university of groningen

Multi-Agent Reinforcement Learning for Cyber Defence

Author: Manos Savvides
First Supervisor: Dr. F. Turkmen
Second Supervisor: R. Fernandes Cunha

Table of Contents

Introduction and Motivation

- Cyberattacks have surged in recent years, costing the global economy over \$1 trillion in 2020 (which is a **50%** increase since 2018)[1].
- **Attackers** increasingly use **AI-driven automation** and adaptive strategies, outpacing traditional, reactive defense methods.
- **Reinforcement Learning** (RL) is well-suited for cyber defense due to its ability to model sequential decision-making problems, enabling **real-time threat responses**.

Introduction and Motivation

- This study explores RL techniques in a **simulated environment**.
- The aim is to **identify** the **most effective methods** to prevent these attacks.
- Research Questions:

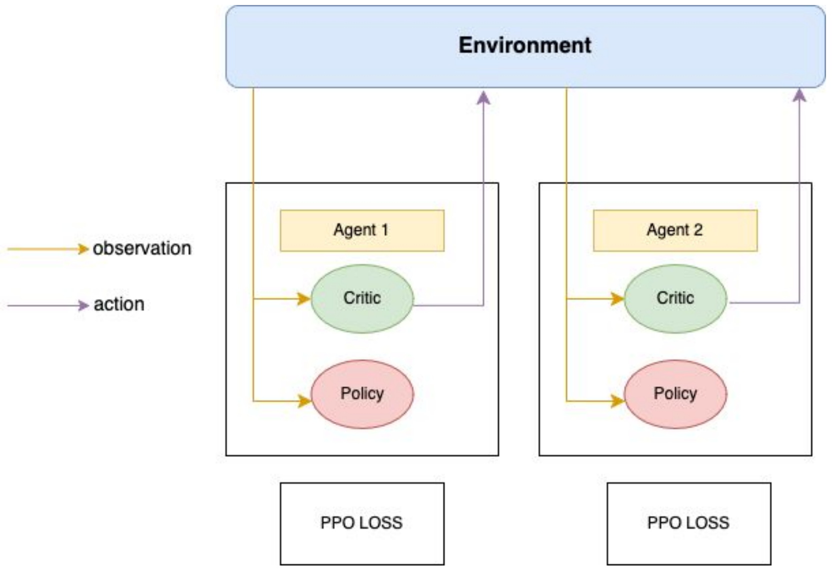
  1. The **optimal hyperparameters** for each algorithm (IPPO and MAPPO) and subsequently evaluate their comparative performance to determine the most effective Proximal Policy Optimization (PPO) approach.

  2. Potential **advantages of hierarchical MARL algorithms** in cyber defence contexts. Specifically, the hierarchical MARL approach will utilize IPPO as its foundational method.

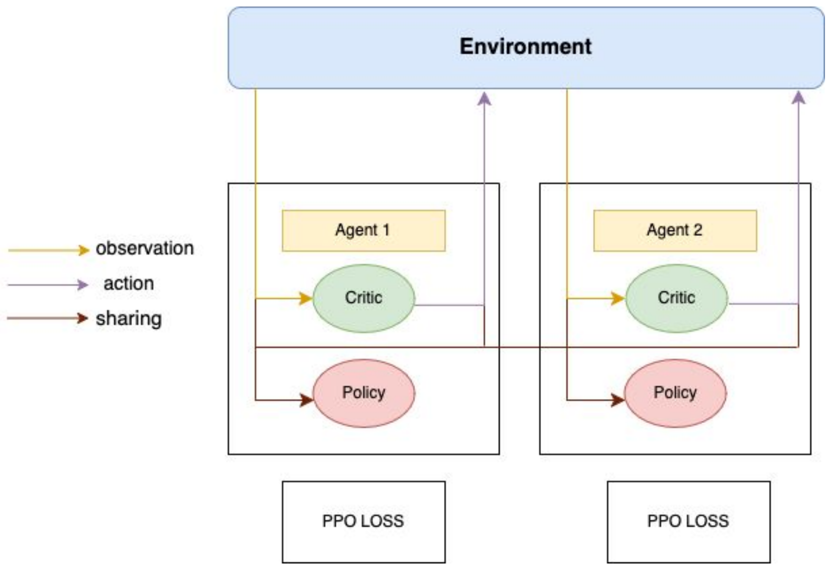
Background

- Building **on the Master's thesis of Davide Rigoni**, this research investigates **hyperparameter optimization** for the top-performing algorithms[2].

Background

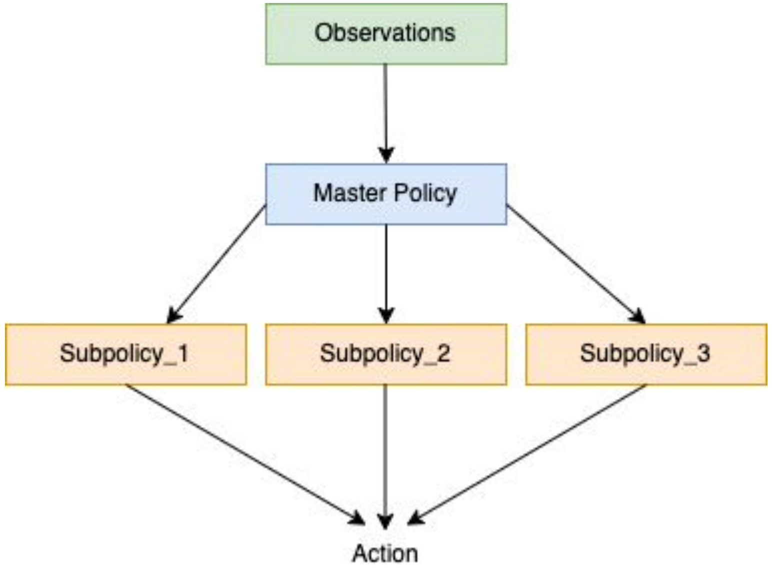


IPPO [4]



MAPPO [5]

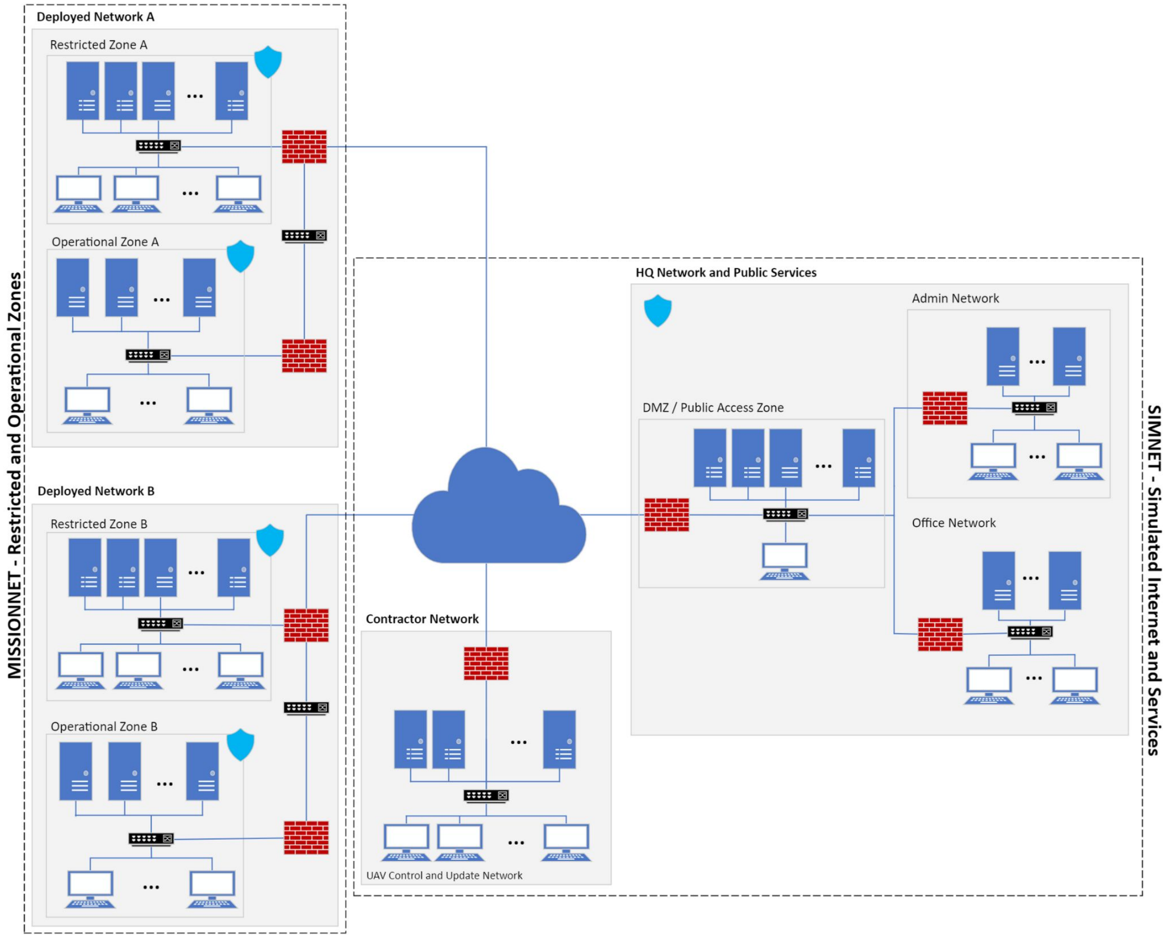
Background



HMARL

Problem Definition

- Cage Challenge 4[6].
- 4 small networks.
- **3 phases** in the simulation.
- Objective is to **prevent red agents from disrupting the green agents** as they carry out their tasks.
- Train **blue agents to take optimal actions** that safeguard the green agents' operations.



https://github.com/cage-challenge/cage-challenge-4/tree/main

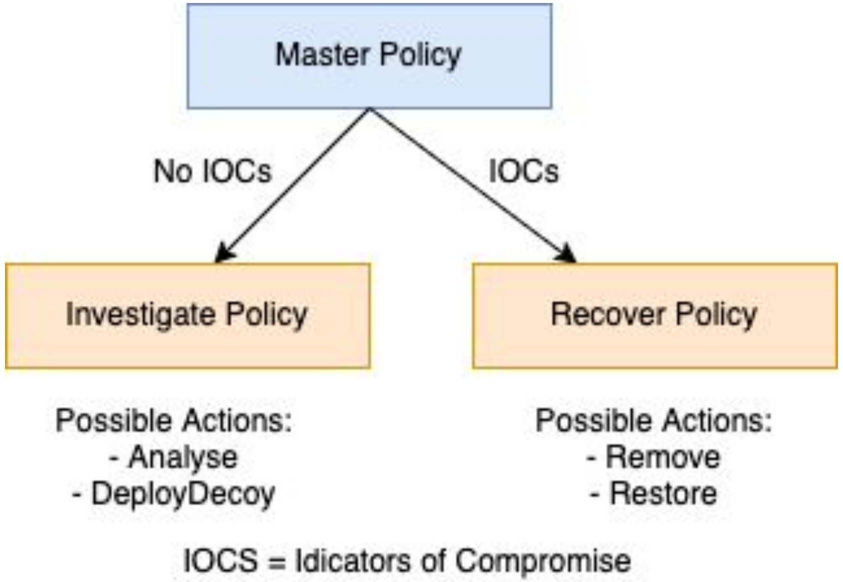
Methodology

- Hyperparameter tuning is critical for PPO performance; we focused on **clip ratio, batch size, minibatch size, and learning rate.**
- Used **Optuna library** with Tree-Structured Parzen Estimator (TPE)[7] sampler for efficient Bayesian optimization, leveraging correlations between parameters.
- Applied **ASHA**[8] for early stopping, **improving tuning speed** in parallel experiments.

Methodology

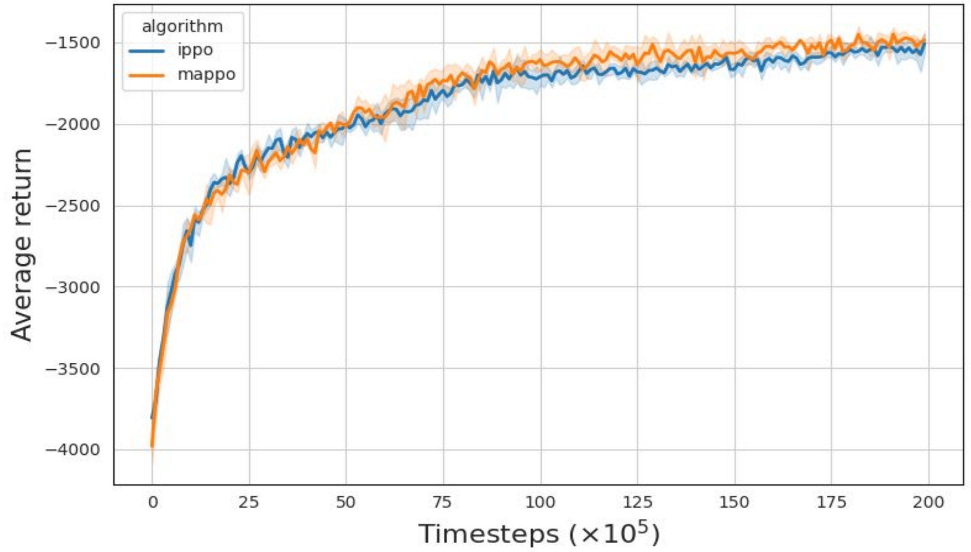
- Implemented Hierarchical Multi-Agent RL (HMARL) with **one master policy** and **two PPO-trained sub policies**: Investigate and Recover.
- The **master policy selects a subpolicy** based on detected **indicators of compromise** (IOCs).
- **Investigate** focuses on **detection** (analyze or deploy decoy)
- **Recover** handles **response** (remove threat or restore network).
- **Block/Allow Traffic were excluded** due to consistently negative impact on performance.

Methodology

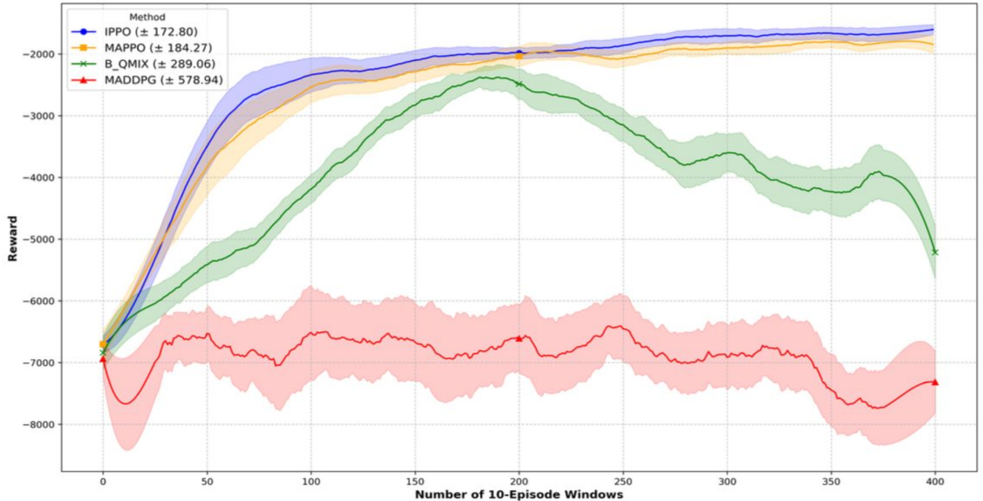


Results (until now)

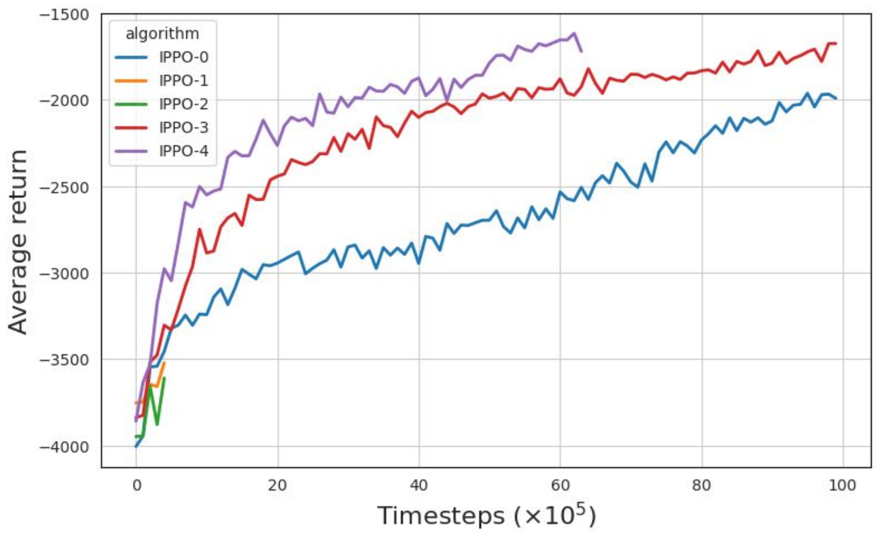
- To answer Research Question 1, **baseline hyperparameters** were set for IPPO and MAPPO using the CybORG environment.
- Training used **20 million timesteps**; each episode lasted **500 steps**, and results were **averaged over 4 runs**.
- **MAPPO** slightly **outperformed IPPO** in average return per episode.

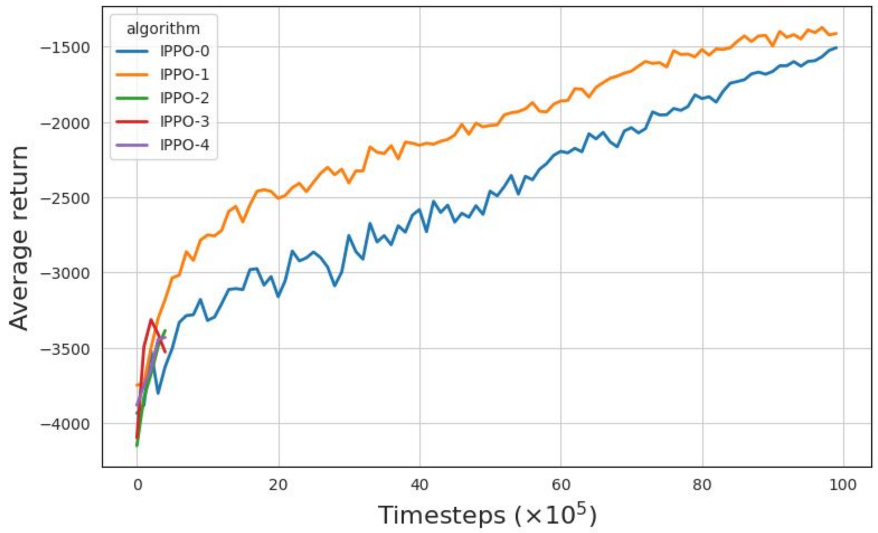


- Around 20 million steps
- Much bigger batch size



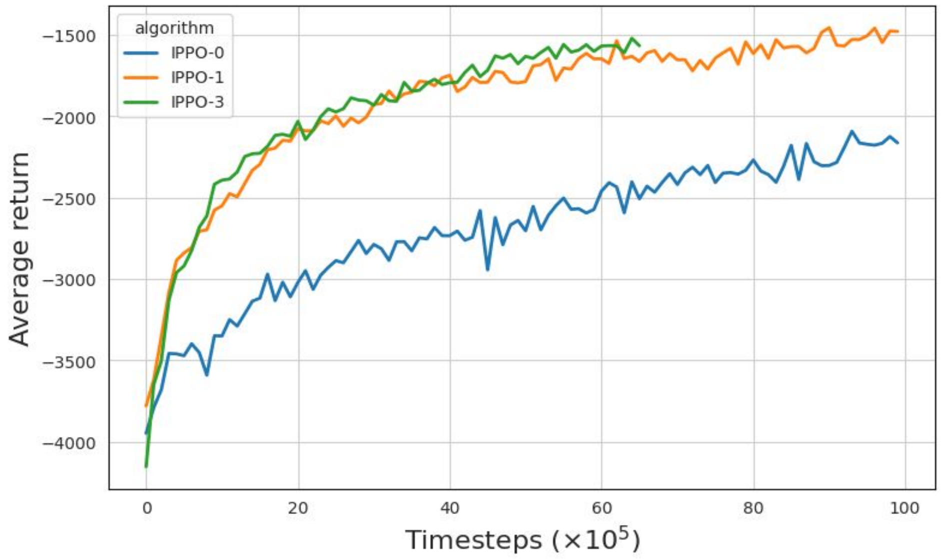
- Around 2 million steps

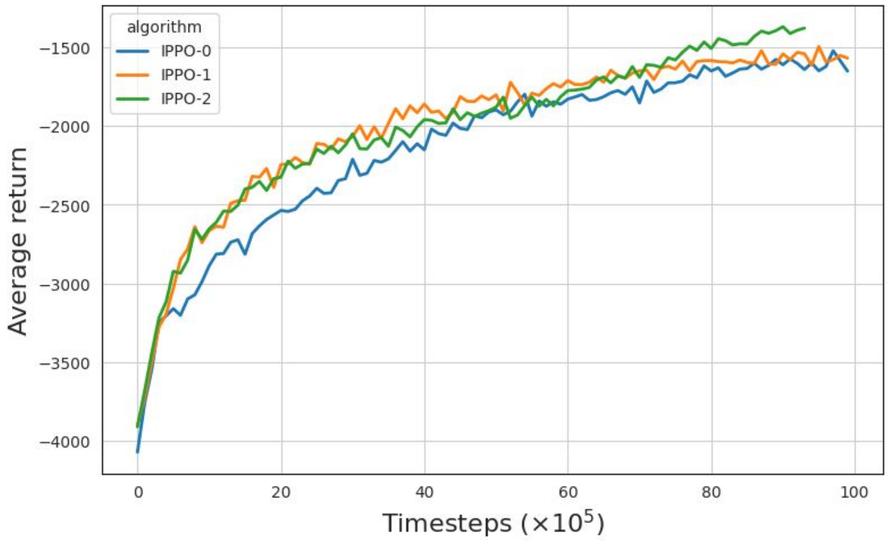




```
IPPO-0_hyper: lr=0.0001873571683731, clip=0.1802740681984156, batch=183553, minibatch=4425
IPPO-1_hyper: lr=0.0002099899276718, clip=0.2375676032007948, batch=110298, minibatch=4865
IPPO-2_hyper: lr=0.0002082773928718, clip=0.1167802436233975, batch=127543, minibatch=2520
IPPO-3_hyper: lr=1.87692869191381e-05, clip=0.1064096544715503, batch=173651, minibatch=2899
IPPO-4_hyper: lr=3.9357180980517874e-05, clip=0.2767812773041417, batch=132497, minibatch=2913
```

```
IPPO-0_hyper: lr=0.0001237904777695, clip=0.1191100372330059, batch=136447, minibatch=3488
IPPO-1_hyper: lr=5.298805928443507e-05, clip=0.1742917932021892, batch=153846, minibatch=3253
IPPO-2_hyper: lr=3.941040459494319e-05, clip=0.1158708550626462, batch=126779, minibatch=5236
IPPO-3_hyper: lr=0.0001312070764956, clip=0.2581465848379486, batch=127763, minibatch=2912
IPPO-4_hyper: lr=0.0001197875593238, clip=0.2828446687866303, batch=117981, minibatch=2071
```





IPPO-0_hyper: lr=0.0002473562799697, clip=0.2024873199372057, batch=132169, minibatch=7145
IPPO-1_hyper: lr=2.144907479114947e-05, clip=0.2710438359092071, batch=167949, minibatch=2849
IPPO-3_hyper: lr=4.546936518956693e-05, clip=0.2391347457923242, batch=188130, minibatch=6286

IPPO-0_hyper: lr=1.4689389186936248e-05, clip=0.1816199713676968, batch=140350, minibatch=4211
IPPO-1_hyper: lr=2.837627136741982e-05, clip=0.263589012911183, batch=132076, minibatch=2980
IPPO-2_hyper: lr=4.06372741487822e-05, clip=0.1756637744517265, batch=177718, minibatch=3889

Challenges

- Training takes a long time (usually around 20 hours), when using Habrok (RUG’s computer cluster).
- It can take around 7-14 days to acquire the processing power on Habrok.
- Narrowing down the hyperparameter range is challenging.

References

1. https://link.springer.com/article/10.1057/s41288-022-00266-6
2. https://github.com/Davide236/Collaborative_RL_CAGE_4
3. https://arxiv.org/abs/2410.17351
4. https://arxiv.org/abs/2011.09533
5. https://arxiv.org/abs/2103.01955
6. https://github.com/cage-challenge/cage-challenge-4
7. https://arxiv.org/abs/2304.11127
8. https://arxiv.org/abs/1810.05934

Thank you for your time!