Selecting Security Mechanisms in Secure Tropos

Michalis Pavlidis, Haralambos Mouratidis
Emmanouil Panaousis and Nikolaos Argyropoulos

University of Brighton
{m.pavlidis,h.mouratidis,e.panaousis,n.argyropoulos}@brighton.ac.uk

Abstract. As security is a growing concern for modern information systems, Security Requirements Engineering has been developed as a very active area of research. A large body of work deals with elicitation, modelling, analysis, and reasoning about security requirements. However, there is little evidence of efforts to align security requirements with security mechanisms. This paper extends the Secure Tropos methodology to enable a clear alignment, between security requirements and security mechanisms, and a reasoning technique to optimise the selection of security mechanisms based on these security requirements and a set of other factors. The extending Secure Tropos supports modelling and analysis of security mechanisms; defines mathematically relevant modelling concepts to support a formal analysis; and defines and solves an optimisation problem to derive optimal sets of security mechanisms. We demonstrate the applicability of our work with the aid of a case study from the health care domain.

Keywords: Security modelling, Secure Tropos.

1 Introduction

Security is an important aspect of modern information systems and it is widely accepted that it should be treated from the early stages of the information system development process, and not as an afterthought [1–4]. As a result, during the last fifteen years the research community has witnessed a significant amount of works [5, 6], which deal with the definition, elicitation, analysis, and reasoning of security requirements. We have contributed to this body of literature, with our work on *Secure Tropos* [7], which is a security requirements engineering methodology that supports elicitation and analysis of security requirements.

Ideally, enough security countermeasures (also known as security mechanisms or security safeguards) should be applied to a system to satisfy those security requirements. However, in practice, there is usually a trade-off between security measures and other factors such as cost and time. The literature from the security engineering community has proposed a number of works that focus on security countermeasures selection usually in relation to vulnerabilities [8–10], investment costs [11, 12] or risks [13, 14]. However, such approaches have ignored the relationship between security mechanisms and security requirements.

We believe, this is an important parameter, especially in the current era of information systems, where security requirements can frequently evolve, and

there is a plethora of available security mechanisms. It is only when there is a clear relationship between security requirements, security mechanisms, and potential trade-off factors (such as cost and benefit of such mechanisms) that the decision is fully supported with the right evidence.

On the other hand, despite the impressive amount of work in the (Security) Requirements Engineering area, the literature provides little evidence of works that provide a clear relationship between security requirements and potential security mechanisms, and it lacks automated approaches to optimise the selection of security mechanisms based on security requirements and a set of other factors, such as cost.

In this paper, we address this gap by extending our previous work on Secure Tropos.

The specific contributions of our work can be summarised as follows:

- extension of a security requirements engineering methodology to support a clear alignment of security requirements and security mechanisms through a well defined process;
- definition of benefit, cost, satisfiability, satisfiability weight, non-functional requirements (NFR) cost, financial cost in the context of security requirements analysis, and a clear relationship with related concepts of the Secure Tropos methodology;
- mathematical representation/formulation of the relevant modelling concepts and relations, along with a definition of a graph data structure to support their modelling;
- computation of an optimal set of security mechanisms in relation to security requirements, given certain criteria about maximum NFR costs, minimum satisfiability levels for each security requirement, and an available monetary budget, by solving a multi-objective optimisation problem;
- application and evaluation of the work to a real-case study.

The next section provides an overview and comparison with related work, while Section III provides a summary of Secure Tropos. Section IV introduces our extensions and Section V introduces a case study from the health care domain, and it discusses the application of our approach to the case study. Section VI concludes the paper.

2 Related Work

The domain of requirements engineering is already rich in terms of decision making techniques, which reason about the selection of alternative design options in order to satisfy requirements that are represented as goals in Goal Oriented Requirements Engineering GORE approaches [15] [16] [17] [18] [19] [20] [21] [22].

However, all these works have not been developed with security in mind. As such, on a conceptual level they lack a clear definition of security requirements and security related concepts such as security mechanisms, while on a process level they lack structured methods and processes to support the identification of security mechanisms and the selection of security packages (set of security mechanisms) to satisfy the elicited security requirements.

The literature also provides a large body of work from the security requirements engineering area [3,6]. Mellado et al. [23] introduced the Security Requirements Engineering Process (SREP), which is based on several Common Criteria constructs to elicit and analyse security requirements. The Security Quality Requirement Engineering Methodology (SQUARE) [24] is another security requirements engineering approach similar to SREP. Both SREP and SQUARE are asset-based and risk-driven methods that follow a number of steps, for eliciting, categorising, and prioritising security requirements. Sindre and Opdahl [25] have developed a misuse case driven approach to establish visual link between use cases and misuse cases for eliciting security requirements at an early stage of the development. McDermott and Fox [1] adapt use cases to capture and analyse security requirements, and they call the adaption an abuse case model. Liu et al. [26] analyse security requirements as relationships amongst strategic actors by proposing different kinds of analysis techniques to model potential threats and security measures. Paja et al. [27] provide reasoning techniques for detecting inconsistencies among security requirements. [9,10], base their work on security problem frames, which are patterns that classify security software development problems related to security and support developers in analyzing them. Lamsweerde [28] provides an extension of the KAOS approach, where security goals are refined until they become precise and represent security requirements. Then, once alternative countermeasures have been identified the NFR qualitative framework is employed to support the selection process according to how critical the security goal been threatened is and how well the countermeasure meets the other non-functional requirements. However, these approaches focus on the elicitation and analysis of security requirements and they do not explicitly consider the concept of security mechanism. As such, they lack a clear definition between the security requirements/security mechanisms relationship and lack support in attributing financial cost. In contrast, our selection algorithm has been integrated into the security requirements engineering process, and as such, derives a set of security mechanisms that satisfy identified security requirements based on preferable criteria, which are set by the requirements engineer. As discussed in the introduction, we believe that such alignment is very important where security requirements can frequently evolve, and there is a plethora of available security mechanisms. It is only when there is a clear relationship between security requirements, security mechanisms, and potential trade-off factors (such as cost and benefit of such mechanisms) that the decision is fully supported with the right evidence. Our approach not only provides an alignment between the what and why (security requirements) with the how (security mechanisms), but also a quantitative method that enables requirements engineers to cope with the complexity of selecting the optimal combination of security countermeasures in a systematic way.

In addition, there is a line of research in the area of security risk assessment [29–31] where there is identification, assessment, and mitigation of risks to security mechanisms that will endanger the satisfaction of security requirements. However, the risk are investigated in isolation with limited support for cost/ben-

efit trade-off analysis. In [32] although the security solution design trade-off is addressed, there is no clear alignment of the security solutions with the security requirements. The literature also provides research related to the selection of security mechanisms during the run-time of a system, such as [e.g. [20], [33]] as well as from the Decision support area, such as [11–14]. However, these works are heavily based on risk management and in most cases, the selection criteria are the financial cost of a mechanism and its effectiveness in blocking an attack, the potential impact (i.e. risk) and the attack success likelihood. However, they offer limited support to identify security requirements, and most importantly the security mechanisms are not linked to the security requirements. Therefore, the selection algorithms do not consider which security requirement a security mechanism satisfies and also what the importance of that requirement is. Moreover, although useful during run time, such approaches entail that a wide range of security mechanisms are acquired and implemented without considering their financial cost.

3 Secure Tropos

The Secure Tropos methodology [7] is based on the principle that security should be analysed and considered from the early stages of the software system development process, and not added as an afterthought. To support that approach, the methodology provides a modelling language, a security-aware process, and a set of automated processes to support the analysis and consideration of security from the early stages of the development process. The Secure Tropos language consists of a set of concepts from the requirements engineering domain, and in particular Goal-Oriented Requirements Engineering [34, 35], such as actor, goal, plan, and dependency, which are enriched with concepts from security engineering, such as security constraint, secure plan, and attacks. An actor [35], represents an entity that has intentionality and strategic goals within the software system or within its organisational setting. Within a network of actors, which is usually the case in large software systems with multiple stakeholders, one actor might depend on another actor for a goal, a plan or a resource. A goal [35] represents a condition in the world that an actor would like to achieve. In other words, goals represent actor's strategic interests. A plan represents, at an abstract level, a way of doing something [35]. The fulfilment of a plan can be a means for satisfying a goal. As such, different alternative plans, that actors might employ to achieve their goals, are modelled to enable software engineers to reason about the different ways that actors can achieve their goals, and decide upon the optimal way. A resource [35] presents a physical or informational entity that one of the actors requires. The main concern when dealing with resources is whether the resource is available, and who is responsible for its delivery.

In line with existing literature [6, 36, 37], we define security requirements as constraints on specific functions of a system. Towards this end, security requirements are represented, in Secure Tropos, as *Security Constraints*. A Security Constraint is defined as a security condition imposed to an actor that restricts the achievement of an actor's goals, the execution of plans or the availability

of resources. To support the analysis and evaluation of the developed security solution, the modelling language supports the modelling of security attacks. An attack is an action that might cause a potential violation of security in the system (this definition has been adopted by Matt Bishop's definition of a computer attack). Within the context of an attack, an attacker represents a malicious actor that is interested in attacking the system. As described above, an actor has intentionality and strategic goals within the system. In the case of an attacker, these are related to breaking the security of a system, and identifying and executing malicious goals. To support the modelling of an actor by depending on another actor for a security constraint, Secure Tropos introduces the idea of Secure Dependency. A Secure Dependency introduces one or more Security Constraints that must be fulfilled for the dependency to be valid. *Vulnerabilities* are defined as weaknesses or flaws, in terms of security, that exist from a resource, an actor and/or a goal. Vulnerabilities are exploited by threats, as an attack or incident within a specific context. It is worth stating that legitimate actors might unintentionally introduce vulnerabilities to a system due to failure or mistakes. Threats pose potential loss or indicate problems that can put the system at risk. On the other hand, actors within the system environment have single or multiple goals. The *process* in Secure Tropos is one of analysing the security needs of the stakeholders and the system in terms of security constraints, imposed on the stakeholders and the system, identifying relevant security threats and attacks and analyse and identify potential countermeasures against those attacks.

4 Proposed Extensions

In this section we discuss how we have extended the Secure Tropos methodology. We first discuss the extensions to the modelling language, we then mathematically formulate some of the Secure Tropos components, and finally, we present a new Secure Tropos process.

The modelling language of Secure Tropos, as briefly described in the previous section, is extended with concepts and links required to model and analyse security countermeasures, but also to create models that have a clear explicit alignment between security constraints and security countermeasures. The updated meta-model of the Secure Tropos language is shown in Fig. 1.

When a Security Constraint is introduced, further analysis is required to establish if and how this constraint can be satisfied. A Security Objective represents an objective that is assigned to an actor, and it indicates a course of action that the actor needs to follow to satisfy one or more security constraints, whose satisfaction by a security objective is defined through a `Satisfies` relationship. Countermeasures are defined in our approach in terms of security mechanisms. These represent standard security methods, which contribute to the satisfaction of the security objectives. Some of these methods are able to prevent security attacks, whereas others are able only to detect security breaches.

In the following, we introduce some mathematical notation that helps to formalise the above Secure Tropos concepts. Assume $\mathcal{C}$ a set of security constraints, and $\mathcal{O}$ a set of security objectives. Each security constraint can be satisfied by

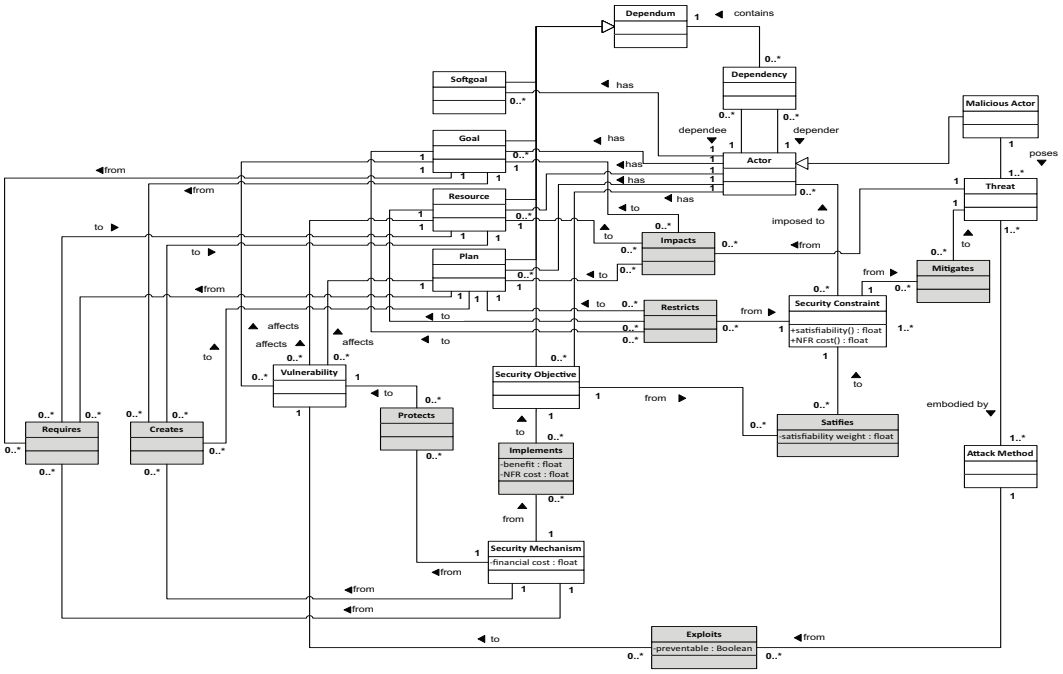

Fig. 1. A part of the Secure Tropos metamodel.

one or more security objectives. A security objective j contributes to the satisfaction of a constraint i at some degree $w_{ji} \in [0,1]$, which is called *satisfiability weight*, with the property that $\sum_j w_{ji} = 1$. For example, when objective j does not satisfy constraint i (i.e. no `Satisfies` link exist between j, i), then $w_{ji} = 0$.

Each objective can be implemented by different mechanisms or *combination of security mechanisms*. In order to represent a combination of security mechanisms that are required for the implementation of a security objective we introduce the notion of a *security package*. This consists of either one security mechanism or a set of mechanisms connected with an AND decomposition. For a set $\mathcal{M}$ of mechanisms, any package z is modelled by the vector $\mathbf{p}_z := [p_{qz}] \in \{0,1\}^{|\mathcal{M}|}$, where p_{qz} equals 1 when mechanism q is included in package z, and 0 otherwise. We define the set $\mathcal{P} := \{\mathbf{p}_z\}$ of π security packages that exist in the model. A security package z contributes to the implementation of a security objective j, and we define its degree of contribution as follows:

Definition 1 (Benefit). *Benefit is defined as the degree of implementation of a security objective by a security package.*

We model the benefit of package z to objective j, by $b_{zj} \in [0,1]$. For example, if package z does not implement objective j then $b_{zj} = 0$. In this case, there is no `Implements` link between z and j.

Definition 2 (Satisfiability). *We define the satisfiability of a security constraint i as $s_i := \sum_j \sum_z w_{ji}\, b_{zj}$.*

As a security constraint mitigates one or more security threats, the satisfiability of a security constraint determines the *mitigation degree* of these *threats*. It is intuitive that the higher the satisfiability the higher the system protection against these threats. Furthermore, each security package comes also with a cost value, defined as follows:

Definition 3 (Financial Cost). *Each security package z has a financial cost value f_z, which refers to potential monetary expenses for acquiring, developing, maintaining, and operating its security mechanisms.*

Definition 4 (Non-Functional Requirements Cost). *Each security package z has a non-functional requirements cost value η_z, which is the cost incurred to non-functional system requirements by the implementation of security mechanisms that comprise this package. This cost can be seen as the negative impact of the security package to, for example, usability, performance, and scalability [38].*

In the rest of this section, we discuss the newly defined security-aware process, which extends the Secure Tropos process. The extended process supports security requirements engineers with the selection of a set of security packages that fulfil identified security constraints. Given the fact that most of the security packages have different benefits and costs, there are multiple combinations of them that might satisfy the identified security requirements. As such, the aim of the engineer is to derive a set of security packages that: (i) meet a minimum satisfiability and a maximum NFR cost, which are set by the requirements engineer during the Secure Tropos modelling process; (ii) its financial cost can be covered by an available budget; (iii) is "optimal", based on a cost-benefit analysis, given the different satisfiability weights.

As part of the new process, after eliciting the security constraints following the traditional Secure Tropos process [3,7], the requirements engineer, with input from the system stakeholders, sets a *minimum satisfiability* α_i, and a *maximum acceptable non-functional requirements cost* β_i for each constraint i, and the different satisfiability weights. Furthermore, the requirements engineer, with the support of a security engineer, identifies the set $\mathcal{P}$ of available security packages that implement the system security objectives $\mathcal{O}$, along with the benefit values b_{zj} of each package z to each objective j. It is worth stressing here that, the process assumes the selection of only one security package per objective. This assumption does not restrict the number of security mechanisms that can be selected for the implementation of an objective, as a security package can be the composition of any security mechanisms. Finally, the requirements engineer, with input from the system stakeholders, provides a maximum available financial budget Φ to cover the financial cost of the selected security packages.

To correlate a security objective j with only one of the security packages that implement this objective, representing a possible *solution* for the implementation of this objective, we define the vector $\mathbf{x}_j := [x_{zj}] \in \{0,1\}^{\pi}$, s.t. $\sum_z x_{zj} = 1$. When package z is selected to implement objective j, then $x_{zj} = 1$, and hence $\mathbf{x}_j = [\underbrace{0,\ldots,0}_{j-1}, 1, \underbrace{0,\ldots,0}_{\pi-j}]$. A *complete solution*, denoted by $\mathbf{X}$, consists of

combinations of different $\mathbf{x}_j$ covering the entire range of objectives. We formally denote a solution by $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\rho]$. The goal of the requirements engineer is to select an $\mathbf{X}$ such that:

(**C-I**) for each constraint $i \in \mathcal{C}$, α_i and β_i are satisfied by packages in $\mathbf{X}$;
(**C-II**) the sum of costs of all selected packages, which comprise $\mathbf{X}$, are within the available budget Φ;
(**C-III**) $\mathbf{X}$ is optimal according to a cost-benefit analysis, which considers the non-functional requirements costs of packages that comprise $\mathbf{X}$, and their benefit values.

To support the aforementioned security-aware process of Secure Tropos, and utilise the newly defined modelling language, we must solve a multi-objective optimisation problem using the OptiMathSAT [39] tool. The solution satisfies criteria **C-I**, **C-II**, and **C-III**, mentioned in the previous section. To facilitate the cost-benefit analysis, which is part of **C-III**, we introduce the notion of the *utility of a security constraint*.

Definition 5 (Utility). *For a given* $\mathbf{X}$*, we define the utility* $u_i(\mathbf{X})$ *of a constraint* i *as* $u_i(\mathbf{X}) := \frac{s_i(\mathbf{X})}{\eta_i(\mathbf{X})}$.

In the following we discuss the computation of $s_i(\mathbf{X})$ and $\eta_i(\mathbf{X})$. From Definition 2 we can deduce that $s_i(\mathbf{X}) := \sum_j \sum_z w_{ji}\, x_{zj}\, b_{zj}$. In order to represent the objectives that satisfy each security constraint we define the latter as the vector $\mathbf{c_i} := [c_{ji}] \in \{0,1\}^\rho, \forall\ i \in \mathcal{C}$. By having this, we can now define the total non-functional requirements cost for a specific constraint as $\eta_i(\mathbf{X}) := \sum_j \sum_z c_{ji}\, x_{zj}\, \eta_z$. We can also derive the financial cost of a solution $\mathbf{X}$ as $f(\mathbf{X}) := \sum_j f(\mathbf{x}_j) = \sum_j \sum_z x_{zj}\, f_z$. We finally compute the optimal solution $\mathbf{X}^*$ by solving the following optimisation problem:

$$\begin{aligned}
&\max_{\mathbf{X}} \sum_i I_i\, u_i(\mathbf{X}) \\
&\text{s.t.} \sum_i I_i = 1 \\
&f(\mathbf{X}) \leq \Phi \\
&s_i(\mathbf{X}) \geq \alpha_i, \forall i. \\
&\eta_i(\mathbf{X}) \leq \beta_i, \forall i,
\end{aligned}$$

where I_i signifies the importance of a security constraint to the system, and it is defined by the requirements engineer, in each case.

5 Case Study

In order to exhibit the applicability of our approach, this section describes how our approach can be applied on the real life case study of the established Greek national e-prescription system [40]. This is a cloud-based system, which is currently used by Greek *healthcare professionals* to handle patients' electronic medication and clinical tests prescriptions. *Medical practitioners*, regardless of specialty, can create an electronic prescription document, which can then be fulfilled,

by using the same platform, by any pharmacist or clinic staff. The healthcare professionals access the e-prescription system via an online portal. The back-end of the system has been created, and it is also maintained, by a non-profit organisation, which is in charge of the e-governance infrastructure of the Greek Ministry of Health. The system was introduced in October 2010, as a pilot, for one healthcare provider. By May 2012, the system was fully released, and it supported all registered Greek healthcare providers. It is estimated that the system handles over 500 thousand prescriptions on a daily basis, accommodating over ten thousand pharmacists and thirty eight thousand medical practitioners [41].

It is worth noting that in this paper, we focus on the main functionalities of the e-prescription system and its security requirements, as described in the relevant act of the Greek parliament [40]. Following the extended Secure Tropos modelling language and security-aware process we have elicited the main security requirements, in terms of security constraints, of the system along with security mechanisms as depicted in Fig. 2. To help with the creation of the model, we have used the SecTro tool. SecTro is a tool for the Secure Tropos methodology based on the ADOxx meta-modelling platform [42], which supports the requirements engineer in creating Secure Tropos models, according to the language metamodel, and it provides a set of analysis functionalities such as identifying security constraints that are not satisfied, and threats that are not mitigated. The rest of this section, describes the application of our work to the case study.

As shown in Fig. 2, for a healthcare professional to get access to the system, a user registration process must be completed. To this end, the users must provide a number of personal information, before they are given their login credentials. This fulfils the system goal of "*Register system users*". The main functionality of the e-prescription system is the handling of the prescription documents from their creation to their fulfilment, as indicated by the goal "*Handle prescription documents*". This process is initiated when a patient visits a medical practitioner, and the latter decides to prescribe some medical treatment, in the form of medication or clinical tests. *Patient Information* is accessed from the system via a unique patient identification number and it is included in each *Prescription Document* along with the prescribed treatment. A prescription is fulfilled when a patient receives consultation or treatment by a pharmacy or a clinic. Each prescription document that a healthcare professional handles can later be accessed via a personal archive. This is maintained by the system, and it generates the system goal "*Archive prescription documents*".

The security requirements of the e-prescription system are described in the legal framework of the e-prescription system [40] and modelled in our approach as security constraints. These constraints are satisfied through the implementation of security objectives as shown in Fig. 2. In particular, the security constraint "*Authorised access only*" aims to ensure that only registered users access and operate the system. This constraint is satisfied by the security objectives "*Authentication*" and "*User Authorisation*". Similarly, the security constraint "*Correct data received and stored*" restricts the creation, fulfilment and access to prescription documents, aiming to ensure the integrity of the information stored in the

Table 1. Security packages.

Package	Mechanism	Package	Mechanism
$\mathbf{p}_1$	m_1 (DoS protection)	$\mathbf{p}_2$	m_2 (2-way mirroring)
$\mathbf{p}_3$	m_3 (3-way mirroring)	$\mathbf{p}_4$	m_4 (RAID5)
$\mathbf{p}_5$	$m_1 \wedge m_2$	$\mathbf{p}_6$	$m_1 \wedge m_3$
$\mathbf{p}_7$	$m_1 \wedge m_4$	$\mathbf{p}_8$	m_5 (Symmetric encryption)
$\mathbf{p}_9$	m_6 (Asymmetric encryption)	$\mathbf{p}_{10}$	m_7 (Network-based IDS)
$\mathbf{p}_{11}$	m_8 (Host-based IDS)	$\mathbf{p}_{12}$	$m_5 \wedge m_7$
$\mathbf{p}_{13}$	$m_5 \wedge m_8$	$\mathbf{p}_{14}$	$m_6 \wedge m_7$
$\mathbf{p}_{15}$	$m_6 \wedge m_8$	$\mathbf{p}_{16}$	m_9 (Role-based AC)
$\mathbf{p}_{17}$	m_{10} (Rule-based AC)	$\mathbf{p}_{18}$	m_{11} (Username/Password)
$\mathbf{p}_{19}$	m_{12} (Smart card)	$\mathbf{p}_{20}$	m_{13} (Biometrics)
$\mathbf{p}_{21}$	$m_{11} \wedge m_{14}$: (m_{14} Anti-logging)		

system. This is satisfied by the implementation of the "*Data Integrity*" security objective. "*Confidentiality of personal information*" requires that the sensitive and personal patient information contained in a prescription document must be accessed only by authorised system users, and it is satisfied by the implementation of the security objective "*Confidentiality*". Finally, the security constraint "*System always available*", satisfied by the "*Availability*" security objective, ensures the uninterrupted functionality of the e-prescription system, regardless of potential infrastructure technical issues or targeted attacks launched against it.

In our work, the aforementioned security objectives are implemented by an optimal set of security packages derived solving a multi-objective optimisation problem. As there is an one-to-one mapping between packages and mechanisms, the afore optimal set can be translated to a set of optimal security mechanisms. Starting from the "*Authentication*" objective, this can be implemented by the use of "*Username / Password*" mechanism. A security package can contain this mechanism along with an "*Anti-logging control*" mechanism. Other alternative security packages that provide user authentication could implement the use of "*Smart card*" or "*Biometrics*" mechanisms. Similarly, for the "*User Authorisation*" security objective, there is a choice between "*Role-based Access Control*" (RBAC) or "*Rule-based Access Control*" (RAC). According to this, each user has access only to specific system information and functionalities.

Different encryption mechanisms (i.e. "*Symmetric*" or "*Asymmetric Encryption*") can be implemented in security packages to contribute towards the achievement of both "*Data Integrity*" and "*Confidentiality*" security objectives. These security packages can contribute towards both objectives, as they can protect transmitted information from being accessed and modified by unauthorised users. In addition to encryption, "*Intrusion Detection Systems*" (IDS) can be deployed for the implementation of the "*Data Integrity*" objective. Thus, "*Host-based IDS*" or "*Network-based IDS*" can be standalone security packages or they can be combined with encryption mechanisms.

Finally, the security objective "*Availability*" can be achieved by creating redundancy at the system infrastructure. Therefore, different types of disk mirroring can be implemented (e.g. "*2-way mirroring*", "*3-way mirroring*" or "*RAID 5*") to ensure the uninterrupted functionality of the system and the availability of the stored data. Alternative to disk mirroring solutions, can be packages that offer protection against denial of service attacks ("*DoS Protection*"). This mechanism can be implemented along with redundancy mechanisms to form security packages able to enforce the "*Availability*" objective.

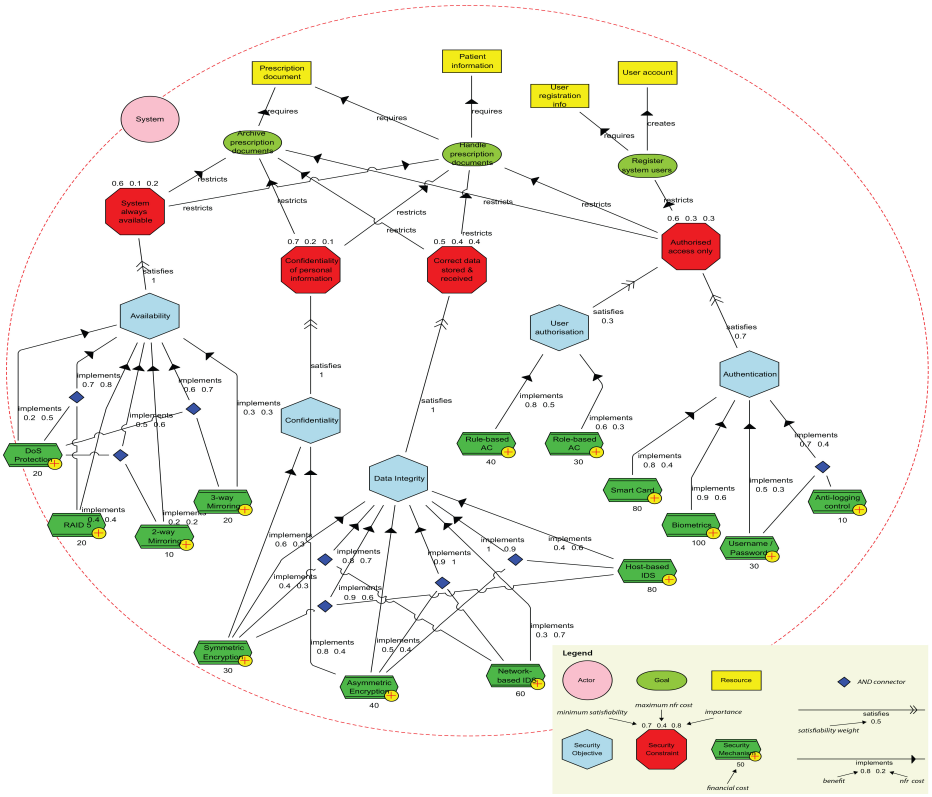

Fig. 2. Security analysis of part of the Greek national e-prescription system

To set the different costs and benefits of the security packages we have combined information found in scientific literature, technical reports, and pricing of commercial security solutions. As an example, literature suggests that while solutions using biometrics are the most secure as opposed to simple passwords or smart card authorisation, they lack in usability and deployability [43]. However, commercial applications used for managing biometric authorisation entail a high financial overhead, due to the infrastructure required at each end-user's terminal. Based on these insights, appropriate values are assigned to these packages resulting in the highest benefit values for "Biometrics" among the 3 available packages, followed by "Smart Cards" and finally "Username/Password". Similarly, due to the high financial and NFR costs associated with them, "Biometrics" also receive the highest cost value compared to the alternative mechanisms. Similar value assignment activities were followed for each of the security packages.

We have undertaken simulations to derive the optimal set of security packages for different financial budgets, as presented in Table 2. The format of the solution in this table is given by the tuple $[\mathbf{p}_x, \ldots, \mathbf{p}_y]$, where its i-th element represents the security package that implements the security objective i, where

Table 2. Simulation Results

Budget	Solution	Utility	Satisf. values	NFR costs	Utilities	Cost
190	$[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_9}, \mathbf{p_{16}}, \mathbf{p_{21}}]$	2.324	0.7, 0.8, 0.5, 0.67	0.4, 0.2, 0.2, 0.35	1.75, 4, 2.5, 1.914	190
	$[\mathbf{p_6}, \mathbf{p_9}, \mathbf{p_9}, \mathbf{p_{16}}, \mathbf{p_{21}}]$	2.317	0.6, 0.8, 0.5, 0.67	0.35, 0.2, 0.2, 0.35	1.714, 4, 2.5, 1.914	190
230	$[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_9}, \mathbf{p_{16}}, \mathbf{p_{19}}]$	2.38	0.7, 0.8, 0.5, 0.74	0.4, 0.2, 0.2, 0.35	1.75, 4, 2.5, 2.114	230
	$[\mathbf{p_6}, \mathbf{p_9}, \mathbf{p_9}, \mathbf{p_{16}}, \mathbf{p_{19}}]$	2.38	0.6, 0.8, 0.5, 0.74	0.35, 0.2, 0.2, 0.35	1.714, 4, 2.5, 2.114	230
	$[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_9}, \mathbf{p_{16}}, \mathbf{p_{21}}]$	2.324	0.7, 0.8, 0.5, 0.67	0.4, 0.2, 0.2, 0.35	1.75, 4, 2.5, 1.914	190
260	$[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_{13}}, \mathbf{p_{16}}, \mathbf{p_{21}}]$	2.524	0.7, 0.8, 0.9, 0.67	0.4, 0.2, 0.3, 0.35	1.75, 4, 3, 1.914	260
	$[\mathbf{p_6}, \mathbf{p_9}, \mathbf{p_{13}}, \mathbf{p_{16}}, \mathbf{p_{21}}]$	2.517	0.6, 0.8, 0.9, 0.67	0.35, 0.2, 0.3, 0.35	1.714, 4, 3, 1.914	260
	$[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_9}, \mathbf{p_{16}}, \mathbf{p_{19}}]$	2.38	0.7, 0.8, 0.5, 0.74	0.4, 0.2, 0.2, 0.35	1.75, 4, 2.5, 2.114	230
300	$[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_{13}}, \mathbf{p_{16}}, \mathbf{p_{19}}]$	2.584	0.7, 0.8, 0.9, 0.74	0.4, 0.2, 0.3, 0.35	1.75, 4, 3, 2.114	300
	$[\mathbf{p_6}, \mathbf{p_9}, \mathbf{p_{13}}, \mathbf{p_{16}}, \mathbf{p_{19}}]$	2.577	0.6, 0.8, 0.9, 0.74	0.35, 0.2, 0.3, 0.35	1.714, 4, 3, 2.114	300
	$[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_{13}}, \mathbf{p_{16}}, \mathbf{p_{21}}]$	2.524	0.7, 0.8, 0.9, 0.67	0.4, 0.2, 0.3, 0.35	1.75, 4, 3, 1.914	260

$i \in [1, 5]$. In the same table, we see that for each financial budget more than one set of packages are given; the optimal, the second optimal, and the third optimal, whenever available. This is because we believe that the requirements engineer, might want to have an idea of the alternative solutions if he wishes, for instance, to reduce the financial budget. We have restricted our results to the best 3 sets, because these might be indicative of the available alternatives. It is obvious that, if the requirements engineer desires to reduce the budget further, then another execution of the program is required. Table 1 facilitates the results discussion by summarising all security packages, of our case study, along with the mechanisms that constitute them. It is worth noting that as mechanisms represent the fundamental notion of security implementations, they are the building blocks of security packages. Hence, there is no sense in combining different alternative packages that implement a security objective, when instead we can combine different mechanisms, by using an AND relationship, to form a new security package.

In our simulations, we have varied the financial budget from very low values and we were increasing this, by using a step of 10 units, until we reach a point where an additional budget provides no improvement in the optimal set of the packages. By using Fig. 2 and Table 1, it is trivial to see the exact security mechanisms that constitute each solution. First, we observed that the minimum financial budget required to satisfy the system security requirements, was 190. This allows two solutions with the same financial cost but with the $[\mathbf{p}_7, \mathbf{p}_9, \mathbf{p}_9, \mathbf{p}_{16}, \mathbf{p}_{21}]$, to perform better. The next financial budget that allows a new solution, i.e. $[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_9}, \mathbf{p_{16}}, \mathbf{p_{19}}]$, is 230. In this case, we notice a 7% improvement of the satisfiability of the fourth security constraint, as opposed to the case where the budget was 190. It is worth noting here that this improvement comes with no increment to any of the non-functional requirements costs. The next budget level that introduces a different solution, and improves security, as determined by its 2.524 utility value, is the $[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_{13}}, \mathbf{p_{16}}, \mathbf{p_{21}}]$. This solution costs 30 extra financial units, but improves the satisfiability of the third security constraint by 40%. Finally, the highest performance is achieved when the budget equals 300, where the optimal solution is $[\mathbf{p_7}, \mathbf{p_9}, \mathbf{p_{13}}, \mathbf{p_{16}}, \mathbf{p_{19}}]$. In other words, the simulation results show that any budget value higher than 300 does not improve security.

6 Conclusion

The selection of appropriate security mechanisms that satisfy the security requirements under a limited budget is an important task, as it determines the final security level of a system. Furthermore, a major consideration when selecting security mechanisms is to maximise the satisfaction of security requirements while minimising their negative side effects to other non-functional requirements. Despite its importance, the computation of an optimal set of security mechanisms is not a straightforward task due to the large decision space. This paper presents a twofold extension of Secure Tropos methodology. First, the relationship between security requirements and security mechanisms is explicitly shown to enable a better understanding and alignment. To this end, the modelling language was enriched with the new concepts so that requirements engineers have a clear understanding of the relationship between what the system needs to do in terms of security, i.e. security requirements, and how they system will do it, i.e. security mechanisms. Second, the Secure Tropos process was enriched with a selection process that enables requirements engineers to derive an optimal set of security packages, which, in effect, is a set of preferable security mechanisms. This set maximises the overall satisfaction of the system's security requirements while respecting a set of criteria, which the engineer sets during the analysis of the system, and a given financial budget. For this purpose, the relevant modelling concepts were defined mathematically to support formal analysis. In this paper, the optimal set of security mechanisms corresponds to a baseline security solution. In the future, we aim to undertake a risk assessment to inform selection. In this way, the requirements engineer will perform threat and vulnerability analysis, update the Secure Tropos model, and then execute selection to derive the set of security mechanisms that maximise the overall system security.

References

1. McDermott, J., Fox, C.: Using abuse case models for security requirements analysis. In: Computer Security Applications Conference, 1999.(ACSAC'99) Proceedings. 15th Annual. pp. 55–64. IEEE (1999)
2. Basin, D., Doser, J., Lodderstedt, T.: Model driven security for process-oriented systems. In: Proceedings of the eighth ACM symposium on Access control models and technologies. pp. 100–109. ACM (2003)
3. Mouratidis, H.: Integrating Security and Software Engineering: Advances and Future Visions: Advances and Future Visions. Igi Global (2006)
4. Haley, C.B., Laney, R., Moffett, J.D., Nuseibeh, B.: Arguing satisfaction of security requirements. Integrating Security and Software Engineering: Advances and Future Visions pp. 16–43 (2006)
5. Fabian, B., Gürses, S., Heisel, M., Santen, T., Schmidt, H.: A comparison of security requirements engineering methods. Requirements engineering 15(1), 7–40 (2010)
6. Dubois, E., Mouratidis, H.: Guest editorial: security requirements engineering: past, present and future. Requirements engineering 15(1), 1–5 (2010)
7. Mouratidis, H., Giorgini, P.: Secure tropos: a security-oriented extension of the tropos methodology. International Journal of Software Engineering and Knowledge Engineering 17(2), 285–309 (2007)

8. Chung, L., Nixon, B., Yu, E., Mylopoulos, J.: Non-functional requirements. Software Engineering (2000)
9. Hatebur, D., Heisel, M.: Problem frames and architectures for security problems. In: Computer Safety, Reliability, and Security, pp. 390–404. Springer (2005)
10. Hatebur, D., Heisel, M., Schmidt, H.: Security engineering using problem frames. In: Emerging Trends in Information and Communication Security, pp. 238–253. Springer (2006)
11. Gupta, M., Rees, J., Chaturvedi, A., Chi, J.: Matching information security vulnerabilities to organizational security profiles: a genetic algorithm approach. Decision Support Systems 41(3), 592–603 (2006)
12. Neubauer, T., Pehn, M.: Workshop-based risk assessment for the definition of secure business processes. In: Information, Process, and Knowledge Management, 2010. eKNOW'10. Second International Conference on. pp. 74–79. IEEE (2010)
13. Viduto, V., Maple, C., Huang, W., López-Peréz, D.: A novel risk assessment and optimisation model for a multi-objective network security countermeasure selection problem. Decision Support Systems 53(3), 599–610 (2012)
14. Sawik, T.: Selection of optimal countermeasure portfolio in it security planning. Decision Support Systems 55(1), 156–164 (2013)
15. Giorgini, P., Mylopoulos, J., Nicchiarelli, E., Sebastiani, R.: Formal reasoning techniques for goal models. In: Journal on Data Semantics I, pp. 1–20. Springer (2003)
16. Amyot, D., Ghanavati, S., Horkoff, J., Mussbacher, G., Peyton, L., Yu, E.: Evaluating goal models within the goal-oriented requirement language. International Journal of Intelligent Systems 25(8), 841–877 (2010)
17. Letier, E., Van Lamsweerde, A.: Reasoning about partial goal satisfaction for requirements and design engineering. In: ACM SIGSOFT Software Engineering Notes. vol. 29, pp. 53–62. ACM (2004)
18. Bryl, V., Giorgini, P., Mylopoulos, J.: Designing cooperative is: Exploring and evaluating alternatives. In: On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE, pp. 533–550. Springer (2006)
19. Kaiya, H., Horai, H., Saeki, M.: Agora: Attributed goal-oriented requirements analysis method. In: Requirements Engineering, 2002. Proceedings. IEEE Joint International Conference on. pp. 13–22. IEEE (2002)
20. Bencomo, N., Belaggoun, A.: Supporting decision-making for self-adaptive systems: from goal models to dynamic decision networks. In: Requirements Engineering: Foundation for Software Quality, pp. 221–236. Springer (2013)
21. Feather, M.S., Cornford, S.L., Hicks, K., Kiper, J.D., Menzies, T., et al.: A broad, quantitative model for making early requirements decisions. Software, IEEE 25(2), 49–56 (2008)
22. Heaven, W., Letier, E.: Simulating and optimising design decisions in quantitative goal models. In: Requirements Engineering Conference (RE), 2011 19th IEEE International. pp. 79–88. IEEE (2011)
23. Mellado, D., Fernández-Medina, E., Piattini, M.: A common criteria based security requirements engineering process for the development of secure information systems. Computer Standards & Interfaces 29(2), 244 – 253 (2007)
24. Mead, N.R., Stehney, T.: Security quality requirements engineering (square) methodology. SIGSOFT Softw. Eng. Notes 30(4), 1–7 (May 2005)
25. Sindre, G., Opdahl, A.L.: Eliciting security requirements with misuse cases. Requir. Eng. 10(1), 34–44 (Jan 2005), http://dx.doi.org/10.1007/s00766-004-0194-4
26. Liu, L., Yu, E., Mylopoulos, J.: Security and privacy requirements analysis within a social setting. In: Requirements Engineering Conference, 2003. Proceedings. 11th IEEE International. pp. 151–161 (Sept 2003)

27. Paja, E., Dalpiaz, F., Giorgini, P.: Managing security requirements conflicts in socio-technical systems. In: Conceptual Modeling, pp. 270–283. Springer (2013)
28. Van Lamsweerde, A.: Elaborating security requirements by construction of intentional anti-models. In: Proceedings of the 26th International Conference on Software Engineering. pp. 148–157. IEEE Computer Society (2004)
29. Franqueira, V.N., Tun, T.T., Yu, Y., Wieringa, R., Nuseibeh, B.: Risk and argument: a risk-based argumentation method for practical security. In: Requirements Engineering Conference (RE), 2011 19th IEEE International. pp. 239–248. IEEE (2011)
30. Asnar, Y., Giorgini, P., Mylopoulos, J.: Goal-driven risk assessment in requirements engineering. Requirements Engineering 16(2), 101–116 (2011)
31. Lee, S.W.: Probabilistic risk assessment for security requirements: A preliminary study. In: Secure Software Integration and Reliability Improvement (SSIRI), 2011 Fifth International Conference on. pp. 11–20. IEEE (2011)
32. Houmb, S.H., Georg, G., Jürjens, J., France, R.: An integrated security verification and security solution design trade-off analysis approach. Integrating Security and Software Engineering: Advances and Future Visions/Mouratidis, Haralambos pp. 190–219 (2007)
33. Tsigkanos, C., Pasquale, L., Menghi, C., Ghezzi, C., Nuseibeh, B.: Engineering topology aware adaptive security: Preventing requirements violations at runtime. In: Requirements Engineering Conference (RE), 2014 IEEE 22nd International. pp. 203–212. IEEE (2014)
34. Van Lamsweerde, A.: Goal-oriented requirements engineering: A guided tour. In: Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on. pp. 249–262. IEEE (2001)
35. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems 8(3), 203–236 (2004)
36. Sommerville, I., Kotonya, G.: Requirements engineering: processes and techniques. John Wiley & Sons, Inc. (1998)
37. Haley, C.B., Laney, R., Moffett, J.D., Nuseibeh, B.: Security requirements engineering: A framework for representation and analysis. Software Engineering, IEEE Transactions on 34(1), 133–153 (2008)
38. Cysneiros, L.M., Sampaio do Prado Leite, J.C.: Nonfunctional requirements: From elicitation to conceptual models. Software Engineering, IEEE Transactions on 30(5), 328–350 (2004)
39. Sebastiani, R., Trentin, P.: Optimathsat: A tool for optimization modulo theories
40. Greek-Parliament: Act 3892: Electronic registration and fulfilment of medical prescriptions and clinical test referrals. FEK 189(1), 4225–4232 (November 2010)
41. Sfyroeras, V.: The electronic prescription system. Pharmacy management and communications pp. 68–69 (September 2012), `http://www.idika.gr/files/synenteyxeis/arthro_pharmacy_management_09.12.pdf`
42. Adoxx Meta-modeling platform, available at `http://www.adoxx.org`
43. Bonneau, J., Herley, C., van Oorschot, P.C., Stajano, F.: The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In: In Proceedings of the 33rd IEEE Symposium on Security and Privacy. San Francisco, CA, USA (May 2012)