Security Threat Models: An Agile Introduction

Security threat modeling, or threat modeling, is a process of assessing and documenting a system's security risks. Security threat modeling enables you to understand a system's threat profile by examining it through the eyes of your potential foes. With techniques such as entry point identification, privilege boundaries and threat trees, you can identify strategies to mitigate potential threats to your system. Your security threat modeling efforts also enable your team to justify security features within a system, or security practices for using the system, to protect your corporate assets.

There are five aspects to security threat modeling:

1. **Identify threats**. The first thing to do is to identify assets of interest, you first model the system either with data flow diagrams (DFDs) or UML deployment diagrams. From these diagrams, you can identify entry points to your system such as data sources, application programming interfaces (APIs), Web services and the user interface itself. Because an adversary gains access to your system via entry points, they are your starting points for understanding potential threats. To help identify security threats you should add "privilege boundaries" with dotted lines onto your diagrams. Figure 1 depicts an example deployment diagram used to explain the boundaries applicable to testing a relational database
. A privilege boundary separates processes, entities, nodes and other elements that have different trust levels. Wherever aspects of your system cross a privilege boundary, security problems can arise. For example, your system's ordering module interacts with the payment processing module. Anybody can place an order, but only manager-level employees can credit a customer's account when he or she returns a product. At the boundary between the two modules, someone could use functionality within the order module to obtain an illicit credit.
2. **Understand the threat(s)**. To understand the

potential threats at an entry point, you must identify any security-critical activities that occur and imagine what an adversary might do to attack or misuse your system. Ask yourself questions such as "How could the adversary use an asset to modify control of the system, retrieve restricted information, manipulate information within the system, cause the system to fail or be unusable, or gain additional rights. In this way, you can determine the chances of the adversary accessing the asset without being audited, skipping any access control checks, or appearing to be another user. To understand the threat posed by the interface between the order and payment processing modules, you would identify and then work through potential security scenarios. For example, an adversary who makes a purchase using a stolen credit card and then tries to get either a cash refund or a refund to another card when he returns the purchase.

3. **Categorize the threats**. To categorize security threats, consider the STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, and Elevation of privilege) approach. Classifying a threat is the first step toward effective mitigation. For example, if you know that there is a risk that someone could order products from your company but then repudiate receiving the shipment, you should ensure that you accurately identify the purchaser and then log all critical events during the delivery process.
4. **Identify mitigation strategies**. To determine how to mitigate a threat, you can create a diagram called a threat tree. At the root of the tree is the threat itself, and its children (or leaves) are the conditions that must be true for the adversary to realize that threat. Conditions may in turn have subconditions. For example, under the condition that an adversary makes an illicit payment. The fact that the person uses a stolen credit card or a stolen debit/check card is a subcondition. For each of the leaf conditions, you must identify potential mitigation strategies; in this case, to verify the credit card using the XYZ verification package and the debit card with the issuing financial institution itself. Every path through the threat tree that does not end in a mitigation strategy is a system vulnerability.

5. **Test**. Your threat model becomes a plan for penetration testing. Penetration testing investigates threats by directly attacking a system, in an informed

or uninformed manner. Informed penetration tests are effectively white-box tests that reflect knowledge of the system's internal design , whereas uninformed tests are black box in nature.	

Figure 1. A UML deployment diagram with threat/privilege boundaries.