Design of Cyber-Physical Security Testbed for Multi-Stage Manufacturing System

Stephen J. Coshatt, Qi Li, Bowen Yang, Shushan Wu
Darpan Shrivastava, Jin Ye, WenZhan Song
Center for Cyber-physical Systems
University of Georgia
Athens, GA 30602
{stephen.coshatt, bowen.yang, qi.li, Shushan.Wu,
darpan.shrivastava, jin.ye, wsong}@uga.edu

Feraidoon Zahiri
402 CMXG/MXDEO
Robins Air Force Base
Warner Robins, GA 31098
feraidoon.zahiri@us.af.mil

***Abstract*—As cyber-physical systems are becoming more wide spread, it is imperative to secure these systems. In the real world these systems produce large amounts of data. However, it is generally impractical to test security techniques on operational cyber-physical systems. Thus, there exists a need to have realistic systems and data for testing security of cyber-physical systems [1]. This is often done in testbeds and cyber ranges. Most cyber ranges and testbeds focus on traditional network systems and few incorporate cyber-physical components. When they do, the cyber-physical components are often simulated. In the systems that incorporate cyber-physical components, generally only the network data is analyzed for attack detection and diagnosis. While there is some study in using physical signals to detect and diagnosis attacks, this data is not incorporated into current testbeds and cyber ranges. This study surveys currents testbeds and cyber ranges and demonstrates a prototype testbed that includes cyber-physical components and sensor data in addition to traditional cyber data monitoring.**

***Index Terms*—Anomaly Detection, Cyber-Physical, Cyber Range, Cybersecurity, Data Generation, Testbed**

I. Introduction

As cyber-physical systems become more widespread, the need to secure these systems becomes more important [1]. Modern cyber-physical systems, often referred to as operational technology (OT) in industry, have external communication capabilities that are increasingly networked to take advantage of advances in Industry 4.0 advanced capabilities including digital engineering, industrial IoT, data analytics, digitization, and integration of the cyber-physical value chain. These communications capabilities increase potential cyber-attack vectors, even in air gapped networks. Industry has growing security concerns that a STUXNET style attack on its cyber-physical systems could degrade or damage their capability to provide services and support. In addition to critical infrastructure, manufacturing systems for the aerospace industry are also an area of concern.

Our research is partially supported by NSF-SaTC-2019311, DOD-FA8571-21-C-0020, DOE-EE0009026 and Georgia Research Alliance.

To our knowledge, limited studies have been done on using side channel information, such as the information embedded in electrical signals for cyber-threat detection in cyber-physical systems. Some cyber-threats including integrity attacks may not be observed in the cyber-space alone and can only be discovered through inter-dependency analysis of multiple cyber and physical signals. Thus, there is a significant opportunity in exploring side channel information from physical signals, together with cyber signals, to advance cyberspace security and trustworthy research and design.

In order to take full advantage of information in cyber networks as well as information in the physical signals from CPS, researchers need access to both types of data when studying attacks and testing defenses. To this end, we propose a cyber-physical testbed that includes data from both sources and demonstrate its effectiveness. The testbed will lay the ground work for building a cyber range for cyber-physcial systems that includes sensor data in addition to cyber data.

II. Background and Related Works

Until recently, attacks on cyber-physical systems were either simulated, along with their physical systems, or physical devices were monitored in isolation. This is due to the nature of cyber-physical systems. Testing attacks and defenses on a live system poses many issues such as the potential to damage systems, harm to people, and loss of access to services they provide [1] [2]. Simulated data and attacks often generated data that are very clean and are not representative of real-world systems. On the other hand, creating realistic faults and attacks in an isolated system were difficult to do. Thus, the Center set out to build a testbed that incorporates physical systems integrated into a network in which a variety of attacks could be created generated, and defenses could be tested. The goal is to have a testbed in which cyber data could be generated and collected along with physical data from sensors. The long-term goal is building a cyber range in which physical CPS components can be added and removed from the range as necessary to test defenses and incorporate realistic attacks.

To accomplish this, a survey of existing testbeds and cyber ranges were performed. The criteria for our testbed are as follows:

- Includes cyber-physical component
- Ability to collect cyber data (network/system data)
- Ability to collect physical data from physical component
- Ability to collect side channel data from physical component(s)
- Ability to implement attacks on cyber-physical component
- Ability to implement defense for the cyber-physical component
- Data Fusion of cyber and physical data in attack defense

Based on our findings a simple prototype testbed was created. During this search, multiple surveys on testbeds and cyber ranges were found and were of great help in conducting this survey [1], [3]–[10]. The information collected from other testbeds aided our team in deciding how to expand upon our prototype. The advantage of the UGA's testbed is that we use side channel information in defense of our cyber-physical systems. While we were able to find cyber-physical testbeds the met a most of our requirements such as [11]–[19], we were unable to locate a testbed or cyber range that used side channel information from sensors in defense of cyber-physical systems. Nor were we able to find a testbed that use data fusion of cyber and physical data in attack defense.

System Designs

The initial phase of the testbed is a basic prototype that contains the proposed components: cyber-physical components, collection of physical sensor data and cyber data in real time, basic attack on the physical components, and a basic detection capability. Unique to our testbed is the incorporation of side channel data for defense of our cyber-physical systems. In the current setup, we use electric-waveform data of our motor's power to detect attacks. See Figure 1 for images of some of the testbed equipment.

Fig. 1. Testbed Equipment.

A. Motors & Sensors

The testbed is constructed to emulate the behaviours of the industrial machines. Such testbed consists of a permanent magnet synchronous machine (PMSM), a three-phase inverter and an ARM-based digital control unit. Table I shows the detail specifications of the testbed motors and Fig. 2 shows the control diagram of the motor drive.

TABLE I
Mini-S&A-Testbed Motor Specifications

Control Unit	NXP S32K144 (Arm Cortex-M4F)
Power Module	SMARTMOS GD3000 3-phase motor driver
Motor	LINIX 45ZWN24-40
Motor Ratings	24V, 40W, 4000rpm, 2.3A, 2 pole pairs
Power Supply	PS-1250APL05/S3: 12 V, 5 A
4*Interfaces	On-board for CAN On-board for LIN On-board OpenSDA debug interface SWD/JTAG debug interface

As shown in Fig. 2, the control unit adopts the field oriented control algorithms to regulate the rotating speed of PMSM according to the requirement from the PC. The PC and the control unit are communicating through the NXP FreeMASTER interface and the Low Power Universal Asynchronous Receiver-Transmitter (LPUART) module. The control algorithms have a two-level feedback control loop: the outer control loop and the inner control loop. The outer loop has a speed regulator associated with the field weakening module to control the motor rotating speed and the air gap flux. The outer loop generates the current references and sends them to the inner loop. The inner loop has two proportional–integral controllers for controlling the d- and q-axis current, respectively. The outputs of the inner control loop are the d- and q-axis voltage commands. Then the Inverse Park Transformation transforms such commands into stationary reference frame, and the PWM modulation module converts these commands into PWM signals, which directly control the six power switches in the inverter. Table II lists the detailed descriptions of all the variables in the control diagram. In addition, National Instrument c-DAQ compact data acquisition board is used to collect and pre-process the physical measurements from the pre-deployed sensors. The sensors implemented for collecting motor line current signals are from Texas Instruments (TMCS1108A4B), which are Hall-effect current sensors with internal reference.

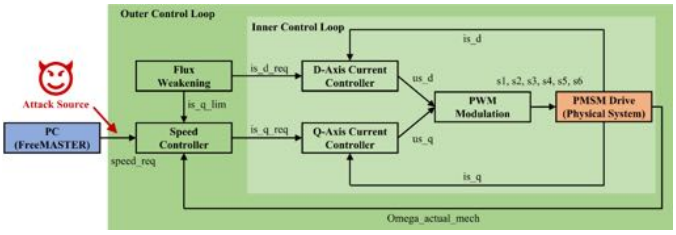

Fig. 2. Control diagram and attack source of the motor testbed.

B. Software

The testbed is a combination of commercial, open source, and in-house software. The software for the motor boards are NXP's FreeMASTER Run Time Debugging Tool, FreeMASTER Lite, and S32 Design Studio IDE. These tools are used for loading software onto the motor board and for setting up communication to the board from a workstation. Additionally,

TABLE II
CONTROL VARIABLE DETAIL DESCRIPTIONS

speed_req	motor rotating speed command
omega_actual_mech	feedback motor rotating speed
is_d_req	d-axis current reference
is_q_req	q-axis current reference
is_q_lim	q-axis current limitation
is_d	feedback d-axis current
is_q	feedback q-axis current
is_a	feedback motor phase-A current
is_b	feedback motor phase-B current
is_c	feedback motor phase-C current
is_alpha	feedback α-axis current
is_beta	feedback β-axis current
us_d	d-axis voltage command
us_q	q-axis voltage command
us_alpha	uncompensated α-axis voltage command
us_beta	uncompensated β-axis voltage command
us_alpha_comp	compensated α-axis voltage command
us_beta_comp	compensated β-axis voltage command
u_dc	feedback DC bus voltage
theta	feedback motor rotor position
theta_enc	rotor position signal from encoder

Eclipse Paho MQTT and custom built Python software are used for setting up motors in their default run-time state and for issuing remote commands to the motors. Motor commands are sent to the local NXP Lite Node Servers via JavaScript Object Notation - Remote Procedure Calls (JSON-RPC). InfluxDB is used for storing all real-time data. Wireshark and Tshark are used for logging network traffic. Hydra is used for password cracking. LabView is used to read data from the sensors. Our LabView program passes data to a Python script that extracts Phasor Measurement Unit (PMU) data from the raw sensor data and sends it to InfluxDB for storage. There is also a Python script that collects information from filtered Tshark packet collection and stores the data in InfluxDB. Lastly, CollectD is used to gather hardware system statistics data on the testbed's Raspberry Pi workstations. A Python script is used to pass the collected data to InfluxDB for storage. There is another Python script which monitors the data for anomaly detection. Grafana is used for visualization of various features in real time as well as for anomaly detection display.

C. Network Description

The testbed is a small network that includes four motors. It is connected to the University of Georgia's PAWS-Secure network. The motors are used to emulate a steel roller in which all four motors should operate at the same speed. Each motor is connected to a Windows 10 workstation that act as the motors controllers. The motors are connect to the workstations by a micro-usb cable. These workstations are connected via Ethernet to a router that is connected to PAWS-Secure. This setup is to loosely represent an Operational Technology (OT) network connected to a traditional Information Technology (IT) network. There are four Raspberry Pi 4 connected to a Wi-Fi router. The Wi-Fi router is connected by Ethernet to the same router as the Windows 10 workstations. The purpose of the Raspberries is to emulate user workstations residing on IT network that can remote into the Windows 10 controllers located on an OT network. Even though our OT workstations are connect to PAWS-Secure, our team needed to avoid attacking any devices in PAWS-Secure, thus we attached a small wireless LAN to represent our IT network. Sensors are attached to each motor in order to read the raw electric wave-form data from each motors power usage. These sensors output data to a Windows 10 workstation via a National Instruments (NI) cDAQ. PMU and Total Harmonic Distortion (THD) data are extracted from the raw waveforms and sent to the InfluxDB database. An Ubuntu workstation hosts our Grafana dashboards that monitor all of our data collection in real-time. The InfluxDB server, Ubuntu workstation, and the LabView workstation are all directly connected to PAWS-Secure. See the Fig. 3 below for details.

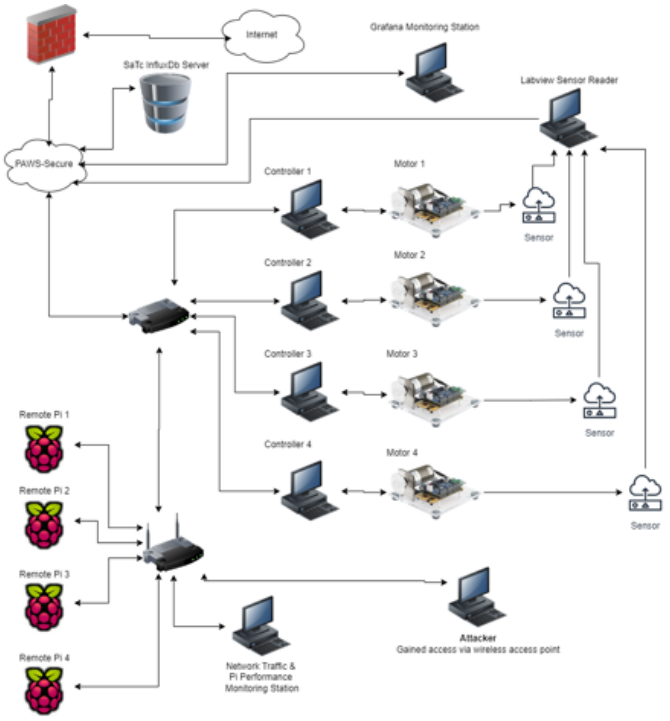

Fig. 3. Schematic of the testbed.

D. Attack Description

For our proposed step-stone attack, we developed a Python script that launches a chain attack in the WLAN based on the input series of IP addresses of the target. The complete workflow is as follows:

1) attacker gained access to the IT network via an unsecured wireless access point shown in Fig. 3
2) attacker launches port scan to locate other workstations on the network
3) brute force password cracking attack based on the target victim IP address, which is arbitrary selected from the scan
4) once the password is cracked, the attacker logs in and loads and runs a script that sends JSON-RPC commands to the NXP Lite server to execute motor commands. This

allows the attacker to speed up, slow down, stop, restart, and disconnect the motor
5) The compromised Pi then launches this attack on another Pi
6) Repeat

III. System Demonstrations and Evaluations

The testbed collects and stores three types of data. Sensor data, network packet data, and workstation system statistics data. The details of each type of data are discussed below.

A. Network Data

Network data statistics play a crucial role in detecting suspicious activities in a network. The testbed uses Wireshark and Tshark to capture and analyze network traffic data. In the testbed's step-stone attack, compromised Raspberry Pis send out ARP request to identify the IP address of the next Raspberry Pi to attack. This happens until the last Raspberry Pi is compromised. To detect the sudden influx of ARP packets, we calculate network statistics, in particular traffic and packet size every 5 seconds. Traffic denotes the number of packets and packet size denotes the size of packet for each identified protocol type. This is done by using a protocol analyzer i.e Tshark, a command line version of the popular tool Wireshark. We then pipe the output of Tshark to a python script for post processing and digest the data into the testbeds InfluxDB server. From InfluxDB, the data is then visualized using Grafana. When the attack is carried out, there is a clear spike in traffic and packet size. This is simple means of identifying the step-stone attack using network traffic data. I more robust means of detecting this anomalous behavior will be added to the test bed in the future. See Figure 5 for an example of Grafana network data dashboard.

Fig. 4. A Grafana panel showing ARP Packet counts. Note the increase in ARP traffic during an attack highlighted in the red box.

B. System Statistics Data

System statistics data represents the status of the workstation, which in our case are of four raspberry Pis. Specifically, it consists of memory usage, CPU core temperature, CPU short/long term load, network traffic (TX and RX), disk usage and disk bandwidth. Collectd is used to collect those data. It is a daemon which collects system and application performance metrics periodically and provides mechanisms to store the values in a variety of ways, such as data pushed to the InfluxDB database. By taking this information into consideration, we could tell the system's current status in real time, which potentially shows if the system is healthy or not. For example, when our designed step-stone attack is happening in a Raspberry Pi, some metrics will show abnormal pattern such as CPU temperature, traffic and CPU short term load. Figure 5 shows an example of Grafana panel. From the picture we could see the status of Raspberry Pi 2 in real time.

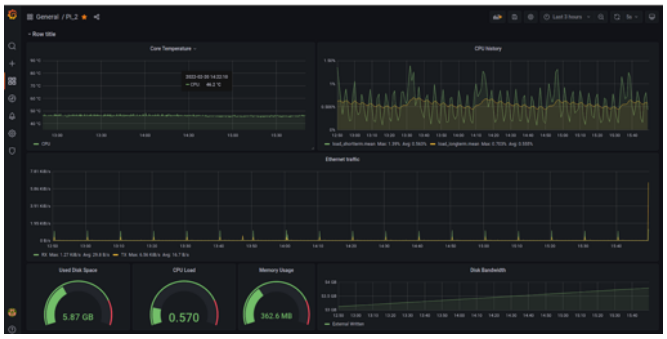

Fig. 5. A Grafana panel showing system statistics data. It is showing the real time status of the Raspberry Pi 2. From the picture, we could monitor the CPU core temperature, Memory usage, CPU short/long term load and disk usage, etc.

Fig. 6. A Grafana panel showing sensor data. It is showing the real time status of a motor. From the picture, you can see when a motor starts up and when its attacked.

C. Sensor Data

Texas Instruments TMCS1108A4B sensors are connected directly to the power cables on the motor. The sensors send current data. For each motor, we currently have two sensors connected to record two of the three phase currents. This information is collected with a National Instrument c-DAQ compact data acquisition board and sent to a workstation with LabView. This workstation displays the raw waveforms and utilizes a python script to extract PMU data (magnitude, frequency, phase angle) and the Total Harmonic Distortion (THD) from the raw waveforms. This extracted data is sent to InfluxDB for storage via the same script. This data is our side channel data that our anomaly detectors utilizes. Not that the sensor data is sent directly to InfluxDB directly from a system on PAWS-Secure and does not traverse the OT network. See Figure 6 for an example of a Grafana dashboard displaying sensor data.

D. Anomaly Detection & Analysis

Singular Spectrum Transformation (SST) [20], [21] is a classical way to decompose the time series and get the change point score (CP score) that could help us find anomalies. It embeds the time subsequences into the subspaces composed by top principle components. Then, we could define a change point score based on an appropriate distance between two subspaces to filter out the noise and feature the signal that drastically change over time.

We first denote the a time sequence by $X_t = (x_{l*t}, ..., x_{l*t+m+w})$, and the next time sequence with lag l is $X_{t+1} = (x_{l*(t+1)}, ..., x_{l*(t+1)+m+w})$ where $t \in \{1, .., n\}$, m is the order of each time window, w is the window length, and l is the time lag between two consecutive time window. For each sequence of time series of length $m+w$, we stack the m subsequences of length w and convert the time sequence X_t, X_{t+1} to matrices H_t, H_{t+1}, respectively. Figure 7 shows how we stack the subsequences for each time window and get the CP score. Each parallelogram represents a stacked time matrix, and the CP score of the present time window is calculated by the distance between the past time window and present time window.

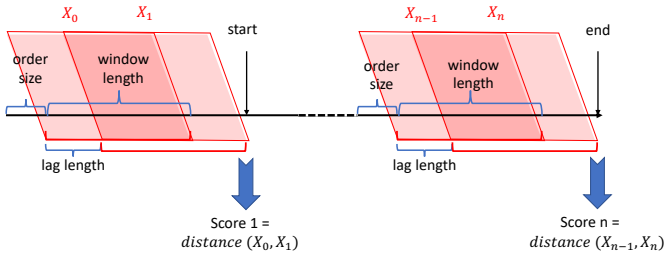

Fig. 7. The procedure of the algorithm processing the sequential data.

Since the signals are often with noise which would contaminate the anomalous information, we would like to use singular value decomposition (SVD) to select the major information. Then the time matrix H_t could embed the time series into subspaces $\mathcal{S}^h = \text{span}\{u^{(1)}, ..., u^{(h)}\}$, where $\{u^{(1)}, ..., u^{(h)}\}$ are top h left singular vectors from applying SVD to matrix H_t. Since two time sequences are now embedded to two subspaces with different dimensions, we could define the distance between two time sequences by distances in the projected spaces.

For example, if the subspace of X_t is $\mathcal{S}^h$ of dimension h, and the subspace of X_{t+1} is $\mathcal{V}^r$ of dimension r, we could apply projection operators $\mathcal{P}_h$ and $\mathcal{P}_r$ to two subspaces $\mathcal{S}^h, \mathcal{V}^r$, respectively. We define the operators as follows:

$$\mathcal{P}_h = \sum_{i=1}^{h} \boldsymbol{u}^{(i)} \boldsymbol{u}^{(i)^\top} \text{ and } \mathcal{P}_r = \sum_{i=1}^{r} \boldsymbol{v}^{(i)} \boldsymbol{v}^{(i)^\top} \quad (1)$$

where $\{v^{(1)}, ..., v^{(r)}\}$ are the top r singular vectors of the time matrix H_{t+1}. We could then define a CP score as the distance between two consecutive subspaces $\mathcal{S}^h, \mathcal{V}^r$.

$$CP_{t+1} = d(\mathcal{S}_h, \mathcal{V}_r)^2 \equiv \min_{\boldsymbol{x} \in \mathcal{S}_h, \|\boldsymbol{x}\|=1} \|(\mathcal{P}_h - \mathcal{P}_r)\boldsymbol{x}\|^2 \quad (2)$$

In this example, we applied the SST algorithm to DC current of the 1^{st} sensor in our testbed. As for the choice of the number of singular vectors of the time matrix, we use the top ones that the corresponding singular values could take up 90% of the sum of all singular values. As for the hyperparameters, we set the window length as 15, lag and order are both 5. Applying the above algorithm, we could get the anomaly score of the DC current for the 2^{nd} sensor. As shown in Figure 8, we could see that after the anomaly happens, the CP score calculated by the anomaly detection algorithm increased drastically with little delay.

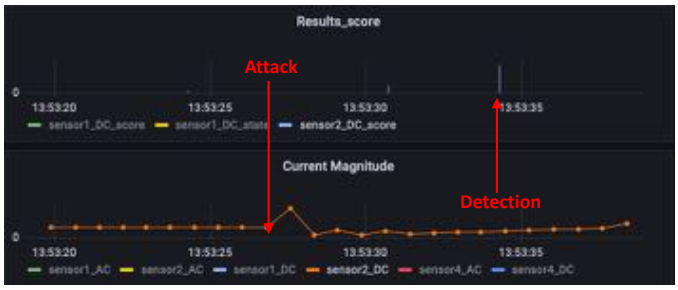

Fig. 8. A Grafana panel showing the anomaly score and the DC current magnitude of sensor 2.

E. Issues & Limitations

During the setup of the testbed, we encountered several problems. The initial sensors that were chosen did not provide good signals when monitoring the motors' electric waveforms which led to us replacing them with the current sensor model. This was partly due to the small power requirements of the motors use.

The motors used in the testbed are small and not built to any industry standards. We performed some simple tests to verify the limits at which the motor could speed up without causing a fault. We determined that approximately 40 rads/sec was the maximum increment we could safely speed up or slow down all of the motors with out causing a fault. Some of the motors could handle larger increments while others could not. When creating our attacks to speed up/slow down the motors, we tried causing the motors to spin faster than their specifications would safely allow. We were able to do so, but repeated attempts eventually damaged the motor such that it could no longer spin properly.

A third issue related to the use of NXP's Node Server. We initially connected all of the motors to on Windows 10 workstation that ran a single instance of Node Server. However, when sending the JSON-RPC commands to multiple motors, only one motor would execute the commands. Thus we moved to multiple workstations with multiple instances of Node Server running. Note that we had to change the service name of each instance in order to have multiple running on the same network.

Currently, our testbed is very limited. While we have approximated an IT/OT network, we do not have true separation using Demilitarized Zones (DMZ) between the IT and OT

networks. Additionally, the network traffic is minimal due to the very small size of our current setup. Thus anomaly activity stands out more in our testbed than would it would in a real world network. Lastly, the system statistics data is monitored on Raspberry Pis that do very little. There is no actual user or many applications running on them. Thus anomalies that stand out in the current set up may not be so obvious on an active real world workstation.

Conclusions and Future Works

The testbed is successful in collecting and storing all of the required data: sensor data, network traffic, and system statistics data. Additionally, we have successfully created a step-stone attack with in the test bed and used anomaly detection to identify the attacks. At this point, we have not yet incorporated data fusion to look at the combined data for attack detection. Our testbed provides an advantage over other testbeds in that we apply sensors to directly monitor the motors electric waveforms for anomalous behaviour. Thus we were able to incorporate the use of side channel information in defense of a cyber-physical systems. Additionally, the collection and analysis of the sensor data is logically separated from the OT network, which provides an added layer of security. Note that the workstations connected directly to the motor-boards could read physical data, such as speed. However, when the workstation is comprised, this data cannot be trusted.

In order to make our testbed a better approximation of a real-world IT/OT network, we need to make several improvements. First, we plan on adding a DMZ between our simulated IT network and our OT network. In both of these, we plan on adding more traffic by using a combination of virtual machines and software the simulates user activity. Second, we plan on adding fault and attack diagnosis in addition to anomaly detection. Third, we plan to use data fusion to use our network, system statistics data, and sensor data for anomaly detection, diagnosis, and localization. Lastly, we intend to implement more sophisticated stealthy attacks to test the efficacy of our defenses.

Acknowledgment

Our research is partially supported by DOE-EE0009026, NSF-SaTC-2019311, DOD-FA8571-21-C-0020 and Aging Aircraft Solutions.

References

[1] G. Kavallieratos, S. K. Katsikas, and V. Gkioulos, "Towards a cyber-physical range," in *Proceedings of the 5th on Cyber-Physical System Security Workshop*, 2019, pp. 25–34.

[2] T. Becejac, C. Eppinger, A. Ashok, U. Agrawal, and J. O'Brien, "Prime: a real-time cyber-physical systems testbed: from wide-area monitoring, protection, and control prototyping to operator training and beyond," *IET Cyber-Physical Systems: Theory & Applications*, vol. 5, no. 2, pp. 186–195, 2020.

[3] R. V. Yohanandhan, R. M. Elavarasan, R. Pugazhendhi, M. Premkumar, L. Mihet-Popa, and V. Terzija, "A holistic review on cyber-physical power system (cpps) testbeds for secure and sustainable electric power grid–part–i: Background on cpps and necessity of cpps testbeds," *International Journal of Electrical Power & Energy Systems*, vol. 136, p. 107718, 2022.

[4] E. Ukwandu, M. A. B. Farah, H. Hindy, D. Brosset, D. Kavallieros, R. Atkinson, C. Tachtatzis, M. Bures, I. Andonovic, and X. Bellekens, "A review of cyber-ranges and test-beds: Current and future trends," *Sensors*, vol. 20, no. 24, p. 7148, 2020.

[5] M. M. Yamin, B. Katt, and V. Gkioulos, "Cyber ranges and security testbeds: Scenarios, functions, tools and architecture," *Computers & Security*, vol. 88, p. 101636, 2020.

[6] Y. Geng, Y. Wang, W. Liu, Q. Wei, K. Liu, and H. Wu, "A survey of industrial control system testbeds," in *IOP Conference Series: Materials Science and Engineering*, vol. 569, no. 4. IOP Publishing, 2019, p. 042030.

[7] Q. Qassim, N. Jamil, I. Z. Abidin, M. E. Rusli, S. Yussof, R. Ismail, F. Abdullah, N. Ja'afar, H. C. Hasan, and M. Daud, "A survey of scada testbed implementation approaches," *Indian Journal of Science and Technology*, vol. 10, no. 26, pp. 1–8, 2017.

[8] M. H. Cintuglu, O. A. Mohammed, K. Akkaya, and A. S. Uluagac, "A survey on smart grid cyber-physical system testbeds," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 446–464, 2016.

[9] H. Holm, M. Karresand, A. Vidström, and E. Westring, "A survey of industrial control system testbeds," in *Nordic Conference on Secure IT Systems*. Springer, 2015, pp. 11–26.

[10] J. Davis and S. Magrath, "A survey of cyber ranges and testbeds," 2013.

[11] C. M. Ahmed, V. R. Palleti, and A. P. Mathur, "Wadi: a water distribution testbed for research in the design of secure cyber physical systems," in *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks*, 2017, pp. 25–28.

[12] I. Ahmed, V. Roussev, W. Johnson, S. Senthivel, and S. Sudhakaran, "A scada system testbed for cybersecurity and forensic research and pedagogy," in *Proceedings of the 2nd Annual Industrial Control System Security Workshop*, 2016, pp. 1–9.

[13] B. Green, A. Lee, R. Antrobus, U. Roedig, D. Hutchison, and A. Rashid, "Pains, gains and {PLCs}: Ten lessons from building an industrial control systems testbed for security research," in *10th USENIX workshop on cyber security experimentation and test (CSET 17)*, 2017.

[14] A. Siddiqi, N. O. Tippenhauer, D. Mashima, and B. Chen, "On practical threat scenario testing in an electric power ics testbed," in *Proceedings of the 4th ACM Workshop on Cyber-Physical System Security*, 2018, pp. 15–21.

[15] S. Poudel, Z. Ni, and N. Malla, "Real-time cyber physical system testbed for power system security and control," *International Journal of Electrical Power & Energy Systems*, vol. 90, pp. 124–133, 2017.

[16] A. P. Mathur and N. O. Tippenhauer, "Swat: A water treatment testbed for research and training on ics security," in *2016 international workshop on cyber-physical systems for smart water networks (CySWater)*. IEEE, 2016, pp. 31–36.

[17] E. Xypolytou, J. Fabini, W. Gawlik, and T. Zseby, "The fuse testbed: establishing a microgrid for smart grid security experiments," *e & i Elektrotechnik und Informationstechnik*, vol. 134, no. 1, pp. 30–35, 2017.

[18] S. Tan, W.-Z. Song, S. Yothment, J. Yang, and L. Tong, "ScorePlus: A Software-Hardware Hybrid and Federated Experiment Environment for Smart Grid," *ACM transaction on Embedded Computing Systems, Special issue on Emerging Embedded Software and Systems (ESS)*, vol. 16, no. 1, p. 19, 2016. [Online]. Available: http://dx.doi.org/10.1145/2964200

[19] G. Lu, D. De, and W. Song, "SmartGridLab: A Laboratory-Based Smart Grid Testbed," in *The 1st IEEE International Conference on Smart Grid Communications (IEEE SmartGridComm)*. IEEE, 2010, pp. 143–148.

[20] V. Moskvina and A. Zhigljavsky, "An algorithm based on singular spectrum analysis for change-point detection," *Communications in Statistics-Simulation and Computation*, vol. 32, no. 2, pp. 319–352, 2003.

[21] T. Idé and K. Inoue, "Knowledge discovery from heterogeneous dynamic systems using change-point correlations," in *Proceedings of the 2005 SIAM International Conference on Data Mining*. SIAM, 2005, pp. 571–575.