SecuriTAS: A Tool for Engineering Adaptive Security

Liliana Pasquale[1], Claudio Menghi[1], Mazeiar Salehie[1]
Luca Cavallaro[1], Inah Omoronyia[1], Bashar Nuseibeh[1,2]
[1]Lero - the Irish Software Engineering Research Centre, University of Limerick, Ireland
[2]Department of Computing, The Open University, UK

ABSTRACT

This paper presents SecuriTAS, a tool to engineer adaptive security. It allows software designers to model security concerns together with the requirements of a system. This model is then used at runtime to analyze changes in security concerns and select the best set of security controls necessary to protect the system.

Categories and Subject Descriptors

D.2 [**Requirements/Specifications**]: Methodologies

General Terms

Design

Keywords

Security, Adaptive Software, Dynamic Access Control, Goals

1. INTRODUCTION

Security is concerned with the protection of valuable assets from harm. Assets can be physical objects (e.g., laptop), sensitive information (e.g., user credentials), or intangible properties (e.g., reputation). Assets have a central role in security and may influence other security concerns, such as threats, attacks, vulnerabilities, risk, security goals, and security controls. Critical assets and system vulnerabilities can change at runtime, and this may affect related security concerns. For example, if new assets need to be protected or assets become more valuable, their related security goals may become more critical. Risk of attacks can also increase if new vulnerabilities are discovered in a system. In both cases, the security controls that are currently present in the system may no longer be effective. To address this, we have proposed adaptive security [4], which aims to continue to protect valuable assets from harm, even when security concerns change dynamically. To prevent possible attacks, adaptive security adjusts active security controls.

SIGSOFT'12/FSE-20, November 11–16, 2012, Cary, North Carolina, USA.

Access control systems [3, 1] can benefit from adaptive security. Unfortunately, beyond the early stages of development of such systems, *what* should be protected (assets) and *why* particular access controls are applied (security goals and requirements) are often ignored. This may introduce difficulties to configure security controls (access control policies) according to what needs to be protected, and adapt existing policies when assets or vulnerabilities change.

This paper presents SecuriTAS[1], a novel tool to support adaptive security from requirements modeling to system execution. SecuriTAS can analyze the impact of changes in assets and vulnerabilities on security concerns, and identify an appropriate set of security controls necessary to protect the system. Potential users are software designers and system adminstrators who have to configure the system and its security controls. SecuriTAS is demonstrated through a set of access control scenarios. The rest of the paper illustrates the steps necessary to engineer adaptive security, describes the functionalities provided by SecuriTAS, and provides a demonstration our tool.

2. ENGINEERING ADAPTIVE SECURITY

We suggest engineering adaptive security in three steps [4]. *a*) *Modeling the security concerns together with the system requirements.* We use a KAOS [5] *goal model* to represent functional and non-functional requirements and a *threat model* to represent threats and attacks. It explicitly represents security goals and associated vulnerabilities. In particular, security goals can be associated with security controls, which can mitigate modeled vulnerabilities. An *asset model* represents assets and their relationships. Assets are linked to the security goals and associated threats.

b) *Configuring adaptive security.* We generate a fuzzy causal network [2] (FCN) from our asset, threat, and goal models. Each node of the FCN is associated with a security concern and has a specific semantics. For example, a node associated with an asset represents its value. The links of the network identify positive and negative causal relationships among security concerns. For example, a link between a security control and a vulnerability indicates how the security control reduces the probability of that vulnerability to occur. The FCN is used at runtime to analyze the security risk and evaluate the utility of all possible configurations of security controls. Furthermore, we identify monitored assets and contextual factors, which may change existing system vulnerabilities. We also map the security controls specified

[1]The tool and the installation guide can be download it at `http://code.google.com/p/securitas/downloads/list`.

in the goal model to the concrete security functions that are implemented to protect the system.

c) Applying Adaptive Security at Runtime. We instrument the activities of a MAPE (Monitor-Analysis-Plan-Execute) loop, by using the configuration generated in the previous step. Monitoring detects the changes that take place in assets and contextual factors that may affect vulnerabilities. Suitable probes are provided by the system to notify the monitor when changes in asset and contextual factors take place. Analysis updates the value of each node of the FCN, by aggregating the contributions of the incoming links. In this way, the security risk and the utility of all configurations of security controls can be re-estimated. Planning selects the configuration of security controls with the best utility. Execution applies the security functions associated with the best configuration of security controls on the system.

3. A WALK THROUGH SECURITAS

SecuriTAS is composed of the **Graphical Modeler** and the **Adaptation Manager**. The Graphical Modeler (see Figure 1) is a visual editor to create the asset, goal, and threat models (*Step a*). It is implemented as an Eclipse plug-in, by using EMF (Eclipse Modeling Framework) and GMF (Graphical Modeling Framework). The Graphical Modeler provides three views. The Package Explorer (1) shows the projects in the workbench and the models contained in each project. The Editor (2) shows the diagram of a selected model and allows us to edit it, by adding/removing elements. The Palette (3) provides the elements and connections that can be dragged and dropped in the model shown by the editor. The Properties view (4) allows us to view and edit the properties of the element/connection selected in the Editor view. After the asset, goal, and threat models are completed, the Graphical Modeler allows us to generate the corresponding FCN and map each security control to the corresponding security function implemented in the system (*Step b*). Monitored assets and contextual factors are also identified from the designed models. This information is sent to the Adaptation Manager to configure its activities at runtime.

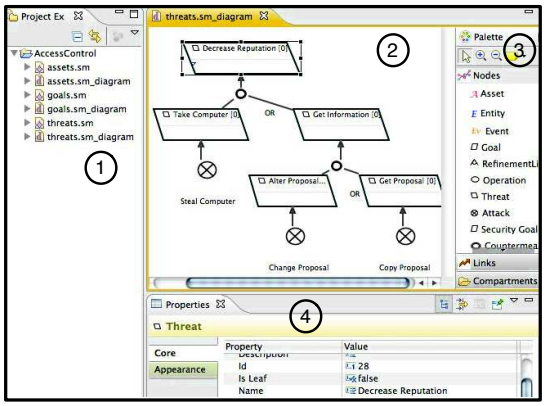

Figure 1: The Graphical Modeler.

The Adaptation Manager implements the activities of the MAPE loop to apply adaptive security at runtime (*Step c*). It exposes a monitoring interface to be notified when assets or context conditions, which affect vulnerabilities, change. It updates the nodes of the FCN, identifies the configuration of security controls with the best utility, and applies the corresponding security functions on the system. The Adaptation Manager is implemented as a web application. It also provides a dashboard (Figure 2) to load and visualize the FCN, update the values of assets and context conditions that may affect vulnerabilities, and visualize the current state of the system (1). If this last functionality is selected, the Input view (2) shows the current assets (and their values) and the context conditions that may affect vulnerabilities. While, the Output view (3) describes the security functions that are currently applied on the system.

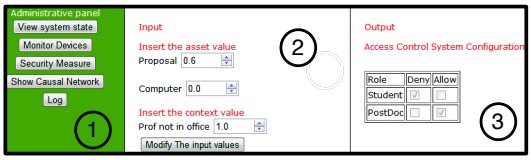

Figure 2: The Adaptation Manager dashboard.

4. DEMONSTRATION

The demonstration scenarios regulate the access to an office for a set of employees. The office contains valuable assets (e.g., patent proposal, computer), while employees can have one of these roles: professor, postdoc, or student. In this scenario, SecuriTAS is used to grant the access to students and postdocs according to the availability of valuable assets and the persons that are currently present in the office. The demonstration of our tool is organized in three parts. First, we model the security concerns and the requirements of the system through the Graphical Modeler. In particular, we create the asset, threat, and goal models of the dynamic access control scenario. Second, we use the Graphical Modeler to configure adaptive security activities. In particular, we generate the FCN from the asset, threat, and goal models, and associate the security controls with the security functions implemented in the system. Third, we show how adaptive security is applied at runtime, by simulating a dynamic access control scenario.

4.1 Modeling assets, threats, and goals

The asset model represents assets and their relationships. Figure 3 depicts the asset model associated with our dynamic access control scenario. As assets, we consider the (patent) `Proposal`, which has a high value (0.6), the `Computer`, which has no value (0.0), since it is not initially located in the office, and the `Office` itself, which has a no initial value (0.0). The value of the `Office` can also increase, depending on whether other valuable assets are located in it (see association `isContained` between assets `Proposal`/`Computer` and `Office`).

The threat model represents the motivations of the attackers to harm the assets in the system boundary. The criticality of a threat directly depends on the value of the asset to be harmed. Threats can be hierarchically decomposed into sub-threats, until they are operationalized as attacks. The main motivation of an attacker is to descrease the reputation of the professor. This threat can be achieved by stealing the computer (attack `Steal Computer`) or by illegitimately modifying or copying the proposal (through attacks `Change Proposal` and `Copy Proposal`).

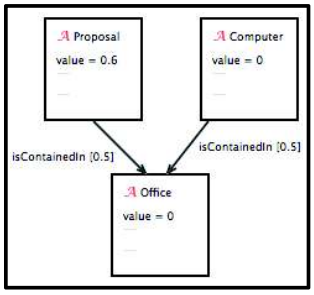

Figure 3: Asset Model.

The goal model represents the main objectives of the system. A goal can be decomposed into sub-goals, until it can be directly associated with a functional/non-functional requirement. Functional requirements can be decomposed into system operations. Vulnerabilities are explicitly represented in the goal model and may be brought by system operations. Vulnerabilities can also increase the probability of success of attacks. The main objective of the system is to support research. In particular, proposals must be completed and students must be supervised. To supervise students, the professor, the postdoc, and the student himself must have access to the office (goals `Stud/Prof/Postdoc Enters Office`). All these goals are achieved through operation `Enter`. In case the professor is not in the office (see contextual factor `Professor not in office` in Figure 4), operation `Enter` brings a vulnerability, since unautorized employees can access to the office. This vulnerability enables attacks `Steal Computer`, `Change Proposal`, and `Copy Proposal`.

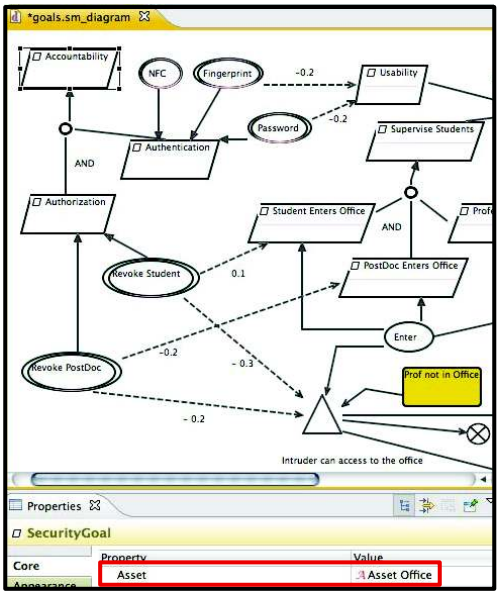

Figure 4: Partial view of the goal model, representing the security requirements.

Security goals represent the CIAA (confidentiality, integrity, availability and accountability) properties. The priority (criticality) of a security goal may depend on the value of the asset that needs to be protected. Security goals have a hierarchical structure and can be gradually decomposed into security controls. Figure 4 depicts a simplified view of our security goals. We want to guarantee Accountability of the `Office` (see Properties View in Figure 4). To this aim, `Authorization` and `Authentication` should be provided to regulate the access to the office. Authentication can be performed in three different ways (security controls): `Password`, `Fingerprint`, and `RFID` (smart card). Authorization is supported by revoking the permission to enter to the student and/or the postdoc (see `Revoke Student` and `Revoke Postdoc`). Security controls are linked to the vulnerabilities they try to mitigate. A weight can be assigned to this link depending, on the effectiveness of the countermeasure. For example, `Revoke Postdoc` and `Revoke Student` mitigate vulnerability `Access of unauthorized employees` of a factor 0.2 and 0.3, respectively. The security controls applied to satisfy security goals can have a negative impact on other goals. For example, `Revoke Student` and `Revoke PostDoc` have a negative impact on goals `Stud Enter Office` and `PostDoc Enter Office`, respectively. Security controls `Password` and `Fingerprint` also have a negative impact on the usability of the system. Note that configuring the relationships between security concerns is an hard task that relies on the expertise of the system designer/administrator.

4.2 Configuring Adaptive Security Activities

The Graphical Modeler allows us to generate the FCN and associate the security controls represented in the goal model with the security functions that can be applied on the system. To generate the FCN, in the Graphical Modeler, we right-click on the project (`AccessControl`) that contains the corresponding asset, threat, and goal models, and select option "Generate FCN". To generate the mappings between the security controls and the security functions, we right-click on project `Access Control` and select option "Generate Sec Controls". Then, a pop-up window appears and allows us to select one of the implemented security function for each security control.

4.3 Applying Adaptive Security at Runtime

We use the AET62 NFC Reader to monitor who accesses to/exits from the office and which assets are moved to/removed from the office. If an employee swipes his/her smart card on the NFC reader and he/she is not in the office, he/she is allowed to enter only in case he/she has the permission to access to the office. We also use smart cards to tag assets in the system. For instance, in case an asset is outside the office and we swipe its card, it means that this asset is moved to the office. In our scenario, we tagged the computer with a smart card to monitor when it is put in the office. We also developed an application (NFCManager) to control the status of the NFC reader and notify the Adaptation Manager when an asset (computer) is added to the office or contextual conditions (presence of the professor in the office) change. The NFCManager also exposes a graphical interface to show what are the available assets in the office (and their value) and which employees tried to access to the office.

The NFC reader also supports smart card authentication. When one of the employee swipes his/her card, the NFCManager checks on the DB whether the employee's role has the permission to access to the office. The door is opened

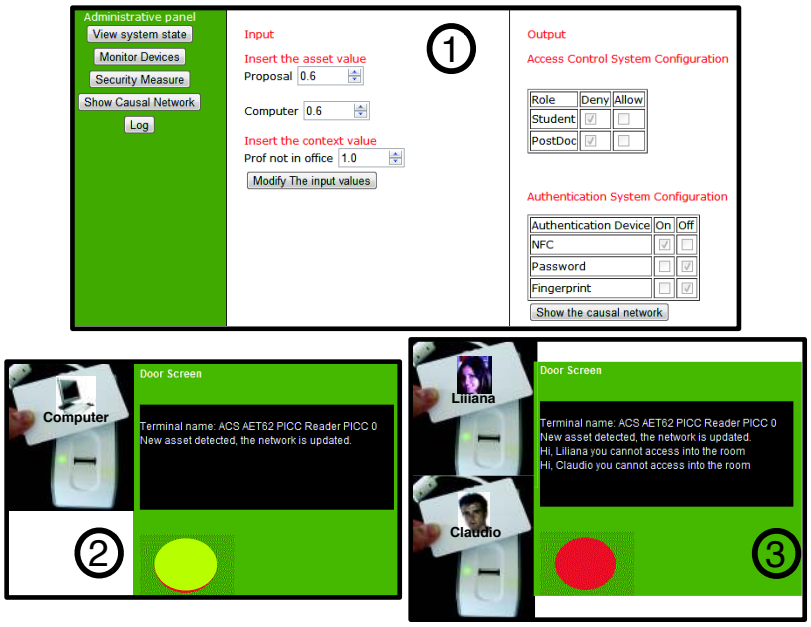

Figure 5: Scenario 2.

only in case the employee's role has the permission to enter. To implement the smart card authorization we store a record of each employee that tracks his/her role(s) and tag. We also store another record for each role to track the corresponding permissions. This way, every time it is necessary to change the permissions associated with a role, the corresponding record in the DB will be updated.

The dynamic access control example is composed of different scenarios. In all scenarios we assume to have three employees (Claudio, Liliana, and Bashar) who have respectively the role of student, postdoc and professor. The initial causal network received from the Graphical Modeler represents the initial situation. In the first scenario, the office only contains the patent proposal (value 0.6) and Bashar is not in the office. After we select function "Modify Input Values", the Adaptation Manager recomputes the best configuration of security controls that should be applied on the system. In this case, it select a single factor authentication (via smart card) and grants the permission to access to the office only to postdocs and not to professors. In this case, when Liliana tries to enter in the office, the access is granted. Instead, when Claudio tries to access to the office, the access control system does not allow him to enter.

In the second scenario the Computer is put in the office, since its corresponding smart card is swiped on the NFC reader (see Figure 5(2)). The Adaptation Manager is notified of this change and updates the state of the system, by setting the value of the Computer to 0.6 (see Input view in Figure 5(1)). After we select function "Modify Input Values", the Adaptation Manager recomputes the best configuration of security controls that should be applied on the system. As Bashar is still not in the office, the risk that the computer is stolen increases and more restrictive security controls should be applied. In this case, the Adaptation Manager selects a single factor authentication (via smart card) and revokes the permission to access to the office to both postdocs and students (see Output view in Figure 5(1)). In this case, when both Liliana and Claudio try to access to the office, the access control system does not allow them to enter (see Figure 5(3)).

In the third scenario, Bashar accesses to the office by swiping his card. The Adaptation Manager is notified of this change and updates the state of the system, by setting the value of context condition "Professor not in the office" to 0.0. After we select function "Modify Input Values", the Adaptation Manager recomputes the best configuration of security controls that should be applied on the system. Since this change reduces the probability of the vulnerabilities to be exploited by potential attacks, the Adaptation Manager selects a single factor authentication (via smart card) and grants the permission to access to the office to both postdocs and students. In this case, when both Liliana and Claudio try to access to the office, the access control system allows them to enter.

5. REFERENCES

[1] F. Cuppens and N. Cuppens-Boulahia. Modeling Contextual Security Policies. *Int. J. Inf. Sec.*, 7(4):285–305, 2008.

[2] R. Howard and J. Matheson. Influence diagrams. *Decision Analysis*, 2(3):127–143, 2005.

[3] M. J. Covington et al. Securing Context-Aware Applications Using Environment Roles. In *SACMAT 2001*, pages 10–20, 2001.

[4] M. Salehie, L. Pasquale, I. Omoronyia, R. Ali, and B. Nuseibeh. Requirements-driven Adaptive Security: Protecting Variable Assets at Runtime. In *RE 2012*, page (to appear), 2012.

[5] A. van Lamsweerde. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. John Wiley, 2009.