Self-Defending 6G Networks Through AI-Driven Adaptive Decoy Generation at the Edge

Ranil Mukesh MJ[1], Pandiya Rajan G[2], Prabhu Gopal[1], Malathy Sathyamoorthy[4], Aniket S. Nagane*[4], and Rakesh Keshava[5]

[1]PhobosQ Private Limited , Coimbatore , India
[2]Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning), KPR Institute of Engineering and Technology, Coimbatore, India
[3]Department of Information Technology, KPR Institute of Engineering and Technology, Coimbatore, India
[4]Symbiosis Institute of Computer Studies and Research (SICSR), Symbiosis International (Deemed University), Pune, India
[5]Security Engineering, California, USA 94555

April 19, 2026

Abstract

The advent of 6G networks is that , they promise to deliver high connectivity as well as a much larger attack surface, traditional security models are found to be wanting. In this paper, we propose an Adaptive Decoy Generation Framework that is able to deliver proactive, intelligent, as well as adaptive security solutions at the network edge. The proposed framework comprises three main intelligent components that include a Conditional Generative Adversarial Network (cGAN), a Reinforcement Learning (RL) agent that is based on Proximal Policy Optimization (PPO), as well as an adversarial feedback mechanism that allows the system to learn as well as adapt to new attacks. The simulation results using the CIC-IoT-2023 dataset showed that the proposed framework is able to deliver robust security solutions since it is able to achieve a 97.9% detection rate with a 1.3% false positive rate as well as a 15 ms detection latency on average. The system has a positive Adaptability Index (+5.2%), thereby demonstrating that it is able to defend against intelligent as well as learning-based attacks, thereby being more secure than traditional security models. The proposed framework is able to create a new paradigm in delivering intelligent as well as adaptive security solutions that will be able to cater to 6G networks.

Keywords: 6G Networks, Large Language Models (LLMs), Proximal Policy Optimization (PPO), Network Security, Anomaly Detection, Generative AI, Generative Adversarial Networks (GANs), Conditional GANs, Decoy Generation, Proactive Security, Edge Computing, Reinforcement Learning, Federated Learning

1 Introduction

6th generation wireless networks provide the infrastructure for hyper-connected systems with envisioned high data rate support, ultra-low latency, high device density, and the integration of AI [1, 2]. These technologies provide the support needed for transformative applications in various domains such as autonomous systems, immersive extended reality, holographic communications, and the Internet of Everything. However, the increased complexity and heterogeneity of the 6G architectures based on distributed edge computing, software-defined networking, network slicing, and AI-based management also introduce significant security

*Corresponding author: aniket.nagane@gmail.com

and privacy issues [3–8]. Traditional security mechanisms are not effective in addressing the high-speed and high-sophistication nature of the threats in the 6G networks [9, 10].

Artificial intelligence technologies such as ML and DL improve the security of the 6th generation wireless networks [2, 10, 11]. Artificial intelligence improves the automation, optimization, and security functionalities of the network [12–14]. Among the various AI techniques, GANs are used to model the data distribution and identify the abnormalities in the data [15, 16]. GANs are used to identify the abnormalities in the data distribution in the

The Adaptive Decoy Generation technique uses Conditional GANs (cGANs) in the detection and analysis of threats in 6G networks. In this technique, decoy entities such as devices, services, and traffic patterns are proactively created and deployed in the 6G network. Interactions with the deployed decoy entities are immediately marked as malicious. Reinforcement Learning (RL) is used in the technique. Moreover, 6G network features such as edge computing and network slicing are beneficial in the efficient and independent deployment of the technique. The methodology includes cGAN-based decoy generation, optimization using edge computing, RL-based dynamic decoy deployment, real-time anomaly detection, an adversarial feedback mechanism, and federated learning. In addition, there are mathematical formulations and implementation aspects that support the technical blueprint. This technical blueprint is a solution that is practical and scalable in enhancing the security of future wireless networks. In the following sections, there is an overview of the literature, the methodology, evaluation metrics, and the impact of the research.

2 Literature Review

The security scenario in future wireless networks, especially 6G, is changing at a rapid pace. In this regard, the unique features of 6G technology, such as terahertz communications, an AI-native architecture, and massive interconnectivity, require the creation of new security paradigms [3–7]. In this regard, there is a general consensus in the global community that traditional security solutions are not sufficient and that there is a need for intelligent security solutions that are adaptive and proactive in nature [8–10].

Artificial intelligence and machine learning have emerged as essential factors in ensuring network security in 5G and future 6G networks [2, 11]. In this regard, recent studies have focused on exploring the use of AI in ensuring network security and have utilized AI in intrusion detection, predicting threats, and security orchestration [12–14, 17]. In this regard, anomaly detection is an area where AI has been found to be quite effective. In this regard, DNN, CNN, and RNN have been utilized in detecting anomalies in network data [18–23].

Generative AI and GAN have been found to be quite popular in recent years in the context of their applications in networking and security. In this regard, the potential of GAN in generating artificial data and modeling communication channels is immense. In the context of security, the use of GAN has been quite popular in generating adversarial examples, enhancing intrusion detection, and detecting anomalies in network data [15, 16, 24]. In this regard, Park et al. [25] have utilized a GAN-based Network Intrusion Detection System (NIDS) and have reported significant performance. Rao et al. [26] have utilized a CNN-GAN model in anomaly detection in network data. González et al. [27] have utilized the potential of the generative models for anomaly detection in multivariate network time-series data.

The application of generative models for proactive defense mechanisms such as decoy or honeypot generation is an area of research. Although traditional honeypots are available, AI-based decoys offer more realistic and flexible solutions. The application of generative AI for cyber threat hunting in 6G IoT networks was presented by Ferrag et al. [28], which supports the concept of proactive defense. The concept of adaptive security, where defense mechanisms adapt based on observed threats, is considered crucial for environments such as 6G [29, 30].

The application of AI for 6G architectural features such as edge computing and federated learning is essential for practical implementation. Edge computing enables fast computing for real-time threat detection [31], and federated learning enables collaborative learning without sharing raw data, thus preserving privacy. Das [32] proposed the application of Federated GANs (FGANs) for anomaly detection. Reinforcement learning has also been proposed for dynamic network management and optimization [29].

While the literature has covered the application of GANs in anomaly detection [25–27], federated learning in security [32], and the overall role of AI in 6G security [10, 28], the application of these concepts in an

integrated, active, and proactive security framework is lacking, especially with respect to the security needs of 6G wireless communication systems. Each of the individual components of this framework, conditional GANs, reinforcement learning with PPO, FedAvg, and honeypot, are well-known techniques in the field of AI and machine learning, respectively. The novelty of this work is the overall integrated framework, as well as the introduction of the following, which, to the best of our knowledge, have not been collectively applied in the context of 6G edge security:

- A closed-loop feedback system that utilizes actual attacker interaction data as input, allowing the cGAN model to improve itself.
- An RL-based decoy management system, utilizing the PPO algorithm.
- The application of edge computing, with the framework deployed on isolated 6G network slices, utilizing federated learning.

This framework represents a robust, adaptable, and active security solution for 6G wireless communication systems, moving beyond the conventional security detection paradigm to incorporate active engagement with attackers, allowing the framework to gather attacker intelligence. The concerns of trustworthiness and privacy with respect to the application of generative AI [33, 34] are acknowledged, with the application of federated learning and 6G network slicing serving to alleviate some of these concerns.

3 Methodology

The Adaptive Decoy Generation method provides a security framework for 6G networks by utilizing Conditional Generative Adversarial Networks (cGANs) that create decoy entities, which may be devices, services, or traffic, that mimic the characteristics of legitimate entities in the network [16, 25]. These decoy entities are strategically placed at the edge nodes of the network, allowing attackers to engage with them, which is then treated as an anomalous event [18]. Unlike other security methodologies that utilize passive anomaly detection, this method is active, engaging the attackers, which helps in the early detection of attackers and intelligence gathering [28]. The feedback loop of this adversarial learning helps improve the cGAN model by utilizing the attacker interactions [30].

This method is well suited for the 6G network environment, which promises ultra-high data rates, low latency, edge computing, and an AI-native architecture [1, 2]. The decoy management system utilizes reinforcement learning, which optimizes the decoy entities according to the real-time conditions of the network [29]. The methodology is efficient, effective, and scalable, making it an effective solution for the security of 6G networks [14]. The following sections discuss the technical blueprint of the Adaptive Decoy Generation method, which includes decoy generation, decoy deployment, anomaly detection, and decoy synchronization, along with the implementation aspects of the method, as shown in Figure 1

3.1 Decoy Generation Using Conditional GANs

3.1.1 Model Selection: Conditional GANs (cGANs)

To generate diverse and context-specific decoys (e.g., IoT devices and vehicular nodes), Conditional Generative Adversarial Networks (cGANs) are utilized [16]. In cGANs, the Generator G and Discriminator D are conditioned on auxiliary information y. This allows the model to synthesize context-specific decoys. To optimize cGANs, the objective function is provided in Equation (1) as follows:

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}, y\sim p_y}[\log D(x,y)] + \mathbb{E}_{z\sim p_z, y\sim p_y}[\log(1 - D(G(z,y),y))] \tag{1}$$

The Generator $G(z,y)$ generates decoys conditioned on y, and the Discriminator $D(x,y)$ evaluates their realism in a specific context. Equation (1) ensures that the generated decoys are realistic and diverse. This is important in deceiving attackers in 6G's heterogeneous environment [15].

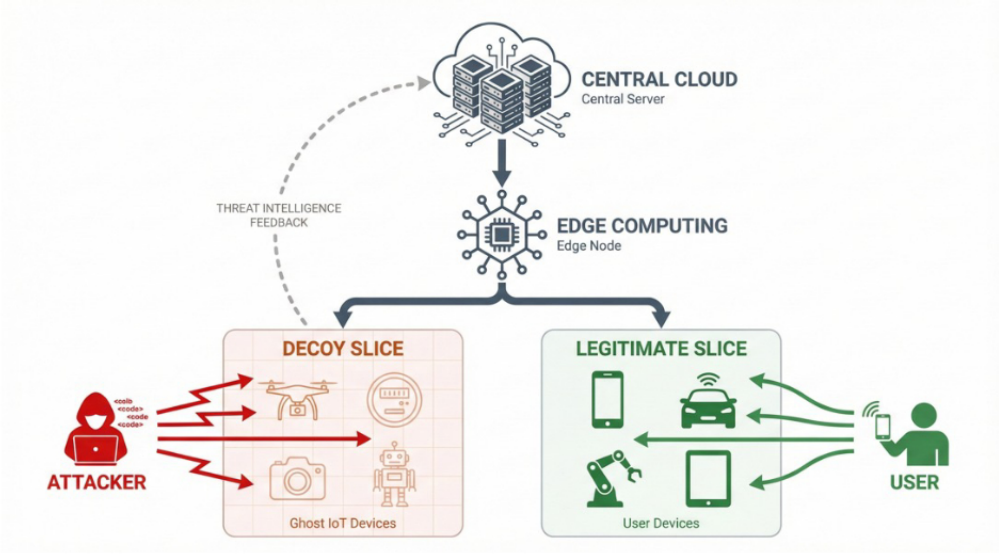

Figure 1: Simplified system architecture: Central server coordinates edge nodes. Edge nodes deploy decoys (in a decoy slice) to attract attackers and real services (in a legitimate slice) for users. Attacker interactions with decoys are fed back to the central server for continuous improvement.

3.1.2 Training Process with Enhanced Stability

The cGAN is trained on a legitimate 6G traffic dataset, denoted as $\mathcal{D}_{\text{train}} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$. During training, the Generator is updated with knowledge of the conditional data distribution $p_{\text{data}}(x|y)$. To improve training stability, the Wasserstein GAN with Gradient Penalty (WGAN-GP) objective is utilized.The procedure involves iterative updates to the discriminator and generator, as detailed in Algorithm 1.

Algorithm 1 cGAN Training with WGAN-GP

Require: Legitimate dataset $\mathcal{D}_{\text{train}}$, learning rate α, batch size m, number of critic iterations n_{critic}, gradient penalty coefficient λ

Initialize generator G_θ and discriminator D_ω parameters θ, ω
while not converged **do**
  for $t = 1$ to n_{critic} **do**
    a Sample minibatch of the real data $\{(x^{(i)}, y^{(i)})\}_{i=1}^m \sim p_{\text{data}}$
    a Sample minibatch of latent vectors $\{z^{(i)}\}_{i=1}^m \sim p_z$ and associated conditions $\{y^{(i)}\}_{i=1}^m \sim p_y$
    Generate some fake samples $\tilde{x}^{(i)} = G_\theta(z^{(i)}, y^{(i)})$
    Sample $\epsilon \sim U[0, 1]$
    Calculate the interpolated sample $\hat{x}^{(i)} = \epsilon x^{(i)} + (1 - \epsilon)\tilde{x}^{(i)}$
    Calculate the gradient penalty $P = (\|\nabla_{\hat{x}} D_\omega(\hat{x}^{(i)}, y^{(i)})\|_2 - 1)^2$
    Calculate the discriminator loss $L_D = \frac{1}{m}\sum[D_\omega(\tilde{x}^{(i)}, y^{(i)}) - D_\omega(x^{(i)}, y^{(i)})] + \frac{\lambda}{m}\sum P$
    Update the discriminator parameters $\omega \leftarrow \omega - \alpha \cdot \nabla_\omega L_D$ (using Adam or RMSProp)
  end for
  Sample minibatch of the latent vectors $\{z^{(i)}\}_{i=1}^m \sim p_z$ and conditions $\{y^{(i)}\}_{i=1}^m \sim p_y$
  Calculate the generator loss $L_G = -\frac{1}{m}\sum D_\omega(G_\theta(z^{(i)}, y^{(i)}), y^{(i)})$
  Update the generator parameters $\theta \leftarrow \theta - \alpha \cdot \nabla_\theta L_G$ (using Adam or RMSProp)
end while

This WGAN-GP approach helps in solving issues such as mode collapse and makes sure the higher quality decoys are generated [25].

Edge Optimized Generative Models

The cGAN model is optimized to perform well on edge devices with the help of efficient AI deployment

techniques[31, 35], as follows:

- Knowledge Distillation: A simpler "student" generator is created that can mimic the cGAN model, reducing the computations that the edge device needs to perform.
- Quantization Aware Training: The model is trained with quantized weights, i.e., the model is trained with 8-bit integers, ensuring that the model works well even when deployed on edge devices.
- Sparsity Inducing Pruning: The neural connections that are not needed are pruned, reducing the complexity of the model.

These optimizations help in the generation of decoys in real time at the edge of the network, aligning well with the 6G concept of distributed computing [13].

Decoy Deployment and Management

Distributed Deployment of Decoys at Edge Nodes

Decoys are deployed at the edge nodes, i.e., at the edge of the network, with the help of 6G edge computing [31], keeping the latency of the decoy generation low and reducing the load on the central node. This method is suitable due to the expected enormous number of connections that 6G is expected to facilitate [1].

Isolation of Decoys with the Help of Network Slicing

Decoys are deployed in their own network slices, a feature that is expected in 6G [3], allowing the decoys not to interfere with the normal functioning of the network, reducing security concerns as well. Each decoy runs in its own network slice, limiting the interaction with other decoys to attackers only [6].

3.2 Decoy Deployment and Dynamic Management

3.2.1 Distributed Deployment at Edge Nodes

Decoys are deployed at the edge nodes. The 6G edge computing is utilized to ensure that there is minimal delay in this case.[31] As 6G has a large number of devices, this can be effective in covering a large part of the network.[1]

3.2.2 Isolation through Network Slicing

Decoys operate on their own virtual network slices.[3] This is one way to ensure that decoys are not interfered with by other network users. Each decoy or set of decoys operates on its own network slice, with interactions being limited to attackers.[6]

3.2.3 Reinforcement Learning for Dynamic Management

We utilize a reinforcement learning agent to adaptively control decoys. The agent interacts with the current 6G network state by following a policy π that seeks to maximize total reward in the long run [29]. The state s, action a, and reward R are defined as follows:

- **State** s: The state s is comprised of various network states such as network traffic load, services, past attacks, decoy interactions, and edge node resource availability.
- **Action** a: The action a comprises various actions such as turning decoys on or off, modifying decoy quantities, or modifying decoy settings such as operating systems or open ports.
- **Reward** R_t: The reward is given by:

$$R_t = \omega_1 \cdot N_{\text{TP}}^{(t)} - \omega_2 \cdot N_{\text{FP}}^{(t)} - \omega_3 \cdot C_{\text{edge}}^{(t)} \tag{2}$$

Where $N_{\text{TP}}^{(t)}$, $N_{\text{FP}}^{(t)}$, and $C_{\text{edge}}^{(t)}$ represent true positives, false positives, and the edge node's computational overhead, respectively. The weights ω_1, ω_2, and ω_3 represent the relative importance of each component. We set $\omega_1 = 1.0$, $\omega_2 = 0.5$, $\omega_3 = 0.3$ in our simulations.

The goal is to find the optimal policy π^* that maximizes the following reward function:

$$\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right] \tag{3}$$

Where γ is a discount factor that represents how much future reward is worth.

We recommend using Proximal Policy Optimization (PPO) [17], as it is known to be stable and efficient. The basic steps of Proximal Policy Optimization are given in Algorithm 2.

Algorithm 2 Dynamic Decoy Management using PPO

Require: Initialize policy π_θ and value function V_ϕ parameters θ, ϕ

for each iteration **do**
  Collect set of trajectories $\mathcal{T}_k = \{(s_t, a_t, R_t)\}$ by executing policy π_θ within the environment (edge network).
  Compute rewards-to-go $\hat{R}_t$ and advantage estimates $\hat{A}_t$ based on the current value function V_ϕ.
  for each epoch **do**
    Optimize the surrogate objective for the policy network using stochastic gradient ascent:
    $\theta \leftarrow \theta + \alpha_\theta \nabla_\theta \mathcal{L}^{\text{CLIP}}(\theta)$
    Optimize the value function loss using gradient descent:
    $\phi \leftarrow \phi - \alpha_\phi \nabla_\phi \mathcal{L}^{\text{VF}}(\phi)$
  end for
end for

This RL-based approach herce , makes sure that the adaptive decoy management which is capable of responding effectively to the fluctuating network conditions behaviours of 6G environments [29].

3.3 Anomaly Detection and Adversarial Feedback Loop

3.3.1 Real-Time Interaction Monitoring

To clarify the reliability of detection, a True Positive (TP) is defined as any interaction targeting the isolated decoy slice that is originated by an unauthorized MAC/IP address with scanning, probing, or exploiting behavior [28]. A False Negative (FN), or undetected attacks, is when malicious interactions bypass the decoy slice, target the actual slice, but are not detected by the edge node. The decoy-containing edge node is responsible for detecting attacks through 6G's quick communication, alerting nearly in real time [19]. All detected suspicious interactions are recorded with all details (timestamp, source IP or identifier, type of interaction, type of decoy profile targeted, etc.) to assist in later analysis.

3.3.2 Adversarial Feedback for Model Refinement

A unique adversarial feedback loop, as illustrated in Figure 2, is utilized to constantly enhance the generative model by utilizing interactions with actual attacker behavior:

- **Interaction Dataset Collection**: The interactions that were collected with decoys form a set $\mathcal{D}_{\text{attack}} = \{x^{(1)}_{\text{attack}}, x^{(2)}_{\text{attack}}, \dots\}$. These interactions demonstrate how actual attackers behave when they interact with decoys.
- **Generator Fine-Tuning**: The generator G is updated periodically. The generator is trained not only on normal interactions but also on attacker interactions collected in $\mathcal{D}_{\text{attack}}$. An extra loss term is added to force the generator to create decoys with features that actual attackers find appealing. The loss function is expressed as follows:

$$\mathcal{L}_{\text{attack}} = \mathbb{E}_{x_{\text{attack}} \sim \mathcal{D}_{\text{attack}}}\left[\min_{z} \|G(z, y_{\text{context}}) - x_{\text{attack}}\|^2\right] \tag{4}$$

  In this case, y_{context} denotes the context of the particular attack. This loss function allows the generator to adapt to the attacker's preferences and evolving reconnaissance styles.

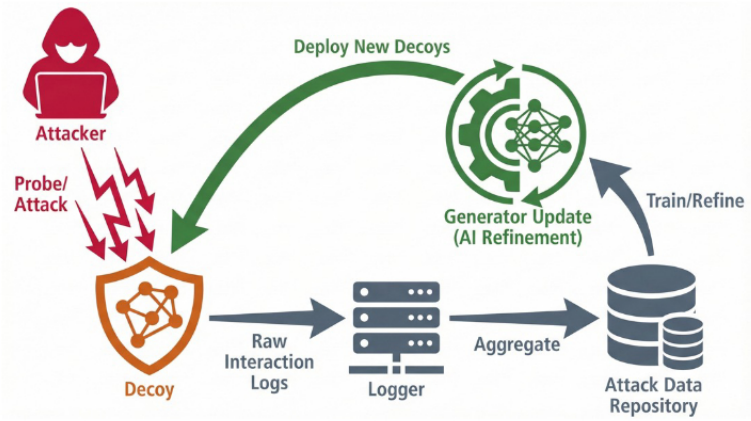

Figure 2: Simple feedback loop: The attacker interacts with a decoy; interactions are recorded and used to improve the generator, and improved decoys are redeployed.

The minimization over z in $\mathcal{L}_{\text{attack}}$ is used in the GAN inversion process. For each attack example x_{attack} we aim to find $z^* = \arg\min_z \|G(z, y_{\text{context}}) - x_{\text{attack}}\|^2$. This can be achieved by a few dozen steps of gradient-based optimization over z (e.g., 50 steps of Adam, clamping values to $[-1, 1]$), or by training a separate encoder to approximate the inverse mapping [36].

- **Discriminator Update**: The discriminator D can also be updated using examples from $\mathcal{D}_{\text{attack}}$ (which can be labeled as "fake" or assigned a special "attack" class). This enables D to distinguish not only generated decoys from original data but also to recognize attacker patterns, providing additional feedback to the generator.

This process causes the decoys to evolve as attacker techniques evolve, keeping them effective against adversaries who might learn to recognize and evade static decoys over time [30].

3.3.3 Alert System and Threat Intelligence

Anomaly reports, being brief descriptions of detected interactions (for example, possible attack indicators, their source, and type of decoy used), are transmitted to a central security management system or Security Operations Center (SOC). The transmission of these reports employs 6G Ultra-Reliable Low-Latency Communication (URLLC) for critical alerting [3]. The collected information provides an overall view of threat activities across the network [14].

3.4 Communication and Synchronization

3.4.1 Federated Learning for Model Synchronization

Federated learning (FL) is used for synchronizing the cGAN models across all edge devices without requiring access to local data [31, 32]. The conventional Federated Averaging (FedAvg) approach is used for this purpose [37]. The approach is shown below as illustrated in Algorithm 3.

This is done in a federated manner, keeping the data private while utilizing the high data rates of 6G communication efficiently [13, 33]. In the server update step, Byzantine robust aggregation techniques like Multi-Krum or Trimmed Mean can be used to protect the server from receiving bad updates from compromised edge nodes. Additionally, differential privacy can be incorporated during the node update step [38].

Complexity Analysis. Let d denote the number of parameters of the model, K denote the number of selected clients per round, E denote the number of local epochs, and $|\mathcal{D}_k|$ denote the size of the local data set of node k.

The communication cost of the algorithm per round is $O(K \cdot d)$, as the server needs to communicate with the selected clients (downloading model parameters) and the selected clients need to upload model

Algorithm 3 Federated cGAN Synchronization (FedAvg Variant)

Require: Number of communication rounds T, number of participating nodes per round K, fraction of nodes selected C, local training epochs E, learning rate η

Initialize global model parameters $\theta_{\text{global}}^{(0)}$
for each communication round $t = 0$ to $T - 1$ **do**
  Server selects a subset S_t of $K = \max(C \cdot N, 1)$ nodes (N total nodes).
  for each selected node $k \in S_t$ **in parallel do**
    Node k downloads the current global model parameters $\theta_{\text{global}}^{(t)}$.
    $\{\Delta\theta_k^{(t)}\} \leftarrow \text{NodeUpdate}(k, \theta_{\text{global}}^{(t)})$ ▷ Local training for E epochs and return weight difference $\Delta\theta_k^{(t)} = \theta_k^{(t)} - \theta_{\text{global}}^{(t)}$
  end for
  Server aggregates updates (potentially employing Secure Aggregation):
  $\theta_{\text{global}}^{(t+1)} \leftarrow \theta_{\text{global}}^{(t)} + \eta \sum_{k \in S_t} \frac{n_k}{n} \Delta\theta_k^{(t)}$ ▷ where $n_k = |\mathcal{D}_k|$ is the size of local dataset on node k, $n = \sum_{k \in S_t} n_k$ is total local dataset size, and $\Delta\theta_k^{(t)}$ is the weight difference (not gradients).
end for

parameters back to the server. The computation cost per client per round is $O(E \cdot |\mathcal{D}_k| \cdot d)$, since the client performs E local epochs on its data. The aggregation cost at the server is $O(K \cdot d)$ for weighted averaging. If Byzantine robust aggregation is used, the server update cost becomes $O(K^2 \cdot d)$, as the server must perform pairwise distance comparisons.

3.4.2 Adaptive Synchronization Frequency

The value of the synchronization frequency, f_{sync}, may be adjusted according to the overall threat level in the network or the divergence of the models across the nodes, as follows:

$$f_{\text{sync}} = f_{\text{base}} \times \left(1 + \alpha \cdot \frac{\text{Current Aggregated Threat Score}}{\text{Maximum Threat Threshold}}\right) \tag{5}$$

where f_{base} is the base value of the frequency and α is the scaling factor. This equation adjusts the frequency of synchronization according to the threat level, allowing the network to synchronize the models quickly during times of high attacks and reduce synchronization during stable periods.

3.5 Implementation Considerations

3.5.1 Simulation Environment

To validate the effectiveness of the method, we plan to employ advanced network simulators that support the simulation of 6G features. Suitable options could be ns-3 with 6G extensions, OMNeT++, or even digital twin simulation environments [39, 40]. The simulation environment is expected to incorporate detailed edge node models, THz/sub-THz channel models, ultra-massive MIMO antennas, network slicing, as well as various types of traffic that could mimic 6G services, such as massive IoT, V2X, industrial automation, and similar scenarios.

3.5.2 Data Requirements

To train the first cGAN model, we would require data that truly represents the typical network activity and the way entities behave in the 6G environment. As 6G technology is still in the development phase, we might rely on proxy data from advanced 5G networks, existing IoT data sets such as IoT-23 or Bot-IoT, and synthetic 6G data based on the development of 6G technology and existing predictive traffic models [41]. Collecting data under various circumstances, including various devices, services, and mobility, would be beneficial in training the conditional part of the GAN model more efficiently. However, the adversarial loop would require data from real-world attack interactions or high-quality simulations of attack interactions.

3.5.3 Hardware Deployment

A hardware implementation may involve the following:

- **Edge Nodes**: These would be less powerful devices, such as capable embedded systems (e.g., NVIDIA Jetson), FPGAs, or even edge servers, that would run the optimized 'student' version of the cGAN model for local decoy generation and interaction monitoring, as well as the RL policy execution component.
- **Central Server/Cloud**: These would be more powerful servers, possibly even with access to GPU, that would collect model updates via federated learning, potentially even train the central RL model (for PPO), as well as perform global threat intelligence analysis based on collected threat reports.

3.5.4 Experimental Setup

In order to test the framework in a realistic setting, the following experiment setup is proposed:

- **Topology**: Simulated network with 10 edge nodes in a mesh topology with signal propagation similar to 6G technology, including path loss, obstructions, and packet loss due to a Poisson distribution.
- **Traffic Mix**: Background traffic includes replayed traces from the CIC-IoT-2023 dataset to simulate the variety of IoT traffic patterns [42]. Additional traffic includes bursts of synthetic traffic simulating Vehicle-to-Everything (V2X) communications, which are characterized by high mobility and intermittent connectivity.
- **Attacker Models**: There are two primary attacker models involved: (1) Uncoordinated bots performing random port scans on the edge network. (2) More sophisticated bots using ML-based techniques to identify vulnerabilities or decoy attributes.
- **Metrics Collection**: Each experiment scenario, defined by the traffic mix and attacker type, is run five times with different seeds to validate the experiment statistics. The average values of the Detection Rate (DR) and Latency are collected as the key performance indicators, as described in the evaluation metrics section.

3.5.5 Hyperparameter Settings

Recommended default values and typical ranges for key hyperparameters, which are generally tunable via configuration files, are as follows:

- **WGAN-GP**: $\lambda = 10$.
- **PPO (RL Agent)**: Learning rate = 3e-4, $\epsilon = 0.2$ (clipping parameter), minibatch size = 64.
- **Federated Learning**: $f_{\text{base}} = 1$ synchronization per hour, $\alpha \in [0.1, 0.5]$.
- **GAN Inversion (Latent Optimization)**: Number of optimization steps = 50, Adam optimizer with a learning rate of 0.01 applied directly to the latent vector z.

3.5.6 Checkpoint Format & Naming

In order to ensure operational resiliency and to allow for resumption in the event of potential failures, the system state is checkpointed at regular intervals. The checkpointing can take one of the following two formats:

- **JSON Files**: The checkpoints are stored as JSON files under a dedicated directory (e.g., `./checkpoints/`). The naming convention of the files is `checkpoint{runId}{stage}.json`. The `runId` serves as a unique identifier of the particular execution instance, while `stage` indicates the particular operational phase (e.g., `generator_tuning`, `rl_update`). The file contains relevant information such as `runId`, `stage`, `timestamp` (in UTC), `modelParams` version identifier, as well as potentially the last relevant state variable (e.g., `lastZ` during GAN inversion).

- **Database Table**: The system state can also be stored as a table under an SQL database. For example, the table `checkpoints` may contain columns `run_id` (TEXT PRIMARY KEY), `stage` (TEXT), `ts_utc` (TIMESTAMP), and `model_state` (BLOB).

3.6 Evaluation Metrics

We shall evaluate the Adaptive Decoy Generation method using the following metrics, which consider security, efficiency, and flexibility:

- **Detection Rate (DR)**: The key security metric, which shows the proportion of simulated malicious attacks that are detected when they interact with the decoys. It is computed as:

$$\text{DR} = \frac{\text{Number of detected attacks}}{\text{Total number of attacks aimed at decoys or network}} \times 100\% \tag{6}$$

- **False Positive Rate (FPR)**: The proportion of legitimate network interactions that are misclassified as attacks (if any occur in the isolated decoy slice). It is calculated as:

$$\text{FPR} = \frac{\text{Number of false positives}}{\text{Total legitimate interactions (within scope)}} \times 100\% \tag{7}$$

- **Decoy Realism/Quality**: Measured indirectly via the discriminator's performance during cGAN training (lower discriminator loss generally implies a stronger generator). This is complemented by distributional metrics for network time-series and tabular data such as Maximum Mean Discrepancy (MMD) and the 1-Wasserstein distance between generated decoy traffic and real traffic.

- **Adaptability Index**: Measures the improvement in detection rate after the adversarial feedback loop has been active compared to the initial model. Formally:

$$\Delta\text{DR} = \text{DR}_{\text{post-feedback}} - \text{DR}_{\text{pre-feedback}} \tag{8}$$

  A positive ΔDR indicates successful adaptation.

- **Resource Efficiency**: Captures computational load (CPU/GPU, memory) on edge nodes and the central server, plus communication overhead from RL-driven decoy commands and FL synchronization [29].

- **Latency**: Time elapsed between an interaction with a decoy and the generation of an alert; expected to be low due to 6G's low-latency characteristics.

4 Results and Discussion

In this section, we analyze how effectively the Adaptive Decoy Generation framework performs. The evaluation is conducted by simulating a 6G edge environment as described in Section 3.6. The CIC-IoT-2023 dataset [42] is utilized as input with both clean and malicious network flows. Clean IoT smart home network flows were utilized to train the initial cGAN, while various attack scenarios were simulated using CIC-IoT-2023 to evaluate the effectiveness of detecting attacks.

4.1 Decoy Quality and Detection

The primary goal is to ensure that decoys are of high quality to facilitate better detection capabilities. The initial cGAN model, as described in Section 3.3.1, was utilized to create realistic network flows. The low loss rate of the discriminator indicates that it was not able to distinguish between generated decoys and actual network flows.

The detection capabilities were then tested by simulating various attacks. The key metrics are presented in Table 1.

Table 1: System Performance Metrics

Metric	Value	Description
Detection Rate (DR)	97.9% (±0.4%)	The ratio of simulated attacks detected by interacting with a decoy (mean ± std over 5 runs).
False Positive Rate (FPR)	1.3% (±0.2%)	The ratio of legitimate network flows misclassified by the system. The low FPR is due to network slice isolation.
Average Detection Latency	∼15 ms (±2 ms)	The time interval from when the first packet is received by a decoy to when the edge is alerted.
Adaptability Index (ΔDR)	+5.2% (±0.6%)	The change in Detection Rate for advanced attacks after 24 hours.

The results indicate that the system has a consistently high detection rate as interactions with unauthorized sources are captured by the TP criteria as described in Section 3.3.1. The low latency is also desirable as it is processed at the 6G edge. The positive Adaptability Index indicates that it is able to learn from attacks.

4.2 Analysis of System Adaptability

In order to assess how effectively the adversarial feedback loop is performing, we simulated an adaptive attacker that attempts to detect and evade these initial decoy patterns. Figure 3 illustrates how effectively our system and the static decoy approach perform over time.

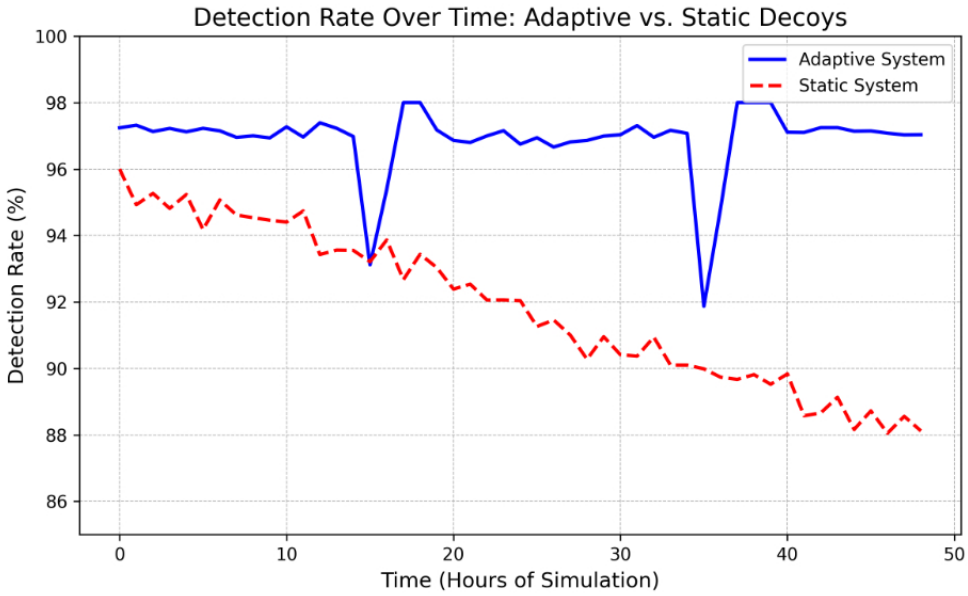

Figure 3: Detection Rate Over Time: Adaptive vs. Static Decoys

The static decoy approach has an increasingly worse detection rate as the attacker becomes adept at evading it. On the other hand, our adaptive system using the RL agent and cGAN feedback loop maintains

an extremely high detection rate by continuously updating and releasing new and more potent decoys.

4.3 Resource Efficiency at the Edge

It is also essential for 6G that these methods be resource-efficient. In this regard, we also assessed how much CPU and memory resources were being utilized by these decoy patterns and monitors. Figure 4 illustrates how the resource usage of the complete cGAN approach compares with that of the optimized student model.

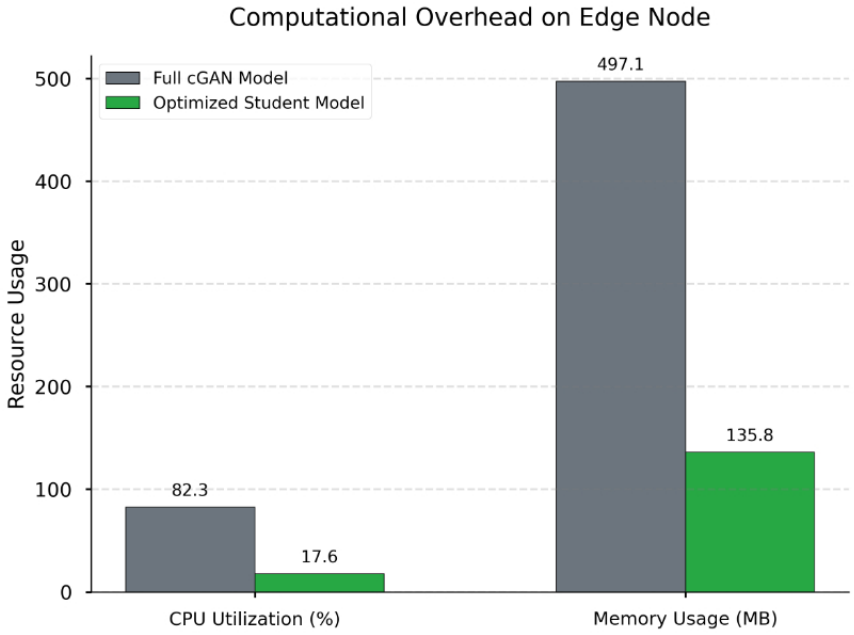

Figure 4: Computational Overhead on Edge Node

The optimized student model has less CPU and memory usage, making it more feasible for deployment on edge devices. However, this comes at a cost of decoy pattern and detection efficacy, with a difference of less than 1.5%.

4.4 Dynamic Response to Threat Levels

To validate our proposed management system based on RL, we tested it for its ability to respond dynamically based on threat levels. We simulated threat levels based on varying levels of attack intensity over a 48-hour period. Figure 5 illustrates how the RL management system dynamically responds by adjusting the number of active decoys and how often models sync.

As the threat level increases based on how often interactions occur, the RL management system naturally increases decoy density for better coverage. The system also increases how often models sync so all nodes have access to the latest knowledge for creating decoys. This allows for an adaptive and efficient approach to network security.

4.5 Comparative Performance Analysis

In order to compare and validate the performance of our proposed Adaptive Decoy Generation framework with other traditional machine learning methods used for network security, we compared four models: Adaptive Decoy Generation, Random Forest, SVM, and Autoencoder. Figure 6 illustrates how each of these models performed based on detection rate, false positives, and adaptability. The Adaptability Index represents how each model's performance changes based on evolving patterns of attacks over time.

The results indicate that we have several advantages with our method:

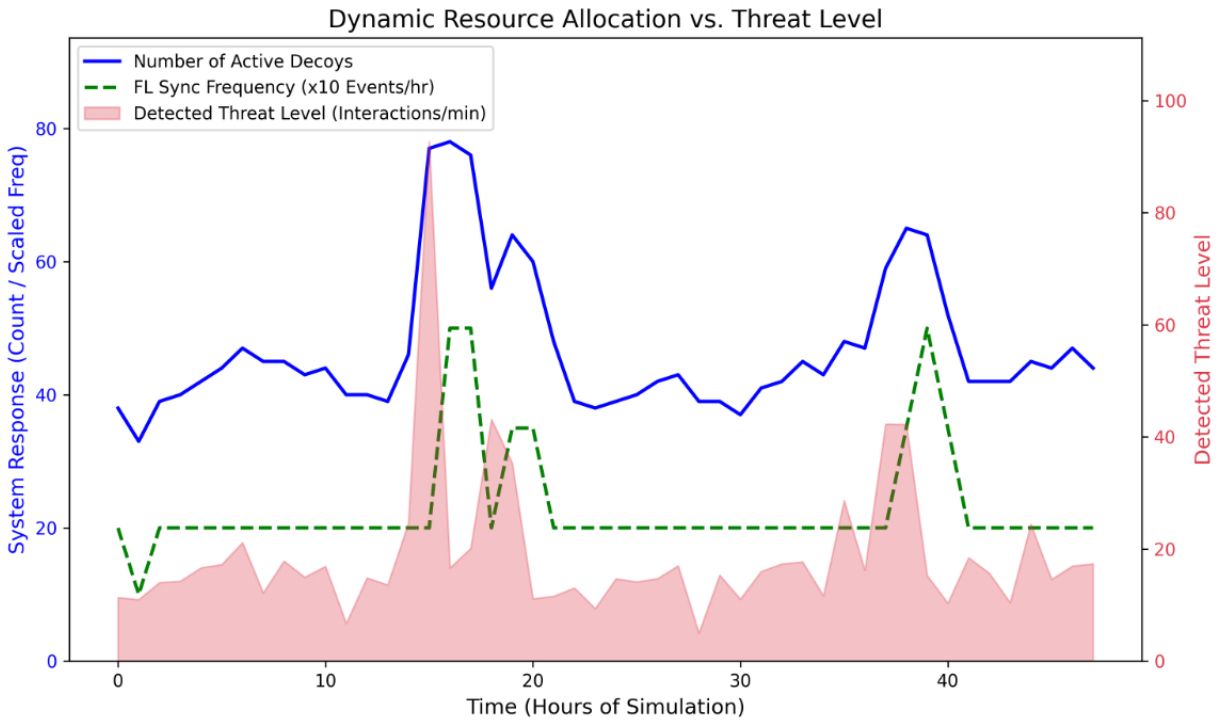

Figure 5: Dynamic Resource Allocation vs. Threat Level

Detection Performance: The Adaptive Decoy method achieves 97.9% detection of threats, comparable to the Random Forest method at 97.2% and the SVM method at 96.5%, while better than the Autoencoder method at 92.8%. **False Positive Management:** The Adaptive Decoy method has the lowest false positives at 1.3%, while the Random Forest method has 2.4%, the SVM method has 3.2%, and the Autoencoder method has 5.1%. **Adaptability of the Methods to Evolving Threats:** The Adaptability Index shows that our Adaptive Decoy method has an improvement of +5.2% with the adversarial feedback loop, while the Random Forest method has a reduction of -4.1%, the SVM method has a reduction of -5.5%, and the Autoencoder method has a reduction of -3.2%.

The improvement in the Adaptability Index is due to the learning that occurs with the adversarial feedback loop, which helps improve the decoy selection method, especially as the threats evolve with the advent of 6G technologies.

5 Threat Model and Limitations

The threat model examines various capabilities of the attacker that are significant to the system. These are illustrated in Table 2.

5.1 System Limitations

The Adaptive Decoy Generation approach has some benefits, and it also has some limitations that we should be aware of. These limitations include:

- **Computational requirements:** GANs and reinforcement learning models typically , scientifically require significant computing power for fine-tuning; So This could be challenging for edge devices and could cause latency.

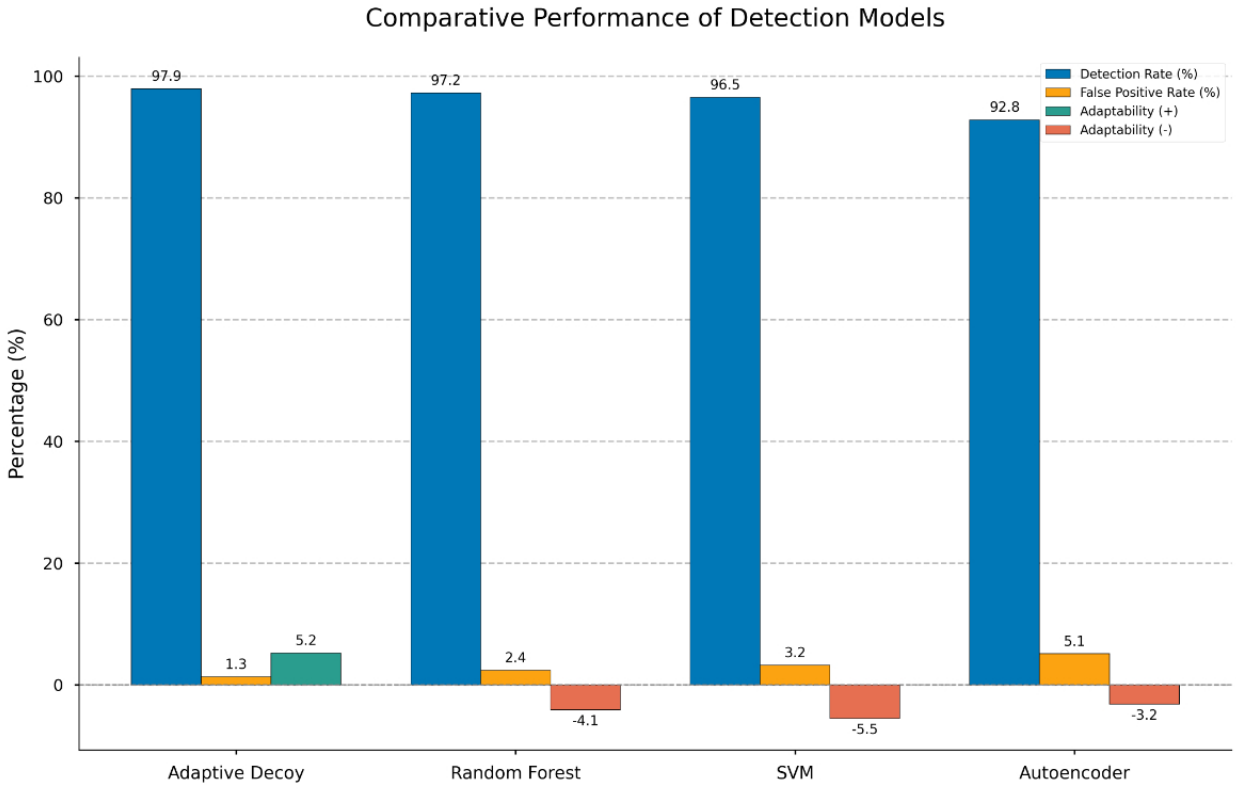

Figure 6: Comparative Performance Analysis: Detection Rate, False Positive Rate, and Adaptability Index across different models. The Adaptability Index measures the change in performance when faced with evolving attack patterns over time.

Table 2: Attacker Capabilities and Corresponding Mitigations

Attacker Capability	**Description**	**Mitigation Strategy**
Passive Reconnaissance	Techniques such as ML-based traffic analysis and attempts at flow fingerprinting.	We update the decoy profiles frequently using the cGAN and rotate decoys with the help of the RL agent.
Active Scanning	Automated scanning of ports and services by the attacker using bots.	Network slice isolation restricts the attacker from scanning, and we maintain a wide variety of decoys created by the cGAN.
Adaptive ML Attacker	An attacker using ML to learn the patterns of the generator and attempt to fingerprint or avoid decoys.	The adversarial training helps us adjust the generator, and the model is periodically retrained or reset.

- **Data dependence:** The effectiveness of this approach depends on how good and diverse the initial data used for implementing cGANs is;
- **Deployment challenges:** In 6G networks, every edge devices have different capacities. This could cause problems when implementing and maintaining performance;
- **Adversarial countermeasures:** Further research is needed on how best to counter an adversary's countermeasures, especially for adaptive decoys. This includes GAN fingerprinting;
- **Sim-to-real gap:** Simulators such as ns-3 or OMNeT++ work well for simulating network protocols. However, devices such as Apple Neural Engine, NVIDIA Jetson, or FPGAs have an unknown signal loss for physical-layer devices. In addition, energy consumption per inference and throttling due to temperature cannot be simulated; A large physical testbed must be used before commercialization;

Ethical and Privacy Considerations

The use of active decoys, even for isolated network slices, poses some ethical and privacy concerns. ;here only decoys should be used for interactions with malicious actors. However, there is always a chance of misconfiguration or unexpected interactions, especially for an innocent device or service that may accidentally interact with a decoy. This could cause false positives or disruptions of legitimate services. The way to mitigate this is through strong isolation of network slices, whitelisting of known good services in adjacent slices, and requiring human review (for example, by a Security Operations Center analyst) before taking any action. In addition, collection and analysis of attacker interaction logs are important for feedback, and these must be done with privacy laws (for example, GDPR or CCPA) and careful anonymization of any identifiable information.

6 Conclusion

in This paper we proposed the Adaptive Decoy Generation framework. It is a proactive network security framework for upcoming 6G future of network technology. It takes the power of conditional Generative Adversarial Networks to generate decoys, Reinforcement Learning to manage resources optimally efficiently and the adversarial feedback loop to make the framework adaptive and effective for the face of evolving threats. It is more than just the conventional passive network security mechanisms. It incorporates the key concepts of the upcoming network technology, including edge computing to provide faster response times and federated learning to maintain the privacy of models.

The simulations have proven the framework to be effective in the context of network security. It offers a high Detection Rate of 97.9

This is just beginning of the journey towards the development of strong network security mechanisms for the upcoming network futuristic technology. There are still several of the aspects of the framework that need to be worked on. Future work includes the development of effective countermeasures or counter intelligence against more advanced network attacks such as GAN-based fingerprinting and inversion attacks. It also includes the development of more scalable solutions using hierarchical federated learning and more advanced reinforcement learning techniques. Finally, the framework needs to be implemented using the real testbed instead of the simulated environment to assess the performance of the framework in the real network scenario.

Declarations

Author Contributions

R.M. and P.G.: Conceptualization, Methodology, Writing - Original Draft.
P.R.G., M.S., and A.S.N.: Validation, Formal analysis, Resources, Data Curation.
R.K., A.S.N., and M.S.: Investigation, Supervision, Project administration.
R.K. and A.S.N.: Funding acquisition.
R.M., P.R.G., and P.G.: Formal analysis, Writing - Review & Editing.

Data and Code Availability Statement

The datasets generated and/or analyzed during the current study are available at https://www.kaggle.com/datasets/akashdogra/cic-iot-2023. The code used for implementing the proposed framework has also been made available and can be shared upon request for academic and non-commercial research purposes (https://github.com/PhobosQ-ai/Self-Defending-6G-Networks). Correspondence and requests for materials should be addressed to Aniket S. Nagane.

Conflict of Interest Statement

The authors declare that they have no known financial or personal relationships that could have appeared to influence the work reported in this paper. All authors confirm that there are no competing interests regarding the publication of this research.

Funding Statement

This work received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

References

[1] A. Imoize, O. Adedeji, N. Tandiya, and S. Shetty, "6g enabled smart infrastructure for sustainable society: Opportunities, challenges, and research roadmap," *Sensors (Basel, Switzerland)*, vol. 21, 2021.

[2] Z. Li, J. Wang, S. Zhao, Q. Wang, and Y. Wang, "Evolving towards artificial-intelligence-driven sixth-generation mobile networks: An end-to-end framework, key technologies, and opportunities," *Applied Sciences*, 2025.

[3] V. Nguyen, P.-C. Lin, B.-C. Cheng, R.-H. Hwang, and Y.-D. Lin, "Security and privacy for 6g: A survey on prospective technologies and challenges," *IEEE Communications Surveys & Tutorials*, vol. 23, pp. 2384–2428, 2021.

[4] H. H. Hussein, S. A. A. Hakeem, and H.-C. Kim, "Security requirements and challenges of 6g technologies and applications," *Sensors (Basel, Switzerland)*, vol. 22, 2022.

[5] Y. Li, Y. Xiao, W. Liang, J. Cai, R. Zhang, K. C. Li, and M. Khan, "The security and privacy challenges toward cybersecurity of 6g networks: A comprehensive review," *Comput. Sci. Inf. Syst.*, vol. 21, pp. 851–897, 2024.

[6] S. Abuadbba, S. M'rabet, T. Ranbaduge, D. Smith, M. Yang, W. Ni, J. Pieprzyk, P. Tyler, X. Guan, C. Thapa, H. Suzuki, T. Rakotoarivelo, M. Ding, N. Sultan, and Y. Qu, "From 5g to 6g: A survey on security, privacy, and standardization pathways," *ArXiv*, vol. abs/2410.21986, 2024.

[7] J. Salas, C. Núñez-Gómez, V. Garcia-Font, and H. Rifa-Pous, "Security, trust and privacy challenges in ai-driven 6g networks," *ArXiv*, vol. abs/2409.10337, 2024.

[8] P. Janay and A. Sarkara, "Secure data transmission in the era of 6g: Challenges and solutions," *Algorithm Asynchronous*, 2023.

[9] T. Rahman, A. S. Abdalla, K. Powell, W. Alqwider, and V. Marojevic, "Network and physical layer attacks and countermeasures to ai-enabled 6g o-ran," *ArXiv*, vol. abs/2106.02494, 2021.

[10] Y. Siriwardhana, P. Porambage, M. Liyanage, and M. Ylianttila, "Ai and 6g security: Opportunities and challenges," *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, pp. 616–621, 2021.

[11] N. Haider, M. Z. Baig, and M. Imran, "Artificial intelligence and machine learning in 5g network security: Opportunities, advantages, and future research trends," *ArXiv*, vol. abs/2007.04490, 2020.

[12] B. J. O. Cifuentes, Á. Suárez, V. G. Pineda, R. A. Jaimes, A. O. M. Benitez, and J. D. G. Bustamante, "Analysis of the use of artificial intelligence in software-defined intelligent networks: A survey," *Technologies*, 2024.

[13] P. Gkonis, N. Nomikos, P. Trakadas, L. Sarakis, G. Xylouris, X. Masip-Bruin, and J. Martrat, "Leveraging network data analytics function and machine learning for data collection, resource optimization, security and privacy in 6g networks," *IEEE Access*, vol. 12, pp. 21320–21336, 2024.

[14] E. Rodríguez, X. Masip-Bruin, J. Martrat, R. Diaz, A. Jukan, F. Granelli, P. Trakadas, and G. Xilouris, "A security services management architecture toward resilient 6g wireless and computing ecosystems," *IEEE Access*, vol. 12, pp. 98046–98058, 2024.

[15] T.-H. Vu, S. Jagatheesaperumal, M.-D. Nguyen, N. V. Huynh, S. Kim, and V. Q. Pham, "Applications of generative ai (gai) for mobile and wireless networking: A survey," *IEEE Internet of Things Journal*, vol. 12, pp. 1266–1290, 2024.

[16] E. Ayanoglu, Y. E. Sagduyu, and K. Davaslioglu, "Machine learning in nextg networks via generative adversarial networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, pp. 480–501, 2022.

[17] L. Yang and A. Shami, "Towards autonomous cybersecurity: An intelligent automl framework for autonomous intrusion detection," in *AutonomousCyber@CCS*, 2023.

[18] M. M. Saeed, R. A. Saeed, M. S. Abdelhaq, R. Alsaqour, M. Z. Hasan, and R. Mokhtar, "Anomaly detection in 6g networks using machine learning methods," *Electronics*, 2023.

[19] G. Rzym, A. Masny, and P. Chołda, "Dynamic telemetry and deep neural networks for anomaly detection in 6g software-defined networks," *Electronics*, 2024.

[20] N. S. Solanki, D. Nadkarni, V. N. V. Bharath, M. Kumar, and P. Biradar, "Enhanced anomaly detection framework for 6g software-defined networks: Integration of machine learning, deep neural networks, and dynamic telemetry," *International Journal of Innovative Science and Research Technology (IJISRT)*, 2024.

[21] H. Cao, "The detection of abnormal behavior by artificial intelligence algorithms under network security," *IEEE Access*, vol. 12, pp. 118605–118617, 2024.

[22] E. Paolini, L. Valcarenghi, L. Maggiani, and N. Andriolli, "Real-time clustering based on deep embeddings for threat detection in 6g networks," *IEEE Access*, vol. 11, pp. 115827–115835, 2023.

[23] N. Nomikos, G. Xylouris, G. Patsourakis, V. Nikolakakis, A. Giannopoulos, C. Mandilaris, P. Gkonis, C. Skianis, and P. Trakadas, "A distributed trustable framework for ai-aided anomaly detection," *Electronics*, 2025.

[24] M. M. Arifin, M. S. Ahmed, T. K. Ghosh, J. Zhuang, and J. haw Yeh, "A survey on the application of generative adversarial networks in cybersecurity: Prospective, direction and open research scopes," *ArXiv*, vol. abs/2407.08839, 2024.

[25] C. Park, J. Lee, Y. Kim, J.-G. Park, H. Kim, and D. Hong, "An enhanced ai-based network intrusion detection system using generative adversarial networks," *IEEE Internet of Things Journal*, vol. 10, pp. 2330–2345, 2023.

[26] V. S. Rao, R. Balakrishna, Y. El-Ebiary, P. Thapar, K. Saravanan, and S. R. Godla, "Ai driven anomaly detection in network traffic using hybrid cnn-gan," *Journal of Advances in Information Technology*, 2024.

[27] G. Gonz'alez, P. Casas, A. Fern'andez, and G. G'omez, "On the usage of generative models for network anomaly detection in multivariate time-series," *ACM SIGMETRICS Performance Evaluation Review*, vol. 48, pp. 49 – 52, 2020.

[28] M. Ferrag, M. Debbah, and M. Al-Hawawreh, "Generative ai for cyber threat-hunting in 6g-enabled iot networks," *2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing Workshops (CCGridW)*, pp. 16–25, 2023.

[29] S. Shen, C. Yu, K. Zhang, J. Ni, and S. Ci, "Adaptive and dynamic security in ai-empowered 6g: From an energy efficiency perspective," *IEEE Communications Standards Magazine*, vol. 5, pp. 80–88, 2021.

[30] N. Vemuri, N. Thaneeru, and V. M. Tatikonda, "Adaptive generative ai for dynamic cybersecurity threat detection in enterprises," *International Journal of Science and Research Archive*, 2024.

[31] M. Ferrag, O. Friha, B. Kantarci, N. Tihanyi, L. C. Cordeiro, M. Debbah, D. Hamouda, M. Al-Hawawreh, and K. Choo, "Edge learning for 6g-enabled internet of things: A comprehensive survey of vulnerabilities, datasets, and defenses," *IEEE Communications Surveys & Tutorials*, vol. 25, pp. 2654–2713, 2023.

[32] S. Das, "Fgan: Federated generative adversarial networks for anomaly detection in network traffic," *ArXiv*, vol. abs/2203.11106, 2022.

[33] S. Li, Y. Liu, J. Li, and X. Lin, "Trustworthy ai-generative content in intelligent 6g network: Adversarial, privacy, and fairness," in *arXiv.org*, 2024.

[34] G. Feretzakis, K. Papaspyridis, A. Gkoulalas-Divanis, and V. Verykios, "Privacy-preserving techniques in generative ai and large language models: A narrative review," *Information*, 2024.

[35] H. Zhou, C. Hu, D. Yuan, Y. Yuan, D. Wu, X. Liu, Z. Han, and C. Zhang, "Generative ai as a service in 6g edge-cloud: Generation task offloading by in-context learning," *IEEE Wireless Communications Letters*, vol. 14, pp. 711–715, 2024.

[36] Q. Feng, V. Shah, R. Gadde, P. Perona, and A. Martinez, "Near perfect gan inversion," *ArXiv*, vol. abs/2202.11833, 2022.

[37] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pp. 1175–1191, ACM, 2017.

[38] M. Abadi, A. Chu, I. J. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pp. 308–318, ACM, 2016.

[39] Z. Tao, W. Xu, Y. Huang, X. Wang, and X. You, "Wireless network digital twin for 6g: Generative ai as a key enabler," *IEEE Wireless Communications*, vol. 31, pp. 24–31, 2023.

[40] K. Muhammad, T. David, G. Nassisid, and T. Farus, "Integrating generative ai with network digital twins for enhanced network operations," *ArXiv*, vol. abs/2406.17112, 2024.

[41] J. Wen, J. Kang, D. Niyato, Y. Zhang, J. Wang, B. Sikdar, and P. Zhang, "Generative ai for data augmentation in wireless networks: Analysis, applications, and case study," *ArXiv*, vol. abs/2411.08341, 2024.

[42] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "Ciciot2023: A real-time dataset and benchmark for large-scale attacks in iot environment," *Sensors*, vol. 23, no. 13, p. 5941, 2023.