

Design for securability – Applying engineering principles to the design of security architectures

Amund Hunstad

Phone number: + 46 13 37 81 18

Fax: + 46 13 37 85 50

Email: amund@foi.se

Jonas Hallberg

Phone number: + 46 13 37 85 14

Email: jonhal@foi.se

Swedish Defence Research Agency

DEPARTMENT OF SYSTEMS ANALYSIS AND IT-SECURITY

P.O. Box 1165, SE-581 11 LINKÖPING, SWEDEN

POC: Amund Hunstad

Design for securability is an approach to obtain distributed information systems possible to secure during operation. To achieve this, three steps have to be supported in the design of these distributed systems. Firstly, the interactions and relations between the system and its environment have to be captured. Secondly, a set of security requirements on the system has to be formulated. Thirdly, the set of requirements has to be implemented in the system. This position paper focus on the third step which requires system models and design methods and tools.

Introduction

There are many problems inherent with the launch and patch approach to security. Also the more careful approach of using red teams/tiger teams rather than waiting for things to break, has its limitations and drawbacks (Gula, 1999), (Schudel & Wood, 2000). The use of vastly distributed information systems will drastically increase the risks associated with security breaches (Schneier, 2001). Thus, the need to get it right from the start should be a cornerstone in all future system development. Unfortunately, systems will always contain flaws. Thus, risk management becomes a necessity and “to get it right from the start” means building systems able to handle failing components and unexpected events. This creates new demands on the ability to comprehend vastly distributed systems and to understand the effects of events in these systems.

Design for securability, our approach to applying engineering principles to system security design, can be described as:

- integrating knowledge of the system and its environment and
- being based on
 - the importance of mutual trust between system owner and operators and in system and organization,
 - requirements engineering,
 - systems modeling,
 - methods and tools for design support and finally
 - how risk management may be eased by such a design approach.

Engineering principles and security architectures

Realizing the fact that no system can be designed to be secure, but can include the necessary prerequisites to be secured during operation; the aim is *design for securability*. The characteristics aspired of such a system vary from case to case and is influenced by factors such as time, cost, throughput and risks. Thus, requirements engineering, as described in (Sommerville & Sawyer, 1997), becomes an important tool for the security engineer. There are other reasons to use requirements engineering in the system development process. The

main benefit is the ability to decide which system functions are required and thus, decrease the total number of functions and number of flaws in the system.

Firstly, the interactions and relations between the system and its environment have to be captured. This usually results in a complex structure, as illustrated by Figure 1, in which trust is a central component. Trust relies on operations performed by operators, by information systems and on operations performed within an organizational context.

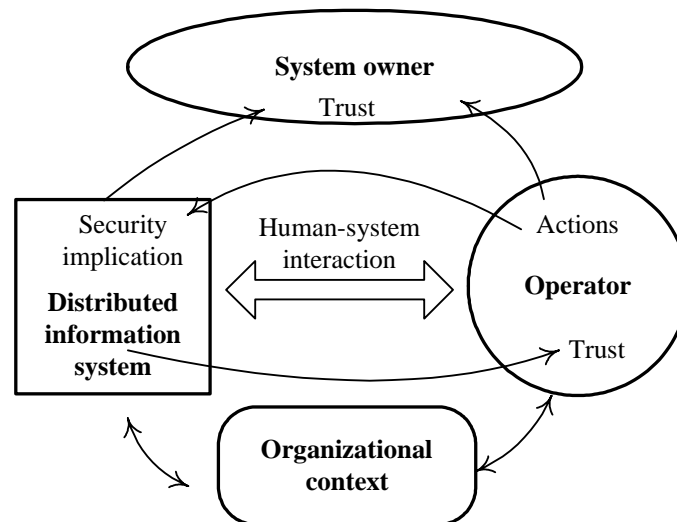


Figure 1: The relations between a system, the organization, operators and the system owner.

At the top level, the system owner's trust in the system relies on the performance of the information system and on different actions taken by operators. The operator's trust is more directly related to the performance of the information system and especially the way the system's performance is experienced through the human-system interaction. Actions taken by an operator has security implications within the information system and the way this makes the system perform influences the operator's trust or possibly lack of trust.

The actions taken by the operator and the functions of the information system is also set within an organizational context, which also has an impact on trust. As an example, a policy regarding backup of data is worthless, if you have no routines to implement the policy.

Secondly, a set of requirements on the system has to be formulated. From this set security-relevant requirements can be extracted, as illustrated by Figure 2. Starting with a textual description of the system requirements, the general system requirements are refined into statements concerning security. These are thereafter validated and checked for consistency. Through this process of requirements engineering, security related issues are integrated at an

early stage of the system development. This is in contrast to what often happens with add-on security at a late stage, perhaps even after the rest of the system development is over.

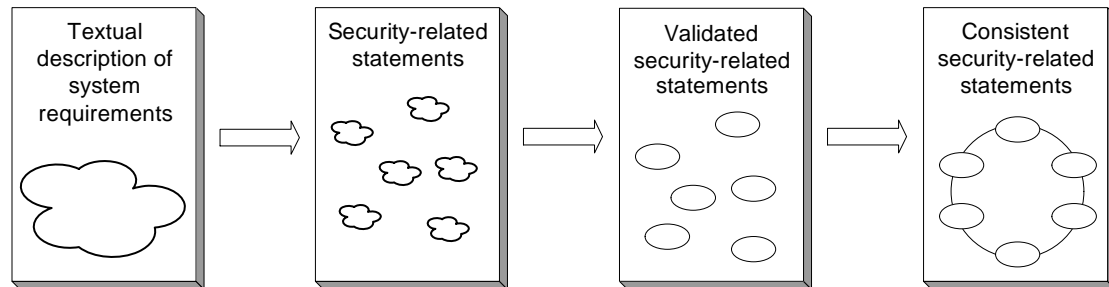


Figure 2: The process of formulating a set of consistent security related statements from a textual description of system requirements.

Thirdly, the set of requirements has to be implemented in the system. This is a complex process that has to extend throughout the lifetime of the system. Ideally, there would be a well-formulated process extending from the set of requirements to the implemented system, and also facilitating and enhancing risk management of the implemented system. However, this demands, on top of the task to design an efficient security architecture, the solution of all traditional system development issues. Therefore, at this point, the formulation of a framework for design and evaluation of security architectures, based on a system implemented at some level of abstraction, would be a great step forward. Such a framework has to be based on the ability to efficiently model the studied systems. Thus, systems modeling and the design framework are discussed in the following section.

Systems modeling and design framework

The designers' ability to model distributed information systems is essential for the comprehension and assessment of the corresponding systems and design decisions. Furthermore, design tools have to support such an ability. Thus, an efficient modeling technique is a prerequisite for the design of distributed information systems. The purpose of system models is to create a notion where system requirements and characteristics meet. System models can be built before the system actually has been implemented (design models) or for a present system (analysis models). As a first step, these system models will enable designers to reason about the modeled systems even before any design methods have been implemented.

To efficiently model systems, system characteristics have to be extracted both from high-level descriptions of the system and from models of system components, as illustrated by Figure 3. Moreover, the models have to be able to capture the system requirements. This is essential in order to be able to verify, validate, or assess system requirements and alternative implementations.

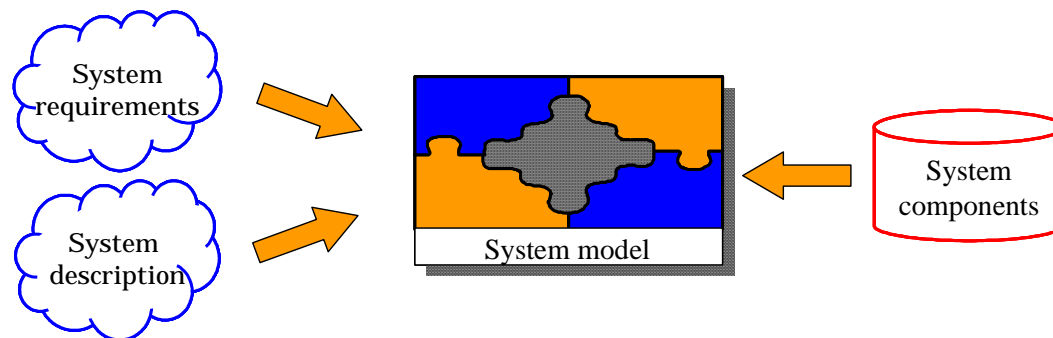


Figure 3: A system model has to capture both the requirements put on a system and its characteristics (from high-level descriptions and component models).

To build the system models a modeling technique is required. An adequate modeling technique has to fulfill design process requirements and enable the capturing of security-relevant system characteristics. Thus, the formulation of an adequate modeling technique requires knowledge of the security-relevant system characteristics that have to be captured in order to efficiently design a security architecture. Consequently, a set of security-relevant system characteristics is needed for two important tasks: to assess the security of a system and to formulate an appropriate modeling technique. It is important to realize that this results in a strong influence on the mechanisms to be included in a modeling technique, e.g. mechanisms to capture system structure or data flow. Still, a modeling technique can hopefully be formulated in such a way that the demands of a dynamic set of security-relevant system characteristics will not require redesign of the modeling technique. A conclusion is that the modeling technique has to be flexible and expressive.

To be able to enumerate important security-relevant system characteristics, a tree structure with the three roots confidentiality, integrity, and availability (CIA) can be used. The tree structure is extended by detecting which security characteristics are descendants of C, I, and A respectively, as illustrated by Figure 4. An effort along these lines, resulting in a structure with 55 distinct characteristics, is presented in (Stjerneby, 2002). The quest for a set of security-relevant system characteristics enabling exact assessments of the security level of a system is indeed a difficult task, as discussed at the ISSRR workshop 2001 (ACSA, 2002). Still, a set as detailed as possible will support the formulation of adequate modeling

techniques and the models created with this modeling technique will enhance the awareness and assessment of security-relevant issues.

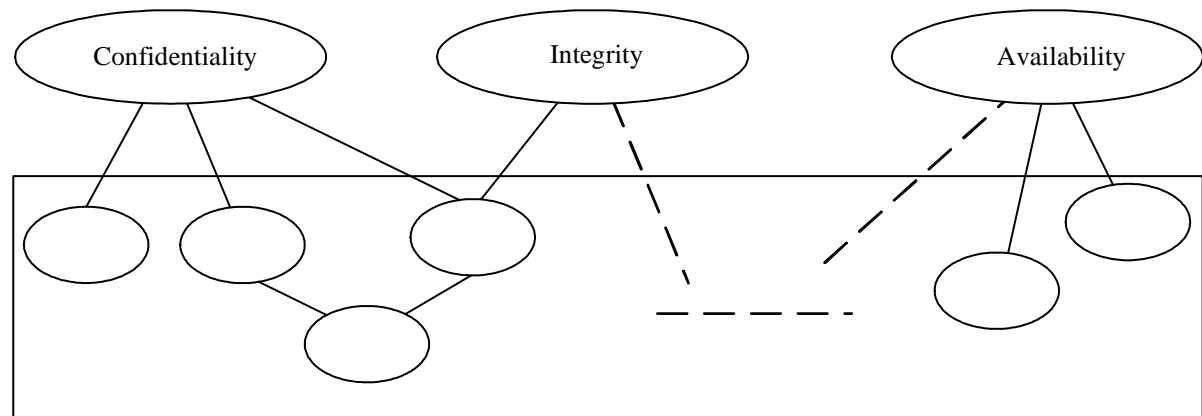


Figure 4: Security characteristics in a tree structure.

Figure 5 illustrates the concept of a framework for assessment and modification of system descriptions. System requirements, high-level descriptions, and component descriptions are used to build system models. The system models are analyzed and modified using design methods and tools. Finally, the result is fed back to the system descriptions and requirements. The number of ways this process can be performed with a mix of manual work and automatic tools is infinite. However, even assuming all analysis, modifications, and feedback to be manual, a systematic design process facilitated by system models would enable the security engineer to validate the requirements specified for the system.

To build comprehensible models capturing all the necessary information, an expressive modeling technique supporting hierarchies (abstraction) and several different views of a system is required. Using a standardized modeling language has several advantages, e.g. utilization of all the work put into the formulation of the language, the possibility of designers already being familiar with the language, and the possibility to use tools developed according to the standard. Considering the requirements on the modeling technique and the advantages of using a standardized language, the unified modeling language (UML) is a strong candidate as a base for the aspired modeling technique. UML is biased towards object oriented software development. However, it contains diagrams for modeling of the structure of a system, although these mechanisms are rarely used (Akehurst & Waters, 1999). The diversity of UML opens the possibility to create modeling techniques for a number of models and model views supporting the use of engineering principles through the whole (security architecture) design process, enabling design for securability.

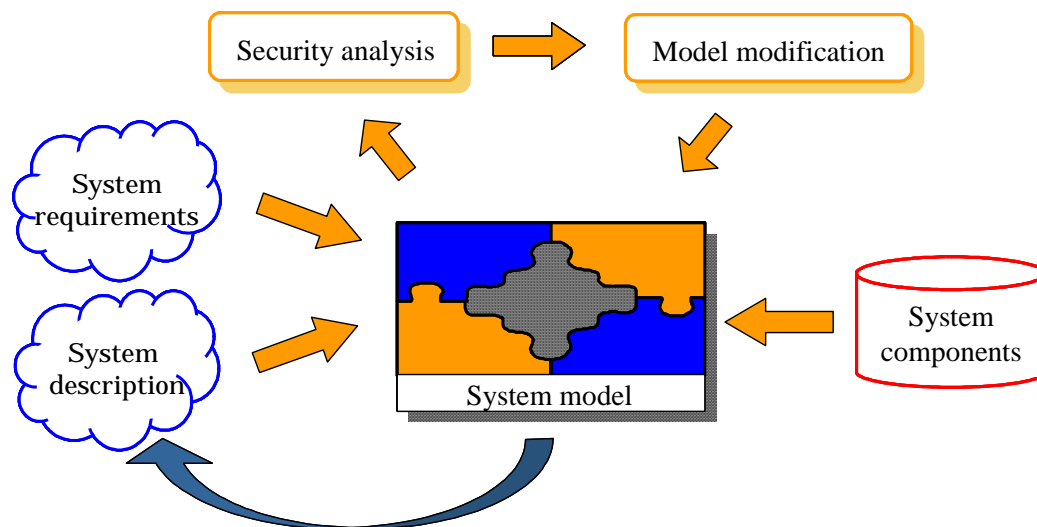


Figure 5: A framework for analysis and modification of system models.

Conclusions

There is a need to formulate methods covering the chain of development steps from mapping the structure of a system and its environment, via the requirements engineering process, to the design of security architectures. Systematic design of security architectures requires powerful modeling techniques and design methods and tools. The process is called design for security since a system cannot be designed secure.

Even though the feasibility of creating a set of security-relevant system characteristics is an open question, we believe that system models enabling designers and design tools to assess and modify current and future systems are viable. The formulation of the corresponding modeling techniques is greatly improved by the presence of sets of security-relevant system characteristics.

Reference

ACSA (2002). *Proc. Workshop on Information Security System Scoring and Ranking*. Applied Computer Security Associates.

Akehurst, D. & Waters, A. (1999). *UML specification of distributed system environments*. Technical Report : 18-99. Computing Laboratory, University of Kent at Canterbury. UK.

Gula, R. (1999). *Broadening the scope of penetration-testing techniques - The Top 14 Things Your Ethical Hackers-for-Hire Didn't Test.*,
<http://www.enterasys.com/products/whitepapers/security/9012542.pdf>

Schneier, B. (2000). *Secrets & Lies – Digital Security in a Networked World*, John Wiley & Sons.

[Schudel,Wood00] G. Schudel and B. Wood, "Adversary Work Factor as a Metric for Information Assurance", *Proceedings of the New Security Paradigms Workshop*, Cork, Ireland, Sep. 18-22, 2000.

Sommerville, I. & Sawyer, P. (1997). *Requirements engineering: a good practice guide*. Chichester: Wiley.

Stjerneby, A. (2002). *Identification of security relevant characteristics in distributed information systems*. Master's Thesis. Linköping University.