A CASE Tool to Support Automated Modelling and Analysis of Security Requirements, Based on Secure Tropos

Michalis Pavlidis, Shareeful Islam, and Haralambos Mouratidis

School of Architecture, Computing, and Engineering, University of East London, UK
m.pavlidis@ieee.org, {shareeful,haris}@uel.ac.uk

Abstract. Secure Tropos, an extension of the Tropos methodology, considers security requirements alongside functional requirements, from the early stages of the system development process. The Secure Tropos language uses security concepts such as security constraint, secure goal, secure plan, secure resource, and threat to capture the security concepts from both social and organisational settings. These concepts are used to model and reason about security for a specific system context. This paper presents a CASE tool, called SecTro, which supports automated modelling and analysis of security requirements based on Secure Tropos. The tool's architecture, layout, and functionalities are demonstrated through a real world example using the Secure Tropos concepts.

Keywords: Security, Goal Modelling, Requirements Engineering, Secure Tropos, SecTro, and CASE tools.

1 Introduction

Information systems play an important role in education, health care, banking, and transportation. Ensuring security of information systems is a vital and challenging task. Such systems often store sensitive customer and business information, which needs adequate protection [1]. There should be cost effective and realistic protection mechanisms to protect such systems against any potential attack. It is already agreed, by the industry and relevant research communities, that security should be considered from the early phases of the development process [2]. Identifying and analysing the security requirements along with the functional requirements provides a better comprehension of the system's security issues and limits the conflicts between the security and functional requirements for more secure information systems [3, 4, 5].

Secure Tropos is a security requirements engineering methodology that considers security issues throughout the whole development process [6]. The approach identifies, models and analyses the security concepts within the organization and social settings from the early stages of development. However, considering security issues from the early development stage may increase the number of activities during the development, which may not be always affordable depending on the project's specific constraints. Therefore, there is the need for an automated tool to support

S. Nurcan (Ed.): CAiSE Forum 2011, LNBIP 107, pp. 95–109, 2012.

security modelling activities based on the Secure Tropos concepts [3]. This paper demonstrates a tool, named SecTro, which supports the designers during the security modelling activities and assists them in producing the output based on the activities. The tool demonstration includes the architecture, the layout, and the functionalities based on a real example.

The rest of the paper is structured as follows. Section 2 is a review of Secure Tropos. Section 3 illustrates the tool that supports Secure Tropos. Section 4 discusses the related work, while section 5 concludes the paper and presents future work.

2 Secure Tropos Methodology

Secure Tropos is an extension of Tropos methodology that takes security into account. Secure Tropos considers the basic Tropos concepts such as dependency, goal, task, resource, and capability and adds security concepts such as security constraint, secure goal, secure plan, secure resource, and secure capability [6, 7]. Secure Tropos includes the following modelling activities:

- Security reference modelling. The security reference modelling activity involves concepts such as the security features of the system under development, the protection objectives of the system that contribute positively towards the security features, the security mechanisms that contribute positively towards the protection objectives, and the threats that have a negative contribution towards the security features of the system. The security reference diagram can be used as a reference point during the development process.
- Security constraint modelling. Involves the modelling of the security constraints imposed to the actors and the system, and it allows developers to perform an analysis by introducing relationships between the security constraints or a security constraint and its context.
- Secure entities modelling. Involves the analysis of the secure entities of the system and it is considered complimentary to the security constraints modelling.
- Secure capability modelling. Involves the identification of the secure capabilities of the actors that guarantee the satisfaction of the security constraints.

In addition, Secure Tropos consists of four stages:

- Early requirements. Stakeholders are modelled as social actors, which depend on each other in order to achieve goals, carry out plans or deliver resources. The outcome of this phase is an actor diagram and a number of goal diagrams.
- Late requirements. In the late requirements the actor diagram of the early requirements is extended with the introduction of the system to be as an actor that has a number of dependencies with the rest of the social actors. In fact, these dependencies will be the functional and non-functional requirements of the system.

- Architectural design. According to the system requirements the designer in this phase defines the global architecture of the system, which consists of a number of sub-systems. Sub-systems are interconnected with data and control flows that are represented as dependencies. Then, each actor is assigned to an agent who has a number of capabilities. Usually, during the end of the architectural design the security attack testing takes places, where the design of the system is tested against the security requirements.
- Detailed design. In detailed design there is further specification of the agent capabilities and the interactions between the agents. Agent capabilities and their included plans are modelled with the help of UMLsec activity diagrams, while agent interactions are modelled with the help of UMLsec sequence diagrams.

The meta-model of Secure Tropos [8] is shown in Fig. 1 and more detailed information about Secure Tropos can be obtained from [6, 7].

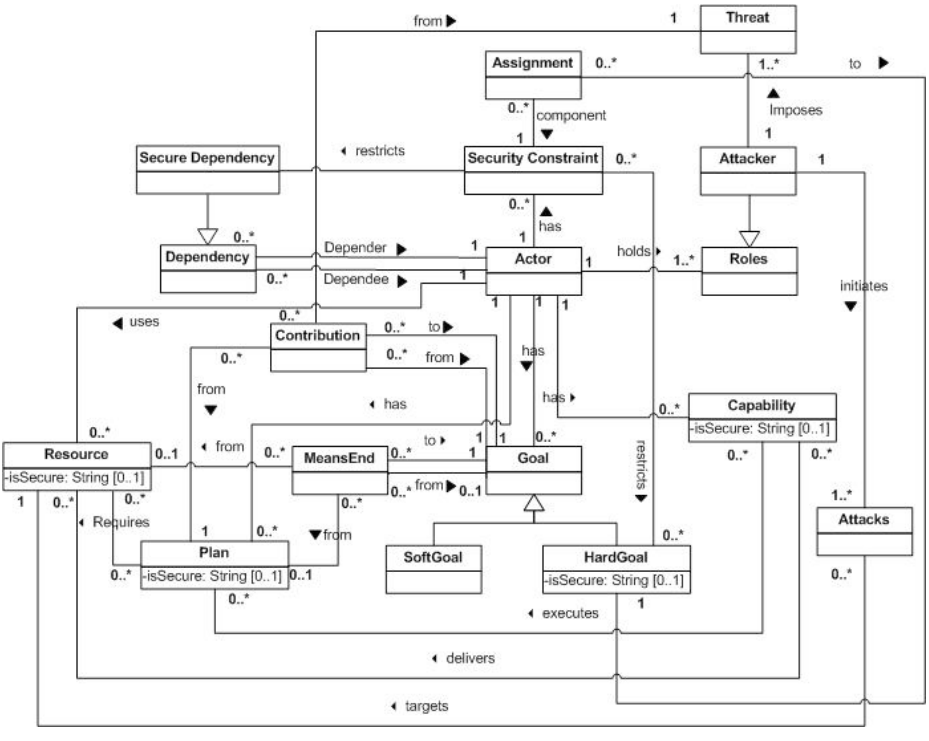

Fig. 1. Secure Tropos meta-model

3 The SecTro Tool

3.1 SecTro Architecture

The SecTro tool [9, 10] was developed to support the Secure Tropos modeling activities for the creation of the visual models and to assist the designers with the

automation of some aspects of the methodology. It also includes a basic validation of the developed models. It is a standalone application that was built with the Java programming language, which makes the tool usable across different platforms.

The package diagram of SecTro is shown in Fig. 2 and a description of the packages is given in Table 1. The main classes for the drawing functionalities of the tool are shown by a class diagram in Fig. 3. The class diagram of the graphical user interface (GUI) package is shown in Fig. 4. All the elements that can be drawn, such as an actor and a hard goal, belong to the ElementType class, while all the links between the elements, such as "plays" and "satisfies" links, belong to the LinkType class.

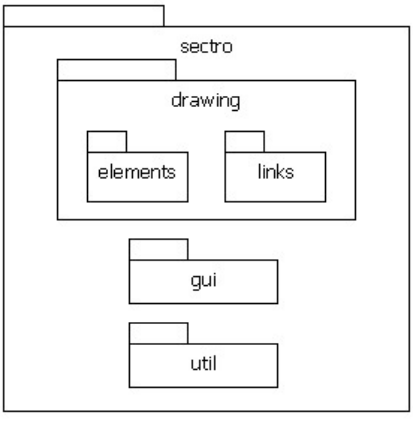

Fig. 2. Package diagram of SecTro

Table 1. Description of the SecTro packages

Package	Description
sectro	The parent package that includes the main class and all the sub packages
sectro.drawing	Contains the generalized class for all the drawing objects (DrawingObject) and the elements and links packages
sectro.drawing.elements	Contains the classes for all the drawing elements (Actor, HardGoal,Resource, Plan, etc.)
sectro.drawing.links	Contains the generalized class for all the Links (Link) and the classes for all the drawing links (LinkDependency, LinkRestricts, LinkPlays, etc.)
sectro.gui	Contains all the classes related to the user interface (MainForm, ToolBar, MenuBar, etc.)
sectro.util	Contains all the utility classes (ImageUtil, XMLUtil, FileUtil, etc.)

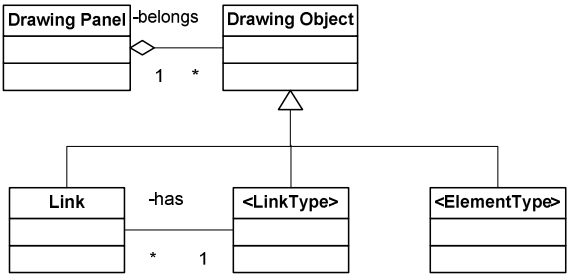

Fig. 3. Class diagram of the SecTro drawing functionality

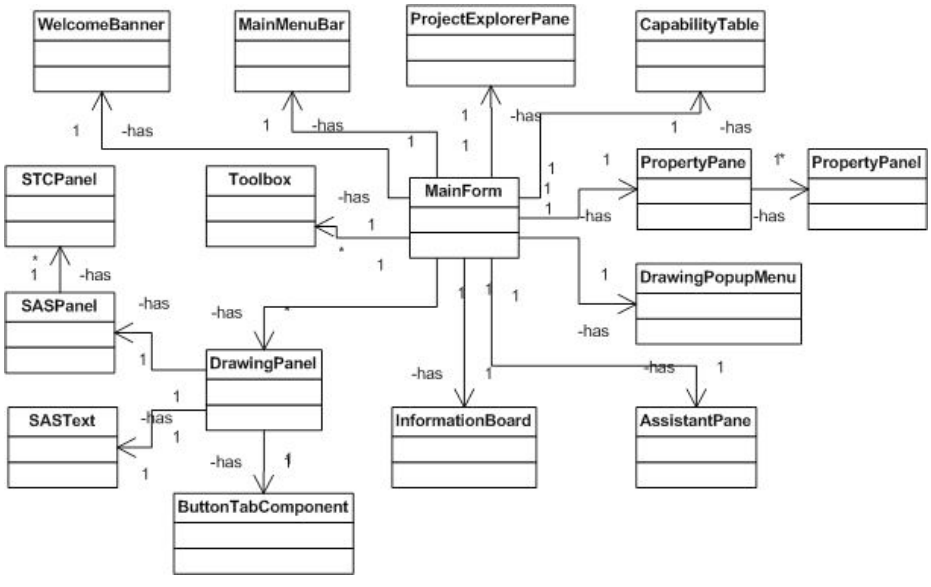

Fig. 4. SecTro GUI class diagram

3.2 SecTro Layout

SecTro's workspace, as shown in Fig. 5, consists of the drawing canvas in the centre of the workspace. The drawing canvas is the space where the designer is drawing the models. The drawing area generally automatically expands depending on the developing model size. On the top of the workspace, there is a series of tabs for showing the developed diagrams for each stage of Secure Tropos. The project explorer and the properties panel are on the right side and the toolbox is on the left side of the workspace. Apart from the tabs, the project explorer is another way for the designer to navigate through the models. Below the project explorer is the properties panel that shows information about the properties of a selected element of a model.

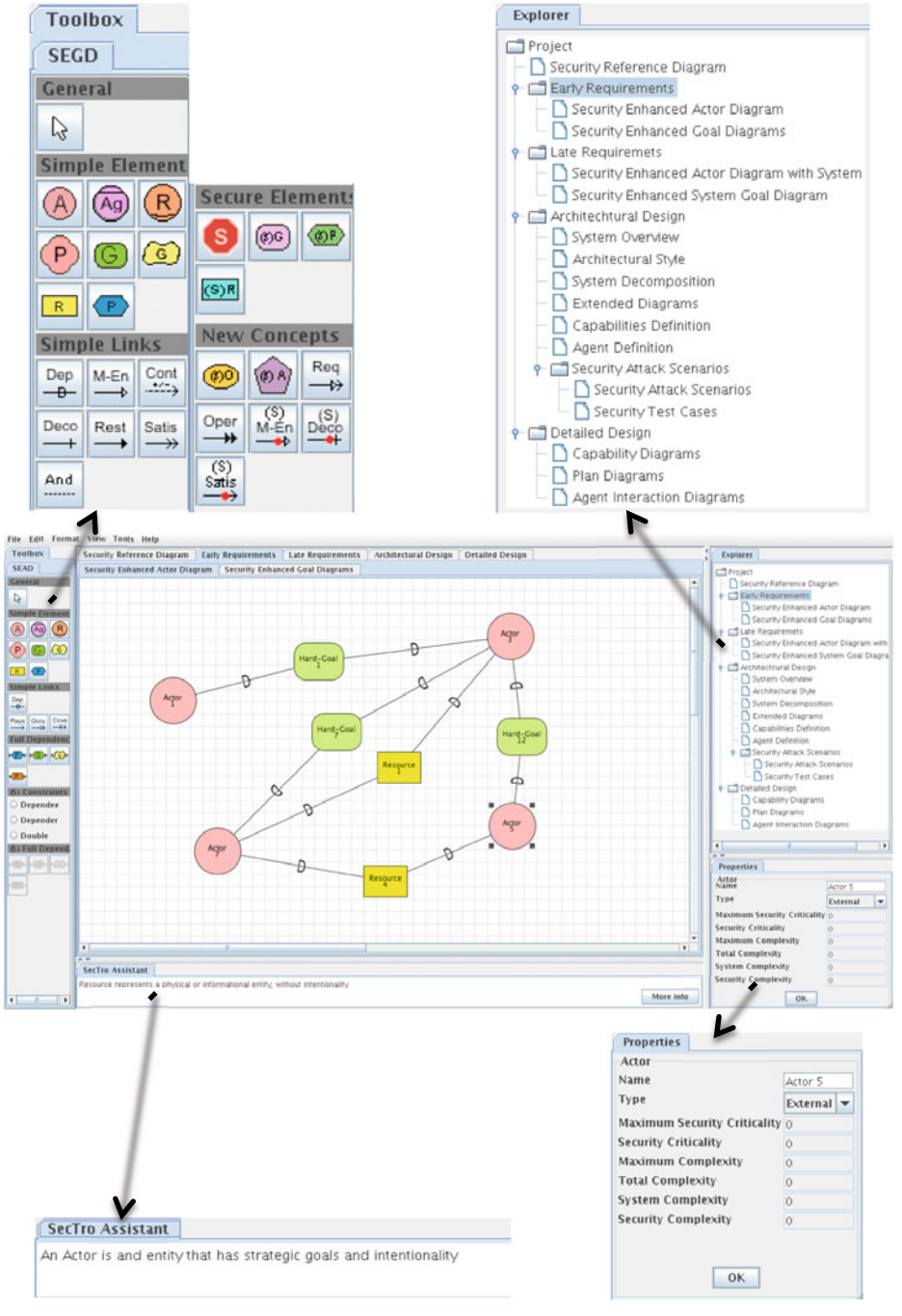

Fig. 5. SecTro workspace

The toolbox on the left contains the graphical elements of the Secure Tropos that can be added to a model. However, not all elements appear on the toolbox, but only the ones that can be used at a specific stage and model. The reason for that is to ensure the syntactical correctness of the developing models by preventing the designer to add invalid elements. Therefore, the elements of the toolbox change according to the tab that is selected. At the bottom of the workspace is the SecTro Assistant that shows information and feedback to the designer about his actions. The graphical representations of the concepts of Secure Tropos, by the SecTro tool are shown in Fig. 6.

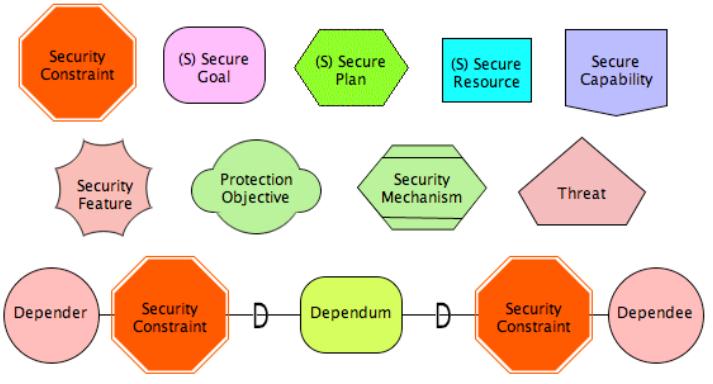

Fig. 6. Secure Tropos notation

3.3 SecTro Functionalities Supporting the Security Aware Process

The main functionalities of the SecTro are to support the security modelling activities of Secure Tropos. Therefore, the tool enables the designer to perform security reference modelling, security constraint modelling, secure entities modelling, and secure capability modelling activities.

To demonstrate the tool functionalities, we use in the rest of the section a real world example. The example is about a value added application for UK located bank that offers its customers use of smart cards (debit/credit card) to pay for purchases. To support that functionality; the bank collaborates with some retailers and the card issuer for the smart card based payment infrastructure. Security is an important aspect for this application because customer identifiable and sensitive data is handled by the participating parties. Based on the above, we identify the following important concepts.

Goals: protect personal data, secure processing.
Actors: customer, application providers (bank, card issuer) and retailer.
Plans: transaction and purchase detail, and update account balance, request payment from the customer account.
Resources: customer identifiable data, customer accounts information.

Security Constraints: Only legitimate customer, transfer minimum data.
Attacker goal: obtain customer data, unauthorised access.

During the early requirements stage the designer constructs the security reference diagram, the actor diagram and a number of goals diagrams. Figures 7, 8, and 9 present screenshots of the aforementioned diagrams created for the purpose of the paper. The designer has the option to construct the goal diagrams in the initial actor diagram or create a new tab in order to construct a goal diagram separately from the actor diagram and the rest of the goal diagrams.

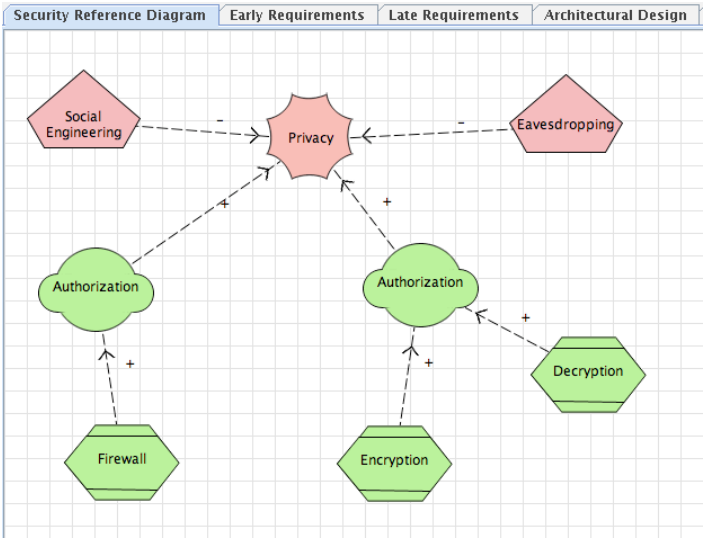

Fig. 7. Security reference diagram

In the late requirements stage the designer constructs the actor diagram with the system and the goal diagram of the system. Similarly, the system's goal diagram can be constructed in the actor diagram with the system tab or the designer can choose to construct it in the dedicated tab. Figures 10 and 11 present an actor diagram with the system and a goal diagram of the system created for the purpose of this paper.

In the actor diagram with the system the dependencies of the system with the rest of the actors are the functional requirements of the system. In the system goal diagram the designer specifies what the system needs to do in order to satisfy the requirements. The security constraints in the system goal diagram are the security requirements of the system. The designer specifies how the system satisfies these security requirements through the concepts of secure goal, plan and resource. This process is not sequential and the designer can goal back at the actor diagram with the system and add more security constraints. The system goal diagram will be updated automatically in order to include the new security constraint.

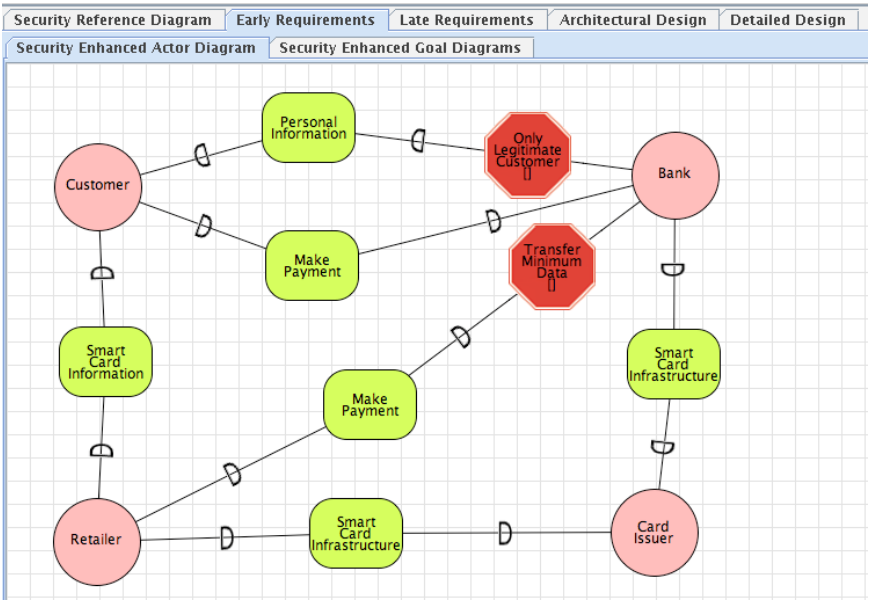

Fig. 8. Actor diagram

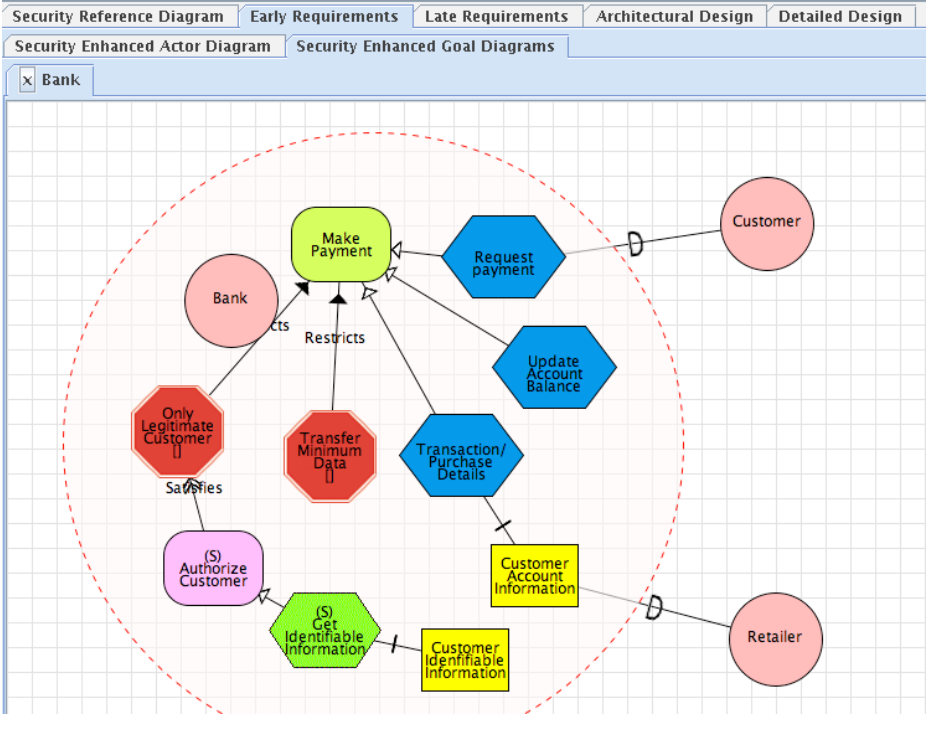

Fig. 9. Goal diagram

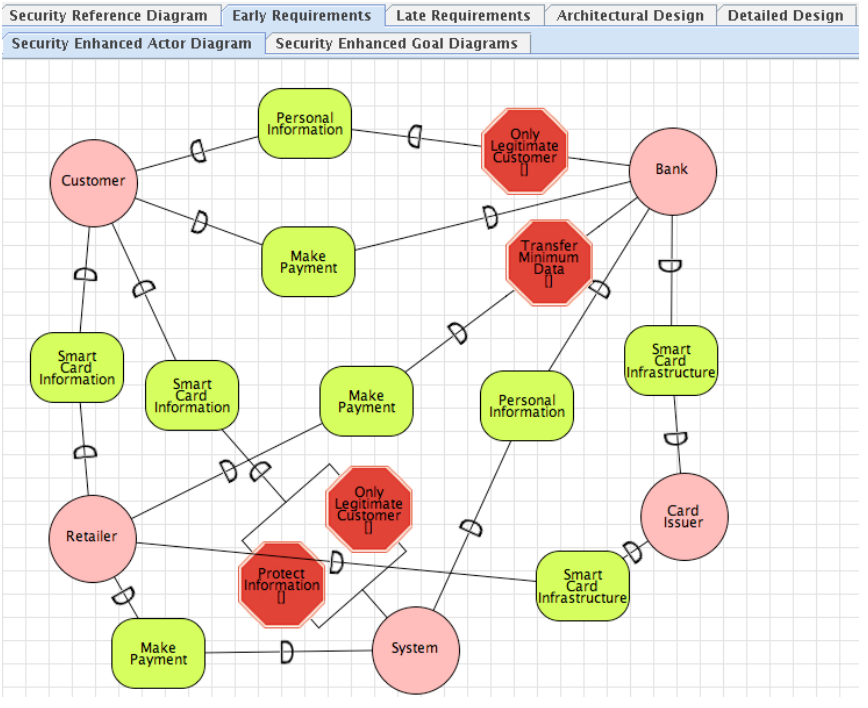

Fig. 10. Actor diagram with the system

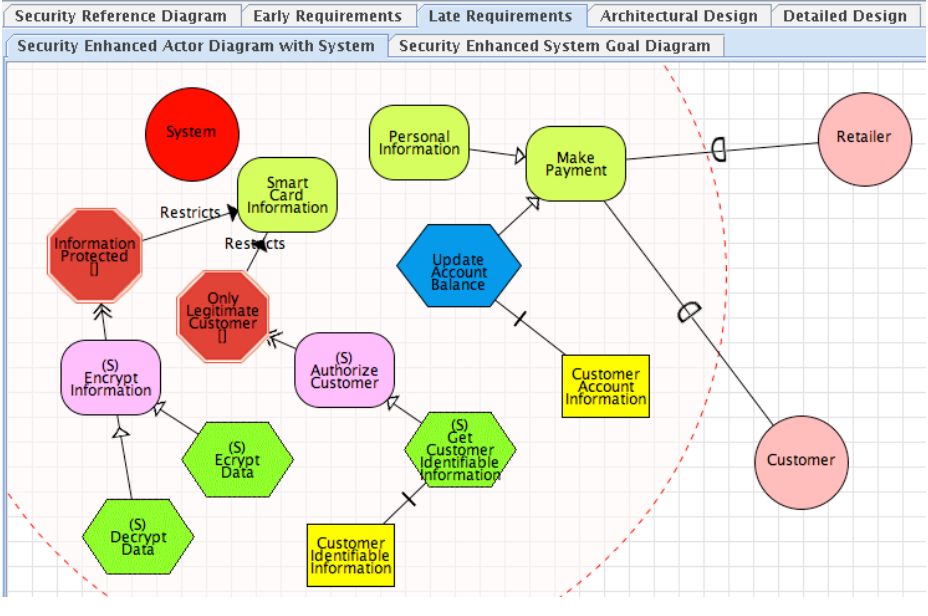

Fig. 11. Goal diagram of the System

During the architectural design the architecture of the system is defined. The tool enables the designer to construct an architecture style selection diagram where he can model different architecture styles and then model how much these architecture styles contribute to the security requirements that has identified in the previous stages.

Finally, in most cases, during the end of the architectural design the security attack testing takes places, where the design of the system is tested against the security requirements [11]. The tool automatically generates for the designer the security attack scenario template and the security test case template where the designer can model the attack and the countermeasures (Fig. 12). In the context of Secure Tropos, Security Attack Scenario is defined *as an attack situation describing the actors of a software system and their secure capabilities as well as possible attackers and their goals, and it identifies how the secure capabilities of the system's actors prevent (if they prevent) the satisfaction of the attackers' goals.*

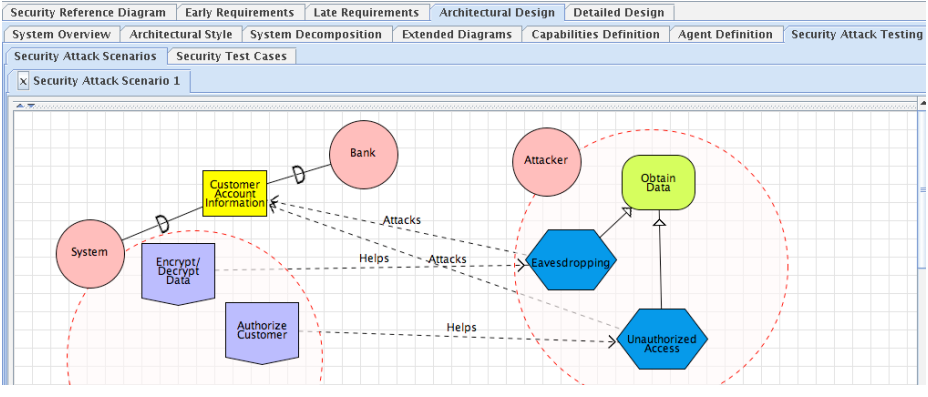

Fig. 12. Security attack scenario template

3.4 SecTro Additional Functionalities

Apart from the above-mentioned functionalities the tool provides additional functionalities such as checking the correctness of a model. The designer has the ability to draw models, but this includes the danger that designers might draw models that are not syntactically correct. Thus, the tool prevents a drawing action by the designer that is not complying with the syntax of Secure Tropos based on the meta-model and shows a respective warning notification. Therefore, a specific concept within a model cannot link with another concept unless rules are defined within the meta-model. This enables the tool to check the correctness of the individual model. For example if the developer tries to connect a security constraint with a goal with a "contributes" link, the tool will not allow such action because syntactically a security constraint connects with a "restricts" link with the goal. Furthermore, the tool checks the consistency between models. When changes are made in one model they are automatically reflected in the other models. For example, when a name of an actor is

changed in the early requirements diagrams it is automatically changed in the diagrams of the late requirements.

Complementary to the above-mentioned functionalities is the functionality of the SecTro Assistant. It is located at the bottom of the workspace and it has been developed to assist designer that are not familiar with Secure Tropos methodology. SecTro Assistant provides feedback on designers' actions, so for example, when a specific toolbox button or tab is clicked, it is showing detailed information about the concept of the button or about the modelling activity of the stage tab. In addition, when the tool has prevented an action by enforcing rule checking on the models, SecTro Assistant will show more information about the reason behind the prevention of the action. Also, by clicking a button in the SecTro Assistant the designer can see parts of the Secure Tropos meta-model that are related to his action.

Furthermore, to increase the interoperability of the tool, the tool includes a functionality that enables the designer to export the developed models in XML format. The output is a .XML file that contains the model in XML format and can be used as an input to another tool.

Last but not least functionalities that SecTro supports are the designer's options to export a model as an image, to print a model, and to customize the workspace. In case of exporting a model as an image the designer has the option to choose the extension of the output image file according to his preference. While in the case of workspace customization, the designer has the option to zoom in and out of the model, to turn the grid on or off, to change the background colour of his model, and to hide the panel around the drawing canvas in order to create more space for his model development.

Therefore, we believe the tool is capable to produce models that are required by the designer during the secure software development process. It provides a usable interface with all the concepts that the designer will need when using the Secure Tropos methodology.

4 Related Work

Although Secure Tropos is still in research and it is difficult to develop a CASE tool for a methodology that is still in research, the i* modelling framework has been out for some years and a number of related CASE tools were developed to support it. OME [12] is a general, goal or agent oriented modelling and analysis tool and its improved version, OpenOME [13], additionally supports aspect oriented engineering while it is integrated in Eclipse Integrated Development Environment (IDE), in Protégé conceptual modelling environment and other graph editing environments such as Microsoft Visio. REDEPEND [14] is a graphical modelling tool that allows the developer to build and analyze Strategic Dependency (SD) models and Strategic Rationale (SR) models and is a plug-in for Microsoft Visio. The REDEPEND-REACT [14] tool is an extension of REDEPEND and involves the ability to define properties, such as security and efficiency. Also, following some heuristics rules, it guides the process of generating an architecture by recommending specific ones. Then the candidate architectures can be evaluated against a variety of properties. TAOM4e

[15] is also a plug-in for the Eclipse IDE and supports the modelling in tropos methodology during all its phases and also generates automatically the code from the tropos specification to JADE or jadex implementations. This can be achieved by mapping the tropos meta-model concepts to the target implementation language constructs. The GR-Tool [16] is a graphical tool, where the developer can construct goal models and run the algorithms that are embedded in the tool for forward and backward reasoning. The T-Tool [17] performs model checking of Tropos specifications while the ST-Tool [17] is a tool to support an extension of tropos that provides trust management process and performs formal analysis of its specifications. J-PRiM [18] tool is a plug-in for Eclipse IDE that uses the i* modeling and supports the J-PRiM methodology, which is a re-engineering methodology, where the specification of the new system starts with the observation of the current system and ends with the specification of the system to be. SNet Tool [19] provides an automatic transformation of graphical network representations based on the extended i* into executable programs. As a result, network scenarios can be simulated and provide valuable feedback. HeRA is a tool that incorporates the concept of heuristic requirements elicitation and uses for elicitation and analyse of security requirements [4]. HeRA uses heuristic rules to represent security related experience and extend further to capture the organisational knowledge for identifying security requirements [5].

The aforementioned tools, although they were developed for different ultimate purposes, they provide support for the i* modelling framework, which is the modelling framework that was adopted by Secure Tropos as well. But, Secure Tropos introduces new concepts that none of the previous tools enables their graphical representation, i.e. security constraint, secure goal, secure plan, secure resource, and secure capability. Also, the previous tools don't provide support for the modelling activities that Secure Tropos introduces, i.e. security constraint modelling, secure entities modelling, and secure capability modelling. So, despite the fact that experienced users with Secure Tropos can make conventions and use the previous tools to construct single diagrams; these tools are not adequate to support the Secure Tropos methodology.

5 Conclusions and Future Work

This paper presented our effort to develop a CASE tool for the Secure Tropos methodology. The tool supports the designers for the security modelling activities in particular for elicitation and analysis of early and late requirements. SecTro aims to provide an editor to develop the security models based on Secure Tropos. It has a user-friendly interface, which makes it easy to use and assists analysts who are not familiar with the methodology, by providing them with information about the methodology concepts, stages, and metamodels. Also, it enforces rules and constraints and provides valuable feedback on various actions of the designers in an interactive way. The tool has already been used by the students of university of East London to model and analyse security issues of a real industry case study. However, the tool does not support the modelling activities of the detailed design stage and we consider

this as future work. In addition, future work includes the extension of the XML Schema in order to validate more models of the methodology.

References

1. Islam, S., Mouratidis, H., Jürjens, J.: A Framework to Support Alignment of Secure Software Engineering with Legal Regulations. Journal of Software and Systems Modeling (SoSyM), Theme Section on Non-Functional System Properties in Domain-Specific Modeling Languages (NFPinDSML) 10(3), 369–394 (2011)
2. Islam, S., Mouratidis, H., Wagner, S.: Towards a Framework to Elicit and Manage Security and Privacy Requirements from Laws and Regulations. In: Wieringa, R., Persson, A. (eds.) REFSQ 2010. LNCS, vol. 6182, pp. 255–261. Springer, Heidelberg (2010)
3. Mouratidis, H., Giorgini, P.: Integrating Security and Software Engineering: Future Vision and Challenges. In: Mouratidis, H., Giorgini, P. (eds.) Integrating Security and Software Engineering: Advances and Future Visions. Idea Group Publishing, London (2007)
4. Houmb, S.H., Islam, S., Knauss, E., Jürjens, J., Schneider, K.: Eliciting Security Requirements and Tracing them to Design: An Integration of Common Criteria, Heuristics, and UMLsec. Requirements Engineering Journal 15(1), 63–93 (2010)
5. Schneider, K., Knauss, E., Houmb, S.H., Islam, S., Jürjens, J.: Enhancing Security Requirements Engineering by Organisational Learning. Requirements Engineering Journal (REJ), Special Issue on REFSQ (2011)
6. Mouratidis, H., Giorgini, P.: Secure Tropos: A Security-Oriented Extension of the Tropos Methodology. International Journal of Software Engineering and Knowledge Engineering 17(2), 285–309 (2007)
7. Giorgini, P., Mouratidis, H., Zannone, N.: Modelling Security and Trust with Secure Tropos. In: Mouratidis, H., Giorgini, P. (eds.) Integrating Security and Software Engineering: Advances and Future Visions. Idea Group Publishing, London (2007)
8. Matulevicious, R.: Summary of Secure Tropos Metamodel. Internal Report, University of Namur (2008)
9. Pavlidis, M., Islam, S.: SecTro: A CASE Tool for Modelling Security in Requirements Engineering using Secure Tropos. In: Nurcan, S. (ed.) Proceedings of the Conference on Advanced Information Systems Engineering (CAiSE) Forum, London, pp. 89–96 (2011)
10. SecTro | Homepage, `http://sectro.securetropos.org/`
11. Mouratidis, H., Giorgini, P.: Security Attack Testing (SAT) – Testing the Security of Information Systems at Design Time. Journal of Information Systems 32, 1166–1183 (2007)
12. OME3, `http://www.cs.toronto.edu/km/ome/`
13. OpenOME, `https://se.cs.toronto.edu/trac/ome/`
14. Grau, G., Franch, X., Maiden, N.: REDEPEND-REACT: An Architecture Analysis Tool. In: 13th IEEE International Conference on Requirements Engineering, Paris, pp. 455–456 (2005)
15. Morandini, M., Nguyen, D.C., Perini, A., Siena, A., Susi, A.: Tool-Supported Development with Tropos: The Conference Management System Case Study. In: Luck, M., Padgham, L. (eds.) AOSE 2007. LNCS, vol. 4951, pp. 182–196. Springer, Heidelberg (2008)
16. Giorgini, P., Mylopoulos, J., Sebastiani, R.: Goal-Oriented Requirements Analysis and Reasoning in Tropos Methodology. Journal of Engineering Applications of Artificial Intelligence 18(2), 159–171 (2005)

17. Giorgini, P., Massacci, F., Mylopoulos, J., Zannone, N.: ST-Tool: A CASE Tool for Security Requirements Engineering. In: 13th IEEE International Conference on Requirements Engineering, Paris, pp. 451–452 (2005)
18. Grau, G., Franch, X., Avila, S.: J-PRiM: A Java Tool for a Process Reengineering i* Methodology. In: 14th IEEE International Conference on Requirements Engineering, Minneapolis, pp. 359–360 (2006)
19. Gans, G., Lakemeyer, G., Jarke, M., Vits, T.: SNet: A Modeling and Simulation Environment for Agent Networks Based on i* and ConGolog. In: Pidduck, A.B., Mylopoulos, J., Woo, C.C., Ozsu, M.T. (eds.) CAiSE 2002. LNCS, vol. 2348, pp. 328–343. Springer, Heidelberg (2002)