8-6-2011

Designing Security Requirements – A Flexible, Balanced, and Threshold-Based Approach

Yanjun Zuo
University of North Dakota, yanjou.zuo@und.edu

Designing Security Requirements – A Flexible, Balanced, and Threshold-Based Approach

Yanjun Zuo
University of North Dakota
Grand Forks, ND 58201
Email: yanjun.zuo@und.edu

ABSTRACT

Defining security requirements is the important first step in designing, implementing and evaluating a secure system. In this paper, we propose a formal approach for designing security requirements, which is flexible for a user to express his/her security requirements with different levels of details and for the system developers to take different options to design and implement the system to satisfy the user's requirements. The proposed approach also allows the user to balance the required system security properties and some unfavorable features (e.g., performance degrading due to tight control and strong security). Given the importance of social-technical factors in information security, the proposed approach also incorporates economic and organizational security management factors in specifying user's security requirements. We demonstrate the application of our approach with the help of a concrete pervasive information system.

Keywords

Security, design, management, model, requirement analysis.

INTRODUCTION

Security requirement specification is a vital component in designing and implementing secure information systems (Tryfonas, 2007; Siponen, *et al.* 2006). Sound security requirements not only ensure that a system satisfying those requirements will provide secure functions in supporting an organization's mission-critical services but also serve as a guideline for the users to evaluate, verify and ascertain that the system meet their expectations. Given the critical role that security requirements play in information security, it is important to make sure that the security requirements are complete, consistent and analyzable by different actors involved in the development and use of the system (Mellado, *et al*. 2007).

Although the importance of security requirements has been well recognized in the research community, there is still a need for a formal approach for designing security requirements with key features such as flexibility, expressiveness, soundness, and ease of use. Our research in this paper presents a flexible, balanced, and threshold-based approach for designing security requirement. It offers a broader view of security requirements by including the socio-technical aspects of information security in security requirement specifications. With infusion of more new services, technologies, and logical or physical devices to an organization's information environment, no system can be completely secure (Straub and Welke, 1998). The security requirements as discussed in this paper not only include technical requirements but also organizational security management aspects (e.g., physical security - organization premise control, monitoring and auditing).

The proposed approach has several unique features. First of all, it is flexible for the user to express his/her security requirements with different levels of details and for the system developers to take different implementation options to satisfy the user's requirements by considering various trade-offs between security compliance and economic factors (e.g., cost). Threshold selection operators are designed based on the utility theories to allow a subset of security properties to be satisfied before the system is considered satisfactory from a particular perspective. To our best knowledge, this work is the first of its kind in applying threshold operations in designing security requirements.

Secondly, the proposed approach is comprehensive in the sense that it allows the user to balance both security requirements and some (unavoidable) concerns on certain system properties which may conflict with tight security control. In engineering terms, more control means additional system layers. More layers mean slower performance and higher implementation costs. The proposed approach allows the user to require that a system must meet the criteria for some mandatory, critical system security features. In the meantime, it will not have any unfavorable properties that cannot be tolerated.

Thirdly, since the terminologies defined in this research (e.g., security characteristic and security primitive) have a general scope and are not limited to technical domains, a user can specify important social, economic, and managerial security requirements. As we mentioned earlier, given the growing importance of social-technical aspects of information security, the

approach allows the user to incorporate the economic and social factors in designing security requirements and allows flexibility in specifying security requirements based on economic factors and feasibility of technical implementations.

Last but not least, since security compliance is critical in information security, the proposed approach supports "provability" – the system provider can compile and submit a proof to show that their system satisfies the user's security requirements. We have already conducted some preliminary work in automatically constructing a compliance proof to indicate that a system meets the user's criteria specified in the security requirements based on the approach presented in this paper.

The rest of the paper is organized as follows. The next section discusses related work. It is followed by the presentation of the security requirements design approach. The manuscript concludes by summarizing the key techniques and unique features of this approach.

RELATED WORK

A considerable amount of research works have been published on security requirements elicitation, analysis, and engineering. We will only mention a few here. Myagmar, *et al.* (2006) discuss threat modeling as a basis of security requirements. They describe a systematic approach towards threat modeling for complex systems. The concept of risk management is also briefly explained. Oladimeji, *et al.* (2006) propose a goal oriented approach to security threat modeling and analysis by using visual model elements. They introduce the notions of negative softgoals for representing threats and inverse contributions for evaluating design alternatives. An analysis procedure is also provided as a guide to context sensitive selection of countermeasures. Haley *et al.* (2006) define security requirements as the implementation of security goals which constraint the functionality of a system. Fabian *et al.* (2010) present a conceptual framework for gathering, analyzing, and reconciling the functional and non-functional, and security requirements of a system. Their work establishes a clear-cut vocabulary and makes explicit the interrelations between the concepts and notions used in security requirements. Mellado, *et al*. (2007) present the security requirement engineering process to describe how to integrate security requirements into the software engineering process in a systematic and intuitive way. Young, *et al.* (2010) propose a holistic framework that incorporates security into the overall software development process instead of only addressing security during the requirement engineering process. Other approaches commonly used by IT security researchers and practitioners such as attack trees, misuse cases, and UMLSec are mainly to identify threats and security flaws. Our model takes a step further – it focuses on how security requirements after the threats/misuse cases have been identified can be formulated to address those concerns. In this regards, those security approaches and our model complement each other and achieve different objectives.

In the literature, security requirements are also studied from different perspectives. For instance, security requirements are expressed by describing the security mechanisms to be used (ISO/IEC, 1999), security requirements are processed as a kind of non-functional requirements (Devanbu & Stubblebine, 2000), described by how they may be violated (McDermott & Fox, 1999), and defined as constraints on the functions of the system, where they operationalize one or more security goals (Haley, *et al.* 2006). In addition, several security requirement engineering approaches such as the Common Criteria, Secure Tropos, SREP, MSRA, and the methods based on UML have also been developed.

Our research represented in this paper makes a significant contribution to the security requirements literature and to the field of developing secure systems in general by addressing several key issues which are not adequately addressed by the existing works. We present a formal approach for systematic reasoning and specification of the security requirements. Most of the existing security requirements frameworks do not consider potential conflicts between security and other functional and non-functional requirements. As we mentioned earlier, our approach offers several advantages: (1) it is flexible for the user to define security requirements and for the system developers to comply with the requirements; (2) it is cost effective, allowing different plans to define and comply with the security requirements based on economic factors and the feasibility of technical implementations; and (3) it is comprehensive by balancing both favorable and unfavorable system security features.

THE EXAMPLE RFID SYSTEM

An RFID system is used as a running example to illustrate our ideas. We have chosen RFID because it is rapidly becoming an important part of enterprises and its security is getting more and more crucial for the success of this technology in business applications. Although we use an RFID system as an example, our approach is general and can be applied to many other systems since no unique feature of RFID is specifically required to use the approach. The structure of a typical RFID system is shown in Figure 1 (Zuo, *et al.* 2009). The major components include: (1) the radio frequency (RF) subsystem, which consists of readers and tags to perform identification and wireless communications and transactions; (2) the enterprise subsystem, which consists of a backend database and a RFID server. The database contains information such as tag identifications, the secret key shared with each tag, and detailed descriptions about the tagged items. The RFID server

consists of the necessary components to communicate with the readers and to process data acquired from the RF subsystem. The server is also connected to the higher-level business applications within the enterprise networks of an organization.

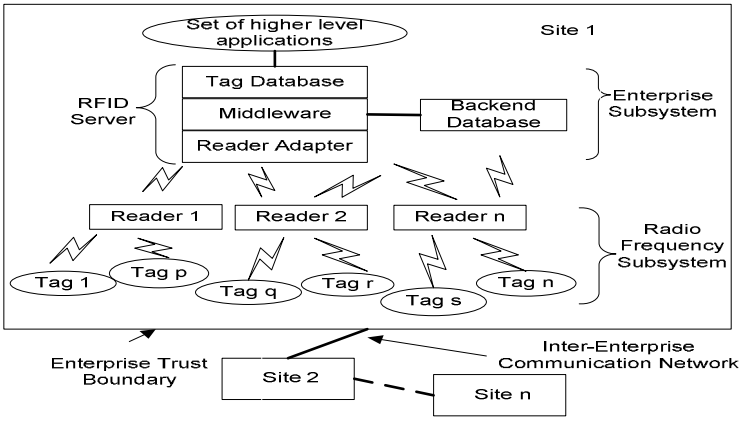

Figure 1: RFID system components

DESIGNING SECURITY REQUIREMENTS

In this section, we first define the key terminologies and then discuss the techniques for designing security requirements.

Terminologies

Security Property Classification and Security Characteristics (SCs)

The security properties of a system can be defined and classified from two technical and semantic aspects. The former focuses on the intrinsic security features of the system and the latter focuses on favorable or unfavorable meanings of the system properties. Some security properties describe the desired features of a system and they express a user's expectations towards those favorable security features. Examples include strong resilience to malicious attacks, solid fault tolerance and robustness, and prompt damage recovery. Other system properties express a user's concerns and wishes to minimize such properties of a system. One example would be system performance degrading. Naturally, tight security control will unavoidably affect system performance. In engineering terms, combining approaches leads to different trade-offs. Unfavorable properties refer to those types of security features that the user wants to limit but it is impossible or too costly to mitigate them.

From the technical aspect, some security properties describe the security features of a system from the same perspective. Logically, they can be grouped into one dimension, called a *security-related characteristic,* or *security characteristic* (*SC*) for short. By its nature, security is a multi-faced concept and a system's security properties can be described from different views. For instance, robustness can be defined as an *SC* for an RFID system, which describes the security features of the RFID system from the perspective of how robust it is to tolerate malicious attacks and/or system failures.

Security Characteristic Hierarchy, Basic Security Characteristic, and Security Primitive

SC is a hierarchical concept - a parent *SC* (more abstract with a broader context) can be refined to multiple sub *SC*s (more specific with a more focused scope). Consider the security characteristic "system robustness" (SC_1) defined for the example RFID system. Three sub *SC*s may be defined for SC_1: "damage masking" ($SC_{1,1}$), "fault tolerance" ($SC_{1,2}$) and "system adaptation" ($SC_{1,3}$). Each of them describes the RFID system's security features from a more specific aspect of system robustness. Technically, each sub *SC* can be further refined to a set of more detailed sub *SC*s until a satisfactory level is achieved. In this way, an *SC* hierarchy is formed from the root *SC*. At the lowest level of the hierarchy, a set of basic *SC*s are obtained, which will not be further refined. In the literature, some research work (Choobineh & Anderson, 2010) also proposes that security requirements are organized in a hierarchical structure where a more abstract higher-level node is broken down into details.

SC refining can be function-based or component-based. For the former, the sub *SC*s are refined based on the technical implications of the parent *SC*. The above example of refining the security characteristic "robustness" (SC_1) is function-based. In a different case, consider the security characteristic "fault tolerance" ($SC_{1,2}$) which describes how an RFID system could

tolerate attacks or system failures in case those faulty conditions cannot be resisted. As we discussed earlier, an RFID system consists of two major parts: enterprise sub-system and RF-subsystem. Therefore, $SC_{1,2}$ can be refined to two sub *SC*s: "RF subsystem fault tolerance" ($SC_{1,2,1}$) and "enterprise subsystem fault tolerance" ($SC_{1,2,2}$). This is an example of component-based SC refinement.

The logical relationships among the sub *SC*s for a parent *SC* is either alternative or conjunctive. For instance, the relationship among $SC_{1,1}$ $SC_{1,2}$ and $SC_{1,3}$ as defined for "robustness" (SC_1) is alternative in the sense that if a system has security features in terms of any one of the three, the system is considered satisfactory in terms of robustness (SC_1). In other words, $SC_{1,1}$ $SC_{1,2}$, and $SC_{1,3}$ can substitute each another. For a conjunctive relationship, the sub *SC*s must be complementary in the sense that they all must be satisfied before the system can be considered as satisfactory in terms of the parent *SC*. For instance, "enterprise subsystem fault tolerance" ($SC_{1,2,1}$) and "RF subsystem fault tolerance" ($SC_{1,2,2}$) must both be satisfied before the RFID system as a whole is considered fault tolerant. Therefore, $SC_{1,2,1}$ and $SC_{1,2,2}$ have a logical "composite" relationship in the context of their parent $SC_{1,2}$ i.e., "system fault tolerance".

As we mentioned earlier, a basic *SC* represents the smallest unit of "aspect" or "perspective" to describe a system's security features. The concrete, specific security properties can be defined from the perspective of each *SC*. We name such a security property as *security primitive* (*SP*) since it represents the most primitive security property to describe a concrete system security feature. For instance, if "redundancy-based damage masking" is refined as a basic *SC* ($SC_{1,1,1}$) in the *SC* hierarchy discussed above, two *SP*s may be defined from this perspective of damage masking: (1) service redundancy; and (2) component redundancy and system partition. We can see that both of the two *SP*s are within the context of $SC_{1,1,1}$ since they both describe the security properties from the perspective of redundancy-based damage masking. More examples of basic *SC*s and their respective *SP*s can be found in Table 1.

Balancing Favorable and Unfavorable System Security Properties

From the semantic perspective, an *SC* is called a *favorable* (*unfavorable*, resp.) *SC* if it describes desirable (undesirable, resp.) security properties of a system. We require that a favorable (unfavorable, resp.) *SC* only contains favorable (unfavorable, resp.) sub *SC*s and security primitives (*SP*s)[1]. A favorable *SC* represents a user's overall expectations towards some desired security features of the system from a particular perspective. Any system to meet the user's requirements must possess those required favorable features. In the meantime, the user may also specify some unfavorable *SC*s to describe potential unwanted features of the system. Those unfavorable features of a system, if any, must not go below a minimum acceptable level in order for the system to be acceptable. In the next section, we use the RFID example to show how the user's security requirements are defined by balancing both favorable and unfavorable system features.

Designing Security Requirements

The major steps of designing security requirements include: (1) specifying the high-level *SC*s that represent the most general and containable perspectives to describe security properties of a system. If the system satisfies the requirements in terms of those *SC*s, it is considered to have a certain level of security; (2) refining each high-level *SC* to a set of more specific sub *SC*s until a desired level of detail is reached and a set of basic *SC*s are defined; and (3) specify the corresponding *SP*s for each basic *SC*. A threshold structure is defined for each basic *SC* to express the user's requirement in terms of the basic *SC* using those *SP*s. In the next two sub sections, we will discuss those steps in detail using the RFID example.

Specifying Security Requirements in Terms of Security Characteristics (SCs)

The basic principles of defining high-level *SC*s include: (1) the defined *SC*s are comprehensive – they cover all the vital perspectives from which the system security properties need to be addressed; (2) the defined *SC*s balanced – both favorable and unfavorable aspects of the system's security-related properties need to be specified; and (3) the defined *SC*s are appropriate – the depth (detail) and breadth (scope) of refining those *SC*s reflect the technical and business requirements of the organization and its mission. In general, if a system satisfies the security properties in terms of those *SC*s, then the system is considered with the desired level of security. For instance, in order for the RFID system to be considered reaching a high level of security, three *SC*s are defined (see Figure 2): "system robustness" (SC_1), "resilience" (SC_2), "intrusion detection and response" (SC_3), and "concerns on system performance degrading" (SC_4). The first three represent favorable *SC*s and the last one represents an undesired *SC*. Those four *SC*s represent the user's high-level vision of a "secure" RFID system.

[1] Technically, one *SC* can always be broken into two *SC*s so that each only contains the sub *SC*s and *SP*s with the same type of semantics (favorable or unfavorable).

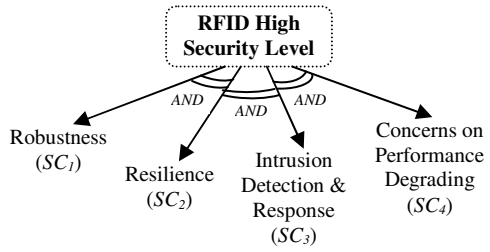

Fig. 2: The High-Level Security Characteristics

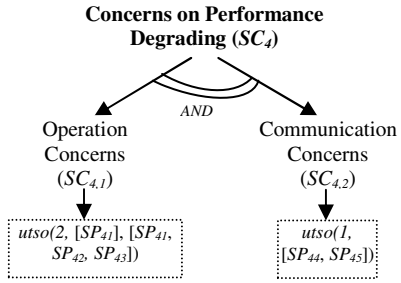

Fig. 3 (a): The Hierarchy of the Security Characteristic "Tolerable Performance Degrading" including the Security Primitives for each Basic *SC*

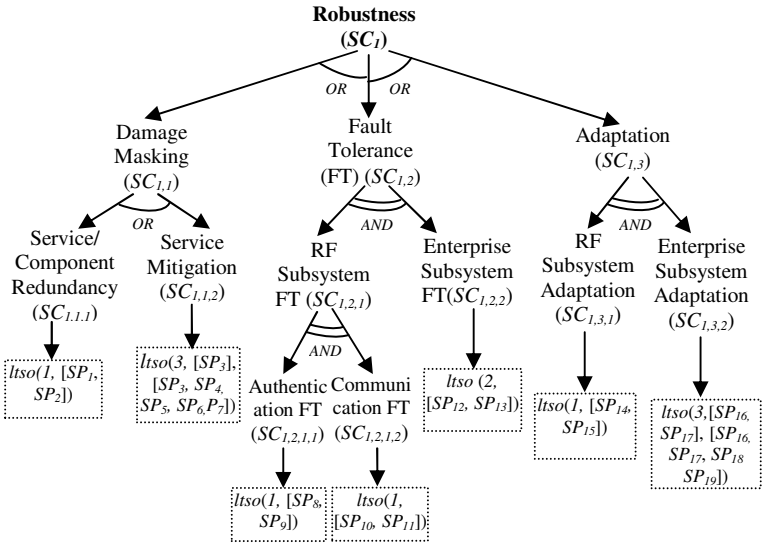

Fig. 3 (b): The Hierarchy of the Security Characteristic "Robustness" including the Security Primitives for each Basic *SC*

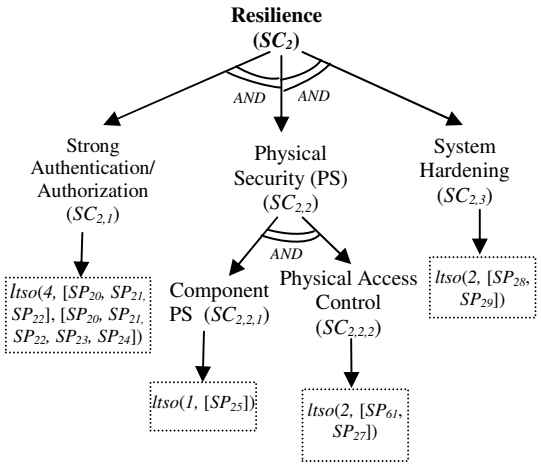

Fig. 3 (c): The Hierarchy of the Security Characteristic "Resilience" including the Security Primitives for each Basic *SC*

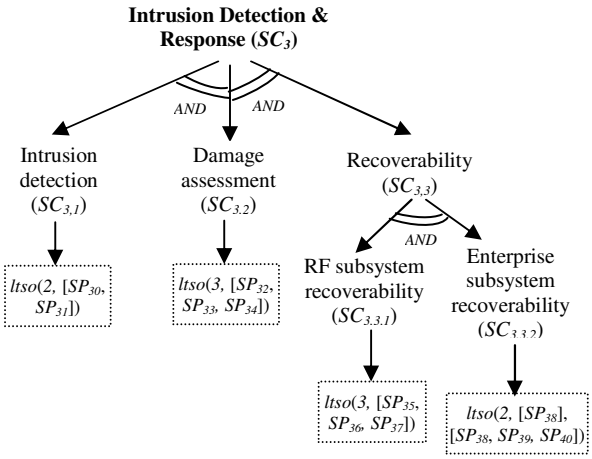

Fig. 3 (d): The Hierarchy of the Security Characteristic "Intrusion Detection & Response" including the Security Primitives for each Basic *SC*

Table 1: Security Primitives (*SP*s) Defined for the Basic Security Characteristics (*SC*s) as Shown in Figure 3 (a)-(d)

Security Primitive (*SP*)	The Basic *SC* To which it Belongs	Explanations of the *SP*
SP_1: Service redundancy	Redundancy based damage masking ($SC_{1,1,1}$)	RFID system's ability to strategically duplicate services to mask damage
SP_2: Component redundancy		RFID system's ability to strategically duplicate critical components and partition the system, and, if necessary, to provide damage masking
SP_3:Process mitigation	Service mitigation ($SC_{1,1,2}$)	RFID system's ability to transfer the processes that support critical services from compromised components to other clean and safe components in face of attacks
SP_4:Component/service distribution ability		RFID system's ability to deploy the critical components and services in a distributed model to avoid single-point failures and to increase system damage masking ability
SP_5:Interoperability		The ability of the RFID system's services/components to interoperate with each other
SP_6:Connectivity		RFID system's ability to maintain its components connected in a hostile environment
SP_7:Scalability		RFID system's ability to spread the services in a large scale to avoid vulnerable points

SP_8: Proxy-based authentication fault tolerance	RF subsystem authentication fault tolerance ($SC_{1,2,1,1}$)	RFID RF subsystem's ability to provide fault tolerable component authentications through proxy service, e.g., a more powerful mediation device conducts authentication with external devices on behalf of the RF components (e.g., RFID tags)
SP_9: Protocol fault tolerance		RFID RF subsystem's ability to provide fault tolerant tag/reader authentication protocols (e.g., robust to desynchronization, signal blocking, bogus data injection attacks)
SP_{10}: Proxy-based communication fault tolerance	Communication fault tolerance ($SC_{1,2,1,2}$)	RFID RF subsystem's ability to conduct fault tolerant inter-components communications through proxy service, e.g., a more powerful mediation device communicates with external devices on behalf of the RF components (e.g., RFID tags)
SP_{11}: Anti-interference		RFID RF subsystem's ability to counter communication interferences by attackers (e.g., single jamming, collision, or blocking)
SP_{12}: Endurability	Enterprise subsystem fault tolerance ($SC_{1,2,2}$)	RFID enterprise subsystem's built-in capability to endure internal and external faults
SP_{13}:Self-healing		RFID enterprise subsystem's ability to promptly repair and recover damages due to attacks or system failures in support of fault tolerance
SP_{14}: Distance sensitive authentication & responses	RF subsystem adaptation ($SC_{1,3,1}$)	RFID RF subsystem's ability to adjust its components' behaviors (for the purposes of authentication & query response) according to the physical distance to the querying entities. For instance, RFID tags may respond differently according to the distance to the querying entities by measuring the signal strengths
SP_{15}: Component reconfiguration		RFID RF subsystems' ability to reconfigure the components to adapt to the changing environment (e.g., an RFID tag can be put into sleep, temporarily deactivated, or working in a degraded mode in a hostile environment)
SP_{16}: Reconfigurability	Enterprise subsystem adaptation ($SC_{1,3,2}$)	The ability of the major components of the RFID enterprise subsystem to be reconfigured
SP_{17}: Context awareness		RFID enterprise subsystem's ability to sense the dynamically changing environment
SP_{18}: Static adaptation		RFID enterprise subsystem's ability to adapt to environment based on pre-defined rules
SP_{19}: Dynamics		RFID enterprise subsystem's ability to automatically reconfigure its major components and/services and to dynamically adapt to the changing environment
SP_{20}: Anti-spoofing	Strong system authentication/ authorization ability ($SC_{2,1}$)	RFID system's ability to resist spoofing and impersonation attacks
SP_{21}: Anti-traffic analysis		RFID system's ability to resist traffic analysis (e.g., communication anonymity)
SP_{22}: Authentication soundness		RFID system's ability to correctly accept legitimate components and reject any non-self or suspicious components
SP_{23}: Time sensitive authentication		RFID system's ability to resist relay attacks by deploying time sensitive authentication protocols and procedures (e.g., a timer is used to measure the response duration)
SP_{24}: Distance bounding authentication		RFID system's ability to resist relay attacks by deploying distance bounding authentication protocols and procedures (e.g., signal strength is used to estimate the querying partner's distance)
SP_{25}: Tamper resilience	Component physical security ($SC_{2,2,1}$)	RFID components' ability to physically protect themselves using mechanisms such as tamper-proof hardware/software, password protection and physical unclonable functions
SP_{26}: Component traceability	Physical access control ($SC_{2,2,2}$)	RFID system's ability to track critical components physically and to search for possible missing components (e.g., RFID tags and readers)
SP_{27}: Environment monitoring and control		RFID system's ability to deploy surveillance devices to monitor its components behaviors and physically control access to critical components in an organization premise
SP_{28}: Software assurance	System hardening ($SC_{2,3}$)	RFID system hardening through sound software engineering practices with solid software design, testing, verification, and analysis
SP_{29}: Vulnerability control		RFID system hardening through threat modeling, vulnerability identification and mitigation, system patching, and update management
SP_{30}: Fault traceability	Intrusion detection ($SC_{3,1}$)	RFID system's ability to detect and trace system abnormal behaviors which may represent the symptoms of malicious attacks and/or system failures in their early stages
SP_{31}: Testability		RFID system's ability to test and verify system abnormal behaviors and conduct the corresponding diagnosis and fault analysis
SP_{32}: Data verifiability	Damage assessment ($SC_{3,2}$)	RFID system's ability to verify data integrity
SP_{33}: Component searching ability		RFID system's ability to search for missing components (e.g., RFID tags and readers) and to verify the security status of the critical components
SP_{34}: Server auditability & impact evaluation		RFID system's ability to audit server status and to perform impact analysis in case that some servers may have been compromised
SP_{35}: Tag reset	RF subsystem recoverability ($SC_{3,3,1}$)	RFID RF subsystem's ability to reprogram, reset, or reconfigure tags
SP_{36}: Reader repairability		RFID RF system's ability to repair RFID readers if they are damaged
SP_{37}: State resynchronization		RFID RF system's ability to resume the states of tags and readers in case of state been desynchronization resulted from malicious actions such as a denial of service attack
SP_{38}: Restorability	Enterprise subsystem recoverability ($SC_{3,3,2}$)	RFID enterprise subsystem's ability to restore damaged components/services to their pre-attack clean states
SP_{39}: Predictability		RFID enterprise subsystem's ability to predict possible causes of damage and/or the techniques used by the attackers in order to quickly recover the compromised components in case of an actual attack case
SP_{40}: Reusability		The ability of the RFID enterprise subsystem's components to be reused to provide similar functions (it is desired that the components are multi-functional and compatible with others)
SP_{41}: Computation accuracy degrading	Operation concerns ($SC_{4,1}$)	User's concern on RFID system's degrading acceptable computation accuracy (given the more resources deployed for security, computation accuracy could be degraded)
SP_{42}: Service consistency concern		User's concern on the acceptable level of service consistency provided by the RFID system
SP_{43}: Resource allocation fairness concern		User's concern on a reasonable level of fairness in resource allocations to different services/functions (since more resources may be dedicated to security)
SP_{44}: Transmission delay concern	Communication concerns ($SC_{4,2}$)	User's concern on RFID data/service transmission delays
SP_{45}: Service availability concern		User's concern on RFID system service availability

To make a balance between the favorable security features and some unfavorable features of a system, the user may require that the system must have at least certain "strong" security properties in terms of some crucial (favorable) *SC*s but at the same time the undesirable security-related features in terms of other non-crucial unfavorable *SC*s are not below a tolerable level. Flexibility is desired to allow for a limited number of non-critical unfavorable features of a system to be accepted as long as the system possesses other significant and critical security features. For the RFID system, it satisfies the user's requirements with a "high level of security" if it meets the following three critical criteria (see Figure 2): (1) it is resilient to most of the malicious attacks (SC_2); (2) it is robust to mask damage and/or tolerate fault if the attacks cannot be resisted (SC_1); and (3) it has strong intrusion detection and incident response ability to recover damage (SC_3). Those three *SC*s represent the user's expectations towards the necessary favorable security features that the RFID system must have. At the same time, the user may require the system's possible performance degrading must not go below a minimum tolerable level represented by SC_4.

After the high-level *SC*s are specified, the user's security requirements can be further refined to different levels of sub *SC*s until a set of basic *SC*s is reached. Each basic *SC* represents the most preliminary perspective (or dimension) to describe the security properties of a system. To illustrate, we show the *SC* hierarchy for each of the three high-level security characteristics $SC_1 - SC_4$ in Figure 3 (a), (b), (c), and (d), respectively (the meaning of each *SP* is explained in Table 1). We call such a hierarchy structure a *security requirement tree*. The relationship of the sub *SC*s within the context of a parent *SC* is represented by an "AND" or "OR" symbol in a security requirement tree, representing the conjunctive and alternative relationship, respectively. As we can see, "resilience" (SC_2) is refined to three sub *SC*s: "strong authentication/authorization" ($SC_{2,1}$), "physical security" ($SC_{2,2}$), and "system hardening" ($SC_{2,3}$). $SC_{2,2}$ is further refined to two sub *SC*s: "component physical security" ($SC_{2,2,1}$) and "physical environment access control & monitoring" ($SC_{2,2,2}$). Finally four basic *SC*s are obtained from SC_2: $SC_{2,1}$, $SC_{2,2,1}$, $SC_{2,2,2}$, and $SC_{2,3}$. In the next sub section, we discuss how the user's security requirement is defined in terms of those basic *SC*s.

Security Requirement Specification in Terms of Security Primitives (SPs)

As we mentioned earlier, a set of security primitives (*SP*s) are defined in the context of each basic *SC*. An *SP* represents a concrete security property to describe the system's security features from a specific perspective of that *SC*. Table 1 shows 45 *SP*s defined for the 16 basic *SC*s for the RFID example. In our approach, the user's security requirements in terms of a basic *SC* are represented by the requirements in terms of the *SP*s defined for the *SC*. More specifically, a threshold selection structure is defined for each basic *SC* to indicate that a subset of *SP*s must be satisfied before the system is considered satisfactory in terms of that *SC*. We first use an example to illustrate the idea and then explain the rationale of developing such a threshold approach.

Consider the basic *SC* "strong authentication/authorization" ($SC_{2,1}$) as represented in Figure 3(c). Five *SP*s are defined for $SC_{2,1}$: "anti-spoofing" (SP_{20}), "anti-traffic analysis" (SP_{21}), "soundness of tag/reader authentication protocols" (SP_{22}), "time sensitive-enabled authentication" (SP_{23}), and "distance bounding-enabled authentication" (SP_{24}). To satisfy the user's security requirements for strong authentication/authorization ($SC_{2,1}$), a system must satisfy SP_{20}-SP_{22} since each of them is critical. But only one of SP_{23} and SP_{24} is required since those two *SP*s represent the complementary or replaceable security features, i.e., the RFID system only needs to have either time sensitive or distance bound authentication protocols to provide strong authentication/authorization to prevent such attacks as relay, replay or man-in-the-middle.

To represent the above idea, a low bound threshold operator $ltso(j, ML_j=[SP_j,\ldots, SP_k], SPL_j=[SP_1, SP_2, \ldots, SP_n])$ is defined for a favorable basic *SC* to indicate that a system must satisfy at least j out of n *SP*s in SPL_j, among which all the *SP*s in ML_j ($ML_j \subseteq SPL_j$) must be satisfied. If that is the case, the system is considered as satisfying the user's requirements in terms of *SC*. Using the notation of threshold selection structure, the requirements for $SC_{2,1}$ as shown in the above example can be represented by $ltso(4, [SP_{20}, SP_{21}, SP_{22}], [SP_{20}, SP_{21}, SP_{22}, SP_{23}, SP_{24}])$.

Theoretical Foundation for Defining Low Bound Threshold Operator

Requiring only a subset of *SP*s for each basic *SC* is based on the Utility Fusion Theory (Zuo & Panda, 2008), which is further based on the Law of Diminishing Marginal Utility (Greene & Baron, 2001). Basically, the law states that the marginal utility of any good or service decreases as the quantity of the good increases. The law is expressed from the viewpoint of a consumer, and is a general principle of economics. The Utility Fusion Theory expresses a similar idea but was developed for the evaluation of an intellectual object (e.g., a piece of information or knowledge in various forms such as a data item or a file) from different perspectives (called dimensions). Basically, the theory indicates that the aggregated utility that an evaluator measures in terms of a dimension as a whole is less than the sum of all the component attribute utilities in that dimension. More specifically, the Utility Fusion Theory is expressed as: *the utility for an attribute dimension D integrated*

from a set of component utilities based on the attributes A_1, …, A_n of D is less than the mathematical addition of those component utilities, i.e., $U(D) \leq \sum_{i=1}^{i=n} U(A_i)$, where U(.) represents an utility function.

Based on the above theories and when applied to our model, for a basic *SC*, incorporating any more *SP*s may only contribute a decreasing margin to the total utility for that *SC*. In other words, those additional *SP*s will not add much more utility. Therefore, as long as the accumulated utility based on the chosen subset of *SP*s is good enough, the user may consider that the security features of the system are acceptable from the viewpoint of that *SC*. Given the complexity of modern systems and variety of usage scenarios, it is desired to have flexibility in defining user's security requirements and to allow the system developers to choose different options to satisfy the user's requirements by meeting the criteria of their chosen *SP*s.

For an undesirable security characteristic *SC'*, a *upper-bound threshold selection operator*, denoted as *utso(i, ML_i=[SP_t, …, SP_q], SPL_i=[SP_1, SP_2, …, SP_m])* is defined, which indicates that the system must not have serious concerns for more than *i* out of *m SP*s defined in SPL_i, among which all *SP*s in ML_i must not be seriously concerned. Essentially, *utso(.)* defines the "most tolerable" upper bound for possible unfavorable features of a system, indicating a "better than minimum" acceptable criteria in evaluating the system's security-related features from the point of view of *SC'*. As we mentioned before, since it is impossible to have a perfectly secure system, it may be necessary to allow for some controlled non-critical unfavorable features of a system to be accepted under the condition that the system possesses other significant and critical positive survivability features.

As an example, consider an unfavorable security characteristic $SC_{4.1}$ (see Figure 3 (a)), which represents the user's concerns on system operation degrading. Three *SP*s are defined: computation accuracy concern (SP_{41}), service consistency concern (SP_{42}), and resource allocation fairness concern (SP_{43}). A threshold selection structure *ulto*(2, [SP_{41}], [SP_{41}, SP_{42}, SP_{43}]) is defined for $SC_{4.1}$. To satisfy this structure, the system first must not have any unacceptable property defined by SP_{41}-SP_{43}. For instance, a system must not have unacceptable level of computation accuracy or unacceptable service consistency. If the system cannot meet this criterion, it is considered unsatisfactory in terms of the entire *SC*. But, even the system marginally meets those requirements (e.g., it offers a minimum level of acceptable computation accuracy, has a minimum level of service consistency, and provides a barely acceptable resource allocation fairness – all those may be caused by excessive allocations of resources to security functions), it still does not satisfy the user's requirement for a "highly secure" system (particularly when a balanced approach for service quality is concerned). Essentially, *ulto*(2, [SP_{41}], [SP_{41}, SP_{42}, SP_{43}]) requires that a satisfactory system must not have concern for SP_{41} (i.e., it must have a above-minimum level of computational accuracy) and has a "better than minimum" security feature in terms of either SP_{42} or SP_{43}. *utso(.)* is defined to indicate that the system should not have too many "boundary" security-related features in order to address the user's concern for a high-grade system.

The Master Security Requirement Tree

Figure 2 and 3 (a)-(d) denote the user's security requirements for the RFID system at different level of details represented by the corresponding *SC* hierarchies. Logically, those trees can be combined to form a comprehensive hierarchy structure to represent the user's overall security requirements for the RFID system, called a *master security requirement tree T* (due to page limitation, we will not show the complete tree here). The root of *T* represents the desired security level. An intermediate node represents an *SC*. Each leaf node represents the user's security requirements in terms of a basic *SC*, i.e., the threshold selection structure *op*(*j*, ML_j, SPL_j), where *op* represents a threshold selection operator *ltso(.)* or *ulso(.)*. As we can image, *T* clearly specifies the user's security requirements from different perspectives with different levels of details. If a system satisfies all the requirements specified in *T*, then it is considered reaching a certain level of security as specified by the user. Since we have presented all the individual components of a master security requirement tree, we will not discuss it further.

Discussions

The proposed approach provides a way for the users to incorporate their particular needs in defining security requirements. It also offers a considerable degree of flexibility at the system implementation level when the security options need to be chosen to carry out the security plan. To apply our approach, the users need to apply their domain knowledge to identify the favorable and unfavorable features of a system, the relative importance of those features, and their expectations for those features (e.g., the parameters in a threshold structure can be adjusted to customize the users' different levels of expectations).

This work contributes to the security requirement research in the literature by proposing a flexible and balanced approach in specifying security requirements. However, there are two major issues which remain unsolved and we will address in our future work. First of all, we assume that a security primitive can be either fully satisfied or not satisfied at all. However, in many cases it is unrealistic to make such a binary judgment. We plan to introduce the concept of "fuzzy satisfaction" for the

case when a system has certain features which partially satisfy the users' requirements in terms of a security primitive. Another issue we plan to address is the inter-dependencies of a subset of security primitives in a security characteristic. For instance, some security primitives may functionally depend on others and they must be all satisfied in the mean time when a security characteristic is considered. Developing a systematic approach to incorporate those non-trivial inter-dependency relationships in security requirement specification is needed.

CONCLUSION

In this paper, we proposed a formal approach for designing security requirements. Different from the existing research works, our approach allows a user to specify his/her security requirements with different levels of details. The approach also allows the user not only to specify the required security features but also to address his/her concerns towards some unfavorable system properties which are either unavoidable or too costly to mitigate. Since a threshold approach is used, flexibility is also achieved for the system developers to design and implement the system with possible different features but still satisfying the user's security requirements. We have shown how to apply the proposed approach using an RFID system.

ACKNOWLEDGEMENTS

This material is based upon work supported by the US Air Force Office of Scientific Research (AFOSR) under award FA9550-09-1-0215. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of AFOSR.

REFERENCES

1. Choobineh, J. and Anderson, E. (2010) Security Management Life Cycle (SMLC): A Comparative Study, *Proc. of the Sixteenth Americas Conference on Information Systems*, Lima Peru, August 12-15, 2010.
2. Devanbu, P. and Stubblebine, S. (2000) Software Engineering for Security: A Roadmap, *The Future of Software Engineering*, ACM Press.
3. Fabian, B., Gurses, S., Heisel, M., Santen, T. and Schmidt, H. (2010) A Comparison of Security Requirement Engineering Methods, *Requirements Engineering* (15), pp. 7-40.
4. ISO/IEC: Information Technology – Security techniques – Evaluation Criteria for IT Security – Part I, II, and III, ISO/IEC, Geneva Switzerland, Dec. 1, 1999.
5. Greene, J. and Baron, J. (2001) Intuitions about Declining Marginal Utility, *Journal of Behavioral Decision Making* 14(3), pp. 243-255.
6. Haley, C. B., Moffett, J. D., Laney, R. and Nuseibeh, B. (2006) A Framework for Security Requirement Engineering, *Proc. of the SESS Conference*, pp. 35-41, Shanghai, China.
7. McDermott, J. and Fox, C. (1999) Using Abuse Case Models for Security Requirement Analysis, *Proc. of the 15th Computer Security Applications Conference*, pp. 55-64, Phoenix, AZ, USA, Dec. 6-10, 1999.
8. Mellado, D., Fernandez-Medina, E. and Piattini, M. (2007) A Common Criteria Based Security Requirements Engineering Process for the Development of Secure Information Systems, *Computer Standards & Interfaces* (29), pp. 244-253.
9. Myagmar, S., Lee, A. and Yurcik, W. (2006) Threat Modeling as a Basis for Security Requirements, *Proc. of Symposium on Requirements Engineering for Information Security*.
10. Oladimeji, E., Supakkul, S. and Chung, L. (2006) Security Threat Modeling and Analysis: A Goal Oriented Approach, *Proc. of the 10th LASTED International Conference on Software*.
11. Siponen, M., Baskerville, R. and Heikka, J. (2006) A Design Theory for Secure Information Systems Design Methods, *Journal of the Association for Information Systems* (7)11, pp. 725-770.
12. Straub, D. W. and Welke, R. J. (1998) Coping with System Risk: Security Planning Models for Management Decision Making, *MIS Quarterly* (22)4, pp. 441-469.
13. Tryfonas, T. (2007) On Security Metaphors and How They Shape the Emerging Practice of Security Information Systems Development, *Journal of Information Systems Security* (3)3, pp. 21-50.
14. Young, D. and Conklin, W. A. (2010) Re-Examining the Information Systems Security Problem from a System Theory Perspective, *Proc. of the Sixteenth Americas Conference on Information Systems*, Lima Peru, August 12-15, 2010.
15. Zuo, Y., Pimple, M. and Lande, S. (2009) A Framework for RFID Survivability Requirements Analysis and Specification, *Proc. of International Joint Conference on Computer, Information and System Sciences and Engineering*, Dec. 4-12, 2009.
16. Zuo, Y. and Panda, B. (2008) Two Level Trust-based Decision Making Model for Information Assurance in a Virtual Organization, *Decision Support Systems* (45)2, pp. 291-309.