LANCER

Cornell Site Visit
November 30, 2023

Outline

- [15 minutes] Introduction (Nate & Wen)
  - Team Introductions
  - Technical Approach
- [10 minutes] Progress Since Kick-Off (Nate & Wen)
  - Executive Summary
  - Planned Trajectory for end of Phase I
- [15 minutes] Collaboration Efforts (Nate & Rebecca)
  - CAGE
  - Talking to Kryptowire
  - Network Action Space
- [60 minutes] Early Results
  - [20 minutes] NetKAT (Jules & Nate)
  - [30 minutes] Inverse RL (Nico/Rebecca & Wen)
  - [10 minutes] Aether: Pronto + OnRamp (Hussain & Nate)
- [20 minutes] Response to Crawl Questions (Everyone)
- [30 minutes] Budget & Contracting (Shailja & Nate)

Introduction

Progress

Progress

- Got going with Kryptowire TA1 Platform
- Started development using CAGE 2
- Started Modeling Red Agents Using Inverse RL
- Fast NetKAT implementation
- Standing Up Aether OnRamp

Trajectory

- Crawl (6 month)
- Walk (6 month)
- Run (6 month)

Collaboration Efforts

Cage Challenge: Overview

- Scenario of a network attack
  - **Red Agent** (malicious): infiltrates network
  - **Blue Agent** (defensive): protects the network
  - **Green Agents** (neutral users): generate noise
- Integrated with CybORG, a reinforcement learning gym

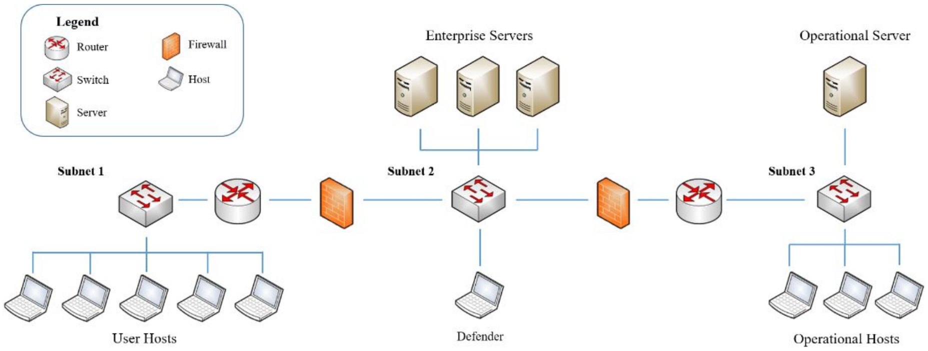


Figure 1: Network Topology (Cage Challenge 2)

Cage Challenge: Red Agent Actions

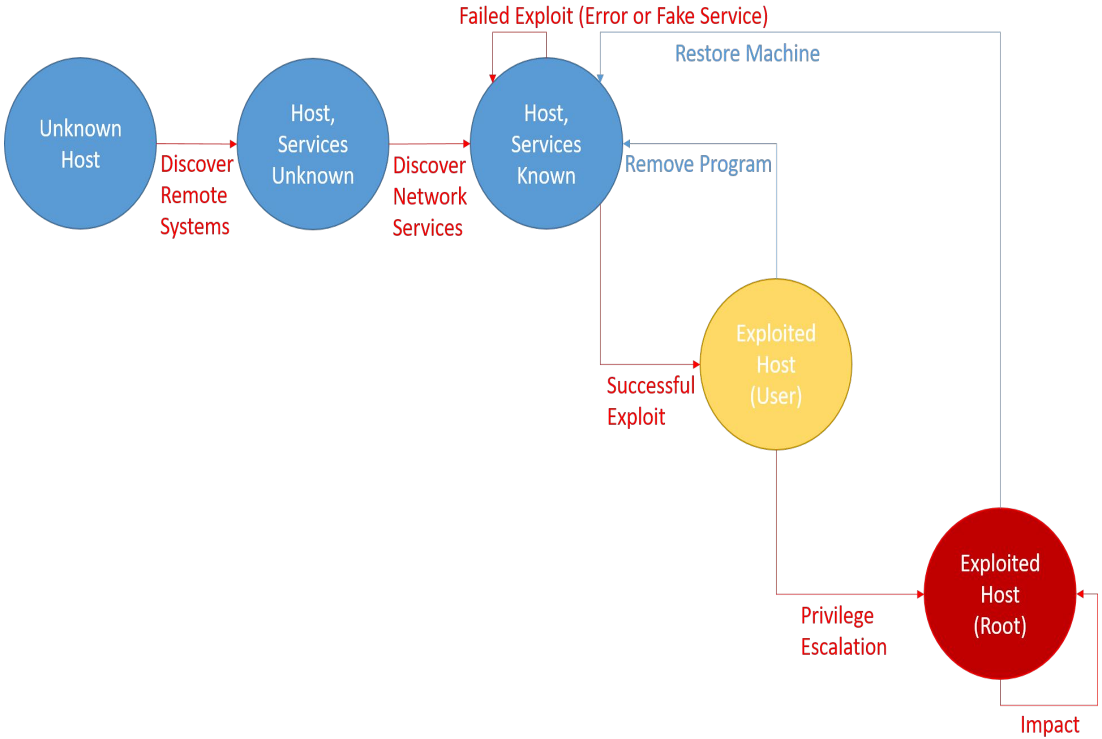


Figure 2: Effect of actions on host state (Cage Challenge 2)

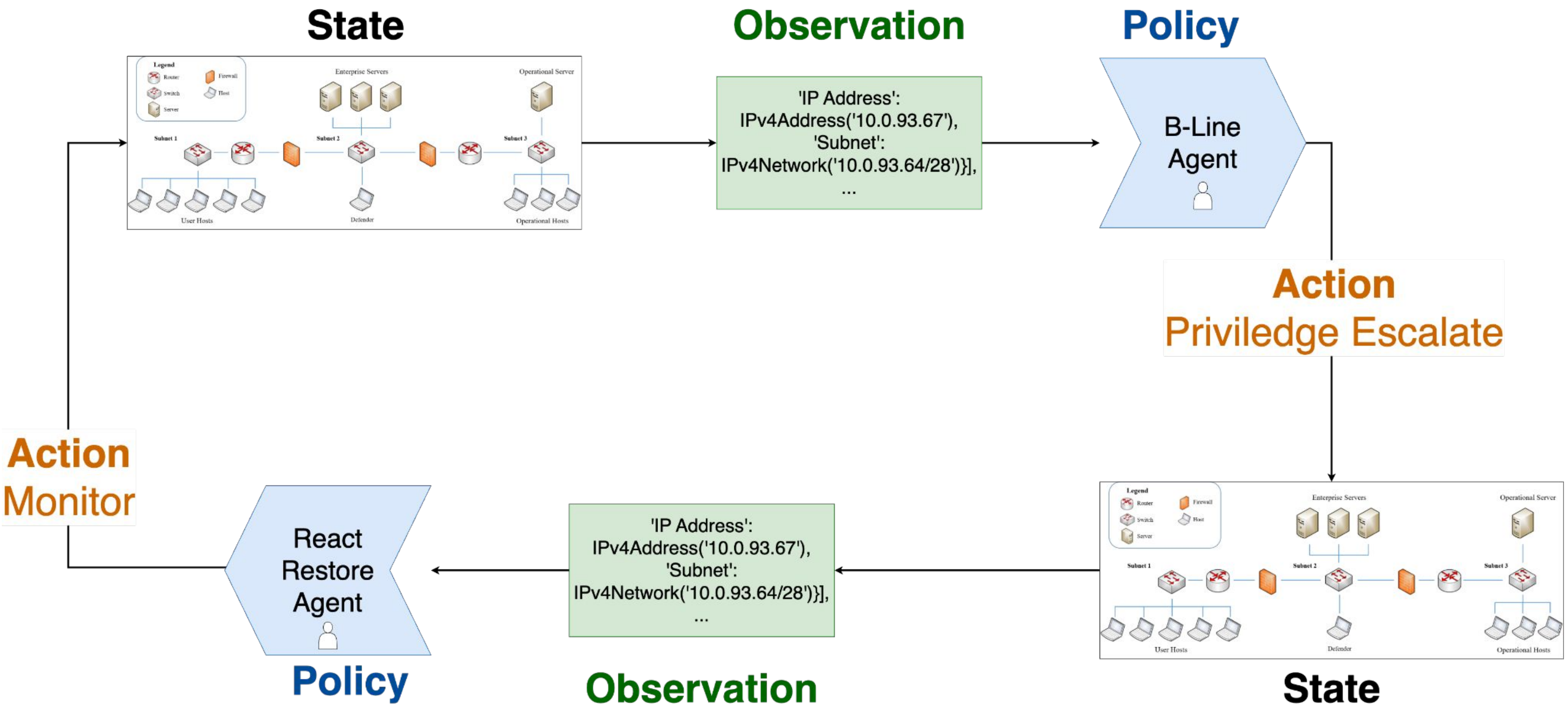
Early Results

Learning Approach for Modeling Red Agents

Imitation Learning: learn red agents' behavior from their traces

- Real-world scenario: only have examples (data) of network exploit (i.e. Red agent infiltration)
  - No access to novel Red agents for simulation
- Once red agents are learned: train blue agents against them
  - Targeted RL training
  - Adversarial RL training: train Blue and Red to fight each other
    - Often results in very conservative behaviors

Reinforcement Learning Terminology



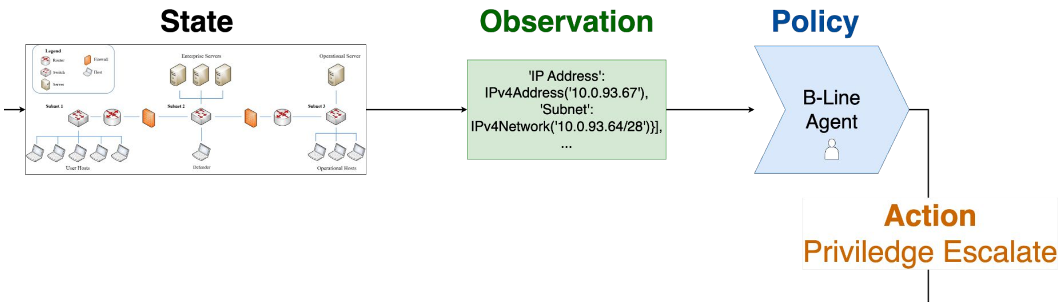
Reinforcement Learning Terminology

States: configuration of the environment

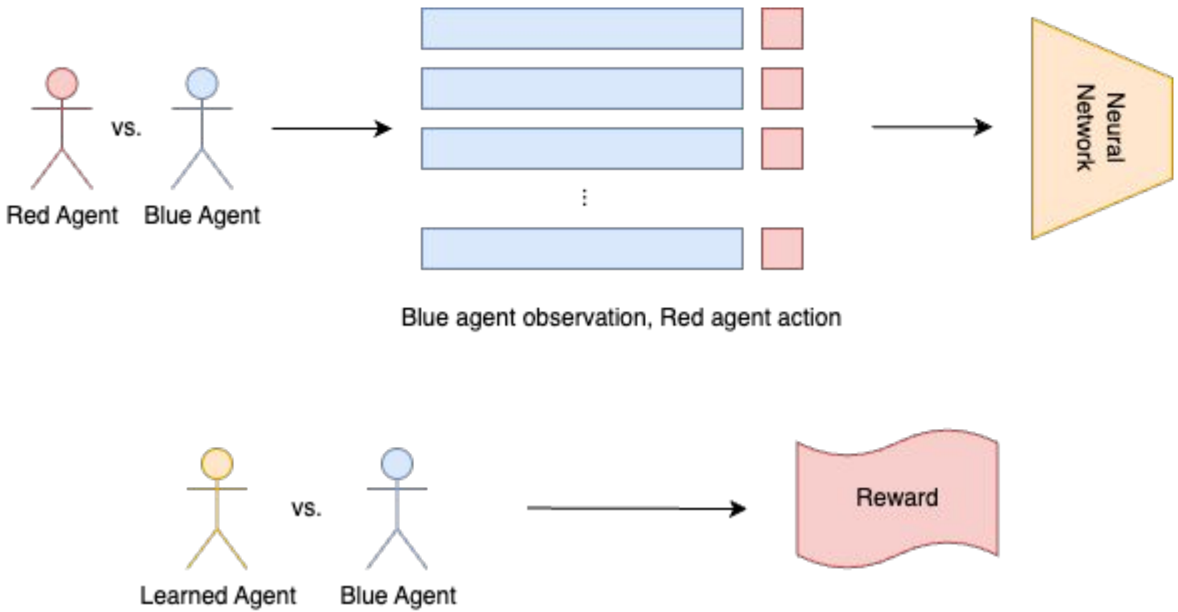
Observation: environment information observed by an agent

Policy: how an agent decides what action to take

Rollout: a sequence of states, actions, and associated reward



Behavior Cloning (BC)



Behavior Cloning (BC)

1. Collect data from environment with Blue, Green, Red agents
    - (Blue agent observation, Red agent action)
2. Train neural network on collected data
    - Blue agent observation → *predicted* Red agent action
3. Created a learned Red agent: used trained neural network as policy
4. Collected reward during rollout: environment with Blue, Green, and learned Red agent
    - Measure of learned Red agent’s quality: reward collected during rollout

BC: 1 Input Observation

Red Agent	Blue Agent	Training Metrics			Learned Red Agent		True Red Agent	
		Train Loss	Train Accuracy	Validation Accuracy	Reward	Standard Deviation	Reward	Standard Deviation
B-Line	React Remove	0.16	0.95	0.93	556	361	947	193
B-Line	React Restore	0.64	0.77	0.77	-10.0	0.0	508	366
Meander	React Remove	0.71	0.72	0.67	11.1	39.5	630	259
Meander	React Restore	1.10	0.56	0.53	3.55	7.77	185	210

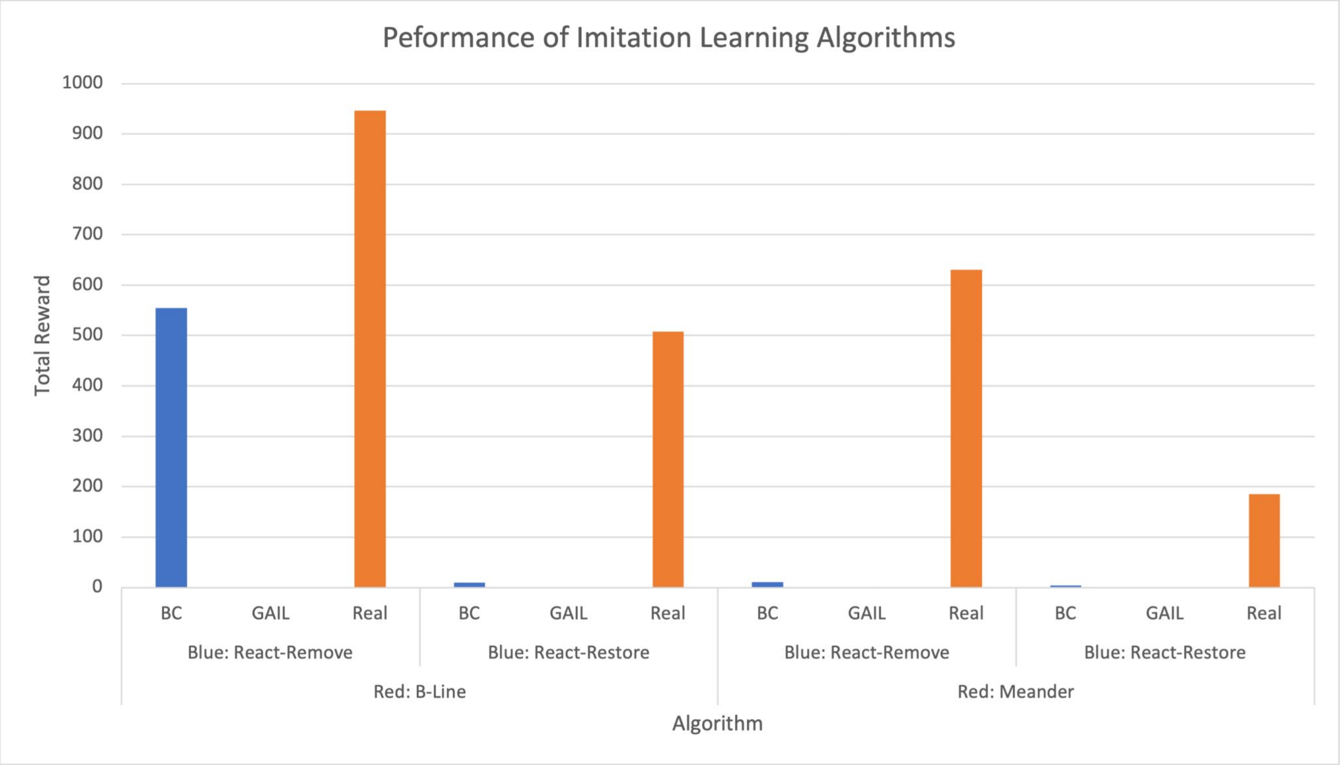
BC: 4 Input Observations

		Training Metrics			Learned Red Agent		True Red Agent	
Red Agent	Blue Agent	Train Loss	Train Accuracy	Validation Accuracy	Reward	Standard Deviation	Reward	Standard Deviation
B_Line	React Remove	0.038	0.986	0.967	694	305	947	193
B-Line	React Restore	0.0372	0.987	0.965	484	336	508	366
Meander	React Remove	0.327	0.870	0.710	255	246	630	259
Meander	React Restore	0.615	0.762	0.587	77	141	185	210

BC Plot: Reward vs. Number of Input Observations

BC Plot: Reward vs. Dataset Size

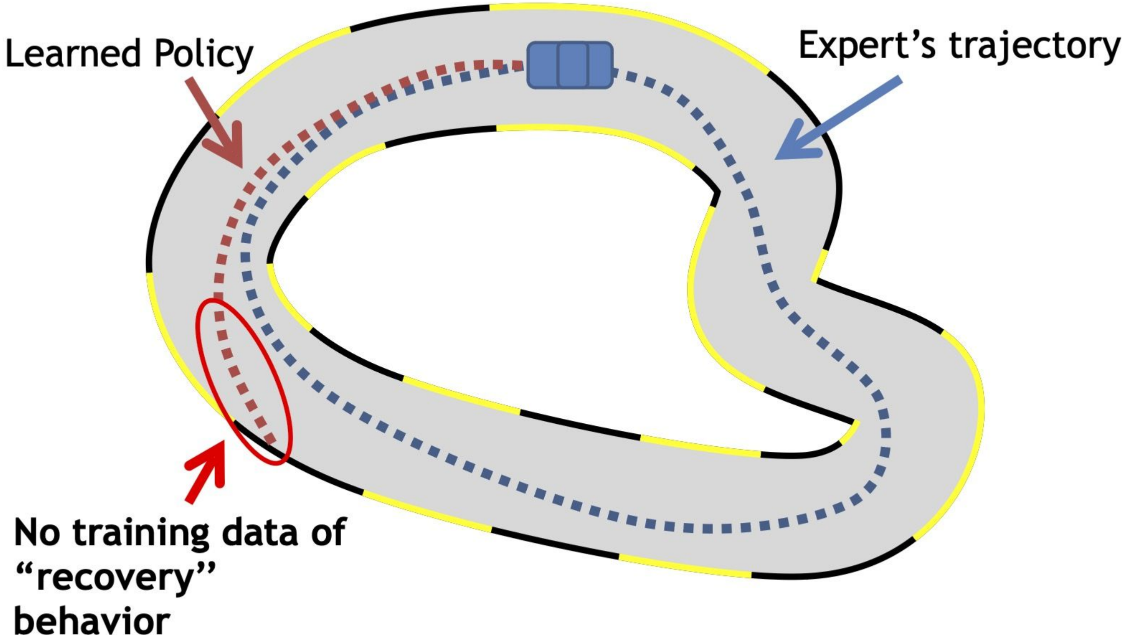
Results



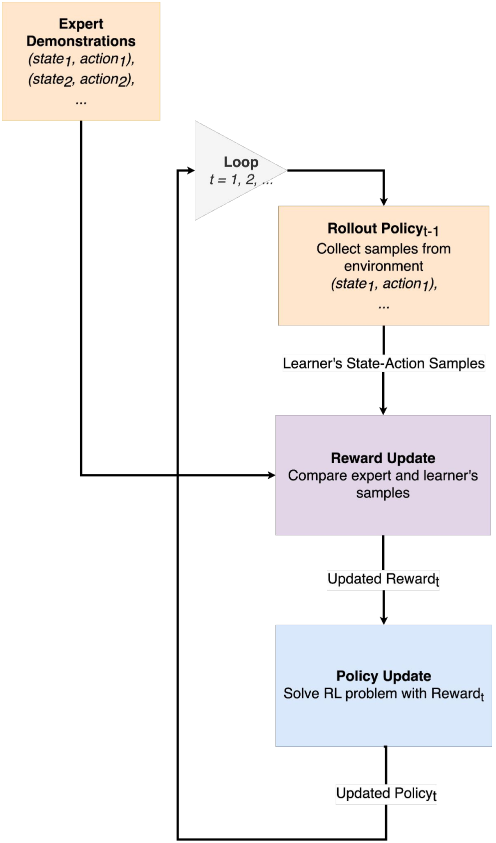
Peformance of Imitation Learning Algorithms
Total Reward
1000
900
800
700
600
500
400
300
200
100
0
BC
GAIL
Real
BC
GAIL
Real
BC
GAIL
Real
BC
GAIL
Real
Blue: React-Remove
Blue: React-Restore
Blue: React-Remove
Blue: React-Restore
Red: B-Line
Red: Meander
Algorithm

Issue of Behavior Cloning: Distribution Mismatch

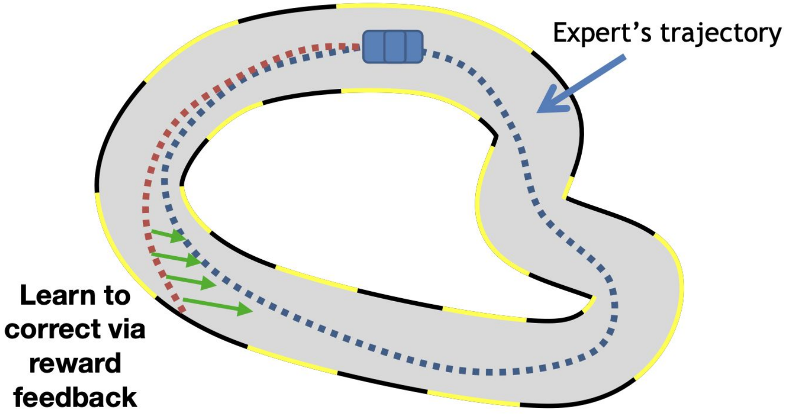
Learning to Drive
Compounding error makes learner deviate from the expert track quickly



Inverse RL to the Rescue



1. Inverse RL aims to learn a reward model from the data (e.g., red agent's reward function when they plan attacks)
2. It then learns a policy to optimize the learned reward
3. The learned policy acts as the predictive model for the red agent

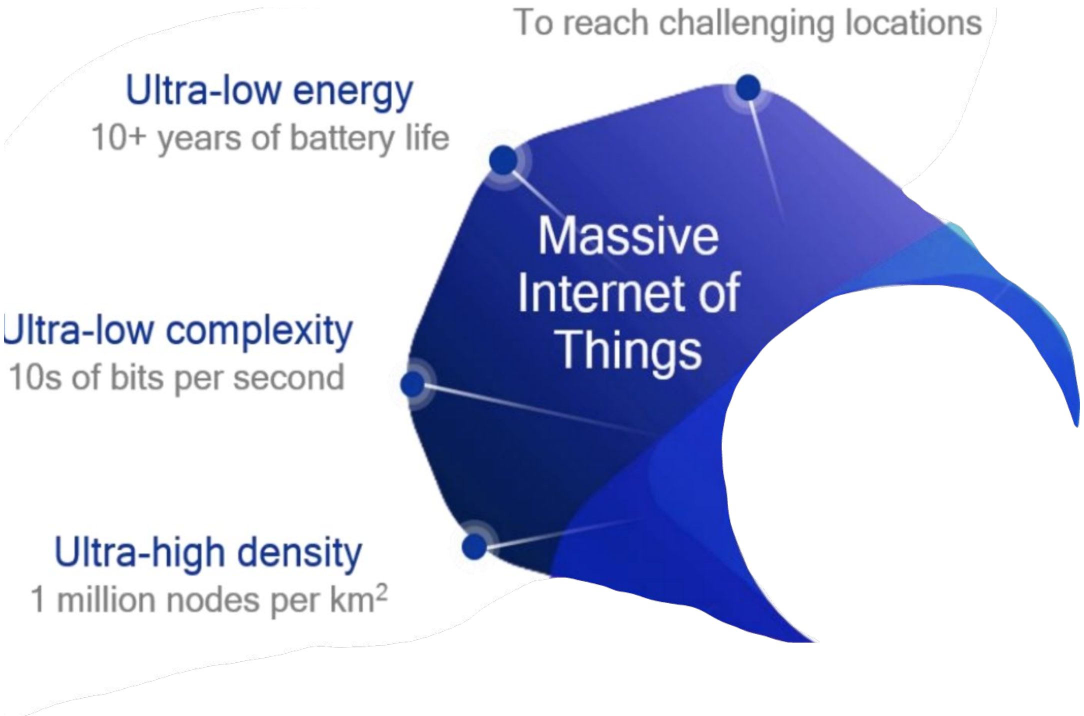


IRL Results

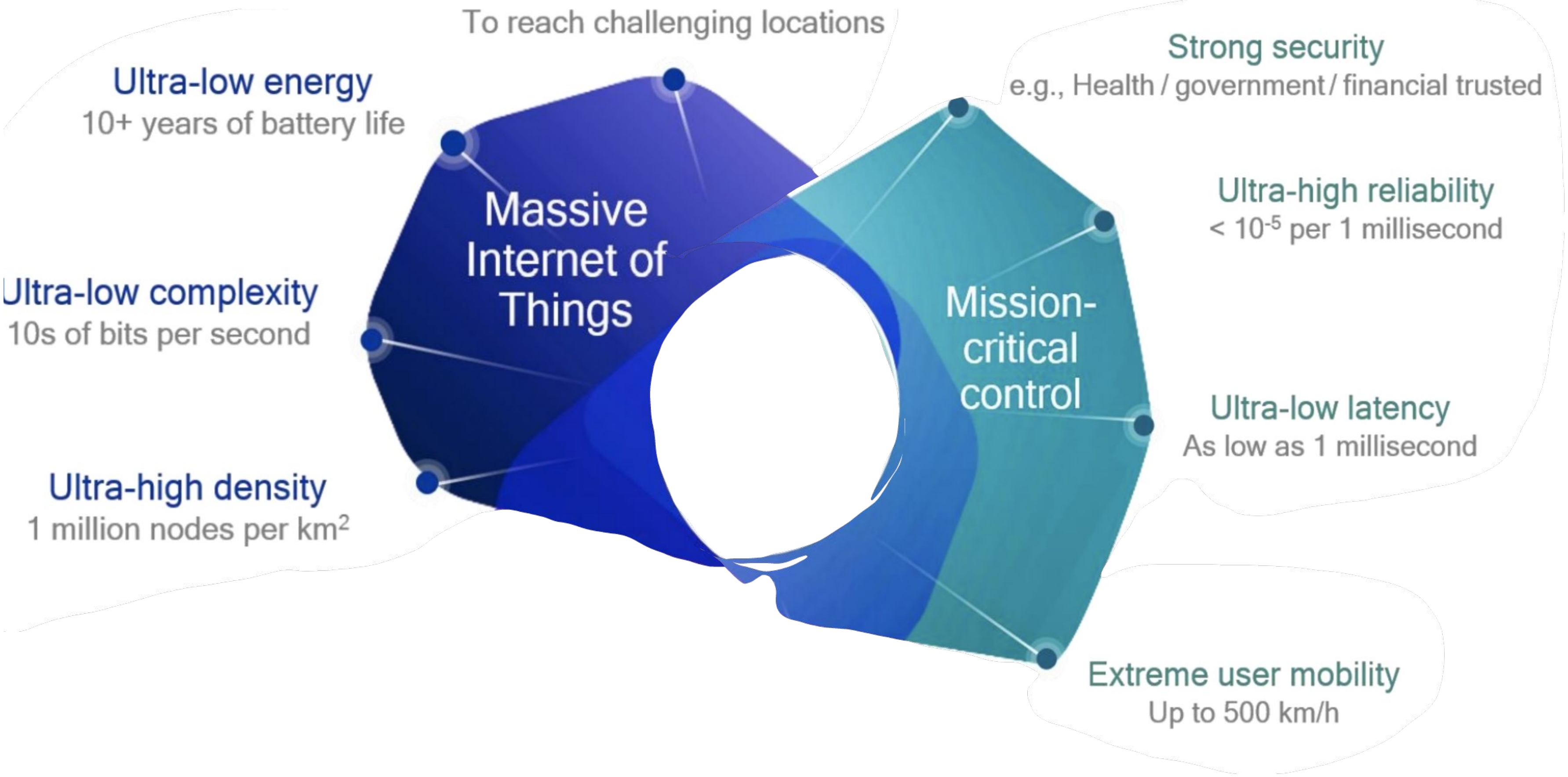
Next Steps

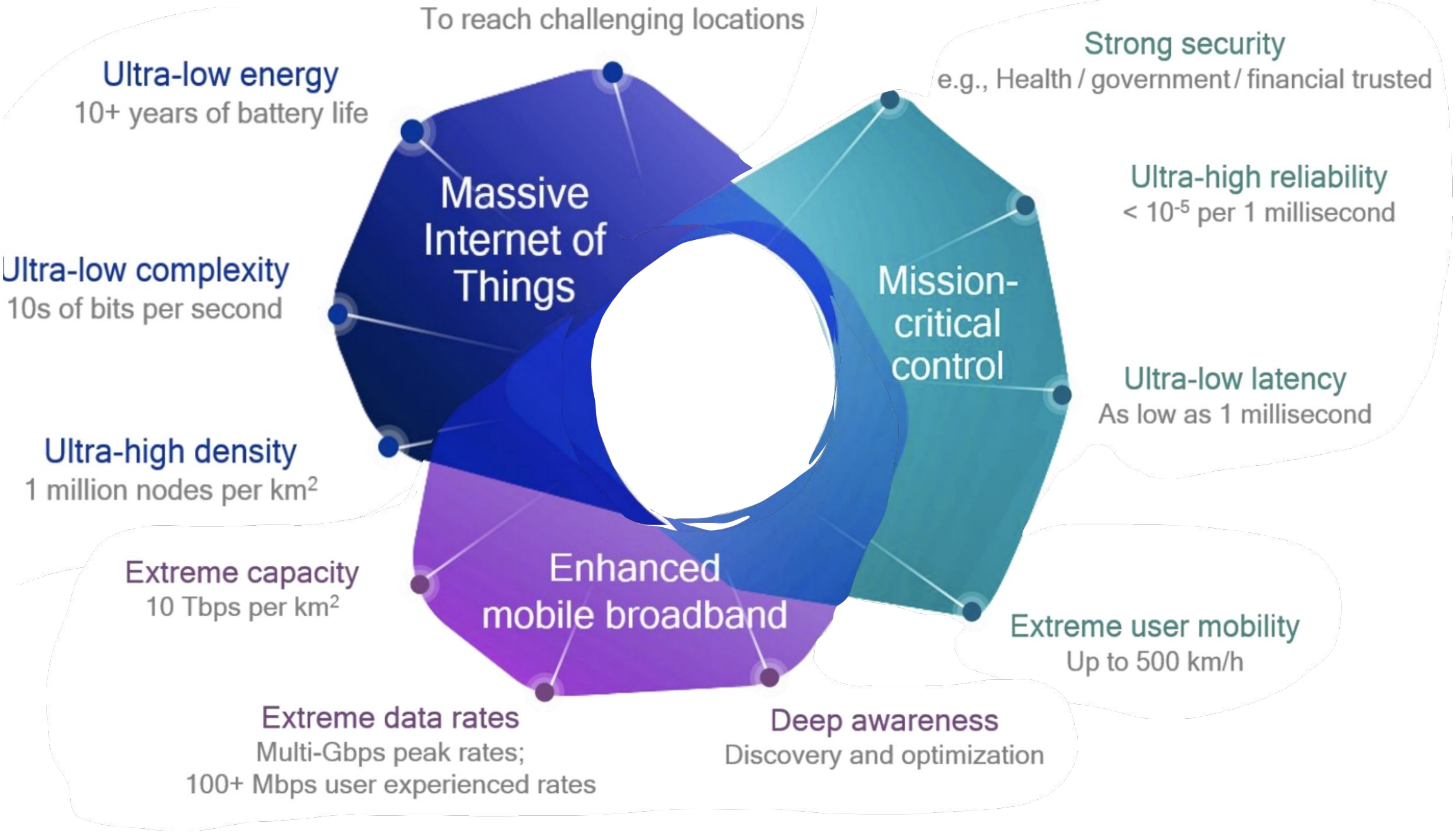
1. New IRL algorithms for improving modeling red agents;
2. Training RL agents against the learned red agents

NetKAT

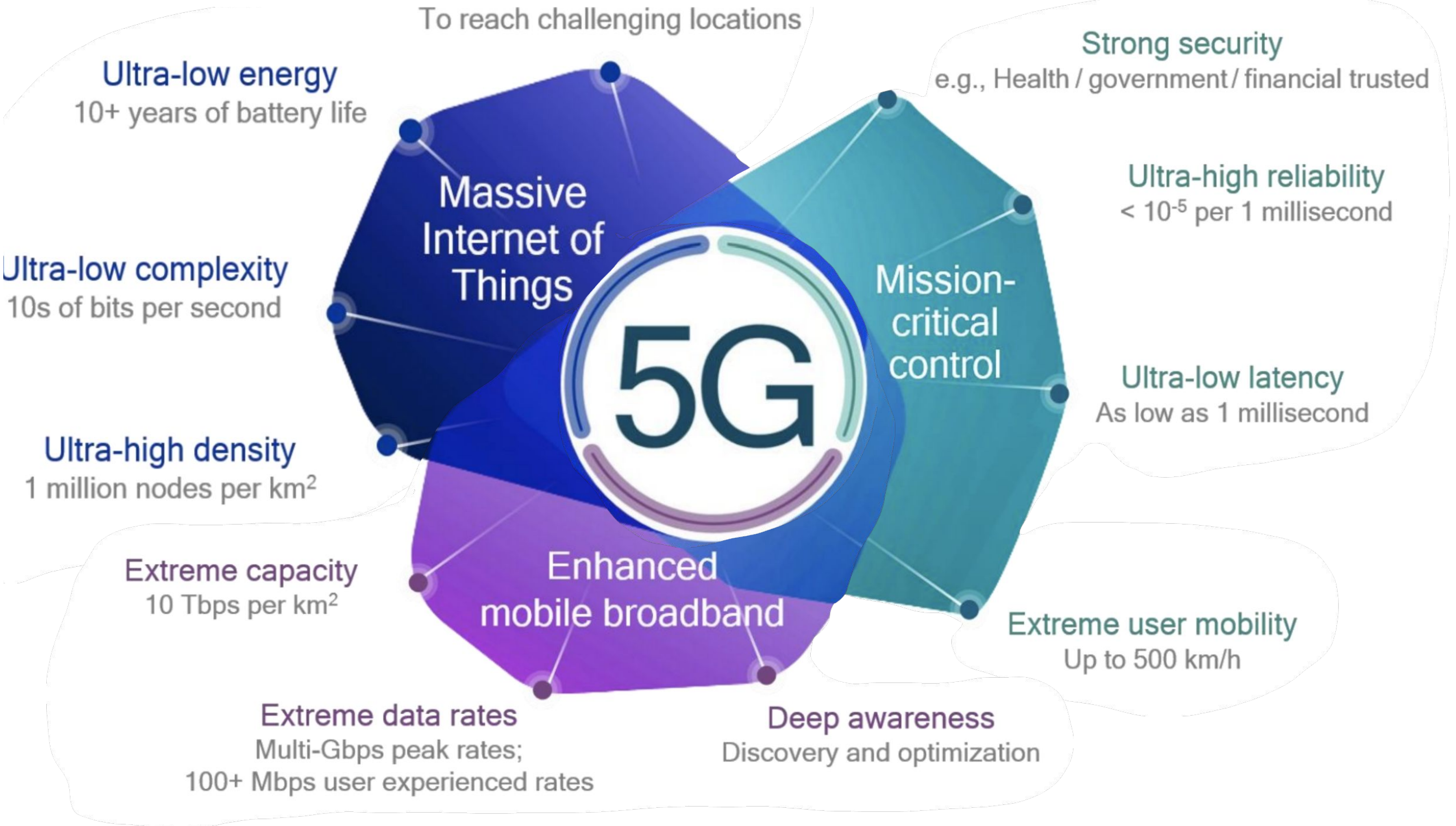


To reach challenging locations
Ultra-low energy
10+ years of battery life
Massive Internet of Things
Ultra-low complexity
10s of bits per second
Ultra-high density
1 million nodes per km²


To reach challenging locations
Ultra-low energy
10+ years of battery life
Ultra-low complexity
10s of bits per second
Ultra-high density
1 million nodes per km²
Massive Internet of Things
Strong security
e.g., Health / government / financial trusted
Ultra-high reliability
< 10⁻⁵ per 1 millisecond
Mission-critical control
Ultra-low latency
As low as 1 millisecond
Extreme user mobility
Up to 500 km/h



To reach challenging locations
Ultra-low energy
10+ years of battery life
Ultra-low complexity
10s of bits per second
Ultra-high density
1 million nodes per km^2
Massive Internet of Things
Strong security
e.g., Health / government / financial trusted
Ultra-high reliability
< 10^{-5} per 1 millisecond
Mission-critical control
Ultra-low latency
As low as 1 millisecond
Extreme user mobility
Up to 500 km/h
Extreme capacity
10 Tbps per km^2
Enhanced mobile broadband
Extreme data rates
Multi-Gbps peak rates;
100+ Mbps user experienced rates
Deep awareness
Discovery and optimization



To reach challenging locations
Ultra-low energy
10+ years of battery life
Strong security
e.g., Health / government / financial trusted
Massive Internet of Things
Ultra-high reliability
< 10-5 per 1 millisecond
Ultra-low complexity
10s of bits per second
Mission-critical control
5G
Ultra-low latency
As low as 1 millisecond
Ultra-high density
1 million nodes per km2
Extreme capacity
10 Tbps per km2
Enhanced mobile broadband
Extreme user mobility
Up to 500 km/h
Extreme data rates
Multi-Gbps peak rates;
100+ Mbps user experienced rates
Deep awareness
Discovery and optimization

5G networks -

5G Mobile Network two main subsystems :

1. RAN - manages radio resources(spectrum)
2. Mobile Core - provide packet data network to mobile subscribers

AetheronRamp - Private Enterprise 5G network

- operational cluster that is capable of running 24/7 and supports live 5G workloads.
- Cluster containerizing subsystems components, can scale horizontally with dynamic workloads.

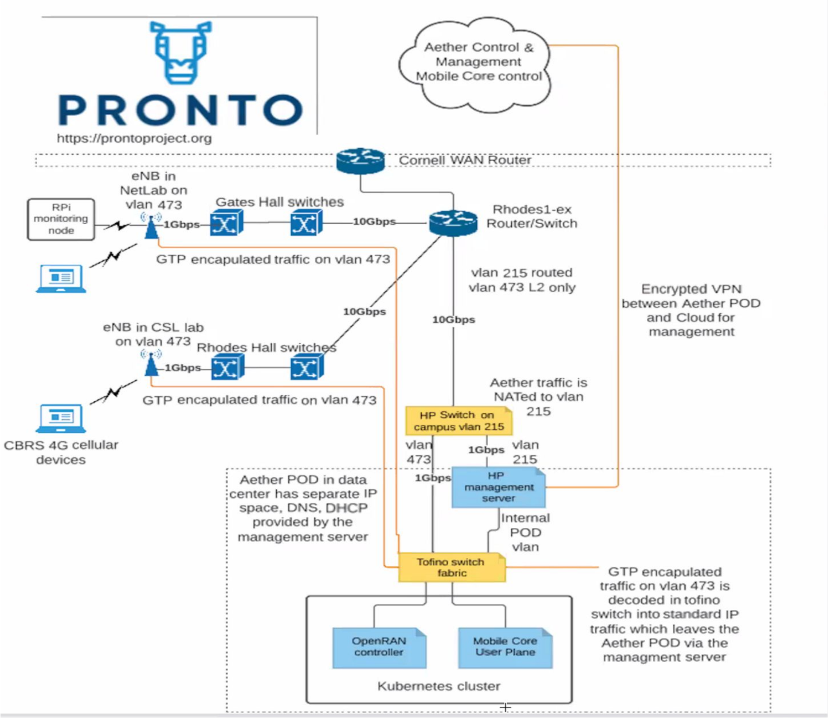
Pronto & AetherOnRamp Demo

Pronto 4G network

- Current testbed located at Gates Lab and Robotics Lab.
- Supports both direct access 4G connectivity, extended with APN connectivity

AetherOnRamp 5G network

- Work in Progress, currently emulate UEs(mobile devices) control and data plane connectivity



Crawl Questions

- Learn about one another's approaches, find integration points, and collaborate on shared infrastructure
- What network should we model first and what workflows should be present?
- What agent actions will be simulated and executed?
- What is a ‘good’ resiliency criteria and how will we judge whether your approach is successful?
- What data types are needed for each performer and what data can be provided by each performer?
  - Data for attackers
  - Reward function for defenders (domain knowledge, Inverse RL)
- How do we collaborate on API design and code interfaces?
- What open-source technology can enable an end-to-end integration demo quickly?
- Who is the intended operator of your approach and what is the desired impact/benefit to their job?