

A Hybrid Game Theory and Reinforcement Learning Approach for Cyber-Physical Systems Security

Joseph Khoury, Mohamed Nassar

American University of Beirut

Department of Computer Science

Beirut, Lebanon

Emails: jck06@mail.aub.edu, mn115@aub.edu.lb

Abstract—Cyber-Physical Systems (CPS) are monitored and controlled by Supervisory Control and Data Acquisition (SCADA) systems that use advanced computing, sensors, control systems, and communication networks. At first, CPS and SCADA systems were protected and secured by isolation. However, with recent industrial technology advances, the increased connectivity of CPSs and SCADA systems to enterprise networks has uncovered them to new cybersecurity threats and made them a primary target for cyber-attacks with the potential of causing catastrophic economic, social, and environmental damage. Recent research focuses on new methodologies for risk modeling and assessment using game theory and reinforcement learning.

This paperwork proposes to frame CPS security on two different levels, strategic and battlefield, by meeting ideas from game theory and Multi-Agent Reinforcement Learning (MARL). The strategic level is modeled as imperfect information, extensive form game. Here, the human administrator and the malware author decide on the strategies of defense and attack, respectively. At the battlefield level, strategies are implemented by machine learning agents that derive optimal policies for run-time decisions. The outcomes of these policies manifest as the utility at a higher level, where we aim to reach a Nash Equilibrium (NE) in favor of the defender. We simulate the scenario of a virus spreading in the context of a CPS network. We present experiments using the MiniCPS simulator and the OpenAI Gym toolkit and discuss the results.

Keywords—CPS, ICS, SCADA Systems, Virus Spreading, Game Theory, Multi-Agent Reinforcement Learning

I. INTRODUCTION

Supervisory Control and Data Acquisition (SCADA) deployed in Cyber-Physical Systems (CPS) such as smart cities, and Industrial Control Systems (ICS) use advanced computing, sensors, control systems, and communication networks to monitor and control industrial processes and distributed assets. These systems, due to the extreme level of impact on human lives, are ranked first among the targets for malicious attackers or in cyber-wars.

In 2010, the cyber world witnessed the arrival of the most technologically sophisticated and malicious program developed for a targeted attack; The first cyber warfare weapon ever known as Stuxnet [1]. Since then, similar attacks proliferated, such as Havex and Industroyer [2]. Back in the days, traditional ICS were protected and secured by isolation from the Internet. However, with the technological industrial advances, these systems became more distributed and exposed to the Internet. Examples from Internet of Things (IoT) and smart grids

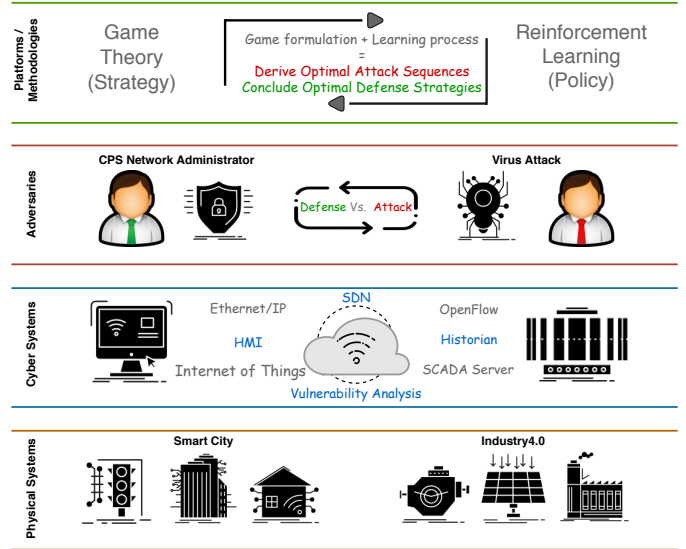


Fig. 1: Our approach in a nutshell: cyber-physical systems, human agents and machine agents, proposed modeling solution.

are abundant in the literature [3], [4], [5]. The interconnection between these systems and organizational enterprise networks is making them vulnerable and prone to cyber-attacks.

Recent research focuses on new methodologies for risk modeling and assessment. An increasing interest in game theory, control theory, and network optimization are witnessed amongst other methods in the will of deriving appropriate mitigation strategies to countermeasure potential attacks.

In this paper, we propose a hybrid human-machine approach to model the security game in between adversaries in the CPS ecosystem, as depicted in Fig. 1. At the bottom, the physical systems layer gives a general idea about the application of CPS and SCADA systems, including smart cities, industry 4.0, Critical Infrastructures (CI), and IoT. Second, the cyber systems layer handles the operations related to monitoring and controlling the physical systems. The third layer is for adversaries: the CPS network administrator trying to defend the cyber systems and reduce potential threats, and the rogue attackers trying to breach the cyber systems to cause physical damage. On top comes the methodology of modeling layer.

We propose a hybrid approach based on game theory as a tool to formalize the game interactions between the human

adversaries who decide on strategies. For instance, how to deal with security situations with different intensity levels. From the defender's point of view, which defense components are deployed, how firewalls are configured, how system patching is managed, etc. From the attacker's point of view, which infiltration techniques are used, how the malware is packaged, etc. This game has an extended form since the security status may change from low danger estimation to more severe states, and most attacks are multi-stage. It is also a game with imperfect information, since most of the time, both the defender and the attacker may not know the exact security state at the current time.

Based on the formulated game and decided strategies, MARL-powered machine learning agents may decide on the best actions (proactive or reactive) to take on the battlefield. The states in the RL model are different since they are run-time and perfect information states. Based on the knowledge of the network and the alerts of the defense systems, the defender chooses actions and navigates its own state space. The attacker navigates a parallel state space based on an attack tree. Therefore, the MARL agents learn the best policy for an automated response at run-time. These best policies for both the attacker and the defender manifest themselves as the utilities at the strategic game level. Therefore, we can compute a Nash Equilibrium (NE) at the strategic level that represents the best possible strategy in the presence of the adversary. A NE is a point that no party would like to deviate from without losing utility.

We focus in particular on a virus spreading scenario, with realistic assumptions such as discovering and using zero-day vulnerabilities. We run simulations using the MiniCPS simulator, and we use OpenAI Gym for the MARL implementation.

Additional novelties of the proposed model are as follows:

- Simulate a virus attack in a modern CPS network composed of four subnets.
- The use of predefined known and discovered vulnerabilities, in addition to the chance of discovering zero-day vulnerabilities.
- Derive optimal attack sequences and defense policies using MARL and Q-learning.
- Create a simulation framework composed of a network simulator and an RL toolkit.

The rest of the paper is organized as follows: Section II gives background information. Section III presents related work. In section IV, the strategic level and the battlefield level are analyzed. Implementation setup is discussed in section V. Section VI depicts simulation scenarios and results. Finally, section VII concludes the paper.

II. BACKGROUND INFORMATION

A. Game Theory

Game Theory is an effective formal tool for strategic behavior analysis. Game theory describes agents that are self-interested and interacting through a shared environment. Each agent has its description of the states of the world and which

states it likes most. Each agent acts based on this description and has a utility function. The utility function quantifies the degree of preference across alternative actions given the other agents choices. An agent has decision-theoretic rationality in the sense that it acts to maximize its expected utility in the long run. The goal of using game theory is to reach a Nash Equilibrium state (NE). NE is defined as a state where no agent would like to deviate from without losing utility to other agents.

B. Multi-Agent Reinforcement Learning (MARL)

MARL [6] is considered as an adaptation of game theory with the additional feature of machine learning. Learning in MARL is achieved through systematic trial and error in a shared and dynamic environment. The agents learn to choose actions that tend to increase their overall expected reward. A policy is a function that maps a state to an action. Optimal policies can be learned using a wide variety of algorithms including deep reinforcement learning. The goal of MARL is to derive optimal policies. MARL assumes that the game converges to a Nash Equilibrium after the learning phase.

III. RELATED WORK AND LIMITATIONS

Recent studies show that game theory is increasingly used in the cybersecurity domain. [7] proposes a dynamic game framework to model long-term interaction between stealthy attacker and proactive defender. Pure and mixed strategies are discussed in similar settings [8].

Reinforcement learning is equally used in the context of cybersecurity decision making. However, current studies mostly focuses on the last (physical) layer and control loops between actuators and sensors, without modeling the network at large. In [9], cybersecurity simulations in a vulnerable network are modeled as a Markov game with incomplete information and stochastic elements. An adversarial, sequential decision-making game is played with two agents, an attacker and a defender. Both agents use RL and involve tuning of different hyperparameters. However, the model contains only four network nodes including a starting node (where the attacker launches an attack), two vulnerable hosts, and the data asset node (also vulnerable). A notable result in [9] is that for the defender, a neural networks based agent performs worse than a table-driven agent. This can be explained by the fact that neural networks require much more data for training.

False Data Injection (FDI) attacks intend to threaten the security of power systems. FDI attacks have been recognized as significant threats to smart grids. In [10], a novel strategy of FDI attacks is proposed, which aims to distort the regular operation of a power system regulated by Automatic Voltage Controls (AVCs). The attack strategy is modeled as a Partial Observable Markov Decision Process (POMDP) problem. Attack actions are determined based on believed state inferred from recorded observations. A Q-learning algorithm with Nearest Sequence Memories (NSM) is adopted because this problem is impossible to be solved analytically. This algorithm helps the attacker to reach an optimal strategy. As

a mitigation to the proposed FDI Attack, bad data detection and correction method is developed, which is based on Kernel Density Estimation (KDE). Results have shown that false data injection can create voltage collapse with minimal knowledge and information about the whole environment.

In [11], a security decision-making game model with stochastic elements is proposed to characterize the interaction between attackers and defenders in Industrial Cyber-Physical Systems (ICPS), to generate optimal defense strategies, and to minimize system losses. The added value of their approach is that quantitative vulnerability analysis and time-based unified payoff quantification. The proposed game is considered a mixed-strategy game. Q-learning algorithm is used to approximate the optimal strategies by continually updating a Q-function.

The previously mentioned related work propose novel ideas in the field of ICS and CPS networks with the use of game theory and RL algorithms. However, they still lack in many aspects. The work proposed in [9] studies cybersecurity simulations in a small and limited network and thus does not reflect realistic CPS network implementations, in addition to the vulnerability analysis for the network hosts and data set that does not reflect a precise real case. In our work, we do not blindly use neural networks before trying the by-book approach for Q-learning. In [10], the attacker's action in the proposed FDI problem is evaluated in the absence of the defender, and the information about the vulnerable elements is not identified. These arguments also make the problem lack the realism aspects of vulnerabilities and cyberattacks in CPS networks.

Regarding the proposed work in [11], the attacker had complete information about the environment that is under attack, which contradicts the clear case of an outsider attacker trying to breach an unknown system. The attacker was bounded to exploit a limited number of pre-defined vulnerabilities. The fact of discovering and exploiting zero-day vulnerabilities is neglected.

IV. PROPOSED MODEL

The proposed hybrid approach is divided into two levels: first, a higher game strategic level and second, a lower battlefield level.

A. Strategic level

We model the strategic level using an imperfect information extensive form game. The states of this game are the overall security status: Low, medium, high, and critical danger. Obviously, the defender does not know for sure what is the actual state (e.g., the attacker has discovered a zero-day vulnerability). The attacker does not know what the state is since he does not have the full information about the target network or all the defender countermeasures. Therefore, we consider an imperfect information model. This game has an extended form since the defender will try to recover from corrupt states to the original (low) state.

The attacker will try to reach the critical state. Therefore the game has multiple stages. The defender chooses the strategy in terms of countermeasures to impede the progress of the attacker. The strategy is translated into a set of actions to choose from at the Battlefield level. The attacker selects the strategy in terms of attack methodologies. Attacker strategies are translated as a set of actions, vulnerabilities, and penetration tools at the Battlefield level.

B. Battlefield level

This level is modeled using MARL and represents a multi-stage stochastic game with a learning component; each agent has a game turn with a random transition probability and creates a new scenario for the other agent to act accordingly. The state of the game is composed of the state of the defender based on the alerts, and the state of the attacker based on its attack tree. The nature of this game is stochastic, since some attacks can fail with some probability, and some countermeasures can fail with another probability.

1) *Network Architecture*: The game model is based on a modern CPS network with an architecture segregated into four subnets and divided into two layers describing a real CPS network. First, a cloud-based network hosting the web server, email server, and the database server of the CPS organization. Second, an enterprise network that hosts all the office's computers for the CPS enterprise. Third, a SCADA network that hosts the SCADA Server responsible for data acquisition, the Human Machine Interface (HMI) used by operators to monitor sensor values and take commands accordingly, and the SCADA historian where acquired data are stored. Finally, The field network contains the goal of the attacker, which is the control units represented by a Programmable Logic Controller (PLC). PLCs execute commands sent from the SCADA network on the physical processes and also retrieves sensor values from sensors deployed all over the industrial process line to be afterward sent to the SCADA network for processing and storing.

Each host deployed on the different subnets of the CPS network is explicitly assigned a unique private directory acting as its storage unit. These directories will be used as a ground for virus spreading among the hosts, the four subnets, and the two layers of the CPS network. To correctly mimic a modern CPS network with its two different layers and its subnets, MiniCps [12] is used as a network simulator to target network communication, control logic, and physical layer interaction. Using MiniCps features, a terminal can be launched at each host to execute a specific network command. In this paperwork, we focus on implementing a virus spreading among the hosts; Netcat command was used to read and write files using a TCP connection, which allows the virus to pass from a directory to another until reaching its final destination.

2) *Vulnerability Analysis*: The CPS network that is designed and implemented in this paperwork is a vulnerable CPS network segregated into four subnets. The cloud-based subnet is hosted on the cloud by an outside organization such as Amazon Web Services (AWS) [13], which is a modern

CVE ID	Type of Vulnerability	Vulnerability Access	Exploitability Score	CVSS Score
CVE-2019-11133	DoS	Local	2 (Low)	4.6
CVE-2011-5163	Overflow	Local	1 (Low)	4.6
CVE-2018-1168	Code Execution	Local	3 (Low)	7.2
CVE-2016-2281	Gain Privileges	Local	7 (High)	6.0
CVE-2016-9360	Gain Information	Local	6 (Medium)	4.4
CVE-2015-3940	Directory Traversal	Local	4 (Medium)	6.9
CVE-2013-0654	DoS	Remote	4 (Medium)	9.3
CVE-2011-1918	Overflow	Remote	2 (Low)	10.0
CVE-2015-7908	Code Execution	Remote	5 (Medium)	9.3
CVE-2019-6564	Gain Privileges	Remote	4 (Medium)	9.3
CVE-2018-11517	Gain Information	Remote	2 (Low)	5.0
CVE-2014-0751	Directory Traversal	Remote	2 (Low)	7.5

TABLE I: Chosen sample of vulnerabilities with different vulnerability types, access, and scores.

practice adopted by all big firms these days. Also, the rest of the subnets are hosted locally in the CPS organization. The base metric provided by Common Vulnerability Scoring System (CVSS) [14] is adopted to quantify vulnerabilities on network nodes in this paperwork. More specifically, the exploitability score that describes best the complexity of exploiting a vulnerability.

Since the first subnet is cloud-based, it is characterized by a high level of security with a secure Intrusion Detection System (IDS) integration. However, the rest of the subnets hosted locally in the CPS organization are vulnerable with an uncertain IDS integration. This paperwork only studies viruses' behaviors; for that reason, local and remote access vulnerabilities are adopted with different types.

Table I shows a small chosen sample of the used vulnerabilities related to the different CPS subnets; they are collected from Common Vulnerabilities and Exposures (CVE) Details [15]. CVE Details is an online database that gathers vulnerabilities with their corresponding information such as attack types, access types, exploitability scores, and CVSS scores. We gathered in total 65 vulnerabilities for the network devices. Each network device assigned with a set of five vulnerabilities. These vulnerabilities, depending on their exploitability scores, determine the transition probabilities for the attacker to bypass the network defenses, and reach the end goal. The defender can invest in patching these vulnerabilities to strengthen the system, with rewards proportional to the exploitability scores. The proposed game also gives the chance to discover zero-day vulnerabilities that are unknown to the CPS network administrator until discovered and exploited by the virus attack. This intended feature provides the model with a sense of real-life virus targeted attacks. The defender must be aware that undiscovered vulnerabilities may exist.

3) *Game Description:* The setting of this game is a vulnerable CPS network based on a dynamic environment with two adversaries and a stochastic transition. The attacker represented as a malicious virus spreading and targeting the control units hosted on the field network, and the defender described as the CPS network administrator defending the CPS network.

The game is played in a multistage manner, where each

agent applies an action on the environment and receives a reward, and at the same time creates a new situation for the environment to be faced by the adversary, and this makes the environment dynamic. Fig. 2 shows the proposed game model for virus spreading in a multi-subnet and cross-layer CPS network.

The CPS network is formed of two main layers. The first layer that represents the cyberspace layer contains three subnets: the cloud-based system, which has a low impact on the overall CPS security. The enterprise network, with a medium effect on the security of the CPS network. Finally, the SCADA network in this layer is considered of high-security impact due to its nature of monitoring and controlling control units and edge devices hosted on the field network [16].

The second layer represents the physical space layer containing both the field network that is stamped with an extreme security impact on the overall CPS operations; due to the presence of the attacker goal, in addition to the physical industrial process that depends on the functionality nature of the CPS organization.

The purpose of the attacker resides only in the physical layer and more precisely on the field network. Also Fig. 2 shows all the possible transition paths of the virus on the CPS network. The spread of the virus is done by exploiting vulnerabilities on network devices and switches having firewalls capabilities until reaching the end goal. Each node of the graph is associated with a set of vulnerabilities, a vector of defense values containing the exploitability scores of the corresponding vulnerabilities. The defender is assumed to have complete information about the architecture of the CPS network.

On the other hand, the attacker has no information about the CPS network; information is collected in the process of attacks only. At each game step, the attacker tries to find and exploit a specific vulnerability on a particular node while the defender counters the attack by increasing the defense values to protect the system. Whenever the success rate of an attack exceeds the exploitability score of the selected vulnerability, the offense is considered auspicious, and the virus advances to the exploited node. However, the defender at each game step tries to make the system harder to breach by leveraging the

complexity of exploiting vulnerabilities.

First, RL is applied using Q-learning to find the optimal sequence attack path to reach the goal node. Moreover, MARL handling Q-learning is applied for both agents to reach the NE, and find the best defense strategies.

4) *Game Formulation*: Next is a six-tuples defining the essential parameters of the proposed game model: the adversaries agents tuple, a set of states representing the 14x6 game model matrix (14 nodes, five vulnerabilities, level of protection), two sets for attacker and defender action samples, a set of reward corresponding to each state, and finally a set of transition probabilities that is represented by the exploitability score of the vulnerabilities.

$$\left\{ \begin{array}{ll} \text{Agents:} & X = \{X_a, X_d\} \\ \text{States:} & S = \{\{S_1^1, \dots, S_1^6\}, \dots, \{S_N^1, \dots, S_N^6\}\} \\ \text{Att. actions:} & A = \{\{A_{S_1^1}, \dots, A_{S_1^5}\}, \dots, \{A_{S_N^1}, \dots, A_{S_N^5}\}\} \\ \text{Def. actions:} & D = \{\{D_{S_1^1}, \dots, D_{S_1^6}\}, \dots, \{D_{S_N^1}, \dots, D_{S_N^6}\}\} \\ \text{Rewards:} & R = \{\{R_{S_1^1}, \dots, R_{S_1^6}\}, \dots, \{R_{S_N^1}, \dots, R_{S_N^6}\}\} \\ \text{Trans. prob:} & P = \{\{P_{S_1^1}, \dots, P_{S_1^5}\}, \dots, \{P_{S_N^1}, \dots, P_{S_N^5}\}\} \end{array} \right.$$

The defender uses extra parameters during the simulation, which helps determine a win-game for the defender; a value W that returns zero if the virus is not detected and one if identified by the IDS, and another value T that indicates if the game is in a terminated state. In addition to value Y that delimits the steps of an attacker, and a success rate of V for the attacker actions. N is used to indicate the index of the last node on the system:

$$\left\{ \begin{array}{ll} \text{IDS:} & W \in \{0 : \text{non-detected}, 1 : \text{detected}\} \\ \text{Terminal state:} & T \in \{0 : \text{non-terminal}, 1 : \text{terminal}\} \\ \text{Maximum steps:} & Y \in [0; 99] \\ \text{Success rate:} & V \in [0; 99] \\ \text{Last node index:} & N = 14 \end{array} \right.$$

Each agent of the game model has its goal state for winning a game:

- The goal state of the attacker is to reach the control units of the CPS network. It is represented by the end node of the proposed game model in Fig. 2 where the attack success rate should exceed the exploitability score; formally described in equation (1):

$$X_a^{goal} = \exists S_N^j \in S, A_{S_N^j} \in A \text{ and } V_{A_{S_N^j}} \in \mathbb{R} \mid \forall j \in [1; 5], T = 1 \text{ and } V_{A_{S_N^j}} > P_{S_N^j} \quad (1)$$

- For the defender goal state, a game is considered auspicious in two cases: First whenever the attacker actions success does not exceed the transition probability on the last node, and the IDS detects the attack then blocks it; formally depicted in equation (2). Second when the maximum attack steps are exceeded; presented in equation (3):

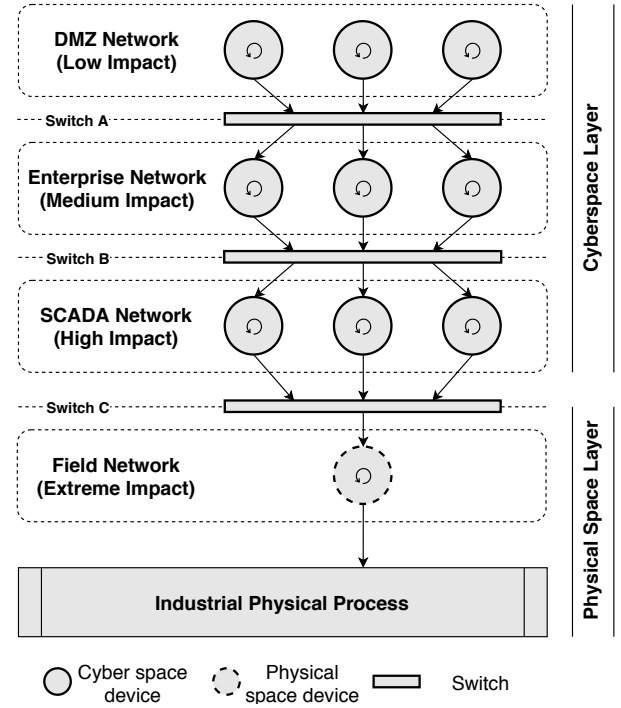


Fig. 2: Proposed game model for virus spreading in a multi-subnet and cross-layer CPS network.

$$X_d^{goal} = \exists S_N^j \in S, A_{S_N^j} \in A, V_{A_{S_N^j}} \in \mathbb{R} \text{ and } Y \in \mathbb{R}$$

$$\mid \forall j \in [1; 5], T = 1, W = 1, V_{A_{S_N^j}} \leq P_{S_N^j} \text{ and } Y \in [0; 99] \quad (2)$$

Or,

$$X_d^{goal} = \exists S_i^j \in S, A_{S_i^j} \in A, V_{A_{S_i^j}} \in \mathbb{R} \text{ and } Y \in \mathbb{R} \mid$$

$$\forall j \in [1; 5], \forall i, T = 0, \forall V_{A_{S_i^j}} \text{ and } Y > 99 \quad (3)$$

V. IMPLEMENTATION SETUP

The goal of this paperwork is to design and implement a CPS network that is the closest in architecture and features to modern CPS networks and, at the same time, simple to be used for simulation and learning purposes. Fig. 3 shows the proposed framework architecture that is created with the combination of MiniCps network simulator, and OpenAI Gym toolkit to achieve the requirements of the proposed hybrid approach. First, the CPS network is created using MiniEdit, a helper tool in MiniCps [12] that allows the creation of a network with a drag and drop functionality and at the end generates a Python file describing the designed network with all its hosts, switches, and links.

Each network device from Fig. 2 was implemented as a host connected by switches for routing. Upon generation, the file is edited to support explicitly (1) private directories for each host

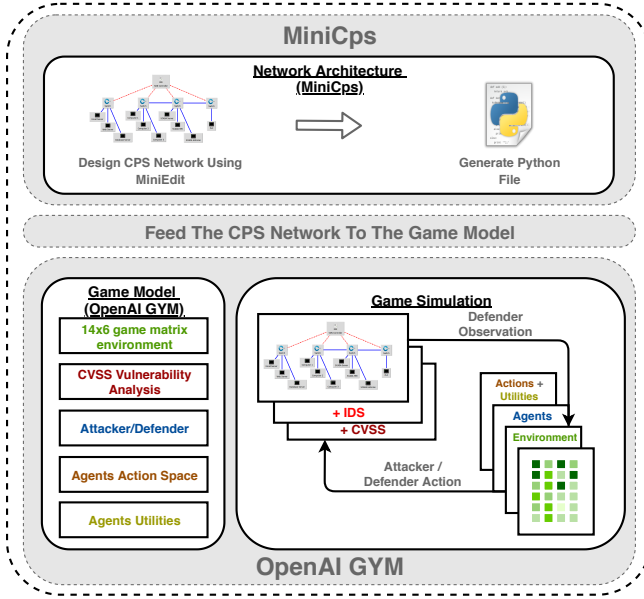


Fig. 3: Proposed framework architecture using OpenAI Gym toolkit, and MiniCps network simulator.

on the CPS network, (2) IDS implementation to detect attacks and return proper observation for learning purposes, and (3) integration of vulnerability analysis on the hosts and switches of the CPS network. Using OpenAI Gym, the environment of the proposed game model is created; it is a 14x6 matrix representing the 13 nodes of the network plus a starting node as rows and the set of five vulnerabilities plus the IDS strength for each host on the network as columns, at each node the first five cells of the matrix contains the exploitability score of a vulnerability on a specific node and the last cell contains the strength of the IDS on each node.

These exploitability scores represent the defense values of the defender and how the network is immune against attacks. Also, using the Gym library, the action spaces for both agents are determined, the rewards for each node of the environment are set, and helper functions are created to apply the chosen action on the designed CPS network that is fed into the game model.

By merging the CPS network created using MiniCps and its add-ons (Directory creation, IDS integration, and vulnerability analysis) with the created game model using OpenAI Gym, a robust framework is designed to conduct simulation, execute a different kind of strategies, and apply learning algorithms to derive optimal defense policies. In this study, Q-learning was used to achieve the learning element. The environment design has a limited number of states with a limited number of actions allowed for agents; this argues the use of an algorithm based on tabular representations instead of an algorithm based on neural networks. In addition to the problem that neural networks have with adversarial learning, where the network is slow in adapting to the continually changing environment [9].

The proposed framework is used along MARL implementation to conduct different game strategies, derive optimal

attack and defense policies using Q-learning, and evaluate the proposed adversarial multistage game-theoretical model. In the beginning, two Q-tables are initialized with zeros for both agents. The rows are equal to the number of states, and columns corresponding to the number of actions for the agents. At each game stage, both agents take action and change the state of the environment while receiving a reward evaluating their actions, respectively.

Parameters	Parameters Sets	Best Parameter
Learning Rate	0.3, 0.4, 0.5, 0.6	0.5 / 0.6
Discount Factor	0.6, 0.7, 0.8, 0.9	0.8 / 0.6
Decay Rate	.005, .006, .007, .008	.008 / .005

TABLE II: Simulation setup parameters.

In table II, we present implementation parameters for the simulation setup that including the learning rate, discount factor, and decay rate. The learning rates describe the degree of accepting new values compared to previous values. The discount factor affects how much weight is considered for future rewards. Moreover, a trade-off between exploration and exploitation is determined using a decay rate. When applying Q-learning for both agents, each agent can reach its optimal policies using different parameter values. The Q-tables are also updated at each game step, and at the end of the simulation runs, Q-tables are concluded, showing the best action to choose for each state of the environment. The completed Q-tables for the attacker help discover potential vulnerabilities in the network to ease the attack and reduce the attack steps to reach the control units. Moreover, the completed Q-table for the defender helps discover optimal defense policies to counter the attacker's action by patching the network vulnerabilities and improving the IDS strength in detecting attacks.

VI. DISCUSSIONS AND RESULTS

To conclude an optimized sequence of attacks, and determine defense strategies with optimal policies, two different simulation scenarios are used to show first the potential of dedicated viruses attacks on a CPS network with no defense strategies, and second the urge of having optimized defense strategies to counter these kinds of attacks and reduce system losses.

A. Strategy Vs. Policy

Fig. 4 gives an idea about the use of game theory and MARL; it correlates the notion of strategies in game theory with the concept of policies in RL. The problem faced in CPS networks is first modeled and formalized based on game theory principles. Each player has defined strategies and action space to act accordingly. Also, with the use of MARL, the formalization of the game model is implemented into an environment to apply the learning process using the Q-learning algorithm. Furthermore, derive optimal policies for the selected strategies.

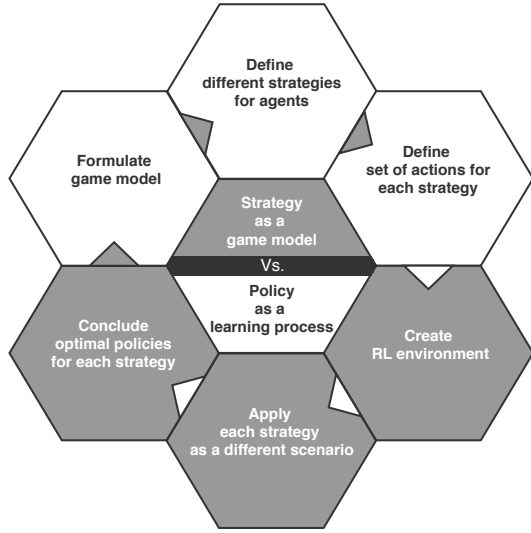


Fig. 4: Strategy Vs. Policy.

B. Game Strategies

Each agent has three strategies; a common strategy is adopted by both agents, which is the random strategy that acts as a baseline. In this strategy, both agents will choose and apply actions to the environment based on arbitrary decisions. The attacker has a stealthy strategy that relies on reducing the number of steps and staying low key to avoid being detected. And the second strategy depends on budget spending to increase the chance of successful attacks. The defender, in addition to the randomized strategy, also has two strategies first one relies only on human capabilities and patch management, and the second relies on machine capabilities and IDS enhancement.

A simulation of 10^6 games is conducted with the application of MARL using the Q-learning algorithm with the proposed framework Fig. 3. The framework requires several inputs, such as the network architecture, in addition to some configuration parameters required for the game model design. To depict the simulation results provided by the framework, a win-factor plot is used, which is a running average of the gained games for both agents over the entire simulation runs. A scale of $[1, -1]$ is used: (1) means the defender is the winner, (-1) means the attacker is the winner and presented with a green and red horizontal lines, respectively.

C. MARL Policies

First, attacker strategies are applied to learn their optimal policies and map actions to states. Fig. 5 shows three curves; In gray, the random strategy acting as a baseline, and showing no clue for the virus attack in reaching the end goal and maximizing its rewards. In olive, a stealthy strategy adopted by the attacker that shows how the attacker is learning with the increase of iterations runs. Finally, in brown, a budget spending strategy that performs better than the previous strategy due to the increase of attack success rate. In Fig. 5 optimal policies were derived for stealthy and budget spending

strategies, that showed excellent performance compared to a randomized strategy.

Second, defender strategies are applied to learn their optimal policies and map actions to states. Fig. 6 shows three curves; In gray, the same as previously described acting as a baseline. In olive, a strategy that relies on human capabilities only; this strategy showed noticeable performance in preventing the attacker from winning more games. Finally, in brown, a strategy that relies only on machine capabilities also showed remarkable performance in limiting the attacker's success rate. In Fig. 6 optimal policies were derived for human and machine strategies; that showed close results and prominent performance compared to only attacker strategies conducted in Fig. 5.

D. Testing Strategies With Learned Policies

In this section, testing is conducted on the learned policies for both attacker and defender strategies. The results show the percentage of successful attacks in a 10^6 game simulation runs for a virus attack starting from all four subnets independently and also shows the success rate by layer noting that the work incorporates both cyber and physical layer.

First, in Fig. 7, testing for attacker learned policies is done. Results show that virus attacks starting from all subnets the attack success rate was way above the average for both strategies. And it is summarized by layers, where a virus attack on average managed to succeed by 77% on the cyber layer and by 99% on the physical layer.

Second, in Fig. 8, testing for defender learned policies is done with the presence of attacker learned policies. Results show noticeable performance in limiting the success rate of virus attacks. Optimized policies for the selected defender strategies showed that the defender is able to reduce the success rate of attacks, especially in the first three subnets that represent the cyber layer. However, these strategies barely changed the results and managed to limit viruses attacks on the last subnet, which is hosted on the physical layer; due to the presence of the attacker goal where the defender has

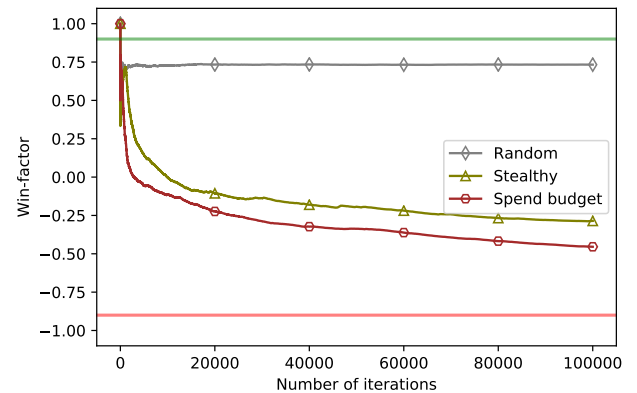


Fig. 5: Win-factor for learned attacker vs. random defender.

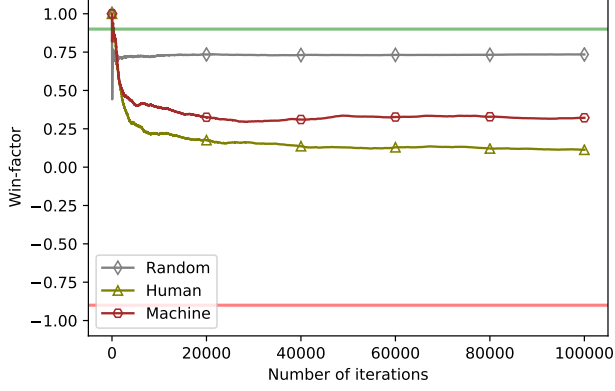


Fig. 6: Win-factor for learned attacker vs. learned defender.

little to no chance in blocking attacks. As a summary, a virus attack on average managed to succeed by 25% on the cyber layer and by 94% on the physical layer. In conclusion, both defender strategies managed to limit the virus attack success rate.

VII. CONCLUSION AND FUTURE WORK

Future malware will undoubtedly be equipped with automated learning components. This paperwork takes this assumption and proposes a framework based on a hybrid approach using game theory and RL to model an adversarial game for cross-layer virus spreading in CPS networks. The work presented makes use of previous literature and adds what we consider more realistic features such as the simulation of a virus spreading, vulnerability analysis, zero-day exploits, and a larger CPS network architecture. The formulation includes a network architecture with different layers and subnets that mimics a real and modern CPS network. The utilities of the game are the cumulative rewards that can be achieved by the RL agents after the learning phase.

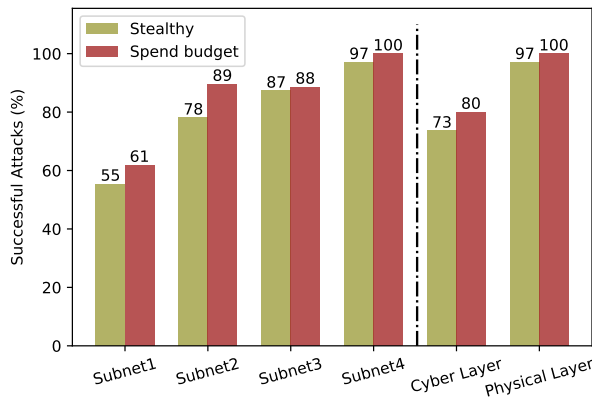


Fig. 7: Testing only attacker strategies with derived policies.

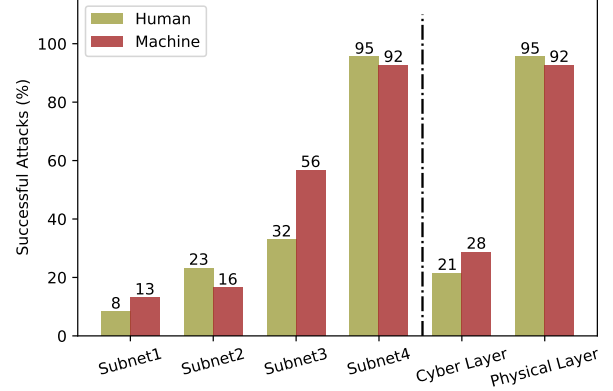


Fig. 8: Testing both attacker and defender strategies with derived policies.

MARL was applied using Q-learning for both agents to learn optimal policies. The obtained results show the ability of the defender to derive optimal defense policies based on a human-selected strategy to reduce losses and prevent viruses from spreading across the CPS networks. A mixed defense strategy can lead the game to a NE, where the attacker would not like to change behavior since it can be countered more easily. A particular focus in this paper was put on the RL level in terms of design and simulations. In future work, we will focus more on the strategic level, by dissecting the extended form security game with imperfect information.

ACKNOWLEDGMENT

This work was supported partially by a grant from the university research board of the American university of Beirut (URB-AUB-2019/2020).

REFERENCES

- [1] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.
- [2] "Seven years after stuxnet: Industrial systems security once again in the spotlight." <https://www.welivesecurity.com/2017/06/16/seven-years-stuxnet-industrial-systems-security-spotlight/>. Accessed: 2019-01-15.
- [3] A. Ghasempour, "Internet of things in smart grid: Architecture, applications, services, key technologies, and challenges," *Inventions*, vol. 4, no. 1, p. 22, 2019.
- [4] A. Ghasempour and T. K. Moon, "Optimizing the number of collectors in machine-to-machine advanced metering infrastructure architecture for internet of things-based smart grid," in *2016 IEEE Green Technologies Conference (GreenTech)*, pp. 51–55, IEEE, 2016.
- [5] A. Ghasempour, "Optimum number of aggregators based on power consumption, cost, and network lifetime in advanced metering infrastructure architecture for smart grid internet of things," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pp. 295–296, IEEE, 2016.
- [6] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine learning proceedings 1994*, pp. 157–163, Elsevier, 1994.
- [7] L. Huang and Q. Zhu, "A dynamic games approach to proactive defense strategies against advanced persistent threats in cyber-physical systems," *arXiv preprint arXiv:1906.09687*, 2019.

- [8] C. Y. Ma, N. S. Rao, and D. K. Yau, "A game theoretic study of attack and defense in cyber-physical systems," in *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 708–713, IEEE, 2011.
- [9] R. Elderman, L. J. Pater, A. S. Thie, M. M. Drugan, and M. Wiering, "Adversarial reinforcement learning in a cyber security simulation.," in *ICAART (2)*, pp. 559–566, 2017.
- [10] Y. Chen, S. Huang, F. Liu, Z. Wang, and X. Sun, "Evaluation of reinforcement learning-based false data injection attack to automatic voltage control," *IEEE Transactions on Smart Grid*, vol. 10, no. 2, pp. 2158–2169, 2019.
- [11] K. Huang, C. Zhou, Y. Qin, and W. Tu, "A game-theoretic approach to cross-layer security decision-making in industrial cyber-physical systems," *IEEE Transactions on Industrial Electronics*, 2019.
- [12] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.
- [13] "Amazon web services (aws)." <https://aws.amazon.com>. Accessed: 2019-08-05.
- [14] "Common vulnerability scoring system." <https://www.first.org/cvss/>. Accessed: 2019-08-02.
- [15] "Common vulnerabilities and exposures (cve)." <https://www.cvedetails.com>. Accessed: 2019-08-05.
- [16] H. Herry, E. Band, C. Perkins, and J. Singer, "Peer-to-peer secure updates for heterogeneous edge devices," in *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–5, IEEE, 2018.