Embedding Security Functionality In Formal Specifications Of Requirements

Gregory W. Ulferts, (gregory.ulferts@udmery.edu), University of Detroit Mercy
Antonio Drommi, University of Detroit Mercy
Daniel Shoemaker, University of Detroit Mercy

ABSTRACT

The methodology in this paper will let designers specify the security properties defined through the functional families of the ISO/IEC 15408 Standard, graphic representations. This blueprint will allow both business and technical participants, to discuss and refine a common solution. It also serves as a roadmap, to guide the implementation process. We feel this can become a useful supporting methodology for the construction of effective security responses, because it ensures both the widest possible participation in the design process as well as the greatest degree of understanding. The fact that the advice of the world's experts is readily available and easy to use as a result of this process might also serve to make the mission to protect America's information assets a little more effective.

Keywords: Information assurance, best practice, standards, UML modeling, information security system (ISMS) implementation

INTRODUCTION: HOW ARE WE DOING SO FAR?

The security of America's information infrastructure is a critical concern in a world where "terrorism" is a household word. It is especially important because every part of our economy from defense, through manufacturing, finance and banking, telecommunications, public health, energy, transportation and emergency services, right down to agriculture and the post office, depends on reliable information for even the most basic activities. As a consequence, advancing our capability in safeguarding that information is a vital national interest.

Security is an increasingly important matter because the threats become more compelling every day. According to the CERT Coordinating Center (CERT/CC, 2003), which is the most authoritative reporting agency in the United States, there were 319,992 security incidents recorded in the period 1988-2003. Each of these represented a discrete event that entailed one of four attack types, unauthorized access, malicious code, denial of service, or inappropriate usage. The important point to note is that, 219,623 of these events occurred in the years 2002 and 2003, which means that sixty nine percent of all attacks have taken place in the past two years.

The same is true for vulnerabilities. Product vulnerabilities are flaws that might allow an attacker to usurp privileges, regulate operation, or compromise data on a users system. There have been 12,946 product vulnerabilities reported to the CERT/CC over the past sixteen years. Of these, 7,913 were recorded for the years 2002 and 2003. Or in real terms, almost two thirds of the known product vulnerabilities have surfaced since 9/11.

There are probably a lot of factors that underlie these statistics, but the direction is clear. America's information assets are increasingly under attack and as a consequence, an effective and continuously evolving defense is an absolute necessity. Unfortunately, the evidence so far indicates that the adequacy of that protection is not what it should be. The U.S. General Accounting Office (GAO, 2002) has done one of the most thorough studies of that question and it found significant weaknesses in the national information infrastructure. Moreover it hypothesized that, as the body of audit evidence grows it is likely that… "Additional significant deficiencies will be identified". The major findings of the GAO study were:

- Risk-based security plans not developed for major systems
- Security policies not documented
- Programs for evaluating the effectiveness of controls not implemented
- Controls for application development and change control not implemented
- Inadequate control over the implementation and use of software products
- No expertise to select, implement, and maintain security controls

It is important to note, that the collective weaknesses identified in this study were procedural, not technical. That is, most of the failings were in the area of process and infrastructure, not technology. The tangible outcome of this failure is that eighty percent of U.S. businesses lose an average of $2.5 million dollars a year to cyber incidents and that cost is growing (CSI, 2003).

THE PROBLEM OF DIVERSITY

The problem originates in corporate culture itself. It is hard to criticize the IT function for failing to adequately assess business risks, document organizational policies; select and implement organization-wide controls, or adequately manage software development and use. That is because those responsibilities all rest with executive management.

Nevertheless, it is also difficult to fault America's corporate executives for not knowing what to do. That is because information assurance is a very complex topic. Proper information security embodies substantive activities from a diverse range of disciplines, including business, computer science, ethics, law, mathematics and military science (Spafford, 1998). Each of these areas contributes to the overall goal of information protection and each is essential to formulating an effective solution. Yet their diversity also illustrates the root of the problem, which is that it is hard to know which bases to touch.

The practical result of this lack of clarity is that accountability for the various critical functions of the process is dispersed throughout the organization, and thus diffused. For example, such disparate entities as strategic planning, accounting, IT/IS and even human resources all share some part of the responsibility for assurance. Yet they rarely communicate with each other, let alone work together in a coordinated way to ensure corporate security. An anonymous CSO quoted in CSO Magazine's "Undercover" feature describes the consequences of that lack of coordination… *"What's missing in most companies is the concept of an organizational vision and voice for the protection mission* (CSO, 2004).

SUPPORTING UNDERSTANDING THROUGH EXPERT MODELS

Some part of that lack of vision stems from the fact that security solutions are necessarily multi-layered and multifaceted and all of the components are essential to establish a coherent response. The black-hat community is very smart. If it can find a hole in a defense it will exploit it and it is always looking. Accordingly, all of the required controls have to be in place and seamlessly embedded in day-to-day operation in order to prevent that.

However, because most organizations approach that responsibility with the same astute vision as the six blind men and the elephant the outcome is typically a set of uncoordinated actions, where "though each was partly in the right they all were in the wrong". The specific consequence of that lack of coherence can be seen in the abysmal statistics laid out in the survey's cited here (CSI, 2003, DTI 2001, CERT, 2003, GAO, 2002), and every other current study that has been done in the past five years.

So clearly, the critical first step in building a COMMON understanding of the ENTIRE problem is to get everybody who is involved in security on the same page. The primary disconnect in that process is the fact that none of the participants speak the same language. For example, the CEO and the CFO are responsible for the policies that are implemented by the system programmers but there are probably no two different species on the planet and they would never be caught talking to each other. Yet they must all share a common vision of the problem and the steps that must be taken to solve it in order to define and implement an effective response.

That necessity, that all of the nuances of the situation must be understood, is a particular barrier to success. That is because there are always a lot of conflicting points-of-view and political undercurrents swirling around the process and information assurance is an intangible condition, not a concrete object. So there is no such thing as an obvious solution that everybody can see and agree on. Instead, the specific requirements for each situation have to be made tangible by some sort of formal process of description, which for lack of a more meaningful term is called design.

The ambiguous understanding of the components of a proper security response is the reason why business and public agencies worldwide have come to rely on expert models for guidance. And for that reason, standards like ISO 17799/BS 7799, DoD 5200.28 (TCSEC), GASSP, COBIT, and even the NSRB's proprietary IBOK and ISC^2's CBK are used by organizations to facilitate the requisite understanding and communication process.

BASING THE DESIGN ON A COMMON PERSPECTIVE

From the standpoint of the scope and depth of the security requirements it contains, the ISO/IEC 15408 international standard, which is called the "Common Criteria", is probably the most detailed of these models. 15408 is a catalogue of security attributes that could potentially be embodied in any given security application. It is meant to provide all of the security advice necessary to for software to address most ordinary threats. As such it is also an extremely effective reference point for selecting and embedding a detailed set of security best practices within an organizational application.

The purpose of 15408 is to provide the worldwide user community with a standard method for evaluating software applications. The standard employs a set of commonly accepted criteria for trusted systems (hence the name). It is considered to be complete and it is assumed to satisfy the overall security requirements of most situations. 15408 enumerate all of the known security attributes that can be confirmed through direct observation. The intention is to provide an encyclopedic collection of standardized security properties, which might be adapted to any situation depending on the specific policies and assumptions driving the process.

Its original intent was to support the evaluation of software products by providing an exhaustive set of attributes to benchmark inherent security behaviors against. However because of the extent of those criteria, 15408 can also be assumed to provide a common set of desirable behaviors that might be present in any security situation. The advantage of 15408 is that, because of its focus on observation, these behaviors are commonly understandable and therefore easier to visualize and implement.

The level of agreement that this common understanding underwrites can also be employed to further develop the process. Managers can use the standard properties itemized by 15408 to verify that the security objectives and dependencies specified in any security system design will satisfy the requirements of best practice. These security properties are represented by ISO as being the current state of the art in assurance. The advantage of the standard is that it substantiates every requirement that is considered to be known and valuable.

EXPRESSING COMMON CRITERIA REQUIREMENTS

There are three parts to the standard: Part 1: Introduction and General Model, Part 2: Security Functional Requirements and Part 3: Security Assurance Requirements. The security functional requirements are itemized in the 15408 part 2. These are extensive and in text form, which makes them difficult to utilize in a design process, particularly if common understanding is required.

In order to support the proper level of understanding it is important to have graphic model of the entire process. The old adage of a picture being worth a thousand words is true when people are trying to understand complex relationships. It is particularly true when they are coming from a number of different cultural directions, as they are when the business side sits down with the technical people. Consequently, almost the only feasible way to represent a security system design is by providing a blueprint, which everybody can see and grasp.

There are many techniques that can be employed to accomplish this but the most widely accepted approach utilized in IT work today is the object model, which is the approach that we adopted for this paper. We did that for two reasons. First, and most importantly the attributes of the common criteria specifically lend themselves to object representation. But another equally compelling reason was the ease of communication that objects modeling, particularly use-case diagrams, provides for non-technical users. Given that, in this paper we will demonstrate how best practice solutions among a diverse set of participants can be developed using the advice of the Common Criteria and an object modeling technique.

DESCRIBING SECURITY REQUIREMENTS USING THE UML

Since our approach employs an object technique we adopted the Unified Modeling Language (UML) as the means to describe the security requirements that are contained in the second part of the standard. UML is the common graphical language used for modeling abstract functions in IS work. Its purpose is to "*visualize, specify, construct and document the artifacts of a software intensive system*" (Booch, 1997). It allows inexperienced users to both understand general functional requirements as well as produce a graphical representation of them. It is also possible to describe abstract requirements in a complete and unambiguous fashion using this technique.

The class diagrams that we produced using this method translate the common criteria best practice recommendations into a complete set of easy to understand and apply object models. Using this approach we transformed all the functional requirements of each individual component in ISO/IEC 15408 part 2 into a class structure using the UML Signature (syntax). The outcome of that process was a 150 page document which space restrictions prevent us from presenting here. However, as an illustration, we will explain how we did this for one requirement. That approach will apply in every other instance.

MODELING SECURITY FUNCTIONS USING CLASS DIAGRAMS

The technique is based around UML class diagrams. A class diagram details a set of classes, interfaces, and collaborations and their relationships. Class diagrams are used to illustrate the static view of a system. Operations in a class are designated by strong verbs that describe how to implement the functionality of the objects. Therefore they should be named in a manner that effectively communicates that functionality.

For people unfamiliar with UML classes are depicted as a three compartment rectangular box, the first compartment representing the Name of the class, the second representing the attributes and the third one representing the operations performed. In UML classes are modeled by relationships. These relationships are indicated in the form of lines that document the specific Association, Dependency and Inheritance relationships.

EXAMPLE OF THE APPLICATION OF UML TO THE COMMON CRITERIA

Let's Consider the Class FCO: Communication from ISO/IEC 15408-2:-199(E). This class provides security specifications that are related to proof of origin and proof of receipt. In total the family ensures that an originator cannot deny having sent the message, nor can the recipient deny having received it. For our explanations let's take only proof of origin.

Example:

Reference: FCO_NRO.1 Selective proof of origin [3, pg no 33]
Hierarchical to: No other components

FCO_NRO.1.1 The TSF shall be able to generate evidence of origin for transmitted [assignment: list of information types] at the request of the [selection: originator, recipient, [assignment: list of third parties]].

FCO_NRO.1.2 The TSF shall be able to relate the [assignment: list of attributes] of the originator of the information, and the [assignment: list of information fields] of the information to which evidence applies.

FCO_NRO.1.3 The TSF shall provide a capability to verify the evidence of origin of information to [selection: originator, recipient, [assignment: list of third parties]] given [assignment: limitations on the evidence of origin].

The first step in the process is to understand the behavior of the component. This can be done by creating a use case diagram that details the form of the requirement (Booch, 1997). A use case specifies the behavior of a system, or a part of a system, and is a description of a set of sequence of actions. People utilize use cases to capture the intended actions. Each sequence represents the interaction of the things outside the system with the system itself and its key abstractions. Their advantage is that these can be easily developed with the participation of even the most naïve end users, which supports common understanding.

This process requires the identification of the actors. From the text of the requirement we can identify the actors as three groups' *originator, recipient* and *list of third parties*. These actors can be human users or another system. The next step is to write the sequence.

- FCO_NRO.1.1 (use case one **Generate evidence**) - The actor (Recipient or the third party) makes a request to the system for evidence of origin; the system generates the evidence of origin provided users are eligible to receive that information.
- FCO_NRO.1.2 (use case two **Relate evidence**) - then the system relates the attributes of the originator (originator identity, time of origin etc...) and the information transmitted (body of the message).
- FCO_NRO.1.2 (use case 3 **Authorize user**) the user makes a request to the verification of origin. The system verifies whether the users are permitted to access. If so then the system will enable the user to verify the evidence of origin under the given conditions.

Here we have three use cases and we have three types of users (originator, receiver and third party) who interact. In a use case diagram the cases are represented as an ellipse and the actors are represented as a line drawing of a human. This is a typical use case description for the above requirement. Following that it is necessary to develop an interaction diagram to describe these flows. This takes place as we refine our understanding of the requirement and its outside interactions. It is embodied in a SEQUENCE diagram, which specifies the use case's main flow. However, before generating the SEQUENCE diagram, it is important to understand the static structure of the target build. This is done thru a class diagram.

In the class diagram we grouped a set of real time objects, which embody a common set of attributes and behaviors into a specific higher-level abstraction called a class. Then these different classes were joined by relationships. The resultant class diagram shows the basic architecture of the system. The process of developing the understanding needed to draw the class diagram starts with identification of the objects in the functional requirement set.

In our example we identify the classes as security functions (TSF) required, Originator, Recipient, Third parties and an Interface. The interface in the case of the example we have selected is" Timing of Identification" (shown in Figure 1.2). These are specified as <name of the class> in the first compartment of the class diagram.

Next we identify the characteristics of the class, which are listed in the second compartment; finally the behavior of each class is identified and listed in the third compartment.

Once the classes are depicted they are related by three kinds of relationships association, dependency and inheritance. An association relationship is bi-directional relationship shown as a simple line connecting two classes. Dependency is a relationship, which implies the functionality of one class is depending on the functionality of the other. This is shown as a dotted line with an open arrow. Inheritance is a relationship, which indicates the parent child relationship indicated by a filled arrowhead towards a parent (shown in Figure 1.1). The roles of each object can be depicted by designating it over the relationship line and also the multiplicity (one to many or many to many) is also designated using the same process.

Here (for illustration purposes only) we have supplied the label to indicate that relationship. One of the important aspects of class diagram is the visibility factor. That can be used in the security functionality description. This is part of the general UML technique. Using this notation we can indicate which of the attributes and operations are private, public and protected by the following signs in order -, + and #. Figures 1.1 and 1.2 on the following pages illustrate how a class diagram can be drawn for a single security function Selective proof of origin.

The next step in the UML process is to develop a sequence diagram. This is illustrated in Figure 1.3. Sequence diagrams are necessary to complete the description of the interaction because they provide the time ordering of the messages. Graphically, a sequence diagram is a table that shows objects arranged along an X axis and Messages, ordered in increasing time, along the Y axis. This drawing makes it increasingly easy to understand the precise functional requirement and the dependencies of the other objects. Thus it is possible to apply this same process to turn all of the security functional requirements classes into class diagrams. Since this involves the specification of 66 class structures the entire set will not be presented here. However, as this paper demonstrates, the UML representation of the security components in 15408-2 can help anybody develop a correct and understandable model of the security system.

CONCLUSION

This introduces a methodology that can be used to specify security best practices for a given assurance situation. It is based on the security advice embodied in the ISO/IEC 15408 Standard, popularly known as the “Common Criteria”. That Standard itemizes an exhaustive set of security attributes, a tailored subset of which can be assumed to apply to all instances.

The methodology described in this paper allows designers to transform the security properties specified by 15408 into a graphic representation of a particular response. This common blueprint is one of the easiest and most feasible ways for all of the stakeholders and participants, both business and technical, to discuss and refine the solution. More importantly it also serves as a mutually acceptable roadmap, to guide the implementation process.

We have developed a UML drawing for every one of the eleven general classes and 67 sub-functions contained in the Common Criteria. We have provided an example to demonstrate how that process works here. We feel this can become an effective supporting methodology in the construction of effective security responses, because it ensures both the widest possible participation in the design process as well as the greatest degree of understanding.

More importantly it also allows organizations to benefit from the best practice advice of the world’s experts. It is not an insignificant advantage that a great deal of the thinking about what constitutes correct security practice has already been, and will continue to be, done by ISO. The fact that this expert advice is readily available and easy to use might serve to make the protection of America’s information assets a little more effective than it has been in the past is both a benefit to the profession and to the society at large.

REFERENCES

1. Booch, Grady, James Rumbaugh and Ivar Jacobson, *The Unified Modeling Language User Guide*, Addison-Wesley, 2002?
2. Critical Infrastructure Taskforce, National Strategy to Secure Cyberspace (Draft), Department of Homeland Security, September 18, 2002
3. Computer Security Institute (CSI), *Results of 2001 Joint Survey on Computer Security* (with the Federal Bureau of Investigation – FBI) CSI, San Francisco, 2002
4. Department of Trade and Industry, *Information Security Breaches Survey,* DTI Great Britain, 2001
5. *Internet Business News*, CSI survey, FBI/Computer Security Institute, April 8, 2002
6. ISO/IEC 15408-1 First Edition 1999-12-01
7. ISO/IEC 15408-2 First Edition 1999-12-01
8. Ritt, William, Government Information Security Reform – The First Year, General Accounting Office, 2002
9. Rumbaugh, James, Michael Blaha, William Premerlani, Fredrick Eddy, William Lorensen, *Object Oriented Modeling and Design*, (publisher, date)?
10. http://www.modelingstyle.info , accessed March 2004

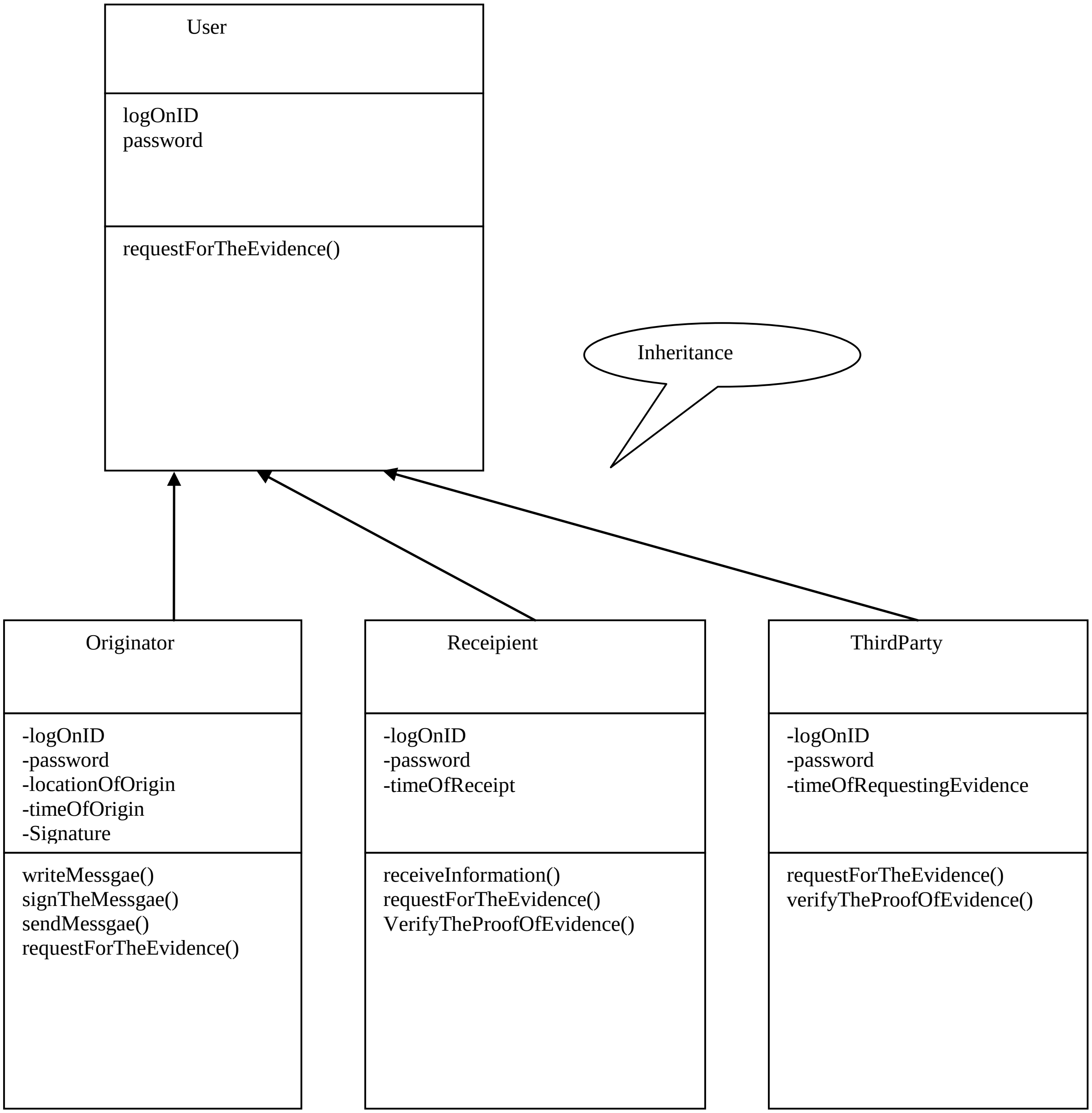

Figure1.1 Class diagram Illustrating the Inheritance of the user group

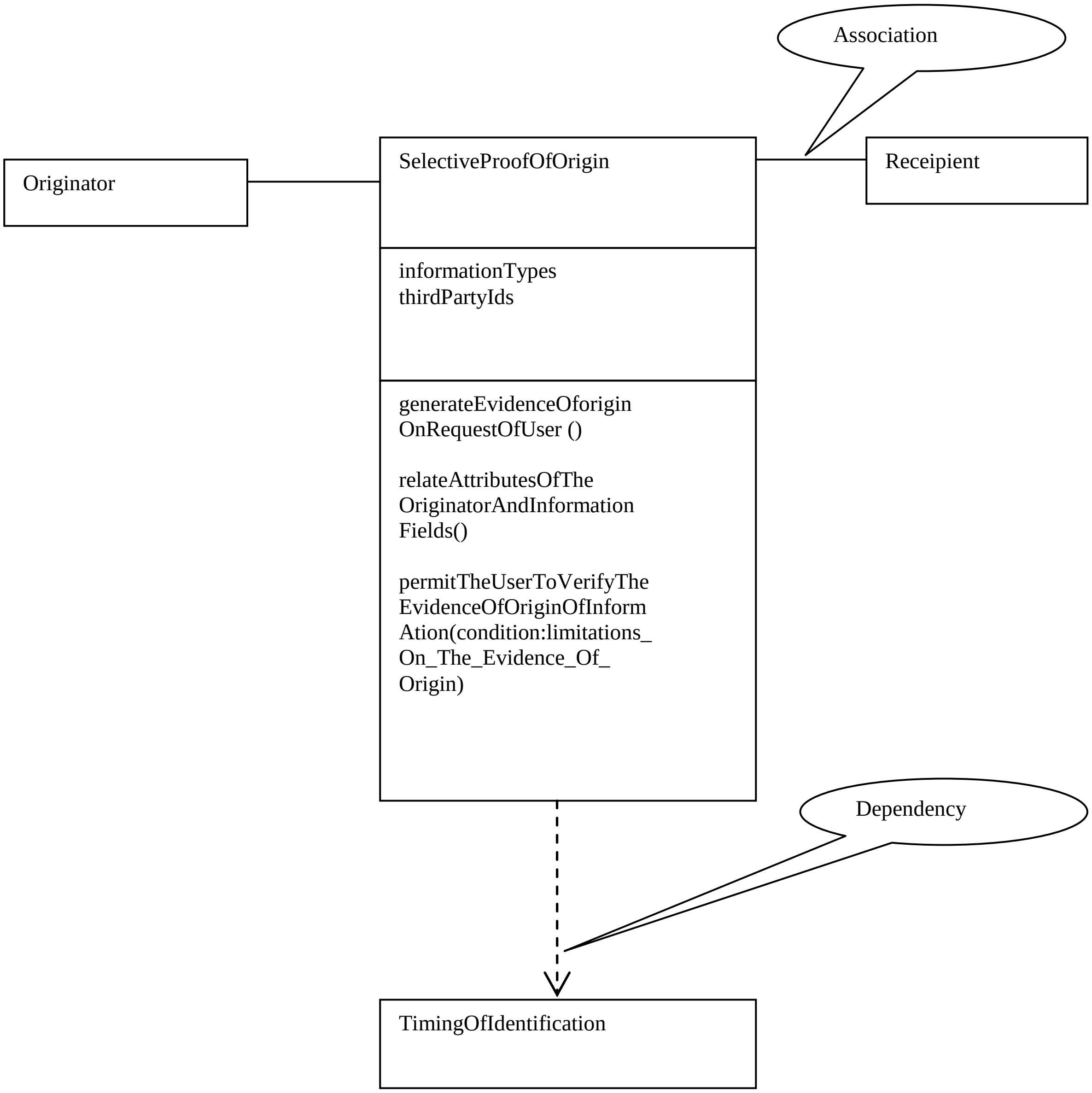

Figure 1.2 Class Diagram illustrating The Security Component

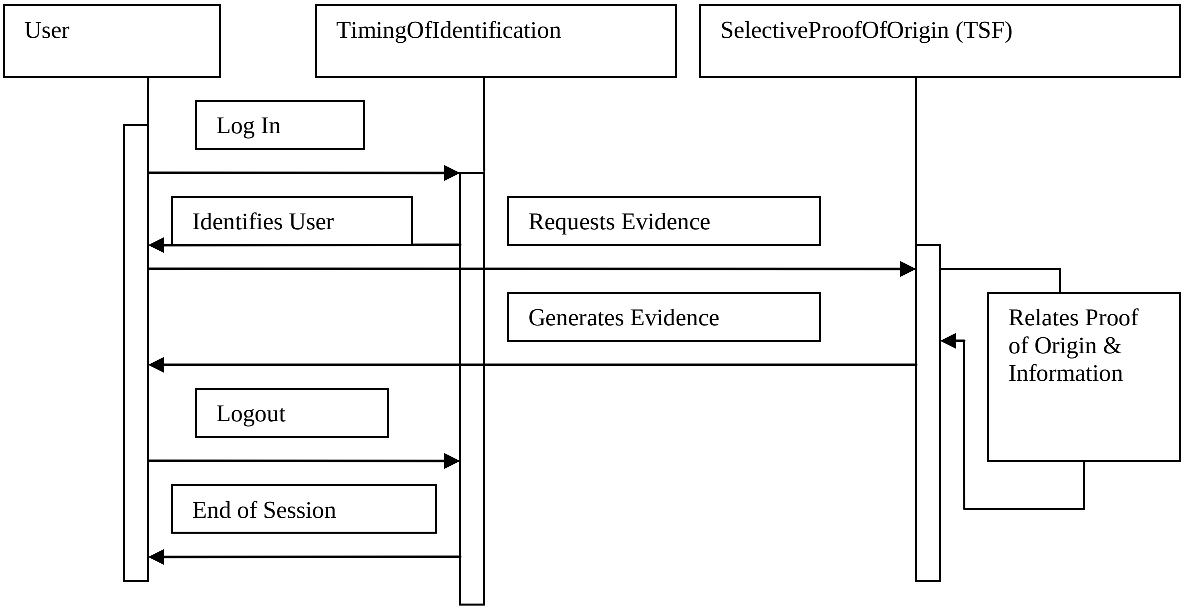

Figure1.3 Resulting Sequence Diagram