*Grenze International Journal of Engineering and Technology, Jan Issue*

# A Survey on Intrusion Detection Systems using Deep Reinforcement Learning

Arsalan Anwar[1] and Dr. D.G. Jyothi[2]
[1]Data Scientist, West Pharmaceutical Services, Bangalore, India
Email: arsalan.anwar@westpharma.com
[2]Professor and Head, Department of AI & ML
Bangalore Institute of Technology, Bangalore, India
Email: dgjyothi@bit-bangalore.edu.in

***Abstract*—The significant increase in cyber-attacks particularly during the COVID-19 pandemic, calls for efficient cyber security systems which help organizations detect and mitigate cyber threats. Several traditional cyber defense systems such as Intrusion Detection Systems (IDS) are in place, but the ever evolving and complex novel attacks demand advanced and self-learning systems that can adapt to detect such threats. Such autonomous systems have been implemented using modern reinforcement learning (RL) techniques. This paper presents a survey on various IDSs implemented using Deep RL methods. We cover numerous architectures and present ways by which different authors address the common issues in IDSs such as the tradeoff between accuracy and the false positive rate (FPR), high computational requirements, etc. We expect that this comprehensive survey provides the foundations for and facilitates future studies on exploring the potential of implementing robust and advanced IDSs using DRL by addressing some of the common limitations like low accuracy, high FPR & high computational requirements.**

***Index Terms*— Deep Reinforcement Learning, Intrusion Detection System, Q-Learning, Deep Q Networks, Cybersecurity, NSL-KDD, AWID, CIC-IDS17, UNSW-NB15.**

## I. INTRODUCTION

Since the onset of the COVID-19 pandemic, when the global economy transitioned into the work from home (WFH) culture, a significant increase in cyber-attacks has been observed. These cyber-attacks are said to be caused by various internal and external factors such as negligent and incautious user actions, failures in systems & technology, and evolving cyber-attacks. While the enterprises are trying to address the internal factors by providing appropriate training to their employees, the external factors are uncontrollable. Enterprises that are aware of the quantum of damage that these attacks can have on them, understand the importance of having robust cyber security systems, especially Intrusion Detection and Prevention Systems so that they can identify the threats & mitigate the risks immediately. However, these cyberattacks & threats are constantly evolving, with 360,000 new malware signatures detected every day which calls for advanced cyber security systems in place [1].

Traditional intrusion prevention methods such as encryption and access control firewalls do not fully prevent the systems from advanced attacks [2] and it is observed that these attacks have now become dynamic which means

*Grenze ID: 01.GIJET.9.1.95*

the attacks circumvent the cyber security systems that are in place at each level of the organization. This dynamism in the attacks calls for dynamic and responsive Intrusion Detection Systems which can quickly alert unforeseen events and help organizations reduce the impact of a cyber-attack [3].
This dynamism can be achieved using RL. RL is a branch of machine learning where an agent learns to make decisions through trial and error. Unlike the other two branches of machine learning namely supervised and unsupervised learning, reinforcement learning uses the feedback obtained from its actions to learn from its mistakes. This feedback mechanism helps the agent to provide dynamic and sequential responses to the ever-evolving cyber threats.
Through this survey, we first understand the importance of having robust and dynamic cyber security systems. We then cover IDS and its types in Section II. After that, in Section III, we understand deep reinforcement learning and the Markov Decision Process (MDP) which represents the RL problem at hand. Following that, we cover the Agent-Environment interaction and types of DRL. In section IV, we cover the various implementations of IDSs using DRL and understand various architectures used, the underlying DRL models along with their advantages and disadvantages.

## II. Intrusion Detection System

### *A. What is IDS?*

Intrusion detection elucidates identifying unauthorized use, misuse, and abuse of computer systems by both internal users and external intruders. The main function of an Intrusion Detection System (IDS) is to secure a computer system or computer network by detecting malicious attacks on a network system or host device by monitoring inbound network traffic to uncover any anomalous and abnormal behaviour [4]. These intrusions can prove to be very costly and hence Intrusion Detection Systems play an important role in early detection of intrusions, reducing the damage to the network structure or the data exchanged across the network.

### *B. Types of IDS*

There are many ways to classify types of IDS in a production network. These classifications are not mutually exclusive; for instance, a network-based IDS may be using the signature-based approach to detection [4][5]. The widely classified types of IDS are:
1) *Host-Based IDS:* A host-based IDS (HIDS) is an IDS that generally operates within a computer, node, or device. Its main function is internal monitoring. This was the first type of intrusion detection software to have been designed, with the original target system being the mainframe computer where outside interaction was infrequent. A HIDS monitors and collects the characteristics of hosts containing sensitive information, servers running public services, and suspicious activities. For example, it can detect a malicious/ hostile program that accesses a system's resources in a suspicious manner or discover that a program has modified the registry in a harmful way.
2) *Network-Based IDS*: A network-based IDS (NIDS) is usually placed along a LAN wire which differs from HIDS. NIDS attempts to discover unauthorized and malicious access to a LAN by analyzing traffic that traverses the wire to multiple hosts. There are many algorithms for detecting malicious traffic, but they generally read inbound and outgoing packets and search for any suspicious patterns. Any alert generated by a NIDS allows it to notify administrators or take active actions such as blocking the source IP address.
3) *Anomaly-based IDS:* Anomaly-based IDS (AIDS) works by identifying patterns from users or groups of users already defined. The detection system utilizes machine learning to recognize a normalized baseline which represents how the system normally behaves, and then all network activity is compared to that baseline. Then anomaly-based IDS simply identifies any abnormal behavior to trigger alerts.
4) *Signature-based IDS:* Signature-based IDS (SIDS) is typically best used for identifying known threats. It operates by using a pre-programmed list of known threats and their indicators of compromise (IOCs). An IOC might be a specific behavior or signature that generally precedes a malicious network attack, known byte sequences, file hashes, malicious domains, etc. SIDS monitors the packets traversing the network and compares these packets to the database of known IOCs or attack signatures to flag any suspicious behavior

## III. Deep Reinforcement Learning

### *A. What is Deep Reinforcement Learning?*

Deep reinforcement learning (DRL) combines Reinforcement Learning (RL) with Deep Learning to utilize the benefits of the latter such as its capability to handle complex, high dimensional data with minimal manual

feature engineering. DRL algorithms incorporate deep learning to solve the mathematical representation of the problem called the Markov Decision Process (MDP) [6]. A popular RL method is Q-learning [7] (Q stands for quality) whose aim is to maximize the discounted cumulative reward based on the Bellman equation which is given as follows:

$$Q(s, a) = R(s) + \gamma\,[\max Q_{old}(s, a')], \quad (1)$$

where $s$ is the current state, $a$ is the action taken and $\gamma$ is the discount factor.

1) *Markov Decision Process (MDP)*: MDPs formally describe an environment for RL where the environment is fully or partially observable. Almost all RL problems can be formalized as MDPs where MDPs formally describe an environment for RL where the environment is fully or partially observable. MDP is given as a tuple $(S, A, P, R, \gamma)$ where $S$ defines the state space, $A$ defines the action space, $P$ defines the transition probability function which is given as:

$$_{ss'}P^{a} = p(s_{t+1} = s' \mid s_t = s, a_t = a), \quad (2)$$

$R$ is the reward function given as:

$$_{ss'}R^{a} = E(r_{t+1} \mid s_t = s), \quad (3)$$

and $\gamma \in [0, 1]$ is the discount factor which helps us define how much weightage immediate rewards and the future rewards should be given. Deep reinforcement learning algorithms incorporate deep learning to solve such MDPs, often representing the policy $\pi(a|s)$ or other learned functions as a neural network.

2) *Agent-Environment Interaction*: In Reinforcement Learning, the agent learns to solve a given problem by trial and error i.e., the agent takes decisions and gets a reward based on how good or bad that decision is. The agent is the decision maker comprising the RL model. This agent interacts with the outside world called the environment which can be anything that is out of the control of the agent. For example: In an autonomous car, the model that is taking the action is called the agent and everything apart from the model like the wheels of the car, roads, wind movement, traffic, etc. is the environment.

The sequence of events in a typical Agent-Environment Interaction for a time step t is shown in the Fig. 1. At time $t$, the agent takes an action $a_t$ based on the initial state $s_t$. The environment then evaluates the goodness of the action $a_t$ using a reward function $R$ and sends the reward $r_t$. Based on the action $a_t$, the environment computes the next state $s_{t+1}$ and sends $s_{t+1}$ to the agent. On receiving the reward $r_t$, the agent learns how to take optimal actions. Additionally, by using the information of the new state $s_{t+1}$, the agent takes an action $a_{t+1}$ at time step $t+1$. This interaction keeps continuing until the agent learns how to solve the problem efficiently.

*B. Types of DRL*

Reinforcement Learning Algorithms are broadly classified into Model-Based and Model-Free Methods. To understand the classification, we need to revisit the MDP and its components. For any decision to be taken efficiently, the agent requires itself to know the reward function R and the transition probability function P. But these components of MDP are not always known to the agent at the beginning. And in cases where these are unknown to the agent, the agent tries to estimate the reward function $R$ and transition probability function $P$ such that the policy obtained maximizes the expected future reward.

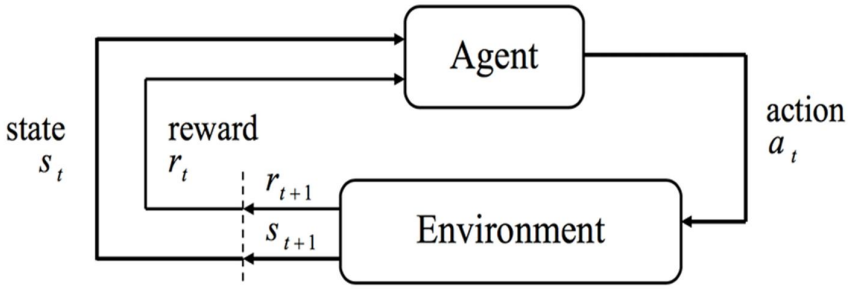


Figure 1. Agent-Environment Interaction

In the model-based approach, the agent tries to learn a model of how the environment works from its observations and then find an appropriate policy using that model. This is done by observing the sequence of actions taken by the agent, the states it transitions into, and the rewards obtained for the action chosen. That is, if the agent is currently in state $s_1$, takes action $a_1$, and then observes the environment transition to state $s_2$ with reward $r_2$, that information can be used to improve its estimate of $P(s_2|s_1, a_1)$ and $R(s_1, a_1)$. Once the agent has adequately modelled the environment, it can use a planning algorithm such as value iteration, policy iteration, etc. to find the policy.

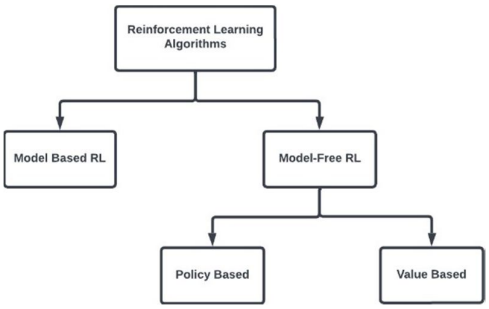


Figure 2. Types of Deep Reinforcement Learning

In the model-free approach, instead of learning a model of the environment to find a good policy, the agent directly tries to estimate the utility (optimal values) of each action in each state. For example, in Q-Learning, the agent directly estimates the optimal Q-values of each action in each state from which a policy can potentially be derived by choosing the action with the highest Q-value in the current state. Therefore, as these approaches do not learn a model of the environment, they are called model-free algorithms.

RL algorithms can also be classified as value-based, policy-based or as a combination of both. In value-based methods, the goal is to find an optimal value function that maps the states to values. Once the value function is evaluated, a policy is derived from the value function. Note that the policy here is implicit as it uses the value function to pick the action with the best value. Value-based methods like DQN (Deep Q Networks) take longer training times and have limitations in solving problems with continuous action spaces. These methods evaluate the goodness of an action given a particular state using the Q-value function. When the number of states or actions is large/ infinite, they tend to show inefficiency and/or even impracticality. Other examples of value-based methods include Q-Learning and SARSA (state-action-reward-state-action) [8].

Unlike value-based methods, policy gradient methods learn the policy explicitly and by mapping states to actions ($\pi : s \rightarrow a$) and keep it in memory during learning. This mapping would be a probability distribution over all possible actions. Policy gradient methods tend to solve the problems faced by value-based methods as they work well with continuous action spaces. REINFORCE [9], vanilla policy gradient [10], trust region policy optimization (TRPO) [11], and proximal policy optimization (PPO) [12] are some of the policy gradient methods. The gradient estimation method used often suffers from a large fluctuation [13].

The combination of value-based and policy gradient methods has been developed to aggregate the advantages and address the disadvantages of the two methods. This kind of combination has resulted in actor-critic methods. The actor-critic method is a combination of two components: an actor and a critic, both of which can be characterized by Deep Neural Networks. The actor attempts to learn a policy by obtaining feedback from the critic which estimates a value function, and this iterative process continues until the actor converges into an optimal policy. Deep deterministic policy gradient (DDPG) [14], distributed distributional DDPG (D4PG) [15], advantage actor critic (A2C) [16], asynchronous advantage actor critic (A3C) [16], and unsupervised reinforcement and auxiliary learning (UNREAL) [17] are some of the actor-critic methods.

## IV. Intrusion Detection System Using Deep Reinforcement Learning

Taylor et al. [2] present an agent-based anomaly intrusion detection and prevention system using deep reinforcement learning techniques. The proposed system makes use of two agents: the first agent which attacks the network system and the second agent which detects and classifies the attack. The authors have used the Deep Q Learning algorithm which helps the agent detect the attack by estimating the optimal Q-values as shown in Fig. 3. The authors make use of the NSL-KDD [18] dataset and the agent classifies the attack as either normal, DOS, probe, Remote to Local (U2L), or User to Root (U2R) attack. Their experiment states that the attacking agent got a reward of 5 whereas their defending agent got a reward of 95 which indicates that their defending agent was able to detect and classify the attacks made by the first agent efficiently. On further exploring the accuracy of their defending agent on individual categories, it is shown that the defending agent performed the best on U2R with 99% accuracy followed by DoS, Probe, and R2L with 94%, 94%, and 88% respectively. For the normal entries in the dataset, the agent was successfully able to classify around 79% of them.

In [3], Alavizadeh et al. have implemented an intrusion detection method that combines a Q-learning-based reinforcement learning with a deep feed-forward neural network method for network intrusion detection. Their

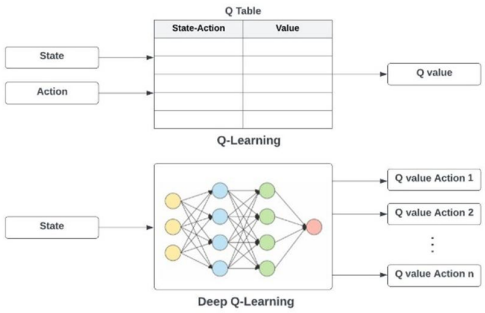


Figure 3. Deep Q-Learning Architecture

proposed Deep Q-Learning (DQL) model provides a continuous self-learning capability for a network environment that can detect network intrusions using a trial-and-error approach to enhance its detection capabilities. They perform fine-tuning of different hyperparameters involved in the DQN model to make it more efficient. The authors state after their extensive experimentation using the NSL-KDD dataset, that the lower discount factor when set as 0.001 with 250 episodes of training yields the best performance results with accuracy of 0.8084 (normal), 0.9237 (DoS), 0.9463 (probe) and 0.8848 (R2L).

Nguyen et al. [19] cover different important aspects involving DRL and its applications in the cyber- security space namely DRL-based security methods for cyber-physical systems, intrusion detection systems, and multiagent DRL-based game theory simulations for defence strategies, etc. The authors of this paper have conducted the survey extensively and made a detailed report of the algorithms used, the action space, state space, and the rewards. The DRL algorithms listed include DQN, DDQN, A3C, TRPO, etc. which provides a comprehensive understanding of the applications of DRL in Cybersecurity.

Lopez-Martin et al. [20] present the results of applying some of the widely applicable Deep Reinforcement Learning models namely Deep Q Network (DQN), Double Deep Q-Network (DDQN), Policy Gradient (PG), and Actor-Critic (AC). The authors apply these DRL techniques to two datasets: NSL-KDD and AWID [21] and show that the best performing model obtained was DDQN. Taking an interesting approach, the authors have made a conceptual alteration of the standard DRL paradigm which is based on the interaction with a live environment and replacing the environment with a sampling function of recorded training intrusions. The authors claim that this unique pseudo-environment, with the sampled training dataset, generates rewards based on detection errors found during the training. Furthermore, they show that Deep Reinforcement Learning, with their model and some parametric changes, can improve the results of intrusion detection in comparison with the current machine learning techniques.

Sethi et al. [22] propose a multi-agent IDS using DQN which is designed as a distributed attack detection platform. The authors explain that the agents work in a coordinated fashion to provide a scalable, fault-tolerant and a guided security system. The proposed system consists of a central IDS and a distributed intrusion detection agent logic as shown in Fig. 4. The central IDS receives messages from agents and gets feedback from the environment about intrusions and communicates the feedback back to the agents. The distributed agents use Deep Q Learning to classify the attack and communicate it to the central IDS for the response. In addition, the authors implemented a denoising autoencoder (DAE) to further enhance the robustness of the system by identifying adversarial (corrupted) attacks from non-adversarial attacks. With this system in place, on the CIC-IDS 2017 dataset [23], the weighted average F1 score was found to be 0.989 with an FPR of 0.0082 whereas, on the NSL-KDD dataset, the weighted average F1 score was 0.978 with FPR of 0.0124. A major limitation of this system is that it does not consider computational and network costs which include delays between the agents in the routers and the central IDS.

In [24], the authors also try to address one of the fundamental problems present in the existing IDSs which is the trade-off between accuracy and false positive rate (FPR). The authors claim that the existing IDSs are unable to provide higher accuracy with low FPR and hence propose a context adaptive IDS that uses multiple independent DRL agents distributed across the network for accurate detection and classification of attacks. The authors too utilize the central IDS and distributed agent logic. Along with the DRL, the system also uses five different classification models like Random Forest (RF), AdaBoost (ADB), Gaussian Naive Bayes (GNB), K-Nearest Neighbours (KNN), and Quadratic Discriminant Analysis (QDA) for fine-grained classification of the attacks. The authors have also used DAE to further enhance the robustness of the system. They have experimented with their system on three benchmark datasets namely NSL-KDD, UNSW-NB15 [25], and AWID, and found that


Figure 4. Central IDS and Distributed Agent Logic

along with DWN, QDA performs well for UNSW-NB15 and AWID datasets with an FPR of 8.4% and 1.6% respectively and either of QDA or GNB for NSLKDD with FPR of 1.59% for QDA and 1.688% for GNB thus creating a balance of high accuracy and low FPR.

Sethi et al. [26] propose a DRL based adaptive cloud IDS architecture that addresses the shortcomings of most of the traditional cloud IDSs that are vulnerable to novel insider attacks and have a non-deterministic nature which makes them unsuitable for cloud-based environments. The proposed architecture performs accurate detection & fine-grained classification of new and complex attacks. The authors have done thorough experimentation using the UNSW-NB15 dataset that shows better accuracy & less FPR compared to the then state-of-the-art IDSs. The proposed system consists of three components i.e., the host network, the agent network, & the administrator network. The host network shows the virtualized environment where different virtual machines are hosted. The agent network comprises the agent nodes that collect network traffic information from various hosts and execute pre-processing and feature selection on these data to extract feature vectors. Using DQN [27] on the input feature vectors, the agent generates results & shares it with the administrator network. The administrator network then uses a voting system to detect whether a feature vector is vulnerable or not and if a feature vector is found vulnerable, the administrator network performs attack type classification. With their proposed adaptive cloud IDS architecture, they were able to achieve an accuracy of 83.80% with an FPR of 2.6%.

When using IDSs as protection against cyber-attacks, the impact of classifying normal samples as attacks (False Positives) or attacks as normal traffic (False Negatives) is varied. To prioritize the absence of one kind of error, Blanco et al. [28] use RL strategies which allow them to build a cost sensitive meta-classifier using a DQN architecture over a Multi-Layer Perceptron (MLP). The authors claim that while the DQN introduces extra effort during the training steps, it does not cause any penalty on the detection system. The authors show the feasibility on two benchmark datasets- UNSW & NSL-KDD, achieving reductions up to 100% in the desired error by changing the rewarding strategies. They have chosen the hyperbolic tangent activation function to grant the convergence of the model. The authors state that, when they set unbalanced negative rewards for the False Positive (FPOS) and False Negative (FNEG) predictions to -10 & -100 respectively (what they call Soft FNEG), they managed to reduce the number of FNEG predictions from 1 to 0 in the validation test & from 9 to 2 in the full test dataset, which is a reduction of 100 and 78%. Furthermore, when they set the rewards to -1 & -100 in the same cases (Hard FNEG) they manage to increase the reduction from 78 to 89%. Hence depending on the application of the system, it is more important for the IDS to focus either on reducing FNEG or on reducing FPOS rather than reducing errors globally.

Bachi et al. [29] address one of the common limitations of traditional IDSs, which is high computational expense, by using a novel approach called SparseIDS after which, the number of consumed packets were reduced by more than three-fourths while keeping the classification accuracy high. The authors show that a shared neural network (NN) can be used for both the classifier and the RL logic which can reduce the need of having additional resources during deployment. The proposed framework consists of three independent neural networks: a classifier (LSTM based supervised classifier), a critic & an actor as shown in Fig. 5. The classifier's goal is to predict correctly if a flow is an attack or not by providing its confidence as output. The critic aims to estimate the future expected average classification performance and the future expected average sparsity that can be achieved at a packet, say n. The actor outputs a probability distribution at packet n, which is sampled to determine the number of packets that should be skipped next. It aims to change the distribution so that the actions which result in a higher reward are chosen more frequently, thereby maximizing the utility function. The

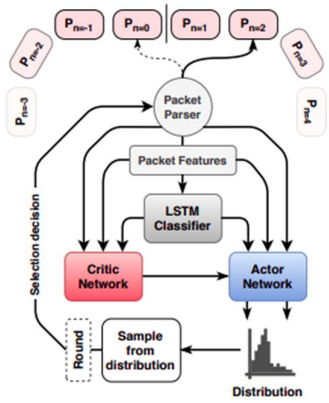


Figure 5. Sparse IDS framework showing the classifier, the actor network and the critic network

authors state that the reward that the agent tries to maximize should ideally be a combination of classification accuracy and the number of packets that could be skipped (sparsity) & they also mention that an operator would be allowed to specify how much accuracy is allowed to be traded-off for achieving higher sparsity. Thus, by opting for an Actor-Critic (AC) approach model rather than a DRL approach based on Deep Q Network, the authors were able to increase computational efficiency of IDS.

## V. Conclusion

In this survey, we have presented an overview of IDS and DRL along with their types. Then, some of the latest implementations of IDSs using DRL techniques have been reviewed to understand their architectures, underlying models, their advantages, and their limitations. We have then, through a comprehensive survey, elucidated the threat posed by increased cyber-attacks & specified the need for robust and autonomous cyber defense systems using advanced techniques like Deep Reinforcement Learning such as Actor-Critic methods, Policy Gradient methods, etc. with pertinent optimization to amalgamate the advantages of the existing IDSs such that the re-engineered system can address the common limitations highlighted in the survey.

## Acknowledgment

The authors wish to thank Ichcha Baliga, Cyber Security Engineer, GE Corporate for her valuable contribution towards this survey.

## References

[1] "The number of new malicious files detected every day increases by 5.2% to 360,000 in 2020", *kaspersky.com*, https://www.kaspersky.com/about/pressreleases/2020_the-number-of-new-malicious-filesdetected-every-day-increases-by-52-to-360000-in-2020 (accessed Feb. 12, 2022)

[2] O. E. Taylor, P. S. Ezekiel, and C.G. Igiri, "Anomaly based intrusion detection/prevention system using deep reinforcement learning algorithm," *Int. Journal of Adv. Research in Computer and Communication Engineering*, Vol. 10, Issue 1, 2021, pp. 58-65

[3] H. Alavizadeh, J. Jang-Jaccard and H. Alavizadeh, "Deep q-learning based reinforcement learning approach for network intrusion detection," in *Computers* 11, 41.

[4] L. H. Yeo, X. Che and S. Lakkaraju, "Understanding modern intrusion detection systems: a survey," *arXiv*, arXiv:1708.07174

[5] S. Othman, T. Nabeel, F. Ba-Alwi and A. Zahary, "Survey on intrusion detection system types," *Int. Journal of Cyber-Security and Digital Forensics*, pp. 444-462.

[6] K. Arulkumaran, M. P. Deisenroth, M. Brundage and A. A. Bharath, "A brief survey of deep reinforcement learning," *IEEE Signal Proc. Magazine*, 34(6)

[7] C. J. Watkins, and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279-292

[8] R. S. Sutton, A. G. Barto, "Introduction to reinforcement learning," *MIT Press Cambridge*, MA, USA.

[9] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3-4, pp. 229-256

[10] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems*, 2000, pp. 1057-1063

[11] J. Schulman, S. Levine, P. Abbeel, M. Jordan and P. Moritz, "Trust Region Policy Optimization," *International Conference on Machine Learning*, 2015, vol. 37, pp. 1889-1897
[12] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint*, arXiv:1707.06347, 2017.
[13] C. Wu, A. Rajeswaran, Y. Duan, V. Kumar, A. M. Bayen, S. Kakade, ... and P. Abbeel, "Variance reduction for policy gradient with action dependent factorized baselines," *arXiv preprint*, arXiv:1803.07246, 2018
[14] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, ... and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint*, arXiv:1509.02971, 2015.
[15] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, A. Muldal, ... and T. Lillicrap, "Distributed distributional deterministic policy gradients," *arXiv preprint*, arXiv:1804.08617, 2018.
[16] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, ... and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in International Conference on Machine Learning, 2016, pp. 1928-1937.
[17] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu, "Reinforcement learning with unsupervised auxiliary tasks," *arXiv preprint*, arXiv:1611.05397, 2016.
[18] Nsl-Kdd dataset, https://www.unb.ca/cic/datasets/nsl.html, (accessed Feb. 25, 2022)
[19] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," in *IEEE Transactions on Neural Networks and Learning Systems*, 2021
[20] M. Lopez-Martin, B. Carro and A. Sanchez-Esguevillas, "Application of deep reinforcement learning to intrusion detection for supervised problems," *Expert Systems with Applications*, Volume 141, 2020
[21] Awid dataset—wireless security datasets project, http://icsdweb. aegean.gr/awid/. (accessed Mar. 22, 2022)
[22] K. Sethi, Y. V. Madhav, R. Kumar, P. Bera, "Attention based multi-agent intrusion detection systems using reinforcement learning," *Journal of Information Security and Applications*, Volume 61, 2021
[23] Intrusion detection evaluation dataset (Cic-Ids2017), https://www.unb.ca/cic/datasets/ids-2017.html (accessed Mar. 10, 2022)
[24] K. Sethi, E. S. Rupesh, R. Kumar, P. Bera and Y. V. Madhav, "A context-aware robust intrusion detection system: a reinforcement learning-based approach," *Int. J. Inf. Secur.* 19, pp. 657–678
[25] The unsw-nb15 dataset description, https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFANB15-Datasets/. (accessed Mar. 10, 2022)
[26] K. Sethi, R. Kumar, N. Prajapati and P. Bera, "Deep reinforcement learning based intrusion detection system for cloud infrastructure," *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*, 2020, pp. 1-6
[27] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou et. al, "Playing atari with deep reinforcement learning," *arXiv*, arXiv:1312.5602
[28] R. Blanco, J. J. Cilla, S. Briongos, P. Malagon and J. M. Moya, "Applying cost-sensitive classifiers with reinforcement learning to IDS," *Intelligent Data Engineering and Automated Learning – IDEAL 2018*, vol 11314, Springer
[29] M. Bachl, F. Meghdouri, J. Fabini and T. Zseby, "SparseIDS: learning packet sampling with reinforcement learning," *2020 IEEE Conference on Communications and Network Security (CNS)*, 2020, pp. 1-9.