JOURNAL OF OBJECT TECHNOLOGY

Online at www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2003**

Vol. 2, No. 3, May-June 2003

Security Use Cases

Donald G. Firesmith, Software Engineering Institute, U.S.A.

Abstract

Although use cases are a popular modeling approach for engineering functional requirements, they are often misused when it comes to engineering security requirements because requirements engineers unnecessarily specify security architectural mechanisms instead of security requirements. After discussing the relationships between misuse cases, security use cases, and security mechanisms, this column provides examples and guidelines for properly specifying essential (i.e., requirements-level) security use cases.

1 INTRODUCTION

Over the past decade, use cases have become one of the most popular modeling approaches for analyzing and specifying functional requirements. However, use case modeling has not been as successfully applied to engineering quality requirements, such as operational availability, performance, portability, reliability, reuse, security, and usability. When it comes to engineering security requirements, use cases are typically misused to unnecessarily specify security architectural mechanisms (e.g., the use of user identifiers and passwords) rather than actual security requirements (e.g., mandating some level of identification and authentication). Thus, typical example use cases for an automatic teller machine might include initial interactions for inserting an ATM card to identify the customer and entering a PIN number for authentication (i.e., verifying the identity of the customer). Whereas this is the current standard security mechanism for implementing identification and authentication requirements for ATM machines, it unnecessarily prevents the use of other, perhaps improved means of access control such as biometrics (e.g., face recognition, fingerprint analysis, or retinal scan).

Security requirements should be based on an analysis of the assets and services to be protected and the security threats from which these assets and services should be protected. Thus, as illustrated in Figure 1, there are clear relationships between assets and services, which are vulnerable to security threats, which necessitate security requirements, which require security mechanisms that counter these security threats and thereby protect the assets and services.

Cite this column as follows: Donald Firesmith: “Security Use Cases”, in *Journal of Object Technology*, vol. 2, no. 3, May-June 2003, pp. 53-64. http://www.jot.fm/issues/issue_2003_05/column6

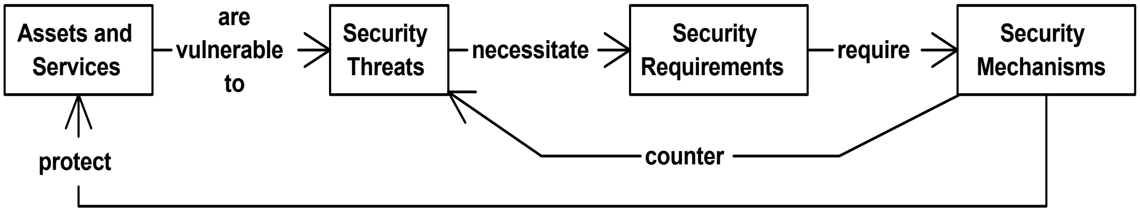

Fig. 1: Security Threats, Requirements, and Mechanisms

Historically, the emphasis of security engineering has been on the development and use of numerous security mechanisms to protect vulnerable assets and services by countering known security threats. The analysis and documentation of security threats and security requirements has received considerably less attention.

Misuse Cases for the Analysis of Security Threats

A relatively recent approach to addressing security threat analysis has been the development of misuse cases. As illustrated in Figure 2, misuse cases (a.k.a., abuse cases) are a specialized kind of use cases that are used to analyze and specify security threats [Sindre and Opdahl 2001] [Alexander2003]. Unlike normal use cases that document interactions between an application and its users, misuse cases concentrate on interactions between the application and its misusers (e.g., cracker or disgruntled employee) who seek to violate its security. Because the success criteria for a misuse case is a successful attack against an application, misuse cases are highly effective ways of analyzing security threats but are inappropriate for the analysis and specification of security requirements. Instead, security use cases should be used to specify requirements that the application shall successfully protect itself from its relevant security threats.

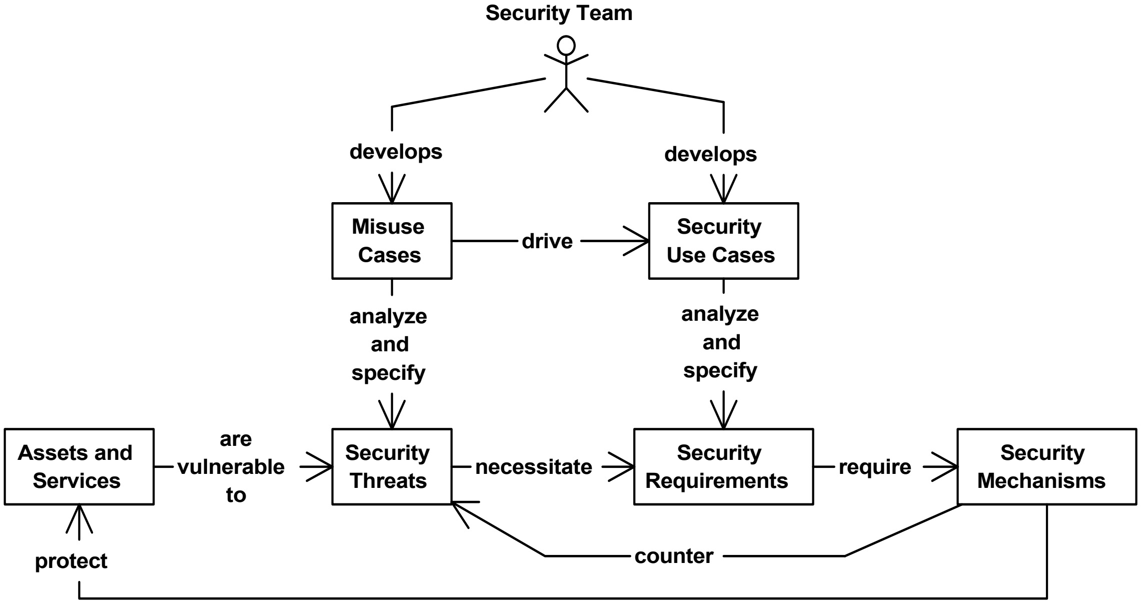

Fig. 2: Misuse Cases vs. Security Use Cases

The following table summarizes the primary differences between misuse cases and security use cases.

	Misuse Cases	Security Use Cases
Usage	Analyze and specify security threats.	Analyze and specify security requirements
Success Criteria	Misuser Succeeds	Application Succeeds
Produced By	Security Team	Security Team
Used By	Security Team	Requirements Team
External Actors	Misuser, User	User
Driven By	Asset Vulnerability Analysis Threat Analysis	Misuse Cases

To further illustrate the differences between normal use cases, security use cases, and associated misuse cases, consider Figure 3. The traditional use cases for an automated teller machine might include Deposit Funds, Withdraw Funds, Transfer Funds, and Query Balance, all of which are specializations of a general Manage Accounts use case. To securely manage one's accounts, one can specify security use cases to control access (identification, authentication, and authorization), ensure privacy (of data and communications), ensure integrity (of data and communications), and ensure nonrepudiation of transactions. The resulting four security use cases specify requirements that protect the ATM and its users from three security threats involving attacks by either crackers or thieves.

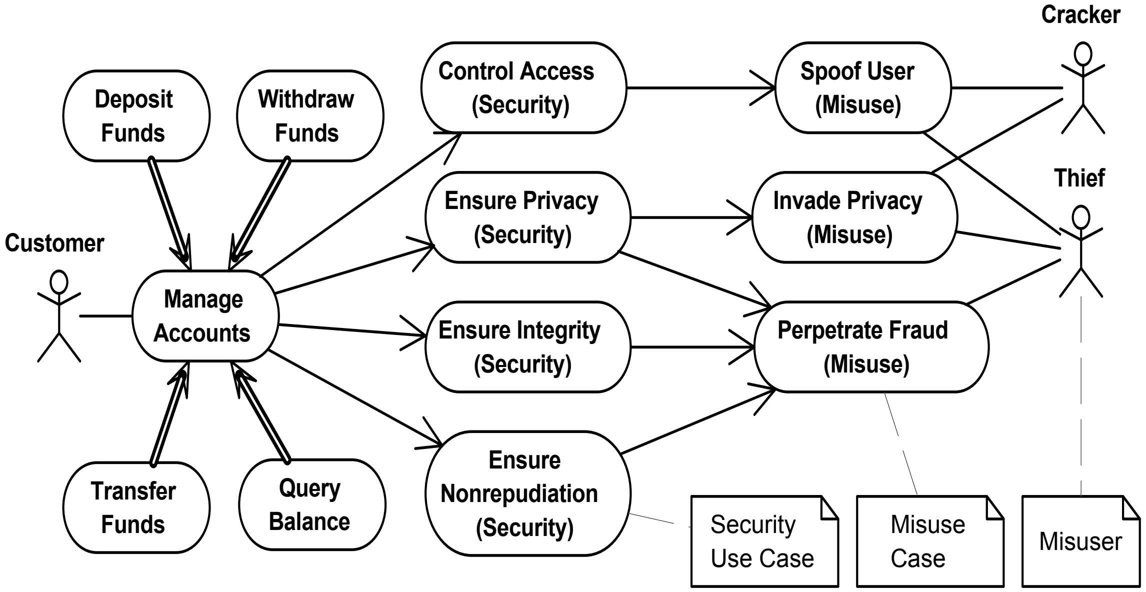

Fig. 3: Example Security Use Cases and Misuse Cases

2 EXAMPLE SECURITY USE CASES

As documented in [Firesmith 2003], there are numerous kinds of security requirements. Although each kind of security requirement typically has its own security use case, given the limited space available in this column, I have selected access control (identification and authentication), integrity, and privacy to illustrate the proper use of security use cases. To maximize the reusability of the following use case path specifications, I have also kept them at the highest, most-generic level of abstraction (i.e., as paths through essential use cases). When reused on real projects, each path specification can easily be made more specific to the application being specified without devolving into an architecture or design level specification, often merely by replacing the general words "system" and "user" with the specific application name and the specific type of user.

Access Control Use Case

Access control is the extent to which a a business enterprise, application, component, or center controls access by its externals (e.g., human users and applications). Access control consists of identification, authentication (i.e., verification of identification), and authorization. The following three tables document example use case paths through a highly-reusable essential security use case that specifies access control requirements:

- Use Case Path - Attempted Spoofing using Valid User Identity
- Use Case Path - Attempted Identity and Authentication Theft
- Use Case Path - Attempted Spoofing using Social Engineering

Use Case: Access Control		
Use Case Path: Attempted Spoofing using Valid User Identity		
Security Threat: The system authenticates and authorizes the misuser as if the misuser were a valid user.		
Preconditions: 1) The misuser has a **valid** means of user identification. 2) The misuser has an **invalid** means of user authentication.		
Misuser Interactions	**System Requirements**	
	System Interactions	**System Actions**
	The system shall request the misuser's means of identification and authentication.	
The misuser provides a valid means of user identity but an invalid means of user authentication.		

		1) The system shall misidentify the misuser as a valid user. 2) The system shall **not** authenticate and authorize the misuser.
	The system shall reject the misuser by canceling the transaction.	

Postconditions:
1) The system shall not have allowed the misuser to steal the user's means of authentication.
2) The system shall not have authenticated the misuser as a valid user.
3) The system shall not have authorized the misuser to perform any transaction that requires authentication.
4) The system shall have recorded the access control failure.

Use Case: Access Control

Use Case Path: Attempted Identity and Authentication Theft

Security Threat: The misuser steals the user's means of identification and authentication, thereby allowing the misuser to impersonate a valid user.

Preconditions:
1) The misuser has no valid means of user identification.
2) The misuser has no valid means or user authentication.

User Interactions	**Misuser Interactions**	**System Requirements**	
		System Interactions	**System Actions**
		The system shall request the user's identity and authentication.	
The user identifies and authenticates himself or herself.	The misuser attempts to steal the user's means to identify and authenticate.		The system shall protect the user's identity and authentication during the interaction.
			The system shall identify and authenticate the user.
		The system shall request the user's choice of interaction.	

Postconditions:
1) The system shall have prevented the misuser from stealing the user's means of identification and authentication.
2) The system shall have identified and authenticated the user.

Use Case: Access Control

Use Case Path: Attempted Spoofing using Social Engineering

Security Threat: The misuser gains access to an unauthorized resource.

Preconditions:
1) The misuser has a valid means of user identification enabling the impersonation of a valid user that is authorized to use a protected resource.
2) The misuser does not have an associated valid means of user authentication.
3) The misuser has basic knowledge of the organization including the ability to contact the contact center.

Misuser Interactions	**Contact Center Requirements**	
	Contact Center Interactions	**Contact Center Actions**
The misuser contacts the contact center.		
	A user support agent shall request the misuser's identity and authentication.	
1) The misuser provides the valid user identity. 2) The misuser states that he or she has a temporary inability to authenticate himself or herself. 3) The misuser states that he or she has an urgent need to access a resource requiring authentication and authorization.		
	The user support agent shall request one or more alternate forms of authentication.	The user support agent shall check the appropriate procedures for the proper action.
The misuser fails to provide a valid alternate form of authentication.		
	The user support agent shall refuse authentication and authorization to the requested	

	resource.	
Alternative Paths: The misuser can quit at any point.		
Postconditions: 1) The system shall not have authenticated the misuser. 2) The system shall not have authorized the misuser to access the protected resource. 3) The system shall have recorded the access control failure.		

Integrity Use Case

Integrity is the extent to which a business enterprise, application, component, or center ensures that its data and communications are not intentionally corrupted via unauthorized creation, modification, or deletion. The following three tables document example use case paths through a highly-reusable essential security use case that specifies integrity requirements:

- Use Case Path - System Data Protected
- Use Case Path - System Data Corrupted
- Use Case Path - System Message Integrity
- Use Case Path - Use Message Integrity
- Use Case Pase - Denial Of Service (DOS) Attack

Use Case: Integrity		
Use Case Path: System Data Protected		
Security Threat: A misuser may corrupt (e.g., add, modify, delete) sensitive data that is stored by the system.		
Preconditions: The system stores sensitive data that must not be corrupted.		
Misuser Interactions	**System Requirements**	
	System Interactions	**System Actions**
The misuser attempts to corrupt (e.g., add, modify, delete) sensitive data stored by the system.		
		The system shall prevent the data from being corrupted.
	The system shall notify the security officer that an attempt to corrupt data occurred.	

Postconditions: The system shall ensure that no sensitive data has been corrupted.

Use Case: Integrity			
Use Case Path: System Message Integrity			
Security Threat: A misuser corrupts a message that is sent from the system to a user.			
Preconditions: 1) The misuser has the means to intercept a message from the system to a user. 2) The misuser has the means to modify an intercepted message. 3) The misuser has the means to forward the modified message to the user.			
User Interactions	**Misuser Interactions**	**System Requirements**	
		System Interactions	**System Actions**
		The system shall send a message to a user.	The system shall ensure that modifications to the message will be obvious to the user.
	The misuser intercepts and modifies the system's message and forwards it on to the user.		
The user receives the corrupted message.			The system shall recognize that its message was corrupted.
		The system shall notify the user that its message was corrupted.	
Postconditions: The system shall have notified the user that the system's message was corrupted.			

Use Case: Integrity		
Use Case Path: User Message Integrity		
Security Threat: A misuser corrupts a user's message to the system.		
Preconditions: The misuser has the means to intercept a message between the user and the system.		
User Interactions	**Misuser Interactions**	**System Requirements**

		System Interactions	**System Actions**
The user sends a message to the system.			
	The misuser intercepts, modifies, and forwards the user's message.		
			The system shall recognize that the user's message was corrupted.
		The system shall notify the user that the user's message was corrupted.	
Postconditions: The system shall have notified the user that the user's message was corrupted.			

Privacy Use Case

Privacy is the extent to which a business enterprise, application, component, or center keep its sensitive data and communications private from unauthorized individuals and programs. The following three tables document example use case paths through a highly-reusable essential security use case that specify privacy requirements:

- Use Case Path - Data Privacy
- Use Case Path - System Message Privacy
- Use Case Path - User Message Privacy

Use Case: Privacy		
Use Case Path: Data Privacy		
Security Threat: The misuser accesses private data that is stored by the system.		
Preconditions: The system stores private data.		
Misuser Interactions	**System Requirements**	
	System Interactions	**System Actions**
		The system shall make the private stored data unreadable.

The misuser accesses the private data that is stored by the system.		
Postconditions: The system shall have stored the private data in a form that is not readable by the misuser.		

Use Case: Privacy			
Use Case Path: System Message Privacy			
Security Threat: The misuser accesses a private message from the system to the user.			
Preconditions: The misuser has the means to intercept a message from the system to the user.			
User Interactions	**Misuser Interactions**	**System Requirements**	
		System Interactions	**System Actions**
			The system shall make the private message unreadable while in transit.
		The system shall send a private message to the user.	
	The misuser intercepts the system's private message.		
Postconditions: The system shall have sent the private message in a form that the misuser cannot read.			

Use Case: Privacy			
Use Case Path: User Message Privacy			
Security Threat: The misuser accesses a private message from the user to the system.			
Preconditions: 1) The misuser has the means to intercept a message from the user to the system. 2) The system has requested private information from the user.			
User Interactions	**Misuser Interactions**	**System Requirements**	
		System	**System Actions**

		Interactions	
The user sends a private message to the system.			
			The system shall make the private message unreadable while in transit.
	The misuser intercepts the user's private message.		
Postconditions: The system shall have ensured that the misuser cannot read the user's private message.			

3 SECURITY USE CASE GUIDELINES

The following guidelines are recommended when developing security use cases during requirements engineering:

- Use essential (i.e., requirements only) use cases that do not specify unnecessary security architectural mechanisms such as user IDs, passwords, digital signatures, biometrics, encryption, etc. Leave such decisions to the architecture and security teams who are better qualified than the requirements team to make such decisions.
- Carefully differentiate requirements (e.g., by using the word "shall") from ancellary information. System interactions, system actions, and the postconditions should be specified as requirements on the system, whereas preconditions, user interactions, and misuser interactions should not be.
- To avoid unnecessarily specifying design constraints, the clearly note if the sequencing of the interactions can occur in different orders.
- Explicitly document the individual paths through the security use cases in order to specify the actual security requirements.
- Base the security use cases on the different types of security requirements, which provide a natural organization to the use cases.
- Document the security threats that justify the individual paths through the security use case.
- Clearly distinquish between user and misuser interactions.
- Clearly distinguish between externally-visible system interactions and hidden system actions.
- Document both preconditions and postconditions, which capture the essence of the individual path.

4 CONCLUSION

Whereas misuse cases are an excellent means of analyzing security threats, they are inappropriate for analyzing and specifying security requirements because they are based on misusers successfully attacking the system. On the other hand, essential security use cases provide a highly-reusable way of organizing, analyzing, and specifying security requirements if they are kept at the appropriate level and if unnecessary architectural and design constraints are avoided.

REFERENCES

[Alexander2003] Ian Alexander: Misuse Case Help To Elicit Nonfunctional Requirements, IEE CCEJ, 2001, http://easyweb.easynet.co.uk/~iany/consultancy/papers.htm.

[Firesmith2003] Donald Firesmith: OPEN Process Framework (OPF) Website, www.donald-firesmith.com.

[Sindre and Opdahl 2001] Guttorm Sindre and Andreas Opdahl: Templates for Misuse Case Description, 2001, http://www.ifi.uib.no/conf/refsq2001/papers/p25.pdf.

ACKNOWLEDGEMENTS

The contents of this article are taken from my informational website on process engineering as well as the contents of my upcoming book on requirements engineering. Guttorm Sindre and Andreas Opdahl provided very valuable insights regarding the relationship between misuse cases and security use cases.

About the author

Donald Firesmith is a senior member of the technical staff at the Software Engineering Institute. He has worked exclusively with object technology since 1984 and has written 5 books on the subject. He is currently writing a book on requirements engineering. Most recently, he has developed a 1000+ page informational website on the OPEN Process Framework at www.donald-firesmith.com. He can be reached at donald_firesmith@hotmail.com.