Selecting Exploits for Capture the Flag Security Games – Literature Review

Crosley, Zackary
SEFCOM, CIDSE
Arizona State University
Tempe, United States of America
zcrosley@asu.edu

***Abstract*—Cyber warfare scenarios present new challenges for national defense and strategic offense. Research has been done on the strategies for a state actor to follow in the cyber warfare environment, often through the lens of Capture the Flag (CTF) games, as they share many of the same decisions. One decision that is more often unique to Capture the Flag is the availability of multiple vulnerabilities to exploit on a single service. In this paper, we review the previous research into strategies for electronic warfare using Markov chains and game theory. An implementation of a new Markov Decision Process (MDP), known as a Functional Reward Markov Decision Process (FRMDP) which will be used as a basis for future research. The final project goal is to produce an optimal policy for a CTF attack and defense player on what exploits to run and when.**

Keywords — Markov decision process, ctf, cybersecurity, cyber strategy, cyber warfare, functional rewards.

I. Literature Review

Extensive research has been done into the optimal strategies to be used in cyber security and cyber war scenarios. With the rise of the internet, the opportunities for electronic attacks to combat have risen dramatically. While electronic warfare can work alongside traditional "kinetic" combat, it follows very different rules in many key ways. To articulate this, researchers Duggan and Parks have produced a paper detailing the unique characteristics of cyber warfare that must be considered when developing strategies [1]. This is kin to the principles of war that have been produced over millennia for kinetic warfare through texts like Sun Tzu's 'The Art of War'. Duggan and Parks identified eight such defining characteristics for cyber-attacks:

- Lack of Physical Limitations – Cyber weapons do not suffer from the same physical restrictions as kinetic weapons, including the need for locality, transportation, and physical resources to produce.
- Kinetic Effects – To be effective, cyber weapons must cause physical effects to real world objects. This could include damaging infrastructure or manipulating information to influence real-world decisions.
- Stealth – All cyber offensives leave traces if the subject is looking in the correct places. Since all traffic is visible, stealth is blending in as normal traffic.
- Mutability and Inconsistency – Cyberspace is constantly changing, where behavior change chaotically in response to differing environments. As a result, software is less reliable than kinetic hardware.
- Identity and Privileges – Impersonation of higher authority users is key to most exploits.
- Dual Use – Where most kinetic weapons are developed with a singular purpose, computers are inherently multipurpose. Software used for offensive security are the same as those used in defensive security.
- Infrastructure Control – Where in the physical world the military controls their own infrastructure, the military only controls about 10% of their cyber infrastructure, the rest owned by third party companies.
- Information as Operational Environment – There is no sensor encoding the operational environment as data, as the operational environment is data.

These base principles setup the base for understanding cyber warfare strategy. These concepts have been applied in subsequent research papers analysing the optimal policy for various cyber warfare scenarios. One such analysis compared cyberattack strategies to US nuclear policy, contrasting the differences to an idealized cyber policy [2]. The researcher identified that while the source of a nuclear strike would be determinable with near certainty, a cyber attack may be near impossible to source. Additionally, an attacker would not know the response of another country to a major cyber attack – the US for instance has no stated policy on retaliation to electronic warfare and their capabilities are not public. Finally, and possibly most importantly, defense is not a meaningful deterrent against nuclear strikes – the risk of defense failing is to high. In cyber however patching the vulnerability is the main method of deterrence. This implies a strategy of defense, unlike with nuclear policy, is a valid means of deterring attack.

While the previous policy analysis was from a high level, political view, other research used game theory and Markov chains to produce optimal policies. One such paper by T. Moore asks if offensive security is a meaningful means of security and uses a Markov chain to produce an optimal policy. The first simulation ran identified the optimal policy choosing between stockpiling and disclosing exploits. This model considers these two actions with states varying the likelihood of each party to discover the exploit over an infinite horizon. The optimal strategy indicated that an exploit should be stockpiled if neither

party is likely to find it. In a separate analysis of whether to immediately attack or disclose. In this analysis, the optimal policy was only to defend if the odds of finding the vulnerability was low for this player and the other player is likely to attack if they find the attack. The main takeaway from this research being that the optimal policy for each party always includes attacking under certain circumstances. Research building off of this to take into account all the possible actions at once revealed a different result [8]. When actors are choosing whether to attack, disclose, or stockpile they choose to attack only when both are likely to discover the exploit. However, when exploit reflection, that is returning the exploit to the attacker, is guaranteed the nash equilibrium results in every exploit being disclosed; a far more desirable outcome than the first analysis.

One common security strategy topic is network security, do to the extensive efforts by malicious parties to deliver malicious code through the internet. A survey of other papers written on game theory applied to network security found multiple such analysis across all different types of games [6]. Implementations of static games with complete information performed risk assessments, and those with incomplete information modeled DDOS attacks and intrusion detection. Most models surveyed were dynamic models, with imperfect models being used for finding equilibriums and incomplete perfect models for modeling worm propagation through a network. One particularly interesting example of a complete and perfect dynamic game the paper referenced calculated the Nash equilibrium between one network defender and many network attackers [3]. Attackers were able to deface a website, launch a DOS attack, sniff traffic, steal data, and a few other typical hacking actions. Network administrators oversaw a network of three machines (a file system, a workstation, and a web server) each with their own states and had actions to remove worms, block users, patch holes, etc. This game, unlike many other models, was not zero sum – the network could lose more than the attackers gain. In their analysis, the researchers found that on average the maximum damage could be performed by the attackers through simply defacing the website, as more complicated attacks network administrators were more likely to see and prevent. They also found a network administrator always benefitted from removing the attacker as soon as possible, rather than monitoring them in hopes of finding attacker information or more vulnerabilities.

While most of the network security research focused on game theory, some did use a Markov-chain to detect cyber attacks on a network. In one such paper, Markov chain was trained on real world data from 1613 audit events on an MIT server [5]. Rather than looking just at the immediate previous state, like a normal Markov model, this algorithm looked at all items within a window of size N and would use the last N events to determine if the traffic was considered normal. Since the sample size was small, unseen events were given non-zero probability in favor of a small probability. The researchers found this model accurately detected network anomalies when the training data contained no noise. Given more realistic training material, however, led to a less successful result. Due to the inability of the model to handle noise, the researchers determined that the model was not ideal for this purpose.

One final paper implemented a generalization of the MDP algorithm for functional rewards [7]. This paper was of interest since modeling a CTF as an MDP will have a more complicated reward than a simple number, or a decreasing number, and the Functional Reward MDP in the paper provided a nice abstraction for the reward to be separated from the rest of the dynamic programming algorithm. Using this MDP as a base and some ideas from other security strategies in these papers, an MDP for CTF attacks may be formed.

II. Benchmark Implementation

The base algorithm chosen to implement as the basis for the CTF attacking MDP was the Functional Reward MDP by Spanjaard and Weng due to its clean methodology of adding a more generalized reward function to the standard MDP. A standard MDP uses the recursion:

$$v_t^*(s) \leftarrow r_s^a + \sum_{s' \in S} p_{ss'}^a v_{t-1}^*(s')$$

This reward function does not consider the next state, s', nor the value of the next state. Thus to account for such dependencies, the recursive function itself must be altered, making it hard to read and easily identify as the MDP dynamic programming algorithm.

Take for instance the sample problem provided in the paper to demonstrate this difficulty. Suppose we have a robot on a 3x3 grid representing an office whose job is to bring coffee from section (1,1) to a researcher in section (3,3). The robot can move North, South, East or West and with each action has a small chance to spill the coffee while bringing it to the researcher. If the coffee spills, the game is over. The office is made from a variety of different flooring material, and as a result the probability of spilling the coffee and the negative reward from doing so vary by square. Additionally, the robot tends to drift to the left of its intended square and will do so with a ten percent probability.

(3,3)

-1	-2	0
-3	-3	-4
-1	-2	-1

(1,1)

(3,3)

1%	5%	3%
2%	4%	2%
3%	5%	3%

(1,1)

Figure 1: The costs from the above example by state (left) and the probabilities (right).

Encoding this MDP requires modifications to the base function. Since the probabilities vary by state, these values must be added to the MDP algorithm as follows:

$$v_t^*(s) \leftarrow P_s r_s + (1 - P_s) \sum_{s' \in S} p_{ss'}^a v_{t-1}^{\pi}(s')$$

Where P_s indicates the probability of spilling the coffee and r_s indicates the negative cost of spilling the coffee. While this is doable, it does make the function far less readable. Furthermore it prevents any simple abstraction of the generic MDP process for all implementations, resulting in individual solutions for each problem. Functional Reward MDPs solve this by reforming the algorithm into:

$$v_t^*(s) \leftarrow \sum_{s' \in S} p_{ss'}^a f_{ss'}^a \left(v_{t-1}^\pi(s')\right)$$

Where $f_{ss'}^a$ is the reward function that returns the reward evaluated from the current state, the action, the next state, and the utility of the next state. In this model, a standard MDP is implemented by setting $f_{ss'}^a \left(v_{t-1}^\pi(s')\right) = r_{ss'}^a + v_{t-1}^\pi(s')$.

To implement the coffee robot, a generic FRMDP class was constructed. This class took a list of states, a list of actions, a list of termination rewards by state, the horizon (number of iterations to run), and a termination array. The termination array is a tensor where the index of a state, action, and next state from the preceding lists maps to $p_{ss'}^a$. An is_terminal function and get_reward function is required to be implemented by the child class. A value iteration function implements the value iteration algorithm which is applicable to any child class provided it implements the is_terminal and get_reward method. Similarly, a policy_iteration and simulation function are provided that run iteration over a policy and run a policy from a start state to a terminal state respectively. For efficiency purposes, the termination tensor is implemented using the Numpy library and uses vector math to quickly evaluate over all possible next_states for a given action. The value iteration and policy iteration also use memoization to quickly look up the value of states from previous iterations, though this is at the expense of high memory usage.

Since creating the tensor by hand would be time consuming (9 states and 4 actions results in 324 cells), a generate_transition script was developed. This iterates over the transition tensor and fills it with values derived from a set of provided lambda functions. This creation model should generalize from any set of lambdas, thus allowing for the efficiency of the tensor representation of transitions with the ease of specifying simple transformation functions.

This base class was extended to form a standard MDP with a reward function $r_{ss'}^a + v_{t-1}^\pi(s')$ for comparison purposes. From these two classes, a BasicCoffeeRobot and CoffeeRobot were produced. The BasicCoffeeRobot extended MDP as a control and the CoffeeRobot extended the MDP with a fixed probability of spilling coffee. The FRMDP coffee robot with differing probability of spilling produces a similar policy, though with the key difference of choosing to move up in state (1,2) instead of right to avoid the higher probability of spilling for less immediate reward. This matches the expected output described in the paper and can be seen below.

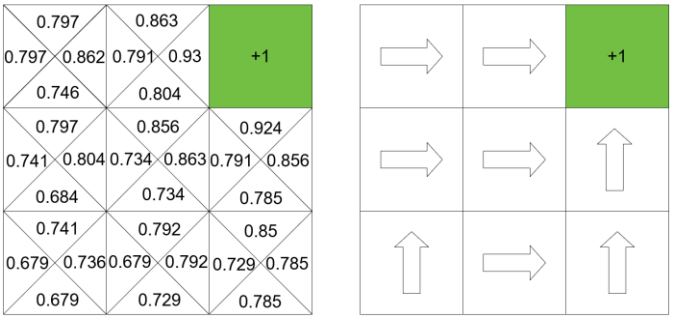

Figure 2: MDP Coffee Robot without state dependent spill probability.

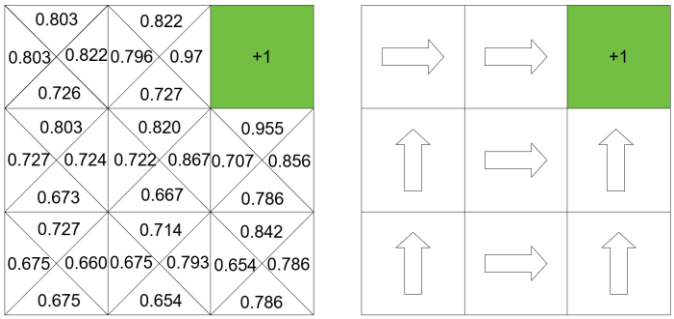

Figure 3: FRMDP Coffee Robot with state dependent spill percentage.

To apply this to the original question, what exploit should be run in a CTF out of many (if you have determined to exploit), we will adjust this MDP to evaluate an optimal exploit policy. Our initial state will be a tuple of M Ns, where M represents the number of exploits we have available and N the number of opponents. This indicates that at the beginning of the game, each exploit will earn one point from each opponent. With each turn there is a probability of any arbitrary reward count dropping by one as a player patches their service from the exploit. With each use of an exploit, this probability goes up the exploit that was used, indicating the likelihood of a victim of this attack spotting the network traffic and repairing their service. The valid actions will be the M exploits plus NoOp (do nothing). This model will be conducted to simulate an in-class ctf (common for security courses), which would be one hour with approximately sixty ticks (a horizon of 60).

Additional modifications may be made as needed for this implementation. After implementation testing if it is found that this simplified score model is not sufficient and would be better replaced with a POMDP or Stochastic Game, the FRMDP developed will be generalized further to support the necessary functionality.

References

[1] D. P. Duggan and R. C. Parks, "Principles of Cyberwarfare," in IEEE Security & Privacy, vol. 9, no. , pp. 30-35, 2011. doi:10.1109/MSP.2011.138

[2] D. Elliott, "Deterring Strategic Cyberattack," in IEEE Security & Privacy, vol. 9, no. 5, pp. 36-40, Sept.-Oct. 2011. doi: 10.1109/MSP.2011.24I.

[3] Lye, K. & Wing, J. IJIS (2005) 4: 71. https://doi.org/10.1007/s10207-004-0060-x

[4] Moore, Tyler et al. “Would a 'cyber warrior' protect us: exploring trade-offs between attack and defense of information systems.” NSPW (2010).

[5] Nong Ye, Yebin Zhang and C. M. Borror, "Robustness of the Markov-chain model for cyber-attack detection," in IEEE Transactions on Reliability, vol. 53, no. 1, pp. 116-123, March 2004. doi: 10.1109/TR.2004.823851

[6] S. Roy, C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya and Q. Wu, "A Survey of Game Theory as Applied to Network Security," 2010 43rd Hawaii International Conference on System Sciences, Honolulu, HI, 2010, pp. 1-10. doi: 10.1109/HICSS.2010.35

[7] Spanjaard O., Weng P. (2013) Markov Decision Processes with Functional Rewards. In: Ramanna S., Lingras P., Sombattheera C., Krishna A. (eds) Multi-disciplinary Trends in Artificial Intelligence. MIWAI 2013. Lecture Notes in Computer Science, vol 8271. Springer, Berlin, Heidelberg

[8] T. Bao, Y. Shoshitaishvili, R. Wang, C. Kruegel, G. Vigna and D. Brumley, "How Shall We Play a Game?: A Game-theoretical Model for

Cyber-warfare Games," 2017 IEEE 30th Computer Security Foundations Symposium (CSF), Santa Barbara, CA, 2017, pp. 7-21. doi: 10.1109/CSF.2017.34