Introduction to (STRIDE) Threat Modeling
Part 1

@KevinWWall <kevin.w.wall@gmail.com>

Agenda – Part 1

- Course Format
- What is Threat Modeling?
- Benefits of Threat Modeling
- Threat Modeling Process Overview
- STRIDE Introduction

Course Format

- 3 parts to the instruction
  - Part 1: Introduction of concepts; lecture and interactive discussion
  - Part 2: More in-depth theory and hands-on examples
  - Part 3: Break down into small, application-specific teams, to work on an application your team is familiar with
- Each part will last approximately 2 hours, with a 5-10 minute break about halfway through
- Likely will space them out 2 weeks apart

What is Threat Modeling?

- First, some questions about terminology:
  - What is a “Threat”?
  - What is “Modeling” (in the IT sense)?
- Main goal:
  - Identify assets in the system we wish to protect.
  - Identify means of protecting these assets.

Threat Modeling 101: Basic Survival Skills

- While today's threat modeling is a bit more formal, making it a bit more repeatable, humanity has been doing it since the earliest days.
- And so have you!
  - It is intuitive.
  - What does Threat Modeling help us answer?
  - We might call it "*assessing personal safety*".

Modern Day Scenario

- You got stuck at work so are running 15 minutes late to meet some friends at a local downtown restaurant.
- After parking on the street, you look around and at your GPS and see if you walk through the streets, it will take you at least another 10 minutes to arrive, but you can get there in about 2 minutes if you are willing to cut through a mostly dark alley.

- What do you do and how to you arrive at your decisions?

Personal Safety Assessment – Informal Threat Modeling

- Discuss the kinds of things that run through your head when you make a decision of choosing the street versus the alleyway?

Let's brainstorm!

My “answers” - not a complete list

- How’s the lighting on the street vs alleyway? (Time of day plays a factor.)
- Length of alleyway? (Can someone hear me scream? Short enough to run?)
- How many storefronts do I pass on the sidewalk?
- Any observable obstructions in the alley? (Dumpsters, doorways, stairs, etc. Weather conditions like snow or ice. Likely cell reception?)
- Is this section of town considered high crime area?
- Do I have anything that could be used as weapon if attacked?
- Fear of rats or other critters in the alley?

Some Benefits of Threat Modeling

- Helps to reveal implicit assumptions, especially areas of implicit trust
- Identifies and helps reduce attack surface
- Helps to identify and eliminate single points-of-failure (e.g., where there is only a single mitigating security control protecting an asset; i.e., lacking in defense-in-depth)
- Helps to prioritize threats and mitigation efforts
- Assists in developing security test plan
- Produces artifacts that may be useful in addressing compliance and regulatory issues

Threat Modeling Stakeholders

- Development team(s) – Ideally, we'd like them to *own* the TM.
- InfoSec / AppSec teams – First as trainers, then mentors, then just as Subject Matter Experts (SMEs).
- System Architecture – As needed.
- Technology SME(s) – As needed.
- Project Management – Because this *will* impact the schedule and they *love* meetings!

4 Basic Questions: Threat Modeling Manifesto

1) What are we working on?
2) What can go wrong?
3) What are we going to do about it?
4) Did we do a good enough job?

4Qs - Q1) What are we working on?

This is a question about scope.

- Initially pick *somewhere* to start. E.g., new features in next sprint.
- Scope may depend on you or your management's risk tolerance:
  - Is this the first threat model for the product?
  - How much has changed since previous threat model? How many vulnerabilties previous identified and addressed?
  - Have you done things like SAST, DAST, SCA, pen testing, security design reviews?
- We may revisit scope and shrink or expand it, as needed, depending on how we answer the next 2 questions.
  - Consider time and budget issues as well. Be realistic.
  - Start small the first few times.

4Qs - Q2) What can go wrong?

Questions to ask that will help you drill down:

- Can I list my explicit and implicit assumptions? (E.g., what are we assuming about threat actors in terms of money, time, and knowledge in terms of their resources? What are we assuming is trusted or not? Etc.)
- Where am I potentially exposed to attack? (I.e., what does my “attack surface” look like?)
- What assets should I be trying to protect?
- Where am I most vulnerable to attack? (Think about impact in terms of CIA triad [Confidentiality, Integrity, Availability].)
- What types of attacks might be the most successful?

Aside: Explicit vs. Implicit Assumptions

- Explicit assumptions:
  - Easy to recognize. We all agree and usually are conscious of these.
  - Might be based on company standards or conventions.
  - Example: We will use X for web application server, or Y for a programming language, or Z for our DBMS.
- Implicit assumptions:
  - These are not obvious and generally not broadly shared.
  - Example: Access control will RBAC (hierarchical) vs groups; we will use linked list vs hash table for this data structure.

4Qs - Q3) What are we going to do about it?

- Decide what you're going to do about each identified threat. Answer might be:
  - Eliminate the threat by remediating a vulnerability (may be by removing the vulnerable code / feature!).
  - Add a mitigating security control to lower the risk to acceptable level.
  - Buy more cybersecurity insurance and sell all your company stock. (JK about the stock! ☺)
  - Accept the risk and do nothing (but be sure to record it anyway)!
- How do I mitigate / remediate these potential vulnerabilities?

4Qs - Q4) Did we do a good enough job?

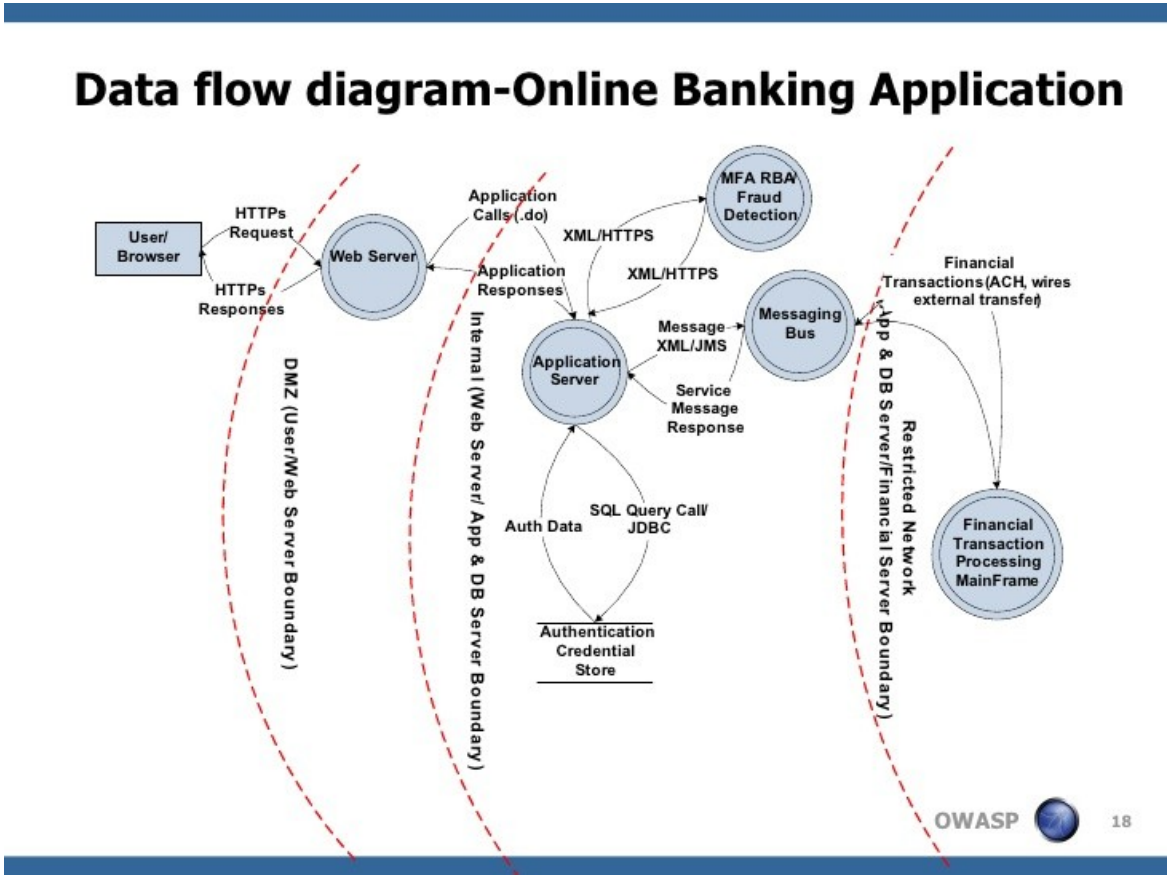
- Did you do a “good enough” job for the system at hand?
- Baseline:
  - Do you have something that you can present to the system architects and development teams that they will find usable?
  - Do you have something that you can hand to your security teams so that they can devise test plans?
  - Have you collected some metrics (e.g., staff time expected, # of threats identified, etc.) that you can provide to your upper management?

Essential Threat Modeling Steps

- Decomposing a system or feature
  - Start with your use cases and your high-level architecture or Unified Modeling Language (UML) diagrams, and create a context-level Data Flow Diagram (DFD).
- Identify potential threats
- Prioritize discovered threats
- Plan for threat mitigation / remediation

Aside: DFDs

- External Entities – rectangle
- Processes – circle
- Data stores – 2 parallel lines
- Data flows – lines with arrows
- Trust boundaries – dashed lines



Group Exercise (10-15 minutes)

- You have been hired to do a physical threat model for a new combination grocery / small department store.
  - List the assets that need protecting.
  - List and prioritize the potential threats based on perceived risk.
  - List one or more security controls to address each of the threats.
  - Have fun!
- We will focus on *intentional* threats only!

Assets to Protect

- Store merchandise
- Cash in safe
- Cash in cash registers
- Customers!
- Employees!
- Store equipment
- Building itself

Prioritized Asset Values

- Safety of the employees & customers
- Safety of the building
- Damage / theft of store equipment
- Store merchandise
  - Stock on hand:
    - Merchandise on display
    - Mechandise in backstock
  - Stock on order
- Cash in safe
- Cash in cash registers

Potential Threats (high to low)

1) Shoplifting
2) Payment fraud
3) Armed robbery
4) Arson
5) Employee theft / fraud
6) Sustained power outage
7) Human caused water damage

Security Controls

A) Insurance – (Transfer risk!) – Applies to all these risks.

B) Audible & silent alarms (theft, fire, water damage, freezer, power, HVAC, exits, police/911) – 1, 2, 4, 5, 7, 8

C) Internal & external surveillance cameras (real & fake!) – All

D) Locked entrances – 2, 4, 6

E) Deposit-only safes – 3, 4

F) RFID theft monitoring – 2, 6

G) Shopping carts with locked wheels – 2

H) Locked displays / drawers for small, high-value / highly targeted items – 2, 6

I) Employee education – 2, 3, 5, 6

Threat / Security Control Coverage Matrix

	A	B	C	D	E	F	G	H	I
1	✔	✔	✔	✔		✔	✔	✔	✔
2	✔		✔						✔
3	✔	✔	✔	✔	✔				
4	✔	✔	✔		✔				✔
5	✔		✔	✔		✔		✔	✔
6	✔	✔	✔						
7	✔	✔	✔						

Introduction to STRIDE

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

Mapping STRIDE

Threat	Security Property Violated	Threat Details / Examples	Typical DFD Artifact (Attack Surface)
Spoofing	Authentication	Pretending to be a different identity, corroborated or not (i.e., pseudo-anonymous).	Processes, external entities
Tampering	Integrity	Unauthorized modification of data	Data stores, data flows, processes
Repudiation	Non-Repudiation	Claiming that you didn't do something, or were not responsible.	Process
Information Disclosure	Confidentiality	Information is leaked to unauthorized party.	Processes, data stores, data flows
Denial of Service	Availability	Doing something to resources to disrupt service	Processes, data stores, data flows
Elevation of Privilege	Authorization	Allowing someone to do something they're not authorized to do. Generally involves multiple roles.	Process

Source: From Figure 3 in - https://learn.microsoft.com/en-us/archive/msdn-magazine/2006/november/uncover-security-design-flaws-using-the-stride-approach

Insightful Shostack quote

- *You'll sometimes hear STRIDE referred to as “STRIDE categories” or “the STRIDE taxonomy.” This framing is not helpful because STRIDE was not intended as, nor is it generally useful for, categorization. It is easy to find things that are hard to categorize with STRIDE. For example, earlier you learned about tampering with the authentication database and then spoofing. Should you record that as a tampering threat or a spoofing threat?* ***The simple answer is that it doesn't matter****.* ***If you've already come up with the attack, why bother putting it in a category? The goal of STRIDE is to help you find attacks.*** *Categorizing them might help you figure out the right defenses, or it may be a waste of effort. Trying to use STRIDE to categorize threats can be frustrating, and those efforts cause some people to dismiss STRIDE, but this is a bit like throwing out the baby with the bathwater.*

Spoofing

- Spoofing – A threat actor attempts to assume the identity of another person or entity.
  - Entity could be another process, file, host, etc.
- Common techniques:
  - User: Credential stealing / guessing
  - Process: Be the first process to open socket on expected endpoint
  - File: Trojan horse / backdoored library
  - Host: IP spoofing via DNS cache poisoning; host names via *IDN homograph* attacks
  - Other: Faked phone calls, deep fakes, etc.

Tampering

- Tampering – Threat actor attempts to maliciously alter / delete persistent data or data in use
- Common techniques:
  - SQL injection and other injection attacks
  - Altering unencrypted data in transit over network
  - Changing insufficiently protected files
  - HTTP parameter tampering through POSTs
  - Injecting fake or altered data into interprocess communication channels
  - Poisoning unauthenticated services (e.g., Redis cache)

Repudiation

- Repudiation – Claim made by attacker that they did not take some action or are not responsible for some action.
- Common techniques / causes:
  - Log injection (to cause reasonable doubt of forensics evidence)
  - Insufficient logging (e.g., failure to log unauthorized attempts, etc.)
  - Leveraging shared user accounts and credentials

Information Disclosure

- Information Disclosure – Where a threat actor is able to obtain information that they are not legitimately authorized to view
- Common techniques:
  - Exploiting weakness in file permissions or other access control
  - Looking through code (e.g., HTML comments, hard-coded secrets in code repos, etc.)
  - Leaking information via logs (URL exposure, Referer [sic] headers, etc.)
  - Exploiting misconfiguration

Denial of Service

- DoS – Attempt to disrupt a provided service by consuming or tying up available resources
- Common techniques:
  - Tie up memory (e.g., large payloads)
  - Consume lots of CPU resources (e.g., CPU intensive tasks, such as verifying digital signatures)
  - Consume / tie up network resources (e.g., SYN floods)
  - Cause loss of other fixed system resources (e.g., file descriptor leaked)

Elevation of Privilege

- Elevation of Privilege – Allow an attacker to assume an unauthorized role so that they may perform actions they are otherwise not authorized to do
- Common techniques:
  - Social engineering
  - Exploiting setuid or setgid programs
  - Exploiting process running with elevated systems privilege
  - Exploiting weakness in permissions systems (e.g., DAC in filesystems)

When you identify a threat...

- Write down the threat and your assumptions
  - Consider filing a bug report even if you decide not to address that threat
- Write down the planned mitigation ideas & maybe approximate cost
- Try to write 1 or more test cases or test plans for it. (Or abuse cases)
- Spitball its relative priority to other threats (e.g., High, Medium, Low) – *First* pass should just be a SWAG

References (1/2)

- OWASP resources:
  - OWASP Threat Modeling
  - OWASP Threat Modeling Process
  - OWASP Threat Modeling Methodology
  - OWASP Threat Modeling Cheat Sheet
  - OWASP Slack - https://owasp.org/slack/invite, #threat-modeling
- Other:
  - https://www.threatmodelingconnect.com/
  - https://www.threatmodelingmanifesto.org/

References (2/2)

- Microsoft
  - The STRIDE Threat Model:

    https://learn.microsoft.com/en-us/archive/msdn-magazine/2006/november/uncover-security-design-flaws-using-the-stride-approach
  - Other:
    https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling
- This slide deck; see
  - https://github.com/kwwall/presentations/blob/master/threat-modeling/README.md

Questions

- Ask now
- Email me at <kevin.w.wall@gmail.com>
- DM me on Twitter @KevinWWall
- But, the answers may cost you:
  - Answers: $10
  - Answers correct: $100
  - Answers requiring thought: $1000
  - Dumb looks are still free!