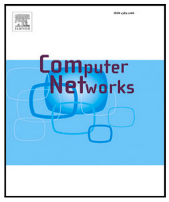
Computer Networks 251 (2024) 110655

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



CGAN-based cyber deception framework against reconnaissance attacks in ICS

Xingsheng Qin [a], Frank Jiang [a,*], Xingguo Qin [b,*], Lina Ge [c], Meiqu Lu [c,*], Robin Doss [a]

[a] Deakin Cyber Research and Innovation Centre, School of Information Technology, Deakin University, Geelong, VIC, Australia
[b] School of Computer Science and Information Security, Guilin University of Electronic and Technology, Guilin, China
[c] School of Artificial Intelligence, Guangxi Minzu University, Nanning, China

ARTICLE INFO

Keywords:
CGAN
Cyber deception
Hybrid defense
ICS
SDN

ABSTRACT

In recent years, Industrial Control Systems (ICSs) have faced increasing vulnerability to cyber attacks due to their integration with the Internet. Despite efforts to enhance cybersecurity, reconnaissance attacks remain a significant threat, prompting the need for innovative defensive strategies. This paper introduces a novel approach to strengthen the defensive capabilities of ICS networks against reconnaissance attacks using machine learning-driven cyber deception techniques. Leveraging Conditional Generative Adversarial Networks (CGANs), the proposed framework dynamically generates defensive network topologies to network shuffling and implement deception strategies, prioritizing system availability. Extensive simulations demonstrate the superior efficacy of the proposed framework in enhancing cybersecurity while minimizing computational overhead. By effectively mitigating reconnaissance attacks, this solution reinforces the resilience of ICS networks, safeguarding critical industrial infrastructure from evolving cyber threats. These findings underscore the significance of adopting machine learning-based cyber deception as a pragmatic security measure for protecting ICS networks in real-world industrial contexts.

1. Introduction

The convergence of information technology (IT) and operational technology (OT) systems, driven by the emergence of smart manufacturing and Industry 4.0, has introduced a new paradigm for ICSs. Despite the increased automation, enhanced connectivity, improved efficiency, and greater flexibility, this integration has also exposed legacy ICS devices to unprecedented cyber threats [1]. Recent incidents underscore the escalating risks, with ICS systems becoming prime targets for malicious attacks. The Stuxnet malware's 2010 exploitation of an Iranian nuclear power plant vulnerability potentially led to catastrophic consequences by granting remote control [2], alongside subsequent deliberate Supervisory Control and Data Acquisition (SCADA) intrusions affecting Ukrainian power companies in 2015 [3] and disruptive Distributed of Service (DoS) attacks on a Utah energy company in 2019 [4], emphasizing the critical necessity for robust cybersecurity in the ICS domain.

While IT security emphasizes the principles of confidentiality, integrity, and availability (CIA), operational technology demands distinct cybersecurity strategies, with availability taking precedence within ICS [5,6]. Unlike IT environments, OT systems face challenges in applying conventional patching and updates due to operational constraints, such as the imperative for continuous uptime and the risk of disrupting critical processes, leading to a prioritization of system availability over security measures and leaving OT systems vulnerable to cyber-attacks. Preventing early-stage cyber-attacks, especially during reconnaissance, is crucial in the ICS domain to mitigate significant financial and human losses [7,8].

Researchers have proposed various defensive strategies to protect ICSs from cyber threats. Strategies like honeypots, honeynets, and honeytokens have been recognized for their effectiveness in providing an additional layer of protection [9,10]. Later, Defensive cyber deception emerged as a potent strategy to enhance the cybersecurity of the ICS systems, encompassing various approaches such as monitoring, preplanned countermeasures, and deception techniques [11]. Recent advancements explore hybrid defense strategies, integrating cyber deception, Machine Learning (ML), or Moving Target Defense

* Corresponding authors.
E-mail addresses: qinxing@deakin.edu.au (X. Qin), frank.jiang@deakin.edu.au (F. Jiang), xgqin@guet.edu.cn (X. Qin), 66436539@qq.com (L. Ge), meiqu.lu@gxmzu.edu.cn (M. Lu), robin.doss@deakin.edu.au (R. Doss).

https://doi.org/10.1016/j.comnet.2024.110655
Received 4 March 2024; Received in revised form 11 July 2024; Accepted 13 July 2024
Available online 20 July 2024

(MTD) for heightened efficacy [10]. For instance, Mengmeng et al. [12] combined optimal network shuffling with honeypots in a smart hospital context, while Yifan et al. [13] introduced a bi-level MTD mechanism to counter false data injection (FDI) attacks in smart grids. However, these approaches often require substantial alterations to ICS devices, posing deployment challenges.

Axel et al. [14] suggested leveraging deep reinforcement learning (DRL) for improved defense strategy selection, encompassing deception or MTD tactics. Shuo et al. pioneered an intelligent deployment policy for deception-based cyber defenses, dynamically adjusting deception resource locations based on network security status, and employing model-free Q-Learning algorithms for optimal strategy identification [15]. Innovations like the DQ-MOTAG system by Xinzhong et al. have leveraged DRL to optimize security in Cyber-Physical Systems (CPS), dynamically adjusting shuffle periods and minimizing resource consumption for enhanced protection [16]. Despite the enhanced effectiveness of cyber defense, implementing ML-based methods in ICS environments presents unique challenges, including resource intensiveness, computational complexity, and the lack of emphasis on prioritizing availability.

Building upon insights from hybrid defense strategies, this study aims to develop a tailored defense framework for ICSs, specifically targeting the mitigation of malicious reconnaissance attacks, which signify the early stage of Advanced Persistent Threat (APT) attacks. This framework utilizes the CGAN model for defensive network topology generation while placing a strong emphasis on system availability. Additionally, leveraging software-defined networking (SDN) minimizes the need for modifications to traditional OT devices. The core concept revolves around strengthening system availability through diverse network topologies, dissimilar redundancy, and real-time updates. Furthermore, this research introduces innovative CGAN-based network shuffling techniques and heterogeneous subnets, diverging from conventional approaches. Subsequent evaluation of the proposed defense framework is conducted through experiments on an SDN-based platform.

1.1. Main contributions

This article addresses the aforementioned challenges by presenting an ML-based solution. The primary contributions are outlined below:

- Introduction of an innovative ML-based cyber deception defensive framework prioritizing the preservation of ICS system availability during cyber attacks.
- Deployment of the CGAN-based model to dynamically generate defensive network topologies, offering robust defensive measures tailored for ICS environments.
- Comprehensive simulation studies were conducted to rigorously evaluate the performance of the proposed defense framework, demonstrating significant enhancements in defense efficacy and notable reductions in runtime.
- Discussion of the advantages and limitations of the proposed framework for ICS networks.

1.2. Outline

The subsequent sections of the paper follow this structure: Section 2 provides an in-depth exploration of related works, surveying existing cyber defense strategies utilized in ICS systems and offering background insights into ML-based defense strategies. Section 3 meticulously introduces our proposed defensive framework, showcasing the application of the CGAN model. Further elaboration on the simulation setup and performance metrics is presented in Section 4. Section 5 offers a comprehensive discussion of the framework's performance, emphasizing the utilization of a trained CGAN model for optimal defensive network generation with decoys in a test network. Finally, Section 6 presents the conclusion, summarizing the study's findings and suggesting avenues for future research.

2. Related work

The subsequent section provides a comprehensive overview of cyber defense strategies within ICS and the landscape of ML-based cybersecurity strategies prevalent in current research. It delves into the existing approaches aimed at fortifying ICS networks against evolving threats and investigates the advancements in ML techniques that have facilitated the development of innovative cyber defense measures.

2.1. Cyber defense strategies in ICS

The integration of IT and OT systems within the framework of Industry 4.0 has exposed traditional ICSs to a heightened risk of cyber threats [1]. Unfortunately, ICS devices often lack timely updates and are not inherently designed with cybersecurity as a primary concern, leaving them vulnerable to emerging vulnerabilities [17]. Traditional security solutions face significant challenges in addressing zero-day attacks and previously unknown threats [18]. Moreover, critical components of ICS networks, such as Intelligent Electronic Devices (IEDs) and Remote Terminal Units (RTUs), are susceptible to false data injection attacks, posing a substantial risk of altering control logic and causing damage to the ICS infrastructure [19]. While past research has predominantly focused on single-layer security approaches, integrating multiple defensive techniques, some spanning various domains like smart grids and water treatment systems, has proven challenging [20–23]. Despite concerted efforts, unresolved issues persist in current cyber defense strategies for ICSs, with a notable scarcity of research on dynamic deception-based strategies prioritizing system availability without necessitating extensive device modifications [14]. This research gap underscores the necessity for our proposed development of an SDN-based defense framework, which emphasizes availability as a foundational aspect of cyber defense in ICS environments.

2.2. Machine learning-based cyber defense strategies

The rapidly advancing field of ML has spurred the development of innovative strategies to fortify network security through intrusion detection, attack analysis, and cyber defense. Notable contributions exemplify the power of ML in addressing various challenges. Cheolhee et al. introduced an AI-based Network Intrusion Detection System (NIDS) that effectively handles data imbalances using Generative Adversarial Networks and deep learning (DL) techniques, significantly enhancing network threat detection [24]. Meanwhile, researchers like Meejoung Kim have harnessed deep learning and Conditional GANs to bolster attack analysis in network traffic, even in situations of data scarcity, resulting in substantial advancements in network security [25]. Shuo et al. pioneered an intelligent deployment policy for deception-based cyber defenses, dynamically adjusting deception resource locations based on network security status, and employing model-free Q-Learning algorithms for optimal strategy identification, achieving an impressive defense success rate of nearly 80% that outperforms existing methods [15]. Innovations like the DQ-MOTAG system by Xinzhong et al. have leveraged deep reinforcement learning to optimize security in Cyber-Physical Systems, dynamically adjusting shuffle periods and minimizing resource consumption for enhanced protection [16].

Furthermore, Axel et al. devised a model that integrates deception and MTD strategies with a Deep Q-Learning algorithm, enabling the simulation of asymmetrical attacker–defender confrontations in computer systems and the selection of the most suitable defensive responses based on observed attacker actions [14]. Despite these remarkable strides, applying ML-based methods to ICS environments poses distinctive challenges. While GANs and deep reinforcement learning show promise for enhancing cybersecurity, their computational complexity and resource demands may impede their suitability for ICS settings. Additionally, the effectiveness of deception-based and

Table 1
Comparison of state-of-the-art.

Ref.	Application	Methodology	Attacks
[12]	IoT	Deception and MTD	Reconnaissance
[13]	Smart grid	MTD	FDI
[14]	Computer system	DRL and MTD	Simplified APT
[15]	Web	Deception and RL	Penetration attack
[16]	CPS	DRL and MTD	DDoS
[25]	Attack analysis	DL and CGAN	DDoS
Ours	ICS	ML and MTD	Reconnaissance

MTD strategies relies on assuming consistent network security status, which may not hold in the complex reality of ICS systems. Despite recent advancements suggesting potential for ML-based cybersecurity in ICS, a notable research gap persists. Addressing this gap involves developing lightweight ML algorithms tailored for ICS, establishing robust methodologies for real-time network security assessment, and adapting attacker–defender models to meet the specific operational and security needs of ICS environments. The comparison of the previous studies is listed in Table 1

In comparison, this research targets ICS and leverages ML and MTD to defend against reconnaissance attacks. The novelty lies in the use of CGANs, which enable the generation of more controlled outputs by incorporating additional information, such as network status, into the generative process. CGAN model offers significant potential for bolstering cybersecurity by generating diverse and realistic samples through adversarial training with conditional data and random noise [26]. While CGANs have excelled in tasks like image synthesis and manipulation, their application in cybersecurity, particularly for intrusion or attack detection, is relatively unexplored but promising. There is ample opportunity to further utilize CGANs in cybersecurity, especially for defense. Leveraging the capabilities of CGANs to generate various network topologies for deception defense could introduce a new dimension to ICS networks' security, enhancing overall defense by complicating the attack surface. The potential of CGANs suggests a promising future for ML-based ICS defense strategies.

By incorporating network status as a condition, this approach produces network topologies specifically tailored to the current security needs of the ICS environment. This targeted generation is crucial for effectively countering reconnaissance attacks by continuously altering the attack surface. Additionally, the proposed method introduces a dynamic, real-time defense mechanism through MTD. While previous works have focused on static or less adaptive approaches, the proposed framework leverages the strengths of both CGAN and MTD to create a constantly shifting network landscape. This makes it significantly harder for attackers to gain a foothold, as the network topology they initially encounter can change before they can exploit it.

Previous research has often overlooked the unique requirements of ICS security, particularly the need to prioritize system availability. Our approach addresses this by ensuring the generated network topologies maintain critical system functions while enhancing security. This balance between availability and security aligns with the practical demands of ICS environments. By incorporating these advancements, our approach builds on and significantly enhances current cyber deception strategies, providing a more resilient and efficient defense against sophisticated cyber threats in ICS networks.

3. The proposed CGAN-based cyber deception framework

This section details the design and implementation principles of our proposed CGAN-based cyber deception framework.

3.1. Design overview

Following the defense-in-depth (DiD) strategy, our approach introduces an ML-based cyber deception framework to enhance defensive efficacy within ICS networks. Fig. 1 illustrates the proposed framework operating within an SDN-based ICS network, incorporating key components such as the Network Monitoring System (NMS), Trained CGAN model, network topology deployment module, sub-networks, and critical nodes.

The primary focus of the study is on leveraging an ML-based model to generate defensive network topologies for each sub-network within the ICS network based on real-time network status monitored by the NMS. These sub-networks share identical functions with distinct network topologies and are connected to critical nodes (Servers, PLCs, IEDs, etc.), significantly contributing to ICS network system availability.

By employing a CGAN-based model, defensive network topologies are proactively generated at regular intervals for the sub-networks within the proposed framework. Initially, a dataset of network topologies must be constructed using a combination of deception resources, network shuffling algorithms, and redundant sub-networks. Following the training phase, the Trained CGAN model can generate deceptive defense sub-networks.

The network topology deployment module is crucial in orchestrating the regular updates of sub-networks network topologies. These updates are based on real-time network status, such as the number of compromised nodes within a sub-network monitored by the NMS. Additionally, the updated network topologies integrate industrial decoys sourced from the honeynet server and hardware honeynet.

Subsequently, these interconnected sub-networks are linked via a switch. Upon detecting an intrusion, abnormal sub-networks are swiftly isolated from critical nodes. Meanwhile, normal sub-networks and critical nodes collaborate to ensure the continuous operation of ICS systems. This proactive approach, coupled with redundant sub-networks and updated defensive network topologies, provides an added layer of protection for system availability.

3.2. Attack model

The cyber kill chain (CKC) serves as a widely adopted framework in cybersecurity, offering a structured representation of the stages encompassing various attack processes. This model, consisting of seven distinct phases, commences with reconnaissance efforts to understand the target's attack surface and culminates in achieving the attacker's intended objectives. Within the context of ICS, APTs are notably regarded as particularly severe [27]. This study focuses on defending against the initial stages of APT attacks, specifically targeting reconnaissance activities. To achieve this, we establish certain assumptions about the attacker's behavior and objectives:

- Our assumption revolves around malicious actors possessing limited insight into the network's details. Nevertheless, they possess the capability to conduct network scans employing tools like Nmap or Metasploit to identify and exploit potential vulnerabilities [28].
- Once an attacker interacts with a decoy, their actions become observable. Once they ascertain the presence of a decoy, communication ceases abruptly, prompting the search for an alternative system to compromise. The interaction between the attacker and decoys would be evaluated using the metrics detailed in Sections 4.1 and 4.2.
- The attacker's pattern involves exploiting unpatched or undisclosed vulnerabilities, utilizing compromised real nodes as gateways for lateral movement within the network, thereby jeopardizing system integrity.

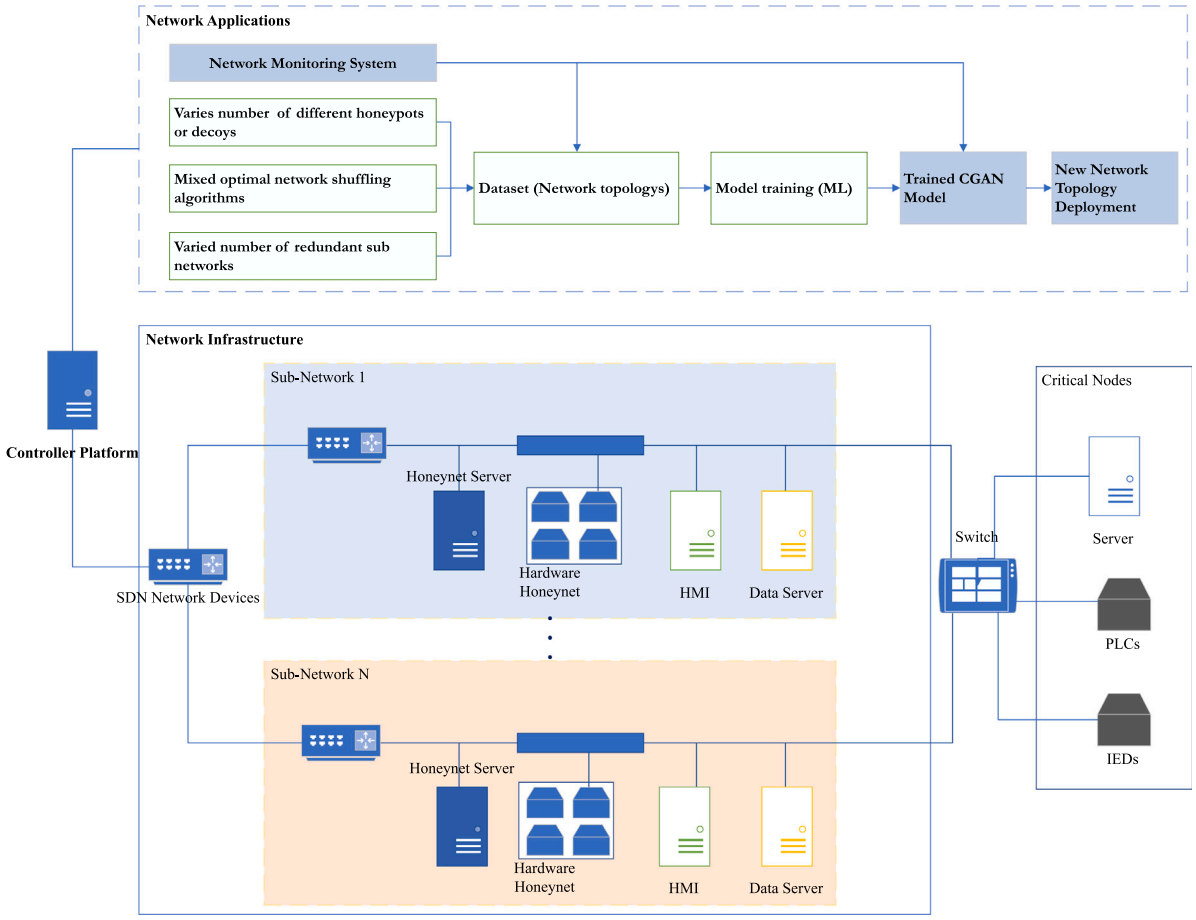


Fig. 1. The proposed CGAN-based cyber deception framework for ICS.

- Subsequently, having established a foothold, the attacker's primary objective is to compromise critical components of the ICS system, such as robots, PLCs, and Central Monitors. This strategic move compromises the system's availability by targeting critical elements.

3.3. CGAN-based network topology generation

CGAN model is employed to generate the defensive network topology in the proposed framework, and Fig. 2 shows the structure of the CGAN model used in this study. We will explain the network generation process in detail in the following subsections.

3.3.1. CGAN-based image generation

The utilization of CGANs amalgamates the foundational principles of GANs with the inclusion of conditional inputs, facilitating data generation with specific constraints or prescribed criteria. In the realm of research, CGANs find application in diverse tasks, encompassing image-to-image translation, super-resolution, data augmentation, and text-to-image synthesis [24,25]. The ability to condition both the generator and discriminator on supplementary input information imparts a remarkable level of control over the characteristics of the generated data, endowing CGANs with the capability to generate synthetic data imbued with particular attributes. In essence, CGANs have significantly propelled the field of generative modeling, casting their influence across domains such as artificial intelligence, computer vision, and data generation endeavors.

Within the proposed defensive framework, the incorporation of the CGAN model, depicted in Fig. 2, facilitates the generation of images (Y) that embody the revised network topology, predicated on insights obtained from the NMS, effectively serving as the conditional input (Y_{fake}). By integrating the conditional input from the NMS alongside a latent vector (Z), the CGAN generator produces fresh network configurations (X_{fake}). This synthesized configuration is engineered to align optimally with the prevailing network conditions. The generated images subsequently undergo evaluation by the discriminator, which has been trained on labeled authentic data (X_{real}). This iterative learning process equips the CGAN model with the ability to create output images (Y) that exhibit akin patterns to those observed in the labeled images (X_{real}) contained within the dataset. Consequently, this adaptive mechanism empowers the dynamic updating of the network's architecture in real time, elevating the overall network performance and enhancing its adaptability to ever-changing conditions.

3.3.2. Dataset for CGAN training

Building a dataset for a CGAN model involves collecting and organizing data to facilitate conditional data generation. The dataset must consist of paired samples, each comprising an input data sample Y representing the conditional information and a corresponding output data sample X representing the target generation based on the condition Y.

Formally, the dataset D for the CGAN model can be represented as Eq. (1):

$$D = \{(X_1, Y_1), (X_2, Y_2), \ldots, (X_N, Y_N)\} \tag{1}$$

where: X_i denotes the i_{th} output data sample corresponding to the input Y_i. Y_i denotes the i_{th} input data sample in the dataset. N is the dataset's total number of paired samples.

The input data samples Y represent the conditional information and can be in various formats, such as images, text, numerical features, or any other relevant representation, depending on the application. For

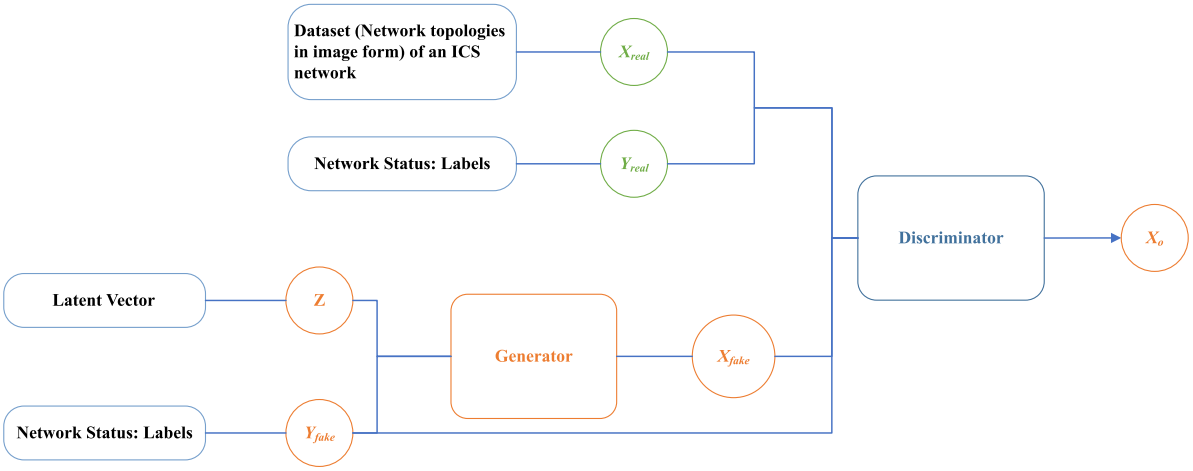


Fig. 2. The CGAN model used to generate network topologies.

example, in image-to-image translation tasks, Y could be a set of images from one domain, and X could be the corresponding images in another field.

The output data sample X represents the target generation that the CGAN generator aims to produce based on condition Y. These output samples should align with the given conditional input's ground truth or desired output.

To build the dataset for this study, the number of compromised real nodes within each sub-network serves as the condition Y. This condition is associated with the ongoing network topology of each sub-network, including optimal network shuffling and deception strategies, represented as a 2D array. This array captures the interconnections and layouts of the sub-networks, which are then converted into images compatible with the CGAN model. The conversion process involves mapping the elements of the 2D array to pixel values, resulting in visual representations of the network topology.

To ensure dataset diversity, various software and hardware honeypots, and decoys are deployed within the sub-networks. These honeypots and decoys include artificial vulnerabilities, leading to varying numbers of compromised nodes in each sub-network, reflecting different attack scenarios and conditions.

Additionally, network shuffling is implemented using various optimal algorithms to enhance the dataset's robustness. This process entails periodically reconfiguring network connections and altering node relationships while preserving the overall sub-network structure. By employing optimal network shuffling, dynamic variations in the network topology are introduced, resulting in a more comprehensive and diverse dataset for training the CGAN model.

The resulting dataset, comprising the conditional inputs Y (number of compromised nodes) and corresponding network topology images X (2D array representations converted to images), is then used to train the CGAN model. The model learns to generate network topologies based on the condition Y of compromised nodes, thereby contributing to advancing the network security of ICS networks.

3.3.3. Network topology and image conversion

The process of converting selected network topologies into images to create the dataset involves encoding the network topology of each sub-network into a 2D matrix. Within each sub-network, multiple nodes are categorized as either real nodes (r_i) or decoy nodes (d_j), and their interconnections are captured in the 2D matrix. This matrix contains the information necessary for building the dataset used to train the CGAN model.

There are several methods for converting a 2D matrix to a grayscale image. One common approach is linear mapping, which normalizes matrix values to a range between 0 and 1 and uses these normalized values as grayscale intensities. Another method is logarithmic mapping, which compresses the dynamic range of matrix values, enhancing image details but at the cost of increased complexity and potential distortion of relative differences in values.

In this study, we adopt the linear mapping method for converting the 2D matrix to a grayscale image. This method involves normalizing the matrix values to a range between 0 and 1, and then proportionally amplifying these normalized values to enhance detail visibility. The advantage of this approach lies in its simplicity and its ability to preserve relative differences in matrix values, making it suitable for the dataset creation and CGAN model compatibility.

We assume that each sub-network has a m number of real nodes and n number of decoy nodes. The interconnections of these real and decoy nodes can be listed in a matrix, which can be represented as follows using Eq. (2).

$$NT_i = \begin{bmatrix} r_2 & \cdots & r_m & d_1 & \cdots & d_n \\ r_1 & \cdots & r_m & d_1 & \cdots & d_n \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ r_1 & \cdots & r_{m-1} & d_1 & \cdots & d_n \\ d_2 & \cdots & d_n & r_1 & \cdots & r_m \\ d_1 & \cdots & d_n & r_1 & \cdots & r_m \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ d_1 & \cdots & d_{n-1} & r_1 & \cdots & r_m \end{bmatrix} \tag{2}$$

In Eq. (2), the matrix NT_i represents the network topology, where r_i represents the i_{th} real node with $i \in (1, m)$, and d_j represents the j_{th} decoy node with $j \in (1, n)$. The rows and columns in the matrix correspond to real nodes and decoy nodes. The i_{th} rows with $i \in (1, m)$ represent the connections between the i_{th} real node and other nodes within the sub-network. The j_{th} rows with $j \in (m+1, m+n)$ represent the connections of the j_{th} decoy node with other nodes. Thus, the row and column indices of elements in this matrix are associated with the numbers of real and decoy nodes.

Subsequently, the linear mapping method converts the matrix into a grayscale image. The topology information encoded in grayscale images serves as the training data for the CGAN model, facilitating the generation of adaptive network topologies based on the conditions of compromised real nodes.

4. Evaluation

SDNs offer a solution to the limitations of traditional IP networks through innovative architectures [29]. Leveraging SDNs' flexibility and cost-effectiveness, we assess our CGAN-based cyber deception framework using an open-source SDN project as a proof of concept. This controlled testbed allows us to simulate and analyze the framework's performance in an industrial manufacturing scenario.

A test network was constructed in an industrial scenario to evaluate the effectiveness and performance of the proposed framework, as depicted in Fig. 3. Within each sub-network of the test network, our cyber deception framework was implemented, incorporating diverse optimal network shuffling algorithms. To prepare data for training the CGAN model, we continuously monitored the compromised count (Y) of real nodes and recorded the corresponding network topology in 2D arrays as output data (X). Subsequently, thorough cleaning and preprocessing of the collected data resulted in a dataset comprising 28 labeled conditions, each with 200 samples, obtained through 2D array-to-image conversion, yielding a total of 5600 images. The CGAN model was then trained using this dataset. After successful model training, a comprehensive evaluation of the cyber deception framework was conducted on an open-source SDN testbed. This allowed for the simulation of real-world cyber-attack scenarios and in-depth analysis of the framework's capability to deceive potential attackers during reconnaissance.

4.1. Simulation setup

Simulations were executed on a ThinkStation equipped with an Intel(R) Xeon(R) W-2133 CPU running at 3.60 GHz, along with 32 GB of RAM and an NVIDIA GeForce RTX2080 SUPER. The operating system utilized was Windows 10 Enterprise, and PyCharm CE 2022 served as the integrated development environment (IDE), leveraging Python 3.8.0 and Keras 2.12.0, TensorFlow-gpu 2.5.0, CUDA 11.8, cuDNN 8.9.0, and NumPy 1.19.2 for the simulation implementations. These hardware and software specifications provided a robust and capable environment for conducting the simulations and effectively evaluating the proposed defense framework's performance.

To validate the effectiveness of the proposed defense framework, a simulation testbed was established to emulate the typical network structure of an ICS, as depicted in Fig. 3, utilizing an open-source SDN-based project [12]. Through diverse simulation scenarios, the security performance of the defense framework was comprehensively evaluated, with a specific focus on an industrial scenario within the ICS domain. The real nodes in the two target networks included cameras, laptops, sensors, meters, master terminal units (MTUs), PLCs, and one server. Notably, the server was identified as a critical component significantly influencing the system's availability and overall performance. This simulation-based approach enabled a rigorous assessment of the proposed defense framework's capabilities in safeguarding ICS networks, providing valuable insights into its effectiveness and potential impact on critical industrial infrastructure.

To construct the test network, pertinent information on vulnerabilities associated with real nodes and decoys was gathered from the Common Vulnerabilities and Exposures (CVE)/NVD database [30]. Specifically, vulnerabilities selected for this research pertained to real devices and had been publicly disclosed. In the test network, each real node was assumed to contain one vulnerability that an attacker could exploit to access the system. The vulnerability information for the selected real nodes and the corresponding exploitability measure based on the Common Vulnerability Scoring System (CVSS) base score is presented in Table 2. This information served as a basis for estimating the performance of the defense measure in the SDN-based ICS test network, as elaborated in previous studies [12,31].

In the initial deployment of the cyber deception framework, four decoys were included in each sub-network, with each decoy configured to have multiple vulnerabilities that an attacker could exploit to gain control over the node. Emulated decoys were utilized for the cameras, Personal Computers (PCs), MTUs, and servers. The vulnerability information for these decoys is provided in Table 3. By incorporating real nodes and decoys with relevant vulnerability details, we created a realistic and comprehensive test network, facilitating a thorough evaluation of the proposed cyber deception framework's effectiveness against potential attacks.

Table 2
Real nodes and vulnerabilities.

Real node	Network	CVE ID	CVSS
MTU	1	CVE-2023-23609	7.4
	2	CVE-2020-3527	8.6
Camera	1	CVE-2023-33244	8.2
	2	CVE-2023-31996	8.8
Sensor	1	CVE-2023-35830	9.8
	2	CVE-2022-39124	5.5
Laptop	1	CVE-2023-3633	7.5
	2	CVE-2023-35326	5.5
PLC	1	CVE-2022-27480	7.5
	2	CVE-2021-34578	8.1
PC	1	CVE-2023-32734	7.8
	2	CVE-2023-32397	7.5
Meter	1	CVE-2023-31238	4.8
	2	CVE-2018-12921	7.5
Server	1	CVE-2023-36884	8.8
	2	CVE-2023-36631	7.8

Table 3
Decoy nodes and vulnerabilities.

Decoy node	Network	CVE ID	CVSS
MTU	1	CVE-2022-30264	9.8
		CVE-2019-10936	7.5
	2	CVE-2019-10923	7.5
		CVE-2017-6041	9.8
Camera	1	CVE-2023-30356	7.5
		CVE-2023-30353	9.8
	2	CVE-2023-26609	7.2
		CVE-2023-24506	7.5
Sensor	1	CVE-2022-24389	8.8
		CVE-2022-0486	7.8
	2	CVE-2021-38548	5.9
		CVE-2021-38546	5.9
Laptop	1	CVE-2020-4631	5.5
		CVE-2020-4494	7.5
	2	CVE-2023-38565	7.8
		CVE-2023-38410	7.8
PLC	1	CVE-2020-16850	7.5
		CVE-2021-27477	7.5
	2	CVE-2021-22791	6.5
		CVE-2020-28211	7.8
PC	1	CVE-2023-36458	8.8
		CVE-2022-23222	7.8
	2	CVE-2022-23119	7.5
		CVE-2022-23511	6.8
Meter	1	CVE-2016-5782	8.6
		CVE-2020-246252	7.5
	2	CVE-2021-22713	7.5
		CVE-2017-5166	9.8
Server	1	CVE-2023-35365	9.8
		CVE-2023-35308	6.5
	2	CVE-2023-35012	6.7
		CVE-2023-22883	7.8

During the simulation, the reconnaissance attack was emulated using the HARM model within the testbed, enabling the exploitation of vulnerabilities to compute potential attack paths [12]. The attacker could randomly select an entry point from one of the attack paths and proceed to compromise nodes along that path until they successfully accessed the system's availability. This approach allowed for the realistic simulation and evaluation of the impact of reconnaissance attacks and the effectiveness of our proposed cyber deception framework in mitigating such threats.

4.2. Dataset

Creating a dataset is crucial for the simulation, comprising images that capture network topology information under distinct conditions. As detailed in Section 3.3.2, network topologies are initially encoded as 2D matrices, documenting node interconnections within sub-networks.

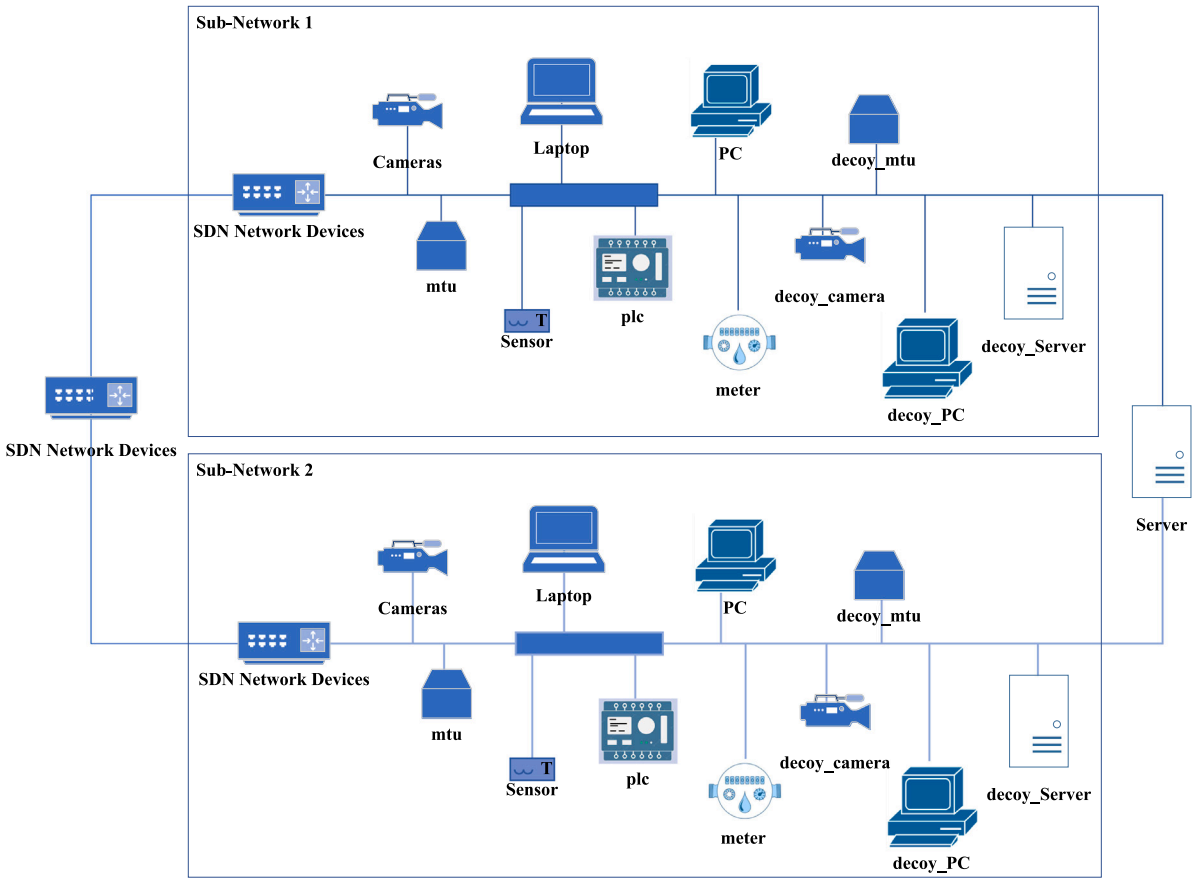


Fig. 3. SDN-based test network with two subnetworks.

These matrices are then transformed into images, making them compatible with the CGAN model, by mapping matrix elements to corresponding pixel values. This transformation entails mapping the elements of the matrix to corresponding pixel values, ultimately producing a visual representation of the network topology in images.

The dataset was created using the same open-source SDN-based testbed with the same test network but combines optimal algorithms and cyber deception techniques against malicious reconnaissance attacks. Besides the Non-dominated Sorting Genetic Algorithm II (NSGA-II) used in [12], we further extend the simulation using particle swarm algorithms (PSO), Estimation of Distribution Algorithm (EDA), Differential Evolution (DE) algorithm, and Evolutionary Computation (EC) algorithms. These optimization algorithms aim to solve complex optimization problems efficiently. They leverage population-based strategies, probabilistic modeling, and evolutionary principles to explore the search space effectively and find optimal solutions. Their advantages lie in their ability to handle multi-objective problems, mimic natural processes for solution generation, and achieve convergence to optimal solutions in various domains. Leveraging these algorithms, we aim to generate the network topologies that maximize the time attacker spend while minimizing their success probability, which is a multi-objective optimization problem (MOO) with two objectives. These two objectives are considered two performance metrics for the proposed defensive approach including Mean Time To Security Failure ($MTTSF$) and Attack Success Probability (ASP).

The MTTSF metric assesses the average duration attackers spend compromising all target networks, indicating system availability failure condition (SAFC) when all sub-networks' system integrity is compromised. It is calculated using Eq. (3), with p rounds of scanning and $MTTC_k$ representing the maximum Mean Time To Compromise (MTTC) in each sub-network [31]. K denotes the total number of sub-networks used in the defense framework. The k refers to the k_{th} sub-network, and K refers to the total number of sub-networks used in the defense framework.

$$MTTSF = \frac{\sum_{i=1}^{p} max(MTTC_k)}{p}, k \in K \tag{3}$$

The MTTC, detailed in Eq. (4), measures the time taken for an attacker to compromise a specified percentage of real nodes, signifying system integrity failure when one-third of the real nodes are compromised [31]. The calculation involves p rounds of scanning, with m_{SIFC} representing the compromised nodes when the system reaches the system integrity failure condition (SIFC) when one-third of the real nodes are compromised. And T_{mvet} denoting the mean vulnerability exploitation time (from the CVSS) and the software vulnerabilities are selected from the NVD. M is a finite set of nodes in each sub-network, and j is one of the nodes in M.

$$MTTC = \frac{\sum_{i=1}^{p} \sum_{j=1}^{m_{SIFC}(i)} T_{mvet}(j)}{p}, j \in M \tag{4}$$

The Attack Success Probability (ASP) is a performance metric that gauges the effectiveness of defense strategies, as defined in Eq. (5). It represents the ratio of successful attacks compromising real nodes during q simulation rounds. In each simulation (i_{th}), r_c denotes the compromised real nodes, while N_r is the total number of real nodes.

$$ASP = mean \sum_{i}^{q} \frac{r_i^c}{N_r} \tag{5}$$

For optimal network shuffling in the sub-networks, we aim to maximize the MTTSF while minimizing the ASP. One of the most common methods for solving MOO problems is scalarizing it to a single objective optimization (SOO) problem. So, We first normalize the metrics

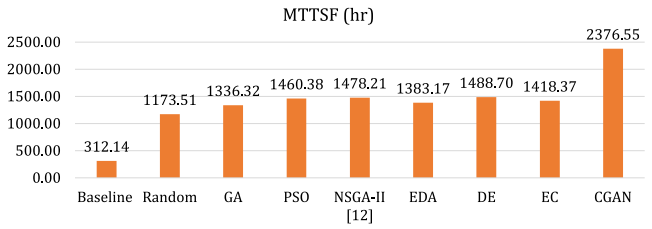


Fig. 4. Mean time to security failure.

and provide a weight to each metric based on the scalarization-based MOO technique to convert the MOO problem into an SOO problem to select one optimum solution among the Pareto optimal solutions. The objective function we used in this work is listed in Eq. (6).

$$max\ \omega_M \widehat{MTTSF} + \omega_A(1 - \widehat{ASP}), \tag{6}$$

where ω_M and ω_A are weights to the above metrics with $\omega_M = \omega_A = \frac{1}{2}$. $\widehat{MTTSF}, \widehat{ASP}$ are the normalized metrics of mean time to security failure and attack success probability. The optimal solution is the network topology with the maximum objective value.

Before the test network is compromised, a range of conditions n_c manifests via compromising different n_r real nodes out of the total N_r real nodes. The number of these conditions can be calculated using Eq. (7).

$$\binom{N_r}{n_r} = \frac{N_r!}{n_r!(N_r - n_r)!} \tag{7}$$

In the context of the depicted test network as showcased in Fig. 3, characterized by eight real nodes, compromising one-third of these nodes – precisely two out of eight – significantly affects system integrity, which leads to 28 combinations. This array constitutes 28 conditions for the CGAN-based deception framework. For each condition, a meticulous selection of 200 samples occurred based on real-time calculated metrics (ASP). Subsequently, a dataset encompassing 5600 images was meticulously curated for the CGAN model's training.

4.3. Performance analysis

After the creation of the dataset (D), the training of a conventional CGAN model (M_{CGAN}) was initiated, spanning 500 epochs. The trained CGAN model is capable of generating images in real time using an input label. This foundational training served as the cornerstone for subsequent phases of the study.

With the trained CGAN model, the next step entailed its integration into the SDN-based testbed. This integration facilitated the continuous monitoring of the current combinations of compromised real nodes within each sub-network, serving as the label (L_i) for the trained CGAN model. When invoked, this model generated an image (I_i) encapsulating the reconfigured network topology. Subsequently, the generated image underwent decoding and was converted into a 2D array (A_i). This 2D array became the input for a network topology generation function within the SDN testbed. Notably, the network topology within the two sub-networks of the proposed framework was constantly updated using the CGAN model. The culmination of these orchestrated steps precipitated the evaluation of the proposed framework's performance, meticulously gauged through the ASP metric. This evaluative phase enabled a comprehensive understanding of the framework's efficacy in countering reconnaissance attacks, with ASP serving as a crucial barometer of its defensive prowess. A detailed illustration of the workflow of the proposed framework can be found in Algorithm 1.

Algorithm 1: CGAN deception defense workflow

Input : Dataset of network topology images D, Trained CGAN model M_{CGAN}
Output: Updated network topology in sub-networks
Train M_{CGAN} with D for 500 epochs;
foreach *Simulation round i* **do**
  Monitor compromised real nodes in each sub-network and obtain labels L_i;
  Generate updated topology image I_i using M_{CGAN} with L_i;
  Decode I_i to 2D Matrix A_i;
  Update network topology using A_i in SDN testbed;
end
Algorithm Termination: SAFC is reached ;

4.3.1. MTTSF

The mean time to security failure represents the duration attackers spent compromising the test network until the defense framework reached a system integrity failure condition, with higher values indicating better performance. In the simulation, 50 rounds were executed for each optimization algorithm alongside the CGAN model. First, utilizing the $MTTSF$ as the performance metric within the test network, as depicted in Fig. 3. Termination of each round was predicated on all sub-networks within the proposed framework reaching the SIFC, indicative of system availability failure condition. The outcome of this evaluation is visually presented in Fig. 4.

Initially, the framework was evaluated without defense strategies to establish a baseline (312.14 h). Subsequently, random network shuffling with decoys was employed, resulting in a significant improvement (1173.51 h) compared to the baseline. Upon deploying optimal network shuffling techniques, NSGA-II and DE exhibited superior performance, recording MTTSF values of 1478.21 and 1488.70 h, respectively, representing increments of 125.9% and 126.8% compared to random shuffling. The CGAN model emerged as the most promising contender, boasting the highest MTTSF at 2376.35 h, which is 159.6% higher than DE, the highest among all optimal algorithms tested, signifying a significant stride towards enhancing the cost of time attackers spent during reconnaissance attacks.

4.3.2. Attack success probability

Following the same workflow, the ASP is also measured, quantifying the probability of successful attacks compromising real nodes in a system over multiple simulation rounds. It provides a measure of the effectiveness of defense strategies. And the outcome is visually presented in Fig. 5.

The evaluation of the framework began with a baseline assessment without any defense strategies. Subsequently, random network shuffling with decoys was implemented, leading to a substantial improvement (48.86% reduction) in the ASP compared to the baseline. Further enhancement was observed with the application of optimal network shuffling techniques, where NSGA-II and DE demonstrated superior performance with ASP values of 43.88% and 42.78%, respectively, representing reductions of 8.26% and 8.36% compared to random shuffling. Notably, the CGAN model exhibited the most promising results, achieving the lowest attack success probability at 32.99%, which is 9.79% lower than DE, the lowest among all optimal algorithms tested. This significant enhancement validates the efficacy of the CGAN model, indicating a substantial advancement in security and resilience.

4.3.3. Time complexity

Time complexity is a pivotal metric for assessing algorithms and models by quantifying their computational efficiency [32]. We gain insights into its scaling efficiency by analyzing an algorithm's operational growth concerning input size. This analysis aids in selecting the best approach for a given problem, considering real-world

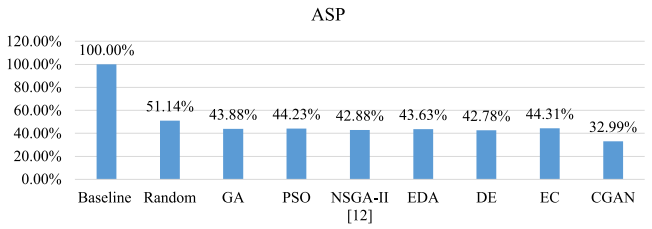


Fig. 5. Attack success probability.

Table 4
Time complexity of optimization algorithms and CGAN.

Algorithm	Time complexity
GA	$O(N \cdot (GenInd + 1) + C \cdot L + M \cdot L + G)$
PSO	$O(N \cdot (Dim + 1) + G)$
NSGA-II	$O(N \cdot (GenInd + G) + N^2 + C \cdot L + M \cdot L)$
EDA	$O(N \cdot (Dim + G \cdot (Dim + 1)))$
DE	$O(N \cdot (Dim + 1) + G \cdot (2 \cdot Dim + 1))$
EC	$O(N \cdot (Dim + 1) + G \cdot (2 \cdot Dim + 1))$
CGAN	$O(Z + CL + FC + NA)$

scenarios and available resources. Time complexity informs resource allocation, guides algorithmic decisions, and influences solution deployment choices. Moreover, it propels algorithmic progress, enhancing methods and efficiency innovations. Time complexity offers a standardized metric to assess and contrast various computational methods, ensuring informed choices that optimize efficiency, resource utilization, and problem-solving capabilities. To further evaluate the proposed approach, we compare the time complexity of the optimal algorithm mentioned with the CGAN model.

The time complexity of algorithms is often denoted using big O notation, which disregards coefficients and lower-order terms. This notation categorizes algorithms based on their responsiveness to input size variations. Determining when and how to employ a specific algorithm enhances our comprehension of its performance characteristics. Evaluating the time complexity of optimization algorithms holds a crucial significance in measuring their computational efficiency.

We first estimate the time complexity of a GA and the various components involved, such as selection, crossover, mutation, and fitness evaluation. The time complexity of a GA can be analyzed based on the following key parts:

Initialization: The population of individuals is created at the beginning of the algorithm. The time complexity of this step depends on the population size N and the complexity of generating an individual, denoted as $O(GenInd)$.

Selection: Selection involves choosing individuals for reproduction based on their fitness. This step typically takes $O(N)$ time as each individual's fitness needs to be evaluated.

Crossover: Crossover involves combining genetic material from selected individuals to create new offspring. The time complexity depends on the crossover rate, which determines how many crossover operations are performed. If C is the number of crossover operations, and L is the average length of the individuals, the time complexity for crossover is $O(C \cdot L)$.

Mutation: Mutation involves changing certain genes in the individuals' genetic material. Like crossover, the time complexity depends on the mutation rate and the length of the individuals. If M is the number of mutation operations, and L is the average length of the individuals, the time complexity for mutation is $O(M \cdot L)$.

Fitness Evaluation: Each individual's fitness needs to be evaluated to guide selection. This step is $O(N)$ as it requires evaluating the fitness function of each individual.

Termination: The algorithm continues until a termination condition is met. The number of iterations or generations is denoted as G.

Considering these components, an approximate time complexity for G generations of the GA can be given as Eq. (8):

$$O(G \cdot N \cdot (\text{GenInd} + 1) + G \cdot (C + M) \cdot L) \tag{8}$$

Similar to the GA analysis, we can analyze the time complexity of PSO, NSGA-II, EDA, DE, and EC based on their key components, as shown in Table 4. Note that G represents the number of generations, N represents the population size, $GenInd$ represents the complexity of generating an individual, C represents the number of crossover operations, M represents the number of mutation operations, L represents the average length of individuals, and Dim represents the dimensionality of the problem.

To calculate the time complexity of the conventional CGAN, we used the following steps:

Input Noise Vector: The generator typically takes an input noise vector of size Z. The time complexity of creating and passing this noise vector to the generator is $O(Z)$.

Convolutional Layers: Convolutional layers are a major component of the generator in a CGAN. For each convolutional layer, you must consider the number of filters, kernel size, and input/output dimensions. The time complexity of a convolution operation depends on these factors and can be expressed as $O(filters \times kernelsize \times inputdimensions \times outputdimensions)$. Summing up the complexities for all convolutional layers gives you the convolutional layers' total time complexity.

Fully Connected Layers: If the generator includes fully connected (dense) layers, each neuron's computation depends on the number of input connections, output connections, and the size of the input vector. The time complexity of a fully connected layer can be approximated as $O(inputconnections \times outputconnections)$. Summing up the complexities for all fully connected layers provides the fully connected layers' total time complexity.

Normalization and Activation: Batch normalization, activation functions (e.g., ReLU), and other operations also contribute to the time complexity, but they are usually relatively minor compared to convolutions and fully connected layers.

We sum up the time complexities for each component to estimate the total time complexity for generating an image using the trained CGAN. Here leads to Eq. (9) that represents the simplified time complexity estimation of a conventional CGAN used to generate an image:

$$O(Z + CL + FC + NA) \tag{9}$$

where Z represents the time complexity of creating and passing the input noise vector. CL represents the combined time complexity of all convolutional layers in the generator. FC represents the combined time complexity of all fully connected layers (if present) in the generator. NA represents the time complexity of batch normalization, activation functions, and other minor operations.

PSO exhibits superior time complexity compared to other optimization algorithms. Its efficiency surpasses GA, NSGAII, EDA, DE, and EC, where complexities involve N and G factors. PSO's linear scaling with population size N and dimensionality Dim, alongside an additive term for iterations, stems from its inherent simplicity and direct particle interaction. Conversely, GA, NSGAII, EDA, DE, and EC entail intricate genetic operations, resulting in higher complexities. Regarding time complexity, PSO is the computationally efficient choice among assessed algorithms. PSO also outperforms random shuffling, with a 6.91% lower ASP in the framework. NSGA-II and DE, despite higher time complexity, exhibit relatively improved ASP performance compared to PSO, each 1.35% and 1.45% lower.

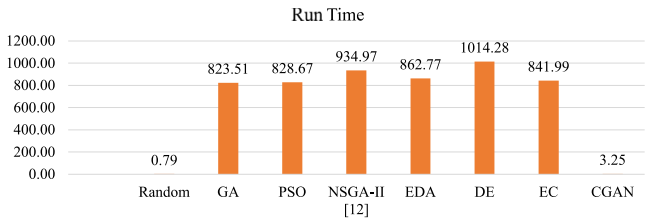


Fig. 6. The runtime of different optimal algorithms and CGAN.

For the comparison of time complexity between CGAN and optimization algorithms, the significant difference in time overhead can be attributed to several factors, particularly the nature of their operations and the size or scalability of the test network. CGANs, during training, are computationally intensive due to their architecture involving two neural networks (generator and discriminator) trained iteratively. This process requires multiple forward and backward passes through the networks, leading to increased computational time. In contrast, optimization algorithms, while also iterative, may involve simpler operations like function evaluations and parameter updates, resulting in lower computational overhead. Additionally, the complexity of the objective functions and search spaces can contribute to the observed differences in time complexity. After training, a trained CGAN can generate new network topologies with relatively low inference time. This is because the generator network in the CGAN has already learned the underlying distribution of the data, allowing it to quickly generate new samples without the need for further training. This characteristic makes CGANs efficient for generating new network topologies on demand.

In a small-sized test network, the trained CGAN exhibits lower runtime due to its relatively small inference time, especially when compared to optimization algorithms that often require more computation to search for optimal solutions in a potentially large search space. Additionally, the iterative nature of optimization algorithms, such as NSGA-II or PSO, may contribute to longer runtime, as they typically involve multiple iterations to converge to a solution, whereas CGAN inference for generating network topologies might be more direct and computationally efficient, particularly for smaller networks.

In this scenario, the trained CGAN model is used in the defensive framework and provides faster results than the optimal algorithms, considering the complexity of the network topology generation. Because the time consumption of the image generation using trained CGAN is lower than optimal algorithms to generate updated network topology. To gain a straightforward comparison of the computational intricacies of these algorithms and the CGAN model, we carried out 50 simulation rounds using the same testbed. This approach enabled us to measure the time required by the trained CGAN and the selected optimization algorithms to generate updated network topologies. Fig. 6 illustrates each round's average run-time (second), showcasing the computational complexity of optimal network shuffling algorithms and the trained CGAN. NSGA-II (934.97 s) and DE (1014.28 s) have higher time complexity and subsequently show considerably higher run-time than the trained CGAN (3.25 s).

The simulation results provide compelling evidence for the effectiveness of the ML-based cyber deception strategy. The trained CGAN model demonstrates considerable performance improvements in $MTTSF$, ASP, and a remarkable reduction in runtime. Compared with the findings from [12], which utilized the same simulation platform, this consistency allows for a direct and fair comparison under similar simulation scenarios, ensuring that the comparison is meaningful and relevant to the specific context of our research. Our approach showed significant improvements in prolonging the time spent by the attacker, reducing ASP, and enhancing operational efficiency. These outcomes, achieved even within a limited-scale test network, validate the feasibility of the proposed approach and highlight its potential to enhance the security and operational efficiency of ICSs significantly. This comparison underscores the robustness and practical applicability of our approach in realistic cyber threat scenarios.

5. Discussion

This section acknowledges and discusses the implications, limitations, and potential avenues for future work regarding the proposed cyber deception framework.

Implication: The proposed CGAN-based cyber deception defense framework aims to fortify the ICS network while minimizing device modifications and computational complexity using AI techniques. Simulation results carry significant implications for enhancing ICS security by improving system availability and reducing deployment runtime for defensive strategies. Strategically lowering the computational complexity of network shuffling against reconnaissance attacks, the framework also decreases the likelihood of attackers compromising system availability. This deliberate delay reduces the probability of successful attacks, bolstering ICS network resilience. Integrating AI into a defense framework establishes an effective line of defense, enhancing overall resilience against evolving threats and complex attack vectors.

Limitations: While implementing the proposed defense strategies, it is essential to assess the impact on individual ICS devices' performance. Tactics like generating deceptive network topologies and executing network shuffling via SDN may disrupt flow patterns, affecting critical devices such as PLCs and microcontrollers. A thorough evaluation of induced overhead within our testbed is crucial to understanding potential disruptions and informing fine-tuning strategies. Additionally, scalability becomes a concern as the framework extends to more complex ICS networks. Integrating AI-driven deception techniques may require additional training for new nodes, potentially hindering scalability. The effectiveness of optimal network shuffling algorithms used for CGAN model training also influences the framework's performance and scalability.

Future Work: Looking forward, the ever-evolving landscape of cyber threats demands continuous advancement in defensive strategies. Addressing scalability challenges in AI-based defensive frameworks opens several avenues for future research. One promising direction involves refining the architecture of CGAN-based deception frameworks to seamlessly adapt to dynamic attack scenarios, leveraging transfer learning to reduce the training burden when incorporating new real nodes. Additionally, integrating real-world experiments and field tests can provide valuable insights into framework performance in authentic ICS environments. Exploring complementary strategies, such as integrating machine learning for attack detection and leveraging distributed learning techniques, holds significant potential in fortifying overall ICS network security and proactively countering emerging threats.

6. Conclusion

This paper presents a novel ML-driven cyber deception framework tailored to enhance ICS defensive capabilities against reconnaissance attacks. Harnessing CGANs, the proposed approach focuses on generating deceptive network topologies to thwart potential attackers early on. Extensive simulations demonstrate our framework's impressive defensive performance, characterized by reduced computational demands and lower attack success probability. By effectively deterring reconnaissance attacks, our approach significantly bolsters ICS network cybersecurity, crucial for safeguarding critical industrial infrastructure against cyber threats. These promising results underscore the practicality and effectiveness of employing an AI-based cyber defensive framework in real-world industrial settings. Future efforts should explore advanced transfer and distributed learning techniques to further enhance adaptability to evolving threat landscapes.

CRediT authorship contribution statement

Xingsheng Qin: Validation, Methodology, Formal analysis, Conceptualization. **Frank Jiang:** Writing – review & editing. **Xingguo Qin:** Writing – review & editing, Data curation. **Lina Ge:** Writing – review & editing. **Meiqu Lu:** Writing – review & editing. **Robin Doss:** Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

The authors would like to express their gratitude to the anonymous reviewers for their valuable contributions and suggestions, which have significantly improved the quality of this paper.

References

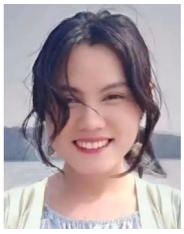
[1] S. Maesschalck, V. Giotsas, B. Green, N. Race, Don't get stung, cover your ICS in honey: How do honeypots fit within industrial control system security, Comput. Secur. 114 (2022) 102598.
[2] D. Kushner, The real story of stuxnet, IEEE Spectr. 50 (3) (2013) 48–53.
[3] Z. Liu, L. Wang, Leveraging network topology optimization to strengthen power grid resilience against cyber-physical attacks, IEEE Trans. Smart Grid 12 (2) (2021) 1552–1564.
[4] S. Lyngaas, Utah renewables company was hit by rare cyberattack in March, 2019, [Online]. Available: https://www.cyberscoop.com/spower-power-grid-cyberattack-foia/.
[5] U.D. Ani, N. Daniel, F. Oladipo, S.E. Adewumi, Securing industrial control system environments: the missing piece, J. Cyber Secur. Technol. 2 (3–4) (2018) 131–163.
[6] W. Hofer, T. Edgar, D. Vrabie, K. Nowak, Model-driven deception for control system environments, in: 2019 IEEE International Symposium on Technologies for Homeland Security (HST), 2020, pp. 1–7.
[7] R. Jhaveri, R. Sagar, G. Srivastava, T.R. Gadekallu, V. Aggarwal, Fault-resilience for bandwidth management in industrial software-defined networks, IEEE Trans. Netw. Sci. Eng. 8 (4) (2021) 3129–3139.
[8] A. Saad, S. Faddel, T. Youssef, O.A. Mohammed, On the implementation of IoT-based digital twin for networked microgrids resiliency against cyber attacks, IEEE Trans. Smart Grid 11 (6) (2020) 5138–5150.
[9] S. Abe, Y. Tanaka, Y. Uchida, S. Horata, Developing deception network system with traceback honeypot in ICS network, SICE J. Control Meas. Syst. Integr. 11 (4) (2021) 372–379.
[10] X. Qin, F. Jiang, M. Cen, R. Doss, Hybrid cyber defense strategies using honey-X: A survey, Comput. Netw. 230 (2023) 109776–109793.
[11] C.-Y.J. Chiang, S. Venkatesan, S. Sugrim, J.A. Youzwak, R. Chadha, E.I. Colbert, H. Cam, M. Albanese, On defensive cyber deception: A case study using SDN, in: MILCOM 2018-2018 IEEE Military Communications Conference, MILCOM, IEEE, 2018, pp. 110–115.
[12] M. Ge, J.-H. Cho, D. Kim, G. Dixit, I.-R. Chen, Proactive defense for internet-of-things: Moving target defense with cyberdeception, ACM Trans. Internet Technol. 22 (1) (2022) 1–31.
[13] Y. Hu, P. Xun, P. Zhu, Y. Xiong, Y. Zhu, W. Shi, C. Hu, Network-based multidimensional moving target defense against false data injection attack in power system, Comput. Secur. 107 (2021) 102283.
[14] A. Charpentier, N.B. Cuppens, F. Cuppens, R. Yaich, Deep reinforcement learning-based defense strategy selection, in: Proceedings of the 17th International Conference on Availability, Reliability and Security, 2022, pp. 1–11.
[15] S. Wang, Q. Pei, J. Wang, G. Tang, Y. Zhang, X. Liu, An intelligent deployment policy for deception resources based on reinforcement learning, IEEE Access 8 (2020) 35792–35804.
[16] X. Chai, Y. Wang, C. Yan, Y. Zhao, W. Chen, X. Wang, DQ-MOTAG: Deep reinforcement learning-based moving target defense against ddos attacks, in: 2020 IEEE Fifth International Conference on Data Science in Cyberspace, DSC, 2020, pp. 375–379.
[17] T. Alladi, V. Chamola, S. Zeadally, Industrial control systems: Cyberattack trends and countermeasures, Comput. Commun. 155 (2020) 1–8.
[18] B. Galloway, G.P. Hancke, Introduction to industrial control networks, IEEE Commun. Surv. Tutor. 15 (2) (2013) 860–880.
[19] N. Tuptuk, S. Hailes, Security of smart manufacturing systems, J. Manuf. Syst. 47 (2018) 93–106.
[20] N. Cifranic, R.A. Hallman, J. Romero-Mariona, B. Souza, T. Calton, G. Coca, Decepti-SCADA: A cyber deception framework for active defense of networked critical infrastructures, Internet Things 12 (2020) 100320.
[21] B. Li, Y. Shi, Q. Kong, C. Zhai, Y. Ouyang, Honeypot-enabled optimal defense strategy selection for smart grids, in: 2021 IEEE Global Communications Conference, GLOBECOM, IEEE, Madrid, Spain, 2021, pp. 1–6.
[22] K. Wang, M. Du, S. Maharjan, Y. Sun, Strategic honeypot game model for distributed denial of service attacks in the smart grid, IEEE Trans. Smart Grid 8 (5) (2017) 2474–2482.
[23] A.F.M. Piedrahita, V. Gaur, J. Giraldo, Á.A. Cárdenas, S.J. Rueda, Leveraging software-defined networking for incident response in industrial control systems, IEEE Softw. 35 (1) (2018) 44–50.
[24] C. Park, J. Lee, Y. Kim, J.-G. Park, H. Kim, D. Hong, An enhanced AI-based network intrusion detection system using generative adversarial networks, IEEE Internet Things J. 10 (3) (2023) 2330–2345.
[25] M. Kim, ML/CGAN: Network attack analysis using CGAN as meta-learning, IEEE Commun. Lett. 25 (2) (2021) 499–502.
[26] J. Huang, D. Hu, Z. Ding, X. Wu, Attack detection and data generation for wireless cyber-physical systems based on self-training powered generative adversarial networks, IEEE Wirel. Commun. 29 (2) (2022) 38–43.
[27] Y. Li, W. Dai, J. Bai, X. Gan, J. Wang, X. Wang, An intelligence-driven security-aware defense mechanism for advanced persistent threats, IEEE Trans. Inf. Forensics Secur. 14 (3) (2019) 646–661.
[28] H. Poston, Top 10 network recon tools, 2019, [Online]. Available: https://resources.infosecinstitute.com/topic/top-10-network-recon-tools/.
[29] D. Kreutz, F.M.V. Ramos, P.E. Veríssimo, C.E. Rothenberg, S. Azodolmolky, S. Uhlig, Software-defined networking: A comprehensive survey, Proc. IEEE 103 (1) (2015) 14–76.
[30] NIST, National vulnerability database (NVD), 2005, [Online]. Available: https://nvd.nist.gov/.
[31] M. Ge, J.B. Hong, S.E. Yusuf, D.S. Kim, Proactive defense mechanisms for the software-defined internet of things with non-patchable vulnerabilities, Future Gener. Comput. Syst. 78 (2018) 568–582.
[32] S. Sepahyar, R. Vaziri, M. Rezaei, Comparing four important sorting algorithms based on their time complexity, in: Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence, 2019, pp. 320–327.

Xingsheng Qin is currently pursuing a Ph.D. degree with the Faculty of Science, Engineering, and Built Environment (SEBE) at Deakin University. After obtaining his master's degree from the University of Shanghai for Science and Technology, he has been working in academia and industry for over ten years. His research interest is information security, cyberSecurity, edge computing, and Industrial Internet of Things (IIoT).

Frank Jiang received a Ph.D. from the University of Technology Sydney in 2008 and a master's degree in computer science from the University of New South Wales (UNSW). He gained 3.5 years of post-doctoral research experience at UNSW. He has published over 80 highly reputed SCI/EIindexed journals and conference articles. His main research interests include biologically inspired learning schemes and their applications in context-aware systems, data-driven cyberSecurity, predictive analytics, and blockchain techniques.

Xingguo Qin is pursuing a Ph.D. degree with the School of Computer and Information Security at Guilin University of Electronic Technology. He received an M.S. degree from Shantou University in 2010. His research interests mainly focus on cyberSecurity, natural language processing, generative adversarial networks.

Lina Ge received the M.S. degree in computer science and technology from Guangxi University, Nanning, in 2004, and the Ph.D. degree in computer science and technology from the South China University of Technology, Guangzhou, China, in 2009. Since 2010, she has been a Professor with the School of Artificial Intelligence, Guangxi Minzu University. She is the author of more than 70 articles. She holds three patents and more than ten software copyrights. Her research interests include information security, the IoTs, and intelligent computing.



Meiqu Lu received her Ph.D. degree from Sun Yat-sen University in 2023. She is currently a lecturer with the School of Artificial Intelligence, at Guangxi Minzu University. Her research interests include artificial intelligence, big data, and information security.

Robin Doss (Senior Member, IEEE) is currently a Professor and the Research Director of the Deakin Cyber Research and Innovation Centre, Deakin University. In this role, he provides scientific leadership for this multidisciplinary research centre focused on cybersecurity's technical, business, human, policy, and legal aspects. He also leads the Next-Generation Authentication Technologies theme for the Critical Infrastructure Security Research Program of the National Cyber Security Cooperative Research Centre (CSCRC). His research interests include system security, protocol design, and security analysis, focusing on smart, cyber–physical, and critical infrastructures. In 2019, he received the Cyber Security Researcher of the Year Award from the Australian Information Security Association (AISA). He is the Founding Chair of the Future Network Systems and Security (FNSS) conference series. He is an Associate Editor of the Journal of Cyber-Physical Systems.