Threat Modelling (Web)Apps
Myths and Best Practices

Matthias Rohr
www.matthiasrohr.de
mail@matthiasrohr.de

OWASP 7.11.2012

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document under the terms of the OWASP License.
About me

- Matthias Rohr
- Dipl. Medieninf. (FH), CISSP, CSSLP, CCSK
- Focus: Application Security Management
- Contractor in London – from 2013 on back in Hamburg
- Active in OWASP since 2007:
  - OWASP ASVS/Java/Skavenger Project
  - Review of “BSI Baustein Webanwendungen”
  - WAF Best Practice Paper
  - OWASP Summits
Motivation I: Pushing Appsec Left in the SDLC

- **Costs** to fix a bug
- **Level of Security** (derived also from the costs)
- **Planability**: Sec tests may lead to “surprises”
- **Visibility** within SDLC:

“60% of all weaknesses are visible in the application design”

*(Principles of Software Engineering Management, T. Gilb)*
Motivation II: The Transformation Problem

Security Management
“Risk & Compliance World”

Dev
“Feature and Bug World”

Pentester
“Vulnerability and Exploit World”

Application

e.g. what is the risk of this vulnerability?

e.g. what are the risks of this feature?

e.g. what feature does this vulnerability relates to?

Other stakeholders:
- Project Managers
- QA
- Operation

e.g. what is the risk of this vulnerability?
Threat Modelling - Goals

- **Primary**
  - Early identification, assessment and correction of **potential** security problems in an IT system (such as a Web application)
  - Link technical implementation to IT Risk Mgmt & ISMS

- **Secondary**
  - Improvement of planability & quality of later security tests (pentests, code reviews, etc.)
  - Documentation and discussion of the application security architecture
What is a Threat?

- Asset (Information, Func., Sys)
- Protection Req. (CIA+A)
- Vulnerability
- Risk
- Likelihood / Impact

- Activity (Access, Store, Trans)
- Threat (Attack, Weakness, BI)
- Threat Agent (Motivation + Capabilities)
  - Spoofing
  - Tampering
  - Repudiation
  - Information Disclosure
  - Denial of Service
  - Escalation of Privilege
  - malicious / non-malicious
Existing Methodologies

- Microsoft I (2003, “DREAD”)
- Microsoft II (2009, “Bug Bars”)
- OWASP I + OWASP II
- PASTA
- T-MAPS
- PTA
- SANS
- Trike

Difficult to compare due to different concepts.
Tools

- Word, Excel, Visio or any Wiki, etc.
- Microsoft Threat Modelling Toolkit (TAM): free MS Visio Plugin, but limited (DfD* analysis only)

DfD = Data flow Diagram
Myths
(or just misunderstandings……)
Myth 1: Threat Modelling is too Complicated

- Threat modelling is a **best effort** approach
  - Identifying only some threats is better than nothing at all
  - Objective is not 100% threat coverage
  - Learning and integration process: Start simple & informal
- Every stakeholder can conduct some sort of **threat assessment** in principle (e.g. developers, project managers, ...)

* A threat assessment is not necessarily a threat modelling!
Myth 2: Threat Modelling = Design Review

- Many threats are already visible in the specification!
- Hence: See TM as a **conceptual security analysis**!

- A threat model can be created in **iterations** (allows us to start very early and with a limited model)
- A threat model can be updated with details from implementation and operation phase.
Myth 3: TM Output = a List of Threats

- Lists are **static**, models can be **dynamic**
- Change of a system’s property (e.g. a data flow) may effect its threats and therefore the threat model too.
- Lists as result of a generic “threat analysis” ok of course.

See also: http://www.curphey.com/2012/03/is-threat-Modelling-overrated
Myth 4: Decide for ONE Perspective

- **Attack-centric**: Focuses on attacks
  - May suit a pentester
  - Example: “XSS attack to steal cookies”

- **Software-/system-centric**: Focuses on weaknesses
  - May suit a developer or SW architect
  - Example: “Insufficient output validation controls”

- **Asset-/Risk-centric**: Focuses on business impact (BI)
  - May suit an infosec manager
  - Example: “Attacker may access customer data via ...”

Multiple perspectives may lead to a lot overlapping threats, but will also increase threat coverage!!!
Myth 5: One Methodology suits them all

- For example Microsoft’s TM:
  - Methodology is based on DfD analysis
  - Software-centric = focused on SW developers
- Instead, the approach should be specific to
  - The (development) organisation
  - Both SDLC and SDL
  - The qualification of the analyst
  - The protection requirements of the app
  - Existing resources
  - ...
- Known as: Tailoring
Best Practices
(based on my personal experiences)
Threat Intelligence (TI)

- Main idea: Mapping of **expert know-how** and other intelligence to a threat modelling exercise
- Examples: Gen. threats, metrics, countermeasures, etc.
- Essential for integrating threat modelling into SDLC, improving quality & reducing resources

See also “Attack Models” practice in BSIMM study: http://bsimm.com/online/intelligence/am
Step 0: Preparation

- Plan threat modelling exercise early in project mgmt:
  - Select suitable threat modelling methodology (internal or external)
  - Input requested from whom and when?
  - Output provided to whom and when?
  - Early kick-off (after this: update planning)
  - Estimate required SMEs*

- Consider exercise as a quality gate

- Use RACI to define responsibilities / estimate resources

*SME = Subject Matter Expert
Step 1: Assessment Definition

- Describe the application
  - Name, version, etc.
  - Business objectives
  - Sec requirements
  - Stakeholder

- Define **scope**
  - Target of Assessment (ToA)
  - Exclude platform, IDM, container, etc.

- Define **constrains**: Trust assumptions, etc.,
  - “Data from IDM or SAP FI system is trust worthy”
  - Irrelevant threat scenarios to be ignored
Step 2: Application Decomposition (AD)

- **Identify** sub-systems, system boundaries and external dependencies.
- **Describe** assets, actors (including trust levels!), DfDs*, use cases*, entry points (channels)
- **Derive** (link) these information as shown left (e.g. using Word refs).
- This step may delivered as part of the development documentation.

* focus on DfDs and use cases that affect identified assets!
Step 2: AD: Application Overview

- Create a layer 7 view of the security architecture (no backup, cluster or other network devices).
- Don’t bother with UML standards.
- Instead: use **hybrid diagrams**. Focus: **Visualisation**!

Dashed lines are **trust boundaries** (= architectural trust assumptions)
Step 3: Clustering (optional)

- Applications can consist technically heterogeneous components leading to different threat profiles.
- Common example:
  - External Web interface for end-users
  - Internal admin GUI
- Clustering is used to identify such components and divide the threat model respectively.
Step 4: Threat Identification

- **Objective:** Maximization of coverage (don’t be afraid of duplicates/overlapping threats!).
- **Where/How may protection requirements of an asset be affected***:
  - **Primary:** Mainly confidentiality, integrity
  - **Secondary:** Authentication, loss of repudiation, etc.
  - **Indirect:** Design Principles (Least Priv., etc.)

* = potential damage to it
Step 4: Threat Identification – Building Blocks

- Questionnaires
- Attribute threat mapping
- Known vulnerability analysis
- Roles and permissions analysis
- Abuse & misuse case modelling
- Security control analysis
- Attack models / attack patterns
- Attack surface analysis
- Attack trees
- DFD analysis: STRIDE mapping, trust boundary analysis, ...
- Input of pentests, other threat models, ...
Step 4: Threat Identification - Tips

- Selection of activities depends on
  - **Protection requirements** (of the app)
  - Level of **maturity** (of the organisation)
  - **Qualification** (of the analyst)
  - **Resources** & time

- Tip: Do not focus on STRIDE*. Use own categories instead that helps you to derive threats from them:
  - e.g. “Threats regarding roles and permissions.“ (see example in appendix!)

Step 4: Misuse & Abuse Cases

■ Misuse Case Modelling
   ■ Based on use cases (of identified assets)
   ■ Analyze cases step-by-step:
     What could happened / should not happen that could cause damage to an asset?

■ Abuse Case Modelling
   ■ Not based on use cases
   ■ What can a specific threat agent (e.g. admin, specific user such as a trader, hacker) do that could result in damage to an asset?
Step 4: Attribute Threat Mapping (ATM)

- Idea: Use threat intelligence to map application properties to generic (or known) threats (expert system).

- Technical ATM (simple approach):

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Threats (Weaknesses, Attacks, BI)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Func.Register</td>
<td>- An attacker may enumerate users names&lt;br&gt; - Missing anti-automation</td>
</tr>
<tr>
<td>Func.Auth.Custom</td>
<td>- Insecure Session Identifier (CWE-330)&lt;br&gt; - Authentication Bypass (CAPEC-115)&lt;br&gt; - Insecure Password Storage (CWE-261)&lt;br&gt; - PW Eavesdropping (CAPEC-94)</td>
</tr>
<tr>
<td>Func.Auth.PWReset</td>
<td>- Weak Password Recovery (CWE-640)</td>
</tr>
</tbody>
</table>

Better approach: Map certain attributes using a logic (and, or, not) to specific threats.

- Create **threat profiles** for certain app types (e.g. collaboration, HR app, etc.)
Step 5: Threat Revision

- **Consolidation**
  Combine similar threats

- **Identify Mitigating Factors**
  Incl. controls, existing and planned

\[ T = \text{Threat} \]
\[ TA = \text{Threat Ident. Activity} \]
Step 5: Threat Revision

- **Consolidation**
  Combine similar threats

- **Identify Mitigating Factors**
  Incl. controls, existing and planned

- **Pre-Assessment (optional)**
  Check relevance / known issues
Step 6: Threat Rating

- **Threat Criticality Rating**
  - Option 1: DREAD: Criteria's are mapped indirectly to a numerical value using a metric (MS TM I) => Often very subjective!!
  - Option 2: CWSS: Similar to DREAD but more granularly and precise (= more work)
  - Option 3: Bug Bars: Criteria's that are mapped directly to low, medium, high, etc. (MS TM II)
  - ...

- **Risk Assessment**
  - Threat Modelling → Risk Assessment

CWSS: http://cwe.mitre.org/cwss/
Step 7: Threat Treatment (Countermeasures)

- Implemental
  - E.g. code changes
- Configurative
  - E.g. system hardening
- Architectural
  - E.g. installation of a PKI, IDM solution
- Other
  - Guidelines
  - Tests
  - ...

Talk - Action = Shit
Threat 8: Threat Validation (Test Cases)

- Derive **test plan** & test cases from countermeasures
- Can easily include **generic test cases** (TI)
- Result: Threat-based security testing

```
Threat
  ↓
Threat Treatment
  ↓
Sec Test Cases
  ↓
Validation
```

- Gen. Countermeasures
  - e.g. via Pentest (Vulnerability Assessment)
- Gen. Test Cases
Step 9+10: Threat Retrospective & Update

- Update threat intelligence:
  - Known issues
  - Security test cases
  - Attribute threat mappings
  - Abuse cases
  - Metrics
  - ...

- Continuous improvement of threat modelling exercises
- Update of the threat model after a specific time / changes
Threat Modelling & Risk Assessments

**Approach I:**
Assessing threats only

- Threat Modelling
- Threats / Countermeasures

**Approach II:**
Assessing threats and risks separately

- Threat Modelling
- Threat Model
- Risk Assessment
- Threats / Risks / Risk Mitigations

**Approach III:**
Assessing threats and risks in one activity

- Threat & Risk Assessment
- Threats / Risks / Risk Mitigations

Try to implement approach II or III
This TM Approach

- Application Decomposition
- Clustering
- Threat Identification
- Threat Revision
- Threat Rating
- Threat Treatment
- Threat Validation

NIST RA (SP 800-30)

- System Characterization
- Threat Identification
- Control Analysis
- Vulnerability Identification
- Likelihood Determination
- Impact Analysis
- Risk Determination
- Control Recommendations

Easy to combine both exercises. The **WHERE** is specific to an existing RM methodology!
So Where to Start?

- Begin simple, informal and learn! (e.g. as a pilot)
- Collect threat intelligence wherever possible
  - **Lessons learned** after pentests, projects, etc.
- Integrate stakeholders: Dev team, TPMs, SME, pentester, etc.
- Build a **roadmap**:
  - Prioritize critical apps and platforms
  - Process maturity / SDLC integration
- **Get help**: E.g. let complicated threat models may be conducted by experienced consultants companies and learn from them!
Thank You! Any Questions???
APPENDIX: Possible Threat Groups

- Insecure systems or missing hardening threats (HRD)
- Local threats (LOC)
- Threats by privileged users (PRV)
- Denial-of-Service threats (DOS)
- Threats to authentication & identities (ATN)
- Access control threats (ATZ)
- Threats regarding roles and permissions (RLP)
- Manipulation or disclosure of data in motion (DMM)
- Manipulation or disclosure of data at rest (DMR)
- Business-logic specific threats (BIL)
- Privacy threats (PRV)
- Accountability threats (ACC)
APPENDIX: Overview of Methodology

Input
- Specification
  - Requirements
  - Design
- Threat Intelligence
  - Known Threats
  - Pentest Results
  - Authorization Model
- Preassessments
- Predefined Countermeasures
- Predefined Testcases

Define Assessment
- Constraints
- Describe App
- Define Approach

Decompose System
- Use Cases
- Actors
- Entry Points
- Security Controls
- Trust Boundaries
- Security Controls
- Assets
- System Diagram
- Data Flows

Clustering

Threat Identification
- Attribute Mapping
- Pentest Results
- Misuse Cases
- Questionnaires
- Abuse Cases
- Role Permissions

Threat Revision
- Consolidate Threats
- Identify Mitigating Factors
- Check Relevance

Measure Threats
- Risk Assessment
- Threat Criticality Rating

Treat Threats
- Mitigate
- Accept
- Avoid
- Transfer
- Config
- Implement
- Architect
- Organisat

Threat Validation
- Testcases

Update & Retrospective
- Update ToA
- Update Threat Profile
- Update Threat Model

Output
- ToA Specification
  - Update
- Threat Model
  - Update
- Risk Model
  - Update
## APPENDIX: RACI Example

<table>
<thead>
<tr>
<th>Step</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>App Owner</td>
</tr>
<tr>
<td>Preparation</td>
<td>C</td>
</tr>
<tr>
<td>Assessment Definition</td>
<td>C</td>
</tr>
<tr>
<td>App Decomposition</td>
<td>C</td>
</tr>
<tr>
<td>Threat Identification</td>
<td>C</td>
</tr>
<tr>
<td>Threat Revision</td>
<td>C</td>
</tr>
<tr>
<td>Threat Rating</td>
<td>I</td>
</tr>
<tr>
<td>Define Action Plan</td>
<td>C</td>
</tr>
</tbody>
</table>

R – Responsible  
A – Accountable  
C - Consulted (in the loop)  
I - Informed (in the picture)