Emergency Systems Modelling using a Security Engineering Process

Jose Fran. Ruiz[1], Antonio Maña[2], Marcos Arjona[2] and Janne Paatero[3]
[1]*Fraunhofer SIT, Darmstadt, Germany*
[2]*Computer Science Department, University of Malaga, Malaga, Spain*
[3]*RUAG Switzerland Ltd., Bern, Switzerland*

Keywords: Security, Conceptual Modeling, Emergency Support Tools, Domain-specific Tools, Case Studies.

Abstract: The engineering and development of complex security-sensitive systems is becoming increasingly difficult due to the need to address aspects like heterogeneity (of application domains, requirements, regulations, solutions, etc.), dynamism and runtime adaptation necessities, and the high demands for security and privacy of the users and agencies involved in scenarios where these systems work (natural disasters, accidents, terrorism, etc.). Moreover, security knowledge is highly domain-dependent and dynamic. These characteristics make the development of those systems hard because the amount of security knowledge required to dealing with such a huge variety of situations, which becomes way too large for a human. We propose in this paper a security-oriented engineering process that is especially useful for these systems. It makes security fit naturally in the systems by interleaving security into the initial architecture and system description. In particular, the proposed process provides means to identify and manage security properties in a consistent and intuitive manner. To illustrate our experience we use a real-world emergency response scenario. More concretely, we focus on the establishment of a secure ad-hoc wireless mesh communication, which is a key component in the domain of spontaneous broadband communication among crisis management vehicles.

1 INTRODUCTION

The engineering and development of complex security-sensitive systems is becoming increasingly difficult due to the need to address aspects like heterogeneity (of application domains, requirements, threats, regulations, suitable solutions, etc.), dynamism and runtime adaptation necessities, and the high demands for security and privacy of the users and agencies involved in scenarios where these systems work (natural disasters, accidents, terrorism, etc.). Moreover, security knowledge is very domain-dependent and dynamic. Threats, properties, solutions, etc. that are valid or relevant in a given domain, are not applicable to other domains and are subject to constant changes. These characteristics make the development of those systems hard because the amount of security knowledge required for dealing with such a huge variety of situations, becoming way too complex for a human.

Besides, most of the time, these systems must use externally developed components that have a complex set of characteristics according to their security features. In particular, security requirements are especially hard to properly address because of the domain-specific knowledge required to understand their security concepts.

Modelling is one of the most important activities in the process of engineering these systems because it establishes the foundations for the rest of the engineering activities. Unfortunately, current modelling formalisms do not seamlessly and naturally integrate security. Ideally, a security modelling activity should (i) be able to coherently deal with the different elements and concepts that are related to security (properties, requirements, threats, attacks, verification, assurance, etc.) and (ii) be useful as a basis for the selection and integration of appropriate security solutions during the design phase, for the configuration of the security components during the deployment phase, for facilitating the testing phase and even for streamlining system evolution.

This paper describes the experience of using a novel secure modelling and engineering process developed in the EU SecFutur project (SecFutur Consortium, 2010). The main objective of SecFutur is supporting the development of dependable and secure systems composed of embedded components.

Ruiz J., Maña A., Arjona M. and Paatero J..
Emergency Systems Modelling using a Security Engineering Process.
DOI: 10.5220/0004480201170124
In *Proceedings of the 3rd International Conference on Simulation and Modeling Methodologies, Technologies and Applications* (SIMULTECH-2013), pages 117-124
ISBN: 978-989-8565-69-3

To this aim, SecFutur has developed a new security modelling framework and an associated engineering process that can flexibly integrate security considerations into the system design and can be easily incorporated into existing engineering processes. The security solutions are provided in terms of SecFutur Patterns (SFP) and Security Building Blocks (SBBs) (Grawrock, 2009) (Pearson, 2002), which integrate existing hardware and software security mechanisms in order to provide complex security properties. The architecture of the SecFutur Modelling Framework (Jose Fran. Ruiz and Maña, 2011) is based on the UML metamodeling capabilities and is composed of three different layers that cover different objectives.

To illustrate our experience we use a real-world emergency response scenario. More concretely, we focus on the establishment of a secure ad-hoc wireless mesh communication, which is a key component in the domain of spontaneous broadband communication among crisis management vehicles. The communication between the vehicles occurs on a tactical level, without any fixed or deployable communication infrastructure available. Depending on the type of emergency or crisis, the fixed and deployed infrastructure may be totally destroyed or partially available. In the latter case, the tactical communication is augmented with backhaul links to headquarters.

The rest of the paper is structured as follows: in Section 2, we describe the state of the art of the current modelling and engineering processes. Section 3 presents the proposed security engineering process. Section 4 describes the real-world scenario we use as an example. Section 5 contains the modelling of the scenario and Section 6 discusses the benefits and conclusions of the engineering process.

2 RELATED WORK

Currently, although some companies use security engineering processes in the development of their systems, usually, these practices for modelling systems with security properties consist of a traditional system modelling, subsequent implementation and testing, combined with ad-hoc methods (”gut feeling”) to define security issues and, in the best cases, some isolated systematic threat modelling. Most of the times those issues are only considered after the architectural and functional design of the system.

The Unified Modeling Language (UML) has become the de-facto standard notation for system development. This language is well supported by platforms and tools and suited for defining designs at high abstraction levels. It offers an excellent opportunity to close the gap between software and security engineering and to support the development of security-critical systems in an industrial setting. Although originally not based on UML, the work proposed in (Peter Herrmann, 2006) includes an extension intended to support the development of an abstract UML-based business process specification.

UMLsec (Jürjens, 2001) is proposed as an extension of UML for modelling security properties of computer systems. Unfortunately, UMLsec only addresses a few specific security requirements and doesn’t allow to create complex or composed ones that can be necessary for a specific scenario.

Model Driven Security (Basin et al., 2003) is a specialization of the MDA approach that proposes a modular approach combining languages for modelling system design with languages for modelling security. One of these languages for modelling security, called SecureUML (T. Tryfonas, 2001), is only used to describe role-based access control policies.

Another recent approach proposes to integrate security and systems engineering using elements of UML within the Tropos methodology (Castro et al., 2001) and (Mouratidis et al., 2003). This approach is a requirement-driven process, therefore it collects all the requirements and the system is modeled as a composition of subsystems interconnected through data. The complexity of obtaining how the actors interact and the sequence of their actions hinders a friendly approach to this engineering process. Also, the engineer needs to have specific knowledge of the scenario and the necessary security properties to fulfill them.

Another interesting approach to the introduction of security, focused on the concept of risk, in the development cycle is presented in (Dimitrakos et al., 2002). This process begins with a rigorous analysis of the context, collecting different aspects and concerns such as the specific requirements. The risks are identified and analyzed, in order to be included in later stages of the process, but this approach merges entities with different natures and behaviors such as threats, vulnerabilities or incidents, treating them in the same way and ignoring their own characteristics.

Finally, on the conceptual level, the Systems Security Engineering Capability Maturity Model (SSE-CMM) (SSE-CMM,) highlights the relationship between security engineering and systems engineering, regarding the former as an integral part of the latter and not an end by itself. On the other hand it does not provide any concrete realization of the proposed integrated treatment of security and systems engineering.

This overview shows that, based on the variety of different approaches, and although several specific properties can be considered more or less rigorously

in the development process, there is still a large gap between the integration of security and systems engineering. The goal of the SecFutur process is precisely to fill that identified breach by providing an security engineering process with tool support, allowing rigorous treatment of security and reliability requirements and making the experience and knowledge of security experts available for system engineering.

3 SECURITY ENGINEERING PROCESS AND ARTEFACTS

The main objective of the SecFutur Security Engineering Process is to help developers and engineers in the treatment of security aspects and the use of security elements in order to enhance system models with security. This is done by applying security properties in order to fulfill the security requirements of the system, which are obtained by means of a security analysis of the targeted scenario. Our proposed security engineering process integrates security solutions into a modelling framework which can be easily integrated with other processes.

3.1 Artefacts

The architecture of the artefacts of the security engineering process is composed of three different layers that aim at different objectives. Each layer is based on the previous one and defines a more specific model of security aspects. Figure 1 shows the dependencies between the different layers. Due to the limitation of the paper we only present the description and functionality of each layer. The three layers of the SecFutur framework are based on the UML Standard Metamodel. However, it is important to clarify that we use the term domain to refer to specific application domains (e.g. wireless sensor networks, smart metering systems, etc.), while in the field of modelling it is normally associated with the notion of Domain Specific

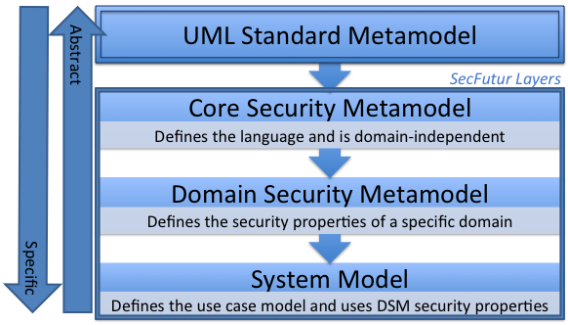

Figure 1: Layers in the SecFutur Engineering Process.

Language (DSL), and focused on code generation.

The first (most abstract) SecFutur layer is called Core Security Metamodel (CSM) and is materialized as a metamodel that contains elements (in the form of UML meta-classes) and relations to represent relevant security concepts, such as properties, requirements, threats, attacks, assumptions, actors, tests, etc. This layer deals only with concepts (e.g. ”attack”) and not with instances of these concepts (e.g. ”distributed denial of service”). Thus, this layer is domain-independent and we could say that it defines the common language (vocabulary, rules, etc.) to express security-related information. Figure 2 shows an example of a CSM.

The CSM presented in Figure 2 shows only the classes and relations between the elements. Due to size limitations we do not show the attributes of each element but these metaclasses contains relevant fields for the language usability, such as attributes to identify the author, version, description or to include references to external components. The elements of the CSM focus on different targets. In order to facilitate the understanding of the CSM we have divided it into six different expertise subsets. Each one focuses on a security concept. The different subsets are:

- Properties Model (SF_PM): focuses on the definition of security properties and its characteristics
- Requirements Model (SF_RM): describes the relations between security properties, requirements and solutions
- Threat Model (SF_TM): defines and describes security threats, attacks and attackers
- Domain Model (SF_DM): represents the domain to which the DSM is applicable, including the domain-related elements of the real world (database, communication networks, etc.) and the list of known actors or roles of the domain
- Assurance Model (SF_AM): represents the assurance and certification mechanisms used in the domain
- V&V Model (SF_VVM): this part of the metamodel is used to represent validation and verification approaches such as testing strategies, etc.

Despite the conceptual division in subsets aiming at facilitating the understanding of such a complex metamodel, the CSM is a single metamodel and cannot be splited.

The specification of the security knowledge for a specific domain is done in the second layer of the framework by creating a Domain Security Metamodel (DSM). DSMs allow experts to capture security knowledge (properties, solutions, threats, etc.) for

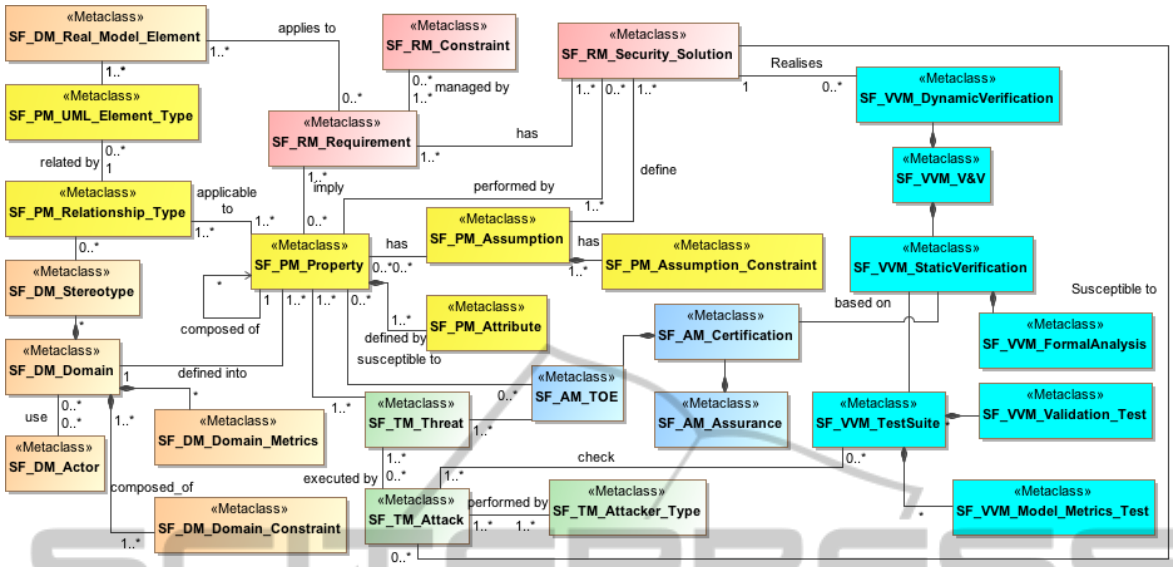

Figure 2: Extract of the CSM.

a specific domain. Some examples of domains are Trusted Meters, Secure Cash Systems, etc. and every one has a unique set of characteristics, constraints, security properties, etc. One domain and its corresponding DSM is distinguished from another one by means of the definition of the domain instance, its attributes, objective and identifier.

SecFutur promotes separation of duties, allowing experts to define models in their domain of expertise (e.g. a standardization body) or domain of authority (e.g. the security officer of a company). An additional advantage of this approach is that applications are decoupled from the specific security aspects of the context, which are captured in the DSM. This ensures that application designs remain valid for all those contexts. There can be as many DSMs as application domains, and these can be combined in a modular way.

Finally, the System Model (SM) describes the UML model of the targeted system. The SM has the information of the targeted scenario and the security properties that make it secure (according to the security requirements obtained in the security analysis of the system done by the System Engineer). Besides, the security properties have information such as the attacks that could compromise that property, the tests that could be done to check the resilience of the system, the assumptions of the solution, etc.

3.2 Security Modelling Activities

The SecFutur Security Engineering Process for the creation of complex systems is done by means of the System Engineers. They access DSM repositories in order to obtain the data and objects for their modelling scenarios. In particular, they select DSMs for their domains, import them into their model and then use the elements of the DSM to express the security aspects of their systems.

A System Engineer can use one or more DSMs for her modelling. It depends on the number of domains the scenario has. She uses many security properties of the DSM (or DSMs) in order to fulfill the security requirements of the system. Once she uses one she needs to define the requirement class of the security property, as it will help later in the implementation phase. A description of the creation and use of the other artefacts of the Security Engineering Process can be found in (Jose Fran. Ruiz and Maña, 2011).

3.3 Tool Support

The SecFutur Process Tool (SPT) is a plugin for MagicDraw (NoMagic, 1995), designed to help and support security engineers in using the SecFutur Security Engineering Process. The tool provides specific functionalities for the different actors and activities described above. Figure 3 depicts the Imported Properties browser showing the properties contained in a DSM (in this case ”MANET (Mobile Ad-hoc NETwork) for Emergency Systems”). In this Figure, the properties are classified according to the type of element they can be applied to. For instance, the property ”Storage Integrity” can be applied to Classes and Attributes. The Figure shows other info such as the CSM used to create the DSM (SecFutur CSM v2.6), the name and version of the DSM, the info of a selected security property (Secure Confidentiality), etc.

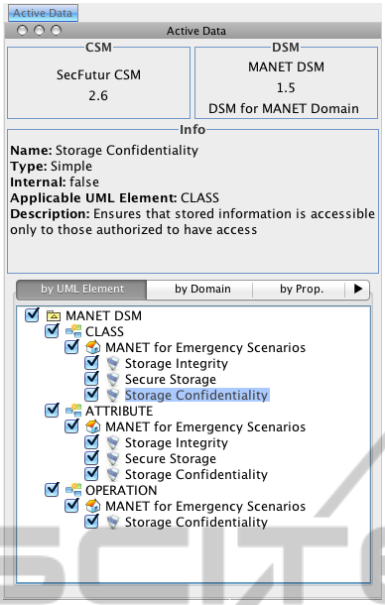

Figure 3: SPT Property Browser.

4 SCENARIO DESCRIPTION

The mobile command post (MCP) use case is illustrated in Figure 4. Typically, in this scenario, various agencies such as fire fighters, police and rescue teams, work in a joint mission. Such missions can be known in advance and thus pre-planned (e.g. in the case of major events, like concerts or large sports events) or they can be spontaneous (e.g. in case of environmental catastrophes or serious accidents). The action forces of the agencies build a MCP network (referred to as Mesh in the figure) on demand on site.

The MCP network is based on ad-hoc mesh network technology. This type of network operates autonomously in a self-configuring manner, thus decreasing the deployment burden by reducing the entire networking configuration effort. The interconnection between vehicles and action forces is typically based on wireless communication technology. In addition to the communication services in the operational area, an interconnection to external networks, backbones and terrestrial infrastructure is typically required in order to reach various information systems available in headquarters as shown in Figure 4.

In the MCP network scenario we adopt the architectural approach of an infrastructure mesh. The network is based on a series of trusted mesh network nodes (TMN) with the functionality of a mesh router and used to build a mesh backbone. TMNs are deployed in vehicles and are responsible for building the mesh infrastructure, providing network connectivity, packet routing and forwarding to other mesh routers. Action forces use client devices (e.g. phone, PDA, computer) with different applications installed in them. These client devices are connected to TMNs through wireless or wired links.

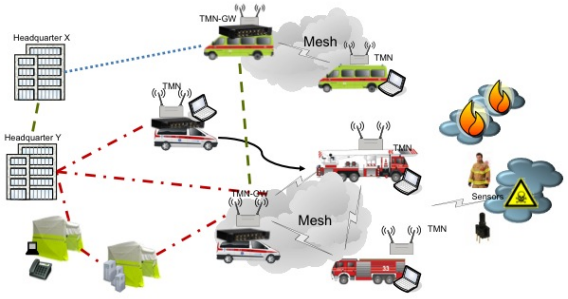

Figure 4: Mobile Command Post Scenario Overview.

Our use case focuses on the dependability of the network infrastructure itself, excluding the application, clients and user domain based security issues. The rationale behind this decision is the following: in an ad-hoc mesh network, new security challenges emerge. Since nodes build the network spontaneously and they can join and leave the network at any time, nodes have to be able to recognize trusted communication partners, to detect malicious or compromised nodes and to exclude such nodes from the network. Therefore, authentication and authorization of nodes are key security functions to ensure authorized access to the network and to information sources on nodes. Additionally, secure storage of information on nodes and secure key distribution and storage for security protocols are key features. Because of wireless communication links, integrity and confidentiality of transferred data is especially important.

The security requirements of the use case are listed below.

- Authentication and trust establishment between TMNs: before connecting to each other the TMNs authenticate mutually using cryptographic means and prove their trust states.
- Authenticity of mesh network topology: only authenticated and trusted TMNs can be part of the trusted mesh network.
- Confidentiality of mesh network topology: the mesh network topology information must remain confidential to trusted TMNs.
- Authenticity of payload traffic: payload traffic (all traffic except the mesh routing protocol traffic) sent between TMNs must be authentic.
- Confidentiality of payload traffic: the payload traffic sent between TMNs remains confidential for trusted TMNs.

- Protection of log data: the integrity of the information about events and system logs stored in TMNs must be protected.

5 MODELLING OF THE SCENARIO

This section presents the modelling of the use case using the Security Engineering Process. We focus on the parts of the process that are directly related to the modelling of the system. First we start with the description of the key characteristics of the emergency system scenario, next we describe the results of the analysis of the security requirements of the system. We also summarize the creation of the DSM used in this scenario and the development of the SM with the security aspects included.

Following on from this, we present some (of many) of the technical key characteristics, assumptions and constraints of the TMN devices:

- TMNs are installed in vehicles, therefore power consumption (e.g. battery life) is not a primary design criterion. Physical access to the installed devices is secured.
- Communication between the vehicles is primarily wireless and must be secured.
- We assume that the mission planning and configuration phase is carried out by trustworthy personnel in a physically secure environment.
- The devices are assumed to be free of malicious behaviour when correct states are defined in the mission planning and configuration phase.

The Security Requirements and Functionality Analysis describes the analysis and description of the emergency use case. The analysis is focused on the modelling activity, so all the descriptions are high-level abstractions of the system functionality.

In order to present a better analysis of the system we divided it into different sub-scenarios. Due to the limitation in size of the paper we present only the security requirements analysis of the Mission Planning and Configuration phase of the TMNs. This scenario deals with the preparation of different missions by creating different models of typical mission configurations that are later activated when necessary. It is composed of three different use cases: define and generate mission-specific configuration, activate or remove functionality and update software packages.

After analyzing these use cases, we describe and model their functionality and security requirements using diagrams. The security definitions are:

- Define and generate mission-specific configuration security properties: secure authentication, generate configurations (uses confidentiality and integrity) and acknowledgments (secure communication property).
- Activate or remove functional security properties: authentication, secure install or remove software components and acknowledgments.
- Update software packages: authentication, secure install and secure remove.

Regarding the modelling part, this scenario has only one domain (from the security point of view): MANET for Emergency Systems. Consequently, we must import into our model the corresponding DSM. Normally, this DSM should be available when the system modelling starts, acting as a security library ready to be imported. In the specific case of our experience, we developed such DSM, as the modelling approach is still under development in the project. The design of DSMs is done by the Security Domain Experts, therefore, we consulted security experts at RUAG . The creation of DSMs gathers and represents the security knowledge of the domain, which is not related to a particular application. Therefore, the DSM contains a large number of elements.

To illustrate this modelling activity, we use the features of the SPT from the approach of a Security Domain Expert. We focus on one specific security property (Storage Integrity) and show the part of the model that is directly linked to this property. In order to create the DSM we start with an empty model containing the CSM . Once the CSM language is loaded, we start adding instances of the CSM metaclasses for the elements. We present only some of them in this paper. We create an instance for the domain called ”MANET for emergency scenarios”. We add an instance for the Property ”Storage Integrity”, we define the Element Type ”Class” where it can be applied using an standard ”Association” and we add a security threat called ”Data Alteration”. We continue adding other related elements such as attacks, attacker types, solutions (in the form of SBBs), etc. For each element we also complete additional internal information to further describe it. Completely fill the DSM is beyond the scope of this paper, but to help readers understand the result, Figure 5 shows a partial view focused on the Storage Integrity property.

The development of the System Model is essentially a normal UML modelling. We (in this case with the role of System Engineer) start by importing the appropriate DSM from a repository of DSMs, as if they were security libraries, and then we model the system including classes, methods, relations, etc.

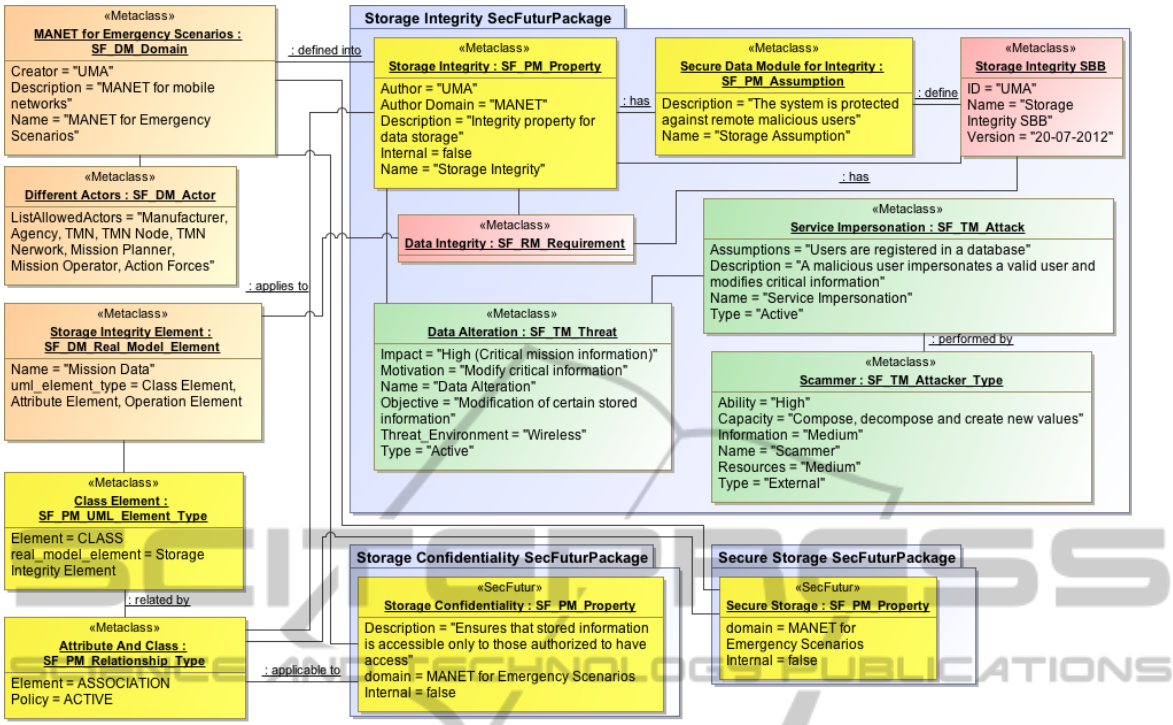

Figure 5: Partial view of the MANET for emergency scenarios DSM.

As we add classes to represent elements of our system, we link the ones with security requirements (obtained after doing the security analysis of the system) to the corresponding security properties of the DSM in order to fulfill the security requirements. This is done by using the SPT. If we select an element and right-click on it, it will show the possible properties that can be attached to it(this is defined in the DSM). When the system engineer selects one of the properties the tool automatically creates and imports all the related elements of the security property such as solutions (SBBs), threats, assumptions, etc. to the element of the system model, as we can see in Figure 6. The tool also allows users to select the visibility of the different elements that are automatically added to the model, in order to avoid overcomplicating view. For instance, we can choose to see only the requirements. In a later stage, when all the security requirements of the scenario are fulfilled, the tool will allow us to replace security properties by specific solutions (SBBs), thus evolving the security elements from the requirements phase to the design phase.

6 CONCLUSIONS

The main goal of the security engineering process is a clear separation between the expertise domains. Security engineering is not only the task of the highly experienced security expert. Using this approach, also system engineers are in a position to make sound security engineering decisions. In general terms, the integration of security engineering into the regular UML-based system engineering has at least two benefits. First and foremost, as security engineering is naturally integrated from the beginning into the modelling process, it helps to avoid design decisions that are contrary to the security requirements (a frequent problem when security is only considered in the final stages of development). Second, the SecFutur artefacts provide a common language for the roles involved in the engineering of the system, facilitating their communication.

The SecFutur process is especially designed to fit into modern iterative (or even agile) development process as it is based on a model-driven philosophy. The approach is based on the following principles:

- Modelling and implementation should be considered, communicated and managed to be iterative. The model should be allowed to change between iterations, as design is emergent.
- Modelling and implementation can be done even in parallel. Often an implementation (even partial) helps to identify which design makes sense. This emphasizes that design is emergent, and usually the form (design) follows the function (imple-

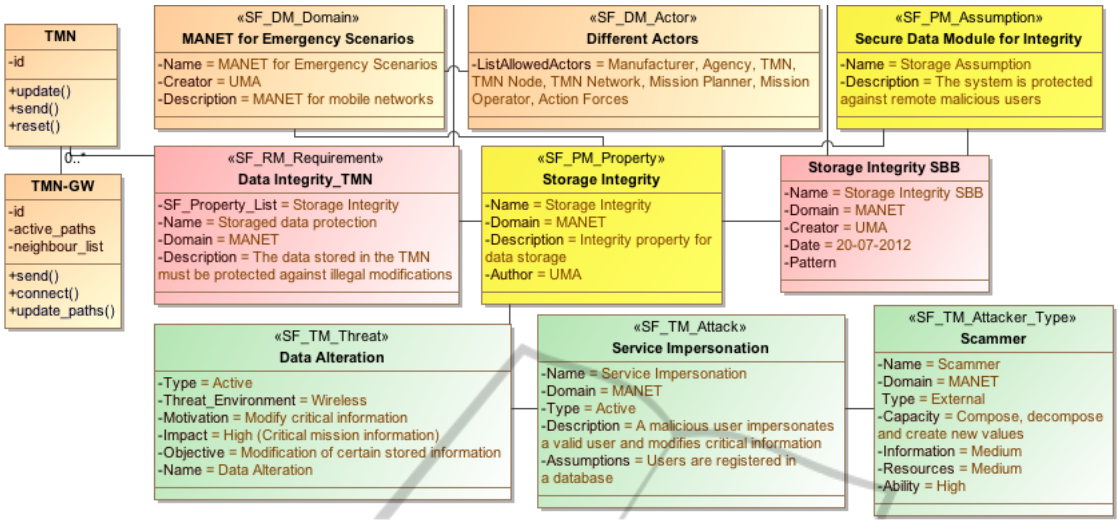

Figure 6: Security Property Imported into a System Model.

mentation).

- The savings originate from reuse. The SecFutur engineering process supports reuse in the form of the DSMs and, in particular, the SBBs.

Summarizing our conclusions, we have presented and applied the novel security engineering approach to a real engineering use case, the emergency systems scenario. The experience suggests (from real experiences in the SecFutur project) that the novel approach is combinable with the already existing engineering formalisms and iterative and agile development methods.

ACKNOWLEDGEMENTS

This work is partially supported by the E.U. through FP7 project SECFUTUR (IST-256668) and FP7 project CUMULUS (IST-318580).

REFERENCES

Basin, D., Doser, J., and Lodderstedt, T. (2003). Model driven security for process-oriented systems. In *SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies*. ACM Press.

Castro, J., Kolp, M., and Mylopoulos, J. (2001). A requirements-driven development methodology. In *Proc. of the 13th Int. Conf. On Advanced Information Systems Engineering (CAiSE)*.

Dimitrakos, T., Ritchie, B., Raptis, D., and Stølen, K. (2002). Model based security risk analysis for web applications: the coras approach. In *Proceedings of the 2002 international conference on EuroWeb*.

Grawrock, D. (2009). Dynamics of a Trusted Platform: A Building Block Approach. Intel Press (2009).

Jose Fran. Ruiz, R. H. and Maña, A. (2011). A security-focused engineering process for systems of embedded components. *SD4RCES 2011*.

Jürjens, J. (2001). Towards development of secure systems using umlsec.

Mouratidis, H., Giorgini, P., and Manson, G. (2003). Integrating security and systems engineering: Towards the modelling of secure information systems. In *Proceedings of the 15th Conference On Advanced Information Systems Engineering (CAiSE)*. Springer-Verlag.

NoMagic (1995). Magicdraw uml tool.

Pearson, S. (2002). Trusted computing platforms, the next security solution. Technical report, Trusted Systems Lab, HP Laboratories.

Peter Herrmann, G. H. (2006). Security-oriented refinement of business processes. springer-verlag. *Electronic Commerce Research Journal*, 6.

SecFutur Consortium (2010). Design of secure and energy-efficient embedded systems for future internet applications (secfutur), ist-25668. fp7.

SSE-CMM. The systems security engineering capability maturity model (sse-cmm).

T. Tryfonas, E. Kiountouzis, A. P. (2001). Embedding security practices in contemporary information systems development approaches. *Information Management & Computer Security*, (4).