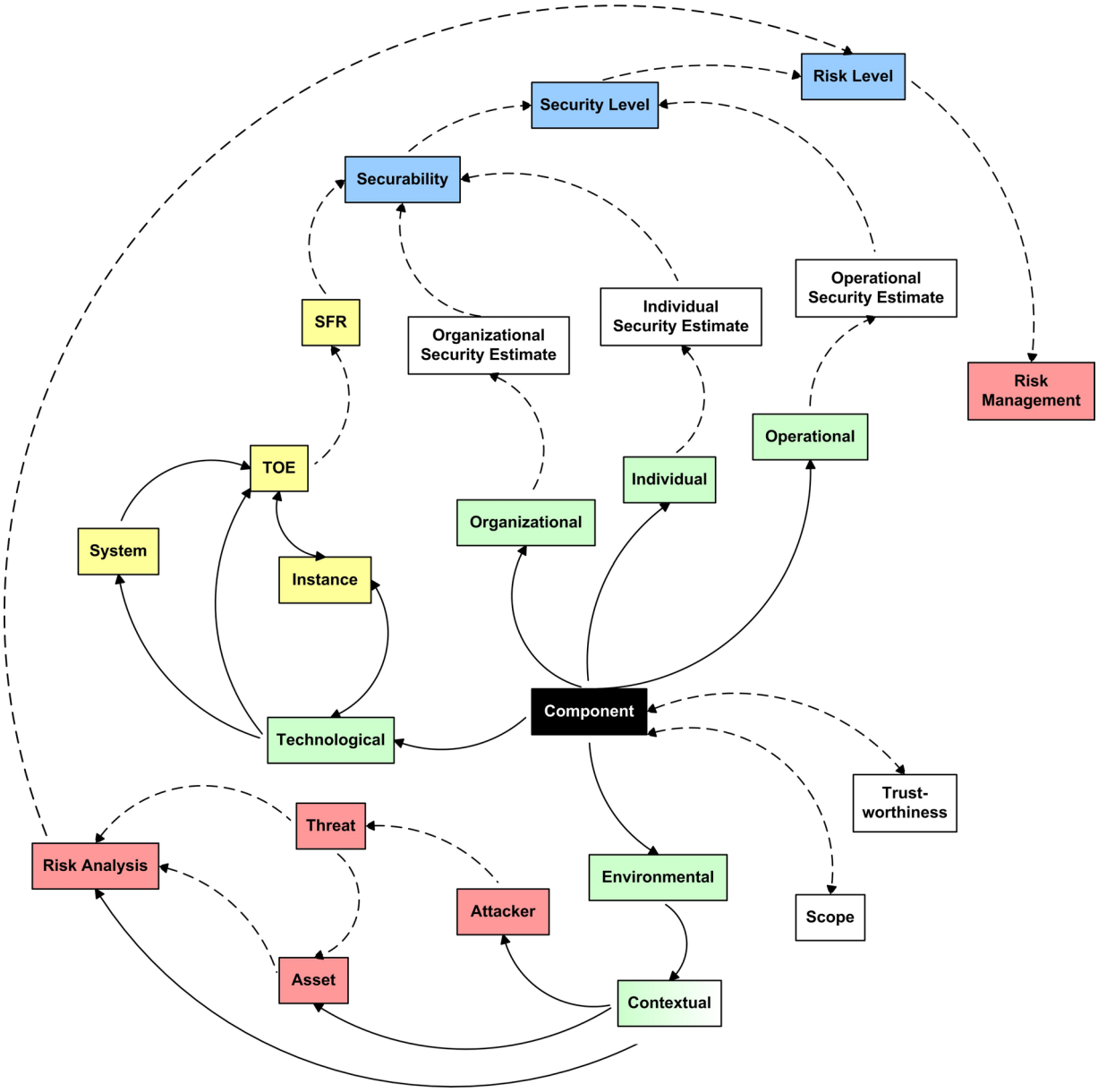
FOI
SWEDISH DEFENCE
RESEARCH AGENCY

FOI-R--1042--SE
November 2003
ISSN 1650-1942
Scientific report

Richard Andersson, Amund Hunstad, Jonas Hallberg

Evaluation of the Security of Components in Distributed Information Systems



Command and Control Systems
SE-581 11 Linköping

SWEDISH DEFENCE RESEARCH AGENCY

Command and Control Systems
P.O. Box 1165
SE-581 11 Linköping

FOI-R--1042--SE

November 2003

ISSN 1650-1942

Scientific report

Richard Andersson, Amund Hunstad, Jonas Hallberg

Evaluation of the Security of Components in Distributed Information Systems

Issuing organization	Report number, ISRN	Report type
FOI – Swedish Defence Research Agency Command and Control Systems P.O. Box 1165 SE-581 11 Linköping	FOI-R--1042--SE	Scientific report
	Research area code 4. C4ISR	
	Month year November 2003	**Project no.** E7046
	Customers code 5. Commissioned Research	
	Sub area code 41 C4I	

Author/s (editor/s)	Project manager
Richard Andersson Amund Hunstad Jonas Hallberg	Jonas Hallberg
	Approved by
	Sponsoring agency Swedish Armed Forces
	Scientifically and technically responsible Jonas Hallberg

Report title

Evaluation of the Security of Components in Distributed Information Systems

Abstract (not more than 200 words)

A networked defense, and the networked information society, requires both trustworthy information systems and that users and societies trust these systems. Since the trustworthiness of systems depends on the level of IT security, the ability to assess the IT security ability is vital. Currently, there are no efficient methods for establishing the level of IT security in information systems. This far, most methods are targeted at parts of a system, this is a severe limitation since it rather is the system perspective that should be in focus.

The main results produced by the efforts described in this report are:

- a survey of contemporary security assessment techniques for distributed information systems,
- a set of terms for the field of security assessment,
- a framework structuring the security evaluation process and enabling different aspects of the modeled system to be emphasized,
- a set of security functions needed in systems, based on the security functional requirements of the Common Criteria (CC, 1999), and
- a method using the set of security functions to assess the securability of distributed information systems.

Keywords

IT security, distributed systems, modeling techniques, Common Criteria, security assessment

Further bibliographic information	Language English
ISSN 1650-1942	**Pages** 63 p.
	Price acc. to pricelist

Utgivare Totalförsvarets Forskningsinstitut - FOI Ledningssystem Box 1165 581 11 Linköping	**Rapportnummer, ISRN** FOI-R--1042--SE	**Klassificering** Vetenskaplig rapport
	Forskningsområde 4. Spaning och ledning	
	Månad, år November 2003	**Projektnummer** E7046
	Verksamhetsgren 5. Uppdragsfinansierad verksamhet	
	Delområde 41 Ledning med samband och telekom och IT-system	
Författare/redaktör Richard Andersson Amund Hunstad Jonas Hallberg	**Projektledare** Jonas Hallberg	
	Godkänd av	
	Uppdragsgivare/kundbeteckning Försvarsmakten	
	Tekniskt och/eller vetenskapligt ansvarig Jonas Hallberg	

Rapportens titel (i översättning)

Evaluering av komponenters säkerhet i distribuerade informationssystem

Sammanfattning (högst 200 ord)

Det nätverksbaserade försvaret och det nätverksbaserade informationssamhället kräver både pålitliga informationssystem och att användare och samhällen bedömer dessa system som pålitliga. I och med att pålitligheten i systemen beror på nivån av IT-säkerhet, är förmågan till värdering central. För närvarande finns inga effektiva metoder för att fastställa IT-säkerhetsnivån i informationssystem. Hittills är de flesta metoder inriktade på delar av system, vilket är en allvarlig begränsning, ty systemperspektivet bör vara i fokus.

Huvudresultaten som beskrivs i denna rapport är följande:

- en översikt av aktuella säkerhetsvärderingstekniker för distribuerade informationssystem,
- en uppsättning begrepp rörande området säkerhetsvärdering,
- ett ramverk som strukturerar säkerhetsevalueringsprocessen och möjliggör viktläggning av olika aspekter av modellerade system,
- en uppsättning för systemen nödvändiga säkerhetsfunktioner, vilka baseras på Common Criteria:s ”security functional requirements” och
- en metod som använder uppsättningen av säkerhetsfunktioner för att värdera säkringsbarheten hos distribuerade informationssystem.

Nyckelord

IT-säkerhet, distribuerade system, modelleringstekniker, Common Criteria, säkerhetsvärdering

Övriga bibliografiska uppgifter	**Språk** Engelska
ISSN 1650-1942	**Antal sidor: 63** s.
Distribution enligt missiv	**Pris: Enligt prislista**

Contents

1. Introduction

Network centric warfare (Alberts, Garstka & Stein, 1999), that is a networked defense, and the networked information society at large, require both trustworthy information systems and that users and societies trust these systems. Trust is largely a subjective issue. Users may trust a low-security system, among other possible reasons, because they do not know better or because they think security is irrelevant for the particular system.

IT security is far from the only crucial factor when achieving trustworthiness or trust, but it is *a crucial factor*, and even more so when the trustworthiness of systems is considered. Thus, it becomes an important ability for individuals, organizations, and the society as a whole.

Since the trustworthiness of systems depends on the level of IT security, being able to assess the IT security ability is vital. The concept of a networked society by default results in widely distributed information systems that are difficult to control or even comprehend and, consequently, the complexity of the task to assess the level of security in these systems is overwhelming.

Thus, we observe that:

1. Information systems need to be trustworthy and trusted.
2. IT security is a crucial factor considering trust and, especially, trustworthiness.
3. To be able to judge the trustworthiness of information systems, methods for assessing the level of IT security in these systems are needed.

1.1 Motivation

Currently, there are no efficient methods for establishing the level of IT security in information systems. This is troublesome, but not surprising, since most evaluation methods rely on testing and testing security is immensely difficult (Gula, 1999; Schudel & Wood, 2000). This far, most methods are targeted at specific technical parts of a system. This is a severe limitation since it rather is the system perspective that should be in focus. Consequently, novel methods that can handle both the complexity and the vastness of networked information systems are needed.

1.2 Problem Formulation

The first issue to consider when discussing methods to assess the security of distributed information systems is what to actually measure. This problem is twofold. Firstly, the meaning of security has to be clearly defined for the particular system and situation, in order for the assessment to emphasize the characteristics required of the system. This will enable the specification of a security metric. Secondly, since security cannot be directly measured, other system properties have to be measured. Thereafter, an assessment of the security properties of system components or subsystems can be based on the measured properties. Based on the assessment of system components or subsystems the security of the corresponding system can be evaluated.

Consequently, the main issues that have to be resolved in order to realize system security assessments are:

- what is actually to be measured, that is, security metrics are needed,
- the definition of a set of measurable security-related characteristics,
- the correlations between these characteristics,
- association of the appropriate set of characteristics to system components,
- estimations of the security strength of system components, regarding the associated set of characteristics, and
- the combination of the results for individual system components, possibly including other factors, to system-wide security measures.

This report targets the first five of the above steps, leaving the last step to future work.

The security of a system is affected by a number of factors, such as the implementation of adequate security mechanisms, human system interaction, and organizational aspects. This is essential and considered in chapter 3 and 4. However, regarding actual measurable security-related characteristics, the scope of this work, that is chapter 5, is limited to technical aspects of distributed information systems. The reason for this limitation is to be able to produce viable results in the selected area.

1.3 Contributions

The main results produced by the efforts described in this report are:

- a survey of contemporary security assessment techniques for distributed information systems,
- a set of terms for the field of security assessment,
- a framework structuring the security evaluation process and enabling different aspects of the modeled system to be emphasized,

- a set of security functions needed in systems, based on the security functional requirements of the Common Criteria (CC, 1999), and
- a method using the set of security functions to assess the securability of distributed information systems.

1.4 Report Layout

In chapter 2, background to the work presented in this report is discussed. In chapter 3, the concept of security measurement is discussed and related terminology is introduced. In chapter 4, a framework for security evaluation is introduced. In chapter 5, a method for securability assessment is introduced. The method is based on the security evaluation framework and the Common Criteria security functional requirements. Finally, in chapter 6, conclusions are drawn.

2. Background

In this chapter, the framework of the topic of the paper is set by introducing the concepts of IT security, distributed systems, and design for securability which includes contextual modeling, requirements engineering, and security architecture design. Previous attempts to model systems are discussed and the issue of certification is also introduced with a special focus on Common Criteria.

2.1 IT Security

When defining IT security, there are several different dimensions that can be used. Firstly, IT security has often been divided into required characteristics; the most commonly used are confidentiality, integrity, and availability. Secondly, IT security can be defined using high-level functions such as the triplet protect, detect, and react, which states that adequate IT security requires the ability to protect a system, detect attacks against the system, and react to these attacks. Lately, the ability to survive attacks has been added, thereby forming the quartet protect, detect, react, and survive. Thirdly, IT security can be defined by stating what has to be secured. In this case, IT security can be defined as the part of information security relating to the use of IT. Which of these definitions that is most appropriate depends on the current perspective on IT security.

In this report, IT security is considered to be the process of upholding the confidentiality, integrity, and availability of information and services provided by IT-based information systems. An important aspect of this definition is that IT security is a process, not a product (Schneier, 2000). This is why we introduce *securability* as the main feature of a system supporting the process of securing a system in operation.

2.2 Distributed Systems

Since the purpose of this report is to discuss evaluation of the security level of components in distributed systems, a definition of distributed systems is necessary. Like IT security, distributed systems can be defined in different dimensions, such as physical location, processing power, information, and services. Leslie Lamport intuitively captured the nature of distributed systems with the statement “You know you have a distributed system when the crash of a computer you’ve never heard of stops you from getting any work done”.

Coulouris, Dollimore, & Kindberg (2001) define distributed system as “one in which components located at networked computers communicate and coordinate their actions only by passing messages.” This is a general

definition comprising most contemporary systems, such as, the Internet, intranets, and mobile and ubiquitous computing.

Client-server and peer-to-peer are the two main principal architectures of distributed systems. The *client-server model* has been, and still is, the most used. On the other hand, *peer-to-peer architectures* can be used to build distributed systems without any distinction between clients and servers. Special methods are required to maintain consistency of resources and synchronize events when necessary.

In this report, a *distributed system* is a computing system where the resources are spatially distributed and connected by some kind of network. This is in contrast with a centralized system where the resources are gathered in a single location. The definition of a distributed system clearly includes both hardware and software.

The notion of *distributed information systems* is used to emphasize the distribution of information in the system and the fact that users and organizations are considered to be part of the system. Thus, there is a substantial difference between distributed systems and distributed information systems as used in this text.

An important aspect of distributed systems design is that in practice new systems are not self-contained but rather has to incorporate or share infra-structure and legacy systems. This is probably one reason why most devel-opment is targeted towards the software parts of these systems (Akehurst & Waters, 1999). One approach to handle the complexity and diversity of widely distributed information systems is to employ evolutionary system development.

2.3 Design for Securability

Design for securability was introduced in (Hunstad & Hallberg, 2002). This section contains a revised version of chapter 3 in that report. Regarding the issue of IT-security, and especially the above stated observation that security is a process, not a product, the goal has to be to design systems that can be secured to a required level during operation. Thus, design for securability is needed. Since systems have to be dynamic, there is nothing such as a delimited design phase. On the contrary, systems have to be continuously developed. Design for securability can, as illustrated in Figure 1, be divided into the three main processes of contextual modeling, requirements engineering, and security architecture design (requirements implementation). These processes have to run during the whole life-cycle of systems. Moreover, they are interdependent and need to interact.

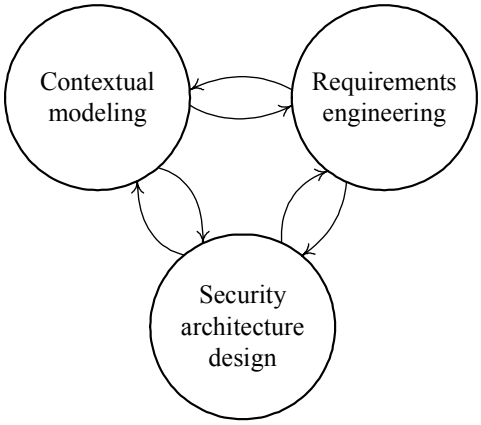


Figure 1: Design for securability can be divided into the three main processes of contextual modeling, requirements engineering, security architecture design and the interactions between these processes.

Contextual modeling

To understand both the needs of a distributed information system and the requirements to be put on it, the interactions and relations between the system and its environment have to be captured. In a pure form, contextual modeling relates to the interactions between the system and its environment. However, in this case, since the concept of distributed information system includes both users and organization, contextual modeling includes the interaction between these entities of the system, that is, organization, users, and distributed system. Trust has a large influence on the interactions between the system and its environment. Trust relies on operations performed by operators, by distributed systems and on operations performed within an organizational context. This results in a complex structure, as illustrated in Figure 2.

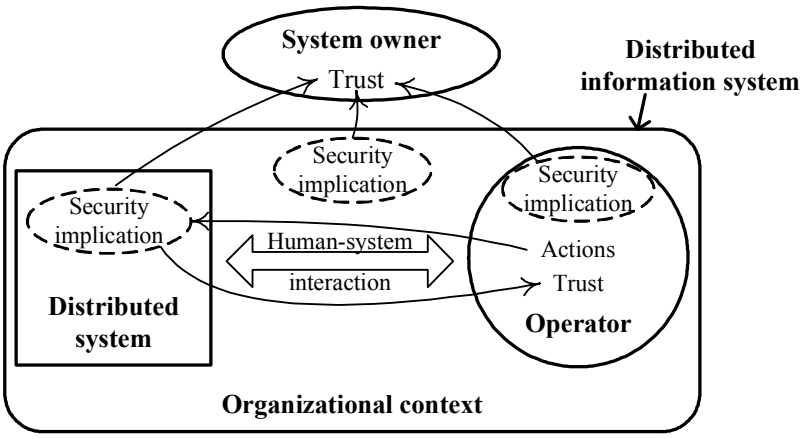


Figure 2: The relations between a distributed system, the organization, operators and the system owner.

At the top level, the system owner's trust in the system relies on the performance of the distributed system and on different actions taken by operators. The operator's trust is more directly related to the performance of the distributed 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 distributed 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 distributed system is also set within an organizational context, which also has an impact on trust. For example, a policy regarding user authentication is worthless, if there are no procedures to implement the policy.

Further studies on trust and information security exist. Of special interest is (Jösang 1996) focusing on differences between trust in humans based on honesty, and trust in systems based on whether they are secure. Both require knowledge as a basis for trust. Jösang compares the concepts of security and reliability based on distinctions between passionate (human-like) and rational entities (essentially technical systems).

Clearly, the interactions and relations between the main entities of the system and the system and its environment have to be modeled at a far more detailed level than the overview depicted in Figure 2. To be effective, this requires the use of, at least, semi-formal modeling techniques, based on for example ideas in (Jösang, 1996).

Requirements engineering

To be able to enforce an adequate level of security in a system, the security requirements on the system have to be formulated. This process has to be performed in concert with the specification of the general system requirements. Starting with a general description of the system requirements, statements concerning security can be extracted. These

statements should be validated and checked for inconsistencies. Implementing such a process of requirements engineering, as described in (Sommerville & Sawyer, 1997), would alleviate the often stated problem that security is considered to late in the system development process. To increase the usefulness of the security requirements, they need to be modeled using well-specified formalisms.

The requirements engineering process has to be performed continually and has to capture the dynamics of the system environment and the system entities, for example considering technical developments enabling the use of new security techniques. Risk management is a core process deciding which requirements should actually be implemented in the system.

Security architecture design

Security architecture design for distributed information systems involves several different architectures regarding distributed systems, information, organization, and operative aspects. Implementing the security requirements is a complex task that has to extend throughout the life-cycle of the system. Ideally, there would be a well-formulated process extending from the set of requirements to the current implementation of the system, as illustrated by Figure 3. To maximize the securability of systems the following issues have to be addressed.

- Selection of architecture.
- Selection of system components.
- System comprehension and assessment.
- Component evaluation.
- Relation to security policy, threat and risk analysis etc.
- Identification of vulnerabilities in the system.

These issues are vital when designing new systems as well as when assessing and redesigning existing systems. To produce viable security architectures, the issues need to be handled jointly. However, this demands, on top of the task to design an efficient security architecture, the solution of all traditional system development issues, making the process more complex.

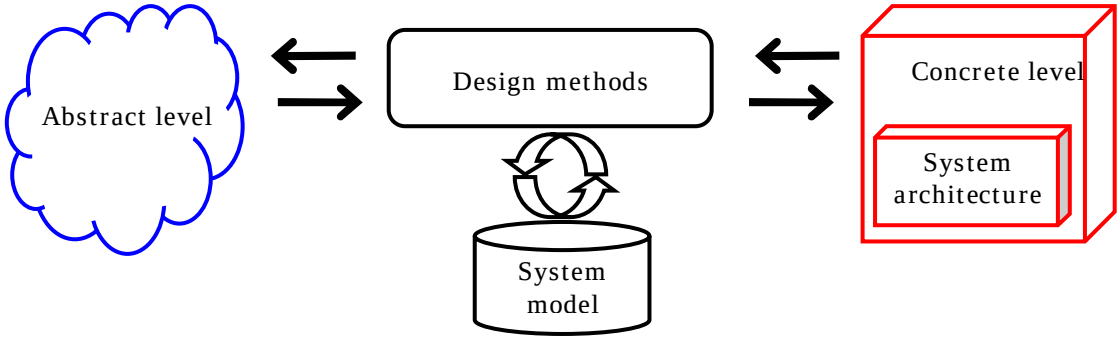


Figure 3: Ideally, there would be design methods to handle the interaction between the abstract specifications and the current implementation of a distributed information system.

An alternative is to introduce a security assessment step into an evolutionary design process, that is, the security is evaluated at different levels of completeness of the system. To enable this, a framework, as illustrated in Figure 4, for design and assessment of security architectures is needed. It should be stressed here that security assessment and system assessment is not separated. The system model is a model of the system as a whole, although the focus of the discussion here is on security issues.

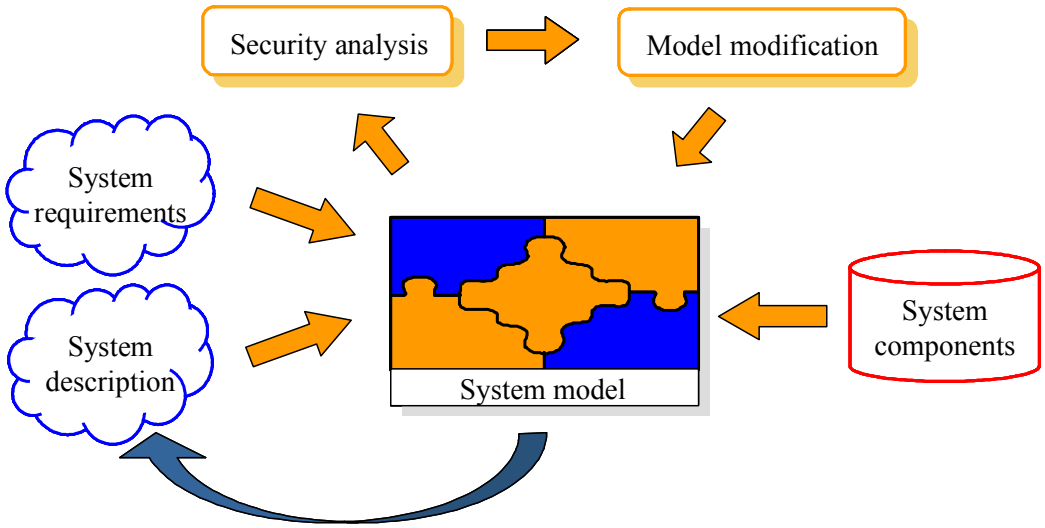


Figure 4: A framework for analysis and modification of system models.

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.

Identification of Security Relevant Characteristics in Distributed Information Systems

In Hunstad & Hallberg (2002b), a set of characteristics is suggested as a first step towards a technique for modeling, building, and evaluation of distributed information systems. The following three steps are performed to find measurable security-relevant characteristics in a distributed information system.

- A definition of basic physical system components.
- A set of security relevant system characteristics for the components.
- Categorization of the system characteristics into a structure.

The resulting tree-structure commences with Confidentiality, Integrity and Availability (CIA) as root nodes. The children nodes are anything from security methods, policies to security products and the leaves are attributes that can be assigned a security value. The evaluation will then be performed by traversing the values from the leaves upwards in the tree, yielding new security values in each step.

In the CIA-tree, nodes of different kinds are mixed. There is no possibility to distinguish nodes that signify security objectives and functions from nodes that denote system components and their characteristics. A limited set of relations can be modeled; these are *"children to"* or *"parent of"*. The tree contains doubles of nodes since certain components support several other components. A more stringent and dynamic structure is needed. This is why an ontology is preferable, instead of the advocated tree-structure.

2.4 Previous Attempts to Model Systems

An interesting question is what previous attempts to model systems have focused on. A literature study discussed in (Hunstad & Hallberg, 2002b) divided a number of articles into five broad categories of relevance especially to design of security architectures. Those categories are: policy modeling, attack modeling, structural modeling, layer-based modeling, and security-indicators modeling. The result of the literature study was limited in terms of finding viable modeling techniques and security-relevant characteristics applicable in the design for securability approach. However, the categories layer-based modeling, with its component-based "plug-in"-approach, as represented by Olivier (2001), and security indicators modeling, as represented by ACSA (2002), seem promising regarding contributions to a design for securability framework. The attack modeling approach, as represented by Jonsson & Olovsson (1997) and Apostal, Foote-Lennox,

Markham, Down, Lu & O'Brien (2001), can be regarded as a complementary approach.

In context of the securability approach, Hunstad & Hallberg (2002b) propose a system modeling technique utilizing UML (OMG, 2001). To capture the necessary information, a modeling framework consisting of three parts is used. The parts are security-relevant characteristics, component library, and system model. The interactions between these three parts results in the ability to capture the dynamics of different aspects of the system without the need to alter the other parts. To implement the modeling framework, a UML-based modeling technique is used. Class diagrams are used to capture the characteristics and the components, while deployment diagrams are used to capture the structure and configuration of the current system design.

Wang and Wulf (1997) present an approach addressing several relevant questions when considering security measurements:

- A framework to calculate scalar values on high-level security attributes is proposed. While the security values of components are assumed to be known, valuable insights on how to acquire these values are given.
- A decomposition method that corresponds to a combination of our characteristics structure and structural system model is described.
- A few functional relationships are introduced in order to model interaction between the factors.
- A method to calculate weights in the resulting tree is presented.
- Component sensitivity analysis is introduced as a means to find sensitive components and possible flaws in the system model.

The approach targets system-wide security measurements, assuming the existence of security values for system components. This differs from the approach in this report, which targets methods to measure security values for system components in order to enable the assessment of system-wide security.

Siponen (2002) categorizes previous attempts to formulate approaches to secure information system design. The result includes two analytical frameworks that relates to our approach, namely the “Security-modified IS development” and “Viable and survivable system” approaches. The “Security-modified IS development” approach is “used to describe approaches that are influenced by IS or software development methods” (Siponen 2002, p.35). The category includes logical modeling by Baskerville (1993). The “Viable and survivable system” approach is represented by two research groups. “Both Karyda and coauthors (2001) and Hutchinson and Warren (2000) have their roots in Beer's viable system model, which consists of five systemic functions which need to be performed in order for an organization to be viable (or survivable).” (Siponen 2002, p. 35)

Baskerville (1993) states that modeling is essential to find “the ideal system solution”, which we acknowledge as central to the design for securability approach. However, Baskerville also states “During one or more design steps, the system is removed from its dependence on concrete technology”. Our firm belief is that the technological aspects and high-level design issues have to be integrated during the whole life-cycle of the system. One reason for this is possibly Baskerville’s focus on design of new systems, whereas the securability approach targets evolutionary system development.

2.5 Certification Approaches

The use of certified products and systems provides a high-level of confidence that the claims being made about security functionality have been independently verified and tested.

A large variety of different certifications exist; most of them have different usage and concerns. Separation could be made between technical and organizational certifications.

Technical certifications

Trusted Computer System Evaluation Criteria (TCSEC), often referred to as the orange book, was initiated 1983 in USA. It was developed mainly to provide a metric to evaluate a degree of trust for computer systems, guidance to manufactures and a basis for specifying security requirements.

In 1990 France, Germany, the Netherlands and the United Kingdom published the Information Technology Security Evaluation Criteria (ITSEC) based on existing work in their respective countries. ITSEC is a structured set of criteria for evaluating computer security within products and systems.

Further discussions concerning TCSEC and ITSEC can be found in (Gollmann, 1999).

Organizational certifications

The most recognized certification in this category is ISO/IEC 17799 (formerly the British Standard BS7799, published 1995). The standard identifies a number of “critical success factors” that an organization must achieve if it is to be successful implementing information security. It addresses most of the physical, procedural, personnel and management issues not addressed by the certifications concentrating on technological aspects (Eloff & von Solms, 2000).

There are also other certifications for individual and organizational evaluations, like the Information Security Awareness Certification from

Information Technology Association of America (ITAA, 2003). Here individuals make web-based tests and have to pass the tests with a minimum score in order to receive the personal certification. Furthermore, the organization must have 90% of its staff to pass the individual test in order for the organization to receive a certification.

Common Criteria

The Common Criteria for Information Technology Security Evaluation (CC, 1999) was initiated 1993 and represents the outcome of international efforts to align and develop the existing European (ITSEC) and North American (TCSEC) criteria towards a common standard for carrying out security evaluations. By establishing a common base, the results of an IT security evaluation become more widely accepted.

CC has a catalogue of standard Security Functional Requirements (SFR) which holds a set of functional components used to express functional requirements of products and systems. CC also has a catalogue of Standard Assessment Requirements (SAR) that is applied to verify that the functional capabilities are implemented correctly. The Security Functional Requirements can be used to develop a Protection Profile (PP) and as a means for developing a Security Target (ST). A PP specifies a profile of the implementation-independent requirements for a class of products or systems that meet specific customer needs. An ST specifies the implementation-dependent *"as-to-be-built"* or *"as-built"* requirements that are to be used as a basis for a particular product or system.

An IT product that is the subject of an evaluation is in CC called the Target of Evaluation (TOE). The security of the TOE is ruled by the TOE Security Functions (TSF). The Security Functions that the TSF consist of are later referred to as the SFs.

A CC evaluation is carried out against a set of predefined assurance levels, called the Evaluation Assurance Levels (EAL0 to EAL7). This scale represents the ascending levels of confidence that can be placed in the TOEs security functions. It covers more the system process evaluation than the system evaluation itself.

The SFR of CC is divided into eleven different classes. Each class contains several families, which each consists of one or more components. The components are also made up of one or several elements. An element is a specific description of a single security task. This structure can be seen in Figure 5 where the class contains three families. Each family contains several numbered components.

The components are hierarchically grouped, which can be interpreted as the component which has the lower hierarchical order is a subset of the component with a higher order. For example, in Family 1 (Figure 5), the

first component is a subset of the second, and both the first and second components are subsets of the third. Components may also depend on other components.

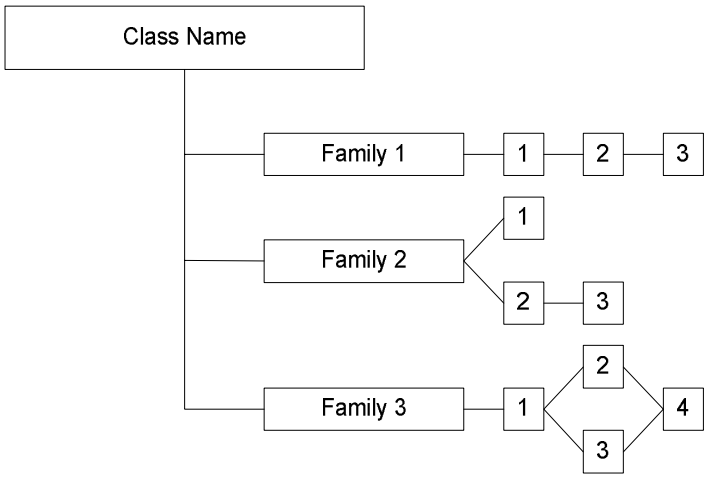


Figure 5: Sample class decomposition diagram (CC, 1999).

Advantages of CC

CC is one of the most commonly used security evaluation standards of today. Security experts have spent substantial time developing CC. Improvements are still made as the area of IT security evolves. Although the purpose of CC differ from the approach proposed in this report, the completeness and usefulness of the security functions still makes them an ideal choice as a bedrock for the evaluation.

The major beneficial functionality of the whole Common Criteria plan is that those who write Protection Profiles, often done with the interests of the customers in mind, will be able to drive the market. Thus the information security can be seen as a market driven industry. The role of CC is that of a meta-standard, providing a framework for spawning more specific standards. What drives the development of CC is explained in greater detail in (Olthoff, 2000; Ware, 1997).

Disadvantages of CC

Unfortunately, there are drawbacks associated with the CC approach.

- CC is an evaluation of design methods, not an evaluation of security functionality. It is the system development process that is being evaluated, not the system itself. This means that the given EAL only states whether a large enough pile of paperwork over the design process exists or not. The correctness and importance of those papers does not even have to be verified and examined.
- Objections may arise to whether CC is usable for large IT systems (Whitmore, 2001). Arguments are being made that the CC is more a standard of evaluation of security functionality focusing on *products*,

thus giving them limitations in describing end-to-end security since their use in complex IT solutions is not intuitive.

- There is a strong emphasis on the *"all or nothing"* nature of an evaluation. A product either meets the profile or it does not. The lack of official feedback to the profile writers leaves them guessing as to what requirements to relax or delete (Olthoff, 2000).
- Another complication is that even a slight change of the configuration renders the evaluation completely unusable.
- One drawback is that CC assumes a static set of threats for the environment. That means that the number of threats and attacks that will endanger the component from its environment are presumed and the evaluation is performed under the influence of this presumption. This environmental assumption, as observed in (Smith, 2003), does not coincide with the usual view; that computer security deals with the worst case scenarios when dealing with risk analysis (while the rest of computer science deals with the average case). If a non-hostile environment is assumed, the evaluation is useless if the evaluated product ends up in a hostile environment instead (Shapiro, 2003).
- Questions about the objectivity of the evaluator may arise, since it is up to his or her own judgment to rule whether a product is to pass or fail a CC evaluation.

Approaches based on CC

A method for designing secure solutions

To be able to use CC as a tool for designing secure solutions, Whitmore (2001) proposes the following extensions.

- A system model that is representative of the functional aspects of security within complex solutions.
- A systematic approach for creating security architectures based on the Common Criteria requirements taxonomy and the corresponding security system model.

The SFR of CC has also been categorized into five strictly operational categories instead of the original eleven:

- credentials/identity
- audit
- integrity
- access control
- flow control.

With these alterations of CC, the SFs are defined, modeled and documented in order to facilitate greater trust in the operation of resulting IT solutions.

A Common Criteria framework for the evaluation of Information Technology system security

Kruger & Eloff (1997) suggests a method of evaluation with three steps that uses CC as a basis to define all security functions.

- In the first step, a list of all functions that could have an effect on the defined security objectives is produced.
- In the second step, this list is then shortened as the most effective functions are singled out.
- The last step deals with comparing the list with the functionality of the existing TOE.

Moreover, Kruger & Eloff appoint a number to the security functions, referred to as *"strength of association"* (SOA). This number determines the effect the security function will have on the objective. The functions are structured in a tree together with the objectives, and the vertices have been given SOA-values. Then the impact of every function on the objective is calculated to see their respective effects on the objective.

3. Security Measurement

IT security measures are needed as input to several processes regarding:

- risk management,
- requirements engineering,
- system design, and
- system operation.

Such measures are vital in order to tune systems and security efforts towards the goal of obtaining “good enough” security. Inadequate security will prove costly, either through security breaches, in the case of to poor security, or expensive systems and administration and hampered organizations, in the case of too rigid security.

At a general level, the problem is how to perform efficient risk management, that is, to detect, analyze and mitigate the security risks posed by the use of a system. To achieve this, the sensitivity of information has to be connected to the system components processing, storing, and transmitting the information and put into the context of the system. Thus, a fundamental input to risk management results from the assessment of different security levels in systems and system components. To quantify security in a meaningful way is extremely difficult and requires vastly differing system aspects to be considered, e.g. cryptographic algorithms and social engineering. The problem can be divided into the tasks of assessing the security qualities of the components of the system and how the system is operated. Consequently, the process of identifying, analyzing and describing system risks can be divided into securability assessment, security level assessment, context modeling, and risk level assessment, as illustrated in Figure 6. Each of these tasks requires input and generates results as illustrated in the figure.

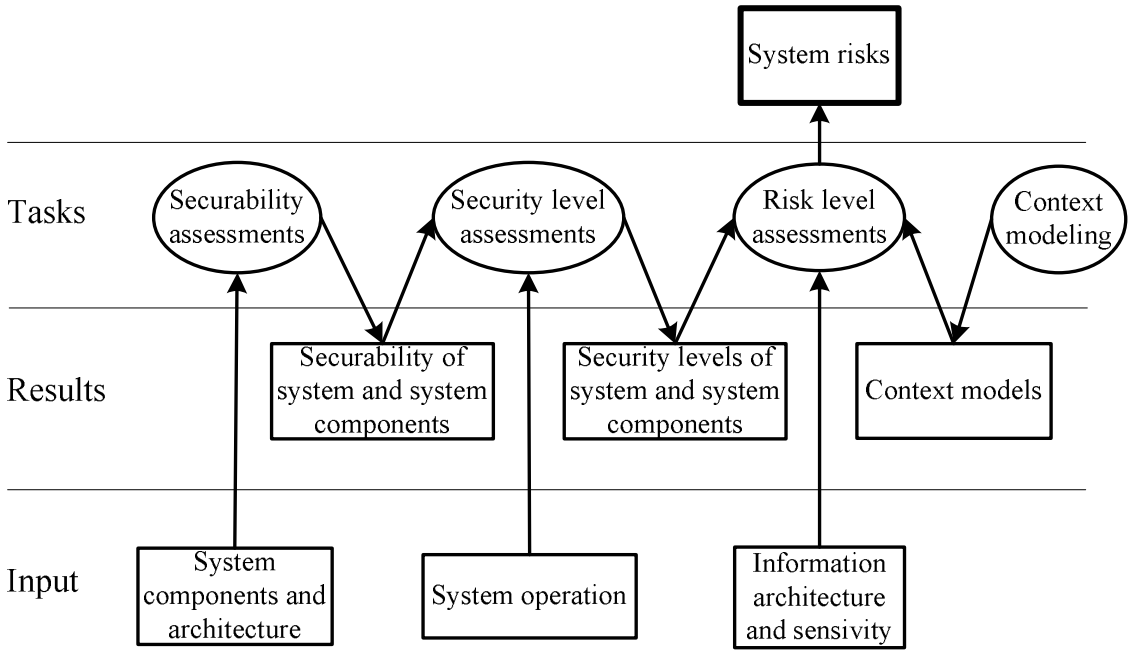


Figure 6: Securability and security level assessment are essential to efficient risk assessment.

In the remainder of this chapter, a number of security assessment related terms are introduced. These terms are central for the interpretation of the rest of the report. The first term relates to the core problem of defining what actually is meant by measuring security.

> **Security metric**: A security metric is a ruler against which the security of systems is measured, that is the correspondence to meter or foot when measuring length or distance.

A possible security metric is the mean time between security violations, although it will be difficult to measure before the particular system is in operation. Another possibility is to use real numbers from 0 to 10, although in this case there will be arguments against the validity of such a metric since a value on this scale does not result in any intuitive feeling about the security of the corresponding system (ACSA, 2002).

As mentioned earlier, the notion of measuring security is interpreted vastly differently. Here, three different scopes of assessment, namely securability, security level, and risk level, are introduced to set the scene. These scopes set a context for the definition of security metrics.

> **Securability**: The goal of designing for securability is that systems can be secured to a required level during operation (Hunstad & Hallberg, 2002). Thus, securability assessment aims at evaluating the strengths and weaknesses of security mechanisms considering technical, organizational, and individual aspects. Consequently, securability is assessed pre-operational or during operation ignoring the actual influence of operation on the security level.

Security level: The security value for a system that is in use and, thus, have its operational aspects included in the model.

Thus, the problem of assessing system securability is focused on the security mechanisms implemented in systems. It does not include the operational aspects yielding the actual security levels of systems or the information sensitivity and value required to decide the security risks.

Securability is measured on the design of an information system, including technical, organizational, and individual aspects, aiming at an estimate of the level to which systems can be secured during operation and the effort required to achieve a certain level of security. Thus, the securability is constant as long as the design is not changed.

Security level is measured on an implemented information system in operation, including technical, organizational, and individual aspects. During system operation, the current security posture is the major concern, e.g. which nodes are functioning and trustworthy, which users are active and how are they connected and authenticated. Thus, the security level of a particular system can change whenever the implementation is altered, that is with system reconfiguration, or with other events affecting the security posture of the system. For example, when a new user account is added, the security level is affected, although with sensible access control mechanisms, the influence may be insignificant. If a user account is locked because the (legitimate) user fails to provide the correct password, the security level is affected (hopefully mainly considering availability).

Securability and security level are measured on different scales and, thus, are not possible to compare. Naturally, the measures can be compared using some reference transformation, i.e. assumptions about the operation, but then the operation has been modeled in some way. A metaphor is car and driver. A car has a performance, which can be estimated in various ways, e.g. the torque of the engine can be measured to give an estimate of the power of the car and its fuel consumption. However, the performance of the car on the road is highly dependent on the driver and the environment in which the car is driving.

Risk level: When the contextual environment, i.e. threats and assets, are considered, an assessment will yield a risk level.

If the operated system is put in a context, risk can be estimated. Apart from security level, risk depends on antagonists and the possible damage caused by security breaches. Thus the security level is assumed to be independent of the context of the system. Efficient evaluation of risk levels enables powerful risk management to be performed.

Security value: Security value is the common term used to denote securability, security level, or risk level.

Security strength: Security strength denotes the ability of a system to uphold the security standards specified for the system or, even more vaguely, just a relative term indicating the general standard of security enforcement in the system.

Security indicator: When assessing the security value of systems, input is needed. This input consists of security indicators that can be measured.

To be able to assess security values, systems have to be modeled. As illustrated in the top right of Figure 7, the securability of a system is reasoned upon in the context of a model, not reality which is the context of the actual securability (top left). Ideally, the securability estimated from the model would correspond directly to the securability of the real system. However, there will always be discrepancies resulting from the transformations between the real system and the model, that is, modeling and implementation respectively. In order to minimize the errors, models that capture the adequate characteristics of the system have to be formulated.

The security level depends on the securability of systems and how they are operated. Thus, to assess the security level of systems, system operation has to be modeled, as illustrated by the arrow between the top right and middle right of Figure 7. Correspondingly, the risk level of a system is assessed based on models of the system, as illustrated in the lower part of Figure 7.

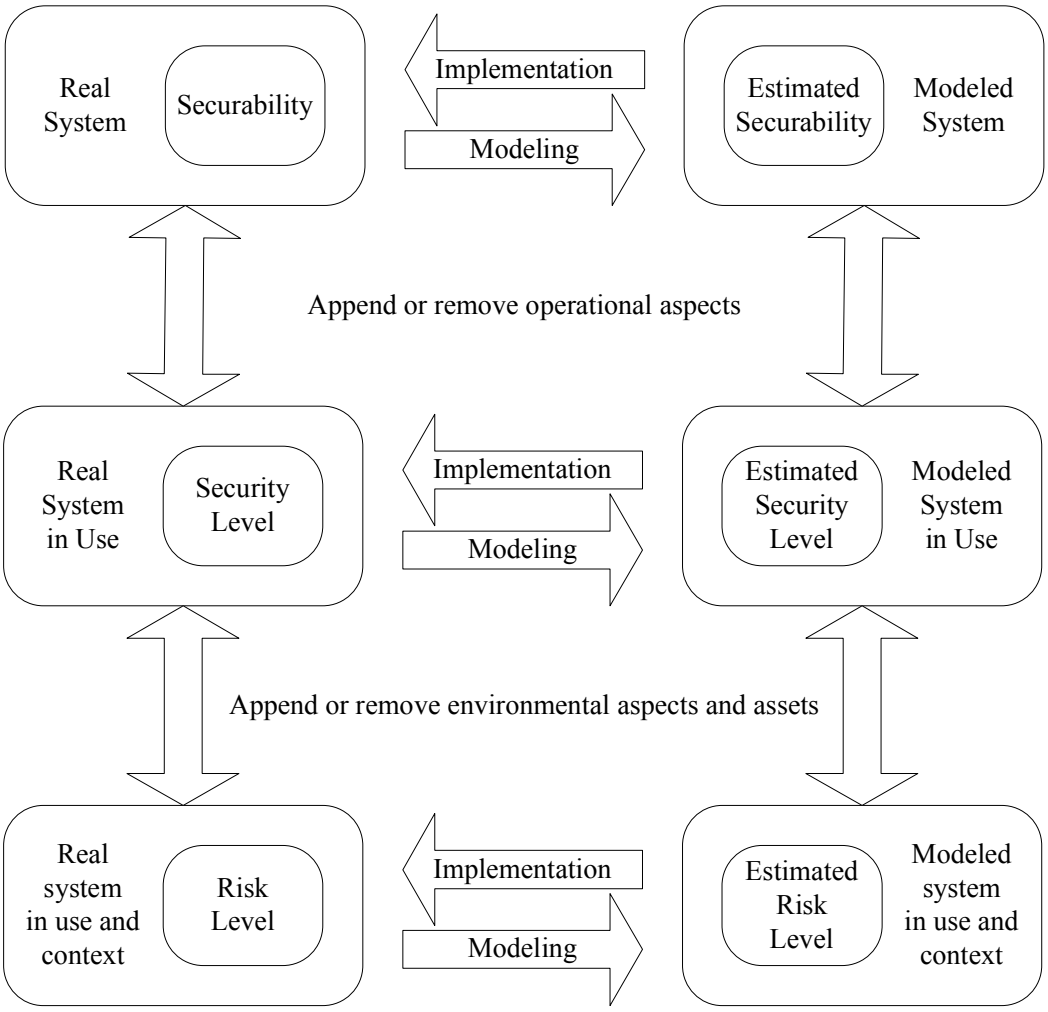


Figure 7: The correspondence between models needed for security value assessment and reality. The inner boxes represent the aspect of security value, and the outer ones the system scope.

4. Security Evaluation Framework

The main purposes of the security evaluation framework are to support the development of system modeling techniques and security evaluation methods. A security evaluation method should be capable of evaluating the security of a system. The evaluated system is modeled with the system modeling technique. These relations, together with the possibility to implement a distributed information system given the system model, are illustrated in Figure 8.

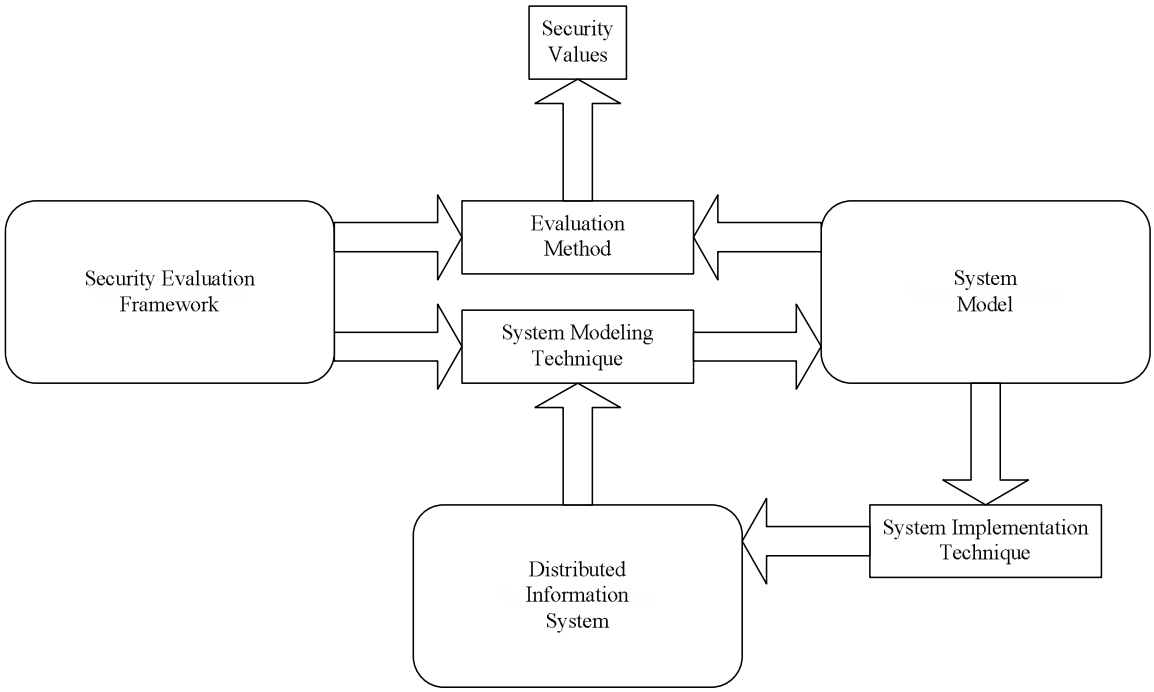


Figure 8: Overview of the relations between the security evaluation framework and other relevant concepts.

The security evaluation framework is illustrated in Figure 9. The framework represents the entire evaluation process in which a security value is being estimated. It models the reality and those restrictions that are being made in order to get accomplishable evaluation methods. The evaluation process is being divided into parts, where each part in the framework of evaluation can be viewed as one independent step in the process of evaluating a distributed information system. Every part is represented as a block in the figure and the arrows show the dependencies between the blocks and how to expand the results in order to calculate meaningful security values. Every block can also be thought of as a module, where all modules work together toward the common goal of the framework. This concept of modularity is further explained below.

The framework is built around the black component in the middle of Figure 9, which all other parts of the framework relate to in some way, as shown by the arrows. The solid arrows indicate that the concepts are inherited, thus indicating a strong relationship. The dashed lines indicate a much softer relation with a meaning that will be explained later on in this chapter. Recursion is indicated by arrows pointing in both directions (between TOE, instance and technological), showing that evaluated instances are stored in the component library for later use.

The component (represented by the black block) is partitioned into several concepts depending on which partition the component belongs to, represented by green blocks in the figure. The environmental component leads to a contextual concept that divides into concepts dealing with risk analysis, which are marked red in the figure. The technological component deals with technical system components, their conversions into targets of evaluation (TOEs), evaluated instances of components, and systems made up of instances of components. The TOE relates to the different security values via the Common Criteria (CC) security functional requirements (SFR). All these concepts originating from the technological component are in the figure marked in yellow. Different security values (blue blocks) are reached depending on which component partitions that were regarded. The technological components and the security values, the concepts visualized in yellow and blue in the figure, are described in chapter 5.

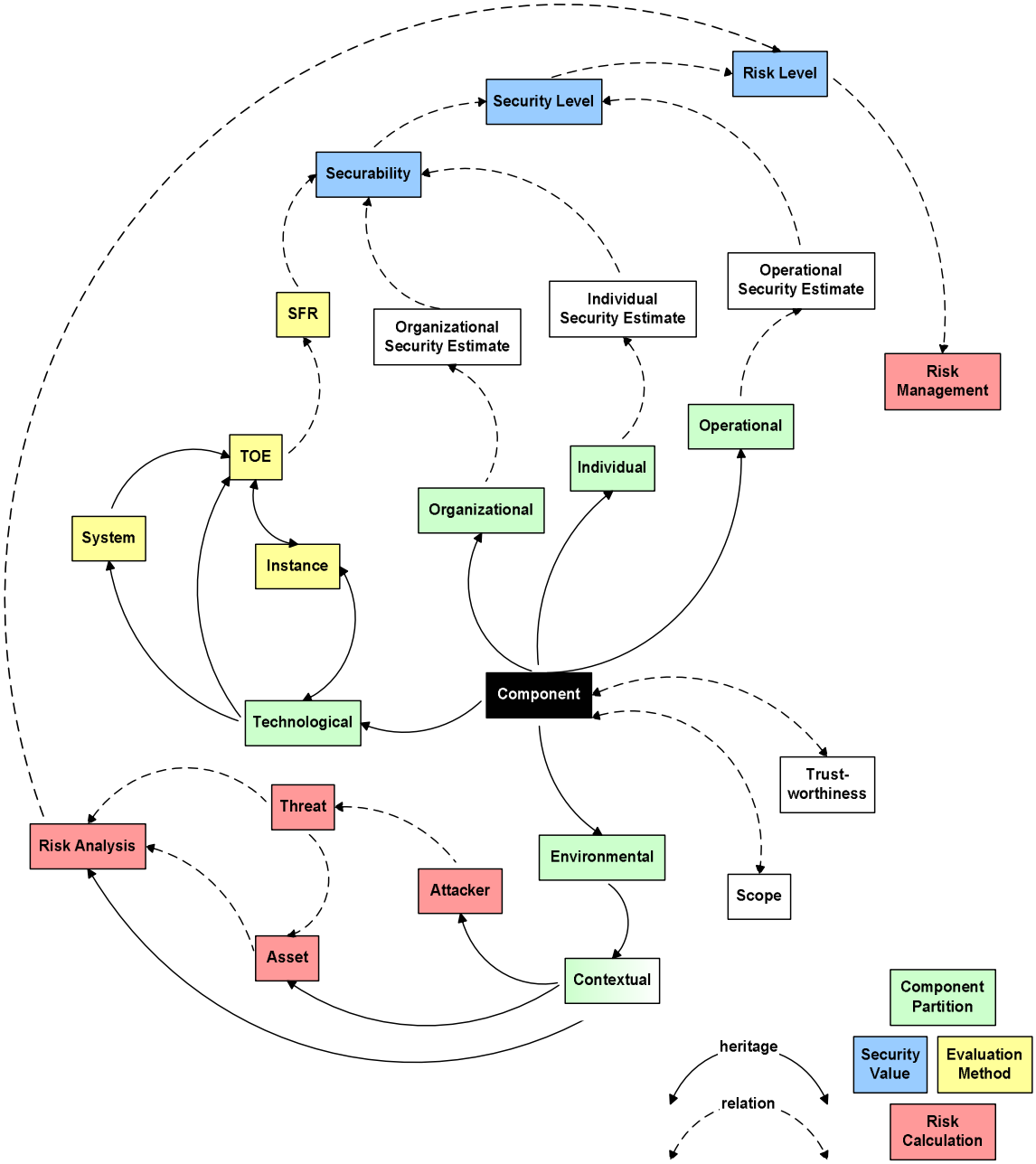


Figure 9: Security evaluation framework model.

The framework is a highly dynamical structure, giving evaluators an opportunity to change it according to their own interests and needs. There exist no apparent restrictions in the framework method as a whole; only in the modules themselves as it is here the restrictions are being made. Most restrictions are visible and marked, making the problem of accommodating for them, perhaps not always easy, but at least a specified, and hopefully, solvable problem. For example, by letting the module "*SFR*" represent the evaluation of a TOE towards a security value, restrictions on how the security of the component is measured and evaluated are introduced. Some of these restrictions come from the fact that the set of security functions

contained in SFR does not cover all security functionality, others from the mapping and evaluation of the security functions.

In the following sections, the characteristics and features of the security evaluation framework will be explained in detail. In section 4.1, the modularity concept is explained. Depending on what the evaluation should focus on, different scopes may be useful. The scopes segment and model the distributed information system differently. Some examples of scopes together with clarifications can be found in section 4.2. Evaluated TOEs are stored as an instance in a component library together with the results of the security evaluation as explained in section 4.3. The system with its components and instances and their relations to the SFR and the security evaluation are structured and visualized by the use of an ontology, as described in section 4.4. A security evaluation method that rates the security properties of a TOE according to the security functions of the Common Criteria is explained in section 4.5.

4.1 Modularity

An advantage of the framework is its flexibility. When creating methods for security evaluations, every restriction introduced in a module should be clearly defined. In this way, restrictions in evaluation methods become visible and may be relaxed at a later stage, since a less restricted module can be created and introduced into the framework. The entire evaluation method will not have to be changed, only the module itself. This will make the framework easy to evolve into an increasingly better structure for security evaluation. Moreover, modules suited for specific demands or tasks can be created.

By dividing modules in the framework, the complexity of the resulting modules decreases. This, together with the potential for improved performance of the modules, gives reason to believe that the restrictions in the framework will decrease, making the framework more complete over time.

4.2 Scope

Different scopes may be chosen for the segmentation of the system, depending on focus and goals of the evaluation. Several scopes may be regarded simultaneously, to get a greater perception of the specific view of the system or model evaluation. Examples of different scopes are the following:

- **Component structure**: How the components, both hardware and software, fit together is one possible aspect to consider. This is the most intuitive scope and is the one which is considered in this report, if nothing else is mentioned.

- **Information flow**: How the information flows in the system could be important for certain security tasks. If the security evaluation were to focus on the importance of the assets, and which subjects that were to be granted access to these assets, this scope of interest would probably be adequate.
- **Services provided**: It is possible to view the system as a set of services. This scope is similar to information flow, but on a higher level of abstraction since it is the services and applications that are in focus and not the information. Services could for example be to supply internet-pages or download files from ftp-servers.
- **Physical structure**: This model is a physical segmentation of the network and its surroundings, thus making it possible to model threats regarding physical intrusion.
- **Attack model**: An attack model shows how the system is supposed to handle intrusion-attempts. The system states are represented by nodes, and vertices represent the events that lead to a change of state. The other scopes are likely to benefit from collaboration with this scope, as it is useful in highlighting weak spots in the security modeled by other scopes.

4.3 Component Library

A component library is built by storing evaluated TOEs instances in a component knowledge base. The typical instance will be a component of a specified brand and model or a general standard component. Thus, evaluations will become less demanding, since already evaluated components will not have to be analyzed and evaluated further.

An example of a component structure can be seen in Figure 10. The physical scope has been chosen in this example in order to better illustrate the features of the component library, as this scope better divides the information system into clear blocks. Every block is a TOE. In order to evaluate it, all blocks that are contained within have to be previously evaluated and put into the component library. The system described in the example contains several workstations, each containing a computer accessible by several different users. These workstations are supposed to be situated in the same room area (denoted compartment). The compartment also contains physical protection (i.e. lock, bio-scanner or anything else that aims to keep unauthorized people out of the compartment). Different compartments are then hooked up to each other and to the Internet by connectivity links.

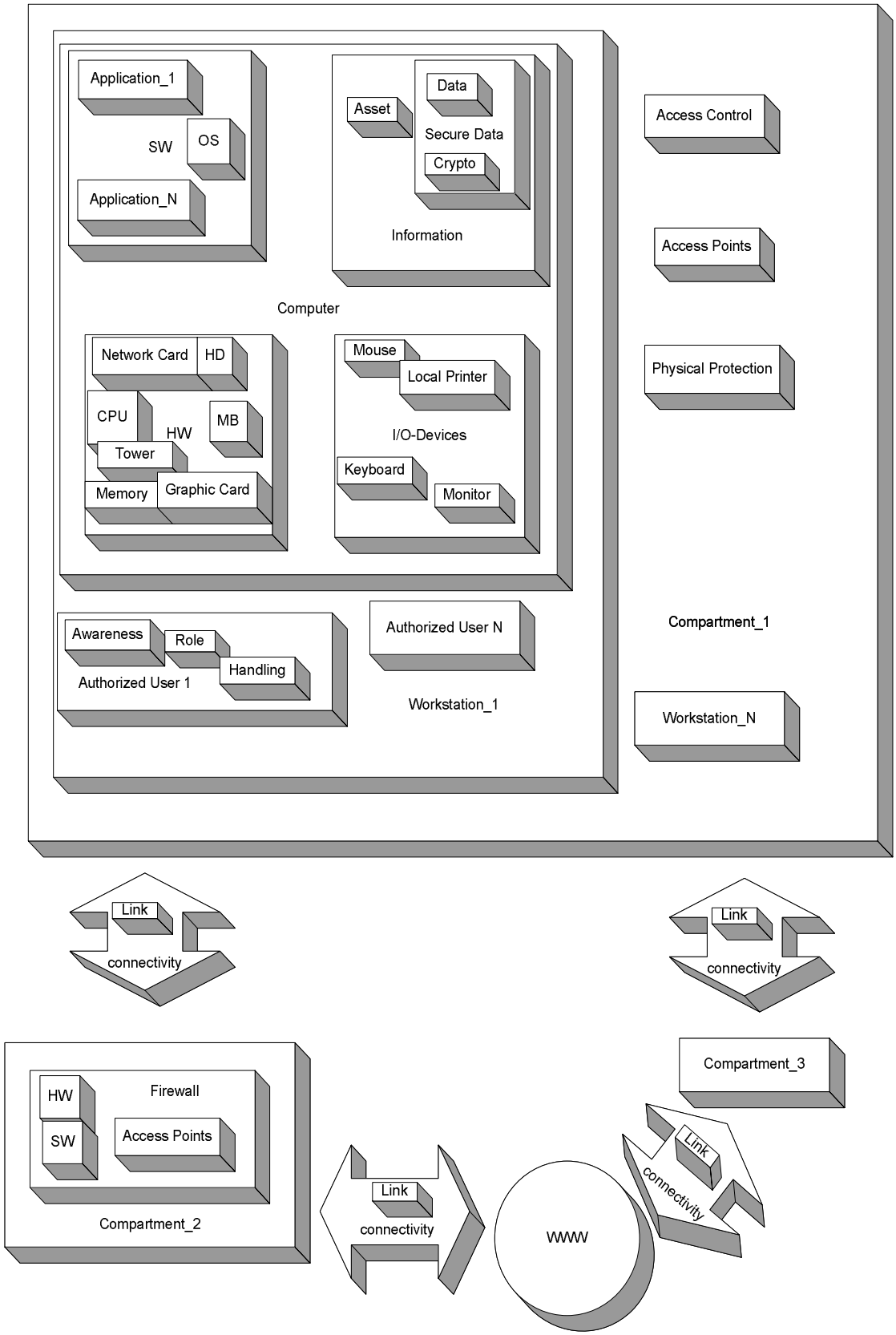


Figure 10: Example of a component structure for a distributed information system.

4.4 Ontology

Donner (2003) expresses why ontologies are regarded to be necessary for the future of computer security.

"*What's missing* [in computer security] *is a broader context that we can use to organize our thinking and discussion. What the field need is an ontology – a set of descriptions of the most important concepts and the relationships among them*. [...] *A great ontology will help us report incidents more effectively, share data and information across organizations, and discuss issues among ourselves.*"

Objects and concepts in the security evaluation framework are easily described using ontologies. This is a highly versatile way to describe a specific conceptualization of the world. An ontology is very dynamic in its structure, using heritage-relations to specialize specific concepts. All concepts may contain slots of attributes and relations to other concepts. Instances of concepts may be created to turn the ontology into a knowledge base.

The main reason for choosing this way of representation is that it makes the framework model and its modularity-thinking, described above, easier to represent. Every module in the framework may be described with its own ontology and then the ontologies may be put together with relations over the ontology-boundaries. If a specific module is replaced by a, at least in some specific aspect, better module, then only a small ontology has to be thrown away, not the whole one. The meaning of the relations between the discarded ontology and the others may easily be put into the new ontology instead.

The ontology created for this work tries to cover two diverse aspects. The first is to have a more dynamical tool to express the security objectives, functions, system components, their properties and the different relations that connect them all. The second reason for using an ontology is the one advocated in the introduction of this section; that we need a common defined terminology in computer security to be able to cooperate and be able to express ourselves understandable.

In Figure 11, the security evaluation framework of Figure 9 has been described using an ontology. The root node, *thing*, is the concept from which all other concepts inherit. From here, the concepts component, TOE, security functional requirements, security value, instance and risk handling originates. The component concept is divided into five partitions depending on focus. One of the partitions, the technical component, has been specified further to exemplify the structure, although complete specification of this component would be too extensive. Another partition, the environmental component, is a generalization of the contextual concepts of attacker and asset. These few concepts that are mentioned under the technological and

environmental components are only a fraction of the whole component library idea presented above. The security functional requirements concept is a generalization of the eleven SFR classes of CC. They can all be further specified into families, CC components and elements, even though these concepts are not shown in the figure.

The concepts of securability, security level, and risk level all inherit from security value. Risk management, risk analysis, and threat inherit from risk handling. These concepts are connected by named relations. For example, the technical component is translated into a TOE, which after it has been evaluated will be stored as an instance together with its security values. The individual, organizational and operational components are needed to be able to estimate different security values. There are also some relations between the contextual environmental component and the concepts under risk handling.

The main advantage with the ontology compared to the framework model in Figure 9 is that the objects in the model and their relations become more structured. It becomes easier to state relations between objects and the hierarchical inheritance introduces taxonomy-like partitions wherever it is convenient in the model.

Some more general reasons for developing and using an ontology, apart from the ones mentioned above, are the following (Noy & McGuiness, 2000),.

- Ontologies will make it possible to share common understanding of the structure of information among people or software agents.
- Creating ontologies will enable reuse of domain knowledge by integrating several existing ontologies that describe portions of a large domain or reuse a general ontology and extend it.
- The use of ontologies makes domain assumptions explicit, making it possible to change these assumptions easily.
- Ontologies separate domain knowledge from the operational knowledge. It will be possible to describe a task of configuring a product from its components, according to a required specification and implement a program that does this configuration independent of the products and components themselves.
- Ontologies are helpful in analyzing domain knowledge. A formal analysis of terms is valuable when attempting to reuse existing ontologies and extending them. Developing an ontology is akin to defining a set of data and their structure for other programs to use.

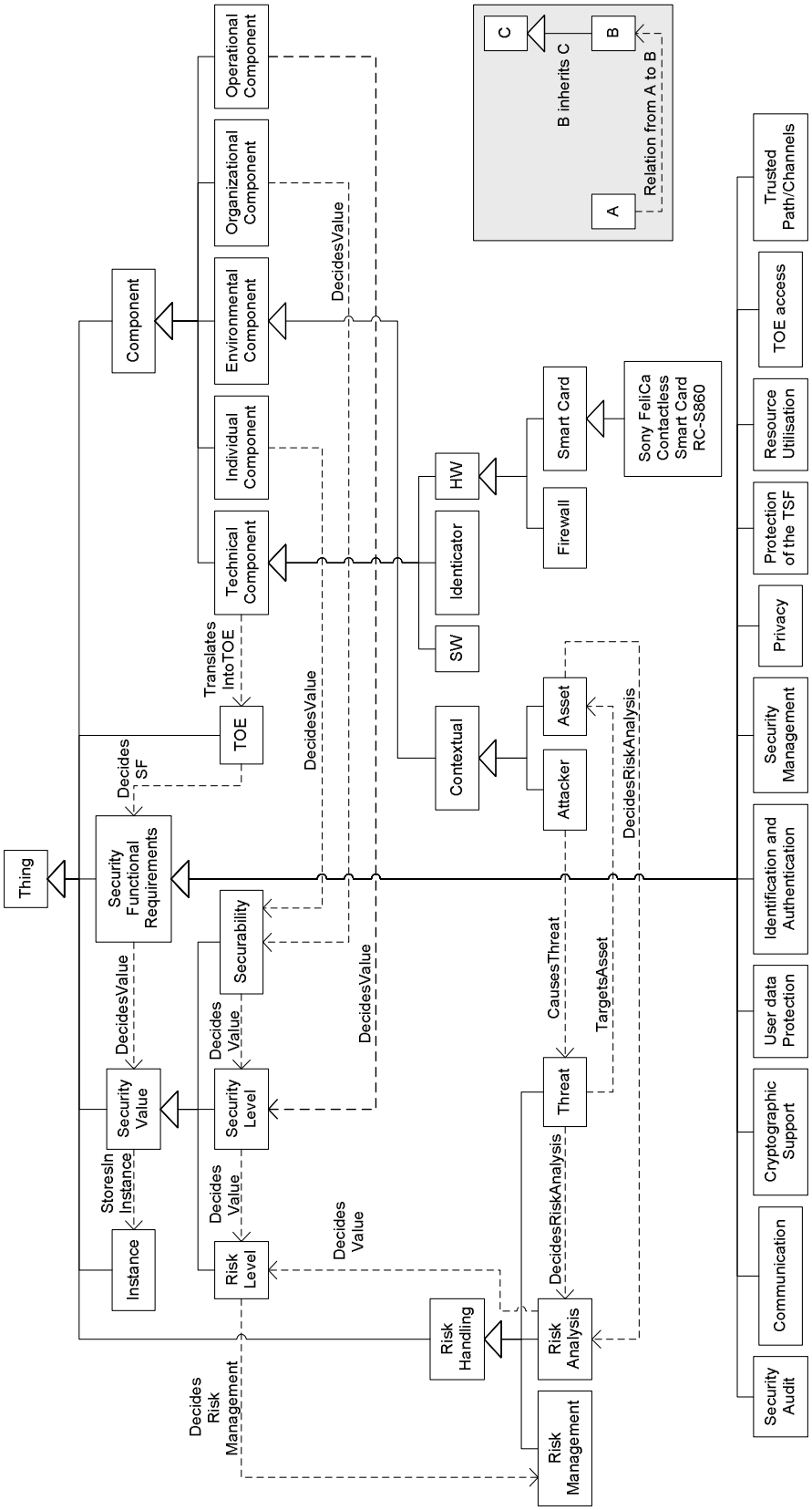


Figure 11: Ontology of Security Level Evaluation.

4.5 A Security Evaluation Method

To illustrate the use of the proposed framework, a security evaluation method considering technical aspects is described. The method uses Common Criteria (CC) as a foundation, or more specifically; its set of Security Functional Requirements (SFR). The component whose security is being measured is called the Target of Evaluation (TOE). An ST or a PP may be obtained and analyzed to clearly map the needed characteristics of the system component to the components of the Security Functions of CC.

Here lies a problem, both the components of the system that are evaluated, and the components that build up the CC Security Functions are being called "*components*". If the meaning of the single word component may be misinterpreted, it is referred to as system component or CC component depending on what is intended.

The system component gives specific values to the CC components, depending on how good the component is at enforcing the specific functionality.

These values can be mapped to CIA or PDR to generate more specific security values. A securability-evaluated TOE is then stored together with its securability characteristics in the component library, so it can be used at a later stage as a building block for a subsystem.

This method of evaluation is further explained in the next chapter.

5. An Approach to Securability Evaluation

This chapter explains the four steps of a securability evaluation method. The method targets the evaluation of system components and is limited to the technical aspects of those components. However, possible extensions to enable system-wide evaluations are discussed.

The first section explains the need for a metric and security values in security evaluation. It also covers some of the problems that occur when trying to reach these values. The second section explains the Security Functional Requirements (SFR) of CC, the meaning of the TOE, and the changes made in the SFR in order to suit the securability evaluation method. The third section explains how to interpret evaluated SFs and also explains and exemplifies the steps to follow in order to reach meaningful security values. The fourth section explains how to map the characteristics of a system component to the SFR of CC.

The fifth and last section explains ideas for combining evaluated components into an evaluated subsystem.

5.1 Security Values and Metric

Adequate security values will be needed for efficient decision support, since it is easier to assess a security property if it is assigned a security specific value. It will also be easier to report the status of an information system if such security values do exist. Exactly what it should describe is difficult to define, since most evaluators tend to demand and focus on completely different aspects of security. For example, government and commercial sectors have different agendas; the former is policy driven and the latter is profit driven. IT security is a general term and therefore no single security value will successfully quantify the security of a system as a whole. Additionally, when information systems evolve and extend, the problem of making statements of their properties increases drastically.

As observed by Wang & Wulf (1997), the solution of proposing a universal security measure will not solve these problems of quantifying, but it might be a step in the right direction; to best approximate the security strength of a system.

This security measure can not be made directly by simply measuring the strength of the components, since the dependencies and structures of information systems are too complex. Instead an estimation of these properties must be made.

The proposed method aims at evaluating the securability of individual system components. Thus, the method does not consider operational aspects and, moreover, it is limited to the technical parts of the securability aspect, that is, the organizational and individual parts are not regarded.

The proposed range of values associated with securability is in the interval [0, 1]. This value can be regarded as a probability estimate. It could for example give an indication of the probability that a random attack or vulnerability lead to a security failure. A zero (0) for a specific CC SF, will indicate the significant possibility of a vulnerability. A one (1) will imply that the system component is secure regarding the specific security functionality. There is no possibility whatsoever for the particular security function to fail. If an SF is not valid for a specific TOE, the NULL-value is used.

Unfortunately, it is impossible to reach an exact evaluation of such a number. Exactly what a specific value will mean is not decided at this stage, that is, no metric is defined.

The use of securability values differs the proposed evaluation method from the approach of certifications like CC, since systems can be rated and compared to each other. The only thing in CC that is similar to a securability value, is the *Strength of Function* (SOF), that is estimated for a few CC components (e.g. cryptographic operation). SOF is a discrete three-valued estimation (low, medium or high).

5.2 CC Security Functional Requirements

This section explains the CC Security Functional Requirements (SFR) and the use of SFR in the evaluation method. Background information about CC can be found in chapter 2. For the purpose of an information system model, it is only the SFR of CC that is interesting since there is no place in the evaluation process for the assurance, as it is regarded in CC. This is because of the fact that the classes defined in CC Standard Assessment Requirements (SAR) deals with the development process of a system component, not the component itself.

CC aims at establishing trust in existing IT products by estimating their level of assurance, while the method proposed here uses the SFR from CC as a tool to estimate and assign securability values to the different security functions in a system component.

CC deals with a TOE, a concept which this report also has adopted. The structure of a TOE is illustrated in Figure 12 below. Users interact with subjects within the TOE, and the TSF, containing most of the security functions, decides which subjects are permitted access to which objects.

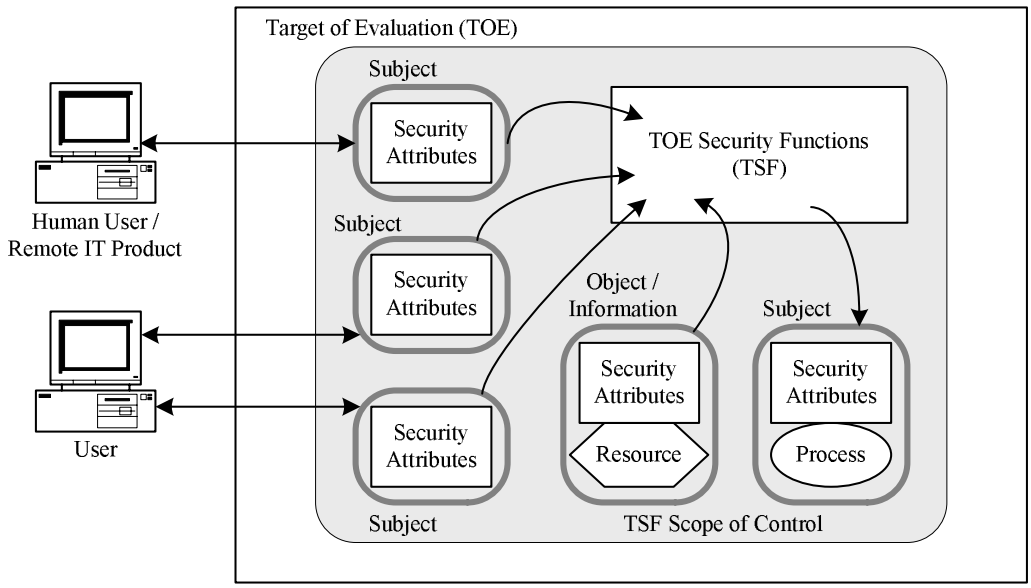


Figure 12: Security functions in a TOE of a system according to CC (1999).

In Figure 13 below, a distributed system is the TOE, which leads to a different view. The primary difference is in the number of connections and the main problem when evaluating the TOE lies in the combination of the SFs.

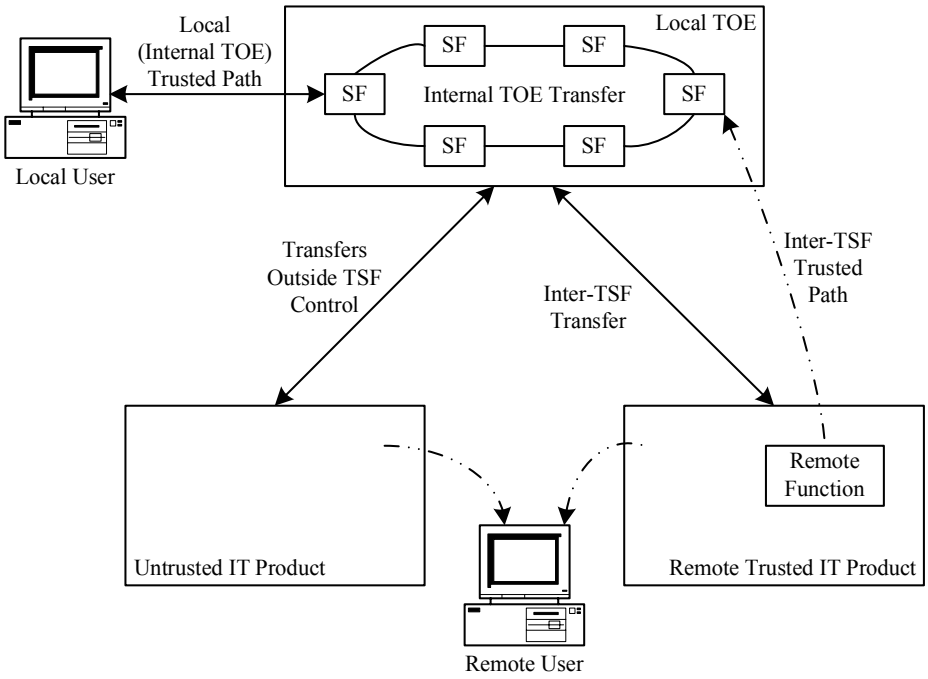


Figure 13: Security functions in a TOE consisting of a distributed system according to CC (1999).

The eleven classes of SFs are, as mentioned in chapter 2, divided into several families that consist of one or many components. The components consist of elements, that is, specifically stated security tasks. The classes,

together with a description of them and an explanation of how they are to be interpreted, are described below.

The most important alteration of the CC SFR is how to regard the hierarchical ordering of the CC components. This ordering is, as mentioned earlier, based on the fact that the previous components are complete subsets of the following ones. This means that all security functions in the components exist in the hierarchically following ones, and new functionality is added as well. In CC, the foremost meaning of the components is to provide a vocabulary to describe the security functions in a computer system. However, since we have changed the purpose of the CC components from description to evaluation, these hierarchically ordered components are hardly justified anymore. A better way would be to change the component structure in the following way.

- If the intersection of the security functions of two hierarchically ordered components is of the same type as the conjunction of the components, but with a higher degree of security, both components should be combined into a single one. When evaluating these components in a system using an ST, the only difference is that the component being a subset of the other should be given a lower security value than the other. Components that are combined in this way are marked with an asterisk (*) under the ID-column in table 3 in appendix. For example, in FCO_NRO (non-repudiation of origin), the first component (selective proof of origin) is very similar to the second one (enforced proof of origin). The main difference is that the first component specifies exactly which sort of transmissions that are guaranteed proofs, while that second component guarantees proofs for all transmissions that are being made. Therefore they should be combined into one component, and if the second component was to be found in an ST, the system component would be assigned a higher security value than if only the first component was found.
- If the intersection of the two hierarchically ordered components is of a different type than the complement, the intersection itself should, if possible, form a new component, while the complement forms another. Components that are created in this way have their component number put inside parentheses under the ID-column in table 3 in appendix. For example, in FDP_SDI (stored data integrity), the first component (stored data integrity monitoring) is very similar to the second one (stored data integrity monitoring and action). The difference is that the second component will perform an action to accommodate for the loss of integrity. Since this action is of a completely different type than the monitoring, it should be put into a new class. The two original classes would instead become: stored data integrity monitoring, and action due to loss of stored data integrity.
- For families that consist of only a single component, the purpose of the component will be only to introduce symmetry in the component

structure, since the meaning and functionality of the family and the single component will be equivalent. Since the number of families containing only a single component will increase when combining components as mentioned above, it will be easier to manage the SFR structure if these single components were removed. The loss of symmetry in the structure has no negative influence. If a symmetric structuring for some reasons is needed, the family that is without components could be considered as a single component as well. These families are marked with an asterisk (*) under the ID-column in table 3 in appendix.

In the following subsections, the classes and their families are explained (Bottomly, 2002). It should be noted that even though the elements are not thoroughly discussed in this chapter, they play a great part in the evaluation process, since they are the smallest part of the security functions that the CC component can be broken up into, and it is ultimately in the element that the evaluation of the SFs are being made.

A complete list of all the classes, families and components, after the changes discussed above have been made, can be found in table 3 in appendix.

FAU – Security Audit

The 6 families in this class address:

- recognition and responding to security-relevant events and activities (FAU_ARP)
- recording security-relevant events and activities (FAU_GEN, FAU_SEL)
- storing and protecting security-relevant events and activities (FAU_STG)
- review and analysis of security-relevant events and activities (FAU_SAA, FAU_SAR).

This class is used for detecting security events, with help from the classes FDP and FPT.

The Security Audit class is special, since it affects almost every component in CC. The minimal requirements for audit that every CC component is supposed to fulfill is specified in CC. Based on these statements a classification of the dependability of each component of this class is specified, depending on how much it is affected by it. For example, if the effectiveness for a certain component to succeed is heavily dependent on that the security audit log-files are controlled at regular intervals, then the dependability will be very high. But if the impact of not monitoring the log-files will not result in severe security failures, then the dependability values will be low.

The audit values will be statically set for each role in a subsystem, i.e. the one responsible for controlling and acting on the log files will be evaluated according to his skill. All audit security functions that are controlled by that person will yield the same effect on the dependent components, but how much that effect will be is decided for each SF.

Components that are dependant on any kind of audit management are marked with an *"A"* in the last column of table 3 in appendix.

FCO – Communication

The 2 families in this class address:

- proof of origin of transmitted information (FCO_NRO)
- proof of receipt of transmitted information (FCO_NRR).

This class is used for enforcing proof of transmission in communications.

FCS - Cryptographic Support

The 2 families in this class address:

- generation, distribution, access and destruction of cryptographic keys (FCS_CKM)
- operational use of cryptographic keys (FCS_COP).

This class is used for cryptographic protection, with help from FDP and FPT.

Characteristics for deciding the security value are crypto algorithm, key length and key distribution method. Evaluations concerning the practical strength of cryptographic keys are hard to accomplish since there are, for apparent reasons, hardly ever any feedback on when or how cryptographic systems fail (Anderson, 1993).

FDP - User Data Protection

The 13 families in this class address:

- security function policies for protection of user data (FDP_ACC, FDP_IFC)
- access control and information flow control functions for protection of user data (FDP_ACF, FDP_IFF)
- authenticity and integrity for protection of user data (FDP_DAU, FDP_ITT, FDP_SDI)
- reuse and rollback for protection of user data (FDP_RIP, FDP_ROL)

- protection of import and export of user data (FDP_ETC, FDP_ITC)
- protection of user data for communications between the TOE and SFs of other trusted IT products (FDP_UCT, FDP_UIT).

This class is used to protect user data when controlling access to information (together with FMT and FPT), monitoring loss of user data integrity when detecting security events (together with FAU and FPT), and protecting transmitted user data in cryptographic protection (together with FCS and FPT).

Access control is the control of subjects, objects and the operations among them, while information flow control rules the subjects, information and operations which causes information to flow to or from subjects. The latter control should be measurable and possible to evaluate on each of the 7 OSI layers (Open Systems Internetwork).

FIA - Identification and Authentication

The 6 families in this class address:

- establishing claimed user identity (FIA_ATD, FIA_SOS, FIA_USB)
- verifying claimed user identity (FIA_UAU, FIA_UID)
- failures when authenticating claimed user identity (FIA_AFL).

This class is used for controlling access to systems, with help from FTA and FTP.

A major characteristic for deciding correctness of validation in FIA_UAU is the False Acceptance Rate (FAR) that states the percentage that a person is falsely granted access when the person should rightfully have been denied access. This rate is sometimes calculated for specific products like bio-scanners and card-readers. The False Rejection Rate (FRR) is a minor characteristic, only affecting the availability. It states the chance of being denied access although access should have been granted.

FMT - Security Management

The 7 families in this class address:

- specifying the management functions of the TOE (FMT_SMF)
- management of TSF data (FMT_MTD)
- management of security attributes of the TOE (FMT_MSA, FMT_REV, FMT_SAE)
- management of the security functions of the TOE (FMT_MOF)
- security roles of the TOE (FMT_SMR).

This class is used for managing TSF data to control access to information, with help from FDP and FPT.

Much like the Security Audit class, the Security Management classes is also special and affects many other CC components. For example, if the effectiveness of a certain CC component to succeed depends heavily on that an administrator has updated and set the correct security attributes, then the dependability will be very high. If the impact on incorrectly set attributes by the administrator will not result in severe security failures, then the dependability values will be low.

The management values will be statically set for each role in a subsystem, i.e. an administrator is evaluated according to his skill. All functions that are administrated by that person will yield the same effect on the dependent components.

Components that are dependant on any kind of security management are marked with an "*M*" in the last column of table 3 in the Appendix.

FPR – Privacy

The 4 families in this class address:

- discovery of an individual's identity by others (FPR_ANO, FPR_PSE)
- association with actions of an individual's identity by others (FPR_UNL, FPR_UNO).

This class is used for anonymity and identity protection.

This class might be of no interest if the users only are regarded as part of an organization, and therefore does not need their privacy to be protected.

FPT - Protection of the TSF

The 16 families in this class address:

- testing of the TSF mechanisms and data (FPT_AMT, FPT_TST)
- physical and anti-tamper protection of the TSF mechanisms and data (FPT_PHP)
- secure TSF data transfer of the TSF mechanisms and data (FPT_ITA, FPT_ITC, FPT_ITI, FPT_ITT, FPT_RPL, FPT_TDC, FPT_TRC)
- failure and recovery of the TSF mechanisms and data (FPT_RCV, FPT_FLS)
- state and timing of the TSF mechanisms and data (FPT_SSP, FPT_STM)

- reference mediation and domain separation of the TSF mechanisms and data (FPT_RVM, FPT_SEP).

This class is used to protect TSF data when controlling access to information (together with FMT and FDP), monitoring loss of TSF data integrity when detecting security events (together with FAU and FDP), and protecting transmitted TSF data in cryptographic protection (together with FCS and FPT).

FRU - Resource Utilisation

The 3 families in this class address:

- availability of resources (FRU_FLT)
- prioritizing of resources (FRU_PRS)
- allocation of resources (FRU_RSA).

This class is used for controlling use of resources, thus enforcing availability in the system.

FTA - TOE Access

The 6 families in this class address:

- attributes of a user session (FTA_LSA, FTA_TAB, FTA_TAH)
- establishment of a user session (FTA_MCS, FTA_SSL, FTA_TSE).

This class is used for controlling access to systems, with help from FIA and FTP.

FTP - Trusted Path / Channels

The 2 families in this class address:

- trusted communication paths between users and the TSF (FTP_TRP)
- trusted communication channels between the TSF and other trusted IT products (FTP_ITC).

This class is used for controlling access to systems, with help from FIA and FTA.

5.3 Security Evaluation of CC SFs

This section explains and argues on how to interpret the evaluated Security Functions.

A security function of CC often affects other security functions in some way. Some may be specified in CC as being dependent on other functions, and thus their security values will depend on these functions. However, there can also exist a more subtle dependency, not specifically stated in the CC. This might depend much on the situation, for example, circumstances regarding the components or system, that decides whether functions are dependable or not.

Interpretation of securability values

Once the components have been evaluated according to the CC Security Functions, the question arises about what the values for the SFs really mean in terms of IT security.

- One possible solution is to present these values as a result. Although they may be hard to interpret, they will be more precise than any other value.
- Another solution is to traverse the values for the SFs upwards, yielding results for the more general security functions, and finally at the top level there will be measurable security values for the 11 classes of CC SFR.
- A third solution is to translate the values into a more widely used terminology. The two most used categorizations of security are CIA and PDR. They cover entirely different aspects of security, the former is on how information assets may be compromised and the latter covers abilities required to maintain the system security.

  Which of the CIA-categories CC components and families are judged to belong to is marked in the third column of table 3. How the SFRs map to CIA is also visualized in Figure 14. The table and the figure are found in the Appendix. Which of the PDR-categories CC components and families are judged to belong to is marked in the fourth column of table 3. How the SFRs map to PDR is also visualized in Figure 15. The table and the figure are found in the Appendix. Prevent is further divided into storage (S), transportation (T) or execution (E) to specify the area of prevention. If applicable, this is marked in the fifth column of table 3 in appendix.

  Thus, the different SFR families and classes have been analyzed and categorized both according to the CIA and the PDR terminologies. Some families belong to more than one category. This categorization has also been done on the component level and for those families where the category of the components are not the same, the category of the family is either a combination of them, or the one of the categories that best fits to the family. The purpose of the categories is to reflect what characteristics that are applicable or valid for each family, and perhaps even component.

There is really no restriction on how to represent the security values. One of the strategies discussed above may be used, or any combination of them if that is found useful. Sometimes several different kinds of security values are needed for the same evaluation in order to cover different aspects of IT security.

There should be a possibility to multiply the values with a weighting-matrix in order to specify properties in the system that are more important (or less important) for the specific evaluation. This matrix has values ranging from zero (>0) to one (1) or a NULL-value. A one means that the property is fully regarded, while lower values means the property is proportionally less regarded. A NULL-value means that the selected security function should be entirely neglected, (e.g. a matrix filled with only ones has no effect to the resulting security values whatsoever).

Calculations of securability values

The following steps explain how to reach meaningful securability values once the evaluation of a system component according to CC SFs has been made.

1. Choose which of the above explained way(s) to represent the security values.

2. Calculate mean values of the above chosen type(s) for every family.

3. If there are CC components that should be prioritized before others, their security values should be multiplied with a weighting matrix to reflect this priority as explained above.

4. NULL-values have no effect whatsoever during the calculations and should simply be ignored.

5. Calculate mean values for every class, or corresponding concept depending on the chosen representation.

Example 1

To exemplify the steps above, security values for one of the 11 CC classes are calculated. The system component for this example is the Sony FeliCa Contactless Smart Card (Sony, 2002). How the security values are estimated for this component is explained in example 2 in section 5.4. For those families which have more than one component (according to table 3 in appendix), the value of the family is calculated as the mean value of the components.

The resulting security value for the class FPT (Protection of the TOE Security Functions) is then calculated as the mean value of the 16 components (grey rows in Table 1). The security value for the specific class would be:

$$\frac{0.95+0.92+0.85+0.80+0+0+0+0.91+0+0+0}{11} \approx 0.403.$$

If the resulting security value would be better to calculate according to CIA, first check the fourth column of Table 1 to see what category each component and family belong to. Note that one family that represents more than one category is divided equally among the categories. The three security values for the FPT-class of the smart card would according to Table 1 be the following:

$$C: \frac{0.85+\frac{1}{2}\cdot 0+\frac{1}{3}(0+0+0)}{1+\frac{1}{2}+\frac{3}{3}} = 0.34$$

$$I: \frac{0.80+0.91+0+\frac{1}{2}\cdot(0.95+0.92+0+0)+\frac{1}{3}(0+0+0)}{3+\frac{4}{2}+\frac{3}{3}} \approx 0.44$$

$$A: \frac{\frac{1}{2}(0.95+0.92+0)+\frac{1}{3}(0+0+0)}{0+\frac{3}{2}+\frac{3}{3}} \approx 0.37$$

Note that compared to the general security value above, the confidentiality (C) and the availability (A) security values are worse. However, the integrity (I) value is better than the general security value.

Table 1: Security values for components in grey columns, and families in white columns, of the FTP-class (Protection of the TOE Security Functions).

Component	Component description	Value	CIA
FPT_AMT	Underlying abstract machine test	0.95	IA
FPT_FLS	Fail secure	0.92	IA
FPT_ITA	Availability of exported TSF data	NULL	A
FPT_ITC	Confidentiality of exported TSF data	0.85	C
FPT_ITI	Integrity of exported TSF data	0.80	I
FPT_ITI.1	Inter-TSF detection of modification	0.80	I
FPT_ITI(2)	Correction of modified Inter-TSF data	NULL	I
FPT_ITT	Internal TOE TSF data transfer	0	CI
FPT_ITT.1	Basic internal TSF data transfer protection	0	CI
FPT_ITT(2)	Transfer separation of TSF data	NULL	CI
FPT_ITT.3	TSF data integrity monitoring	0	I
FPT_PHP	TSF physical protection	0	CIA
FPT_PHP.1	Passive detection of physical attack	NULL	CIA
FPT_PHP(2)	Notification when detection of physical attack	NULL	CIA
FPT_PHP.3	Resistance to physical attack	0	CIA
FPT_RCV	Trusted recovery	0	IA
FPT_RCV.1*	Recovery	0	IA
FPT_RCV(3)	No undue loss after recovery	NULL	IA
FPT_RCV.4	Function recovery	NULL/0.9	IA
FPT_RPL	Replay detection	0.91	I
FPT_RVM	Reference mediation	0	CIA
FPT_SEP	Domain separation	0	CIA
FPT_SEP.1	TSF domain separation	0	CIA
FPT_SEP.2	SFP domain separation	NULL	CIA
FPT_SEP.3	Complete reference monitor	0	CIA
FPT_SSP*	State Synchrony Protocol	NULL	IA
FPT_STM	Time stamps	NULL	CIA
FPT_TDC	Inter-TSF TSF data consistency	NULL	I
FPT_TRC	Internal TOE TSF data replication consistency	NULL	I
FPT_TST	TSF self test	0	I

5.4 Map CC Security Functions to Evaluated Component Characteristics

This section explains the process of evaluating the system components according to their security properties and how this evaluation relates to the security functions of Common Criteria.

The appropriate security functions that are desired for a specific component could be found reading a PP or ST. However, these documents do not give any information about the securability values of the SFs for that component. The reality is that most of these, for IT security so essential security characteristics, are hard to evaluate. Even basic security methods that are commonly used today, and regarded as having a high security assurance, may in fact not be reliable at all. Too many aspects of deployed measures

are dreamed up in the absence of any empirical evidence (Smith, 2003). For example, one commonly used security method is to freeze the accounts of the users when an incorrect password has been entered three times in a row. Where is the analysis that guarantees the soundness of the exact value three (3) for this parameter? Could it instead be that the number ten (10) is almost as secure, but gives the administrator more time for relevant security work, resulting in a better overall security for the system environment?

A way to decide the correct security values for components could be to practically test them (e.g. penetration tests for firewalls). However, testing could be difficult to perform objectively and independently. Another way could be to fill in surveys with multiple-choice questions regarding the system. The accuracy of the evaluation will of course suffer, but the query forms will be easy to create and the results easy to fill in and evaluate. The ideal way is of course if all security-relevant characteristics could be found and measured for all of the components, which will result in an objective and just securability value for each evaluated component.

There are four different smaller steps that together form this mapping of SFs to a component.

1. Select those SFs that are relevant for the component. Here a PP may be of good use.
2. Add those SFs that the SFs selected in step 1 are depending on. Those SFs that are not regarded at all are given the NULL-value, as they are not considered important or valid for the given component.
3. Determine which SFs from step 2 that actually exist and are covered in the specific component. Here an ST may be of good use, if one exists for the component. Those SFs from step 2 that are not covered in step 3 are assigned the zero value to signify that here resides a security void.
4. Evaluate all existing SFs from step 3 and assign values, in the range 0 to 1, representing the strength of the implementation of the SF.

It is important for the security of a distributed information system that components with one or more functions containing a zero are combined with components that cover these gaps, resulting in better values for those security functions. If a system with zero-valued functions is put into use, there will be substantial security vulnerabilities in that system.

Observe that a component may also affect the information system negatively. For example, if some really time-consuming methods are used to verify the identification of a user, the availability of the entire information

system will worsen. Protections may cause other functionality to fail, or at least to make the actual work harder and more time-consuming to perform. Another aspect of this problem that is focusing on availability was observed by John Ousterhout, who claimed that security was *"anti-CS"* (computer science) because it was getting in the way of people getting things done (Smith, 2003).

Example 2

As an example on how to map SFs to a component, we look at the FPT class (Protection of the TSF) for the same Sony FeliCa Contactless Smart Card (Sony, 2002), as in Example 1, following the four steps introduced above.

According to a PP (NIAP, 2001) and some additional reasoning about smart cards (Gollmann, 1999), the relevant SFs were chosen to be the ones marked in the first two columns of Table 2.

After checking for dependencies in (CC, 1999), we add TST, since RCV.1 is dependant on that component. Also, TST depends on AMT, and ITT.3 on ITT.1, but these two components already exist in the set. When summarizing, the relevant SFs for a smart card turned out to be the following: AMT, FLS, ITC, ITI.1, ITT.1, ITT.3, PHP.3, RCV.1, RPL, RVM, SEP.1, SEP.3 and TST.

When the Smart Card is evaluated according to these requirements, the ST indicates the SFs of the product, as shown in column three in Table 2.

Null- and zero-values are now easily determined when comparing what SFs the smart card is supposed to contain to what SFs it actually do contain. The security values of other SFs, ranging from above zero to one are harder to estimate. For the sake of this example, these values were, without any further proofs or motivations, found to be the ones shown in the last column of Table 2.

Table 2: Lists all components of the FPT-class, in which papers they were found to be relevant for the smartcard-component and their estimated security values. Note that some components are merged according to 5.2.

PP	Gollman	ST	Component	Component description	Value
X	X	X	FPT_AMT	Underlying abstract machine test	0.95
	X	X	FPT_FLS	Fail secure	0.92
			FPT_ITA	Availability of exported TSF data	NULL
	X	X	FPT_ITC	Confidentiality of exported TSF data	0.85
X		X	FPT_ITI.1	Inter-TSF detection of modification	0.80
			FPT_ITI(2)	Correction of modified Inter-TSF data	NULL
X	X		FPT_ITT.1	Basic internal TSF data transfer protection	0
			FPT_ITT(2)	Transfer separation of TSF data	NULL
X			FPT_ITT.3	TSF data integrity monitoring	0

PP	Gollman	ST	Component	Component description	Value
			FPT_PHP.1	Passive detection of physical attack	NULL
			FPT_PHP(2)	Notification when detection of physical attack	NULL
X	X		FPT_PHP.3	Resistance to physical attack	0
X			FPT_RCV.1*	Recovery	0
			FPT_RCV(3)	No undue loss after recovery	NULL
		X	FPT_RCV.4	Function recovery	NULL/0.9
X		X	FPT_RPL	Replay detection	0.91
X			FPT_RVM	Reference mediation	0
X			FPT_SEP.1	TSF domain separation	0
			FPT_SEP.2	SFP domain separation	NULL
X			FPT_SEP.3	Complete reference monitor	0
			FPT_SSP*	State Synchrony Protocol	NULL
			FPT_STM	Time stamps	NULL
			FPT_TDC	Inter-TSF TSF data consistency	NULL
			FPT_TRC	Internal TOE TSF data replication consistency	NULL
(X)			FPT_TST	TSF self test	0

5.5 Evaluation of Component-Based Systems

This section raises questions regarding how evaluated components can be combined to enable evaluations of subsystems.

A problem exists in the fact that the CC functions are assembled by combinations of system components. Thus, it is a difficult task to single out exactly which CC functions a specific system component affect. There is also a problem in deciding how one system component affects the others, since they could be dependent on each other.

The larger the system to be evaluated, the more complex the evaluation becomes, since the security evaluation of the system is a complex combination of the evaluations of all of the components. It should be possible to “zoom in” on a specific part of the system to get a more accurate view of this part and its security values. It should also be possible to “zoom out” from a system focus in order to get a better overview of the entire system.

To solve these issues, powerful means of describing the complex combinations of system components are needed. The system components and how they interact could be modeled using a graph with nodes representing the system components, with assigned security values, and vertices representing the relations between the components. Different graphs could be used for different scopes.

6. Conclusions

The networked world of today demands development of efficient IT security solutions and products. To be able to assess the quality of these IT security products, as well as the security in large distributed information systems, some sort of estimation or evaluation has to take place. This is required in order to produce a measure, and measures are needed to rate security products as well as information systems, and be able to compare different products or systems. Unfortunately, few approaches to solve these problems have been proposed and the viability of those proposed is limited. This report describes an ongoing effort to find methods for security evaluation of distributed information systems.

As a first step, a common terminology and understanding of what to be assessed and measured is needed. In this report, three different scopes of security evaluation have been introduced. Theses scopes, together with the aspects of security evaluation introduced in the security evaluation framework, will facilitate consistent categorization of security evaluation approaches. However, being a first step, it will not directly result in any viable security metrics.

The Security Evaluation Framework establishes foundation for the development of security evaluation methods and security-focused system modeling techniques. The framework divides the evaluation into several steps, called modules. Thus, evaluation methods and modeling techniques can be specified in a flexible and manageable way. For example, the component concept, which denotes the system or system component under evaluation, is partitioned into the five aspects technological, organizational, individual, operational, and environmental, stating the need to handle all of these aspects during security evaluations. Evaluation methods and modeling techniques may ignore some of these aspects, but then that will become an explicit limitation of the corresponding approach. Arguments may arise whether the different problem aspects, covered by the framework, easily can be divided and handled by separate modules. Currently, there is nothing that contradicts this approach, but because of the, at this point, abstract state of the framework, there are no proofs of the opposite. Moreover, it is possible that the framework makes the problem space too wide, and that a more narrow and restricted problem space would yield better results.

The framework is described with an ontology. The ontology structure gives a dynamical and versatile way to structure all the objects in the framework. The ontology divides the objects into taxonomy-like partitions, and also helps in defining a common terminology. It also divides and partitions the objects of the framework more strictly than, for example a tree-structure

does, which gives a more precise and detailed structure. One additional advantage of the ontology is that it helps in defining a common terminology.

A security evaluation method considering technical aspects of system components is proposed. It uses the Security Functional Requirements (SFR) of the Common Criteria as a basis. The securability of system components is evaluated by mapping security values to the SFR representing the specific components. A further specification of the security values can be obtained via the provided mappings from SFR to the security categorizations CIA and PDR. The evaluation method consists of four steps:

1. The securability values and a simplified metric that are needed for the purpose of the evaluation are introduced. The values were given a probabilistic meaning, but the exact meaning of this probability value is not specified, since only the endpoints of the metric have an associated meaning.

2. The security functional requirements of the Common Criteria were adopted as a base for the evaluation. There may exist better bases for the evaluation than the SFR. Still, CC has undergone an extensive development and is widely used and renowned. Thus, it has been chosen as the starting point for this effort.

3. The values of the evaluated security functions can be used differently. One approach is to calculate the values for the extensive eleven SFR-classes. Alternatively, a thorough mapping has been made that relates the SFR to CIA and PDR respectively. A weighting matrix is proposed, enabling prioritization of different aspects of the security. The approach is illustrated by an example based on a smart card.

4. How component characteristics should be reflected in the security functional requirements of CC is not straightforward. Sometimes PPs and STs can be used to extract which functions that are deemed relevant for certain products. An algorithm of the process is defined and exemplified, using a Smart Card. One problem is that even if a PP or an ST might state what functions that are needed for a product, they do not rate those functions. The actual rating of the security functions is the most difficult part of this step and should benefit from further formalization.

The proposed method for security evaluations is limited to system components. To achieve methods that are practical in an operative environment, system-wide evaluations have to be handled. This is a complex issue, as discussed in section 5.5, due to the intricate relations between the components in a distributed information system.

Bibliography

ACSA (2002), *Proc. Workshop on Information Security System Scoring and Ranking*. Applied Computer Security Associates, http://www.acsac.org/measurement/proceedings/wisssr1-proceedings.pdf

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

Alberts, D. S., Garstka, J. J. & Stein, F. P. (1999). Network Centric Warfare: Developing and Leveraging Information Superiority. Second edition (Revised). CCRP Publication Series.

Anderson R. (1993), *Why Cryptosystems fail*, 1st Conference – Computer & Comm. Security '93.

Apostal, D., Foote-Lennox, T., Markham, T., Down, A., Lu, R., O'Brien, D. (2001) Checkmate network security modelling. *Proceedings of DARPA Information Survivability Conference & Exposition II, 2001*. DISCEX '01. Vol. 1, pp. 214-26

Baskerville R (1993) Information systems security design methods: implications for information systems development. ACM Computing Surveys 25: 375-414.

Bottomly, R. (2002), *CC Part 2: Security Functional Requirements*.

CC (1999). *Common Criteria for Information Technology Security Evaluation. Part 1: Introduction and general model, Part 2: Security functional requirements, Part 3: Security assurance requirements. Version 2.1*, August 1999. Visited 2002-05-31 at http://www.commoncriteria.org/cc/cc.html

Coulouris, G., Dollimore, J., & Kindberg, T. (2001). *Distributed Systems– Concepts and Design, third ed.*, Addison-Wesley.

Donner, M. (2003), *Toward a Security Ontology*, IEE Security & Privacy, May/June 2003, pp 6-7.

Eloff & von Solms (2000), Information Security Management: An Approach to Combine Process Certification And Product Evaluation, Computers & Security, Volume 19, Issue 8, 2000, pp 698-709.

Gollmann, D. (1999). *Computer Security*. John Wiley & Sons.

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

Hunstad, A. & Hallberg, J. (2002). Design for securability - Applying engineering principles to the design of security architectures. ACSA workshop on the application of engineering principles to system security design. Boston, Nov. 6-8, 2002. Linköping, FOI 2002, 8 p. (FOI-S--0721--SE)

Hunstad, A. & Hallberg, J. (2002b), *Modeling of Distributed Systems Focusing on IT Security Aspects*, FOI-R—0712-SE, FOI, Linköping, Sweden.

Hutchinson W & Warren M (2000) Using the viable systems model to develop an understanding information system security threats to an organisation. Proceedings of the 1st Australian Information Security Management Workshop, Deakin University, Geelong, Australia.

ITAA, Information Technology Association of America, http://www.itaa.org (acquired 22 October 2003).

Jonsson, E. & Olovsson, T. (1997). A Quantitative Model of the Security Intrusion Process Based on Attacker Behavior. *IEEE Transactions on Software Engineering*. vol. 23. no. 4.

Jøsang, A (1996). *The right type of trust for distributed systems*. In Proceedings of the 1996 New Security Paradigms Workshop, ACM.

Karyda M, Kokolakis S & Kiountouzis E (2001) Redefining information systems security: viable information systems. Proceedings of the IFIP TC11 16th International Conference on Information Security (IFIP/SEC'01), Paris, France, p 453-468.

Kruger, R. & Eloff, J. (1997). A Common Criteria framework for the evaluation of Information Technology systems security. *IFIP TC11 13 international conference on Information Security (SEC '97)*. Copenhagen, Denmark. Chapman & Hall.

NIAP, National Information Assurance Partnership (2001), *Validation Report Smart Card Protection Profile (SCPP)*, Version 3.0, CCEVS-VR-01-0007, October 2001.

Noy, N. F., McGuiness, D. L. (2000), *Ontology Development 101: A Guide to Creating Your First Ontology*, KSL 01-05, Stanford University, 2001.

Object Management Group (2001). *OMG – Unified Modeling Language, v1.4*. September 2001.

Olivier, M. (2001). Towards a configurable security architecture. *Data & Knowledge Engineering*, Volume 38, Issue 2, August 2001, pp. 121-145.

Olthoff, K. (2000). Thoughts and Questions on Common Criteria Evaluations. 23rd National Information Systems Security Conference.

Schudel, G. & Wood, B. (2000). Adversary Work Factor as a Metric for Information Assurance. *Proceedings of the New Security Paradigms Workshop*. Cork, Ireland, Sep. 18-22, 2000.

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

Shapiro, J. S. (2003), *Understanding the Windows EAL4 Evaluation*, IEEE Security & Privacy, February 2003, pp 103-105.

Siponen, M. (2002) Designing Secure Information Systems and Software. Critical evaluation of the existing approaches and a new paradigm. Dept Information Processing Science and Infotech, Oulu, University of Oulu, Oulu 2002 A 387, ISBN 951-42-6789-3.

Smith, S.W. (2003), *Humans in the Loop: Human-Computer Interaction and Security*, IEEE Security & Privacy, May/June 2003, pp 75-79.

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

Sony Corporation (2002), *FeliCa Contactless Smart Card RC-S860*, EAL4, March 2002.

Wang, C. and Wulf, W. (1997). A Framework for Security Measurement. Proceedings of the National Information Systems Security Conference, Baltimore, MD, pp. 522-533, Oct. 1997.

Ware, W. H. (1997), New vistas on info-system security, Chapman & Hall.

Whitmore, J.J. (2001). A method for designing secure solutions. IBM Systems Journal, Armonk 2001, Volume 40, Issue 3.

Appendix A

Table 3: Revised CC structure (according to 5.2) including categorization of class, family or component. The classes are coloured dark grey, the families are light grey and the components are white. The meanings of the id and the last four columns are explained in the text.

ID	Descriptive Name	CI	PD	ET	M
FAU	Security audit				
FAU_ARP	Security audit automatic response	CIA	R		MA
FAU_GEN	Security audit data generation	CIA	D		
FAU_GEN.1	Audit data generation	CIA	D		
FAU_GEN.2	User identity association	CIA	D		
FAU_SAA	Security audit analysis	CIA	D		MA
FAU_SAA.1	Potential violation analysis	CIA	D		
FAU_SAA.2	Profile based anomaly detection	CIA	D		
FAU_SAA.3*	Attack heuristics	CIA	D		
FAU_SAR	Security audit review	CIA	D		
FAU_SAR.1	Audit review	CIA	D		MA
FAU_SAR.2	Restricted audit review	C	P	E	A
FAU_SAR.3	Selectable audit review	CIA	D		A
FAU_SEL	Security audit event selection	CIA	D		MA
FAU_STG	Security audit event storage	IA	PDR	S	
FAU_STG.1	Protected audit trail storage	IA	PDR	S	
FAU_STG(2)	Guarantees of audit trail storage	A	P	S	M
FAU_STG.3*	Prevention of audit data loss	A	PDR	S	MA
FCO	Communication				
FCO_NRO*	Non-repudiation of origin	I	P	E	MA
FCO_NRR*	Non-repudiation of receipt	I	P	E	MA
FCS	Cryptographic Support				
FCS_CKM	Cryptographic key management	CI	P	T	MA
FCS_CKM.1	Cryptographic key generation	CIA	P	T	MA
FCS_CKM.2	Cryptographic key distribution	CIA	P	T	MA
FCS_CKM.3	Cryptographic key access	CIA	P	T	MA
FCS_CKM.4	Cryptographic key destruction	CIA	P	T	MA
FCS_COP	Cryptographic operation	CI	P	T	A
FDP	User data protection				
FDP_ACC*	Access control policy	CIA	P	E	
FDP_ACF	Access control functions	CIA	P	E	MA
FDP_DAU	Data authentication	I	P	E	MA
FDP_DAU.1	Basic data authentication	I	P	E	MA
FDP_DAU(2)	Identity of guarantor of data	I	P	E	MA
FDP_ETC	Export to outside TSF control	CIA	P	T	
FDP_ETC.1	Export of user data without security attributes	CIA	P	T	A
FDP_ETC.2	Export of user data with security attributes	CIA	P	T	MA
FDP_IFC*	Information flow control policy	CIA	P	E	
FDP_IFF	Information flow control functions	CIA	P	E	
FDP_IFF.1*	Security attributes	CIA	P	R	MA
FDP_IFF.5*	No illicit information flows	CA	DR		A
FDP_IFF.6	Illicit information flow monitoring	CA	D		MA
FDP_ITC	Import from outside TSF control	CIA	P	T	MA
FDP_ITC.1	Import of user data without security attributes	CIA	P	T	MA
FDP_ITC.2	Import of user data with security attributes	CIA	P	T	MA
FDP_ITT	Internal TOE transfer	CI	P	T	MA
FDP_ITT.1	Basic internal transfer protection	CI	P	T	MA

ID	Descriptive Name	CI	PD	ET	M
FDP_ITT(2)	Internal transfer separated by attribute	CI	P	T	MA
FDP_ITT.3*	Integrity monitoring	I	D		
FDP_RIP*	Residual information protection	C	P	S	M
FDP_ROL*	Rollback	I	P	E	MA
FDP_SDI	Stored data integrity	I	D		
FDP_SDI.1	Stored data integrity monitoring	I	D		A
FDP_SDI(2)	Action due to loss of stored data integrity	I	DR		MA
FDP_UCT	Inter-TSF user data confidentiality transfer protection	C	P	T	A
FDP_UIT	Inter-TSF user data integrity transfer protection	I	D		A
FDP_UIT.1	Data exchange integrity	I	D		A
FDP_UIT.2*	Data exchange recovery	I	R		A
FIA	Identification and authentication				
FIA_AFL	Authentication failures	CIA	R		MA
FIA_ATD	User attribute definition	CIA	P	E	M
FIA_SOS	Specification of secrets	CIA	P	E	MA
FIA_SOS.1	Verification of secrets	CIA	P	E	MA
FIA_SOS.2	TSF Generation of secrets	CIA	P	E	MA
FIA_UAU	User authentication	CIA	P	E	
FIA_UAU.1*	Timing of authentication	CIA	P	E	MA
FIA_UAU.3	Unforgeable authentication	C	DR		A
FIA_UAU.4	Single-use authentication mechanisms	CIA	P	E	A
FIA_UAU.5	Multiple authentication mechanisms	CIA	P	E	MA
FIA_UAU.6	Re-authenticating	CIA	P	E	MA
FIA_UAU.7	Protected authentication feedback	C	P		
FIA_UID*	User identification	CIA	P	E	MA
FIA_USB	User-subject binding	CIA	P	E	MA
FMT	Security management				
FMT_MOF	Management of functions in TSF	CIA	P	E	MA
FMT_MSA	Management of security attributes	CIA	P	E	
FMT_MSA.1	Management of security attributes	CIA	P	E	MA
FMT_MSA.2	Secure security attributes	CIA	P	E	A
FMT_MSA.3	Static attribute initialisation	CIA	P	E	MA
FMT_MTD	Management of TSF data	CIA	P	E	
FMT_MTD.1	Management of TSF data	CIA	P	E	MA
FMT_MTD.2	Management of limits on TSF data	CIA	P	E	MA
FMT_MTD.3	Secure TSF data	CIA	P	E	A
FMT_REV	Revocation	CIA	P	E	MA
FMT_SAE	Security attribute expiration	CIA	P	E	MA
FMT_SMR	Security management roles	CIA	P	E	
FMT_SMR.1*	Security roles	CIA	P	E	MA
FMT_SMR.3	Assuming roles	CIA	P	E	A
FPR	Privacy				
FPR_ANO	Anonymity	C	P	E	A
FPR_ANO.1	Anonymity	C	P	E	
FPR_ANO(2)	No solicit information while having anonymity	C	P	E	
FPR_PSE	Pseudonymity	C	PD	E	A
FPR_PSE.1	Pseudonymity	C	P	E	
FPR_PSE(2)	Reversability in pseudonymity	C	P	E	
FPR_PSE(3)	Alias used in pseudonymity	C	P	E	
FPR_UNL	Unlinkability	C	P	E	MA
FPR_UNO	Unobservability	C	P	E	
FPR_UNO.1*	Unobservability	C	P	E	MA
FPR_UNO.3	Unobservability without soliciting information	C	P	E	
FPR_UNO.4	Authorised user observability	C	P	E	MA
FPT	Protection of the TOE Security Functions				
FPT_AMT	Underlying abstract machine test	IA	D		MA
FPT_FLS	Fail secure	IA	P	E	A
FPT_ITA	Availability of exported TSF data	A	P	E	MA
FPT_ITC	Confidentiality of exported TSF data	C	P	T	

ID	Descriptive Name	CI	PD	ET	M
FPT_ITI	Integrity of exported TSF data	I	D		
FPT_ITI.1	Inter-TSF detection of modification	I	D		A
FPT_ITI(2)	Correction of modified Inter-TSF data	I	R		MA
FPT_ITT	Internal TOE TSF data transfer	CI	P	T	
FPT_ITT.1	Basic internal TSF data transfer protection	CI	P	T	M
FPT_ITT(2)	Transfer separation of TSF data	CI	P	T	M
FPT_ITT.3	TSF data integrity monitoring	I	D		MA
FPT_PHP	TSF physical protection	CIA	P	S	
FPT_PHP.1	Passive detection of physical attack	CIA	D		A
FPT_PHP(2)	Notification when detection of physical attack	CIA	R		MA
FPT_PHP.3	Resistance to physical attack	CIA	P	S	M
FPT_RCV	Trusted recovery	IA	R		
FPT_RCV.1*	Recovery	IA	R		MA
FPT_RCV(3)	No undue loss after recovery	IA	R		MA
FPT_RCV.4	Function recovery	IA	R		A
FPT_RPL	Replay detection	I	D		MA
FPT_RVM	Reference mediation	CIA	P	E	
FPT_SEP	Domain separation	CIA	P	E	
FPT_SEP.1	TSF domain separation	CIA	P	E	
FPT_SEP.2	SFP domain separation	CIA	P	E	
FPT_SEP.3	Complete reference monitor	CIA	P	E	
FPT_SSP*	State synchrony protocol	IA	P	E	A
FPT_STM	Time stamps	CIA	P	E	MA
FPT_TDC	Inter-TSF TSF data consistency	I	P	T	A
FPT_TRC	Internal TOE TSF data replication consistency	I	P	T	A
FPT_TST	TSF self test	I	D		MA
FRU	Resource utilisation				
FRU_FLT*	Fault tolerance	IA	P	E	A
FRU_PRS*	Priority of service	A	P	E	MA
FRU_RSA*	Resource allocation	A	P	E	MA
FTA	TOE access				
FTA_LSA	Limitation on scope of selectable attributes	CIA	P	E	MA
FTA_MCS*	Limitation on multiple concurrent sessions	A	P	E	MA
FTA_SSL	Session locking	CIA	P	E	MA
FTA_SSL.1	TSF-initiated session locking	CIA	PR	E	
FTA_SSL.2	User-initiated locking	CIA	P	E	
FTA_SSL.3	TSF-initiated termination	CIA	PR	E	
FTA_TAB	TOE access banners	I	P	E	M
FTA_TAH	TOE access history	CIA	D		
FTA_TSE	TOE session establishment	CIA	P	E	MA
FTP	Trusted path/channels				
FTP_ITC	Inter-TSF trusted channel	CIA	P	T	MA
FTP_TRP	Trusted path	CIA	P	T	MA

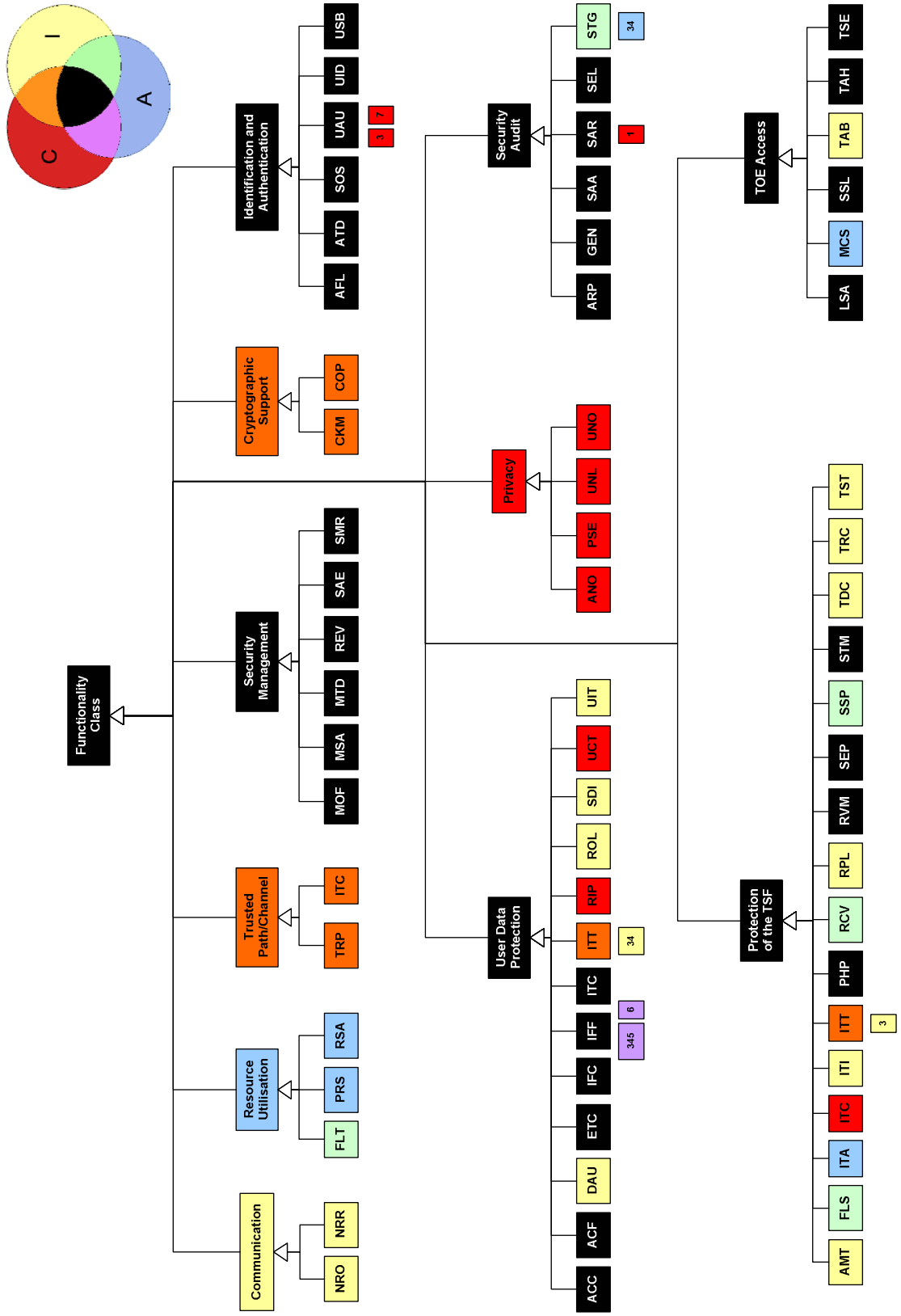


Figure 14: SFR coloured according to CIA. Those components that do not entirely share the view of their family are marked with a different colour than that of their family.

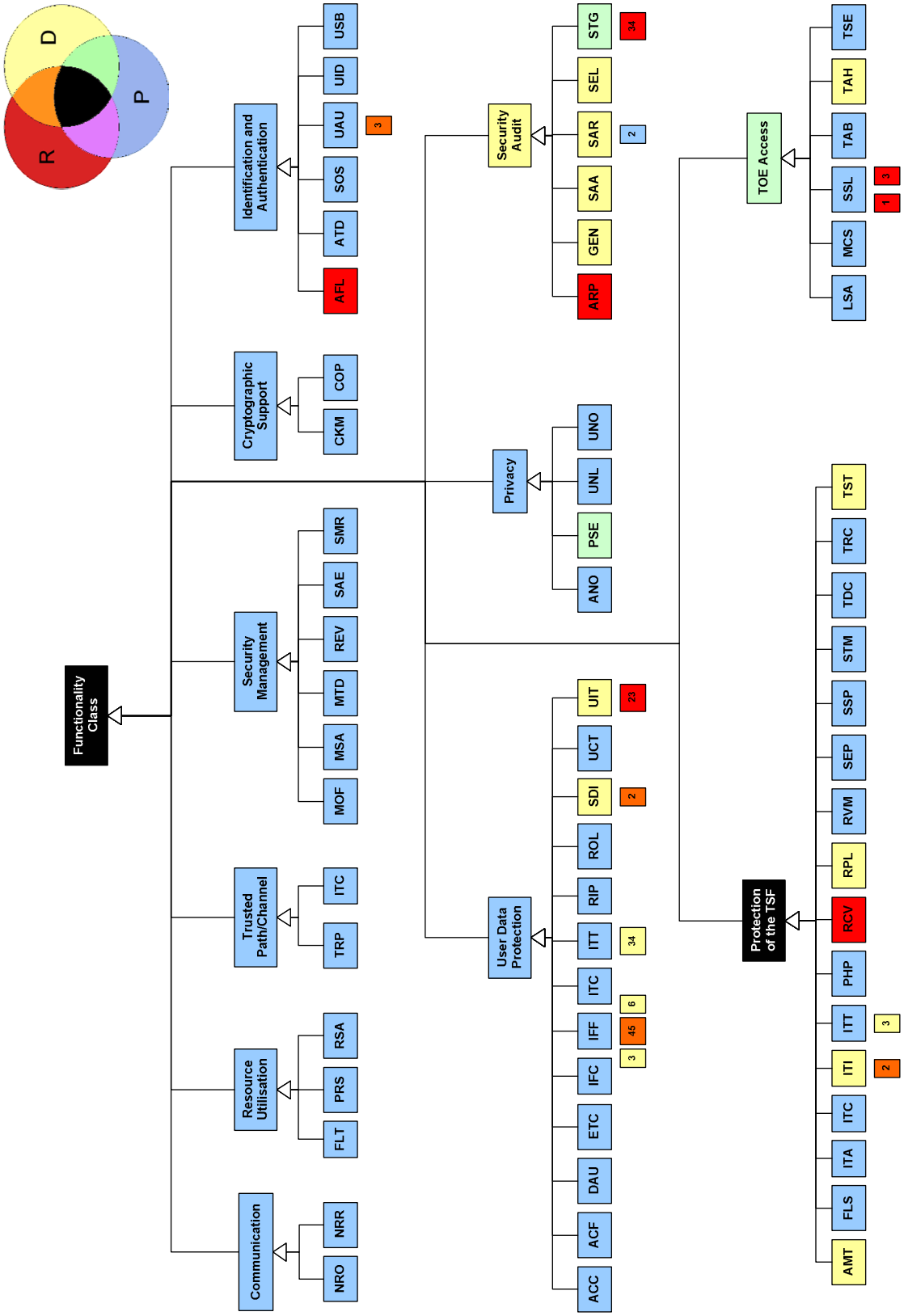


Figure 15: SFR coloured according to PDR. Those components that do not entirely share the view of their family are marked with a different colour than that of their family.