Better Than Thinking Like a Hacker
Threat Modeling - A Practical Way to Develop Secure Applications

Michael Solomon
June 2016
Wouldn’t it be great to find security issues before writing any code?

Security issues (problems/bugs) are hard to find. Intuitive approaches are limited (some in serious ways). Threat Modeling is effective, but difficult to do.
Agenda

- Ways to find security issues
- How to threat model
- Strategies and methods
This talk is based on the book “Threat Modeling: Designing for Security” by Adam Shostack, and the course I taught (ISOL 536) for the University of the Cumberlands.
Ways to find security issues

1. Static code analysis
2. Fuzzing / dynamic testing
3. Pen testing / Red team
4. Just wait … (customers will report bugs)

List progresses from early to late in the development process. Late identification results in higher cost (impact) to fix.
Or ... Threat modeling!
Results of a live poll of attendees of the talk at PUG Challenge Americas 2016.
Why use threat modeling?

- Early security integration
- Clearer understanding of requirements
- Avoid defects in the first place
How to threat model
Think like an attacker? A popular approach.

But, how can you know what they know? Or, what they will do?

What if you get it wrong?
Similar to “cook like a professional chef”. You can’t really think like every chef – they’re all different!

Chefs meet the needs of their specific kitchens (and environments.) It is FAR BETTER to know how to be a chef, and then meet the needs of your situation.

To take this further, what would a state-sponsored attacker do to get at your data? How about a hacktivist?

Thinking like at attacker isn’t good enough – you’ll miss important things.
It's hard to judge what an attacker will want: maybe they want a random computer to serve pirated music. What do you protect? Everything inside the perimeter, because every computer is a stepping stone to the inner-sanctum. So what do you learn by making an asset list?
You’re more likely to understand the thing you’re building than any other possible starting point.
Threat Modeling outline

- What are you building?
- What can go wrong?
- What are you going to do about it?
- Check your work
What are you building?

- Create a system model
  - Abstracts away the details
- Diagrams are key
  - Mathematical models are rare in industry
- Primary focus in threat modeling
  - Data flows
  - Threat boundaries
- Common diagram types
  - Data Flow Diagrams (DFD)
  - Swim Lanes
  - State machines
Developed in the early 70s, and still useful
   Simple: easy to learn, sketch
   Threats often follow data
Abstracts programs into:
   Processes: your code
   Data stores: files, databases, shared memory
   Data flows: connect processes to other elements
   External entities: everything but your code & data. Includes people & cloud software
   Trust boundaries now made explicit
Show two or more entities communicating, each “in a lane”
Useful for network communication
Lanes have implicit boundaries between them
Helpful for considering what changes security state
For example, unauthenticated to authenticated
User to root/admin
Rarely shows boundaries
Fun to brainstorm
Mnemonics, trees or libraries of threats can all help structure thinking
Structure helps get you towards completeness and predictability
STRIDE is a mnemonic
  Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege
STRIDE

- S – Spoofing
- T – Tampering
- R – Repudiation
- I – Information disclosure
- D – Denial of Service
- E – Elevation of privilege
Spoofing is pretending to be something or someone you’re not, such as paypal.com, the C library, a Nigerian Prince with money to send you, or a Stormtrooper.
Tampering is modifying something you’re not authorized to modify, such as files on disk, packets on a network, data in memory, or the controls to a tractor beam.

http://pinlac.com/LegoDSTractorBeam.html
Repudiation is claiming that you’re not responsible for something, or lying to avoid responsibility. That might be saying that you didn’t eat the last cookie, you didn’t click buy it now, you never saw the email, or that you’ve had a reactor meltdown, and it’s very dangerous.
Information disclosure is about being able to read information that you’re not authorized to read. That might be files on disk, packets on a network, or the secret plans to a battle station.

Instructors might prefer the next (hidden) slide; this one gives you a chance to stress how central information disclosure is to the movie.
Information Disclosure
Denial of Service

Not having access to a resource, such as CPU, disk, bandwidth, or Han Solo

Alternately http://lego.wikia.com/wiki/Han_Solo
Doing things beyond your authorization, including running code as root, changing the files on a web server, getting a program to run your code, or telling stormtroopers that these aren’t the droids they’re looking for
<table>
<thead>
<tr>
<th>Threat</th>
<th>Property Violated</th>
<th>Definition</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spoofing</td>
<td>Authentication</td>
<td>Impersonating something or someone else.</td>
<td>Pretending to be any of Bill Gates, Paypal.com or ntdll.dll</td>
</tr>
<tr>
<td>Tampering</td>
<td>Integrity</td>
<td>Modifying data or code</td>
<td>Modifying a DLL on disk or DVD, or a packet as it traverses the network</td>
</tr>
<tr>
<td>Repudiation</td>
<td>Non-repudiation</td>
<td>Claiming to have not performed an action.</td>
<td>“I didn’t send that email,” “I didn’t modify that file,” “I certainly didn’t visit that web site, dear!”</td>
</tr>
<tr>
<td>Information Disclosure</td>
<td>Confidentiality</td>
<td>Exposing information to someone not authorized to see it</td>
<td>Allowing someone to read the Windows source code; publishing a list of customers to a web site.</td>
</tr>
<tr>
<td>Denial of Service</td>
<td>Availability</td>
<td>Deny or degrade service to users</td>
<td>Crashing Windows or a web site, sending a packet and absorbing seconds of CPU time, or routing packets into a black hole.</td>
</tr>
<tr>
<td>Elevation of Privilege</td>
<td>Authorization</td>
<td>Gain capabilities without proper authorization</td>
<td>Allowing a remote internet user to run commands is the classic example, but going from a limited user to admin is also EoP.</td>
</tr>
</tbody>
</table>
Using STRIDE

- How can each STRIDE threat can impact each part of your model
  - How could an attacker tamper with this part of the system?
- Make it easier
  - Elevation of Privilege Game
    - https://www.thegamecrafter.com/games/elevation-of-privilege
  - Attack Trees
  - Experience
- Track issues as you find them
  - Track assumptions too
What are you going to do about it?
Threats and assumptions

- For each threat
  - Fix – remove functionality
  - Mitigate
  - Accept – Be careful about accepting customer risk
  - Transfer – License agreements, TOS
- For each assumption
  - Check
  - Reconsider wrong assumptions
## Ways to mitigate threats

<table>
<thead>
<tr>
<th>Threat</th>
<th>Mitigation Technology</th>
<th>Developer Example</th>
<th>Sysadmin Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spoofing</td>
<td>Authentication</td>
<td>Digital signatures, Active directory, LDAP</td>
<td>Passwords, crypto tunnels</td>
</tr>
<tr>
<td>Tampering</td>
<td>Integrity, permissions</td>
<td>Digital signatures</td>
<td>ACLs/permissions, crypto tunnels</td>
</tr>
<tr>
<td>Repudiation</td>
<td>Fraud prevention, logging, signatures</td>
<td>Customer history risk management</td>
<td>Logging</td>
</tr>
<tr>
<td>Information disclosure</td>
<td>Permissions, encryption</td>
<td>Permissions (local), PGP, SSL</td>
<td>Crypto tunnels</td>
</tr>
<tr>
<td>Denial of service</td>
<td>Availability</td>
<td>Elastic cloud design</td>
<td>Load balancers, more capacity</td>
</tr>
<tr>
<td>Elevation of privilege</td>
<td>Authorization, isolation</td>
<td>Roles, privileges, input validation for purpose, (fuzzing*)</td>
<td>Sandboxes, firewalls</td>
</tr>
</tbody>
</table>
Quality assurance
Check that you covered all the threats & assumptions
Check that each is covered well
What are you building?

What can go wrong?

What are you going to do about it?

Check your work
www.solomonconsulting.com