Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Innovative Applications of O.R.

Scalable min-max multi-objective cyber-security optimisation over probabilistic attack graphs

MHR. Khouzani, Zhengliang Liu[*], Pasquale Malacaria

School of Electronic Engineering and Computer Science at Queen Mary University of London (QMUL), Mile End Rd, London E1 4NS, UK

A R T I C L E I N F O

Article history:
Received 10 October 2018
Accepted 23 April 2019
Available online 29 April 2019

Keywords:
Computing science
Multi-objective optimisation
Cyber-security
Probabilistic attack graph

A B S T R A C T

We present a framework to efficiently solve a multi-objective optimisation problem for cyber-security defence. Facing an attacker who can mount a multi-stage attack (modelled using attack graphs), the defence problem is to select a portfolio of security controls which minimises the security risk and the (direct and indirect) costs of the portfolio of controls. The main challenges for the optimisation are: (a) the effect of the security controls is in general probabilistic, for example, the effect of staff anti-phishing training; moreover, some controls like taking regular back-ups do not have an attack-preventing effect, but rather, mitigate the losses of a successful attack; (b) each control may affect multiple vulnerabilities; and each vulnerability may be affected by multiple controls; (c) there can be a prohibitively large number of attack paths, each involving exploitation of different vulnerabilities. Our mathematical framework deals with all these problems. In particular, we model the problem as a min-max multi-objective optimisation. Using techniques such as ILP conversion, exact LP relaxation and dualisation, we convert the problem into a very efficient MILP. For instance, it returns the optimal solution for attack graphs with 20,000 nodes in less than four minutes typically.

1. Introduction

Cyber-security is a concern for any organisation that has IT-related assets or uses IT services. Organisations can improve their overall security state by adopting a cyber-security *plan* (or portfolio). A (cyber-security) plan in part specifies a subset of security counter-measures, which we will refer to as *security controls*, along with the details of how they should be implemented.

The choice of a cyber-security plan is not an easy task, as there are many security controls to chose from, and each can provide some defence against exploitation of some potentially overlapping *vulnerabilities*. Examples of security controls include (The National Cyber Security Centre, 2018): managing the inventory of authorised and unauthorised devices and software, identity management and access control, employing a password policy, patch management, setting up network and/or application firewalls, installing anti-malware and anti-phishing software, training of staff against social engineering attacks, data back-up and resource redundancies, physical security, etc.

For each cyber-security control, there can be different intensity levels of implementation. For instance, the security control of patch management can be implemented at one of the following levels, with increasing intensity: 1: `patch once`, 2: `patch monthly`, 3: `weekly`, 4: `daily`, 5: `upon release`. Similarly, the security control of deploying a network firewall can be implemented at one of the following levels: 1: `packet filter`, 2: `circuit-level gateway`, 3: `stateful inspection firewall`, 4: `application proxy firewall with deep-packet-inspection (DPI)`. The choice of controls and their intensity can even be specific to a physical or logical part of an organisation and vary among them.

Choosing a cyber-security plan is a multi-criteria decision problem (Dewri, Poolsappasit, Ray, & Whitley, 2007; Noel, Jajodia, O'Berry, & Jacobs, 2003; Viduto, Maple, Huang, & LóPez-PeréZ, 2012; Wang, Noel, & Jajodia, 2006): the improvement in the overall security state of an organisation needs to be balanced with the *direct* and *indirect* costs of the plan. The direct costs are represented in monetary fees: this includes both the capital expenditure (capex) for obtaining and installing a security control, and the operational expenditure (opex) for its maintenance, involving IT/security staff salaries, license renewal, etc. The indirect costs, on the other hand, are those that cannot be clearly represented in a monetary value, and are rather associated with the negative im-

[*] Corresponding author.
E-mail addresses: arman.khouzani@qmul.ac.uk (MHR. Khouzani), zhengliang.liu@qmul.ac.uk, ethan_liu@ymail.com, p.malacaria@qmul.ac.uk (Z. Liu), p.malacaria@qmul.ac.uk (P. Malacaria).

https://doi.org/10.1016/j.ejor.2019.04.035

pact of the controls on the normal operation of an organisation. For instance, the downtime of a database server for applying a patch, or the call to help desk by employees who forget their passwords, or the slowdown of network caused by an application proxy. A decision problem with multiple criteria of distinct natures is usually approached using multi-objective optimisation whose solution concept is the Pareto-optimality, and the Pareto-front, as the set of Pareto-optimal solutions (Ehrgott, 2005). The Pareto-front allows the administrators to investigate the best possible trade-offs and subjectively choose among them, with the guarantee that all the options for which all objectives can be simultaneously improved are excluded.

The defensive mechanism of security controls can be roughly classified in three different categories (Franke, 2016): blocking an exploitation by removing the underlying vulnerability or tightening the access control; detecting an exploitation attempt so that it can be thwarted; or limiting the damages in case of a successful attack and recovering from it. Irrespective of the mechanism, the counteracting effect of a security control is not necessarily absolute in general, but rather, probabilistic or partial, as the following examples argue:

- An attack may be able to evade an intrusion detection and protection system (IDPS). This is even often part of the design of an IDPS in order to cap the rate of false positives.
- Even though training of the staff against social engineering attacks (for example, phishing, baiting, etc) or physical attacks (tailgating, protecting and reporting their devices against loss), decreases the chances of them falling victim to such attacks, it does not guarantee the removal of these vulnerabilities.
- Even in the case of installing security patches, the effect can be interpreted as probabilistic: although the patch may remove a specific vulnerability, it only curtails its window of exposure: There is a non-zero probability that the vulnerability is exploited as a zero-day attack (before discovered by whitehats), or after the patch is released but not installed yet. This probability is specially higher in the latter window, given the reverse-engineering and automatic-exploit development tools of black-hats, Frei, May, Fiedler, and Plattner (2006).
- A recovery scheme such as taking back-ups can recover some but not all of the corrupted data (depending on the regularity of the back-ups) and cannot deal with surreptitious stealing of sensitive information.

Attack graphs in general (Homer & Ou, 2009; Kordy, Piètre-Cambacédès, & Schweitzer, 2014; Ou, Boyer, & McQueen, 2006; Roschke, Cheng, Schuppenies, & Meinel, 2009) and probabilistic attack graphs in particular (Poolsappasit, Dewri, & Ray, 2012; Sommestad, Ekstedt, & Holm, 2013; Wang, Islam, Long, Singhal, & Jajodia, 2008) have been used to model and analyse the overall security state of an information system. An attack graph is an abstraction of a real world cyber system. In its most basic form, nodes of an attack tree represent different security states, and a directed edge represents a potential transition in the security state as a result of an attacker's exploitation. An attacker penetrates the system by choosing a sequence of exploitation actions, which correspond to a path from source nodes (initial access privileges of an attacker) to target nodes (access to critical assets) in the attack graph.

The security plan should provide a well-rounded security by covering the most critical vulnerabilities most effectively. This motivates the need for a sound measure of security risk. A suitable candidate is the highest expected losses coming from the most effective attack paths in the probabilistic attack graph. A fundamental challenge is that changing the security plan affects the entire attack graph and, in particular, can affect the critical attack paths in a non-trivial way. This non-triviality is partly due to the fact that each security control can affect multiple vulnerabilities, and hence multiple paths; and each vulnerability can be affected by more than one security control.

It has been suggested that this cybersecurity problem can be modelled as a Stackelberg game (Simaan & Cruz, 1973). In particular, the defender (the leader) chooses a security plan seeking to minimise its security risk, while the attacker (the follower) intends to maximise it via the most effective attack path. We model this as a min-max optimisation problem, where the inner (maximisation) problem is the attacker's, and the outer (minimisation) problem is the defender's, keeping in mind the reaction of the attacker.

Solving this min-max optimisation can be particularly challenging, since the representation of risk is non-linear and the number of attack paths can exponentially grow with the size of the attack graph. Moreover, an approach that iteratively identifies the most effective attack paths and chooses the best defences accordingly, besides being computationally burdensome, does not guarantee to provide an optimal solution.

This paper main *contribution* is to provide a series of "exact" conversions that transform the defender's problem into a highly efficient mixed integer linear programming (MILP), allowing for the efficient solution of the optimisation in large probabilistic attack graphs. In particular, first we show that the attacker's non-linear integer programming problem can be re-cast exactly as a linear program (LP). Next, using the strong duality of LPs, we convert this inner LP maximisation to its minimisation dual LP, and combine it with the minimisation problem of the defender to yield a MILP, which our numerical evaluations show to be highly efficient.

2. Related work

Perhaps the closest work related to this paper is Almohri, Watson, Yao, and Ou (2016). We share a similar probabilistic attack graph model and min-max formulation. However, our methodology is quite different. The method in Almohri et al. (2016) provides an approximate solution for the inner optimisation problem based on Taylor expansion and sequential linear programming. Unlike the approximate approach we convert the non-linear min-max integer optimisation to a MILP that efficiently solves the problem. This reformulation method allows us to solve attack graphs of much larger scale. Khouzani, Malacaria, Hankin, Fielder, and Smeraldi (2016) and Fielder, Panaousis, Malacaria, Hankin, and Smeraldi (2016) propose similar models to this study in terms of multiple objective functions, security risk, indirect and direct costs, etc. However, those models only consider single-step attacks rather than a sequence of attack steps. Another "single-step" approach using a MILP linearisation for optimal defence is presented in Schilling and Werners (2016) and subsequently extended with robustness and sensitivity analysis in Schilling (2017). The "reduction coefficient of the selected safeguard" used in these works can be used in many contexts to estimate the p_{ecl} used in this paper. Other well known "single-step" approaches are Viduto et al. (2012) which uses genetic optimisation for optimal trade-off between financial costs and risk and Sawik (2013) which use financial tools like conditional value at risk, and Rakes, Deane, and Rees (2012), which considers a model in which the threat rates are external parameters that do not change with respect to the security profile of an organisation.

Zhang, Albert, Luedtke, and Towle (2017) also adopt the attack graph structure and propose new coverage models for selecting optimal portfolio of security controls to reduce threat vulnerability. A polynomial-time heuristics and a Benders branch and cut algorithm are proposed, which provide near-optimal solutions and exact solutions of large scale efficiently. Zheng and Albert (2019) extend the work by considering three different robustness models providing insights into modelling uncertainty. These models assume that the attack paths are enumerated and given as a complete list. In contrast we relax this assumption allowing the attacker to exploit the best attack path to its interest among a large number of alternatives. Similar to our work, Zhang and Albert (2018) propose a new Stackelberg game framework for cyber-security planning. In it the defender's object is to maximally delay the attacks, whereas the attacker seeks the fastest completion time of an attack. In contrast, we assume that the attacker selects an attack path with the highest probability of success.

An interactive approach is proposed in Qian, Mao, Rayes, and Jaffe (2012), which iterates between the defender's and the attacker's problem until all vulnerabilities are protected. However, this may result in sub-optimal equilibria. Game theoretical approaches have been proposed also by Cavusoglu, Raghunathan, and Yue (2008) and Baykal-Guersoy, Duan, Poor, and Garnaev (2014) to determine an ideal investment level and by Panaousis, Fielder, Malacaria, Hankin, and Smeraldi (2014) in the context of "one step" cyber-security investment. An "online" counter-measure selection scheme is proposed in Chung, Khatkar, Xing, Lee, and Huang (2013), in which a counter-measure is selected once an attack is detected and the threat is close enough to the critical target nodes. Another work in a similar spirit is Zonouz, Khurana, Sanders, and Yardley (2009). They propose an automatic intrusion response and recovery system for cyber attacks, in which the attacker and the response engine are modelled as a follower and a leader in a sequential Stackelberg stochastic game. These approaches require monitoring and analysing the cyber system at all times, and is rather perpendicular to the objective of selecting a well-rounded cyber-security plan which may include intrusion detection and prevention as a constituent.

In this study we formulate the security selection problem as a multi-objective min-max optimisation problem. Our formulation is close to a category known as "network interdiction problems", which has been a subject of research for decades. These address many practical resource allocation problems, including combating drug smuggling (Wood, 1993), flood management (Ratliff, Sicilia, & Lubore, 1975) and more recent applications such as enhancement of drone delivery systems (Sanjab, Saad, & Basar, 2017), infection control (Nandi & Medal, 2016) and cyber attack mitigation (Bhuiyan, Nandi, Medal, & Halappanavar, 2016; Nandi, Medal, & Vadlamani, 2016; Nguyen & Sharkey, 2017). The network interdiction problem includes several forms such as maximum flow interdiction (Akgün, Tansel, & Wood, 2011), shortest path interdiction (Israeli & Wood, 2002), and minimum cost flow interdiction (Zenklusen, 2010). Most of the previous work consider complete edge interdiction, in which an edge is completely removed (Nandi et al., 2016; Nguyen & Sharkey, 2017). However, in practice cyber attacks can only be mitigated to some extent instead of being completely blocked. In this paper we specify the effectiveness of a control by means of the probability of intrusion success. Specifically, the higher the effectiveness, the less chance that the attacker is likely to pass through the edge successfully.

The complexity of interdiction problems has been studied by many researchers. The directed shortest path interdiction is strongly NP-complete on general graphs, (Malik, Mittal, & Gupta, 1989). The maximum flow interdiction problem is strongly NP-hard on general graphs (Wood, 1993). Altner, Ergun, and Uhan (2010) reduced a simple form of the problem from the clique problem. The results proposed by Pan and Schild (2016) show that the interdiction problem on planar graphs is strongly NP-complete. Specifically, the interdiction problem in this study is the shortest path maximisation problem, which is an NP-complete problem, (Israeli & Wood, 2002).

Fortunately, this problem can be solved by the dualisation technique (Wood, 1993). This approach replaces the inner minimisation problem with its dual problem and merges it with the outer maximisation problem. The reformulation leads to a MILP, which can be solved efficiently by many existing solvers such as `CPLEX` and `Gurobi`.

Some of the previous work involve probability and randomness. A similar probability network interdiction is considered by Washburn and Wood (1995). This model assigns a probability to each edge indicating the successful detection rate if an attack is inspected on that edge. However, in this model only one edge is selected for inspection, whereas in our model a set of edges is chosen in terms of control selection.

A more detailed review on network interdiction can be found in Smith and Lim (2008).

3. Modelling and notations

We start by defining our measure for estimating the overall security risk (Section 3.1) using probabilistic attack graphs: we take the expected damage caused by the *most effective (attack) paths* in the probabilistic attack graph as a measure of overall security risk, and pose it as the "Attack" problem.

Next, we describe the "Defence" problem as a multi-objective optimisation for choosing security controls that minimise the overall security risk, and the overall direct and indirect costs, and overview Pareto-optimality as the solution concept.

3.1. Security risk with probabilistic attack graphs: the attack problem

We use attack graphs to model the cyber-security risk of an organisation. The vertices (nodes) in the attack graph represent the "privilege states" of an attacker in the organisation's IT infrastructure. The edges between them represent the exploitation of a vulnerability that allows the attacker to change (create/escalate) its privilege state. Each edge is associated with the success probability of the exploitation attempt corresponding to that edge.

For simplicity, we combine all the potential attackers into one monolithic entity called *the attacker*. The attacker starts from the vertex labelled as *source*, which represents his initial privilege state. For instance, this can be an "outside" attacker with no a priori privilege on the internal IT infrastructure of the organisation. Such an attacker can only access publicly available services, such as the Internet-facing website or mail-server of the organisation as well as the publicly available contact and potentially social network information of the staff. In each privilege state, i.e., each vertex, the attacker can take one of the available atomic attack actions, corresponding to a directed edge out of that vertex. All of such edges share the same head, but may have different tails.

We thus define the probabilistic attack graph $G = (V, E, h, t, p, s, T)$ as a directed multi-graph where:

- V: is the set of vertices (or nodes); each vertex can be thought of as the privilege state of an attacker in the organisation.
- E: is the multi-set of directed edges. Note that there can be multiple edges between two vertices, corresponding to different atomic attacks between two attackers' privilege states. Equivalently, an edge e can be represented by the ordered triplet $e = (i, j, k)$, where i and j are the *tail* and *head* of the edge, and k is its index among all such edges that go from i to j;
- h: $E \to V$: returns the vertex that is the head of the edge, i.e., $h(i, j, k) = j$;
- t: $E \to V$: returns the vertex that is the tail of the edge, i.e., $t(i, j, k) = i$;
- p: $E \to (0, 1]$: represents the conditional success probability of the attack taking the (atomic) attack step corresponding to an edge given that it has successfully reached the head of that edge. That is, if an attacker is already at the privilege

state i and wants to get to privilege state j by taking the atomic attack action e, where $j = h(e)$ and $i = t(e)$, its success probability will be p_e. These success probabilities depend on the choice of the security counter-measures, as we formalise in Section 3.2. For now, consider p_e's to be given.

$s \in V$: one of the vertices labelled as *source*, specifying the initial privilege state of an attacker;

$T \subset V$: a subset of the vertices labelled as *targets* (or sink vertices). These are the privilege states that constitute the potential goal of an attacker.

Note that consecutive nodes can be either tied to a single device (for example, related to a local privilege escalation using a buffer overflow) or related to two different connected entities (for example, using brute-force password attack on an ssh service) or may not even be tied to any physical nodes (for example, when an attacker has found a password but needs a text-message to complete a two-factor authentication). Also, the probabilities on each edge represent the likelihood that exploitation attempts corresponding to that edge are *ever* successful. This is in accordance with the "belief" interpretation of these probabilities as done for example, in Almohri et al. (2016) and Singhal and Ou (2017).

For simplicity of exposition, for now, we assume that there is only a single target node, i.e., $T = \{t\}$. In Section 5.2, we relax this assumption. A path $\omega_{s\to t}$ is a sequence of directed edges $(e_1, \dots, e_j)$ starting from s and ending at t, i.e., $t(e_1) = s$, $h(e_i) = t(e_{i+1})$ and $h(e_j) = t$. A path describes an attack scenario: an attacker in the starting privilege state s, takes the sequence of atomic attack actions corresponding to the edges in that path, and reaches the target privilege state t. If we assume that the successes of the attack steps are independent from each other, then the overall success of an attack represented by the attack path $\omega_{s\to t}$ can be computed as: $\prod_{e\in\omega_{s\to t}} p_e$.

Then a measure of the security risk, which we denote by R, can be defined as the highest success probability of attack across all possible paths. We will call an attack path that yields this value as the *most effective attack path*. The problem of estimating the security risk can be then cast as the following optimisation, which we will refer to as the *Attack* (or the *inner*) problem:

$$R := \max_{\omega_{s\to t}} \prod_{e\in\omega_{s\to t}} p_e, \qquad (1)$$

where the optimisation is taken with respect to all the attack paths from the source to the target.

3.2. The defence problem as a multi-objective min-max optimisation

The organisation seeks to reduce its cyber-security risk, R, by choosing a cyber-security plan. A plan specifies which counter-measures, i.e., *controls*, should be selected and how each should be implemented. Let $\mathcal{C}$ represent the set of all available controls. Note that a control can be at the level of the entire organisation, for example, training the staff against social engineering attacks, or be a particular device or set of devices, for instance, the placement and rule-set of a network firewall. In the latter case, the same control can be applied to multiple places in the organisation. For instance, different network firewalls can be potentially implemented in different places, each with its own level of strictness. In our model, these are labelled as distinct controls.

Recall that a control c can be potentially implemented at different intensity levels. Let the set of available implementation levels for control c be denoted by $\mathcal{L}(c)$. Using this notation, a plan can be expressed using indicators x_{cl} satisfying:

$$x_{cl} \in \{0,1\}\ \forall c \in \mathcal{C},\ \forall l \in \mathcal{L}(c), \qquad \sum_{l\in\mathcal{L}(c)} x_{cl} \le 1, \forall c \in \mathcal{C}. \qquad (2)$$

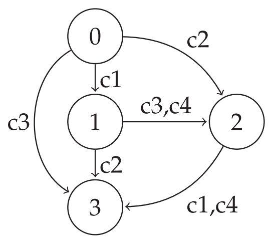


Fig. 1. Atoy example.

The interpretation of the decision variables is as follows: $x_{cl} = 1$ if control c is selected to be implemented at level l, and is 0 otherwise. The second inequality follows the logical requirement that a control can be implemented in at most one of its levels. It is allowed that a control may not be selected at all if $x_{cl} = 0$ for all $l \in \mathcal{L}(c)$.

Each control is an actionable security counter-measure that can affect a certain set of (atomic) attacks. A control may be effective on multiple edges and an edge may be affected by multiple controls. We denote the subset of controls that affect an edge e by $\mathcal{C}(e)$.

Even in the absence of any defensive mechanism, the probability of a successful exploitation corresponding to an edge may not be 1. This is for instance related to how easy it is to exploit the underlying vulnerability, i.e., its "exploitability" score, which can be obtained from its CVSS, as explored for example, in Frigault, Wang, Singhal, and Jajodia (2008) and Singhal and Ou (2017). To incorporate this in our model, for each edge e, we let π_e be the "baseline" probability of success of the attack step associated with that edge. Implementation of security controls can reduce this success probability depending on their efficacies. However, since multiple controls can affect an edge, careful consideration should be taken regarding how their efficacies are "combined". A reasonable candidate can be as follows: Suppose control c implemented at level l can block exploitation attempts associated with edge e with (conditional) probability $p_{ecl} \in (0, 1]$, allowing the effect of a control to be edge-dependent to keep generality. Then one can assume that exploitation attempts associated with edge e, should "defeat" each of the controls that affect that edge to succeed. If these events are considered independent, then the overall (conditional) success probability of exploitation attempts on edge e, which we will denote by p_e, can be expressed as:

$$p_e(x) = \pi_e \prod_{\substack{c\in\mathcal{C}(e),\\ l\in\mathcal{L}(c)}} (p_{ecl}x_{cl} + (1 - x_{cl})). \qquad (3)$$

To interpret this relation, observe that if $x_{cl} = 0$, i.e., control c at level l is not part of the defender's plan, then the corresponding term in the product is 1, and if $x_{cl} = 1$, the corresponding term is p_{ecl}. Wherever not ambiguous, we will avoid using $p_e(x)$ and make the dependence on $x = (x_{cl})$ implicit.

The coefficients p_{ecl} can be estimated for example using similar surveyed data sources as for the "effectiveness coefficient of a safeguard" in Schilling and Werners (2016).

Example: To clarify the basics of our setup, let us consider the toy example in Fig. 1. The initial state of the attacker is vertex 0 and the target state is vertex 3. Associated with each edge are security controls which are effective against that attack step. For simplicity, assume that each control has only one level of implementation; also, $\pi_e = 1$ for all edges. Suppose that when c1 is deployed, the attacker probability of success on that edge is 0.5, for c2 and c4 is 0.2, and for c3 is 0.1. Suppose the defender can only choose two security controls Then the optimal choice is c2 and c3, which limits the probability of reaching 3 to 0.2.

3.3. Justifying independence

The assumption that controls "compose independently" which leads to the multiplicative form in (3) deserves some justification, as it plays a critical role in our method in Section 4. We provide some supporting arguments:

(A) Independence is mostly justifiable in this context: Different security controls typically use distinct defensive "mechanisms", based on distinct operations or features. For instance, the defensive mechanism of patching is by removing the potential vulnerabilities in an application, while the defensive effect of a network firewall is based on restricting outside access to a resource. So if an attack involves exploiting an application vulnerability, it first needs to access it by bypassing the firewall rules. The assumption of independence here is natural given the distinct nature in operations and mechanisms involved. Also notice that even when the "same" control appears several times along a path, this still usually represents "different" instances of that control, each deployed at different "depths" of the organisation. For example, the first occurrence of the control "firewall" would represent the one facing the external world, while the second occurrence could be the firewall separating the "demilitarized zone" from the the intranet, etc. Each of these firewalls will have their distinct rules: an attacker that has successfully bypassed the first rules, still needs to bypass the second.

(B) The model can be extended to partial relaxation of the independence assumption: For security controls with high degree of correlation (positive, or negative), the model can be extended to incorporate them (at least partially), as we describe next. Suppose, e.g., that controls A, B are deemed highly correlated. Then the analyst could add a new control labelled $\langle A, B\rangle$ to the model, which represents control A and B acting together when both are implemented. Then $\langle A, B\rangle$ will have its own effectiveness (which could be different from the multiplicative form). We then only need to add a linear constraint allowing only one choice among A, B and $\langle A, B\rangle$, i.e., $x_A + x_B + x_{\langle A,B\rangle} \le 1$. The optimisation would then choose $\langle A, B\rangle$ if appropriate.

Another possibility, sometimes more reasonable, is to have one control to appear as a higher implementation level of another control; For instance, "intrusion prevention system (IPS)" as a higher implementation level of "intrusion detection system (IDS)".

(C) It is consistent with diminishing marginal utility of security investments As pointed out in related works such as Schilling and Werners (2016), Sawik (2013), and Rakes et al. (2012) where a similar model is adopted, the multiplicative relation captures the expected property that implementation of more security controls improve the overall (blocking) efficacy, but the rate of improvement slows down.

3.4. Multi-objective optimisation & Pareto-front

The organisation faces two main types of costs, namely, direct costs and indirect costs. The direct cost of a control is its "implementation cost" i.e. the cost of the software/hardware that constitute the control and its maintenance over a fixed interval of time. The indirect cost of a control is a measure of the negative impact on the organisation of deploying that control at that level. For example think of a password policy, as the password policy is stricter (for example, change password once a month vs no need to change password ever), more staff may require help desk support and may lose working time: these are the indirect costs. Let

$$D(x) = \sum_{c\in\mathcal{C}, l\in\mathcal{L}(c)} x_{cl}\texttt{Cost}_{c,l}, \qquad I(x) = \sum_{c\in\mathcal{C}, l\in\mathcal{L}(c)} x_{cl}\texttt{IndirectCost}_{c,l}$$

denote the direct cost and indirect cost of a security plan x, respectively. Then, we propose the following multi-objective optimisation model.

$$\min_{x \text{ s.t. } (2)} (R(x), D(x), I(x)), \qquad \text{where} \qquad R(x) = \max_{\omega_{s\to t}} \prod_{e\in\omega_{s\to t}} p_e(x)$$

Let us briefly review some basic notions of multi-objective optimisation. Consider the *multi-objective optimisation problem*

$$\min_{\vec{x}\in\mathcal{X}} (F_1(\vec{x}), \dots, F_N(\vec{x})),$$

where $N \ge 2$. Then a solution $\vec{x}_1 \in \mathcal{X}$ is said to *Pareto-dominate* another solution $\vec{x}_2 \in \mathcal{X}$ if and only if (iff) $\forall i \le N$, $F_i(\vec{x}_1) \le F_i(\vec{x}_2)$ and $\exists j \le N$ for which $F_j(\vec{x}_1) < F_j(\vec{x}_2)$. A solution $\vec{x} \in \mathcal{X}$ is *Pareto-optimal* (*Pareto-efficient*) iff it is not Pareto-dominated by any other solution. The set of all Pareto-optimal solutions is the *Pareto-front(ier).*

There are several techniques to compute Pareto-fronts including ϵ-constraint and weighting methods (Chinchuluun and Pardalos, 2007, Section 6). We have chosen the ϵ-constraint method, where one of the objective is left as a single objective optimisation and the other objectives are transformed as constraints. The Pareto front is then built by iteratively increasing the constraints' bounds by a small amount in each step.

This leads to the following series of single-objective optimisations, which we refer to as the Defence problem:

$$\begin{aligned} & \min_x \max_{\omega_{s\to t}} \textstyle\prod_{e\in\omega_{s\to t}} p_e(x) \\ \text{s.t.} \quad & D(x) \le B_D, \quad I(x) \le B_I \\ & \sum_{l\in\mathcal{L}(c)} x_{cl} \le 1, \forall c \in \mathcal{C}, \\ & x_{cl} \in \{0,1\}, \ \forall c \in \mathcal{C}, \forall l \in \mathcal{L}(c). \end{aligned} \tag{4}$$

In particular, the values of B_D (direct cost budget) and B_I (indirect cost budget) are swept from 0 to a large value, with steps equal to the minimum difference between direct (resp. indirect) costs of the controls. Each "new" solution belongs to the Pareto-front.

4. Methodology and solution method

4.1. Solving the attack problem

The Attack problem (1) is to find an attack path with the highest probability of success among all probabilistic paths. This may appear as a contrived problem, since the objective function is highly non-linear. However, this problem can be *exactly* converted to an equivalent simple Linear Programming (LP).

The first lemma shows how we can change the optimisation variable from $s \to t$ paths to new variables defined only on edges $(y_e)_{e\in E}$: for each edge e, y_e represents whether the attacker should choose that edge as part of his optimal attack path.

Lemma 1. *Introducing binary variables y_e for each edge $e \in E$, the Attack problem (1) can be equivalently expressed as:*

$$\max_y \prod_{e\in E} (y_e p_e + 1 - y_e) \tag{5}$$

$$\text{s.t.} \quad y_e \in \{0,1\} \ \forall e \in E,$$

$$\sum_{e:h(e)=i} y_e - \sum_{e:t(e)=i} y_e = \begin{cases} 1 & i = t \\ -1 & i = s \\ 0 & \forall i \in N \setminus \{s,t\} \end{cases}$$

Proof. The constraint enforces the following: for each intermediate vertex, the number of exiting edges whose $y_e = 1$ should be equal to the number of entering edges whose $y_e = 1$. For the source (target, resp.), the number of exiting edges whose $y_e = 1$ must be 1 more (less, resp.) than the entering edges whose $y_e = 1$. Now, either there is no $s \to t$ path in the attack graph, in which case both problems are infeasible, or this forces the edges along an $s \to t$ path to have $y_e = 1$ and all other edges to have $y_e = 0$.

Hence, in the maximisation (5) the only y's equal to 1 will be edges along the path maximising (5). All other y's will have value 0 and hence for those terms $(y_e p_e + 1 - y_e) = 1$. Hence (5) is equivalent to the Attack problem (1). □

Note that in both problems, the path does not have to be "simple", i.e., it can contain loops, however, the optimisations will never pick a path with a loop, as it yields a lower utility.

The problem in (5) is still a non-linear integer programming due to the product form in the objective function. However, since $\log(x)$ is strictly monotone for $x > 0$, we can equivalently maximise the log of the objective, which converts the product to a sum:

$$\log \prod_{e\in E} (y_e p_e + (1 - y_e)) = \sum_{e\in E} \log (y_e p_e + (1 - y_e)) \tag{6}$$

Noting that the y_e's can only take 0 or 1 values, this can be further simplified to the linear form $\Sigma_{e\in E} y_e \log p_e$. This is because if $y_e = 0$, then $\log (y_e p_e + (1 - y_e)) = \log(1) = 0$, and if $y_e = 1$, then $\log (y_e p_e + (1 - y_e)) = \log p_e$. Hence:

Lemma 2. *The problem in (5) is equivalent to the (binary) integer linear programming (ILP) of* $\max_y \Sigma_{e\in E} y_e \log p_e$ *subject to the same constraints.*

The ILP in Lemma 2 can be relaxed to a LP. Crucially, this relaxation is *exact*. That is:

Proposition 1. *The Attack problem in (1) is equivalent to the following linear program (LP):*

$$\begin{aligned} &\max_y \ \sum_{e\in E} y_e \log(p_e) \\ &\text{s.t.} \ \ y_e \geq 0 \ \forall e \in E, \\ &\sum_{e:h(e)=i} y_e - \sum_{e:t(e)=i} y_e = \begin{cases} 1 & i = s \\ -1 & i = t \\ 0 & \forall i \in N \setminus \{s,t\} \end{cases} \end{aligned}$$

Proof. A polyhedron whose vertices have only integer coordinates is called "integral". Following Theorem 19.1 in Schrijver (1986, Chapter 19), the polyhedron $\{x | Ax \leq b\}$ where A is a "totally unimodular matrix" and b is an integral vector, is integral. Hence, for an ILP whose constraints can be expressed as $Ax = b$ and $l \leq x \leq u$ where b, l, u are integral vectors and A is a *totally uni-modular* matrix, the relaxation to LP is exact – in Schrijver (1986, Chapter 19, Corollary 19.2a and Theorem 19.3), this is referred to as the Hoffman and Kruskal's theorem. An important example of totally uni-modular matrices is the incidence matrix of a directed graph. Specifically, a matrix that has only {0, +1, −1} entries with exactly one +1 and exactly one −1 in each column is totally unimodular (Schrijver, 1986, Chapter 19, example 2). Now, note that the constraints in (5) can be written as $Ay = b$, $0 \leq y \leq 1$ where A corresponds to the incidence matrix of the attack graph, and b is an integral vector (specifically, b has a 1 at its sth element, a −1 at its tth element, and has zero everywhere else). □

4.2. Solving the defence problem

Even though we turned the non-linear (min-max) integer programming of the attacker to an LP over the edges, the defender's problem is still a complicated min-max optimisation problem. However, thanks to the strong duality in LP, we can turn the inner maximisation LP to its dual, which is a minimisation LP. This will allow us to convert the min-max optimisation of the defender into a single minimisation problem by combining the outer problem with the dual of the inner. Hence, we first derive the dual of the inner problem through the well-known Lemma 3:

Lemma 3. *The dual of the LP in (10) is the following LP:*

$$\begin{aligned} &\min_\lambda \ \lambda_s - \lambda_t \\ s.t. \quad &\lambda_{t(e)} - \lambda_{h(e)} \geq \log(p_e) \qquad \forall e \in E \end{aligned} \tag{7}$$

Applying $\log(\cdot)$ to (3):

$$\log(p_e) = \log \pi_e + \sum_{\substack{c\in\mathcal{C}(e),\\ l\in\mathcal{L}(c)}} x_{cl} \log p_{ecl}$$

Therefore combining the outer optimisation with the dual of the LP-equivalent of the inner problem, we arrive at our main result:

Proposition 2. *The Defence problem in (4) is equivalent to the following mixed integer linear programme (MILP):*

$$\begin{aligned} &\min_{\lambda,x} \quad \lambda_s - \lambda_t \\ \text{s.t.} \quad &\lambda_{t(e)} - \lambda_{h(e)} \geq \sum_{\substack{c\in\mathcal{C}(e),\\ l\in\mathcal{L}(c)}} x_{cl} \log p_{ecl} + \log \pi_e \qquad \forall e \in E \\ &\sum_{c\in\mathcal{C}, l\in\mathcal{L}(c)} x_{cl} \texttt{Cost}_{c,l} \leq B_D \\ &\sum_{c\in\mathcal{C}, l\in\mathcal{L}(c)} x_{cl} \texttt{IndirectCost}_{c,l} \leq B_I \\ &x_{cl} \in \{0,1\} \quad \forall c \in \mathcal{C}, \forall l \in \mathcal{L}(c), \quad \sum_{l\in\mathcal{L}(c)} x_{cl} \leq 1, \ \forall c \in \mathcal{C} \end{aligned} \tag{8}$$

5. Enriching the model and dealing with uncertainty

In this section we enrich our model to incorporate recovery controls in Section 5.1, and we consider multiple target nodes in Section 5.2. In Section 5.3 we study, using sensitivity analysis, how variations in the parameters affect an optimal solution. Section 5.4 addresses input data uncertainty by robust optimisation.

5.1. Incorporating recovery controls

As we discussed earlier, there are generally 3 different classes of security controls: prevention, detection, recovery. Our model so far captures the first two categories of controls, as their impact is effectively as reducing the attack success probabilities on each edge. Here, we describe how our model extends to incorporate recovery-type controls as well.

The effect of a recovery type control is to abate the impact of the attack if it is successful. In particular, suppose the attack's impact, i.e., the damage if the attack is successful to reach target state t, is Λ_t. Then, the effect of a recovery control implemented at level "l" for target t, is that the impact is smaller, say $\Lambda'_{tl} < \Lambda_t$. This can be captured in our model as follows: suppose the indicator for that recovery control is represented by z_{cl}, to distinguish it from x_{cl}. Also, let the set of recovery type controls and a subset of those affecting target t be respectively denoted by $\mathcal{R}$ and $\mathcal{R}(t)$ (to distinguish it from $\mathcal{C}$ and $\mathcal{C}(e)$). Suppose the recovery control c implemented at level l at target t reduces the (conditional) expected losses of a successful attack on target t (conditioned on the attack successfully reaching state t) from Λ_t to Λ'_{tcl}, and define $p_{tcl} = \Lambda'_{tcl}/\Lambda_t$. Then, the new optimisation objective will be:

$$\min_{x,z} \max_y \quad \Lambda_t \prod_{\substack{c\in\mathcal{R}(t),\\ l\in\mathcal{L}(c)}} (p_{tcl} z_{cl} + (1 - z_{cl})) \prod_{e\in E} \left[y_e \prod_{\substack{c\in\mathcal{C}(e),\\ l\in\mathcal{L}(c)}} \pi_e (p_{ecl} x_{cl} + (1 - x_{cl})) + (1 - y_e) \right]$$

In terms of the inner problem, taking the $\log(\cdot)$, we obtain (eliminating the constant term $\log(\Lambda_t)$):

$$\min_{x,z} \max_y \left\{ \sum_{\substack{c\in\mathcal{R}(t),\\ l\in\mathcal{L}(c)}} z_{cl} \log p_{tcl} + \sum_{e\in E} y_e (\log \pi_e + \sum_{\substack{c\in\mathcal{C}(e),\\ l\in\mathcal{L}(c)}} x_{cl} \log p_{ecl}) \right\}$$

Now, we make the observation that the first summation is not a function of y_e at all. Hence, our exact LP equivalence for the inner problem as stated in Proposition 1 still holds. Therefore, after dualisation of the inner problem, we get the following outer (Defence) problem:

$$\min_{x,z,\lambda} \left\{ \sum_{\substack{c\in\mathcal{R}(t),\\ l\in\mathcal{L}(c)}} z_{cl}\log p_{tcl} + \lambda_t - \lambda_s \right\}$$

subject to the same linear constraints as before on λ's and x's, with the obvious additional constraints on z's similar to those of x's:

$$\sum_{l\in\mathcal{L}(c)} z_{cl} \le 1 \qquad \forall c\in\mathcal{R}$$

$$z_{lc}\in\{0,1\} \qquad \forall c\in\mathcal{R}, \forall l\in\mathcal{L}(c)$$

Finally, the direct and indirect constraints should also include z's:

$$\sum_{c\in\mathcal{C}, l\in\mathcal{L}(c)} x_{cl}\texttt{Cost}_{c,l} + \sum_{c\in\mathcal{R}, l\in\mathcal{L}(c)} z_{cl}\texttt{Cost}_{c,l} \le B_D$$

$$\sum_{c\in\mathcal{C}, l\in\mathcal{L}(c)} x_{cl}\texttt{IndirectCost}_{c,l} + \sum_{c\in\mathcal{R}, l\in\mathcal{L}(c)} z_{cl}\texttt{IndirectCost}_{c,l} \le B_I$$

5.2. Extension to multiple targets with different impacts

In an attack graph, there can be damage associated with more than a single attack privilege state. Here, we show how our model can be extended to multiple target vertices. Suppose that there are n targets in G, i.e., $T=\{t_1,\dots,t_n\}$. Furthermore, let Λ_t represent the (normalised, i.e. scaled in the interval [0,1]) "impact" if target t is successfully attacked. Then a measure of security risk for such a setting can be (compare with (1)):

$$R := \max_{t\in T}\Lambda_t \max_{\omega_{s\to t}} \prod_{e\in\omega_{s\to t}} p_e \tag{9}$$

We can incorporate this into our model by modifying the attack graph as follows: introduce an auxiliary sink vertex, σ, and extend the set of edges E by introducing simple edges that connect each of the targets to the sink. Set p_e of such a new edge connecting target t to σ to be Λ_t. Then, a similar argument that lead to Proposition 1 yields:

Proposition 3. *The multi-target attack problem (9) is equivalent to the following LP in the extended attack graph:*

$$\max_{y} \sum_{e\in E} y_e\log(p_e) \tag{10}$$

$$\text{s.t.}\quad y_e\ge 0\ \forall e\in E,$$

$$\sum_{e:h(e)=i} y_e - \sum_{e:t(e)=i} y_e = \begin{cases} 1 & i=s \\ -1 & i=\sigma \\ 0 & \forall i\in N\setminus\{s,\sigma\} \end{cases}$$

The defence problem in (8) can hence be also readily extended.

5.3. Sensitivity analysis

Sensitivity analysis addresses the following kind of questions: Does the optimal investment plan remain optimal if the input data changes? More specifically, what are the ranges in which the input data are allowed such that the optimal plan remains optimal? We present a simple uni-dimensional sensitivity analysis, more sophisticated development being left for future work.

We are particularly interested in how changes to the parameter p_{ecl} affect the optimal plan (x_{cl}). Specifically, we would like to find "tolerance" ranges for p_{ecl}, identified by lower and upper-bounds, within which, the optimal plan (x_{cl}) stays the same.

From Proposition 1 the attack problem is equivalent to a shortest path problem when re-written as a minimisation:

$$\min_{y} \sum_{e\in E} -\log(p_e)y_e.$$

where p_e is defined in (3). In Ramaswamy, Orlin, and Chakravarti (2005) sensitivity analysis is studied for the shortest path problem. Using similar notations therein, let P^* denote the shortest path from s to t; $c(P^*)$ the length of the shortest path; $d(e,\infty)$ the length of the shortest path from s to t when the length of edge e is changed to ∞; and $d(e,0)$ the length of the shortest path from s to t when the length of edge e is changed to 0. Following Corollary 1 in Ramaswamy et al. (2005), we have:

Proposition 4. *For an $e\in E$, if $e\in P^*$, then P^* will stay the shortest path as long as $p_e\in[p_e\exp(-d(e,\infty)+c(P^*)),1]$. If $e\notin P^*$, then P^* will stay the shortest path as long as $p_e\in[0,\exp(c(P^*)-d(e,0))]$.*

In order to compute these bounds, we need to compute $d(e,\infty)$ and $d(e,0)$. This can be achieved by using any shortest path solver. Note that when some p_{ecl} changes we can easily compute the new p_e value. If the new p_e lies within the range calculated in Proposition 4, the shortest path remains unchanged, which means that the attacker takes the same path as before. Therefore, the optimal defence plan is still optimal. If the new p_e exceeds either the lower or the upper bound, the attacker will choose a new attack path. In such a case, the current defence plan may not be optimal any more. We have to solve the model again with the new p_{ecl} value. Note that if more than one p_{ecl} changes at the same time, as long as they are on the same edge e, we are still able to compute the new p_e value and determine whether these changes affect the optimal plan. However, the changes to multiple p_{ecl} across different edges may lead to changes to multiple p_e.

6. Evaluation

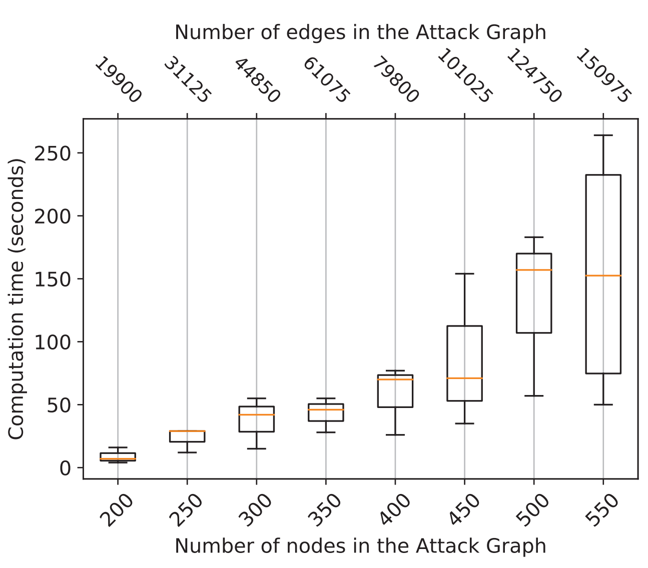
This section shows that the optimisation model scales incredibly well. In particular it can compute the precise optimal solution over complete graphs with 550 nodes in under 3 minutes. For this, we build complete directed graphs so that we can precisely calculate the number of paths as a function of the number of vertices. Optimal solutions over more sparse attack graphs, specifically, Erdős-Rényi random graphs are also computed: the model easily compute, in less than four minutes, the optimal solution for graphs with tens of thousands of edges. This numerical evaluation illustrates that the methodology scales well to realistic scenarios.

All computations were performed on a `MacBook Pro` with a 2.9 gigahertz Intel Core i5 processor and 8 gigabytes RAM. The optimisation was programmed in python using the `PulP` modeller and the `Gurobi` solver.

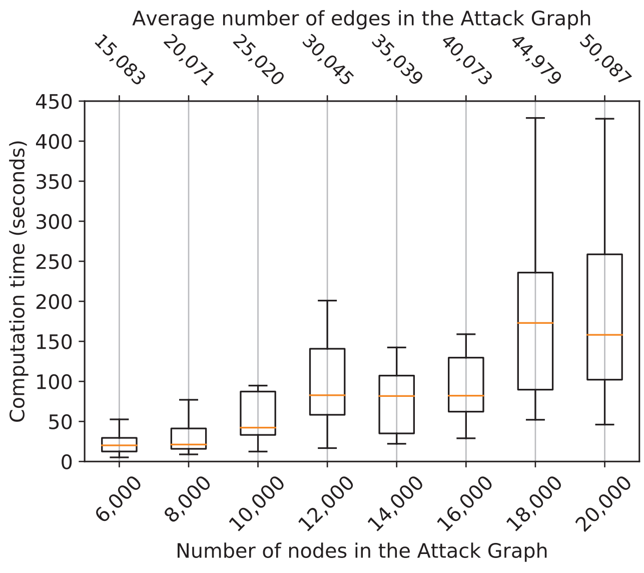
6.1. Optimisation over directed complete graphs

Let us define a directed complete graph over nodes $0,\dots,N-1$ to be a graph such that for each pair of nodes i, j such that $0\le i<j\le N-1$ the edge (i,j) exists. For example, the complete directed graph over nodes 0,1,2 has edges (0,1), (0,2) and (1,2). It is easy to show that the number of paths in a directed complete graph over nodes $0,\dots,N-1$ is $2^{(N-2)}$, and the number of edges is $N(N-1)/2$.

In the experiments, complete directed graphs are generated over an increasing number of nodes N, and 10 controls with two levels each (which is equivalent to 20 controls) are considered. For each control, random (success) probabilities and random direct and indirect costs are generated. To each edge in the graph, a randomly selected number of controls (between 1 and 4) is associated. The optimal solution is then computed: results are shown in Fig. 2a. As



(a) Directed complete graphs.



(b) Random graphs.

Fig. 2. Scalability results for (a) directed complete graphs, and (b) random graphs; The *y*-axis is the time it takes to solve the optimisation (in seconds), *x*-axis is the number of nodes in the attack graph. For the random graph, for each x-value, 20 runs are considered. The "boxplots" show the median (middle bar) as well as the upper and lower quartile values and the range of the computation times.

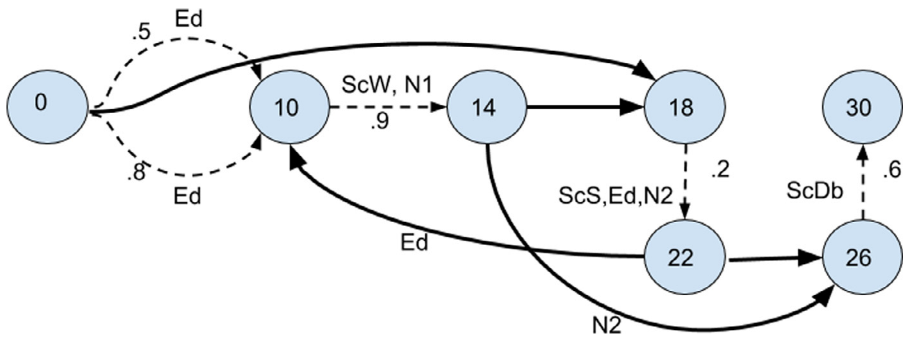


Fig. 3. The attack graph of the case study.

can be seen, the optimal solution over a complete graph with 550 nodes (hence 2^{548}) paths is computed in less than 3 minutes.

6.2. Optimisation over random graphs

Real attack graphs are rarely complete. We have hence also carried a numerical evaluation to test our scalability by processing Erdős–Rényi random graphs into attack graphs with a similar topology to real attack graphs. First a directed Erdős–Rényi random graph with the probability parameter set at $3/N$ where N is the number of vertices in the graph is generated. The parameter $3/N$ translates in the average out-degree of a vertex being 3. A path is added from node 0 to the target going through each node to avoid the trivial situation where there is no feasible attack path. Also graph edges (pairs of indices) were made into triples (multi-edges between nodes) and 37 controls (as in Khouzani et al., 2016) each with two levels were generated with random coefficients and random direct and indirect costs. Each attack step (edge) can be mitigated by 2 randomly chosen controls. The direct and indirect cost budgets were set to 40, which means around 7 controls can be chosen. The running times of the optimisations are shown in Fig. 2b. As the figure shows the optimisation is very fast. For example, for attack graphs with 20,000 nodes (and about 50,000 edges on average), it takes in general less than four minutes. In fact the bottleneck in the experiments wasn't the optimisation but the generation and processing of the random graphs.

6.3. Case study: a network attack graph

This section analyses, using our framework, an attack scenario from the literature: the attack graph is example 3 from Singhal and Ou (2017).

This example models a scenario where an attacker can exploit three possible CVEs: one on a workstation, one on the webserver and one on the database server. The attacker's target is the database server. There are several possible paths: the attacker could start by compromising the web server and use it to escalate the attack to the database server. In another path the attacker could first gain control on a user workstation through a social engineering step, and then attack the database server from the workstation.

The main steps and states of the attack scenario are depicted in Fig. 3 (we have left the same numbering for the nodes as in Singhal & Ou (2017)). State 10 is the state where a malicious input has been accessed, by browsing either a malicious or a compromised website (the two edges leading to 10). State 14 represents

Table 1
values for controls in case study, $\epsilon = 0.0000001$.

Control	ScW	ScS	ScDb	N1	N2	Ed1	Ed2
Cost	1	1	1	1	1	1	1
Indirect cost	1	1	5	1	1	1	1
Efficacies	ϵ	ϵ	ϵ	0.7	0.01	0.5	0.3

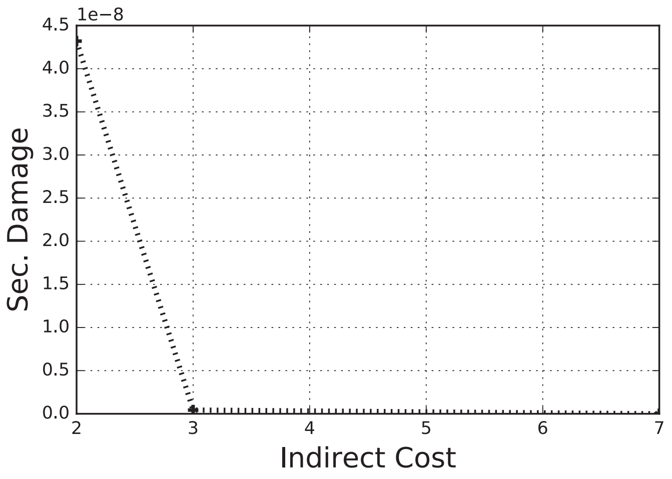


Fig. 4. Pareto front for case study 1.

the privilege state that the attacker can execute user code on a workstation. State 18 is a state where the attacker has net access to the webserver via `http` protocol. In state 22, it is possible to execute code on the webServer running `apache`. State 26 represents net access to the database server via `dbprotocol`. In state 30, the attacker can run code on the database server as `root`. Edges represent possible attack steps, e.g., social engineering (dotted arrows), or exploiting a vulnerability (line-dotted arrows). Each attack step has a baseline success rate which the authors in Singhal and Ou (2017) estimate using various sources. In our model, these correspond to the π_e values. Also on each edge, the possible mitigation for that attack step are shown. For example “`ScW`” is the patching control for `CVE-2009-1918` (an `Internet Explorer` vulnerability). For this case study each patching control has only one level; if applied the probability of success of the attacker over that step is (almost) 0. We set all direct and indirect costs of controls to 1 apart from “`ScDb`” (patch database) which has indirect cost of 5 (due to downtime). A subtle point made in Singhal and Ou (2017) is that patching the database server may have very high indirect costs in terms of downtime and so it may not always be the “best choice”. Our model can capture this aspect naturally because the indirect costs are part of the optimisation.

The values for the controls used in this case study are shown in Table 1. The Pareto front solution for this case study is shown in Fig. 4. The plans chosen at the Pareto points are:

```
point=(2,4.319999e-08),controls=[ScW 1, ScS 1], cost=2
point=(3,4.319999e-10),controls=[ScW 1, ScS 1, N2 1], cost=3
point=(7,4.319999e-15),controls=[ScW 1, ScS 1, ScDb 1], cost=3
```

Consistently with the argument from Singhal and Ou (2017), we see that when the indirect cost budget is very low, the optimal solution chooses patching workstation and web-server. As the indirect cost budget increases the optimal solution adds firewall, blocking connection between workstation and database server. Finally when the indirect cost budget is large enough, the optimal solution chooses all three patches.

7. Conclusion

This work presents an efficient and scalable model for solving cyber-security planning problems. The problem is modelled as a multi-objective bi-level optimisation which can be efficiently solved using exact linear programming relaxation and dualisation. The framework and solution concept are extended to capture recovery type controls, multiple target nodes and uncertainty in the parameters.

The focus of this paper has been on developing a sound mathematical framework with efficient precise solutions. A big challenge ahead is to provide stronger validation methodologies and test suites for this kind of models. Validation could be achieved using complex real world scenarios where the optimisation results are compared with statistics and/or expert decisions.

A more theoretical challenge is to relax the assumption of independence: a preliminary investigation suggests that for a general case, there is a duality gap and hence different solution concepts like Benders decomposition would be required. Both robustness and sensitivity analysis can also be extended to multiple dimensions to better model the impact of uncertainty. Another challenging problem is to more accurately incorporate cybersecurity controls whose main purpose is detection of an active attack. The model should then allow re-allocation of dynamic resources in response to a detection event. This will lead to a challenging tri-level Defender-attacker-defender optimisation on a probabilistic attack graph. Finally, a systematic analysis of the impact of risk-aversion will be a desirable future step.

Acknowledgment

This work was supported by EPSRC grant number EP/R004897/1.

References

Akgün, I., Tansel, B. c., & Wood, R. K. (2011). The multi-terminal maximum-flow network-interdiction problem. *European Journal of Operational Research, 211*(2), 241–251.
Almohri, H. M. J., Watson, L. T., Yao, D., & Ou, X. (2016). Security optimization of dynamic networks with probabilistic graph modelling and linear programming. *IEEE Transactions on Dependable and Secure Computing, 13*(4), 474–487.
Altner, D. S., Ergun, O., & Uhan, N. A. (2010). The maximum flow network interdiction problem: Valid inequalities, integrality gaps, and approximability. *Operatons Research Letters, 38*(1), 33–38.
Baykal-Guersoy, M., Duan, Z., Poor, H. V., & Garnaev, A. (2014). Infrastructure security games. *European Journal of Operational Research, 239*(2), 469–478.
Bhuiyan, T. H., Nandi, A. K., Medal, H., & Halappanavar, M. (2016). Minimizing expected maximum risk from cyber-attacks with probabilistic attack success. In *Proceedings of the IEEE symposium on technologies for homeland security, hst 2016*.
Cavusoglu, H., Raghunathan, S., & Yue, W. (2008). Decision-theoretic and game-theoretic approaches to IT security investment. *Journal of Management Information Systems, 25*(2), 281–304.
Chinchuluun, A., & Pardalos, P. M. (2007). A survey of recent developments in multiobjective optimization. *Annals of Operations Research, 154*(1), 29–50.
Chung, C. J., Khatkar, P., Xing, T., Lee, J., & Huang, D. (2013). Nice: Network intrusion detection and countermeasure selection in virtual network systems. *IEEE Transactions on Dependable and Secure Computing, 10*(4), 198–211.
Dewri, R., Poolsappasit, N., Ray, I., & Whitley, D. (2007). Optimal security hardening using multi-objective optimization on attack tree models of networks. In *Proceedings of the 14th ACM conference on computer and communications security (CCS)* (pp. 204–213). Alexandria, Virginia, USA: ACM.
Ehrgott, M. (2005). *Multicriteria optimization*. Berlin, Heidelberg: Springer-Verlag.
Fielder, A., Panaousis, E., Malacaria, P., Hankin, C., & Smeraldi, F. (2016). Decision support approaches for cyber security investment. *Decision Support Systems, 86*, 13–23.
Franke, D. (2016). *Cyber security basics: Protect your organization by applying the fundamentals* (1st). United States: CreateSpace Independent Publishing Platform.
Frei, S., May, M., Fiedler, U., & Plattner, B. (2006). Large-scale vulnerability analysis. In *Proceedings of the SIGCOMM workshop on large-scale attack defense* (pp. 131–138). ACM.
Frigault, M., Wang, L., Singhal, A., & Jajodia, S. (2008). Measuring network security using dynamic bayesian network. In *Proceedings of the 4th ACM workshop on quality of protection*. In *QoP '08* (pp. 23–30). New York, NY, USA: ACM.

Homer, J., & Ou, X. (2009). SAT-solving approaches to context-aware enterprise network security management. *IEEE Journal on Selected Areas in Communications, 27*(3), 315–322.

Israeli, E., & Wood, R. K. (2002). Shortest-path network interdiction. *Networks, 40*(2), 97–111.

Khouzani, M., Malacaria, P., Hankin, C., Fielder, A., & Smeraldi, F. (2016). Efficient numerical frameworks for multi-objective cyber security planning. In *Computer security – ESORICS 2016* (pp. 179–197). Heraklion, Greece: Springer International Publishing.

Kordy, B., Piètre-Cambacédès, L., & Schweitzer, P. (2014). DAG-based attack and defense modeling: Don't miss the forest for the attack trees. *Computer Science Review, 13–14*, 1–38.

Malik, K., Mittal, A., & Gupta, S. (1989). The k most vital arcs in the shortest path problem. *Operations Research Letters, 8*(4), 223–227.

Nandi, A. K., & Medal, H. R. (2016). Methods for removing links in a network to minimize the spread of infections. *Computers and Operations Research, 69*, 10–24.

Nandi, A. K., Medal, H. R., & Vadlamani, S. (2016). Interdicting attack graphs to protect organizations from cyber attacks: A bi-level defender-attacker model. *Computers and Operations Research, 75*, 118–131.

Nguyen, H., & Sharkey, T. C. (2017). A computational approach to determine damage in infrastructure networks from outage reports. *Optimization Letters, 11*(4), 753–770.

Noel, S., Jajodia, S., O'Berry, B., & Jacobs, M. (2003). Efficient minimum-cost network hardening via exploit dependency graphs. In *Proceedings of the 19th annual computer security applications conference.* (pp. 86–95). IEEE.

Ou, X., Boyer, W. F., & McQueen, M. A. (2006). A scalable approach to attack graph generation. In *Proceedings of the 13th ACM conference on computer and communications security*. In *CCS '06* (pp. 336–345). New York, NY, USA: ACM.

Pan, F., & Schild, A. (2016). Interdiction problems on planar graphs. *Discrete Applied Mathematics, 198*, 215–231.

Panaousis, E., Fielder, A., Malacaria, P., Hankin, C., & Smeraldi, F. (2014). Cybersecurity games and investments: A decision support approach. In *Decision and game theory for security* (pp. 266–286). Springer International Publishing.

Poolsappasit, N., Dewri, R., & Ray, I. (2012). Dynamic security risk management using Bayesian attack graphs. *IEEE Transactions on Dependable and Secure Computing, 9*(1), 61–74.

Qian, Z., Mao, Z. M., Rayes, A., & Jaffe, D. (2012). Designing scalable and effective decision support for mitigating attacks in large enterprise networks. In *Security and privacy in communication networks* (pp. 1–18). Berlin, Heidelberg: Springer Berlin Heidelberg.

Rakes, T. R., Deane, J. K., & Rees, L. P. (2012). IT security planning under uncertainty for high-impact events. *Omega, 40*(1), 79–88.

Ramaswamy, R., Orlin, J. B., & Chakravarti, N. (2005). Sensitivity analysis for shortest path problems and maximum capacity path problems in undirected graphs. *Mathematical Programming, 102*(2), 355–369.

Ratliff, H. D., Sicilia, G. T., & Lubore, S. H. (1975). Finding the n most vital links in flow networks. *Management Science, 21*(5), 531–539.

Roschke, S., Cheng, F., Schuppenies, R., & Meinel, C. (2009). Towards unifying vulnerability information for attack graph construction. In *Information security* (pp. 218–233). Berlin, Heidelberg: Springer Berlin Heidelberg.

Sanjab, A., Saad, W., & Basar, T. (2017). Prospect theory for enhanced cyber-physical security of drone delivery systems: A network interdiction game. *Proceedings of the IEEE International Conference on Communications (ICC)* (pp. 1–6).

Sawik, T. (2013). Selection of optimal countermeasure portfolio in IT security planning. *Decision Support Systems, 55*(1), 156–164.

Schilling, A. (2017). A framework for secure IT operations in an uncertain and changing environment. *Computers & Operations Research, 85*, 139–153.

Schilling, A., & Werners, B. (2016). Optimal selection of IT security safeguards from an existing knowledge base. *European Journal of Operational Research, 248*(1), 318–327.

Schrijver, A. (1986). *Theory of linear and integer programming*. New York, NY, USA: John Wiley & Sons, Inc.

Simaan, M., & Cruz, J. B. (1973). On the Stackelberg strategy in nonzero-sum games. *Journal of Optimization Theory and Applications, 11*(5), 533–555.

Singhal, A., & Ou, X. (2017). Security risk analysis of enterprise networks using probabilistic attack graphs. *Network security metrics* (pp. 53–73)). Springer International Publishing.

Smith, J. C., & Lim, C. (2008). Algorithms for network interdiction and fortification games. In *Pareto optimality, game theory and equilibria* (pp. 609–644). Springer New York.

Sommestad, T., Ekstedt, M., & Holm, H. (2013). The cyber security modeling language: A tool for assessing the vulnerability of enterprise system architectures. *IEEE Systems Journal, 7*(3), 363–373.

The National Cyber Security Centre (2018). *Cyber essentials*. https://www.cyberessentials.ncsc.gov.uk/. Accessed January 7, 2019.

Viduto, V., Maple, C., Huang, W., & LóPez-PeréZ, D. (2012). A novel risk assessment and optimisation model for a multi-objective network security countermeasure selection problem. *Decision Support Systems, 53*(3), 599–610.

Wang, L., Islam, T., Long, T., Singhal, A., & Jajodia, S. (2008). An attack graph-based probabilistic security metric. In *Data and applications security XXII* (pp. 283–296). Berlin, Heidelberg: Springer Berlin Heidelberg.

Wang, L., Noel, S., & Jajodia, S. (2006). Minimum-cost network hardening using attack graphs. *Computer Communications, 29*(18), 3812–3824.

Washburn, A., & Wood, K. (1995). Two-person zero-sum games for network interdiction. *Operations Research, 43*(2), 243–251.

Wood, R. K. (1993). Deterministic network interdiction. *Mathematical and Computer Modelling, 17*, 1–18.

Zenklusen, R. (2010). Matching interdiction. *Discrete Applied Mathematics, 158*(15), 1676–1690.

Zhang, K., & Albert, L. (2018). Interdiction models for delaying adversarial attacks against critical information technology infrastructure. *Technical Report*. University of Wisconsin-Madison, Department of Industrial and System Engineering.

Zhang, K., Albert, L., Luedtke, J., & Towle, E. (2017). Interdiction models for delaying adversarial attacks against critical information technology infrastructure. *Technical Report*. University of Wisconsin-Madison, Department of Industrial and System Engineering.

Zheng, K., & Albert, L. A. (2019). A robust approach for mitigating risks in cyber supply chains. *Risk Analysis*. doi:10.1111/risa.13269.

Zonouz, S. A., Khurana, H., Sanders, W. H., & Yardley, T. M. (2009). Rre: A game-theoretic intrusion response and recovery engine. In *Proceedings of the IEEE/IFIP international conference on dependable systems networks* (pp. 439–448).