A Comparative Evaluation of Three Approaches to Specifying Security Requirements

Mamadou H. Diallo, Jose Romero-Mariona, Susan Elliott Sim, Thomas A. Alspaugh, and Debra J. Richardson

Donald Bren School of Information and Computer Sciences
University of California, Irvine
444 Computer Science Building, Irvine, 92697-3425, USA
{mdiallo, jromerom, ses, djr}@ics.uci.edu

Abstract. As software systems and networks continue to evolve, so do threats to their security. Unfortunately, most security issues come to light only after completion of the system because security is often managed in an ad hoc fashion late in the software lifecycle. There are many advantages to incorporating security specification into the requirements phase and a number of approaches have been proposed. In this paper, we present a comparative evaluation of three such approaches: The Common Criteria, Misuse Cases, and Attack Trees. We applied each of these approaches to a common problem, a wireless hotspot, and evaluated them for learnability, usability, solution inclusiveness, clarity of output, and analyzability. We found that each approach has strengths and weaknesses, and that they can be complimentary when combined. The Common Criteria are difficult to learn and use, but are easy to analyze. Misuse Cases are easy to learn and use, but produces output that is hard to read. In contrast, Attack Trees produce clear output, but are difficult to analyze.

Keywords: Security, method, evaluation, case study, wireless network

1 Introduction

As software systems become larger and more connected, security threats that compromise these technologies are also growing [3]. However, most of the current software development practices deal with security concerns after completion of the system. If a security vulnerability problem is discovered later, protection is added in an ad hoc manner. As a result, our confidence in the security of such systems is low. There are many advantages to confronting security as early as possible in the software lifecycle, in particular, in the requirements phase.

Many techniques and approaches have been proposed to specify security requirements. While these have been applied to a variety of problems and domains, there is no work
that we know of that has compared and evaluated them. In this paper, we report a critical
analysis and comparison of three popular security requirements techniques. These three
techniques are the Common Criteria, Use and Abuse Cases, and Attack Trees. Even
though all these three techniques facilitate the specification of security requirements,
they have differences in their approaches which impact their effectiveness.

As a basis for our evaluation, we applied all three approaches to a sample problem,
wireless hotspots. The three approaches were evaluated for learnability, usability,
solution inclusiveness, clarity of output, and analyzability. Our findings highlighted that,
even though these three techniques were capable of specifying security requirements of
the wireless hotspot, more confidence can be achieved when they are combined. For
example, Common Criteria are hard to learn and use, but they produce outputs that are
easy to read, analyze, and identify solutions. Misuse Cases are simple to learn and use,
easy to analyze and identify solutions, but they produce diagrams that can be difficult to
read. Finally, Attack Trees are simple to learn and use, produce clear output, but their
solutions are not represented and the results are difficult to analyze.

We selected these techniques for a number of reasons. One, the techniques selected
covers a wide spectrum of qualities, ranging from a very methodical technique (Common
Criteria) to a more informal one (Attack Trees). Also, this range covers various output
formats, from text to graphics. Two, the techniques have different intended uses, for
instance, Common Criteria intended for use on larger projects, than Misuse Cases.

Our paper is organized as follows. Prior research on Security and Requirements is
reviewed in Section 2. The three techniques we have evaluated are described in Section 3.
In Section 4, we present our example problem that is wireless hotspots. In Section 5, the
results and discussion of the study are presented and the comparison of the techniques
including the criteria is described. In Section 6, the future work is introduced. Finally, we
conclude with a summary of the study in Section 7.

2 Security and Requirements

Security issues have risen in recent years due to the increased connectivity, extensibility,
and complexity of systems [12]. Networks provide additional attack vectors, extensibility
reveals systems internals, and complexity is accompanied by unanticipated effects. In our
work, we use the definition of security developed by Redwine et al. from the Software
Process Subgroup, which includes the concepts of confidentiality, integrity, and
availability [17]. They wrote: “Software security has as its primary goals three aspects,
the preservation of the confidentiality, integrity, and availability of the information assets
and resources that the software creates, stores, processes, or transmits including the
executing programs themselves.”

In this sense, confidentiality preservation refers to the prevention of unauthorized
disclosure, integrity preservation is about preventing unauthorized alteration, and
availability preservation is about preventing unauthorized destruction or denial of access
or service. Furthermore, security is a system property and needs to be considered from the outset of system development [21]. Consequently, we are focusing on security in the requirements phase.

A number of studies have focused on the question of addressing security concerns at the requirements phase. In their paper, Ware et al. presented an approach to eliciting security requirements for IT systems with use cases using Common Criteria methodologies [25]. Other work that has investigated the applicability of the Common Criteria is presented by Shi and Sun [19]. However, both of these efforts are limited only to investigation of the use of the Common Criteria in providing security confidence for software systems.

Hope et al. analyzed Misuse Cases and Abuse Cases in the perspective of putting them to work [22]. They emphasized the need of addressing security throughout the lifecycle starting from the requirements engineering, where Misuse and Abuse Cases can be used to specify security requirements. There have been many studies that have employed Misuse and Abuse Cases for specifying security requirements [20] [2] [10] [26] [27] [28]. Our work differs from these studies in that we are not using a single method; rather we are using a number of them, along with their comparison.

To our knowledge, there have not been any other comparative evaluations of techniques or approaches to specifying security requirements. The choice of these methods is based not only on their popularities, but also on their efficiency in specifying security requirements.

### 3 Three Approaches to Specifying Security Requirements

In this section, we describe three approaches in specifying security requirements. These approaches are Common Criteria, Misuse Cases, and Attack Trees. We present them here before applying them to our sample problem of wireless hotspots.

#### 3.1 Common Criteria

The Common Criteria (CC) is a comprehensive, standardized method for security requirements elicitation, specification, and analysis originating from the combination of U.S., Canadian, and European criteria [29]. It represents efforts to develop criteria for the evaluation of Information Technology (IT) security that is used internationally. It is intended to be a repeatable method for documenting IT security requirements and has been adopted by seven governmental organizations worldwide, including the United States. The CC can be used not only by software developers, but also by evaluators and consumers in their respective tasks.

The Common Criteria includes a method for specifying Security Functional Requirements. In this method, a Target of Evaluation (TOE) is examined to identify the
security environment and security objectives. These in turn are then analyzed to produce
the security requirements. The security environment defines the nature and scope of the
security problem to be addressed by the system. It is composed of usage assumptions,
threats, and relevant organizational security policies. The security objectives are based on
the security environments and establish the basis for the selection of security
requirements for a particular system. The security objectives are intended to be a concise
statement of the intended response, or solution, pertaining to the security problem.
Finally, the security requirements are selected so that they meet the security objectives.

As a result of this three-step process, the CC produces two kinds of documents, a
Protection Profile (PP) and a Security Target (ST). The PP is a document used to identify
the desired security properties of a product created by a group of users while the ST
identifies what a product actually accomplishes that is security-relevant.

3.2 Misuse Cases

Misuse Cases is a requirements specification method that is based on use cases [20] [2]
[27]. Use cases are helpful in describing functional requirements of a system, but they are
less helpful with non-functional requirements, such as security. Misuse Cases were
introduced to address this shortcoming of use cases [20]. In contrast to use cases, a
Misuse Case describes the malicious behavior of an unwanted user. Misuse Cases can
then be used to identify potential threats and elicit security requirements in the same
manner that use cases elicit requirements. Abuse Cases is another method based on use
cases [10]. Both of these methods are promising and complement existing analysis,
design, and verification practices.

Both Misuse and Abuse Cases are described by use case diagrams and descriptions.
The actors in a Misuse or Abuse Case model are the same kinds of external agents that
participate in use cases. Misuse Cases are created through the following steps [20].

1. Describe actors and use cases in the normal way suggested by UML methods.
2. Introduce the major mis-actors and Misuse Cases.
3. Investigate the potential “includes” relations between Misuse Cases and use cases.
4. Introduce new use cases for detecting or preventing Misuse Cases.
5. Continue with a more detailed requirements documentation.

Both Misuse Cases and Abuse Cases have been investigated by a number of
researchers. Pauli and Xu have applied Misuse Cases in the design of software
architecture to identify and analyze components and connectors for possible security
concerns [26]. Herrmann and Paech took this approach further and applied Misuse Cases
to model quality attributes, such as usability and efficiency, in developing a Misuse-based
method for deriving detailed nonfunctional and functional requirements [28].
3.3 Attack Trees

Attack Trees are a systematic method used to characterize system security based on varying attacks [30]. This technique models the attacker’s decision process by organizing attacks into a tree, with the attacker’s goal as the root node and the different ways of accomplishing that goal as leaf nodes.

Attack Trees are composed of AND nodes and OR nodes. OR nodes represent alternative choices while AND nodes represent different steps toward achieving the same goal. In addition to give more semantics to the trees, nodes are assigned labels (for example possible or impossible), which can be used to carry out calculations about the nodes to derive the overall security goals [5]. The value of an OR node is possible only if at least one value its children is possible. If all of its children’s values are impossible, then the value of the OR node is impossible. The value of an AND node is possible only if the values of all its children are possible, otherwise it is impossible.

Attack Trees are similar to Fault Trees, which traditionally have been used for the analysis of system safety properties [5]. Attack Trees are mainly used in requirements elicitation while Fault Trees are used in requirements analysis. Furthermore, Attack Trees are used in the development of intrusion scenarios, while Fault Trees are used to model intrusions and help identify requirements for intrusion detection systems. Creating an attack tree to make security decisions requires the following steps.

1. Identify the possible attack goals. Each goal forms a separate tree, although they may share sub-trees and nodes.
2. Identify all attacks against each goal and add them to the tree. Repeat this process down the nodes and branches.

Attack Trees have been used in varying situations to specify security requirements. Moore et al. proposed using Attack Trees to model security requirements for a specific domain of survivable systems [31]. In a different approach, Helmer et al. [5] proposed a mechanism for using software Fault Trees to analyze requirements of an Intrusion Detection System [9]. An approach for analyzing and specifying security goals and anti-goals using systematic Attack Trees generation has also been developed [32].

3.4 Commensurability of the techniques

The three techniques selected have similarities and differences in the ways they address security requirements. Their differences allow us to make meaningful comparisons and their similarities allow us to apply them to same requirements problem. Each of the three techniques incorporates requirements elicitation, analysis, and specification. Each technique begins by eliciting the threats to the software system to be implemented. In the Common Criteria technique, the Security Environment is addressed at the beginning of the process, which includes the usage assumptions, threats, and relevant organizational security policies are identified. With the Misuse Cases technique, mis-actors and misuse cases based on the actors and use cases are defined early in the process. In the case of the
Attack Trees technique, the goals of the attacker are identified at the starting point of the process. For each method, a step involving an analysis is necessary to guarantee that all identified threats are addressed or formally described. Furthermore, each method provides a notation for specifying the security threats and the output produced by each method is described using the provided notation.

The main difference between the three methods is the extent to which they address security requirements. The Common Criteria, in addition to specifying security functional requirements, is supplemented with security assurance requirements. The requirements assurance process is the basis for gaining confidence to ensure that the claimed security measures are effective and implemented correctly. In this regard, Misuse Cases and Attack Trees are limited to the elicitation, analysis, and specification of security requirements. Attack Trees differ from the other two in that the trees that represent the attacks are formally defined and facilitate automation.

Even though it is difficult to isolate this process from the others in this study, the main focus continues to be directed toward the security requirements specification process. We are comparing and analyzing the methods based on the process of producing the security requirements specification document. Amongst the questions we will address include the following: What is the level of difficulty in developing the security requirements specification beginning with learning the technique, applying the technique, and producing a final output? In addition, we will investigate the quality of the output given by each technique.

4 Comparative Evaluation Using Wireless Hotspots

In this section, we will describe the comparative evaluation that we performed on the three approaches. We applied each method to a common problem, wireless hotspots. Wireless hotspots, described further in Section 4.1, are ad hoc wireless connectivity providers in public spaces such as airports and libraries. We selected wireless hotspots because they have a number of well-understood security vulnerabilities, which we present in Section 4.2.

The study was undertaken primarily by the two first authors, who are Master’s of Science students. They studied the requirements specification techniques and applied them to the demonstration problem. Since they did not have extensive background nor training in the area, we feel that their experiences are comparable with average software engineers.

4.1 Wireless Hotspots

Wireless hotspots are extensions of network connectivity to public places for wireless computers such as laptops and personal digital assistants. In recent years, wireless
hotspots have become widely used due to significant advancements in networking technologies, in particular, wireless local area networks [33]. Wireless hotspots are becoming available in airports, shopping malls, hotels, libraries, and other public avenues. However, as the number of users increase, the risk of possible misbehavior grows with it [33]. This reality leads to more design challenges of hotspot systems which include security, coverage, management, location services, billing, and interoperability. The security challenge is recognized as being the most significant obstacle to the proliferation of open wireless networks [18]. In this paper, we focus only on the security aspect of wireless hotspots, in particular, the security requirements elicitation.

Current wireless hotspot technology is based on IEEE 802.11, the Wireless Fidelity (WI-FI) standard, which denotes a set of Wireless Local Area Network (WLAN) standards. The MAC-layer of the IEEE 802.11 has a greedy behavior that, if not used properly, might lead to security risks.

We consider a simplified model of a wireless hotspot, which includes an access point, a good user [a legitimate user] and a bad user [an attacker]. The system model is illustrated in Figure 1. The functional requirements are limited to three user activities in the system. The first activity is authentication of the user identity at the access point. Following the authentication there is surfing of the internet. The last activity is the power saving mode. Here the user is still connected to the access point but is not active and it is assumed that the wireless hotspot uses WI-FI technology.

![Fig. 1. System Model](image)

4.2 Security Threats

There are a number of vulnerabilities associated with the 802.11 wireless LAN, and these are well-understood and well-documented [32] [1] [16] [23]. There two mainly types of vulnerabilities in the 802.11 wireless LAN; identity vulnerabilities and media access Carrier (MAC) vulnerabilities. These vulnerabilities lead to denial of service (DoS)
attacks, which prevent the transmission of data to and from access points. The identity vulnerabilities arise from the implicit trust wireless LAN networks place in a user’s source address. Wireless LAN ‘Class 1’ messages, which include most management and control messages, do not incorporate a mechanism for verifying the correctness of the self-reported identity. Thus, an attacker can spoof, or pretend to be, other nodes, and request various MAC-layer services on their behalf. The identity vulnerabilities cause three kinds of attacks: De-authentication, Disassociation, and Power Saving Mode attacks. The MAC vulnerabilities arise due to the collision avoidance mechanism of the wireless LAN MAC layer. The two types of possible attacks are Time Window and Virtual Carrier Sense attacks. Finally, there is the Evil Twin attack which works around encryption technologies.

There are a total of six different attacks, but due to space limitation, only the Evil Twin attack will be described in detail.

4.2.1 Evil Twin Attack
This is the latest form of attack in the wireless LAN. The Evil Twin attack can be carried out through a laptop equipped with special software that broadcasts a radio signal that overpowers hotspots. Then, masquerading as the real hotspot, they view the activities of wireless users within several hundred feet of the hotspot.

One solution is to use encryption in every step, from the start of the login process to the service in wireless hotspot, as this encryption helps avoid the Evil Twin attack. Another solution is to use an identity authentication system that requires mutual identity authentication.

5 Results and Discussion

In this section, we present the results of applying the three approaches to our test problem. After studying the techniques ourselves, we applied them to wireless hotspots. In each subsection, we will describe the process, the outputs, and our evaluation from the CC, Abuse Cases, and Attack Trees.

5.1 Common Criteria

The design of the Common Criteria for the wireless hotspots is accomplished through security environment definition, security objectives identification, security requirements elicitation, and the mappings between these elements. Since there are no particular organizational security policies and assumptions about the wireless hotspot system, the security environment is limited in defining threats to the system. Once the threats are known, the next step is to identify the security objectives that can address those threats.
In Table 1 the first column lists the threats while the second column shows the security objectives.

Table 1. Threats and Security objectives.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Security objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1: Deauthentication/ disassociation attacks</td>
<td>O1: Delay effect de-authentication messages</td>
</tr>
<tr>
<td>T2: Power saving mode attack</td>
<td>O2: Control time window for SIFS</td>
</tr>
<tr>
<td>T3: Time window attack</td>
<td>O3: Control duration of received frames</td>
</tr>
<tr>
<td>T4: Virtual carrier sense attack</td>
<td>O4: Data encryption</td>
</tr>
<tr>
<td>T5: Evil twin attack</td>
<td>O5: Identity authentication system</td>
</tr>
</tbody>
</table>

Table 2. Security requirements drawn from the CC components.

<table>
<thead>
<tr>
<th>Security Functional Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>0. Communication:</strong></td>
</tr>
<tr>
<td>R01: Enforced proof of origin [FCO_NRO.2]</td>
</tr>
<tr>
<td>R02: Enforced proof of receipt [FCO_NRR.2]</td>
</tr>
<tr>
<td><strong>1. Cryptographic support:</strong></td>
</tr>
<tr>
<td>R10: Cryptographic key generation [FCS_CKM.1]</td>
</tr>
<tr>
<td>R11: Cryptographic operation [FCS_COP.1]</td>
</tr>
<tr>
<td><strong>2. Identification and authentication:</strong></td>
</tr>
<tr>
<td>R21: User authentication before any action [FIA_UAU.2]</td>
</tr>
<tr>
<td>R22: Unforgeable authentication [FIA_UAU.3]</td>
</tr>
<tr>
<td>R23: User identification before any action [FIA_UID.2]</td>
</tr>
<tr>
<td><strong>3. User data protection:</strong></td>
</tr>
<tr>
<td>R31: Complete access control [FDP_ACC.2]</td>
</tr>
<tr>
<td>R32: Security attribute based access control [FDP_ACF.1]</td>
</tr>
<tr>
<td>R33: Integrity monitoring [FDP_ITT.3]</td>
</tr>
<tr>
<td><strong>4. Trusted path/channels:</strong></td>
</tr>
<tr>
<td>R41: Inter-TSF trusted channel [FTP_ITC.1]</td>
</tr>
<tr>
<td>R42: Trusted path [FTP_TRP.1]</td>
</tr>
</tbody>
</table>

Table 3. Mapping between threats, security objectives, and security requirements.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Security objectives</th>
<th>Security requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>O1, O4</td>
<td>R01, R02, R22</td>
</tr>
<tr>
<td>T2</td>
<td>O4</td>
<td>R01, R02, R11</td>
</tr>
<tr>
<td>T3</td>
<td>O2, O4</td>
<td>R01, R02, R32, R33</td>
</tr>
<tr>
<td>T4</td>
<td>O3, O4</td>
<td>R01, R02, R31, R33</td>
</tr>
<tr>
<td>T5</td>
<td>O4, O5</td>
<td>R41, R42, R31, R32</td>
</tr>
</tbody>
</table>
A mapping between threats and security objectives identifies threats that are addressed by at least one security objective. This mapping is illustrated by the first two columns of Table 3. In this table, we can see that each threat is addressed by at least one security objective. Security Functional Requirement components are selected from the CC method to specify security requirements that meet the security objectives of the wireless hotspot. These components are presented in Table 2. Finally, columns one and three of Table 3 show the final step of mapping the threats to the security requirements.

The document produced by the CC is organized into tables, which makes it simple to read and interpret. The Common Criteria can express requirements completely in the sense that they can specify any threat and incorporate the solution to the threat if available. For example, in our case study, the Evil Twin attack is mapped to its solution, encryption. This characteristic produces clear output, and also permits easy analysis of the security requirements. In particular, other parts of the CC provide guidance in performing various analyses, such as Security Confidence.

One drawback with the CC is that the learning curve associated with it is high due to its complexity. In addition, there are many steps required to complete the document, which add complexity and compromise to its usability. This complexity was evident in our process of selecting the security requirements to address the security objectives for the wireless hotspot. Another problem with the CC is that its security requirements can be too general. As a result, these security requirements might not always satisfy the needs of system designers since systems designers aim at solving specific security needs.

5.2 Misuse Cases

To designing a use and Misuse Cases diagram, the identification of use cases, the definition of the Misuse Cases, and the identification of the mitigations against the threats are necessary. Figure 2 shows the complete diagram of use and Misuse Cases. The boxes to the right of the diagram represent the use cases. While, the boxes on the left illustrate the Misuse Cases. The solid arrows show the direction of the “includes”, the dashed arrows show the direction of the attacks, and the dotted arrows show the direction of the mitigations. Notice that data encryption mitigation addresses both Evil Twin and saving Mode attacks.

Like the Common Criteria, Misuse Case diagrams incorporate the mappings between the use cases, Misuse Cases, and mitigation against the threats. The graphical representation of these relationships facilitates the analysis of the security requirements. In addition, any known threat and mitigation can be described by the diagrams, which make the technique complete. Other advantages of Misuse Cases are learnability and usability. Since they are based on use cases, they are easy to learn. In this case study, after identifying the threats and their mitigations, it was simple draw the diagram.

Although combining threats and solutions in the diagram is useful in the analysis process, this can be a drawback when it comes to the clarity of the output, particularly
with large systems. Even though our case study is a simple system, the use and misuse diagram contains crossing edges that affect readability. Additionally, Misuse Cases only capture high-level security issues. For example, the diagram shows the Disassociation attack but fails to give details on how this attack can be achieved.

5.3 Attack Trees

An Attack Tree is represented by a tree structure where the root becomes the goal of the attacker. The structure and semantics of this are as follows. A node of the tree is decomposed as a set of attack sub-goals. In an AND-decomposition, all the sub-goals must be achieved in order for the attack to succeed, while in OR-decomposition any one of the sub-goals can be achieved before the attack succeeds. Figure 2 shows a partial Attack Tree for the wireless hotpot where the root represents a successful attack of the access point. From here, we can construct all scenarios that lead to a successful attack of the access point and harm the user.

Similar to Misuse Cases, Attack Trees are simple to learn and use since they are based on a known requirements technique, scenarios. Each scenario is a path in the tree and shows step by step how an attack goal can be achieved. The tree gives a broad view of all possible ways in which to achieve an attack goal. Another advantage of Attack Trees is the clarity of the output. The diagram does not only display information hierarchically, but it can also allow designers to add detailed information to it.
While details provide more information, they can increase the cost of both the creation and the maintenance of systems, particularly those that are large. Another drawback of this method is that it does not express the relationship between the attacks and their solutions, thus leaving designers with more work to do. Unfortunately, Attack Trees are difficult to analyze. For example, the Attack Tree for the wireless hotpot shows an Evil Twin attack, but it is not possible to analyze the different solutions against this attack using the tree. Another problem with Attack Trees is that they capture attacks that are sequences of events. However, they may not be appropriate for attacks that involve concurrent actions.

![Access Point Attack Tree](image)

**Fig. 3. Access Point Attack Tree.**

### 6 Comparison of Techniques

#### 6.1 Criteria for Evaluation

In this section we describe the five main criteria used to evaluate the three selected security requirement specification approaches. Before going further, it is important to note that the comparison criteria, as well as the metrics used, are defined in a post hoc manner. Due to the lack of established standard criteria and metrics for comparing security requirements methods, we defined these criteria based on our experience and analysis working with the three methods. The evaluation of the techniques with the different criteria is based on the amount of time spent on each technique, the effort
required to use the techniques, and lastly the analysis of the output produced by each method. We believe this is a good starting point for establishing standard criteria and metrics for comparing security requirements methods.

**Learnability.** How long would it take designers to learn and use the techniques? How many new concepts or readings are required to apply these methods? For the three methods we selected, we evaluated them in terms of how long it took us to learn how to use them.

**Usability.** The usability principle measures how simple or complex a technique can be used. In the context of our evaluation, we are looking at how much work was involved in applying the technique, gathering information, and producing the output. Once the information is collected, how usable is the process of designing a diagram, tree, or table.

**Solution Inclusiveness.** While it is difficult to know whether a particular set of requirements is complete, it is reasonable to ask whether a format for specifying requirements is complete. In other words, are there certain types or classes of threats or information that are not easily captured by the format or technique? Can these techniques specify any threat and its solution? Can the techniques we selected incorporate the threats and their solutions?

**Clarity of Output.** This principle is concerned with the ease associated with reading and using the outputs from the specification technique. We were also interested in the kinds of details about the attacks and remediations that represented. It would be more helpful if the techniques provide clear solutions to attacks that are understandable and usable by software designers. We include examples of output in this paper to help illustrate this property.

**Analyzezability.** The analyzability measures how easily a designer can interpret and analyze the results provided by the techniques. For example, can we check all possible combinations of known threats that lead to a successful attack to a system? Can you choose among alternatives in addressing the mitigations to the threats? Is it possible to verify the security assurance provided by the technique?

### 6.2 Comparison of the Techniques

Each of the three methods has advantages and disadvantages, but they can be combined to increase confidence in a security specification. Table 4 summarizes the comparison between the three methods based on our evaluation criteria presented earlier: learnability, usability, clarity of output, solution inclusiveness, and analyzability.

Attack Trees and Misuse Cases provide complementary information. In the one hand, Attack Trees provide a systematic way to symbolize attacks. On the other hand, a misuse case that has additional information such as preconditions may be more accessible and useful to a developer. In our case study, a precondition for an attacker to use the Power Saving Mode attack is finding a user in a Power Saving Mode. This information is available in use and misuse diagrams, but not in the Attack Trees. The combination of Attack Trees and Misuse Cases gives more information to designers, but cannot provide
other advantages found in the Common Criteria. The Common Criteria gives guidance on downstream activities to provide assurance that a system meets its security objectives. No comparable guidance is available with Misuse Cases and Attack Trees. In addition, the Common Criteria is more suitable for evaluators and customers to use.

Table 4. Comparison of the three approaches.

<table>
<thead>
<tr>
<th></th>
<th>Common Criteria</th>
<th>Misuse Cases</th>
<th>Attack Trees</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Learnability</strong></td>
<td>Difficult to learn –</td>
<td>Simple to learn –</td>
<td>Simple to learn – based</td>
</tr>
<tr>
<td></td>
<td>large documents</td>
<td>based on use cases</td>
<td>based on scenarios</td>
</tr>
<tr>
<td><strong>Usability</strong></td>
<td>Difficult to use</td>
<td>Simple to use – based</td>
<td>Simple to use – based</td>
</tr>
<tr>
<td></td>
<td></td>
<td>on use cases</td>
<td>on scenarios</td>
</tr>
<tr>
<td><strong>Solution</strong></td>
<td>Solution included</td>
<td>Solution included</td>
<td>Solution not included</td>
</tr>
<tr>
<td><strong>Inclusiveness</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Clarity of</strong></td>
<td>Clear output – uses</td>
<td>May be difficult to</td>
<td>Clear output – uses a</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>tables</td>
<td>read for large systems</td>
<td>tree</td>
</tr>
<tr>
<td><strong>Analyzability</strong></td>
<td>Easy to analyze -</td>
<td>Easy to analyze</td>
<td>Difficult to analyze</td>
</tr>
</tbody>
</table>

One aspect common to all three methods is the possibility of reuse. Security requirements specification for the wireless hotspot is described as a set of tables by the Common Criteria, as a tree by the Attack Trees, and as a diagram by the Misuse Cases all of which can be reused in similar applications. Any of these documents can be extended to address new threats in the wireless network. It is important to note that mitigations, whether in the CC or Misuse Cases, rarely counteract all security threats. For example, an attacker can compromise the access point by other means than the ones we have described here. However, partial mitigations, such as the ones found in the CC tables and the use and misuse cases diagram, are still useful as long as they decrease the vulnerability of a system at a reasonable cost.

**6. Future Work**

One of the difficulties we encountered in this study was the lack of standard criteria for comparing security requirements methods. In the future, we plan to elaborate on the criteria for comparing security requirements methods. Furthermore, we will focus on defining quantitative metrics that provide a more precise comparison. This in turn will allow us to reach a better cost-benefit analysis of the techniques.

It is not surprising that each of the techniques has shortcomings, so the next obvious question is how to remedy this, in particular, by combining the techniques. Thus, as part of our future work we propose the development of a framework that would assist in combining multiple security requirements methods appropriately. This framework would also provide guidance on when to use a particular technique or representation. In this
study we only considered specification, so we need to broaden our examination to include
elicitation, analysis, and traceability.

Analyzing and comparing security requirements techniques by applying them to a
specific case is an important initial step in evaluating current security requirements
methods. Specific details about the similarities and differences, and the advantages and
disadvantages can be obtained through this approach. A complementary approach would
be to perform an analytic study of security requirements techniques based on creating
meta-models. Such a study would provide insight into the concepts or threats that it is
possible to represent using the various methods.

7 Conclusion

The main focus of this paper was to critically analyze and compare three popular
techniques for specifying security requirements, the Common Criteria, Misuse Cases, and
Attack Trees. The analysis and comparison were performed by applying the techniques to
an example problem (wireless hotspots) and were based on five criteria, learnability,
usability, solution inclusiveness, clarity of output, and analyzability. We found that each
of these techniques worked well but had different strengths and weaknesses. While the
Common Criteria is comprehensive and thorough, it is not easy to learn and use.
Furthermore, it does not provide a way to specify detailed information such as
preconditions in security objectives that might prove useful for the designers. In contrast,
Misuse Cases can embed detailed information about the mitigations of the threats. Misuse
Cases have two main drawbacks as illustrated by our study. They represent only general
information about the threats and the produced diagram can be difficult to read and
understand. While Attack Trees can represent highly detailed information about security
requirements, they do not include security objectives, which make them less useful in the
design process.

Our study exposes the benefits associated with combining techniques when specifying
security requirements. While it is not possible to guarantee the security of a system based
solely on the requirements, it is imperative for system designers to receive as much
information as possible from the requirements specification. By combining techniques,
more information can be obtained regarding the security of a system. This observation
directs the way of future research in specifying security requirements. Given that each of
the techniques that we studied is known and used, it would be beneficial to combine their
strengths to mitigate their weaknesses.

Security affects us today more than ever. Software security problems have become
frequent, widespread, and serious. Security challenges are increasing due to the modern
technologies that raise connectivity, extensibility and complexity of systems [10].
Requirements technologies for security will need to keep pace with not just system
builders, but ever more innovative attackers as well.
References

Compared the three skin segmentation approaches (RehgJones, HybridClustering, NeuralGasColorClustering. Method of evaluation:
Ground truth dataset of about 500 images. Approaches to Comparative Analysis - bruce walton all reports in mast log in as mastcomparative. why compare performance?. importance of context identifying potential for improvement citizens are encouraged to.
Intrusion Detection -. outline. intrusion detection and computer security current intrusion detection approaches data mining approaches for intrusion detection summary. intrusion detection and computer security . computer security goals FE-VER thus strives for a fair comparative evaluation of different approaches by considering both the effectiveness and configura- tion effort. 1. INTRODUCTION. Entity resolution (also referred to as object or entity matching, duplicate identification, record linkage or reference reconciliation) is a fundamental problem for data integration and data cleaning [11]. The operator tree concept provides a high flexibility to specify a tailored workflow for a given match task and supports its com-parative evaluation with other approaches. In particular, it allows the selection and combination of several match approaches. Developed operator trees can be saved as building blocks and can then be reused as a sub-tree in other workflow definitions. A comparative overview of resilience measurement frameworks, analysing indicators and approaches. E. Lisa F. Schipper & Lara Langston. July 2015. PPCR Pilot Programme for Climate Resilience. SHARP UN Food and Agriculture Self-evaluation and Holistic Assessment of Climate Resilience. TAMD Tracking Adaptation and Monitoring Development. UN United Nations.