Automated Post-Breach Penetration Testing through Reinforcement Learning

Sujita Chaudhary, Austin O'Brien, Shengjie Xu

***Abstract*—Predicting cyber attacks to networks is ever present challenges in the security domain. Rapid growth of Artificial Intelligence (AI) has made this even more challenging as machine learning algorithms are now used to attack such systems while defense systems continue to protect them with traditional approaches. Penetration testing (pentest) has long been one way to prevent security breaches by mimicking black hat hackers to expose possible exploits and vulnerabilities. Using trained machine learning agents to automate this process is an important research area that still needs to be explored. The objective of this paper is to apply machine learning in the post-exploitation phase of penetration testing to assess the vulnerability of the system and hence, contribute to the automation process of penetration testing. We train the agent using reinforcement learning by providing an appropriate environment to explore a compromised network and find sensitive files. By utilizing several different network environments during training, we hope to generalize our agent as much as possible, allowing for more widespread application. Extended research may include training our agent for further lateral exploration and exploitation in the system.**

I. Introduction

Penetration testing (or vulnerability assessment) is a vulnerability identification and exploitation process. As one of the key methods used by organizations to strengthen the defense of their systems against cyber threats, it assesses and evaluates the security of digital assets by planning, generating and executing all possible attacks that can exploit existing vulnerabilities [1]. With increasing cyber attacks day by day, pentest is becoming even more crucial and troublesome at the same time, manifesting the need for intelligence oriented pentest.

Pentest has long been one way to prevent security breaches by mimicking black hat hackers to expose possible exploits and vulnerabilities. However, use of traditional approaches to defend systems is more resource and time consuming with high chances of human error caused by repetitive tasks. Using trained machine learning agents to automate this process is an important area of research that still needs to be conducted. The need for the real-time identification of exploitable vulnerabilities adds to the number of potential research challenges and opportunities in modern AI.

In this research, we aim to apply a reinforcement learning algorithm in the post-exploitation phase of pentest in order to assess the vulnerability of the system. We train the agent using reinforcement learning by providing an appropriate environment to explore a compromised network and find sensitive files. By utilizing several different network environments during training, we hope to generalize our agent as much as possible, allowing for more widespread applications.

S. Chaudhary, A. O'Brien, and S. Xu are with The Beacom College of Computer and Cyber Sciences, Dakota State University, Madison, SD 57042. E-mail: sujita.chaudhary@trojans.dsu.edu, austin.obrien@dsu.edu, shengjie.xu@dsu.edu.

II. Reinforcement Learning for Penetration Testing

Researchers have analyzed the impact of pentest and studied different approaches to enhance them. The ability of reinforcement learning to enable an agent to learn in an interactive environment by trial and error approach using feedback from its own actions and experiences makes it more suitable for this work than other general machine learning algorithms.

A. Previous Studies in Reinforcement Learning

In [2], the authors have conducted research on vulnerability assessment, and pointed out different methods used for those studies and comparison between them. The paper also presents potential research direction in using AI for penetration testing. In [1], the authors proposed a hypothesis that reinforcement learning can be used to enhance pentest. They modeled a system using partially observable Markov Decision Process (POMDP) and asserted that this approach allows intelligent and autonomous pentest. Researchers also proposed a general framework called DQEAF using deep Q-network to evade anti-malware engines [3], and an architecture called DQFSA on Deep Q-learning based Feature Selection Architecture [4].

B. Limitations of Previous Studies

Despite the use of automated tools, current pentest practice is becoming complex, repetitive and resource consuming [1]. Deep Exploit, a pentest tool using Metasploit framework also works as a fully automated penetration test tool, however, it is not fully automated for a complete penetration test. At the post-exploitation phase of pentest, there is still a huge gap to cover with machine learning algorithms. It is rare to find studies using reinforcement learning for the automation in the post-breach penetration testing which makes this research novel.

III. Proposed Study on Automated Post-Breach Penetration Testing through Reinforcement Learning

Reinforcement learning, a subset of machine learning, analyzes actions taken by a software agent in an environment in order to maximize a reward. Fig. 1 presents the general architecture of reinforcement learning. In recent years, methods using deep Q-learning networks have been successful in

978-1-7281-4760-4/20/$31.00 ©2020 IEEE

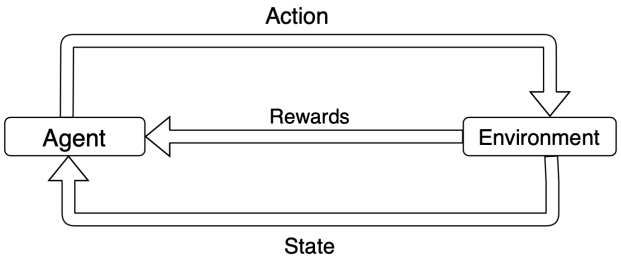

Fig. 1: General Architecture of Reinforcement Learning.

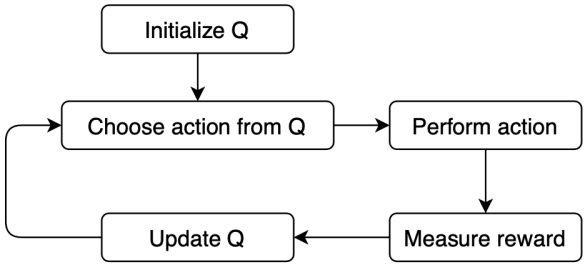

Fig. 2: Work Flow of Q-learning.

human level control of systems, such as game-playing [4]. Considering the rare research studies adopted reinforcement learning in the post-exploitation phase of pentest, a model trained using Deep Q-learning is proposed for the post-breach of the network.

A. Training Agent using Deep Q-Learning

Deep Q-learning is a model-free approach which can be used for building a self-exploring agent. Fig. 2 presents the work flow of Q-learning. The method uses neural networks to estimate Q values by feeding initial state into the network and returning the Q-values of all possible actions as output. The Q value is calculated as:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_{\alpha} Q(S_{t+1}, a) - Q(S_t, A_t)] \quad (1)$$

where α is the learning rate that determines the extent to which newly acquired information overrides the old one.

Q-Learning is a good candidate as our reinforcement training algorithm due to its performance when the time steps between rewards are scarce. In our research, we will train a Deep Q Network with TF Agents, on top of the TensorFlow library. The trained agent will then be placed in Linux and Windows servers on virtual machines to explore compromised networks and find sensitive files, just as would occur during a penetration test. The training of our agent will be on the modified networks used in cyber defense competitions which provide environments conducive to our task. The dataset are the files over those networks such as password, shadows, configurations and so on. Based on the exploration and exploitation performance of the agent, scores will be provided as reward in order to train the agent.

The actions available to our agent will comprise of a finite set of terminal commands, with placeholders that will be taken from the environment. An example command could be:

```
> cd [placeholder]
```

Here, we can see that the placeholder will be a file directory scanned from the current environment. Using a softmax function for the neural net output, we will determine which actions to take based on maximized Q-value estimates, while still allowing for agent exploration.

B. Preliminary Performance Evaluation

We have written a directory traversal script using python, which is run through the command line to find files in a specific directory. We have tested the script to find configurations and log files over both Windows and Linux servers which is set as a baseline for files exploitation using our trained agent. Some of those files are listed in table I that will be considered for reward upon exploitation and corresponding actions to be taken.

From the agent created using randomized policy, we will calculate the loss between Q-value estimates and their true values, along with the maximum reward obtained. We will then compare these values to that of our trained agent's policy.

Server	File Type	File Name	User	Action
Windows	Employee information	data.csv	Admin	Exploit
Linux	Password	passwd	Admin	Exploit
Linux	Shadow	shadow	Root	Report back the directory information

TABLE I: Example of Files Granting Rewards.

IV. Conclusion and Future Work

In this paper, we discuss the limitations of current penetration testing practices and the need of automation mechanisms in such tradition. We propose an idea to automate the post exploit phase of penetration testing using reinforcement learning and present our current environment for training the agent. In the future, we will be implementing this idea by training the agent and devising other rewards and goals that will help generalize our model to a broader set of networks.

References

[1] M. C. Ghanem and T. M. Chen, "Reinforcement Learning for Intelligent Penetration Testing," *Proceedings of 2018 Second World Conference on Smart Trends in Systems, Security and Sustainability*, London, UK, 2018.

[2] D. R. McKinnel, T. Dargahi, A. Dehghantanha, and K.K. R. Choo, "A systematic literature review and meta-analysis on artificial intelligence in penetration testing and vulnerability assessment," *Computers & Electrical Engineering*, vol. 75, pp. 175–188, 2019.

[3] Z. Fang, J. Wang, B. Li, S. Wu, Y. Zhou and H. Huang, "Evading Anti-Malware Engines With Deep Reinforcement Learning," *IEEE Access*, vol. 7, pp. 48867-48879, 2019.

[4] Z. Fang, J. Wang, J. Geng and X. Kan, "Feature Selection for Malware Detection Based on Reinforcement Learning," *IEEE Access*, vol. 7, pp. 176177-176187, 2019.