CSE-8713 Advanced Cyber Operations Team 1 CybORG Report

MATTHEW JONES and MORGAN REECE

1 PROJECT INTRODUCTION

The course project for CSE-8713 Advanced Network Operations at Mississippi State University consists of utilizing an autonomous agent to defend a network against malicious attackers. The project is to train autonomous agents utilizing reinforcement learning enabling network protection empowered by artificial intelligence.

2 BACKGROUND REVIEW/RESEARCH

Autonomous Cyber Operations (ACO) is a technology that utilizes Artificial Intelligence (AI), and specifically reinforcement learning (RL). The objective of RL is to use data sets created dynamically during 'training' runs of the agents within the environment. This is different from supervised learning which uses a fixed dataset to train the agents.[4] Our team settled on using the Stable-Baseline3 library for RL. In the evaluation of algorithms within the Stable-Baseline3 library, we selected the Advantage Actor-Critic (A2C) algorithm. A2C is a deterministic, synchronous implementation of the RL algorithm that updates only after each actor finishes its segment. [2] Benefits were seen with A2C in that it is a hybrid method of RL, combining value-based and policy-based methodologies. The benefits of this hybrid method helps to overcome the challenges of the value-based(discretization) and policy-based (high variance) algorithms, but the negative of A2C is that it lengthens the run time. There are variants to the Actor-Critic algorithms such as Asynchronous Advantage Actor-Critic (A3C) which uses multiple agents and multiple environments run in parallel to reduce execution time. [2]

3 PROJECT SPECIFICATION

The project scenario is based on the movie "The Princess Bride", where the war between two countries, Florin and Guilder, is used as the back drop of a love story. The project

specification outlines that Guilder is suspected of directing cyber attacks on Florin's network infrastructure to ultimately disrupt the manufacture of a new weapons system. Our company was contracted by Florin to defend their computer network using the autonomous defense agents. The execution scenario is that Florin defense agents receive alerts and then address the attackers that are found. Attackers are analysed to determine the attack methodology and the agents utilize different protective actions. [5]

4 PROBLEM ANALYSIS

The problem being addressed in this project was to create a blue agent that could respond to the actions of a red agent for a given scenario using machine learning. In the given scenario, the network is divided into three subnets as demonstrated in Figure 1. The first subnet is comprised of non-critical user hosts while the second subnet contains enterprise servers. The final subnet consists of three user hosts and a critical operational server. The purpose of the blue agent is to prevent or delay the red agent in its goal to compromise the critical operational server. The goal of the red agent, however, is to disrupt the services of the operational server for as long as possible. [1]

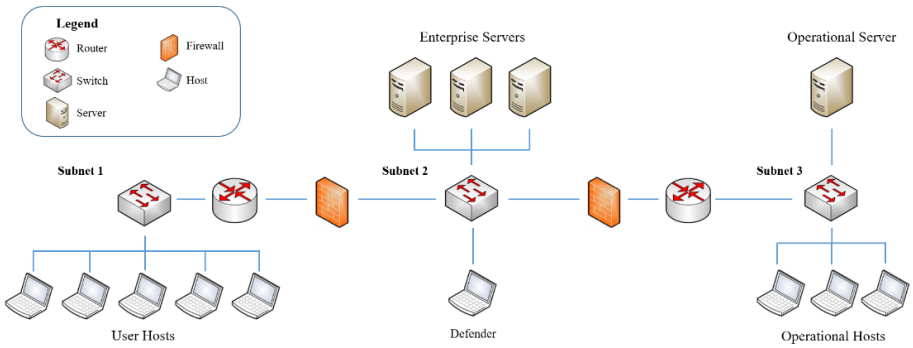

Fig. 1. Scenario Network Diagram [1]

5 SOLUTION DESIGN

To complete the project, our team utilized a reinforcement learning (RL) multi-layer perception (MLP) policy, a feedforward artifical neural network that is used for giving input to a model utilizing backpropogation for training, provided through the Python Stable-Baselines3 library. [3] We then trained the blue agent over two thousand timesteps

before observing how the agent reacts to the actions of the red agent. We assessed our agent using "Scenario 2" provided in the project documentation.

The custom agent file, Team1Agent.py, consists of a single class with five functions and was the only file created or significantly altered within the project. The first function, train, trains the agent as detailed in the paragraph above using a multilayer perception policy over two thousand timesteps. The second function, get_action, retrieves the current directory, loads the provided scenario, creates the CybORG object, and sends that object, or agent, to the train function.

The third and fourth functions, end_episode and set_initial_values respectively, are not utilized, however, they are required for the successful execution of the program. Both functions have pass statements within. The final function is the __init__ function which simply loads the model that was created and starts the assessment.

6 IMPLEMENTATION, TESTING, AND EVALUATION

To evaluate how the blue agent reacted to the actions of the red agent, we utilized the provided evaluation.py python file. The only change made to this file was on line 13 to point the script to our custom agent. This python script loads the agent it is provided and requests user input for the name, team, and the name of the technique utilized. These variables are stored in the output file along with the assessment. After the variables are provided, the program loads the custom blue agent and begins training. Once the training is complete, the evaluation begins. [1]

The assessment is conducted with three distinct red agents with a fixed number of steps representing a fixed period. Within the scenario, the red agent begins with an initial foothold of access into a user machine within the first subnet. The red agent can then select actions to conduct reconnaissance of hosts within the second subnet to exploit one using privilege escalation. The first agent, B_lineagent, has prior knowledge of the network and moves directly to the operational server, while the RedMeanderAgent has no prior knowledge and explores the network one subnet at a time. The RedMeanderAgent will attempt to compromise all hosts in one subnet before moving to the next. The third agent is the SleepAgent which does not act offensively. [1]

The custom blue agent is evaluated against each red agent for 30, 50, and 100 steps resulting in nine assessments. The blue agent score is deducted for the successful actions of

the red agent. For hosts within the first subnet and third subnet, except for the operational server, the blue agent's score is decreased by 0.1 points for each turn that the red agent maintains access to the system. The score is deducted by one point for enterprise servers in the second subnet and the operational server in the second subnet. For each turn the red agent impacts the operational server, the blue agent has ten points deducted, however, if the blue agent restores any hosts, a single point is removed. [1]

7 SUMMARY/CONCLUSIONS

The custom agent created by Team 1 was able to successfully defend the network from the red agent! Against the B_lineagent, the Team1Agent scored -165.88 with a standard deviation of 77.71, -375.53 with a standard deviation of 150.9, and -858.37 with a standard deviation of 326.9 over 30, 50, and 100 steps, respectively. The blue agent scored significantly better against the RedMeanderAgent with no prior knowledge of the network resulting in scores of -31.2 with a standard deviation of 9.68, -185.4 with a standard deviation of 89.88, and -696.83 with a standard deviation of 281.92 over 30, 50, and 100 steps, respectively. Against the SleepAgent, the blue agent scored -1.87 with a standard deviation of 1.25, -2.9 with a standard deviation of 1.48, and -5.55 with a standard deviation of 2.23 over the same number of steps as the previous two agents. In our opinion, our custom blue agent successfully thwarted the red agent in many aspects by utilizing a reinforcement learning (RL) multilayer perception (MLP) policy.

REFERENCES

[1] CAGE. 2022. TTCP CAGE Challenge 2. https://github.com/cage-challenge/cage-challenge-2.

[2] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. https://doi.org/10.48550/arxiv.1602.01783. (2 2016).

[3] Antonin RAFFIN. 2021. Policy Networks. https://stable-baselines.readthedocs.io/en/master/modules/policies.html.

[4] Antonin RAFFIN. 2022. Stable-Baselines3 Docs - Reliable Reinforcement Learning Implementations. https://stable-baselines3.readthedocs.io/en/master/index.html.

[5] Maxwell Standen, Martin Lucas, David Bowman, Toby J Richer, Junae Kim, and Damian Marriott. 2021. CybORG: A gym for the development of autonomous cyber agents.