AI vs. AI

Autonomous Cyber Defence (ACD) against AI-Driven Threats

Ian Miles (i.miles@fnc.co.uk)

More detail available in our paper:
"Reinforcement Learning for
Autonomous Cyber Defence"

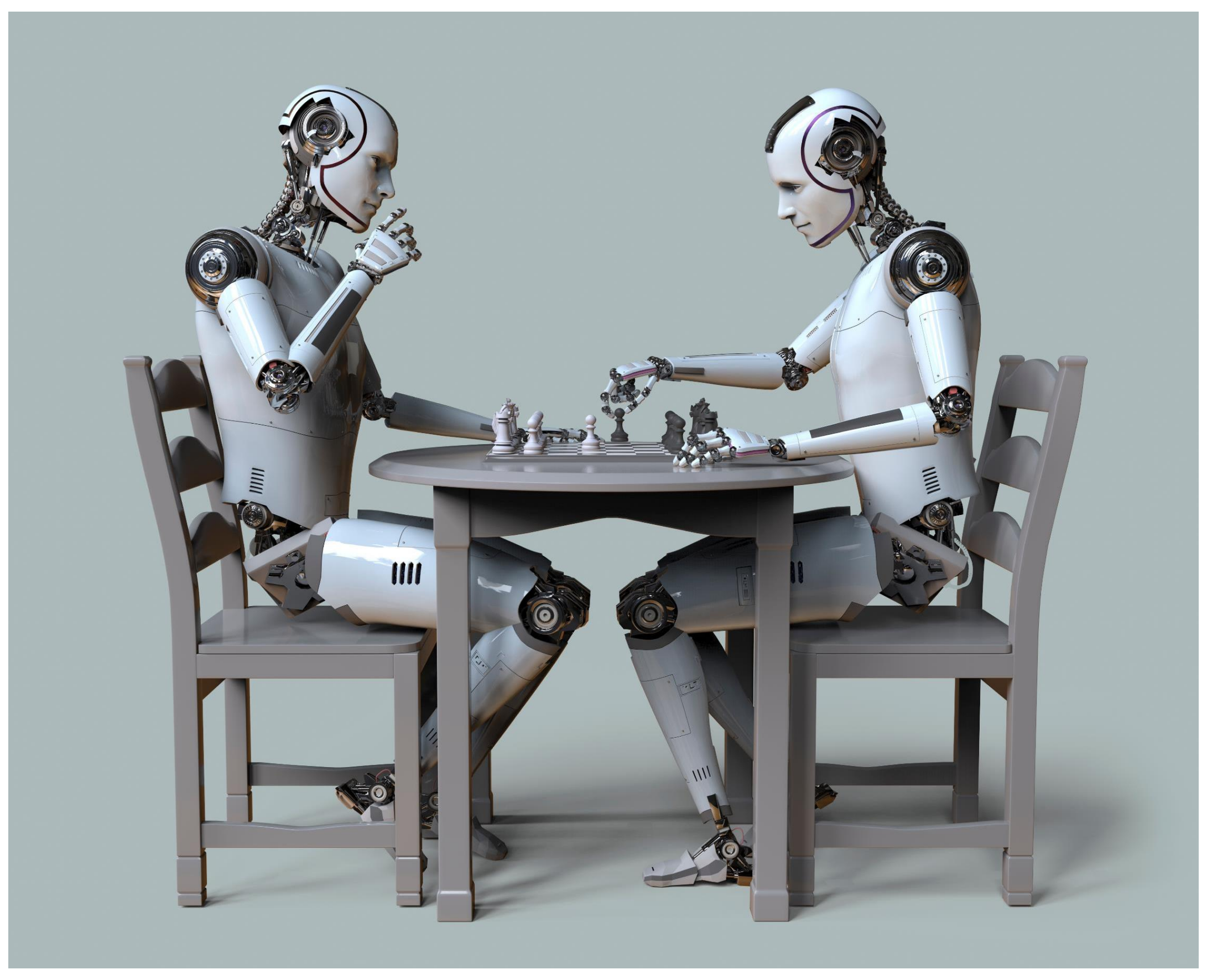
The Need

Robust ACD agents (blue agents) must avoid overfitting to training adversaries, particularly in a Defence context where:

- Blue agents are likely to be deployed at the edge, where connectivity, safety and security constraints will limit the ability for live retraining or updates.
- Adversaries are more likely to have extensive resources and expertise, leading to a higher probability of novel attacks.

Robust blue agents require vast volumes of training data covering a broad range of cyber attacker (red agent) Tactics, Techniques and Procedures (TTPs). Using human experts to generate sufficient volumes of training data has significant feasibility challenges in terms of availability and cost.

Deep Reinforcement Learning (RL) offers opportunity to overcome these training limitations by developing red agents as training adversaries. We can also extend ACD to Autonomous Cyber Operations (ACO), where RL red and blue agents train and operate within the same environment. Such approaches, including adversarial learning, could identify next generation attack TTPs, that target the ACD agent as an attack vector.



Sim-to-real transfer of generalisable red & blue RL agents

Trustworthy AI combined Deep RL with Heterogenous Graph Neural Networks to train red and blue agents that can generalise across multiple network topologies. Typically this would require a lot of manual effort to develop training network configurations. Instead, GPT4 was used to create 80 synthetic network configurations (with an average of 24 computers). Evaluation in network topologies not seen in training showed improved blue agent performance as the number of training network topologies increased (Figure 1).

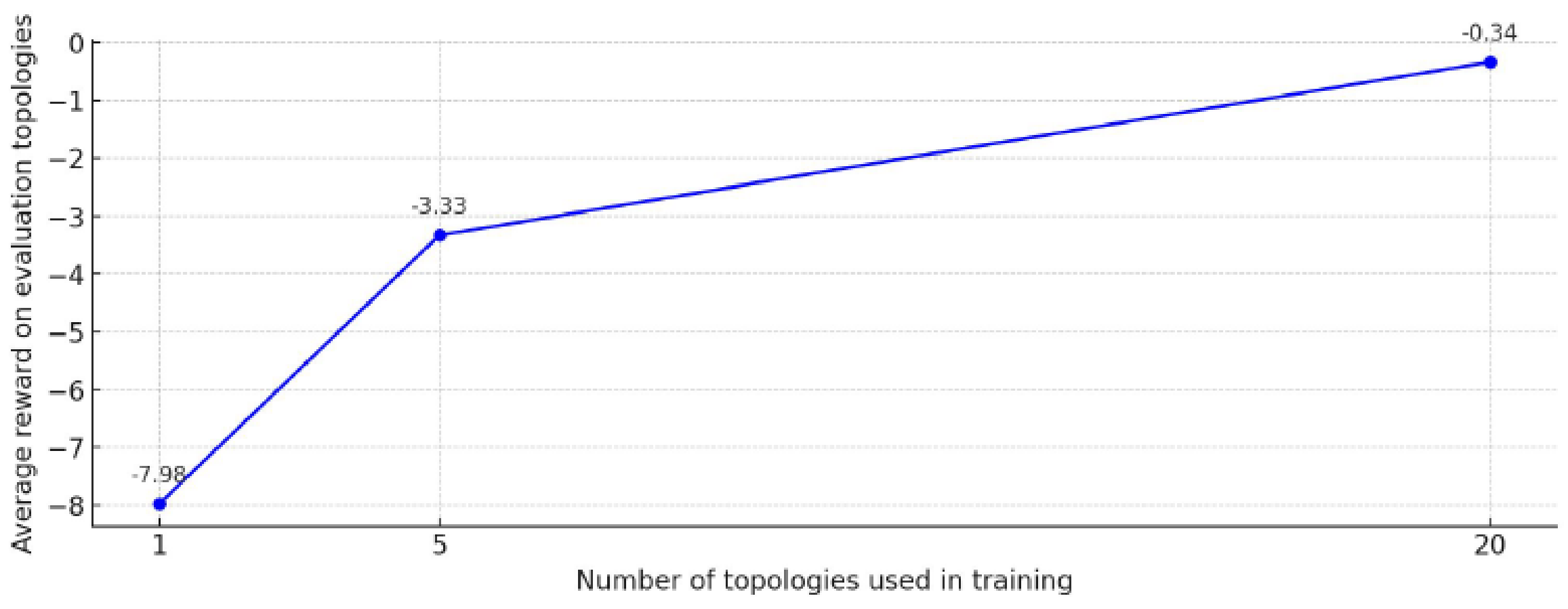


Figure 1: Autonomous cyber defence performance in evaluation network topologies (unseen in training) improving as the number of training topologies increases.

A custom simulation and emulation environment, CAESAR (Cyber Adaptive Environment for Simulating Autonomous Response) has been developed for training agents, with an action & observation space guided by real-world cyber tooling (Figures 2 & 3). Blue agent actions are executed through the Elastic Security Agent, leveraging both natively supported actions and custom commands executed on hosts via the agent. Similarly, the red agent utilises Cobalt Strike, incorporating native Cobalt Strike commands, .NET assemblies, and Beacon Object Files (Figure 3).

The CAESAR environment is designed around these actions and their associated assets. For the emulated environment, we currently support network deployment on AWS via Ansible and User Data scripts, fully customised through a network configuration file. The tight coupling of our emulator and simulator provides the ability to train our agents efficiently whilst maintaining the close relationship to a real system. The transfer learning approach has been >1000x faster than training in a real system and we have recently conducted our first successful transfer of agents from simulation into emulation.

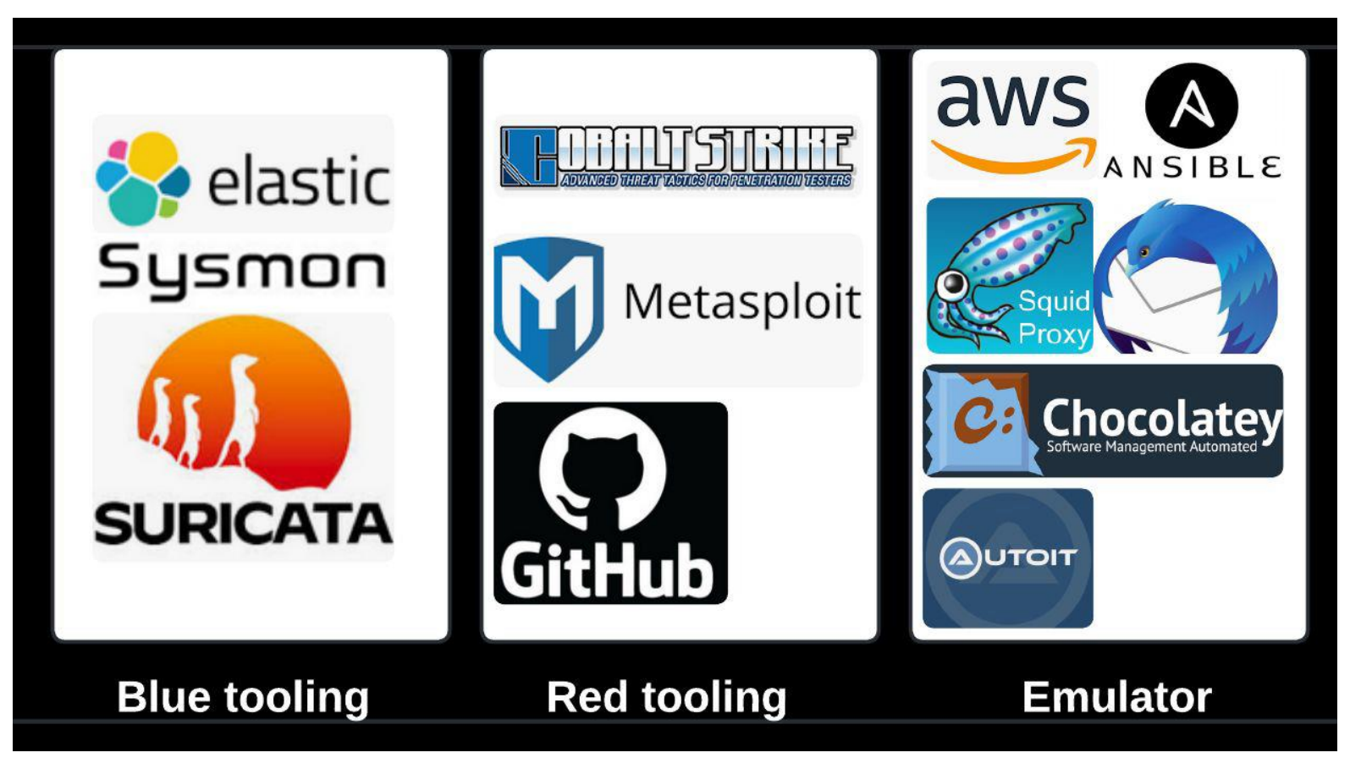
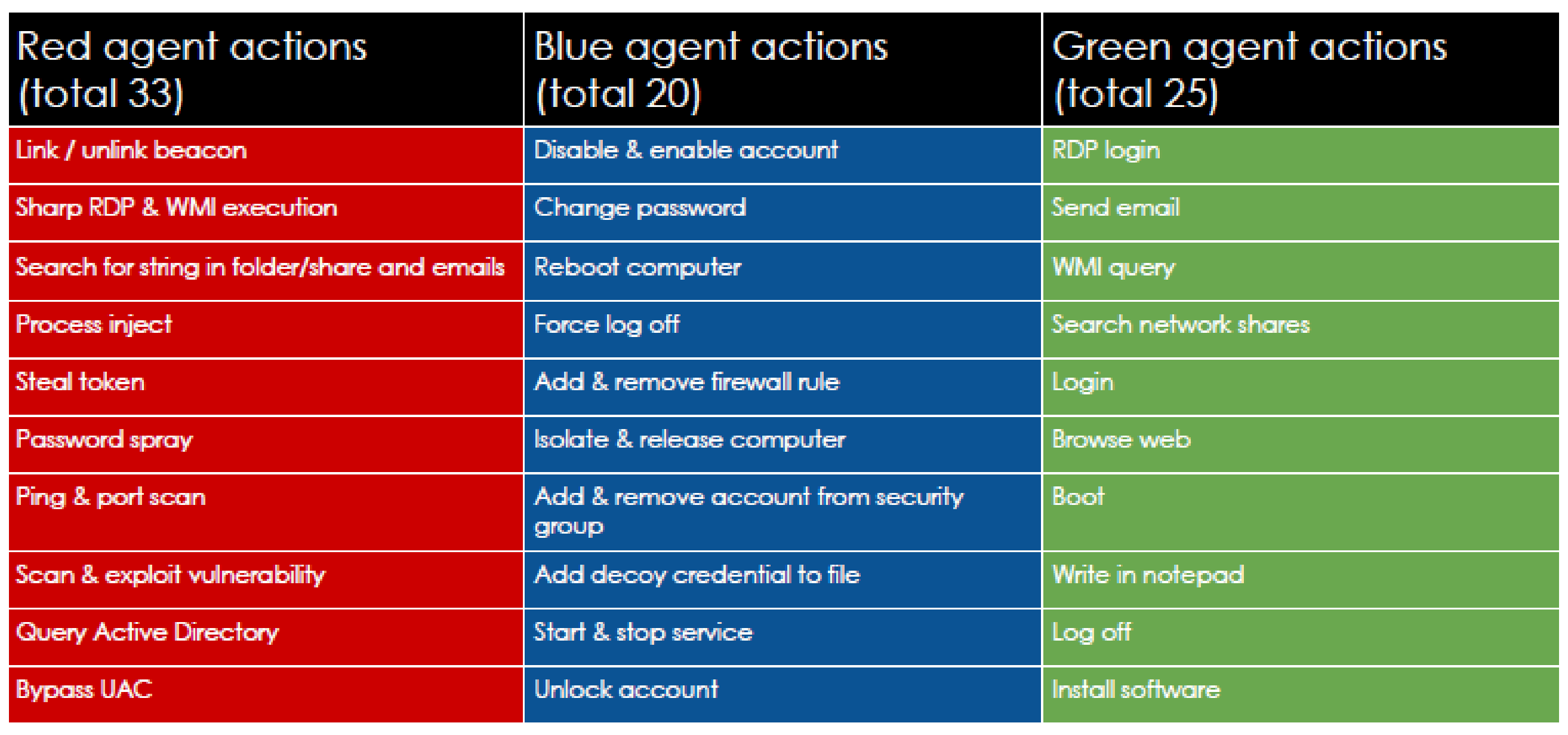


Figure 2: Tools & technologies used in CAESAR



Red agent actions (total 33)	Blue agent actions (total 20)	Green agent actions (total 25)
Link / unlink beacon	Disable & enable account	RDP login
Sharp RDP & WMI execution	Change password	Send email
Search for string in folder/share and emails	Reboot computer	WMI query
Process inject	Force log off	Search network shares
Steal token	Add & remove firewall rule	Login
Password spray	Isolate & release computer	Browse web
Ping & port scan	Add & remove account from security group	Boot
Scan & exploit vulnerability	Add decoy credential to file	Write in notepad
Query Active Directory	Start & stop service	Log off
Bypass UAC	Unlock account	Install software

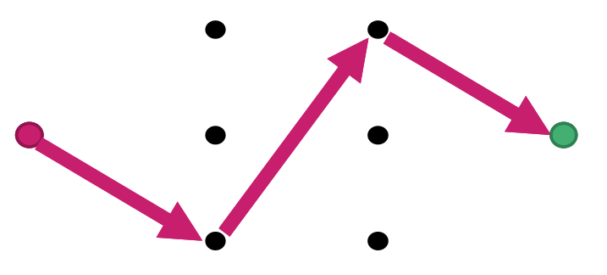
Figure 3: CAESAR action space

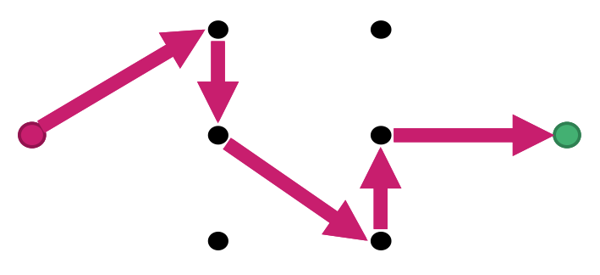
RL red agents developed with the UK's largest national telecoms provider

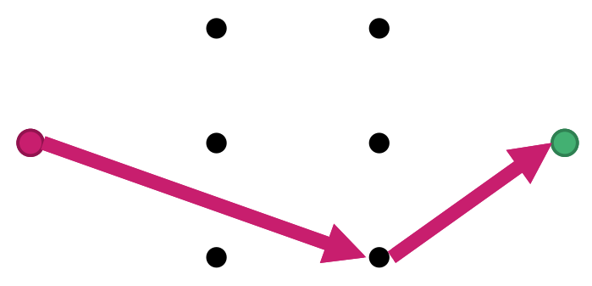
Red agents start with a foothold in the network and no other information. They must uncover new devices and vulnerabilities. Our RL red agent actions are:

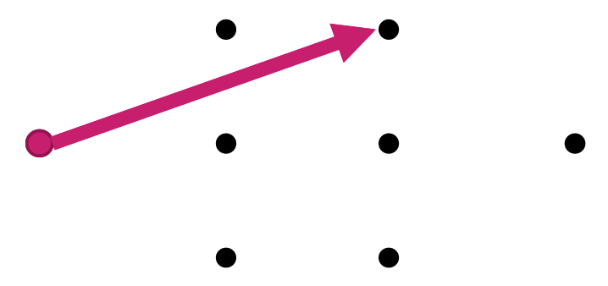
- **Network discovery** – discover other devices in the network that are within its range.
- **System discovery** – discover information about a specific device, e.g., vulnerabilities.
- **Exploit** – run an exploit on a target device in an attempt to infect it and gain control.

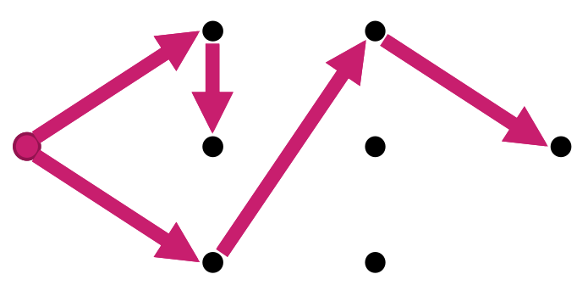
Red agents strategies were developed with BT's Offensive Security Team and represented in reward functions and trained within BT's Inflame cyber simulation tool. Strategies included:


Fastest Time to Target


Stealthiest Path to Target


Shortest Path to Target


Stealthy Persistence


Maximum Coverage

For the fastest route to target strategy the agent is given a target node tag which it must find and exploit in the network. Trained red agents were able to find and exploit their designated target in networks with 10 nodes (Figure 4). We are now training red and blue agents together to explore co-evolution in larger, more representative network topologies.

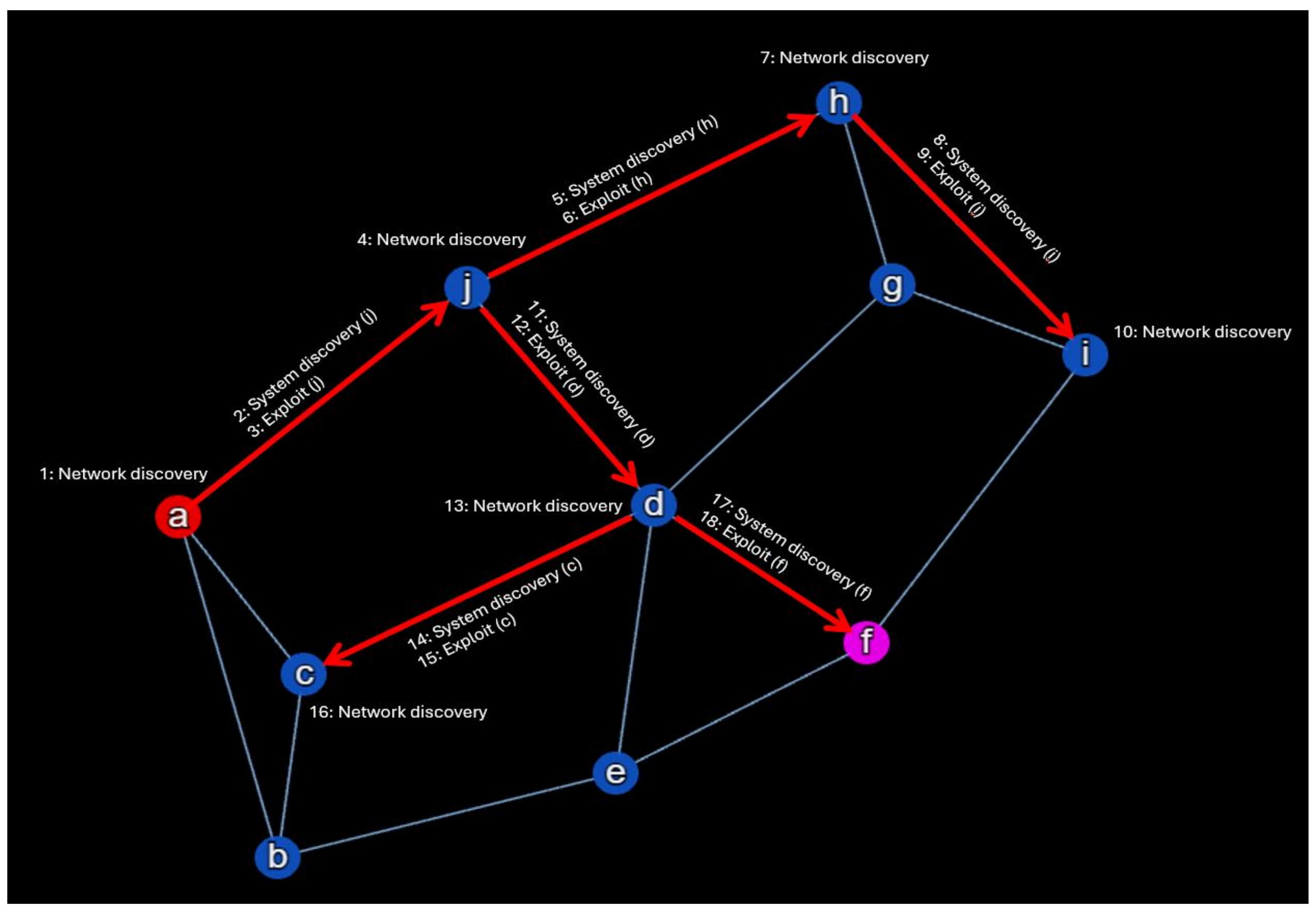


Figure 4: Example of trained red agent action sequence
(fastest time to target strategy, initial outbreak on device 'a', targeting device 'f')

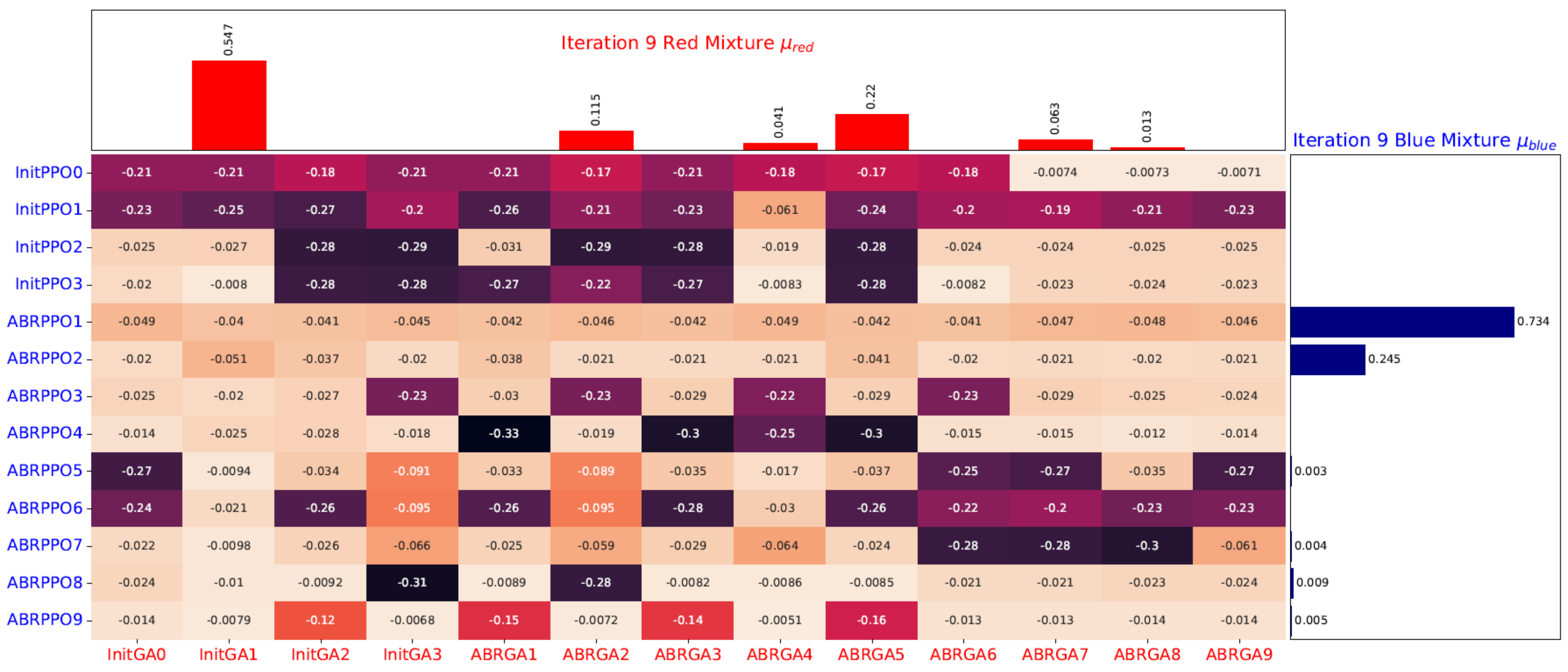
Reducing the exploitability of cyber defence agents

Introduction of ACD agents also introduces a new attack surface to our system. BAE Systems have developed a principled adversarial learning framework which minimises exploitability[1]. The framework is underpinned by a novel Multiple Response Oracle approach, which generates a weighted ensemble of cyber-defence agents, determined via a game theoretic evaluation of generations of agents. The resulting policy sampling weights are determined by the Nash equilibrium for Blue and Red policies (Figure 5). We empirically evaluated our framework, finding it reduced blue agent exploitability[1] (below).

Environment	Red Agent	Initial Exploitability	Final Exploitability
CAGE Challenge 2	Deep RL (PPO)	39.52	-12.50 or 132% reduction: red unable to win
PrimAITE	Genetic Algorithm (GA)	0.01938	0.0036 or -81% reduction

[1] "Exploitability" is the difference in the payoff that Red receives when switching from its current TTPs to the set of TTPs that our current Blue agent finds most challenging (the Approximate Best Response (ABR)). If exploitability ≤0, then red was unable to find an ABR against blue, which is our desired outcome.

Figure 5 → : A simple empirical payoff matrix capturing the payoffs observed when pairing blue & red agents in PrimAITE. To the right and at the top of the payoff table we visualize the optimal Blue and Red mixtures, computed via a Nash solver:



	InitGA0	InitGA1	InitGA2	InitGA3	ABRGA1	ABRGA2	ABRGA3	ABRGA4	ABRGA5	ABRGA6	ABRGA7	ABRGA8	ABRGA9
InitPPO0	-0.21	-0.21	-0.18	-0.21	-0.21	-0.17	-0.21	-0.18	-0.17	-0.18	-0.0074	-0.0073	-0.0071
InitPPO1	-0.23	-0.25	-0.27	-0.2	-0.26	-0.21	-0.23	-0.061	-0.24	-0.2	-0.19	-0.21	-0.23
InitPPO2	-0.025	-0.027	-0.28	-0.29	-0.031	-0.29	-0.28	-0.019	-0.28	-0.024	-0.024	-0.025	-0.025
InitPPO3	-0.02	-0.008	-0.28	-0.28	-0.27	-0.22	-0.27	-0.0083	-0.28	-0.0082	-0.023	-0.024	-0.023
ABRPPO1	-0.049	-0.04	-0.041	-0.045	-0.042	-0.046	-0.042	-0.049	-0.042	-0.041	-0.047	-0.048	-0.046
ABRPPO2	-0.02	-0.051	-0.037	-0.02	-0.038	-0.021	-0.021	-0.021	-0.041	-0.02	-0.021	-0.02	-0.021
ABRPPO3	-0.025	-0.02	-0.027	-0.23	-0.03	-0.23	-0.029	-0.22	-0.029	-0.23	-0.029	-0.025	-0.024
ABRPPO4	-0.014	-0.025	-0.028	-0.018	-0.33	-0.019	-0.3	-0.25	-0.3	-0.015	-0.015	-0.012	-0.014
ABRPPO5	-0.27	-0.0094	-0.034	-0.091	-0.033	-0.089	-0.035	-0.017	-0.037	-0.25	-0.27	-0.035	-0.27
ABRPPO6	-0.24	-0.021	-0.26	-0.095	-0.26	-0.095	-0.28	-0.03	-0.26	-0.22	-0.2	-0.23	-0.23
ABRPPO7	-0.022	-0.0098	-0.026	-0.066	-0.025	-0.059	-0.029	-0.064	-0.024	-0.28	-0.28	-0.3	-0.061
ABRPPO8	-0.024	-0.01	-0.0092	-0.31	-0.0089	-0.28	-0.0082	-0.0086	-0.0085	-0.021	-0.021	-0.023	-0.024
ABRPPO9	-0.014	-0.0079	-0.12	-0.0068	-0.15	-0.0072	-0.14	-0.0051	-0.16	-0.013	-0.013	-0.014	-0.014