R. Rios, C. Fernandez-Gago, and J. Lopez, "Privacy-Aware Trust Negotiation", *12th International Workshop on Security and Trust Management (STM)* vol. LNCS 9871, pp. 98-105, 2016.
http://doi.org/10.1007/978-3-319-46598-27
NICS Lab. Publications: https://www.nics.uma.es/publications

Privacy-Aware Trust Negotiation

Ruben Rios[1], Carmen Fernandez-Gago[1], and Javier Lopez[1]

Network, Information and Computer Security (NICS) Lab,
University of Malaga, Spain
{ruben,mcgago,jlm}@lcc.uma.es

Abstract. Software engineering and information security have traditionally followed divergent paths but lately some efforts have been made to consider security from the early phases of the Software Development Life Cycle (SDLC). This paper follows this line and concentrates on the incorporation of trust negotiations during the requirements engineering phase. More precisely, we provide an extension to the SI* modelling language, which is further formalised using Answer Set Programming specifications to support the automatic verification of the model and the detection of privacy conflicts caused by trust negotiations.

Keywords: Secure Software Engineering, Requirements Engineering, Goal-Oriented Modelling, Privacy, Trust

1 Introduction

In recent years, the number of vulnerabilities and attacks present in software systems have led to a growing interest in incorporating security from the early phases of the SDLC. Also, the notions of trust and privacy are gaining momentum due to the proliferation of new computing paradigms where devices from different security domains interact with each other and exchange valuable information. This paper deals with the confluence between secure software engineering, privacy and trust.

This paper presents a framework for identifying privacy threats caused by the uncontrolled disclosure of information during a trust negotiation. Trust negotiation systems [15] model how the exchange of information between entities, wishing to establish a relationship, is done. Our framework is capable of modelling such systems and detecting potential threats automatically early in the specification and design of the system. Thus, it facilitates the incorporation of privacy-aware trust negotiations in the development of socio-technical systems. To that end, we build our framework as an extension of SI* [6], which is designed to capture the objectives and relationships between various entities within an organisational setting and already supports the definition of some security concepts, such as delegation and trust.

The proposed framework models trust negotiations as a relationship between two entities that pursue a common goal[1]. To that end, they need to exchange

[1] N.B. That we assume that the goal is always common. The consideration of different goals is out of the scope of the paper

information that may be sensitive and thus impact their privacy. Therefore, informational resources are labelled with a particular sensitivity level that defines how important it is to keep control of this information. The framework detects inconsistencies with the privacy policy by comparing it with the sensitivity level of the resources being exchanged during a trust negotiation.

The rest of this paper is organised as follows. Section 2 introduces the related work in the area whereas Section 3 deeps into SI*, which is the basis of our work. Our proposal for a privacy-aware trust negotiation methodology is presented in Section 4 and its formalisation in Section 5. Section 6 concludes the paper and outlines the future work.

2 Related Work

A common approach to requirements engineering is to follow a goal-oriented methodology based on concepts such as actors and goals rather than on programming concepts. The KAOS framework [14] is based on temporal logics and Tropos [3] is founded on the i* organisational modelling framework [16]. These frameworks have been extended to deal with security requirements. The notions of obstacle [13] and anti-goal [12] have been introduced to KAOS. Secure Tropos [7] extends Tropos by making explicit ownership relationships and actor entitlements. The modelling language used by Secure Tropos is SI* [6], which incorporates a number of security concepts. New goal-oriented methodologies have recently been proposed, such as STS [10], which puts more emphasis on authorisation and the notion of document.

Although research on security engineering is extensive, privacy has traditionally been left out. The only support to privacy in most of these frameworks, including SI* and STS, is considering it as data confidentiality. Notwithstanding, several privacy engineering methods exist. Authors in [8] tackle privacy issues by defining a set of best practices in the different stages of the development process. LINDDUN [4] defines a mapping among privacy threats and the software components in order to elicit privacy requirements. Pris [5] models privacy requirements as organisational goals and later privacy patterns are used for identifying architectures.

The closest approach to ours is the one followed by PP-Trust-$\mathcal{X}$ [11]. The main difference with our proposal is that our framework detects privacy conflicts during the requirements engineering phase rather than at runtime. To the best of our knowledge no other works address this problem early in the SDLC.

3 The SI* Modelling Language

We provide next an overview of SI* modelling language [6], that is, a description of core elements and some extensions, which are relevant to our work.

3.1 SI* Core Elements

SI* defines a set of concepts, which are necessary to identify the actors[2] involved in the system, their goals, entitlements and the relationships between them. An *agent* is an active entity of the system, which plays a particular *role*. This is represented by means of the *play* relationship. The notion of *service* is used to refer to either a goal, a task or a resource. A *goal* is a desirable situation or interest expressed by an entity, a *task* is a set of actions that can be executed to fulfil a goal, and a *resource* is an artefact produced or used by a goal or task. The connection between services and actors are expressed by means of three relationships: *own* denotes the authority of entities over resources and goals; *provide* represents the ability of an actor to accomplish a goal or to provide a resource; and *request* denotes the interest of an entity over a goal or resource.

There are some additional predicates to denote that a goal can be attained by fulfilling a set of subgoals, and predicates to deal with social relationships such as delegation and trust. The formalisation of the aforementioned elements is done using answer set programming (ASP) syntax [2], as shown on the left side of Table 1. Note that only the most relevant predicates are provided here.

Goal model
actor(Actor: a)
agent(Agent: a)
role(Role: r)
service(Service: s)
goal(Goal: g)
task(Task: t)
resource(Resource: r)
Actor properties
play(Agent: a, Role: r)
own(Actor: a, Service: s)
request(Actor: a, Service: s)
provide(Actor: a, Service: s)
Goal refinement
subgoal(Service: s_1, Service: s_2)
AND_decomp(Service: s, Service: s_1, Service: s_2)
OR_decomp(Service: s, Service: s_1, Service: s_2)
means_end(Service: s_1, Service: s_2)
Social relations
del_perm(Actor: a_1, Actor: a_2, Service: s)
del_exec(Actor: a_1, Actor: a_2, Service: s)
trust_perm(Actor: a_1, Actor: a_2, Service: s)
trust_exec(Actor: a_1, Actor: a_2, Service: s)

Resource model
stored_in(Resource: r, Resource: r_1)
part_of(Resource: r, Resource: r_1)
require(Resource: r, Resource: r_1)
Permission model
permission(Actor: a, Resource: r, PType: pt)
del_perm(Actor: a, Actor: a_1, Resource: r, PType: pt)
trust_perm(Actor: a, Actor: a_1, Resource: r)
Security and Threat model
secure_req(Resource: r, SProperty: sp)
secure_req(Goal: g, SProperty: sp, Resource: r)
threat(Actor: a, Resource: r, SProperty: sp)
threat(Actor: a, Goal: g, SProperty: sp, Resource: r)
Asset model
asset(Service: s, Actor: a)
sensitivity(Service: s, SLevel: sl, Actor: a)
secure_req(Service: s, SProperty: sp, Actor: a)
Trust model
trust_perm(Actor: a, Actor: a_1, Service: s, PType: pt)

Table 1. Relevant SI* Predicates

[2] The notion of *actor* is inherited from i* and is used only when it is not necessary to distinguish between the concepts of agent and role.

3.2 SI* Extensions

Although SI* is a very powerful language, some extensions have been proposed to support the modelling of new scenarios.

Asnar et al. [1] introduced different levels of permissions on resources and relationships between them. The *stored_in* relationship indicates the physical location of an informational resource, *part_of* denotes that a resource is composed of other resources, and *require* denotes that a resource needs another resource to function. Moreover, resources are marked with a *security requirement* label that indicates the security property (confidentiality, integrity and availability) that must hold for it. Actors can be provided with three different types of *permissions*: access, modify or manage permission. Finally, the *threat* predicate holds if an actor violates the security property on a resource. Paci et al. [9] introduce two additional extensions to detect insider threats in organisations. The first one is based on the notion of *asset*, which is a service for which the owner specifies the *sensitivity level* as well as a *security property* that denotes the level of protection demanded by the actor for protecting the service. The second extension is a *trust* model that enables to specify the trust level that an actor places on another actor with respect to a given permission on a particular asset. A summary of the aforementioned predicates is presented on the right side of Table 1.

4 Trust Negotiation Extension

We present here our privacy-aware extension of SI* for trust negotiations.

4.1 Overview

A trust negotiation [15] is a *dual relationship* in which the participants exchange (accredited) information in order to establish trust as a means to achieve a goal.

Based on the above definition, we propose to model trust negotiations based on existing features of the SI* modelling language, as shown in Fig. 1. In this figure we can distinguish two main components that play a fundamental role in the modelling of trust negotiations. First, the trust relationship in which data is demanded by each of the actors and the goal to be accomplished. Second, the informational resources owned by the actors, which need to be under control. For that reason, these are marked with a privacy requirement label and a sensitivity level to indicate the risk of sharing these data.

4.2 Trust Negotiaton Relationship

Trust negotiations pose a natural tension due to the conflicting objectives of privacy and trust. On the one hand, trust is founded on the availability of information about other actors. On the other hand, privacy refers to the ability to keep control of sensitive information. As a result, trust negotiations are ruled by the amount of information that each participant *demands* and the amount of data that is willing to *offer*.

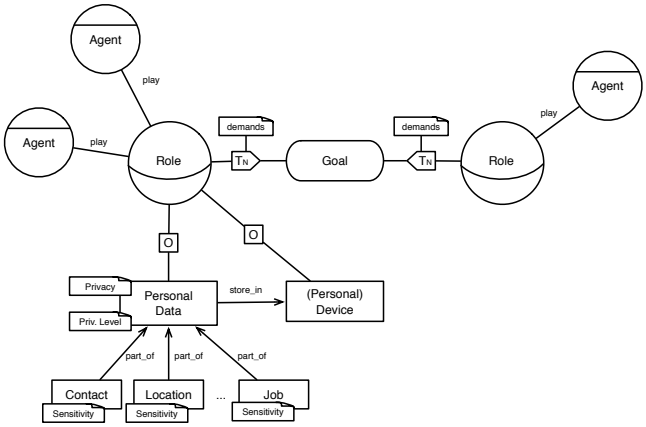

Fig. 1. Trust Negotiation Representation in SI*

This type of relationship can be represented with a notation that is consistent with SI*. Actors can be represented as circles, the goal as a squared oval. These elements are connected by a labelled arc. The arc is further parameterised with the information being requested. Since a trust negotiation is a dual relationship it would be necessary to have one arc in each direction. However, for the sake of clarity and simplicity, we propose an alternative notation with a single arc, as depicted in Fig. 1[3].

4.3 Privacy-Aware Data Disclosure

The modelling of trust negotiation must also take into consideration the ownership of data and whether there are any privacy requirements for these data. Similar to previous extensions that incorporate *security requirement* labels, we propose the use of a *privacy requirement* label to indicate that a particular resource must maintain a specific level of privacy.

Privacy violations are usually associated with a loss of control over data. To this end, we adopt the *part_of* relationship to represent the composition of data resources. Data resources may be additionally labelled with a *sensitivity level* to indicate how valuable this information is. Moreover, the sensitivity is related to the level of detail of data offered. For the sake of simplicity, we consider only 3 sensitivity levels: *Low*, *Medium*, and *High*; and, consequently, we also deal with 3 levels of granularity for each data type. This depends on the data type being considered. Note that the requirements engineer can easily extend this feature to incorporate as many sensitivity and granularity levels as desired.

[3] Note that pentagons point to the party whose information is being demanded.

5 Reasoning Support

We present in this section how verification of the model can be done.

5.1 Predicates

First, a predicate for representing the trust negotiation relationship itself is needed. The predicate *trust_neg* indicates that actors[4] a_1 and a_2 can initiate a negotiation to achieve a common goal g. Note that it is not reasonable to have a trust negotiation where the two actors are the same. This will be reflected later in the rules presented in Section 5.2.

The predicate *offers* denotes that actor a is willing to offer resource r up to a given granularity level $l \in \{Low, Medium, High\}$. The granularity level is inversely proportional to the *sensitivity* of a resource. Similarly, the predicate *demands* indicates that an actor a requests a resource r with at least a given granularity l. This predicate indicates to which actor the resource is demanded since an actor can be involved in several trust negotiations. However, the predicate *offers* does not consider this as it expresses the level of detail that the agent will release regardless of who is involved in the negotiation. Finally, the predicate *privacy_req* denotes the level of privacy that needs to be satisfied for a particular resource. These are predicates P_1 to P_5.

P_1: trust_neg(Actor: a_1, Actor: a_2, Goal: g)
P_2: offers(Actor: a, Resource: r, Level: l)
P_3: demands(Actor: a_1, Actor: a_2, Resource: r, Level: l)
P_4: sensitivity(Resource: r, Level: l)
P_5: privacy_req(Resource: r, Level: l)
P_6: satisfy(Actor: a_1, Actor: a_2)
P_7: data_exposure(Actor: a, Resource: r, Level: l)
P_8: establish_trust(Actor: a_1, Actor: a_2, Goal: g)
P_9: privacy_threat(Actor: a, Resource: r, Level: l)

Besides the aforementioned predicates, other intermediate predicates are needed. The predicate *satisfy* denotes that an actor satisfies the demands of another actor. The predicate *data_exposure* indicates that the resource r belonging to an actor a is exposed to a certain degree l. Two additional predicates indicate whether the trust negotiation process can be fulfilled (*establish_trust*) and whether there is a privacy breach (*privacy_threat*) with respect to the established privacy policy. These are predicates P_6 to P_9.

[4] Actors are used for simplicity but the actual predicates and rules should consider roles and agents as arguments.

5.2 Rules

The first set of rules, from R_1 to R_3, express that the sensitivity of a resource is inversely proportional to its granularity level. Rule R_4 denotes that one actor satisfies the demands of another actor if the resource is offered with at least as much granularity as desired[5]. The actors that demand and offer resources cannot be the same.

R_1: offers(A, R, *High*) $\leftarrow$ owns(A, R) $\wedge$ sensitivity(R, *Low*)
R_2: offers(A, R, *Medium*) $\leftarrow$ owns(A, R) $\wedge$ sensitivity(R, *Medium*)
R_3: offers(A, R, *Low*) $\leftarrow$ owns(A, R) $\wedge$ sensitivity(R, *High*)
R_4: satisfy(A_1, A_2) $\leftarrow$ offers(A_1, R, G_{L_1}) $\wedge$ demands(A_2, A_1, R, G_{L_2}) $\wedge$ ($G_{L_1} \succeq G_{L_2}$) $\wedge$ ($A_1 \neq A_2$)
R_5: establish_trust(A_1, A_2, G) $\leftarrow$ trust_neg(A_1, A_2, G) $\wedge$ satisfy(A_1, A_2)
R_6: data_exposure(A_1, R, E_L) $\leftarrow$ offers(A_1, R, E_L) $\wedge$ satisfy(A_1, A_2)
R_7: data_exposure(A_1, R, E_L) $\leftarrow$ offers(A_1, R_1, E_L) $\wedge$ satisfy(A_1, A_2) $\wedge$ part_of(R_1, R)
R_8: sensitivity(R, *Low*) $\leftarrow$ not sensitivity(R, _) $\wedge$ resource(R)
R_9: sensitivity(R_1, S_L) $\leftarrow$ not sensitivity(R_1, _) $\wedge$ resource(R_1) $\wedge$ sensitivity(R, S_L) $\wedge$ part_of(R_1, R)
R_{10}: privacy_threat(A, R, E_L) $\leftarrow$ privacy_req(R, P_L) $\wedge$ data_exposure(A, R, E_L) $\wedge$ ($E_L \succeq P_L$)

R_5 states that it is possible to establish a trust relationship whenever the trust negotiation has been satisfied. Rules R_6 and R_7 express the amount of information being exposed due to the fulfilment of a trust negotiation.

Rules R_8 and R_9 consider the case of having resources without a predefined sensitivity level. The former assigns a *Low* sensitivity level while the latter impose the same sensitivity level as the one defined for the parent resource. Note that the '_' symbol represents that this argument is irrelevant for the rule to be triggered. Finally, rule R_{10} states that the privacy policy is violated when the level of exposure of a resource exceeds its desired privacy level.

6 Conclusion

This paper presents a framework to include trust negotiation models in the early phases of the SDLC. The framework is based on the SI* modelling language and enables the automatic detection of privacy threats due to disclosure of data beyond a sensitivity level. The detection of privacy threats can aid in the refinement of privacy policies in the system.

We are currently working on extending the features of our framework to capture more complex scenarios. A future research line will be to consider multiple data exchanges when an actor is engaged in multiple trust negotiations.

[5] We use the $\succeq$ symbol to compare ordinal values: $High \succ Medium \succ Low$.

Acknowledgements

This work has been partially funded by the European Commission through the Marie Curie Training Network NeCS (H2020-MSCA-ITN-2015-675320), the Spanish Ministry of Economy and Competitiveness through PERSIST (TIN2013-41739-R) and PRECISE (TIN2014-54427-JIN), which is co-financed by FEDER.

References

1. Y. Asnar, T. Li, F. Massacci, and F. Paci. Computer Aided Threat Identification. In *13th IEEE Conference on Commerce and Enterprise Computing*, pages 145–152, 2011.
2. G. Brewka, T. Eiter, and M. Truszczyński. Answer Set Programming at a Glance. *Commun. ACM*, 54(12):92–103, Dec. 2011.
3. J. Castro, P. Giorgini, M. Kolp, and J. Mylopoulos. Tropos: A Requirements-Driven Methodology for Agent-Oriented Software. In B. Henderson-Sellers and P. Giorgini, editors, *Agent-Oriented Methodologies*. Idea Group, 2005.
4. M. Deng, K. Wuyts, R. Scandariato, B. Preneel, and W. Joosen. A privacy Threat Analysis Framework: Supporting the Elicitation and Fulfillment of Privacy Requirements. *Requir. Eng.*, 16(1):3–32, Mar. 2011.
5. C. Kalloniatis, E. Kavakli, and S. Gritzalis. Addressing privacy requirements in system design: the PriS method. *Requir. Eng.*, 13:241–255, 2008.
6. F. Massacci, J. Mylopoulos, and N. Zannone. Security Requirements Engineering: The SI* Modeling Language and the Secure Tropos Methodology. In *Advances in Intelligent Information Systems*, pages 147–174. Springer Berlin Heidelberg, 2010.
7. H. Mouratidis and P. GiorginiI. Secure Tropos: A Security-Oriented Extension of the Tropos Methodology. *Int J Softw Eng Know*, 17(02):285–309, 2007.
8. N. Notario, A. Crespo, Y. Martín, J. M. del Álamo, D. L. Métayer, T. Antignac, A. Kung, I. Kroener, and D. Wright. PRIPARE: Integrating Privacy Best Practices into a Privacy Engineering Methodology. In *International Workshop on Privacy Engineering*, pages 151–158, 2015.
9. F. Paci, C. Fernandez-Gago, and F. Moyano. Detecting Insider Threats: A Trust-Aware Framework. In *8th International Conference on Availability, Reliability and Security (ARES)*, pages 121–130, Sept 2013.
10. E. Paja, F. Dalpiaz, and P. Giorgini. Modelling and Reasoning about Security Requirements in Socio-Technical Systems. *Data and Knowledge Engineering*, 98:123–143, 2015.
11. A. Squicciarini, E. Bertino, E. Ferrari, F. Paci, and B. Thuraisingham. PP-Trust-X: A System for Privacy Preserving Trust Negotiations. *ACM Trans. Inf. Syst. Secur.*, 10(3), July 2007.
12. A. van Lamsweerde. Elaborating Security Requirements by Construction of Intentional Anti-Models. In *26th International Conference on Software Engineering*, ICSE '04, pages 148–157, Washington, DC, USA, 2004. IEEE Computer Society.
13. A. van Lamsweerde and E. Letier. Handling Obstacles in Goal-Oriented Requirements Engineering. *IEEE T Software Eng*, 26(10):978–1005, Oct. 2000.
14. A. van Lamsweerde; R. Darimont ; E. Letier. Managing Conflicts in Goal-Driven Requirements Engineering. *IEEE T Software Eng*, 24(11):908–926, 1998.
15. M. Winslett, T. Yu, K. E. Seamons, A. Hess, J. Jacobson, R. Jarvis, B. Smith, and L. Yu. Negotiating Trust on the Web. *IEEE Internet Comput*, 6(6):30–37, 2002.

16. E. Yu. *Modelling Strategic Relationships for Process Reengineering.* PhD thesis, University of Toronto, Canada, 1996.