Threat Modeling Introduction

John E. Savage
Computer Science
Brown University
Source Material


Threat Modeling

• Model ways to violate security and do damage
  – A good model helps to improve security
  – The STRIDE acronym helps to find threats

• Organize your threat analysis
  – Structure your analysis
  – Choose a starting point, e.g. on external entities
  – Don’t ignore threats, you may forget them later
  – Focus on feasible threats, not very unlikely ones.
Building a Threat Model

• Illustration: A House Break-in
  – What are the usual entry points?
  – What are unusual entry points?

• Levels of threat
  – First level, e.g. via windows and doors
  – Second level threats are threats to first level mitigations
  – Ditto for third and higher levels

• Defense in depth copes with multiple threat levels
Functioning Under Attack

• Should a computer system be able to function after being penetrated?
  – How effective would a fighter jet be if it was disabled by the first hit?

• Defense in depth must anticipate effects of penetrations and allow operations to continue, perhaps in a degraded mode.
How to Find Threats

1. **Describe** what you are building
2. Find **damage** that might be done to it
3. Try to **mitigate** the potential damage
4. **Assess** effectiveness of your mitigation
5. During deployment **return** to 2.
Creating a System Model

• Is it better to model top down or bottom up?
  – What are advantages/disadvantages of each?
• Bottom-up may overlook interactions
• Top-down preferred because it sees interactions
  – It can’t see low level threats but they can they wait
• Is depth-first or breadth-first preferred?
  – If time is limited, use breadth-first.
Techniques to Identify Security Threats

• Brainstorming
  – Idea generation followed by analysis
  – Guidance from an expert very helpful

• Scenario Analysis
  – Consult attacker type lists (Appendix C of Shostack)
    • Barnard’s list, Verizon list, OWASP list, Intel TARA
  – Study personas of attackers
  – Invent attack scenarios, construct

• Do literature review looking for similarities
Techniques to Identify Security Threats

• Structured threat identification types:
  – Focus on assets
    • What attackers want, you treasure, or stepping stones
  – Focus on attackers
    • What are their motivations, how hard will they work? Helps to make threat real
  – Focus on software
    • Communicate decisions to team members, and
    • Data flow diagrams of software are useful.

• This process is designed to sensitize designers
Critiques of Techniques

• Asset-centric threat modeling
  – Focus on assets does \textit{not} improve threat modeling

• Attacker-centric threat modeling
  – Lists and personas \textit{don’t provide} enough structure to address threats

• Software-based approach
  – \textit{The best bet}
    – Having teams explain their designs at a whiteboard is surprisingly effective
Graphical Aids to Threat Analysis

- Data Flow Graphs (DFGs)
  - See following slides
- Unified Modeling Language
- Swim Lane Diagrams
- State Diagrams
Modeling System Threats

• *Data flow analysis* preferable to *exploring assets or motivations of attackers.*
Modeling System Threats

• *Data flow analysis* preferable to *exploring assets or motivations of attackers*.

• Group components by *trust boundaries*
Modeling System Threats

- *Data flow analysis* preferable to *exploring assets or motivations of attackers*.
- Group components by *trust boundaries*.
- Number entries to avoid missing components.

![Diagram of system components](image)
Data Flow Diagram Issues

• What events drive the system?
• What processes are activated by these events?
• What responses will processes generate?
• What are the sources for requests & responses?
• What are the recipients to responses?
• What is the scope of each action?
• What techniques help explore possible failures?
• How do you deal with overly complex diagrams?
Information Security Attributes
The CIA Triad

• **Confidentiality**: Access to information is limited to those with proper authorization.

• **Integrity**: Maintaining the consistency, accuracy and trustworthiness of data during its life cycle.

• **Availability**: Reliable access is maintained to resources by authorized parties.
STRIDE Threats*

- S – Spoofing
- T – Tampering
- R – Repudiation
- I – Information Disclosure
- D – Denial of Access or Service
- E – Elevation of Privilege

* Microsoft’s mnemonic for types of software threats
STRIDE Explained

• S – pretending to be another person or thing
• T – modifying something one should not
• R – falsely claiming not to have taken an action
• I – exposing information to those unauthorized
• D – denying users access to a service
• E – acquiring access at an elevated level
# STRIDE Elaborated

<table>
<thead>
<tr>
<th>Threats</th>
<th>Objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spoofing</td>
<td>Authenticity</td>
</tr>
<tr>
<td>Tampering</td>
<td>Integrity</td>
</tr>
<tr>
<td>Repudiation</td>
<td>Non-Repudiation</td>
</tr>
<tr>
<td>Information Disclosure</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>Denial of Service</td>
<td>Availability</td>
</tr>
<tr>
<td>Elevation of Privilege</td>
<td>Proper Authorization</td>
</tr>
</tbody>
</table>
Spoofing Examples

• Replacing a process/file with a corrupt one
• Changing a link surreptitiously
• Replacing a source IP address in packets
• Responding to a DNS resolution request before legitimate resolver can do so
• Faking an email address - phishing
• Pretending to be a repair person in phone call
Tampering Examples

- **Changing file contents**, e.g. replacing XML schemas or Javascript on file servers
- **Modifying memory**, e.g. passing data by reference when crossing a trust boundary
- **Tampering with a network**, e.g. new Software Designed Networks (SDNs) are programmable, giving new opportunities to tamper with wireless protocols.
Repudiation Threats

• These threats often appear at business layer
• Failure to use logs, retain them, or analyze them makes it difficult to deal with repudiation threats
• Beware attackers who fill logs

• Note: Delivery by email of a message is not evidence that it has been read!
  – Is delivery by USPS considered legal notification?
Information Disclosure

• Can **lose information** from processes, memory or data in movement

• **Process data leaks** can provide ID/passwords or design details, such as ASLR parameters

• **Data store leaks** can provide cryptographic keys as well as critical information via file names

• **Data flow leaks** may result from failure to encrypt long distance communications or just through side-channel attacks.
Denial of Service

• Rogue process can **consume CPU cycles**

• **Distributed denial of service (DDoS)** on Internet
  – See Low Orbit Ion Cannon

• If hashing algorithm for web server known, may be able to **hash many requests to one linked list, soaking up cycles.**
  – Can you think of other algorithmic DoS attacks?
Elevation of Privilege Threats

• Can occur if **control can be seized** when a process invokes another with higher privilege.
  – E.g. Subroutine invocation requires OS intervention. If buffer overflow occurs in subrtn and overwrites the return address, instead of resuming process, a new command is executed with OS privileges!

• Can result from **errors in authorization**
  – E.g. directory access is granted only to a group of users but directory files are accessible to everyone
Options to Address Threats

• Mitigate a threat
  – Increase the work to exploit it

• Eliminate a threat
  – Usually requires elimination of features

• Transfer of a threat
  – Let some other system element cope with it

• Accept a threat
  – Risk acceptance may be less costly than other steps
Revue of Basic Approach

• A flow graph that captures data movement helps to identify trust boundaries

• Use *libraries* of threats, e.g. CAPCEC or OWASP Top 10, or *mnemonics*, e.g. STRIDE or CIA, to find threats

• Mitigate, eliminate, transfer or accept threats

• Validate – Assess your work