Logical Methods in Computer Science
Volume 22, Issue 2, 2026, pp. 7:1–7:27
https://lmcs.episciences.org/

Submitted Mar. 22, 2025
Published Apr. 17, 2026

A SCALABLE GAME-THEORETIC APPROACH FOR SELECTING SECURITY CONTROLS FROM STANDARDIZED CATALOGUES *

DYLAN LÉVEILLÉ AND JASON JASKOLKA

Carleton University, Canada
e-mail address: dylan.leveille@carleton.ca, jason.jaskolka@carleton.ca

Abstract. Selecting the combination of security controls that will most effectively protect a system's assets is a difficult task. If the wrong controls are selected, the system may be left vulnerable to cyber-attacks that can impact the confidentiality, integrity, and availability of critical data and services. In practical settings, as standardized control catalogues can be quite large, it is not possible to select and implement every control possible. Instead, considerations, such as budget, effectiveness, and dependencies among various controls, must be considered to choose a combination of security controls that best achieve a set of system security objectives. In this paper, we present a game-theoretic approach for selecting effective combinations of security controls based on expected attacker profiles and a set budget. The control selection problem is set up as a two-person zero-sum one-shot game. Valid control combinations for selection are generated using an algebraic formalism to account for dependencies among selected controls. Using a software tool, we apply the approach on a fictional Canadian military system with Canada's standardized control catalogue, ITSG-33. Through this case study, we demonstrate the approach's scalability to assist in selecting an effective set of security controls for large systems. The results illustrate how a security analyst can use the proposed approach and supporting tool to guide and support decision-making in the control selection activity when developing secure systems of all sizes.

1. Introduction

With computers becoming more interconnected than ever, there emerges an even greater need to secure computer systems and to effectively manage security risks. Security risks are mitigated by the implementation of a set of security controls. A *security control* refers to a safeguard or countermeasure prescribed for an information system or an organization designed to protect the confidentiality, integrity, and availability of its information and to meet a set of defined security requirements [RPG+21]. *Control selection* is an activity commonly found as part of a risk management process [Int18], a systems engineering process [RPG+21], the Risk Management Framework [Joi18], the Cybersecurity Framework [Nat24], or the Privacy Framework [Nat20]. Control selection involves selecting and documenting the security controls

Key words and phrases: Security controls, Control selection, Security standards, Game theory, Knapsack problem.

* A preliminary version of this article was presented at the 15th International Symposium on Games, Automata, Logics, and Formal Verification (GandALF 2024) [LJ24a]. In this version, we extend the article by applying the proposed approach to a much larger use case system with the use of a custom tool.

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.46298/LMCS-22(2:7)2026

necessary to protect the information system and organization commensurate with risk to organizational and system operations and assets, individuals, other organizations, and the nation [Joi18].

During the control selection activity, security analysts typically select security controls from standardized security control catalogues, such as NIST SP 800-53 [Joi20b], ITSG-33 [Gov14], ISO 27002 [Int22a], CIS Critical Security Controls [Cen21], and MITRE D3FEND™ [KS20], among others. However, selecting combinations of controls from these catalogues can be difficult for several reasons. First, these control catalogues are large, and many possible controls could be selected to mitigate the risks identified for a given system. In practical settings, it is not possible to select and implement every control possible. Considerations such as budget, effectiveness, and dependencies among various controls, must be considered to choose a combination of security controls that best achieve a set of system security objectives. Second, control selection is largely a human-oriented activity. The dynamics between security analysts (defenders) strategizing to protect critical systems and assets and achieve a set of security objectives, and attackers aiming to impact critical systems and assets and violate those same security objectives must be considered when deciding on the most effective and cost-efficient combination of security controls. Although numerous optimization-based solutions are adept at accounting for various properties of the controls themselves, they fail to capture the human element that is inherently part of the control selection activity.

To address the above-mentioned challenges, we propose a game-theoretic approach for security control selection. The human aspects of the control selection problem, as well as the large space of possible control combinations, and their dependencies and constraints, lend themselves well to an application of game theory. Specifically, we set up a two-person zero-sum one-shot game which is played by a security analyst. The analyst selects their strategy based on an attacker profile, characterized by the expected targeted assets and security objectives. Each analyst strategy corresponds to a combination of security controls from a chosen control catalogue that can achieve the security objectives. Valid control combinations are generated using an algebraic formalism (akin to product family algebra [HKM11]) to account for dependencies among selected controls. The outcome of the game is a combination of suggested security controls that can effectively defend against the considered attacker profile. To demonstrate the approach's scalability towards large systems, we apply the approach on an illustrative Canadian military system using a previously proposed tool [LJ24b].

The rest of this paper is organized as follows. Section 2 provides an overview of existing works on the topic of control selection and of game theory applications in cybersecurity. Section 3 presents the proposed game-theoretic approach for control selection. Section 4 presents a previously proposed tool that automates the proposed approach. Section 5 provides a large illustrative example in which the proposed approach is applied using the tool from Section 4. Section 6 discusses the benefits and potential limitations of the proposed approach. Lastly, Section 7 concludes and briefly discusses future work.

2. Related Work

In this section, we present existing works that propose solutions for assisting with control selection. Additionally, we discuss the lack of scalability of these approaches, thus highlighting their impracticality for large real-world systems.

2.1. **Control Selection Approaches.** Many existing approaches to support the security control selection activity are based on setting and solving optimization problems. For example, for each considered control, Yevseyeva et al. [YBFEV15] assign a probability of "survival" for each possible threat (i.e., the probability that the threat persists in the presence of the control). Probabilities are also assigned for the expected loss of successful attacks. The goal of the proposed approach is to minimize this expected loss, under constraints such as cost and system resources. Similarly, Almeida and Respício [AR18] also assign probabilities to controls based on their expected performance in mitigating certain vulnerabilities. For the proposed approach, the goal is to find the optimal controls for the system that will minimize an objective function accounting for both loss and cost. Many such optimization approaches have been proposed, differing only by the metrics used for optimization and their specific optimization techniques. This includes the works by Tsiodra et al. [TCG+21], Schmidt et al. [SAZ21], and Sarala et al. [SZV16]. A different approach was proposed by Dewri et al. [DPRW07] where systems are modelled as trees, in which the leaf nodes represent possible attacks. Controls therefore mitigate one or many leaf nodes. With the attack impact, attack frequency, and cost of each control known, the optimal controls can be found by optimization. A similar tree-like approach was also proposed by Park and Huh [PH20]. Using an Attack-Defense tree, in which attacker motivations and defender mitigations are modeled as a tree, is also a popular approach for assisting with control selection [KMRS14, FW20, JYFF16]. Although these trees can be modeled in various ways, optimal solutions are obtained through optimization. Lastly, Shahpasand et al. [SSGG15] and Uuganbayar et al. [UYMM21] have both developed custom algorithms that solve control selection as an optimization problem. While optimization-based approaches can account for important considerations and constraints such as cost and effectiveness, they depend heavily on assumptions about probabilities for threat likelihoods or control success rates. Such probabilities are not likely to be accurately known in a practical setting.

Several other approaches for control selection that are not based on optimization have also been proposed. Bettaieb et al. [BSS+20] presented an approach where a machine learning model is trained with historical data from previous security assessments to make predictions using certain features of interest from a given security assessment to determine optimal controls. However, using historic data to determine how to protect a system has several limitations as every system is unique and may operate in widely different environments. In another work, Kiesling et al. [KEG+16] proposed a simulation-based approach to determine the optimal controls for a system. To do this, expected attacks are simulated on different components of the system using different possible control combinations to find the optimal ones. This approach is noteworthy as it simply uses the properties of the controls and of the current system (such as different threats) to find the most optimal control combinations and does not depend on any probabilities.

Although the approaches presented above may be suitable for solving the control selection problem, they fail to account for the human behavioural aspect of an attacker, opting instead to categorize potential attackers as a series of possible attacks. Additionally, the main issue with many of these approaches is the need to define probabilities for threat likelihoods or control success rates. Such probabilities are not likely to be known in any realistic scenario. These works also fail to acknowledge possible dependencies between the controls. In contrast to existing work, the proposed approach aims to leverage game theory to address the shortcomings of current control selection approaches by placing a central focus on possible attacker behaviours, while also considering the dependencies and constraints that limit the

selection of certain combinations of security controls to effectively mitigate the threats to a system.

2.2. **Scalability of Control Selection.** While existing approaches fail to capture the human nature of this problem, they lend themselves well to being automated as many of them simply involve solving an optimization problem. In fact, most of these works mention the necessity of automation to apply their approaches practically. For example, Almeida and Respício [AR18] developed an Excel tool to automate their approach. However, solutions are generated through brute-force optimization. Brute-force methods are also used to automate the approach by Park and Huh [PH20]. While brute-force optimization is functional, given the numerous constraints that exist in control selection, it is not scalable for cases in which a large number of controls are considered. Although the works of Shahpasand et al. [SSGG15] avoid brute-force optimization through the development of a custom algorithm, they provide no details on the computational complexity of their algorithm. Similarly, while Uuganbayar et al. [UYMM21] provides experimental showing that their algorithm finds results within five to ten minutes, the computational complexity of the algorithm is not provided.

The literature shows that there is a lack of research in the automation of control selection approaches as the computational complexity of this problem is overlooked. Additionally, as existing approaches mostly rely on optimization, an automated game-theoretic approach for control selection has yet to be developed. Although the approach automated in this work is based on game theory, the constraints inherent in control selection necessitates that a custom algorithm be developed for efficient automation. Lastly, this work is the first to automate a control selection approach that considers control dependencies.

3. The Proposed Approach

In this section, we present our game-theoretic approach for security control selection. An overview of the approach is shown in Figure 1. The approach consists of two main stages shown as swim lanes and six steps shown in blue. All steps are to be conducted by a security analyst. A detailed description of each step of the proposed approach is provided in the sections below.

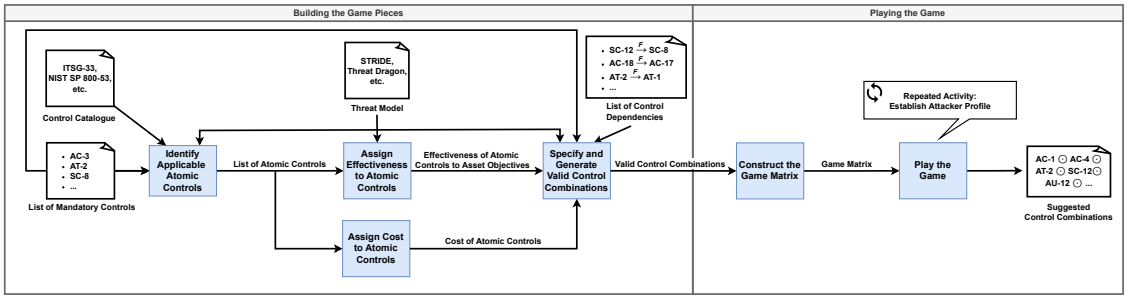


Figure 1. An overview of the game-theoretic approach for security control selection

3.1. **Identify Applicable Atomic Controls.** Before the approach can be applied, the analyst must be provided with a security control catalogue, such as NIST SP 800-53 [Joi20a] or ITSG-33 [Gov14], from which security controls for the system will be considered. The approach begins with the security analyst identifying applicable atomic controls from this given control catalogue. In our context, the *atomic controls* are the smallest (indivisible) security controls that can be selected from a control catalogue. We say that a control is *applicable* to a system if it could provide any form of protection from the threats to the assets of the system.

We assume that a list of threats and assets are available to the security analyst in the form of a threat model. A threat model is defined as "a structured representation of all the information that affects the security of an application" [Dra]. Threat models typically include identified system threats and their impact on the assets within the system [Dra, Mur16]. The threat model can be obtained by applying a well-known threat modelling methodology such as STRIDE [Mic22] or PASTA [UM15].

To determine the applicability of an atomic control, the security analyst must carefully consider each atomic control from the given control catalogue and decide if the control can mitigate the identified threats to the assets. Additionally, certain organizational needs or standards and regulations for the system's application domain may require that specific security controls be present in the system. The security analyst must therefore ensure that these *mandatory controls* are included as part of the set of applicable controls identified. This is a manual process. However, it should be noted that the effort required for this activity is reasonable as security control catalogues are typically separated by control families which help guide an analyst in finding suitable controls [RGJ23]. Instructions and guidance for performing this task is well documented by ISO 27005 [Int22b] and NIST SP 800-53B [Joi20a].

At the end of this step, the analyst will have a set of applicable atomic controls for the system. Note that the combination of suggested controls found by applying the proposed approach will be a subset of the controls gathered in this initial step.

3.2. **Assign Effectiveness to Atomic Controls.** For each identified atomic control, the analyst proceeds by assigning an effectiveness of the control at satisfying each *security objective* on each asset in the system. Security objectives represent the security needs of the assets on the system, such as confidentiality, integrity, and availability [CEL+20]. These objectives are normally included as part of the threat impacts described in the threat model. It is important to remember that the goal of the proposed approach is to create a game. In every game, there needs to be strategies, and payoffs defined for each strategy. Assigning the effectiveness of each atomic control therefore defines the payoffs of each atomic control in the game.

To perform this step of the approach, the atomic payoff matrix presented in Table 1 must be completed. The rows represent each atomic control that was identified in the previous step (denoted $C_1, \ldots, C_N$). The columns represent the security objectives for each asset (denoted $O_1, \ldots, O_M$). We expect the analyst to assign a value between 0 and 1 in each cell of this matrix. A value of 0 means that the atomic control is not effective at satisfying the specified objective for an asset, while a value of 1 means that the atomic control is completely effective at satisfying the specified objective for an asset. Each payoff value is therefore normalized. Provided that the rating scheme is selected and used consistently throughout the approach, the analyst is free to choose any method for assigning the effectiveness values for the atomic payoff matrix. For example, the analyst may choose to use a quantitative

approach as in the Defect Detection and Prevention (DDP) risk reduction strategy developed by NASA [FCHJ05], or they may alternatively choose to use a qualitative rating mapped to quantitative values as in [LZ11].

Table 1. General form of the atomic payoff matrix

	Asset 1			*Asset 2*			...	*Asset X*		
	O_1	...	O_M	O_1	...	O_M	...	O_1	...	O_M
C_1										
$\vdots$										
C_N										

At the end of this step, the analyst will have the effectiveness of each applicable atomic control for satisfying each security objective on each asset in the system.

3.3. **Assign Cost to Atomic Controls.** In practical settings, cost or time constraints limit how many controls can be part of a system; if there are too many controls they may exceed a certain budget or cannot be implemented in reasonable time. In fact, without such constraints, there could technically be no limitations on the number of controls that can be selected for a system, and the best solution would be to select them all.

At the same time as assigning effectiveness, the analyst will also need to assign a cost for each identified atomic control. We expect the analyst to assign a cost from the set of real numbers $\mathbb{R}$. The units for cost could be represented as dollars, thousands of dollars, or any other form of currency as long as the same units are consistently used for all cost values. Furthermore, no units could be used if desired. Without units, costs simply represent an implementation effort.

After this step, the analyst will have the cost associated with each applicable atomic control.

3.4. **Specify and Generate Valid Control Combinations.** Given a set of applicable atomic controls, the analyst needs to specify and generate the set of valid control combinations that satisfies their constraints. To formally capture these constraints, we have decided to use an algebraic specification based on product family algebra [HKM11] to specify and generate valid combinations of security controls.

Product family algebra extends the mathematical notions of semirings to describe and manipulate product families. A semiring is an algebraic structure $\big(S, +, \cdot, 0, 1\big)$ consisting of a set S with a commutative and associative binary operator $+$ and an associative binary operator $\cdot$. An element $0 \in S$ is the identity element with respect to $+$, while an element $1 \in S$ is the identity element with respect to $\cdot$. Additionally, $\cdot$ distributes over $+$ and element 0 annihilates S with respect to $\cdot$. A semiring is commutative if $\cdot$ is commutative and a semiring is idempotent if $+$ is idempotent.

For ease of presentation, we recast the vocabulary of product family engineering into the vocabulary of security controls by first defining a security control algebra to express families of security control combinations generated from a set of atomic controls.

Definition 3.1 (Security Control Algebra)**.** *A* security control algebra *is a commutative idempotent semiring* $\mathcal{C} \stackrel{\text{def}}{=} \big(C, \oplus, \odot, 0, 1\big)$ *where each element of the semiring* $c \in C$ *is a security control family.*

In a security control algebra, the operator $\oplus$ is interpreted as a choice between two security control families and the operator $\odot$ is interpreted as a mandatory composition of two security control families[1]. The element 0 represents a non-implementable security control combination that cannot exist and the element 1 represents the empty security control combination which has no controls. A security control family is called a *security control combination* if it is indivisible with regard to the choice operator $\oplus$. Additionally, it is called a *proper security control combination* if $c \neq 0$. A security control combination is an *atomic control* if is it is indivisible with regard to the mandatory composition operator $\odot$. Optional controls are expressed as a choice between the controls and the empty security control combination 1. A list of optional controls $c_1, \ldots, c_n$ is denoted by $opt\big[c_1, \ldots, c_n\big] \stackrel{\text{def}}{=} (c_1 \oplus 1) \odot \cdots \odot (c_n \oplus 1)$.

For two security control families c_1 and c_2 in a security control algebra, the *refinement relation* ($\sqsubseteq$) is defined as $c_1 \sqsubseteq c_2 \stackrel{\text{def}}{\iff} \exists(c_3 \ \mid: \ c_1 \leq c_2 \odot c_3\ )$ where $\leq$ is the natural semiring order (i.e., $c_1 \leq c_2 \stackrel{\text{def}}{\iff} c_1 \oplus c_2 = c_2$). To specify constraints, such as dependencies between controls, we use the requirement relation.

Definition 3.2 (Requirement Relation [HKM11])**.** *For elements* c_1, c_2, c_3, c_4 *and security control combination* x *in a security control algebra, the requirement relation* ($\rightarrow$) *is defined inductively as:*

$$\begin{aligned} c_1 \xrightarrow{x} c_2 \quad &\stackrel{\text{def}}{\iff} \quad x \sqsubseteq c_1 \implies x \sqsubseteq c_2 \\ c_1 \xrightarrow{c_3 \oplus c_4} c_2 \quad &\stackrel{\text{def}}{\iff} \quad c_1 \xrightarrow{c_3} c_2 \ \wedge \ c_1 \xrightarrow{c_4} c_2 \end{aligned}$$

For elements c_1, c_2 and x, the requirement relation $c_1 \xrightarrow{x} c_2$ can be read as "c_1 requires c_2 within x."

With this setting, all security control combinations can be specified algebraically by expressing the mandatory and optional controls as terms of a security control algebra along with requirement relations describing control dependencies.

The resulting specification serves as the basis for generating all possible proper security control combinations. However, not all control combinations are possible as some may exceed our defined budget. To make this determination we first define how to calculate the cost of a proper security control combination. In what follows, let $P \subseteq C$ be the set of all proper security control combinations in a security control algebra $\mathcal{C}$.

Definition 3.3 (Cost of a Proper Security Control Combination)**.** *The cost of a proper security control combination* $Cost : P \rightarrow \mathbb{R}$ *is a function defined inductively for any proper security control combinations* $a, b \in P$ *in a security control algebra* $\mathcal{C}$ *as:*

$$\begin{aligned} Cost(1) &= 0 \\ Cost(a) &= G(a) \text{ if } a \text{ is atomic} \\ Cost(a \odot b) &= Cost(a) + Cost(b) \end{aligned}$$

where G *is a function that returns the cost assigned to an atomic control (see Section 3.3).*

[1]When the context is clear, we omit the mandatory composition operator $\odot$ when specifying security control algebra terms.

Now that we can compute the cost of a proper security control combination, we determine the set of valid security control combinations. A *valid security control combination* is a proper security control combination that does not exceed the prescribed cost budget. The validity of a control combination is formalized in the following rule.

Definition 3.4 (Budget Rule)**.** *For any $p \in P$ and budget B:*

$$Valid(p) \iff Cost(p) \leq B$$

After this step, the analyst will have a set of valid security control combinations that satisfy the prescribed budget. These valid security control combinations become the strategies that an analyst can select when playing the game.

3.5. **Construct the Game Matrix.** In this step, the analyst constructs the game matrix. The general form of the game matrix can be seen in Table 2. The rows represent the valid security control combinations found from the last step (denoted $Combo_1, \ldots, Combo_N$). The columns represent the security objectives for each asset (denoted $O_1, \ldots, O_M$). Note that the game matrix is identical in style to that of the atomic payoff matrix (see Table 1). The game matrix simply has control combinations as rows rather than atomic controls. In the game, the strategies of the security analyst will be the valid security control combinations, while the strategies of the attacker will be each security objective that could be violated on every asset.

Table 2. General form of the game matrix

	Asset 1			*Asset 2*			...	*Asset X*		
	O_1	...	O_M	O_1	...	O_M	...	O_1	...	O_M
$Combo_1$										
⋮										
$Combo_N$										

Each outcome in a game is tied to a payoff [Str93]. In our game, the payoffs are represented from the perspective of the analyst and represent the effectiveness of the security control combinations towards every asset's security objectives. Just as cost was defined inductively, we can define a proper control combination's effectiveness towards an asset's security objective in a similar manner.

Definition 3.5 (Effectiveness of a Proper Security Control Combination)**.** *The effectiveness of a proper security control combination towards an asset's security objective $Eff : P \to \mathbb{R}$ is a function defined inductively for any proper security control combinations $a, b \in P$ in a security control algebra $\mathcal{C}$ as:*

$$\begin{aligned} Eff(1) &= 0 \\ Eff(a) &= E(a) \text{ if } a \text{ is atomic} \\ Eff(a \odot b) &= 1 - (1 - Eff(a))(1 - Eff(b)) \end{aligned}$$

where E is a function that returns the effectiveness assigned to an atomic control for an asset's security objective (see Section 3.2).

With Definition 3.5, the payoff values in the game matrix can be calculated. Note that the calculation of the effectiveness of a security control combination is inspired from the combined effectiveness calculation as part of NASA's DDP approach [FCHJ05], in which effectiveness is calculated by subtracting the compounded ineffectiveness of each control in the combination from complete effectiveness (i.e., 1).

After this step, the analyst will have the game matrix so that they can proceed to play the game.

3.6. **Play the Game.** The game is a *two-person zero-sum one-shot game*. The game is played by *two persons*: the security analyst and the attacker. The attacker may embody one or multiple entities, but acts as a unified adversary. The goal of the security analyst is to select the security control combination that will best protect the security objectives for the assets they believe will be targeted by the attacker. Only one security control combination can be selected, hence it is a *one-shot* game. On the other hand, the goal of the attacker is to attack assets and violate corresponding security objectives. An attacker could attack one or many assets and violate one or more objectives from a series of attacks. Regardless, an attacker will select which assets and objectives they will target and will commit to attacking the selected assets and objectives. The attacker will naturally prefer attacking assets which are not properly defended, i.e., those for which there are minimally effective security controls. The effectiveness values in the game matrix (payoffs) do not directly correlate to a loss to the attacker. However, it is easy to see that the higher the values, the more difficult it is for an attacker to conduct a successful attack leading to corresponding security objective violations. Therefore, what the security analyst gains in effectiveness is what the attacker loses in their ability to successfully conduct their attack; hence, it is a *zero-sum* game. Note that this game is strictly non-cooperative; the analyst and attacker are competing directly and would never want to cooperate.

Using the game matrix, the analyst must select a strategy (i.e., a valid security control combination) to play that will best protect the system assets and security objectives that they believe are most important. To do this, an analyst must establish the expected attacker profile. An *attacker profile* is an expected set of the assets and corresponding security objectives targeted by the attacker. One can imagine different classes of attackers having different capabilities, and different targets, thereby establishing different attacker profiles. In the context of a game, an attacker profile corresponds to guessing the attacker strategy so that it can be defended. This consideration of the dynamics of the analyst and the attacker strategies in this game is what differentiates it from existing security control selection approaches.

It is impossible to know exactly which security objectives on which assets will be attacked, so assumptions must be made. One way to do this is to determine where most of the critical information flows in the system and which assets may be prone to more attacks (i.e., have more expected threats). The combination of these ideas can help localize assets that are more attractive for attacks, and therefore puts the security objectives of these assets at higher risk of violation. Another way to do this is to consider the risk to each asset and corresponding security objectives for the identified threats to the system (which we consider known to the analyst). In this case, prioritizing defence of assets and security objectives targeted by high risk threats may be a good approach. Regardless, once the attacker profile is determined, then the suggested strategy (i.e., the most effective security control combination) can be found.

Regardless of the approach taken to establish the attacker profile, it will articulate the objectives that are expected to be violated by an attacker. For this work, we establish an attacker profile by considering and prioritizing different attacker objectives. Attacker objectives correspond to a set of security objectives for some assets that are equally expected to be targeted by an attacker. Within an attacker profile, several attacker objectives may be prioritized according to their perceived likelihood of being targeted by the attacker to obtain a priority order for the objectives. For example, the security analyst could establish an attacker profile in which the attacker has two ordered attacker objectives: (1) to target the confidentiality of two specific assets equally, and (2) to target the integrity of two other assets equally. The security analyst may consider as many attacker objectives as they desire when developing an attacker profile. The suggested analyst strategies for an attacker profile will be those which maximize the *total effectiveness* across each attacker objectives (i.e., the sum of the effectiveness returned by Definition 3.5 for the security objectives in the attacker objectives is maximized in the priority order). To better understand this concept, an example of the strategies found by playing the game with an attacker profile with two ordered attacker objectives is visualized in Figure 2.

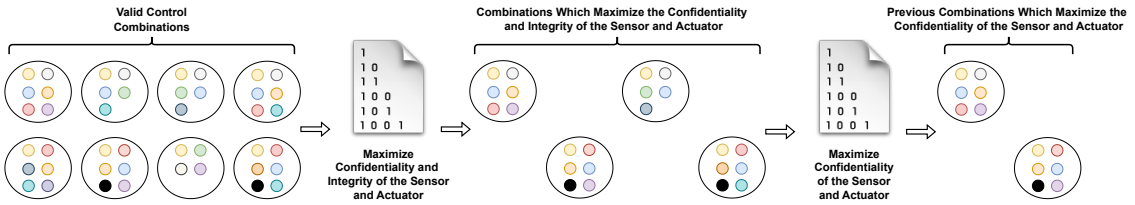


Figure 2. Finding the suggested controls for an attacker profile with multiple ordered attacker objectives

In this example, there are initially eight valid security control combinations. Each security control combination has a unique set of controls (denoted by the different coloured dots in the figure). Only two assets exist in this system; a Sensor and an Actuator. The attacker profile has two ordered attacker objectives: (1) the confidentiality and integrity of the Sensor and the Actuator and then (2) the confidentiality of the Sensor and the Actuator. From all valid control combinations, the control combinations which maximize the first set of attacker objectives is found, yielding four different combinations. From these four combinations, the combinations which maximize the second set of attacker objectives is found, yielding two control combinations. As there are no more ordered attacker objectives, the two resulting control combinations are considered equally valid, and represent the suggested strategies. Note that since the suggested strategies are derived through a series of maximization problems, it may be possible for more than one strategy to be the most effective for a given attacker profile.

At the end of this step, the analyst will obtain at least one strategy that best protects against the considered attacker profile and that corresponds to the suggested security control combinations to be implemented in the system. It is important to remember that this approach is a game. Therefore, as with any game, it is recommended that the game be re-constructed with different maximum budget values and re-played with different attacker profiles (as illustrated in Figure 1). This can help gauge and compare the control combinations that should be used for the system under different constraints and goals.

4. CSAT: Control Selection Assistant Tool

In this section, we present the Control Selection Assistant Tool (CSAT) [LJ24b] that automates the approach presented in Section 3. The primary goal of CSAT is to eliminate the manual effort required to find suggested control combinations for a given system. In fact, with N optional controls, there could exist up to 2^N valid security control combinations for the approach, making it impractical when applied manually to large systems. With CSAT, results to the approach can be obtained without having found all valid security control combinations. An overview of CSAT's design and functionality can be seen in Figure 3. The remainder of this section describes the components of this design. CSAT and the example data presented in this section are available at: `https://gitlab.com/CyberSEA-Public/CSAT`.

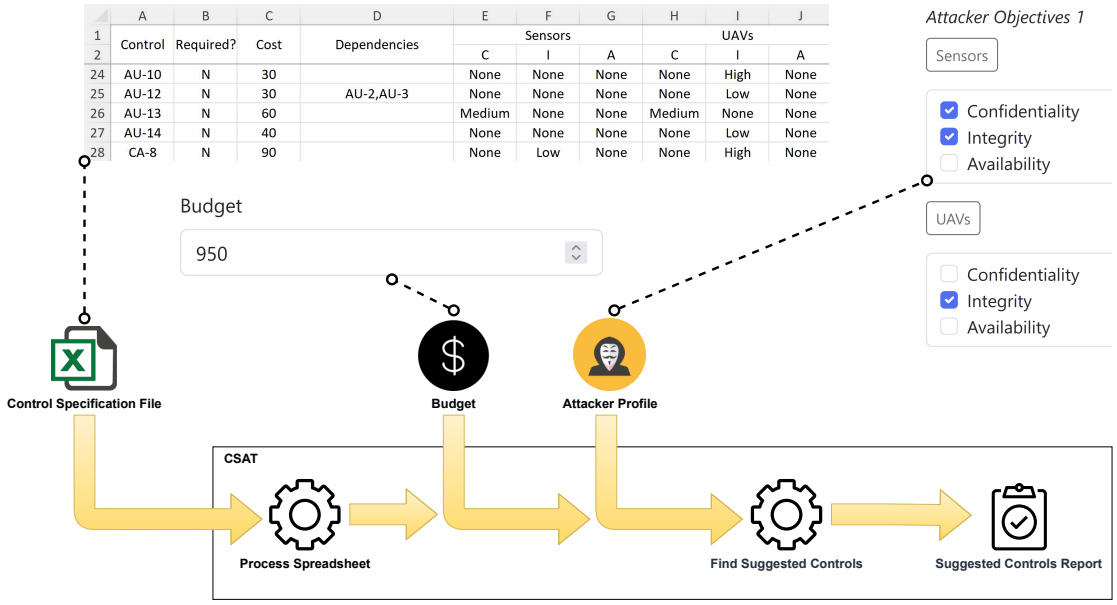


Figure 3. Overview of CSAT's design and functionality

4.1. **Control Specification File.** The atomic controls considered for the system, their effectiveness values, their cost, whether they are mandatory or optional, and their dependencies, are provided through an Excel spreadsheet (i.e., `.xlsx`) referred to as the control specification file. An example of this spreadsheet being filled with controls can be seen in Figure 4. As shown in the figure, CSAT currently only supports security objectives based around the CIA triad. Additionally, each effectiveness value must be one of *None*, *Low*, *Medium*, *High*, or *VeryHigh*. These represent the normalized values of 0, 0.2, 0.5, 0.8, and 0.9 respectively. These values are adopted and adapted from the metrics in the Common Vulnerability Scoring System (CVSS) [MSR07]. No rating was assigned to the value of 1 as it is unrealistic to expect a single control to fully protect a security objective. To account for uncertainty, multiple effectiveness values can be provided for a control's effectiveness towards an asset's security objective (e.g., in the figure, *Low* versus *Medium* for C1's effectiveness towards the confidentiality of Asset1).

	A	B	C	D	E	F	G	H	I	J
1	Control	Required?	Cost	Dependencies	Asset1			Asset2		
2					C	I	A	C	I	A
3	C1	N	7		Low,Medium	None	None	Low	None	None
4	C2	Y	5		Low	Medium	Low	Low	Low	Medium
5	C3	N	10		Medium	Medium	None	Medium	Low	Low
6	C4	N	6	C1	Medium,High	None	Medium	Medium,High	Low	Low
7	C5	N	2		Low	Medium	Low	Low	Medium	Low
8	C6	N	4		Low	Medium	Low	Low	None	Low
9	C7	N	2	C5,C6	Low,Medium,High	Medium	High	Low	VeryHigh	None

Figure 4. An example of a filled CSAT control specification file

4.2. **Process Spreadsheet.** As CSAT is a Python-based web tool, it is accessed through a web browser. The control specification file can simply be passed into the file picker on the initial page. CSAT will parse the specification file provided using `pandas` [Num24] and store the information internally.

4.3. **Budget and Attacker Profile.** After the control specification file is submitted and processed, a new page will be shown prompting for the budget and attacker profile. This can be seen in Figure 5.

In the tool, the budget can simply be entered in the budget field. To specify the attacker profile, the security objectives in the first set of attacker objectives is specified by selecting the security objectives to be protected for each asset (each asset is a button on the tool which expands to its security objectives). Attacker objectives can be added and removed to create an attacker profile with as many ordered attacker objectives as desired. This is done using the "Add Attacker Objectives" button and the "Remove Attacker Objectives" button respectively.

In this figure, the budget is 20, and we have an attacker profile with two ordered attacker objectives: the first targets the confidentiality and integrity of Asset1 and the confidentiality of Asset2 equally, and the second targets the availability of Asset1 and the integrity of Asset2 equally. Once submitted, CSAT will parse and store the budget and attacker objectives provided and will begin finding the suggested security controls for the system.

4.4. **Find Suggested Controls.** The logic used by CSAT to find the suggested control combinations is visualized in Figure 6. To begin, CSAT identifies a set of candidate security control combinations that are effective against the first set of attacker objectives. From this set of candidate security control combinations, CSAT will keep those which maximize the effectiveness towards the security objectives of the first set of attacker objectives. From these combinations, CSAT then keeps those which maximize the effectiveness towards the security objectives of the second set of attacker objectives. This process is repeated until there are no more ordered attacker objectives. Note that this logic is identical to that of the example provided in Figure 2. A minor difference is that CSAT first identifies candidate security control combinations. Finding these candidate control combinations is essential for reducing the large number of security control combinations to evaluate, thus enabling the tool to generate its solutions within a reasonable timeframe.

Finding these candidate control combinations corresponds to solving a well-known NP complete problem known as the bounded 0-1 knapsack problem [KV01]. The algorithm

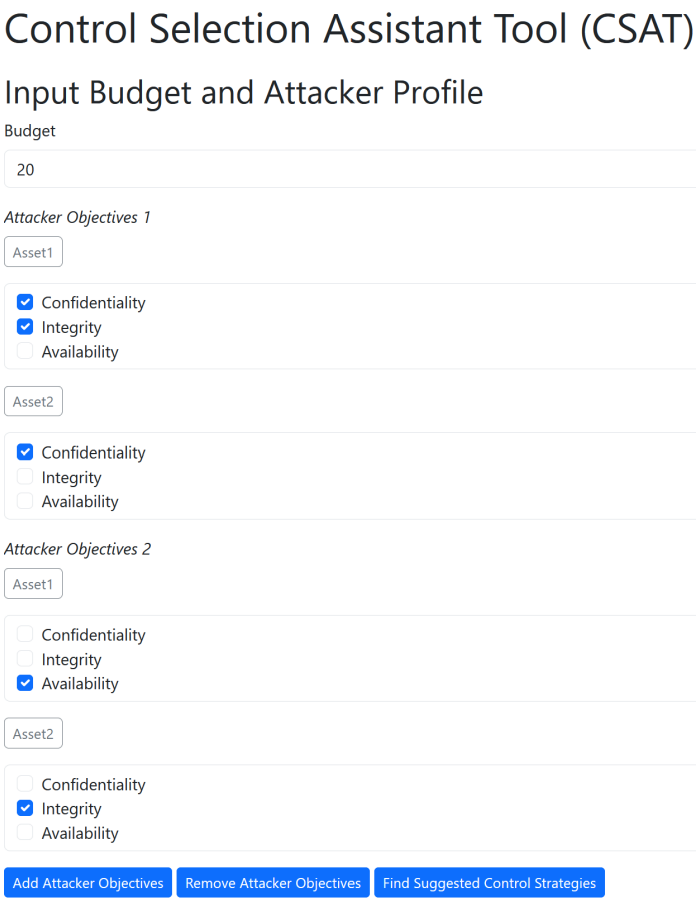


FIGURE 5. The CSAT user interface before the suggested controls are found

developed to find these candidate control combinations (shown in in green in Figure 6) has computational complexity $O(n)$ in special cases and was previously described alongside the tool in [LJ24b].

4.5. **Suggested Security Control Combinations Report.** With the suggested control combinations found, CSAT will generate a report for the suggested control strategies. Figure 7 shows part of the report generated by the tool after pressing submit for our running example.

The "Case" heading is used to denote the suggested control combinations that were obtained for a particular case. As some controls were assigned multiple effectiveness values towards some of the security objectives, the suggested security controls must be found for each possible permutation of these uncertain values (which we define as a case). The case seen in the figure is the one were the confidentiality of C1 towards Asset 1 is *Medium*, the confidentiality of C4 towards Asset 1 is *High*, the confidentiality of C4 towards Asset 2 is *High*, and the confidentiality of C7 towards Asset 1 is *High*.

In the report, the suggested security control combinations and their associated costs are displayed for each case. All suggested combinations for a case share the same cost as CSAT resolves ties in the results by selecting those with the lowest cost. As many cases could share the same suggested security controls, cases with identical results are printed sequentially.

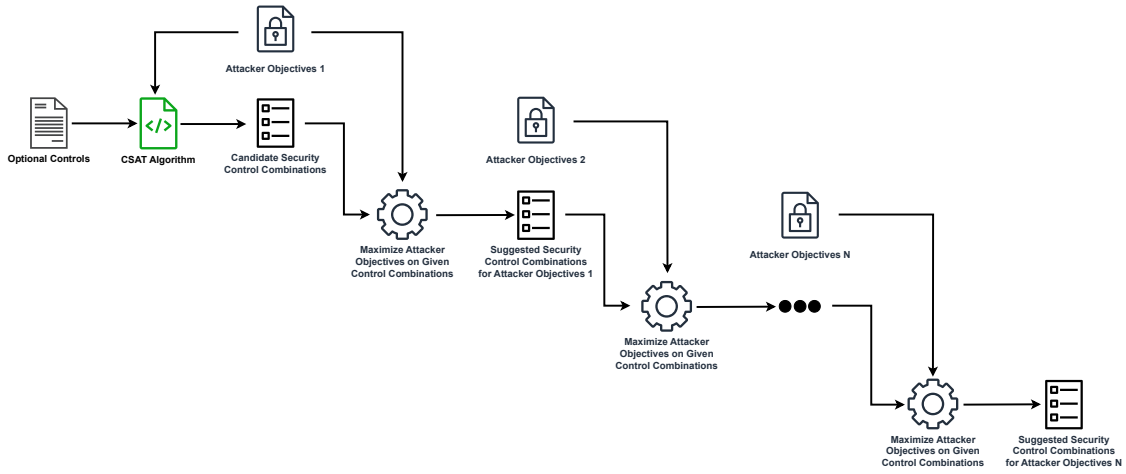
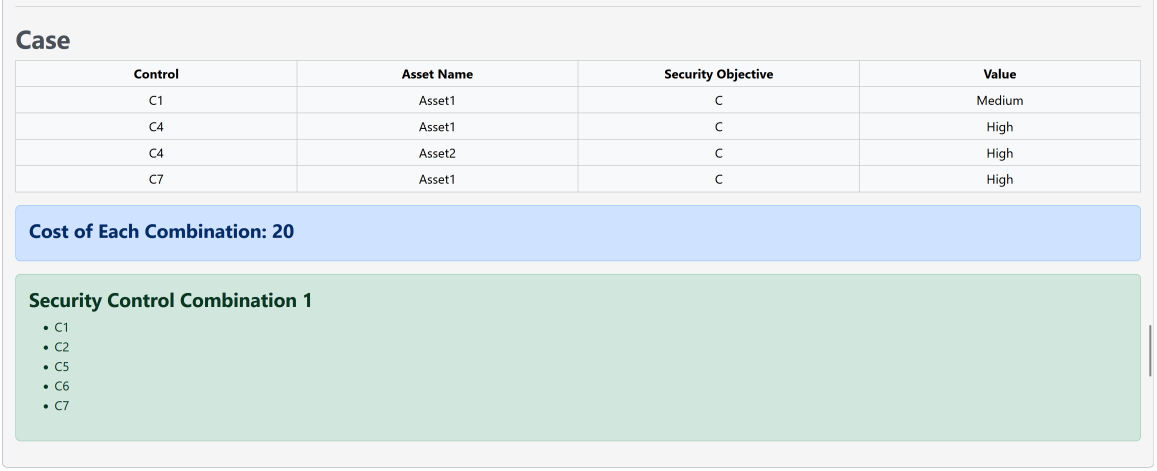


Figure 6. A visualization of the logic used by CSAT to find the suggested control combinations



Case

Control	Asset Name	Security Objective	Value
C1	Asset1	C	Medium
C4	Asset1	C	High
C4	Asset2	C	High
C7	Asset1	C	High

Cost of Each Combination: 20

Security Control Combination 1

- C1
- C2
- C5
- C6
- C7

Figure 7. Excerpt of a CSAT report

5. Illustrative Example

In this section, we demonstrate how the approach presented in Section 3 could be applied to support the control selection activity for a large illustrative system. The system that will be used for this example is a fictional Internet of Things (IoT) system for the Canadian military called *Ravenclaw*. Its architecture has already been designed and can be seen in Figure 8. Note that the arrows in this picture refer to the directions in which data flows in the system.

In this system, sensors collect battlefield information to be reported back for intelligence analysis. This data is processed with the processing system and stored in the database shortly after. An intelligence officer can then view the collected data from an analyst interface. Intelligence officers can also create and edit analysis reports from this interface, which are stored to the database when saved. Lastly, based on the analyses made, intelligence officers can send commands from this interface to UAVs in the battlefield.

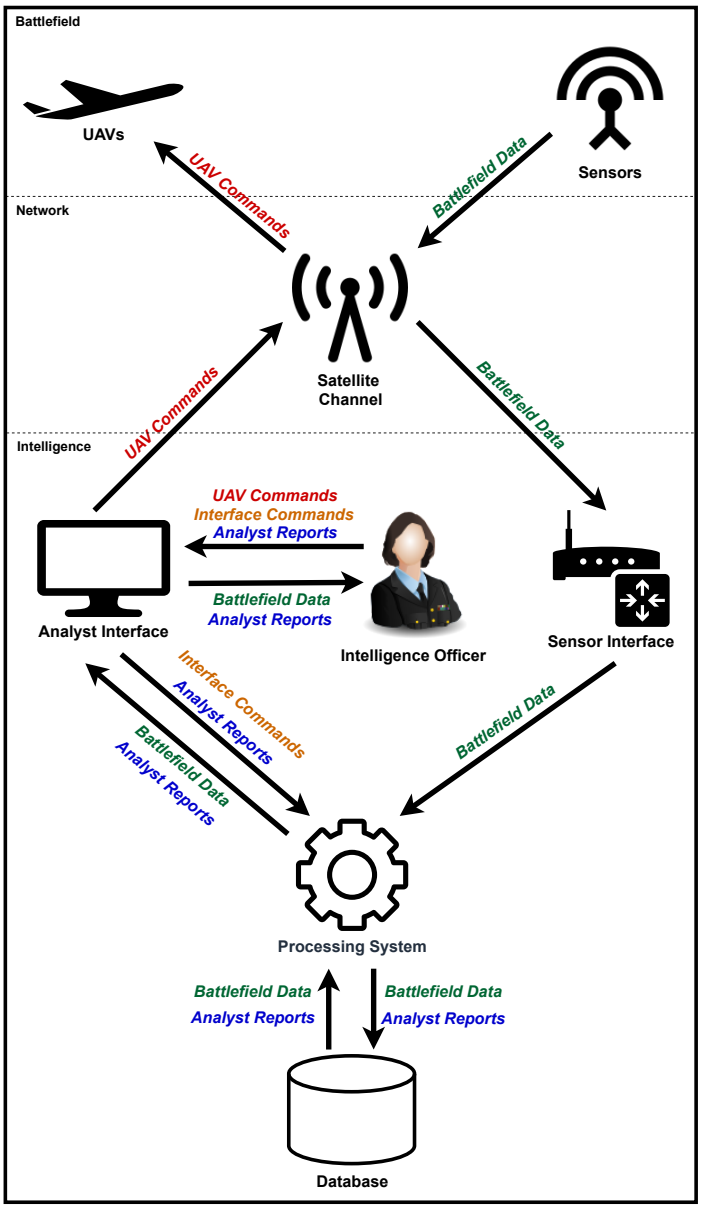


FIGURE 8. An overview of the *Ravenclaw* system architecture

The security analyst will apply the proposed approach to the *Ravenclaw* system as described in the following sections.

5.1. **Identify Applicable Atomic Controls.** From the architecture provided in Figure 8, we identify seven assets: the sensors, the UAVs, the satellite channel, the sensor interface, the analyst interface, the processing system, and the database. Note that although there are

many sensors and UAVs, the sensors and UAVs will each be considered as a singular asset. This is because each sensor in the system is identical. The same can be said for the UAVs.

Because *Ravenclaw* is a Canadian government system, the analyst selects controls from the ITSG-33 control catalogue[2]. To comply with departmental requirements, it was decided by the organization that the following controls must be present in the system:

- *AC-1: Access Control Policy and Procedures*
- *AC-3: Access Enforcement*
- *AC-4: Information Flow Enforcement*
- *AC-17: Remote Access*
- *AC-23: Data Mining Protection*
- *AU-1: Audit and Accountability Policy and Procedures*
- *IA-3: Device Identification and Authentication*
- *SC-8: Transmission Confidentiality and Integrity*

As for the identified threats, the organization has provided to the security analyst the identified threats to each system asset using a STRIDE threat model. The STRIDE model for the sensors can be seen in Table 3. To reduce cluttering this article, the STRIDE threat models for the other assets in the system can be viewed in the CSAT code repository linked above. Note that the organization has determined that the three security objectives of concern in the system are confidentiality, integrity, and availability.

Table 3. STRIDE threat model for the *Ravenclaw* sensors

STRIDE Category	**Threats**	**Security Objectives Violated**
Spoofing	• Data request from impersonating body	• Confidentiality
Tampering	• Modification of device • Outgoing data is modified	• Integrity
Repudiation	N/A	N/A
Information Disclosure	• Outgoing data is read	• Confidentiality
Denial of Service	• Physical destruction of device	• Availability
Elevation of Privilege	• Improper SSH access	• Confidentiality • Integrity

With the threats provided, the analyst can use the control catalogue to find relevant controls, and must ensure that each mandatory control is included at least once across all assets. All other applicable controls identified are therefore optional. The applicable controls for the sensors can be seen in Table 4. The applicable controls for the other assets are found in the CSAT code repository.

[2]ITSG-33 is the standard control catalogue to assist security practitioners in their efforts to protect information systems in compliance with applicable Government of Canada legislation, policies, directives, and standards [Gov14].

TABLE 4. Applicable atomic controls for the *Ravenclaw* sensors

STRIDE Category	**Threats**	**ITSG-33 Controls**
Spoofing	• Data request from impersonating body	• *AC-4: Information Flow Enforcement* • *AC-7: Unsuccessful Login Attempts* • *AC-17: Remote Access* • *AC-18: Wireless Access* • *IA-2: Identification and Authentication (Organizational Users)* • *SC-40: Wireless Link Protection*
Tampering	• Modification of device • Outgoing data is modified	• *PE-1: Physical and Environmental Protection Policy and Procedures* • *PE-3: Physical Access Control* • *PE-6: Monitoring Physical Access* • *PE-20: Asset Monitoring and Tracking* • *SA-18: Tamper Resistance and Detection* • *SC-8: Transmission Confidentiality and Integrity* • *SI-7: Software, Firmware, and Information Integrity*
Repudiation	N/A	N/A
Information Disclosure	• Outgoing data is read	• *AU-13: Monitoring for Information Disclosure* • *SC-8: Transmission Confidentiality and Integrity* • *SC-12: Cryptographic Key Establishment and Management*
Denial of Service	• Physical destruction of device	• *PE-1: Physical and Environmental Protection Policy and Procedures* • *PE-2: Physical Access Authorizations* • *PE-3: Physical Access Control* • *PE-6: Monitoring Physical Access* • *PE-20: Asset Monitoring and Tracking* • *SC-40: Wireless Link Protection*
Elevation of Privilege	• Improper SSH access	• *AC-6: Least Privilege* • *AC-7: Unsuccessful Login Attempts* • *AC-12: Session Termination* • *AC-24: Access Control Decisions* • *AU-2: Auditable Events* • *AU-8: Time Stamps* • *CA-8: Penetration Testing* • *IA-3: Device Identification and Authentication*

5.2. **Assign Effectiveness to Atomic Controls.** The analyst must now assign the effectiveness values for each identified applicable atomic control at mitigating the threats and protecting the security objectives listed in Table 3. The analyst has elected to assign qualitative ratings for the effectiveness of each atomic control that are mapped to quantitative

values. These values, *None* (0.0), *Low* (0.2), *Medium* (0.5), *High* (0.8), and *Very High* (0.9), align with the effectiveness values used for CSAT.

With the applicable controls gathered, the atomic payoff matrix can be created. Since there are many controls and assets, the atomic payoff matrix shall be broken into smaller matrices for readability; one for each asset. Using the metrics previously described, the analyst creates the atomic payoff matrix for the sensors. This matrix can be seen in Table 5 The matrices for the other assets are found in the CSAT code repository.

Table 5. Atomic payoff matrix for the *Ravenclaw* sensors

	Sensors		
	C	*I*	*A*
AC-1: Access Control Policy and Procedures	None	None	None
AC-2: Account Management	None	None	None
AC-3: Access Enforcement	None	None	None
AC-4: Information Flow Enforcement	Medium	None	None
AC-5: Separation of Duties	None	None	None
AC-6: Least Privilege	High	Medium	None
AC-7: Unsuccessful Login Attempts	None	Medium	Low
AC-8: System Use Notification	None	None	None
AC-9: Previous Logon (Access) Notification	None	None	None
AC-12: Session Termination	None	Low	None
AC-17: Remote Access	None	Low	None
AC-18: Wireless Access	None	Medium	None
AC-23: Data Mining Protection	None	None	None
AC-24: Access Control Decisions	Low	Low	None
AT-2: Security Awareness	None	None	None
AU-1: Audit and Accountability Policy and Procedures	None	None	None
AU-2: Auditable Events	None	Low,Medium	None
AU-3: Content of Audit Records	None	None	None
AU-6: Audit Review, Analysis, and Reporting	None	None	None
AU-8: Time Stamps	None	Low	None
AU-9: Protection of Audit Information	None	None	None
AU-10: Non-Repudiation	None	None	None
AU-12: Audit Generation	None	None	None
AU-13: Monitoring for Information Disclosure	Medium	None	None
AU-14: Session Audit	None	None	None
CA-8: Penetration Testing	None	Low	None
IA-1: Identification and Authentication Policy and Procedures	None	None	None
IA-2: Identification and Authentication (Organizational Users)	None	VeryHigh	None
IA-3: Device Identification and Authentication	None	Medium	None
IA-10: Adaptive Identification and Authentication	None	None	None
PE-1: Physical and Environmental Protection Policy and Procedures	None	None	None
PE-2: Physical Access Authorizations	None	Low	Low
PE-3: Physical Access Control	None	Medium	High
PE-6: Monitoring Physical Access	None	Medium	High
PE-20: Asset Monitoring and Tracking	None	Low	Low
SA-18: Tamper Resistance and Detection	None	High	None
SC-5: Denial of Service Protection	None	None	None
SC-7: Boundary Protection	None	None	None
SC-8: Transmission Confidentiality and Integrity	VeryHigh	High	None
SC-12: Cryptographic Key Establishment and Management	High	None	None
SC-40: Wireless Link Protection	Medium	Medium	Medium
SI-4: Information System Monitoring	None	None	None
SI-5: Security Alerts, Advisories, and Directives	None	None	None

TABLE 5. Atomic payoff matrix for the *Ravenclaw* sensors (continued)

	Sensors		
	C	*I*	*A*
SI-6: Security Functional Verification	None	None	None
SI-7: Software, Firmware, and Information Integrity	None	High	None
SI-10: Information Input Validation	None	None	None
SI-15: Information Input Filtering	None	None	None

5.3. **Assign Cost to Atomic Controls.** The cost of each atomic control can be seen in Table 6. Note that the costs were assigned in consultation with the system designers. Also note that the costs below are for all the controls identified, not just those of the sensors.

For this example, each cost includes the effort required for an atomic control to be implemented in *Ravenclaw*. This is why policy controls have a cost associated to them (as they require an effort to develop). Note that the physical controls have a much higher cost than the other controls. This is because physical security will be more expensive as it requires physical measures to be purchased and installed (versus cyber controls which only require a software implementation). Also note that the *AT-2: Security Awareness* control is expensive as well as much effort is required to develop a training course and provide this training to employees. Lastly, note that the organization has allocated a total budget of $B = \$950,000$.

5.4. **Specify and Generate Valid Control Combinations.** Next, the analyst must determine the valid security control combinations that could be considered for the system. To do this, they use security control algebra to specify the security control family from the mandatory and optional atomic controls identified in the previous steps. Denoting the security control family as F, the security control family for this case study is specified as the following security control algebra term.

$$
\begin{aligned}
F \;=\; & \mathit{AC\text{-}1} \odot \mathit{AC\text{-}3} \odot \mathit{AC\text{-}4} \odot \mathit{AC\text{-}17} \odot \mathit{AC\text{-}23} \odot \mathit{AU\text{-}1} \odot \mathit{IA\text{-}3} \odot \mathit{SC\text{-}8} \odot \\
& \mathit{opt}\big[\mathit{AC\text{-}2}, \mathit{AC\text{-}5}, \mathit{AC\text{-}6}, \mathit{AC\text{-}7}, \mathit{AC\text{-}8}, \mathit{AC\text{-}9}, \mathit{AC\text{-}12}, \mathit{AC\text{-}18}, \\
& \mathit{AC\text{-}24}, \mathit{AT\text{-}2}, \mathit{AU\text{-}2}, \mathit{AU\text{-}3}, \mathit{AU\text{-}6}, \mathit{AU\text{-}8}, \mathit{AU\text{-}9}, \mathit{AU\text{-}10}, \\
& \mathit{AU\text{-}12}, \mathit{AU\text{-}13}, \mathit{AU\text{-}14}, \mathit{CA\text{-}8}, \mathit{IA\text{-}1}, \mathit{IA\text{-}2}, \mathit{IA\text{-}10}, \mathit{PE\text{-}1}, \\
& \mathit{PE\text{-}2}, \mathit{PE\text{-}3}, \mathit{PE\text{-}6}, \mathit{PE\text{-}20}, \mathit{SA\text{-}18}, \mathit{SC\text{-}5}, \mathit{SC\text{-}7}, \mathit{SC\text{-}12}, \\
& \mathit{SC\text{-}40}, \mathit{SI\text{-}4}, \mathit{SI\text{-}5}, \mathit{SI\text{-}6}, \mathit{SI\text{-}7}, \mathit{SI\text{-}10}, \mathit{SI\text{-}15}\big]
\end{aligned}
$$

The control names were omitted for clarity in the security control family above. For this case study, let's suppose the security analyst and organization have determined the dependencies shown in Table 7. This table presents the dependencies as requirement relations. It also presents the reasoning for which each dependency exists.

At this point in the approach, we would typically expand the security control family F to find all possible security control combinations, find those which respect the requirement relations, and filter the remaining combinations by the budget rule. This would be reasonable for small examples. However, it is not feasible to expand the security control family above as the number of optional controls is very large; specifically, there are 39 optional controls. This security control family therefore generates 2^{39} possible security control combinations

Table 6. Atomic control costs for *Ravenclaw*

Control	Cost
AC-1: Access Control Policy and Procedures	$20,000
AC-2: Account Management	$60,000
AC-3: Access Enforcement	$40,000
AC-4: Information Flow Enforcement	$30,000
AC-5: Separation of Duties	$10,000
AC-6: Least Privilege	$30,000
AC-7: Unsuccessful Login Attempts	$10,000
AC-8: System Use Notification	$20,000
AC-9: Previous Logon (Access) Notification	$10,000
AC-12: Session Termination	$30,000
AC-17: Remote Access	$50,000
AC-18: Wireless Access	$30,000
AC-23: Data Mining Protection	$50,000
AC-24: Access Control Decisions	$20,000
AT-2: Security Awareness	$200,000
AU-1: Audit and Accountability Policy and Procedures	$20,000
AU-2: Auditable Events	$40,000
AU-3: Content of Audit Records	$30,000
AU-6: Audit Review, Analysis, and Reporting	$20,000
AU-8: Time Stamps	$10,000
AU-9: Protection of Audit Information	$30,000
AU-10: Non-Repudiation	$30,000
AU-12: Audit Generation	$30,000
AU-13: Monitoring for Information Disclosure	$60,000
AU-14: Session Audit	$40,000
CA-8: Penetration Testing	$90,000
IA-1: Identification and Authentication Policy and Procedures	$30,000
IA-2: Identification and Authentication (Organizational Users)	$30,000
IA-3: Device Identification and Authentication	$30,000
IA-10: Adaptive Identification and Authentication	$40,000
PE-1: Physical and Environmental Protection Policy and Procedures	$10,000
PE-2: Physical Access Authorizations	$40,000
PE-3: Physical Access Control	$50,000
PE-6: Monitoring Physical Access	$80,000
PE-20: Asset Monitoring and Tracking	$100,000
SA-18: Tamper Resistance and Detection	$100,000
SC-5: Denial of Service Protection	$40,000
SC-7: Boundary Protection	$60,000
SC-8: Transmission Confidentiality and Integrity	$30,000
SC-12: Cryptographic Key Establishment and Management	$30,000
SC-40: Wireless Link Protection	$50,000
SI-4: Information System Monitoring	$60,000
SI-5: Security Alerts, Advisories, and Directives	$20,000
SI-6: Security Functional Verification	$70,000
SI-7: Software, Firmware, and Information Integrity	$40,000
SI-10: Information Input Validation	$20,000
SI-15: Information Input Filtering	$20,000

and cannot be found manually. Similarly, the requirement relations and budget rule cannot be applied manually as well. As a result, we will use CSAT to complete this illustrative example.

TABLE 7. Atomic control dependencies for *Ravenclaw*

Dependency	Reasoning for Dependency
AC-18 $\xrightarrow{F}$ *AC-17*	*AC-18* is a refinement of *AC-17*
AU-2 $\xrightarrow{F}$ *AU-1*	*AU-2* requires an audit policy (*AU-1*)
AU-3 $\xrightarrow{F}$ *AU-1*	*AU-3* requires an audit policy (*AU-1*)
AU-6 $\xrightarrow{F}$ *AU-1*	*AU-6* requires an audit policy (*AU-1*)
AU-8 $\xrightarrow{F}$ *AU-1*	*AU-8* requires an audit policy (*AU-1*)
AU-9 $\xrightarrow{F}$ *AU-1*	*AU-9* requires an audit policy (*AU-1*)
AU-10 $\xrightarrow{F}$ *AU-1*	*AU-10* requires an audit policy (*AU-1*)
AU-12 $\xrightarrow{F}$ *AU-1* · *AU-2* · *AU-3*	*AU-12* requires an audit policy (*AU-1*), and depends on auditable events and content (*AU-2* and *AU-3*)
AU-13 $\xrightarrow{F}$ *AU-1*	*AU-13* requires an audit policy (*AU-1*)
AU-14 $\xrightarrow{F}$ *AU-1*	*AU-14* requires an audit policy (*AU-1*)
IA-2 $\xrightarrow{F}$ *IA-1*	*IA-2* requires an authentication policy (*IA-1*)
IA-10 $\xrightarrow{F}$ *IA-2*	*IA-10* is an additional authentication mechanism to *IA-2*
PE-2 $\xrightarrow{F}$ *PE-1*	*PE-2* requires a physical protection policy (*PE-1*)
PE-3 $\xrightarrow{F}$ *PE-1*	*PE-3* requires a physical protection policy (*PE-1*)
PE-6 $\xrightarrow{F}$ *PE-1*	*PE-6* requires a physical protection policy (*PE-1*)
PE-20 $\xrightarrow{F}$ *PE-1*	*PE-20* requires a physical protection policy (*PE-1*)
SC-12 $\xrightarrow{F}$ *SC-8*	Cryptographic key establishment/management (*SC-12*) requires encryption to be employed (*SC-8*)

5.5. **Construct the Game Matrix.** Since the valid security control combinations could not be generated manually, and that these combinations represent the strategies of the security analyst in the game matrix, the game matrix cannot be constructed manually. However, to illustrate the general structure of this matrix, we have manually constructed a portion of this matrix in Table 8 using two randomly selected control combinations. Only the sensor asset is included in the matrix as the inclusion of all the assets would make it excessively wide. This again emphasizes the need to use CSAT to complete this example.

Table 8. Sample atomic payoff matrix for *Ravenclaw*

	Sensor		
	C	*I*	*A*
AC-1 ⊙ *AC-2* ⊙ *AC-3* ⊙ *AC-4* ⊙ *AC-5* ⊙ *AC-6* ⊙ *AC-7* ⊙ *AC-12* ⊙ *AC-17* ⊙ *AC-18* ⊙ *AC-23* ⊙ *AC-24* ⊙ *AU-1* ⊙ *AU-2* ⊙ *AU-3* ⊙ *AU-6* ⊙ *AU-8* ⊙ *AU-10* ⊙ *AU-12* ⊙ *IA-1* ⊙ *IA-2* ⊙ *IA-3* ⊙ *PE-1* ⊙ *PE-3* ⊙ *SC-5* ⊙ *SC-8* ⊙ *SC-40* ⊙ *SI-4* ⊙ *SI-7*	0.996	0.99999	0.920
AC-1 ⊙ *AC-2* ⊙ *AC-3* ⊙ *AC-4* ⊙ *AC-5* ⊙ *AC-6* ⊙ *AC-7*⊙ *AC-8* ⊙ *AC-12* ⊙ *AC-17* ⊙ *AC-18* ⊙ *AC-23* ⊙ *AC-24* ⊙ *AU-1*⊙ *AU-2* ⊙ *AU-3* ⊙ *AU-6* ⊙ *AU-8* ⊙ *AU-9* ⊙ *AU-13* ⊙ *AU-14* ⊙ *IA-1*⊙ *IA-2* ⊙ *IA-3* ⊙ *IA-10* ⊙ *SC-8* ⊙ *SI-4* ⊙ *SI-5* ⊙ *SI-10* ⊙ *SI-15*	0.996	0.99974	0.2
...	...	...	...

5.6. **Play the Game.** We use CSAT to play the game. As mentioned in Section 4, CSAT can find the suggested controls for a given attacker profile without having found all valid control combinations. The control specification file for this case study is publicly available in the CSAT project repository. Additionally, recall that the budget B for *Ravenclaw* is $950,000.

To play the game, we will consider two different scenarios. Each scenario presents a different possible attacker profile for this system, and aims to show how the suggested controls for the system change based on this profile. These scenarios and their outcomes are described in the following subsections.

5.6.1. *Scenario 1.* This scenario considers an attacker profile where the attacker has three ordered attacker objectives to target: the confidentiality, integrity and availability of the sensor interface and analyst interface, followed by the confidentiality, integrity and availability of the processing system, and lastly followed by the confidentiality, integrity and availability of the database. Using CSAT, we find that for all cases, there is but one suggested security control combination. This result has a cost of $940,000. The suggested security control combination is as follows.

$$\begin{aligned} \textit{Combo1.1} \quad = \quad & \textit{AC-1} \odot \textit{AC-2} \odot \textit{AC-3} \odot \textit{AC-4} \odot \textit{AC-5} \odot \textit{AC-6} \odot \textit{AC-7} \odot \textit{AC-9} \odot \\ & \textit{AC-12} \odot \textit{AC-17} \odot \textit{AC-18} \odot \textit{AC-23} \odot \textit{AC-24} \odot \textit{AU-1} \odot \textit{AU-2} \odot \\ & \textit{AU-3} \odot \textit{AU-8} \odot \textit{AU-10} \odot \textit{AU-12} \odot \textit{AU-14} \odot \textit{IA-1} \odot \textit{IA-2} \odot \textit{IA-3} \odot \\ & \textit{IA-10} \odot \textit{SC-5} \odot \textit{SC-8} \odot \textit{SC-12} \odot \textit{SI-4} \odot \textit{SI-7} \odot \textit{SI-10} \end{aligned}$$

This combination contains a variety of controls that protect the assets of concern. For example, *IA-2: Identification and Authentication (Organizational Users)* protects a multitude of security objectives across multiple assets, namely the confidentiality and integrity of the sensor interface, analyst interface, and database. Additionally, there are no physical controls in this combination which is logical as the expected attacker has no intention of targeting the physical assets of the system. Note this combination considers 22 of the 39 applicable optional controls. Without CSAT, determining that this combination best protects against

the expected attacker under the given constraints would have been infeasible. It took 10.659 seconds for CSAT to generate this result.

5.6.2. *Scenario 2.* This scenario considers an attacker profile where the attacker targets the integrity and availability of the sensors, UAVs, satellite channel, sensor interface, and analyst interface. Using CSAT, we find there are different suggested security control combinations due to the analyst uncertainty. Specifically, we find one combination in four cases when *AU-12: Audit Generation* has *High* effectiveness towards the integrity of the analyst interface (*Combo2.1*), and we find another in the other four cases when *AU-12* has *Medium* effectiveness towards the integrity of the analyst interface (*Combo2.2*). Note that *Combo2.1* has a cost of $930,000, and *Combo2.2* has a cost of $940,000. The suggested security control combinations are as follows.

$$
\begin{aligned}
\textit{Combo2.1} \;=\; & \textit{AC-1} \odot \textit{AC-2} \odot \textit{AC-3} \odot \textit{AC-4} \odot \textit{AC-5} \odot \textit{AC-6} \odot \textit{AC-7} \odot \textit{AC-12} \odot \\
& \textit{AC-17} \odot \textit{AC-18} \odot \textit{AC-23} \odot \textit{AC-24} \odot \textit{AU-1} \odot \textit{AU-2} \odot \textit{AU-3} \odot \\
& \textit{AU-6} \odot \textit{AU-8} \odot \textit{AU-10} \odot \textit{AU-12} \odot \textit{IA-1} \odot \textit{IA-2} \odot \textit{IA-3} \odot \textit{PE-1} \odot \\
& \textit{PE-3} \odot \textit{SC-5} \odot \textit{SC-8} \odot \textit{SC-40} \odot \textit{SI-4} \odot \textit{SI-7} \\
\textit{Combo2.2} \;=\; & \textit{AC-1} \odot \textit{AC-2} \odot \textit{AC-3} \odot \textit{AC-4} \odot \textit{AC-5} \odot \textit{AC-6} \odot \textit{AC-7} \odot \textit{AC-12} \odot \\
& \textit{AC-17} \odot \textit{AC-18} \odot \textit{AC-23} \odot \textit{AC-24} \odot \textit{AU-1} \odot \textit{AU-2} \odot \textit{AU-3} \odot \\
& \textit{AU-6} \odot \textit{AU-8} \odot \textit{AU-10} \odot \textit{IA-1} \odot \textit{IA-2} \odot \textit{IA-3} \odot \textit{IA-10} \odot \textit{PE-1} \odot \\
& \textit{PE-3} \odot \textit{SC-5} \odot \textit{SC-8} \odot \textit{SC-40} \odot \textit{SI-4} \odot \textit{SI-7}
\end{aligned}
$$

Both combinations differ by only one control; *Combo2.1* has *AU-12 Audit Generation* while *Combo2.2* has *IA-10 Adaptive Identification and Authentication.* Although both controls are functionally different, and neither protect availability, they are both helpful in protecting the integrity of the targeted assets (audits and behavioural analyses can detect unauthorized data modifications). Hence, regardless of this uncertainty, we may say that both security control combinations are adequate in protecting against the expected attacker profile. In this scenario, the suggested combinations each consider 21 of the 39 applicable optional controls which, like *Scenario 1*, would have been hard to identify without CSAT. Additionally, given that the uncertainty of an effectiveness value resulted in different suggested combinations for some of the cases, an analyst would have had to determine this manually by re-playing the game for each of the cases, which took CSAT just 16 minutes and 44 seconds to compute.

6. Discussion

Approaching security control selection as a game emphasizes the human element in deciding how to most effectively protect a system's assets under various considerations such as budgetary constraints. Selecting controls for a system is indeed a human-centric problem as the large number of potential controls to use from a control catalogue can be overwhelming and could lead to many mistakes in the chosen combination of security controls selected for the system. To expand on this point, selecting security controls exclusively on technical considerations while overlooking attacker behaviours is a fundamentally flawed approach in addressing this issue, as ultimately, it is humans which are conducting attacks. Given that the proposed approach emphasizes the need to reflect on potential attacker behaviours, it prioritizes the human-centric aspect to find its solutions. Additionally, viewing this problem

as a game captures the opposing dynamics of the attacker and analyst, aligning with the real-world motivations of both actors.

Unfortunately, a limitation with many existing game-theoretic approaches from addressing cybersecurity challenges is their heavy reliance on assumptions. Specifically, game theory depends on assumptions on the players, such as the players knowing every strategy available to them, knowing the probabilities of every move, and knowing the payoff functions [Owe15]. Compared to existing works, the proposed approach can be used practically as it does not rely on assigning probabilities for the likelihood of attacks succeeding. In fact, such probabilities would generally be unattainable in practice without sufficient historical data. Instead, the approach focuses on general assumptions about which security objectives could be violated by the attacker. Security analysts cannot realistically predict exact probabilities of attack, but they can make informed assumptions regarding which system components might be more attractive to attackers. These considerations ensure that the proposed approach is systematic, repeatable, and realistic, thereby minimizing the influence of human bias on the results of the game and eliminating many of the required assumptions with existing game-theoretic approaches.

Although CSAT can automate the approach, finding the suggested security controls remains an NP-complete problem. Therefore, CSAT cannot be expected to generate its solutions rapidly. However, a performance analysis was conducted for a typical[3] usage of CSAT and revealed that the tool has an upper limit capacity of 50 optional controls. With this number of optional controls, results were generated in approximately one hour. This running time is acceptable given that 50 optional controls is a significantly large number of inputs and that the results of CSAT are not needed in real-time. To support this claim, considering 50 controls from ITSG-33 is equivalent to considering 17% of the controls from this catalogue (given that ITSG-33 has approximately 300 security controls [Gov14]). To further support this claim, despite having identified 39 optional controls in Section 5, the results for both scenarios were generated in a reasonable amount of time. The approach can therefore be used practically towards large systems using CSAT.

7. Conclusions and Future Work

Ensuring effective security controls are selected for a system can greatly impact its security. In this work, a game-theoretic approach to security control selection is proposed in which a game is played by a security analyst to determine security controls which best mitigate expected attacker profiles. The suggested controls can help make a security analyst feel more confident in their decision to implement some controls over others. To demonstrate the approach's scalability for large systems, the approach was applied to a large fictional Canadian military system using a previously proposed tool [LJ24b]. Despite 39 optional controls being considered in this example, results were generated in reasonable time, demonstrating the scalability of the approach when automated and applied towards large systems.

In future work, we aim to investigate the outcomes of the game when played from the attacker perspective. As the game matrix used in this approach is known only to the security analyst, it is not realistic to play the game as an attacker. However, if the attacker were to gain access to the game matrix and wished to play the game, they would make assumptions about possible security analyst behaviours and create *defender profiles*, representing security

[3]A typical usage of CSAT refers to a scenario in which the optional controls considered include a balanced mix of controls with varying effectiveness and cost.

control combinations that the security analyst is likely to choose for the system. The attacker could then play the game from the opposite perspective of the analyst, by selecting the security objectives that are least protected against security control combinations that might be chosen. While playing the game from this perspective may not be realistic, it could potentially identify weaknesses in certain security controls or pinpoint security objectives that are more likely to be successfully violated by an attacker (i.e., weaker links in the system). Additionally, for future work, changes to CSAT, such as allowing for other formats than `.xlsx` for inputting control information, or improving its general look-and-feel, could be made to improve the usability of the tool.

Acknowledgment

This research is funded in part by the Human-Centric Cybersecurity Partnership under the SSHRC Partnership Grants program.

References

[AR18] Luís Almeida and Ana Respício. Decision support for selecting information security controls. *Journal of Decision Systems*, 27:173–180, 2018. doi:10.1080/12460125.2018.1468177.

[BSS+20] Seifeddine Bettaieb, Seung Yeob Shin, Mehrdad Sabetzadeh, Lionel C. Briand, Michael Garceau, and Antoine Meyers. Using machine learning to assist with the selection of security controls during security assessment. *Empirical Software Engineering*, 25(4):2550–2582, 2020. doi:10.1007/s10664-020-09814-x.

[CEL+20] Jennifer Cawthra, Michael Ekstrom, Lauren Lusty, Julian Sexton, and John Sweetnam. Data integrity: Detecting and responding to ransomware and other destructive events. Special Publication (NIST SP) 1800-26, National Institute of Standards and Technology, December 2020. doi:10.6028/NIST.SP.1800-26.

[Cen21] Center for Information Security. CIS Critical Security Controls – Version 8. https://www.cisecurity.org/controls/v8 [Accessed: 2024-06-21], May 2021.

[DPRW07] Rinku Dewri, Nayot Poolsappasit, Indrajit Ray, and Darrell Whitley. 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*, pages 204–213, New York, NY, USA, 2007. Association for Computing Machinery. doi:10.1145/1315245.1315272.

[Dra] Victoria Drake. Threat Modeling. https://owasp.org/www-community/Threat_Modeling [Accessed: 2023-12-11].

[FCHJ05] Martin S. Feather, Steven L. Cornford, Kenneth A. Hicks, and Kenneth R. Johnson:. Applications of tool support for risk-informed requirements reasoning. https://www.researchgate.net/publication/220403935_Applications_of_tool_support_for_risk-informed_requirements_reasoning [Accessed: 2024-06-21], 2005.

[FW20] Barbara Fila and Wojciech Wideł. Exploiting attack–defense trees to find an optimal set of countermeasures. In *2020 IEEE 33rd computer security foundations symposium (CSF)*, pages 395–410. IEEE, 2020.

[Gov14] Government of Canada. IT Security Risk Management: A Lifecycle Approach – Security Control Catalogue. https://www.cisecurity.org/controls/v8 [Accessed: 2024-06-21], December 2014.

[HKM11] Peter Höfner, Ridha Khedri, and Bernhard Möller. An algebra of product families. *Software and Systems Modeling*, 10(2):161–182, 2011. doi:10.1007/s10270-009-0127-2.

[Int18] International Organization for Standardization. ISO/IEC 31000:2018 Risk Management – Guidelines. https://www.iso.org/standard/65694.html [Accessed: 2024-06-21], February 2018.

[Int22a] International Organization for Standardization. ISO/IEC 27002:2022 Information security, cybersecurity and privacy protection – Information security controls. https://www.iso.org/standard/75652.html [Accessed: 2024-06-21], February 2022.

[Int22b] International Organization for Standardization. ISO/IEC 27005:2022 Information security, cybersecurity and privacy protection – Guidance on managing information security risks. https://www.iso.org/standard/80585.html [Accessed: 2023-12-11], October 2022.

[Joi18] Joint Task Force Interagency Working Group. Risk management framework for information systems and organizations: A system life cycle approach for security and privacy. Special Publication (NIST SP) 800-37 Revision 2, National Institute of Standards and Technology, December 2018. doi:10.6028/NIST.SP.800-37r2.

[Joi20a] Joint Task Force Interagency Working Group. Control baselines for information systems and organizations. Special Publication (NIST SP) 800-53B, National Institute of Standards and Technology, September 2020. doi:10.6028/nist.sp.800-53b.

[Joi20b] Joint Task Force Interagency Working Group. Security and privacy controls for information systems and organizations. Special Publication (NIST SP) 800-53 Revision 5, National Institute of Standards and Technology, September 2020. doi:10.6028/NIST.SP.800-53r5.

[JYFF16] Xiang Ji, HuiQun Yu, GuiSheng Fan, and WenHao Fu. Attack-defense trees based cyber security analysis for cpss. In *2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pages 693–698. IEEE, 2016.

[KEG+16] Elmar Kiesling, Andreas Ekelhart, Bernhard Grill, Christine Strauss, and Christian Stummer. Selecting security control portfolios: a multi-objective simulation-optimization approach. *EURO Journal on Decision Processes*, 4(1-2):85–117, 2016. doi:10.1007/s40070-016-0055-7.

[KMRS14] Barbara Kordy, Sjouke Mauw, Saša Radomirović, and Patrick Schweitzer. Attack–defense trees. *Journal of logic and computation*, 24(1):55–87, 2014.

[KS20] Peter Kaloroumakis and Michael Smith. Toward a knowledge graph of cybersecurity countermeasures. https://apps.dtic.mil/sti/citations/AD1156977 [Accessed: 2024-06-21], April 2020.

[KV01] Bernhard Korte and Jens Vygen. Combinatorial optimization: Theory and algorithms. *Computers & Mathematics with Applications*, 41(3):538, 2001.

[LJ24a] Dylan Léveillé and Jason Jaskolka. A game-theoretic approach for security control selection. In *15th International Symposium on Games, Automata, Logics and Formal Verification*, volume 409 of *GandALF 2024*, pages 103–119, Reykjavic, Iceland, 2024. Electronic Proceedings in Theoretical Computer Science.

[LJ24b] Dylan Léveillé and Jason Jaskolka. A tool for enabling scalable automation in security control selection. In *17th International Symposium on Foundations & Practice of Security*, FPS 2024, pages 1–16, Montreal, Canada, 2024. Springer.

[LZ11] Qixu Liu and Yuqing Zhang. VRSS: A new system for rating and scoring vulnerabilities. *Computer Communications*, 34:264–273, 2011. doi:10.1016/j.comcom.2010.04.006.

[Mic22] Microsoft. Microsoft threat modeling tool – threats. https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats [Accessed: 2024-06-21], 2022.

[MSR07] Peter Mell, Karen Scarfone, and Sasha Romanosky. The common vulnerability scoring system (CVSS) and its applicability to federal agency systems. NIST Interagency Report 7435, National Institute of Standards and Technology, August 2007. doi:10.6028/NIST.IR.7435.

[Mur16] Murugiah Souppaya and Karen Scarfone. Guide to data-centric system threat modeling. https://csrc.nist.gov/pubs/sp/800/154/ipd [Accessed: 2024-06-21], March 2016.

[Nat20] National Institute of Standards and Technology. The NIST privacy framework: A tool for improving privacy through enterprise risk management. Cybersecurity White Papers (CSWP) 10, National Institute of Standards and Technology, January 2020. doi:10.6028/nist.cswp.10.

[Nat24] National Institute of Standards and Technology. The NIST cybersecurity framework (CSF) 2.0. Cybersecurity White Papers (CSWP) 29, National Institute of Standards and Technology, February 2024. doi:10.6028/NIST.CSWP.29.

[Num24] NumFOCUS, Inc. pandas - python data analysis library. https://pandas.pydata.org/ [Accessed: 2024-08-29], 2024.

[Owe15] Guillermo Owen. Game theory. In James D. Wright, editor, *International Encyclopedia of the Social & Behavioral Sciences (Second Edition)*, pages 573–581. Elsevier, Oxford, second edition edition, 2015. doi:10.1016/B978-0-08-097086-8.43045-X.

[PH20] Jun Young Park and Eui Nam Huh. A cost-optimization scheme using security vulnerability measurement for efficient security enhancement. *Journal of Information Processing Systems*, 16(1):61–82, 2020. doi:10.3745/JIPS.02.0128.

[RGJ23] Quentin Rouland, Stojanche Gjorcheski, and Jason Jaskolka. Eliciting a security architecture requirements baseline from standards and regulations. In *2023 IEEE 31st International Requirements Engineering Conference Workshops*, REW, pages 224–229, Hannover, Germany, 2023. doi:10.1109/rew57809.2023.00045.

[RPG+21] Ron Ross, Victoria Pillitteri, Richard Graubart, Deborah Bodeau, and Rosalie Mcquaid. Developing cyber-resilient systems: A systems security engineering approach. Special Publication (NIST SP) 800-160, Volume 2 Revision 1, National Institute of Standards and Technology, December 2021. doi:10.6028/NIST.SP.800-160v2r1.

[SAZ21] Adam Schmidt, Laura A. Albert, and Kaiyue Zheng. Risk management for cyber-infrastructure protection: A bi-objective integer programming approach. *Reliability Engineering and System Safety*, 205, 2021. doi:10.1016/j.ress.2020.107093.

[SSGG15] Maryam Shahpasand, Mehdi Shajari, Seyed Alireza Hashemi Golpaygani, and Hoda Ghavamipoor. A comprehensive security control selection model for inter-dependent organizational assets structure. *Information and Computer Security*, 23(2):218–242, 2015.

[Str93] Philip D. Straffin. *Game Theory and Strategy*. The Mathematical Association of America, second edition, 1993.

[SZV16] R. Sarala, G. Zayaraz, and V. Vijayalakshmi. Optimal selection of security countermeasures for effective information security. *Advances in Intelligent Systems and Computing*, 398:345–353, 2016. doi:10.1007/978-81-322-2674-1_33.

[TCG+21] Maria Tsiodra, Michail Chronopoulos, Matthias Ghering, Eirini Karapistoli, Neofytos Gerosavva, and Nicolas Kylilis. The SPIDER cyber security investment component (CIC). *Proceedings of the 2021 IEEE International Conference on Cyber Security and Resilience, CSR 2021*, pages 415–421, 2021. doi:10.1109/CSR51186.2021.9527924.

[UM15] Tony UcedaVélez and Marco M. Morana. *Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis*. John Wiley & Sons, first edition, 2015. doi:10.1002/9781118988374.

[UYMM21] Ganbayar Uuganbayar, Artsiom Yautsiukhin, Fabio Martinelli, and Fabio Massacci. Optimisation of cyber insurance coverage with selection of cost effective security controls. *Computers and Security*, 101, 2021. doi:10.1016/j.cose.2020.102121.

[YBFEV15] Iryna Yevseyeva, Vitor Basto-Fernandes, Michael Emmerich, and Aad Van Moorsel. Selecting optimal subset of security controls. *Procedia Computer Science*, 64:1035–1042, 2015. doi:10.1016/j.procs.2015.08.625.