SHOSTACK
+ ASSOCIATES

The Jenga View of Threat Modeling

Supporting Delivery of Resilient Software

Secure Products
Critical Thinking
Brainstorming
Requirements
Focus on Solutions
Patience
Shift Left
Humility
Business Goals
CAPEC
STRIDE
Collaboration
Attack Trees
DFD
Kill Chains
Training
Knowledge
Executive Support
Metrics
Bugs
Good Intent
Office Hours
DoS
Active Listen
Tools
Tampering
What are we working on?

Shostack + Associates White Paper #1
June 2020
by Adam Shostack

Executive Summary

This paper presents a new view of how security can be a part of technology designs from the start. We introduce a metaphor of Jenga blocks supporting a secure product. In the game of Jenga, there's a tower of blocks. Play involves removing them one at a time, and then piling the removed blocks on top of the tower until it collapses. In the business world, we aim to construct a tower that is both stable and lightweight, using only as many blocks as needed, because each block comes at a cost.

This paper is intended for leadership in engineering and application security. Security can and should be a part of the design of new products, services, features, system architecture and other technological systems. Including security at design time results in faster and more predictable launches of better designs with fewer security problems.

Context

Security, in the sense of resilience to attack, is increasingly acknowledged by executives as a requirement. This requirement often leads to the creation of a secure development lifecycle program (SDL or SSDL). These programs often have early wins from penetration testing, fuzz testing or static analysis. As these programs start, they often focus on tools and outsource-friendly techniques, which may limit their impact. They may reduce bugs without dramatically improving resilience.

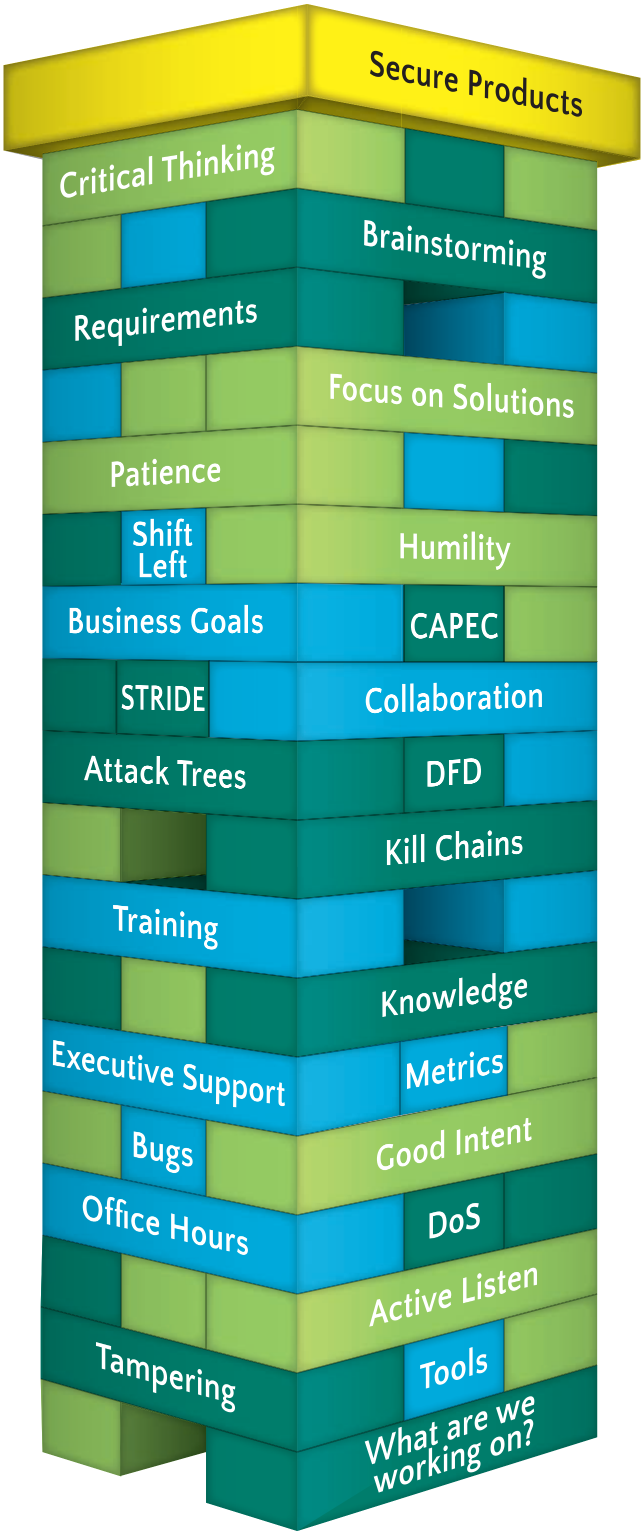
Threat modeling is a key to achieving resilience. Without threat modeling, without considering what could go wrong, it is hard to have confidence that software or a service will be free of unpleasant and hard to fix surprises. Tools and automation are important, but as long as software development requires judgement and human decisions, some of those decisions will involve security or have security effects that tools don't see.

Structure is key to any practice growing from something that a few people do to something an organization does. Without structure, it's random activity. It's hard to measure or justify. It's hard to judge if the work has been productive.

Threat modeling has been an artisanal activity that some people perform to great effect. As the discipline matures, it has become clear that to reach the goal of resilient software we need more than just technical skills in threat modeling. We need soft skills such as active listening, respect, and assumption of good intentions. We need organizational support, including processes and policies. We need standards for how the work is to be done, including gates and nets to prevent or catch mistakes. We might want software to help, and training that covers each of these.

To support the goal of resilient software, we have started using a metaphor of Jenga blocks, where each block is a skill, technique, or tool that helps ensure appropriate threat modeling. The blocks come in three flavors: technical, interpersonal, and organizational. With more blocks in place, the structure is more stable (the real game of Jenga is more complex, but our metaphor does not include piling blocks on top.)

Types of Building Blocks



A successful threat modeling program includes:

TECHNICAL

INTERPERSONAL

ORGANIZATIONAL

building blocks. The mix of blocks is different from those needed to support other software security activities. For example, static analysis is very much centered on technical blocks, and interpersonal ones are less important.

The technical skills of threat modeling are the ones that help us answer the first three questions in the four-question framework:

1. What are we working on?

- Model systems using approaches like whiteboards, data flow diagrams, state machines, swim lanes and other diagramming techniques
- Specific modeling techniques such as C4, DFD3 or UML

2. What can go wrong?

- Discover threats using STRIDE or Kill Chain
- Analogize from existing threats using CAPEC or ATT&CK
- Educate peers about technical threats

3. What are we going to do about it? [1]

- Advise on remediation approaches
- Analyze prioritization approaches
- Write tests for TDD, acceptance testing, or anywhere in between
- File actionable, compelling bugs or tickets
- Write user stories, epics, or reports to capture findings

For example, being able to draw a data flow diagram (DFD) is a technical skill that helps people be precise in explaining "what are we working on?" Various forms of STRIDE and Kill Chains can help address "what can go wrong?" STRIDE is a mnemonic of various threats (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege), while Kill Chains are a model of how attackers chain activities together to reach (kill) their objectives.

Security professionals are often advised to start from risk assessment. My experience is there are often improvements that can be made faster than assessing their risk. Examples include moving files from /tmp or changing permissions on an S3 bucket. When the analysis is harder, various prioritization analyses can help.

Additional technical skills and knowledge-style building blocks are broadly helpful, and many require no threat modeling specific adaptation. These include:

- Critical thinking
- Knowledge of a repertoire of attacks
- General technical knowledge of the systems being used
- Understanding of software delivery models including DevOps, agile, Scrum and the local customizations to these models
- Teaching (especially around security)
- Agile approaches to work, including small incremental delivery, testing, and improvement.

[1] The fourth question, "Did we do a good job" does double duty, giving a place to assess work at in both technical and organizational senses.

INTERPERSONAL Building Blocks for Threat Modeling

The best technical skills in the world can be overshadowed by interpersonal issues. Even with the best technical skills, interpersonal issues can not only ruin a threat modeling session, but can also shake trust, destroy rapport, and reduce collaboration between security and other teams. Critical analysis is harder to accept when delivered condescendingly. Even good suggestions from "know-it-alls" are left on the cutting room floor.

In contrast, effective communication has many elements that we see consistently across very different work cultures. Those elements include:

- Active listening
- Focus on solutions
- Patience
- Humility
- Respect
- Assumption of good intent
- Moderation and facilitation
- Understanding the working culture
- Working the organization[2]
- Developing a support network
- Commitments and predictability

Of course, any of these can be taken to a counter-productive extreme. Moreover, we prefer to label these as "interpersonal skills" because "soft skills" get a bad rap, often for being imprecise. Each of these is specific enough to be taught or assessed.

[2] Working the organization essentially means before a formal meeting, informally talking to those who will make a decision and those who influence them. These conversations provide an opportunity to gather feedback and address criticisms before a formal decision.

We split organizational blocks into rough groups of governance and operations. Governance means that the executive leadership has accepted the importance of the work, possibly incorporated into a vision such as secure by design or product security, and assigned someone to govern it. That person is accountable for the operational decisions to implement the goal.

To this point, we have been considering threat modeling not only as a set of activities, but also the skills that allow people to execute on those activities. Some of those activities have outputs (deliverables), and those can be specified by an organization to great effect. Other activities simply inform the work, but the steps are not written down. For example, if a back of an envelope analysis quickly shows that a possible design choice has many side effects, recording that the analysis was done may not be worthwhile.

SOFTWARE

People often want to threat model in a specific tool. This as a common approach; the other common approach is to integrate threat models into other software artifacts. In that case, the software, such as Jira or Word, is not threat modeling specific. And threat modeling is more dependent on human thought than other aspects of secure software development, which are more amenable to automation.

ROLES AND RESPONSIBILITIES

Software and operations engineers must be deeply involved in considering what can go wrong and what are they going to do about it. When they are not, the results will be less impactful. They can and should be supported by people in other roles.

Governance

- Clear goals
- Executive support
- Defined stakeholders and accountability
- Definitions, monitoring, and optimization

Operations

1. Roles and responsibilities

- Who is responsible for what?
- Are new roles created, or are existing roles altered? For example, are there security architects or champions?
- How does threat modeling relate to expectations at different levels of seniority?
- Who is empowered to sign off on exceptions, and how are those tracked?
- Are meetings expected, and who participates in what capacity?

2. Enablement

- How are people made aware of the change?
- What training do people need to enable them to meet the new requirements?
- What support is offered (Office hours, Slack channel)? What SLA is committed?

3. Deliverables (new and changed)

- How and where are the deliverables stored?
- What form do they take, or what existing forms are altered?

4. Software (threat modeling software can help with or enforce)

- Process: Software can help model systems, discover threats and/or mitigations, and connect to gates and nets.
- Documentation: software can help produce documentation of the threat model (system models, analysis and mitigations.) It can also produce documentation of what's being done, and integrate that into systems for process visibility.
- Automation: Discover threat types, file bugs, track changes, and the like.

5. Standards, policies, and procedures

- Gates and nets[3]
- What deliverables become gated on threat modeling deliverables?

Of course, as statistician George Box teaches us, "all models are wrong and some models are useful." Some blocks may fit into several of the above categories without detracting from the usefulness. For example, software that helps technical execution of threat modeling is a technical building block as it helps answer the question "what can go wrong?", and an organizational block when it helps measure the program.

[3] Gates and nets is a way of thinking about how projects proceed. Hollywood famously "green lights" projects: it's a known gate that projects must pass through. Other projects proceed on their own, but with safety nets to catch their failures, possibly including penetration testing or bug bounties. Smart organizations will often use and instrument nets, and convert the common failures to mandated gates of various types.

Conclusion: What Structure Do You Need?

As anyone who's played Jenga knows, a tower is stable when the game begins, and as blocks are removed, stability decreases until the whole thing comes crashing down. We remove blocks in the game because that's the game. But in business, we may not start from a stable tower. We are probably missing blocks when we start, and need to insert some. Also, we remove blocks because each block has a cost. There's the cost of executing on the block, training, measuring, dealing with exceptions and escalations, and so on. From an organizational viewpoint, a bit of a risky tower may be an acceptable steady state. These questions are separate from those of maturity.

A key question to consider is how much structure your organization wants or needs. Your organization may want more structure because you make a high-assurance product, or you may need it because you work in a highly regulated field. Either way, you likely have a framework like FMEA or HAZID, possibly embedded in an approach like Safety-II. You may be working in a software startup that applies a YAGNI test to everything. (YAGNI stands for "You Ain't Gonna Need It.").

If you have a security or threat modeling program that's not standing up on its own, perhaps studying the building blocks will help you see that you've been focused on technical skills, or organizational matters such as software, at the expense of interpersonal skills. If your program has collapsed under its own weight or been pushed to a separate "safety" team, perhaps questioning the value of each block can help. The combination of blocks to make a successful program is as varied as the companies deploying threat modeling.

How Can We Help?

The Jenga model emerged from our coaching work. Frustration and confusion were constant problems. If you're struggling with rolling out threat modeling, with fights over how threat modeling gets done, tracked, managed or measured, we can help. Our coaching service is designed to help you see the behaviors and impacts so you can adjust and drive better outcomes. Why not drop us a note?

FAQ

Q. Does the positioning of the blocks matter?

A. No.

Q. Is the set of blocks complete?

A. No. It's intended to be representative and evocative.

Q. Can I buy a set?

A. Not from us, and not anywhere we're aware of. Sorry, it's a metaphor, not a product.

ABOUT SHOSTACK + ASSOCIATES

Shostack + Associates is a trusted specialized security consultancy, focused on meeting the unique needs of each client through a variety of services including threat modeling, security engineering and risk management. For more, please visit: *https://shostack.org/about/shostack+associates.*

Get In Touch

If threat modeling isn't delivering what you hope for, then it's our hope that this paper will help. If we can help further, please don't hesitate to reach out for a confidential consultation, at *adam@shostack.org.*

ABOUT ADAM SHOSTACK

Adam is a leading expert on threat modeling, and a consultant, expert witness, author and game designer. He has decades of experience delivering security. His experience ranges across the business world from founding startups to nearly a decade at Microsoft.

His accomplishments include:

- Helped create the CVE. Now an Emeritus member of the Advisory Board.
- Fixing Autorun for hundreds of millions of systems
- Led the design and delivery of the Microsoft SDL Threat Modeling Tool (v3)
- Created the *Elevation of Privilege* threat modeling game
- Wrote *Threat Modeling: Designing for Security*
- Co-authored *The New School of Information Security*

While not consulting or training, Shostack serves as an advisor to a variety of companies and academic institutions.

ACKNOWLEDGMENTS

Cédric Lévy-Bencheton, John Benninghoff, Edward Bonver, Izar Tarandach and others gave early helpful feedback. Jenga is an awesome party game from Pokonobe Associates. Adrian Lane's blog post "Enterprise DevSecOps: Security's Role In DevOps" *https://securosis.com/blog/15000* inspired expansion of the building block list.

LICENSE