# Dynamic Information Flow Tracking Games for Simultaneous Detection of Multiple Attackers

Dinuka Sahabandu, Shana Moothedath, Joey Allen, Andrew Clark, Linda Bushnell, Wenke Lee, and Radha Poovendran

*Abstract*— **Dynamic Information Flow Tracking (DIFT) has been proposed to detect and prevent various cyber attacks in computer systems. DIFT tracks suspicious information flows in the system and generates security analysis when anomalous behavior is detected. A system threatened by attackers of different capabilities demands simultaneous analysis of multiple flows. As the number of information flows in a system is typically large and the amount of resource required for analyzing different flows varies, an optimal allocation of the limited resources available to DIFT is essential. We address the problem of detecting multiple attackers using resource constrained DIFT and develop a model that captures the interaction of adversaries and a DIFT-based defender as a multi-player dynamic game. Our model consists of a multi-stage game, in which each stage represents the subset of processes in the system that correspond to the locations of the information flows, and captures the notion of benign flows. Given the attackers' strategies, we prove that finding an optimal defense strategy is equivalent to maximizing an increasing DR-submodular function that enables us to propose an approximation algorithm. Further, given a defense strategy and strategies of other attackers, we show that finding an optimal attacker strategy is equivalent to solving a shortest path problem, where the edge weights are derived from the strategies of the other players. Based on this mapping we propose a polynomial-time algorithm for computing an optimal attacker strategy. Finally, we evaluate the performance of our algorithm on a real-world dataset of a nation state attack obtained using the Refinable Attack INvestigation (RAIN) framework.**

## I. Introduction

With the increasing proliferation of information and communication technologies, cyber security is a global issue affecting companies and organizations of all sizes. A great deal of effort has been placed into discovering new ways to determine the origin and nature of cyber attacks, including signature-based and behavior-based detection methods. As modern attacks become more stealthy, adaptive, and persistent detecting them using conventional security schemes is challenging. However, adversarial interactions introduce information flows consisting of data and control commands in the system (e.g., an instance of a computer program). Dynamic Information Flow Tracking (DIFT) is a detection mechanism that has been proposed and widely used to detect attacks by analyzing the information flows in a system [1].

DIFT tags suspicious information flows, tracks their propagation across the system, and performs security analysis if the system observes any anomalous behavior. As the number of benign flows in a system is large in comparison to the adversarial flows, it is essential to minimize the allocation of the limited memory of DIFT in analyzing tagged benign flows. Further, when a system is threatened by multiple simultaneous attacks, in addition to differentiating benign and adversarial flows, DIFT must also detect adversarial flows introduced by multiple attackers.

The granularity of security analysis for detection and risk assessment varies across different types of attackers [1], [2]. For instance, while a fine-grain level analysis to detect an Advanced Persistent Threat (APT) incurs high memory, a coarse-grained analysis with lesser memory suffices to identify cyber criminals [1]. Any modeling framework should accommodate these levels of granularities to address the fundamental trade-off between detection of multiple attackers, which makes the memory allocation problem further challenging.

In this paper we present an analytical model of a resource constrained DIFT that analyzes a set of information flows to simultaneously detect $K$ attackers in a system. The effectiveness of detection depends on the actions of the attackers and the attackers' chances of achieving their goals depend on the allocation of the defense resources. This strategic interaction motivates a game theoretic modeling as it allows us to investigate the trade-off between detection probability of different attackers and memory allocation of DIFT. The $(K+1)$-player game is dynamic where at each stage the attackers choose their next operation in the system and the defender chooses the type of analysis at each flow which characterizes the allocation of the memory.

We make the following contributions in this paper:

- We prove that finding an optimal defense strategy against given attackers' strategies is equivalent to maximizing an increasing DR-submodular function subject to a polytope constraint. Using the equivalence, we provide an algorithm to compute an approximate optimal strategy of the defender.
- We show that finding an optimal attacker's strategy, for given strategies of other attackers and the defender, is equivalent to solving a shortest path problem on the information flow graph of the system. Based on this mapping we propose a polynomial-time algorithm for computing an optimal attacker strategy.

D. Sahabandu, S. Moothedath, L. Bushnell, and R. Poovendran are with the Department of Electrical and Computer Engineering, University of Washington, Seattle, WA 98195 USA. {sdinuka,sm15,lb2,rp3}@uw.edu.

A. Clark is with the Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA 01609 USA. aclark@wpi.edu.

J. Allen and W. Lee are with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332 USA. jallen309@gatech.edu, wenke@cc.gatech.edu.

This work was supported by ONR grant N00014-16-1-2710 P00002 and DARPA TC grant DARPA FA8650-15-C-7556.

- We evaluate the performance of our algorithm on data sets collected using Refinable Attack INvestigation (RAIN) system and analyze the optimal system memory allocation.

This paper is organized as follows: Section II summarizes the related work. Section III introduces the notion of information flow graph and then discusses the attacker and defender models. Section IV describes the formulation of the dynamic game between DIFT and multiple adversaries. Section V analyzes the best responses of the players of the game. Section VI presents the numerical results. Section VII gives the concluding remarks and future work.

## II. Related Work

Game theoretic framework is used to analyze security and privacy in computer and communication networks in [3]. A zero-sum game is presented in [4] to model the interaction between malicious attackers and the Intrusion Detection System (IDS) who allocates system resources for detection. A nonzero-sum game model is given in [5] to analyze joint threats from APT attacker and the insiders and the best response of the players are derived. In contrast to references [3]-[5], our model focuses on a case where multiple attackers, such as APTs and cyber criminals, interplay with a resource constrained defense mechanism in a system environment rather than a network domain.

Multiple actors competing against a common resource in the system domain is modeled as a *FlipIt* game in [6], [7]. In our case the multiple adversaries have different targets to achieve in the system and they strategize against a common defender. The CPU allocation problem between an APT attacker and the system is modeled as a *colonel-blotto* game in [8]. An extensive game model to detect APTs in a system is given in [9]. Our focus is to develop a game model that evaluates the feasibility of a resource constrained DIFT, a widely used information flow tracking-based real-world dynamic defense strategy, against multiple attackers that include APTs and cyber criminals.

Defending against APTs using a DIFT-based defense is studied in [10], [11]. A stochastic variant of the problem is later considered in [12]. In contrast to references [10]-[12], this paper consider the case of simultaneous detection of multiple adversaries by a resource constrained defender. In our analysis, we use the notion of DR-submodularity. Submodular functions are a wide class of non-convex/non-concave continuous functions with constant approximation guarantees [13]. They find applications in combinatorial optimization [14] and machine learning [15]. To the best of our knowledge, this is the first time DR-submodularity is used in the context of a security problem.

## III. Preliminaries

### A. Information Flow Graph (IFG)

An IFG [1], $\mathcal{G} = (V_\mathcal{G}, E_\mathcal{G})$, is a graphical representation of log data collected from a system, where $V_\mathcal{G} = \{v_1, \ldots, v_N\}$ and $E_\mathcal{G} \subset V_\mathcal{G} \times V_\mathcal{G}$. Here, $\mathcal{G}$ represents the whole-system execution during the entire period of logging and $N$ denotes the total number of nodes in the IFG. The nodes of the IFG represent processes (e.g., an instance of a computer program), file objects, network sockets and memory objects. The edges represent the feasibility of pushing information flows from one process to another process. We use IFGs in our formulation to detect adversaries in a system.

### B. Attacker Model

We consider an attack model that consists of $K$ adversaries, denoted $\mathcal{P}_{A_1}, \ldots, \mathcal{P}_{A_K}$, of varying attack capabilities. Highly capable adversaries, such as nation-states, are capable of developing APTs with sophisticated targets and attack goals. Additionally, less capable adversaries, such as small cyber criminal gangs may also attack the system, for purposes such as creating a botnet. Without loss of generality, assume that $\{\mathcal{P}_{A_k}\}_{k=1}^{K}$ is an ordered list in the increasing order of the attacking capabilities of the adversaries. We capture the different capabilities of the attackers in the payoff functions of the players (see Section IV).

Adversaries enter into the system through a subset of nodes $\lambda \subset V_\mathcal{G}$ in the IFG $\mathcal{G} = (V_\mathcal{G}, E_\mathcal{G})$. Each adversary has a distinct goal to achieve in the system. The goals of adversaries are identified in the IFG as a set of destination nodes denoted by $\mathcal{D} := \{d_1, \ldots, d_K\} \subset V_\mathcal{G}$, where each $d_k$ is the specific target of the $k^{\text{th}}$ adversary $\mathcal{P}_{A_k}$. For all $k \in \{1, \ldots, K\}$, adversary $\mathcal{P}_{A_k}$ performs operations in the system, i.e., transition between nodes in the IFG, so as to reach its goal $d_k$. Furthermore, all the adversaries are capable of abandoning their respective attacks before reaching their corresponding destination nodes to avoid getting detected by the defender.

### C. Defender Model: DIFT

We consider a DIFT-based defense mechanism [2]. DIFT consists of three main components: (i) tainting at tag sources, (ii) tag propagation policies, and (iii) trapping at tag sinks. Firstly, DIFT taints all the information flows incoming to the set of nodes $\lambda \subset V_\mathcal{G}$ in the IFG, where $\lambda$ denotes the set of vulnerable locations in the system which an adversary can exploit to enter into the system. Then, DIFT tracks the propagation of the tainted flows through the system during various system operation. Finally, DIFT performs security analysis called 'trapping' in order to verify the authenticity of the tagged information flows when an unauthorized usage of a tainted flow is detected. While DIFT is widely used for detection of cyber attackers (see [2], [1] and the references therein), its excessive memory and runtime overhead makes it difficult to integrate into ordinary systems. In this paper, we consider a memory constrained DIFT to defend against multiple adversaries.

We consider a defender who observes $W$ number of tagged flows at each time instance, where $W > K$, and aims to simultaneously detect the $K$ adversaries before they achieve their respective goals. Out of the $W$ tagged flows, $K$ tagged flows correspond to $K$ adversarial flows which are indistinguishable to the defender from the $(W-K)$ tagged benign flows. DIFT can detect an adversarial flow of $k^{\text{th}}$ adversary by performing type-$k$ security analysis on the tagged flows, where $k = 1, \ldots, K$. The amount of resources such as memory and processing power required to do type-$k$ trapping varies with the capabilities of the $k^{\text{th}}$ adversary and also depends on the trapping location of the tagged flow in the IFG. Moreover, the amount of resources (memory) that can be allocated for trapping at each time instance is constrained because of the limited memory. Hence, DIFT can incur additional resource cost by undertaking any type-$k$ trapping on tagged benign flows (cost of false positives).

Let $M$ denote the total amount of memory available to the DIFT at each time instant when it observes $W$ tagged flows at different processes and files in the system. In our model consisting of multiple attackers, the resource constrained DIFT at each time step needs to allocate its available memory, $M$, among different security analyses across $W$ tagged flows to maximize the probability of detection and minimize the harm caused by the attackers to the system.

## IV. Game Formulation

In this section, we model the $(K+1)$-multi-player dynamic game, $\Gamma$, between the defender player $\mathcal{P}_D$ and the adversarial players $\mathcal{P}_{A_1},\dots,\mathcal{P}_{A_K}$. All players act simultaneously at each stage of the game and no player can observe the action of any other player. As a result, the game is an *imperfect* information game. We assume that all players are rational and know the payoff function and the goal of every other player. Hence the game is a *complete* information game. The game evolves in the time $t = 0,1,2,\dots,T$, where $t = T$ denotes the termination time of the game and $t$ is referred to as the stage of the game.
**State Space**: Let $\mathbf{S}$ denote the state space of the game. Then each state $\mathbf{s} \in \mathbf{S}$ is structured as $\mathbf{s} = (s_1,\dots,s_W)$. The entries $s_1,\dots,s_W$ represent the current status of the $W$ tagged information flows in the system as observed by the defender at time $t$. We let the first $K$ entries in any state $\mathbf{s}$, i.e., $s_1,\dots,s_K$, represent the status of the $K$ adversarial flows in the IFG. The defender observes all the entries in $\mathbf{s}$ but is unaware of whether each tagged flow observed at $s_1,\dots,s_W$ is benign or adversarial. Each type-$k$ adversary only observes the $s_k^{\text{th}}$ entry of a state $\mathbf{s}$. Notice that the players of the game have asymmetric information on the states of the game. Furthermore, we introduce a new state, $\mathbf{s_0}$, where $s_1 = \dots = s_W = s_0$, to denote the initial state of the game at time $t = 0$. We augment $s_0$ into the underlying IFG of the game as a virtual node. The set of outgoing edges from the virtual node, $s_0$, to each node in the set $\lambda \subset V_\mathcal{G}$ (entry points of the $K$ adversaries) models the initial foothold of the $K$ adversaries in the system.

For a state $\mathbf{s} = (s_1,\dots,s_W)$, $s_i = v_j$ indicates the arrival of $i^{\text{th}}$ information flow to node $v_j \in V_\mathcal{G}$ in the IFG. Here, $i = 1,\dots,W$ and $j = 1,\dots,N$. We let $s_i = \phi$, for any $i = 1,\dots,W$, represent the case where $i^{\text{th}}$ information flow observed by the DIFT is no longer being continued in the system (i.e., dropped out). Also, $s_k = \tau$, for any $k = 1,\dots,K$, represents the case where DIFT has successfully trapped the $k^{\text{th}}$ adversarial flow before it reaches its destination node $d_k$. Moreover, if any $s_i = \phi$ (or $s_k = \tau$) for a state $\mathbf{s}$ at time $t$, then that entry $s_i$ ($s_k$, respectively) will remain at $\phi$ ($\tau$, respectively) for any future state $\mathbf{s}'$ at time $t' > t$. Hence, $s_k \in V_\mathcal{G} \cup \{\phi,\tau\}$ for all $k = 1,\dots K$ and $s_i \in V_\mathcal{G} \cup \{\phi\}$ for all $i = K+1,\dots W$. The total number of states in the game is $O((N+2)^K N^{W-K}) + 1$.

A state $\mathbf{s} = (s_1,\dots,s_W)$ is a terminal state if $s_k \in \{d_k,\phi,\tau\}$, for all $k \in \{1,\dots,K\}$. Let the set $\mathbf{S}_T \subset \mathbf{S}$ denote the set of terminal states in $\Gamma$. This implies that at a state $\mathbf{s} \in \mathbf{S}_T$, each of the $k$ adversaries either reached its respective destination $d_k$ in the system or abandoned the attack or is detected by the defender before reaching its goal.
**Action Space**: The action sets of players are finite. At every state in the game except the terminal states $\mathbf{s} \in \mathbf{S}_T$, adversarial players $\mathcal{P}_{A_1},\dots,\mathcal{P}_{A_K}$ decide which out-neighboring node of the IFG to transition to or drop the attack. The defender player decides whether to trap or not to trap each tagged information flow, $s_1,\dots,s_W$, observed at time $t$. Moreover, if defender decides to trap a tagged information flow then it must also decide which security policy it should choose to analyze the trapped flow. Let $\mathcal{A}_{A_1},\dots,\mathcal{A}_{A_K}$ and $\mathcal{A}_D$ denote the action sets of $\mathcal{P}_{A_1},\dots,\mathcal{P}_{A_K}$ and $\mathcal{P}_D$, respectively. Then, at a state $\mathbf{s} = (s_1,\dots,s_W)$, $s_k \notin \{d_k,\phi,\tau\}$, $\mathcal{A}_{A_k}(s_k) = \{v_i : (v_j,v_i) \in E_\mathcal{G}$ and $s_k = v_j\} \cup \varnothing$. Here, the action $v_i$ indicates $k^{\text{th}}$ adversary transitioning to a neighboring node in the IFG while $\varnothing$ represents $k^{\text{th}}$ adversary dropping out of the game.

Let $\Pi = \{\pi_1,\dots,\pi_K\}$ denote the set of all $K$ security analysis and $\pi_k \in \Pi$ be the type-$k$ security analysis, where $k = 1,\dots,K$. Then define $\Pi(\mathbf{s}) \subseteq \Pi$ to be the set of security analysis feasible in the state $\mathbf{s}$, i.e., $\pi_k \in \Pi(\mathbf{s})$ if $s_k \neq \{\tau\}$ in $\mathbf{s}$. This implies that after successfully detecting a type-$k$ adversarial flow, DIFT can stop analyzing rest of its tagged flows using type-$k$ analysis as the $k^{\text{th}}$ attacker is already detected by the defender. Then $\mathcal{A}_D(\mathbf{s}) = \mathcal{A}_D(s_1) \times \mathcal{A}_D(s_2) \times \dots \times \mathcal{A}_D(s_W)$, where $\mathcal{A}_D(s_i) = \Pi(\mathbf{s}) \cup \{0\}$ for $i = 1,\dots,W$ gives the set of feasible actions for the defender at the current location of the tagged flow, $s_i$. The action 0 represents the defender not trapping a tagged flow observed at $s_i$. At states $\mathbf{s} \in \mathbf{S}_T$, $\mathcal{A}_{A_1}(\mathbf{s}) = \dots \mathcal{A}_{A_K}(\mathbf{s}) = \mathcal{A}_D(\mathbf{s}) = \emptyset$. In our formulation, the defender is unaware whether an incoming flow is malicious or not and the adversaries do not know whether the adversarial flows are going to get trapped at the nodes they reach at time $t$. The game captures the information asymmetry between the players as all players take actions simultaneously and no player observe others' actions.
**Player Strategies**: If we allow all players to use information obtained from previous game stages (i.e., the previous state transitions, previous trapping strategies) to determine their action at each stage, the size of the set of possible strategies for the defender and the adversaries can be very large. To reduce the computational complexity and to model the fact that trapping an adversarial information flow is a lower-level processes with limited computation capability, we restrict our focus to stationary strategies.

**Definition IV.1.** *A player strategy is said to be stationary if it depends only on the current state of the player in the game.*

We consider mixed strategies in which players randomize over their action spaces. Let $\mathbf{P}_{A_1},\dots,\mathbf{P}_{A_k}$ and $\mathbf{P}_D$ denote the set of (mixed) stationary strategies of the players. Strategy of the $k^{\text{th}}$ adversary at a state $\mathbf{s} \in \mathbf{S}$ is $p_{A_k}(\mathbf{s}) \in \mathbf{P}_{A_k} : s_k \to [0,1]^{|\mathcal{A}_{A_k}(s_k)|}$. The defender's strategy at a state $\mathbf{s}$ is $p_D(\mathbf{s}) \in \mathbf{P}_D : \mathbf{s} \to [0,1]^{W|\mathcal{A}_{D_k}(\mathbf{s})|}$. Furthermore, $p_D(\mathbf{s}) = \begin{bmatrix} p_D^{\mathbf{s}}(s_1) & \dots & p_D^{\mathbf{s}}(s_W) \end{bmatrix}$, where $p_D^{\mathbf{s}}(s_i) = \begin{bmatrix} p_D^{\mathbf{s}}(s_i,1) & \dots & p_D^{\mathbf{s}}(s_i,K) \end{bmatrix}$ for all $i = 1,\dots,W$. $p_D^{\mathbf{s}}(s_i,k)$ denotes the probability of using type-$k$ trapping for a tagged information flow observed at a location corresponding to $s_i$ in the state $\mathbf{s}$. Here, $s_i$ is the location of the $i^{\text{th}}$ tagged information flow observed according to the state $\mathbf{s}$.

**Payoffs**: Payoff of an adversarial player consists of three components. An adversarial player $\mathcal{P}_{A_k}$ gets (i) a reward of $\beta_k^A > 0$ for reaching its destination $d_k$ without being detected by the defender, (ii) a penalty of $\alpha_k^A \leqslant 0$ for getting detected by the defender, and (iii) a penalty of $\sigma_k^A \leqslant 0$ for abandoning the attack by dropping out, for $k \in \{1,\dots,K\}$. Recall that the adversaries are ordered based on their attack capability as $\{\mathcal{P}_{A_1},\dots,\mathcal{P}_{A_K}\}$. Thus, $\beta_1^A \leqslant \dots \leqslant \beta_K^A$, $\alpha_1^A \geqslant \dots \geqslant \alpha_K^A$, and $\sigma_1^A \geqslant \dots \geqslant \sigma_K^A$. On the other hand, the defender payoff consists of four components. The defender player $\mathcal{P}_D$ gets (i) a penalty of $\beta_k^D \leqslant 0$ for $k^{\text{th}}$ adversary reaching its destination $d_k$ without being detected by the defender, (ii) a reward of $\alpha_k^D > 0$ for detecting the $k^{\text{th}}$ adversary, (iii) a reward of $\sigma_k^D \geqslant 0$ if the $k^{\text{th}}$ adversary abandon the attack by dropping out, and (iv) a resource constraint $M$. Then, $\beta_1^D \geqslant \dots \geqslant \beta_K^D$, $\alpha_1^D \leqslant \dots \leqslant \alpha_K^D$, and $\sigma_1^A \leqslant \dots \leqslant \sigma_K^A$. At every state of the game the resource (memory) constraint should be satisfied. Moreover, the memory requirement to perform security analysis for different types of adversaries are different. The amount of memory required to perform security analysis related to highly capable adversaries is higher than the memory required to deploy security analysis to detect less capable adversaries. Also, the memory required to perform security analysis depends on the node in the IFG as the required memory can vary depending on the type of the process (e.g. process, file or network socket) and the busyness of the process. Let $\mathcal{C}_k(v_i)$ denote the amount of memory required to perform type-$k$ trapping analysis at node $v_i \in V_{\mathcal{G}}$. Then, under a policy $p_D \in \mathbf{P}_D$ at every state $\mathbf{s} \in \mathbf{S}$, $\sum_{i=1}^{W}\sum_{k=1}^{K} p_D^{\mathbf{s}}(s_i,k)\mathcal{C}_k(s_i) \leqslant M$.

Let $\bar{p}_T(k)$, $\bar{p}_R(k)$, and $\bar{p}_\phi(k)$ denote the probability that adversary $\mathcal{P}_{A_k}$ is detected by the defender $\mathcal{P}_D$, the probability that adversary $\mathcal{P}_{A_k}$ reach destination $d_k$, and the probability that adversary $\mathcal{P}_{A_k}$ drops out of the game, respectively. Here, $\bar{p}_T(k)$, $\bar{p}_R(k)$, and $\bar{p}_\phi(k)$ are functions of strategies $p_{A_1},\dots,p_{A_K},p_D$. Now we define the payoff functions of the players of $\Gamma$. Let $U_{A_1},\dots,U_{A_K},U_D$ denote the payoff functions of the $K$ adversarial players and the defender player, respectively. For a given strategy, $p_D$ and $p_A = \{p_{A_k}\}_{k=1}^{K} = \{p_{A_1},\dots,p_{A_K}\}$, the payoffs $\{U_{A_k}\}_{k=1}^{K}$, and $U_D$ are

$$U_D(p_A,p_D) = \sum_{k\in\{1,\dots,K\}} \Big(\bar{p}_T(k)\alpha_k^D + \bar{p}_R(k)\beta_k^D + \bar{p}_\phi(k)\sigma_k^D\Big) \quad (1)$$

$$U_{A_k}(p_A,p_D) = \bar{p}_T(k)\alpha_k^A + \bar{p}_R(k)\beta_k^A + \bar{p}_\phi(k)\sigma_k^A. \quad (2)$$

Additionally, the defender policy $p_D$ must satisfy the following memory constraint

$$\sum_{i=1}^{W}\sum_{k=1}^{K} p_D^{\mathbf{s}}(s_i,k)\mathcal{C}_k(s_i) \leqslant M, \text{ for all } \mathbf{s} \in \mathbf{S}. \quad (3)$$

In this paper, our goal is to find best responses of the players which is defined below.

**Definition IV.2.** *Let $p_D : \mathbf{S} \to [0,1]^{|\mathbf{S}|}$ denote a defender strategy and $p_A : \mathbf{S} \to [0,1]^{|\mathbf{S}|}$ denote an adversary strategy. The set of best responses of the defender is given by*

$$BR(p_A) = \arg\max_{p_D}\{U_D(p_D,p_A) : p_D \in [0,1]^{|\mathcal{A}_D|}\}.$$

*Similarly, the best responses of the adversary given by*

$$BR(p_D) = \arg\max_{p_A}\{U_A(p_D,p_A) : p_A \in [0,1]^{|\mathcal{A}_A|}\}.$$

In what follows, we focus on finding best responses of the players of the game.

## V. Best Responses of the players

### A. *Best Response of the Defender*

For a given strategies $p_{A_1},\dots,p_{A_K}$ of $K$ adversarial players, the best response of defender is characterized by the following optimization problem.

**Problem V.1.** The defender's problem is as follows:
$\max_{p_D\in\mathbf{P}_D} \sum_{k\in\{1,\dots,K\}} \Big(\bar{p}_T(k)\alpha_k^D + \bar{p}_R(k)\beta_k^D + \bar{p}_\phi(k)\sigma_k^D\Big)$
*Subject to: $\sum_{i=1}^{W}\sum_{k=1}^{K} p_D^{\mathbf{s}}(s_i,k)\mathcal{C}_k(s_i) \leqslant M$, for all $\mathbf{s} \in \mathbf{S}$.*

There exists a reduction of a general instance of a multi-set cover problem to an instance of Problem V.1 which provides the following hardness result.

**Proposition V.2.** *The defender's best response problem, Problem V.1, is NP-hard.*

We omit the proof of Proposition V.2 in the interest of space. Also note that Problem V.1 is a non-convex optimization problem. To this end, we investigate on additional structure of the payoff function of the defender to obtain a tractable solution to Problem V.1.

**Definition V.3** ([16])**.** *A function $f(\cdot)$ defined over $\mathcal{X} \in \Re^n$ satisfies the diminishing returns (DR) property if for all $a \leqslant b \in \mathcal{X}$, for all $i \in n$, for all $k \in \Re_+$ such that $(k\mathcal{X}_i + a)$ and $(k\mathcal{X}_i + b)$ are still in $\mathcal{X}$, it holds, $f(k\mathcal{X}_i + a) - f(a) \geqslant f(k\mathcal{X}_i + b) - f(b)$. $f(\cdot)$ is called a DR-submodular function.*

For a twice differentiable function Definition V.3 is equivalent to $\nabla_{ij}^2 f(x) \leqslant 0$, for all $i \neq j$ [13]. We introduce the following concepts which is used in Theorem V.4 to prove that Problem V.1 is equivalent to maximizing a DR-submodular function.

Notice that a strategy $p_{A_k} \in \mathbf{P}_{A_k}$ of the $k^{\text{th}}$ adversary induces a set of sample paths in the underlying state space. All the sample paths resulting from a given $p_{A_k}$ can be divided into one of the following two categories: i) a sample path, $\omega_k$, starts from the node $s_0$ and ends at the node $d_k$ in $\mathcal{G}$. Let $\Omega_k^{d_k}$ denote the set of such paths and ii) a sample path, $\omega_k$, starts from the node $s_0$ and ends at a node $v_j \in V_{\mathcal{G}} \setminus \{d_k\}$, where $p_{A_k}(v_j,\varnothing) = 1$ and $j \in \{1,\dots,N\}$. That is, $\mathcal{P}_{A_K}$ abandons its attack at a node $v_j$ in $\mathcal{G}$ before reaching its destination $d_k$. Let $\Omega_k^{\phi}$ denote the set of such paths.

**Theorem V.4.** *The defender's payoff function, $U_D(p_A,p_D)$, is DR-submodular in the defender's strategy, $p_D$.*

*Proof.* For a given $p_A \in \mathbf{P}_A$, let $p_R(\omega_k)$, $p_T(\omega_k)$, and $p_\phi(\omega_k)$ be the probability of $\mathcal{P}_{A_K}$ reaching $d_k$ without getting detected, the probability of getting detected, and the probability of getting trapped when $k^{\text{th}}$ adversary is following the sample path $\omega_k$, respectively. Define $p(\omega_k)$ to be the probability of $\mathcal{P}_{A_K}$ choosing sample path $\omega_k$ in $\mathcal{G}$. Now, $U_D(p_A, p_D)$ can be rewritten as follows.

$$\begin{aligned}U_D(p_A,p_D) = &\sum_{k\in\{1,\dots,K\}}\Big(\sum_{\omega_k\in\Omega_k^{d_k}} p(\omega_k)\Big(p_T(\omega_k)\alpha_k^D + p_R(\omega_k)\beta_k^D + \\ &p_\phi(\omega_k)\sigma_k^D\Big) + \sum_{\omega_k\in\Omega_k^{\phi}} p(\omega_k)\Big(p_T(\omega_k)\alpha_k^D + p_\phi(\omega_k)\sigma_k^D\Big)\Big) \\ = &\sum_{k\in\{1,\dots,K\}}\Big(\sum_{\omega_k\in\Omega_k^{d_k}} p(\omega_k)\Big(\alpha_k^D + [\beta_k^D-\alpha_k^D]p_R(\omega_k) + \\ &[\sigma_k^D-\alpha_k^D]p_\phi(\omega_k)\Big) + \sum_{\omega_k\in\Omega_k^{\phi}} p(\omega_k)\Big(\alpha_k^D + [\sigma_k^D-\alpha_k^D]p_\phi(\omega_k)\Big)\Big) \quad (4)\end{aligned}$$

Eq. (4) holds as $p_T(\omega_k) = 1 - p_R(\omega_k) - p_\phi(\omega_k)$. Notice that $p_R(\omega_k) = \prod_{s_i\in\omega_k}[1-p_D^{\mathbf{s}}(s_i,k)]$ and $p_\phi(\omega_k) = \sum_{s_{i'}\in\omega_k}\prod_{s_i\in\hat{\omega}_k}[1-p_D^{\mathbf{s}}(s_i,k)]p_{A_k}(s_{i'},\varnothing)$, where $\hat{\omega}_k$ is the subpath in $\omega_k$ from the node $s_0$ to the node $s_{i'}$ in $\mathcal{G}$.

Let $\nabla_i^2 U_D(p_D)$ denote the $i^{\text{th}}$ entry of the gradient of $U_D$ and $\nabla_{ij}^2 U_D(p_D)$ denote the $(i,j)^{\text{th}}$ entry of Hessian of $U_D(p_D)$, where $i$ and $j$ are indices of the vector $p_D$. Then,

$$\begin{aligned}\nabla_i U_D(p_D) \;=\; &\sum_{k\in\{1,\dots,K\}}\Big(\sum_{\omega_k\in\Omega_k^{d_k}} p(\omega_k)\Big([\beta_k^D-\alpha_k^D]\nabla_i p_R(\omega_k) \\ &+[\sigma_k^D-\alpha_k^D]\nabla_i p_\phi(\omega_k)\Big) + \sum_{\omega_k\in\Omega_k^{\phi}} p(\omega_k)\Big([\sigma_k^D-\alpha_k^D]\nabla_i p_\phi(\omega_k)\Big)\Big)\end{aligned}$$

$$\begin{aligned}\nabla_{ij}^2 U_D(p_D) \;=\; &\sum_{k\in\{1,\dots,K\}}\Big(\sum_{\omega_k\in\Omega_k^{d_k}} p(\omega_k)\Big([\beta_k^D-\alpha_k^D]\nabla_{ij}^2 p_R(\omega_k) \\ &+[\sigma_k^D-\alpha_k^D]\nabla_{ij}^2 p_\phi(\omega_k)\Big) + \sum_{\omega_k\in\Omega_k^{\phi}} p(\omega_k)\Big([\sigma_k^D-\alpha_k^D]\nabla_{ij} p_\phi(\omega_k)\Big)\Big)\end{aligned}$$

Here, $\nabla_i p_R(\omega_k) = -\prod_{\substack{s_l\in\omega_k\\ s_l\neq s_i}}[1-p_D^{\mathbf{s}}(s_l,k)]$ and for any $i\neq j$, $\nabla_{ij}^2 p_R(\omega_k) = \prod_{\substack{s_l\in\omega_k\\ s_l\notin\{s_i,s_j\}}}[1-p_D^{\mathbf{s}}(s_l,k)]$. Also, for any $i\neq j$,

$$\nabla_i p_\phi(\omega_k) = -\sum_{\substack{s_{l'}\in\omega_k\\ s_{l'}\geqslant s_i}}\prod_{\substack{s_l\in\hat{\omega}_k\\ s_l\neq s_i}}[1-p_D^{\mathbf{s}}(s_l,k)]p_{A_k}(s_{l'},\varnothing) \text{ and}$$

$$\nabla_{ij}^2 p_\phi(\omega_k) = \sum_{\substack{s_{l'}\in\omega_k\\ s_{l'}\geqslant\{s_i,s_j\}}}\prod_{\substack{s_l\in\hat{\omega}_k\\ s_l\notin\{s_i,s_j\}}}[1-p_D^{\mathbf{s}}(s_l,k)]p_{A_k}(s_{l'},\varnothing).$$

The notation $s_{l'}\geqslant s_i$ (and $s_{l'}\geqslant\{s_i,s_j\}$, resp.) implies that the node $s_{l'}$ in the sample path $\omega_k$ appears as or after the node $s_i$ ($s_r$, where the index $r=i$ if the node $s_i$ appears after the node $s_j$ in the sample path $\omega_k$ and vice versa, resp.).

Note that $\nabla_{ij}^2 p_R(\omega_k)\geqslant 0$ and $\nabla_{ij}^2 p_\phi(\omega_k)\geqslant 0$. Then from Eq. (5), $\nabla_{ij}^2 U_D(p_D)\leqslant 0$ for any $i\neq j$ as $[\beta_k^D-\alpha_k^D]\leqslant 0$ and $[\sigma_k^D-\alpha_k^D]\leq 0$. Thus, $U_D(p_D)$ is DR-submodular in $p_D$. $\square$

In order to propose an approximation algorithm that leverages the DR-submodularity of $U_D$, we first show some additional properties of $U_D$. Specifically, in Lemma V.5 we show that $U_D$ is point-wise monotone increasing in $p_D$ and in Theorem V.6 we show a Lipschitz condition.

**Lemma V.5.** *The defender's payoff function, $U_D(p_A, p_D)$, is monotone increasing in the defender's strategy, $p_D$.*

*Proof.* Consider the defender's payoff function given by Eq. (4). Notice that when $p_D$ is increasing both $p_R(\omega_k) = \prod_{s_i\in\omega_k}[1-p_D^{\mathbf{s}}(s_i,k)]$ and $p_\phi(\omega_k) = \sum_{s_{i'}\in\omega_k}\prod_{s_i\in\hat{\omega}_k}[1-p_D^{\mathbf{s}}(s_i,k)]p_{A_k}(s_{i'},\varnothing)$ terms decreases. Also $[\beta_k^D-\alpha_k^D]<0$ and $[\sigma_k^D-\alpha_k^D]<0$. Then for any $p_D,\hat{p}_D\in\mathbf{P}_D$ with $p_D<\hat{p}_D$, $U_D(p_A,p_D)<U_D(p_A,\hat{p}_D)$. $\square$

A univariate auxiliary function $g_{p_D,\hat{p}_D}(\zeta)$ is introduced below to prove the required Lipschitz condition.

**Theorem V.6.** *The univariate axillary function $g_{p_D,\hat{p}_D}(\zeta) = U_D(p_A, p_D+\zeta\hat{p}_D)$ with respect to $\zeta$ has L-Lipschitz continuous derivative in $[0,1]$, where $\zeta\in\mathbb{R}_+$ and $p_D,\hat{p}_D\in\mathbf{P}_D$.*

*Proof.* Second derivative of $g_{p_D,\hat{p}_D}(\zeta)$ with respect to $\zeta$ can be written as follows:

$$\begin{aligned}\nabla_\zeta^2 g_{p_D,\hat{p}_D}(\zeta) = &\sum_{k\in\{1,\dots,K\}}\Big(\sum_{\omega_k\in\Omega_k^{d_k}} p(\omega_k)\Big([\beta_k^D-\alpha_k^D]\nabla_\zeta^2 p_R(\zeta,\omega_k) + \\ &[\sigma_k^D-\alpha_k^D]\nabla_\zeta^2 p_\phi(\zeta,\omega_k)\Big) + \sum_{\omega_k\in\Omega_k^{\phi}} p(\omega_k)[\sigma_k^D-\alpha_k^D]\nabla_\zeta^2 p_\phi(\zeta,\omega_k)\Big),\end{aligned}$$

where $p_R(\zeta,\omega_k) = \prod_{s_i\in\omega_k}[1-p_D^{\mathbf{s}}(s_i,k)-\zeta\hat{p}_D^{\mathbf{s}}(s_i,k)]$ and $p_\phi(\zeta,\omega_k) = \sum_{s_{i'}\in\omega_k}\prod_{s_i\in\hat{\omega}_k}[1-p_D^{\mathbf{s}}(s_i,k)-\zeta\hat{p}_D^{\mathbf{s}}(s_i,k)]p_{A_k}(s_{i'},\varnothing)$. Notice that both $p_R(\zeta,\omega_k)$ and $p_\phi(\zeta,\omega_k)$ are polynomials of $\zeta$. Hence they can be expressed as $p_R(\zeta,\omega_k) = a_0\zeta^n + a_1\zeta^{n-1}+\ldots+a_{n-1}\zeta+1$ and $p_\phi(\zeta,\omega_k) = b_0\zeta^m+b_1\zeta^{m-1}+\ldots+b_{m-1}\zeta+1$. Furthermore, $0\leq a_i,b_j\leq 1$ for $i\in\{0,\ldots,n-1\}$ and $j\in\{0,\ldots,m-1\}$, where $n$ represent the number of distinct nodes (excluding the node corresponding to $d_k$) traversed in the sample paths related to $\omega_k\in\Omega_k^{d_k}$ and $m$ is the number of distinct nodes traversed in $\omega_k\in\Omega_k^{\phi}$.

From the polynomial forms of $p_R(\zeta,\omega_k)$ and $p_\phi(\zeta,\omega_k)$, we can write their second derivatives with respect to $\zeta$ as $\nabla_\zeta^2 p_R(\zeta,\omega_k) = a_0 n(n-1)\zeta^{n-2} + a_1(n-1)(n-2)\zeta^{n-3}+\ldots+6a_{n-3}\zeta+2a_{n-2}$ and $\nabla_\zeta^2 p_\phi(\zeta,\omega_k) = b_0 m(m-1)\zeta^{m-2} + b_1(m-1)(m-2)\zeta^{m-3}+\ldots+6b_{m-3}\zeta+2b_{m-2}$. The terms $a_i$ for $i\in\{0,\ldots,n-2\}$ and $b_j$ for $j\in\{0,\ldots,m-2\}$ are products of probability terms. Hence, $a_i$ and $b_j$ are upper bounded by 1. $\max(n) = N-1$ when the sample path from $s_0$ to $d_k$ has to traverse through all the $N$ distinct nodes in the $\mathcal{G}$ without getting trapped for a corresponding $k^{\text{th}}$ analysis at $N-1$ nodes (note that defender is not allowed to trap at the $N^{\text{th}}$ node which is related to the $d_k$ in this case). Similarly, $\max(m) = N-1$ when the $k^{\text{th}}$ adversary traverse through $N-1$ distinct nodes in $\mathcal{G}$ on a path where $N^{\text{th}}$ node is $d_k$ (i.e $\omega_k\in\Omega_k^{d_k}$) and abandon the attack at the node $N-1$ with a non zero probability or $k^{\text{th}}$ adversary traverse through $N-1$ distinct nodes in $\mathcal{G}$ that does not contain the node related to $d_k$ and abandon the attack at node $N-1$ with probability one (i.e. $\omega_k \in \Omega_k^{\phi}$).

For $\zeta \in [0,1]$, we can bound the maximum value of $\nabla^2 p_R(\zeta,\omega_k)$ and $\nabla^2 p_\phi(\zeta,\omega_k)$ by $(N-2)(N-1)N/3$, where $N \geq 3$. Let $\omega'_k$ be a path in $\Omega_k^{d_k}$ that yield the highest probability $p(\omega'_k)$ and similarly let $\omega''_k$ be a path in $\Omega_k^{\phi}$ that is gives the highest probability $p(\omega''_k)$. Let the total number of paths in the set $\Omega_k^{d_k}$ and $\Omega_k^{\phi}$ denoted by $|\Omega_k^{d_k}|$ and $|\Omega_k^{\phi}|$, respectively. Define $\hat{L}_k = \max\{|p(\omega'_k)|\Omega_k^{d_k}|[\beta_k^D - \alpha_k^D]|, |p(\omega''_k)|\Omega_k^{d_k}|[\sigma_k^D - \alpha_k^D]|, |p(\omega''_k)|\Omega_k^{\phi}|[\sigma_k^D - \alpha_k^D]|\}$ and $\hat{L} = \max_k\{\hat{L}_k\}$ for $k = 1,\dots,K$. Hence we obtain the following upper bound on the $|\nabla_\zeta^2 g_{p_D,\hat{p}_D}(\zeta)|$:

$$|\nabla_\zeta^2 g_{p_D,\hat{p}_D}(\zeta)| \leq \hat{L}_k \frac{(N-2)(N-1)N}{3} \leq \hat{L}\frac{(N-2)(N-1)N}{3}$$

Since the second derivative of $g_{p_D,\hat{p}_D}(\zeta)$ with respect to $\zeta$ is bounded in the case where $\zeta \in [0,1]$, first derivative of $g_{p_D,\hat{p}_D}(\zeta)$ with respect to $\zeta$ has $L$-Lipschitz continuous derivative in $[0,1]$. Furthermore, $L = \hat{L}\frac{(N-2)(N-1)N}{3}$. □

The Lipschitz constant $L$ derived in Theorem V.6 is used later in Algorithm V.1 and in Proposition V.9 to claim an approximation guarantee on Problem V.1.

**Definition V.7.** *A set $\mathcal{P}$ is said to be a down-closed convex set if $x \in \mathcal{P}$ and $0 \leqslant y \leqslant x$ implies $y \in \mathcal{P}$.*

**Lemma V.8.** *Let $\mathbf{P}_D$ be the strategy space of the defender and $\mathcal{P} \subset \mathbf{P}_D$ be the feasible strategy space of the defender. That is, $\mathcal{P} := \{p_D \in \mathbf{P}_D : \sum_{i=1}^{W}\sum_{k=1}^{K} p_D^{\mathbf{s}}(s_i,k)\mathcal{C}_k(s_i) \leqslant M$, for all $\mathbf{s} \in \mathbf{S}\}$. Then, $\mathcal{P}$ is a down-closed convex polytope in the positive orthant.*

*Proof.* The strategy space of the defender is a positive orthant as probabilities are bounded between 0 and 1. Thus $\mathcal{P} \subseteq [0,1]^{|p_D|}$ satisfies $\mathcal{P} := \{p_D \in \mathbf{P}_D | 0 \leqslant [0,1]^{|p_D|}, Ap_D \leqslant M\mathbb{1}\}$, where $\mathbb{1}$ is a vector of all ones of size $|\mathbf{S}|$. Here, $|\mathbf{S}|$ is the number of states in $\mathbf{S}$, $|p_D|$ is the size of $p_D$, $M$ is the available memory, and $A$ is the $(|\mathbf{S}| \times |\mathbf{S}|)$ matrix that captures the memory constraint. Note that the feasible space is a subset of the polytope $[0,1]^{|p_D|}$ that is constrained by $Ap_D \leqslant M\mathbb{1}$. For any $\hat{p}_D$ satisfying $0 \leqslant \hat{p}_D \leqslant p_D$, $A\hat{p}_D \leqslant M\mathbb{1}$. This implies that $\mathcal{P}$ is a down-closed convex polytope in the positive orthant. □

In what follows, we present an algorithm to compute an approximate optimal strategy of the defender.

**Proposition V.9.** *Let $U_D^\star(p_A)$ be the maximum defender's payoff for given attackers' strategies $p_A = \{p_{A_k}\}_{k=1}^{K}$. Then, Algorithm V.1 which takes as input $p_A$ returns an approximate optimal defense strategy $\bar{p}_D$ for the defender such that $U_D(p_A,\bar{p}_D) \leqslant (1 - 1/e + \varepsilon)U_D^\star(p_A)$. Further, Algorithm V.1 takes $O(1/\varepsilon)$ number of iterations and the number of operations in each iteration is linear in the action space of the defender.*

*Proof.* Theorem V.4 and Lemma V.5 show that the defender's payoff function, $U_D$, is DR-submodular and monotonically increasing in $p_D$, respectively. Thus Problem V.1 is equivalent to maximizing an increasing DR-submodular function. Using

**Algorithm V.1** Best response computation of the defender

```
Input: Attacker's strategies $\mathcal{P}_A$, $\mathcal{P}$, stepsize $\gamma \in (0,1]$
Output: Best response of the defender $\bar{p}_D$
1: Initialize $p_D^0 \leftarrow 0, t \leftarrow 0, r \leftarrow 0$
2: while $t < 1$ do
3:     Find $\hat{p}_D^r : \langle \hat{p}_D^r, \nabla U_D(p_D^r)\rangle \geqslant \eta \max_{\hat{p}_D \in \mathcal{P}} \langle \hat{p}_D, \nabla U_D(p_D^r)\rangle$
       $-\frac{1}{2}\delta L$, where $L > 0$ is the Lipschitz constant from
       Theorem V.6, $\eta \in (0,1]$ is the multiplicative error level,
       $\delta \in [0,\bar{\delta}]$ is the additive error level
4:     Find stepsize $\gamma_r \in (0,1]$, e.g., $\gamma_r \leftarrow \gamma$; and set $\gamma_r \leftarrow$
       $\min\{\gamma_r, 1-t\}$
5:     $p_D^{r+1} \leftarrow p_D^r + \gamma_r \hat{p}_D^r$, $t \leftarrow t + \gamma_r$, $r \leftarrow r+1$
6: end while
7: return $\bar{p}_D \leftarrow p_D^r$
```

the Lipschitz constant of the gradient of $U_D$ derived in Theorem V.6, the proof follows from Theorem 1 and Corollary 1 in [16]. □

### B. Best Responses of the Adversaries

The best response of the kth adversarial player $\mathcal{P}_{A_k}$ to a given defender strategy, $p_D$, and other $K-1$ adversarial players' strategies, $p_{A\setminus k} := \{p_{A_k}\}_{k=1}^{k=K} \setminus p_{A_k} = p_{A_1},\dots,p_{A_{k-1}},p_{A_{k+1}},\dots,p_{A_K}$, is given by the following optimization problem.

**Problem V.10.** The adversary's problem is as follows: *for any $k \in \{1,\dots,K\}$*

$$\max_{p_{A_k} \in \mathbf{P}_{A_k}} \left(\bar{p}_T(k)\alpha_k^A + \bar{p}_R(k)\beta_k^A + \bar{p}_\phi(k)\sigma_k^A\right)$$

In this section we derive a pure strategy best response for the kth adversary, where $k \in \{1,\dots,K\}$.

**Definition V.11** (Pure strategies of $\mathcal{P}_{A_k}$)**.** *The kth adversary's strategy can be interpreted as selecting a sample path, $\omega_k$, in $\mathcal{G}$ from node $s_0$ to $d_k$. Then for a given pair of nodes $v_i$ and $v_j$ in $\mathcal{G}$, where $i,j \in \{1,\dots,N\}$, a pure strategy of $\mathcal{P}_{A_k}$ can be defined as follows:*

$$p_{A_k}(v_i,v_j) = \begin{cases} 1, & \text{if } (v_i,v_j) \in \omega_k \\ 0, & \text{otherwise.} \end{cases}$$

The following lemma characterizes the best response of $\mathcal{P}_{A_k}$ under pure strategies.

**Lemma V.12.** *Let $\Omega_k^{d_k}$ denote the set of paths that start from the node $s_0$ and end at the node $d_k$ in $\mathcal{G}$. Then for any $\omega_k \in \Omega_k^{d_k}$, define $p_R(\omega_k)$ to be the probability of $\mathcal{P}_{A_k}$ reaching $d_k$ without being detected by $\mathcal{P}_D$ when $\mathcal{P}_{A_k}$ is following the path $\omega_k$. Furthermore, let $\omega^* = \arg\max\{p_R(\omega_k) : \omega_k \in \Omega_k^{d_k}\}$. Then, $\omega^* \in BR(p_D, p_{A\setminus k})$.*

*Proof.* Note that $p_{A_k}(v_i,\phi) = 0$, for all $i \in \{1,\dots,N\}$, when $\mathcal{P}_{A_k}$ follows a pure strategy. Let $p(\omega_k)$ be the probability of $\mathcal{P}_{A_K}$ choosing a path $\omega_k \in \Omega_k^{d_k}$. Then the payoff of $\mathcal{P}_{A_k}$ under a pure strategy is as follows:

$$U_{A_k}(p_A,p_D) \quad = \quad \bar{p}_T(k)\alpha_k^A + \bar{p}_R(k)\beta_k^A$$

$$= \sum_{\omega_k \in \Omega_k^{d_k}} p(\omega_k)\Big([1-p_R(\omega_k)]\alpha_k^A + p_R(\omega_k)\beta_k^A\Big)$$
$$= \sum_{\omega_k \in \Omega_k^{d_k}} p(\omega_k)\Big(\alpha_k^A + [\beta_k^A - \alpha_k^A]p_R(\omega_k)\Big)$$

For a given $p_D$ and $p_{A\setminus k}$, any path $\omega_k \in \Omega_k^{d_k}$ that maximize $U_{A_k}(p_A, p_D)$ will yield the same payoff to $\mathcal{P}_{A_k}$. Hence we can rewrite the best response of $\mathcal{P}_{A_k}$ (under pure strategies) as

$$\mathrm{BR}(p_D, p_{A\setminus k}) \in \arg\max_{\omega_k \in \Omega_k^{d_k}} \Big(\alpha_k^A + [\beta_k^A - \alpha_k^A]p_R(\omega_k)\Big).$$

Since $\alpha_k^A$ is a constant value and the term $[\beta_k^A - \alpha_k^A]$ is a positive scalar, $\mathrm{BR}(p_D, p_{A\setminus k}) \in \arg\max\{p_R(\omega_k) : \omega_k \in \Omega_k^{d_k}\}$. □

Next we define the set of states reachable to a state $\mathbf{s} \in \mathbf{S}$ in the following definition.

**Definition V.13.** *A state* $\mathbf{s} \in \mathbf{S}$ *is said to be* one-step reachable *from a state* $\bar{\mathbf{s}} \in \mathbf{S}$ *if each of the position* $s_j$ *in the state* $\mathbf{s}$ *is one-step reachable from each of the position* $\bar{s}_j$ *in* $\bar{\mathbf{s}}$*, for all* $j \in \{1, \dots, W\}$*. Here, we say a position* $s_j$ *is one-step reachable from a position* $\bar{s}_j$*, if* $s_j \in \{\phi, \tau\}$ *or if* $(v_i, v_r) \in E_{\mathcal{G}}$ *where* $\bar{s}_j = v_i$ *and* $s_j = v_r$*. Then,* $\bar{\mathbf{S}}(\mathbf{s}) \subseteq \mathbf{S}$ *is the set of states from which state* $\mathbf{s}$ *is one-step reachable.*

Let $p_B$ be the distribution of the benign information flows in the system. More precisely, it provides transition probabilities of a benign flow between two nodes $v_i, v_j \in V_{\mathcal{G}}$. In the following, we define the probability of type-$k$ trapping at node $v_i$ in $\mathcal{G}$, $p_D(v_i, k)$, for a given $p_D$, $p_{A\setminus k}$, and $p_B$.

**Definition V.14.** *Without loss of generality, assume each* $k^{\text{th}}$ *position in the state* $\mathbf{s} \in \mathbf{S}$ *related to the* $\mathcal{P}_{A_k}$*. Let* $\hat{\mathbf{S}}(v_i) \subseteq \mathbf{S}$ *be the set of states such that for all* $\hat{\mathbf{s}} \in \hat{\mathbf{S}}(v_i)$*,* $s_k = v_i$ *for* $i \in \{1, \dots, N\}$*. Then the probability of type-k trapping at node* $v_i$ *in* $\mathcal{G}$*,* $p_D(v_i, k)$*, can be defined as follows:*

$$p_D(v_i,k) = \sum_{\hat{\mathbf{s}} \in \hat{\mathbf{S}}(v_i)} \Big(\sum_{\bar{\mathbf{s}} \in \bar{\mathbf{S}}(\hat{\mathbf{s}})} \prod_{k'} p_{A_{k'}}(\bar{s}_{k'}, \hat{s}_{k'}) \prod_{k''} p_B(\bar{s}_{k''}, \hat{s}_{k''})\Big) p_D^{\hat{\mathbf{s}}}(\hat{s}_k, k),$$

*where* $k' \in \{1, \dots, K\} \setminus k$*,* $k'' \in \{K+1, \dots, W\}$*,*

$$p_{A_{k'}}(\bar{s}_{k'}, \hat{s}_{k'}) = \begin{cases} 1, \text{ if } \bar{s}_{k'} = \hat{s}_{k'} \text{ and } \bar{s}_{k'} \in \{\phi, \tau, d_{k'}\} \\ p_D^{\bar{\mathbf{s}}}(\bar{s}_{k'}, k'), \text{ if } \bar{s}_{k'} \in V_{\mathcal{G}} \text{ and } \hat{s}_{k'} = \tau \\ p_{A_{k'}}(\bar{s}_{k'}, \hat{s}_{k'})[1 - p_D^{\bar{\mathbf{s}}}(\bar{s}_{k'}, k')], \text{ otherwise} \end{cases}$$

*and*

$$p_B(\bar{s}_{k''}, \hat{s}_{k''}) = \begin{cases} 1, \text{ if } \bar{s}_{k''} = \hat{s}_{k''} \text{ and } \bar{s}_{k''} \in \{\phi, d_{\bar{k}}\} \\ p_B(\bar{s}_{k''}, \hat{s}_{k''}), \text{ otherwise} \end{cases}$$

*for all* $\bar{k} \in \{1, \dots, K\}$*.*

The following theorem presents a method to calculate a best response of $\mathcal{P}_{A_k}$ for a given $p_D$, $p_{A\setminus k}$, and $p_B$.

**Theorem V.15.** *The best response of adversary* $\mathcal{P}_{A_k}$*,* $BR(p_D, p_{A\setminus k})$*, under pure strategies, is a path* $\omega^*$ *returned by a shortest path algorithm on the IFG,* $\mathcal{G}$*, with edge weight of each incoming edge to a node* $v_i$ *given by* $-\log([1 - p_D(v_i, k)])$*, for* $i \in \{1, \dots, N\}$*.*

*Proof.* Notice that using $p_D(v_i,k)$ we can write $p_R(\omega_k) = \prod_{v_i \in \omega_k} [1 - p_D(v_i,k)]$. From Lemma V.12, we can derive the following expression[1] for the payoff of $\mathcal{P}_{A_k}$ under best response strategy,

$$\begin{aligned} \max \prod_{v_i \in \omega_k} [1 - p_D(v_i,k)] &= \max \prod_{v_i \in \omega_k} \log([1 - p_D(v_i,k)]) \\ &= \min \prod_{v_i \in \omega_k} -\log([1 - p_D(v_i,k)]) \end{aligned}$$

This implies that solving Problem V.10 is equivalent to solving a shortest path algorithm on $\mathcal{G}$, with $-\log([1 - p_D(v_i,k)])$ as the edge weight of each incoming edge to node $v_i$. Notice that edge weights are positive values since $0 < [1 - p_D(v_i,k)] \leqslant 1$ Furthermore, Definition V.14 can be used to calculate $p_D(v_i,k)$ for given $p_D$, $p_{A\setminus k}$ and $p_B$. Hence a Dijkstra's shortest path algorithm [17] will compute an optimal attacker strategy. □

## VI. Numerical Study

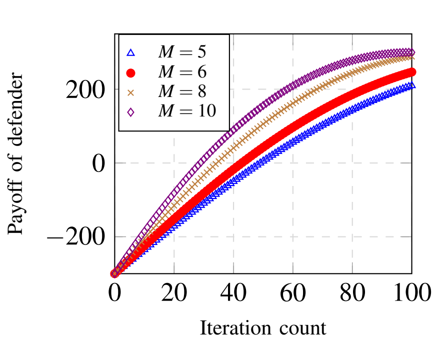
We validate our theoretical results using real-world attack data obtained using RAIN [1] for a three day nation state attack. We implement our model and run Algorithm V.1 on day one attack data of the nation state. A brief description of the dataset we used and the steps involved in the construction of the IFG for that attack are given below.

The goal of the adversaries' campaign was to steal sensitive proprietary and personal information from the targeted company. We considered two attackers operating in the system with distinct goals and entry points. Both attackers established their initial foothold in the system through network (e.g., spear-phishing attack, watering hole attack). Once the system is compromised, attacker 1 leveraged common system utilities to perform internal reconnaissance. The goal of this attacker was to fingerprint the compromised system to detect running processes and network information. On the other hand, attacker 2 wrote a malicious program to a disk that was eventually executed to establish a backdoor which is used to continuously exfiltrate the company's sensitive data.

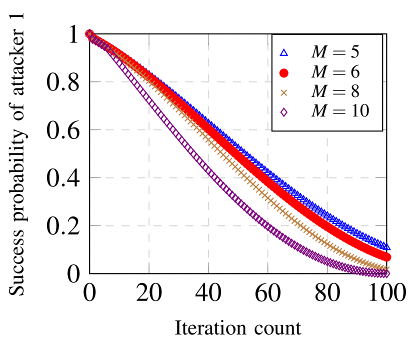
Initial conversion of the attack data into an IFG resulted in a coarse-grain graph with $\approx$ 132,000 nodes and $\approx$ 2 million edges. We pruned the coarse-grained graph by starting from the destinations of each attacker. Then we constructed the subgraph related to all the nodes in the coarse-grained graph that have at least one directed path to the destination of that attacker. We performed our analysis on the resulting refined IFG consisting of 30 nodes related to the attacks.

The parameters chosen are: $\alpha_1^D = 100$, $\alpha_2^D = 200$, $\beta_1^D = -100$, and $\beta_1^D = -200$. Thus attacker 2 is more capable as the impact of attack 2 is more compared to attack 1, which is captured by $\alpha_2^D > \alpha_1^D$. Figure 1(a) shows the variation of the payoff of the defender returned by Algorithm V.1 with respect to iteration count for four instances with different values of $M$. In order to analyze the impact of the defender

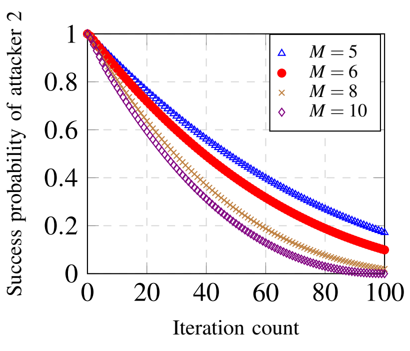
[1]The expression gives a scaled version of the exact payoff. In order to calculate the exact payoff under best responses of $\mathcal{P}_{A_k}$, one need to multiply the value of this expression by $[\beta_k^A - \alpha_k^A]$ and add a constant value $\alpha_k^A$.



(a)



(b)



(c)

Fig. 1: The parameters chosen are: $\alpha_1^D = 100$, $\alpha_2^D = 200$, $\beta_1^D = -100$, and $\beta_1^D = -200$. The memory values, $\mathcal{C}_k$, for the nodes in the IFG where chosen from a random distribution such that $\mathcal{C}_1(v_i) < \mathcal{C}_2(v_i)$, for all $i \in \{1,\ldots,N\}$. Figure 1(a) shows the payoff of the defender obtained by Algorithm V.1 v.s. iteration count for four instances with different values of $M$. Figures 1(b) and 1(c) show the probability of attacker one and two, respectively, reaching the target with increasing values of $M$.

on the different attackers, we compute attack success probabilities of each attacker with varying the amount of memory. Figures 1(b) and 1(c) show the probabilities of attacker one and two, respectively, reaching their targets while increasing the value of $M$. Notice from Figures 1(b) and 1(c) that for a fixed memory constraint, attacker 2 has higher probability of reaching the target compared to attacker 1 which is expected as attacker 2 is more capable.

## VII. Conclusion

In this paper, we studied the problem of detecting multiple attackers in a computer system. We presented an analytical model of a resource constrained DIFT that allocate scarce resources across multiple flows to simultaneously detect different attackers. We modeled the strategic interaction between $K$ adversarial information flows and the DIFT defense as a dynamic $(K+1)$-player game. Each stage of the game corresponds to the propagation of the attacks through the system, in which each attacker must determine the next operation and the defender must decide an efficient memory allocation. Given attackers' strategies, we proved that finding an optimal defense strategy is equivalent to maximizing an increasing DR-submodular function which enabled us to propose an approximation algorithm. Further, given a defense strategy and strategies of $(K-1)$ attackers, we showed that finding an optimal attacker strategy is equivalent to solving a shortest path problem, where the edge weights are derived from the strategies of the other players. Based on this mapping we proposed a polynomial-time algorithm for computing an optimal attacker strategy. We evaluated the performance of our algorithm on a real-world attack dataset obtained using RAIN [1]. In future, we plan to study the solution of the dynamic game by leveraging the best responses of the players to characterize optimal strategies of the players.

## References

[1] Y. Ji, S. Lee, E. Downing, W. Wang, M. Fazzini, T. Kim, A. Orso, and W. Lee, "RAIN: Refinable attack investigation with on-demand inter-process information flow tracking," *ACM SIGSAC Conference on Computer and Communications Security*, pp. 377–390, 2017.

[2] J. Newsome and D. Song, "Dynamic taint analysis: Automatic detection, analysis, and signature generation of exploit attacks on commodity software," *Network and Distributed Systems Security Symposium*, 2005.

[3] M. H. Manshaei, Q. Zhu, T. Alpcan, T. Başar, and J.-P. Hubaux, "Game theory meets network security and privacy," *ACM Computing Surveys*, vol. 45, no. 3, p. 25, 2013.

[4] T. Alpcan and T. Başar, "An intrusion detection game with limited observations," *International Symposium on Dynamic Games and Applications*, vol. 26, 2006.

[5] P. Hu, H. Li, H. Fu, D. Cansever, and P. Mohapatra, "Dynamic defense strategy against advanced persistent threat with insiders," *IEEE Conference on Computer Communications*, pp. 747–755, 2015.

[6] M. Van Dijk, A. Juels, A. Oprea, and R. L. Rivest, "FlipIt: The game of "stealthy takeover"," *Journal of Cryptology*, vol. 26, no. 4, pp. 655–713, 2013.

[7] P. Lee, A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, "A host takeover game model for competing malware," *IEEE Conference on Decision and Control*, pp. 4523–4530, 2015.

[8] M. Min, L. Xiao, C. Xie, M. Hajimirsadeghi, and N. B. Mandayam, "Defense against advanced persistent threats in dynamic cloud storage: A Colonel Blotto game approach," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4250–4261, 2018.

[9] S. Rass, S. König, and S. Schauer, "Defending against advanced persistent threats using game-theory," *PLoS one*, vol. 12, no. 1, pp. e0 168 675: 1–43, 2017.

[10] D. Sahabandu, B. Xiao, A. Clark, S. Lee, W. Lee, and R. Poovendran, "DIFT games: Dynamic information flow tracking games for advanced persistent threats," *IEEE Conference on Decision and Control*, pp. 1136–1143, 2018.

[11] S. Moothedath, D. Sahabandu, A. Clark, S. Lee, W. Lee, and R. Poovendran, "Multi-stage dynamic information flow tracking game," *Conference on Decision and Game Theory for Security*, vol. 11199, pp. 80–101, 2018.

[12] D. Sahabandu, S. Moothedath, J. Allen, A. Clark, L. Bushnell, W. Lee, and R. Poovendran, "A game theoretic approach for dynamic information flow tracking with conditional branching," *American Control Conference*, 2019.

[13] F. Bach, "Submodular functions: From discrete to continuous domains," *Mathematical Programming*, pp. 1–41, 2016.

[14] A. Krause and C. Guestrin, "Submodularity and its applications in optimized information gathering," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 4, pp. 32:1–20, 2011.

[15] H. Lin and J. Bilmes, "A class of submodular functions for document summarization," *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 510–520, 2011.

[16] A. A. Bian, B. Mirzasoleiman, J. M. Buhmann, and A. Krause, "Guaranteed non-convex optimization: Submodular maximization over continuous domains," *International Conference on Artificial Intelligence and Statistics*, pp. 111–120, 2017.

[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. MIT press: Cambridge, 2001.