## Jun 10, 2024 | [Rony / Yinuo](https://www.google.com/calendar/event?eid=NGUyODQ2dTI4amo5Z2hhZWY0aHQ2Njc0amwgeWludW9kQGFuZHJldy5jbXUuZWR1)

Attendees: [Rony Dahdal](mailto:rdahdal@andrew.cmu.edu) [Yinuo Du](mailto:yinuod@andrew.cmu.edu)

Action items

* Explain the C2 structure
  + What are the functions of the server
  + What are the functions of the client
  + How does the server talk to the client
  + Where should you run the server / client code
* Demo the interaction between controller and the hosts
  + Redeploy the controller
* Demo whatever you can do as an attacker (optional)

## Jun 11, 2024 | [Rony Dahdal/Fei Fang Meeting](https://www.google.com/calendar/event?eid=M3BvOWFjMTVpY2hqZjA3NnJvcGVycnQ0dnYgcmRhaGRhbEBhbmRyZXcuY211LmVkdQ)

Attendees: [Rony Dahdal](mailto:rdahdal@andrew.cmu.edu) [Fei Fang](mailto:feif@andrew.cmu.edu)

Notes

* Explain the task (diagram) (note that the communication with AWS is already hardcoded, implemented by Adam and Pau)
* Model
  + Yinuo’s model
  + Adam & Pau’s model
  + Differences between them and why
* IoT paper (RL in AWS) – how to leverage their design for our context/experiment

Action items

* Reach out to IoT team to request code of RL agent and AWS (if code isn’t available)
* Begin designing custom gym environment, similar to HoneyIoT – simulate the network behavior and train a PPO agent inside the wrapped gym (think about attacker and defender – what they observe, effects of actions, how to represent current state of attacker)

## Jun 13, 2024 | [Rony / Yinuo](https://www.google.com/calendar/event?eid=NDQ3dG1lYTdsMDJuYms0bW1mc2E4dWdya2hfMjAyNDA2MTNUMjAwMDAwWiB5aW51b2RAYW5kcmV3LmNtdS5lZHU)

Attendees: [Rony Dahdal](mailto:rdahdal@andrew.cmu.edu) [Yinuo Du](mailto:yinuod@andrew.cmu.edu) [Fei Fang](mailto:feifang@cmu.edu)

Notes

* Explanation
* Demo as defender
* Demo as attacker
* Translation between RL agent and C2
  + observation as a vector
    - currently event counter (feature vector), need to make it machine specific (such as attacker at this IP)
  + action as a number mapping
    - how do I approach IP selection as an action here? (Defender must select action such as “start cowrie” but also *where*

Action items

Jun 21, 2024

Main milestone: Build the interface to evaluate a trained RL policy in AWS

1. Load RL policy
2. AWS -> observations -> input of RL policy
3. Output of RL policy -> sampled action -> executable action -> AWS

Steps to be completed

1. Network configuration in AWS, Actions for defender and attacker
2. What does one time step correspond to in the real environment?
3. Input / output of RL policy: vector representation of the observation, vector representation of action or list of actions if limited number of actions are allowed, action masking rules
4. (simple) neural network architecture of the RL policy network
5. Collect observations from the real system and then convert them into the format that RL policy takes as input
6. Procedure of getting executable actions from RL policy network
7. How to execute each of the possible actions
8. Integrate the entire workflow

## Jun 25, 2024 | [Rony Dahdal/Fei Fang Meeting](https://www.google.com/calendar/event?eid=NzlmYmZ1cDJnYjc3dDZrZm9tc3E5bHB0ZW8geWludW9kQGFuZHJldy5jbXUuZWR1)

Attendees: [Rony Dahdal](mailto:rdahdal@andrew.cmu.edu) [Fei Fang](mailto:feif@andrew.cmu.edu) [Yinuo Du](mailto:yinuod@andrew.cmu.edu)

Notes

* Automated attacker
  + Different types of heuristic attacker. Check out existing literature to backup your design of the attacker policy
  + Design the input to the attacker’s policy, which does not have to be a neural network policy.
* Timestep definition
  + Note that the attacker might not follow your schedule of one action per 30 seconds.
  + Keep in mind the implications of time step. The defender needs to keep track of the execution status of the actions and be aware of whether the actions are finished. If the actions enacted in previous steps are not finished, those actions need to be filtered out with action masks.
* Observation
  + Rule of thumb: Only include information from the telemetry. Exclude the “inferred” information.
    - Remove the node value and attack position
  + Represent the attack’s actual state with one-hot encoding. Four bits for one host.
    - foothold : [0,1,0,0]
    - Privesc: [0,0,1,0]
    - data.ex: [0,0,0,1]
  + Represent the attacker’s perceived state in a similar format
    - Not tricked: [0, 0, 0, 0]
    - Fake foothold : [0,1,0,0]
    - Fake Privesc: [0, 0,1,0]
    - Fake data.ex: [0, 0,0,1]
  + A vector to represent the existence of fake edge: [0, 0,0,0,0,0] for no fake edge deployed
* Assumptions
  + We expect the existence of the normal user and a list of trusted ip so that we can differentiate normal users from attackers.
  + We assume the normal user activity is stationary so that we can estimate the loss of actions like shutting down a host.
  + We assume there’s only one attacker existing in the network.
  + The benefit of recording the existence of normal users is a more accurate estimate of loss for actions like shutting down a host.
* Budget represent strict constraint
* Cost represent the loss
* How to represent the actions?

## 

## Jul 2, 2024 | [Rony Dahdal/Fei Fang Meeting](https://www.google.com/calendar/event?eid=NzlmYmZ1cDJnYjc3dDZrZm9tc3E5bHB0ZW9fMjAyNDA3MDJUMTcwMDAwWiB5aW51b2RAYW5kcmV3LmNtdS5lZHU)

Attendees: [Rony Dahdal](mailto:rdahdal@andrew.cmu.edu) [Fei Fang](mailto:feif@andrew.cmu.edu) [Yinuo Du](mailto:yinuod@andrew.cmu.edu)

Notes

* Defender policy network
  + A host-selection network + An action-selection network
  + The output of host-selection network is part of input to the action-selection network
  + Implement a simple policy network (relu+softmax) and randomly initialize the network
* Attacker
  + Think about how does the attacker observe the network
  + Implement two types of attacker, a powerful one and a weaker one.

## Jul 16, 2024 | [Rony Dahdal/Fei Fang Meeting](https://www.google.com/calendar/event?eid=NzlmYmZ1cDJnYjc3dDZrZm9tc3E5bHB0ZW9fMjAyNDA3MTZUMTcwMDAwWiB5aW51b2RAYW5kcmV3LmNtdS5lZHU)

Attendees: [Rony Dahdal](mailto:rdahdal@andrew.cmu.edu) [Fei Fang](mailto:feif@andrew.cmu.edu) [Yinuo Du](mailto:yinuod@andrew.cmu.edu)

Notes

* Trigger logic
  + ON/OFF determines whether to send the observations to the policy network.
  + INITIALLY the switch is set as OFF
  + Switch to ON when attacker activity is observed (i.e., when the observation is not empty)
  + Send the observation to the policy network as long as the switch is ON, no matter whether that specific observation is empty or not.
  + Switch to OFF when 10 consecutive observations are empty.
* LSTM
  + Save the embeddings from last step

TODO

* Basics of LSTM
* Check the existing implementation

## Jul 31, 2024 | [Rony Dahdal/Fei Fang Meeting](https://www.google.com/calendar/event?eid=NzlmYmZ1cDJnYjc3dDZrZm9tc3E5bHB0ZW9fMjAyNDA3MzBUMTcwMDAwWiB5aW51b2RAYW5kcmV3LmNtdS5lZHU)

Attendees: [Rony Dahdal](mailto:rdahdal@andrew.cmu.edu) [Fei Fang](mailto:feif@andrew.cmu.edu) [Yinuo Du](mailto:yinuod@andrew.cmu.edu)

Notes

* Check with Zhuorui how they use PPO
  + To have more control over the algorithm and less constraints
* Heuristic defense policy
  + Log human demonstration
    - Simulated attacker
    - Manual label of defender action
  + Train neural network with imitation learning