Using Abuse Frames to Bound the Scope of Security Problems

Luncheng Lin Bashar Nuseibeh Darrel Ince Michael Jackson
Department of Computing
The Open University
Walton Hall
Milton Keynes MK7 6AA, U.K.
E-mail: {L.C.Lin, B.Nuseibeh, D.C.Ince, M.Jackson}@open.ac.uk

Abstract

Security problems arise from the concern for protecting assets from security threats. In a systems development process, the security protection of a system is specified by security requirements, identified from the analysis of the threats to the system. However, as it is often not possible to obtain a full system description until late in the RE process, a security problem often has to be described in the context of a bounded scope, that is, one containing only the domains relevant to some part of the functionality of the full system. By binding the scope of a security problem, it can be described more explicitly and precisely, thereby facilitating the identification and analysis of threats, which in turn drive the elicitation and elaboration of security requirements. In this poster, we elaborate on an approach we developed based on abuse frames and suggest how it can provide a means for structuring and bounding the scope security problems.

1. Introduction

Security problems arise when there is a need to protect assets from threats. In systems requirements engineering, a security problem can be addressed by identifying and analysing the threats to a system, and specifying appropriate security requirements to counter those threats. Our research focuses on the explicit representation of security threats to facilitate the analysis of security problems. Our general research objectives are: *1) to define a precise notion of security threat, 2) to devise an appropriate representation for expressing such a threat, and 3) to use such representation to structure and bound the scope of security problems.*

We have previously addressed the first two objectives in [3, 4] by proposing the *abuse frames* technique to represent security threats. In this poster, we focus specifically on the third objective, and elaborate our technique to facilitate the bounding and structuring of security problems.

The abuse frames technique adopts the concepts and notation of problem frames [2] for two reasons. First, the problem frame notation extends on the context diagram notation, providing a convenient means of describing a system context in terms of the domains and their interactions in the system. Second, the classification of elementary problem frames provides a granularity of problem descriptions that allows the subproblems of a system development problem to be structured and analysed individually.

2. Abuse Frames and Terminology

In contrast to problem frames, abuse frames consider threats to a system from the viewpoint of a malicious user. We define a threat to be the potential for use of domains in the system to cause harm. An *attack* is defined as a realisation of a threat.

In [1], we introduced the notion of *anti-requirement*s (AR) to represent the requirements of users with malicious intent. We incorporate anti-requirements into abuse frames to represent a security threat [3, 4]. Figure 1 is *one* generic structure of an abuse frame.

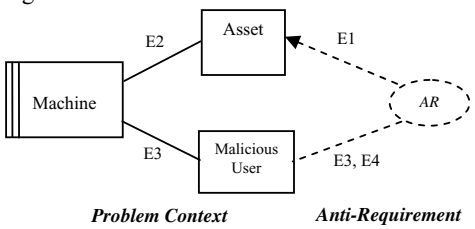


Figure 1: A generic abuse frame diagram.

The plain rectangles represent problem domains. The Asset domain is the domain under attack. The Machine domain, which is represented by the rectangle with two vertical stripes, acts as the interface between the malicious user and the Asset. The Malicious User domain represents the domain that is imposing the threat. Phenomena shared between two domains are represented by an annotated line connecting the two domains. The problem context represents a projection of the system, which can be the whole or a part of the system that is under attack

The anti-requirement, AR, indicated by the dashed oval, specifies the *observable* and undesirable phenomena E1 in the Asset domain as the result of E2. The interface of the Machine describes the relation between the E3 of the Malicious User and the E2 of the Asset that, in conjunction with the properties of the Malicious User and

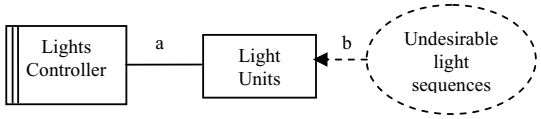
Proceedings of the 12th IEEE International Requirements Engineering Conference (RE'04)
1090-705X/04 $ 20.00 IEEE

Asset domains, will satisfy the anti-requirement. In cases where the threat is realised without the active participation of a malicious user, the malicious user domain may be omitted in the diagram.

3. Bounding the Scope of Security Problems

Consider the example problem of developing a traffic light system as discussed in [2]. The problem can be decomposed into two subproblems: the traffic light controller and the light regime editor. Each subproblem is concerned with a different aspect of functionality of the traffic light system, and each can be represented by a problem frame and analysed separately.

The light controller is directly connected to the traffic light units; its purpose is to cause the light units to show light signals in compliance with a predefined light regime. The threat to this subproblem can be represented by an abuse frame. As an example, one potential anti-requirement to the light controller is to cause the light units to follow an incorrect light sequence (Figure 2).



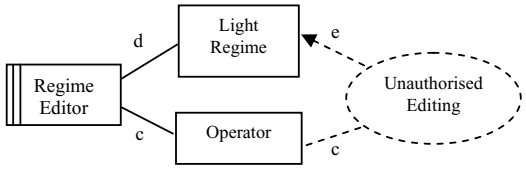
a: Lights Controller sends signals to control the Light Units.
b: The Light Units showing Stop and Go.

Figure 2: An abuse frame bounding the scope of light controller subproblem.

The anti-requirement, *undesirable light sequences,* specifies the undesirable behaviour of the Light Units. The scope of threat analysis for this subproblem is defined by the abuse frame, and includes only the Light Controller and the Light Units. The threat analysis is concerned with the evaluation of the security properties of the lights controller subproblem with respect to the satisfaction of the anti-requirement. It has to show whether the threat can be realised (i.e., the anti-requirements is satisfied) through some form of attack.

On the other hand, suppose that we would like to analyse the security properties of the light regime editor first because the requirements for the light controller are still being elaborated.

The light regime editor allows the traffic light operator to edit the light sequence described in the light regime. Figure 3 shows potential security threat to this subproblem. The Operator is assumed to be malicious and the anti-requirement, *unauthorised editing,* specifies the unauthorised modification of the light regime. This abuse frame defines the scope of threat analysis for this subproblem. The scope includes only the Regime Editor, Lights Regime, and the Operator domains, and no references to other subproblems are made. The threat analysis conducted using this abuse frame is concerned with the evaluation of the security properties of the lights regime editor subproblem with respect to the unauthorised editing by a malicious operator.



c: Editing commands entered by the Operator.
d: The edit operations performed by the Regime Editor.
e: The effects of the edits on the Light Regime.

Figure 3: An abuse frame bounding the scope of the light regime editor subproblem.

Figure 4 shows the traffic light system that is obtained by composing the two subproblems through the Light Regime domain. The two anti-requirements identified previously are also shown in the diagram.

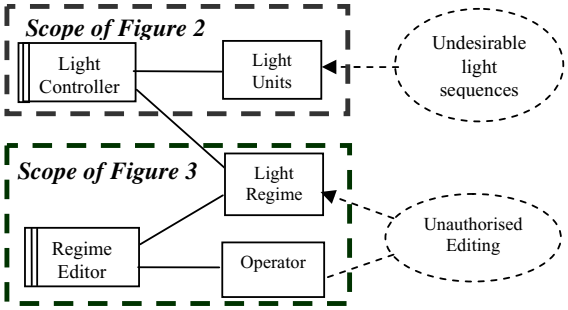


Figure 4: An abuse frame describing two security problems in a traffic light system.

The large dotted rectangles indicate the scope of threat analysis for each subproblem. Because each scope includes only the domains relevant to the respective subproblem, separation of concerns in describing the security problem in each subproblem is achieved. The security problems identified in each subproblem can be represented individually for threat analysis during the elaboration of systems requirements.

References

[1] R. Crook, D. Ince, L. Lin, and B. Nuseibeh, "Security Requirements Engineering: When Anti-requirements Hit the Fan", *Proc. of the 10th IEEE Requirements Engineering Conference (RE'02)*, Germany, Sept. 2002.

[2] M. Jackson, *Problem Frames: Analysing and structuring software development problems*, Addison Wesley, 2001.

[3] L. Lin, B. Nuseibeh, D. Ince, M. Jackson, and J. Moffett, "Analysing Security Threats and Vulnerabilities Using Abuse Frames", *Technical Report 2003/10*, Department of Computing, The Open University, October 2003.

[4] L. Lin, B. Nuseibeh, D. Ince, J. Moffett, and M. Jackson, "Using Abuse Frames to Analyse Security Requirements", *Proc. of 11th IEEE International Requirements Engineering Conference (RE'03)*, USA, Sept. 2003.