An adversarial reinforcement learning based system for cyber security

1st Song Xia
College of Electronic and Information
Wuhan University
Wuhan, Hubei, China
songxia@whu.edu.cn

2nd Meikang Qiu
Department of Computer Science
Texas A&M University-Commerce
Commerce, TX, USA
meikang.qiu@tamuc.edu

3rd Hao Jiang
College of Electronic and Information
Wuhan University
Wuhan, Hubei, China
jh@whu.edu.cn

***Abstract*—In this paper, we proposed a reinforcement learning based system for defending the network users from malicious network traffics. By training two reinforcement learning agents: network attack generation agent and network defense agent, and based on the environment of deep neural networks, this system not only aims to outperforme the traditional machine learning algorithm (such as *CNNs* and *LSTM*), but also can to detect the adversarial example, which is the one of the biggest challenges for current machine learning based intrusion detection system.**

***Index Terms*—adversarial reinforcement learning, adversarial samples, intrusion detection system, deep neural network.**

I. Introduction

With terabits of information stored in the network and and much of this information being confidential, how to detect the malicious attack and protect the security becomes a very important issue in today society. Cyber-security is a measure protecting computer systems, networks, and information from disruption[3]. Recently, with the popularity of machine learning algorithm, such as *SVM*[4], deep neural network (*DNN*)[5], and reinforcement learning (*RL*)[6], the traditional intrusion detection methods are replaced by new machine learning based methods. Although those new methods have made a great progress in this field, there are still many challenges need to be overcome. In the *DNN* based system, the authors can detect the malicious attack from outside successfully with a rate over 99%. However, this deep learning based system is extremely vulnerable to the adversarial example. In *SVM* and *RL* based systems, although making a better immunity to the adversarial example, the detection performance is not satisfying enough.

Aiming to solve the challenges mentioned above, we proposed a reinforcement learning based system, combined with deep neural network environment. The main contributions of this work will be as follows:

1. We proposed a brand new adversarial reinforcement learning based intrusion detection system for cyber security.

2. This work introduces the trained CNNs as the interacting environment for attack generation agent, which aims to generate adversarial samples to beat the trained CNNs.

3. In theory, this system can outperform the traditional deep neural network and detect the adversarial samples.

The rest of the paper is oranginzed as follows: section II introduce the related work about the recently proposed cyber-security systems; section III introduce the framework of our system; in section IV, we give a case study to prove the system and section V concludes our work and gives a prospect.

II. Related work

Many scholars such as K. Gai, J. Wang and M. Qiu et al. [8][9] have done numerous valuable work in the Cyber-security field. However, with the diversification of cyber-attacks, traditional defense products cannot fully guarantee the security of the network. AlFayyad et al. evaluated the performance of personal firewall systems but it indicated that personal firewalls were poorly used which leaded to vulnerabilities in security [7]. Intrusion detection systems are similar to burglar alarms which could look for suspicious activity and alert the system and network administrators [10], but traditional *IDS* might miss some unknown threats [11].

The *IDS* based on machine learning, such as deeping learning and reinforcement learning have achieved a better performance in this field. Elike Hodo et al. [12], established a neural network based threat detection system to protect the Internet of things network and achieved relatively high accuracy. However this system is ad-hoc for *DOS* threat. In [13], G. Caminero et al. proposed an adversarial environment reinforcement learning algorithm for intrusion detection and although gave a comprehensive comparison with current deep learning based algorithm. However, due to the learning environment and simple reword function, in their experiment, the accuracy perdiction rate in around 90%, much lower than many machine based algorithms.

III. System design and an example

As shown in Fig. 1 and Fig. 2, our model main includes two sub-systems: attack generation system and adversarial reinforcement learning system. The attack generation system is based on the trained *CNNs* environment and used to generate the adversarial samples. The adversarial reinforcement system composes of two reinforcement learning agents: trained attack generation agent (based on deep Q-learning[1]) and an innocent defense agent (based on deep deterministic policy gradient[2]). Here we will give the details about these two sub-systems.

978-1-7281-5506-7/19/$31.00 ©2019 IEEE
DOI 10.1109/SmartCloud.2019.00046

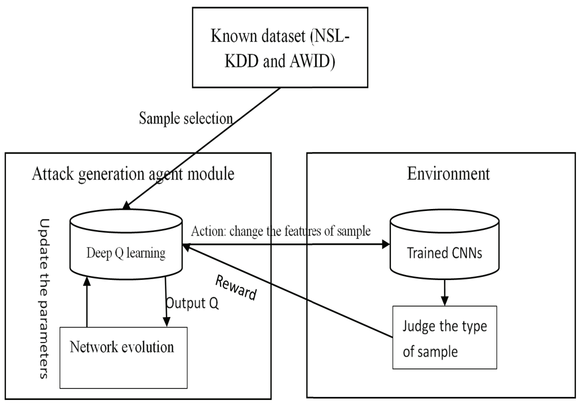


Fig. 1. the structure of attack generation system.

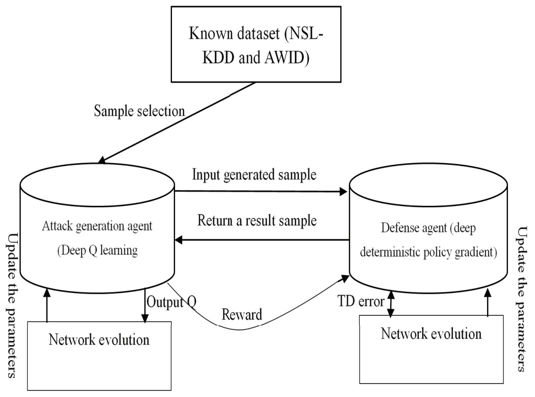


Fig. 2. the structure of adversarial reinforcement learning system.

A. Attack generation system

In this system, we choose deep Q-learning algorithm to train the agent. Deep Q-learning is a method based on Q-learning and neural network. In Q-learning, the Q value is updated by the formula (1). s means the state of the agent and a represents the action that agent does in current state. s' means the next state of the agent and a' represents the action that agent chooses in next state. r is the reward given to the agent, when it enters in the next state. Q is the motivation of the action a in state s , which means the agent will choose the action with the greatest Q. γ and α are the attenuation factors.

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)] \quad (1)$$

From the above formula, the value of $Q(s,a)$ is updated by the max Q value in the next state and the current Q value. In deep Q-learning, it introduces two neural networks with the same structure: an evaluation network and a targeted network. The evaluation network has a fast updated speed and is used to estimate the value of $\hat{Q}$, while the targeted

```
**Algorithm 1: deep Q-learning with experience replay.**
Initialize replay memory $D$ to capacity $N$
Initialize action-value function $Q$ with random weights $\theta$
Initialize target action-value function $\hat{Q}$ with weights $\theta^{-} = \theta$
For episode = 1, $M$ do
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  For $t$ = 1,T do
    With probability $\varepsilon$ select a random action $a_t$
    otherwise select $a_t = \mathrm{argmax}_a Q(\phi(s_t), a; \theta)$
    Execute action $a_t$ in emulator and observe reward $r_t$ and image $x_{t+1}$
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $D$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $D$
    Set $y_j = \begin{cases} r_j & \text{if episode terminates at step j+1} \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^{-}) & \text{otherwise} \end{cases}$
    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ with respect to the
    network parameters $\theta$
    Every $C$ steps reset $\hat{Q} = Q$
  End For
End For
```

```
**Algorithm 2 central control algorithm for the system**
Initialize set the structure of evaluation network to equal with CNNs'
Initialize the state $s$ and action $a$ with value 0
For episode in max range
  Randomly select one sample from the dataset
  Set $s$ = features of current sample
  With the probability of $\varepsilon$ to select a random $a$
  Otherwise choose an $a$ with the max Q
  Change the features of the sample according the action $a$
  Pass the modified sample to the trained CNNs
  Return the classification result $l$ in CNNs
  Compare $l$ with the label
  If true
     Set the $r = -1$
  Else
     Set the $r = +1$
  Return the $r$
  Call the algorithm 1
  Update the $\theta$
```

network gets a slow updated speed and is used to estimate the value of Q . The algorithm of Deep Q learning is shown in algorithm 1. In this algorithm, θ is the parameter for targeted network and θ^{-} is the parameter for evaluation network. Both of them are used to calculate the value of Q . The deviation between the $r_j + \gamma \max_{a'} Q(\phi_{j+1}, a', \theta^{-})$ and $Q(\phi_j, a_j, \theta)$ is back propagated to the evaluation network and used to update the θ^{-}. To cooperate different modules in this system, we propose a scheme algorithm to control the work flow, which is shown in algorithm 2.

In this algorithm, we first set the neural networks in the Deep Q-learning and *CNNs* have the same structure, in which we can exploit the same loss function to generate the adversarial samples more effectively, and then, the features of sample is assigned as the state of agent. The *CNNs* is proposed in my previous work, which gets the detection accuracy over 99%. Each action of agent will change the features of the

Algorithm 3 adversarial reinforcement learning algorithm

```
Initialize set the two agents with DQL and DDPG respectively
Initialize the state $s$ and action $a$ with value 0
Initialize the state $s'$ and action $a'$ with value 0
For episode in max range
    Randomly select one sample from the dataset
    Set $s$ = features of current sample
    With the probability of $\varepsilon$ to select a random $a$
    Otherwise choose an $a$ with the max Q
    Change the features of the sample according the action $a$
    Pass the modified sample to the trained DDPG
    Return the classification result $l$ in DDPG
    Compare $l$ with the label
    If true
        Set the $r = -1$
        Set the $r' = +1$
    Else
        Set the $r = +1$
        Set the $r' = -1$
    Return the $r$
    Update the $\theta$
```

samples to mislead the *CNNs*. From the algorithm, we can know that, if the *CNNs* network get the wrong classification result, the agent will get a positive reward, otherwise it will get a negative reward. After training, the attack generation agents can generate the adversarial samples and defeat the original *CNNs* easily.

B. Adversarial reinforcement learning system

In this system, we choose two adversarial agents: attack generation agent and defense agent, to fight with each other. To avoid the agents getting the same training result, we assign different reinforcement algorithm for them. The attack generation agent, with a deep Q-learning algorithm, has been trained in the first system and the defense agent, with a deep deterministic policy gradient algorithm, is innocent. The algorithm of this adversarial reinforcement learning system is shown in algorithm 3.

In this algorithm, the adversarial sample generation agent with a DQN is trained in the first subsystem and to improve the adversarial training efficiency, we choose a different algorithm for the defense agent. Then, the features of sample is assigned as the state of sample generation agent. With a previous experience, this agent can modify the chosen sample to make it hard being detected. The action of defense agent is to give a predicted label for the current sample. From the algorithm, we can know that, if the DDPG agent gets the right classification result, the defense agent will get a positive reward, otherwise it will get a negative reward. After training, the defense generation agent can gain experience from the adversarial training and improve its efficiency.

IV. Case study

In this section, we will give a case to support the proposed system. The structure of this system and parameters are shown in figure 3. In this case, a sample with four parameters is selected from the dataset, and then, passed through the DQN. After that, the action of the agent is decided by the Q estimation. The modified sample then will be passed to the neural network to get a corresponding label. The classification result will background to the DQN algorithm to update the network parameters. The calculation process is shown as follows:

$$\begin{aligned} z^{[1]} &= W^{[1]}x + b^{[1]} \\ a^{[1]} &= \sigma(z^{[1]}) \\ z^{[2]} &= W^{[2]}a^{[1]} + b^{[2]} \\ a^{[2]} &= \sigma(z^{[2]}) \end{aligned} \tag{2}$$

$$\sigma = \begin{bmatrix} 1,0,0 \\ 0,1,0 \\ 0,0,0 \end{bmatrix} \tag{3}$$

Before training, the parameters in the DQN are all nil. As a result, the action of the agent will not change the value of the input sample. The sample with a value (0.0 1.0 0.6) is passed to the neural network. Through the eq. 2, we get the result y = 0.67. In this case, the classification result is 1, the same as the original label, thus returning a -1 reward. We assume that after iteration, the trained parameters for DQN is in table 1. Through the eq.2, and adjust the σ as eq.3, we get the result = (-0.1 +0.2 -0.1). Each value means the extent of modification value for corresponding parameter in the input sample; the value of sample is modified from (0.0 1.0 0.6) to (-0.1 1.2 0.5). The modified sample with value of (-0.1 1.2 0.5) is passed to neural network again and through the eq.1, we get the result is 0.48, which is shown in table 2. In this case, the trained neural network is misled by the adversarial generation agent successfully.

TABLE I
PARAMETERS OF TRAINED *DQN*

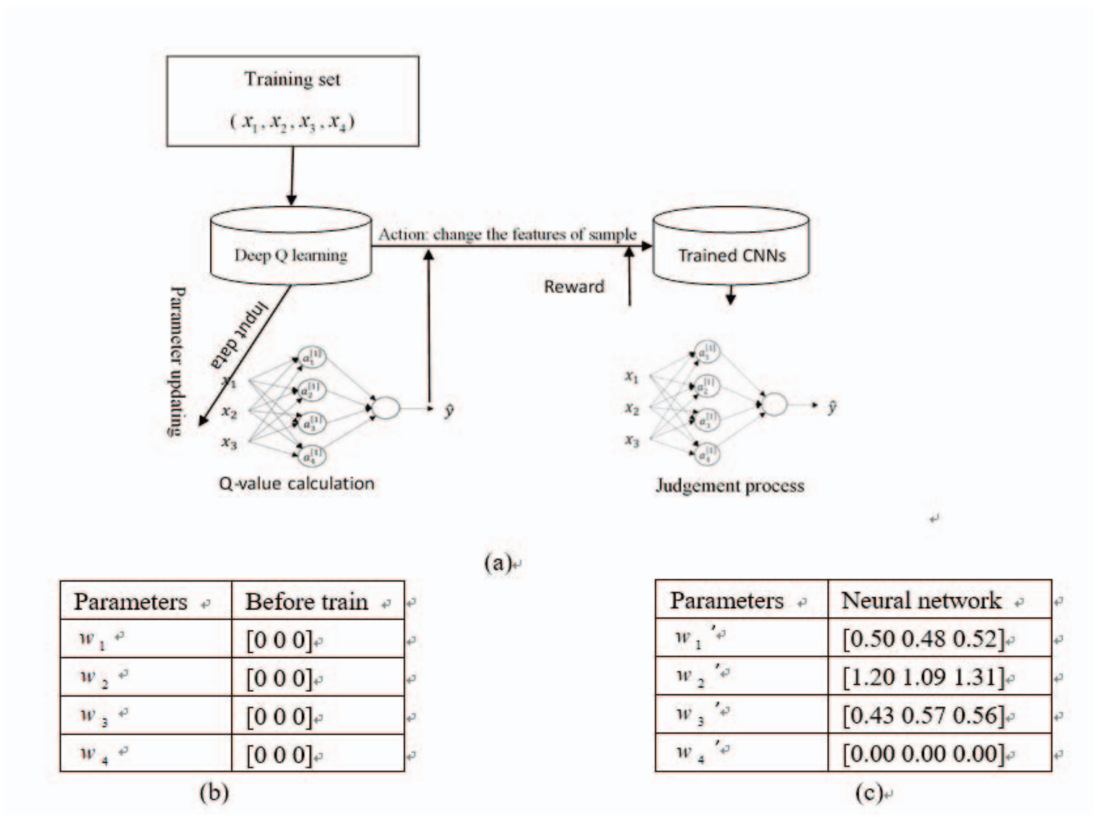
Parameters	***After train***
w_1	[0.063 0.058 0.060]
w_2	[0.125 0.130 0.122]
w_3	[0.060 0.055 0.060]
w_4	[0.000 0.000 0.000]

TABLE II
THE RESULT IN CASE STUDY *DQN*

Parameters	***Before modification***	***After modification***
x_1	0.0	-0.1
x_2	1.0	1.2
x_3	0.6	0.5
Label	1.0	1.0
Prediction result	1.0	0.0

V. Conclusion

In this paper, we proposed a reinforcement learning based system to handle the adverasial samples, which were trouble-some for DL based method. By training two adversarial agents:



Parameters	Before train
w_1	[0 0 0]
w_2	[0 0 0]
w_3	[0 0 0]
w_4	[0 0 0]

Parameters	Neural network
w_1'	[0.50 0.48 0.52]
w_2'	[1.20 1.09 1.31]
w_3'	[0.43 0.57 0.56]
w_4'	[0.00 0.00 0.00]

Fig. 3. the parameters and structure.(a)the system sturcture (b)the parameters of innocent *DQN* (c)the parameter of train neural network

network attack generation agent and network defense agent, and based on the environment of deep neural networks, this system aimed to propose a novel adversarail sample generation and defense method. The future work includes conducting the experiment in the NSL-KDD and AWID dataset and making a comprison with extant DL methods.

ACKNOWLEDGMENT

This work is supported by the Advanced network and Intelligent Information Service Institute of Wuhan University, Zhongshan City (No.2017F1FC0001) and the Intelligent front-end recognition technology based on deep learning and its application in industry (No.2019AG012).

REFERENCES

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabisn, “Human-level control through deep reinforcement learning,” Nature. vol. 518, pp. 529–533, Feb. 2015.

[2] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver,D. Wierstra,“ Continuous control with deep reinforcement learning,“ arXiv preprint arXiv:1509.02971

[3] K. Thakur, M. Qiu, K. Gai, M. Ali. “An investigation on cyber security threats and security models”. In: 2015 IEEE 2nd International Conference on Cyber Security and Cloud Computing, pp. 307-311. IEEE, New York, Nov. 2015.

[4] Ross, K.: SQL injection detection using machine learning techniques and multiple data sources. Master’s Projects. 650. DOI: https://doi.org/10.31979/etd.zknb-4z36

[5] W. Li, P. Yi, Y. Wu, L. Pan, and J. Li, “A new intrusion detection system based on KNN classification algorithm in wireless sensor network”. Journal of Electrical and Computer Engineering, vol.2014, no. 240217, pp. 1-9, Jun. 2014.

[6] C. Kim, J. Park,“Designing online network intrusion detection using deep auto-encoder Q-learning“Computers & Electrical Engineering Vol. 79, 106460, Oct. 2019.

[7] B. Alfayyadh, J. Ponting, M. Alzomai, A. Jøsang. “Vulnerabilities in personal firewalls caused by poor security usability”. In: 2010 IEEE International Conference on Information Theory and Information Security, pp. 682-688. Beijing, Jan.2011.

[8] J. Wang, M. Qiu, B. Guo, “Enabling real-time information service on telehealth system over cloud-based big data platform”.Journal of Systems Architecture, Vol. 72, pp. 69-79, Jan. 2017.

[9] K. Gai, M. Qiu, Z. Xiong, M. Liu, “Privacy-preserving multi-channel communication in Edge-of-Things“Future Generation Computer Systems Vol. 85, pp. 190-20, Aug. 2018.

[10] F. Rietta, “Application layer intrusion detection for SQL injection”. In: ACM-SE 44 Proceedings of the 44th annual Southeast regional conference, pp. 531-536. Florida, Mar.2016.

[11] R. Bace, P. Mell. “NIST Special Publication on Intrusion Detection Systems”. Technical Report, pp.800-31, Jan.2001.

[12] E. Hodo, X. Bellekens, A. Hamilton, P. Dubouilh, L. Ephraim, T. Christos, A. Robert. “Threat analysis of IoT networks using artificial neural network intrusion detection system“. 2016 International Symposium on Networks. Computers and Communications (ISNCC). pp. 1-6, Yasmine, May. 2016.

[13] G. Caminero, M. Lopez-Martin, B. Carro, “Adversarial environment reinforcement learning algorithm for intrusion detection“, Computer Networks, vol. 159, pp. 96-109, Aug. 2019.