Simplifying Threat Modeling

Mike Ware
Cigital, Inc.
1.703.404.9293, x1251

OWASP
9/23/2011

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document under the terms of the OWASP License.
Today’s Threat Modeling Theme

Keep it simple.
Objective:
Provide a framework to facilitate a threat modeling roundtable

Builders
- Developers
- Vendors

Gluers
- Enterprise Arch
- CTO
- Shared Services

Owners
- Program
- Product
- Project
- Business
- Requirements
- ISO
- IRM

Defenders
- Infrastructure
- Ops

Breakers
- SSG
- External Pen Testers
What is a Threat?

- Anything (e.g., object, human) capable of performing unauthorized actions against a software system
- Possess skills, access, and resources

OWASP NoVA Chapter: https://groups.google.com/forum/#!forum/novaowasp_threatmodeling
Threat Example – Mobile Architecture

**Malicious Device User** (1)

**Skills**
- Jailbreak device
- Reverse engineer software
- Install/modify software

**Access**
- Access to device
- Access to apps/browsers
- Access to device SDK

**Resources**
- Possess device/app credentials
- Disassemblers, proxies
A neighboring network user eavesdrops on Internet traffic using a network sniffer and steals another user’s session id. Attacker replaces her browser session id with victim’s id and gains access to victim’s account thereby impersonating the victim.
## Threat Traceability Matrix

<table>
<thead>
<tr>
<th>Who</th>
<th>Where</th>
<th>What</th>
<th>How</th>
<th>Impact</th>
<th>Mitigation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threat</td>
<td>Attack Surface</td>
<td>Conceptual Goals</td>
<td>Tech-Specific Exploits</td>
<td>Consequence</td>
<td>Control</td>
</tr>
</tbody>
</table>

This matrix helps in tracking and understanding the progression of threats from conceptual goals to specific technical exploits and their impact on control measures.
Elements of a Threat Model

- **Software architecture** – structure, interaction, control flow, frameworks, services, design patterns
- **Threats**
- **Assets** (data and function)
- **Attack Vectors**
- **Security Controls**
- **Notion of ‘trust’**
Simplified Threat Modeling Framework

Views of a software system
 Builders

• Developers
• Vendors

Gluers
• Enterprise Arch
• CTO
• Shared Services

Defenders
• Infrastructure
• Ops

Breakers
• SSG
• External Pen Testers

Owners
• Program
• Product
• Project
• Business
• Requirements
• ISO
• Risk

Traceability Matrix

Abuse/Misuse

Trust Boundaries

Asset Flow

Attack Surface

• Enterprise Arch
• CTO
• Shared Services

• Developers
• Vendors

• Infrastructure
• Ops

• SSG
• External Pen Testers

• Program
• Product
• Project
• Business
• Requirements
• ISO
• Risk

• Enterprise Arch
• CTO
• Shared Services

• Developers
• Vendors

• Infrastructure
• Ops

• SSG
• External Pen Testers

• Program
• Product
• Project
• Business
• Requirements
• ISO
• Risk
Keep it simple.

7+1 Threat Modeling Steps
1. Diagram Software Architecture
2. Enumerate Attack Surface(s)
**Viewpoints**

- Gluers
- Builders
- Breakers
- Defenders

**SDLC**

- Design
  - High level architecture
  - Low level design

**Characteristics**

- Interfaces enabling interaction
  - Web, services, middleware, data tier, etc.

- Interaction model
  - Synch, async, transactional
  - Stateful, stateless

- Technology enabling interaction
- Authentication/authorization

**Inputs/Usage**

- Design/architecture changes
- Integration with:
  - Frameworks, toolkits, 3rd party libraries
  - Partners, service providers
  - Other enterprise systems
- Discovery, mapping, and other tool usage

- ‘WHERE’ traceability matrix column

**Attack Surface View**
Threat Traceability Matrix

<table>
<thead>
<tr>
<th>Who</th>
<th>Where</th>
<th>What</th>
<th>How</th>
<th>Impact</th>
<th>Mitigation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Threat
- Conceptual Goals
- Consequence
- Attack Surface
- Tech-Specific Exploits
- Control
### 3. Each User Class Becomes a Threat

<table>
<thead>
<tr>
<th>User</th>
<th>Threat</th>
<th>Malicious Intent</th>
<th>Non-Malicious Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account Holder</td>
<td>Malicious Customer</td>
<td>Fraud, steal money, sabotage accounts</td>
<td>Inadvertent account lockout</td>
</tr>
<tr>
<td>Customer Support Representative (CSR)</td>
<td>Malicious CSR</td>
<td>Sell sensitive customer information</td>
<td>Backup customer data</td>
</tr>
<tr>
<td>Phone User</td>
<td>Malicious Device User</td>
<td>Install malware, reverse engineer app, jailbreak phone</td>
<td>Lose phone</td>
</tr>
</tbody>
</table>
### Malicious Intent Creates New Threat

<table>
<thead>
<tr>
<th>User</th>
<th>Threat</th>
<th>Malicious Intent</th>
<th>Non-Malicious Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account Holder</td>
<td>Malicious Customer</td>
<td>Fraud, steal money, sabotage accounts</td>
<td>Inadvertent account lockout</td>
</tr>
<tr>
<td>Customer Support Representative (CSR)</td>
<td>Malicious CSR</td>
<td>Sell sensitive customer information</td>
<td>Backup customer data</td>
</tr>
<tr>
<td>Phone User</td>
<td>Malicious Device User</td>
<td>Install malware, reverse engineer app, jailbreak phone</td>
<td>Lose phone</td>
</tr>
<tr>
<td>Malicious Device</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Visualize Normal Users as Threats
Re-consider Attack Surface(s)
Abuse/Misuse Case View

- Owners
  - Business
  - Product
  - Requirements
- Breakers

Viewpoints

- Use cases, user story elicitation
- High level requirements definition
- List of threat actor profiles
  - Skills
  - Access
  - Resources

- Link abuse/misuse to ‘WHERE’
- ‘WHO’, ‘WHAT’, ‘HOW’

Characteristics

- Abuser/misuser (actor)
- System interface to actor (attack surface)
- Preconditions
- Inputs
- Actor’s actions
- Expected outcomes

SDLC

- Functional
- Non-functional

Requirements
## Capture ‘Who’, ‘Where’, and ‘What’

<table>
<thead>
<tr>
<th>Who</th>
<th>Where</th>
<th>What</th>
<th>How</th>
<th>Impact</th>
<th>Mitigation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Malicious Account Holder</td>
<td>User’s Browser</td>
<td>• Execute fraudulent transactions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Malicious CSR</td>
<td>Desktop Client</td>
<td>• Steal customer PII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Malicious Mobile Device</td>
<td>Phone OS, SDK</td>
<td>• Capture and transfer application data</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7. Malicious Third Party</td>
<td>User’s Browser</td>
<td>• Steal user credentials</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. Illuminate Assets

Browser/Client
- DOM
- AJAX
- Javascript
- Untrusted

Internet
- Desktop Client
- NET CLR
- Phone
- Browser
- Native App

Application Server
- Servlet Mapping
- JSPs
- Interactions
- Banking Controller
- UserAccount Form Controllers
- Session Identifier
- Credentials
- Principal
- PII
- Account Info: balance, IDs, withdrawal, deposit, transfer

Database Host
- DB
- Accounts Gold Source

Hosting LAN
- Middleware
- LAN
- LDAP

Cache, Store
- Internet

Data Tier

Application Tier
**Asset Flow View**

- Owners
  - Risk (IRM)
- Gluers
- Builders
- Breakers

**Viewpoints**

**Characteristics**

- Data and functionality
- Threat agent(s) level of access
- Exposure to attack surface(s)
- Asset classification
- Protection mechanisms
  - Rest, process, transit
  - Egress, ingress
- Qualifying technologies

**SDLC**

**Requirements**

- Information architecture
- High level architecture diagram

**Design**

**Inputs/Usage**

- Data View + CRUD
- Schemas, config, DTDs
- SCR, VA assessment results

- Enhance ‘WHAT’, ‘HOW’ with contextual information
- Evaluate ‘IMPACT’ of abuse/misuse
5. Illuminate Trust Boundaries

- Browser
  - DOM
  - AJAX
  - Javascript

- Desktop Client
  - .NET CLR

- Phone Browser
  - Native App

- Untrusted

- Internet

- Feed Aggregator App
  - LDAP
  - LAN
  - Middleware
  - Cache, Store

- Hosting LAN

- Application Server
  - Rate Aggregation
  - ACH Verify
  - REST Services

- Servlet Mapping
  - Banking Controller
  - UserAccount Form Controllers
  - Pending Transaction Controller
  - User Controller
  - Account Controller
  - Hibernate

- Database Host
  - DB

- Accounts Gold Source
  - LAN

- Spring

- Session Identifier
- Credentials
- Principal
- PII
- Account Info: balance, IDs, withdrawal, deposit, transfer

- Input Presumed Validated
- User Presumed Authenticated
- Acct GUID Presumed Protected
# 6. Postulate Attacks Against Assets

<table>
<thead>
<tr>
<th>Who</th>
<th>Where</th>
<th>What</th>
<th>How</th>
<th>Impact</th>
<th>Mitigation</th>
</tr>
</thead>
<tbody>
<tr>
<td>3. Malicious Mobile Device User (unauthenticated)</td>
<td>User’s Browser, Native Phone App</td>
<td>Execute fraudulent transactions</td>
<td>• Directly make REST API requests using another customer’s account identifier</td>
<td>• CSRF attack against another customer</td>
<td></td>
</tr>
</tbody>
</table>


## 7. Evaluate Impact

<table>
<thead>
<tr>
<th>Who</th>
<th>Where</th>
<th>What</th>
<th>How</th>
<th>Impact</th>
<th>Mitigation</th>
</tr>
</thead>
<tbody>
<tr>
<td>3. Malicious Mobile Device User (unauthenticated)</td>
<td>User’s Browser, Native Phone App</td>
<td>Execute fraudulent transactions</td>
<td>• Directly make REST API requests using another customer’s account identifier</td>
<td>• Fines</td>
<td>• Brand damage (PR incident)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• CSRF attack against another customer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Authenticated Malicious User</td>
<td>User’s Browser, Native Phone App</td>
<td>Modify user account information</td>
<td>• Account recovery costs</td>
<td>• Lose customer(s)</td>
<td></td>
</tr>
</tbody>
</table>

**Traceability Matrix**
## 8. Mitigate

<table>
<thead>
<tr>
<th>Who</th>
<th>Where</th>
<th>What</th>
<th>How</th>
<th>Impact</th>
<th>Mitigation</th>
</tr>
</thead>
</table>
| 3. Malicious Mobile Device User (unauthenticated) | User’s Browser, Native Phone App | Execute fraudulent transactions | • Directly make REST API requests using another customer’s account identifier | • Fines  
• Brand damage  
• Account recovery costs | R.1.a: Authenticate REST API requests (user level)  
R.1.b: Authorize all REST API calls (message level)  
S.1.a: Implement request tokens for all state changing servlets |
Trust Boundaries View

- Gluers
- Breakers
- Defenders

Viewpoints

Characteristics

- Boundaries defined by set of security properties
  - AuthN/AuthZ
  - I/O Controls
  - Privileged functionality/data
  - Connections & protocols
  - Object marshaling and remoting
  - Queues, channels
  - …

- Postulate ‘HOWs’ by speculating about weaknesses in trust boundary implementations

SDLC

Design

- High level architecture
- Low level design

Inputs/Usage

- ‘Attack Surface View’
- ‘Asset Flow View’
7+1 Threat Modeling Steps

1. Diagram Software Architecture
2. Enumerate Attack Surface(s)
3. Document Threats
4. Illuminate Assets
5. Illuminate Trust Boundaries
6. Postulate Attacks
7. Evaluate Impact
8. Mitigate
Acting on Threat Modeling Results

Threat Modeling

- Software Architecture
- Intrinsic Risk
- Use Cases

- Secure Design
- Assessment (SCR, VA, Pen Test)
- Risk Management
Contact

- Mike Ware
- Sr. Security Consultant, Cigital
- mware at cigital dot com