

1-1-2010

A HOLISTIC APPROACH FOR SECURITY REQUIREMENT SPECIFICATION FOR LOW-COST, DISTRIBUTED UBIQUITOUS SYSTEMS

Yanjun Zuo

University of North Dakota, yanjun.zuo@und.edu

Follow this and additional works at: http://aisel.aisnet.org/icis2010_submissions

Recommended Citation

Zuo, Yanjun, "A HOLISTIC APPROACH FOR SECURITY REQUIREMENT SPECIFICATION FOR LOW-COST, DISTRIBUTED UBIQUITOUS SYSTEMS" (2010). *ICIS 2010 Proceedings*. Paper 9.
http://aisel.aisnet.org/icis2010_submissions/9

This material is brought to you by the International Conference on Information Systems (ICIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in ICIS 2010 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

A HOLISTIC APPROACH FOR SECURITY REQUIREMENT SPECIFICATION FOR LOW-COST, DISTRIBUTED UBIQUITOUS SYSTEMS¹

Completed Research Paper

Yanjun Zuo

University of North Dakota

Grand Forks, ND USA

Email: yanjun.zuo@und.edu

Abstract

The class of low-cost, distributed ubiquitous systems represents a computing mode where a system has small, inexpensive networked processing devices, distributed at all scales throughout business activities and everyday life. The unique features of such a class of ubiquitous systems make the security analysis different from that for the centralized computing paradigms. This paper presents a holistic approach for security requirement analysis for low cost, distributed ubiquitous systems. Rigorous security analysis needs both quantitative and qualitative approaches to produce the holistic view and the robust data regarding the security features that a system must have in order to meet users' security expectations. Our framework can assist system administrators to specify key security properties for a low-cost, distributed ubiquitous system and to define the specific security requirements for such a system. We applied Bayesian network and stochastic process algebra to incorporate probabilistic analysis to the framework.

Keywords: IS security, ubiquitous technology, requirement analysis, research framework, IT risk management

¹ This work was supported in part by US AFOSR under grant FA 9550-09-1-0215.

Introduction

Ubiquitous computing, also described as pervasive computing, represents a post-desktop model of human-computer interaction in which information processes have been thoroughly integrated into everyday objects and daily business activities (http://en.wikipedia.org/wiki/Ubiquitous_computing). Ubiquitous computing has been used in a wide range of areas. There is recognition that in many ways we are already living in a ubiquitous computing world. For example, a domestic ubiquitous computing environment might interconnect lighting and environmental controls with personal biometric monitors woven into clothing so that illumination and heating conditions in a room might be continuously and imperceptibly modulated. Another common scenario posits refrigerators "aware" of their suitably-tagged contents, able to both plan a variety of menus from the food actually on hand and warn users of stale or spoiled food.

The term ubiquitous computing is generic and includes a set of techniques. In this paper, we focus on a class of low-cost, distributed ubiquitous systems, which plays important roles in the current and future pervasive computing. A system in this class has the following characteristics: (1) it has a large set of small, inexpensive, networked processing devices. Those devices have limited computation and storage capabilities and use wireless means to communicate with their peers or with other more powerful interrogating devices; (2) it is highly decentralized with the components distributed in a large scope; and (3) it focuses on collecting information through the device nodes to support higher-level applications.

The most important low-cost, distributed ubiquitous systems include wireless sensor networks (WSNs) (Akyildiz, 2002), radio frequency identification (RFID) (Karygiannis, et al. 2007) and mobile ad hoc networks (Toh, 2002). They represent the most widely used ubiquitous applications in business and many personal activities. A WSN consists of a large number of sensor nodes and are able to collect and disseminate data in areas where ordinary networks are unsuitable for environmental and/or strategic reasons (Zia, 2008). WSNs have been used in many applications such as in-building energy usage monitoring; military and civilian surveillance; natural habitats monitoring; and data gathering in instrumented learning (to name a few). An ad hoc network is a self-configuring network of mobile devices connected by wireless links. Such a network does not rely on a preexisting infrastructure. Instead, each node forwards traffics unrelated to its own use, and therefore becomes a router. The decentralized nature of ad hoc networks makes them suitable for a variety of applications where central nodes can't be relied on. Applications include emergency situations like natural disasters or military conflicts (http://en.wikipedia.org/wiki/Wireless_ad_hoc_networks). An RFID system consists of a set of low-cost tags (passive or active), each of which is physically attached to an item with a unique identification. An RFID reader is a device capable of communicating with an RFID tag. RFID has been used for automatic item identification and data collection. RFID enabled systems have been applied in various areas including supply chain, military, healthcare, and crisis management, just to name a few. The US Department of Defense was requiring its 60,000 suppliers to tag items with RFID. Major retailer chains such as Wal-Mart and Target have mandated that all suppliers introduce RFID.

Although ubiquitous computing holds the promise of enhancing human life quality and business productivity, it also raises great security challenges. The security and privacy issues are considered as the largest barriers for the proliferation of ubiquitous applications. For instance, since most ubiquitous devices rely on wireless communications to connect to each other and with the backend supporting systems, this opens various opportunities for an attacker to eavesdrop, intercept, modify, or replay the messages transferred in the wireless channels and to jam and interfere with the wireless communications. In addition, since many ubiquitous systems have no fixed network structure and the ubiquitous devices tend to be unattended, the attacker could physically capture those devices. By reverse engineering or other similar techniques, the sensitive data stored in those devices could be retrieved, and consequently those victim-devices are compromised.

There are also a great deal of privacy issues in ubiquitous computing, which must be addressed by deploying effective security mechanisms together with legal regulations and organizational policies. Imagine, for example, that a system allows an unattended person to view the video of a conference as captured by camera and RFID readers. In an RFID-enabled inventory system, imagine that a competitor-sponsored attacker queries logistic and commercial data about an organization's inventory and strategic operation plans. Many ubiquitous applications make information much easier to collect and access than ever before. There must be effective security mechanisms to ensure that sensitive data is only available to the right user and at the right time. Therefore, it is critical to define

appropriate security requirements for those systems so that actions can be taken based on those requirements to ensure that they can meet the desired security level.

Security in ubiquitous computing is challenging, however, due to several reasons. First of all, a low-cost, distributed ubiquitous system has a great deal of components deployed in a large scope possibly with a high level of mobility. This makes the system monitoring and access control difficult. Secondly, many ubiquitous devices (e.g., RFID tags and WSN sensors) have limited physical protection. They could be physically attacked and controlled by adversaries. Thirdly, some ubiquitous devices (e.g., low-cost RFID tags and ad hoc nodes) have limited resources (e.g., memory storage and computational power) for security. Many standard security mechanisms cannot be implemented on those devices. All those factors make securing a ubiquitous system a challenging task.

In this paper, we present a holistic approach for security requirement analysis for the class of low-cost, distributed ubiquitous systems as mentioned earlier. Rigorous security requirement analysis needs both quantitative and qualitative approaches to produce the holistic view and the robust data regarding the security features that a system must have in order to meet users' expectations. Our framework provides a systematic security requirement analysis for such class of ubiquitous systems with three major components: (1) qualitative identification of security properties that the system must have; (2) quantitative measurement of the compromise rate(s) of the critical system components based on a security requirement specification; and (3) simulative analysis of the security level of the system. The framework helps the system administrators answer the following types of questions: (a) how to systematically identify the security properties essential to low-cost, distributed ubiquitous systems; (b) to which degrees those security properties must be satisfied by such a system in order for it to be considered with the required security level; and (c) how to determine the security level of the system and whether the system meets users' security criteria?

Our quantitative analysis and simulation model are particularly suitable for the class of low-cost, distributed ubiquitous systems for the following reasons:

(1) Given the low-cost and high level of distribution of the components in such a system and the insecure wireless communications among those components, compromises or failures of some nodes are considered common. Therefore, a low-cost, distributed ubiquitous system must have a certain level of repair ability to recover any compromised nodes as soon as possible. Our framework models the compromise and recovery of some system components and analyzes their dynamic effects on the overall system security given a security requirement specification for the system. This is particularly suitable for a low-cost, distributed ubiquitous system and different from those security analysis methods developed for centralized computing paradigms;

(2) Given the security challenges that a low-cost, distributed ubiquitous system may face and the computational constrictions of the system components, the system as a whole must meet users' security requirements even though some individual components could be compromised. Our framework applies a "bottom-up" approach – we study how the integrated effects of component security affect the overall system security. A holistic approach is proposed to help users specify the security requirements for individual components in order to ensure that the system as a whole satisfies the users' security criteria;

(3) Since low-cost ubiquitous devices are vulnerable to various attacks and they tend to be unattended in many operational environments, the security status of those devices is often uncertain. Given various factors that could affect the system components' security features, probabilistic reasoning of the security requirements for the system and their components is both necessary and beneficial. Our framework applies a probabilistic model to study how the security features of a critical system component determine the rate at which the component could be compromised and how likely the system will meet users' security criteria given those components' security status.

Although we use an RFID system as a running example throughout the paper to illustrate how to apply our framework, the methodologies and methods presented in this paper can be applied to any low-cost, distributed ubiquitous system. As we mentioned earlier, such class of ubiquitous systems is featured with a large set of small, inexpensive networked components which are highly distributed and may be vulnerable to various attacks. For those systems, although some components could be compromised, the system as a whole must still provide useful services. We develop a framework for security requirement specifications so that the system can meet users' security criteria.

The rest of the paper is organized as follows. We first discuss related work. In the following section, we describe our framework in detail. We first discuss security properties identification for a low-cost, distributed ubiquitous system, and then present security requirement quantification. We also summarize the framework and show how the proposed components are integrated. Finally, we conclude our paper.

Related Work

We discuss two major streams of work closely related to our research.

Security Requirement Engineering Security requirement analysis has attracted many studies from the research community. Threat modeling as a basis of security requirement is discussed by Myagmar (Myagmar, et al. 2006). The authors present an approach to systematic threat modeling for complex systems. They also discuss the concept of risk management. A goal-oriented approach is proposed by Oladimeji (Oladimeji, et al. 2006) to security threat modeling and analysis by using visual model elements. The authors introduce the notions of negative soft-goals for representing threats and inverse contributions for evaluating design alternatives. An analysis procedure is provided as a guide to context-sensitive selection of countermeasures. Mead (Mead, 2003) describes the current state of requirement engineering for survivable systems. The requirements for survivability strategies are enumerated as resistance, recognition and recovery.

In the literature, however, there is little work on systematic reasoning and specification of the security requirements for a ubiquitous system. Most of the existing works consider a centralized computing paradigm and model the system as a whole to transit from one state to another under different attack scenarios. Those approaches may not be suitable for a low-cost, distributed ubiquitous system with a set of small, inexpensive networked devices vulnerable to various attacks. Our framework is proposed specifically for this class of ubiquitous systems and considers the impact of the component security features on the overall security level of the system. It assists system administrators to enumerate security requirements for such a ubiquitous system in a systematic way. The security level of the system is quantified given a security requirement specification (i.e., a set of detailed security requirements). The administrators can then determine if the proposed security requirements can ensure that the system satisfies the users' security expectations. Various what-if types of questions can be answered to allow a set of more flexible and realistic security requirements for a ubiquitous system. The existing works do not sufficiently address those issues.

In terms of threat modeling, our work differs from the existing studies in twofold. First, our work extends the existing threat models with mapping risks to specific security requirements for a low-cost, distributed ubiquitous system. We present the approach and structure for a systematic security requirement mapping. Secondly, it incorporates probabilistic reasoning to quantify the security requirements. While other threat modeling and security requirement analysis studies mainly focus on qualitative analysis, our work also provides quantitative analysis and simulations to specify the measurable and verifiable security requirements for a low-cost, distributed ubiquitous system.

Bayesian Network Applications Bayesian network has been applied to various applications, where evidence and cause-effect relationships can be used to predict probabilistic unknown values of variables of interest. Several Bayesian network-based models in security have been proposed in the literature. We only discuss a few here. Mehdi (Mehdi, et al. 2007) propose a new approach of an anomaly Intrusion detection system (IDS). It consists of building a reference behavior model and the use of a Bayesian classification procedure associated to unsupervised learning algorithm to evaluate the deviation between current and reference behavior. Continuous re-estimation of model parameters allows for real time operation. Abouzakhar (Abouzakhar, et al. 2003) develop a probabilistic approach using Bayesian network to detect distributed network attacks as early as possible. Their approach shows how a Bayesian network probabilistically detects communication network attacks. Herath (Herath, et al. 2008) develop a model for information security investments using Bayesian inferences for valuation and post-auditing. Their work uses Bayesian statistics to model learning options and post-auditing of sequential real options, an idea that can be applied to a variety of other types of sequential investment problems. Sebyala (Sebala, et al. 2002) present the application of Bayesian technology in the development of an anomaly detection system for proxylets (third party executable codes). Their model can be incorporated into an intrusion detection system that will provide runtime security to ensure that active platform integrity is maintained while running third party executable codes.

In this paper, we also applied Bayesian network to security requirement analysis; but our work has different goals from the above frameworks. We use Bayesian network to represent the cause-effect relationships between various factors that affect the compromise rate of critical system components of a ubiquitous system. Based on those relationships and possibly user input evidence, systems administrators can estimate the mean probabilistic compromise rate of those critical components. This quantification is important to determine the overall system security level given other factors such as system recovery and adversary capability.

The Framework

There are two major phases of activities in our framework of security requirement analysis for a low-cost, distributed ubiquitous system (as shown in Figure 1): **phase one** – system security property identification; and **phase two** - security requirement quantification. The first phase provides a qualitative view of security requirements for a ubiquitous system in terms of security properties that the system (and its major components) should have in order to be considered secure. In order to systematically identify the important security properties of a system, it is necessary to formally characterize the system, pinpoint its critical components and major access points, and specify the security threats to the system and the critical components. The second phase is an extension of the first. It quantitatively measures how the system security features of the proposed security requirement specification determine the security level of the system. This phase provides quantification that indicates whether the proposed security requirements will result in a level of security of the system that meets users' criteria. This analysis helps system administrators define and evaluate a security requirement specification for a ubiquitous system. From the perspective of system engineering, analyzing security requirements is the important first step in security specification and compliance evaluation.

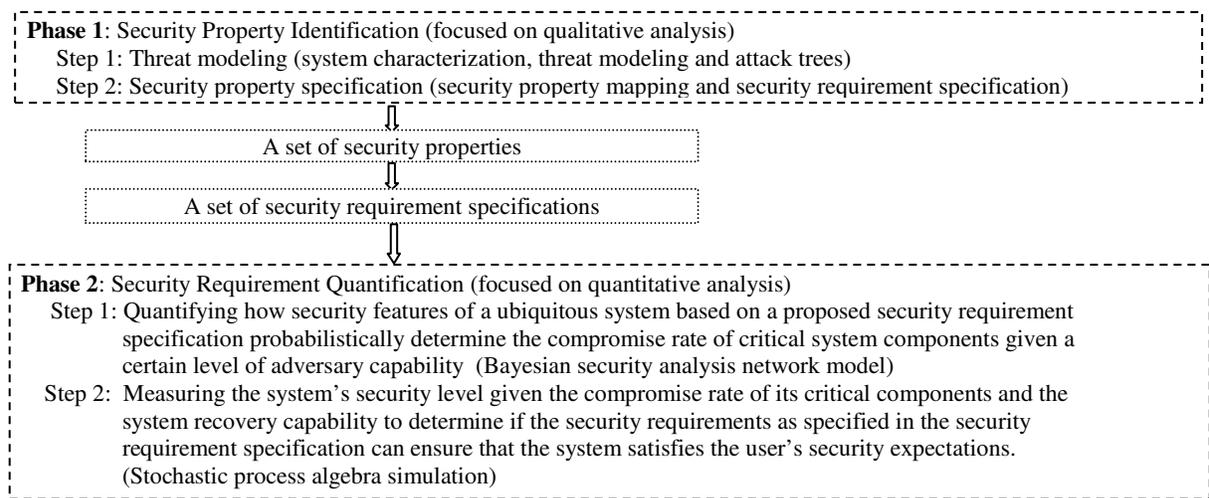


Figure 1: The Two Phases of the Framework and the Tools and Methodologies Used in Each Phase

Identifying Security Properties

In this section, we present the methodologies and approaches to identify the security properties that a low-cost, distributed ubiquitous system must have in order for it to be considered secure. This phase provides a qualitative analysis for further security requirement quantification. The goal is to identify which kinds of security features are essential for the system and its major components.

The term security property refers to a system characteristic which is essential to the security of the system. Some examples of security properties of such a ubiquitous system include: (1) strong authentication and authorization abilities of its critical components; (2) the robustness of the services that those components provide and the abilities of the components to tolerate system faults (due to malicious attacks or system failures); and (3) the physical security of the components themselves. For each security property, there is a degree of measurement given a particular system: to which level that the system satisfies that security property. This satisfaction level is determined by the security requirements for that system. Therefore, the security requirements proposed for a system can be represented in terms of a set of key system security properties. For instance, we can specify the security requirements for a ubiquitous system as follows: (1) the critical components of the system must have a high level of authentication and authorization capability in order for them to identify other legitimate system components; (2) the system's critical components must be robust to internal and external faults caused by either malicious attacks or system failures; (3) the critical components must have at least a medium level of physical security.

From the perspective of security, it is always desired that a system has a high level of satisfaction for every security property. However, due to cost constraints in system development and security technical challenges, it is often

difficult to develop and maintain a perfectly secure system. Consequently, it is unrealistic to require that the system have a high level of satisfactions in every security property. This is particularly true for a low-cost, distributed ubiquitous system whose components have limited computation and storage capabilities. Therefore, the system administrators need to determine whether the set of proposed security requirements for a system will guarantee that the system meets users' security criteria and in the mean time with an acceptable level of implementation costs. There are two general steps in security property identification: threat modeling and security property specification. Details are discussed in the next two sub-sections.

Threat Modeling

Threat modeling is an important step towards security requirement specification of a system. It involves understanding the complexity of the system and identifying possible threats to the system. Based on the identified threats, the system administrators can define the appropriate security requirements for the system in order to ensure that the system has the desired security features to withstand malicious attacks. With the corresponding security mechanisms put in place, the likelihood for the attackers to compromise the system can be greatly reduced.

We use an RFID system as a running example throughout this paper to illustrate our ideas. We chose an RFID system because RFID is rapidly becoming an important part of enterprises and an RFID system is subject to various threats which can affect system security due to the high distribution nature and vulnerability of its components.

(1) Characterizing critical system components and major access points

The first step in threat modeling is to identify critical system components and major access points to this system. An important approach is to step through the system's assets, reviewing a list of possible attacks for each asset (Westmark, 2006). A set of threats that could possibly compromise the system and its critical components can be identified. Then, the threat model is used as a basis to identify the security properties that the system must have given the threats.

Figure 2 shows a typical structure of an RFID system. There are three major categories of components: tags, readers, and backend servers. A tag is a small device physically attached to an item with a unique identifier. Each tag has an antenna for radio communication with one or more readers. Compared with a tag, an RFID reader is a more powerful device which can identify the presence of a tag and read information stored in the tag (Glover & Bhatt, 2006). The backend RFID server stores tag and reader authentication information as well as other data such as product descriptions and ownership. Low- to mid-range RFID tags have limited computational capabilities and memory storage. For those reasons, many standard security mechanisms (e.g. public key cryptography protocols) cannot be applied to those tags. This further makes securing an RFID system challenging.

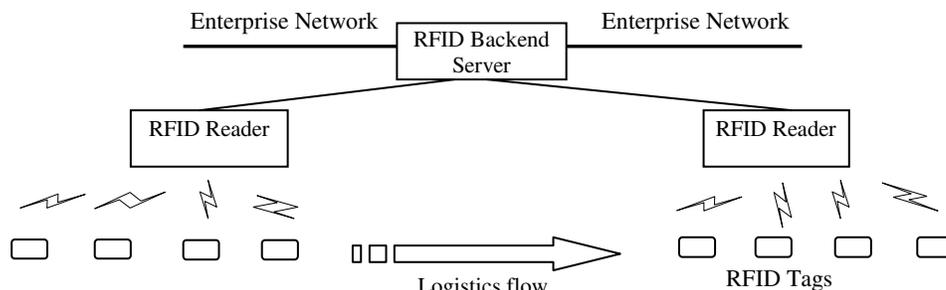


Figure 2: An RFID System Structure

Formally, an RFID system can be represented as a tuple $\Sigma = \{R, T, D, O_1, O_2\}$, where R represents a set of RFID readers; T represents a set of tags; D represents a set of backend servers; and O_1 represents a set of relations between R and D , i.e., $O_1 \subseteq D \times R$. An instance of O_1 , e.g., $o = (r, d) \in O_1$, where $r \in R$ and $d \in D$, indicates that an RFID reader r is a legitimate reader authorized by a backend server d to retrieve further detailed tag information; and O_2 presents a set of relations between T and D , i.e., $O_2 \subseteq T \times D$. An instance of O_2 , e.g., $o' = (t, d) \in O_2$, where $t \in T$ and $d \in D$ indicates that tag t is a legitimate tag in the system and there is a record for this tag as maintained by the backend server d .

An access point is a communication connection point among the system components or with external systems. Access points are often used as entries for an adversary to penetrate into a system. In an RFID system, for example, tags and readers communicate through wireless channels, which are vulnerable to various attacks such as signal jamming as well as message interception, eavesdropping, interception, and replay. Therefore, the tag-reader wireless communication channel is one of the major access points for an attacker. On the other hand, since readers and the backend servers typically communicate through secure channels (wired or wireless), they are less concerned. Furthermore, RFID tags have limited physical security. For instance, an attacker can break into the tag memory by force or use fuming nitric acid to remove the resin coat of the tag, disconnect the microprocessor and attach on a single micro-probing needle to the data bus. Then by providing the address signal on the address bus, the secret memory content in that tag can be accessed. Other means of physical attacks include shaped charges, laser etching, and ion-probing. So, physical contact is another major access point for the adversary to attack an RFID tag.

(2) Identifying threats

There are several approaches to identify security threats to a system and its critical components. In this paper, we use attack trees (Schneier, 1999) to identify the critical threats related to a critical system component (e.g., an RFID tag). An attack tree provides a formal, methodical way of describing the security of the system based on varying attacks. In an attack tree, the root represents the threat corresponding to an asset (the attacker goal) and each node represents the attacker decision process. Figure 3 shows an attack tree for an RFID tag. It outlines the common approaches that an adversary can use to attack an RFID tag. Those threats are discussed next and interested readers may refer to (Mitrokovska, *et al.* 2009; Zuo, 2010) for a more complete list of attacks on RFID systems.

Tag corruption: an attacker physically destroys a tag or uses special devices to read the tag memory and obtain identification related information (e.g., the tag secret key).

Tag cloning: an attacker creates duplicated “authentic” RFID tags after capturing tag keys via either physical attacks of the tags or eavesdropping on the communications between the tag and a reader.

Reply attack: an attacker makes the reader believe that the tag is in its close vicinity by surreptitiously forwarding the signal between the reader and an out-of-field tag.

Replay attack: an attacker reuses a tag’s response to the reader in a previous session to impersonate the tag in the current session.

Tag-reader communication channel jamming and interference: tag readings are interfered or intercepted through signal jamming by attackers using a powerful device.

Tag-reader state desynchronization: An attack causes a tag to assume a state from its normal operation. The tag becomes temporarily or permanently incapacitated.

Impersonation attack: an attacker impersonates a tag without knowing the tag’s internal secrets. The attacker with knowledge of the internal state of a tag is able to impersonate the valid reader to the tag.

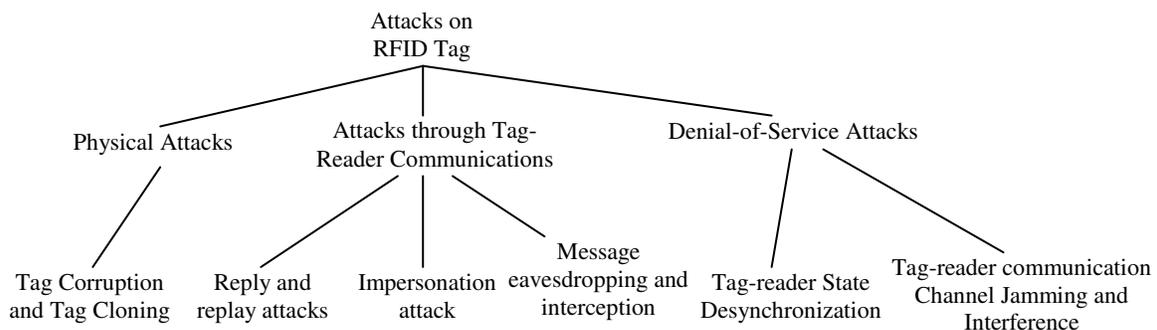


Figure 3: Attack Tree for RFID tag

Similarly, threats can be identified for other important components and major entry points of an RFID system. For instance, the major threats to an RFID reader include impersonation, eavesdropping, relay attacks, spoofing, and passive/ active interference of tag-reader communications. The major threats to an RFID backend server include denial-of-service attacks, buffer overflow attacks, and malicious code injections. Due to page limitation, we will not present the attack trees corresponding to the RFID reader and backend server. But, we must point out that those

attacks are much harder to mount as compared with the attacks on RFID tags. Since RFID readers and the backend server are much more powerful devices, we assume that the readers and the backend server are relatively secure. Our focus is on the low-cost RFID tags with limited computational capacities and storage space. RFID tags are the most vulnerable components of an RFID system.

Specifying Security Properties Based on the Identified Threats

After the major threats are identified, the corresponding security properties that the system and its critical components must have in order to eliminate or mitigate those security threats can be specified. Identifying the important security properties is essentially a mapping between the security threats and the set of security features that the system must have to withstand those threats. We develop a security property mapping graph (similar to the survivability mapping graph presented by Zuo (Zuo, et al. 2009)) as a formal model to relate the security goals to a set of security properties of an RFID tag based on the threats identified in the subsection “Threat Modeling”.

Conceptually, a security property mapping graph is a directed graph $G = \{V, E\}$, where V represents a set of vertices and E represents a set of directed edges. Each vertex in V represents one of the three types of nodes: (1) security goal or sub-goal node, denoted by an oval, representing a security objective to achieve; (2) threat node, denoted by a rectangle, representing a threat to the security goal or sub-goals; and (3) security property node, denoted by a wrapped rectangle, representing a security characteristic of the system to overcome the threat and achieve the desired security goals. These three types of vertices appear alternatively along a path in G from the root to a leaf (in the reverse edge direction). The goal node must be at the root, followed by an optional sub-goal node, followed by a threat node, and finally, followed by the appropriate security property node at the leaf.

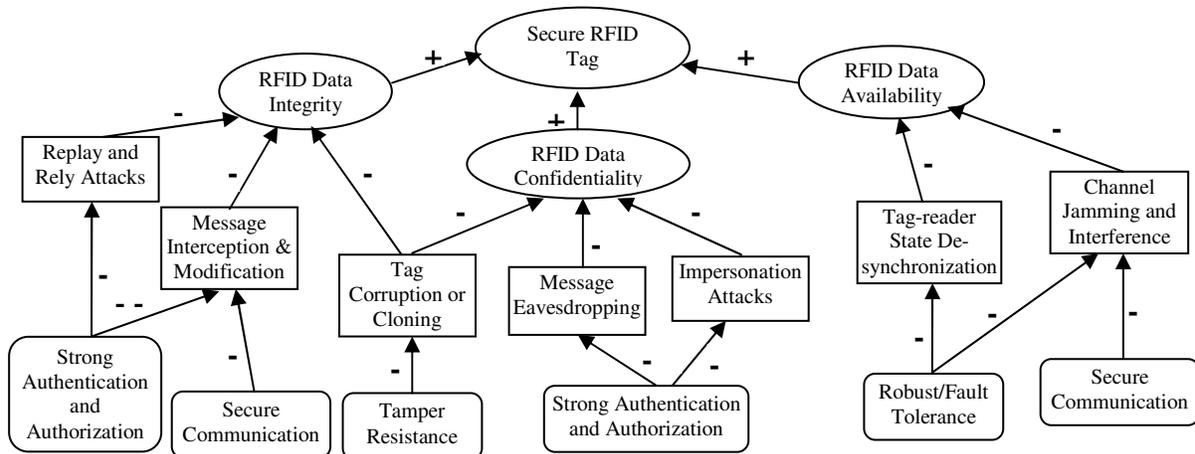


Figure 4: RFID Security Property Mapping Graph

Each edge $e \in E$ in a security property mapping graph G represents a relationship between the two connected nodes. The relationship can be a positive contribution represented by the “+” symbol or an inverse (or negative) contribution represented by the “-” symbol. Positive contribution increases the chance of meeting a goal and inverse contribution decreases it. For instance, suppose that the main security goal for an RFID tag is to ensure that the RFID system can provide trustworthy, timely, and reliable information about the tagged items to support higher-level applications. We can refine this goal to three sub-goals: RFID data confidentiality, integrity, and availability. The three sub-goals positively contribute to the main goal. So, the edge from each of the sub-goal node to the main goal node should be marked as “+”. On the other hand, a threat such as a denial-of-service attack on an RFID tag (through its communications with the reader, for example) can negatively contribute to the sub-goal of data availability since this attack could cause RFID data to be unavailable and consequently inversely affects information availability. Therefore, the edge from the threat node “denial-of-service-attack” to the sub-goal node “RFID data availability” should be marked as “-”. After the threats are specified, appropriate security properties can be identified so that, if the system has the security features in terms of the identified security properties, the system can have the ability to resist those threats. Since the security properties represent counter-attack security requirements, they negatively contribute to the respective threats.

We give the security property mapping graph for an RFID tag in Figure 4. As we can see from the figure, after the main security goal and sub-goals as shown above are specified, a set of potential threats that may hinder the main security goals or the sub-goals are outlined. The possible threats that could affect RFID data confidentiality include eavesdropping and interception of messages transferred between an RFID tag and a reader, physically corruption of a tag, cloning of a tag, and impersonation attack (e.g., a fake reader attempts to fool a tag for sensitive data). Other threats to a tag are shown in the figure, which aim at compromising data integrity and availability. After the threat specification, a set of survivability properties are specified for each threat which, if satisfied, can effectively thwart the corresponding attacks. For example, for the threat “physical corruption of a tag”, the security property “RFID tag physical tamper resistance” is specified – if an RFID tag has a strong tamper resistance property, it negatively contributes to threat of physically corrupting tags and thus positively contributes to the security sub-goals “RFID data integrity” and “RFID data confidentiality”. We summarize the security properties for an RFID tag in Table 1.

Table 1: Security Properties of an RFID Tag

Tamper resistance	The tag’s ability to resist malicious attacks such as a physical break-in the tag, access control over sensitive data stored in the tag, anti-cloning, and anti-reverse engineering.
Tag-reader Secure communications	The communications between a tag and a reader should be secure against various wireless attacks. For instance, all the sensitive data should be appropriately encrypted using cryptographic protocols suitable for low-cost RFID tags.
Tag Robustness/fault tolerance	The tag’s ability to provide a reasonable level of robustness in critical services and fault tolerance. Those include basic damage masking mechanisms and prompt recovery from tag-reader state desynchronization as caused by malicious attacks.
Tag’s Authentication and authorization Ability	The tag’s ability to identify legitimate readers and only allow authorized entities to access sensitive data stored in the tag. This ability is represented by appropriately deployed anti-spoofing mechanisms, distance-bound authentication, and multi-factor authentications.

Security Requirement Quantification

In this section, we discuss security requirement quantification – the second phase of our framework. This phase focuses on quantitatively specifying the security requirements (called *a security requirement specification*) for a ubiquitous system. A security requirement specification is defined in terms of a set of security properties of the system as shown below. It is to ensure that the system has the required level of security. We use the term security level to represent the ability of a system to withstand malicious attacks and provide secure services to users.

The process of security property identification as shown in the above qualitative analysis identifies a set of security properties $\{sp_1, sp_2, \dots, sp_n\}$ for a low-cost, distributed ubiquitous system. A security requirement specification, denoted as SRS_i , represents the users’ security requirements for the system in terms of those security properties. It is represented by a set of values to the identified security properties. For example, Table 1 shows a set of security properties of an RFID tag {“Tamper resistance”, “Secure communications”, “Robustness/fault tolerance”, “Authentication and authorization ability”}. The security requirement specification SRS_i may be determined as {“tamper resistance” = high, “Secure communications” = medium, “Robustness/fault tolerance” = capable, “Authentication and authorization”=high}. Since cost is an important factor to consider in security analysis, we also include a cost factor for each security requirement, which indicates the resources needed to implement the security mechanisms in order to meet the particular security requirement. Formally, a security requirement specification SRS_i is represented as a tuple in the following format.

$$\Phi = \{(sp_1 = a_{i,1}, C(a_{i,1})), (sp_2 = a_{i,2}, C(a_{i,2})), \dots, (sp_n = a_{i,n}, C(a_{i,n}))\}$$

where $a_{i,1}, a_{i,2}, \dots, a_{i,n}$ represent the set of values assigned to the corresponding set of security properties $\{sp_1, sp_2, \dots, sp_n\}$ based on SRS_i ; and $C(a_{i,1}), C(a_{i,2}), \dots, C(a_{i,n})$ represent the costs associated with the security requirements $sp_1 = a_{i,1}, sp_2 = a_{i,2}, \dots, sp_n = a_{i,n}$, respectively. We also define a cost function for the security requirement specification SRS_i as a whole, denoted as C_{SRS_i} as shown below.

$$C_{SRS_i} = \sum_{j=1}^{j=n} C(a_{i,j}) \cdot$$

Incorporating cost into security requirement analysis is important for ubiquitous systems. Some security requirements may be very costly to improve, while others may be relatively inexpensive. For a low-cost, distributed ubiquitous system with a set of components distributed in a large scope with a high level of mobility, it may be very costly to implement advanced security functions. Since a ubiquitous device typically has limited physical resources (e.g., computation, communication, and storage) which can be devoted to security functions, system administrators may not be able to freely refine the security properties and choose the corresponding mechanisms to meet a security requirement. For instance, currently it is still not realistic to implement standard cryptographic mechanisms (e.g., public-key algorithms) in low-cost RFID tags as limitations in area and power are quite severe in those devices. Rather, for low-cost RFID tags, several efficient implementations of hash functions are available. For instance, non-linear feedback shift registers can be used to build a low-cost hash function. Another alternative is to implement low-cost hash functions for a block size of 64 bits (Yuksel, 2004). Regarding pseudorandom functions, a variety of well-known and validated constructions are established. As discussed by Burmester (Burmester, *et al.*, 2007), it is possible to build a pseudorandom function that makes a call to a pseudorandom generator (PRG) per bit of input processed. In turn, a very efficient PRG implementation can be achieved using linear feedback shift registers. This results in a small number of bit operations per input and output bit. In addition, block ciphers can be used to implement pseudorandom functions through standard constructions. Therefore, the costs to implement those functions are relatively low and suitable for RFID tags. In general, the cost to implement a security mechanism in a system is domain dependent and there is no general formula which can be applied in all the cases. In this paper we focus on security requirement quantifications and will leave the cost issues to our future work.

A security requirement specification SRS_i specifies the guidelines and security requirements for a system so that a set of security actions can be taken and the corresponding security mechanisms can be deployed in order for the system to achieve a desired level of security L . The problem statement of the security requirement quantification is specified as “Given a security requirement specification SRS_i with an acceptable level of implementation cost C_{SRS_i} , can the security features of the ubiquitous system based on SRS_i ensure that the system has a desired level of security and thus meet the users’ security criteria?”

Two consecutive steps are specified for security requirement quantification: (1) analyzing the cause-effect relationships between the specified security requirement specification for a critical system component and the rate at which the component could be compromised given a certain level of adversary capability; and (2) measuring the security level of the system as a whole given the compromise rate of the critical components and the system recovery capability. The two steps will be discussed in the next two subsections. The output of the first step is used as input to the second step.

Determining the Compromise Rate of a Critical System Component

We apply a probabilistic model to study how the security features of a critical system component determine the rate at which the component could be compromised by an adversary. A compromise rate represents how fast a system component could be possibly attacked and damaged. It is measured as the number of the components in a category that can be compromised by an adversary per unit of time. We perceive that the compromise of a system component is dependent on several security features of that component in such a way that relationships exist between those security features and the compromise rate. We represent such probabilistic relationships using a Bayesian network. Bayesian network is a very useful tool to represent evidence and cause-effect relationships and to predict probabilistic unknown values of variables of interest. In the following discussion, we show how the compromise rate of an RFID tag is probabilistically determined by the tag’s security characteristics as specified by a security requirement specification.

(1) The Bayesian security analysis network

In this section, we present our Bayesian security analysis network model. A Bayesian network consists of a structure and a set of parameters. The structure is composed of nodes representing domain variables and edges connecting these nodes. The Bayesian network structure encodes the probabilistic dependencies in the data – the presence of an edge between two variables means that there exists a direct dependency between them. Although the structure of a Bayesian network can be learned from data, it may be less expensive to incorporate domain knowledge and apply heuristic rules to determine the Bayesian network. In our framework, the qualitative analysis presented earlier has already identified important security properties of a low-cost, distributed ubiquitous system and those

parameters have a great impact on the security status of a system component. Therefore, we apply the results of the qualitative analysis and incorporate domain heuristic rules to determine the Bayesian network structure. The detailed procedure consists of three steps as shown below.

(a). Determine the important domain variables. The goal of our Bayesian security analysis network is to represent how the security features of a component of a ubiquitous system probabilistically determine the security status of the component (represented as the compromise rate of the component) given a level of adversary capability. The qualitative analysis has already identified important security properties (see Table 1). Those domain variables will be used as the nodes of the Bayesian security analysis network. In addition, since the adversary capability has a great impact on the security status of a component, we also include such a variable in the Bayesian network.

(b). Identify the cause-effect relationships (represented as parent-children relationships in a Bayesian network) between the domain variables. Heuristic rules allow the incorporation of user knowledge through the prior probabilistic and relative strength over the associations between the variables. For example, system administrators may determine that a set of security features of an RFID component determine the security baseline of the component. Furthermore, the component’s security baseline and its robustness and fault tolerance determine the component’s compromise probability (e.g., highly possible, possible, low possible, and unlikely) given an adversary’s capability to launch attacks. Therefore, there are strong cause-effect relationships between the security properties of the component and its security baseline. In turn, the level of the component’s security baseline is an important factor in determining the compromise rate of the component under an attack. The encoding of the dependency between those variables provides the basic structure of the Bayesian network.

(c). Direction of the edge by following the oriental rules. The fundamental oriental rule is that a parent node precedes a child node. The cause-effect relationships identified in step (c) then determine a set of directed edges from the nodes that represent the “cause” side of the factors to the nodes that represent the “effect” side of the factors.

Following the above procedure, we developed a Bayesian security analysis network, where each node of a Bayesian network represents a state, and a cause-effect relationship between two nodes is denoted by an arrow, called an edge. An instance of the Bayesian network corresponds to a security requirement specification. For example, the instance of the Bayesian network corresponding to the security specification SRS_i is shown in Figure 5, where $SRS_i = \{$ “RFID tag tamper resistance” = “Very resistant”; “Tag-reader secure communication level” = “Medium”; “Tag authentication/authorization ability = “High”; “Tag robustness/fault tolerance”= “Capable”} given a high level of adversary capability. In the next few sub-sections, we discuss the major components of the Bayesian network.

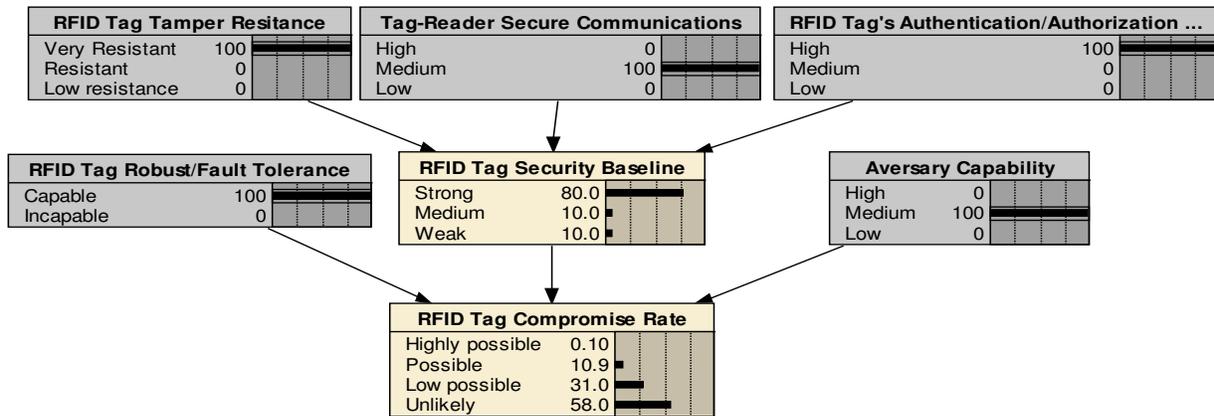


Figure 5: An Instance of the Bayesian Security Analysis Network

Nodes

As we discussed earlier, a node in a Bayesian network represents a domain variable in the situation being modeled. Each node is represented graphically by a labeled rectangle. A node can take different values, each of which is referred to as a state of the node. A nature node models the nature or reality about one aspect of the system. Based on the domain variables identified in the first phase of the framework, our Bayesian security analysis network has seven nature nodes (as explained in Table 2).

Table 2: The Bayesian Network Nodes

Bayesian Network Node	Semantic Meanings	Domain Values
Tamper resistance (TR)	This node represents a security property of an RFID tag - the degree that the tag is resistant to physical corruption, tag cloning, tag disabling, malicious modification, and other physical attacks.	Very resistant, resistant, or low resistance.
Secure communications (SC)	Secure communication channels between the tags and readers are critical since various attacks (e.g., jamming, interference, channel hijacking, message interception and replay) can be mounted. This node is used to represent the security property of an RFID in terms of the security characteristics in terms of its communications with a reader.	High, medium, or low.
Reliable authentication/authorization (RAA)	This node represents an RFID tag's ability to authenticate legitimate readers and allow only authorized parties to access tag data. Virtually, every tag has at least some secret shared with a legitimate reader. However, the authentication/authorization ability varies from tag to tag.	High, medium, or low.
Security Baseline (SB)	The combined effect of RFID tag tamper resistance, authentication & authorization ability, and the degree of secure communications with a reader determines the level of security baseline of the tag. This node represents such a security baseline.	Strong, medium, or weak.
Adversary capability (AC)	This node represents an adversary's ability to compromise an RFID tag.	High, medium, or low.
Robustness/fault tolerance (RFT)	This node represents an RFID tag's basic ability for function/service robustness and fault tolerance functions as well as damage masking. For instance, some RFID tags are programmed to resynchronize with the readers in case that the adversary desynchronizes their communications and tag-reader state.	Capable or incapable.
Tag Compromise Rate	Given the values of the three nodes: "security baseline", "adversary capability" and "robustness/fault tolerance", the rate at which an RFID tag could be compromised by the adversary is presented by this node. We use "negligible" (e.g., 0.005) to represent the case that the tag is unlikely to be compromised.	High rate (0.8), average rate (0.5), low rate (0.1) or negligible (0.005).

Edges

An edge of a Bayesian network represents a cause-effect relationship between two nodes. It is represented graphically by an arrow between the two nodes and the direction of the arrow indicates the logical causality (see Figure 5). The intuitive meaning of an edge drawn from node A to node B is that A has a direct influence on B. The conditional probability table (CPT) associated with a node defines quantitatively how the node (called the child node) is influenced by others (called the parent nodes). In a relationship, the child node is conditionally dependent upon their parent nodes. For instance, as shown in Figure 5, the three parent nodes "Tamper resistance", "Secure communications" and "Reliable authentication and authorization" jointly influence the child node "Security baseline" of an RFID tag.

Conditional Probability Tables

In a Bayesian network, the relationship between two nodes (events) A and B is defined as a conditional probability, $P(A | B)$, which represents the probability of A conditional on a given outcome of B . The conditional probability is calculated using Bayes' Theorem (Jensen, 2002) as shown in Formula (1), where $P(B|A)$ is the conditional probability of B given A , and $P(B)$ and $P(A)$ are the unconditional probabilities of B and A , respectively. The probabilistic dependency is maintained by CPT, which shows all possible outcomes of a given node and the conditional probability corresponding to each outcome given all its parent nodes.

$$P(A | B) = \frac{P(B | A) * P(A)}{P(B)} \quad (1)$$

To present the conditional probabilities of parent nodes on a child node, every intermediate and leaf node has a CPT associated with it. For every possible combination of the parent states, there is a row in the child node's CPT that describes the possible state that the child node should be. Nodes with no parents also have CPTs but they only consist of the probabilities for each state of that node. In our model, the CPT tables are generated through a Bayesian learning process given a set of training data. Due to page limitation, we will not discuss this issue here.

(2) Determining the compromise rate of critical system components

We implemented the Bayesian security analysis network using Netica (<http://www.norsys.com>). The Bayesian network takes as input a set of values for the security properties (e.g., a high level of RFID tag tamper resistance ability, a medium level of tag authentication and authorization ability with other system components, and capability of tag robustness/fault tolerance). For a given level of adversary ability, the network probabilistically determines the rate that the tag could be compromised given the input. That information is important for the system administrator to determine if the current security features corresponding to a security requirement specification for an RFID tag can guarantee that the system meets users' security expectations.

Given all the possible combination of different values of each input node in our Bayesian security analysis network (see Figure 5), there are in total $3 \times 3 \times 3 \times 2 \times 3 = 162$ possible results for the goal node "Tag Compromise Rate". We conducted various simulations but only show a few sample results here as summarized in Table 3. Each entry in table corresponds to a testing case, which lists the value for each node of the Bayesian network. Essentially, a testing case corresponds to a security requirement specification proposed for an RFID tag. For instance, the first entry represents the case specifying the security requirements for an RFID tag as follows: (a) it is required to be very tamper resistant (to physical attacks), (b) the communications between tag and the reader are highly secure according to RFID standards, (c) the tag has a relative high level of authentication and authorization ability (relative to general low-cost devices), and (d) the tag has a reasonably high level robustness/fault tolerance. In this case, given the adversary with a high level of capability to attack the RFID system, the Bayesian network determines that the probabilities of the compromise rate of the tag being high (i.e., 0.8), medium (0.5), low (0.1) and negligible (0.005) are 1.02%, 14.1%, 54.7%, and 30.2%, respectively. Due to probabilistic nature of the Bayesian network, a set of values of compromise rate with different probabilities are returned. For each case, the mean compromise rate of the tag, denoted as c , is calculated as the weighted average of those compromise rates with the corresponding occurrence probabilities. The last column of Table 4 shows the mean compromise rate for each testing case.

As the first step of security requirement quantification, the Bayesian security analysis network model determines the compromise rate of an RFID tag. However, for a ubiquitous system (e.g., an RFID system) with a large number of tags distributed components, damages to some components (e.g., RFID tags) do not necessarily mean that the entire system is not useful. Due to the system's self-recovery and fault tolerance abilities, even if some components are compromised, the system may still be trusted to provide an acceptable level of services. In our second step of security requirement quantification, we will measure the security level of a ubiquitous system as a whole given two opposite factors – the compromise rate of individual components and the recovery rate of damaged components.

Table 3: The Bayesian Security Analysis Testing Case

TR	SC	RAA	RFT	AC	RFID Tag Compromise Rate Possibilities				Mean Compromise Rate (c)
					High (%)	Average (%)	Low (%)	Negligible (%)	
Very resistant	High	High	Capable	High	1.02	14.1	54.7	30.2	0.104
Resistant	High	Medium	Capable	High	1.4	16.2	51.3	31	0.109
Very resistant	Medium	Medium	Capable	High	11.8	27.8	32.7	24.2	0.201
Resistant	Medium	High	Capable	High	16.5	17.7	48.1	32.5	0.231
Low resistant	Medium	Low	Capable	Medium	9.49	18.9	37.3	32.5	0.167
Resistant	High	High	Capable	Low	0	1.5	6.2	92.3	0.028
Resistant	Medium	Medium	Incapable	Low	0	1.85	28.1	70	0.047

Measuring the Security Level of a Ubiquitous System

In this section, we study the security level of a low-cost, distributed ubiquitous system (e.g., an RFID system) given the compromise rate of the system components (e.g., RFID tags) and the system's recovery rate. As an important security requirement, virtually all the ubiquitous systems have some intrusion detection and damage recovery mechanisms put in place. Therefore, when an intrusion is detected and some components are damaged, the system can repair and restore the compromised components to their pre-attack clean states. Component recovery can take various forms. For instance, an RFID tag can be reprogrammed when it is compromised or resynchronized with a

legitimate reader if it is out of state with the reader due to malicious attacks or communication failures. The security level of a ubiquitous system, denoted as L , is determined by two factors: how fast the critical system components could be possibly compromised (i.e., the compromise rate) and how fast the compromised components can be successfully recovered (i.e., the recovery rate). We use Formula (2) to represent a generic method to calculate the security level of a low-cost, distributed ubiquitous system, where n_h represents the number of healthy (uncompromised) components and n_t represents the total number of components in the system. The security level is calculated as the percentage of healthy components out of the total number of components in the system.

$$L = \frac{n_h}{n_t} \quad (2)$$

Questions remain regarding how to determine the number of compromised components given the compromise rate of a critical system component and the system recovery ability. Since a compromise rate probabilistically represents how fast a component could be compromised, there is a need for the model to reflect the uncertainty nature of the attack and system recovery. In this paper, we have chosen the stochastic process algebra to simulate the interactions between compromise and recovery activities on RFID tags. The system status at each state can be determined probabilistically. The corresponding security level of the system as a whole can be determined. Research (McDemott, 2005; Zuo, 2010) has already shown that stochastic process algebra is a useful model to simulate attacks and recovery. Stochastic process algebras introduce timing and probability qualifications to pure process algebras. The timed and probabilistic process algebras are well suitable for modeling the concurrent, dynamic interactions between the adversary and a system such as ubiquitous computing with uncertainty (such as the random time for a component to be compromised). The simulation model will determine the probability of each system status, i.e., the number of components compromised given a certain compromise rate and a system recovery rate.

$$L = \sum_{i=0}^{i=n} \frac{S_i * i}{n} \quad (3)$$

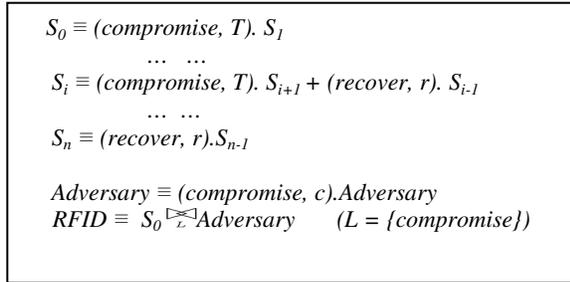


Figure 6: PEPA Model of RFID Attack and System Recovery

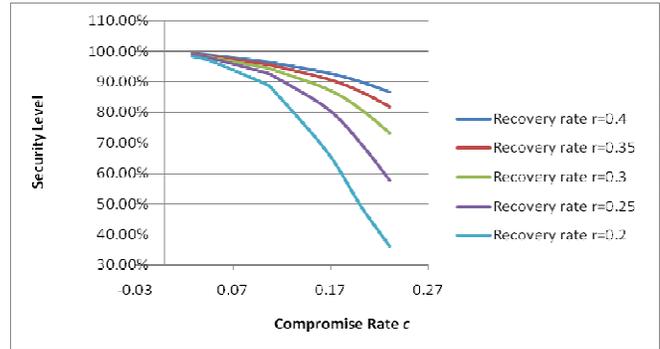


Figure 7: Security Level of an RFID System

We model the stochastic process algebra process using the Performance Evaluation Process Algebra (PEPA) tool (Hillston, 1994). PEPA is a stochastic description technique. Its models are constructed by the composition of components (processes) which perform individual activities or cooperate on shared ones. The basic operators of PEPA model include: (1) activity prefix $(\alpha, r_\alpha).P$: prefixing the activity α to process P to describe its behavior. Activity α has a duration that is negatively exponentially distributed with a variable r_α . Thus, the process algebra can be used to represent a Markov process. The basic elements of PEPA, i.e., the components and activities, correspond to the states and transitions of the underlying Continuous Time Markov Chain; (2) process choice $P+Q$: the behavior of the system could be either P or Q depending on some random variables. If one process is chosen, the activities of that process will be complete before the other; and (3) a cooperation operator $P \overset{\Delta}{\underset{L}{\bowtie}} Q$: a cooperating process synchronizes on the activities in the cooperation set L . The activities in L will occur together in processes P and Q but with the duration of the slower activity.

In our simulation, the attacks on RFID tags are represented as stochastic faults whose occurrence is predicted by the tag compromise rate as we discussed in earlier. The RFID system is modeled as interactions between the adversary process and the RFID tags. The attack-recovery process is executed for an extended time so that the stochastic

model representing the interactions of the attack and the RFID system reaches a set of steady states. Each state (call RFID *system status* in the following discussions to avoid confusion with other similar terms) represents an equilibrium point, i.e., the number of compromised tags and the number of recovered tags are equal. Due to the random nature of the stochastic algebra process, the RFID system has a probabilistic occurrence at every status in the state space. For instance, for an RFID system with n tags, there are $(n+1)$ possible status S_0, S_1, \dots, S_n , which represent that $0, 1, \dots, n$ tags are compromised, respectively. Each status S_i ($0 \leq i \leq n$) has an occurrence possibility p_i . Given the stochastic description of the RFID system under attack, the security level of the system as calculated using Formula (2) can be refined to incorporate the probabilistic nature of different system status to reflect the uncertainty of attack and recovery effects. The refined security level calculation is shown in Formula (3). Our PEPA simulation model will determine the possibilities of different system status given the tag compromise rate c and system recovery rate r .

The PEPA model representing attacks on the tags and system recovery is shown in Figure 6. For the sake of simplicity, we considered a small RFID system comprising of only 10 tags. However, it is straightforward to generalize to larger systems. The model has two components: tags and attacker. The process S_i represents the status of the RFID system where i tags have been compromised. Each S_i process has two possible behaviors: (1) moving to next status S_{i+1} , indicating that one more tag is compromised; and (2) going back to the previous status S_{i-1} , indicating that one compromised tag is recovered. The behaviors of the adversary are modeled by the PEPA process *Adversary* with a compromise rate c . The activities of the model are (1) *compromise*, representing the corruption of a tag. This activity has an unspecified rate denoted by the symbol T . This rate will be determined by the tag compromise rate c when the two processes *Attacker* and S_i are synchronized; and (2) *recovery*, representing the recovery of a compromised tag with a recovery rate r . The recovery rate of a tag by the RFID system is determined by the system recovery and repair ability. We assume this rate is known. The RFID system is modeled using the PEPA cooperation operator $S_0 \bowtie_L \text{Adversary}$ (where $L = \{\text{compromise}\}$) to represent the interactions of the adversary and the tags via synchronized participation in the event *compromise*.

The PEPA model has been solved using the PEPA Eclipse Plug-in software. Given the occurrence probability of each RFID system status, we can apply Formula (3) to calculate the security level of an RFID system. As an example, for a given recovery rate² (e.g., $r=0.3$), if the (mean) compromise rate of a tag is 0.104 and the adversary capability is determined as high, then the security level of the RFID system is 94.69%. This indicates that about 94.69% of total number of tags are healthy (i.e., not compromised by the adversary) given the two opposite effects of malicious attacks and system recovery.

Since a system recovery rate has a significant impact on the security level of a ubiquitous system, we executed our PEPA simulation model and calculated the different security levels of an RFID system with various combinations of tag compromise rates and system recovery rates. Figure 7 shows the results, where we can see that a higher compromise rate or a low recovery rate results in a lower security level of the RFID system. Furthermore, we can observe that the security level under a smaller recovery rate (e.g., $r=0.2$) is more sensitive to any small change in compromise rate. For instance, given a lower recovery rate, the system's security level decreases more significantly for the same increase of the compromise rate as compared with the situation with a higher recovery rate. This has an important implication to the system administrator in determining security budgets. Any investment to improve recovery rate is always beneficial because it improves the system security level to a greater extent.

Putting Everything Together

We now summarize the major components of our framework and show how to integrate them to determine whether a specific security requirement specification proposed for a low-cost, distributed ubiquitous system can guarantee that the system has a required level of security. The flowchart to apply our framework is shown in Figure 8 and the four major steps are explained next.

Step 1: The process of security property identification as shown in the qualitative analysis of our framework determines a set of security properties $\{sp_1, sp_2, \dots, sp_n\}$. A security requirement specification represents user's

² We assume that the system recovery rate is available since it depends on the system's inherent features determined by its design and implementation, which are supposed to be known to the system administrators.

security requirements for the system and specifies the values for those security properties. Since there could be a set of plans for different security requirement specifications, $SRS_1, SRS_2, \dots, SRS_i, \dots, SRS_n$, we assume that the system administrator chooses a security requirement specification SRS_i with an acceptable level of security cost (e.g., the one with the minimum cost) and see if the security level of the system based on SRS_i can meet the users' security criteria. Suppose that for the set of n security properties identified, the corresponding n values for SRS_i are represented in a set $\{a_{i,1}, a_{i,2}, \dots, a_{i,n}\}$. If those security requirements are satisfied, then the system has the set of desired security features.

Step 2: By applying the Bayesian security analysis network model, the security features of the system (as determined in step 1) determines a compromise rate of the critical components of the ubiquitous system given an adversary capability. The compromise rate has a set of values $\{c_1, \dots, c_m\}$ with their corresponding probabilities of occurrence $\{p_{c1}, p_{c2}, \dots, p_c\}$. A mean compromise rate of the tag, represented as c , is calculated using Formula (4).

$$c = \sum_{j=1}^{j=m} c_j * p_{c_j} \quad (4)$$

Step 3: Given the (mean) tag compromise rate c and a system recovery rate r , the PEPA model generates the possibilities of the system in different status S_0, S_1, \dots, S_n . As discussed earlier, each status S_i ($0 \leq i \leq n$) represents the system state where i out of n components have been compromised.

Step 4: The security level of the ubiquitous system is calculated using Formula (3). If a pre-defined threshold, denoted as L^* , has been set up to represent the user's minimum required security criteria for the system, then the system administrator can decide if the proposed security requirement specification in step 1 can ensure that the system satisfies the user's security expectations. As shown in Figure 8, if the calculated security level L given the specific security requirements is greater than the user's required security level L^* , i.e., $L \geq L^*$, then the security requirement specification (as represented as the values of the security properties, $\{a_1, a_2, \dots, a_n\}$) can be accepted. As a result, the security requirements can be enforced in the system design, development, or security evaluation processes since they can guarantee that the user's required security level can be satisfied. Otherwise, the security properties must be refined towards higher criterion or another security requirement specification SRS_j ($i \neq j$) should be selected so that a higher level of security can be resulted (by applying the above steps 2-4).

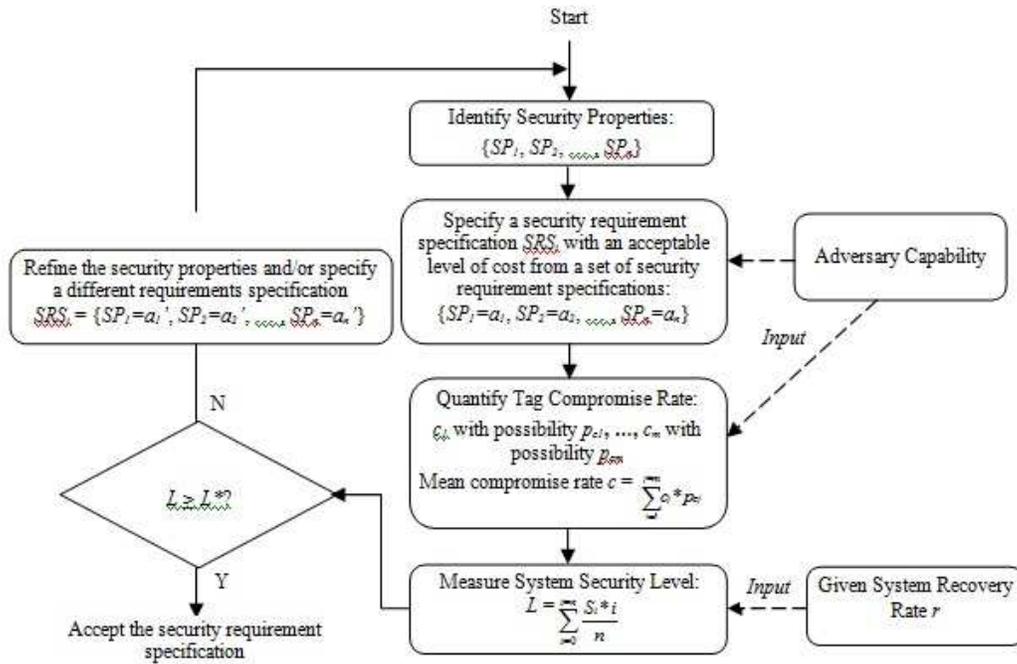


Figure 8: Flow Chart of the Framework

To illustrate, we have conducted a set of simulations by following the flow chart to quantify the security level of a ubiquitous system under different sets of security requirement specifications. The results are summarized in Table 4 (we only show the results given the recovery rate $r=0.3$). Each entry in the table represents a case with the set of identified security properties for a ubiquitous system, the values of those security properties given a particular

system (i.e., a security requirement specification for the ubiquitous system), the mean compromise rate of a system component, the adversary capability, the tag recovery rate, and finally the security level of the ubiquitous system. For instance, as shown in the first entry of the table, given a security requirement specification for a system component (e.g., an RFID tag), i.e., {high levels of tamper resistance, secure communications, authentication/authorization ability, and robustness/fault tolerance}, and a high level of adversary capability, the mean component compromise rate is calculated as 0.104. The security level of the system is calculated as 94.69% (given the system recovery rate $r=0.3$). This security level is above the pre-defined threshold ($L^*=90\%$), and therefore the proposed security requirement specification can be accepted. Other entries in Table 4 can be interpreted in the same way.

Table 4: Security Requirement Quantification Cases

Security Requirement Specification				Adversary Capability (AC)	Mean Tag Compromise Rate (c)	Tag Recovery Rate (r)	System Security Level	Security Level Threshold	Security Requirements Acceptable?
TR	SC	RAA	RFT						
Very resistant	High	High	Capable	High	0.104	0.3	94.69%	90%	Yes
Resistant	High	Medium	Capable	High	0.109	0.3	94.29%	90%	Yes
Very resistant	Medium	Medium	Capable	High	0.201	0.3	81.06%	90%	No
Resistant	Medium	High	Capable	High	0.231	0.3	73.10%	90%	No
Low resistant	Medium	Low	Capable	Medium	0.167	0.3	87.62%	90%	No
Resistant	High	High	Capable	Low	0.028	0.3	98.14%	90%	Yes
Resistant	Medium	Medium	Incapable	Low	0.047	0.3	94.29%	90%	Yes

Our framework can assist system administrators in answering various what-if questions. For instance, if we change some security requirements for the critical components of a system, can we still ensure that the system satisfies users' security criteria? Can we make some aspects of the system security features stronger and in the mean time relax some other requirements for the system while the system can still maintain a satisfied level of security? What will be the security requirements given a different adversary capability, after the threat model shows additional evidence about the possibility of the attacks? The answers to those questions can make the system design and security specification more flexible. In some applications, certain aspects of the system are hard to implement towards a higher level of security (e.g., it is difficult to require a higher level of fault tolerance for low-cost ubiquitous devices such as RFID tags due to their computational and memory constraints). Our framework allows the system designers and administrators to empower other aspects of a system to "compensate" for some relative weak areas that may be technically challenging to implement.

Conclusion

From the perspective of system engineering, security requirement analysis is the important first step in security specification and compliance evaluation. In this paper, we present a framework for security requirement analysis for the class of low-cost, distributed ubiquitous system with three major components: identification of essential security properties, determination of the compromise rates of critical system components, and measurement of security level of the system given a security requirement specification. Our framework is suitable for such a ubiquitous system with a large number of small, inexpensive, networked processing devices (e.g., RFID tags, wireless sensors, and Ad hoc notes). We used an RFID system as a running example throughout the paper to illustrate our ideas.

Acknowledgement

This work was supported in part by US AFOSR under grant FA 9550-09-1-0215. The author is thankful to Dr. Robert Herklotz for his support, which made this work possible.

References

- Abouzakhar, N., Gani, A., Manson, G., Abuitbel, M. and King, D. 2003. "Bayesian Learning Networks Approach to Cybercrime Detection," in *Proceedings of the 2003 PostGraduate Networking Conference*, Liverpool, United Kingdom.
- Akyildiz, I., Su, W., Sankarasubramaniam, Y. and Cayirci, E. 2002. "A Survey on Sensor Networks," *IEEE Communications Magazine* 40(8), pp.102-114.
- Burmester, M. and Medeiros, B. 2007. "RFID Security: Attacks, Countermeasures and Challenges," in *Proceedings of 5th RFID Academic Convocation, the RFID Journal Conference*, April 30 - May 2, Orlando, USA.
- Glover, B. and Bhatt, H. 2006. *RFID Essentials*, O'Reilly Publisher.
- Herath, H. and Herath, T. 2008. "Investments in Information Security: A Real Options Perspective with Bayesian Post-audit," *Journal of Management Information Systems* 25(3), pp.337-375.
- Hillston, J. 1994. "A Compositional Approach to Performance Modeling," Ph.D. Thesis, University of Edinburgh, CST-107-94.
- Jensen, F. 2001. *Bayesian Networks and Decision Graphs*, Springer-Verlag New York.
- Karygiannis, T., Eydt, B., Barber, G., Bunn, L. and Phillips, T. 2007. "Guidelines for Securing Radio Frequency Identification (RFID) Systems," Special Publication 800-98, National Institute of Standards and Technology. Available: <http://csrc.nist.gov/publications/nistpubs>.
- Mead, N. 2003. "Requirements Engineering for Survivable Systems," Technical Report, Carnegie Melon University.
- Mehdi, M., Zair, S., Anou, A. and Bensebti, M. 2007. "A Bayesian Networks in Intrusion Detection Systems," *Journal of Computer Science* 3(5), pp. 259-265.
- McDemott, J. 2005. "Attack-potential-based Survivability Modeling for High-consequence Systems," in *Proceedings of third IEEE International Information Assurance Workshop*, Washington DC, USA.
- Mitrokotsa, A., Rieback, M. and Tanenbaum, A. 2009. "Classification of RFID Attacks," *Information System Frontiers: A Journal for Innovation and Research*, Springer Netherlands, 2009.
- Myagmar, S., Lee, A. and Yurcik, W. 2005. "Threat Modelling as a Basis for Security Requirements," in *Proceedings of Symposium on Requirements Engineering for Information Security (SREIS)*, Pairs, France.
- Oladimeji, E., Supakkul, S. and Chung, L. 2006. "Security Threat Modeling and Analysis: A Goal Oriented Approach," in *Proceedings of the 10th IASTED International Conference on Software*, Dallas, TX, USA.
- Schneier, B. 1999. "Attack Trees", *Dr. Dobb's Journal of Software Tools* 24, pp. 12-29.
- Sebyala, A. Olukemi, T. and Sacks, L. 2002. "Active platform security through intrusion detection using naive Bayesian network for anomaly detection," in *Proceedings of the 2002 London Communications Symposium*, London, UK.
- Toh, C. 2002. "Ad Hoc Mobile Wireless Networks", Prentice Hall Publishers.
- Westmark, V. 2004. "A Definition for Information System Survivability," in *Proceedings of the 37th Hawaii International Conference on System Sciences*, Hawaii, USA.
- Yuksel, K. 2004. "Universal Hashing for Ultra-low-power Cryptographic Hardware Applications," Master's Thesis, Worcester Polytechnic Institute, USA.
- Zia, T. A. 2008. "A Security Framework for Wireless Sensor Networks", Ph.D. Dissertation, the University of Sydney.
- Zuo, Y., Pimple, M. and Lande, S. 2009. "A Framework for RFID Survivability Requirement Analysis and Specification," in *Proceedings of International Joint Conference on Computing, Information and Systems Sciences and Engineering*, Bridgeport, CT, USA.
- Zuo, Y. 2010. "RFID Survivability Quantification and Attack Modeling," in *Proceedings of 3rd ACM International Conference on Wireless Network Security*, pp. 13-19, Hoboken, NJ, USA.
- Zuo, Y. 2010. "Survivability RFID Systems: Issues, Challenges, and Techniques," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, Special Issue on Availability, Reliability and Security*.