StEVe: A Rational Verification Tool for Stackelberg Security Games

Surasak Phetmanee$^{1[0000-0002-8913-1124]}$, Michele Sevegnani$^{1[0000-0001-6773-9481]}$, and Oana Andrei$^{1[0000-0002-1306-0219]}$

School of Computing Science, University of Glasgow, Glasgow G12 8RZ, UK
`{surasak.phetmanee, michele.sevegnani, oana.andrei}@glasgow.ac.uk`

Abstract. We present StEVe, a prototype tool modelling Stackelberg Security Games (SSGs) and employing rational verification based on bespoke Stackelberg equilibrium computation. StEVe automatically extracts technical details from public vulnerability databases, transforming these into Attack Defence Trees and then into SSG models. By using the temporal logic rPATL, the tool enables the synthesis of optimal defence strategies through Stackelberg equilibrium analysis, which is implemented as a PRISM-games extension. Preliminary results demonstrate StEVe's ability to model and counteract cyber threats, reducing potential damages and financial losses.

Keywords: Rational Verification · Security Games · Temporal Logic

1 Introduction

Stackelberg Security Games (SSGs) [16,17], which utilise the concept of Stackelberg equilibrium for their solution, have proven effective for resource allocation in security contexts. In these games, the leader (typically a defender) commits to a strategy first, and the follower (the attacker) observes this strategy and responds optimally. This framework is particularly relevant in cybersecurity, where defenders must allocate finite security resources to protect multiple targets, and attackers choose targets based on the observed security measures.

To determine the optimal strategy for defenders, several factors need to be considered, including the value of the targets, the potential impact of a breach, and the evolving nature of cyber threats. One promising approach to solving this problem is Rational Verification (RV) [18], which verifies agent systems by assuming that agents act rationally to pursue their preferences. RV checks if a temporal logic formula is satisfied in some or all game equilibria, ensuring rational behaviour in the system. Recent works [1,8,11] have demonstrated the effectiveness of RV in multiple agent systems. For example, Aslanyan et al. [2] present quantitative verification methods for cybersecurity scenarios, illustrating how RV can be applied to cybersecurity. While tools like EVE [5] and PRISM-games [9] already support several types of equilibria, expanding their capabilities to support Stackelberg equilibria would allow their applicability to a wider set of cybersecurity scenarios, particularly those involving sequential decision making, where defenders commit to strategies in advance and attackers respond after observing these strategies.

To bridge this gap, we propose a method for computing the Stackelberg equilibrium in SSGs and introduce SᴛEVᴇ (**St**ackelberg Security Games and **E**quilibrium **Ve**rification), a tool that integrates game theory and formal verification for rationality in cybersecurity. SᴛEVᴇ enables defenders to compute optimal resource allocation strategies and verify rational attacker behaviour. It transforms data from public vulnerability databases (such as CVEs from NIST's National Vulnerability Database) into SSG models, automating the process of generating and verifying defensive strategies using rational verification and temporal logic. Our work also extends the PRISM-games framework to support Stackelberg equilibrium computation in cybersecurity scenarios.

2 Background

SᴛEVᴇ's implementation is based on the concepts introduced in this section.

Attack Defence Trees (ADTs) [6] are graphical models used to express potential security threats and their countermeasures. We restrict the ADT models to one level of child nodes and construct them solely from the perspective of a defender, which we denote as ADT(1). This is sufficient to express SSGs. The syntax of ADT(1) comprises four terms: $c^d(b',\ b)$; $c^d(b',\ f(b_1,...,b_k))$; $c^d(f'(b'_1,...,b'_{k'}),\ b)$; $c^d(f'(b'_1,...,b'_{k'}),\ f(b_1,...,b_k))$, where a represents an attacker and d is a defender, $f \in \{\vee^a, \wedge^a\}$, $f' \in \{\vee^d, \wedge^d\}$, $b,\ b_i \in \mathbb{B}^a$ for $1 \leq i \leq k$ and $k \geq 2$, $b',\ b'_j \in \mathbb{B}^d$ for $1 \leq j \leq k'$ and some $k' \geq 2$, and $\mathbb{B}^d, \mathbb{B}^a \subset A$ are disjoints sets of actions for the defender and the attacker, respectively. For example, an ADT(1) for ensuring data confidentiality against employee attacks can be represented as: $c^d(\mathsf{firewall},\ \wedge^a(\mathsf{send\_vurs},\ \mathsf{get\_password}))$, where $\wedge$ represents a conjunction (both actions must occur) and $\vee$ represents a disjunction (either action can occur). The defensive action is firewall, while the attacker's actions include send_vurs and get_password. This indicates that the defender can protect the system by investing in high performance firewalls, but the system remains vulnerable to a coordinated attack involving both sending a virus and obtaining a password.

An *extensive form game* [10] $\mathcal{G}$ is a tuple $(N, A, H, Z, \chi, \pi, \gamma, u)$ where $N = \{1, ..., n\}$ is a set of players, $n \in \mathbb{N}$, $n \geq 2$, A is a finite set of actions, H is a set of non-terminal history (or choice) nodes, Z is a set of terminal history nodes disjoint from H, i.e. $H \cap Z = \varnothing$, $\chi : H \to 2^A$ maps each non-terminal history node to its possible actions, $\pi : H \to N$ is the player function, which assigns to each non-terminal history node a player $i \in N$ who chooses an action at that node, $\gamma : H \times A \to H \cup Z$ is the successor function mapping a choice node and an action to a new node, either terminal or non-terminal, and $u = (u_1, ..., u_n)$ is a profile of utility functions, where $u_i : Z \to \mathbb{R}$ is a utility function for player i that assigns a real-valued utility to each terminal node.

A *Stackelberg Security Game (SSG)* [17] is a tuple $(\mathcal{G}, AP, L)$ where $\mathcal{G}$ is a two-player (i.e., $N = \{1, 2\}$) extensive form game, AP is a set of atomic propositions, and $L : H \cup Z \rightarrow 2^{AP}$ is a labelling function. In this game, player 1 as the *defender* chooses an action and then player 2 as the *attacker* chooses an action after observing the choice of the defender. In SSGs a *strategy* is a set of actions that a player follows to make decisions in a game, based on available information and their objectives.

The logic rPATL [3], which is used within RV, is a temporal logic for expressing properties of Stochastic Multiplayer Games (SMGs), with SSGs a subclass of SMGs. This logic incorporates operators, including probabilistic (P), reward (R), next (X), bounded until ($\mathsf{U}^{\leq k}$), until (U), instantaneous reward after k steps ($\mathsf{I}^{=k}$), accumulated reward over k steps ($\mathsf{C}^{\leq k}$), and reachability reward (F). The syntax of rPATL is given by the following grammar:

$$\phi := \texttt{true} \mid \texttt{a} \mid \neg\phi \mid \phi \wedge \phi \mid \langle\langle C \rangle\rangle \mathsf{P}_{\sim q}[\psi] \mid \langle\langle C \rangle\rangle \mathsf{R}^{r}_{\sim x}[\rho] \mid \langle\langle C : C' \rangle\rangle_{max \sim x}(\theta)$$

$$\theta := \mathsf{P}[\psi] + \mathsf{P}[\psi] \mid \mathsf{R}^{r}[\rho] + \mathsf{R}^{r}[\rho] \qquad \psi := \mathsf{X}\ \phi \mid \phi\ \mathsf{U}^{\leq k}\ \phi \mid \phi\ \mathsf{U}\ \phi$$

$$\rho := \mathsf{I}^{=k} \mid \mathsf{C}^{\leq k} \mid \mathsf{F}\ \phi$$

with $\texttt{a}$ an atomic proposition, $C, C' \subseteq N$ sets (or *coalitions*) of players such that $C' = N \setminus C, \sim \in \{<, \leq, \geq, >\}, q \in [0, 1], x \in \mathbb{R}, r$ a state reward structure mapping a state to a non-negative rational reward, and $k \in \mathbb{N}$. The full semantics is defined elsewhere [7].

The informal semantics for rPATL is as follows: $\langle\langle C \rangle\rangle \mathsf{P}_{\sim q}[\psi]$ indicates that coalition C has a strategy to ensure that the probability of ψ to hold meets the bound $\sim q$. $\langle\langle C \rangle\rangle \mathsf{R}^{r}_{\sim x}[\rho]$ means that coalition C can guarantee the expected reward r accumulated over paths satisfying ρ meets the bound $\sim x$. $\langle\langle C' \rangle\rangle_{max \sim x}(\theta)$ states that coalition C can ensure the maximum value of θ with respect to C' is within the bound $\sim x$. For the temporal operators, $\mathsf{X}\ \phi$ expresses that ϕ holds in the next state, $\phi_1\ \mathsf{U}^{\leq k}\ \phi_2$ means ϕ_1 holds until ϕ_2 holds within k steps, and $\phi_1\ \mathsf{U}\ \phi_2$ indicates ϕ_1 holds until ϕ_2 holds unbounded. $\mathsf{I}^{=k}$ represents the instantaneous reward after k steps, $\mathsf{C}^{\leq k}$ denotes the accumulated reward within k steps, and $\mathsf{F}\ \phi$ denotes the reward cumulated until eventually reaching a state where ϕ holds.

3 Rational Verification under Stackelberg Equilibrium for Reachability Reward Properties

For an SSG $\mathcal{G}$, we extend rPATL to support rational verification under Stackelberg equilibrium for reachability reward properties in the coalition game of $\mathcal{G}$ induced by coalition $C = \{\text{defender, attacker}\}$ (denoted by $\mathcal{G}_C$), where coalition C can consist of player 1 (the defender), player 2 (the attacker), or both players 1 and 2 together. The defender selects a strategy (denoted as σ_1) that maximises their payoff, assuming that the attacker will best respond to this strategy (denoted as σ_2). On the other hand, given the defender's strategy, the attacker chooses a strategy that maximises their own payoff. In a typical SSG, the defender first selects a strategy to maximise their payoff, assuming that the attacker will respond optimally. The attacker, in response, attempts to exploit observed weaknesses. Using rPATL, we verify whether a given strategy is optimal under these conditions. We expand state formulas ϕ in the grammar by adding the syntax $\langle\langle C\rangle\rangle \mathsf{R}^{r}_{\mathsf{SE}=?}$ to denote the Stackelberg equilibrium (SE) and define its semantics as follows:

$$\begin{aligned}\langle\langle C\rangle\rangle \mathsf{R}^{r}_{\mathsf{SE}=?}[\,\mathsf{F}\phi\,] &\overset{\text{def}}{=} \mathbb{E}^{\max,\max}_{\mathcal{G}_C,s}[\text{rew}(r,\mathsf{c},\text{Sat}(\phi))] \\ &= \sup_{\sigma_1\in\Sigma_1}\ \sup_{\sigma_2\in BR(\sigma_1)}\ \mathbb{E}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}[\text{rew}(r,\mathsf{c},\text{Sat}(\phi))].\end{aligned} \tag{1}$$

$\mathbb{E}^{\max,\max}_{\mathcal{G}_C,s}$ represents the expected outcome for the coalition C starting from state s, assuming that both players are acting optimally in their own interests in the game $\mathcal{G}_C$. Reward calculation is done by $\text{rew}(r,\mathsf{c},\text{Sat}(\phi))$, where r is a defined reward structure, c denotes that the reward is accumulated along the paths to the target set of states, and computes the set of states in which ϕ holds.

Σ_1 is the set of strategies for the defender, and $BR(\sigma_1)$ is the attacker's best response strategy to the defender's strategy σ_1. Given the defender's strategy σ_1, $\sup_{\sigma_1\in\Sigma_1}\sup_{\sigma_2\in BR(\sigma_1)}$ maximises the attacker's payoff based on this strategy, and given the attacker's strategy σ_2, it maximises the defender's payoff accordingly. For now, we only support rPATL formulas in the form $\langle\langle C\rangle\rangle \mathsf{R}^{r}_{\mathsf{SE}=?}[\,\mathsf{F}\,\mathsf{a}\,]$. This restriction to only reachability properties is sufficient for evaluating specific system states that are necessary for security analysis.

4 Tool Overview

SᴛEVᴇ is the first rational verification tool for SSGs. It can automatically extract technical exploit descriptions into ADTs, which are then transformed into SSGs for further analysis using rPATL. The overall architecture is in Fig. 1.

The tool starts analysing vulnerabilities by gathering specific CVE details and their CVSS data from public vulnerability databases in JSON format. This includes descriptions of the vulnerability, attack sequences, mitigation steps, and impact scores. Security experts can manually refine this data and consider realistic cost values before proceeding.

In creating the ADT(1), the tool extracts the mitigation steps as the defender's actions, maps attack sequences to the attacker's actions, and adds dependencies between actions using conjunctive or disjunctive operators. The obtained ADT(1) is then transformed into an SSG. Preferences are calculated using a risk estimation based on impact scores and cost, then defined at the endpoints of the game.

The tool models defensive and offensive behaviours as a PRISM-games model. A set of rPATL properties is generated to focus on security aspects such as confidentiality, integrity, or availability, based on the mitigation technique in the CVE. Security experts can specify additional security requirements as temporal formulae. The outcomes of this process include a model of the game and expressions in temporal logic.

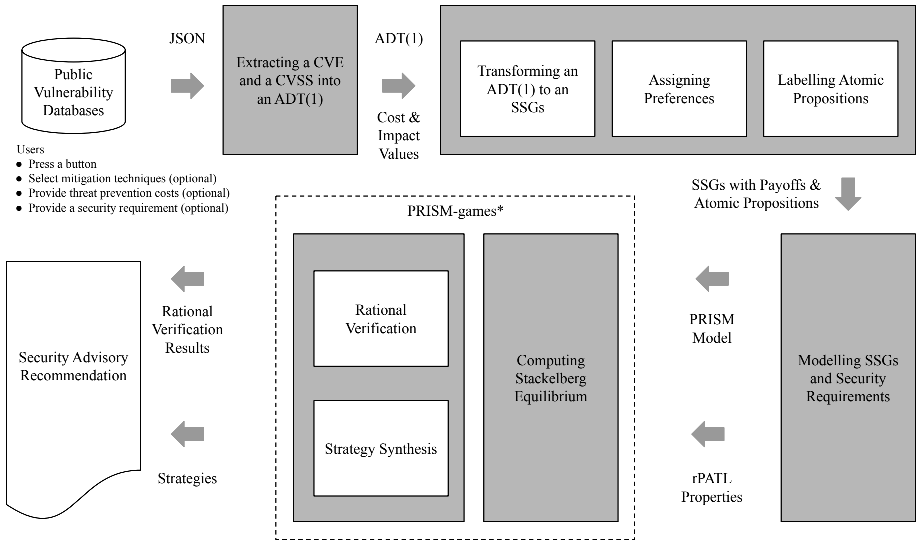
Public Vulnerability Databases
JSON
Extracting a CVE and a CVSS into an ADT(1)
ADT(1)
Cost & Impact Values
Transforming an ADT(1) to an SSGs
Assigning Preferences
Labelling Atomic Propositions
Users
Press a button
Select mitigation techniques (optional)
Provide threat prevention costs (optional)
Provide a security requirement (optional)
SSGs with Payoffs & Atomic Propositions
PRISM-games*
Rational Verification
Strategy Synthesis
Computing Stackelberg Equilibrium
Modelling SSGs and Security Requirements
PRISM Model
rPATL Properties
Rational Verification Results
Strategies
Security Advisory Recommendation

Fig. 1. Overview of StEVe's architecture and workflow. PRISM-games* is the version we patched to support Stackelberg equilibrium.

The process for computing the Stackelberg equilibrium determines the strategies for a defender and the corresponding responses from an attacker. StEVe relies on our extension of PRISM-games to support Eq.(1) for the analysis of the payoffs for both the defender and the attacker and the computation of the set of states in the equilibrium. Rational verification allows us to answer whether there is a defender strategy that comprises the equilibrium and satisfies their goals. Furthermore, we can synthesise strategies that are rational for the defender. A defender strategy that comprises the equilibrium refers to a strategy that satisfies the conditions for Stackelberg equilibrium, meaning it optimally coordinates the defender's and attacker's objectives. A rational strategy for the defender is one that minimises potential loss while considering the attacker's best response. As a final output we produce security advisory recommendations that include rational verification results and strategies.

5 Security Examples and Experiments

In this section, we present some experimental results. We select the two common vulnerabilities CVE-2017-8759 and CVE-2024-3400 due to their high severity and impact [12,13]. These have been exploited in targeted attacks, received extensive media attention, and highlight the need for robust defensive measures.

CVE-2017-8759, known as the .NET Framework remote code execution, serves as our primary example. In this CVE, the defensive solutions include the

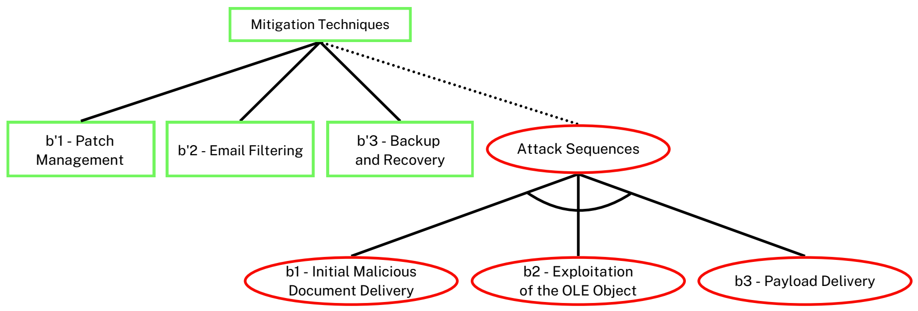

Fig. 2. Graphical representation of the ADT(1) term $c^d(\vee^d(b'_1, b'_2, b'_3), \wedge^a(b_1, b_2, b_3))$ generated from the CVE-2017-8759 example, which is then automatically transformed into an SSG. The defender (green nodes) can choose mitigation techniques for each of the following separately: b'_1 - patch management, b'_2 - email filtering, or b'_3 - backup and recovery. However, the attacker (red nodes) can initiate an attack by combining actions involving b_1 - initial malicious document delivery, b_2 - exploitation of the OLE object, and b_3 - payload delivery.

following actions: b'_1 – patch management, b'_2 – email filtering, b'_3 – backup and disaster recovery, b'_4 – application whitelisting, b'_5 – network segmentation and monitoring, b'_6 – user training and awareness, b'_7 – multi factor authentication, b'_8 – endpoint protection, and b'_9 – regular security audits. The exploit activities involve: b_1 – malicious document delivery, b_2 – exploitation of the OLE object, b_3 – payload delivery, b_4 – execution of the HTA file, b_5 – establishment of persistence, b_6 – command and control communication, and b_7 – lateral movement and further exploitation. In this experiment, we assume that the defender takes the following disjunctive actions: b'_1, b'_2, and b'_3. The attacker uses the following conjunctive actions: b_1, b_2, and b_3. The ADT(1) term encoding this scenario is $c^d(\vee^d(b'_1, b'_2, b'_3),\ \wedge^a(b_1, b_2, b_3))$ and graphically represented in Fig. 2, which is then transformed into an SSG with payoffs derived from adjustment costs. Each of the defender's strategies $\sigma_1, \ldots, \sigma_8 \in 2^{\{b'_1, b'_2, b'_3\}}$ is verified against the security requirements $\rho_1 = \mathsf{F}\ \texttt{applyPatch}$, $\rho_2 = \mathsf{F}\ \texttt{filterEmail}$, and $\rho_3 = \mathsf{F}\ \texttt{backup}$, where `applyPatch`, `filterEmail`, and `backup` are user-defined atomic propositions corresponding to the actions b'_1, b'_2, b'_3 respectively. Then the tool generates the three corresponding rPATL formulae $\langle\langle C \rangle\rangle \mathsf{R}^r_{\mathsf{SE}=?}[\rho_i]$, for $i \in \{1,\ 2,\ 3\}$.

We analyse multiple defensive strategies, as shown in Table 1. The cost of each strategy is calculated from deployment and maintenance expenses, while the attacker's gain is estimated based on potential financial damages from successful exploits, including data loss and business disruption. These estimates are informed by historical data collected by the company. The strategies are evaluated based on their effectiveness in minimising the defender's loss and the attacker's gain. The Stackelberg equilibrium is highlighted in bold. Among the evaluated strategies, σ_5 (patch management) is the optimal choice as it satisfies the Stackelberg equilibrium, resulting in the least loss for the defender and a moderate gain for the attacker. Rational verification with STEVE returns "true" when a security requirement satisfies some of the rPATL formulae and the Stackelberg equilibrium, and "false" when it fails to meet these criteria or does not achieve the Stackelberg equilibrium. If the user does not provide a security requirement, STEVE will synthesise strategies that satisfy both the Stackelberg equilibrium and the applicable rPATL formulae.

Table 1. Summary of defensive & attacking strategies. Stackelberg equilibrium in bold.

Strategy	Def. Actions	Att. Actions	Def. Loss (£)	Att. Gain (£)
σ_1	$\{b'_1, b'_2, b'_3\}$	$\{b_1, b_2, b_3\}$	-85,015	1,504.5
σ_2	$\{b'_1, b'_2\}$	$\{b_1, b_2, b_3\}$	-35,900	1,770
σ_3	$\{b'_2, b'_3\}$	$\{b_1, b_2, b_3\}$	-76,018	6,018
σ_4	$\{b'_1, b'_3\}$	$\{b_1, b_2, b_3\}$	-72,537.5	3,761.25
$\boldsymbol{\sigma_5}$	$\{\mathbf{b'_1}\}$	$\{\mathbf{b_1, b_2, b_3}\}$	**-24,750**	**4,425**
σ_6	$\{b'_2\}$	$\{b_1, b_2, b_3\}$	-43,600	7,080
σ_7	$\{b'_3\}$	$\{b_1, b_2, b_3\}$	-100,150	15,045
σ_8	$\varnothing$	$\{b_1, b_2, b_3\}$	-59,000	17,700

To further validate STEVE, we conduct additional experiments on CVE-2024-3400, which affects Palo Alto Networks, potentially allowing arbitrary code execution on the firewall. Table 2 summarises the defensive strategies and their outcomes for both CVE examples. The "Securing with Stackelberg" and "Best Case" scenarios both demonstrate significantly lower defender losses and moderate attacker gains, indicating that these strategies are highly effective in mitigating damage and deterring attackers. In contrast, the "Worst Case" scenario results in the highest financial losses for defenders and the greatest gains for attackers, highlighting a critical failure in defensive measures. Additionally, strategies such as "Minimise Rewards for the Attacker" and "Exhaustive Approach" result in identical defender losses and notably low attacker gains, underscoring their effectiveness in discouraging attackers by minimising their potential rewards.

Table 2. Experimental results in various scenarios.

Scenario	CVE-2017-8759	CVE-2024-3400
Securing with Stackelberg	**-£24,750, £4,425**	**-£13,000, £3,150**
Worst Case	-£100,150, £15,045	-£60,000, £25,200
Best Case	-£24,750, £4,425	-£13,000, £3,150
Minimise for the Attacker	-£85,015, £1,504.5	-£29,000, £450
Exhaustive Approach	-£85,015, £1,504.5	-£29,000, £630
Do Nothing	-£59,000, £17,700	-£60,000, £25,200

6 Discussion and Conclusion

SteVe's capability to automatically derive ADTs from CVEs and transform them into SSG models represents a new contribution. By using the temporal logic rPATL and the extended PRISM-games framework, the tool supports the computation of Stackelberg equilibria, ensuring that the defence strategies synthesised are not only optimal but also rational given the attacker's best responses. This shows how rational verification can guide security experts in making informed decisions about the allocation of limited resources. Our experiments with two CVEs demonstrate SteVe's potential to model security scenarios and suggest optimal defensive strategies. However, further validation in practical, real world scenarios is necessary to fully assess its impact on reducing damages and financial losses. We chose PRISM-games as the backend for SteVe due to its robustness in handling game-theoretic analysis, even in non-probabilistic settings. The ability to compute Stackelberg equilibria, along with its support for rPATL, makes it the most suitable option.

There are currently some limitations to our approach. SteVe does not support social engineering attacks. Since these exploit human psychology rather than technical vulnerabilities, it is difficult to model them, within the framework of SSGs. Another limitation is that the tool cannot analyse scenarios where multiple vulnerabilities are combined by an attacker to achieve their objectives.

We plan to address these limitations in future work by extending SteVe to support a diverse range of vulnerabilities. We consider using requirements elicitation tools like FRET [4] to help users specify their logical specifications. Additionally, moving beyond single shot games to represent repeated SSGs [17] by integrating probabilistic and Bayesian games [14] will allow for better modelling of uncertainties and incomplete information prevalent in cyber-defence scenarios. We also foresee the possibility of integrating SteVe's approach into the EVE [5] framework, which would further enhance its ability to handle rational verification in cybersecurity scenarios by enabling the computation of Stackelberg equilibria. This would allow EVE to support the analysis of leader-follower interactions in security games, improving its capability to model interactions between defenders and attackers. Finally, our code and the examples presented in this paper are publicly available [15].

Acknowledgments. S.P. is supported by a Royal Thai Government Scholarship. M.S. is supported by an Amazon Research Award on Automated Reasoning.

References

1. Abate, A., Gutierrez, J., Hammond, L., Harrenstein, P., Kwiatkowska, M., Najib, M., Perelli, G., Steeples, T., Wooldridge, M.J.: Rational verification: Game-theoretic verification of multi-agent systems. Appl. Intell. **51**(9), 6569–6584 (2021)
2. Aslanyan, Z., Nielson, F., Parker, D.: Quantitative verification and synthesis of attack defence scenarios. In: Proc. of CSF 2016. pp. 105–119. IEEE Computer Society (2016)

3. Chen, T., Forejt, V., Kwiatkowska, M., Parker, D., Simaitis, A.: Automatic verification of competitive stochastic systems. Formal Methods in System Design **43**(1), 61–92 (2013)
4. Giannakopoulou, D., Mavridou, A., Rhein, J., Pressburger, T., Schumann, J., Shi, N.: Formal requirements elicitation with FRET. In: International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ-2020). No. ARC-E-DAA-TN77785 (2020)
5. Gutierrez, J., Najib, M., Perelli, G., Wooldridge, M.J.: EVE: A tool for temporal equilibrium analysis. In: Proc. of ATVA'18. LNCS, vol. 11138, pp. 551–557. Springer (2018)
6. Kordy, B., Mauw, S., Radomirovic, S., Schweitzer, P.: Attack defence trees. J. Log. Comput. **24**(1), 55–87 (2014)
7. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: Equilibria-based probabilistic model checking for concurrent stochastic games. In: Proc. 23rd International Symposium on Formal Methods. LNCS, vol. 11800, pp. 298–315. Springer (2019)
8. Kwiatkowska, M., Norman, G., Parker, D.: Probabilistic model checking and autonomy. Annu. Rev. Control. Robotics Auton. Syst. **5**, 385–410 (2022)
9. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: PRISM-games 3.0: Stochastic game verification with concurrency, equilibria and time. In: Proc. of CAV'2020. LNCS, vol. 12225, pp. 475–487. Springer (2020)
10. Leyton-Brown, K., Shoham, Y.: Essentials of game theory: A concise multidisciplinary introduction. Synthesis Lectures on AI and ML, Morgan & Claypool Publishers (2008)
11. Najib, M.: Rational verification in multi-agent systems. Ph.D. thesis, UK (2020)
12. National Vulnerability Database (NVD): CVE-2017-8759. https://nvd.nist.gov/vuln/detail/CVE-2017-8759
13. National Vulnerability Database (NVD): CVE-2024-3400. https://nvd.nist.gov/vuln/detail/CVE-2024-3400
14. Osborne, M.J., Rubinstein, A.: A course in game theory. MIT Press (1994)
15. Phetmanee, S., Sevegnani, M., Andrei, O.: StEVe: A rational verification tool for Stackelberg security games. https://doi.org/10.5281/zenodo.11004420
16. Sinha, A., Fang, F., An, B., Kiekintveld, C., Tambe, M.: Stackelberg security games: Looking beyond a decade of success. In: Proc. IJCAI18 (2018)
17. Tambe, M.: Security and game theory. Cambridge University Press (2012)
18. Wooldridge, M., Gutierrez, J., Harrenstein, P., Marchioni, E., Perelli, G., Toumi, A.: Rational verification: From model checking to equilibrium checking. Proc. of the AAAI Conference on Artificial Intelligence **30**(1) (2016)