An introduction to threat modeling

COOCK+ SECDES

13 September 2024

Koen Yskout, Prof. dr. ir. | KU Leuven Campus Diepenbeek

KU LEUVEN DistriNet

About me

koen.yskout@kuleuven.be

https://distrinet.cs.kuleuven.be/people/koeny

- Associate Professor in Computer Science
  - KU Leuven (Campus Diepenbeek), Belgium
  - Faculty of Engineering Technology (“industrieel ingenieur”)
  - DistriNet & ACRO research group
- About 20 years of research experience on
  - (automated) threat modeling
  - security by design
  - secure design patterns
  - model-driven security
  - empirical research on secure software design

KU LEUVEN

DistriNet

Warm-up

What do you know/have you heard about threat modeling?

Context: security by design

What is ‘software design’?

- Decisions about the UI/UX?
- Whiteboard sketches?
- UML models? C4 models?
- Early decisions about the structure of the software?
- Source code?

In essence: the ‘real’ software design = the source code
Designing = all activities that affect the source code
Including architecture, implementation, testing and debugging

Reeves, J. W. (2005). Code as Design: Three Essays by Jack W. *Developer.* Magazine*, 1–24.

When is software (in)secure?

1. There must be an adversary (threat)

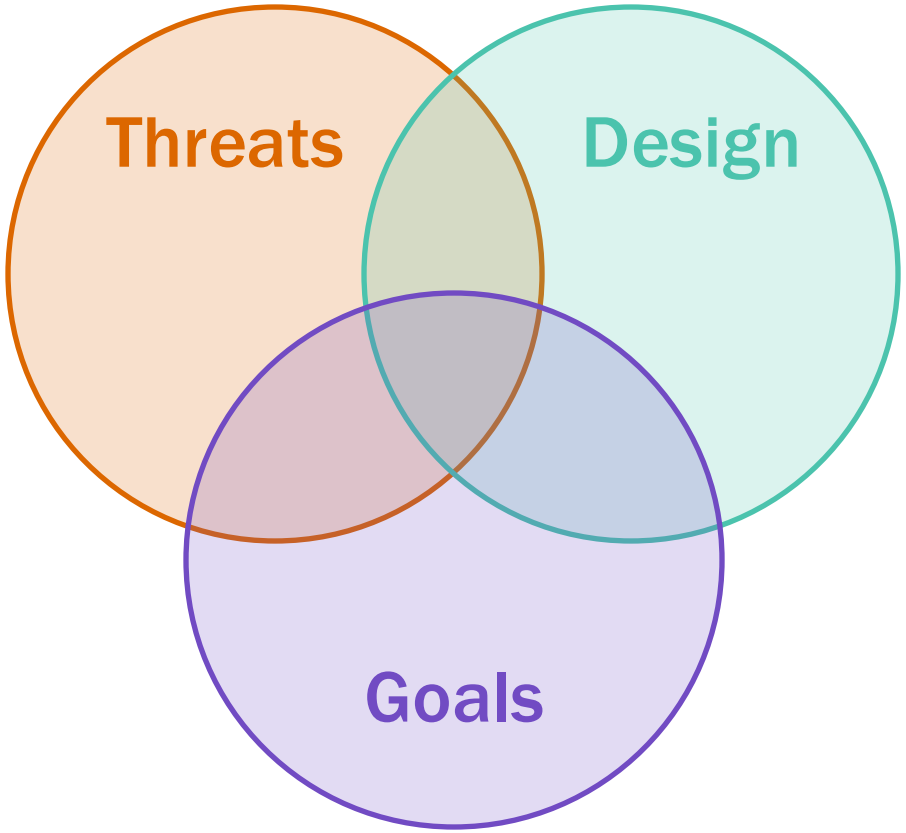
   Not under your control

2. The system must be vulnerable (design)

   You have a lot of control over this

3. The negative impact must matter (goal)

   Helps to prioritize



Being 'secure' heavily depends on context!

Turpe, S. (2017). The Trouble with Security Requirements. *2017 IEEE 25th International Requirements Engineering Conference (RE)*, 122–133. https://doi.org/10.1109/RE.2017.13

When is software (in)secure?

1. There must be an adversary (threat)

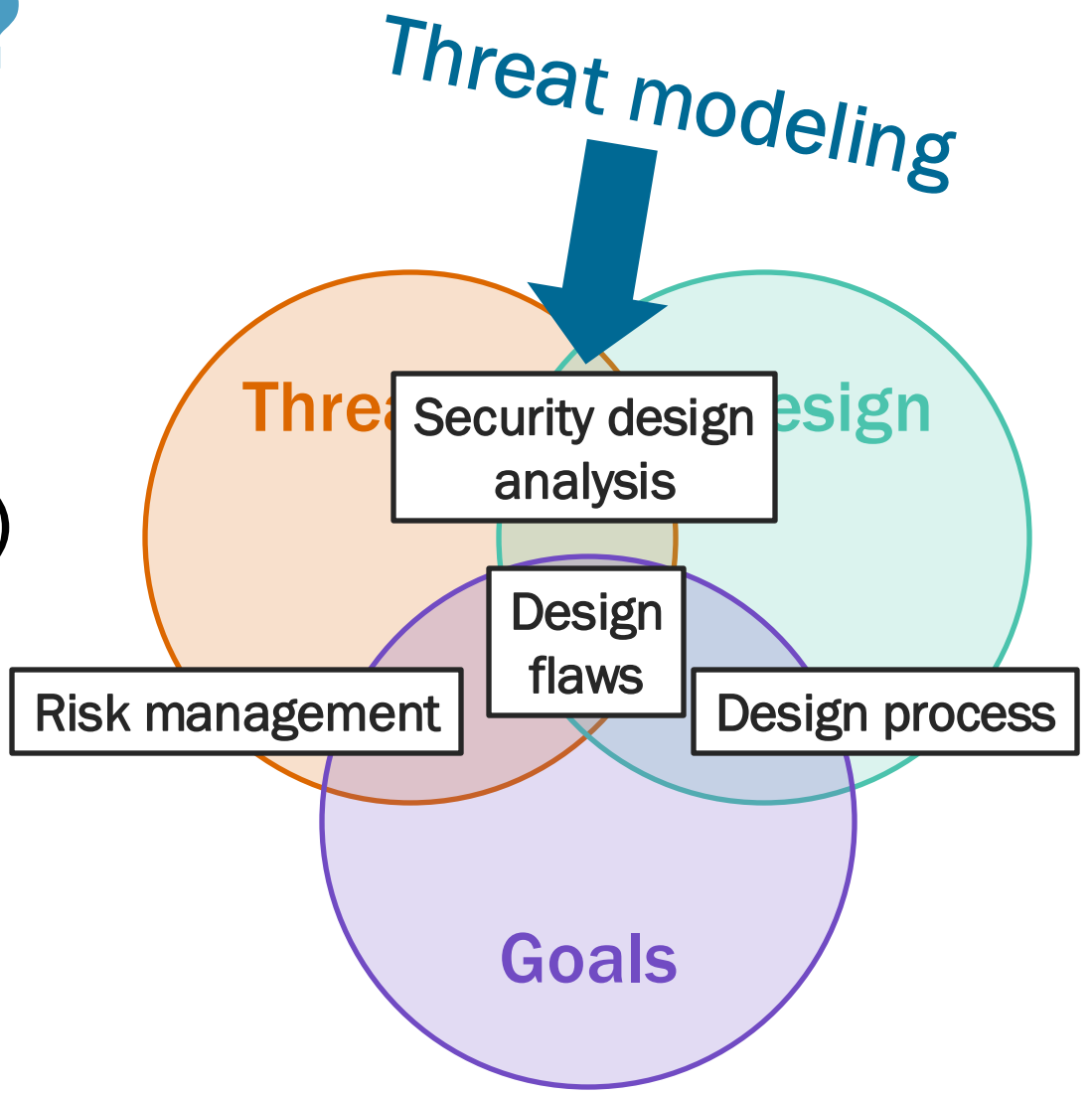
   Not under your control

2. The system must be vulnerable (design)

   You have a lot of control over this

3. The negative impact must matter (goal)

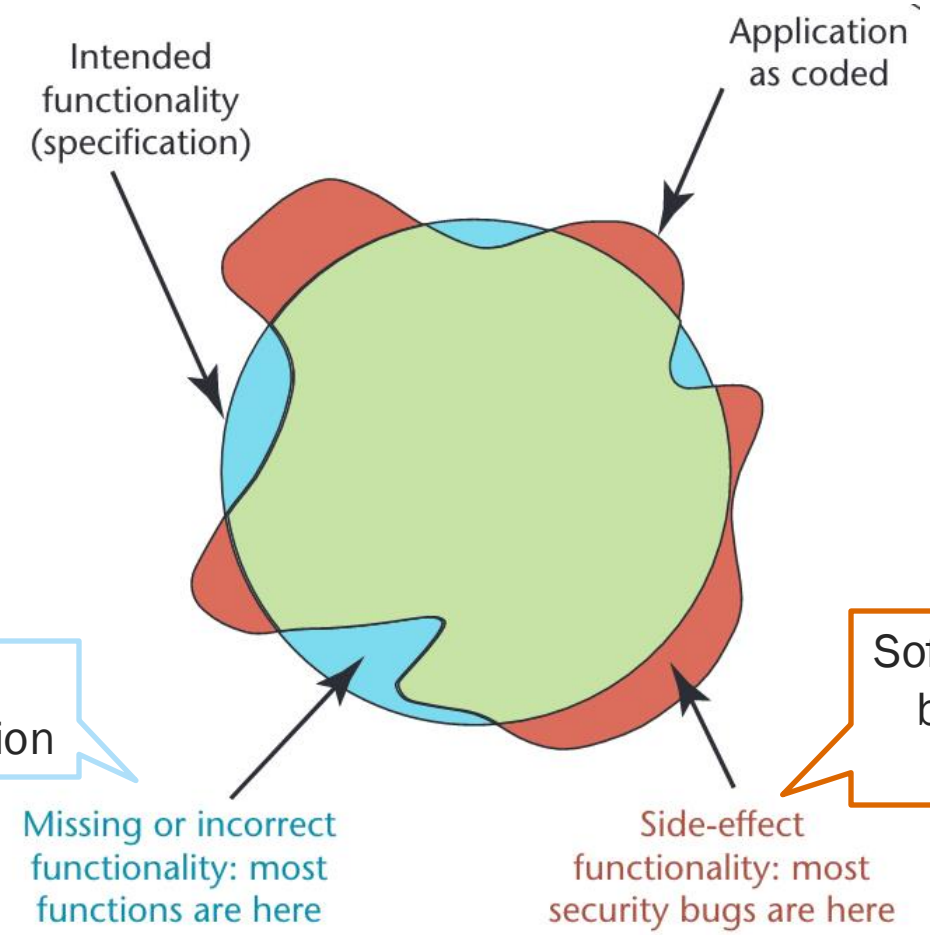
   Helps to prioritize



Being 'secure' heavily depends on context!

Turpe, S. (2017). The Trouble with Security Requirements. *2017 IEEE 25th International Requirements Engineering Conference (RE)*, 122–133. https://doi.org/10.1109/RE.2017.13

Why is security hard?



Software doesn't yield correct result under regular operation

Software mostly behaves as expected, but also causes security problems in specific situations

H. H. Thompson, "Why security testing is hard," *IEEE Security & Privacy*, vol. 1, no. 4, pp. 83–86, Jul. 2003, doi: 10.1109/MSECP.2003.1219078.

What is ‘security by design’?

- Also referred to as ‘shift left’ (DevSecOps), ‘build security in’
- Consider security during **all** software design activities
- **Not** as an ‘afterthought’ (when the design is ready)
  - i.e., not only with a pentest right before releasing/deploying
  - i.e., not only by focusing on the environment/network
  - i.e., not only after a security incident

What ‘security by design’ is not?

- The use of tools
  - Can be part, but there’s more
- Requiring “big up-front design” or extensive modeling
  - You can be agile and do security by design
- Strive for perfection
  - Secure *enough* by design
- All-or-nothing
  - You can start simple and grow towards a more extensive approach

Main objective of 'security by design'

Detect and mitigate important security flaws as soon as possible

- Avoid re-design/delays due to security
- Avoid 'security debt' (~ technical debt)

My own working definition:

processes, practices, and tools to make the security of the developed software product inevitable

Secure design activities

SDLC, SSDLC, …

- (Software Development Lifecycle: SDLC)
- Secure Development Lifecycle: SDLC
- Secure Software Development Life Cycle (SSDLC)
- Secure Development Lifecycle (SDL)
- Security by Design (SbD)
- …

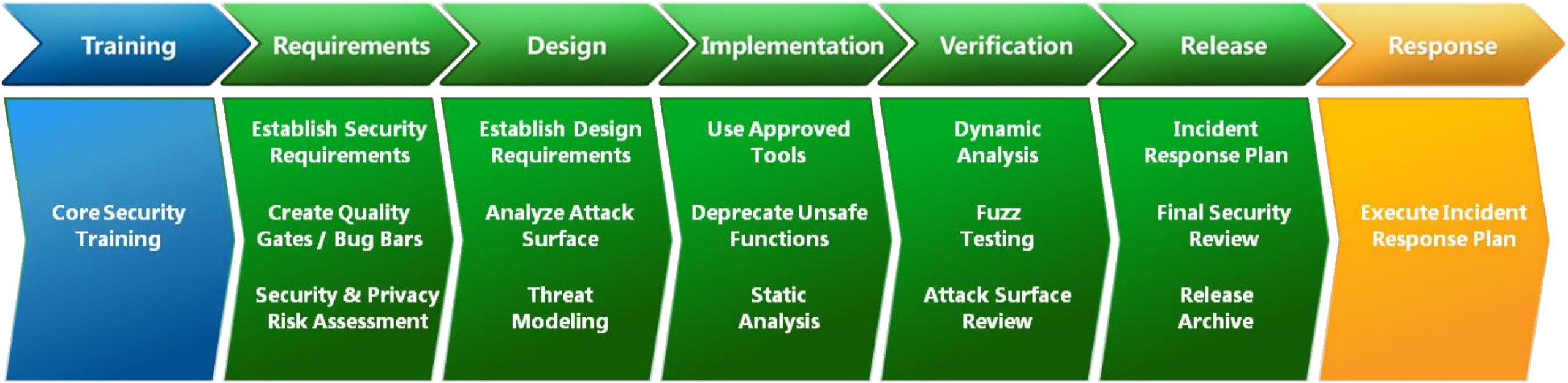
A set of activities to build secure software by design

Which SDLC?

- There are many Secure SDLC models (see later)

- We take a well-known one (from Microsoft) as an example
  - Initiated around 2002 (memo by Bill Gates)
    - Microsoft was under fire after a series of worms (Code Red and Nimda)
  - Their SDLC has undergone many changes since (in tandem with changing development practices and tools, e.g., agile development, CI/CD, …)

Example: Microsoft SDLC (v5.2, until 2012)



https://www.microsoft.com/en-us/download/details.aspx?id=29884
(Note: this picture isn't used anymore by Microsoft)

Example: Microsoft SDLC (current)

1 | Training

Make sure everyone understand the need for and the basics about security

3 | Metrics & compliance

Set a *bug bar* for security; track and report progress

5 | Design requirements

Ensure security functionality is correctly used

2 | Security requirements

Define and update what ‘secure’ means for your application

4 | Threat modeling

Think early about security implications

6 | Cryptography standards

Only use industry-vetted crypto libraries

https://www.microsoft.com/en-us/securityengineering/sdl/practices

Example: Microsoft SDLC

7 | 3rd party components

Inventorize and assess risk of 3rd party components

8 | Approved tools

Standardize and update tool options and versions

9 | SAST

Analyze source code for security vulnerabilities on every commit (code quality, linters, ...)

10 | DAST

Run-time verification of your software (scanners, fuzzing, ...)

11 | Pentesting

Let an attacker assess the security of your software

12 | Incident response plan

Have a plan for handing a (newly discovered) vulnerability or attack

https://www.microsoft.com/en-us/securityengineering/sdl/practices

Alternatives

- Is the Microsoft SDLC the only/best source for security activities? **No!**
- Many resources exist that guide you in implementing a secure software development program. **Pick what works for your situation.**
- Two other well-known examples:
  - NIST SSDF
  - OWASP SAMM

NIST Secure Software Development Framework (SSDF)

- V1.1 (NIST SP800-218), February 2022, freely available online (https://csrc.nist.gov/Projects/ssdf)
- Collection of best practices and links to other resources
- Possible **uses**:
  - Implement practices to increase software security
  - Use as a common language for procurement and management
- **Impact**: if you sell software to the US government, you now must (self-)attest compliance to the SSDF
  - https://www.klgates.com/Secure-Software-Regulations-and-Self-Attestation-Required-for-Federal-Contractors-5-19-2023

NIST SSDF: best practices in 4 groups

1. Protect the organization (PO)

Organizational policies, processes, roles
Infrastructure security requirements, shared security requirements, toolchains, quality gates, …

2. Protect the software (PS)

Managing software artefacts
Access control to source code, integrity-protected releases, archiving, …

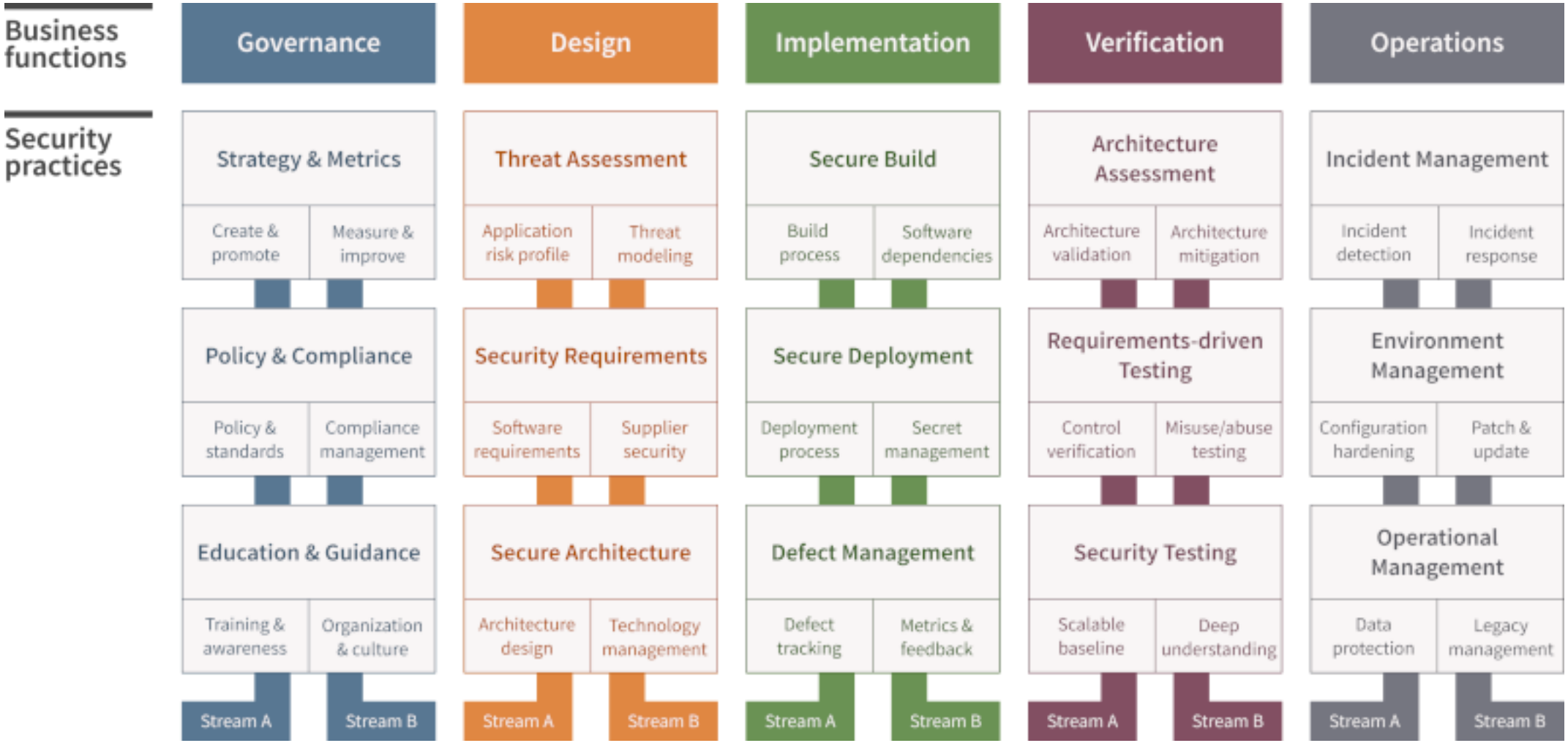
3. Produce well-secured software (PW)

Software engineering activities
Threat modeling, code reviews, use of libraries, compiler configuration, testing, …

4. Respond to vulnerabilities (RV)

Handling residual vulnerabilities
Identify and confirm vulnerabilities, review for similar instances, analysis and mitigation, …

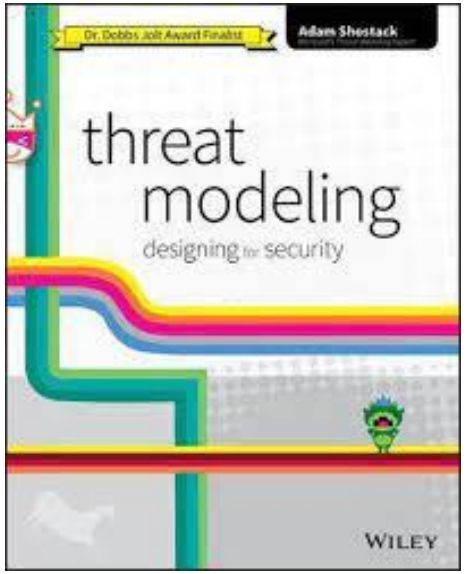
OWASP Software Assurance Maturity Model (OWASP SAMM)



Threat modeling

Threat modeling

- What is threat modeling?
  - E.g., see book of Shostack or https://www.threatmodelingmanifesto.org/
  - Analyzing representations of a system to highlight concerns about security (and privacy) characteristics.
- Popular techniques
  - Attack trees / attack-defense trees / …
  - STRIDE / LINDDUN (for privacy)
  - Misuse cases / Abuser stories / …



What is threat modeling?

"Identifying the likely threats to a system to inform the design of security countermeasures"

– Alyssa Miller

Look! There's a Threat Model in My DevOps (BSidesATL 2020)

https://www.youtube.com/watch?v=4KL7t1-FYBk

Threat modeling?

THREAT MODELING MANIFESTO

What is threat modeling?

Threat modeling is analyzing representations of a system to highlight concerns about security and privacy characteristics.

At the highest levels, when we threat model, we ask four key questions:

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?

https://www.threatmodelingmanifesto.org/

Threat modeling?

THREAT MODELING MANIFESTO

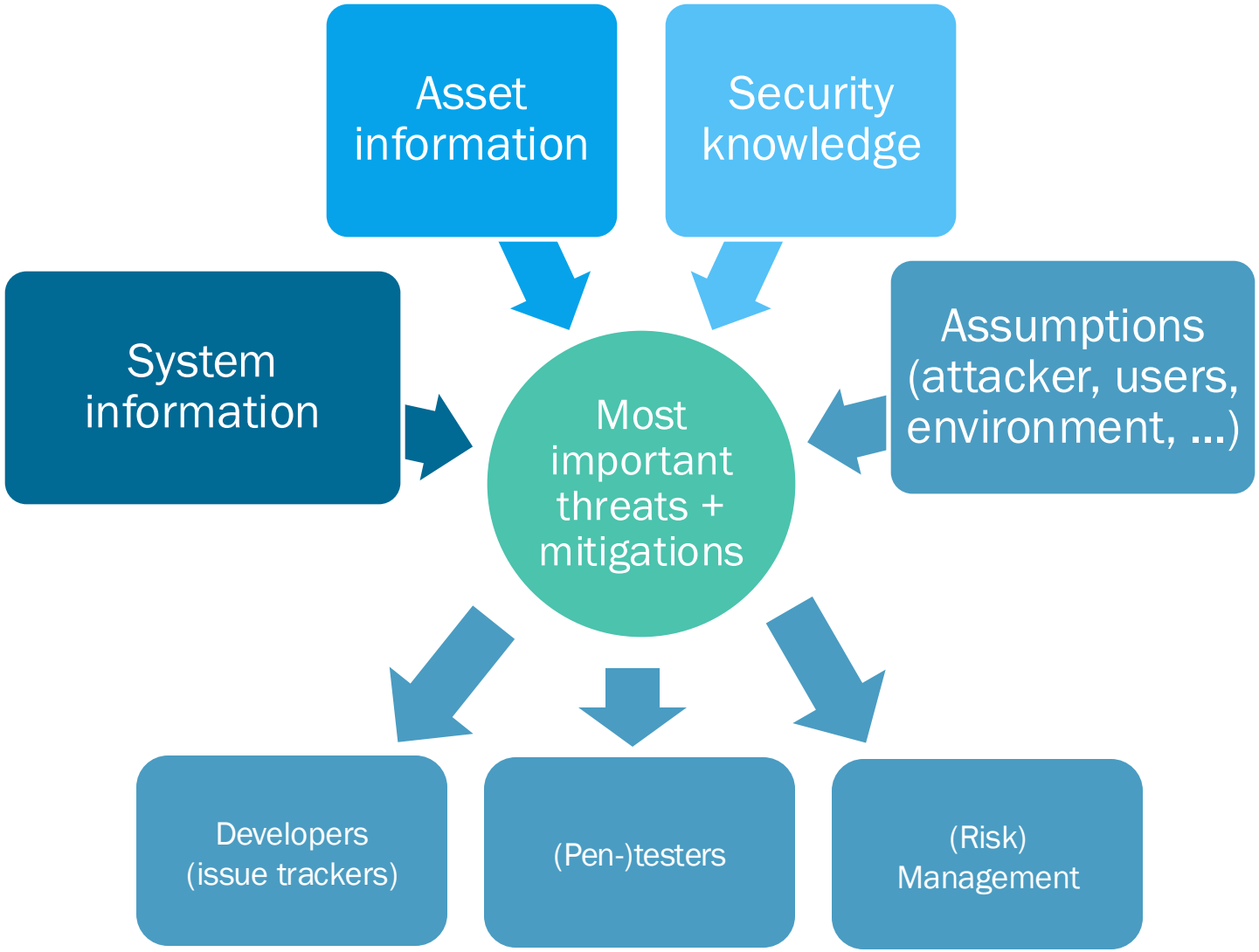
Principles

We follow these principles:

- The best use of threat modeling is to improve the security and privacy of a system through early and frequent analysis.
- Threat modeling must align with an organization's development practices and follow design changes in iterations that are each scoped to manageable portions of the system.
- The outcomes of threat modeling are meaningful when they are of value to stakeholders.
- Dialog is key to establishing the common understandings that lead to value, while documents record those understandings, and enable measurement.

https://www.threatmodelingmanifesto.org/

The big picture of threat modeling



Why threat modeling?

1. Find and address design problems that other techniques (e.g., SAST/DAST) don't cover
   - E.g., logical errors with security implications, feature interaction, missing authorization, …
2. Become aware of security problems early (€) rather than late (€ € €)

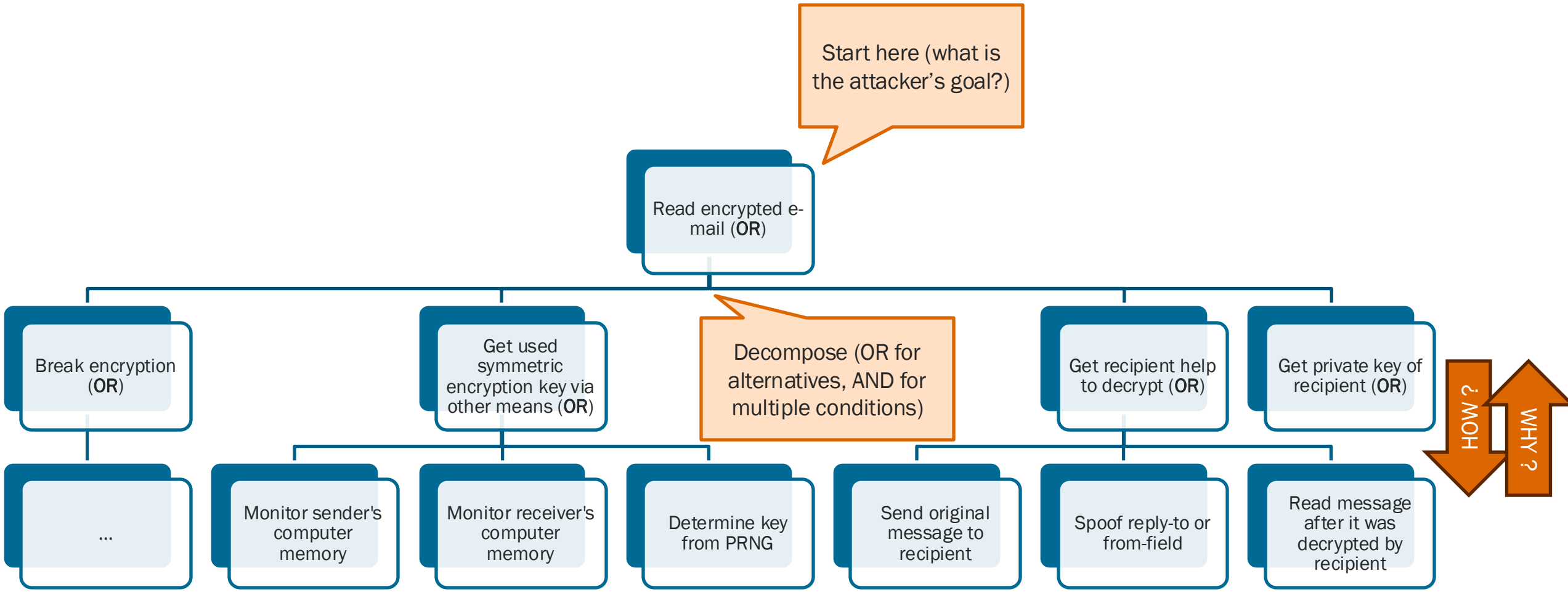
Why threat modeling?

3. Understand your security requirements and assumptions
   - What do we *really* want to protect against? If I have 1 € to spend on security, where would I spend it most effectively?
   - What security assumptions do we (as designers) make? (they need to be checked!)
   - Threat modeling as a driver for **pentesting**
4. An opportunity to (re)construct the design of the system

Shostack, A., 2014. Threat Modeling. Wiley.

Threat modeling techniques

Example technique: Attack tree



Schneier, Bruce. "Attack Trees." *Dr. Dobb's Journal*, no. December (1999).

Example technique: STRIDE

Informal meanings

- **S**poofing: Assuming an identity that isn't yours
- **T**ampering: Unauthorized modification of something (on disk, on a network, in memory)
- **R**epudiation: (Being able to plausibly) claim that you didn't do something (i.e., no logs/proof)
- **I**nformation disclosure: Providing information to someone not authorized to see it
- **D**enial of service: Absorbing resources to disturb/disable services for legitimate users
- **E**levation of privilege: Executing authorized (unexpected) actions

S. Hernan, S. Lambert, T. Ostwald, and A. Shostack, "Uncover Security Design Flaws Using The STRIDE Approach," *MSDN Magazine*, Nov. 2006.

Applying STRIDE

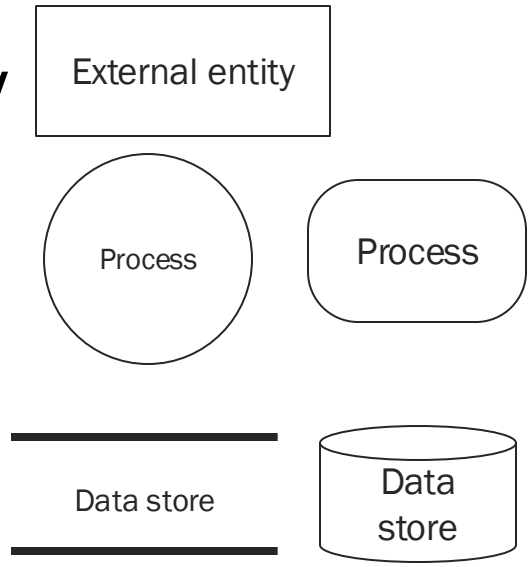
- Use STRIDE mnemonic when looking for threats
  - Brainstorming, EoP card game, … (‘*whiteboard hacking*’)
  - Focus on assets, attackers, **software**
- More systematic variants (~ algorithmic)
  - STRIDE per element
  - STRIDE per interaction

No completeness guarantees! (Involving a security expert is useful)

Only the discovery of a threat matters, not its precise categorization! (STRIDE is not a taxonomy)

STRIDE input: data flow diagram (DFD)

- External entity
- Process
- Data store
- Data flow
- Trust boundary



ELEMENT	APPEARANCE	MEANING	EXAMPLES
Process	Rounded rectangle, circle, or concentric circles	Any running code	Code written in C, C#, Python, or PHP
Data flow	Arrow	Communication between processes, or between processes and data stores	Network connections, HTTP, RPC, LPC
Data store	Two parallel lines with a label between them	Things that store data	Files, databases, the Windows Registry, shared memory segments
External entity	Rectangle with sharp corners	People, or code outside your control	Your customer, Microsoft.com

Shostack, A., 2014. Threat Modeling. Wiley.

Trust boundary meanings

- Different levels of trust or privilege in the system
- Representing information or assumptions on the attacker

  (e.g., parts system inaccessible to external attacker)
- Deployment information (which resources on same network)
- Separation of principals by some control (i.e., a countermeasure)

Laurens Sion, Koen Yskout, Dimitri Van Landuyt, Alexander van den Berghe, and Wouter Joosen. 2020. Security Threat Modeling: Are Data Flow Diagrams Enough? In Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops (ICSEW'20). Association for Computing Machinery, New York, NY, USA, 254–257. https://doi.org/10.1145/3387940.3392221

STRIDE

per element

	S	T	R	I	D	E
External Entity	x		x			
Process	x	x	x	x	x	x
Data Flow		x		x	x	
Data Store		x	?	x	x	

For each DFD element:

For each STRIDE threat type:

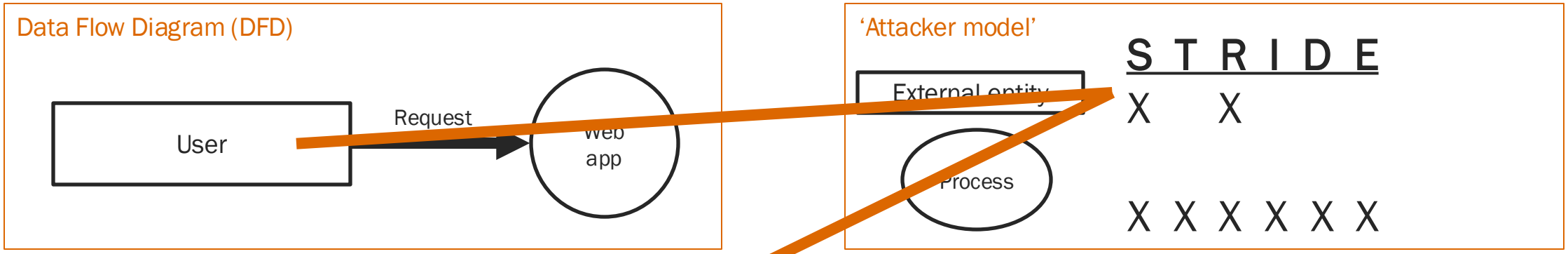
If table contains an ‘x’ at intersection, you’ve found a (potential) threat

STRIDE

per element

	S	T	R	I	D	E
External entity	Someone acts as **<the external entity>** when interacting with a process	-	**<The external entity>** claims not to be involved in some action	- *(Information of external entity is disclosed; see privacy)*	-	-
Process	Someone acts as **<the process>** when interacting with another process, external entity, or data store	Control flow or state of **<the process>** is tampered with	There is no convincing evidence of **<the process>** being involved in some action	Information handled or generated by **<the process>** is disclosed	**<The process>** becomes unavailable	**<The process>** executes actions it is not allowed to execute
Data flow	-	Information transmitted over **<the flow>** is tampered with	-	Information transmitted over **<the flow>** is disclosed	**<The data flow>** (channel) becomes unavailable	
Data store	-	Information stored in **<the data store>** is tampered with	**<The data store>** contains data for non-repudiation (e.g., logs) that is the target of an attack	Information stored in **<the data store>** is disclosed	**<The data store>** becomes unavailable	

Example: STRIDE per element



	S	T	R	I	D	E
User (External entity)	Someone acts as **the user** when interacting with a process	-	**The user** claims not to have sent a particular request	- *(Information of user is disclosed; see privacy)*	-	-
Web app (Process)	Someone else acts as **the web app** when interacting with another element (e.g., phishing)	Control flow or state of **the web app** is tampered with	There is no convincing evidence of **the web app** having processed the request	Information handled or generated by **the web app** is disclosed	**The web app** becomes unavailable	**The web app** executes actions it is not allowed/expected to execute

Knowledge base: threat trees

- You could say ‘generic attack trees’
- Provide refinements of top-level STRIDE threats
  - Add technical detail
  - AND/OR decomposition
- E.g., to spoof a client, you can
  - obtain the client’s credentials
    - by observing them in transit
    - OR by using functionality to change the credentials
    - OR …
  - OR …

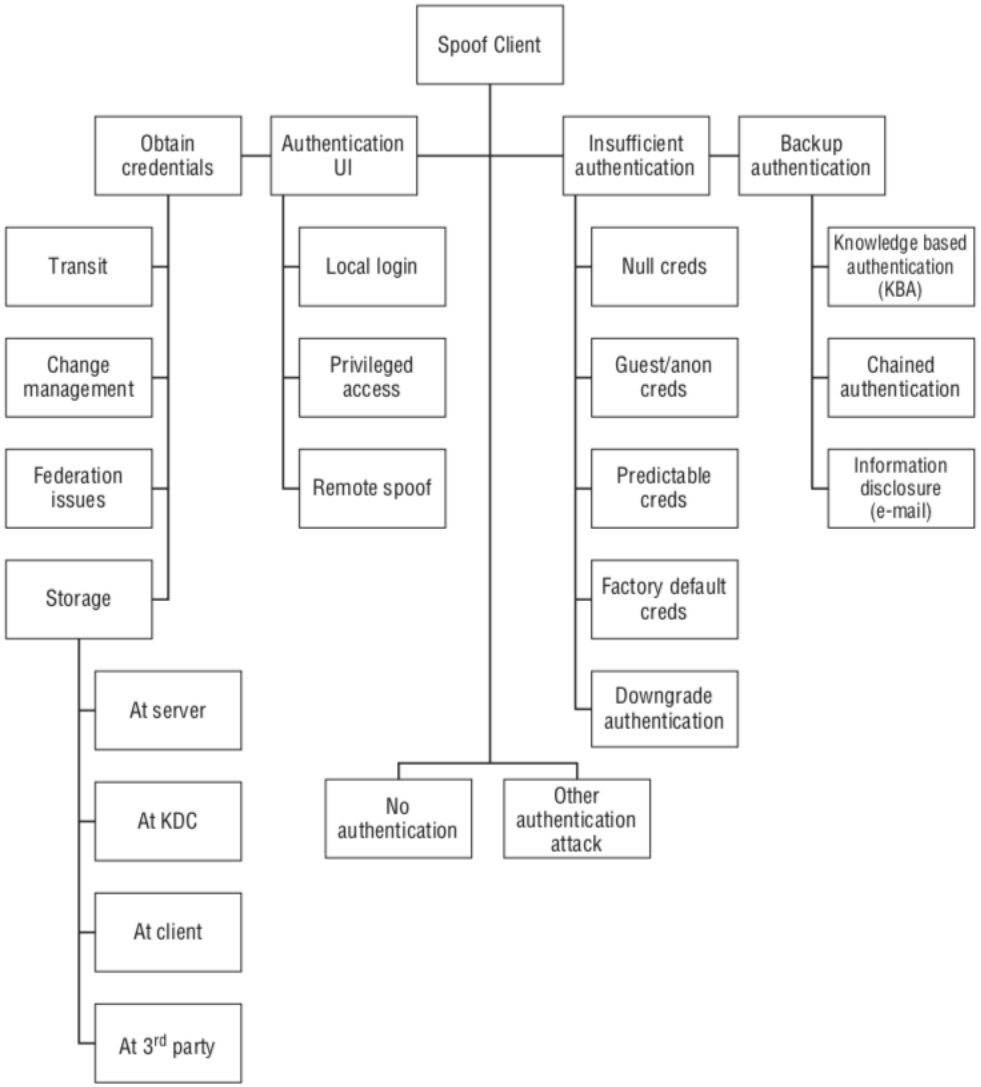
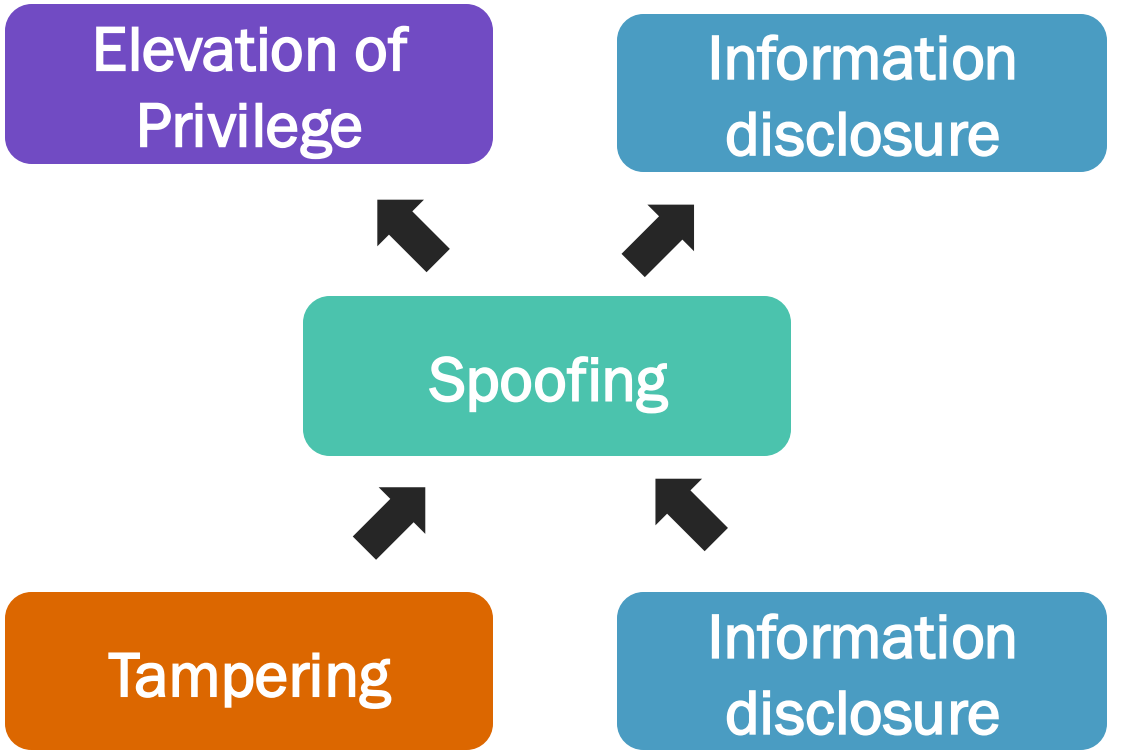


Figure B-1: Spoofing an external entity (client)

Relationships between the STRIDE threat types

E.g., Gain access to system commands because you impersonated an authorized user



E.g., Gain read access to a sensitive resource because you impersonated an authorized user

E.g., Forging a cookie that is used for identification in a web application

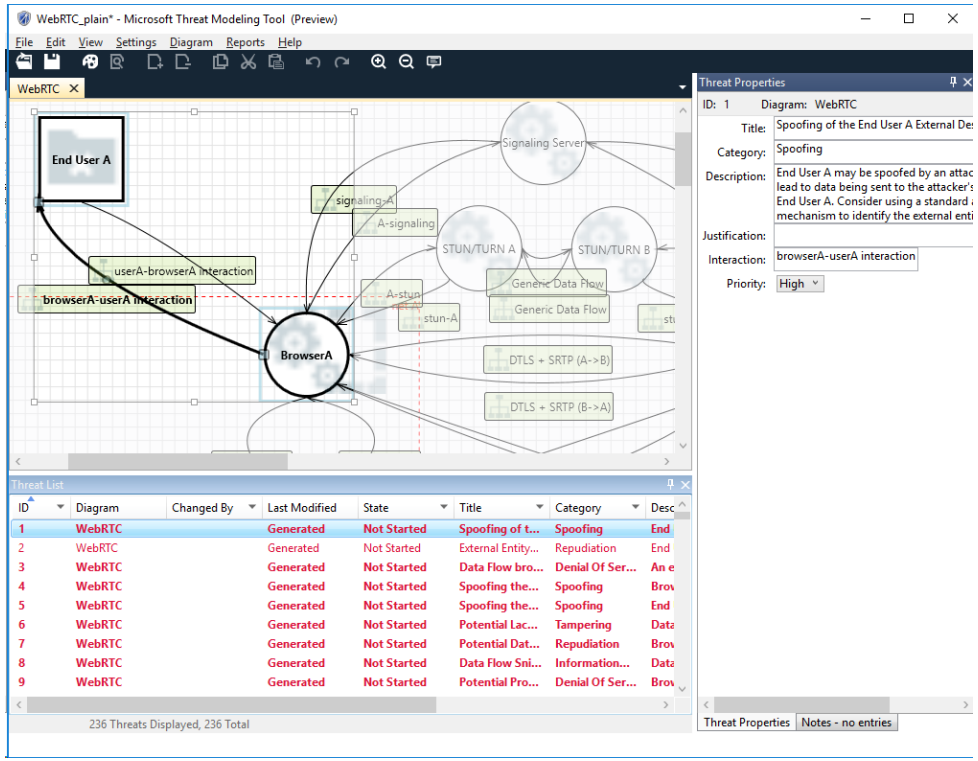
E.g., Leakage of an access token because it is not encrypted during transmission

Threat modeling: heavy effort? NO!

- An informal session of a few hours per project may already yield important insights

- For each new feature, ask yourself “how could this go wrong/be abused?”

Threat modeling tools



- Diagram-based (DFD or similar)
  - Microsoft Threat Modeling Tool
  - OWASP Threat Dragon
  - ThreatModeler
  - IriusRisk
  - Threats Manager Studio
  - SPARTA
- Code-oriented ('threat modeling as code')
  - PyTM: describe your threat model with Python code (and generate reports)
  - ThreatSpec: add threat modeling annotations to your source code (and generate reports)
  - Threagile: add a threat model as a yaml file to your repository (and generate reports)

For more tools and a taxonomy, see

SHI, Z., GRAFFI, K., STAROBINSKI, D. AND MATYUNIN, N., 2021. Threat Modeling Tools: A Taxonomy. *IEEE Security & Privacy*, pp.2–13.

Theory vs. practice

What is the state of practice regarding threat modeling?

- In collaboration with NCSC (NL)
- Semi-structured **interviews** with

  13 **practitioners** from 7 large Dutch organizations
- 1-hour interview (recorded, transcribed, coded)

S. Verreydt, K. Yskout, L. Sion, and W. Joosen, “Threat modeling state of practice in Dutch organizations,” presented at the Twentieth Symposium on Usable Privacy and Security (SOUPS 2024), 2024, pp. 473–486. Accessed: Sep. 11, 2024. [Online]. Available: https://www.usenix.org/conference/soups2024/presentation/verreydt

KU LEUVEN DistriNet

Nationaal Cyber Security Centrum
Ministerie van Justitie en Veiligheid

Research questions

RQ4
Experiences

RQ1
How is threat modeling embedded in the organization?

RQ2
Which organizational roles are involved in threat modeling activities?

RQ3
How is threat modeling performed within the organization?

Research questions

RQ4
Experiences

RQ1
How is threat modeling embedded in the organization?

RQ2
Which organizational roles are involved in threat modeling activities?

RQ3
How is threat modeling performed within the organization?

Purpose of threat modeling

- Finding potential **vulnerabilities**
- Raising security **awareness**
- **Communicating** about security issues with non-technical people

“a way for [developers] to discuss information security in a practical way within their team”

Motivation for threat modeling

- A culture of finding and fixing design issues over checkbox compliance.

- Seldom mandated, except for high-risk applications
- Focus on **internal motivation**

"[…] the moment you start forcing threat modeling, people naturally lose enthusiasm and do it because they have to and not because they see the usefulness and necessity of it."

Research questions

RQ4
Experiences

RQ1
How is threat modeling embedded in the organization?

RQ2
Which organizational roles are involved in threat modeling activities?

RQ3
How is threat modeling performed within the organization?

Who is involved in threat modeling activities?

Varied Viewpoints

Assemble a diverse team with appropriate subject matter experts and cross-functional collaboration.

- **Security team** creates **awareness**
- **Development** teams **initiate** threat modeling sessions
- Main participants: Developers, Product owner, Facilitator (security team)
- Usually **not involved**: testers, information security officers, architects, IT admins

"[…] they don't have the capacity [to attend threat modeling sessions]"

Who introduced threat modeling in the organization?

- Triggered by **(previous) experience** of a security team member
- Hiring **external expert** to support introduction is well-received

"[the external expert] does not have the bias of the organization and its processes [...] such that we can first determine what [threat modeling] is and what it adds, before it finds the right spot [in the organization]"

Involvement of management

- Not always aware of threat modeling and its benefits
- Demonstrating the effectiveness of threat modeling is challenging
- **Difficult to get support**, time and resources for threat modelling
- Don’t realize that participation is valuable
- Lack of follow-up

“management, according to me, does play a role in accepting [threat modeling], seeing the added value of it and being able to translate that back to their stakeholders as well”

Research questions

RQ4
Experiences

RQ1
How is threat modeling embedded in the organization?

RQ2
Which organizational roles are involved in threat modeling activities?

RQ3
How is threat modeling performed within the organization?

Planning

- Continuous refinement over a single delivery.

Preferably **early on** in the development lifecycle and **periodic** re-assessment

➔ But this is **difficult** in practice:

- **Scope** may not be clear early on
- **Mitigating** threats may be difficult later on
- Security team lacks resources / backlog of high-risk applications
- Finding a hole in everyones **schedule**

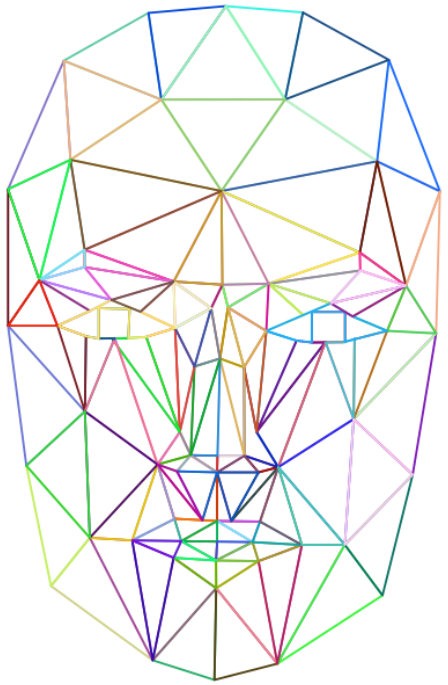
" [the security team] simply doesn't have the capacity for that yet, because we just have so many development teams."

Modeling an application

Useful Toolkit

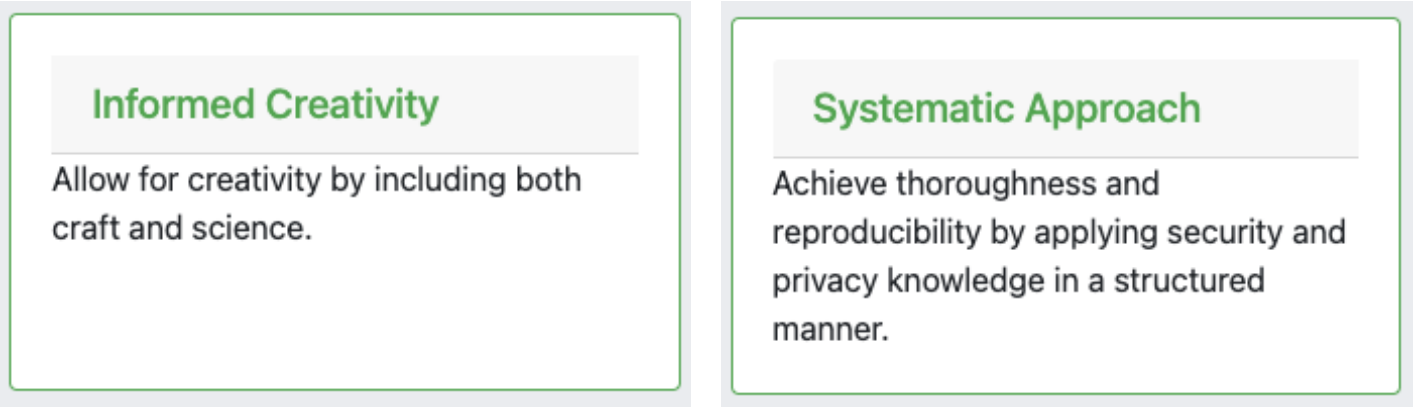
Support your approach with tools that allow you to increase your productivity, enhance your workflows, enable repeatability and provide measurability.

- Ranging from white board drawings to structured notations like data flow diagrams
- Input (architectural documentation) not always available
  - ➔ Creating a model may be **time-consuming**
- Advantage: create **mutual understanding** of the architecture
- **Limited use of tool support** for drawing diagrams
  - May require too much (detailed) inputs and complicate the process
- Balance between model quality/correctness and overhead



"there is no single record, with the truth, not even on a conceptual level"

Method



- Mostly STRIDE, other methodologies depending on the context
- Prefer **pragmatism** over strict methodologies

  "[...] it's really not so much about whether it's done very well. The point is that we do it, and that we learn from it together and gain knowledge [...]."
- Prevent lengthy discussions on the specific methodology
- Teams tend to focus on provided examples in learning material

Output

- A journey of understanding over a security or privacy snapshot.

- Report includes the system model, identified threats, existing mitigations and mitigation advice.
- Threats may be prioritized
  - But **risk estimation difficult** due to lack of security knowledge
- Preferable to **limit reporting overhead**

"writing takes a lot of time, and I don't know if it's always worth the effort. Going through the process is perhaps the most fruitful."

Follow-up

- Mitigate severe threats
- Input for **pentests**
- Other follow-up **depends on priorities**
- Interpreting threat modeling output may not be straightforward

"It's not that they don't want to do security, but they have so many other things to think about besides security"

Limitations

- Sample size
- Selection bias
  - Large organizations providing critical services
  - Dedicated security department
  - Mostly in-house software
  - Knowing/Willing to talk about threat modeling
- ‘Threat modeling’ may not cover similar activities (e.g., ‘secure design review’)

(Our) research in this area

- Automation
  - Leverage the relationships between threats
  - Extract (DFD) models from source code, IaC configuration files, …
  - Encode security knowledge (threat trees, security patterns)
- Literature review and controlled experiments on effectiveness of threat modeling
- Follow-up studies “threat modeling in practice”
  - Smaller companies, other sectors, …?
  - Situation in other countries? (Belgium?)

- **Talk to me or send an e-mail if interested in any of the above!**

koen.yskout@kuleuven.be

Key points

- Threat modeling is a good first step towards security by design
  - Also to raise security awareness
- Start now (and be pragmatic)!
- STRIDE is a useful mental framework to start from

Koen.Yskout@kuleuven.be

KU LEUVEN DistriNet

Q&A and discussion