Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

# Computing Optimal Monitoring Strategy for Detecting Terrorist Plots

**Zhen Wang**
School of Computer Engineering
Nanyang Technological University
Singapore 639798
wangzhen@ntu.edu.sg

**Yue Yin**
The Key Lab of Intelligent Information Processing, ICT, CAS
University of Chinese Academy of Sciences
Beijing 100190, China
melody1235813@gmail.com

**Bo An**
School of Computer Engineering
Nanyang Technological University
Singapore 639798
boan@ntu.edu.sg

## Abstract

In recent years, terrorist organizations (e.g., ISIS or al-Qaeda) are increasingly directing terrorists to launch coordinated attacks in their home countries. One example is the Paris shootings on January 7, 2015. By monitoring potential terrorists, security agencies are able to detect and stop terrorist plots at their planning stage. Although security agencies may have knowledge about potential terrorists (e.g., who they are, how they interact), they usually have limited resources and cannot monitor all terrorists. Moreover, a terrorist planner may strategically choose to arouse terrorists considering the security agency's monitoring strategy. This paper makes five key contributions toward the challenging problem of computing optimal monitoring strategies: 1) A new Stackelberg game model for terrorist plot detection; 2) A modified double oracle framework for computing the optimal strategy effectively; 3) Complexity results for both defender and attacker oracle problems; 4) Novel mixed-integer linear programming (MILP) formulations for *best* response problems of both players; and 5) Effective approximation algorithms for generating *suboptimal* responses for both players. Experimental evaluation shows that our approach can obtain a robust enough solution outperforming widely-used centrality based heuristics significantly and scale up to realistic-sized problems.

## 1 Introduction

In recent years, the rise of terrorist organizations (e.g., ISIS, al-Qaeda in the Arabian Peninsula (AQAP)) leads new challenges to the world's security. Such organizations are increasingly directing and encouraging their supporters in the West to carry out coordinated attacks in their home countries. To thwart such attacks, it is extremely important for domestic security agencies to detect the terrorist plot in the planning stage by monitoring potential terrorists[1]. Unfortunately, though security agencies may have knowledge about the suspects (e.g., who they are, whom they interact with), they do not have enough resources to monitor them all (Woo 2009). For example, the recent Paris shootings on January 7, 2015 is a successful plot directed by AQAP owing to the limited resources of France's domestic intelligence agency and most probably its poor monitoring strategy.



[1]We use 'terrorist' and 'potential terrorist' interchangeably.

It is challenging to develop efficient monitoring strategies since the terrorist planner can arouse any connected subgroup of a terrorist network, and different subgroups of terrorists can cause different levels of damage. A subgroup of terrorists who can cooperate well is more dangerous than a subgroup of terrorists who cannot (Enders and Jindapon 2010). Therefore, to decide whether to monitor a potential terrorist, the defender should consider the terrorist's capability as well as all subgroups he may be involved in. In addition, a strategic terrorist planner may take advantage of patterns in the security agency's monitoring strategies (Enders and Su 2007). Thus randomized monitoring strategies are needed. In this paper, we aim at designing the optimal allocation of security resources to monitor terrorists.

There has been lots of research on game-theoretic security resource allocation (Tambe 2011; Yin, An, and Jain 2014) and many systems based on attacker-defender security game models have been successfully deployed (Shieh et al. 2012). However, an attacker chooses a target to attack in standard security games (Blum, Haghtalab, and Procaccia 2014; Gan, An, and Vorobeychik 2015; Yin et al. 2015) or a path to execute an attack in graph based security games (Vorobeychik et al. 2014). While, in our domain, considering potential terrorists as targets, an attacker can choose a connected subgroup of targets and the strategy space is significantly larger than the set of targets or paths. Thus, existing approaches cannot be directly applied to our domain. Law enforcement and intelligence agencies often use network centrality measures (e.g., degree, closeness and betweenness) to priorities the potential terrorists (I2 2010). Recently, Lindelauf et al. (2013) introduced a Shapley value-based centrality metric. However, all these existing centrality metrics did not consider defender's combinatorial pure strategy space and failed to generate optimal monitoring strategy as we will show later.

To fill the research gap, we build a novel Stackelberg game model (*TPD*), where the leader (defender) chooses vertices of a terrorist network to monitor while the follower (terrorist planner) arouses a connected subgraph of the network to launch a coordinated attack. The major challenge brought by this model is the exponential growth (in terms of the network size) of both players' strategy spaces.The double oracle framework is a standard approach for handling problems with such large strategy spaces. Unfortunately,

the state-of-the-art algorithm (Jain, Conitzer, and Tambe 2013) cannot be directly applied to our domain because of the difference in players' strategies and the more complicated payoff function in our problem. Indeed, both players' oracles (i.e., computing one's best response strategy given the other's strategy) are shown to be NP-hard. To overcome these computational barriers, we propose our solution algorithm *DO-TPD* which incorporates the following key components: 1) An efficient initialization procedure to efficiently provide an initial feasible solution; 2) Novel mixed-integer linear programming (MILP) formulations to compute the best response for both players; 3) A greedy algorithm with constant-factor approximation ratio to efficiently generate improving strategies for the defender; and 4) An iterated local search heuristic to effectively identify improving attacker strategies. We conduct extensive experiments showing that our algorithm can obtain a robust enough solution outperforming widely-used centrality based heuristics significantly and scale up to realistic-sized problems.

## 2 Motivating Domain

In the morning of 7 January 2015, two brothers, Chérif Kouachi and Saïd Kouachi, stormed into the Paris office of satirical magazine *Charlie Hebdo* and gunned down 12 people. A few hours later, the third gunman, Amedy Coulibaly was involved in and killed a policewoman in Montrouge and four hostages at a kosher supermarket in east Paris. All the three gunmen were dead after two-day massive manhunt.

This is a well-coordinated plot, directed by AQAP (CNN 2015c). The three gunmen were known to French's domestic intelligence agency (DGSI) several years ago. They were connected to each other, and to a large France terrorist cell known as the Buttes-Chaumont network, which has connections to both ISIS and AQAP. At least one of the brothers travelled to Yemen in 2011, trained with AQAP and met local al-Qaeda handlers. The leadership of AQAP claimed their direction of this event. Amedy Coulibaly coordinated his attack with the brothers to increase the impact.

DGSI ascribed this successful plot to their limited resources. Some 1,400 French nationals have gone to fight in Iraq and Syria in the last few years, in which more than 100 have returned. It required more than 10 security personnel, working in shifts, to keep full monitoring (e.g., listen to phone calls and watch suspicious contacts) on a single suspect, while DGSI only has the manpower to monitor one fifth of all suspects in home (BBC 2015a). France monitored the brothers for over two years, but the monitoring terminated at the end of 2013 and June 2014, respectively. Because they seemed to be inactive targets that were quiet for a long time, the intelligence agency shifted limited resources to other high-priority targets (CNN 2015a). However, because Chérif Kouachi was previously arrested based on intercepted phone conversations, the Kouachis were aware that they were being watched and avoided any activities that might draw the attention of the agency. When DGSI took eyes off them, they grabbed the opportunity. According to the Paris Public Prosecutor's office, Amedy Coulibaly and Chérif Kouachi had exchanged more than 500 phone calls on their wives' phones in 2014 (BBC 2015b); even Chérif Kouachi had texted Amedy Coulibaly one hour before the *Charlie Hebdo* attack (CNN 2015b).

The challenge for DSGI is how to allocate limited resources efficiently to make the best use of them. Normally, DGSI just allocated limited resources to the high-priority suspects based on the assessment of their dangerousness (CNN 2015a). However, in order to plan such coordinated attacks, the terrorists need to communicate with each other, thus it is better for the agency to **take the network structure into account**. Intuitively, since the three gunmen had to communicate with each other to plan this attack, the agency did not need to keep both the brothers under surveillance simultaneously to prevent such a plot. In addition, considering that the terrorists could observe the surveillance on them and take advantage of its patterns, the agency might do better if **randomized monitoring schedules** were used. This paper aims to compute the optimal randomized monitoring strategy for domestic intelligence agencies such as DGSI.

## 3 Terrorist Plot Detecting Game (*TPD*)

We define the problem of detecting terrorist plots as a leader-follower (defender-attacker) Stackelberg game. The security agencies (e.g., DGSI) are the leader (defender) who acts first and the terrorist planners (e.g., AQAP, ISIS) observe the leader's strategy and then respond to it. The communication network of the terrorists is represented by a graph $G(V, E)$ with each vertex $v \in V$ representing a terrorist (or a group behaving like a single terrorist), and an edge between two vertices representing that two terrorists can communicate/cooperate with each other. In addition, each vertex is associated with a capability value $\tau_v \geq 0$, representing the terrorist's financial means, weapons accessibility or bomb building skills. Although the size and structure of the terrorist network are growing or shrinking over time, the terrorist planner will not plan a plot using the just returned terrorists or newly constructed communication links. Thus we assume that the network structure is static. Considering that the defender may have uncertainty associated with terrorists' capabilities, we also conduct sensitivity analysis in Section 5.

Formally, denote $\mathcal{N}_G(v)$ (or simply $\mathcal{N}(v)$) as the set of neighbors of $v \in V$ in $G$, i.e., $u \in \mathcal{N}_G(v)$ iff $(v, u) \in E$. For a subset $V' \subseteq V$, we can similarly denote the subset's neighbors as $\mathcal{N}(V') = \{u \in V \setminus V' | (v, u) \in E, v \in V'\}$. The subgraph $(V', E \cap \binom{V'}{2})$ induced by $V'$ is denoted by $G[V']$.[2] If there exists a path in $E \cap \binom{V'}{2}$ between every pair of vertices in $V'$, the subgraph $G[V']$ is connected. We denote the collection of all the subsets $V' \subseteq V$ whose induced subgraphs of $G$ are connected as $\mathcal{C}(G)$ (or simply $\mathcal{C}$).

**Strategies.** A pure defender strategy $S = \langle S_v \rangle$ is an assignment of the $R$ monitoring resources to $R$ vertices, i.e., $\sum_{v \in V} S_v = R$, where $S_v \in \{0, 1\}$ and $S_v = 1$ indicates that the vertex $v$ is monitored. The defender's pure strategy space is denoted by $\mathcal{S}$. A mixed defender strategy is a probability distribution over pure strategies, i.e., $\mathbf{x} = \langle x_S \rangle$ with

[2] Since non-induced subgraphs are not considered, we will often omit the word *induced* in this paper.

$x_S$ representing the probability that $S$ is played.

The attacker can choose any subset of vertices to plan an attack. In order to fulfill his planning or coordination activity, he must ensure that information can be communicated among all selected terrorists. Thus the attacker will choose a subset $V' \in \mathcal{C}$, and we denote an attacker's pure strategy as a vector $A = \langle A_v \rangle$, where $A_v = 1$ if $v \in V'$, 0 otherwise. The attacker's pure strategy space is denoted by $\mathcal{A}$. A mixed attacker strategy is denoted by $\mathbf{y} = \langle y_A \rangle$ with $y_A$ representing the probability that $A$ is played. With a slight abuse of notation, we also use $S$ and $A$ to denote subsets of vertices, such that $v \in S$ (or $A$) if $S_v = 1$ (or $A_v = 1$).

**Utility.** As in most related works, we assume a zero-sum game. Given a defender allocation $S$ and an attacker subset $A$, if $S \cap A = \emptyset$, the terrorist plot is not detected by the security agency and the attacker succeeds; otherwise the attacker fails. If the attacker succeeds, he will gain a payoff $P(A) = P(G[A])$ and the defender will receive $-P(A)$, otherwise both players will gain 0.

The payoff of a connected subgraph $G[A]$ represents the potential damage of the terrorist attack by the subgraph, which generally depends on both the capabilities of vertices and network structure of $G[A]$ (Enders and Su 2007; Enders and Jindapon 2010; Lindelauf, Hamers, and Husslage 2013). Firstly, to successfully launch a terrorist attack, tasks such as financing and weapon acquiring/transport have to be conducted. Each individual terrorist $v$ has some capability $\tau_v$ of performing such tasks. A more skillful terrorist can perform such tasks better and cause more damage. Secondly, if the terrorists can coordinate their activities easily, they can cause greater damage. Therefore, we assume that each vertex $v \in A$ can provide an external effect $\delta \cdot \tau_v$ to each of its neighbor vertices, where $\delta$ measures the extent of this positive network externality. Thus, $P(A)$ is defined as:

$$P(A) = \sum\nolimits_{v \in A} (\tau_v + \delta \sum\nolimits_{u \in \mathcal{N}_{G[A]}(v)} \tau_u) \tag{1}$$

Notice that we assume homogenous positive network externality effect, while our approach can be easily extended to handle other payoff functions such as heterogenous externality effect (Lindelauf, Hamers, and Husslage 2013).

Given a defender's mixed strategy $\mathbf{x}$ and an attacker's pure strategy $A$, the expected attacker utility is:

$$U_a(\mathbf{x}, A) = P(A) \sum\nolimits_{S \in \mathcal{S}} (1 - z_{S,A}) x_S \tag{2}$$

Where $z_{S,A}$ indicates whether the defender strategy $S$ and attacker strategy $A$ overlap, i.e.,$z_{S,A} = 1$ if $S \cap A \neq \emptyset$, 0 otherwise.

Similarly, the attacker's expected utility $U_a(S, \mathbf{y})$ of playing mixed strategy $\mathbf{y}$ against $S$ is:

$$U_a(S, \mathbf{y}) = \sum\nolimits_{A \in \mathcal{A}} (1 - z_{S,A}) y_A P(A) \tag{3}$$

We can define $U_a(\mathbf{x}, \mathbf{y}) = \sum_S x_S U_a(S, \mathbf{y}) = \sum_A y_A U_a(\mathbf{x}, A)$. And we have $U_d = -U_a$ due to the zero-sum assumption.

**Equilibrium.** The Stackelberg equilibrium (SSE) is the same as the maxmin equilibrium given the zero-sum assumption (the defender maximizes her minimum utility, or equivalently, minimizes the maximum attacker utility). Thus, the optimal mixed strategy $\mathbf{x}$ of the defender can be computed by solving the following linear program (LP).

$$\begin{aligned}
\max \quad & U & & (4)\\
\text{s.t.} \quad & U \leq U_d(\mathbf{x}, A) \qquad \forall A \in \mathcal{A} & (5)\\
& \sum\nolimits_{S \in \mathcal{S}} x_S = 1 & (6)\\
& x_S \geq 0 \qquad \forall S \in \mathcal{S} & (7)
\end{aligned}$$

However, it is impractical to directly solve this LP since the defender strategy space $\mathcal{S}$ grows exponentially with the number of resources $R$, whereas the attacker strategy space $\mathcal{A}$ grows exponentially with the network size $|V|$.

# 4 Approach

The double oracle framework is a standard approach for solving zero-sum games with large strategy spaces (Jain, Conitzer, and Tambe 2013). It first computes the equilibrium strategy for a significantly smaller restricted game and then iteratively computes improving strategies for both players and finally converges to a global equilibrium. The key problem thus reduces to computing the players' improving strategies, which are called *defender oracle* and *attacker oracle*, respectively. Unfortunately, the state-of-the-art double oracle algorithm *SNARES* (Jain, Conitzer, and Tambe 2013) cannot compute players' oracles in *TPD*s due to different strategy settings and payoff functions. Specifically, *SNARES* solves games where the defender blocks edges (instead of vertices as in *TPD*s) in a graph, and the attacker chooses a path connecting a source and a target (instead of a connected subgraph as in *TPD*s) to launch an attack, bearing a much smaller strategy space than that in *TPD*s. Moreover, all the paths with the same target have the same payoff in SNARES's domain, while each subgraph has a unique associate payoff decided by the vertices and their network structure in *TPD*s. Therefore, to apply the double oracle framework to solving *TPD*s, improvements need to be made on top of the existing framework. We propose our algorithm *DO-TPD* (an Double Oracle algorithm for *TPD*s) with the following novel features: 1) An efficient initialization procedure; 2) Novel mixed-integer linear programming (MILP) formulations for both players' oracles; 3) A greedy algorithm with constant-factor approximation ratio to speed up the computation of defender oracle; and 4) An iterated local search heuristic to speed up the computing of attacker's improving strategies. Next, we start with an overview of *DO-TPD*, and then present its key components in detail.

## 4.1 DO-TPD Overview

*DO-TPD* is sketched in Algorithm 1. Line 1 first initializes *DO-TPD*. A small strategy space $\langle \mathcal{S}', \mathcal{A}' \rangle$ is generated by solving a linear program *LWA-LP* where attacker strategy is restricted to "lone-wolf" attack using a single vertex. Then *DO-TPD* solves a restricted version of *TPD*, i.e., Eqs.(4)-(7) with $\langle \mathcal{S}, \mathcal{A} \rangle$ replaced by $\langle \mathcal{S}', \mathcal{A}' \rangle$ (*CoreLP*, Line 3). The restricted *TPD* can be solved very efficiently as the strategy space is rather small. Obviously, the solution obtained, being a SSE of the restricted *TPD*, does not necessarily form SSE to the original *TPD*. Both players may want to use other

**Algorithm 1:** *DO-TPD* overview

1 Initialize $\mathcal{S}'$, $\mathcal{A}'$ using *LWA-LP*;
2 **repeat**
3 $(\mathbf{x}, \mathbf{y}) \leftarrow$ *CoreLP*$(\mathcal{S}', \mathcal{A}')$;
4 $\mathcal{S}^+ \leftarrow$ *betterO-D*$(\mathbf{x}, \mathbf{y})$;
5 **if** $\mathcal{S}^+ = \emptyset$ **then** $\mathcal{S}^+ \leftarrow \{$*bestO-D*$(\mathbf{x}, \mathbf{y})\}$ ;
6 $\mathcal{S}' \leftarrow \mathcal{S}' \cup \mathcal{S}^+$; /*Lines $4-6$: Defender Oracle*/
7 $\mathcal{A}^+ \leftarrow$ *betterO-A*$(\mathbf{x}, \mathbf{y})$;
8 **if** $\mathcal{A}^+ = \emptyset$ **then** $\mathcal{A}^+ \leftarrow \{$*bestO-A*$(\mathbf{x}, \mathbf{y})\}$;
9 $\mathcal{A}' \leftarrow \mathcal{A}' \cup \mathcal{A}^+$; /*Lines $7-9$: Attacker Oracle*/
10 **until** $\mathcal{S}^+ = \emptyset$ *and* $\mathcal{A}^+ = \emptyset$;
11 **return** $(\mathbf{x}, \mathbf{y})$.

strategies out of $\langle \mathcal{S}', \mathcal{A}' \rangle$ to improve their utilities. *DO-TPD* allows them to do so with the subsequent oracles (Lines 4–6 and Lines 7–9). Specifically, in Lines 4–6, *DO-TPD* first calls efficient *betterO-D* (better Oracle for Defender) to find a set of improving strategies for the defender. If *betterO-D* fails to return an answer, *DO-TPD* proceeds to *bestO-D* (best Oracle for Defender) which returns at most one improving strategy and runs slower, but guarantees that no improving strategy exists when it fails to find one. In Lines 7–9, *DO-TPD* searches for improving attacker strategies in the similar manner. The process repeats until no improving strategy can be found for both players (Line 10), when the solution obtained is provably optimal to the original *TPD* (McMahan, Gordon, and Blum 2003).

## 4.2 Initialization Procedure

To initialize *DO-TPD*, we first restrict the attacker's pure strategy to "lone-wolf" attack, i.e., the terrorist planner only inspires a single terrorist to launch a attack, thus $\mathcal{A}' = \{\{v\} | v \in V\}$. In this case, the graph structure can be totally ignored and the game degenerates to the simplest form of security game, where the defender's strategy can be compactly represented by marginal coverage rate (i.e., the probability of being protected) of each target (vertex). The following LP, called *"lone-wolf" attack LP* (*LWA-LP*), computes the optimal marginal coverage $\mathbf{c}$.

$$\max \quad U \tag{8}$$
$$\text{s.t.} \quad U \leq -\tau_v(1 - c_v) \qquad \forall v \in V \tag{9}$$
$$\sum\nolimits_{v \in V} c_v \leq R \tag{10}$$
$$c_v \in [0, 1] \qquad \forall v \in V \tag{11}$$

Furthermore, a mixed strategy $\mathbf{x}$ can be efficiently sampled to implement this marginal $\mathbf{c}$ with existing approaches such as the Comb Sampling algorithm (Tsai et al. 2010). The support set is then used as the initial pure strategy set for the defender (i.e., $\mathcal{S}' = \{S | x_S > 0\}$).

## 4.3 Defender Oracle (DO)

***bestO-D*** The goal of defender oracle is to find pure strategies that can improve the defender's utility (given that the attacker sticks to the current strategy). We call such strategies *improving strategies*. Given a restricted game $\langle \mathcal{S}', \mathcal{A}' \rangle$ and its solution $(\mathbf{x}, \mathbf{y})$, the defender's utility can be improved if any

**Algorithm 2:** *betterO-D* $(\mathbf{x}, \mathbf{y})$

1 $\mathcal{S}_{better} = \emptyset$
2 **for** $v \in V$ **do**
3 $S \leftarrow \{v\}$;
4 **while** $|S| < R$ **do**
5 $v^* \leftarrow \arg\max_v U_d(S \cup \{v\}, \mathbf{y})$;
6 $S \leftarrow S \cup \{v^*\}$;
7 **if** $U_d(S, \mathbf{y}) > U_d(\mathbf{x}, \mathbf{y})$ **then** $\mathcal{S}_{better} = \mathcal{S}_{better} \cup \{S\}$ ;
8 **return** $\mathcal{S}_{better}$.

strategy $S$ satisfying $U_d(S, \mathbf{y}) > U_d(\mathbf{x}, \mathbf{y})$ is added to the current pure strategy space $\mathcal{S}'$. The best defender oracle *bestO-D* thus finds an improving strategy by maximizing $U_d(S, \mathbf{y})$ over the entire pure strategy space, and is formulated as the following MILP.

$$\max \quad -\sum\nolimits_{A \in \mathcal{A}'} (1 - z_A) y_A P(A) \tag{12}$$
$$\text{s.t.} \quad z_A \leq \sum\nolimits_{v \in V} A_v S_v \qquad \forall A \in \mathcal{A}' \tag{13}$$
$$\sum\nolimits_{v \in V} S_v \leq R \tag{14}$$
$$S_v \in \{0, 1\}, z_A \in [0, 1] \tag{15}$$

Here, Eq.(14) enforces that the defender covers at most $R$ vertices; Eq.(13) ensures $z_A = 1$ if $A \cap S \neq \emptyset$ (i.e., some vertex in subgraph $A$ is monitored by $S$), and 0 otherwise.

Unfortunately, solving *bestO-D* turns out to be NP-hard (Theorem 1). Therefore, to speed up the process, we propose a faster oracle *betterO-D*. *betterO-D* trades off by computing *suboptimal* ("better") solutions for Eqs.(12)-(15), which in most cases are good enough to meet the criterion $U_d(S, \mathbf{y}) > U_d(\mathbf{x}, \mathbf{y})$. In this fashion, *bestO-D* only needs to be called when *betterO-D* fails to find an answer.

**Theorem 1.** *The bestO-D problem is NP-hard.*[3]

***betterO-D*** As presented in Algorithm 2, *betterO-D* repeatedly starts from each single vertex $v \in V$ and, in a greedy manner, iteratively selects vertices that brings the maximum marginal utility (Line 5). Notably, *betterO-D* can generate multiple strategies in a single run, which significantly reduces the number of overall iterations in practice. Besides, a $(1 - \frac{1}{e})$-approximation of *betterO-D* is shown in Theorem 2. We can even totally rely on *betterO-D* and exclude *bestO-D* from the loop (i.e., remove Line 5 from Algorithm 1), which offers an approximate solution to the original *TPD*, while a $(1 - \frac{1}{e})$-approximation ratio still holds (Theorem 3).

**Theorem 2.** *Let* $U' = \max_{S \in \mathcal{S}_{better}} U_d(S, \mathbf{y})$*, where* $\mathcal{S}_{better}$ *is the solution returned by betterO-D. Let* $U^* = \max_{S \in \mathcal{S}} U_d(S, \mathbf{y})$*. Then* $\frac{U' - U_d(\emptyset, \mathbf{y})}{U^* - U_d(\emptyset, \mathbf{y})} \geq 1 - \frac{1}{e}$.[4]

**Theorem 3.** *Let* $(\mathbf{x}^*, \mathbf{y}^*)$ *be the optimal solution of a* TPD*, and let* $(\mathbf{x}', \mathbf{y}')$ *be the solution computed by Algorithm 1 with Line 5 skipped. Then* $\frac{U_d(\mathbf{x}', \mathbf{y}') - U_d(\emptyset, \mathbf{y}')}{U_d(\mathbf{x}^*, \mathbf{y}^*) - U_d(\emptyset, \mathbf{y}')} \geq 1 - \frac{1}{e}$.

[3] All the proofs in this paper are in the online appendix: http://www.ntu.edu.sg/home/boan/papers/AAAI16_Monitor_Appendix.pdf

[4] In case $\mathcal{S}_{better} = \emptyset$, we simply relax the criterion in Line 7 of Algorithm 2, and accept all solutions.

### 4.4 Attacker Oracle (AO)

***bestO-A*** The attacker oracle works in the similar way. The best oracle *bestO-A* maximizes $U_a(\mathbf{x}, A)$ for an improving strategy $A$ such that $U_a(\mathbf{x}, A) > U_a(\mathbf{x}, \mathbf{y})$. We formulate *bestO-A* as the following MILP.

$$\max \sum_{S\in\mathcal{S}'} x_S \cdot U_S \tag{16}$$

$$\text{s.t. } U_S \le M\cdot(1-A_v) \qquad \forall S\in\mathcal{S}', v\in S \tag{17}$$

$$A_u+A_v-1 \le \gamma_{uv} \le \min(A_u, A_v) \quad \forall (u,v)\in E \tag{18}$$

$$U_S \le \sum_{v\in V} A_v\tau_v + \sum_{(u,v)\in E}\delta\gamma_{uv}(\tau_u+\tau_v) \quad \forall S\in\mathcal{S}' \tag{19}$$

$$\omega_v \le \min(A_v, 1-A_u),\ \sum_{v\in V}\omega_v = 1 \quad \forall u,v\in V, u<v \tag{20}$$

$$\sum_{(u,q)\in E} h^+_{uqv} - \sum_{(u,q)\in E} h^-_{uqv} \ge \omega_u - (1-A_v) \quad \forall (u,v)\in \tbinom{V}{2} \tag{21}$$

$$h^+_{uqv} \le \gamma_{uv},\ h^-_{uqv} \le \gamma_{uv} \qquad \forall (u,q)\in E, v\in V \tag{22}$$

$$A_v\in\{0,1\},\ \gamma_{uv}\in[0,1],\ \omega_v\ge 0,\ U_S\ge 0 \tag{23}$$

Here, $U_S$ captures the attacker's utility against defender's pure strategy $S$. This is guaranteed along with Eqs.(17)-(19), where $\gamma_{uv}$ is a binary indicator such that $\gamma_{uv}=1$ iff both endpoints of edge $(u,v)\in E$ are selected by the attacker. In Eq.(20), $\omega_v = 1$ only when $v$ is the smallest indexed vertex[5] selected by the attacker. Furthermore, Eqs.(21)-(22) ensures connectivity of the selected subgraph by enforcing a unit flow from the smallest indexed vertex, say $v_0$, to any selected vertices via edges in the selected subgraph, where $h^+_{uvq}$ represents the amount of flow from $v_0$ to $v$ that passes through edge $(u,q)\in E$. The above MILP thus maximizes $U_a(\mathbf{x}, A)$ over the entire attacker strategy space.

**Theorem 4.** *The bestO-A problem is NP-hard, even when network externality is zero, i.e., $\delta = 0$.*

***betterO-A*** As presented in Algorithm 3, *betterO-A* generates an attacker pure strategy (i.e., a connected subgraph) containing each starting vertex $v\in V$. The core of each iteration (Lines 3–8) is designed based on the *basic Variable Neighborhood Search* framework (Hansen and Mladenović 2001). It utilizes two key components *LocalSearch* and $k$-distance neighbourhood $\mathcal{F}^{(k)}_v$.

Specifically, *LocalSearch* $(v, A, \mathbf{x})$ (Algorithm 4) starts from a given strategy $A$, and it consecutively tries to add a best vertex or remove a worst vertex in the hope of improving utility $U_a$, and stops when both tries fail. *LocalSearch* never removes the root vertex $v$ designated in the input, and never breaks connectivity of the subgraph, so that $\mathcal{P}(A)\cup\{v\}$ is always excluded from removing the candidates, where $\mathcal{P}(A)=\{v\in A | A\setminus\{v\}\notin\mathcal{C}\}$ is the set of cut vertices of the subgraph induced by $A$. While identifying $\mathcal{P}(A)$ is computationally more expensive, *LocalSearch* only tries to remove a vertex when no vertex can be added to improve the solution. $\mathcal{F}^{(k)}_v(A)$ represents the set of attacker pure strategies containing vertex $v$ and within Hamming distance $k$ of $A$, i.e., $\mathcal{F}^{(k)}_v(A) = \{A'\in\mathcal{C} | v\in A', \sum_{u\in V} | A_u - A'_u | \le k\}$.

[5]We can assign an index to each node, without introducing a new symbol, we simply use the node $v$ to present its index.

**Algorithm 3:** *betterO-A* $(\mathbf{x}, \mathbf{y})$

1 $\mathcal{A}_{better}\leftarrow\emptyset$;
2 **for** $v\in V$ **do**
3  $A\leftarrow$ *LocalSearch* $(v,\{v\},\mathbf{x})$, $k\leftarrow 1$;
4  **while** *no termination condition is met* **do**
5  $A'\leftarrow$ randomly pick one from $\mathcal{F}^{(k)}_v(A)$;
6  $A'\leftarrow$ *LocalSearch* $(v, A', \mathbf{x})$;
7  **if** $U_a(\mathbf{x}, A') > U_a(\mathbf{x}, A)$ **then** $A\leftarrow A'$, $k\leftarrow 1$;
8  **else** $k\leftarrow\min(k+1, k_{max})$;
9  **if** $U_a(\mathbf{x}, A) > U_a(\mathbf{x}, \mathbf{y})$ **then** $\mathcal{A}_{better}\leftarrow\mathcal{A}_{better}\cup\{A\}$;
10 **return** $\mathcal{A}_{better}$.

**Algorithm 4:** *LocalSearch* $(v, A, \mathbf{x})$

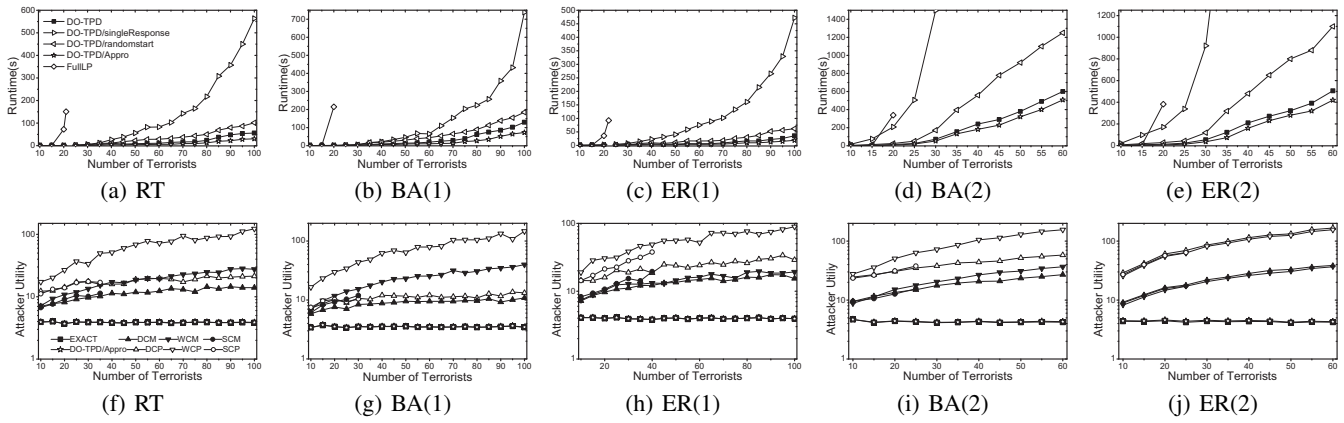
1 **Repeat**
2  $v^*\leftarrow\arg\max_{v\in\mathcal{N}(A)} U_a(\mathbf{x}, A\cup\{v\})$;
3  **if** $U_a(\mathbf{x}, A\cup\{v^*\}) > U_a(\mathbf{x}, A)$ **then** $A\leftarrow A\cup\{v^*\}$;
4  **else**
5  $v^*\leftarrow\arg\max_{v\in A\setminus(\mathcal{P}(A)\cup\{v\})} U_a(\mathbf{x}, A\setminus\{v\})$ **if** $U_a(\mathbf{x}, A\setminus\{v^*\}) > U_a(\mathbf{x}, A)$ **then** $A\leftarrow A\setminus\{v^*\}$;
6  **else return** $A$ ;

After initializing with *LocalSearch* (Line 3), *betterO-A* repeatedly generates a random new starting point $A'$ from the neighborhood of $A$ (Line 5), and applies *LocalSearch* again for a new local optimal solution (Line 6). Afterwards, the incumbent $A$ and neighborhood size $k$ are updated systematically in Lines 7–8, to avoid getting stuck in local optimum. The loop repeats until a termination condition is met: i) the incumbent $A$ is a better response for the attacker, i.e., $U_a(\mathbf{x}, A) > U_a(\mathbf{x}, \mathbf{y})$; ii) for $c_{max}$ consecutive iterations, the incumbent $A$ is not updated, or iii) the total number of iterations reaches a pre-defined constant $t_{max}$.

## 5 Experimental Evaluation

We evaluate the performance of our approach through extensive experiments. All LPs and MILPs are solved with CPLEX (version 12.6). All computations are performed on a machine with a 3.20GHz quad core CPU and 16.00GB memory. The parameters in *betterO-A* (Algorithm 3), $t_{max}$, $c_{max}$ and $k_{max}$ are set to $|V|$, $0.2*|V|$ and 3, respectively. The number of resources $R$ is set to $\lfloor\frac{|V|}{5}\rfloor$ unless otherwise specified. All the results are averaged over 50 samples.

We conduct experiments on three types of graph structures which are widely used to model terrorist organisations: (i) Random trees (RT), where every new vertex is attached to a randomly picked incumbent; (ii) Erdős-Rényi random graphs (ER($V$, $M$)), where exactly $M$ edges are randomly constructed between all the possible pairs of vertices (Erdős and Rényi 1959); (iii) Barabási-Albert scale-free networks (BA($k$)), where each new vertex is connected to $k$ incumbents using a preferential attachment mechanism (Barabási and Albert 1999). Because real-world terrorist networks are commonly sparse (Krebs 2002), we use ER random graphs with parameters $M=V, 2V$, labelled as ER(1) and ER(2), respectively, and use BA scale-free networks with parame-




(a) RT (b) BA(1) (c) ER(1) (d) BA(2) (e) ER(2)

(f) RT (g) BA(1) (h) ER(1) (i) BA(2) (j) ER(2)

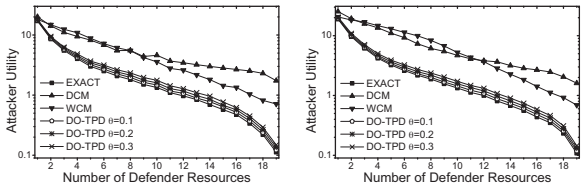
Figure 1: Runtime ((a) - (e)) and attacker utility ((f) - (j), y-axis is plotted on a log scale)

ters $k = 1, 2$, labelled as BA(1) and BA(2), respectively. The capability of each vertex $\tau_v$ is randomly chosen in $[1, 5]$, and the network externality measure $\delta$ is fixed to be 0.1.

We compare the scalability of four versions of our algorithms: (i) *DO-TPD*: Algorithm 1; (ii) *DO-TPD/singleResponse*: revised *DO-TPD* which only generates one improving strategy for both players per iteration; (iii) *DO-TPD/randomstart*: modified *DO-TPD* where $\mathcal{S}'$ and $\mathcal{A}'$ are initialized with several randomly generated pure strategies; (iv) *DO-TPD/Appro*: an approximation variant of *DO-TPD* which does not call *bestO-A* and *bestO-D*. The naïve algorithm of solving the LP in Eqs.(4)-(7) directly (*FullLP*) is also included as a benchmark.

To evaluate the solution quality of our algorithms, we introduce six centrality measure based heuristic baselines. The solution quality of a baseline strategy $\mathbf{x}$ is assessed in terms of attacker utility $U_a(\mathbf{x}, \mathbf{y})$, where $\mathbf{y}$ is the attacker best response of $\mathbf{x}$. A lower attacker utility indicates a higher defender utility given the zero-sum assumption. The baselines are: (i) *DCP*: a pure defender strategy monitoring vertices with top $R$ degree values; (ii) *DCM*: a defender mixed strategy where marginal coverage probability of each vertex is normalized degree centrality; (iii) *WCP*: a pure defender strategy monitoring vertices with top $R$ node capability values; (iv) *WCM*: a defender mixed strategy where marginal coverage probability of each vertex is normalized weight centrality; (v) *SCP*: a pure defender strategy monitoring vertices with top $R$ Shapley values computed by FasterSVCG algorithm in (Michalak et al. 2013); (vi) *SCM*: a defender mixed strategy where marginal coverage probability of each vertex is normalized Shapley value based centrality. Note that, given the marginal coverage probabilities, all the mixed strategies are generated using the Comb Sampling algorithm (Tsai et al. 2010).

**Scalability Analysis.** In Figures 1(a) - 1(e), we compare the scalability of the proposed algorithms on 5 types of networks. Results show that *DO-TPD* and *DO-TPD/Appro* outperform alternatives significantly by several orders of magnitude. The runtime of *FullLP* exponentially increases with the size and runs out of memory when the network size is about 20. Comparing with *DO-TPD/singleResponse* and *DO-TPD/randomstart*, it is revealed that generating multiple better responses and *LWA-LP* initialization procedure in *DO-TPD* can reduce the number of overall iterations. Overall, *DO-TPD* and *DO-TPD/Appro* perform the best and can scale up to realistic-sized problems. As stated in Section 2, France has about **100** such potential terrorists, which is within the scalability of our approach.

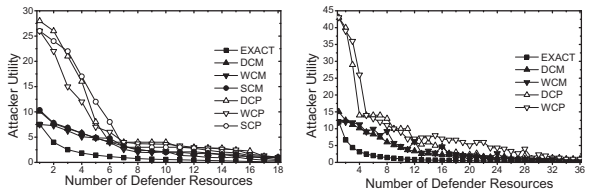



(a) ER(1), $|V| = 20$ (b) BA(2), $|V| = 20$

Figure 2: Robustness results on different networks

**Solution Quality Analysis.** Figure 1 compares the solution quality obtained from our approaches with those obtained from the 6 heuristic baselines, in which *EXACT* repents the solution quality obtained by *DO-TPD*. Results show that the solutions of all the heuristic baselines are very bad in general and become worse with the increasing network size, while *DO-TPD/Appro* can always achieve an almost optimal solution. *WCP* performs dramatically worse in all the cases because it was designed entirely **without considering the network structure**. All the mixed strategy based heuristics (i.e., *DCM*, *WCM* and *SCM*) perform much better than corresponding pure strategy based heuristics (i.e., *DCP*, *WCP* and *SCP*), which confirms the necessity of **using randomized monitoring schedules**. The good performance of *DO-TPD/Appro* indicates the effectiveness of our better-response oracles.

**Robustness.** The defender's estimation of terrorists' capability may not be perfect, thus we analyze the performance of *DO-TPD* considering noise of $\tau$. Let $\hat{\tau}$ be the defender's estimation of $\tau$ while $\hat{\tau}$ is drawn uniformly from




(a) small 9/11 network (b) bigger 9/11 network

Figure 3: Attacker utility results on 9/11 networks

$\tau \cdot [1-\theta, 1+\theta]$. We compare the attacker utility of *DO-TPD* under different degrees of uncertainty, with the optimal solution, denoted by *EXACT*, and two best heuristic solutions (i.e., *DCM* and *WCM*). Figure 2 shows a decreasing efficiency of *DO-TPD* with increasing $\theta$. The solutions are still near-optimal and outperform *DCM* and *WCM* significantly.

**Application on 9/11 Terrorist Network.** We conduct experiments on two 9/11 networks (a small one with 19 vertices, who are directly responsible for this attack, and a bigger one with 37 vertices including accomplices (Krebs 2002)) with the same node capabilities and payoff function as Lindelauf et al. (2013), to compare our *DO-TPD* with these centrality metrics based heuristics, and *DO-TPD* can solve both networks efficiently, no matter how many defender resources are provided. The solution qualities are depicted in Figure 3, from which we can see that *DO-TPD* outperforms all heuristics significantly, especially with scarce defender resources, which is exactly the case in reality.

## 6 Conclusion

This paper presents the first work on applying Stackelberg security games for detecting terrorist plots. We propose an efficient double oracle framework based algorithm with novel MILPs for NP-hard best-response oracles. We also provide a greedy defender better-response oracle with $1 - \frac{1}{e}$ approximation ratio and a fast attacker better-response oracle based on iterated local search. Experimental results show that our algorithm can obtain a robust enough solution outperforming widely-used centrality based heuristics significantly and can scale up to realistic-sized problems.

## References


Barabási, A.-L., and Albert, R. 1999. Emergence of scaling in random networks. *Science* 286(5439):509–512.

BBC. 2015a. *Charlie Hebdo* attack: A French intelligence failure? http://www.bbc.com/news/world-europe-30760656.

BBC. 2015b. Paris attacks: Suspects' profiles. http://www.bbc.com/news/world-europe-30722038.

Blum, A.; Haghtalab, N.; and Procaccia, A. D. 2014. Learning optimal commitment to overcome insecurity. In *NIPS*, 1826–1834.

CNN. 2015a. French monitored Kouachi brothers but lost interest, despite red flags. http://edition.cnn.com/2015/01/10/europe/kouachi-brothers-surveillance/.

CNN. 2015b. Report: Chérif Kouachi texted Coulibaly an hour before Paris attacks began. http://edition.cnn.com/2015/02/17/world/france-charlie-hebdo-attacks/.

CNN. 2015c. Who are suspects in two violent French standoffs? http://edition.cnn.com/2015/01/08/europe/paris-charlie-hebdo-shooting-suspects/.

Enders, W., and Jindapon, P. 2010. Network externalities and the structure of terror networks. *Journal of Conflict Resolution* 54(2):262–280.

Enders, W., and Su, X. 2007. Rational terrorists and optimal network structure. *Journal of Conflict Resolution* 51(1):33–57.

Erdős, P., and Rényi, A. 1959. On random graphs. *Publicationes Mathematicae Debrecen* 6:290–297.

Gan, J.; An, B.; and Vorobeychik, Y. 2015. Security games with protection externality. In *AAAI*, 914–920.

Hansen, P., and Mladenović, N. 2001. Variable neighborhood search: Principles and applications. *European Journal of Operational Research* 130(3):449–467.

I2. 2010. Analyst's notebook 8, Social network analysis whitepaper. www.i2group.com.

Jain, M.; Conitzer, V.; and Tambe, M. 2013. Security scheduling for real-world networks. In *AAMAS*, 215–222.

Krebs, V. E. 2002. Mapping networks of terrorist cells. *Connections* 24(3):43–52.

Lindelauf, R.; Hamers, H.; and Husslage, B. 2013. Cooperative game theoretic centrality analysis of terrorist networks: The cases of Jemaah Islamiyah and Al Qaeda. *European Journal of Operational Research* 229(1):230–238.

McMahan, H. B.; Gordon, G. J.; and Blum, A. 2003. Planning in the presence of cost functions controlled by an adversary. In *ICML*, 536–543.

Michalak, T. P.; Rahwan, T.; Jennings, N. R.; Szczepański, P. L.; Skibski, O.; Narayanam, R.; and Wooldridge, M. J. 2013. Computational analysis of connectivity games with applications to the investigation of terrorist networks. In *IJCAI*, 293–301.

Shieh, E.; An, B.; Yang, R.; Tambe, M.; Maule, B.; and Meyer, G. 2012. PROTECT: An application of computational game theory for the security of the ports of the United States. In *AAAI*, 2173–2179.

Tambe, M. 2011. *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*. Cambridge University Press.

Tsai, J.; Yin, Z.; Kwak, J.; Kempe, D.; Kiekintveld, C.; and Tambe, M. 2010. Urban security: Game-theoretic resource allocation in networked physical domains. In *AAAI*, 881–886.

Vorobeychik, Y.; An, B.; Tambe, M.; and Singh, S. P. 2014. Computing solutions in infinite-horizon discounted adversarial patrolling games. In *ICAPS*, 314–322.

Woo, G. 2009. Intelligence constraints on terrorist network plots. In Memon, N.; David Farley, J.; Hicks, D.; and Rosenorn, T., eds., *Mathematical Methods in Counterterrorism*. 205–214.

Yin, Y.; An, B.; and Jain, M. 2014. Game-theoretic resource allocation for protecting large public events. In *AAAI*, 826–834.

Yin, Y.; Xu, H.; Gan, J.; An, B.; and Jiang, A. X. 2015. Computing optimal mixed strategies for security games with dynamic payoffs. In *IJCAI*, 681–687.