IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 29 March 2023, accepted 22 May 2023, date of publication 2 June 2023, date of current version 16 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3282182

APPLIED RESEARCH

Reinforcement Learning Environment for Cyber-Resilient Power Distribution System

ABHIJEET SAHU, (Member, IEEE), VENKATESH VENKATRAMAN, AND RICHARD MACWAN, (Member, IEEE)
National Renewable Energy Laboratory, Boulder, CO 80305, USA

Corresponding author: Abhijeet Sahu (asahu@nrel.gov)

This work was supported in part by the National Renewable Energy Laboratory (NREL) operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract DE-AC36-08GO28308; and in part by the Laboratory Directed Research and Development (LDRD) Program, NREL.

ABSTRACT Recently, numerous data-driven approaches to control an electric grid using machine learning techniques have been investigated. Reinforcement learning (RL)-based techniques provide a credible alternative to conventional, optimization-based solvers especially when there is uncertainty in the environment, such as renewable generation or cyber system performance. Efficiently training an agent, however, requires numerous interactions with an environment to learn the best policies. There are numerous RL environments for power systems, and, similarly, there are environments for communication systems. Most cyber system simulators are based in a UNIX environment, while the power system simulators are based in the Windows operating system. Hence the generation of a cyber-physical, mixed-domain RL environment has been challenging. Existing co-simulation methods are efficient, but are resource and time intensive to generate large-scale data sets for training RL agents. Hence, this work focuses on the development and validation of a mixed-domain RL environment using OpenDSS for the power system and leveraging a discrete event simulator Python package, SimPy for the cyber system, which is operating system agnostic. Further, we present the results of co-simulation and training RL agents for a cyber-physical network reconfiguration and Volt-Var control problem in a power distribution feeder.

INDEX TERMS Reinforcement learning, OpenDSS, SimPy, OpenAI gym, network reconfiguration, re-routing.

I. INTRODUCTION

A smart electric grid being one of the complex, dynamic and large scale network of electrical and communication devices, its control can be transformed into a sequential-decision making problem. Such problems can be solved using traditional approaches such as convex optimization, stochastic programming or heuristic methods such as genetic algorithms, ant colony optimization etc. These approaches have challenges solving control problems with uncertainty either caused due to higher penetration of renewable power or cyber intrusions. Reinforcement Learning (RL), a branch of machine learning, have been an alternative approach, a data-driven approach to solve challenging sequential decision making problem.

The associate editor coordinating the review of this manuscript and approving it for publication was Bijoy Chand Chatterjee.

Recently, data-driven control methods have been applied for decision support and control in a variety of power systems applications, such as voltage control [1]; frequency control [2]; energy management systems, such as optimal power flow [3]; electric vehicle charging scheduling [4]; battery management [5]; and residential load control [6]. Control in power systems primarily target satisfying load flow equations to meet demand at minimal cost, to keep the system operating frequency constant, and to maintain the bus voltage within limits while ensuring that the power system components are not overloaded. Due to the increased use of advanced communication infrastructures for monitoring and control of power systems [7], the scope of cyber intrusions to enforce malfunction controller logic has proliferated. Hence, the ability to dynamically reconfigure networks will contribute to power system resilience to cyber threats. The

use of software-defined networking (SDN) has been quite prevalent recently in the information technology (IT) space, and numerous researchers have developed SDN-based solutions to mitigate threats in the operational technology (OT) domain [8]. The programmable SDN controller can be trained as an RL agent for determining the optimal configuration considering cyber-physical states from both the IT and OT space.

Although advanced distribution management systems (ADMS) and distributed energy resource management systems (DERMS) integrate software solutions for outage management as well as grid optimization and aggregated distributed energy resource (DER) operations, respectively, the underlying communication systems are monitored and controlled using dedicated network monitoring and intrusion detection applications, such as Security Information and Event Management (SIEM) and Snort. If these power distribution management solutions are upgraded to facilitate cyber-physical situational awareness, i.e. simultaneous visibility to cyber and physical sensor data, the controls can be made more threat resilient. The fusion of both cyber and physical states/information for the control can be accelerated by a model-free, data-driven approach considering that modeling this complex interaction is often challenging. Overcoming the challenges in controls through a data-driven approach necessitates generation of large-scale data sets with varying scenarios. An agent's training can be effective if it interacts with an environment mimicking an actual system. Though numerous works have focused on developing environments using network and power distribution simulators [9], there is a need for an RL environment with co-simulation. This work addresses the need for a co-simulation environment that can assist in faster data generation under varying threats and contingencies. The goal of this environment is to enhance the distribution feeder system to be both cyber and physical resilient by training the reconfiguration and voltage control agents alongwith the rerouting agents, for restoring the communication and feeder network respectively, with minimum steps. The outcomes from the trained agents using this environment, are the optimal policies learned to control the electrical and cyber components, during cyber-induced power system contingencies, which is collectively known as *cyber-physical defense* against those threats.

The major contributions of this paper are to:

1) Develop a discrete event simulation-based cyber RL environment for training agents for network reconfiguration to address network congestion and cyber threats.
2) Develop an OpenDSS-based RL environment for distribution grid reconfiguration using sectionalizing switches under various system faults.
3) Merge both SimPy and the OpenDSS-based environment for cyber-physical defense against cyber threats and feeder contingencies.
4) Validate the environment for different sizes of power and cyber systems.
5) Train well-known RL agents using the environments with the Markov decision process (MDP) model of rerouting and network reconfiguration in the respective environments.

The paper is structured as follows. Section II provides a literature review of various cyber, power system, and cyber-physical RL environments. Section III introduces the proposed the SimPy-based cyber simulator with its development of the Gym-based environment and the amalgamation with the OpenDSS Gym environment. In Section IV, the validation experiments are presented for the novel simulator, along with the evaluation of the environments with some well-known RL agents under various threat scenarios. Finally, in Section V, the possible extension of the current work is discussed, and the paper concludes in Section VI.

II. BACKGROUND

Before delving into the design of the RL environment, a few basics and prior works are illustrated in this section. RL is a subbranch of machine learning whose objective is to train agents for control actions to maximize the cumulative reward. It was initially introduced in the areas of digital gaming, followed by robotics. Before using RL to train an agent, one needs to define the problem as an MDP. An MDP is a discrete-time stochastic process used to describe the agent and the environment interactions. It is defined by a tuple of five components: states (S); action (A); state transition model, $P(s_{t+1}|s_t, a_t)$, which describes the transition of the environment state when the agent performs an action, a, in a current state, s_t; reward model $R(s_{t+1}|s_t, a_t)$, which describes the actual reward value that the agent receives from the environment after the execution is performed; and the discount factor, γ, which controls the future rewards. This work focuses on model-free RL, where the P is unknown in the MDP.

A. OpenAI GYM

OpenAI Gym [10] is an open-source Python library for developing and comparing RL algorithms by providing an application programming interface (API) to communicate between learning algorithms and RL environments as well as a standard set of environments compliant with that API. It primarily consists of definitions of the observation, action space, `reset()`, `step()`, and `render()` function, where in the `step()`, an action is executed and information is extracted from the environment. The setup collects the current state of the environment by calling the following callback functions: a) `GetObservation()`—collect the values of the observed variables and/or the parameters in the simulation; b) `GetReward()`—get the reward achieved during the last step; c) `GetDone()`—check a predefined end-of-episode condition; and d) `GetInfo()`—(optional) get any extra information associated with the current environment state, such as which contingency or, specifically, which line faults are considered in the current episode.

B. CYBER RL ENVIRONMENTS

Currently, there are numerous RL environments developed for cyber emulation targeting resource allocation, Transmission Control Protocol (TCP) congestion control, or addressing security. CyberBattleSim [11] is an artificial intelligence (AI)-based security solution from Microsoft to solve security challenges. It provides an environment for simulating intruders' lateral movements by sequentially exploiting vulnerabilities of an enterprise network to reach a targeted goal using high-level abstraction of computer networks and cybersecurity concepts. While the environment captures the movement of the attacker between the states of the system, network performance parameters in between the transition is not captured, as they are abstracted.

The authors of [12] developed an action recommendation engine based on RL for self-healing operations in the network operation center built using Graphical NS-3 (GNS-3). The authors of [13] proposed a deep RL agent for SDN-based rerouting to minimize network delays using the OMNeT++ discrete event simulator (DES). The authors of [14] developed an OpenAI Gym-based RL environment for interacting with the NS-3 simulator using Protocol Buffers (protobuf) and developed an MDP model for TCP congestion control. The work in [15] models an RL environment for a software-defined wide-area network (WAN) using the Mininet, where an agent switches between WAN links to maximize the bandwidth utilization, and where links are switched between the internet cloud and Multiprotocol Label Switching (MPLS) links, with the goal to mimimize the usage of MPLS links by using the resources from the cloud.

C. POWER SYSTEM RL ENVIRONMENT

PyMGRID [16] provides a platform for generating synthetic microgrids and model them as MDPs for providing both primary and tertiary control. Still the models are not operating with a well-known simulators like OpenDSS and cannot support interfacing other cyber emulation or simulation environment. L2RPN [17] is an Open-AI Gym compatible RL environment created for the IEEE 118 case for network reconfiguration as a competition for different participants world-wide to train agents. They tested 200 $N-1$ and 40000 $N-2$ contingencies in the system. Gym-SolarPVDER [18] is a RL environment where the dynamics of the DER are modelled using dynamic phasors. Deep RL (DRL) have been considered in other physical domains apart from power systems, such as predictive aircraft maintenance where the net maintenance cost for maintenance of aircraft turbofan engines was reduced using DRL [19].

D. CYBER-PHYSICAL RL ENVIRONMENTS

Although there are numerous pure cyber or physical RL environments, there are limited cyber-physical power distribution RL environments. The review paper on RL methods for cyber-physical systems illustrates some co-design framework for network computing and control [20], but they are not pertaining specifically to power system controls. For power system domain, a cyber-transmission grid RL environment, which is the fusion of PowerWorld with NS-3 [21], was designed for guaranteed control through mitigating TCP congestion evaluated for a Western Electricity Coordinating Council (WSCC) transmission grid. Still, both environments operated separately in this work, as the episodic results obtained in the NS-3 simulation were used as input to the PowerWorld environment offline. Similarly, a pure simulation-based, cyber-physical RL environment was developed in [22], but this work did not leverage the actual simulator in the back end. Moreover, it tested the MDP model using the naive value and policy iteration technique.

E. CYBER-PHYSICAL THREAT MODEL

Most of the prior works considers a threat model confined to either cyber or physical space. For instance a physical layer spoofing detection using received signal strength indicators using RL is proposed for wireless networks [23]. In [24], an RL agent is trained to respond against white-box and black-box attacks as a use-case of automative defense in the SDN. Modification of GOOSE, SMV and MMS to malfunction relays is considered as the threat model in [25], but the model doesnt consider what steps an attacker takes to modify such protocols.

III. A NOVEL CYBER-PHYSICAL RL ENVIRONMENT

This section presents the proposed cyber-physical RL environment. The various components of the proposed environment are detailed, and then their interconnection is described.

A. SimPy CYBER SIMULATOR

The cyber simulation environment is developed using the SimPy Python package, an open-source, process-based, DES framework. SimPy is considered in modeling the neighborhood-area network of advanced metering infrastructure systems [26]. Based on this DES framework, packet generators are modeled as the data concentrator (DC), the packet sink as the data aggregator (DA), the forwarding devices as the routers, and the communication channels as the links connecting the nodes. The packets are generated from the DC based on the number of smart meters that communicate to the DC. As a threat model, background traffic is generated to create congestion in the channel and to increase the queueing delay in the routers, which impacts the channel bandwidth and the overall latency. The state is modeled as the packet drop rate in the forwarding device and the available bandwidth in the channels/edges, although the action spaces are related to the routing policy updates in the layer 3 (L3) devices. The IEEE 123-bus distribution case is segregated into 7 zones. Within each zone, a DC receives data from the smart meters and sends them to the distribution system operator (DSO) using a WAN with the mesh topology.

1) ROUTER MODEL

A router is a networking device used for forwarding data packets between two networks. It functions in the network layer of the TCP/IP stack. Conventionally, the router software has two functional processing units: the a) control plane, and the b) forwarding plane. In the control plane, the router maintains a routing table that maintains which path would be used to forward a packet and along which interface. This routing table is statically configured or dynamically updated using dynamic routing protocols, such as open shortest path first (OSPF). The time taken to forward packets depends on the processing time of parsing the packet and searching the next hop information from the routing table. Hence, the service rate, R_{SR}, as well as the queue limit for every router, R_{QL} is modeled. The queue limit indicates the amount of bytes the router can forward at a given instance. In the forwarding plane, the router simply forwards the packet to the desired interface based on the routing rule. In the MDP model, modeling the router drop rate as a state plays a crucial role in enforcing new routes. The router packet drop rate is defined as the ratio of the number of dropped packets to the received packets.

2) CHANNEL MODEL

The channel models the latency and bandwidth between each node in the network model. Based on the traffic, the channels update their utilization rates and compute the available channel capacity with the update frequency of U_f in the SimPy DES. The utilization rate, U_r is defined as,

$$U_r = \frac{Total\ Bytes\ in\ the\ channel}{U_f\ *\ Ch_{BW}} \tag{1}$$

The available channel capacity, Ch_{avail}, is given by

$$Ch_{BW}(1 - U_r) \tag{2}$$

A packet injected into the channel is dropped, if the *Total Bytes in channel* crosses the Ch_{avail} limit. For the evaluation of the environment, the experiments are performed under varying Ch_{BW}.

3) DATA CONCENTRATOR MODEL

The DC acts as a data collector and forwarder for all the smart meters in the zones. When a fault occurs in the distribution feeder, the relay captures it and forwards it to the DC. The DC within the zone forwards the state to the DSO, which acts as the DA. In the simulation model, it is assumed that all the sensor data are accumulated at the DC; hence, the payload size of the packet sent to the DA depends on the number of components, i.e., transmission lines, buses, switches, etc., within a zone. Conventionally, the DC communicates with sensors using power line carriers, and they communicate to the DSO through the WAN.

4) DATA AGGREGATOR MODEL

Packets are received from the DC from different zones in the system. This node acts as the DSO/DA, collecting data from each DC for control purposes. In the MDP model, the goal is the state when the DA accepts at least N_g traffic within a time frame from the DCs of each zone.

5) THREAT MODEL

For the threat model, denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks are considered where an attacker injects unwanted traffic to exhaust the router as well as the communication channel, causing an increase in latency and the packet drop rate of the critical traffic. In the experiments, for every episode, variability is introduced through compromising different sets of core routers. The location would affect the neighboring channels and routers, whereas the attack intensity would affect the available channel bandwidth and the router queue limit.

In this work, the threat model spans across both IT and OT networks in the system. The scenarios are presented in the manner in which while a physical contingency have occurred and due to the cyber threat, what should be an ideal action to be taken to reach the goal of performing network reconfiguration in less number of steps. Because of the cyber-threat, pure physical actions may not restore the contingency due to loss of connectivity due to threat.

6) REROUTING MDP MODEL

1) **State Space** The system states are the packet drop rates at every router and the channel utilization rate, U_r, at every channel.
2) **Action Space** The actions within the discrete action MDP depend on the action at every router to select the highest-priority nearest hop among all the neighbors. The action model is *MultiDiscrete* in nature, where the first discrete variable selects the router, and the second discrete variable selects one from among all the neighbors.
3) **Reward Model** Currently, the reward is defined as the number of packets successfully received at the DSO. Other factors include the average latency of the packet from the source to the destination. The latency is the combination of propagation, queueing, and transmission delays. The propagation delay is kept fixed, whereas the queueing and transmission delays are affected in the MDP model, based on the channel bandwidth and the router queue size limit.
4) **Goal State**: When at least N_g packets are successfully received at the DA from each DC.

B. OpenDSS SIMULATOR

OpenDSS is an open-source electric power distribution system simulator developed for grid modernization with the integration of DERs. This work focuses on developing an RL environment leveraging the *OpenDSSDirect.py* to interface with the OpenDSS modeled distribution feeder for executing the contingencies and restoring them using network reconfiguration using a sectionalizing switch, an automated switching device that is intended to isolate faults and restore loads.

The optimal network reconfiguration is modeled as an MDP, where the variability is introduced at the beginning of each episode through random selection of different load profiles along with a contingency.

1) NETWORK RECONFIGURATION MDP MODEL

1) **State Space** The system states are the critical load bus per-unit voltage magnitude, V. In the IEEE 123-bus system, based on a certain set of line faults, one or two critical load buses are picked from each of the seven zones.
2) **Action Space** The actions with the discrete action MDP depend on the action of either opening or closing one of the available sectionalizing switches at a given step of an episode; hence, the dimension of the action space is N_{sw}, where N_{sw} is the number of sectionalizing switches.
3) **Reward Model** The reward model is defined based on the number of critical load buses yet to be restored, N_{res}.

$$R(s) = \begin{cases} 20 & \text{if } N_{res} = 0 \\ -1 * N_{res} & \text{if } elsewhere \end{cases}$$

4) **Goal State** When all the critical loads in the distribution grid are within the required voltage range.
5) **Contingencies** $N-1, N-2, N-3$ line outages, and $N-1$ DER outage.

2) VOLT-VAR CONTROL MDP MODEL

Due to distribution losses, voltage drops accross transmission lines, possibly causes voltage violations, hence Volt-Var optimization is adopted. Primarily, voltage regulators, capacitor banks, batteries, etc are controlled in this optimization under various constraints. In this work, the *PowerGym [27]* environment is considered for the Volt-Var control in the power distribution systems. The objective of the problem is to minimize voltage violations, control loss, and power loss, while the physical network constraints are maintained by running the power flow in the OpenDSS simulator. The device constraints are usually integer constraints formulated in the action space. In RL, conventionally sequential decision making problems of two kinds are solved: a) Episodic b) Finite Horizon i.e. fixed length episode. The network reconfiguration problem introduced previously have a goal state where an episode terminates when the goal is achieved. While in the Volt-Var MDP model a finite horizon problem is solved where the control elements need to operate with varying load profile for a span of a day or 24 Hrs where the control actions are executed every one hour making finite horizon steps to 24. The MDP model for the Volt-Var control problem is given by:

1) **State Space** The system states are the bus voltages, capacitor status, voltage regulator tap number, state of charge and discharge power of the battery.
2) **Action Space** The actions are a mix of discrete and continuous control: **a)** Capacitor Bank: On/Off, **b)** Regulator Tap Number: Discrete $0, .., N_{reg\_act} - 1$, where N_{reg_act} is the number of taps of a regulator, **c)** Battery Discharge: Discrete $0, .., N_{bat\_act} - 1$, where N_{bat_act} is the number of discrete battery's discharge power. **d)** Battery Discharge: Continuous $[-1, 1]$.
3) **Reward Model** Since, the objective of the problem is to minimize voltage violation, power loss and control loss, the reward model is given by:

$$R(s) = -f_{volt}(s) - f_{ctrl}(s) - f_{power}(s)$$

The definition of the functions f_{volt}, f_{ctrl}, and f_{power} are given in details in the PowerGym [27].

C. INTERCONNECTION

The cyber-physical co-simulation within the environment is performed through passing `Queue` as a shared variables across multiple threads. The events across both the simulator and the triggers generated from the events are carried out depending on the dynamic updates in the shared variables. There are two shared variables created for this purpose: *Phy-Cyb Queue* and *Cyb-Phy Queue*.

1) *Phy-Cyb Queue:* This queue is used to schedule event in the cyber environment based on the control command on re-configuring is given by the DSO in the physical environment. While a control action is selected in the OpenDSS environment, the switch information along with the location is put inside the queue, which is processed by the cyber environment to generate a control command in the respective zone.
2) *Phy-Cyb Queue:* This queue is used to schedule event in the cyber environment based on the control command on re-configuring is given by the DSO in the physical environment. While a control action is selected in the OpenDSS environment, the switch information along with the location is put inside the queue, which is processed by the cyber environment to generate a control command in the respective zone.

Fig. 1 shows the sequential diagram of the steps that takes place within each episode of both the cyber and physical environment. The interconnection in blue indicates the communication between the SimPy and OpenDSS simulators. The interconnects indicates:

1) Indicates the passage of physical-side information to cyber network for determining the packet size.
2) Indicates the passage of cyber and physical contingency to each others environment. Currently, a physical fault adds an event in cyber emulator to generate a fault information to send to the aggregator.
3) Indicates the merge of the cyber and physical state information to feed to the RL algorithm or the Agent.
4) Based on the policy, implement the action by segregating respective action of routing policy and control of sectionalizing switch.

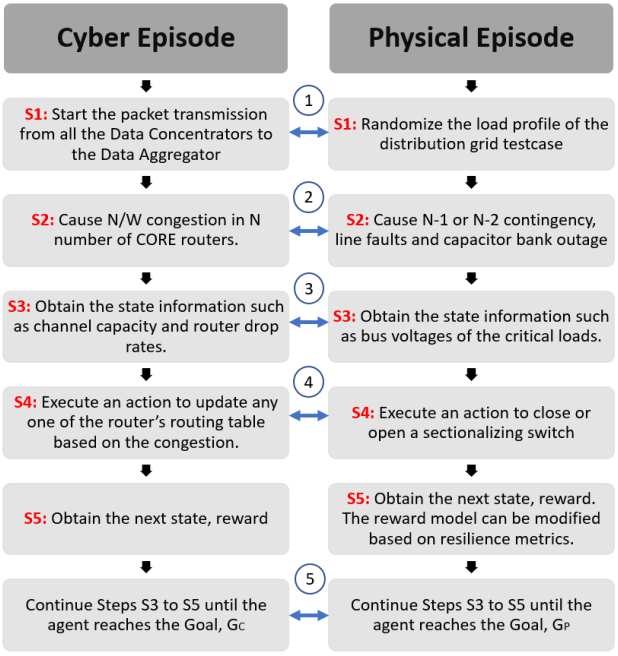

FIGURE 1. Steps in an episode created in the cyber-physical RL environment.

5) Evaluating the goal G_P and G_C for terminating the episode when both goals are reached.

A cyber-physical episode with physical contingency and cyber threat can be visualized as shown in Fig. 2. At time t_0 the OpenDSS environment starts with a random load profile selected for all the load buses. Depending on the zones where the aggregate load injection value is changed, the *DC* of that zone generates a packet to be send to the *DA* at time t_1. This information from OpenDSS to SimPy is shared with the use of *Phy-Cyb Queue*. At t_2, a physical contingency is caused which can be a single or multi-line outage. A packet is generated corresponding to this outage from the *DC* of the zone to which the contingency belongs at t_3. A cyber-threat of exhausting a single or multiple routers through DoS attack is performed at t_4. Further this threat information from SimPy is passed to the OpenDSS through *Cyb-Phy Queue* at t_5, which flags the information regarding which zone is currently secured to be controlled. Then at t_6, both cyber and physical action are randomly taken and at t_7 the cyber and physical states are obtained from the simulators and merged. The steps taken at t_6 and t_7 are carried until the goals G_P and G_C are reached.

IV. VALIDATION EXPERIMENTS

The communication network of the IEEE 123 bus case is based on the geographical segregation of the feeder into seven zones. Within each zones there is a data concentrator that forwards the information to the DSO, or data aggregator.

A. SimPy-BASED CYBER EXPERIMENTS

The experiments in this section focuses on the impact of the cyber network parameters such as channel bandwidth, router queue size, number of core routers compromised, etc on the average episode length, packet drop rates, latency etc. All the results shown are the average of 1000 episodes run with a unique scenario. The goal of each episode is achieved, when atleast N_g packets successfully received at the data aggregator from each data concentrator.

1) EFFECT OF CHANNEL BANDWIDTH ON EPISODE TERMINATION RATE AS WELL AS THE AVERAGE EPISODE LENGTH

The states considered in the MDP model for the re-routing plays a major role in determining optimal action. The communication channel gets impacted by threats such as DoS or DDoS and also due to network congestion caused by benign background traffic such as firmware updates. Hence, in the RL environment the channel is modeled and the impact of varying channel bandwidth on the a) average episode length in reaching the goal, b) success rates in reaching the goal within a certain threshold of steps, as well as, the c) latency are evaluated. The overall latency depends on the transmission, propagation, queueing, and processing delay. The processing and the queuing delay is effected at the routers and the transceiver nodes. While the propagation delay is based on the distance and media of communication. While the transmission delay depends on the packet size and the channel bandwidth. Fig. 3 shows how the latency reduces from 8.7 s to 8.45 s with the increase in channel bandwidth. The channel propagation delay being set to 1 s and the router's processing rate set at 400 bits/sec, considering an average of 4 to 5 hops from source to destination with the average packet size of 25 bytes, the net propagation delay in the channels amounts to 4-5 s, while router's processing delay around 2.0-2.5 s, the rest being transmission delay, the overall latency is more than 8 s. Further, improved bandwidth assist in reaching goal in less episodes increasing the success rate.

2) EFFECT OF THE THREAT INTENSITY

In a DoS attack usually a single server or resource is targeted at a time, while more stealthier attacks such as DDoS target multiple resources. Hence, the threat intensity is varied by targeting different numbers of core routers in the environment to study its impact on the episodic length. From Fig. 4, under no threat scenario, increased channel bandwidth reduces episode length, but increased threat targets resulted in longer episodes. Though there is reduction of episode length from 47 to 36 in the case of two routers compromised, still it is not substantial. Hence, it is essential to train an intelligent agent to reach goal by performing re-routing correctly. Moreover, with higher threats such as four routers being targeted, the episode length rose towards almost 90 to 95 episodes.

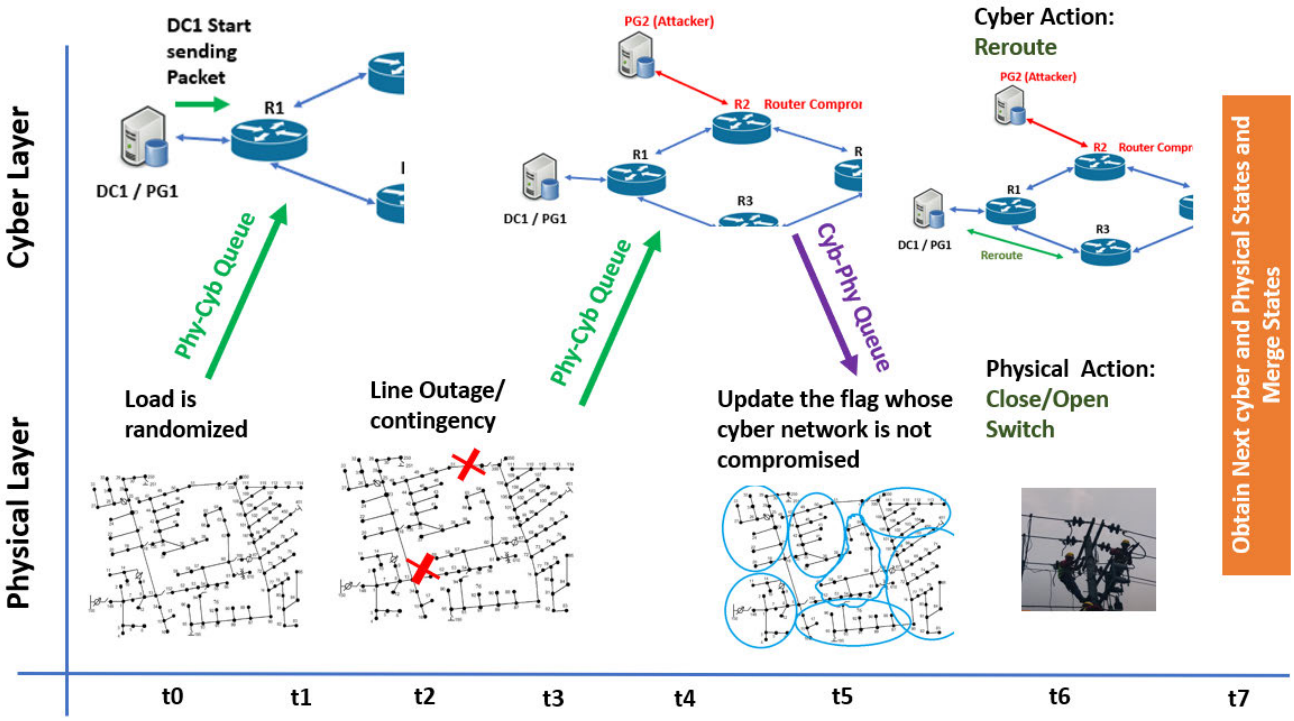

FIGURE 2. This is a sequence diagram of an episode in the co-simulation RL framework.

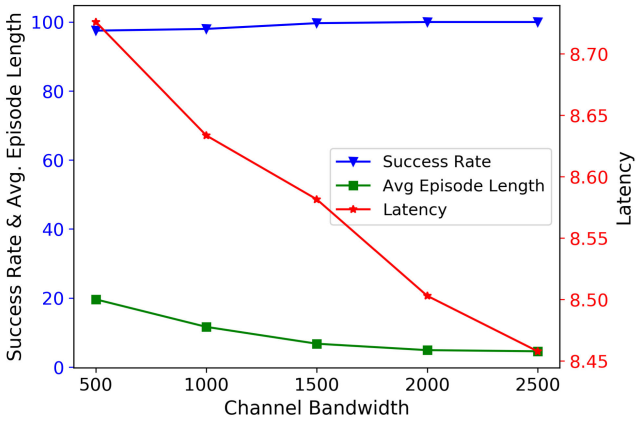

FIGURE 3. Impact of the channel bandwidth on the latency, episode length, success rates.

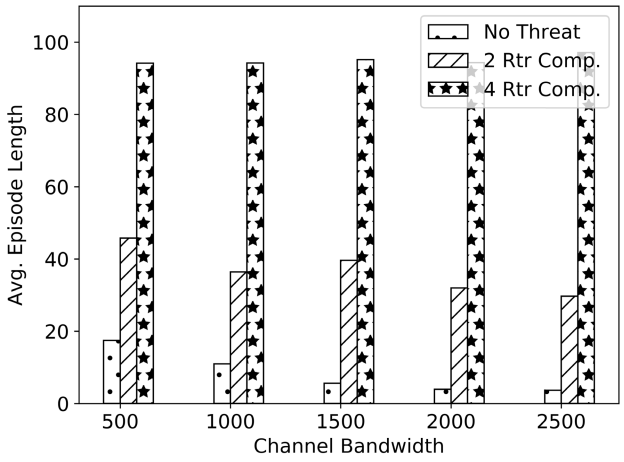

FIGURE 4. Effect of varying threat intensity through number of routers targeted.

3) EFFECT OF ROUTER's QUEUE LIMIT

Every router have got a processing time, hence it creates a queue of the incoming packets. There is a limit on the queue size of the forwarding device, beyond which if a packet arrives it drops the packet. Hence, the DES simulator is validated, by varying the queue limit, R_{QL}. It can be observed from Fig. 5 that the packet drop rate reduces with increase in the queue limit, but this drop is not prominent under threat scenarios.

4) IMPACT OF RE-ROUTING

To evaluate the impact of re-routing under cyber-threat first a miniature cyber network is considered as shown in Fig. 6, where the benign user is transmitting packet from $PG1$ to PS, while the attacker $PG2$ exhaust the resources in the router $R2$. The goal of the re-routing is to make sure the edge router $R1$ re-routes the packets towards $R3$. To validate the effectiveness of re-routing, the selection probability of $R3$ over $R2$ is considered for varying Router queue limits. It can be observed from Fig. 7 that with the increase in selection of $R3$, the average episodic length in the MDP reduces. Moreover increasing the queue limit further improves the performance. When an RL agent is trained, it would prefer to select $R3$ over $R2$, when it moves from exploration to exploitation phase.

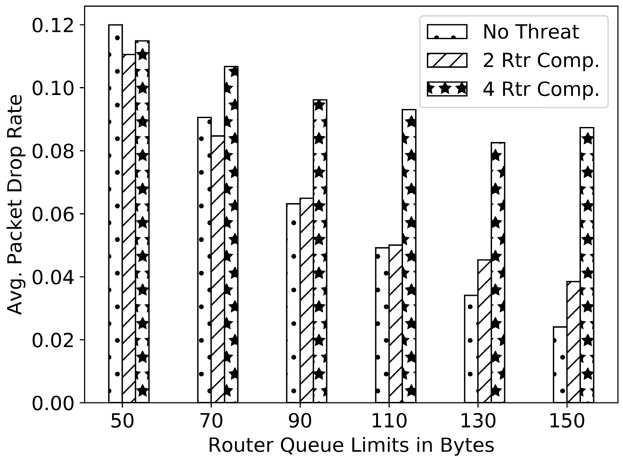

FIGURE 5. Impact of the router queue size on the packet drop rate.

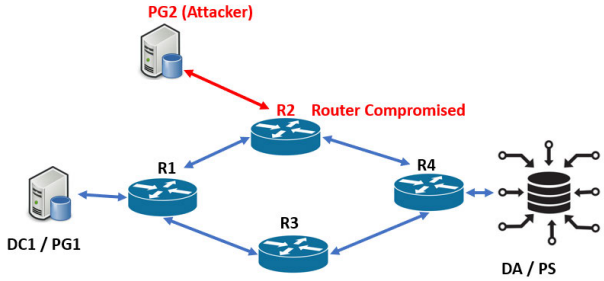

FIGURE 6. A simple network where router *R*2 is compromised.

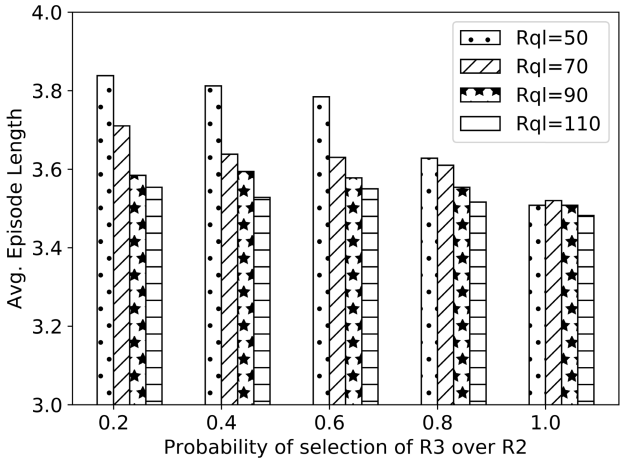

FIGURE 7. Effect of selection of router *R*3 over *R*2, when *R*2 is compromised.

B. COMPARISON WITH MININET

Mininet is a widely known tool that enable creation of a virtual network and control through Software-Defined Network. The communication network developed for the IEEE 123 system is emulated and compared with the SimPy-based model. Static routes with priorities have been configured in the core routers. The 8 edge routers are configured to connect with the 7 DCs and 1 DA. For the experiment, the

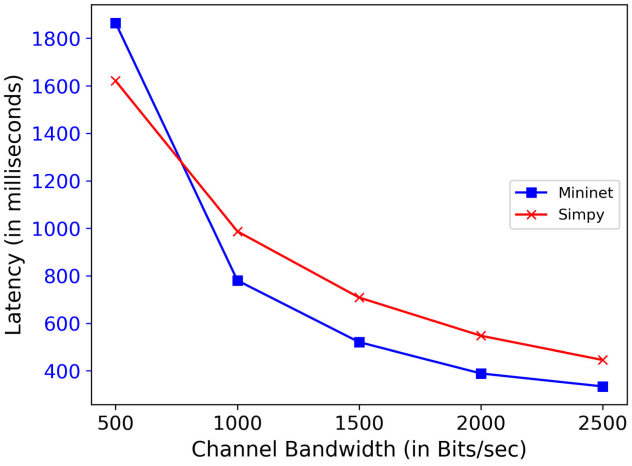

FIGURE 8. Comparison of latency between the data concentrator and the aggregator in the SimPy and Mininet simulators.

DA would simultaneously communicate with the DCs using the *nping* package and obtain the statistics of round trip time. The net latency between the DA and DC has been evaluated for varying channel conditions. In comparison to Fig. 3, the latency for the SimPy-based model reduced from 8 s to 1.6 s as observed from Fig. 8 since the channel propagation delay of 1 sec was reduced to 1 ms and the router processing rate increased from 400 Bits/s to 40 KBits/s. Such translation is performed, since in the Mininet network no propagation delay is introduced and also the router processing delay is negligible. The payload size of 25 bytes is considered for both the experiments.

C. OpenDSS BASED POWER DISTRIBUTION SYSTEM EXPERIMENTS

The goal in the network reconfiguration problem, for a single episode is reached, when all the critical loads in the distribution grid are restored following a contingency. The critical load buses considered in the IEEE 123 bus case includes: 37,39, 48, 50 (zone 3), 58, 59 (zone 4), 78, 88, 93, 94, 99 (zone 5), 111, 114 (zone 7). There are 6 DERs added to the base case at bus 35, 48, 64, 78, 95 and 108.

1) EFFECT OF TYPE OF CONTINGENCIES ON EPISODE LENGTH

Depending on the type of contingencies number of episodes for the restoration of voltages of the critical load buses would vary. For instance it is hypothesized that an $N-x$ contingency, with higher x would require more actions for restoration in comparison to a lower one. For the experiments, four types of contingencies are considered. $N-3$ indicates three line-faults, $N-2$ indicates two line-faults, $N-1$ being one, while $N-1$, *DER outage* indicates one line-fault along with a set of DER outages. The selection of switching is randomly done but as an agent is trained, the action would be based on the trained policy. It can be observed from Fig. 9 with the random action selection, the $N-3$ contingency need more than

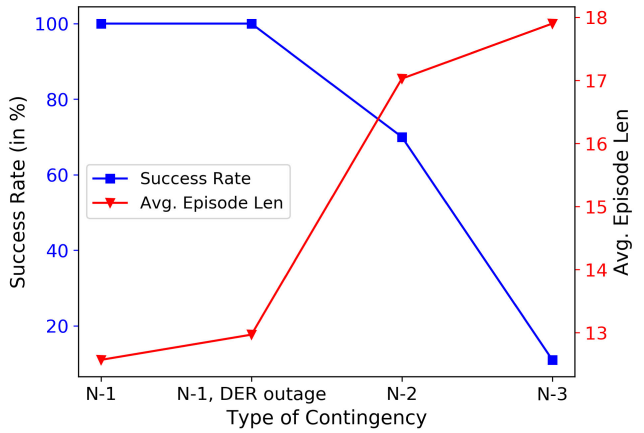

FIGURE 9. Effect of different physical contingency on the episode lengths to successfully reach the goal of restoring critical load bus.

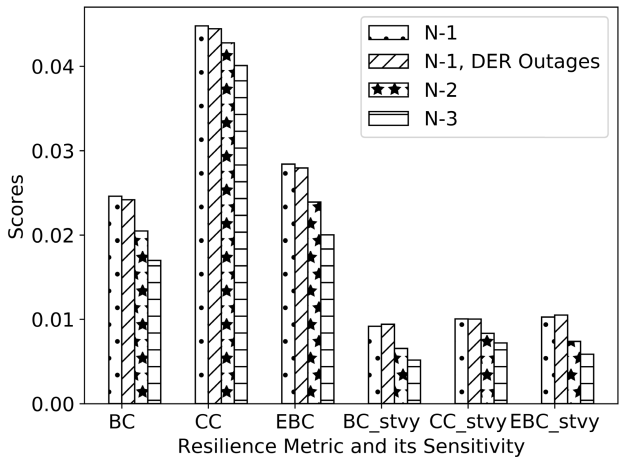

FIGURE 10. Topological Resilience Metrics Betweenness Centrality, Closeness Centrality, Edge Betweenness Centrality alongwith their sensitivity to each step in an episode.

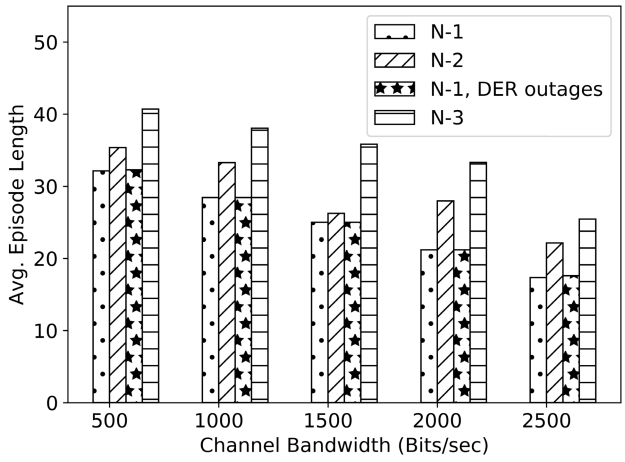

FIGURE 11. Effect of different physical contingency under varying channel bandwidth on the episodic length of the cyber-physical RL environment.

1.5 times of episodes in comparison to the $N-1$ contingency. Since restoration of critical load buses is the primary target, $N-1$, *DER* have similar behavior with $N-1$ types. If one consider, to maintain the voltage within a stringent limit, the outage of DER will effect the episodic length dramatically.

2) EFFECT ON GRAPH CENTRALITY BASED RESILIENCE METRIC

The dynamics of the resilience metric is evaluated for every switching action in the environment to quantify the effectiveness of agent's action towards reaching a more resilient topology. In Fig. 10 the first three scores are the centrality-based resilience metric Betweenness Centrality (*BC*), Closeness Centrality (*CC*), Edge Betweenness Centrality (*EBC*) and the next three is their sensitivity to every switching action. *BC* of a node is the sum of the fraction of all the pairs of shortest paths that pass through the node, while *EBC* of an edge is the sum of the fraction of all the pairs of shortest paths that pass through the edge. *CC* of a node is the inverse of the average shortest path to the node from all the reachable nodes in the network. For $N-2$ contingencies, the resilience is lower, still the positive sensitivity indicates switching action improves resilience.

D. SimPy-CUM-OpenDSS BASED MIXED DOMAIN EXPERIMENTS

The goal in the mixed-domain experiment in an episode is reached, when both critical loads in the distribution grid are restored and minimum number of packets are successfully received at the destination. Here the impact of both channel bandwidth and physical contingency on the episodic length and success rate is evaluated. Unlike the individual environment, the episodic length of the combined environment require more steps to accomplish the goal. More impactful contingencies such as $N-3$ have expected behavior of increased episodic length as evaluated in the physical-environment. Similar observation of reduced episode length under increased channel bandwidth is witnessed, as seen from Fig. 11.

Fig. 12 shows the impact of both cyber threat strength, as well as, the physical contingencies, on the episode length. Episodic length are primarily dominated by the cyber environment since the physical contingency type didn't impact the average episode length much like the attack strength. This indicates that, if a non-compromised path is found among the DA and DCs, then all the critical load could be restored, through successful command packet transmission to close the sectionalizing switch. If the re-routing is not successful to transmit the command, then although the physical RL environment execute the step, the goal is still unreached.

E. TRAINING RL AGENTS

The MDP models will be now utilized to train some well-known RL agents such as DQN, PPO, A2C, etc. While DQN is a value learning based method, Proximal Policy Optimization (PPO) [28] is a policy gradient technique while Advantageous Actor Critic (A2C) [29] use both value learning and policy gradient in the process of learning critic and

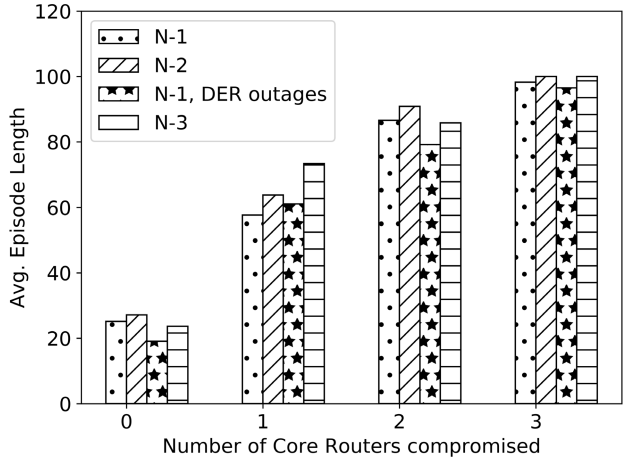

FIGURE 12. Effect of different physical contingency under varying cyber threat strength on the episodic length of the cyber-physical RL environment.

actor function respectively. The DQN method does not fare well for the scenario with large state and action space. Moreover, existing libraries such as *stable-baselines* [30] does not support *MultiDiscrete* action space for DQN.

1) RE-ROUTING

In this problem, a *MultiDiscrete* action is formulated, because an agent has to learn which router to re-route as well as based on the selected router determine the next-hop router. The action space size for the re-routing problem is of the order of $O(mn)$ where m is the number of routers and n is the degree of the router node. Hence, before training the agent for a large-scale network for the IEEE 123 case with 18 routers as shown in Fig. 15, the PPO and A2C-based agents are first trained for the small network with 4 nodes (Fig. 6) as well as, a few mid-sized networks (Fig. 13). Table 1 shows the results of the average episode length without training and with training with PPO and A2C agent for four different cyber network. Though there is reduction in average episode length with training, still for the larger network the reduction is not quantifiably high. Training each router separately using multi-agent RL framework is a viable solution, but this is currently out of scope of this paper.

The RL based method are further compared with heuristic approach. Alg. 1 presents the algorithm for selecting one of the optimal action for the re-routing method for communication under threat scenarios. An action in this MDP is defined by a tuple $< r, r_{nh} >$ where the first element represent the router, r, selected to update the route while the r_{nh} is the next hop router selected to update in the router r routing table. It can be observed from Table 1 that the RL based approach is at par with the number of steps to goal in comparison with the heuristic approach.

2) NETWORK-RECONFIGURATION

For the network-reconfiguration problem, the state space model with continuous variable representing the critical load bus's p.u. voltage had challenges in training agents. Hence, the state space is represented through the restoration status of the critical load bus, through one-hot encoding. Since the state space is discrete for this problem, DQN is considered for training under 4 different types of contingencies as introduced in the OpenDSS based experiments section. Table 2 shows the reduction in average episode length for different contingency scenarios. In most of the scenario there is atleast 40% reduction in episode lengths when compared without the DQN training.

A spanning tree based approach (STA) [31] is adopted for optimal network reconfiguration for the reference. Due to the radial structure of a distribution network it is represented as a spanning tree. The switching operations is based on adding an edge to the spanning tree to create a cycle and deleting another edge within this cycle for a transition to a new spanning tree.

Algorithm 1 Re-Routing Expert Heuristic Method

From MDP states infer compromised router set R_{comp}
Initialize the set of possible optimal policies Π
for $r \in R_{comp}$ **do**
  Extract the parent routers, Pa_r in forward path to DA.
  for $p_r \in Pa_r$ **do**
    Extract all paths to DA from p_r, that doesnt include r.
    From paths get the immediate next hop routers.
    Select next hop router with lowest channel packet drop rate, $Ch^*_{p_r}$.
    $\Pi = \Pi \bigcup (p_r, Ch^*_{p_r})$
  end for
end for
r, r_{nh} = Sample a policy from Π

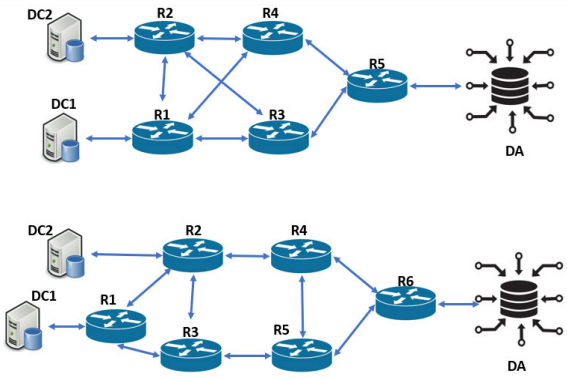

FIGURE 13. Two mid-sized communication network with 5 and 6 routers with 3.4 and 3 average node degree.

TABLE 1. Evaluation of PPO & A2C based RL agent for varying size network for re-routing.

NW	Rtrs	Chs	N_g	Epi. Len.	A2C	PPO	Heuristic
Small	4	6	10	12.63	3.02	3.028	1.8
Mid-Sized	5	10	5	13.8	7.54	5.77	5.2
Mid-Sized 2	6	11	5	11.03	6.18	6.23	6.0
IEEE-123	18	36	2	42.3	32.4	29.7	37.2

TABLE 2. Evaluation of DQN based RL agent for different contingencies type on IEEE 123 distribution system.

Contingency Type	Epi. Len. (Random)	DQN	STA
N-1	10.33	6.097	4.8
N-1, DER Outage	11.26	5.98	5.0
N-2	14.27	8.05	5.4
N-3	17.19	12.51	6.8

TABLE 3. System details with the type of controls and their quantities.

System	# Capacitor Banks	# Voltage Regulators	# Batteries
IEEE 13	2	3	1
IEEE 34	2	6	2
IEEE 123	4	7	4
8500 Nodes	10	12	10

TABLE 4. Average episodic reward.

System	Random	PPO	A2C
IEEE 13	-48	-11.45	-5.68
IEEE 34	-99	-24.78	-7.77
IEEE 123	-180	-39.98	-11.4
8500 Nodes	-6504	-123.45	-67.98

The optimal final topology along with the sequential order of switching is provided by the proposed method [31]. Based on the different line outages considered in this work, the sequence of switching is obtained and it can be observed from Table. 2 that adopting STA's approach, the episode length is almost one-third of the random agent and even better than DQN. Use of advanced policy gradient technique can better improve the performance. But since such STA methods cannot be leveraged when the physical environment is combined with cyber, RL based framework is developed.

3) VOLT-VAR CONTROL

For the Volt-Var control problem, four benchmark distribution systems are considered: a) IEEE 13 bus b) IEEE 34 bus c) IEEE 123 bus, and d) 8500 nodes cases, as presented in the *PowerGym* [27]. For the experiments, only the discrete controls from Section III-B2 are considered. The control elements of each distribution system is shown in the Table 3. Since the problem is formulated as 24 step-sized fixed horizon problem for control action for a single day i.e. 24 Hours with varying load profile, the environment is evaluated on the basis of average episodic reward with the trained *A2C*, *PPO* and a random agent. Table 4 shows the evaluation of the *A2C* and *PPO* agents considered for the four systems. As the grid size increases, the state and action space increases, but with the use of *A2C* RL agent the performance is still better in comparison to PPO. But since *PPO* prevent large gradient updates it can be preferred over *A2C* in the continuous control environments.

F. VISUALIZATION/ENVIRONMENT RENDERING

Most of the RL environment present a rendering application to visualize how a trained agent can assist in reach the goal fast. Following the similar approach, the proposed RL environment also presents visualization platform for both the SimPy and OpenDSS environments. The application is developed using the *PyQT5* python package.

To visualize the RL environment under threats and contingencies, that varies the topology and resilience metrics, a rendering application (Fig. 14) is developed that shows in real-time the execution of every steps within an episode while training the agent. The color-map in the figure indicates the p.u. voltage of the buses in the IEEE 123 system.

A visualization application is implemented for the Simpy-based Cyber RL environment (Fig. 15). For every steps in the episode, the tool visualize the channel utilization rates on the channels connected to the core and edge router.

V. DISCUSSION

The proposed CP-RL environment can assist in faster generation of data pertaining to various threat and contingency scenarios, in comparison to the emulated RL environment. For instance thousands of episodes in a SimPy environment could be run in less than 5 minutes, while an emulation platform like Mininet would take more than an hour. Since training efficiently with RL algorithms is a data-hungry process, this DES based simulation platform is proposed, which is not purely simulation or emulation based, since unlike simulators it has event handlers, and unlike emulators it doesn't run in synchronization with real-time clock. The pipeline between the environments is not comprehensive like a co-simulation platform, HELICs, but still it assist in developing use-cases such as incorporating cyber-induced threats and its impact on the distribution feeder in the OpenDSS space, and DER outage or line-faults from the OpenDSS space to the SimPy, through handling the event by generating a packet in cyber space. Since majority of the power distribution system simulator such as OpenDSS or GridLab-D are Windows operating system supportable, integration of this cyber environment can assist power system RL researcher to overcome the challenges of co-simulation and the necessity to connect with UNIX-based network simulators.

There are numerous scope of improvement in the current environment. For instance, in the rerouting problem, each action within an episode were executed at an interval of 50 sec. If this interval is reduced, the steps to reach the goal may reduce but it will result in more unstable solution were the router's policy are changed quite frequently. While if it's increased, the network restoration will be delayed as more steps would be required to reach the goal. Hence, deciding an ideal interval between steps can be an another research question.

In the MDP model, how efficiently an agent learns also depends on the states considered in the MDP. For the re-routing experiment, first the channel utilization rates was considered, which were not impactful, hence each router's packet drop rates was incorporated in the state-space model. In a realistic scenario, identifying an ideal state-space, reward and goal is a challenging problem. In future, we are envisioning to work on adaptively learning a reward depending on varying scenarios through inverse reinforcement learning.

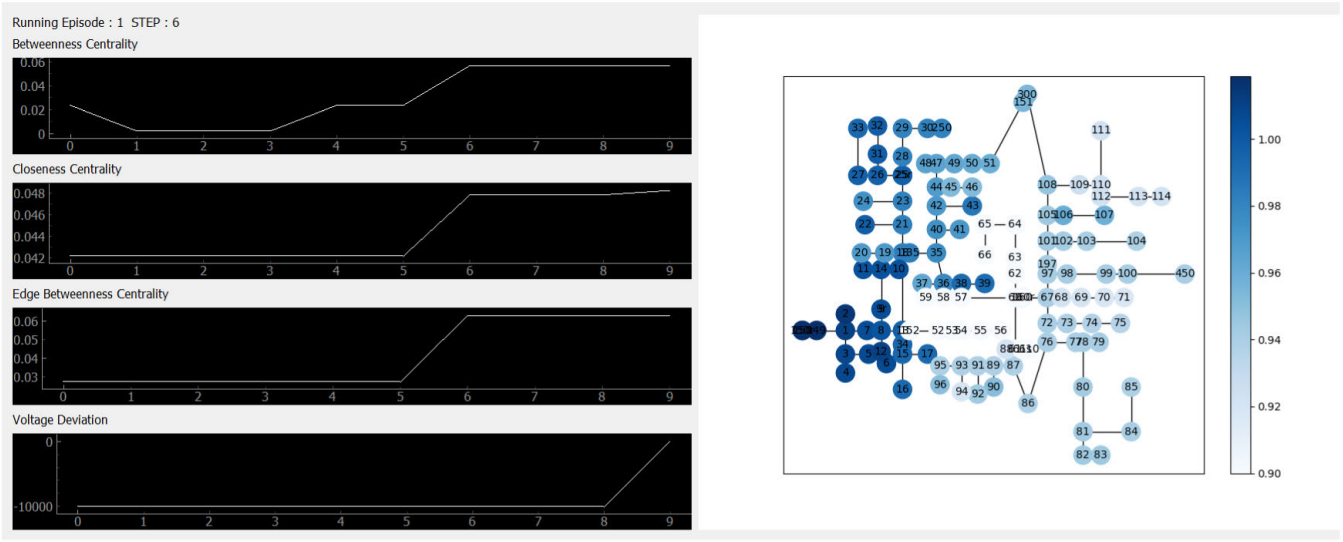

FIGURE 14. Visualization of IEEE 123 bus network and the respective resilience metric, with interaction through the RL environment.

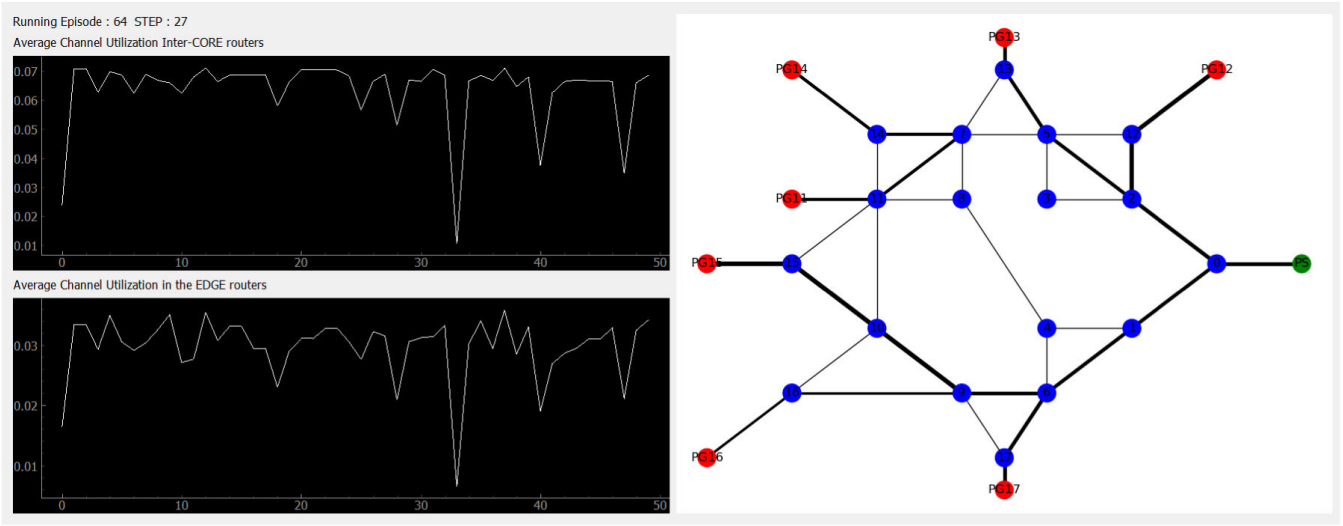

FIGURE 15. Visualization of the communication network for the IEEE 123 system.

The code base for the proposed RL framework is currently available for peer-review at the following Github repository [32].

The time taken for training an RL agent is dependent on the MDP model. An MDP model with large state-space and action-space model would require more amount of simulation data to learn an optimal policies. For instance, a network reconfiguration for a IEEE 13 bus case with one breaker in the model, can be trained faster in comparison to a IEEE 123 bus case with 8 sectionalizing switches. Similarly, training time for a cyber network with routers having large number interfaces will be more in comparison to a network constituting of routers with less interface. The exact time complexity cannot be defined, since it will depend on the RL algorithm considered. Moreover, the training time would also depend on the time taken by the simulator to execute the action and obtain the next state. The evaluation of the time complexity of training various RL algorithm is not the scope of this paper. Since after a model is trained, the accuracy of learned policy in the testing scenario is more essential.

VI. CONCLUSION

In this paper, an OpenDSS-cum-SimPy based Gym environment is presented, that simplifies the usage of reinforcement learning for solving problems in the area of cyber-physical security. This is achieved by interconnecting the OpenAI Gym with the OpenDSS and SimPy based network simulator. As the framework is generic, this work can be extended to

other power system problem such as state estimation or other threat scenario simulation. For the future, it is planned to extend environment to support multi-agent training, which can expedite the process of learning for larger test system. Utilizing this environment, both re-routing and network reconfiguration are implemented using both value function learning and policy gradient methods. This light-weight cyber environment would enable power system RL researcher to incorporate cyber threat scenarios without the hassle of environment issues, since majority of the network simulator are UNIX based.

ACKNOWLEDGMENT

The views expressed in the article do not necessarily represent the views of the DOE or the U.S. government. The U.S. government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so for U.S. government purposes.

REFERENCES

[1] L. Yin, C. Zhang, Y. Wang, F. Gao, J. Yu, and L. Cheng, "Emotional deep learning programming controller for automatic voltage control of power systems," *IEEE Access*, vol. 9, pp. 31880–31891, 2021.

[2] Z. Yan and Y. Xu, "A multi-agent deep reinforcement learning method for cooperative load frequency control of a multi-area power system," *IEEE Trans. Power Syst.*, vol. 35, no. 6, pp. 4599–4608, Nov. 2020.

[3] Y. Zhou, B. Zhang, C. Xu, T. Lan, R. Diao, D. Shi, Z. Wang, and W.-J. Lee, "A data-driven method for fast AC optimal power flow solutions via deep reinforcement learning," *J. Mod. Power Syst. Clean Energy*, vol. 8, no. 6, pp. 1128–1139, Nov. 2020.

[4] Z. Wan, H. Li, H. He, and D. Prokhorov, "Model-free real-time EV charging scheduling based on deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5246–5257, Sep. 2019.

[5] J. Cao, D. Harrold, Z. Fan, T. Morstyn, D. Healey, and K. Li, "Deep reinforcement learning-based energy storage arbitrage with accurate lithium-ion battery degradation model," *IEEE Trans. Smart Grid*, vol. 11, no. 5, pp. 4513–4521, Sep. 2020.

[6] X. Chen, Y. Li, J. Shimada, and N. Li, "Online learning and distributed control for residential demand response," *IEEE Trans. Smart Grid*, vol. 12, no. 6, pp. 4843–4853, Nov. 2021.

[7] Y. Isozaki, S. Yoshizawa, Y. Fujimoto, H. Ishii, I. Ono, T. Onoda, and Y. Hayashi, "Detection of cyber attacks against voltage control in distribution power grids with PVs," *IEEE Trans. Smart Grid*, vol. 7, no. 4, pp. 1824–1835, Jul. 2016.

[8] R. Sahay, W. Meng, D. A. S. Estay, C. D. Jensen, and M. B. Barfod, "CyberShip-IoT: A dynamic and adaptive SDN-based security policy enforcement framework for ships," *Future Gener. Comput. Syst.*, vol. 100, pp. 736–750, Nov. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X1930367X

[9] V. Venkataramanan, P. Wang, A. Srivastava, A. Hahn, and M. Govindarasu, "Interfacing techniques in testbed for cyber-physical security analysis of the electric power grid," in *Proc. Workshop Modeling Simulation Cyber-Phys. Energy Syst. (MSCPES)*, Apr. 2017, pp. 1–6.

[10] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI gym," 2016, *arXiv:1606.01540*.

[11] C. Seifert, J. Bono, J. Parikh, and W. Blum. (Feb. 2020). Cyberbattlesim. Microsoft. [Online]. Available: https://www.microsoft.com/en-us/research/project/cyberbattlesim/

[12] B. Altamimi, "Towards super intelligence-driven autonomous network operation centers," Tech. Rep., 2021.

[13] G. Stampa, M. Arias, D. Sanchez-Charles, V. Muntes-Mulero, and A. Cabellos, "A deep-reinforcement learning approach for software-defined networking routing optimization," 2017, *arXiv:1709.07080*.

[14] P. Gawłowicz and A. Zubow, "Ns-3 meets OpenAI gym: The playground for machine learning in networking research," in *Proc. 22nd Int. ACM Conf. Model., Anal. Simulation Wireless Mobile Syst.* New York, NY, USA: Association for Computing Machinery, Nov. 2019, pp. 113–120, doi: 10.1145/3345768.3355908.

[15] A. Chakravarty. (Dec. 2018). *Open-AI Gym for SD-WAN Link Selection*. [Online]. Available: https://towardsdatascience.com/open-ai-gym-for-sd-wan-link-selection-fe7dac671172

[16] G. Henri, T. Levent, A. Halev, R. Alami, and P. Cordier, "Pymgrid: An open-source Python microgrid simulator for applied artificial intelligence research," 2020, *arXiv:2011.08004*.

[17] A. Marot, B. Donnot, C. Romero, L. Veyrin-Forrer, M. Lerousseau, B. Donon, and I. Guyon, "Learning to run a power network challenge for training topology controllers," 2019, *arXiv:1912.04211*.

[18] S. J. Plathottam. (2019). *Gym-SolarPVDER-Environment: A Environment for Solar Photovoltaic Distributed Energy Resources*. Accessed: Mar. 18, 2019. [Online]. Available: https://github.com/sibyjackgrove/gym-SolarPVDER-environment

[19] J. Lee and M. Mitici, "Deep reinforcement learning for predictive aircraft maintenance using probabilistic remaining-useful-life prognostics," *Rel. Eng. Syst. Saf.*, vol. 230, Feb. 2023, Art. no. 108908. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832022005233

[20] X. Liu, H. Xu, W. Liao, and W. Yu, "Reinforcement learning for cyber-physical systems," in *Proc. IEEE Int. Conf. Ind. Internet (ICII)*, Nov. 2019, pp. 318–327.

[21] Z. Mao, A. Sahu, P. Wlazlo, Y. Liu, A. Goulart, K. Davis, and T. J. Overbye, "Mitigating TCP congestion: A coordinated cyber and physical approach," in *Proc. North Amer. Power Symp. (NAPS)*, Nov. 2021, pp. 1–6.

[22] A. Sahu, H. Huang, K. Davis, and S. Zonouz, "SCORE: A security-oriented cyber-physical optimal response engine," in *Proc. IEEE Int. Conf. Commun., Control, Comput. Technol. Smart Grids (SmartGridComm)*, Oct. 2019, pp. 1–6.

[23] L. Xiao, Y. Li, G. Han, G. Liu, and W. Zhuang, "PHY-layer spoofing detection with reinforcement learning in wireless networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 10037–10047, Dec. 2016.

[24] Y. Han, B. I. P. Rubinstein, T. Abraham, T. Alpcan, O. De Vel, S. Erfani, D. Hubczenko, C. Leckie, and P. Montague, "Reinforcement learning for autonomous defence in software-defined networking," in *Decision and Game Theory for Security*, L. Bushnell, R. Poovendran, and T. Başar, Eds. Cham, Switzerland: Springer, 2018, pp. 145–165.

[25] F. Wei, Z. Wan, and H. He, "Cyber-attack recovery strategy for smart grid based on deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 11, no. 3, pp. 2476–2486, May 2020.

[26] A. Goulart and A. Sahu, "Cellular IoT for mobile autonomous reporting in the smart grid," in *Securing the Internet of Things: Concepts, Methodologies, Tools, and Applications*. pp. 1025–1041.

[27] T.-H. Fan, X. Y. Lee, and Y. Wang, "PowerGym: A reinforcement learning environment for Volt-Var control in power distribution systems," 2021, *arXiv:2109.03970*.

[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[29] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," 2016, *arXiv:1602.01783*.

[30] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *J. Mach. Learn. Res.*, vol. 22, no. 268, pp. 1–8, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-1364.html

[31] J. Li, X.-Y. Ma, C.-C. Liu, and K. P. Schneider, "Distribution system restoration with microgrids using spanning tree search," *IEEE Trans. Power Syst.*, vol. 29, no. 6, pp. 3021–3029, Nov. 2014.

[32] A. Sahu. (Mar. 2023). *Open-DSS and SimPy based Cyber-Physical RL Environment*. [Online]. Available: https://github.com/NREL/DSS-SimPy-RL