An Extended Ontology for Security Requirements

Fabio Massacci[1], John Mylopoulos[1], Federica Paci[1],
Thein Thun Tun[2], and Yijun Yu[2]

[1] Department of Information Engineering and Computer Science,
University of Trento, Italy
[2] Department of Computing, The Open University, UK
{fabio.massacci,mylopoulos,paci}@disi.unitn.it, {t.t.tun,y.yu}@open.ac.uk

Abstract. Security concerns for physical, software and virtual worlds have captured the attention of researchers and the general public, thanks to a series of dramatic events during the past decade. Unsurprisingly, this has resulted in increased research activity on topics that relate to security requirements. At the very core of this activity lies the problem of determining a suitable set of concepts (aka ontology) for modeling security requirements. Many proposals for such ontologies exist in the literature. The main objective of this paper is to amalgamate and extend the security ontologies proposed in [1] and [2]. The amalgamation includes a careful comparison of primitive concepts in Problem Frames and Secure Tropos, but also offers a novel account for rather nebulous security concepts, such as those of vulnerability and threat. The new concepts are justified and related to the literature. Moreover, the paper offers a number of security requirements adopted from industrial case studies, along with their respective representation in terms of the proposed ontology.

1 Introduction

Security concerns for physical, software and virtual worlds have captured the attention of researchers and general public, thanks to a series of dramatic events during the past decade. Unsurprisingly, this has resulted in increased research activity on topics that relate to security requirements. At the very core of this activity lies the problem of determining a suitable set of concepts (aka ontology) for security requirements. In other words, the problem consists of selecting a suitable set of primitives through which security requirements can be conceptualised [3] for purposes of modeling, analysis and communication. The problem is clearly articulated in [4], where more than a dozen recent proposals for such security ontologies are reviewed and compared.

Massaccci et al. [1] presents one such proposal for an ontology, based largely on the PhD thesis of Nicola Zannone [5], and founded on the modeling framework of i* and Tropos. In parallel, Haley et al. [2] proposed Abuse Frames to take advantages of the analytical capability of Problem Frames [6]. Both proposals have their advantages: with goal-oriented security requirements analysis, malicious intentions of attackers can be identified through explicit characterization of

C. Salinesi and O. Pastor (Eds.): CAiSE 2011 Workshops, LNBIP 83, pp. 622–636, 2011.

social dependencies among actors; with problem-oriented security requirements analysis, valuable assets that lie within or beyond the system boundary can be identified through explicit traceability of shared phenomena among physical domains and the machine itself. There are also other proposals to address security requirements, such as misuse cases, obstacle analysis of anti-goals, information flow analysis of attack scenarios, etc. Each one of these has unique features that others do not. Having seen the advantages of alternative proposals, we would like to have a unified ontology to reach a shared understanding of the domain of security requirements, and also take advantage of multiple analysis techniques. The main objective of this paper is to amalgamate the security ontologies of SI* [1] and Abuse Frames [2], and also to account for rather nebulous security concepts, such as those of vulnerability and threat. A secondary objective is to develop techniques for using the amalgamated framework in order to model and analyze security requirements.

The research reported in this paper is conducted within the context of the SecureChange EU project [7]. The project has generated a number of realistic case studies including one from the Air-Traffic Management (ATM) domain. A first evaluation of our proposed security ontology is offered in the paper by applying it to a fragment of the ATM case study.

The rest of the paper is structured as follows. Section 2 briefly illustrates the baseline concepts from Secure i*/Tropos (SI*) and Problem Frames, and highlights the analysis facilities of each methodology to analyse security requirements. Section 3 presents the unified ontology, while Section 4 illustrates the application of the ontology to the ATM case study. Section 5 presents related work and Section 6 concludes the paper and highlights future directions.

2 Baseline

Tropos [8] is a development methodology for agent-oriented software, founded on i*. i* [9] is a modeling framework for early requirements. Through it, one can model the stakeholders for a given project, their goals and their social interdependencies. Both frameworks are grounded on the concepts of actor. An actor is an entity that wants (goals) and acts (by carrying out tasks) in order to fulfill its goals. Actors can be agents, roles or positions. Agents are materialized (as organizational or human agents, also systems). A role has goals and can carry out tasks, but only once it has an agent who plays the role. Positions are aggregates of roles. In order to fulfill their goals, actors depend on other actors to fulfill goals, carry out tasks or deliver resources. Such dependencies (aka dodependencies) constitute a basic form of relationship in modeling and analyzing social settings.

SI* [1] adopts and extends this framework. Firstly, actors not only want goals, but also own resources, tasks and goals. The meaning of owning a task or resource is an extension of the concept of owning a resource. There are tasks such as "Take a seat on flight AC847" that cannot be carried out by an actor (e.g., Paolo), unless another actor (e.g., Airline AC) who "owns" the task gives permission. Likewise, the University of Trento (UniTN) owns the goal "earn degree

from UniTN", and another actor is not allowed to fulfill it unless she has permission from the owner. Given that services (i.e., goals, tasks and resources) can be owned, SI* also proposes permission dependencies (also known as can-dependencies) to indicate that one actor has given permission to another to use a given service. SI* also includes trust dependencies between actors and a service to represent a situation where actor A trusts actor B for a given service C, i.e., A trusts B to fulfill/carry out/deliver C.

There are at least two types of security analysis supported by SI*: [1] proposes an analysis where an SI* model is checked to confirm that all actors who have been delegated services (i.e., they are responsible for delivering some service) actually have permission to do so. In earlier work, Liu et al [10] proposed a complementary type of analysis which involves checking the consequences if an actor within a network of actors and delegations actually behaves like an attacker.

Problem Frames [6] (PF) is an intellectual tool to explore the typical contextual relationship between the machine domains to be specified and the related physical domains in the world. The behaviour of each domain is described by a number of phenomena. At the interfaces between these domains, phenomena are shared: some domains control the phenomena whilst other domains observe them. Having the knowledge of these physical domain expressed, requirements are modelled as constraints on the referred phenomena. In the security area, two problem-oriented approaches are interesting. Abuse Frames [11] analyses anti-requirements and their satisfaction arguments. The analysis of the anti-requirements and assoicated problem structure reveals typical situations where security requirements are violated. The intentions of stakeholders behind these anti-requirements are further treated as anti-goals [12]. Another one is the security argumentation framework [2, 13], which challenges the trust assumptions on all phenomena included or excluded in the problem structure, in the context of whether they may cause harm to the assets identified with value to the stakeholders.

Not all phenomena are referred by the requirements, and not all are related to the requirements problem. Therefore two kinds of analysis in Problem Frames methodology are fundamental to the requirements analysts. One analysis is to establish the satisfaction argument semantics between the indicative phenomena in the world domains W and the optative properties of the machine domain to specify S and the constraints in requirements, denoted by $W, S \vdash R$ [14].

The other analysis explores the relationships of phenomena at the boundaries between the system and the context. A phenomena in the knowledge may be excluded from the argument, therefore hidden from the problem analysis. Hidden Domains or Hidden Phenomena are methodological concepts in PF. It helps one focus on the relevant information to analyse a problem, and also helps one consider the relationship between decomposition and composition of problems, reflecting the widsom "Divide and conqueor, unite and rule". The term Frames refers to typical structural relationships between certain types of behaviour domains (e.g., Causal, Biddable or Lexcial), in order to reason about the

associated typical concerns such as interaction and initialisation using Event Calculus [15].

3 The Extended Ontology

A bird's eye view of our proposal has as follows. The very top of the taxonomy is adopted from DOLCE [16], a foundational ontology intended to account for basic concepts that underlie natural language and human cognition. Lower levels of the taxonomy include concepts from i*, problem frames and argumentation frameworks, with security concepts occupying the lowest strata of the taxonomy. Key among the concepts we introduce is the concept of Proposition, with instances such as "Want for customers for our business" and "Paolo is married". The other key concept is that of Situation, representing a partial state of the world, e.g., "High oil prices", or "Many unhappy customers".

The most general concept is *Thing*, which has as instances all the things that can exist in the world.

An object is a thing that persists (endurant in [16]). An event, instead, is an instantaneous happening (perdurant) that changes some objects. Specializations of the concept Object include *Proposition*, *Situation*, *Entity* and *Relationship*.

A proposition is an object representing a true/false statement. A situation is a partial world described by a proposition (its description)[17]. Arbitrary propositions are true/false/ undefined in a situation, given its partial world status. The status of the world is expressed by a predicate over the entities involved[1]. Situations can have structure consisting of relationships and things standing in those relationships. Some entities and relationships according to the common sense always satisfy certain predicates, making them strong beliefs or trust assumptions. Therefore we consider predicates in the logical world on every entity and relationship to question even their absense/existence. This makes a security argumentation more powerful.

An entity is an object that has a distinct, separate existence from all other things, though that existence need not be material. Thus, Santa Claus, my cancelled trip, a square circle are entities. A relationship, on the other hand, is an object that participates in a certain situation along with other objects (its relata); the existence of a relationship depends on that of its relata.

Entities. *Entity* is specialized into *Actor*, *Action*, *Process*, *Resource*, and *Asset*, all concepts adopted from Goal-Oriented Requirements Engineering (GORE) approaches. An actor is an entity that can act and intend to want or desire. *Stakeholder* and *Attacker* are two important specializations of *Actor* for the domain of security requirements. A stakeholder is an actor who has a stake in the system- to-be, while an attacker wants to prevent the fulfillment of the requirements for the system-to-be. An action is an entity performed by an actor, which can generate events, and can have preconditions and post-conditions. A Process

[1] Note that predicates are a special form of propositions, and through reification they can be grounded into sentences of propositions.

is an entity that generates events and changes objects. *Activity* is a specialization of *Process*, consisting of actions. *Attack* specializes *Activity*, and is always carried out by an Attacker. We distinguish between Process and Activity in our ontology because we want to allow for processes that do not involve any actions, e.g., a fire burning, or an earthquake. A resource is an entity without intention or behaviour. An asset is an entity of value that can be owned and used. For example, an asset can be an passenger (actor) whose life needs to be protected, can be an engine (process) whose behaviour has a value to the protector, or can be an aircraft (resource) whose value are tangible for other actors. A relationship such as the organization chart of the air traffic management organisation is also an asset as long as its value need to be protected.

Relationships. Specializations of *Relationship* include *do-dependency*, *can-dependency* and *trust-dependency* adopted from SI*. These are all ternary relationships between two actors and an asset. In addition, there are many binary relationships that characterize other concepts in our ontology. For example, actors are entities who *want* goals and *carries out* actions. *composes*, *contributes*, *uses*, and *provides* are some of the other relationships included in the ontology. AND/OR refinement is a relationship between a goal and two or more other goals that indicates that a goal can be refined into subgoals. contributes relates two goals and indicates that one goal has a positive or negative impact on the satisfaction of the other. provides is a relationship from an actor to resource, which specifies that an actor provides a certain resource. uses is a Relationship from a process to a resource denoting that the process generates or consumes the resource. fulfills relates an entity to a goal that the entity fulfills.

For the sake of security requirement analysis, the ontology includes also the following specializations of Relationship: *damages*, *exploits*, *protects*, and *denies*. damages is a relationship between an attack and an asset, where the attack causes harm to the asset. exploits is a relationship between attack and vulnerability. protects relates a security goal to an asset. Finally, denies relates an anti-goal to a requirement. A complete list of all the possible relationships is found in Figure 1.

Propositions. Proposition is specialized into *Fact*, *Claim*, *Argument*, *Domain Assumption*, *Quality Proposition*, and *Goal*, depending on the different types of proposition modalities. A fact is a true proposition. A claim is a proposition claimed to be true by an actor. An argument is a proposition consisting of a set of claims. A domain assumption is a proposition about the domain assumed to be true by an actor. A quality proposition is a proposition about the quality of the system-to-be. A goal is a concept found in GORE approaches, and represents a proposition an actor wants to make true. For security analysis purposes, Goal is specialized into *Requirement*, *Security Goal*, and *Anti-Goal*. A requirement is a goal wanted by a stakeholder. A security goal prevents harm to an asset through the violation of confidentiality, integrity, and availability [2]. An anti-goal is a goal an attacker wants which denies the fulfillment of a requirement of the system-to-be.

```
Thing :: = Event | Object | ...
Object  :: = Situation | Proposition | Entity | Relationship
Situation :: = Domain | Context | Vulnerability | Threat | Specification
Context :: = Domain
Proposition :: = Argument | Predicate| Claim |Domain Assumption|
Quality  Proposition |Goal
Argument :: = {Claim}
Goal :: = Requirement | Security Goal | Anti Goal
Entity :: =  Action | Process | Actor | Resource | Asset
Activity :: = {Action}
Attack :: = Activity
Actor :: = Stakeholder | Attacker
Specification :: = {Domain Assumption} {Quality Proposition} {Action}
Relationship :: =  fulfills | exploits | protects | denies | damages |
wants |carries out | uses | provides | trust-dependency | do-dependency |
 can-dependency | composes | contributes| ...
```

Fig. 1. Ontology Representation in EBNF

Situations. The *Context* and *Domain* concepts coming from Problem Frames are specializations of Situation. These concepts are useful to define the situation of system boundaries, to allow one place focus on analysis while hide the unnecessary details. For the analysis of every problem or subproblem, a different situation may be selected from the physical world. Thus the context is a situation in which the system-to-be will operate; and a domain is a situation that is part of the context. In Problem Frames, domains can be classified as biddable, causal, and lexical. By biddable, a domain's behaviour is not fully predicable or controllable, usually represented by human actors or natural processes. By causal, a domain's behaviour is predicable or controllable, usually represented by activities. By *lexical*, a domain's behaviour is predefined, usually by a resource.

Another concept adopted from Problem Frames is *Specification*. A specification is an entity consisting of actions, quality propositions, and domain assumptions. Thus, is a collection of indicative propositions about the entities in the system-to-be.

Vulnerability is a specialization of Situation and is adopted from the Security domain. A vulnerability is a situation where some actions that are part of an attack can be carried out (i.e., their preconditions are satisfied). A threat, on the other hand, consists of a situation that includes an attacker and one or more vulnerabilities.

Figure 1 summarizes the elements of our ontology in Extended Backus-Naur Format (EBNF)[2]. A EBNF rule of the form A ::= B | C | ... indicates that concept A has concepts B and C (and possibly others) as specializations. A rule

[2] When ... abbreviation is used in the production rule, its semantics is to rewrite all the previous rules of the same LHS by extension [18].

of the form A ::= {C} indicates that each instance of A consists of (has parts) zero or more instances of C. [] is similar to { } but allows for zero or one instance.

4 ADS-B - A Case Study

Air traffic control management systems are changing. They are moving from systems that rely on radar technology to systems that use precise location data from the global satellite network. Automatic Dependent Surveillance-Broadcast (ADS-B) technology is enabling such evolution. ADS-B-equipped aircrafts determine their own position using a global navigation satellite system, and periodically broadcast their position, identity and other relevant information such as speed, and height to the ground stations and other aircrafts with ADS-B equipment. With ADS-B, both pilots and controllers will see on their displays (the controller working position) highly accurate traffic data from satellites. Thus, ADS-B provides accurate information and frequent updates to airspace users and controllers, and hence supports improved use of airspace, reduced ceiling/visibility restrictions, improved surface surveillance, and enhanced safety.

If on one side, ADS-B has several benefits for air traffic management, on the other side, it arises several security concerns, because ADS-B transmissions can be easily corrupted. A concrete example of corruption of ADS-B transmissions is the spoofing of the GPS Satellite that provides the GPS signal to aircrafts equipped with ADS-B to allow them to determine their position. Such an attack can be easily accomplished using a GPS satellite simulator. To conduct the spoofing attack, an adversary broadcasts a fake GPS signal with a higher signal strength than the true signal. The GPS receiver believes that the fake signal is actually the true GPS signal from space and ignores true signal. The receiver then proceeds to calculate erroneous position or time information based on this false signal.

We now illustrate how Problem Frames and Goal-Oriented approaches, and the security ontology we have proposed here allow one to model a security requirement problem domain at different levels of abstraction.

4.1 Modeling the ADS-B Example in Problem Frames

Using the Problem Frames approach (including Abuse Frames), we first describe the problem context, which include the following problem world domains (Figure 2):

GPS satellite broadcasts position signals

Aircraft (i) receives GPS signals, (ii) calculates their positions, (iii) broadcasts the position and identity information over ADS-B, (iv) follow instructions from the ATC operator, and they need to be routed safely and securely

ADS-B Ground Receivers receive ADS-B broadcasts and reports to the central processor

Central Processor (i) validates the position and identity information, (ii) produces tracking reports for the operator, (ii) sends the instructions from the the ATC operator to the aircrafts.

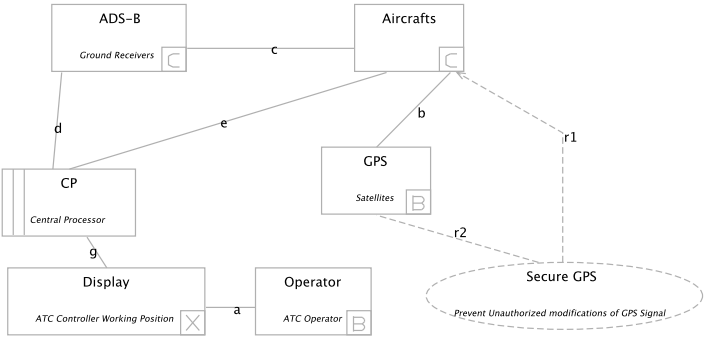

Fig. 2. ADS-B: Problem World Domains

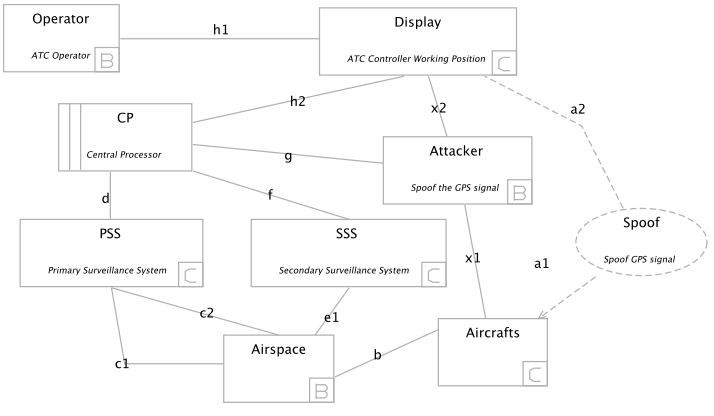

Fig. 3. ADS-B: Problem World Domains from an Attacker's Perspective

Display shows the tracking reports of the controller working position and receives the operator instructions

ATC Operator checks tracking reports and send routing instructions

Figure 2 shows the basic structure of the problem, which includes three types of nodes (Requirements as ellipses, Physical Domains as rectangles, and Machine Domains as rectangles marked by strips). The relationship between them are viewed as three types of links (Interface Phenomena as solid lines, Reference Phenomena as dashed lines and Constraint Phenomena as dashed arrows).

We consider as main security requirement to prevent unauthorized modifications of the aircraft position. In the above context, we can make a tentative argument that the behavior of the problem world domain ensures that the requirement is satisfied.

For the security analysis, we need to introduce into the problem context an attacker with its own requirement: that is, an attacker who wishes to harm an

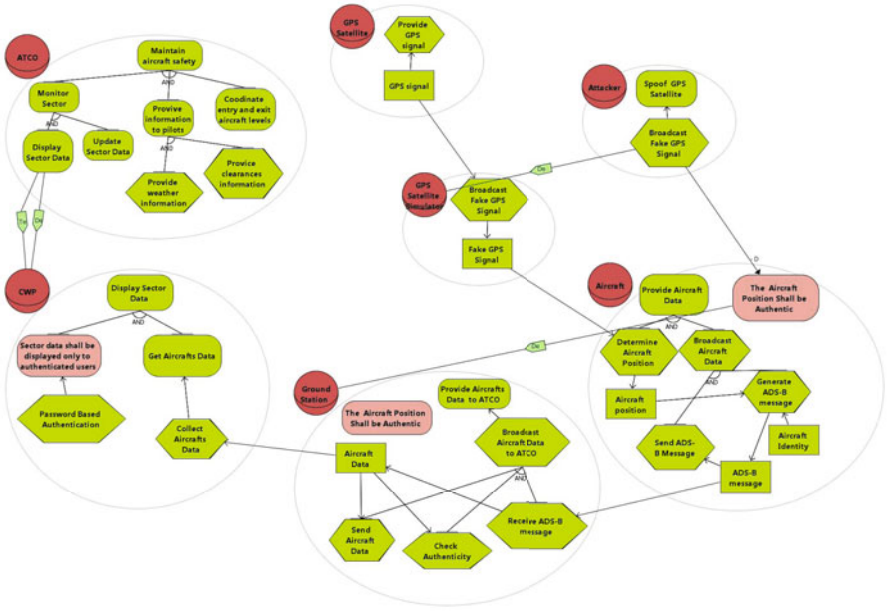

Fig. 4. The ADS-B Example modeled in Secure Tropos

asset protected by the system. Such a requirement of an attacker could be to spoof the GPS signal broadcasted by GPS Satellite to compromise the aircrafts position (as shown in Figure 3). In that case, the asset is the aircraft position.

We can construct a tentative argument showing that the ATM system will satisfy the "Prevent Unauthorized Modifications of Aircraft Positions" requirement, thus preventing the attacker from achieving his/her requirement. However, our analysis has to recognise the attacker is likely to modify parts of the problem world domain by exploiting vulnerabilities, using which the attacker will have his/her requirement satisfied by the ATM system. These vulnerabilities are rebuttals to the tentative arguments.

In the problem context shown in Figure 3, several vulnerabilities could be exploited by an attacker. An attacker could produce fake GPS signals in order to force aircrafts to produce incorrect position information, thus potentially allowing aircrafts to follow flight paths desired by the attacker.

Although Problem Frames concepts help explore the functional requirements that concerns the interfaces at the system boundary, they do not express the intentions behind the actors for both stakeholders and attackers.

4.2 Modeling the ADS-B Example in Secure Tropos

Using Secure Tropos concepts, one would model the physical domains by detailing the rationale ("why") behind them. Note that not all the rationale of physical domains can be modelled as intentions (goals desired by actors), thus

the notion of tasks and resources are used to represent the physical domains interacting with the actors. The goals represent the requirements of the actor, the tasks represent the processes by which the goals are fulfilled, and the resources represent the shared phenomena that are observed or controlled by an actor or a process.

In the following (Figure 4), we present all the domains as actors: the ovals model goals, the rectangles represent resources, and the exagons model the tasks.

ATCO. The main goal of the ATCO is to "Maintain aircraft safety" that is AND-decomposed into three subgoals:"Monitor Sector", "Provide Information to Pilots", and "Coordinate entry and exit aircraft levels". The "Monitor Sector" goal can be further AND-decomposed into the subgoals "Display Sector Data " and "Update Sector Data ". The fulfillment of the goal "Display Sector Data " is delegated by the ATCO to the CWP.

CWP. The "Display Sector Data" goal is AND-decomposed into the goal "Sector Data shall be displayed only to authenticated users" and in the goal of "Display Aircraft Data". Note that the former subgoal is in fact a constraint on the latter in order to protect the confidentiality of the valuable assets, users. Furthermore, the latter is fulfilled by the task "Password Based Authentication"; the former is fulfilled by the task "Display Aircraft Data" that takes in input the resource "Aircraft Data". The "Aircraft Data" resource is provided to the CWP by the Ground Station.

Ground Station. The main goal of Ground Station actor is "Provide Aircraft Data to ATCO". This goal is fulfilled by the task "Broadcast Aircraft Data " which is decomposed into the tasks "Receive ADS-B message", "Check Authenticity", and "Send Aircraft Data". "Check Authenticity" realises the security goal of protecting the confidentiality of data, therefore it is regarded as a security requirement that constraints the other tasks that fulfils the functional requirements.

GPS Satellite. The main goal of GPS Satellite is "Provide GPS Signal" that is fulfilled by the task "Broadcast GPS Signal" that produces the GPS Signal resource.

Attacker. The main goal of the Attacker actor is "Spoof GPS Satellite" whose satisfaction is delegated to the GPS Satellite Simulator actor.

GPS Satellite Simulator. The GPS Satellite Simulator fulfills the goal "Spoof GPS Satellite" of the Attacker by providing the task "Broadcast Fake GPS Signal" that produces the "Fake GPS Signal" resource.

Aircraft. The Aircraft actor has two goals: "Provide Aircraft Data", and the security goal "Aircraft Position shall be Authentic". "Provide Aircraft Data" is fulfilled by the tasks "Determine Aircraft Position", and "Broadcast Aircraft Data". In absence of Attacker, "Determine Aircraft Position" should process the GPS signal resource provided by GPS Satellite and produce the resource "Aircraft Position". Due to the presence of Attacker, "Determine Aircraft Position"

identify the aircraft position on the basis of the "Fake GPS Signal" transmitted by the task "Broadcast Fake GPS Signal" of the GPS Satellite Simulator under control of the Attacker. "Broadcast Fake GPS Signal" fulfills Attacker's goal "Spoof GPS Signal" but has a negative impact on the fulfillment of Aircraft's Security Goal "Aircraft Position shall be Authentic". The satisfaction of "Aircraft Position shall be Authentic" is delegated to the Ground Station actor which fulfills it with the task "Check Authenticity". "Broadcast Aircraft Data" task is decomposed in two tasks, "Generate ADS-B Message", and "Send ADS-B Message".

4.3 Modeling the ADS-B Example Using the Extended Ontology

In this section we show how using the concepts of the security ontology, we are able to model the ADS-B example at a more detailed abstraction level than both PF and SI* approaches and reason about the security argumentations that would not be possible using just PF or just SI*. Figure 5 sketches the spoofing of the GPS Satellite threat scenario, and the impact that it has on the assets and on the satisfaction of the security requirements of the ABS-B-based surveillance system. Due to the lack of space, we model only the domains involved in the threat that is, the Aircraft, Attacker, GPS Satellite Simulator, and the Ground Station. The requirements of each actor are modeled using the notion of goals, and the relationship wants between the actor and the goal the actor desires to achieve.

The threat scenario is represented by the threat "Spoofing GPS Signal" that consists of two attackers "Adversary" and "GPS Satellite Simulator", the vulnerability "Aircraft receiver is not able to distinguish a Fake GPS Signal from a Fake one", and the attack "Broadcast Fake GPS Signal". Thus, the "Adversary" attacker wants the anti goal "Spoof GPS Satellite" to cause harm to the asset "Aircraft Position" of the Aircraft actor. To fullfill the anti-goal, the "Adversary" attacker carries the attack "Broadcast Fake GPS Signal" which exploits the vulnerability "Aircraft receiver is not able to distinguish a Fake GPS Signal from a Fake one". If the attack is successful, the anti goal is satisfied denying the satisfaction of the the security goal "Aircraft Position shall be Authentic" wanted by the Aircraft actor, which aims to protect the "Aircraft Position" asset from.

5 Related Work

The need of the security community to have a common ontology to promote knowledge sharing and understanding of the security domain has been widely recognized [19] as a branch of research. Several ontologies for security [20, 21, 22, 23, 24, 25, 1, 26] have been proposed but they fall short in completeness and generality. We can classify the existing proposals about security ontologies in two main categories: ontologies that includes only security specific concepts such as those coming from threat analysis, and ontologies that includes security related

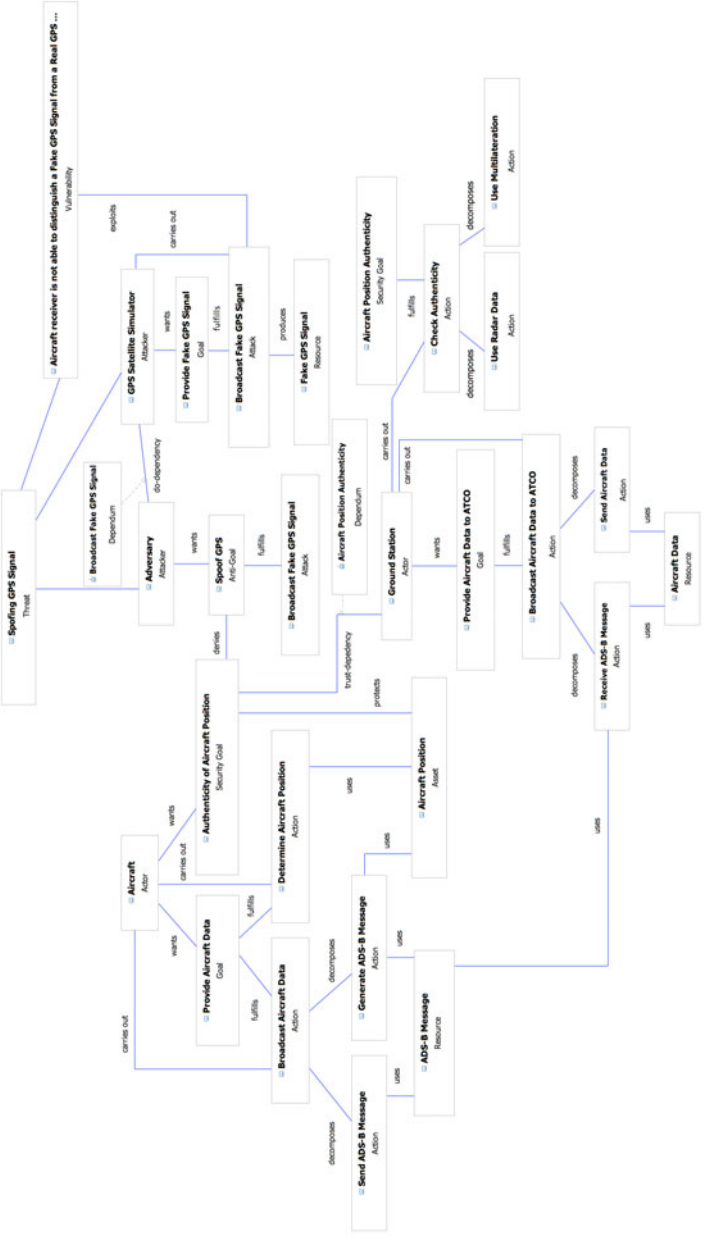

Fig. 5. The ADS-B Example modeled using the security ontology concepts

concepts, concepts for modeling requirements, and the dependencies between them. The ontologies proposed by Denker et al. [20], Kim et al.[25], Fenz et al. [22], Karyda et al. [24] , and Undercoffer et al. [26] belong to the first category; instead the proposals by Firesmith et al. [23], Dobson et al. [21], and Massacci et al. [1] fell in the the second category.

Denker et al. [20] propose an ontology to augment web service and agents descriptions with security annotations: the ontology consists of two subontologies that captures respectively security mechanism and credentials. Similarly to Denker et al. ontology, the security ontology proposed by Kim et al.[25] consists of seven ontologies that include concepts related to security mechanisms, protocols, algorithms, and credentials. In our security ontology we do not include explicitly the concepts of security mechanism and credentials but they can be included in our ontology as a specialization of process and the resource concepts. Fenz et al. [22] and Karyda et al. [24] propose two ontologies including concept used in threat analysis. Fenz et al. ontology consists of five ontologies to model threats, countermeasures, affected infrastructures, the impact of threat, the enterprise being threaten, and the persons involved. Similarly, Karyda et al. ontology includes the concepts of assets, countermeasures, persons, and threats. Undercoffer et al. [26] have defined an ontology to specify computer attacks. Our ontology is not less expressive than the above security ontologies because in our ontology we are able to model the threat that can harm the system-to-be, that is a situation that involves one or more attackers that carries out an attack which exploits a vulnerability. We are, also, able to model which is the asset that is damaged by the attack and the possible countermeasures can be modeled using the action concepts.

Firesmith et al. [23] defined an ontology for safety related requirements which models how the derivation of safety requirements depends on threat analysis. Instead, Dobson et al. [21] have defined an ontology for dependability requirements that includes security issues such as dependability, reliability, availability, integrity, confidentiality, and safety. In our ontology we do not focus on a specific type of requirements but we are able to model both functional and non functional requirements such as security requirements and how these requirements will be fulfilled by the system-to-be. Moreover,we are able to capture the dependencies between the security requirements of the system-to-be and the assets that need to be protected by security requirements, and the threats that can deny the satisfaction of those requirements.

Massacci et al. [1] has proposed a security ontology which extends the i* ontology [9] to model security at an organizational level that is based on the concepts of ownership, trust and delegation dependencies between actors.

Our ontology unifies the concepts from Massacci et al. ontology and the concepts from Problem Frames and Abuse Frames approaches to model requirements, and other security relevant concepts such as threat, attack, attack, vulnerability. Thus, our ontology is the first attempt to define a complete ontology that helps understanding the security problem domain.

6 Conclusions

This paper presents an ontology for security requirements that unifies existing concepts from the Problem Frames and Secure i* methodologies, and security concepts such as asset and threat. The propopsed extended ontology brings together these concepts to facilitate a security argumentation that was not feasible in each method due to the missing constructs. We have illustrated the expressiveness of the proposed ontology with respect to PF and SI* by modelling the security requirements of a case study from the Air Traffic Management domain. The ontology we show here has the potential to be further extended to accommodate more concepts in the area of security requirements by incorporating other methodologies such as misuse cases.

Acknowledgement

This work has been partially funded by EU-SecureChange project, and the EU-NESSoS NoE.

References

1. Massacci, F., Mylopoulos, J., Zannone, N.: Computer-aided support for secure tropos. Automated Software Engg. 14(3), 341–364 (2007)
2. Haley, C.B., Laney, R.C., Moffett, J.D., Nuseibeh, B.: Security requirements engineering: A framework for representation and analysis. IEEE Trans. Software Eng. 34(1), 133–153 (2008)
3. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing. Int. J. Hum.-Comput. Stud. 43(5-6), 907–928 (1995)
4. Blanco, C., Lasheras, J., Valencia-García, R., Fernández-Medina, E., Toval, A., Piattini, M.: A systematic review and comparison of security ontologies. In: ARES 2008: Proceedings of the 2008 Third International Conference on Availability, Reliability and Security, pp. 813–820. IEEE Computer Society Press, Washington, DC, USA (2008)
5. Zannone, N.: A requirements engineering methodology for trust, security, and privacy (2006)
6. Jackson, M.: Problem frames: analyzing and structuring software development problems. Addison-Wesley Longman Publishing Co., Inc., Boston (2001)
7. Secure change project
8. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems 8, 203–236 (2004)
9. Yu, E.S.K.: Modelling strategic relationships for process reengineering. PhD thesis, Toronto, Ont., Canada, Canada, Adviser-Mylopoulos, John (1995)
10. Liu, L., Yu, E.S.K., Mylopoulos, J.: Security and privacy requirements analysis within a social setting. In: [27], pp. 151–161 (2003)
11. Lin, L., Nuseibeh, B., Ince, D.C., Jackson, M., Moffett, J.D.: Introducing abuse frames for analysing security requirements. In: [27], pp. 371–372 (2003)

12. van Lamsweerde, A.: Elaborating security requirements by construction of intentional anti-models. In: ICSE, pp. 148–157. IEEE Computer Society, Los Alamitos (2004)
13. Nuseibeh, B., Haley, C.B., Foster, C.: Securing the skies: In requirements we trust. IEEE Computer 42(9), 64–72 (2009)
14. Hall, J.G., Rapanotti, L., Jackson, M.: Problem frame semantics for software development. Software and System Modeling 4(2), 189–198 (2005)
15. Laney, R.C., Tun, T.T., Jackson, M., Nuseibeh, B.: Composing features by managing inconsistent requirements. In: du Bousquet, L., Richier, J.L. (eds.) ICFI, pp. 129–144. IOS Press, Amsterdam (2007)
16. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening ontologies with DOLCE. Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web, 223–233 (2002)
17. Gangemi, A., Presutti, V.: Ontology Design Patterns. In: Handbook of Ontologies, 2nd edn., Springer, Berlin (pres)
18. Cordy, J.R.: Txl - a language for programming language tools and applications. Electron. Notes Theor. Comput. Sci. 110, 3–31 (2004)
19. Blanco, C., Lasheras, J., Garcia, R.V., Fernandez-Medina, E.: A systematic review and comparison of security ontologies (2008)
20. Denker, G., Kagal, L., Finin, T., Sycara, K., Paoucci, M.: Security for daml web services: Annotation and matchmaking. In: Second International Semantic Web Conference (2003)
21. Dobson, G., Sawyer, P.: Revisiting ontology-based requirements engineering in the age of semantic web. International Seminar on Dependable Requirements Engineering of computerised Systems at NPPs (2006)
22. Fenz, S., Weippl, E.: Ontology based it-security planning. In: Twelve Pacific Rim International Symposium on Dependable Computing (2006)
23. Firesmith, D.: Engineering safety related requirements for software intensive systems. In: 27th International Conference on Software Engineering (2005)
24. Karyda, M., Balopoulos, T., Gymnopoulos, L., Kokolakis, S., Lambrinoudakis, C., Gritzalis, S., Dritsas, S.: An ontology for secure e-government applications. In: International Conference on Availability, Reliability and Security (2006)
25. Kim, A., Luo, J., Kang, M.: Securit ontology for annotating resources. In: 4th International Conference on Ontologies, Databases, and Applications of Semantics (2005)
26. Undercoffer, J., Joshi, A., Pinkston, J.: Modeling computer attacks: An ontology for intrusion detection. In: 6th International Symposium on Recent Advances in Intrusion Detection, pp. 113–135. Springer, Heidelberg (2003)
27. In: RE 11th IEEE International Conference on Requirements Engineering (RE 2003), September 8-12. IEEE Computer Society, Los Alamitos (2003)