Towards a Security Domain Model for Embedded Systems

Simin Nadjm-Tehrani and Maria Vasilevskaya
Dept. of Computer and Information Science
Linköping University, Linköping, Sweden
Email: [simin.nadjm-tehrani, maria.vasilevskaya]@liu.se

I. Introduction

Embedded devices are increasingly involved in applications that store, access, and manipulate sensitive information. This creates a need for protecting such devices from security threats. However, resource-constrained nature of these devices does not allow engineers to apply conventional security mechanisms in a straight forward manner. For instance, Ravi et al. [1] identify resource-oriented gaps which are related to incorporating security solutions in embedded devices; in particular, the so-called battery and processing power gaps. To address these issues we propose to shift security considerations to earlier development stages, and support the embedded engineer in incorporating security solutions.

The embedded systems design process is already quite challenging since an engineer has to consider both hardware and software elements. In addition, it is not easy to understand and apply security solutions' features for system engineers. Here we see a role to be played by domain specific modelling (DSM). DSM is able to capture heterogeneous views of a system providing specific languages for each of them. In our work, we consider the two domains of embedded systems and security. We aim to provide an appropriate view of a final system model that supports cooperation between the two identified domains, while leaving them independent from each other.

We use UML (Unified Modeling Language), since it is widely used as a base for building domain-specific languages. Another motivation is that there already exists an extension of UML, i.e. MARTE (Modeling and Analysis of Real-Time and Embedded systems) [2], that allows capturing the resource-oriented constraints as UML models.

Two other approaches for incorporating security concerns at the design level are aspect-oriented modelling [3] and security patterns [4]. In these approaches a security solution, represented as an aspect or pattern, is integrated with a functional model of a system. While these are demonstrated to work for software-intensive systems, our approach adds the hardware dependent resource perspective. Thus, the security focus does not neglect the resource constraints in embedded systems. Rather, the security domain model includes a resource element that helps to select reasonable security solutions, for example, to deal with energy as a prime concern in mobile applications.

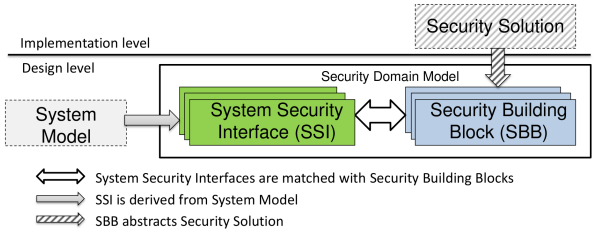

Figure 1. Security domain model components (right) and its relation with existing/evolving system model (left) and a security solution implementation (top).

II. Security Domain Model

The proposed security domain model consists of two components (see Figure 1): (1) System Security Interface (SSI), and (2) Security Building Block (SBB). These are fundamental elements derived within the engineering processes for both system and security solution designs. They will be connected together using matching algorithms that identify the right level of security at the right resource picture available for implementing those security solutions.

Each part of the security domain model abstracts more detailed elements. The SSI is an abstraction of the system design model for communicating security needs and resource availability, and the SBB is an abstraction of the implementation for a security mechanism. Thus, our security domain model is primarily a means of communication between the system/design engineer and a security expert who knows the security building blocks well, and can characterise their needs and capabilities. We proceed to describe the structure of the SSI and SBB.

The SSI component describes the security related parts derived when constructing a system model. It comprises three elements (see Figure 2): *asset & actions* (i.e. an object of a system that should be protected and the actions applied on/within this asset), *required security properties* (i.e. the nature of the protection required by the asset), and *provided resources* (i.e. the system resources available and allocated by a system engineer for security purposes).

The SBB component represents a developed security solution to be integrated into an embedded device. It is created by a security engineer and contains three elements: *security mechanism* (i.e. a model of a security solution that exists or is to be implemented), *provided security properties* (i.e. the nature of the provided security), and *required resources* (i.e. the resources which are needed for a security mechanism to function on a specific platform).

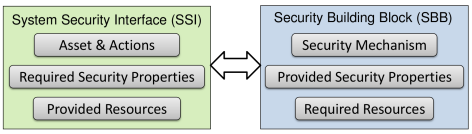

Figure 2. Conceptual view of the security domain model components.

The applicability of this concept has been analysed on three industrial use cases available in the SecFutur project [5]. In this short paper we briefly outline one such instantiation, i.e. a mesh communication scenario.

III. Description and instantiation

In what follows we provide an overview of the mesh communication scenario, followed by description of the six box model and its illustration with the mesh communication system.

The considered system is intended for on-demand provision of communication services within a crisis-related scenario, where different actors could be involved, e.g., fire-fighters and police. These actors are equipped with heterogeneous client devices (e.g., PDA, sensors). A mesh network is built from mobile Mesh Nodes (MNs). The interconnection between MNs can be based on wireless (or even wired) technologies. The functionality of each MN is to build a network, receive and forward messages from peers (MNs and client devices), and store temporary media files from actors. In this paper we consider the requirement of preserving trustworthiness of a mesh network while a new node joins the network.

Now, we describe the system security interface (SSI) component.

Assets& Actions. These are specific to the system and extracted from the system model within a security engineering process. Identification of an asset typically takes place in the risk analysis process, and indicates the need for protection. We distinguish two types of assets, i.e. *passive* and *active*. Passive assets (e.g., data) can be exposed to actions by the system elements. Active assets (e.g., network interface controllers) can themselves perform some actions. Asset and action pairs are derived from the structural, architectural and behaviour descriptions in the system model available in UML.

In case of the mesh network communication the selected (active) asset is the MN platform. This asset includes hardware, system software and services such as communication capabilities, and application-related services as a part of its structural description. A MN platform participates in establishing a mesh network using the join and accept actions as a part of its behaviour description.

Required security properties. These are used to specify the nature of the required protection for the identified assets. In our case it is absence of alterations of the MN platform considered as an asset. This encompasses evidence of hardware tamper resistance and software integrity.

Provided resources. In order to enforce the desired properties the system engineer might be prepared to allocate certain resources for security purposes. In case of the mesh network the system engineer could envisage the provision of a computing unit and some communication bandwidth.

Next we proceed with the security building block (SBB) component.

Security mechanisms. These model generic components created by security experts as a security solution to be integrated. They may enforce security properties in many different ways. Each way may consume different types and amounts of system resources. In the mesh scenario a remote attestation mechanism can be deployed for the MN platform.

Provided security properties. Each mechanism has to show which properties it provides. A given remote attestation mechanism may provide evidence of software and hardware integrity on a target platform.

Required resources. Any security mechanism will claim some system resources in order to function. A remote attestation mechanism, for instance, needs memory to store the code, computing resources to generate certificates, and communication bandwidth to exchange the required packets. All computation and communication, of course, use up energy resources which deplete the battery on a hand-held device.

IV. Conclusions and Further work

In this short paper we have proposed the security domain model to deal with security concerns at the early design stages of embedded systems development. The embedded nature of the applications is captured by the provided and required resources, while the security aspect is encapsulated in required and provided security properties.

As a part of our ongoing work we will refine the languages for each of the components and define a formal semantics for these languages to support reasoning within our engineering process.

References

[1] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady, "Security in embedded systems: Design challenges," *ACM Transactions on Embedded Computing Systems*, vol. 3, no. 3, pp. 461–491, August 2004.

[2] *Modeling and Analysis of Real-Time and Embedded systems, MARTE*, OMG Std. [Online]. Available: http://www.omgmarte.org/

[3] G. Georg, I. Ray, K. Anastasakis, B. Bordbar, M. Toahchoodee, and S. H. Houmb, "An aspect-oriented methodology for designing secure applications," *Journal of Information and Software Technology*, vol. 51, no. 5, pp. 846–864, 2009.

[4] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security Patterns: Integrating Security and Systems Engineering*, 2005.

[5] SecFutur EU project: Design of Secure and energy-efficient embedded systems for Future internet applications. http://www.secfutur.eu/.