Draining the Water Hole: Mitigating Social Engineering Attacks

Zheyuan Ryan Shi[1] , **Aaron Schlenker**[2] , **Brian Hay**[3] and **Fei Fang**[1]
[1]Carnegie Mellon University
[2]University of Southern California
[3]Virginia Tech

Abstract

Cyber adversaries have increasingly leveraged social engineering attacks to breach large organizations and threaten the well-being of today's online users. One clever technique, the "water-hole" style attack, compromises a legitimate website to execute drive-by download attacks by redirecting users to another malicious domain. We introduce a game-theoretic model that captures the salient aspects for an organization protecting itself from a water-hole attack by altering the environment information in web traffic so as to deceive the attackers. Our main contributions are (1) a novel Social Engineering Deception (SED) game model that features a continuous action set for the attacker, (2) an in-depth analysis of the SED model to identify computationally feasible real-world cases, and (3) an iterative algorithm which solves for the optimal protection policy using (i) a characterization of websites that may be compromised, (ii) an LP-relaxation with optimality condition, and (iii) the column generation method. A Chrome extension is being built to field our algorithms in the real world.

1 Introduction

Social engineering attacks are a scourge for the well-being of today's online user and the current threat landscape only continues to become more dangerous [Mitnick and Simon, 2001]. Social engineering attacks manipulate people to give up confidential information through the use of phishing campaigns, spear phishing whaling or water-hole style attacks [Krombholz *et al.*, 2015; Abraham and Chengalur-Smith, 2010]. For example, in water-hole style attacks, the attackers compromise a legitimate website and redirect visitors to a malicious domain where the attackers can infect the user's computer and intrude the user's network. The number of social engineering attacks is growing at a catastrophic rate. In a recent survey, 60% organizations were or may have been victims of at least one targeted attacks [Agari, 2016]. Such cybercrime poses an enormous threat to the peace and security at all levels – national, business, and individual, challenging the Sustainable Development Goals set by the United Nations [UN, 2015].

To mitigate these attacks, organizations take countermeasures which vary from training employees to recognize social engineering to technology-based defenses. Security training on employees [Orgill *et al.*, 2004] has limited efficacy against a motivated adversary's targeted campaign [Junger *et al.*, 2017]. Existing technology-based countermeasures are also largely inadequate. Water-hole attackers typically use zero-day exploits, rendering patching and updating the systems almost useless [Sutton, 2014]. Sandboxing the potential attack by using a VM requires high-end hardware, which hinders its wide adoption [Farquhar, 2017]. White/blacklisting websites is of limited use, since the adversary is strategically infecting trustworthy websites.

We propose a game-theoretic deception framework to mitigate social engineering attacks, and in particular, the water-hole style attacks. Deception is to delay and misdirect an adversary by incorporating ambiguity. Recent papers have explored deception techniques such as sending erroneous responses to scan queries during the reconnaissance phase of an attack [Jajodia *et al.*, 2017; Schlenker *et al.*, 2018]. Water-hole attackers rely on the identification of a visitor's system environment to deliver the correct malware to compromise a victim. Towards this end, the defender can manipulate the identifying information in the network packets, such as the user-agent string, IP address, time-to-live, etc. Consequently, the attacker might get false or confusing information about the environment and send incompatible exploits. Thus, deceptively manipulating employees' network packets provides a potential countermeasure to social engineering attacks.

Our Contributions We provide the first game-theoretic framework for deploying autonomous countermeasures to social engineering attacks. We propose the Social Engineering Deception (SED) game, in which the organization (defender) strategically alters the network packets of its employees' web traffic. The attacker selects some websites to compromise, and captures the organization's traffic to these websites to launch an attack. The identification of the traffic and the success of the attack depends on whether the traffic is altered. We model it as a zero-sum game and consider the minimax strategy for the defender.

Second, we analyze the structure of the SED game. We show that it is NP-hard for the adversary to best respond to an alteration policy, and identify real-world scenarios where the optimal protection policy can be found efficiently. We also characterize a subset of dominated websites to which the defender never needs to alter the traffic and can be eliminated.

Third, we propose a column-generation based algorithm to solve the SED game. The algorithm uses the elimination of

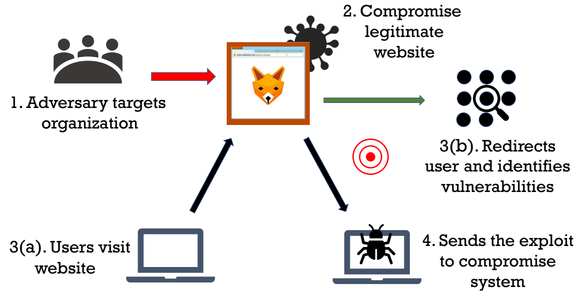

1. Adversary targets organization
2. Compromise legitimate website
3(b). Redirects user and identifies vulnerabilities
3(a). Users visit website
4. Sends the exploit to compromise system

Figure 1: Anatomy of a water-hole attack carried out by hackers targeting an organization.

dominated websites to reduce the problem size. It also leverages a linear program (LP) relaxation which efficiently approximates the optimal solution with an optimality condition.

Finally, we provide extensive experiments to evaluate our algorithms. We show that our algorithm can handle corporate-scale instances on 10^5 websites.

2 Water-hole Attacks

Water-hole attacks are a prominent type of social engineering used by sophisticated attackers to identify the vulnerable systems in an organization's network. For example, in 2013, a water-hole attack with the purpose of technological espionage targeted several large corporate networks, including Apple, Facebook, and Twitter, and succeeded [Whittaker, 2013]. More recent Ratankba malware attacks in 2017 targeted 100 organizations in 31 different countries with the vast majority of the victims being banks [Symantec, 2017].

To gain a better understanding of our contribution, we list the primary steps in executing a water-hole attack, as illustrated in Fig. 1. In step 1, the adversary identifies a target organization. They use tracking services such as KISSMetrics and AddThis to understand the browsing habits of employees of the organization and the connections to the organization's network. This allows the adversary to determine the most lucrative websites to compromise for maximum exposure. In step 2, the adversary attempts to compromise these websites. In step 3, employees visit the compromised website and are redirected to another website which scans their system environment and the present vulnerabilities. To gather this information, several techniques can be used, including analyzing the user-agent string, operating system fingerprinting, etc. Step 4 sees the adversary deliver an exploit for an identified vulnerability to compromise the user's machine. With all these steps, the adversary can navigate the organization's network and access the desired sensitive information.

The countermeasure we study introduces uncertainty and difficulty in Step 3 of a water-hole attack. Identifying the vulnerabilities in a visitor relies on the gathered information from reconnaissance methods. By introducing deception during this phase, it helps the defender prevent a water-hole attack before employees are compromised and provides an opportunity to identify attempted attacks and infected websites.

It is worth noting that in reality, sophisticated attackers typically do not send all, untailored, exploits regardless of the packet information. This is because defense would become easier due to the defender seeing more such unknown exploits which are expensive to acquire otherwise. Also, sending all exploits would be flagged as suspicious and get blocked. The attacker then needs to get a new 0-day – a costly proposition.

3 The Social Engineering Deception Game

In an SED game, an organization (defender) O aims to protect its employees I from a water-hole attack. We model the interaction between the organization and an adversary as a zero-sum game, where the defender chooses an alteration policy and the adversary chooses which websites to compromise and decides the effort spent on scanning traffic. In everyday activities employees visit a set of websites W which contains legitimate sites and potential water-holes set up by an adversary. The defender's alteration policy x_w represents the proportion of the organization's traffic to website $w \in W$ for which the network packet will be altered. We assume that if an employee's packet is altered a drive-by download attack will be unsuccessful; if the traffic is unaltered the attack will succeed. We discuss relaxing this assumption in Section 8.

For each website w, there is a total amount of traffic t_w^{all}. Denote t_w^i as the traffic from employee $i \in \mathcal{I}$ to w. The total traffic from O to w can then be captured by $t_w = \sum_i t_w^i$. As altering the network packet can degrade the webpage rendered, e.g., displaying for Android on a Windows desktop, we consider a cost c_w to alter a single unit of traffic to w. The defender is limited to a budget B_d on the allowable cost due to the alterations, i.e. $\sum_{w \in W} c_w t_w x_w \leq B_d$.

The adversary chooses which websites to compromise, represented by binary decision variables y_w. He has a cost π_w for compromising a website: the work to turn a legitimate site into a water-hole. Compromising google.com is nearly impossible while the Polish Financial Authority, victim of the 2017 Ratankba malware attacks [Symantec, 2017], cannot invest the same security resources. Without this cost, he would compromise all websites – which is not what has been observed from previous water-hole attacks [Parliament, 2018]. The attacker has a budget B_a for compromising websites (we assume $\pi_w \leq B_a \; \forall w \in W$) , i.e., $\sum_{w \in W} \pi_w y_w \leq B_a$.

Given a unit traffic to a compromised website, the attacker spends a certain amount of effort to determine the visitor's system environment and whether it comes from the target organization. We use e_w to represent how much traffic the attacker decides to scan per week for w. He has a budget B_e for scanning the incoming traffic per week, i.e., $\sum_{w \in W} e_w \leq B_e$ and $e_w \leq t_w^{all} \cdot y_w$ for $w \in W$. In the special case where this scanning effort is negligible ($B_e = \infty$), all our complexity and algorithmic results still hold.

We consider an attacker who maximizes the expected amount of unaltered flow from O through scanning. This is justifiable when there is a fixed probability of successfully finding out information about O every time the attacker traces the traffic from O. The adversary's objective is then to maximize $\sum_{w \in W} t_w (1 - x_w) e_w / t_w^{all}$.

4 Analysis of SED Games

We aim to find the defender's minimax strategy, for which we can formulate the following optimization problem $\mathcal{P}_1$.

$$
\begin{aligned}
\mathcal{P}_1: \min_x \max_{y,e} \quad & \sum_{w\in W} \frac{t_w(1-x_w)e_w}{t_w^{all}} && (1)\\
\text{s.t.} \quad & \sum_{w\in W} e_w \le B_e && (2)\\
& \sum_{w\in W} \pi_w y_w \le B_a && (3)\\
& e_w \le t_w^{all}\cdot y_w \qquad \forall w\in W && (4)\\
& y_w \in \{0,1\}, e_w\in[0,\infty) \qquad \forall w\in W && (5)\\
& \sum_{w\in W} c_w t_w x_w \le B_d && (6)\\
& x_w\in[0,1] \qquad \forall w\in W && (7)
\end{aligned}
$$

4.1 Hardness Result

Given a defender strategy x, the adversary's best response problem $\mathcal{P}_2(x)$ is the inner maximization problem of $\mathcal{P}_1$:

$$
\begin{aligned}
\mathcal{P}_2(x): \quad \max_{y,e} \quad & \sum_{w\in W} \frac{t_w(1-x_w)e_w}{t_w^{all}} && (8)\\
\text{s.t.} \quad & \text{Constraints}(2) \sim (5) && (9)
\end{aligned}
$$

First, we note that computing the adversary's best response is NP-hard. Its proof is deferred to Appendix B.1.

Theorem 1. *Finding adversary's best response is NP-hard.*

4.2 Tractable Cases

Despite the hardness result, we are able to show that the optimal solutions of SED games exhibit a greedy allocation of the attacker's effort budget. That is, for at most one website w will the attacker spend scanning effort neither zero nor t_w^{all}. This leads to a tractable case of the SED game $\mathcal{P}_1$. The proof of the following results can be found in Appendix B.2-B.3.

Theorem 2. *Let (x^*, y^*, e^*) be an optimal solution to $\mathcal{P}_1$. Let $W_F = \{w : e_w^* = t_w^{all}\}, W_Z = \{w : e_w^* = 0\}, W_B = \{w : e_w^* \in (0, t_w^{all})\}$. There is an optimal solution with $|W_B| \le 1$.*

Corollary 1. *If $0 < B_e \le t_w^{all} \;\; \forall w \in W$, the defender's optimal strategy can be found in polynomial time.*

Another simplified version of SED, where the attacker has uniform cost π_w and unlimited effort budget B_e, also has real-world implications. In some scenarios, the attacker may have a systematic way of compromising a website, making the cost π_w uniform. If the attacker trusts the IP address is not altered by the defender, the scanning effort e_w is negligible. The attacker's best response is then obviously polynomial time solvable. We show that the defender's optimal strategy can also be found efficiently, with proof given in Appendix B.4.

Theorem 3. *If $\pi_w = 1, \forall w \in W$ and $B_e = \infty$, the defender's optimal strategy can be found in polynomial time.*

4.3 Dominated Websites

Not all websites are equally valuable for an organization as some are especially lucrative for an adversary to target. In a Polish bank, many employees may visit the Polish Financial Authority website daily, while perhaps a CS conference website is only visited by a bored banker once a week. Intuitively, attackers will not consider the conference website and thus, the bank may not need to alter traffic to the conference website. If we can identify these websites ahead of time, we could greatly reduce the size of our problem.

Algorithm 1: FIND-DOMINATED-WEBSITES

```
1 Define $U = \{w \in W : c_w t_w \ge B_d\}$. Let $D = \emptyset$.
2 foreach website $u \in U$ do Calculate $x_u^{max} = \frac{B_d}{c_u t_u}$
3 foreach website $w \in W$ do
4     Set $U_w = \{u \in U : \frac{t_w}{t_w^{all}} \le \frac{t_u(1-x_u^{max})}{t_u^{all}}\}$
5     if exists $U_w^* \subseteq U_w$ such that
6     (1) $\sum_{u\in U_w^*} \pi_u \le \pi_w$, (2) $\sum_{u\in U_w^*} t_u^{all} \ge t_w^{all}$, and (3)
      $\sum_{u\in U_w^*} t_u^{all} \ge B_e$ then $D = D \cup \{w\}$
7 return set of dominated websites $D$
```

We now formalize this intuition. A website w is **dominated by another website** u if the attacker would not attack w unless he has used the maximum effort on u, i.e. $e_u = t_u^{all}$, regardless of the defender's strategy. In other words, the attacker would get less utility on w than u per unit effort, even if the defender does not cover w at all and covers u to the maximum capacity. Suppose $c_u t_u > B_d$, which means the defender can only cover u to the extent $x_u < x_u^{max} := \frac{B_d}{c_u t_u}$. Then, the attacker is guaranteed to get utility at least $\frac{t_u(1-x_u^{max})e_u}{t_u^{all}}$ by compromising this website with an effort e_u. This leads us to the following theorem, whose proof appears in Appendix B.5.

Theorem 4. *Consider websites $u, w \in W$. If the following conditions hold, the website w is dominated by u:*

$$
x_u^{max} := \frac{B_d}{c_u t_u} \le 1, \qquad \frac{t_w}{t_w^{all}} \le \frac{t_u(1-x_u^{max})}{t_u^{all}},
$$
$$
\pi_w \ge \pi_u, \qquad t_w^{all} \le t_u^{all}.
$$

If a website w is often visited worldwide, yet seldom visited from the organization, then it is unlikely a unit of traffic to w, scanned by the adversary, is from the targeted organization. Thus, the adversary may not compromise w as long as he has more profitable, efficient, and feasible options which already exhaust his effort budget. A website w is a **dominated** website if the adversary would not attack w, regardless of his strategy on other websites and the defender's strategy. Theorem 4 suggests the procedure in Alg. 1 to identify dominated websites.

We remark that, if all attack costs π_w are uniform, the condition on Line 6 of Alg. 1 becomes "find $u \in U_w$ such that $t_u^{all} \ge t_w^{all}$ and $t_u^{all} \ge B_e$". Then Alg. 1 can run in polynomial time, and we show its effectiveness in the experiment section.

4.4 An Upper Bound for $\mathcal{P}_1$

Furthermore, we can obtain an upper bound for $\mathcal{P}_1$. This upper bound will play an important role in our algorithm to solve $\mathcal{P}_1$ in the next section. It is also useful in its own right.

Let $\hat{\mathcal{P}}_2(x)$ be the LP relaxation of $\mathcal{P}_2(x)$, i.e.,

$$
\begin{aligned}
\hat{\mathcal{P}}_2(x): \quad \max_{y,e} \quad & \sum_{w\in W} \frac{t_w(1-x_w)e_w}{t_w^{all}} && (10)\\
\text{s.t.} \quad & \text{Constraints}(2) \sim (4) \\
& y_w \in [0,1], e_w \in [0,\infty) \qquad \forall w\in W && (11)
\end{aligned}
$$

We formulate the dual of $\hat{\mathcal{P}}_2(x)$, and denote the dual variables as the $\lambda_1, \lambda_2, \nu, \eta$ for constraints (2) $\sim$ (4) and (11). We then include the variable x for the defender strategy along with the dual problem, and obtain the minimization problem $\hat{\mathcal{P}}_1$.

$$\begin{aligned}
\hat{\mathcal{P}}_1 : \min_{x,\lambda,\nu,\eta} \quad & B_e\lambda_1 + B_a\lambda_2 + \sum_{w\in W}\eta_w && (12)\\
\text{s.t.} \quad & \frac{t_w(1-x_w)}{t_w^{all}} \le \lambda_1 + \nu_w \qquad \forall w \in W && (13)\\
& \pi_w\lambda_2 - t_w^{all}\nu_w + \eta_w \ge 0 \qquad \forall w \in W && (14)\\
& \sum_{w\in W} c_w t_w x_w \le B_d && (15)\\
& x_w \in [0,1],\ \lambda_1, \lambda_2, \nu_w, \eta_w \ge 0 \quad \forall w \in W && (16)
\end{aligned}$$

For an optimization problem $\mathcal{P}$, we denote its optimal value as OPT($\mathcal{P}$). We show that the relaxed problem $\hat{\mathcal{P}}_1$ yields a 3-approximation to $\mathcal{P}_1$, with proof in Appendix B.6.

Theorem 5. *If the attacker's budgets satisfy $B_a \ge \max_w \pi_w$ and $B_e \ge \max_w t_w^{all}$, then $OPT(\hat{\mathcal{P}}_1) \le 3OPT(\mathcal{P}_1)$.*

The following result establishes the connection between the optimization problems we have constructed so far. It illustrates that the optimal value of $\mathcal{P}_1$ is upper bounded by its LP relaxation as well as the adversary best response problems, whose proof appears in Appendix B.7. This immediately suggests a characterization of how little we lose by solving the LP relaxation $\hat{\mathcal{P}}_1$ instead of $\mathcal{P}_1$.

Theorem 6. *Let x^* be an optimal solution to $\mathcal{P}_1$, and $\hat{x}^*$ be an optimal solution to $\hat{\mathcal{P}}_1$. We have*

$$OPT(\mathcal{P}_1) \le OPT(\mathcal{P}_2(\hat{x}^*)) \le OPT(\hat{\mathcal{P}}_1) \le OPT(\hat{\mathcal{P}}_2(x^*)).$$

If $OPT(\hat{\mathcal{P}}_2(x^*)) - OPT(\mathcal{P}_1) \le \epsilon$, $\hat{\mathcal{P}}_1$ is an ϵ-approximation to the SED problem $\mathcal{P}_1$. If $OPT(\hat{\mathcal{P}}_2(x^*)) = OPT(\mathcal{P}_1)$, the SED game can be solved in polynomial time by $\hat{\mathcal{P}}_1$.

Theorem 5 leads to an efficient approximation algorithm for the SED game, yet Theorem 6 does not offer direct algorithmic implications. Nevertheless, Theorem 6 only depends on the SED problem itself. It gives an intrinsic property which characterizes the structure of SED problems, which can be seen as another tractable class of SED games.

4.5 Optimality Conditions for $\hat{\mathcal{P}}_1$

Theorem 5 provides an approximation bound for $\hat{\mathcal{P}}_1$. As the problem size grows, LP relaxation often yields a much better solution in practice than the worst case guarantee. We now present a sufficient condition for the optimal solution $\hat{x}^*$ to $\hat{\mathcal{P}}_1$ to be also optimal for $\mathcal{P}_1$. In the following linear program $\tilde{P}_3(\hat{x}^*)$, we enumerate all the optimal solutions (max effort vectors) to the problem $\mathcal{P}_2(\hat{x}^*)$ and choose an adequately small ϵ. We can obtain $e^{\mathcal{P}_2(\hat{x}^*)}$, the set of these optimal max effort vectors, efficiently in practice when solving $\mathcal{P}_2(\hat{x}^*)$.

$$\begin{aligned}
\tilde{P}_3(\hat{x}^*) : \min_{x,v} \quad & v && (17)\\
\text{s.t.} \quad & v \ge \sum_{w\in W}\frac{t_w}{t_w^{all}}(1-x_w)e_w \quad \forall e \in e^{\mathcal{P}_2(\hat{x}^*)} && (18)\\
& \sum_{w\in W} |x_w - \hat{x}^*| \le \epsilon && (19)\\
& \sum_{w\in W} c_w t_w x_w \le B_d && (20)\\
& x_w \in [0,1] \qquad \forall w \in W && (21)
\end{aligned}$$

Claim 1. *If $OPT(\tilde{P}_3(\hat{x}^*)) \ge OPT(P_2(\hat{x}^*))$, $\hat{x}^*$ is optimal for $\mathcal{P}_1$. If $\tilde{P}_3(\hat{x}^*)$ yields some optimal solution x' with $OPT(P_2(x')) < OPT(P_2(\hat{x}^*))$, $\hat{x}^*$ is not optimal for $\mathcal{P}_1$.*

5 Computing an Optimal Defender Strategy

We first introduce SED-IA, our algorithm for computing the optimal defender strategy, based on Theorem 2 and the column generation method. We then further improve its scalability using dominated website elimination and LP-relaxation.

Define $\hat{e}^{\mathcal{A}}$ as the set of all **max effort vectors**. That is, $e \in \hat{e}^{\mathcal{A}}$ if $\sum_w e_w = B_e$ and $|W_B| \le 1$. By Theorem 2, if we enumerate all of $\hat{e}^{\mathcal{A}}$, we would be able to find an optimal defender strategy x using the LP in Equations (22)-(25) with $e^{\mathcal{A}}$ in Equation (23) replaced by $\hat{e}^{\mathcal{A}}$.

$$\begin{aligned}
\min_{x,v} \quad & v && (22)\\
\text{s.t.} \quad & v \ge \sum_{w\in W}\frac{t_w}{t_w^{all}}(1-x_w)e_w \qquad \forall e \in e^{\mathcal{A}} && (23)\\
& \sum_{w\in W} c_w t_w x_w \le B_d && (24)\\
& x_w \in [0,1] \qquad \forall w \in W && (25)
\end{aligned}$$

However, the order of $\hat{e}^{\mathcal{A}}$ is prohibitively high. In the following section, we refer to this approach as MAX EFFORT, and show that it has poor scalability. We propose an iterative algorithm SED-IA (Alg.2) to find the optimal defender strategy. It first finds a feasible defender strategy x by solving $\hat{\mathcal{P}}_1$, as shown in Section 4.4. SED-IA is based on the column generation technique [Gilmore and Gomory, 1961]. It iteratively solves the adversary's problem and the defender's problem. Given a defender strategy x, we solve the adversary's optimization problem $\mathcal{P}_2(x)$ and obtain an adversary best response effort vector e. Instead of enumerating all of $\hat{e}^{\mathcal{A}}$, we keep a running subset $e^{\mathcal{A}} \subseteq \hat{e}^{\mathcal{A}}$ of max effort vectors, and add the best response effort vector e to $e^{\mathcal{A}}$. We solve the defender's optimization problem in Equations (22)$\sim$(25) with this running subset $e^{\mathcal{A}}$. This process repeats until no new effort vectors are found for the adversary.

Claim 2. SED-IA *terminates with the optimal solution.*

In light of the hardness of finding the adversary's best response, we consider a greedy heuristic. The algorithm allocates the adversary's budget to websites in decreasing order of $r_w = \frac{t_w(1-x_w)/t_w^{all}}{\alpha_w}$, where α_w is a tuning parameter. We also consider an exact dynamic programming algorithm. Details about these algorithms can be found in Appendix A.

Algorithm 2: SED-IA

```
1 Get initial defender strategy $x$ by solving $\hat{\mathcal{P}}_1$.
2 Initialize max effort vector set $e^{\mathcal{A}} = \emptyset$.
3 while new max effort vector was added to $e^{\mathcal{A}}$ do
4     Solve $\mathcal{P}_2(x)$ for adversary's best response $e$
5     Add $e$ to $e^{\mathcal{A}}$.
6     Solve defender's optimal strategy problem w.r.t.
        effort set $e^{\mathcal{A}}$ as the LP in Equations (22)$\sim$(25).
7     Update the game value and defender's strategy $x$.
```

We leverage the previously established results to further improve SED-IA. Before SED-IA starts, we remove the dominated websites using Alg. 1. In line 1 of SED-IA, instead of merely using $\hat{\mathcal{P}}_1$ to find an initial feasible solution, we check the optimality of the LP relaxation using the conditions in Section 4.5. We refer to this enhanced algorithm as SED-ALL, whose details are included in Appendix A.

6 Experiments

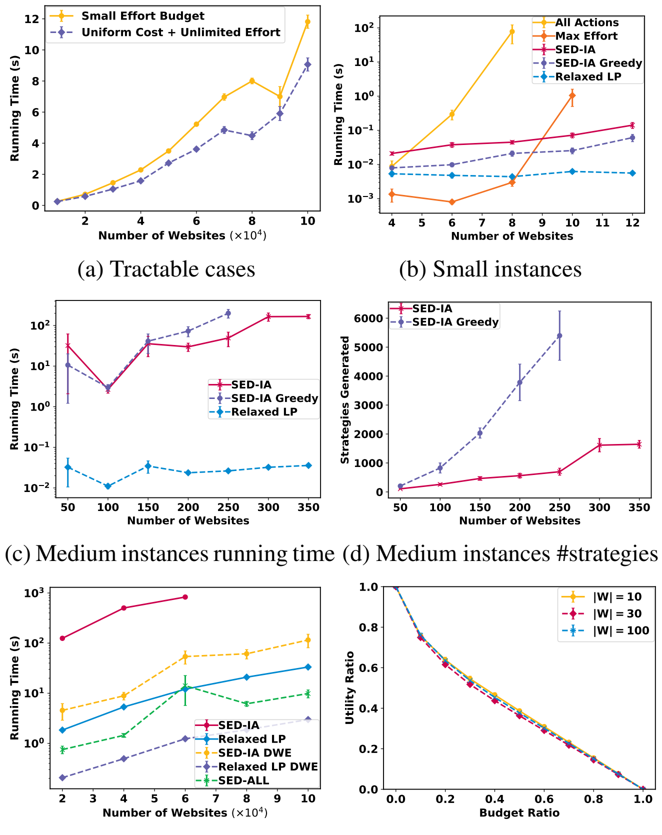
Considering the typical scenarios of social engineering attacks, we developed and tested our algorithms to match the scalability required of large-scale deployment. All tests are run on a 3.8GHz Intel Core i5 CPU with 32G RAM. We solve optimization problems using CPLEX 12.8. Unless otherwise noted, problem parameters are generated from distributions in Table 3 in Appendix D. All results are averaged over 20 instances; error bars represent standard deviations of the mean.

Tractable cases First, we run experiments on the polynomial time tractable cases of SED, according to Corollary 1 and Theorem 3. In the case of small effort budget ($B_e \leq t_w^{all}$ for all $w \in W$), B_e is generated uniformly between 1 and $\min_{w\in W} t_{all}^w$. Fig. 2a shows that in both cases, our solution can be easily scaled to 10^5 websites. This implies these cases could be deployed in real-world corporate-scale applications.

Small instances With this, we move on to the general case of SED games, where we test three algorithms (SED-IA, SED-IA GREEDY, and RELAXED LP) with two other baselines (MAX EFFORT and ALL ACTIONS). SED-IA GREEDY refers to replacing the MILP for $\mathcal{P}_2(x)$ in SED-IA with the GREEDY best response heuristic. We include additional experiments on the best response heuristics in Appendix C. RELAXED LP refers to solving $\hat{\mathcal{P}}_1$. The first baseline algorithm, MAX EFFORT, is explained in Section 5. The other baseline, ALL ACTIONS, decomposes SED into subproblems, each assuming some adversary's maximal effort vector is a best response. The details of this algorithm can be found in Appendix A.

As shown in Fig. 2b, both baseline algorithms quickly become impractical even on problems with less than 12 websites. On the other hand, the iterative approaches are able to find the optimal solutions rather efficiently. In particular, GREEDY slightly improves the performance of SED-IA. RELAXED LP yields the fastest running time, while it also comes with a solution gap above 6% as shown in Table 1.

Medium-size instances In Fig. 2c, we advance to examine our algorithms on larger problem instances. GREEDY stops



(a) Tractable cases (b) Small instances

(c) Medium instances running time (d) Medium instances #strategies

(e) Large instances (f) Trade-off

Figure 2: Experiment results

Size	Gap	# Exact	Size	Gap	# Exact
4	13.19%	2 out of 20	150	7×10^{-8}	16 out of 20
8	8.11%	5 out of 20	200	8×10^{-10}	19 out of 20
12	6.63%	8 out of 20	250	0	20 out of 20
50	2×10^{-6}	18 out of 20	300	2×10^{-3}	17 out of 20
100	8×10^{-9}	19 out of 20	350	2×10^{-8}	18 out of 20

Table 1: Solution quality of RELAXED LP, with the number of instances where RELAXED LP solves the problem exactly.

being helpful: SED-IA GREEDY typically does not terminate in 600 seconds on problems of size $|W| \geq 300$. In fact, the number of "better" max effort vectors generated in SED-IA GREEDY far outnumbers the "best" max effort vectors in SED-IA (Fig. 2d). Thus, GREEDY saves time over MILP at the expense of more iterations, which eventually takes more time. Meanwhile, RELAXED LP has negligible running time, and Table 1 shows it often solves the problem optimally.

Large instances We test our complete algorithm SED-ALL on much larger problems. Different websites have different importance to the organization. We partition W into W_1, W_2, where $|W_1| = 0.1|W|$ and $|W_2| = 0.9|W|$. W_1 contains the websites with a large portion of their traffic from the organization, while W_2 the contains websites with a smaller portion. To account for this distinction, websites in W_1 and W_2 are generated from different distributions (Table 4, Appendix D). We assume the attacker has a uniform cost of attack.

SED-ALL combines SED-IA with dominated website elimination (DWE) and the optimality check for RELAXED LP. As shown in Fig. 2e, SED-ALL solves SED games with 10^5 websites in 10 seconds. Compared to RELAXED LP with DWE, SED-ALL guarantees the optimal solution. Compared to SED-IA with DWE, SED-ALL leverages the optimality of RELAXED LP in many instances to greatly reduce the running time. Fig. 2e also shows the efficacy of DWE. DWE enables SED-IA to solve most SED games of 10^5 websites in around 2 minutes. Meanwhile, without DWE, SED-IA typically does not terminate in 1000 seconds on SED games with $|W| \geq 8 \times 10^4$ websites. A similar speed-up is observed for RELAXED LP. Among the 20 instances for each problem size, in less than 4 of which DWE did not reduce the problem size by much. We thus group the instances according to this criteria, and report in Fig. 2e the results of the majority group where DWE eliminated a significant number of websites.

We note that the ratio $|W_1|/|W|$ could be a lot smaller in reality, and our algorithms with DWE would run even faster.

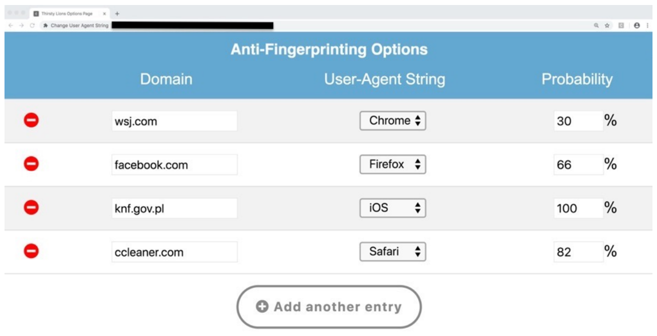
Trade-off analysis Finally, we consider the trade-off between the organization's risk exposure and the degradation in rendering websites, which can be represented by the objective $OPT(\mathcal{P}_1)$ and the defender's budget B_d, respectively. If the defender has budget $\bar{B}_d = \sum_{w \in W} c_w t_w$, the attacker would have zero utility. Meanwhile, if $B_d = 0$, the attacker gets some maximum utility $\bar{U}$. In Fig. 2f, we show how the utility ratio $OPT(\mathcal{P}_1)/\bar{U}$ changes with the budget ratio $B_d/\bar{B}_d$. As the organization increases the tolerance for service degradation, its risk exposure drops at a decreasing rate.

7 Related Work

Deception is regarded as one of the most effective ways to thwart cyber attacks [Virvilis *et al.*, 2014; Rowe, 2007]. For example, Albanese *et al.* [2016] study deceiving the adversary by making the network appear different from the true state, and Jajodia *et al.* [2017] use a probabilistic logic model to deceptively reply to adversary's scan requests. Schlenker *et al.* [2018] consider a defender who strategically replies to network probes. However, the use of deception to counter social engineering and water-hole attacks is underexplored.

Recently, deception has gained popularity with the rise of interest in applying game theory to cybersecurity settings [Serra *et al.*, 2015; Letchford and Vorobeychik, 2013; Basilico *et al.*, 2016]. In particular, many papers focused on the use of honeypots to sway the attacker away from attacking real servers [Kiekintveld *et al.*, 2015; Pıbil *et al.*, 2012; Durkota *et al.*, 2015]. We study a different domain of cybersecurity with different models. Another form of social engineering attacks, spear phishing, studied in [Laszka *et al.*, 2015], shares a few high-level modeling aspects. However, the different natures of attacks lead to different assumptions. For example, in SED it takes the adversary both an effort and a target-specific cost to compromise a website. This leads to different problem formulation and solution techniques.

In the technical vein, the SED problem $\mathcal{P}_1$ is similar to the recent work on bi-level knapsack with interdiction [Caprara *et al.*, 2016; Caprara *et al.*, 2014]. However, our outer problem of $\mathcal{P}_1$ is continuous rather than discrete, and the added dimension of adversary's effort makes the inner problem $\mathcal{P}_2(x)$ more complicated than that being studied in these works.


[redacted]

Anti-Fingerprinting Options

Domain	User-Agent String	Probability
wsj.com	Chrome	30 %
facebook.com	Firefox	66 %
knf.gov.pl	iOS	100 %
ccleaner.com	Safari	82 %

Figure 3: Screenshot of the browser extension

8 Conclusion and Discussion

Our work combines theoretical backing and practical usability, to potentially improve the existing approaches of web traffic tracking.

Proof of concept A browser extension is being built to field our algorithms. The software allows the user to change the user-agent string to appear as some other browser. A screenshot is shown in Fig 3. In future versions, the software will be able to change other information in the network packets. We will use it to gather real-world data to improve our model.

Assumptions and generality We assumed that the attack will succeed if and only if the network packet is unaltered. If the attacker can obtain the true system information with probability p_w even if the packet is altered, we may modify the objective in Eq. (1) to $\sum_w t_w(1-x_w(1-p_w))e_w/t_w^{all}$. If the organization has other countermeasures (e.g. Bromium browser VMs), the attack may fail with probability q_w even if the packet is unaltered, the objective then becomes $\sum_w t_w(1 - x_w)(1 - q_w)e_w/t_w^{all}$. Thus, our algorithm can account for different levels of adversary and defender sophistication.

We do not attempt to claim that altering the network packets is a panacea to all water-hole attacks. Cyber attackers have many tools to circumvent existing deception techniques. Nonetheless, the proposed deception technique increases the adversary's uncertainty about the true nature of the environment, which leads to a cost for the attacker, e.g. technical complexity and increased exposure. This uncertainty ties into our consideration of the attacker's scanning effort e_w and budget B_e, as the attacker cannot easily obtain or trust the basic information in the network packets.

Limitations The generality notwithstanding, we acknowledge a few limitations of our work and the potential problems in its large-scale deployment. First, if an organization is the sole user of our method and if the attacker has (possibly imperfect) clue about the source of traffic from the start, randomizing network packet information might serve as an unintended signal to the attacker, reducing the effort needed e_w to identify traffic from the targeted organization. Second, by manipulating the web traffic, the organization is effectively monitoring its employees' internet activities. Although in many jurisdictions this is allowed when doing properly, the potential ethical issues must be carefully addressed.

References

[Abraham and Chengalur-Smith, 2010] Sherly Abraham and InduShobha Chengalur-Smith. An overview of social engineering malware: Trends, tactics, and implications. *Technology in Society*, 32(3):183–196, 2010.

[Agari, 2016] Agari. *Email Security: Social Engineering Report*, 2016. https://www.ciosummits.com/2016-Social-Engineering-Report-Agari-Survey.pdf.

[Albanese *et al.*, 2016] Massimiliano Albanese, Ermanno Battista, and Sushil Jajodia. Deceiving attackers by creating a virtual attack surface. In *Cyber Deception*. 2016.

[Basilico *et al.*, 2016] Nicola Basilico, Andrea Lanzi, and Mattia Monga. A security game model for remote software protection. In *ARES*, 2016.

[Caprara *et al.*, 2014] Alberto Caprara, Margarida Carvalho, Andrea Lodi, and Gerhard J Woeginger. A study on the computational complexity of the bilevel knapsack problem. *SIAM Journal on Optimization*, 2014.

[Caprara *et al.*, 2016] Alberto Caprara, Margarida Carvalho, Andrea Lodi, and Gerhard J Woeginger. Bilevel knapsack with interdiction constraints. *INFORMS Journal on Computing*, 2016.

[Durkota *et al.*, 2015] Karel Durkota, Viliam Lisỳ, Branislav Bosanskỳ, and Christopher Kiekintveld. Optimal network security hardening using attack graph games. In *IJCAI*, 2015.

[Farquhar, 2017] David Farquhar. *Watering hole attack prevention*, 2017. https://dfarq.homeip.net/watering-hole-attack-prevention/.

[Gilmore and Gomory, 1961] Paul C Gilmore and Ralph E Gomory. A linear programming approach to the cutting-stock problem. *Operations research*, 9(6):849–859, 1961.

[Jajodia *et al.*, 2017] Sushil Jajodia, Noseong Park, Fabio Pierazzi, Andrea Pugliese, Edoardo Serra, Gerardo I Simari, and VS Subrahmanian. A probabilistic logic of cyber deception. *IEEE Trans. Inf. Forensic Secur.*, 2017.

[Junger *et al.*, 2017] Marianne Junger, Lorena Montoya, and F-J Overink. Priming and warnings are not effective to prevent social engineering attacks. *Computers in human behavior*, 2017.

[Kellerer *et al.*, 2003] Hans Kellerer, Ulrich Pferschy, and David Pisinger. Knapsack problems. 2004, 2003.

[Kiekintveld *et al.*, 2015] Christopher Kiekintveld, Viliam Lisỳ, and Radek Píbil. Game-theoretic foundations for the strategic use of honeypots in network security. In *Cyber Warfare*, pages 81–101. Springer, 2015.

[Krombholz *et al.*, 2015] Katharina Krombholz, Heidelinde Hobel, Markus Huber, and Edgar Weippl. Advanced social engineering attacks. *Journal of Information Security and applications*, 22:113–122, 2015.

[Laszka *et al.*, 2015] Aron Laszka, Yevgeniy Vorobeychik, and Xenofon D Koutsoukos. Optimal personalized filtering against spear-phishing attacks. In *AAAI*, 2015.

[Letchford and Vorobeychik, 2013] Joshua Letchford and Yevgeniy Vorobeychik. Optimal interdiction of attack plans. In *AAMAS*, 2013.

[Mitnick and Simon, 2001] Kevin D Mitnick and William L Simon. The art of deception: Controlling the human element of security. 2001.

[Ogryczak and Tamir, 2003] Wlodzimierz Ogryczak and Arie Tamir. Minimizing the sum of the k largest functions in linear time. *Information Processing Letters*, 2003.

[Orgill *et al.*, 2004] Gregory L Orgill, Gordon W Romney, Michael G Bailey, and Paul M Orgill. The urgency for effective user privacy-education to counter social engineering attacks on secure computer systems. In *The 5th conference on Information technology education*, 2004.

[Parliament, 2018] Operation Parliament. *WATERING HOLE ATTACKS*, 2018. https://cyberprosconsulting.com/tag/watering-hole-attacks/.

[Pıbil *et al.*, 2012] Radek Pıbil, Viliam Lisỳ, Christopher Kiekintveld, Branislav Bošanskỳ, and Michal Pechoucek. Game theoretic model of strategic honeypot selection in computer networks. *Gamesec*, 7638:201–220, 2012.

[Rowe, 2007] Neil C Rowe. Deception in defense of computer systems from cyber attack. In *Cyber warfare and cyber terrorism*, pages 97–104. IGI Global, 2007.

[Schlenker *et al.*, 2018] Aaron Schlenker, Omkar Thakoor, Haifeng Xu, Milind Tambe, Phebe Vayanos, Fei Fang, Long Tran-Thanh, and Yevgeniy Vorobeychik. Deceiving cyber adversaries: A game theoretic approach. In *AAMAS*, 2018.

[Serra *et al.*, 2015] Edoardo Serra, Sushil Jajodia, Andrea Pugliese, Antonino Rullo, and VS Subrahmanian. Pareto-optimal adversarial defense of enterprise systems. *TISSEC*, 2015.

[Sutton, 2014] Michael Sutton. *How to protect against watering hole attacks*, 2014. https://www.networkworld.com/article/2451341/security0/how-to-protect-against-watering-hole-attacks.html.

[Symantec, 2017] Security Response Symantec. *Attackers target dozens of global banks with new malware*, 2017. https://www.symantec.com/connect/blogs/attackers-target-dozens-global-banks-new-malware-0.

[UN, 2015] UN. *17 Goals to Transform Our World*, 2015. https://www.un.org/sustainabledevelopment.

[Virvilis *et al.*, 2014] Nikos Virvilis, Bart Vanautgaerden, and Oscar Serrano Serrano. Changing the game: The art of deceiving sophisticated attackers. In *CyCon 2014*, 2014.

[Whittaker, 2013] Zach Whittaker. *Facebook, Apple hacks could affect anyone: Here's what you can do*, 2013. https://www.zdnet.com/article/facebook-apple-hacks-could-affect-anyone-heres-what-you-can-do/.

Appendix
Draining the Water Hole: Mitigating Social Engineering Attacks

A Deferred Algorithms

A.1 Attacker's Better Response Heuristic

In light of the hardness of finding the adversary's best response, we consider a greedy heuristic. Leveraging Theorem 2, GREEDY (Alg. 3) allocates the adversary's budget to websites in decreasing order of the ratio $r_w = \frac{t_w(1-x_w)/t_w^{all}}{\alpha_w}$, where α_w is a tuning parameter. In the iterative algorithm (Alg. 2) for $\mathcal{P}_1$, we replace the MILP for $\mathcal{P}_2(x)$ with Alg. 3 to find an adversary's better response. If it does not yield a new effort vector, the MILP is called. The iterative algorithm terminates if the MILP again does not find a new effort vector. We refer to this entire procedure as SED-IA GREEDY in the following section. Although Alg. 3 does not provide an approximation guarantee, it performs well in practice. As we show in the experiment section, in practice the accuracy of its solution improves as the size of the problem grows. We also considered a dynamic programming algorithm which is exact and runs in pseudo-polynomial time. However, its practical performance is unsatisfactory.

Algorithm 3: GREEDY

```
1 Sort the websites in decreasing order of
    $r_w = \frac{t_w(1-x_w)/t_w^{all}}{\alpha_w}$.
2 foreach website $w$ in the sorted order do
3     if remaining attack budget $\geq$ attack cost $\pi_w$ then
4         Attack this website $w$ with maximum effort
          allowed
5     if running out of budget then break
```

A.2 Enhanced Algorithm SED-ALL

Our enhanced algorithm SED-ALL for solving SED games is shown in Alg. 4. SED-ALL combines dominated website elimination, the optimality check of the LP relaxation, and the iterative algorithm SED-IA.

A.3 Baseline Algorithm for $\mathcal{P}_1$

We show the details of one of our baseline algorithms, All Actions, in Alg. 5. Let $\mathcal{A}$ denote the set of actions available to the adversary such that the budget constraint is satisfied. Each action $a^* \in \mathcal{A}$ is a set of websites being compromised. According to Theorem 2, among all the websites w compromised in a^*, the adversary puts "partial" effort $e_w \in (0, t_w^{all}$ on at most one website w^*. Therefore, the action-website pairs (a^*, w^*) fully characterize the adversary's strategies. Alg. 5 works by finding the optimal defender strategy, assuming each action-website pair is the optimal strategy for the adversary.

Algorithm 4: SED-ALL

```
1  Remove $D \leftarrow$ FIND-DOMINATED-WEBSITES() from $W$.
2  Get initial defender strategy $x$ by solving $\hat{\mathcal{P}}_1$.
3  Solve $\tilde{P}_3(x)$.
4  if $x$ satisfies optimality condition in Sec. 4.5 then return
5  Initialize max effort vector set $e^{\mathcal{A}} = \emptyset$.
6  while new max effort vector was added to $e^{\mathcal{A}}$ do
7      Solve $\mathcal{P}_2(x)$ for adversary's best response $e$
8      Add $e$ to $e^{\mathcal{A}}$.
9      Solve defender's optimal strategy problem w.r.t.
         effort set $e^{\mathcal{A}}$ as the LP in Equations (22)∼(25).
10     Update the game value and defender's strategy $x$.
```

B Deferred Proofs

B.1 Proof of Theorem 1

We reduce from the knapsack problem. In the knapsack problem, we have a set W of items each with a weight ω_w and value p_w $\forall w \in N$, and aim to pick items of maximum possible value subject to a capacity B. We now create an instance of the SED problem. Create a website for each item $w \in W$ with organization traffic and total traffic $t_w = t_w^{all} = p_w$ and attack cost ω_w. Assume that $x = \mathbf{0}^T$. Next, set $B_a = B$ and $B_e = \infty$. Notice that the objective function becomes $\sum_{w \in W} e_w$ where $\sum_{w \in W} e_w \leq \infty$ and $e_w \leq p_w y_w$. Hence, $e_w = p_w$ whenever $y_w = 1$. Then, the adversary's best response problem is given by:

$$\max_y \quad \sum_{w \in W} p_w y_w \tag{31}$$

$$\text{s.t.} \quad \sum_{w \in W} \omega_w y_w \leq B \tag{32}$$

$$y_w \in \{0,1\} \qquad \forall w \in W \tag{33}$$

This is exactly the knapsack problem described above. □

B.2 Proof of Theorem 2

For each $w \in W$, let $k_w = t_w(1 - x_w^*)/t_w^{all}$. Suppose there exist some $w_1, w_2 \in W_B$, and w.l.o.g assume $k_{w_1} \geq k_{w_2}$. Let $\Delta e = \min\{e_{w_2}^*, t_{w_1}^{all} - e_{w_1}^*\}$. Consider the solution $(x^*, y^*, \hat{e})$ where $\hat{e}_{w_1} = e_{w_1}^* + \Delta e$, $\hat{e}_{w_2} = e_{w_2}^* - \Delta e$, and $\hat{e}_w = e_w^*$ for all other websites $w \in W$. This is a feasible solution, and the objective increases by $(k_{w_1} - k_{w_2})\Delta e \geq 0$ compared to (x^*, y^*, e^*). Furthermore, at least one of w_1 and w_2 is removed from W_B. We can apply this argument repeatedly until $|W_B| \leq 1$. □

B.3 Proof of Corollary 1

Since $B_e \leq t_w^{all}$ $\forall w \in W$, we know $|W_F| \leq 1$ for any feasible solution. If $|W_F| = 1$, then we have $|W_Z| = n - 1$ and $|W_B| = 0$. If $|W_F| = 0$, by Theorem 2, we have $|W_B| = 1$ and $|W_Z| = n - 1$. In either case, there is only website w^* such that $e_{w^*} > 0$. It follows that $w^* \in \arg\max_{w \in W} \frac{t_w(1-x_w)B_e}{t_w^{all}}$ given a defender strategy x. The

Algorithm 5: ALL ACTIONS

```
1  foreach $(a^*, w^*) \in \mathcal{A} \times W$ where $w^* \in a^*$ do
2      foreach website $w \in W$ do
3          if $w = w^*$ then
4              Define
               $z_w = \min\{B_e - \sum_{w \in a^*, w \neq w^*} t_w^{all}, t_{w^*}^{all}\}$
5          else if $w \in a^*$ then
6              Define $z_w = t_w^{all}$
7          else
8              Define $z_w = 0$
9          Define $k_w = \frac{t_w}{t_w^{all}} z_w$
10         foreach $(\hat{a}, \hat{w}) \in \mathcal{A} \times W$ where $\hat{w} \in \hat{a}$ do
11             Define $\hat{k}_w$ similarly as above, for each
               $w \in W$.
12             Add to $BR(a^*, w^*)$ the following linear
               constraint
               $\sum_{w \in a^*} k_w(1 - x_w) \geq \sum_{w \in \hat{a}} \hat{k}_w(1 - x_w)$
13     Solve the following LP
```

$$\min_{x,v} \quad v \tag{26}$$

$$\text{s.t.} \quad v \geq \sum_{w \in W} k_w(1 - x_w) \tag{27}$$

$$\text{linear constraints in } BR(a^*, w^*) \tag{28}$$

$$\sum_{w \in W} c_w t_w x_w \leq B_d \tag{29}$$

$$x_w \in [0, 1], \qquad \forall w \in W \tag{30}$$

```
14 Select the best solution out of all the LPs.
```

optimal defender strategy can be found by solving the following LP.

$$\min_{x,v} \quad v \tag{34}$$

$$\text{s.t.} \quad v \geq \frac{t_w(1 - x_w)B_e}{t_w^{all}} \qquad \forall w \in W \tag{35}$$

$$\sum_{w \in W} c_w t_w x_w \leq B_d \tag{36}$$

$$x_w \in [0, 1] \qquad \forall w \in W \quad \square \tag{37}$$

B.4 Proof of Theorem 3

Under these assumptions, the problem $\mathcal{P}_1$ becomes

$$\min_x \max_{y,e} \quad \sum_{w \in W} t_w(1 - x_w)y_w \tag{38}$$

$$\text{s.t.} \quad \sum_{w \in W} y_w \leq B_a \tag{39}$$

$$\sum_{w \in W} c_w t_w x_w \leq B_d \tag{40}$$

$$x_w \in [0, 1], y_w \in \{0, 1\} \qquad \forall w \in W \tag{41}$$

The constraint $\sum_{w \in W} y_w \leq B_a$ must be satisfied with equality because $t_w(1 - x_w) \geq 0$ for all $w \in W$. The defender's problem is to minimize the sum of B_a largest linear functions $t_w - t_w x_w$ among the $n = |W|$ of them, subject to the polyhedral constraints on x_w. This problem can be solved as a single LP [Ogryczak and Tamir, 2003] as follows.

$$\min_{d^+,x,z} \quad B_a z + \sum_{w \in W} d_w^+ \tag{42}$$

$$\text{s.t.} \quad d_w^+ \geq t_w - t_w x_w - z \qquad \forall w \in W \tag{43}$$

$$\sum_{w \in W} c_w t_w x_w \leq B_d \tag{44}$$

$$x_w \in [0, 1], \, d_w^+ \geq 0 \qquad \forall w \in W \quad \square \tag{45}$$

B.5 Proof of Theorem 4

From conditions (1) and (2), we know that for the same amount of effort, the attacker will be better off attacking website u than w, regardless of the defender's strategy.

Suppose $e_w > 0$ and $e_u = 0$ (consequently $y_w = 1, y_u = 0$). Then we could let $e'_w = 0$ and $e'_u = e_w$. This is possible because from condition (4), $e_w \leq t_w^{all} \leq t_u^{all}$ so we have $e'_u \leq t_u^{all}$. Doing this does not increase the attack cost because now $y'_w = 0$ and $y'_u = 1$ and $\pi_w \geq \pi_u$ from condition (3).

Suppose $e_w > 0$ and $e_u > 0$ (consequently $y_w = y_u = 1$). Let $e'_w = e_w - \min\{e_w, t_u^{all} - e_u\}$ and $e'_u = e_u + \min\{e_w, t_u^{all} - e_u\}$. We know that if $e'_w > 0$, then $e'_u = t_u^{all}$. Of course, the attack cost does not increase as well. $\square$

B.6 Proof of Theorem 5

Let x^* be the optimal solution to $\mathcal{P}_1$. Consider the problem $\hat{\mathcal{P}}_2(x^*)$. At optimal solution, the inequality $e_w \leq t_w^{all} \cdot y_w$ in $\hat{\mathcal{P}}_2(x^*)$ is satisfied with equality, as if $e_w < t_w^{all} \cdot y_w$, then we can decrease y_w without changing the objective value and violating any constraints. Then, we can eliminate the variables e_w and $\hat{\mathcal{P}}_2(x^*)$ becomes a standard two-dimensional fractional knapsack problem $\hat{\mathcal{P}}_4(x^*)$. It is well-known that there exists an optimal solution to $\hat{\mathcal{P}}_4(x^*)$ which has at most 2 fractional values y_{w_1} and y_{w_2} [Kellerer *et al.*, 2003]. We have

$$\begin{aligned} &OPT(\hat{\mathcal{P}}_1) \leq OPT(\hat{\mathcal{P}}_2(x^*)) = OPT(\hat{\mathcal{P}}_4(x^*)) \\ &\leq OPT(\mathcal{P}_2(x^*)) + t_{w_1}(1 - x^*_{w_1}) + t_{w_2}(1 - x^*_{w_2}) \\ &\leq 3OPT(\mathcal{P}_2(x^*)) = 3OPT(\mathcal{P}_1) \end{aligned}$$

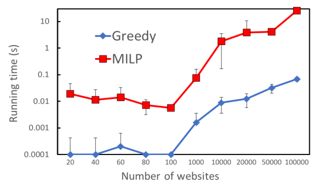
Note that if $B_e = \infty$, $\hat{\mathcal{P}}_1$ is a 2-approximation. $\square$

B.7 Proof of Theorem 6

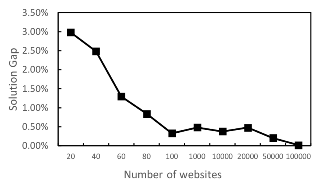
Since $\hat{x}^*$ and its best response calculated by $\mathcal{P}_2(\hat{x}^*)$ form a feasible solution to $\mathcal{P}_1$, the first inequality holds. For any defender strategy x, OPT$(\mathcal{P}_2(x)) \leq$ OPT$(\hat{\mathcal{P}}_2(x))$ as adversary can choose fractional y_w's in $\hat{\mathcal{P}}_2(x)$. For $\hat{x}^*$ specifically, we have OPT$(\hat{\mathcal{P}}_2(\hat{x}^*)) =$ OPT$(\hat{\mathcal{P}}_1)$, since $\hat{\mathcal{P}}_1$ is, by strong duality, equivalent to $\mathcal{P}_1$ except that the adversary is allowed to choose fractional y_w's. This establishes the second inequality. The last inequality holds because x^* and its fractional best response calculated by $\hat{\mathcal{P}}_2(x^*)$ form a feasible solution to $\hat{\mathcal{P}}_1$. $\square$

α_w	$\frac{\text{OPT}-\text{OPT}_{\text{Greedy}}}{\text{OPT}}$
π_w	0.0079
$\pi_w/B_a + 1/B_e$	0.0285
1	0.0082

Table 2: Solution gaps of different greedy heuristics for the adversary best response problem. Results are averaged over 5 runs on different problem sizes $|W| = 100, 200, \ldots, 500$.



(a) GREEDY running time



(b) GREEDY solution gap

B.8 Proof of Claim 1

To see the first condition, suppose $(\hat{x}^*, OPT(P_2(\hat{x}^*)))$ is not an optimal solution for the LP in Equations (22)-(25), thus equivalently $\hat{x}^*$ not optimal for $\mathcal{P}_1$. Any of its neighborhood with radius ϵ contains some $(\hat{x}', v')$ as a better solution, meaning $v' < OPT(P_2(\hat{x}^*))$. This solution $(\hat{x}', v')$ satisfies constraint (23), which is strictly stronger than constraint (18). Therefore $(\hat{x}', v')$ is feasible for $\tilde{P}_3(\hat{x}^*)$; this contradicts $OPT(\tilde{P}_3(\hat{x}^*)) \geq OPT(P_2(\hat{x}^*))$. For the second condition, note that x' is a better solution than $\hat{x}^*$. □

B.9 Proof of Claim 2

SED-IA terminates when no new effort vectors are found for the adversary. Suppose x is the optimal solution to the defender's optimization problem from the previous iteration (Line 6, Alg. 2), and suppose now $\mathcal{P}_2(x)$ does not find a new effort vector (Line 4, Alg. 2). This implies x would still be feasible for the LP in Equations (22)∼(25) even if $e^{\mathcal{A}}$ is replaced by the set of all max effort vectors $\hat{e}^{\mathcal{A}}$. Thus, x is an optimal solution. Indeed, at this point the optimal values of the LP and $\mathcal{P}_2(x)$ are equal. □

C Deferred Experiments

We present additional experiments on the adversary's best response problem. In the GREEDY algorithm (Alg. 3), the adversary selects websites based on a decreasing order of $r_w = \frac{t_w(1-x_w)/t_w^{all}}{\alpha_w}$. Here, α_w is the tuning parameter. With different choices of α_w, we compare the output value $\text{OPT}_{\text{Greedy}}$ of GREEDY with the optimal value OPT obtained by solving the MILP $\mathcal{P}_2(x)$. Table 2 shows the solution gap $\frac{\text{OPT}-\text{OPT}_{\text{Greedy}}}{\text{OPT}}$. We observe that $\alpha_w = \pi_w$ yields the smallest solution gap. We also tested other choices for α_w such as $(\pi_w/B_a)^p + (1/B_e)^q$ for different powers p and q, yet they do not yield better optimization gaps. Hence we fix $r_w = \frac{t_w(1-x_w)/t_w^{all}}{\pi_w}$ in subsequent experiments.

Fig. 4b shows GREEDY's solution gap decreases to near zero as the problem size grows. In addition, GREEDY typically runs within 1% of the time of the MILP.

D Experiment Parameters

Table 3 shows the distribution from which the parameters are generated in most of our experiments. In Table 4, we detail the parameters used in the experiment in Fig. 2e.

Variable	Distribution
t_w^{all}	$U(350, 750)$
t_w	$U(50, 100)$
c_w	$U(1, 4)$
π_w	$U(30, 54)$
B_d	$U(0.11 \sum_{w\in W} c_w t_w, 0.71 \sum_{w\in W} c_w t_w)$
B_a	$U(0.1 \sum_{w\in W} \pi_w, 0.8 \sum_{w\in W} \pi_w)$
B_e	$U(0.2 \sum_{w\in W} t_w^{all}, 0.8 \sum_{w\in W} t_w^{all})$

Table 3: Parameter distribution

For $w \in W_1$		For $w \in W_2$	
Variable	Distribution	Variable	Distribution
t_w^{all}	$U(60, 110)$	t_w^{all}	$U(20, 70)$
t_w	$U(45, 55)$	t_w	$U(3, 8)$
c_w	$U(2, 6)$	c_w	$U(1, 3)$
π_w	3	π_w	3
B_d	$U(0, 10 \sum_{w\in W} c_w t_w/\|W\|)$		
B_a	$U(0.1 \sum_{w\in W} \pi_w, 0.8 \sum_{w\in W} \pi_w)$		
B_e	$U(0, 3 \sum_{w\in W} t_w^{all}/\|W\|)$		

Table 4: Parameter distributions for the experiment on large instances.