Learning to Attack Nonlinear Networked Cyber-Physical Systems

Yu Zheng[1], Satish Vedula[1], Olugbenga Moses Anubi[1]

***Abstract*— Nonlinear networked cyber-physical systems (NCPS) are highly susceptible to malicious attacks due to their complex dynamics and interconnectivity. This paper presents a data-driven generative system for evaluating the vulnerability of nonlinear networked cyber-physical systems (CPS). The proposed framework formulates the vulnerability analysis problem as determining the feasibility of a specific attack set based on two boundary functions that represent the effectiveness and stealthiness of attack signals. The framework incorporates two discriminative models that learn these boundary functions, along with a generative model that generates feasible attack signals. A new loss function is introduced to ensure a high probability of successful attack generation. The approach is evaluated on an IEEE 14-bus NCPS and a gas pipeline NCPS, demonstrating the effectiveness of the proposed attack generation scheme in assessing the vulnerability of nonlinear NCPS. The results show that the proposed approach can be used to learn how to attack nonlinear CPS and identify potential vulnerabilities.**

I. INTRODUCTION

Modern cyber-physical infrastructure systems (CPIS) are built based on the seamless integration of physical devices and computer systems to enable the exchange of data and control of physical processes [1]. As the complexity and scale of CPIS increase, network structures are employed to pool the resources and capabilities of subsystems to provide better functionality and performance [2]. In such networked cyber-physical systems (NCPS), physical agents, sensors, and actuators are interconnected physically or coupled through communication networks to gather and analyze data and perform physical actions in response. NCPSs are used in various applications, such as smart power grids [3], industrial pipeline systems [4], and transportation systems [5]. The networked and distributed integration of computer systems and physical devices provides numerous benefits, including improved efficiency, reduced costs, and increased safety [6]. However, networked information transmission presents significant security and privacy challenges as they are vulnerable to cyber-attacks and data breaches [7], [8]. Therefore, the development of secure and reliable networked CPSs is essential to ensure their safe and effective operation.

Assessing the vulnerability of networked cyber-physical systems (NCPSs) is a challenging task that requires a deep understanding of the complex interplay between the physical and cyber components of the system and the potential attack scenarios. In the literature, researchers have proposed various techniques to address the vulnerability analysis problem in cyber-physical systems (CPS), ranging from mathematical and analytical methods to data-driven and simulation-based approaches. Early researchers incorporated the full system model into the maximization program to generate a feasible attack [9], [10], [11]. The authors in [9] studied false data injection attack (FDIA) against the least-square estimator with a residual-based BDD. The feasibility of FDIA against the Kalman filter with χ^2 detector was studied in [10]. One study defined vulnerability under sensor attacks as the boundedness of the estimation error, and derived sufficient and necessary conditions through analysis of the system's reachable unstable zero dynamics [12]. A sufficient and necessary condition for insecure estimation under FDIA was also derived for the networked control system in [11]. To develop more pragmatic attack generation strategies, several constraints are incorporated to capture the attacker limitations such as limited access to sensors [9], incomplete knowledge of system dynamics [13], and incomplete knowledge of implemented state estimators [14]. However, all these approaches target the linear CPS.

The nonlinear nature of many NCPS and the request for high-fidelity models make the attack generation problem challenging against NCPS [15]. An engineering type of approach is to analyze the vulnerability of the system based on domain knowledge and through tons of simulations. The authors in [16] combined a stochastic adversarial model and a simulation model of interdependent gas-electricity delivery infrastructures to explore possible operational disruptions caused by cyber-attacks. The authors in [17] combined hierarchical Limited stochastic Petri nets and power system network topology to simulate the intrusion attack scenarios. A dynamic network security vulnerability assessment method for the SCADA system was also studied by considering software vulnerabilities and deployed network security defense measures [18].

However, the question is if there exists an unified and analytical approach to analyzing the vulnerability of any NCPS. The authors in [19] characterize attacks using an asymptotic detection performance, defined as the rate of decrease in the worst-case probability of error. Then, an attack-defense game was utilized to search for the boundary of vulnerability. Similarly, the authors in [8] built the connections among robustness, security, and resilience from the perspective of game theory. On the other hand, some researchers view the vulnerability analysis problem as a feasible attack generation problem characterized by effectiveness and stealthiness [12]. Some works have trained generative adversarial networks to learn from existing feasible attack datasets [20], but this relies on the existence of non-generative attack datasets and

[1]Yu Zheng, Olugbenga Moses Anubi and Satish Vedula are with the Department of Electrical and Computer Engineering, Florida State University, FL, USA. yzheng6@fsu.edu, oanubi@fsu.edu, svedula@fsu.edu

good representative quality of the training data. The authors in [21] used data-driven models to approximate the system model, which reduces the complexity of the attack generation problem.

In this paper, we give a new feasible attack generation problem formulation. Based on this problem formulation, we observe that a data-driven generative model could be utilized to solve this problem with a customized loss function. In order to remove the difficulty of the nonlinear model knowledge of NCPS, two data-driven discriminators are utilized to learn the characterization of feasible attacks based only on system runtime data. Consequently, a pure data-driven attack generation framework, based on interactive training among three data-driven models, is proposed with a quantitative performance guarantee. Furthermore, The proposed attack generation framework is also validated on an IEEE 14-bus system and a gas pipeline system with different topologies.

The remainder of the paper is organized as follows. In Section II, the notations employed are summarized. In Section III, a standard setup of nonlinear NCPS systems is introduced, and the attack generation problem is formulated. In Section IV, the proposed data-driven attack generation framework is discussed. In Section V, two case studies are presented, one is on IEEE 14-bus system and the other is on several gas pipeline systems. Conclusions follow in Section VI.

II. Notations

We use $\mathbb{R}^n, \mathbb{R}_+$ to denote the space of real column vectors of length n and positive real numbers respectively. Normal-face lower-case letters $(e.g.\ x \in \mathbb{R})$ are used to represent real scalars, bold-face lower-case letters $(e.g.\ \mathbf{x} \in \mathbb{R}^n)$ represent vectors, while normal-face upper-case letters $(e.g.\ X \in \mathbb{R}^{m\times n})$ represent matrices. We use $\mathbf{x}_i, \mathbf{x}(i)$ to denote the ith element of the vector $\mathbf{x} \in \mathbb{R}^n$ and the vector $\mathbf{x} \in \mathbb{R}^n$ at time i, repectively.

The probability triple is denoted by $(\Omega, \mathcal{F}, \mathbf{P}_\mathbf{z})$, where Ω is a sample space containing the set of all possible outcomes, $\mathcal{F}$ is an event space, and $\mathbf{P}_z$ is the associated probability functions of the events in $\mathcal{F}$. We use the symbol $\mathbb{E}$ to denote the expected value operator. The ReLU function is given by

$$\mathsf{ReLU}(x) = \max(0, x),$$

and the $\mathsf{LeakyReLU}_s$ function is given by

$$\mathsf{LeakyReLU}_s(x) = \max(0, x) + s\min(0, x),$$

where $s < 0$.

An undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ contains vertices, denoted by $\mathcal{E} = \{v_1, v_2, \ldots, v_n\}$, and edges, denoted by $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$. (v_1, v_2) and $(v_1, v_2) \in \mathcal{E}$ represents an edge in $\mathcal{G}$. Consequently, the adjacency matrix, denoted by $A(\mathcal{G})$, is a square matrix of size $|\mathcal{V}|$, defined as [22]

$$A_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in \mathcal{E}, \\ 0 & \text{otherwise } . \end{cases}$$

The support of a vector $\mathbf{x} \in \mathbb{R}^n$ is defined as $\mathsf{supp}(\mathbf{x}) \triangleq \{i \subseteq \{1, \ldots, n\} | \mathbf{x}_i \neq 0\}$. $\Sigma_k \triangleq \{\mathbf{x} \in \mathbb{R}^n | |\mathsf{supp}(\mathbf{x})| \leq k\}$ denotes the set of k-sparse vectors. We use $\otimes$ to denote Kronecker product, i.e. for $B \in \mathbb{R}^{m\times n}$,

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \otimes B = \begin{bmatrix} a_{11}B & a_{12}B \\ a_{21}B & a_{22}B \end{bmatrix} \in \mathbb{R}^{2m\times 2n}.$$

We use $\odot$ to denote element-wise multiplication, i.e.

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \odot \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} \\ a_{21}b_{21} & a_{22}b_{22} \end{bmatrix},$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \odot \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} a_{11}b_1 & a_{12}b_1 \\ a_{21}b_2 & a_{22}b_2 \end{bmatrix}.$$

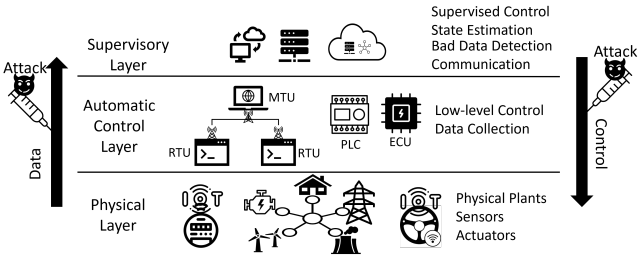

Fig. 1. A typical example of a control system of NCPS under attacks (MTU: master terminal unit, RTU: remote terminal unit, ECU: electrical control unit, PLC: programming logic controller)

III. Problem Formulation

This paper considers a networked CPS under attack. The networked CPS contains n nodes/agents with communication and computation capability. A typical control structure of NCPS is depicted in Fig. 1. The networked physical plants are equipped with either/both IoT sensors or/and IoT actuators which are controlled by the automatic control units. The sensing data are shared through the communication network, and a supervisory control center performs state estimation, bad data detection, and generates the targets for system plants. The target commands are transferred to networked plants through the communication network again. However, networked data transmission is vulnerable to malicious hackers. The toxic data injection would cause catastrophic consequences in the physical world.

Therefore, in this section, we properly build this networked cyber-physical control system and formulate the vulnerability analysis problem as a feasible attack generation problem with a quantitative requirement.

A. System Dynamical Model

The physical process is modeled by the network dynamics given by

$$\dot{\mathbf{x}}_i = \sum_{j\in\mathcal{N}_i} \psi(\mathbf{x}_i, \mathbf{x}_j) + B_i\mathbf{z}_i \tag{1}$$

for each node $i \in \{1, 2, \cdots, n\}$, where $\mathbf{x}_i \in \mathbb{R}^{m_1}$ is i-th node's state vector and $\mathcal{N}_i$ denotes the neighborhood of the i-th node/agent, which is the set of nodes physically linked with the i-th node. The physical layer dynamics in (4) and the interaction function $\Psi : \mathbb{R}^{m_1} \to \mathbb{R}^{m_1}$ is assumed to satisfy the following properties [23]:

1) $\psi(\mathbf{y}, \mathbf{y}) = 0$,
2) $\psi(\mathbf{y}, \mathbf{z}) = -\psi(\mathbf{z}, \mathbf{y})$,

3) The underlying graph is undirected.

Let $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1^\top & \mathbf{x}_2^\top & \cdots & \mathbf{x}_n^\top\end{bmatrix}^\top \in \mathbb{R}^{m_1 n}$ be an augmented network state vector composing of all nodal state vectors, we define the nodal interaction matrix

$$\Psi(\mathbf{x}) \triangleq \begin{bmatrix} 0 & \psi(\mathbf{x}_1,\mathbf{x}_2) & \dots & \psi(\mathbf{x}_1,\mathbf{x}_n) \\ \psi(\mathbf{x}_2,\mathbf{x}_1) & 0 & \dots & \psi(\mathbf{x}_2,\mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ \psi(\mathbf{x}_n,\mathbf{x}_1) & \psi(\mathbf{x}_n,\mathbf{x}_2) & \dots & 0 \end{bmatrix}. \quad (2)$$

Since the underlying graph is undirected, it is straight forward to show that the nodal interaction matrix Ψ satisfies the skew-symmetric properties; $\Psi(\mathbf{x})^\top = -\Psi(\mathbf{x})$ and $\eta^\top \Psi(\mathbf{x})\eta = 0$ for all $\mathbf{x} \in \mathbf{x} \in \mathbb{R}^{m_1 n}$ and $\eta \in \mathbb{R}^n$.

Consequently, the entire network dynamics is given as

$$\dot{\mathbf{x}} = (A \odot \Psi(\mathbf{x}))\,\mathbf{1} + B\mathbf{z}, \quad (3)$$

where $A \in \mathbb{R}^{n\times n}$ is the adjacency matrix of the underlying undirected graph, $B = \mathsf{blkdiag}([B_1, B_2, \cdots, B_n])$ is the block diagonal matrix of all B_i.

Let $\mathcal{S}_a$ denote the index set of actuation nodes, then the actuation state vector $\mathbf{z}_i = \mathbf{0}$ if $i \notin \mathcal{S}_a$, otherwise it follows an actuation dynamics described by a control-affine equation subject to actuation attacks:

$$\dot{\mathbf{z}}_i = f_i(\mathbf{z}_i) + g_i(\mathbf{z}_i)(\mathbf{u}_i + \mathbf{e}_i^u), \quad (4)$$

where $\mathbf{z}_i \in \mathbb{R}^{m_2}$ is i-th node's actuation state, $\mathbf{u}_i, \mathbf{e}_i^u \in \mathbb{R}^p$ are the i-th node's controlled actuation signal and injected attacks on actuators respectively. f, g are Lipchitz continues on compact sets.

Next, a measurement model of equipped IoT sensors under attack is given by

$$\mathbf{y} = h(\mathbf{x}) + \mathbf{e}^y + \mathbf{v}, \quad (5)$$

where $\mathbf{y}, \mathbf{e}^y, \mathbf{v} \in \mathbb{R}^m$ are the measurement, sensor attacks and sensor noise.

Let $\mathcal{T}_u = \mathsf{supp}(\mathbf{e}^u) \subset \{1, 2, \cdots, pn\}$, $\mathcal{T}_y = \mathsf{supp}(\mathbf{e}^y) \subset \{1, 2, \cdots, m\}$ denote the attack supports at actuation and sensing processes respectively. They are usually assumed as sparse in literature [24] such that $\mathbf{e}^u \in \Sigma_{k_u}$ and $\mathbf{e}^y \in \Sigma_{k_y}$, where $k_u < pn, k_y < m$.

B. Nominal Control Design

In this subsection, we consider the control design for the nominal system without attacks (i.e. $\mathbf{e}^u = \mathbf{0}$, $\mathbf{e}^y = \mathbf{0}$). The control design comprises 2 layers; (1) the low-level actuator control (Level 1), and (2) the supervisory target generator (Level 2).

The local controllers are closed-loop schemes $\mathbf{u}_i : \mathbb{R}^{m_2} \to \mathbb{R}^p$ designed to ensure that the actuator state $\mathbf{z}_i$ tracks a given reference $\mathbf{z}_i^r \in \mathbb{R}^p$ such that

$$\|\tilde{\mathbf{z}}(t)\| \le \|\tilde{\mathbf{z}}(0)\|\, e^{-\lambda t}. \quad (6)$$

where $\tilde{\mathbf{z}}_i = \mathbf{z}_i - \mathbf{z}_i^r$ is the associated tracking error. It is assumed that there exists a continuous differentiable, positive definite function $V : \mathbb{R}^{m_2} \times \mathbb{R}^p \to \mathbb{R}$ satisfying [25]

$$\underline{\alpha}(\|\tilde{\mathbf{z}}\|) \le V \le \bar{\alpha}(\|\tilde{\mathbf{z}}\|), \;\; \frac{\partial V}{\partial \tilde{\mathbf{z}}}\dot{\tilde{\mathbf{z}}} \le -\alpha(\|\tilde{\mathbf{z}}\|),$$

Thus, there exists a stabilizing controller of the form $\mathbf{u} = K(V(\tilde{\mathbf{z}}))$ - for example using the Sontag formular [26]

$$K(V(\tilde{\mathbf{z}})) = \begin{cases} -\frac{L_f V(\tilde{\mathbf{z}}) + \sqrt{\left[L_f V(\tilde{\mathbf{z}})\right]^2 + \left[L_g V(\tilde{\mathbf{z}})\right]^4}}{L_g V(\tilde{\mathbf{z}})} & \text{if } L_g V(\tilde{\mathbf{z}}) \neq 0 \\ 0 & \text{if } L_g V(\tilde{\mathbf{z}}) = 0 \end{cases} \quad (7)$$

where $L_f V(\tilde{\mathbf{z}}) \triangleq \left\langle \frac{\partial V}{\partial \tilde{\mathbf{z}}}, f(\tilde{\mathbf{z}}) \right\rangle$, $L_g V(\tilde{\mathbf{z}}) \triangleq \left\langle \frac{\partial V}{\partial \tilde{\mathbf{z}}}, g(\tilde{\mathbf{z}}) \right\rangle$ are the Lie derivatives of V along f and g respectively.

The supervisory target generator generates the state reference $\mathbf{z}^r$ for the local controllers in order to regulate the network states $\mathbf{x}$ at the equilibrium point. We assume the local control speed is much faster than the supervisory generator.

Let $\mathbf{x}_{eq}$ be equilibrium points of network states corresponding to the equilibrium points of inputs $\mathbf{u}_{eq}$. If $\mathbf{u}_{eq} = \mathbf{0}$, then at the equilibrium point, $\mathbf{x}_{eqi} = \mathbf{x}_{eqj}$. Next, a linear model is used to approximate the network dynamical model in (3) around $\mathbf{x}_{eq}$:

$$\dot{\tilde{\mathbf{x}}} = -L(\mathbf{x}_{eq})\tilde{\mathbf{x}} + B\mathbf{z} \quad (8)$$

where $\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{x}_{eq}$ is the regulation error and $L(\mathbf{x})$ is a state-dependent Laplacian matrix defined as

$$L_{ij}(\mathbf{x}) = \begin{cases} -\sum\limits_{j\in\mathcal{N}_i} \frac{\partial \Psi(\mathbf{x_i},\mathbf{x}_j)}{\partial \mathbf{x}_i} & \text{if } i = j \\ -\frac{\partial \Psi(\mathbf{x_i},\mathbf{x}_j)}{\partial \mathbf{x}_j} & \text{if } i \neq j. \end{cases} \quad (9)$$

If the network state transition model is linear $\psi(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x_i} - \mathbf{x}_j$, then the state-dependent Laplacian matrix becomes the normal Laplacian matrix $L = D - A$, where A is the adjacent matrix and D is a diagonal matrix containing all in-degrees such that $D_{ii} = \sum_j A_{ij}$.

Next, given a sample time step T_s, a discrete form of the linear model in (8) is given as

$$\tilde{\mathbf{x}}(k+1) = \underbrace{(-L(\mathbf{x}_{eq})T_s + I)}_{A_d}\tilde{\mathbf{x}}(k) + \underbrace{BT_s}_{B_d}\mathbf{z}(k) \quad (10)$$

Consequently, a model predictive control (MPC) scheme is utilized to regulate the network states at the equilibrium point $\mathbf{x}_{eq}$:

$$\begin{aligned} \underset{\mathbf{z}^r_{[t,t+h]}, \tilde{\mathbf{x}}_{[t,t+h]}}{\mathsf{Minimize}} :\;& \sum_{k=0}^{h} \frac{1}{2}\tilde{\mathbf{x}}(k)^\top M_1 \tilde{\mathbf{x}}(k) + \sum_{k=0}^{h-1} \frac{1}{2}\mathbf{z}^r(k)^\top M_2 \mathbf{z}^r(k) \\ \mathsf{Subject\ to} :\;& \tilde{\mathbf{x}}(k+1) = A_d\tilde{\mathbf{x}}(k) + B_d\mathbf{z}^r(k),\; k = 0, \cdots, h-1 \\ & \tilde{\mathbf{x}}(0) = \tilde{\mathbf{x}}(k) \\ & \tilde{\mathbf{x}}(k) = 0. \end{aligned} \quad (11)$$

This MPC program is to minimize the error from the equilibrium point and the control energy with weight matrices $M1, M2$ respectively. The discrete error dynamics is served as an equality constraint. The initial constraint $\tilde{\mathbf{x}}(0) = \tilde{\mathbf{x}}(k)$ and terminal constraint $\tilde{\mathbf{x}}(k) = 0$ are served as another equality constraints. Since the network state $\mathbf{x}$ is not measurable, the regulation error is given by $\tilde{\mathbf{x}}_t = \hat{\mathbf{x}}_t - \mathbf{x}_{eq}$, where the estimate of the network states $\hat{\mathbf{x}}_t$ is outputted by the following state estimator given sensor measurements $\mathbf{y}$.

The state estimator is an optimizer to minimize the difference between model measurements and real sensing readings $\mathbf{y}$:

$$\begin{aligned} &\underset{\hat{\mathbf{x}}}{\text{Minimize:}} \ \ \|\mathbf{y}-h(\hat{\mathbf{x}})\|_2 \\ &\text{Subject to:} \ \ \dot{\hat{\mathbf{x}}}=(A\odot\Psi(\hat{\mathbf{x}}))\,\mathbf{1}+B\mathbf{z}, \end{aligned} \tag{12}$$

As the estimator estimates the network states from the measurements, a BDD monitors the state estimation process to detect any false inputs. It is defined as a function $D:\mathbb{R}^{m_1 n}\times\mathbb{R}^m\to\mathbb{R}_+$ mapping from the state estimates to a detection residual (i.e. 1-norm or 2-norm residual-based detectors [9], [27]) or detection likelihood (i.e. χ^2 detector [10], [21]).

$$\mathbf{r}_t = D(\hat{\mathbf{x}}_t, \mathbf{y}_t, \mathbf{u}_t). \tag{13}$$

Thus far, the models of the networked system are fully developed. Next, we introduce a problem formulation of the attack generation problem against the networked control system.

C. Attack Generation Problem

Effectiveness and stealthiness are two common criteria characterizing the performance of attacks with respect to the system defined above [7], [21], [19], [27], [10], [14]. We use $l_1:\mathbb{R}^m\times\mathbb{R}^p\to\mathbf{R}, l_2:\mathbb{R}^m\times\mathbb{R}^p\to\mathbf{R}$ to denote functions evaluating the effectiveness and stealthiness of the attacks respectively. Then a feasible set of attacks $\mathcal{S}$ is defined with given thresholds of effectiveness and stealthiness τ_E, τ_S:

$$\mathcal{S}=\left\{\mathbf{e}^u\in\Sigma_{k_u}, \mathbf{e}^y\in\Sigma_{k_y}\middle| l_1(\mathbf{e}^u,\mathbf{e}^y)\geq\tau_E, l_2(\mathbf{e}^u,\mathbf{e}^y)\leq\tau_S\right\} \tag{14}$$

Some research uses the estimation error to represent the effectiveness of sensor attacks such that $l_1=\|\hat{\mathbf{x}}-\mathbf{x}\|$ [9], [21], [28]. The stealthiness function often employs the BDD function $l_2=D$, where D is defined in (13).

Consequently, the attack generation problem is finding a generative model of the form

$$G(\mathbf{z},\theta):(\Omega,\mathcal{F},P_\mathbf{z})\times\mathbb{R}^{n_g}\to\mathbb{R}^n \tag{15}$$

with

1) a constant tunable parameter vector $\theta\in\mathbb{R}^{n_g}$,
2) a prior probability sample space $(\Omega,\mathcal{F},P_\mathbf{z})$ with random variables $\mathbf{z}\sim P_\mathbf{z}$,

such that

$$Pr\{G(\mathbf{z},\theta)\in\mathcal{S}\}\geq\eta \tag{16}$$

for some $\eta\in(0,1)$. Essentially, $G(\mathbf{z},\theta)$ is a generative model for the set $\mathcal{S}$ if $G(\mathbf{z},\theta)\in\mathcal{S}$ with a high probability given by the lower bound η. In addition, the l_1 and l_2 are closed-loop compositions of the models from (3) to (12). Firstly, the high-fidelity dynamical models (3) and (4) are usually unknown. Also, the estimation and control models in (12) and (11) are possibly iterative procedures. Thus, the search for feasible attacks in $\mathcal{S}$ is a highly nonlinear and nonconvex problem.

IV. MAIN RESULTS

In this section, we present a solution to search for feasible attacks in the set $\mathcal{S}$ defined in (14) using only the runtime data of a CPS. Firstly, we present a generator training algorithm with a custom loss function, assuming the knowledge of l_1 and l_2 functions. Secondly, we explore the approximation of l_1 and l_2 functions through data-driven models. Lastly, we describe an interactive training algorithm among multiple data-driven models, addressing various training challenges.

A. Generator

In this subsection, we study the generative problem described in (15) and (16). We assume l_1, l_2 are known functions. The next result gives a condition for a successful generative model.

Theorem 1: Given a set $\mathcal{S}$ whose boundary is of measure zero. If

$$\mathbb{E}\left[D\left(G(\mathbf{z},\theta)\right)\right]\leq 1-\eta, \tag{17}$$

where $D:\mathbb{R}^n\to\mathbb{R}$ is a function given by

$$D(\mathbf{x})\begin{cases}\in[0,1] & \text{if } \mathbf{x}\in\mathcal{S},\\ >1 & \text{otherwise,}\end{cases} \tag{18}$$

then

$$Pr\{G(\mathbf{z},\theta)\in\mathcal{S}\}\geq\eta. \tag{19}$$

Proof: By definition of the expectation operator, for any random variable $\mathbf{x}$,

$$\begin{aligned}\mathbb{E}\left[D(\mathbf{x})\right] &= \sum_{X\in\mathbb{R}^n} D(X)\mathsf{Pr}\{\mathbf{x}=X\}\\ &= \sum_{X\in\mathcal{S}} D(X)\mathsf{Pr}\{\mathbf{x}=X\}+\sum_{X\notin\mathcal{S}} D(X)\mathsf{Pr}\{\mathbf{x}=X\}.\end{aligned}$$

Since $D(x)\geq 0$, it follows that

$$\mathbb{E}[D(\mathbf{x})]\geq\sum_{X\notin\mathcal{S}} D(X)\mathsf{Pr}\{\mathbf{x}=X\}.$$

In addition, since $D(\mathbf{x})>1$ for all $x\notin\mathcal{S}$, $\mathbb{E}[D(\mathbf{x})]$ can be further lower bounded as

$$\mathbb{E}[D(\mathbf{x})]\geq\sum_{X\notin\mathcal{S}}\mathsf{Pr}\{\mathbf{x}=X\}=\mathsf{Pr}\{\mathbf{x}\notin\mathcal{S}\}.$$

Replacing $\mathbf{x}$ with $G(\mathbf{z},\theta)$ yields

$$\mathbb{E}[D(G(\mathbf{z},\theta))]\geq\mathsf{Pr}\{G(\mathbf{z},\theta)\notin\mathcal{S}\}.$$

Using the inequality in (17), it follows that

$$\begin{aligned}\mathsf{Pr}\{G(\mathbf{z},\theta)\in\mathcal{S}\} &= 1-\mathsf{Pr}\{G(\mathbf{z},\theta)\notin\mathcal{S}\}\\ &\geq 1-\mathbb{E}[D(G(\mathbf{z},\theta))]\\ &\geq 1-(1-\eta)\\ &\geq\eta.\end{aligned}$$

■

It is straightforward that the condition in (17) can be achieved to train the generator $G(\mathbf{z},\theta)$ with the loss function

$$L(G(\mathbf{z};\theta))=ReLU\left(\mathop{\mathbb{E}}_{\mathbf{z}\sim\mathbf{P}_z}\left[D\left(G(\mathbf{z};\theta)\right)\right]-(1-\eta)\right). \tag{20}$$

by gradient descent. And this training process is unsupervised. The D function satisfying (18) can be chosen as

$$D(\mathbf{e}) = exp[LeakyReLU_s(\tau_E - l_1(\mathbf{e})) + LeakyReLU_s(l_2(\mathbf{e}) - \tau_S)], \tag{21}$$

which satisfies (18), and $\mathbf{e} = \begin{bmatrix}\mathbf{e}^{u\top} & \mathbf{e}^{y\top}\end{bmatrix}^\top$.

Corollary 1: Let $G(\mathbf{z};\theta) : (\Omega, \mathcal{F}, P_\mathbf{z}) \times \mathbb{R}^{n_g} \to \mathbb{R}^n$ be a generative model with random inputs $\mathbf{z} \sim \mathbf{P}_z$ drawn from a prior distribution and a constant tunable parameters parameter $\theta \in \mathbb{R}^{n_g}$. If $G(\mathbf{z};\theta)$ is continuous in θ and there exists $\theta_0 \in \mathbb{R}^{n_g}$ and $\theta_1 \in \mathbb{R}^{n_g}$ such that $\mathbb{E}[D(G(\mathbf{z};\theta_0))] \leq 1-\eta$ and $\mathbb{E}[D(G(\mathbf{z};\theta_1))] \geq 1$, then

$$\mathbb{E}[D(G(\mathbf{z};\theta^\star))] \leq 1-\eta,$$

where $\eta \in (0,1)$, and $\theta^\star$ is given by

$$\theta^\star = \underset{\theta}{\text{argmin}} \underset{\mathbf{z}\sim\mathbf{P}_z}{\mathbb{E}} L(G(\mathbf{z};\theta)). \tag{22}$$

Proof: Using the Intermediate Value Theorem for $\mathbb{E}[D(G(\mathbf{z};\theta))]$, for any real number $\delta \in [1-\eta, 1]$, there exists $\theta_g \in \mathbb{R}^{n_g}$ such that $\mathbb{E}[D(G(\mathbf{z},\theta_g))] = \delta$. Thus, it follows from the optimality of $\theta^\star$ and definition of ReLU that $L(G(\mathbf{z};\theta^\star)) = 0$, which implies that

$$\mathbb{E}[D(G(\mathbf{z};\theta^\star))] = 1-\eta.$$

■

This corollary gives the training method to obtain the optimal parameter vector $\theta^\star$ leading to

$$Pr\{G(\mathbf{z};\theta^\star) \in \mathcal{S}\} \geq \eta. \tag{23}$$

B. Discriminators

The training of a successful generator requires knowledge of l_1 and l_2 functions. However, it is hard to obtain their closed-form expression. Data-driven approximation offers a practical solution, as it only requires runtime data instead of a high-fidelity model and provides easy calculation of gradients, reducing computation overhead [21]. Furthermore, research has shown that both fully-connected neural network (FCN) and recurrent neural network (RNN) could learn l_1, l_2 [21], [15].

In this paper, we use a deep regression neural network to approximate l_1:

$$a_E = f_1(\mathbf{e};\theta_E). \tag{24}$$

It is trained with the mean-square-error (MSE) loss function given N runtime effectiveness metric data $l_1(\mathbf{e})$ with the corresponding injection of attacks $\mathbf{e}$:

$$\underset{\theta_E}{\text{Minimize:}} \frac{1}{N}\sum_{j=1}^{N}\left\| l_1(\mathbf{e})^{(j)} - f_1(\mathbf{e};\theta_E)^{(j)}\right\|_2^2. \tag{25}$$

The choice of a data-driven model for approximating l_2 depends on the choice of stealthiness index. If l_2 is a classification function outputting the probability of detecting attacks as a stealthiness index, then we use a deep classification network to approximate it. If l_2 is a function outputting stealthiness value, such as estimation residual, then we use a regression neural network to approximate it. Generally, we represent the data-driven model for l_2 by the following:

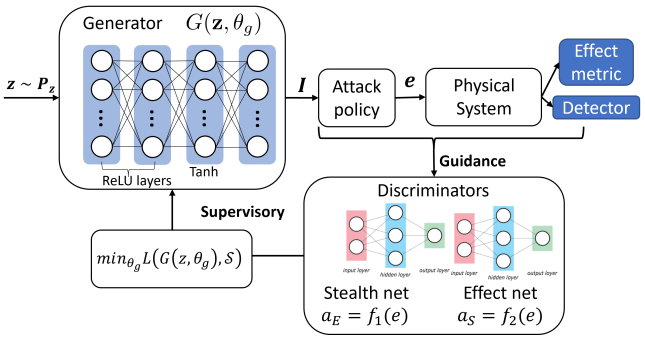

Fig. 2. Schematic description of the proposed data-driven vulnerability analysis framework

$$a_S = f_2(\mathbf{e};\theta_S). \tag{26}$$

If it is a regression network, it is trained with the MSE loss function in (25). If it is a classification network, it is trained with the binary cross-entropy loss function given N runtime stealthiness metric data $l_2(\mathbf{e})$ with the corresponding injection of attacks $\mathbf{e}$:

$$\underset{\theta_S}{\text{Minimize:}} -\frac{1}{N}\sum_{j=1}^{N} f_2(\mathbf{e};\theta_S) log(l_2(\mathbf{e})) + (1-f_2(\mathbf{e};\theta_S)) log(1-l_2(\mathbf{e})). \tag{27}$$

C. Training Algorithm

Consequently, a summary of the proposed data-driven attack generation framework is shown in Fig. 2. The runtime data is generated from the physical experiment which provides guidance for the training of the discriminators. Then, the trained discriminators supervise the learning process of the generator.

The generator maps random samples $\mathbf{z} \sim \mathbf{P}_z$ to the parameter vector $\mathbf{I} \in \mathbb{R}^l$ of a pre-defined attack policy $\pi(\mathbf{I})$. Given $\mathbf{I} \in \mathbb{R}^l$, the attack policy $\pi(\mathbf{I})$ is a deterministic time sequence of the injected attacks given by

$$\pi(\mathbf{I}) \triangleq \{t_0(\mathbf{I}), T(\mathbf{I}), g(\mathbf{I})\}. \tag{28}$$

It comprises the start time of the attack injection $t_0(I)$, the duration of the attack injection $T(I)$, and the attack profile $g(\mathbf{I}) : [t_0(\mathbf{I}) \;\; t_0(\mathbf{I}) + T(\mathbf{I})] \to \mathbb{R}^n$. The specific attack policy is assumed to be predetermined for the purpose of this paper. By using a pre-defined attack policy, the complexity of searching for feasible attacks is reduced since the attacks are searched from a function space as a result, rather than infinite-dimensional point-wise ambient space.

Biased learning is one of the issues of deep generative models [29], especially when it is trained with multiple interrelated data-driven models. The discriminators might end up learning a biased system vulnerability evaluation based on a limited space of attacks covered by the generative model. This in turn affects the training of the generator network. To combat biased learning, it's crucial to have a well-rounded

Algorithm 1 Training algorithm of the physics-guided attack generative model

Hyperparameters: α (effectiveness threshold), ε (stealthiness threshold), η (success probability), π (attack policy), s (scale factor of leakyReLU function)

- Pass random attack policy inputs, obtain *random attack dataset* $\{[I_1,I_2,I_3],[a_E,a_S]\}_{rand}$, where $I_i = rand$ for $i = 1,2,3$;

For i in N_{epoch} **do**:

- Sample minibatch of m random samples $\{\mathbf{z}^{(1)},\mathbf{z}^{(2)},\cdots,\mathbf{z}^{(m)}\}$ drawn from $\mathbf{z}^{(I)} \sim \mathbf{P}_z$;
- Pass the random samples through the generator to obtain $[I_1,I_2,I_3] = G(z,\theta_{i-1})$;
- Pass the generator outputs to the simulation, obtain *generated attack dataset* $\{[I_1,I_2,I_3],[a_E,a_S]\}_{gen}$;
- Train discriminators using both *random attack dataset* and the current *generated attack dataset* $\rightarrow$ (25), (27);
- Train the generator using the trained discriminators in the loss function (21), where $l_1(\mathbf{e}) = f_1(\mathbf{e},\theta_{E_i})$ and $l_2(\mathbf{e}) = f_2(\mathbf{e},\theta_{S_i})$.

End For

and generalizable training dataset for the discriminators. This paper addresses the issue by incorporating both random and generated attack datasets into the training process for the discriminators. The *random attack dataset* is created by simulating random attack policies through a physical system, a metric calculator, and a BDD. The *generated attack dataset*, on the other hand, uses inputs $[I_1,I_2,I_3]$ generated by the last trained generator. The entire training procedure is outlined in Algorithm 1.

V. CASE STUDIES

In this section, we demonstrate the proposed learning-based vulnerability analysis framework in networked control systems (NCS). Two case studies will be presented: a power grid system and a networked gas pipeline system.

A. Power Grid System

In this subsection, we consider a power grid system containing n_g generator buses and n_l load buses. An example on IEEE 14-bus system ($n_g = 5, n_l = 14$) is shown in Fig. 3. It is assumed to be equipped with IoT net meters and frequency meters which measure the net power of buses and generators' frequency. A control center collects the sensor readings through a communication network and performs state estimation, bad data detection, and regulation control of the generator's rotor angles and frequency. The attacker could either spoof the smart meters or compromise the communication channels to maliciously modify sensor readings received by the control center.

Firstly, we introduce the system dynamical models and control setup. Without loss of generality, several assumptions are admitted [30]:

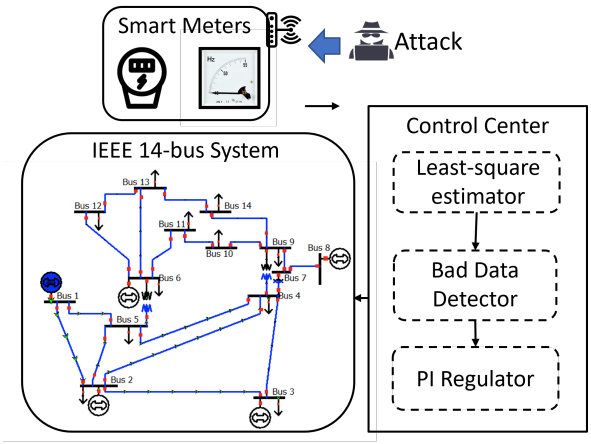

Fig. 3. Schematic description of cyber-physical IEEE 14-bus system

1) The network voltages stay constant at $v = 1p.u.$;
2) The angular difference between each bus is small;
3) The conductance is negligible, therefore the system is lossless.

The active power flow dynamics at bus i is represented as

$$\mathbf{0} = -bsin(\theta_i - \theta_j) - \mathbf{P}_i(\theta) \tag{29}$$

where θ_i denotes i-th bus angle, $\mathbf{P}_i$ denotes the active power at the bus i, and b is the associated susceptance. Since the power flow dynamics converge much quicker than the mechanical power injection at buses, the dynamical equation in (30) is already at its steady status. Then, given $\Psi(\theta)$ defined as in (2) with $\psi(\theta_i,\theta_j) = -bsin(\theta_i - \theta_j)$, the networked power flow dynamics is described by

$$\mathbf{0} = (A \odot \Psi(\theta))\mathbf{1} - \mathbf{P}(\theta). \tag{30}$$

By ordering the buses such that the generator nodes appear first, the admittance-weighted Laplacian matrix can be expressed as $L = \begin{bmatrix} L_{gg} & L_{lg} \\ L_{gl} & L_{ll} \end{bmatrix} \in \mathbb{R}^{N\times N}$, $N = n_g + n_l$, and A is the corresponding adjacent matrix. In addition, we can write the network power active power vector as $\mathbf{P}(\theta) = \begin{bmatrix} \mathbf{P}_g(\theta) \\ \mathbf{P}_l(\theta) \end{bmatrix}$, where $\mathbf{P}_g(\theta), \mathbf{P}_l(\theta)$ are the active power vector at generator buses and load buses. Furthermore, the nonlinear swing dynamics of generator buses are described by a differential algebraic equation (DAE) [30]:

$$\begin{bmatrix} I & 0 \\ 0 & M_g \end{bmatrix} \begin{bmatrix} \dot{\delta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & -D_g \end{bmatrix} \begin{bmatrix} \delta \\ \omega \end{bmatrix} + \begin{bmatrix} 0 \\ P_g - \mathbf{P}_g(\theta) \end{bmatrix}, \tag{31}$$

where $\delta \in \mathbb{R}^{n_g}$ is the vector of generator rotor angles, $\omega \in \mathbb{R}^{n_g}$ is the generator frequency vector, M_g is a diagonal matrix of inertial constants for each generator and D_g is a diagonal matrix of damping coefficients, and P_g denotes net power injected at the generator buses. In addition, at load buses, the active power $\mathbf{P}_l(\theta)$ is equal to the injected power P_l, so that

$$\mathbf{P}_l(\theta) = P_l \tag{32}$$

Next, we choose an equilibrium point of the system $\theta_i = \theta_j$, then linearizing the network dynamical model in (31) yields

$$\begin{aligned}\dot{\mathbf{x}} &= \underbrace{\begin{bmatrix} 0 & I \\ -M_g^{-1}(L_{gg} - L_{gl}L_{ll}^{-1}L_{lg}) & M_g^{-1}D_g \end{bmatrix}}_{A}\mathbf{x} \\ &\quad + \underbrace{\begin{bmatrix} 0 & 0 \\ M_g^{-1} & -M_g^{-1}L_{gl}L_{ll}^{-1} \end{bmatrix}}_{B}\mathbf{u}, \\ \mathbf{y} &= \underbrace{\begin{bmatrix} 0 & I \\ -P(\theta)L_{ll}^{-1}L_{lg} & 0 \end{bmatrix}}_{C}\mathbf{x} + \underbrace{\begin{bmatrix} 0 & 0 \\ P(\theta)L_{ll}^{-1} & 0 \end{bmatrix}}_{D}\mathbf{u}, \\ \theta &= -L_{ll}^{-1}(L_{lg}\delta - P_l),\end{aligned} \tag{33}$$

where the state variables $\mathbf{x} = [\delta^\top \ \omega^\top]^\top \in \mathbb{R}^{2n_g}$ contain the generator rotor angles and the generator frequency, the control input vector $\mathbf{u} = [P_g^\top \ P_l^\top]^\top \in \mathbb{R}^{n_g+n_l}$ contain the power injected at the generator buses and load buses, and the measurements $\mathbf{y} = [\omega^\top \ P_{net}^\top]^\top \in \mathbb{R}^{n_g+n_l}$ contain the generator frequency and the net power injected at the buses. With a sample time $T_s = 0.01s$, we obtain the discrete-time model of (33):

$$\begin{aligned}\mathbf{x}(k+1) &= A_d\mathbf{x}(k) + B_d\mathbf{u}(k) \\ \mathbf{y}(k) &= C\mathbf{x}(k) + D\mathbf{u}(k)\end{aligned} \tag{34}$$

where $A_d = AT_s - I$ and $B_d = BT_s$. A Luenberger observer is employed to estimate the states $\mathbf{x}(k+1)$ based on the discrete model in (34):

$$\hat{\mathbf{x}}(k+1) = A_d\hat{\mathbf{x}}(k) + B_d\mathbf{u}(k) + L(\mathbf{y}(k) - C\hat{\mathbf{x}}(k) - D\mathbf{u}(k)) \tag{35}$$

where L is the observer gain satisfying that $A_d - LC$ has all the eigenvalues inside the unit circle. Then, a PI controller is used to regulate the generators' rotor angles and frequencies. Given a reference profile $\mathbf{x}^r(k) = [\delta^{r\top} \ \omega^{r\top}]^\top$, let $\tilde{\mathbf{x}}(k) = \hat{\mathbf{x}}(k) - \mathbf{x}^r(k)$, the controller is given by

$$\mathbf{u}(k) = K_P\,\tilde{\mathbf{x}}(k) + K_I\int_0^k \tilde{\mathbf{x}}(\tau)\mathrm{d}\tau. \tag{36}$$

Consider a candidate Lyapunov function

$$V = \frac{1}{2}\begin{bmatrix}\tilde{\mathbf{x}}' \\ \tilde{\mathbf{x}}\end{bmatrix}^\top \begin{bmatrix}\tilde{\mathbf{x}}' \\ \tilde{\mathbf{x}}\end{bmatrix}, \tag{37}$$

where $\tilde{\mathbf{x}}' \triangleq \int_0^t \tilde{\mathbf{x}}(\tau)d\tau$, the control gains K_P, K_I are designed to satisfy

$$\begin{bmatrix} 0 & I \\ BK_I & A + BK_P \end{bmatrix} \prec 0. \tag{38}$$

Next, we discuss the implementation of the proposed attack generator on this IEEE 14-bus system. The attacks are injected through the measurements

$$\mathbf{y} = C\mathbf{x} + D\mathbf{u} + \mathbf{e}.$$

The total simulation time is set as $8s$, thereby $T_{sim} = 8/T_s = 800$ time steps in total. We choose the ratio of the mean

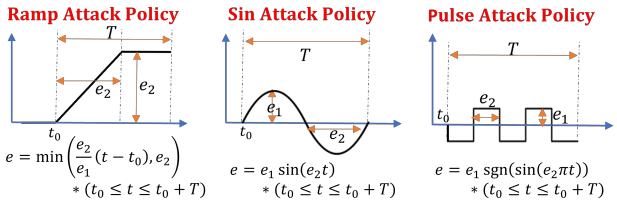

Fig. 4. Three different attack policies tested in the case study

control error of the generator rotor angles to the maximum nominal[1] generator angles to be the effectiveness index

$$a_E = l_1(\mathbf{e}) = \frac{\frac{1}{T_{sim}}\sum_{k=0}^{T_{sim}} \|\delta(k) - \delta^r(k)\|_2}{\max\limits_{0\le k\le T_{sim}} \|\delta_{nominal}(k)\|_2}, \tag{39}$$

and choose the maximum estimation residual to be the stealthiness index

$$a_S = l_2(\mathbf{e}) = \max_{0\le k\le T_{sim}} \|\mathbf{y}(k) - D\mathbf{u}(k) - C\hat{\mathbf{x}}(k)\|_2 \tag{40}$$

Three attack policies are tested in this case study: Ramp attack (RA) policy, Pulse attack (PA) policy, and Sin attack (SA) policy, as shown in Fig. 4. As defined in (28), all three attack policies have two common time parameters: start time of the attack injection $t_0 \in [0.1T_{sim}, 0.2T_{sim}]$ and duration time of the attack injection $T \in [1e-5, T_{sim}]$. Then the attack functions g are given with two parameters e_1, e_2. For RA policy, it is given as

$$g_{RA}(k) = \min\left\{\frac{e_2}{e_1}(k-t_0), e_2\right\} \odot (t_0 \le k \le t_0+T), \tag{41}$$

where the range of the parameters are chosen as $e_1 \in [0.1T_{sim}, 0.6T_{sim}]$, $e_2 \in [-0.04, 0.04]$. For SA policy, it is given as

$$g_{SA}(k) = e_1 sin(e_2 k) \odot (t_0 \le k \le t_0+T), \tag{42}$$

where $e_1 \in [-0.04, 0.04]$, $e_2 \in [-5, 5]$. For PA policy, it is given as

$$g_{PA}(k) = e_1 sgn\,(sin(e_2\pi k)) \odot (t_0 \le k \le t_0+T), \tag{43}$$

where $e_1 \in [-0.04, 0.04]$, $e_2 \in [-5, 5]$. For each measurement node, $\mathbf{I} = \begin{bmatrix} I_1 & I_2 & I_3 & I_4 \end{bmatrix}$, where $I_i \in [0,1]$ for all $i = 1,2,3,4$, is outputted by the generator. For the sake of clean presentation, let $\mathbf{x}_a = \begin{bmatrix} t_0 & T & e_1 & e_2 \end{bmatrix} \in [\underline{\mathbf{x}}_a, \bar{\mathbf{x}}_a]$, then the inputs of the policies are given by

$$\mathbf{x}_a = \underline{\mathbf{x}}_a + \mathbf{I}(\bar{\mathbf{x}}_a - \underline{\mathbf{x}}_a). \tag{44}$$

The total number of measurements is 19, so the output size of the generator for all attack policies is $19*4 = 76$. It is also the input size of the two discriminators. The input size of the generator is 10, and the output sizes of the discriminators are both 1. The generator has 4 layers: *ReLU*, *ReLU*, *Tanh*, *Sigmoid*. The numbers of neurons at the hidden layers are $500, 1000, 500$, respectively. The stealth net consists of 3 *ReLU* hidden layers and *Sigmoid* output layer, and the effect

[1] nominal means no attack.

net consists of 3 *ReLU* hidden layers and *Linear* output layer. The numbers of neurons at hidden layers are $500, 1000, 500$ for both effect net and stealth net. In each training epoch, the discriminators are trained in 20 batches with the size of 2000, and the generator is trained in 10 batches with the size of 5000.

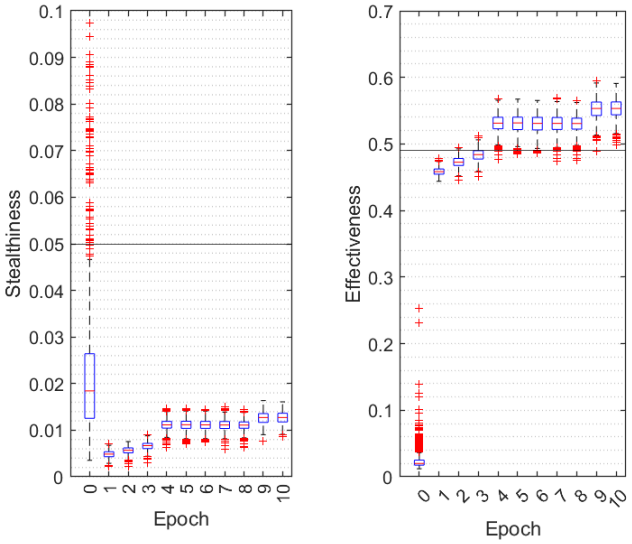

Fig. 5. Testing performance of the attack generator with Ramp attack policy after each epoch (0 epoch means pre-training)

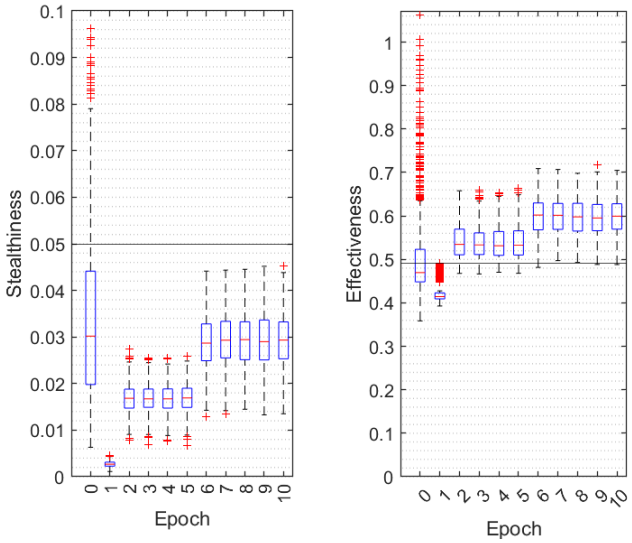

Fig. 6. Testing performance of the attack generator with Sin attack policy after each epoch (0 epoch means pre-training)

All three attack policies are used separately in the training process of Algorithm 1. The same stealthiness and effectiveness thresholds are chosen: $\varepsilon = 0.05$ and $\alpha = 49\%$. For Algorithm 1, other hyperparameters are chosen as $\eta = 0.8$ and $s = 0$. In Fig. 5, Fig. 6 and Fig. 7, we show the testing results of the trained attack generator in the system simulation after each epoch for those three attack policies. All testing results indicate the successful training of the attack generation framework. In the beginning, the generator

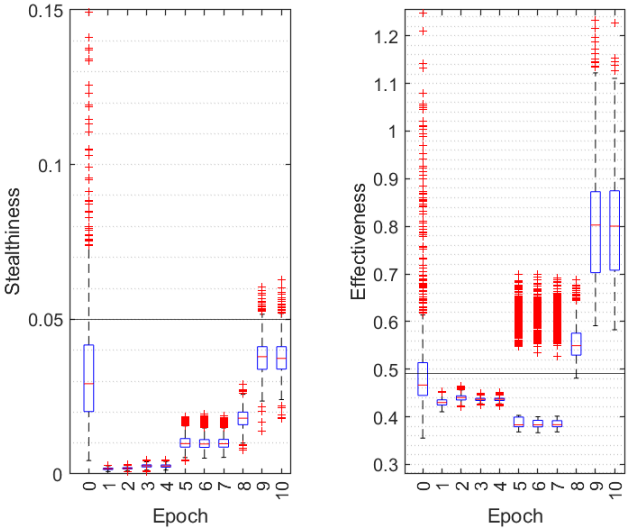

Fig. 7. Testing performance of the attack generator with Pulse attack policy after each epoch (0 epoch means pre-training)

Epochs	Ramp Attack	Sin Attack	Pulse Attack
1	0%	0.2%	0%
2	0.3%	92.3%	0%
3	23%	93.8%	0%
4	99.5%	93.7%	0%
5	99.4%	91.4%	24.8%
6	99.5%	99.8%	23.5%
7	99.6%	100%	23.2%
8	99.5%	100%	99.4%
9	99.9%	99.8%	97.4%
10	100%	99.8%	96.9%

TABLE I

TESTING SCORES OF THREE ATTACK POLICIES AFTER TRAINING

tried to lower the stealthiness, but the effectiveness was also sacrificed due to the positive correlation between the effectiveness and stealthiness. Once the stealthiness is below the threshold, the generator tried to explore more effective attacks until the effectiveness reached beyond the given threshold. By comparing the testing performance policy-wise, it is seen that the attack generation framework could approach the boundary of stealthiness more and achieve bigger effectiveness with SA policy and PA policy compared to RA policy. This indicates that SA policy and PA policy enable more vulnerability space, where the attack generation framework could explore, than RA policy. In addition, the corresponding testing scores, the ratio of feasible attacks in $\mathcal{S}(0.05, 49\%)$ to total testing samples, in epochs are shown in Table I. It shows that the generator with SA policy converges faster than the other two, and the generator with PA policy is the slowest possibly due to its complexity.

Next, to compare the performance of the generated attack with different attack policies, we inject the generated attack into the system's time-series simulation. Based on tabel I, we used the trained attack generators at epoch 10 for RA policy and at epoch 8 for SA and PA policy. The examples of the time-series performance of the generated attacks are shown

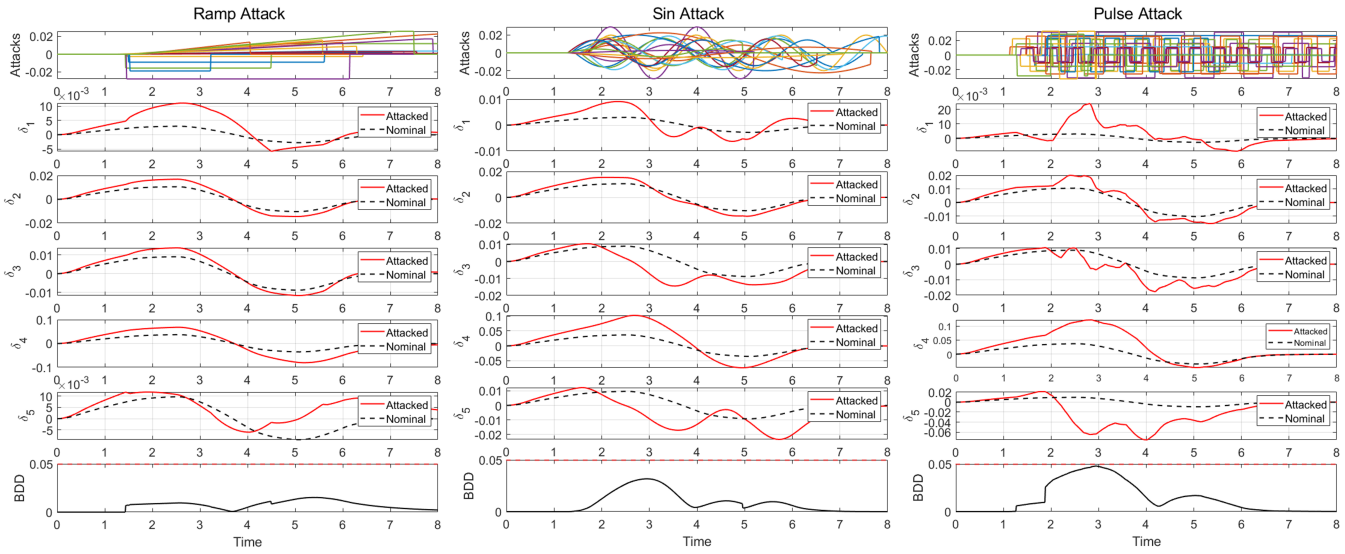

Fig. 8. Time-series performance of generated attacks by the trained attack generators with different attack plocies

in Fig. 8. The first row shows the attack signals generated by the trained attack generators for all 19 sensors. The second to sixth rows show the comparison of generator rotor angles before and after the attack injection. The last row shows the outputted estimation residual (stealthiness index) by BDD. It is seen that PA policy with the proposed attack generation framework could explore the most vulnerable space of the system since the biggest BDD outputs already touch the detection boundary during the simulation. In addition, different attack policy produces different form of effects on the system. RA policy is more like introducing a constant bias on sensor readings, so the shapes of the attacked rotor angles are similar to the nominal ones. SA policy introduces waves in the rotor angles and PA policy introduces higher-frequency fluctuations in rotor angles. However, whichever attack policy you choose, the proposed attack generation framework could find the corresponding feasible attack set.

B. Gas Pipeline System

In this subsection, we consider gas pipeline systems equipped with IoT sensors and actuators, a supervisory controller, and automatic controllers, as shown in Fig. 9.

A nonlinear creep flow model is used to describe the gas flow in a pipeline segment [31]:

$$\begin{aligned}\frac{\partial p}{\partial t} &= -\frac{c^2\rho}{G}\cdot\frac{\partial Q_n}{\partial x}\\ \frac{\partial p}{\partial x} &= -\frac{2f\rho^2c^2Q_n^2}{DG^2p},\end{aligned} \tag{45}$$

where p is the pressure in the pipeline, x is the position of flow, c is the speed of sound in gas [m/s], ρ is the average gas density over the cross-section area of the pipeline [kg/m^3], Q_n is the volumetric flow at standard conditions, f is the friction factor, D is the diameter of the pipeline and $G = \pi(D/2)^2$. While the partial differential equation (PDE) model provides an accurate description of the gas flow dynamics

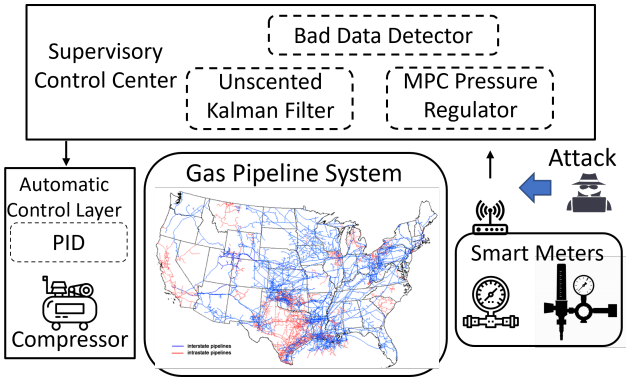

Fig. 9. Schematic description of networked cyber-physical gas pipeline system

for infrequent online optimization, an ordinary differential equation (ODE) model is sufficient to describe the pressure dynamics at nodes [31], [32]. The dynamical pressure model at node i is given as

$$\dot{p}_i = \frac{c^2}{V_i}\sum_{j\in\mathcal{N}_i}\sqrt{\frac{|p_j^2-p_i^2|}{K_{ij}}}\,\mathrm{sgn}(p_j-p_i) - w_i, \tag{46}$$

where

$$K_{ij} = \frac{64 f_{ij} c^2 \Delta x_{ij}}{\pi^2 D_{ij}^5},\quad V_i = \frac{\pi}{8}\sum_{j\in\mathcal{N}_i} D_{ij}^2\Delta x_{ij},$$

and w_i is the mass flow at pipeline i, Δx_{ij} denotes the length of pipeline between node i and node j, $\mathcal{N}_i$ denote the set of neighborhood pipelines of the pipeline i. Let

$$\psi(p_i,p_j) \triangleq \sqrt{\frac{|p_j^2-p_i^2|}{K_{ij}}}\,\mathrm{sgn}(p_j-p_i), \tag{47}$$

then the model of the entire pipeline network is given by

$$\begin{aligned}\dot{\mathbf{p}} &= c^2V^{-1}\left(A\odot\Psi(\mathbf{p})\right)\mathbf{1} - \mathbf{w}\\ &= c^2V^{-1}\left(A\odot\Psi(\mathbf{p})\right)\mathbf{1} + B_{\mathrm{sup}}\mathbf{w}_{\mathrm{sup}} - B_{\mathrm{dem}}\mathbf{w}_{\mathrm{dem}}\end{aligned} \tag{48}$$

where $\Psi(\theta)$ is defined as in (2) with $\psi(\theta_i,\theta_j)$ defined in (47), $V^{-1} = \mathsf{diag}([V_1^{-1}, V_2^{-1},\ldots,V_n^{-1}])$, $\mathbf{w}_{\text{sup}}$ is a vector of mass inflows at the supply points and $\mathbf{w}_{\text{dem}}$ is a vector of mass outflows at the demand points. Clearly $\mathbf{w} = -B_{\text{sup}}\mathbf{w}_{\text{sup}} + B_{\text{dem}}\mathbf{w}_{\text{dem}}$, where B_{dem} and B_{sup} are appropriately dimensioned matrices such that $B_{\text{sup}}^{\top}B_{\text{dem}} = 0$ and $P\begin{bmatrix} -B_{\text{sup}} \mid B_{\text{dem}} \end{bmatrix} = I$ for some projection matrix P. In addition, a dynamical model of the compressor between two nodes i and j is given by [33]:

$$\begin{aligned} \dot{\omega}_{ij} &= \frac{1}{J_{0_i}}(u_{ij} + \rho r_2^2 q_{ij}\omega_{ij}) \\ \dot{q}_{ij} &= \frac{A_c}{L_c}(\phi(\omega_{ij},q_{ij})\,p_i - p_j) \\ \dot{w}_{ij} &= q_{ij}, \end{aligned} \tag{49}$$

where

$$\phi(\omega,q) = \left(1 + \frac{1}{T_{in}c_p 1000}\left(\rho r_2^2\omega^2 + k_f q^2 - \frac{r_1^2}{2}\left(\omega - \frac{q}{A_c r_1 \rho_{in}}\right)^2\right)\right)^{\frac{\gamma}{\gamma-1}},$$

ω_{ij} is the compressor rotor angular velocity [rad/s], q_{ij} is the mass power flow [kg/s], $\phi(\omega,q)$ is the compressor characterization, u_{ij} is the mechanical torque [N-m] applied to the rotor shaft or inertia J_{0_i}[kg-m^2]. p_i and p_j are the input and output pressures of the compressor, respectively. For the system considered, the compressors are at the extremes of the network. Thus, at each compressor node, one of p_i and p_j will be the ambient pressure. $p_i = P_{\text{amb}}$ for upstream compressors, while $p_j = P_{\text{amb}}$ for downstream compressors. The compressor could be controlled by a simple PID controller:

$$u_{ij} = K_P\,\tilde{\mathbf{w}}_{ij}(t) + K_I\int_0^t \tilde{\mathbf{w}}_{ij}(\tau)\mathrm{d}\tau + K_D\frac{\mathrm{d}\tilde{\mathbf{w}}_{ij}(t)}{\mathrm{d}t}, \tag{50}$$

where $\tilde{\mathbf{w}}_{ij}(t) = \mathbf{w}_{ij} - \mathbf{w}_{ij}^r$. Similarly, the PID gains K_P, K_I, K_D are designed to achieve the closed-loop performance in (6). Linearizing the models (48) around the equilibrium point $(\mathbf{p}_{eq},\mathbf{w}_{eq})$ and discretizing with a fixed time step $T_s = 0.01s$ yields

$$\tilde{\mathbf{p}}_{k+1} = A\tilde{\mathbf{p}}_k + B\tilde{\mathbf{w}}_k, \tag{51}$$

where $\tilde{\mathbf{p}}_k = \mathbf{p}_k - \mathbf{p}_{eq}$, $\tilde{\mathbf{w}}_k = \mathbf{w}_k - \mathbf{w}_{eq}$, and

$$A = \left.\frac{\partial f}{\partial \mathbf{p}}\right|_{(\mathbf{p}_{eq},\mathbf{w}_{eq})} T_s + I_n,\quad B = \left.\frac{\partial f}{\partial \mathbf{w}}\right|_{(\mathbf{p}_{eq},\mathbf{w}_{eq})} T_s,\quad C = \left.\frac{\partial g}{\partial \mathbf{p}}\right|_{\mathbf{p}_{eq}}.$$

The supervisory target generator is given in (11) to regulate the node pressures around its equilibrium point at a lower frequency. We assume all node pressure and mass flow rate could be directly measured subject to noise $\mathbf{v}$:

$$\mathbf{y} = \mathbf{x}' + \mathbf{v}. \tag{52}$$

where $\mathbf{x}' \triangleq \begin{bmatrix} \mathbf{x} \\ \mathbf{w}_{sup} \\ \mathbf{w}_{dem} \end{bmatrix}$ composes of the node pressure states and the wellhead supplied mass flow and the demand mass flow. An unscented Kalman filter (UKF) [34] is used to estimate the states from the noisy measurements. Three different pipeline systems with different network topologies are tested in this case study, as shown in Fig 10: Linear topology, Tree topology, and Cyclic Topology [35]. The linear topology has 9 nodes, the tree topology has 11 nodes, and the cyclic topology has 13 nodes. All three topological pipeline systems have two wellheads with gas supplies, one demand node, and one supply node that needs to be controlled.

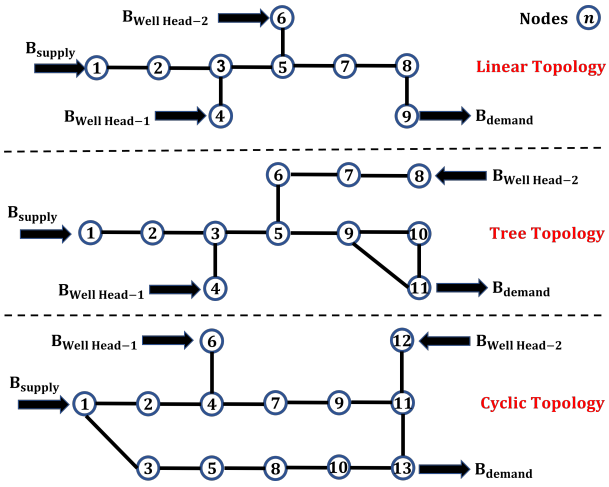

Fig. 10. Three different pipeline topologies tested in the case study

Next, we discuss the implementation of the proposed attack generator on these pipeline systems. The attacks are injected through the measurements, then the measurement model in (52) becomes

$$\mathbf{y} = \mathbf{x}' + \mathbf{v} + \mathbf{e} \tag{53}$$

The total simulation time is set as $200s$, thereby $T_{sim} = 200/T_s = 20000$ time steps in total. To train the attack generator, we use the mean value of the node pressure control error as the effectiveness metric:

$$a_E = l_1(\mathbf{e}) = \sum_{k=0}^{T_{sim}} \|\tilde{\mathbf{p}}(k)\|_2, \tag{54}$$

and use the maximal estimation residual as the stealthiness metric:

$$a_S = l_2(\mathbf{e}) = \max_{0\le k\le T_{sim}} \left\|\mathbf{y}(k) - \mathbf{x}'(k)\right\|_2 \tag{55}$$

In this case study, we only test the Ramp attack policy, defined in (41), with the pre-defined range of injection start time $t_0 \in [0.1T_{sim}, 0.2T_{sim}]$, range of injection duration time $T \in [1e-5, 0.5T_{sim}]$ and the range for two attack function parameters $e_1 \in [0.1T_{sim}, T_{sim}]$, $e_2 \in [0, 0.5*\mathbf{y}]$. The input of the attack policy function (41) is given in (44), where $\mathbf{I}$ is the output of the generator. For linear topological pipeline systems, it has 12 measurements, so the output size of the associated generator is $4*12 = 48$. Similarly, the output size of the generator for the tree topological pipeline system is $4*14 = 56$ and the output size of the generator for the cyclic topological pipeline system is $4*16 = 64$. The input sizes of the generators are the same and equal to 10, and they all have 4 layers, *ReLU* (500 neurons), *ReLU* (1000 neurons), *Tanh* (500 neurons), *Sigmoid*. The stealth net consists of 3 *ReLU* hidden layers and *Sigmoid* output layer, and the effect net

consists of 3 *ReLU* hidden layers and *Linear* output layer. The numbers of neurons at hidden layers are $500, 1000, 500$ for both effect net and stealth net. In each training epoch, the discriminators are trained in 20 batches with the size of 1000, and the generator is trained in 5 batches with the size of 5000. All generators converge in the first epochs, so next, we present the testing performance of the generators from batch to batch in the first epoch.

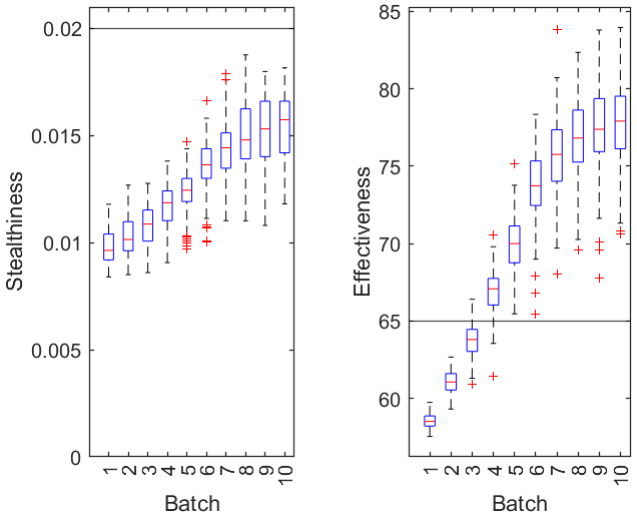

Fig. 11. Testing performance of the attack generator with Ramp attack policy on the Linear topological pipeline system in first epoch (10 batches)

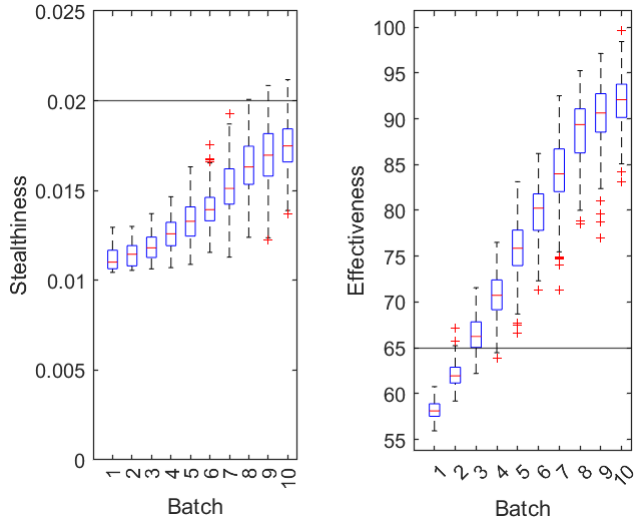

Fig. 12. Testing performance of the attack generator with Ramp attack policy on the Tree topological pipeline system in first epoch (10 batches)

As shown in Fig. 11, Fig. 12, and Fig. 13, it is seen that the generator for the linear topological system tried to increase the effectiveness beyond the threshold while the stealthiness is already below the threshold. Due to the positive correlation between effectiveness and stealthiness, stealthiness was sacrificed when the effectiveness is increased but the generator was able to stop increasing effectiveness further when stealthiness reached close to the boundary. Upon comparing

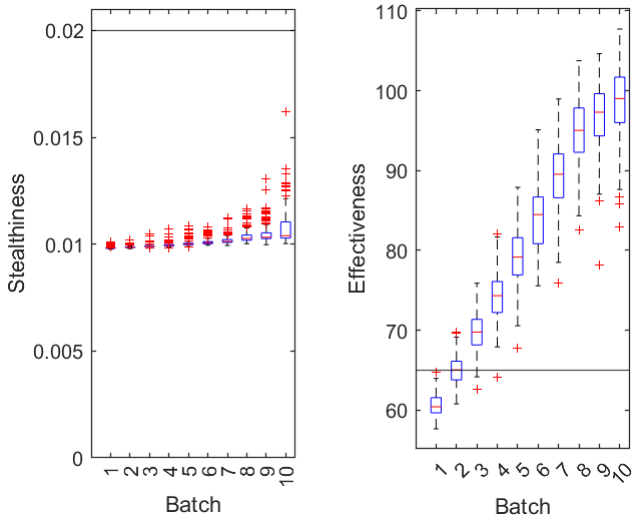

Fig. 13. Testing performance of the attack generator with Ramp attack policy on the Cyclic topological pipeline system in first epoch (10 batches)

Batches	Linear	Tree	Cyclic
1	0%	0%	0%
2	0%	2.78%	51.1%
3	12.78%	76.1%	96.7%
4	93.33%	97.8%	99.4%
5	100%	100%	100%
6	100%	100%	100%
7	100%	100%	100%
8	100%	99.4%	100%
9	100%	96.7%	100%
10	100%	97.8%	100%

TABLE II

TESTING SCORES OF THREE RAMP ATTACK POLICY ON DIFFERENT TOPOLOGICAL PIPELINE SYSTEMS

the testing performance topology-wise, it is seen that within the same stealthiness threshold, the proposed attack generator could explore more effective attacks for cyclic topology than for tree topology, and explore more effective attacks for tree topology than linear topology. In addition, the testing scores of the trained generators after each batch are shown in Table II. It is seen all generators could converge at the fifth batch for different topological pipeline systems.

Next, we injected the generated feasible attacks in those three pipeline systems, and the time-series performances are shown in Fig. 14, Fig. 15, and Fig. 16. On the left top plots, the attack signals $\mathbf{e}$ are shown. The injection start times are between $0.1T_{sim} = 20s$ and $0.2T_{sim} = 40s$, and the attack signals have ramp shapes. The left bottom plots show the BDD output which is the estimation residual $\|\mathbf{y}(k) - \mathbf{x}'(k)\|_2$. It is seen the proposed generator could approach the stealthiness boundary most for tree topology, which means more vulnerability space of the tree topological pipeline network was explored by the generator. In addition, more vulnerability space of cyclic topology could be explored than a linear topology. These indicate that more nodes and more complicated topology of the pipeline network provide more possibility for the proposed attack generation

framework to explore more vulnerability space. Furthermore, the right plots show the node pressures at network nodes during the time-series simulation of attack injection. It is seen that the generated feasible attacks could mislead the system to deviate from the target equilibrium point and arrive at a wrong equilibrium point.

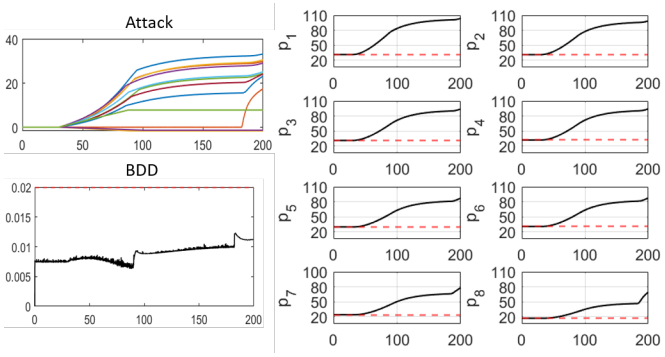

Fig. 14. Time-series performance of an example generated attack on linear topological pipeline system

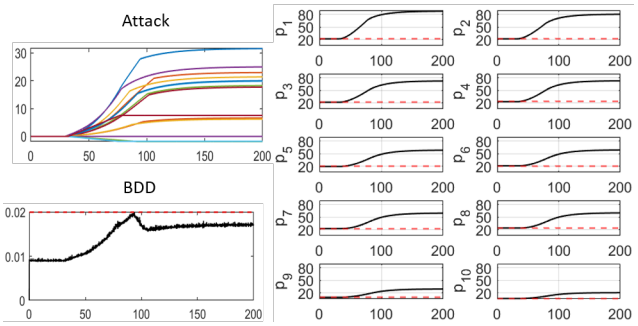

Fig. 15. Time-series performance of an example generated attack on tree topological pipeline system

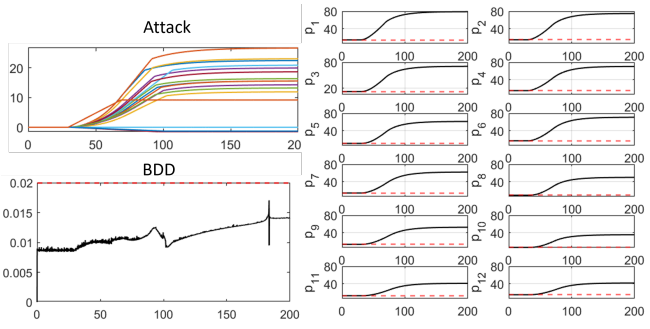

Fig. 16. Time-series performance of an example generated attack on cyclic topological pipeline system

VI. CONCLUSIONS

This paper introduces a physics-guided attack generative model for CPS vulnerability analysis. The framework consists of three interactive data-driven models: two discriminative models to evaluate attack signals and a generative model to generate feasible attacks. A new loss function is proposed to ensure successful attack generation with high probability. The data-driven approach enables wide application to various cyber-physical systems, allowing the learning of stealthiness boundaries while exploring effectiveness. This feature holds potential for use in resilient co-design with learning-based BDDs.

Future work includes addressing questions such as finding an optimal attack policy through an algorithm similar to policy learning, by adding constraints to the training environment. Additionally, further application experiments will be conducted.

REFERENCES

[1] M. Dahleh, E. Frazzoli, A. Megretski, S. Ozdaglar, P. Parrilo, and D. Shah, "Information and control: a bridge between the physical and decision layers," in *NSF Workshop on CyberPhysical Systems*, 2006.

[2] Q. Zhu and L. Bushnell, "Networked cyber-physical systems: Interdependence, resilience and information exchange," in *2013 51st Annual Allerton conference on communication, control, and computing (Allerton)*. IEEE, 2013, pp. 763–769.

[3] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei, "A survey of distributed optimization and control algorithms for electric power systems," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2941–2962, 2017.

[4] A. Marino and E. Zio, "A framework for the resilience analysis of complex natural gas pipeline networks from a cyber-physical system perspective," *Computers & Industrial Engineering*, vol. 162, p. 107727, 2021.

[5] L. Zhang, J. Sun, and G. Orosz, "Hierarchical design of connected cruise control in the presence of information delays and uncertain vehicle dynamics," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 1, pp. 139–150, 2017.

[6] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: the next computing revolution," in *Proceedings of the 47th design automation conference*, 2010, pp. 731–736.

[7] F. Pasqualetti, F. Dörfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE transactions on automatic control*, vol. 58, no. 11, pp. 2715–2729, 2013.

[8] Q. Zhu and T. Basar, "Game-theoretic methods for robustness, security, and resilience of cyber-physical control systems: games-in-games principle for optimal cross-layer resilient control systems," *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 46–65, 2015.

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

[10] Y. Mo and B. Sinopoli, "False data injection attacks in control systems," in *Preprints of the 1st workshop on Secure Control Systems*, vol. 1, 2010.

[11] L. Hu, Z. Wang, Q.-L. Han, and X. Liu, "State estimation under false data injection attacks: Security analysis and system protection," *Automatica*, vol. 87, pp. 176–183, 2018.

[12] T. Sui, Y. Mo, D. Marelli, X. Sun, and M. Fu, "The vulnerability of cyber-physical system under stealthy attacks," *IEEE Transactions on Automatic Control*, vol. 66, no. 2, pp. 637–650, 2020.

[13] X. Liu, Z. Bao, D. Lu, and Z. Li, "Modeling of local false data injection attacks with reduced network information," *IEEE Transactions on Smart Grid*, vol. 6, no. 4, pp. 1686–1696, 2015.

[14] A.-Y. Lu and G.-H. Yang, "False data injection attacks against state estimation without knowledge of estimators," *IEEE Transactions on Automatic Control*, 2022.

[15] A. Khazraei, H. Pfister, and M. Pajic, "Resiliency of perception-based controllers against attacks," in *Learning for Dynamics and Control Conference*. PMLR, 2022, pp. 713–725.

[16] I. L. Carreno, A. Scaglione, A. Zlotnik, D. Deka, and K. Sundar, "An adversarial model for attack vector vulnerability analysis on power and gas delivery operations," *Electric Power Systems Research*, vol. 189, p. 106777, 2020.

[17] R. Fu, X. Huang, Y. Xue, Y. Wu, Y. Tang, and D. Yue, "Security assessment for cyber physical distribution power system under intrusion attacks," *IEEE Access*, vol. 7, pp. 75 615–75 628, 2018.

[18] K. Yan, X. Liu, Y. Lu, and F. Qin, "A cyber-physical power system risk assessment model against cyberattacks," *IEEE Systems Journal*, 2022.

[19] X. Ren and Y. Mo, "Secure detection: Performance metric and sensor deployment strategy," *IEEE Transactions on Signal Processing*, vol. 66, no. 17, pp. 4450–4460, 2018.

[20] M. Mohammadpourfard, F. Ghanaatpishe, M. Mohammadi, S. Lakshminarayana, and M. Pechenizkiy, "Generation of false data injection attacks using conditional generative adversarial networks," in *2020 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*. IEEE, 2020, pp. 41–45.

[21] A. Khazraei, S. Hallyburton, Q. Gao, Y. Wang, and M. Pajic, "Learning-based vulnerability analysis of cyber-physical systems," in *2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2022, pp. 259–269.

[22] C. Godsil and G. F. Royle, *Algebraic graph theory*. Springer Science & Business Media, 2001, vol. 207.

[23] G. F. Coulouris, J. Dollimore, and T. Kindberg, *Distributed systems: concepts and design*. pearson education, 2005.

[24] M. Pajic, J. Weimer, N. Bezzo, O. Sokolsky, G. J. Pappas, and I. Lee, "Design and implementation of attack-resilient cyberphysical systems: With a focus on attack-resilient state estimators," *IEEE Control Systems Magazine*, vol. 37, no. 2, pp. 66–81, 2017.

[25] E. D. Sontag, "A 'universal'construction of artstein's theorem on nonlinear stabilization," *Systems & control letters*, vol. 13, no. 2, pp. 117–123, 1989.

[26] ——, *Mathematical control theory: deterministic finite dimensional systems*. Springer Science & Business Media, 2013, vol. 6.

[27] Y. Zheng and O. M. Anubi, "Attack-resilient weighted l1 observer with prior pruning," in *American Control Conference*, 2021, pp. 340–345.

[28] ——, "Resilient observer design for cyber-physical systems with data-driven measurement pruning," in *Security and Resilience in Cyber-Physical Systems*. Switzerland: Springer, 2022, pp. 85–117.

[29] S. Zhao, H. Ren, A. Yuan, J. Song, N. Goodman, and S. Ermon, "Bias and generalization in deep generative models: An empirical study," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[30] E. Scholtz, "Observer-based monitors and distributed wave controllers for electromechanical disturbances in power systems," Ph.D. dissertation, Massachusetts Institute of Technology, 2004.

[31] A. Osiadacz, *Simulation and analysis of gas networks*. Gulf Publishing Company, Houston, TX, 1987.

[32] G.-Y. Zhu, M. A. Henson, and L. Megan, "Dynamic modeling and linear model predictive control of gas pipeline networks," *Journal of Process Control*, vol. 11, no. 2, pp. 129–148, 2001.

[33] H. Perez-Blanco and T. B. Henricks, "A gas turbine dynamic model for simulation and control," in *Turbo Expo: Power for Land, Sea, and Air*, vol. 78637. American Society of Mechanical Engineers, 1998, p. V002T03A002.

[34] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *Signal processing, sensor fusion, and target recognition VI*, vol. 3068. International Society for Optics and Photonics, 1997, pp. 182–193.

[35] R. Z. Ríos-Mercado and C. Borraz-Sánchez, "Optimization problems in natural gas transportation systems: A state-of-the-art review," *Applied Energy*, vol. 147, pp. 536–555, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261915003013