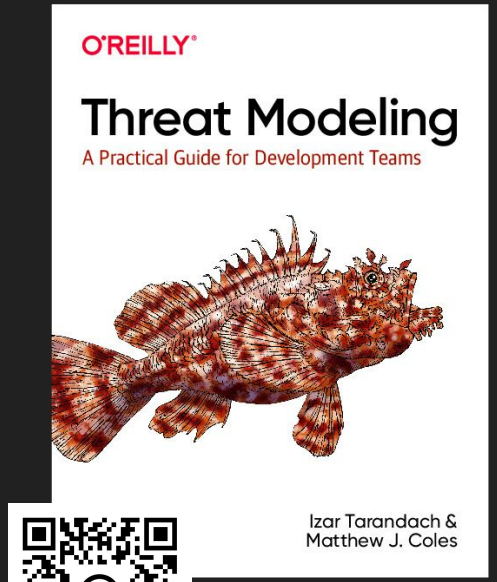
Continuous Threat Modeling

Let Developers Figure It Out

Izar Tarandach - SecAppDev 2025

Your Speaker

- Sr. Principal Security Architect at a major broadcasting company
- Doing security in a way or another since…well, a long time
- Many top and big companies, but some startups as well
- I've had lots of successes and a non-zero number of failures
- My focus is Threat Modeling, uses of AI in Security (like everyone else…) and improving security processes for developers. Big advocate of soft skills in Security
- Creator of CTM, co-author of this book →, some essays, co-author of the TMM and TMC documents, webinars and ⅓ of The Security Table podcast
- I produce walls of text. Sorry. I call them "references".



Who are you ?

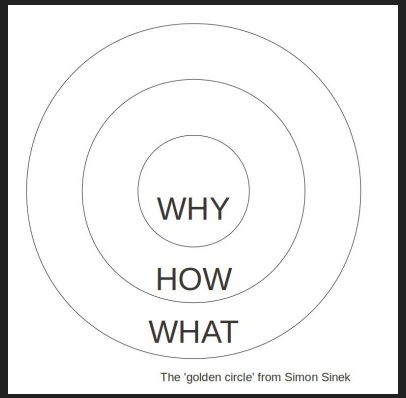
- What is your threat modeling background, if any ?

- How much do you …
  - Use threat modeling on a day to day basis ?
  - Would like to use threat modeling ?
  - Fail at threat modeling?

- Do you participate in threat modeling ?
- Do you lead threat modeling sessions ?

- What would you like to learn by the end of today ?
- What is your biggest security development lifecycle frustration ?

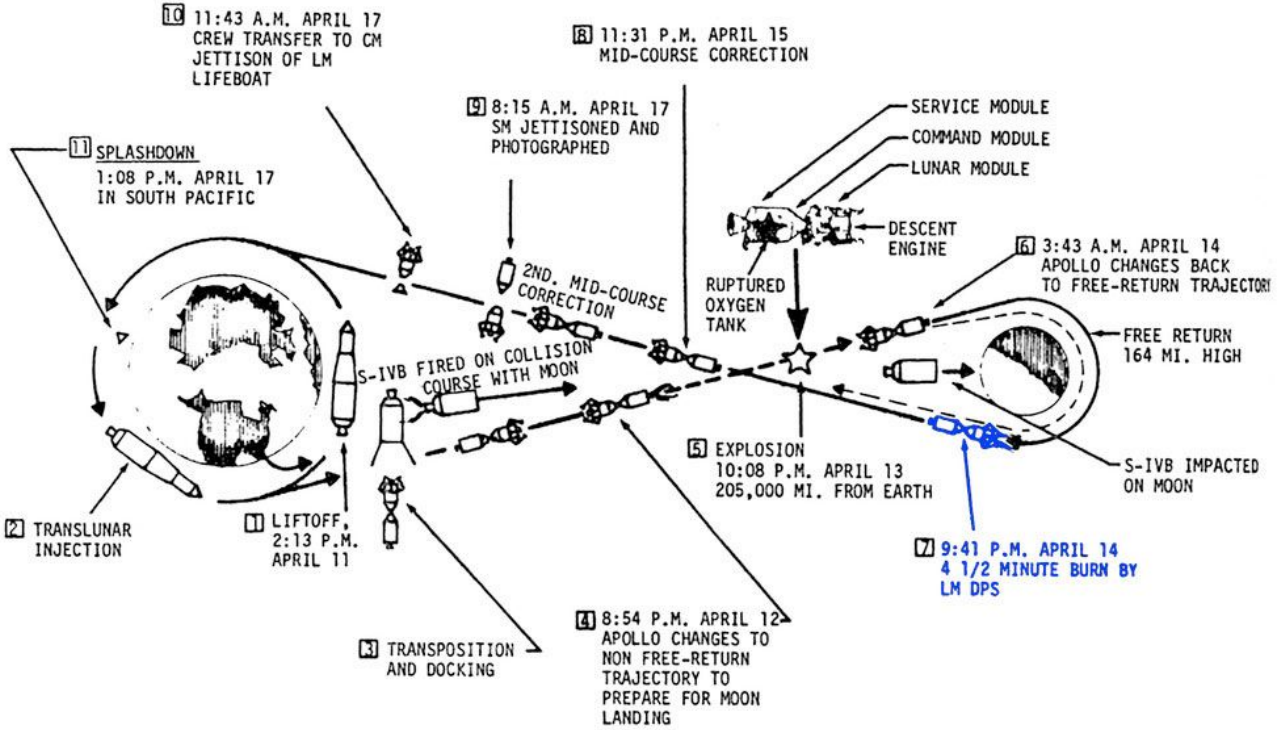
Start with “why”.



The 'golden circle' from Simon Sinek

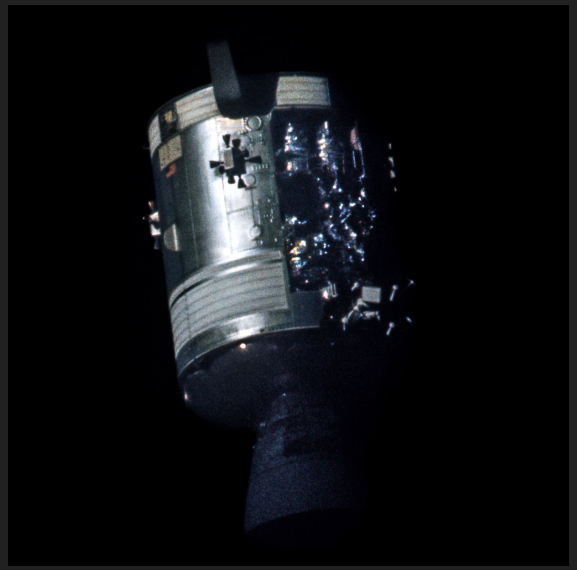
Apollo 13

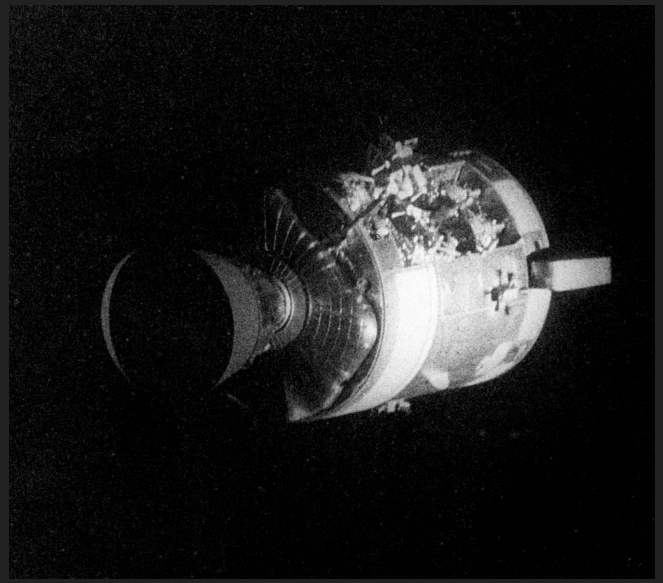
APOLLO 13 FLIGHT PROFILE

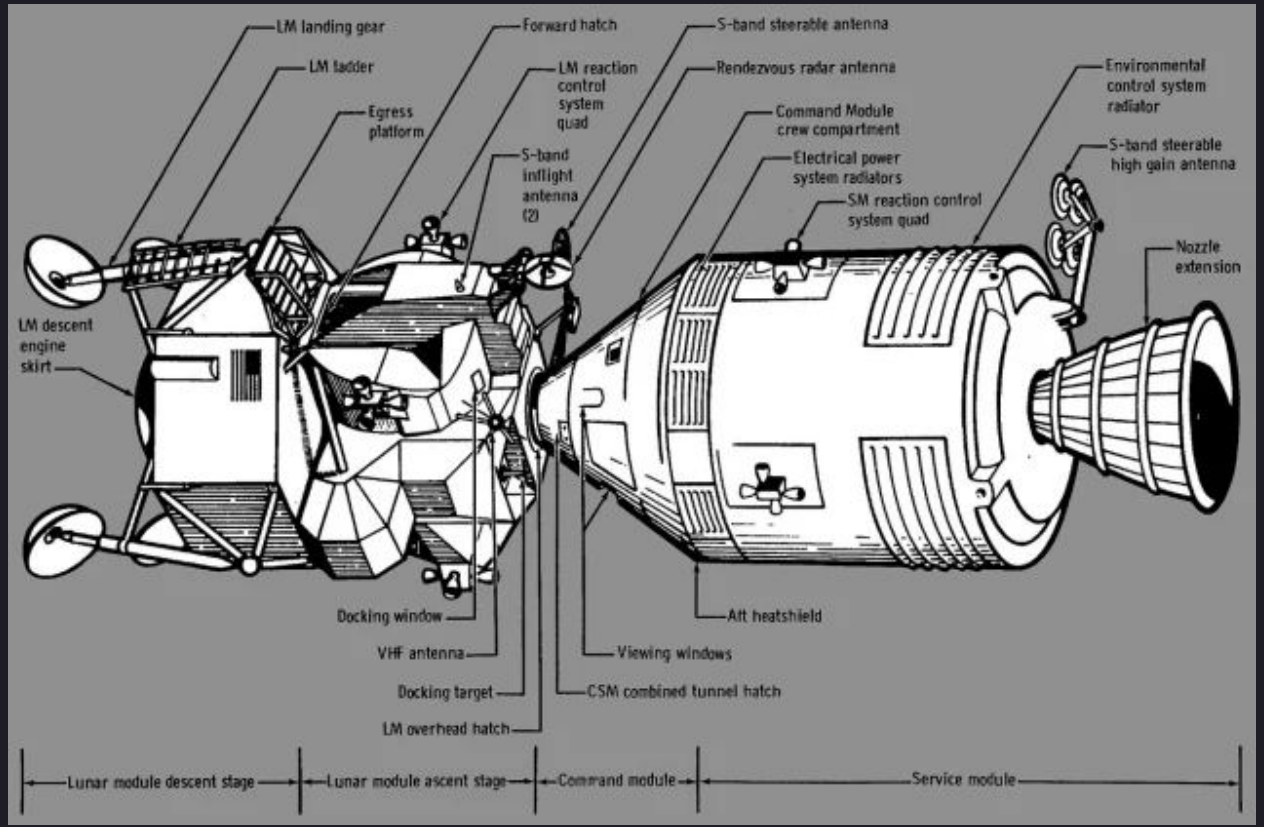


http://www.collectspace.com/review/andyanderson/apollo13_orientation01-lg.jpg

“Houston, we’ve had a problem here.”



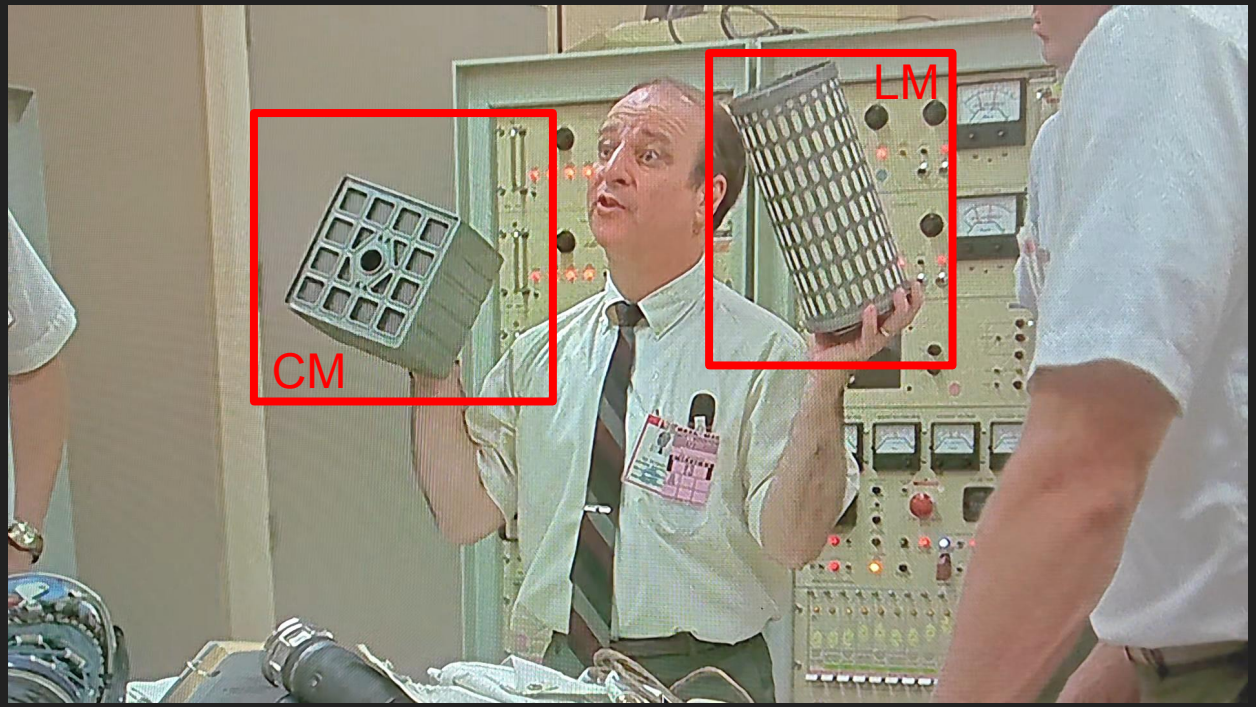




So this is the situation …

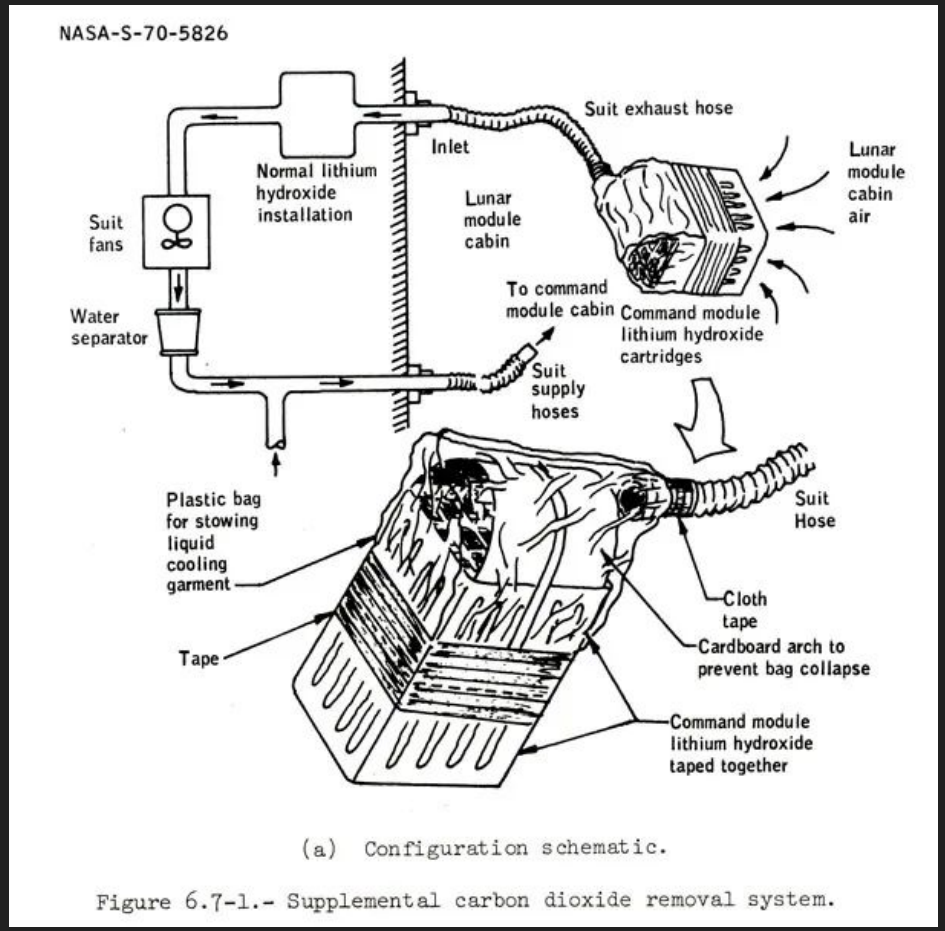
Flying back home

- New software written “on the flight” to use a different engine and nav comp
- Lots of math performed by the astronauts and checked at Mission control
- Lots of flight maneuvers made with only visual references
- About 10.000 things that needed to go absolutely right.
- The “mailbox” had to be imagined and built



“Apollo 13” - Universal Studios - 1995

The “Mailbox”



https://www.nasa.gov/wp-content/uploads/static/history/alsj/a13/ap13-S70-5826a-sm.jpg

The Point Of The Story

I am definitely not criticizing NASA design, I'm no rocket scientist

BUT

While there were contingency plans available…
… they couldn't take into consideration all the possible bad outcomes

The astronauts' life depended in large part on the ability of the ground team to know what was available, how it could be improvised, and create and transmit those instructions, with the astronauts performing flawlessly, under the stress of time and penalty of death.

The Point Of The Story - from the script of “Apollo 13”

GK - What about the scrubbers on the command module?

EE - They take square cartridges.

TM - The ones on the LM are round.

GK - Tell me this isn't a government operation.

TM - This isn't a contingency we've remotely looked at.

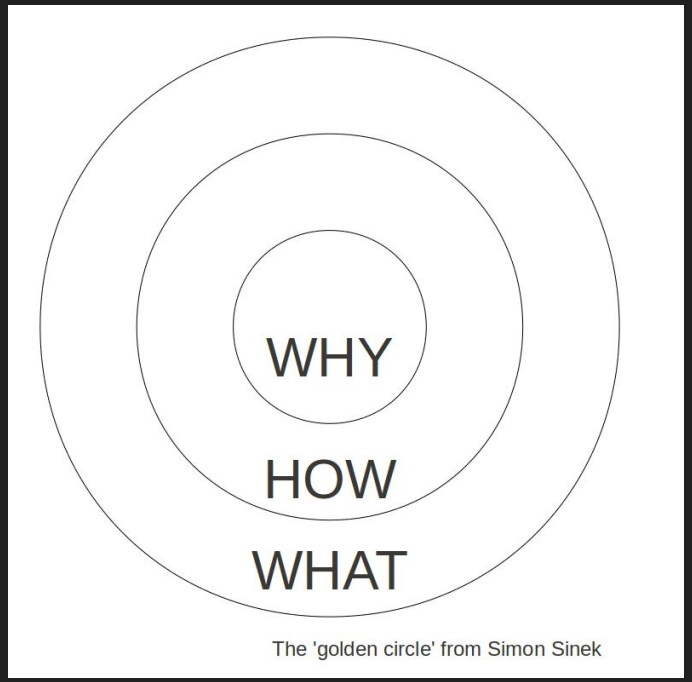
DR - Those CO2 levels are gonna be getting toxic.

Dealing with modes of failure

- A contingency plan is a strategic plan covering a possible bad outcome.
- It aims at emergencies or extraordinary situations.
- It is limited to what can be predicted, and can fail when faced with “different ways” the issue can happen.

- Solving a problem “by design” means removing the possibility of the mode of failure that can cause the problem.

…and so we get to Threat Modeling



The 'golden circle' from Simon Sinek

Threat Modeling is …

- “... analyzing representations of a system to highlight concerns about security and privacy characteristics.” - TMM
- Getting in front of problems before they even have a chance to happen
- Something we all do, all the time

What Threat Modeling is NOT ….

- A security review
- A risk assessment
- A compliance tool
- A checkbox to be filled
- Billable hours to be burned
- A chance to “call someone else’s baby ugly”

Who should threat model?

- You
- Everyone
- Anyone concerned about the privacy, safety, and security of their systems and their users

- You will hear me say many times that “threat modeling is a team sport”.
- Bring as many interested parties into the process as you can - developers, architects, QE/QA, UX, product and program managers, management, etc.
- “But that’s a huge draw on the team! Why do I need everyone?” - Eureka Moments

….which is all awesome, but what does it mean?

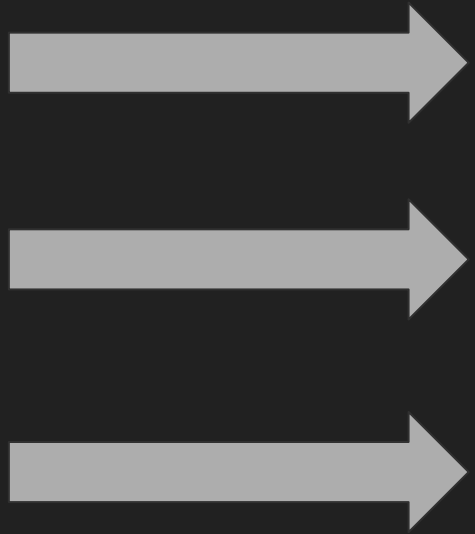
Threat Modeling is a process that intrinsically promotes those two newfangled approaches:

- Secure-by-design = well-known controls +secure-by-default (CISA)
- Secure-by-design = “We designed this thing to be secure” (everyone else)

If ALL the CO_2 scrubbers fit everywhere they are needed, then we CAN use them wherever needed whenever needed and no need for contingency plans.

4 Questions, 2 Parts, 1 artifact:

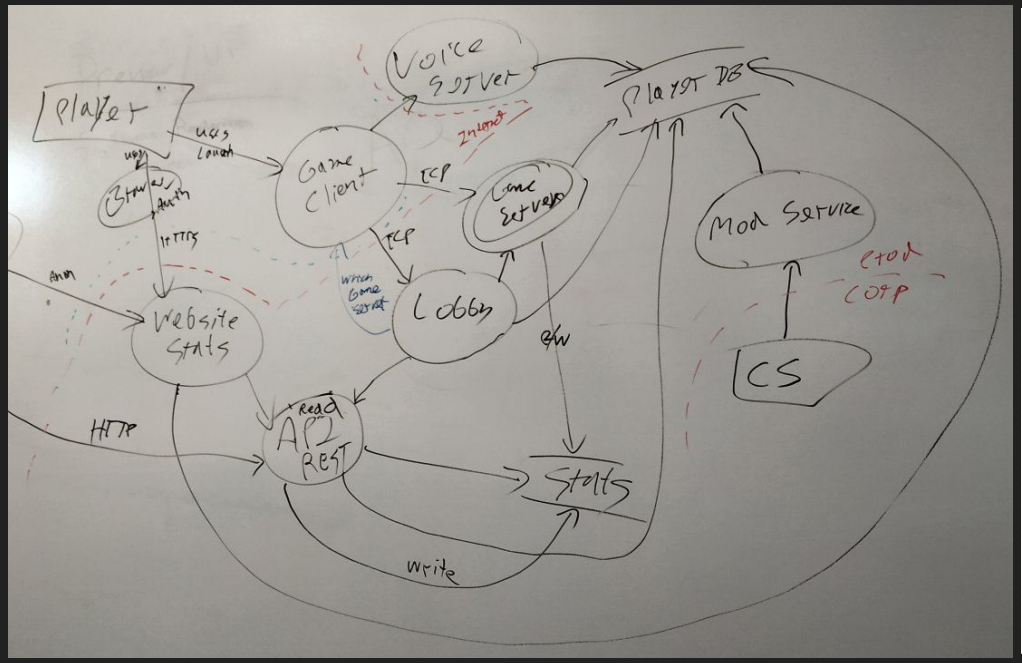
1. What are we building?
2. What could go wrong?
3. What will we do about it ?
4. Did we do a good job?

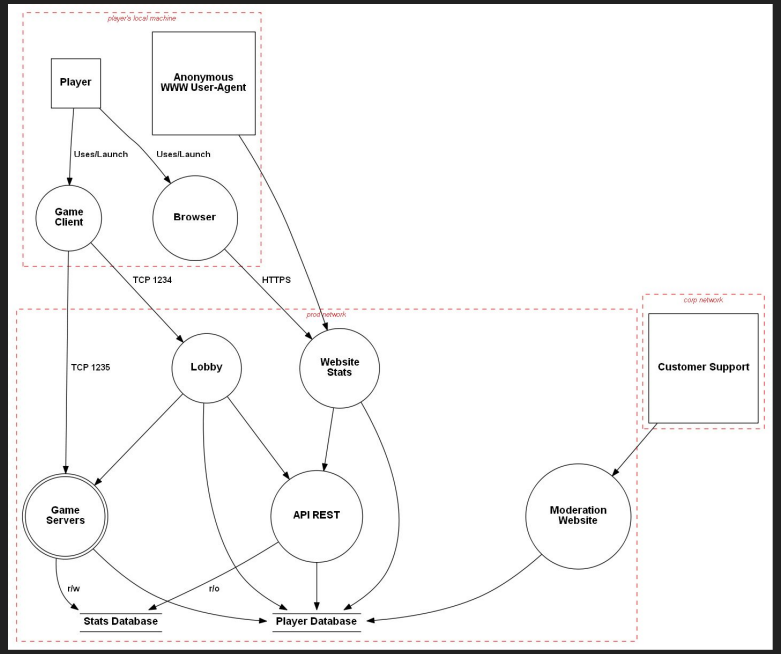


System Modeling

- Basically any way that suits the team: from formal modeling languages like SysML to a drawing on a napkin
- The important bit is that once the system is modeled, two things need to happen:
  - The participants can look back at the representation of the system, and recognize “this is what we are working on”
  - The representation is expressive enough to help elicit threats.

Dataflow Diagrams (DFDs) are a popular representation





https://github.com/OWASP/threat-model-cookbook/tree/master/Flow%20Diagram/online-battleroyale-game

Threat Elicitation

- What could go wrong?
- What could POSSIBLY go wrong?
- How BAD does it have to be?
- What if North Korea decides to break into my dog grooming business site?
- “Think like a hacker”. No. Really don’t.
- Threat libraries, security fundamentals, and everything in between

A bad example …

THREAT: something will break

MITIGATION: avoid something breaking

THREAT: secrets can leak

MITIGATION: stop secrets from leaking

A better example … THREAT/MITIGATION/PRIORITY

THREAT: too many requests too fast will cause the endpoint /abc to stop responding

MITIGATION: perform rate limitation, report on the number of requests/second, the number of successfully served and missed requests inside a configurable time period. This is a high priority issue, as the endpoint is required for the service to run.

PRIORITY: MEDIUM - P2

THREAT: the key used for encrypting data at rest is available for any service account with read rights

MITIGATION: either accept this as a design feature, or limit access to the key to only the accounts that actually need to be able to use it, in which case this is a high priority issue

PRIORITY: REALLY IMPORTANT - P0

The result of a threat model should be ….

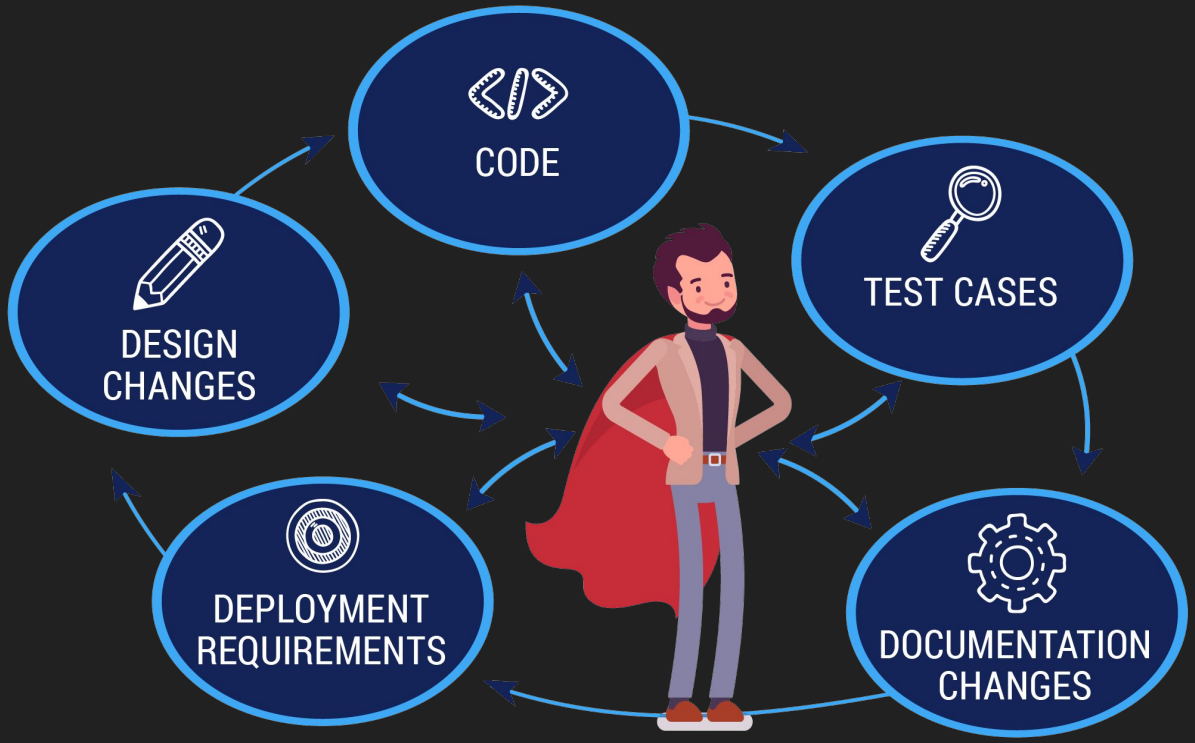
- A list of mitigations, for every flaw identified in the design that is open to exploitation by a threat actor via the existing attack surface

    - Actionable - developers, not security people
    - Shareable - among all who need to see it
    - Valuable - increments the security posture

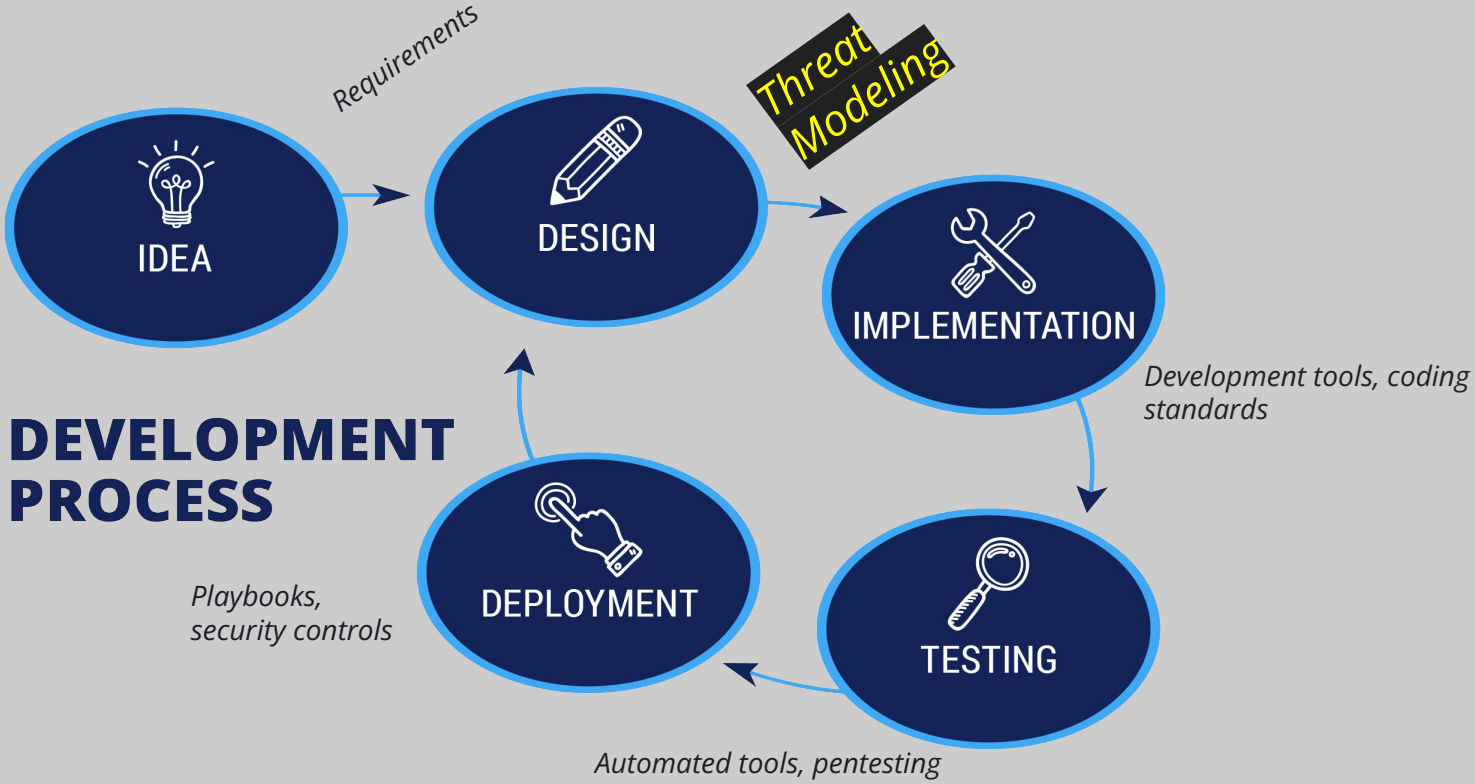
Threat Modeling is **also** good for …

- Creating a common language and view around the system we are working on
- Of course, pointing out possible design flaws in it
- But so much more!
  - Design your own training program
  - Get a security test plan done!
  - Understand development patterns and playbook them
  - Map out and enforce “security contracts”
- “Reasonable security”

The developer in today's Development Lifecycle



Threat Modeling in the Traditional SDLC



What value can Threat Modeling *actually* provide

Threat Modeling is an *expensive* process.

Design	*Secure-by-design*, eliminate whole classes of flaws, shrink attack surface
Implementation	*Secure-by-default* - secure libraries, known-good configurations, guardrails, paved road
Testing	Priorities and initial issues - what to test for and how - *the chewy bits*
Deployment	*Secure-by-default* - how the application interacts with its environment, secure services available, guardrails
Maintenance	New features tested in isolation, old code versus new threats

It is all about VALUE.

“The outcomes of threat modeling are meaningful when they are of value to stakeholders” - TMM

- There isn’t one magical methodology
  - There isn’t one perfect representation of a system
  - There are many ways to do threat elicitation
- The process takes time
  - Developers don’t like to give time to Security
  - Managers like even less
  - Training exists but is there is little absorption
- Requirements on developers:
  - Sparsely trained, but expected to provide perfect security
  - Security team is often a bottleneck

Threat Modeling Manifesto

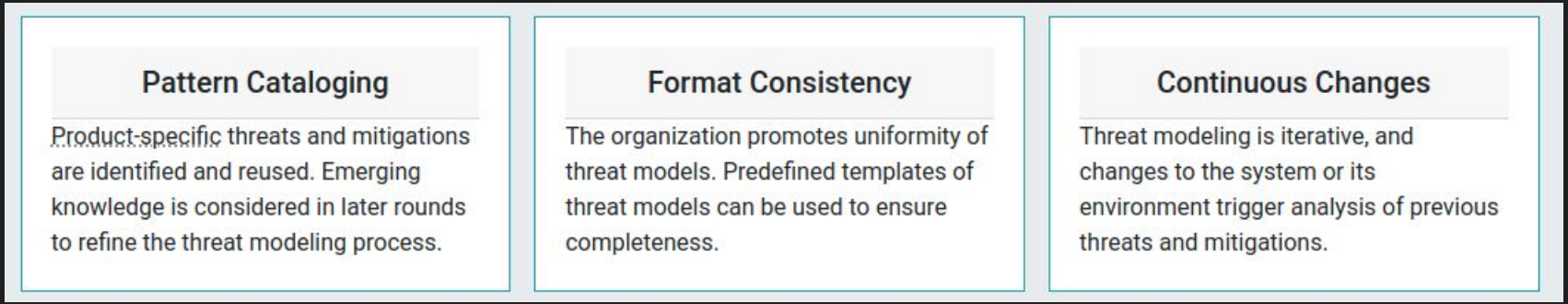
- 15 of the top voices in Threat Modeling education, research and execution sharing their experience

- 5 values:
  - A culture of finding and fixing design issues over checkbox compliance
  - People and collaboration over processes, methodologies, and tools
  - A journey of understanding over a security or privacy snapshot
  - Doing threat modeling over talking about it
  - Continuous refinement over a single delivery

- 4 principles
- 5 patterns
- 4 anti-patterns

Threat Modeling Capabilities

- Same authors, more or less
- It is a catalog of capabilities that help cultivate value from your threat modeling practice
- It is NOT a maturity document - we don't say how well you need to be doing something, only what you need to be threat modeling!
- 7 process areas



But there’s more!

- Managers love processes
- You can threat model processes as well

- In addition to developing contingencies and what-if-it-goes-wrong plans, you can apply threat modeling concepts to remove risk altogether!

Threat Modeling Your Way Into The Office

- 0600AM! Wake up
  - What if my alarm doesn't ring ?
    - Double check it the night before
    - Set two separate alarms in two separate devices
- Get Dressed
  - My clothes are not presentable
    - Wear something else
    - Use a handheld vapor press
- Commute
  - It started raining
    - I didn't bring an umbrella!
- Arrive at the office
  - Have a spare change of clothes in the office, if your region has sudden rains

(better source and material: Dr. Michael Loadenthal, Univ. of Cincinnati - "Taking Threat Modeling Offline for IRL Human Application")

Threat Modeling Fails

- Do not bring value
  - Do not lead to security posture improvement
  - “Hero Threat Modeler”
  - “Admiration for the Problem”
- Do not express threats
  - At all
  - The *right* threats
    - “Tendency to Overfocus”
- Do not represent the system
  - “Perfect Representation”
- Dead threat models
  - No alignment to culture and practices

Orgs/teams/devs don't WANT to threat model

'Pushing security practices is a bit like selling insurance; people know they need it but nobody enjoys the associated costs.'

'...by adopting the perspectives of the decision-makers, we can grasp how they're likely to perceive our plans and attempts at persuasion.'

perspective-taking!

https://leaddev.com/security/supporting-influencing-and-leading-security-practitioner

How You Win

- People will start asking unprompted questions.
- People will refer to the threat model as part of documentation
- People will start asking “how can I ask better questions ?”
- People will start asking “where can I find security training ?”
- People will start asking “where is the threat model for X ?”
- Create a repeatable process and organize a threat modeling program

Takeaways

- It’s all about understanding your “customer” - try perspective-taking
- People cannot be forced to threat model, but they can be shown that they are already doing it
- Don’t expect perfection right at the start, threat modeling is an evolutionary practice
- A “bad” threat model is better than no threat model

More takeaways

- Make threat modeling a verb - say “let’s threat model this feature” rather than “bring me a threat model of this feature”
- Model and foster curiosity
- Everyone should threat model early and often
- You don’t need to be paranoid, but it helps!

Who should threat model?

- You
- Everyone
- Anyone concerned about the privacy, safety, and security of their systems and their users

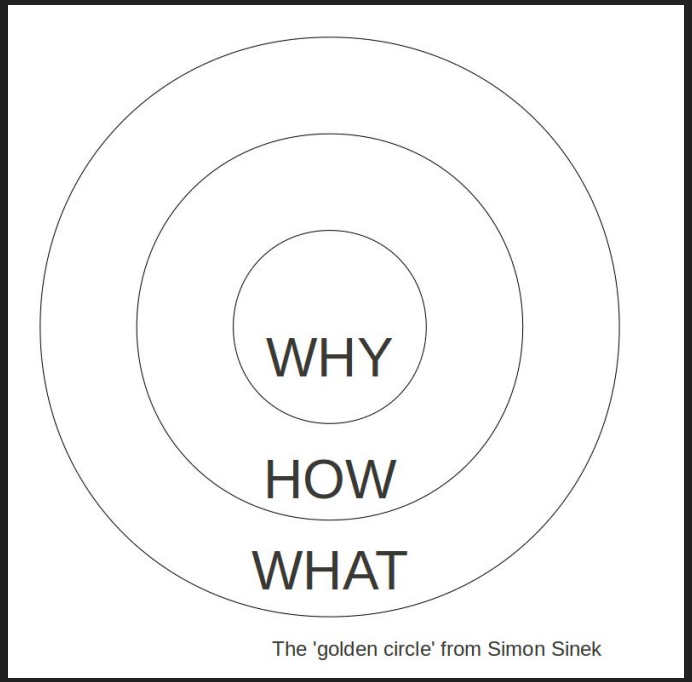
- You will hear me say many times that “threat modeling is a team sport”.
- Bring as many interested parties into the process as you can - developers, architects, QE/QA, UX, product and program managers, management, etc.
- “But that’s a huge draw on the team! Why do I need everyone?” - Eureka Moments

Methodologies

- DREAD IS NOT A THREAT MODELING METHODOLOGY.
- CVSS IS NOT A THREAT MODELING METHODOLOGY.

- STRIDE is a threat modeling methodology
- PASTA
- ATASM
- TARA
- … and CTM

Methodologies!



The 'golden circle' from Simon Sinek

The 4 Questions

It really doesn’t need to be complicated

The 4 Questions framework was authored by Adam Shostack and adopted by the TMM

- What are we building?
- What can go wrong?
- What will we do about it?
- Did we do a good job?

What are we building?

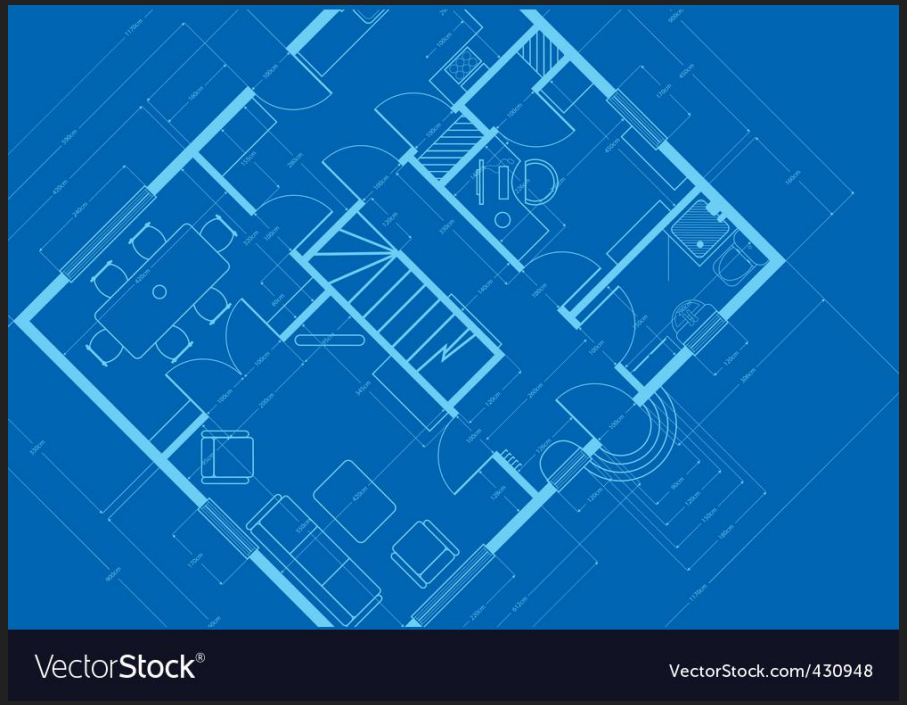
If you don’t know how to explain what you are building, well, then you have bigger problems than threat modeling.

But the real challenge here is - does the whole team agree on what we are building?

What is the scope and the resolution?

What are the “crown jewels” and where are they stored and transiting?

What are the trust boundaries?



DO IT: Given this description, please create a diagram of the system. (paper, excalidraw, anything)

RideShare is a cloud-hosted Service that matches customers (riders) with drivers (partners) in such a way that ETA, wait time, and unnecessary driving are minimized. To access the service, riders install the RideShare rider app and drivers the RideShare driver App.

The rider and driver applications send real-time location and ride data to the backend when a ride has started. For requesting a driver, the rider makes a request with the geographical cell-id, as defined by the google s2 library, the pickup point and the vehicle type to the backend. The driver app is constantly sending the geographic location and occupancy details to the backend.

The backend matches rider and driver based on geographical proximity, traffic, and route length.

What could go wrong?

What are some reasonable ways to make the system behave contrary to expectations?

Can we make it …

… reveal sensitive information?

… stop answering?

… kill itself?

… injury someone?

What will we do about it?

Now that we know what we want to build…

… and how it can fail:

What are some *reasonable* measures we can take in order to prevent the bad outcomes?

Did we do a good job?

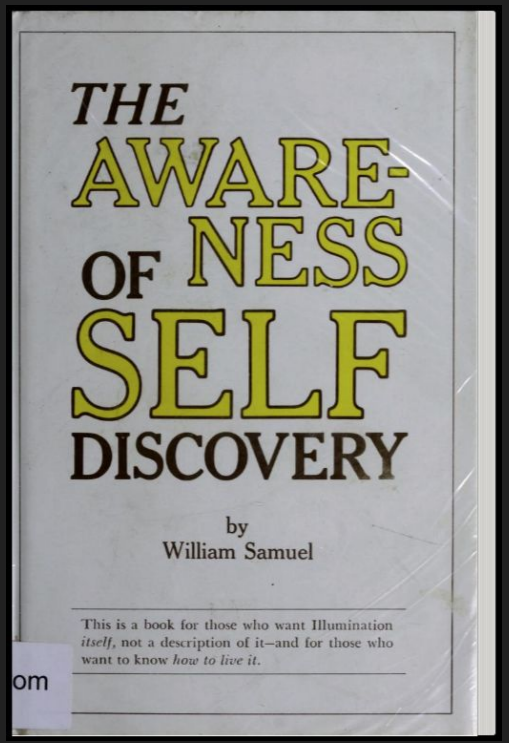
Are the results worth the work?

Did we learn something new?

Did we do better than we did last time?

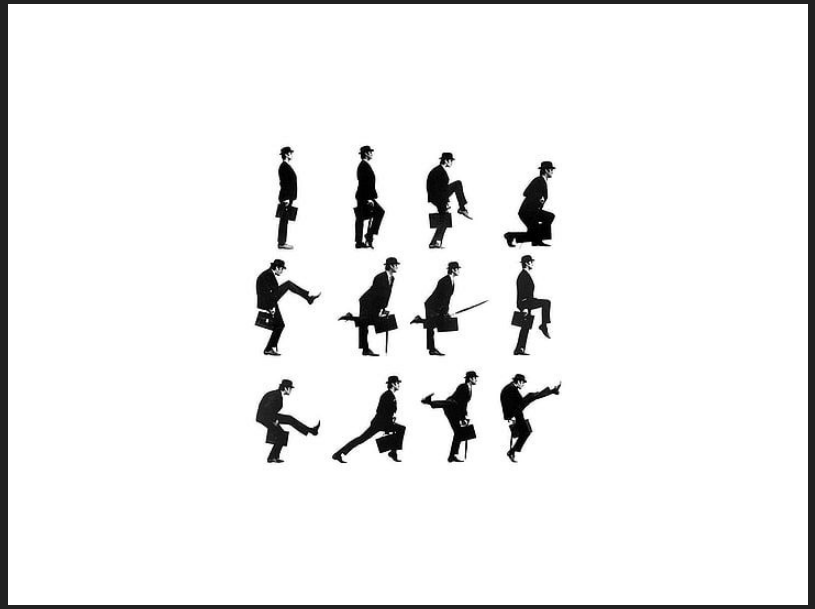
Will we be doing this again?

Did we create value for the participants?



STRIDE

STRIDE is an acronym



S - spoofing

T - tampering

R - repudiation

I - information disclosure

D - denial of servive

E - elevation of privilege

Originated in 1999 at Microsoft, authored by Loren Kohnfelder and Praerit Garg - see “*The Threats To Our Products*”

Spoofing

Email Spoofing:

One of the most common examples of spoofing is email spoofing. An attacker sends an email that appears to come from a trusted source, such as a bank, a friend, or a reputable company.

The email might tell you that there's a problem with your account and ask you to click on a link to verify your account details.

The link actually leads to a fake website set up by the attacker to collect your login credentials.

This is a form of spoofing because the attacker is pretending to be your bank in order to trick you into giving them your login details.

Tampering

One example: Data Tampering, the unauthorized modification of data.

- an attacker targeting a retail store might change the prices of certain items, giving themselves or their friends a significant discount when purchasing those items.
- attacker is able to intercept and modify the data being transmitted. In a Man-in-the-Middle (MitM) attack, the attacker changes the data being sent and the recipient has no way to sense the change

Possible mitigation: sender signs data, tamper-proof checksums, etc.

Repudiation

Online Transaction Repudiation:

A customer makes a purchase on an e-commerce website using their credit card. Once the transaction is completed and the goods are delivered, the customer denies making the purchase and claims that their credit card information was stolen and used fraudulently. They then ask their credit card company to reverse the charges.

Possible mitigation: logs with non-repudiable information, 2FA, RFID/chip card reader

Information Disclosure

A company's database, which contains sensitive customer information such as names, email addresses, and credit card numbers, is not adequately protected. An attacker exploits a SQL injection in the company's system and gains unauthorized access to the database.

Possible mitigation: use an ORM, rigorous data validation and sanitization

Denial of Service

A web interface, with an eye on preventing credential stuffing attacks, where an attacker repeatedly tries combinations of login/password pairs, establishes a policy where after 5 failed tentatives, a user is locked until an administrator manually unlocks access.

An attacker throws garbage as passwords on valid logins, and those logins are locked, denying access to the valid users.

Possible mitigation: passkeys instead of passwords, escalating timeouts rather than lock-outs.

Elevation of Privilege

An email daemon, written in C, runs in a UNIX machine.

In order to open port tcp/25 it needs super-user privileges.

The developer did not relinquish privileges after opening the port, and a string copy operation when parsing the Subject line is writing to a buffer than is shorter than it should be. The input is not checked for length.

An attacker can exploit the short buffer to overwrite the return pointer present in the stack, and direct execution to code inside the string it just sent.

Possible mitigation: use memory-safe libraries and languages, validate input before processing, drop privileges when not needed

How to STRIDE

1. Create a representation of the system being threat modeled
2. Apply STRIDE to the system or to specific elements (STRIDE-by-element)
3. Has a new threat been identified?
    a. Add to the list
4. Return to step 2 until no more threats can be identified
5. Rank the threats by criticality
6. Identify mitigations to the top X threats
7. Document the findings
8. Execute the mitigations!

Variations: STRIDE-per-element, STRIDE-per-interaction, LINDDUN

Let’s STRIDE

SecMess - Security Messaging For The Masses!

This application allows for the secure exchange of messages. It has a client, a server and an encryption module. Each will be explained in the next slide.

Using this information:

1. Create a diagram
2. Identify at least three STRIDE threat scenarios
3. Document in a table - use Critical/High/Medium/Low for severity

Element	**STRIDE**	**Scenario**	**Mitigation**	**Severity**

SecMess

Clients - user interface for registering users and logging into the system, composing and sending messages, receiving and decrypting messages. Users can create accounts in the local system and log into them.

Server - the server handles message storage and delivery. It receives encrypted messages and stores them until the recipient logs in.

Encryption module - responsible for encrypting and decrypting messages, using ROT13.

STRIDE, go! 5 minutes

In 5 minutes, share your threat table and let’s discuss the experience.

PASTA

Process for Attack Simulation and Threat Analysis

- Authored by VerSprite CEO Tony UcedaVélez and security leader Marco M. Morana in 2012
- Focuses on:
  - Risk-centric: “how much is this going to cost me”
  - Attacker-centric
  - Application context referring back to business to determine importance
  - Using existing processes for communication
  - Evidence-based threat modeling to support threat motives and leverage data
  - Focus on probability of attack, likelihood, inherent risk, impact of compromise

Cooking PASTA

1. Define business objectives
2. Define technical scope
3. Decompose the application
4. Perform threat analysis
5. Detect vulnerabilities
6. Enumerate attacks
7. Perform risk and impact analysis

https://gfkitchenadventures.com/gluten-free-one-pot-spaghetti/

1. Define business objectives

A. Define business requirements
B. Define security and compliance requirements
C. Perform a preliminary business impact analysis (BIA)
D. Define a risk profile

Output: a BIAR (Business Impact Analysis Report) - a description of the application functionality and business objectives, constrained by the requirements

2. Define Technical Scope

A. Enumerate software components
B. Identify actors and data sources - where data is created, data sinks - where it is stored away
C. Enumerate system-level services
D. Enumerate third-party infrastructures
E. Assert completeness of secure technical design

Output: lists of all assets involved, mode of deployment, and their dependencies in order to create a high-level end-to-end overview of the system

3. Decompose the application

A. Enumerate all application use cases
B. Construct data flow diagrams for identified components
C. Perform security functional analysis and use trust boundaries in the system

Output: DFDs, abuse cases

4. Perform threat analysis

A. Analyze the overall threat scenario
B. Gather threat intelligence from internal sources
C. Gather threat intelligence from external sources
D. Update threat libraries
E. Map the threat agents to assets mapping
F. Assign probabilities to identified threats

5. Detect Vulnerabilities

A. Review and correlate existing vulnerability data
B. Identify weak design patterns in the architecture
C. Map threats to vulnerabilities
D. Provide context risk analysis based on threats and vulnerabilities
E. Conduct targeted vulnerability testing

6. Enumerate attacks

A. Examine the vulnerabilities and their probabilities and decide what can become an attack
B. Update the attack library, vectors and control framework by using threat intelligence
C. Identify the attack surface of your system and enumerate the attack vectors matching previous analysis
D. Analyze identified attack scenarios by *correlating them with the threat library and attack trees*, and validate which is viable
E. Assess the probability and impact of each viable attack scenario
F. Derive a set of cases to test existing countermeasures

7. Perform risk and impact analysis

A. Mitigate the threats identified that are most probable to result in attacks
B. Apply countermeasures that are relevant to your system:
    a. Determine the risk of each threat being realized (happening)
    b. Identify the countermeasure
    c. Calculate the residual risk: are countermeasures good enough against this threat?
    d. Recommend a strategy to manage residual risk

Exercise time: perform PASTA!

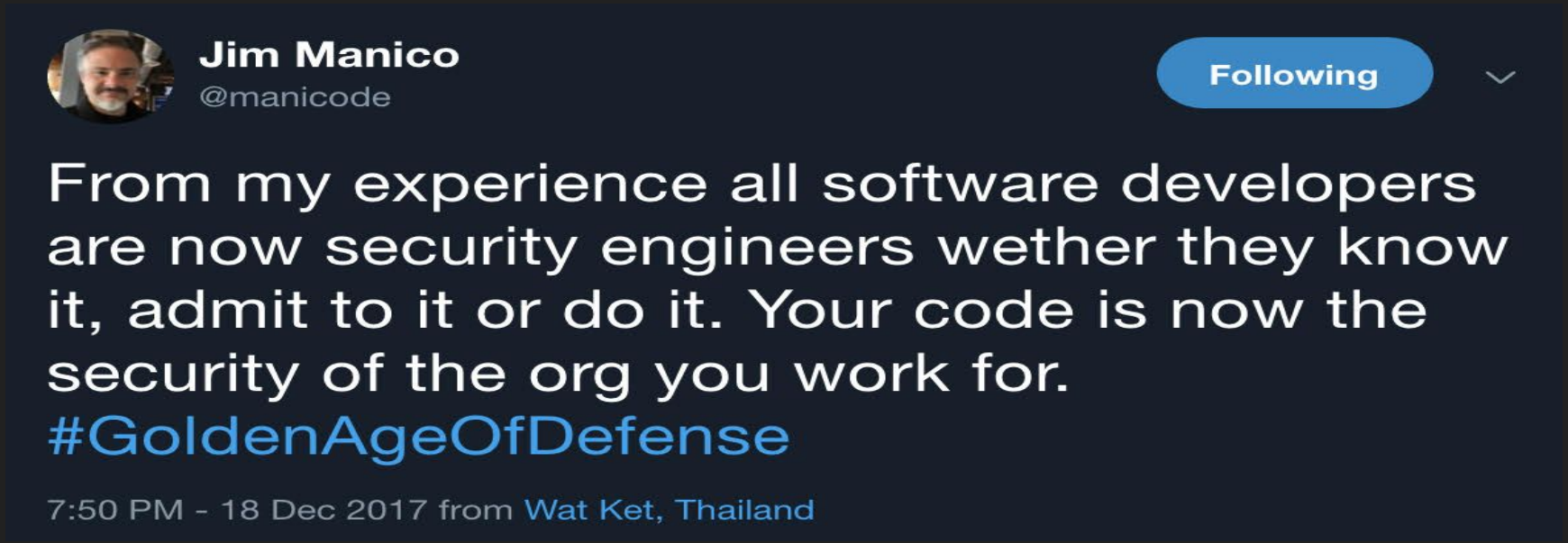
- You have 6 weeks.
- Just kidding. No PASTA for you. I am sorry I put you through that.

Continuous Threat Modeling

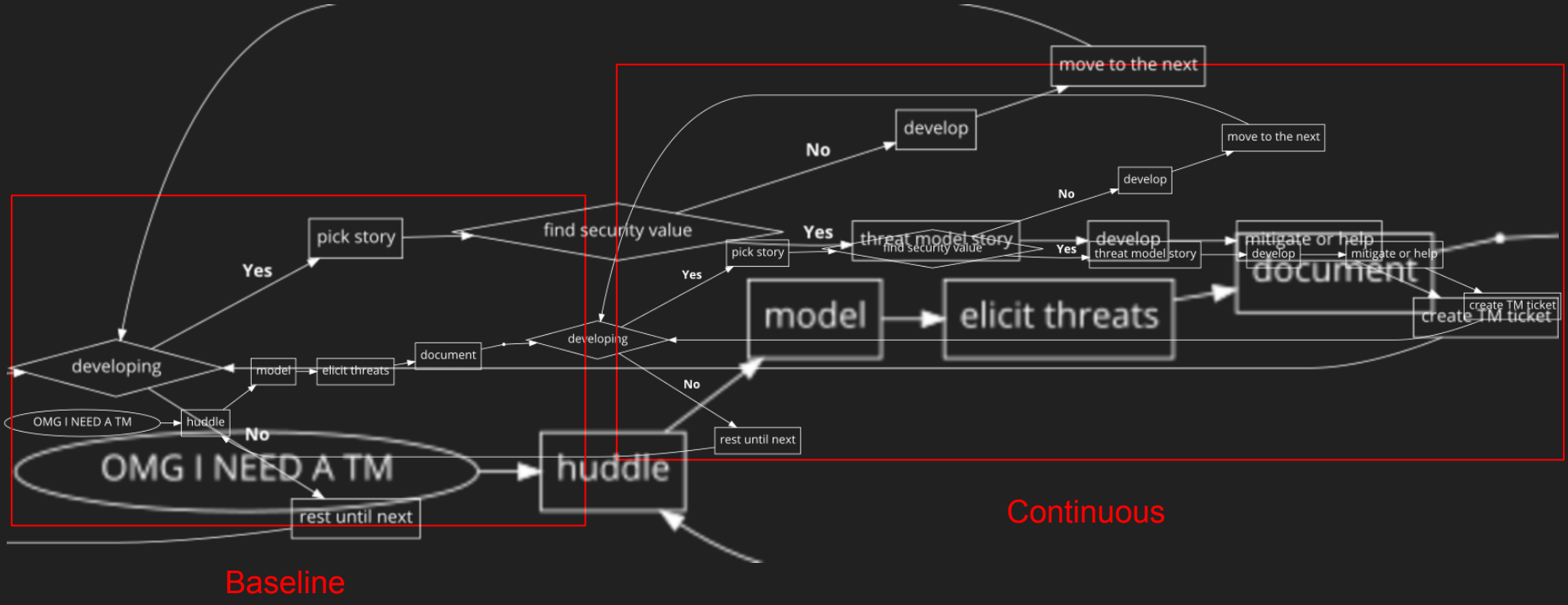
What and why

- Introduced in 2018 at Autodesk by none other than yours truly
- Developed earlier at DellEMC, where an enormous amount and variety of teams made one-size-fits-all impossible
- Focuses on the developer to build a security mindset and threat model as they develop
- *“Teach principles, not formulas”* - Richard Feynman
- *“Threat model every story”* - me

The Case For Continuous TM



Continuous Threat Modeling in a pinch



Threat Model Every Story

- build a baseline - involving everyone. Use whatever technique works for your team. CTM suggests a “subject based” list of points of interest
- designate one or more “threat model curators” who will be responsible for maintaining the canonical threat model document and the findings queue
- instruct your developers to evaluate each one of their stories with focus on security:
  - if the story has no “security value”, continue as usual
  - if the story generates a security “notable event”, either fix it (and document as a mitigated finding) or pop it up as a “threat model candidate finding” for the curator to take notice of (at Autodesk we are doing this using labels on JIRA tickets)
- make sure your curators are on top of the finding and candidate finding queues

But…how do my developers know what has “security value”?

Subject areas

Question and then continue questioning during “official design time” or when building a baseline

Checklist

Verify that the principles have been followed at implementation time

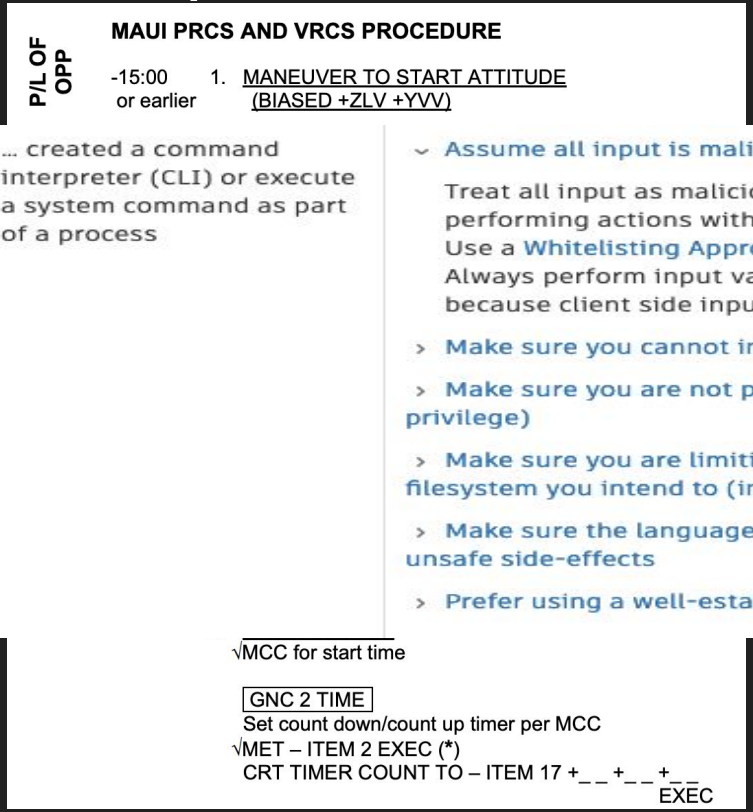
Handbook and Subject areas

- › Autodesk Threat Modeling Mission Statement

Subject	Sample questions under that subject
Authentication and Authorization	• How do users and other actors in the system, including clients and servers, authenticate each other so that there is a guarantee against impersonation? • Do all operations in the system require authorization, and are these given to only the level necessary, and no more (for example a user accessing a database has limited access to only those tables and columns they really need access to)?
Access Control	• Is access granted on a role-based fashion, are all access decisions relevant at the time access is performed? • Are all objects in the system subject to proper access control with the appropriate mechanisms (files, web pages, resources, operations on resources, etc.) ?
Trust boundaries	• Can you clearly identify where the levels of trust change in your model? • Can you map those to access control, authentication and authorization?
Auditing	• Are security-relevant operations being logged? • Are logging best practices being followed: no PII, secrets are logged. Logging to a central location, format compatible with SIEM systems. Is Cloudtrail being properly used?

- › Threat Model and Security Architecture Review

Principles Checklist



P/L OF OPP

MAUI PRCS AND VRCS PROCEDURE

-15:00 or earlier 1. MANEUVER TO START ATTITUDE (BIASED +ZLV +YVV)

… created a command interpreter (CLI) or execute a system command as part of a process

- Assume all input is malicious

  Treat all input as malicious. At a minimum, validate input and sanitize output before performing actions with it. This improves the overall security posture of your application. Use a Whitelisting Approach as opposed to a Blacklisting approach when validating input. Always perform input validation on the server side even if you are doing it on the client side because client side input can be easily bypassed.

- Make sure you cannot inject extraneous commands as arguments
- Make sure you are not providing an elevation of privilege vector to an attacker (least privilege)
- Make sure you are limiting the reach of the command to those operations and areas of the filesystem you intend to (input validation & least privilege)
- Make sure the language mechanism you are using to execute commands does not have unsafe side-effects
- Prefer using a well-established command execution library instead of creating a new one

√MCC for start time

GNC 2 TIME
Set count down/count up timer per MCC
√MET – ITEM 2 EXEC (*)
CRT TIMER COUNT TO – ITEM 17 +_ _ +_ _ +_ _
EXEC

Threat Model Every Story - again, slow

- build a baseline - involving everyone. Use whatever technique works for your team. CTM provides a “subject based” list of points of interest - they’re starting points, not a checklist!
- designate one or more “threat model curators” who will be responsible for maintaining the canonical threat model document and the findings queue
- instruct your developers to evaluate each one of their stories with focus on security:
  - if the story has no “security value”, continue as usual
  - if the story generates a security “notable event”, either fix it (and document as a mitigated finding) or pop it up as a “threat model candidate finding” for the curator to take notice of (at Autodesk we are doing this using labels on JIRA tickets)
- make sure your curators are on top of the finding and candidate finding queues

Reactions from product teams

- “Uh...what?”
- “This is still too heavy”
- “But how do I know I did everything?”
- “I never saw a room of architects excited about threat modeling before”

Caveat Emptor: This Is Not Perfect

- Difficult to convince teams that the Subject List is not a threat library and developers that the Checklist is not a requirements list – not exhaustive, just a starting point
- The resulting TM won't be perfect – evolutionary
- A SME or security group may still be necessary for education
- GIGO – garbage-in, garbage-out

CTM - Exercise

Let’s add some more features to SecMess, and threat model their stories as we go.

Refer to the baseline threat model we created back in STRIDE, or create one with the Subject List in the handbook

Go over the new Stories in the next slide and use the Developer Checklist and the Subject List to identify their security value (or lack of) and write a threat model addition ticket if necessary.

User Stories https://tinyurl.com/mrymnet5

- As a USER, I want to be able to have a central login server that manages all users in the system.
- As an ADMIN, I want to be able to change the cipher used for the cryptography of the system, at will
- As a DEVELOPER, I want to document the latest version of the API
- As a DEVELOPER, I want to be able to create new clients and interfaces at will
- As a LEO, I want to be able to escrow keys so I can intercept messages in transit
- As a QA person, I want to be able to run the whole system in a contained environment so I can test its features

Tools, tools, tools!

Let’s come up with a system to model

Write yourself the description of a system. Something simple like a calculator that stores results, or the operating system for an airliner. Now threat model it.

Can we even automate this ?

- What do YOU think?
- Some things can be automated:
  - Diagraming
  - Low hanging fruit
  - Management of output
  - Communication of findings
- Some … “can’t”
  - Really innovative new findings
  - System modeling
- Somebody say LLM. I dare you. I double dare you.

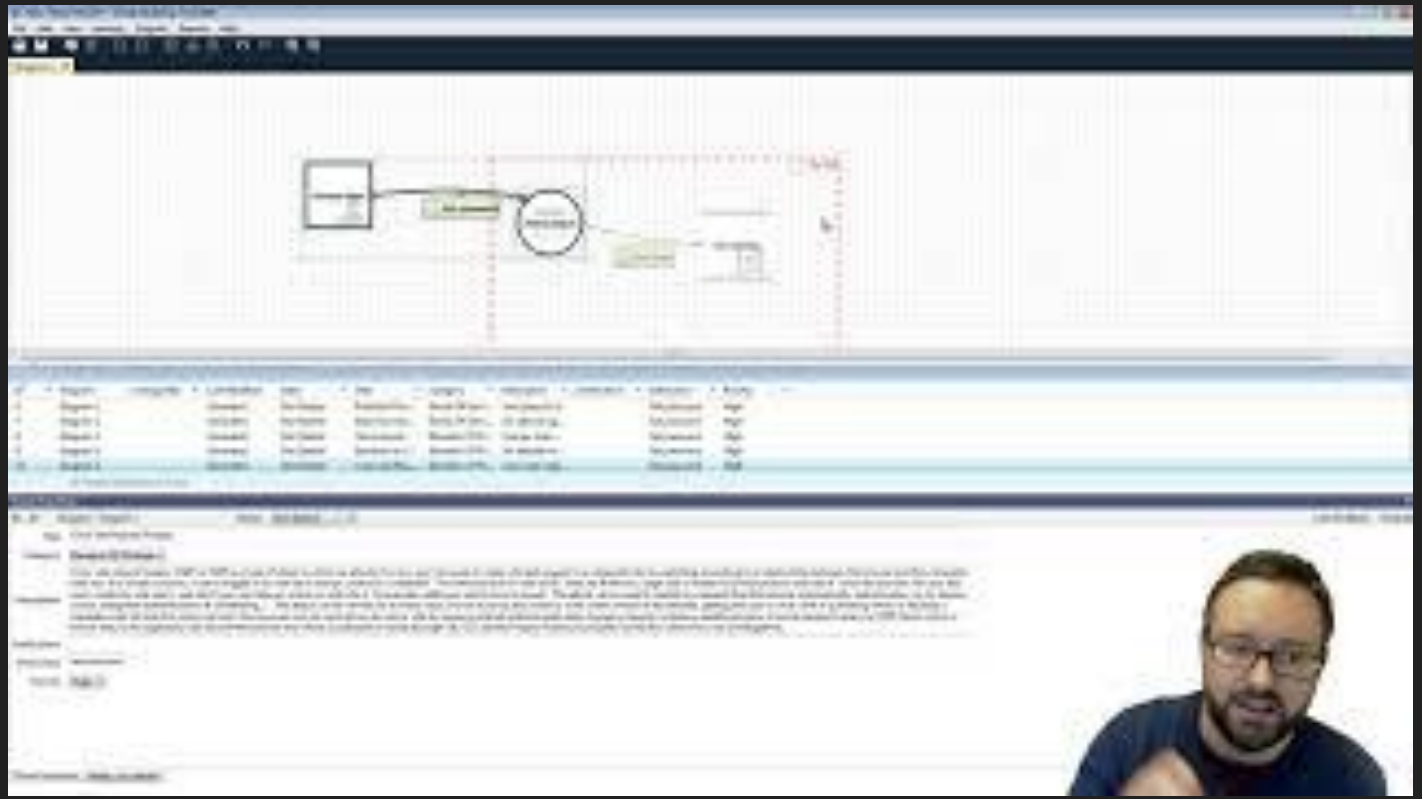
But Izar, what do you have against LLMs ?

- Nothing. They’re great tools. For what they’re good for.
- They are not good for threat modeling.
- The answer to “Please threat model this system” is not a threat model. It is a collection of regurgitated material that at some point links in context to some already known issue with an easy to identify/parse element.
- Do you really want hallucinated threats?
- And no, I am not afraid they’ll replace me any time soon.

OWASP Threat Dragon

- Open a browser and go to https://www.threatdragon.com/#/
- Click on *“Login With Local Session”*
- Click on *“Explore a sample threat model”*
- Click on *“Version 2 demo threat model”*
- Click on “*Main request data flow”*
- Click on elements to see and edit their attributes
- Click on *“+ New Threat”* and create a new threat
- Click on *“x Close”*
- Click on “Report”
- Now do yours!

Microsoft Threat Modeling Tool (TMT)



OWASP Pytm - the pythonic library for threat modeling

Schema

Element

type
name
purpose
role
exposes...
privileges
uses...
contains...
connects to...
listens for...

Rules

```
Weakness w =

(target.type == Process &&
 target.privileges == "root" &&
 len(target.exposes) > 0
)

Threat t =

(target.exposes.port == 80 &&
 source_data.is_hci()
)
```

Engine

Loader

Parser

Sequencer

Analyzer

Renderer

Calculator

Reporter

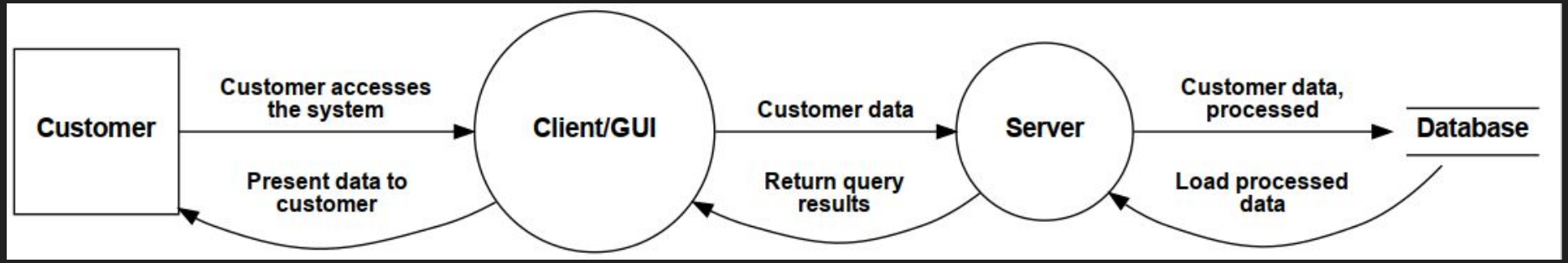
Using pytm

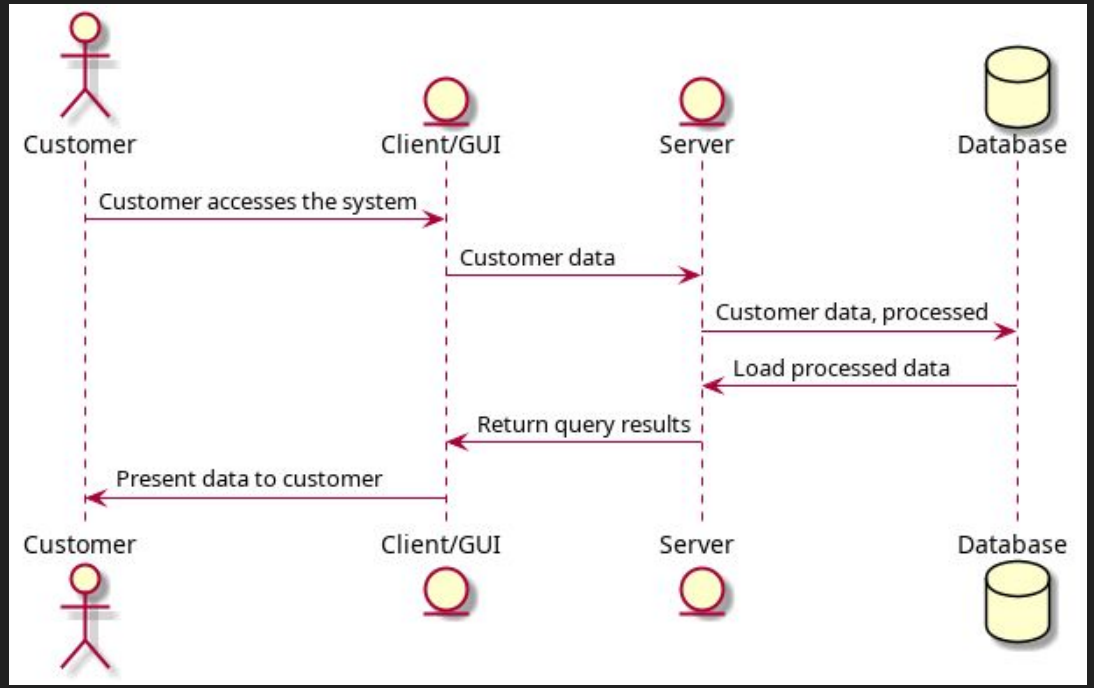
1. Define the components of the model and their relationships (dataflows)
2. Generate a dataflow diagram and/or a sequence diagram
3. Annotate the components with their attributes
4. Generate a report with the threats identified as a function of component and dataflow attributes

```
#!/usr/bin/env python3

from pytm import (
    TM, Actor, Boundary, Classification, Data,
    Dataflow, Datastore, Process, Server
)

tm = TM("TM Demo v0.0.1")

...

tm.process()
```

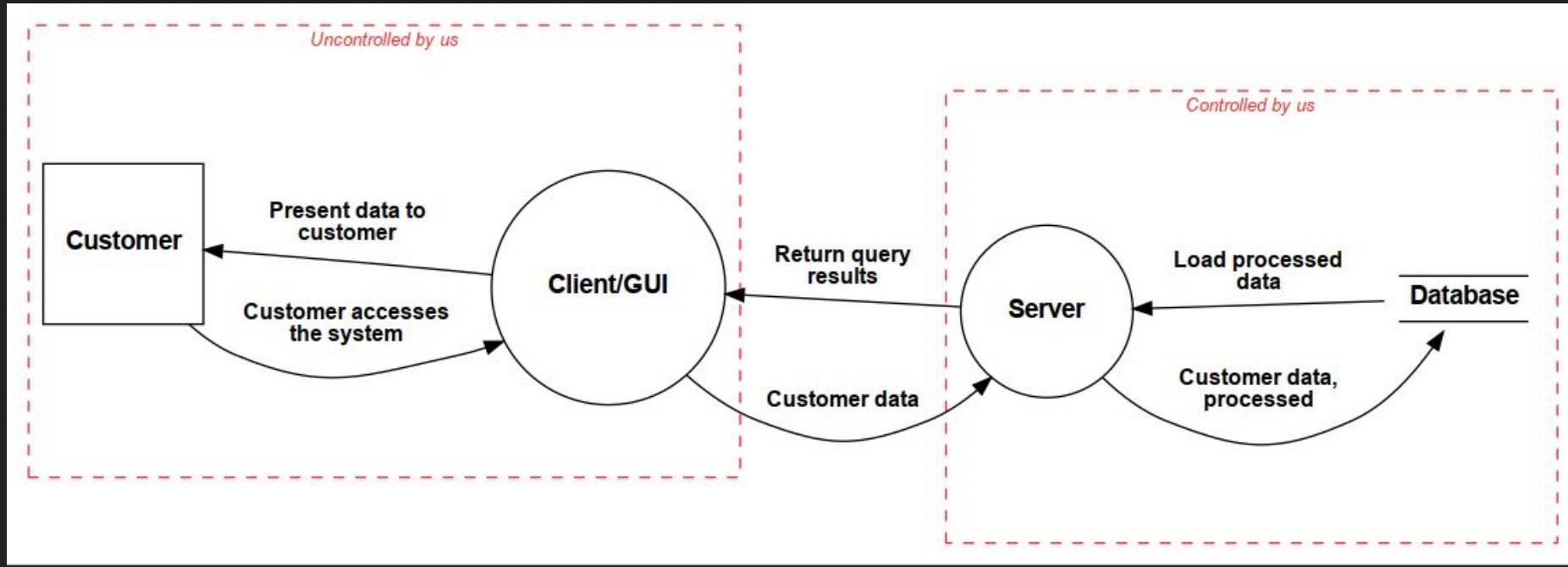
```
tm = TM("TM Demo v0.0.1")

user = Actor("Customer")

client = Process("Client/GUI")

server = Server("Server")

db = Datastore("Database")

tm.process()
```

```
db = Datastore("Database")

interact = Dataflow(user, client, "Customer accesses the system")

enterData = Dataflow(client, server, "Customer data")

saveData = Dataflow(server, db, "Customer data, processed")

loadData = Dataflow(db, server, "Load processed data")

editData = Dataflow(server, client, "Return query results")

present = Dataflow(client, user, "Present data to customer")

tm.process()
```

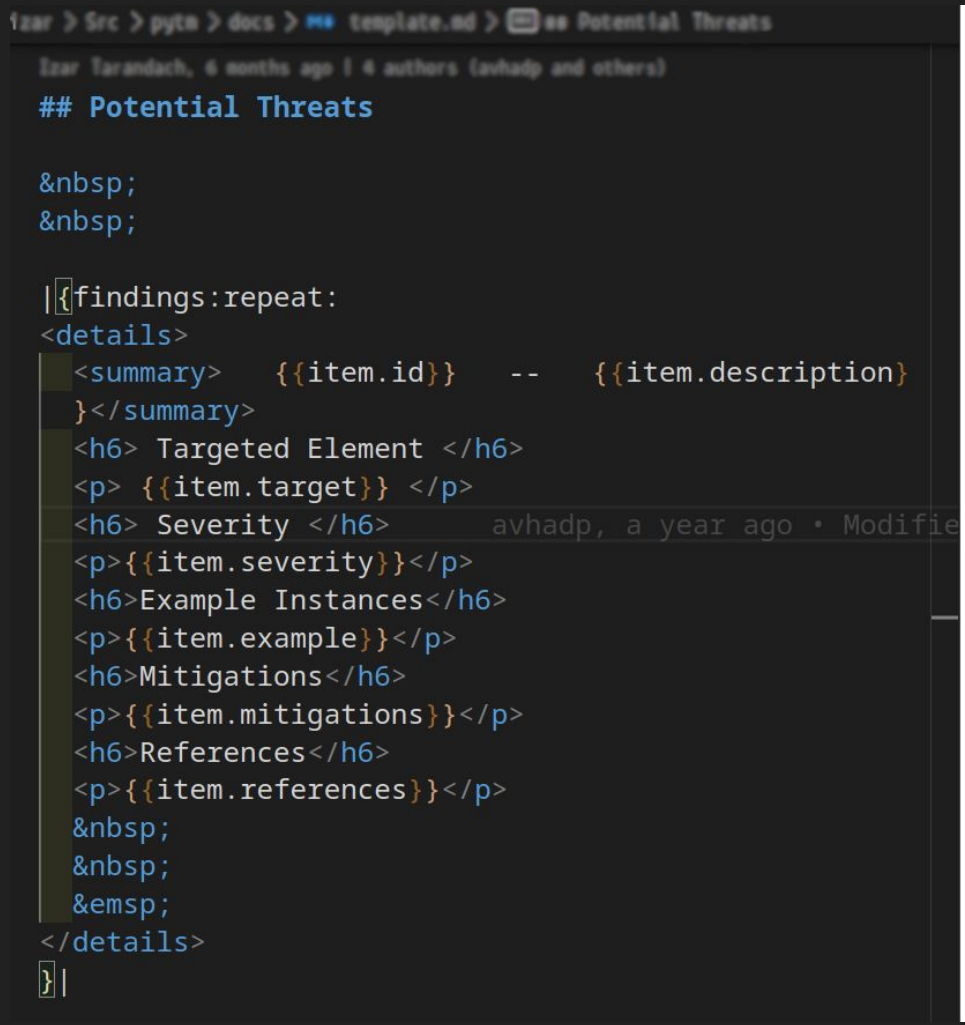

Customer
Customer accesses the system
Client/GUI
Customer data
Server
Customer data, processed
Database
Present data to customer
Return query results
Load processed data


Customer
Client/GUI
Server
Database
Customer accesses the system
Customer data
Customer data, processed
Load processed data
Return query results
Present data to customer
Customer
Client/GUI
Server
Database

```
tm = TM("TM Demo v0.0.1")

publicBoundary = Boundary("Uncontrolled by us")
protectedBoundary = Boundary("Controlled by us")

user = Actor("Customer")
user.inBoundary = publicBoundary
client = Process("Client/GUI")
client.inBoundary = publicBoundary

server = Server("Server")
server.inBoundary = protectedBoundary
db = Datastore("Database")
db.inBoundary = protectedBoundary
```


Uncontrolled by us
Controlled by us
Customer
Present data to customer
Customer accesses the system
Client/GUI
Return query results
Server
Load processed data
Database
Customer data
Customer data, processed

```
server.OS = "Ubuntu"
server.isHardened = True
server.sanitizesInput = False
server.encodesOutput = True
server.authorizesSource = False

db.OS = "CentOS"
db.isHardened = False
db.isSQL = True
db.inScope = True
db.maxClassification = Classification.RESTRICTED

enterData.protocol = "HTTP"
enterData.dstPort = 80
enterData.data = "New items to be stored, in JSON format"

saveData.protocol = "MySQL"
saveData.dstPort = 3306
saveData.data = "MySQL insert statements, all literals"

tm.process()
```



Potential Threats

|{findings:repeat:

▼ {{item.id}} -- {{item.description}}

Targeted Element

{{item.target}}

Severity

{{item.severity}}

Example Instances

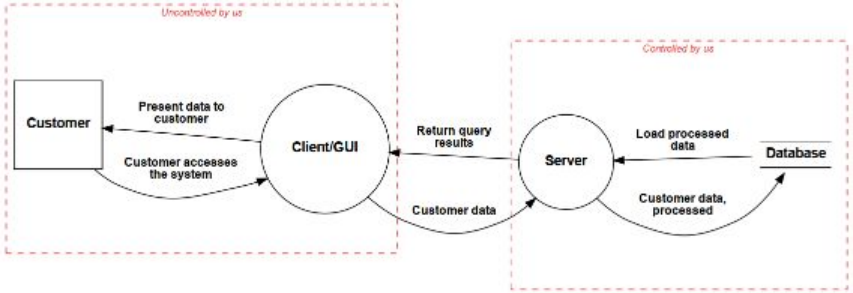
{{item.example}}

Mitigations

{{item.mitigations}}

References

Dataflow Diagram - Level 0 DFD



Dataflows

Name	From	To	Data	Protocol	Port
Customer accesses the system	Customer	Client/GUI	[]		-1
Customer data	Client/GUI	Server	New items to be stored, in JSON format	HTTP	80
Customer data, processed	Server	Database	MySQL insert statements, all literals	MySQL	3306
Load processed data	Database	Server	[]		-1
Return query results	Server	Client/GUI	[]		-1
Present data to customer	Client/GUI	Customer	[]		-1

Data Dictionary

Name	Description	Classification
New items to be stored, in JSON format		PUBLIC
MySQL insert statements, all literals		PUBLIC

Potential Threats

- INP02 – Overflow Buffers
- AA01 – Authentication Abuse/ByPass
- DE02 – Double Encoding
- AC01 – Privilege Abuse
- INP07 – Buffer Manipulation
- DO01 – Flooding
- DO02 – Excessive Allocation
- INP05 – Format String Injection
- INP12 – Client-side Injection-induced Buffer Overflow
- INP13 – Command Delimiters

▼ AC01 – Privilege Abuse

Targeted Element

Client/GUI

Severity

Medium

Example Instances

An adversary that has previously obtained unauthorized access to certain device resources, uses that access to obtain information such as location and network information.

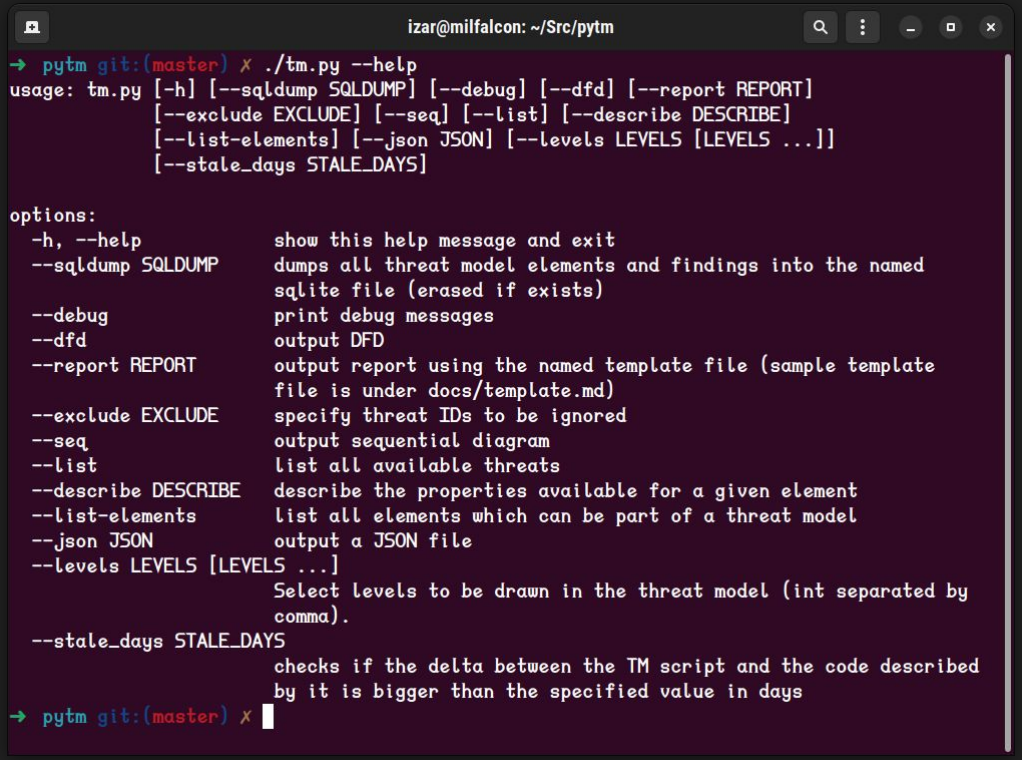
Mitigations

Use strong authentication and authorization mechanisms. A proven protocol is OAuth 2.0, which enables a third-party application to obtain limited access to an API.

References

https://capec.mitre.org/data/definitions/122.html, http://cwe.mitre.org/data/definitions/732.html, http://cwe.mitre.org/data/definitions/269.html

Pytm CLI arguments



Working with pytm

1. Checkout the code
2. Copy (or use) tm.py (the sample code) to a name of your choosing
3. Edit the copied file to reflect the system you are working on
4. Save the pytm script file in the same repo than the code it reflects
5. Create your dfd: ./yourfile.py --dfd | dot -Tpng > sample.png
6. Create your sequence file: ./yourfile.py --seq | java -Djava.awt.headless=true -jar /usr/local/lib/plantuml.jar -tpng -pipe > seq.png
7. Create your report: ./yourfile.py --report docs/basic_template.md > report.md
8. Use *pandoc* to translate your report into whichever format you'd like: pandoc -f md -t pdf report.md report.pdf

Pytm with Docker

- Clone: *git clone https://github.com/izar/pytm/*
- Build: *docker build -t pytm .*
- Run: *docker run -it -v `pwd`:/usr/src/app/ pytm*
- Work:
  - Edit tm.py
  - Run: ./tm.py –dfd | dot -Tpng > tm.png
- In a different terminal, open tm.png

Pytm hands-on

File sharing service

1. Web interface, file server, authentication service, storage and database
2. The web interface and file server use HTTP to talk to each other
3. The database and file server are not hardened
4. The authentication service is not encrypted
5. The web interface does not perform input validation

But I don’t know Python!

- No need - you may think you’re writing C++. Or Java. Javascript. Whatever helps think about `object.attribute = value`

- `Or…you can use PytmGPT!`
  `https://chatgpt.com/g/g-soISG24ix-pytmgpt`

In closing - that AI thing again

I have been playing with AI and Threat Modeling on and off:

- PytmGPT - https://chatgpt.com/g/g-soISG24ix-pytmgpt

- Vibe Threat Modeling - https://www.linkedin.com/pulse/vibe-threat-modeling-really-izar-tarandach--duwse/

Threat Model And Prosper!

Threat Modeling Manifesto - https://www.threatmodelingmanifesto.org

Threat Modeling Capabilities - https://www.threatmodelingmanifesto.org/capabilities/

“Threat Modeling: A Practical Guide For Development Teams” - https://amzn.to/3Ss8vlv

“Threat Modeling: Designing for Security”, Adam Shostack - https://amzn.to/3VLsXkq

Thank you! Questions?

Threat Modeling Manifesto - https://www.threatmodelingmanifesto.org

Threat Modeling Capabilities - https://www.threatmodelingmanifesto.org/capabilities/

“Threat Modeling: A Practical Guide For Development Teams” - https://amzn.to/3Ss8vlv

“Threat Modeling: Designing for Security”, Adam Shostack - https://amzn.to/3VLsXkq

“Building In Security At Agile Speed”, James Ransome & Brook Schoenfield - https://amzn.to/3MTDbuN

CISA “Secure By Design”, https://www.cisa.gov/sites/default/files/2023-10/SecureByDesign_1025_508c.pdf