A Goal Oriented Approach for Modeling and Analyzing Security Trade-Offs

Golnaz Elahi[1] and Eric Yu[2]

[1] Department of Computer Science, University of Toronto, Canada, M5S 1A4
gelahi@cs.toronto.edu
[2] Faculty of Information Studies, University of Toronto, Canada, M5S 3G6
yu@fis.utoronto.edu

Abstract. In designing software systems, security is typically only one design objective among many. It may compete with other objectives such as functionality, usability, and performance. Too often, security mechanisms such as firewalls, access control, or encryption are adopted without explicit recognition of competing design objectives and their origins in stakeholder interests. Recently, there is increasing acknowledgement that security is ultimately about trade-offs. One can only aim for "good enough" security, given the competing demands from many parties. In this paper, we examine how conceptual modeling can provide explicit and systematic support for analyzing security trade-offs. After considering the desirable criteria for conceptual modeling methods, we examine several existing approaches for dealing with security trade-offs. From analyzing the limitations of existing methods, we propose an extension to the i* framework for security trade-off analysis, taking advantage of its multi-agent and goal orientation. The method was applied to several case studies used to exemplify existing approaches.

Keywords: Security Trade-offs, Trade-off Analysis, Goal Modeling, Goal Model Evaluation.

1 Introduction

"Security is about trade-offs, not absolutes."
Ravi Sandhu

In designing software systems, security is typically only one design objective among many. Security safeguards may conflict with usability, performance, and even functionality. For example, if usability concerns are not addressed in the design of a secure system, users respond by circumventing security mechanisms [29, 30]. Achieving a balance between the intrusiveness of security mechanisms [25] and usability goals is an important consideration in designing successful secure software systems. Security goals can have their own contradictions because confidentiality, integrity, privacy, accountability, availability, and recovery from security attacks often conflict fundamentally. For example, accountability requires a strong audit trail and end-user authentication, which conflicts with privacy needs for user anonymity [25].

C. Parent et al. (Eds.): ER 2007, LNCS 4801, pp. 375–390, 2007.

Ultimately, security is about balancing the trade-offs among the competing goals of multiple actors. In current practice, security designers often adopt security mechanisms such as firewalls, access control, or encryption without explicit recognition of, and systematic treatment of competing design objectives originating from various stakeholders. This motivates the question: what conceptual modeling techniques can be used to help designers analyze security trade-offs to achieve "good enough" security?

The remaining parts of the paper are structured as follows. In section 2, we consider the criteria for a suitable conceptual modeling technique for dealing with security trade-offs. In section 3, a number of existing approaches to security trade-off analysis are reviewed and compared to the introduced criteria. From analyzing the limitations of existing methods, we propose a conceptual modeling technique for modeling and analyzing security trade-offs in a multi-actor setting. In section 4, the meta-model of security concepts is introduced, and proposed extensions and refinements to the i* notation are presented. In section 5, we describe the goal model evaluation and trade-off analysis technique. Section 6 summarizes the results of some case studies. Finally, section 7 discusses results and limitations of the approach.

2 Conceptual Modeling Criteria for Security Trade-Offs Analysis

Trade-off analysis in software design refers to achieving the right balance among many competing goals. When some goals are not sufficiently satisfied, designers need to explore further alternatives that can better achieve those goals without detrimentally hurting others. Each potential solution can have positive effects on some goals while being negative on others. A careful and systematic process for security trade-off analysis can be very challenging, because a wide range of security mechanisms, solutions and frameworks need to be considered.

To support security trade-off analysis a conceptual modeling technique should model three kinds of concepts: i) Goals, ii) Actors and iii) Security specific concepts.

i) Goals: Security trade-offs are conflicts among design objectives that originate from stakeholder goals. While selecting a solution among security alternatives is difficult, the more fundamental problem is that designers need to decide about alternatives security mechanisms subject to multiple factors such as cost, time-to-market, non-functional requirements (NFRs), security policies, standards, and individual goals of various stakeholders. Therefore, the "goal" concept is a basic modeling construct required in the conceptual modeling technique for dealing with trade-offs. The technique should provide means for structuring the contributions to goals and modeling the extents and measures of goals satisfaction, contribution and competition. The measures could be quantitative or qualitative. Quantitative approaches can greatly simplify decision making, but can be difficult to apply due to lack of agreed metrics or unavailability of accurate measures. The modeling technique should be able to support analysis despite inaccurate or incomplete knowledge about goals.

ii) Actors: Design objectives typically come from multiple sources and stakeholders such as system's users, administrators, top managers, project managers, and customers. The conceptual modeling technique should be able to model multiple actors that impose competing goals on the designer, and should provide means to

trace back goals to the actors. The modeling technique should be able to model trade-offs that occur within a single actor or across multiple actors.

iii) Security Specific Concepts: The conceptual modeling technique that enables security trade-off analysis should model security specific concepts such as threats, vulnerabilities, and safeguards. Threats can be viewed as malicious actors' goals. Conflicts among stakeholders' goals are usually unavoidable, and the designer needs to balance the trade-offs among conflicting goals. In contrast, threats and attacks must be mitigated. In addition, decision makers need a measurable expression of the security level of solutions [21]; therefore, the modeling technique should provide means to model to what extent attacks are successful, how attacks influence on goals, whether countermeasures control the threats, and whether the goals are at risk.

The modeling concepts need to be accompanied with a procedure for evaluating security alternatives. The proper trade-off analysis method should evaluate the impact of each alternative on goals and potential threats. It should answer to what extent the goals are satisfied or denied, threats are contained, and vulnerabilities are patched. The procedure should be able to analyze the trade-offs in the face of incomplete or inaccurate knowledge about goals' contributions and security measures.

3 Existing Approaches to Security Trade-Off Analysis

Many approaches have been proposed to model security aspects of the software systems. The notion of "abuse case" [14] and UMLsec modeling language [15] are examples of security specific conceptual modeling approaches for modeling security requirements and aspects of the system.

In recent years, agent and goal oriented frameworks in Requirements Engineering have emerged as new approaches to the analysis and design of complex software systems. Examples of such frameworks are KAOS [1], the NFR framework [10], the i* framework [7], and Tropos [2]. Several approaches such as [3, 5, 6, 16, 17, 18] propose frameworks for modeling and analyzing security concepts by taking advantage of agent and goal oriented techniques. The majority of these approaches employ qualitative trade-off analysis, while [16] suggests a quantitative approach for analyzing security requirements. In [22], probabilistic inference on security influence diagrams is used to support trade-off analysis using Bayesian Belief Nets (BBN). The approach in [23] proposes a framework of core security requirements artefacts to describe the security requirements. The meta-model of the core artefacts includes concepts such as assets, threats, security goals, functional requirements, and security requirements. In [20], using the core security artefacts, the authors propose a framework for security requirements elicitation and analysis.

In this section, we review three selected methods for modeling and analyzing security trade-offs as representative of existing approaches. We study Architecture Tradeoff Analysis Method (ATAM) [11] as a general purpose and widely used architectural trade-off analysis method which considers security. We study agent and goals oriented approaches for dealing with security trade-offs. Security Verification and security solution Design Trade-off analysis (SVDT) [21] and Aspect-Oriented Risk-Driven Development (AORDD) [27] are studied as representatives of quantitative analysis methods. We study how well these approaches are matched with the criteria discussed in the previous section.

3.1 ATAM

Bass et al. [11] introduces a framework to model quality attributes and architectural options using the notion of scenarios and tactics respectively. A quality attribute scenario is a quality-attribute-specific requirement, and consists of six parts: Source of stimulus, Stimulus, Environment, Artifact, Response, and Response measure. Achievement of quality scenarios relies on tactics. ATAM is an evaluation method to analyze whether an architecture decision satisfies particular quality goals. ATAM helps designers to prioritize scenarios and evaluate alternative tactics using a "Quality Attribute Utility Tree". Scenarios that have at least one high priority of importance or difficulty are chosen for a detail analysis to examine if the selected tactics satisfy the scenario.

The result of the analysis is an "Architectural Approach Analysis" table for each quality scenario. In this table, evaluators identify and record sensitivity, tradeoff, risks and non-risks points for alternative tactics. Sensitivity and tradeoff points are architectural decisions that have effect on one or more quality attributes, the former positively and the latter negatively. In ATAM, a risk is defined as an architectural decision that may lead to undesirable consequences, and non risk points are defined in the opposite way. The conceptual elements related to trade-offs in ATAM may be captured in a meta-model as in Fig. 1.

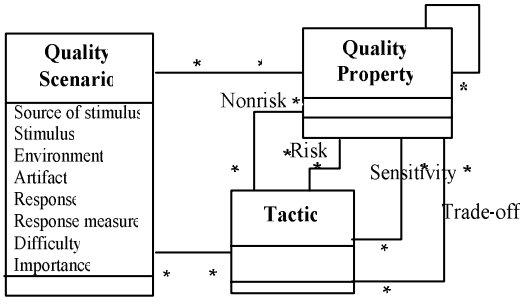

Fig. 1. Meta-model of trade-off elements in ATAM

3.2 SVDT/AORDD Approach

Houmb et al. [21] propose the SVDT approach using UMLsec for modeling security solutions. UMLsec is used to specify security requirements, and UMLsec tools verify if the design solutions satisfy the security requirements. Design solutions that pass the verification are then evaluated using security solution design trade-off analysis. A complementary framework on AORDD provides a risk assessment process and cost-benefic trade-off analysis. AORDD and SVDT use BBN to compute Return on Security Investment (RoSI).

Fig. 2 illustrates the relationship between the main concepts involved in AORDD risk assessment, which specifies the structure of the inputs to the AORDD cost-benefit trade-off analysis. The result of risk assessment is a list of misuses which need security treatments. This list, alternative security treatments, and fixed trade-off parameters such as budget, time-to-market, and policies are fed into the BBN to compute the RoSI.

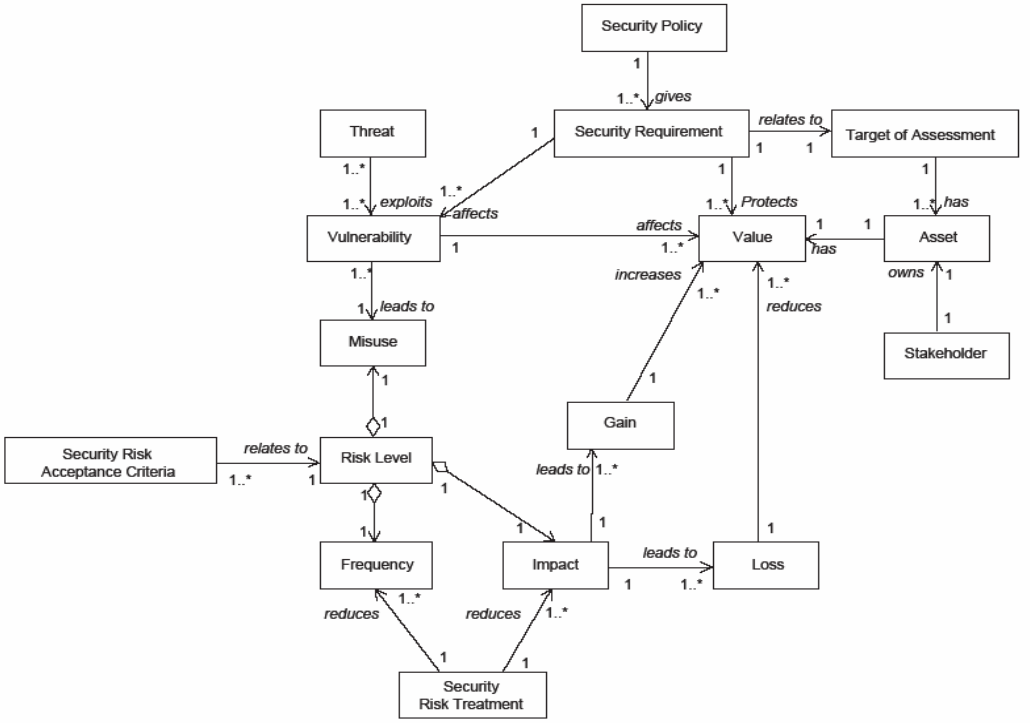

Fig. 2. ARODD risk assessment main concepts and relation [27]

3.3 Secure Tropos/i*

The proposed approaches in [3, 5, 6, 17, 18] take advantage of the i* and Tropos frameworks. In these approaches, systems are modeled as intentional agents collaborating or competing with each other to achieve their goals. Security issues arise when some actors, while striving to achieve their own goals, intentionally or unintentionally threaten other actors' goals; therefore, agent and goal oriented approaches provide a suitable basis for dealing with competing goals of multiple actors.

The approach in [3] suggests using relationships among strategic actors for analyzing security requirements. In [3], potential attackers of the systems are distinguished from

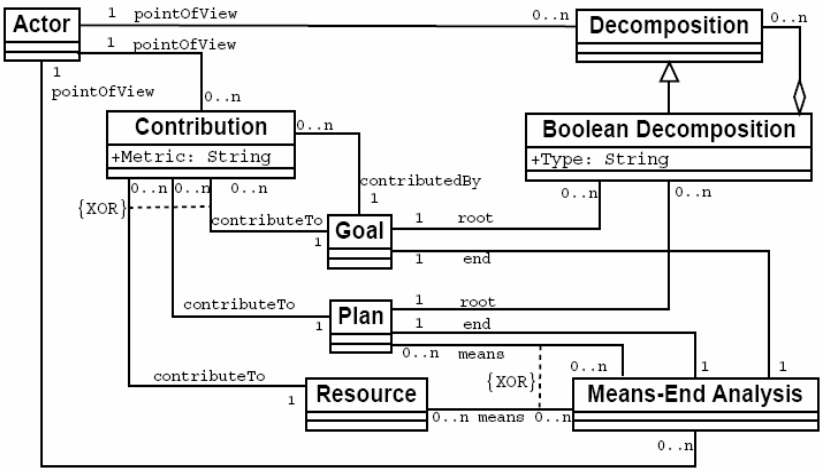

Fig. 3. Part of Tropos meta-model for goals and related concepts [31]

other actors of the system. [5] proposes a methodological framework for dealing with security requirements based on the i* notation. In [6], a framework known as Secure Tropos for modeling and analyzing security requirements based on the notions of trust, ownership, and permission delegation is developed. In [17, 18], the "threat" and "security constraint" modeling elements are added to the i* meta-model. "Threat" elements are employed in the "security diagram" to express potential violation against the security goals, and "security constraints" are used to impose security requirements on actors' dependencies. The meta-model of related concepts to the Tropos goal model, which is the core of all these approaches, is depicted in Fig. 3.

3.4 Limitations of Existing Approaches

In ATAM, trade-offs among quality scenarios and tactics in the "Architectural Approach Analysis" table are indirect and implicit, since trade-off and risk points, instead of referring to quality scenarios, refer to affected quality properties. ATAM lacks considering the impact of each tactic on stimuli of security scenarios (attacks). The impact of tactics on quality attributes are not captured qualitatively or a quantitatively. Finally, the framework of scenarios, tactics and ATAM method does not provide means to model and analyze security concepts specifically.

SVDT and AORDD rely on quantitative computation and probabilistic inference for trade-off analysis. This requires the software designers obtain the quantitative measures of the impact of misuses and solutions. The major limitation is the inaccuracy or unavailability of qualitative data on the impact of misuses and solutions especially in the early stages of the development lifecycle.

Generally, the suggested BBN topologies in SVDT and AORDD do not consider a more general source of trade-off inputs such as NFRs and functionalities, and the trade-off inputs to the designed BBN are limited to factors such as budget, laws and regulation. Besides, the AORDD meta-model of risk assessment concepts (Fig. 2) does not consider the relation between "security risk treatment" and other entities such as "security requirement", "threat", and "vulnerability". The AORDD meta-model could be strengthen by considering more general concepts such as goals, other quality requirements, and actors.

In SVDT and AORDD, the trade-off inputs and information are given to a BBN, and the final RoSI is computed automatically, which makes the analysis efficient. Since, the relationships between various states of the variables are specified in terms of the node probability matrix in BBN, this automatic trade-off analysis process can be traced by the designer. However, it may be difficult for the designer to follow what aspects of the design caused the difference in the final results.

Although agent and goal oriented approaches provide a proper conceptual basis for modeling and analyzing security trade-offs, a mechanism for such analysis has not been elaborated in these frameworks. The method in [5] lacks a direct and explicit way to model the competition among malicious and non-malicious actors' goals, and trade-off modeling among goals is limited to the non-malicious actors. The proposed framework in [6] does not support modeling security concepts such as malicious behavior. In [17, 18], threats are modeled explicitly as a distinct construct in the "security diagram", but they are not traced to the threats' source actors, and the relation between countermeasures and threats are not elaborated.

Table 1 summarizes a comparison of the studied approaches based on the evaluation criteria from section 2.

Table 1. A comparison of existing approaches based on the criteria of the conceptual modeling technique for security trade-off analysis

Method Requirement	ATAM	SVDT/AORDD	i*/Tropos
Goals	Expressed in terms of scenarios	Limited to security requirements and fixed BBN parameters	Explicit goals
Relations of goals	Not model explicitly	Limited to UMLsec models	Modeled using contribution links
Extents of goal satisfaction	Not expressed	Quantitatively	Qualitatively
Goals contribution structure	Utility tree doesn't capture the contributions of scenarios	Not modeled	Modeled in terms of sub goals and contribution links
Multiple actors	Expressed implicitly by multiple stimuli sources	Not modeled	Modeled in terms of agents/actors/ roles/ positions
Trade-off within a single actor or across actors	Single actor	Single actor	Single and multiple actors
Security Specific Trade-off Concepts	Not modeled	Some concepts are modeled	Some concepts are modeled
Trade-off analysis method	Qualitative analysis	Quantitative analysis	Qualitative and quantitative analysis

4 The Security Trade-Offs Modeling Notation

We propose a meta-model of security concepts for systematically addressing security trade-offs (Fig. 4), considering the limitations of existing approaches and reviewing well known security knowledge sources such as NIST's guidelines and standards like [19], CERT [26], and widely used textbooks such as [4, 13]. The core of the meta-model is the concepts of goals and actors guided by the criteria of the conceptual modeling technique that enables security trade-offs analysis.

The proposed notation builds upon the i* framework which provides a notation to model *actors*, their *goals* and intentional *dependencies* and competitions among the actors. Actors achieve goals on their own or depend on each other for goals to be achieved, tasks to be performed, and resources to be furnished. Quality goals, which do not have clear-cut criteria for satisfaction degree, are modeled as softgoals. *Means-ends* relation between goals and tasks is used to model alternative ways to achieve a goal [8]. However, the i* notation lacks explicit modeling constructs for concepts such as threats and vulnerabilities. In this section, we propose some extensions to the i* notation, which provide conceptual structures for modeling and analyzing security trade-offs.

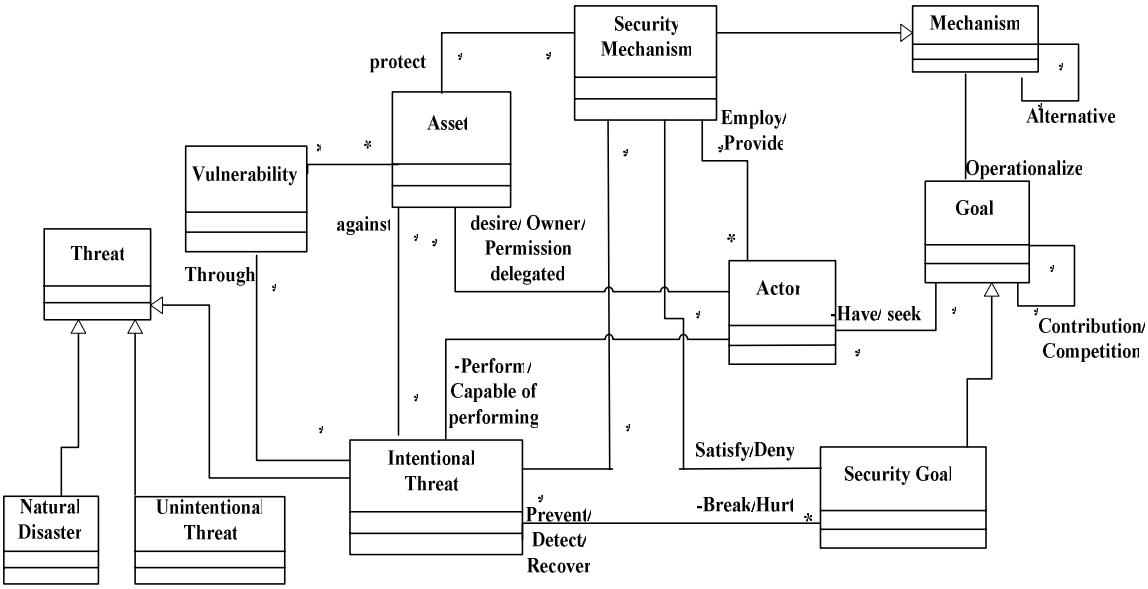

Fig. 4. Meta-model of security concepts used in proposed modeling notation

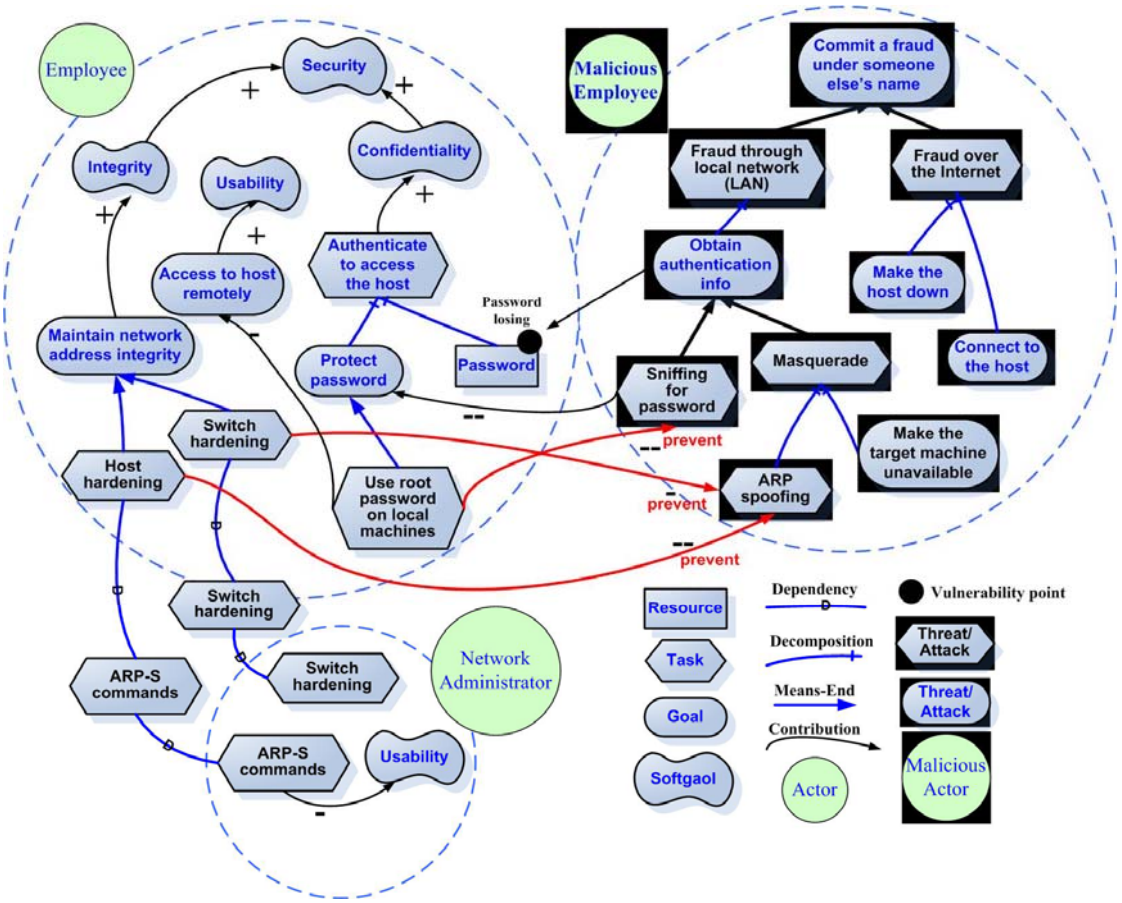

Fig. 5. Example of a multi-actor system modeled using the proposed notation

4.1 Malicious Actor, Goals and Tasks

Actors depend on, or compete with each other to achieve their goals. Meanwhile, malicious actors try to achieve their own goals. Representing a malicious actor with a different modeling construct in i* was first employed in [3] by highlighting them with a black shadow rectangle. This notation was used to model malicious goals in [5]. We make use of this notation in which malicious goals, softgoals, tasks, and actors are highlighted by a black shadow rectangle. By distinguishing malicious modeling elements from non-malicious ones, we emphasize studying the attackers' goals and tasks. Although attacker's behavior might be partially unknown and generic, an important aspect of trade-off analysis depends on studying attackers' options and the risks they pose to other actors' goals.

A security threat is any malicious behavior that interferes with the achievement of other actors' goals. For example, in Fig. 5, Malicious Employee is the malicious actor whose goal is to Commit a fraud under someone else's name, either through the local network or over the Internet. Threats might be unintentional or caused by natural disasters. In this paper, we mainly focus on the security threats caused by actors with malicious intent.

4.2 Assets, Services and Vulnerabilities Points

An asset is any thing that has a value for the organization [13]. Physical resources, information, and people can be counted as assets. In this way, the asset concept is well matched with the "resource" modeling element in i*. Assets can be the services an organization offer or receive, and in this case, can be represented by tasks or goals that actors offer to the "depender" actors.

In security analysis, a vulnerability point is any weakness in, or back door to the system [13]. For example, it is said that *buffer overflow* and *password cracking* are the most common vulnerability points of many computer systems [4]. Generally, a vulnerability point corresponds to an asset or service, and attackers usually try to achieve malicious goals through a vulnerability to reach an asset. In the i* notation, tasks are usually decomposed to goals, softgoals, other tasks, and resources. In this way, harm of an attack can be indicated by the cost of the failed task that relies on the compromised assets. In a similar approach in [20, 23], threats are described in terms of assets, the action that exploits the assets, and the subsequent harm.

Although vulnerability that arises from dependencies among actors is a fundamental concept in i* in [5], there is no explicit modeling construct in i* to represent vulnerability points. We add the vulnerability point modeling element to i*, accompanied with a graphical notation to connect a vulnerability point to the corresponding attacks, and to attach it to a resource. For example, in Fig. 5, to protect confidentially employees are authenticated by the host. Hence, Password is one of the employees' assets they need to protect. On the other hand, Password losing is one of the most important vulnerability points in computer systems. Sniffing for password is an attack against the goal of Protect password. Through this attack and Password losing vulnerability point, the goal of Fraud under someone else's name can be satisfied, and the attacker gains a valuable asset: the Password.

4.3 Relation Between Attacks and Security Mechanisms

In the i* notation, relation between softgoals and other elements is modeled by contribution links [7]. If an element hurts a softgoal, yet not enough to prevent it, the contribution link type is "-". If the element is sufficient to prevent a softgoal, the contribution link type is "--". This qualitative approach is used to model the impact of attacks on softgoals and the impact of security mechanisms on malicious tasks and goals. In security engineering, various mechanisms have different effects on attacks. Contribution of mechanisms to attacks are categorized as 1) Prevent 2) Detect 3) Recover [13]. These categories are added as attributes on the contribution links. "Detect" and "Recover" contribution links may partially mitigate the effect of attacks. Mechanisms which are related to the attacks with "Detect" contribution links can not control any attack. Similarly, "Recover" contribution links indicate that the mechanisms can not control the attack either, but the mechanism would be used to recover the system after the attack. This link would be useful to express availability and integrity goals that rely on recovering the system after the failure. To sufficiently counteract an attack, security mechanisms must be related to the attack with a "Prevent" contribution link.

4.4 Expressing Trade-Offs by the Proposed Conceptual Structure

The proposed approach provides the means to model goals, and trace them back to the source actors. In this approach, trade-offs among goals are modeled by contribution links. Through contribution link types of -, --, + and ++ [10], the qualitative effect of alternative solutions are propagated to the other goals. The i* notation offers the conceptual structure to model trade-offs between refined sub-goals of high level goals as well. For example, in Fig. 5, the employee can Use root password on local machines to completely prevent the attack of Sniffing for password [4]. However, this security solution contributes negatively to the Access to host remotely goal, and it has negative influence on the Usability softgoal consequently. In this way, the trade-off among usability and security is modeled through relationships among their refined sub-goals.

5 Trade-Off Analysis and Decision Making

In the previous section, we proposed a conceptual modeling technique for modeling security trade-offs. In this part, we propose a trade-off analysis method for use with the trade-off model. Designers need to balance the trade-offs to mitigate the security risks and yet satisfy the goals of multiple actors. A goal is at risk when it may be denied (partially or fully) by the successful behavior of malicious actors. Partially or fully denial of goals are expressed through contribution links of type "-" and "--". Hence, for trade-off analysis designers need to examine available alternative security solutions, and verify the impacts of each one on attacks and goals to finally select the one which fits with goals of multiple actors. Goal model evaluation is the procedure to ensure that actors' top level goals are satisfied by the choices they have made [12]. The security goal model evaluation, consisting of interactive qualitative reasoning, is based on the method proposed in [10] and refined in [12]. Fig. 6 depicts the proposed security trade-off analysis procedure.

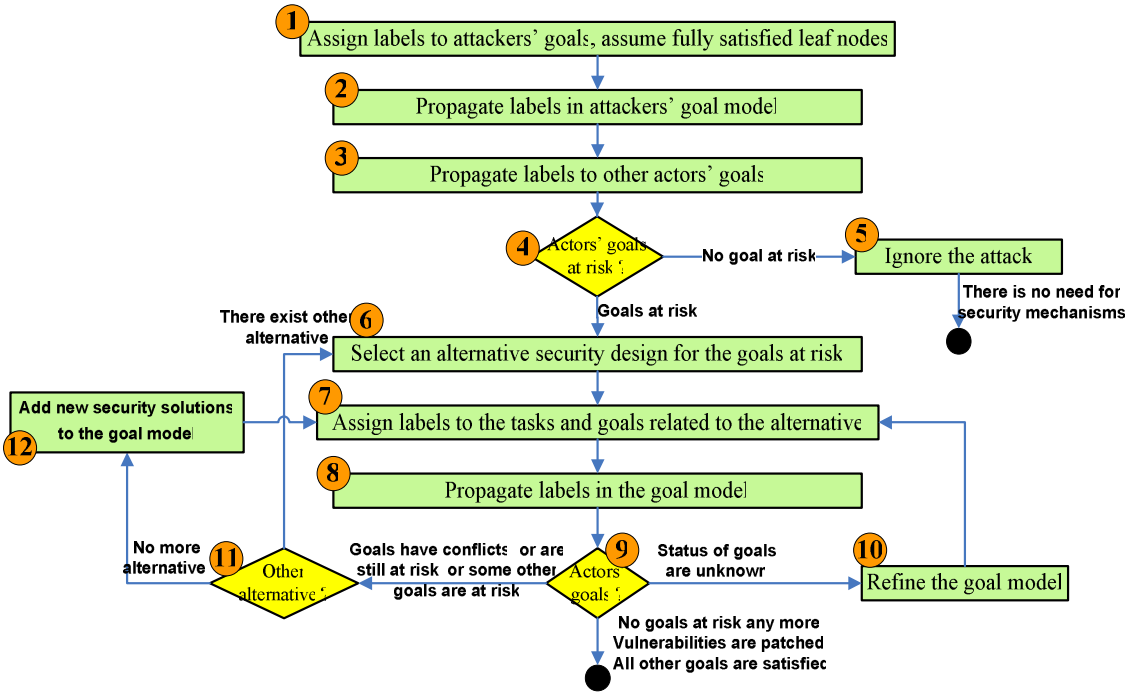

Fig. 6. Security trade-off analysis procedure

In the first step, evaluator assumes that attackers are successful in performing tasks and satisfying their goals, since attackers are usually external actors that designer has no sure knowledge of their abilities and skills. Therefore, the leaf nodes in attackers' goal model are labeled fully satisfied. This assumption does not imply that the risk of attacks is definite, as it is possible that evaluation of attackers' goal model yields to denial of higher goals of attacker. The leaf labels are propagated to upper goals. Once the impact of malicious actors' behavior is propagated to the entire goal model, the evaluator assigns labels to the tasks and goals that operationalize security mechanism (step 7). This label indicates the evaluator's judgment about the success of the actor in performing a security task or achieving a security goal. This judgment could be based on knowledge of previous experiences, empirical studies, or subjective knowledge [21].

In step 9, the goal model indicates which goals are fully or partially satisfied or denied for the examined security solution. The procedure iterates until a security design solution is found that, based of the evaluator's perception, satisfies an acceptable configuration of goals. However, the evaluator may prefer to examine further alternatives to select the security design solution that satisfies more goals. After evaluating an alternative, the status of some goals may be unknown, prompting the designer to elaborate on the models (step 10). In case of conflict of goals, other alternatives should be examined to resolve the conflicts (step 11). An example of security goal model evaluation is shown in Fig. 7.

Propagation of the labels is based on the contribution types and rules summarized in Table 2. [12] provides details about aggregation rules for multiple contributions. The rules provided in Table 2 are merely valid for the "Prevent" contribution type, as we discussed earlier that recovering from, or detecting an attack do not lead to controlling the attack.

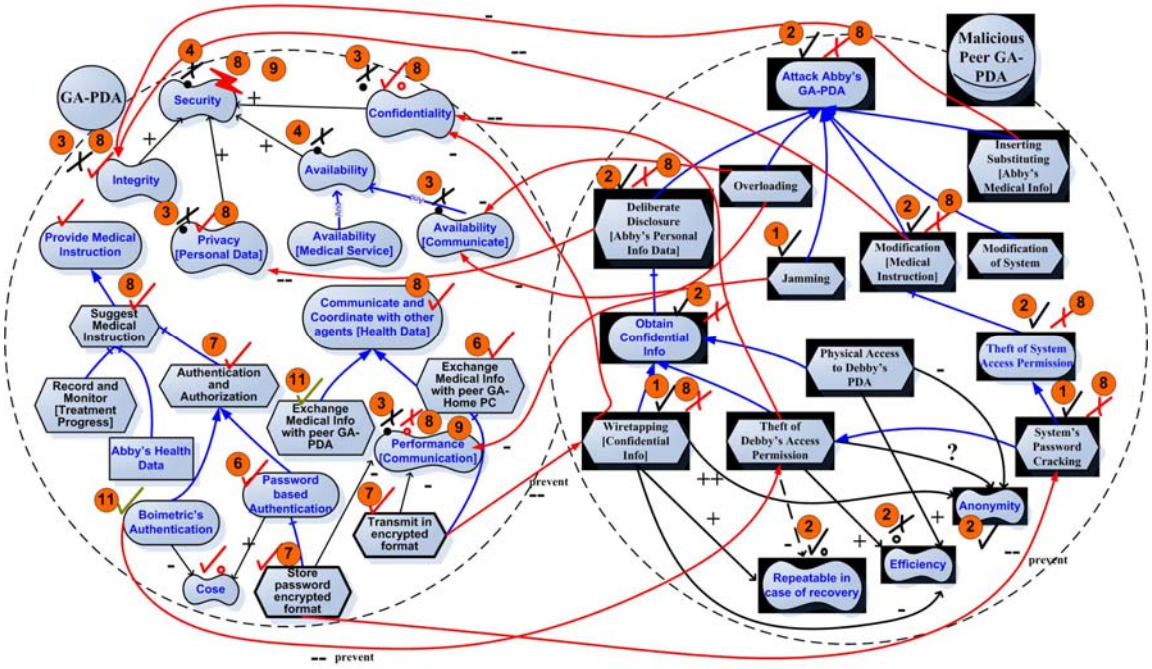

Fig. 7. Part of the attacker and countermeasures model for the Guardian Angel case study annotated with the evaluation steps introduced in Fig. 6

Table 2. Evaluation labels and propagation rules from [10, 12]

Child Node		Contribution Type (Prevent)				
Label Name	Symbol	++	+	-	- -	?
Satisfied	✓	✓	✓.	✗.	✗	?.
Weakly Satisfied	✓.	✓.	✓.	✗.	✗.	?.
Conflict	↯	↯	↯	↯	↯	?.
Unknown	?.	?.	?.	?.	?.	?.
Weakly Denied	✗.	✗.	✗.	✓.	✓.	?.
Denied	✗	✗	✗.	✓.	✓.	?.

6 Case Studies

In developing the proposed notation, we modeled a number of NIST guidelines [19] and security engineering knowledge in [4], using the extended i* notation. In addition to example cases, we applied the notation to three example cases originally used to illustrate other approaches to security trade-offs [28]. In the first example case, we modeled and analyzed the eSAP system, an agent-based health and social care system, which was used as the case study system in [16, 17, 18]. In the second example case system, we modeled and analyzed a simple Course Registration system, using the proposed extensions to the i* and the framework proposed in [11]. Due to space limitations, we present only a third case study in the following. Details of the case studies can be found in [28].

The Guardian Angel (GA) [9] is a patient and physician supporting system using software agents, which is studied in [5]. In vulnerability analysis in [5], each dependency is examined as a potential threat against the system. In this approach, each actor is studied in two roles: its regular role, and its potential malicious role. One of the actors in the dependency relation is substituted by its corresponding attacker. For each malicious actor, a number of attacks and threats are identified, the impact of threats and corresponding security safeguards are added to the goal model. However, resulting models do not capture goals and intentions of the attacker. The goal model evaluation is limited to evaluating impact of security safeguards on threats, while the safeguards may affect other goals such as performance and usability. Generally, the approach in [5] does not consider modeling security mechanisms in terms of the trade-offs they impose to the other goals.

Fig. 7 gives a part of the trade-off models and analysis of GA system using the proposed approach in this paper. The model captures the potential intentions behind an attack, since deciding among different countermeasures depends on the attacker's goals. For example, the designer needs to differentiate between goals of a professional hacker and intentions of a curious kid to select proper security mechanisms. The resulting goal model captures the effects of each alternative attack on malicious and non-malicious actors' goals and softgoals. As a result, the designer can evaluate the risk of threats, and select a more appropriate countermeasure for attackers' behavior based on the consequence of malicious actors' behavior. In the goal model of Fig. 7, the designer decides to employ Authentication and Authorization with Password based Authentication (Steps 6 and 7). The goal model evaluation yields a fully satisfied Privacy goal with Confidentiality partially satisfied, while Performance is partially denied.

7 Conclusion and Future Work

In this paper, we began by considering the criteria for a conceptual modeling technique that enables designers to model and analyze security trade-offs among competing goals of multiple actors to achieve a good-enough security level. We studied existing approaches to trade-off analysis, and identified limitations of these approaches. Based on the evaluation criteria and limitation of previous works, we proposed extensions to the i* notation for modeling and analyzing security trade-offs of a multi-actor system. The proposed modeling notation is accompanied with a qualitative trade-off analysis procedure based on goal model evaluation methods. The procedure provides the designers with assessment of security mechanisms' impact on actors' goals and threats. Table 3 gives the comparison of the proposed approach with the evaluation criteria.

Although the i* notation provides the proper basis for modeling and analyzing trade-offs, the models become complex and inefficient when the goal models scale. Another limitation of the proposed approach is that a comprehensive source of knowledge of security mechanisms and corresponding contributions does not exist.

Table 3. Comparison of proposed approach with the conceptual modeling technique's criteria

Method Requirements	Suggested approach
Goals	Modeled using goals and softgoals elements of i*
Relations of goals	Modeled using i* goal dependency modeling. Competition and trade-offs are modeled by contribution links and relation between attacks and goals.
Extents and measures of goals	Modeled qualitatively by contribution links of type -, - -, +, ++
Inaccurate or incomplete knowledge	Modeled by unknown contribution links, and goal model evaluation propagates them to related elements
Goals contribution structure	Structured by sub-goals, task decomposition, contribution links
Multiple actors	Multiple malicious and non malicious actors can be modeled
Trade-off within a single actor or across actors	Trade-off within a single actor or across actors can be modeled
Security specific trade-off concepts	Modeled by security extensions to i* notation derived from the meta-model
Trade-off analysis method	Security goal model evaluation technique supports qualitative trade-off analysis

In future work, we aim to conduct empirical studies of how security designers make trade-offs in practice, and to adapt the proposed systematic trade-off analysis framework for integration into everyday design practice. We will also build a security requirements and design knowledge base to gather and catalogue reusable knowledge about security trade-offs. Tool support for managing and applying security knowledge will also be studied.

Acknowledgments. Financial support from Natural Science and Engineering Research Council of Canada and Bell University Labs is gratefully acknowledged.

References

1. Dardenne, A., van Lamsweerde, A., Fickas, S.: Goal-Directed Requirements Acquisition. The Science of Computer Programming 20, 3–50 (1993)
2. Castro, J., Kolp, M., Mylopoulos, J.: A requirements-driven development methodology, In Proc. of the 13th Int. Conf. on Advanced Information Systems Engineering, CAiSE'01. In: Dittrich, K.R., Geppert, A., Norrie, M.C. (eds.) CAiSE 2001. LNCS, vol. 2068, pp. 108–123. Springer, Heidelberg (2001)
3. Liu, L., Yu, E., Mylopoulos, J.: Analyzing Security Requirements as Relationships among Strategic Actors. In: 2nd Symp. on Requirements Engineering for Information Security (SREIS) (2002)
4. Anderson, R.: Security Engineering: a guide to Building dependable Distributed systems. John Wiley and Sons, Chichester (2001)

5. Liu, L., Yu, E., Mylopoulos, J.: Security and Privacy Requirements Analysis within a Social Setting. In: IEEE Joint Int. Conf. on Requirements Engineering, pp. 151–161. IEEE Computer Society Press, Los Alamitos (2003)
6. Giorgini, P., Massacci, F., Mylopoulos, J., Zannone, N.: Modeling Security Requirements through Ownership, Permission and Delegation. In: 13th IEEE Int. Requirements Engineering Conf, pp. 167–176. IEEE Computer Society Press, Los Alamitos (2005)
7. Yu, E.: Modeling Strategic Relationships for Process Reengineering, PhD thesis, Department of Computer Science, University of Toronto, Canada (1995)
8. Yu, E.: Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering. In: Proc. of the 3rd IEEE Int. Symp. on Requirements Engineering, pp. 226–235 (1997)
9. Szolovits, P., Doyle, J., Long, W.J.: Guardian Angel: Patient-Centered Health Information Systems: MIT/LCS/TR-604, Available at: http://www.ga.org/ga
10. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. Kluwer Academic Publishing, Dordrecht (2000)
11. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice, 2nd edn. Addison Wesley, London, UK (2003)
12. Horkoff, J.: Using i* Models for Evaluation, Masters Thesis, University of Toronto, Department of Computer Science (2006)
13. Pfleeger, C.P., Pfleeger, S.L.: Security in Computing, 3rd edn. Prentice-Hall, Englewood Cliffs (2002)
14. McDermott, J., Fox, C.: Using Abuse Case Models for Security Requirements Analysis. In: McDermott, J., Fox, C. (eds.) Proc.15th. IEEE Annual Computer Security Applications Conf., pp. 55–64. IEEE Computer Society Press, Los Alamitos (1999)
15. Jürjens, J.: Secure Systems Development with UML. Springer Academic Publishers, Germany (2004)
16. Bresciani, P., Giorgini, P., Mouratidis, H.: On Security Requirements Analysis for Multi-Agent Systems. In: Lucena, C., Garcia, A., Romanovsky, A., Castro, J., Alencar, P.S.C. (eds.) Software Engineering for Multi-Agent Systems II. LNCS, vol. 2940, pp. 35–48. Springer, Heidelberg (2004)
17. Mouratidis, H., Giorgini, P., Manso, G., Philp, I.: A Natural Extension of Tropos Methodology for Modelling Security. In: Proc. of the Workshop on Agent-oriented methodologies, at OOPSLA, pp. 91–103 (2002)
18. Mouratidis, H., Giorgini, P.: Manso, Modelling Secure Multiagent Systems. In: the 2nd Int. Conf. on Autonomous Agents and Multiagent Systems, pp. 859–866 (2003)
19. Grance, T., Stevens, M., Myers, M.: Guide to Selecting Information Technology Security Products, Recommendations of the National Institute of Standards and Technology, NIST Special Publication 800–836 (2003)
20. Haley, C.B., Moffett, J.D., Laney, R., Nuseibeh, B.: A framework for security requirements engineering. In: Software Engineering for Secure Systems Workshop (SESS'06), pp. 35–42 (2006)
21. Houmb, S.H., Georg, G., Jürjens, J., France, R.: An Integrated Security Verification and Security Solution Design Trade-off Analysis. In: Integrating Security and Software Engineering: Advances and Future Visions, pp. 190–219. IDEA Group Publishing, USA (2007)
22. Johnson, P., Lagerstrom, R., Norman, P., Simonsson, M.: Extended Influence Diagrams for Enterprise Architecture Analysis. In: Enterprise Distributed Object Computing Conference, EDOC '06. 10th IEEE Int., pp. 3–12. IEEE Computer Society Press, Los Alamitos (2006)

23. Moffett, J.D., Haley, C.B., Nuseibeh, B.: Core Security Requirements Artefacts, Department of Computing, The Open University, Milton Keynes UK, Technical Report 2004/23 (2004)
24. Mayer, N., Rifaut, A., Dubois, E.: Towards a Risk-Based Security Requirements Engineering Framework, 11th Int. Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'05) (2005)
25. Sandhu, R.: Good-Enough Security: Toward a Pragmatic Business-Driven Discipline," IEEE Internet Computing, Vol. IEEE Internet Computing 07(1), 66–68 (2003)
26. US-CERT Vulnerability Notes Database, United States Computer Emergency Readiness Team, http://www.kb.cert.org/vuls
27. Houmb, S.H., Georg, G.: The Aspect-Oriented Risk-Driven Development (AORDD) Framework. In: Proc. of the Int. Conf. on Software Development (SWDC.REX), pp. 81–91 (2005)
28. Elahi, G., Yu, E.: A Goal Oriented Approach for Modeling and Analyzing Security Trade-Offs, Technical Report, University of Toronto, Department of Computer Science, Available (2007), at http://istar.rwth-aachen.de/tiki-index.php?page=Security+Requirements+Engineering
29. Sasse, M.A.: Computer Security: Anatomy of a Usability Disaster, and a Plan for Recovery, Workshop on Human-Computer Interaction and Security Systems, CHI 2003, Fort Lauderdale (2003)
30. De Witt, A.J., Kuljis, J.: Aligning Usability And Security-A Usability Study Of Polaris. In: Proc. of the Symp. On Usable Privacy and Security (2006)
31. Susi, A., Perini, A., Mylopoulos, J.: The Tropos Metamodel and its Use. Informatica 29, 401–408 (2005)