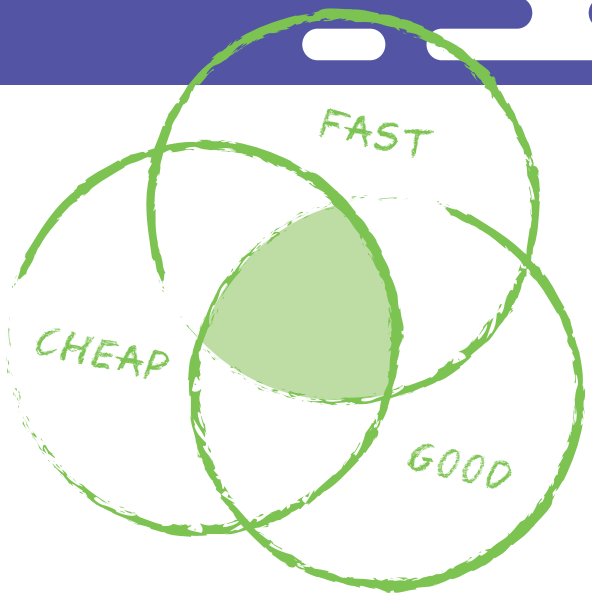
SHOSTACK
+ ASSOCIATES

Fast, Cheap and Good

An Unusual Trade-off Available in Threat Modeling

Shostack + Associates White Paper #3
December 2021
by Adam Shostack



Executive Summary

Threat modeling work can be very rewarding. There is a common pattern where a lightweight proof of concept run by security experts leads to the creation of a heavier process. This heavier process is designed to help developers, operations and others with less security expertise. These approaches are often too heavy for low-risk projects, too big for agile projects, and they don't consistently produce results worthy of the invested energy.

This creates a situation in which fast and cheap ways to figure out **'What can go wrong?'** and **'Do we need to dig deeper?'** are better than heavyweight approaches. This paper presents a set of approaches as simple as asking, "What can go wrong?" It also provides a framework that allows us to consider direct return on effort as well as other common goals for security assurance such as consistency and assurance.

This paper:

- Collects a set of fast, cheap, and good ways to begin threat modeling
- Measures them on several metrics
- Provides you with a decision framework to illuminate when to use more in-depth approaches

Understanding the Situation: Indicators of No Compromise

There is a set of phrases that correlates with problems in threat modeling::

> **“Threat modeling is a waste of time.”**
> **“We have a questionnaire that we ask people to fill out in lieu of threat modeling.”**
> **“We only threat model the highest risk systems.”**

These phrases indicate that the security team brought a no-compromise approach to threat modeling: do it our way, or don’t do it at all! Don’t get me wrong, I love it when people put my books on a pedestal. But I love it even more when they build better products.

Compromises and trade-offs are crucial parts of engineering. Threat modeling helps us bring security into system trade-offs, and the design of a threat modeling system involves compromises. The traditional model of ‘fast, cheap, or good: pick two’ encapsulates the understanding that engineering processes always include prioritization, while asking management or the customer to determine those priorities.

Let’s start by examining and addressing each of these objections in order to respond.

“Threat Modeling is a waste of time.”

Ok! Implicit here is the assumption that threat modeling will take a lot of time, and that the return on investment is low. That’s fair if you think there is only one true way to threat model. If you think of threat modeling as a collection of techniques, and if you agree that some of those techniques take more time than others, you can then say, “There is no threat modeling approach that could provide value,” or you could accept that some might. Here’s a simple technique: ask, “What can go wrong?” You’re now threat modeling. If, however, you consider that question a waste of time, please stop reading. Don’t even delete the file. That, too, would be a waste of your time.

“We have a questionnaire”

My mental model of these questionnaires is that they feature questions such as, “Does your system accept data from the internet?”, “Does it process information about people?” and “Are you using open-source libraries?” Please provide a list.” Additionally, my mental model is that they are usually on the order of 30-100 questions and are slow to fill out, because no single individual has all the responses. This is also threat modeling, albeit in its least fun form. The questions in the list are a model of what can go wrong, presented in a way that implies, but doesn’t explicitly state, the answer. If you answer enough of these questions well, you then get to do more threat modeling.

“We only threat model the highest risk systems.”

How do you know? There is some kind of threat modeling (sometimes called a risk assessment) which determines the risk level for a system. It might be H/M/L, it might be a numeric score, but that work to determine what is high risk is…(wait for it)…threat modeling.

Each of these approaches frames threat modeling as something different than the work that is being done, and to the work that could be done or should be done. By artificially introducing a discontinuity, we make our lives harder. Sometimes we need to go deeper, and we should design our approaches so that going deeper doesn’t create resistance.

Ambient Information

At its heart, threat modeling is a collection of ways to anticipate and address problems. Fast and cheap ways to do that are inherently good.

Software development projects have a huge amount of *ambient information*, which is to say, understood by those involved without having been made crisp or specific. The work to make the ambient explicit can be daunting, and we should treat that work like any other work, requiring justification and prioritization. There are many problems that result from ambient information being inaccurate, incomplete or inconsistent, and good, agile practices are often focused on reducing those problems at a reasonable cost..

Context is important in order to discover security problems. We might think of security problems on two axes: "How much system knowledge is required?" and "How much security expertise is required?"

For example, discovering a remote file inclusion attack requires knowledge of the technique. We might count that as low security expertise. Discovering a new cryptanalytic attack requires a wealth of knowledge of cryptanalysis.

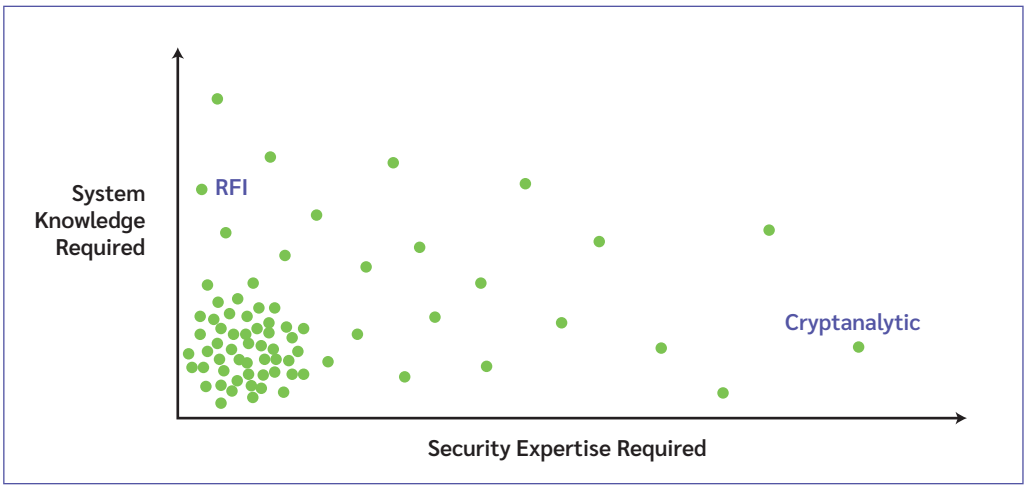


Figure 1: Knowledge required to find bugs

Leverage Non-Specialists

How much security expertise is required to find the problem? How complex is it to execute? Many problems are labeled "obvious", and they are, in hindsight. Once someone looks. If that's the case, then systems that help non-specialists look for security problems might reasonably find the problems in the lower left of Figure 1. And if, as presented, most problems are on the left side of the graph, then these fast, cheap systems are valuable.

Fast, Cheap and Good Approaches

Over the last few years, there has been a profusion of ultra-lightweight methods for threat modeling. Some of these have been promoted at industry conferences, while others have been presented in academic papers. Many of these methods fit within my Four Question Framework for threat modeling: "What are we working on?", "What can go wrong?", "What are we going to do about it?" and "Did we do a good job?", but some of them address a subset. Only taking on a subset of a method can be a reasonable way to optimize, as is scoping to solve a smaller problem.

In fact, the simplest approach to threat modeling is simply to ask "What can go wrong?" or, as in the popular meme variant, "What could possibly go wrong?" You can also include this question in checklists, such as those for a pull request, code review, sprint planning session, or the like.

This approach treats the answers to the other questions as implicit.

Many threat modeling projects are structured for the convenience of consultants starting with interview questions such as, 'What are you working on?', or 'Can you bring us consultants a diagram?' The needs of the consultant are literally put first, because consultants don't have the ambient information.

Some readers may look at these fast, cheap methods and have the same response that I had: "That's not even threat modeling!" But it is. It's using models to anticipate problems. The models are implicit, and that can lead to all sorts of failures. Anticipating those failures and preparing for them can seem wise to security folks, and overwhelming to those whose main focus is product delivery.

It's useful to collect and catalog these fast, cheap methods, which are discussed in greater detail below. These tools can be useful to those with little time, limited threat modeling skills or high uncertainty about the value of deeper analysis, especially in the context of the small work units common in agile environments.

1. **What can go wrong?** Simply ask, "What can go wrong?" as discussed above.
2. **Make it personal:**
   a. How would you attack this?
   b. What are you worried about?
   c. What would make me call you at 2 AM?
   d. How would you steal money with this?
3. **Fortunately/Unfortunately**
4. **Security fictions**
5. **Center of Gravity.** Ask, "What is your work section's objective/mission?", "What assets are used to accomplish this mission?" and "What is your center of gravity?"
6. **Lock architects in a room**
7. **Card games:** *Elevation of Privilege* and derivatives
8. **Use cases and variants**
9. **Libraries of Risk Patterns**

2. Make it Personal

You can see that each of the "make it personal" variants shares a crucial word: you. This can help focus attention. These questions are all variants on what can go wrong. The 2 AM variant helps people think about a potential problem and makes it concrete. This is an informal variant of Klein's "Pre-Mortem," as discussed in his book, *Sources of Power*. The steal money variant focuses attention on a particular type of attack, and it's easy to build on that variation for whatever valuables the system holds. (Klein, 1999)

3. Fortunately/Unfortunately

"Fortunately/unfortunately" comes to us from improv comedy, and works surprisingly well in threat modeling. Someone says, "fortunately, X," and someone else replies, "unfortunately, Y," and the exercise continues back and forth. For example, "Fortunately, we're encrypting all this data!" "Unfortunately, we store the key in the same directory!"

The fortunately/unfortunately structure provides an opportunity for both playful and serious exploration, within a very quick demonstration ("training") in how it works.

4. Security Fictions

Security Fictions adapts tools from speculative design to encourage thinking about threat identification. As a method, Security Fictions is presented as a role-playing game. The system involves two people, one is the organizer, the other, the participant. The setup, presented by the organizer, is, "Our client wants your (the participants') help. Please leave your ethics aside, and describe how they can do these things *without your help."* (Emphasis added.) The prompts the organizer uses in the paper are: "Impersonate another user," "Find the physical location of a particular user", and "Find all the politically conservative users." (Merrill, 2020)

These key prompts are only slightly deeper than the question, 'How would you steal money with this?" However, the preparation and framing as a role-playing game is interesting. Unlike the other methods in this paper, Security Fictions is presented as a one-on-one exercise.

5. Center of Gravity

The Center of Gravity approach is described in an academic paper entitled *"The Battle for New York."* Quoting from the paper, *"As a military concept, a center of gravity is the 'primary entity that possesses the inherent capability to achieve the objective.' As a threat modeling approach, CoG focuses on identifying and defending this central resource."* (Stevens, 2018)

The proper CoG method involves an hour of training and an hour of individual work going through the nine parts of a worksheet, applying it to a system. It's possible that with experience, the worksheet will take less time.

One way to use this concept (which I'll call CoG-micro) might be to simply ask, "What are we protecting here?" This may be enough to release people from a compliance mindset, and get them thinking. (This hasn't been tested the way CoG has, but is inline with the "make it personal" approaches.)

6. Lock Architects in a Room

Dr. Gary McGraw has pointed out that "locking architects in a room and investigating everywhere they disagree" can be an effective technique.[1] Implicitly, there are also security experts listening. This works because it elicits assumptions about designs and reveals where the "seams" in the software are likely to be. This approach has an interesting property, showing security experts where disagreements have happened, while potentially shifting documentation work to those exports. The prep work for this is mostly centered on calendaring and ensuring your architects start the meeting in bad moods.

7. Card Games: Elevation of Privilege

Elevation of Privilege (EoP) is a physical card game that helps draw developers into threat modeling by presenting it within the framework of a game. Each of the 84 cards features a hint about a threat to a system. The cards are organized into six suits, and each suit is aligned with the STRIDE framework. The game includes instructions and a flowchart. (There are also privacy variants which either add a privacy suit or replace STRIDE with another approach.) For more on these, please see *https://shostack.org/games/elevation-of-privilege.html* or (Shostack, 2014)

1 He has also suggested the addition of a bottle of scotch.

8. Use Case Approaches

There are a variety of approaches written up under the label of 'misuse cases' or 'abuse cases.' They share the idea that you write up something approximating a use case. Some frame with the attacker succeeding, others with a defender succeeding. I am not aware of any being used consistently within an organization, never mind an available collection, and therefore will not discuss the approach further.

9. Pattern Libraries

Pattern libraries that correspond to architectural patterns can be used as a way to shortcut the "What could go wrong?" and "What are we going to do about it?" questions. The concept of these patterns is that once you have threat modeled an architectural pattern, like single factor authentication on a web app, for example, then the threats and countermeasures identified can be added to a risk pattern library. The next time a new system is analyzed that also employs single factor authentication on a web app, the pattern can be looked up and used, instead of redoing the whole threat model from first principles.

Pattern libraries involve substantial up-front effort with the hope that there will be ROI. There are both internal libraries and commercial efforts, representing different levels of effort.

Libraries can be considered the threat modeling equivalent of choosing to use a software library when building a system, as opposed to building it from scratch. IriusRisk is a good example of a tool that implements this principle with libraries that include both Threats and Countermeasures. Dell has publicly spoken about their custom internal library at RSA Conference 2010.

Measuring the Approaches

There are a variety of criteria on which these approaches can be compared or even judged. Again, a list is followed by explanation and discussion.

- **License**
- **Availability of a formal definition and instructions**
- **Time involved:**
  - **Training for participants**
  - **Prepwork by a technical specialist**
  - **Effort per sprint or iteration**
- **Value from the methodology: is it structured, systematic or comprehensive?**
- **Some "intrinsic" quality of the output**
- **Output effectiveness/impact**

License

Is there a formal license?

Of the methods, only Elevation of Privilege is formally licensed. Microsoft makes it available with a quite liberal CC-BY-3.0 license. The lack of licensing probably doesn't matter with methods this simple.

Formal Definition

Is there a formal definition of tasks to complete with defined inputs and outputs?

At the lightweight end, a formal definition would be difficult to craft. Perhaps there's value in mentioning that it's important to say, "How would you?" For the same reasons that agile teams spend time talking about what makes a good stand up meeting, a crisp definition can help make for good threat modeling.

A literal definition of Security Fictions can easily be extracted from the paper (Merrill, 2020), but it is not an instruction sheet. It is harder to extract the definition from the CoG paper. EoP is available with instructions. Only EoP is formally versioned, with changes visible in github.

Time Required

As shown in Table 1, the time requirements can be directly compared.

Method	Training	Prep	Per execution
"What can go wrong"	None	None	Minutes
Make it personal	None	None	Minutes
Fortunately/Unfortunately	Minutes	None	Minutes
Security Fictions	None	None/Customize	30 min
CoG	1 hour	None	1 hour
EoP	None	None	1 hour
Pattern Libraries (commercial)	Incorporated into product training	Purchase	Minutes
Pattern library (internally built)	Awareness	Substantial for the creators	Minutes

Table 1: Training, Preparation and Execution Times

Value from the Methodology

Does the methodology provide for consistency or assurance?

CoG and EoP each provide a structure. Both provide a systematic approach to the analysis, although the approaches are very different. EoP's diagram creation step provides a higher likelihood that the system model is "comprehensive," and provides a record of what system was considered. The other methodologies do not provide structure that promote either consistency or assurance.

Output Quality: Predictability vs Effectiveness

We can look at the quality (or qualities) of the output produced by the methodology as "What does the methodology produce?" and "Does the output lead to action?" Separating these aspects may be surprising, but consider the same output from a threat modeling methodology, delivered to two different teams. The first team has management who regularly prioritizes security, the second does not. The output will have a different impact. Let's compare again, this time with output in the form of either a set of bugs, or a well-written consulting report with engaging abuse stories ... a team with security knowledge might embrace the first, while a team with less security knowledge might get more results from the second. So we can evaluate if a methodology produces consistent output, and we can evaluate if that output is valued by an organization.

Output Predictability

The set of "What can go wrong?" approaches will result in less predictable outputs because no guidance is given. Security Fictions, in providing attachment points, will lead to stories that tie to the prompts, and here, time spent on prep and variation may dramatically alter the results. Center of Gravity has a focus on the "core asset." Elevation of Privilege results in a list of issues to be further investigated. Pattern libraries are intended to produce identical output each time they're used.

In my experience, approaches focused on a core asset tend to lose sight of auxiliary systems, and decisions made early will have a directional effect. EoP's effectiveness is tied to the quality of the diagram used, and the willingness of participants to improve it as they go. The instructions are silent on such improvements.

Slow, Expensive and Excellent

Why would anyone go for something that's slower or more expensive? Ask anyone who opts for a fancy dinner over a trip to McDonalds.

Even if fast and cheap are good, they might not be all that we need. We might want records that can be shared with customers or regulators. We might want a higher degree of assurance. In this sense, it's worth thinking about the relationship of the fast, cheap, and good model to the approaches, which frankly, are slower and more expensive.

Further, in practice, there is often a hope that we can be both efficient and thorough at the same time. This lovely turn of phrase is from Erik Hollnagel, who wrote, "If anything is unreasonable, it is the requirement to be both efficient and thorough at the same time — or rather to be thorough when with hindsight it was wrong to be efficient."

We can hope that fast, cheap (efficient) approaches may show us where thoroughness is called for. Once engineers start down the path, if they feel respected when they ask questions, they may begin to raise questions to help you identify where you need to go deeper. But many organizations implicitly declare efficiency is the priority. If people dig deeper in a few places and don't happen to find problems, then that may discourage further deep investigations.

A Flow View of Threat Modeling

There's a concept of 'flow' that has long informed my work in threat modeling. This section introduces it briefly, and demonstrates how flow can help us contextualize the fast, cheap, and good approaches in this paper.

What is Flow?

Psychologists refer to the concept of Flow as "While in this mental state, people are completely involved and focused on what they are doing." The originator of the idea, Mihály Csíkszentmihályi, talks about a balance between activity and challenge, and arranges them into a graph, shown in Figure 2:

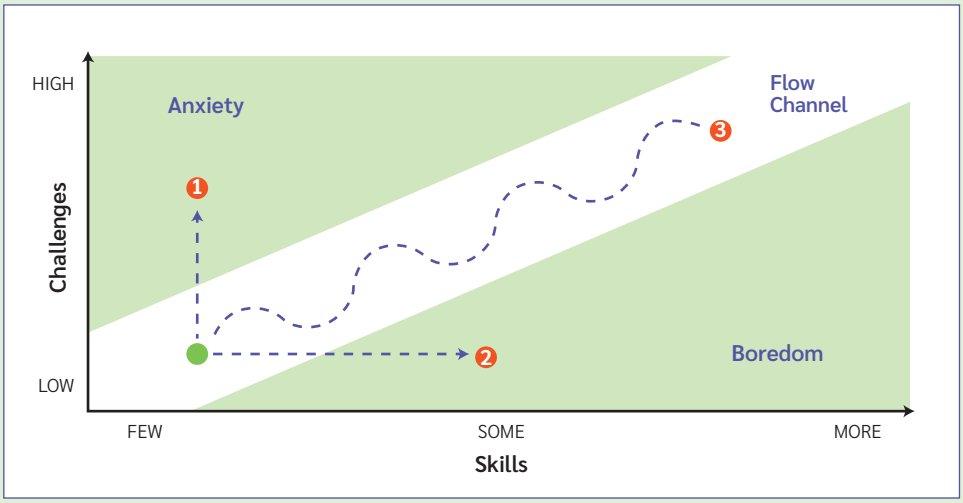


Figure 2: The Flow Channel (Redrawn from *Flow: The Psychology of Optimal Experience*)

The idea is simple but powerful. When someone is embarking on a new challenge, they don't yet have the necessary skills. They can engage with simple problems, but more challenging problems will cause them anxiety (see #1 in Figure 2). As their skills develop, the same low-challenge problems that were engaging become trivial, and they become bored (#2).

As an aside, arguments between peers produce a flow state for many architects. It's important to realize that this is not universal, and some people who might otherwise be good architects are repelled by the practice. If we want a more diverse community of architects, relying less on argument will help us get there. Therefore, the question becomes, if you need to improve skills, how do you do so?

It's reasonable to think of the tools as applicable with various skills and challenges.

A FLOW CHANNEL FOR THREAT MODELING

There are at least three arguments for fast, cheap, and good approaches, including high uncertainty about the value of deeper threat modeling, or limited time or skills. The flow approach is most illustrative in the context of limited threat modeling skills.

The below is intended to be illustrative, not a commitment to the placement of any particular tool in the graph.

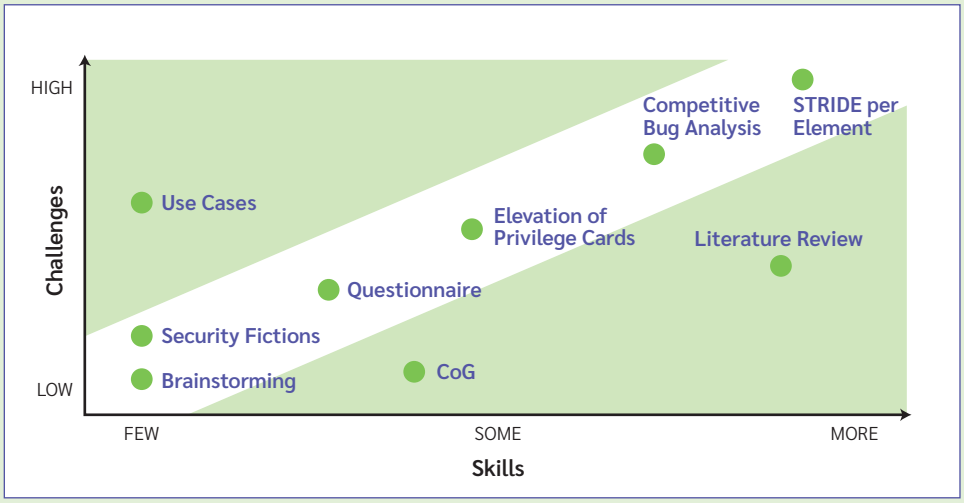


Figure 3: A Flow Channel View of Fast, Cheap Approaches

Ambient Information Can Be Exploited

Bringing consultants "up to speed" is one of the elements that makes threat modeling expensive, as is documenting that information. That's true for both external consultants, or even the security team at a large company, acting as internal consultants. Consultants need to document to test their understanding, and also to memorialize their understanding in case of later disagreements. This is especially true with external consultants.

Keeping Going

If we think of the flow channel as a learning path, then we need to avoid some traps:

- "The way to threat model is..." inhibits our ability to adjust our approach
- The compliance checkbox or crazy long lists
- Thinking of threat modeling as a "one and done," rather than a skillset to bring to bear

Conclusion

This paper's title is a reference to the old engineering rubric: "Fast, cheap or good? Pick any two!" I am sure some readers will take issue with the claim that fast and cheap have value. I stand by that claim, and do not think it must call into question the value of approaches which are more rigorous, in depth, or mathematically grounded.

If there is a practical question of "no threat modeling at all" or "some threat modeling", then a collection of fast and cheap approaches may get organizations to start. If exploiting ambient information facilitates broader involvement, that may substantially reduce cost while maintaining quality.

Approaches to threat modeling that have the engineering teams do the work are likely to be substantially cheaper than consultant-centered approaches. Doing so requires that we find approaches that engineering teams can use today, and that we can grow those teams into approaches that fully meet the needs of the business.

How Can We Help?

This white paper is all about fast, cheap and good ways to get started threat modeling. We know, white papers are supposed to make you anxious about a problem you didn't understand and include a heavy subtext of "buy our stuff." Sorry if this paper is a total failure at that. (Not sorry.) We're optimistic, and hope that this paper is simply helpful.

If you need our help

- Moving these to more structured, systematic or comprehensive approaches;
- Managing the culture change that can come with threat modeling;
- Coaching, training, or other help,

please get in touch: ***adam@shostack.org*** or +1-917-391-2168

We encourage you to follow what we're doing by joining: ***Adam's New Thing*** mailing list, and follow Adam on ***Linkedin***.

SHOSTACK + ASSOCIATES

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.

ACKNOWLEDGEMENTS

This paper has a few important origins. One was conversations over the years with Stephen De Vries, founder of IriusRisk. He has seen the need for lightweight methods for longer than I have. Another was a conversation with Nick Merrill, which started with the Fictions paper, and for which I drew an early version of Figure 3.

The idea of making threat modeling more lightweight has many important antecedents that EMC has long invested in a threat library. (Danny Dhillon spoke about it at RSAC 2010). Izar Tarandach has spoken of the need to "threat model every story."

Many people, including Kate Burnham, Irene Michlin, Mario Platt and Morgan Roman have provided useful feedback on various drafts.

When I was getting ready to launch Elevation of Privilege, I wanted to use the tagline, "The easiest way to get started threat modeling," but a wise lawyer at Microsoft said, "You can't say that. Maybe there are easier ways!"

REFERENCES

Csikszentmihalyi, M. *Flow: The Psychology of Optimal Experience (Vol. 1990).* New York: Harper & Row.

Klein, Gary A. *Sources of Power: How People Make Decisions.* MIT press, 1999.

Merrill, N. (2020). *Security Fictions: Bridging Speculative Design and Computer Security*. DIS 2020 – Proceedings of the 2020 ACM Designing Interactive Systems Conference, 1727–1735. *https://doi.org/10.1145/3357236.3395451*

Shostack, A. *Elevation of privilege: Drawing developers into threat modeling.* In 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14). 2014. *https://www.usenix.org/system/files/conference/3gse14/3gse14-shostack.pdf*

Stevens, R., Votipka, D., Redmiles, E. M., Ahern, C., Sweeney, P., & Mazurek, M. L. (2018). *The Battle for New York: A Case Study of Applied Digital Threat Modeling at the Enterprise Level*. 27th USENIX Security Symposium, 621–637. *https://www.usenix.org/conference/usenixsecurity18/presentation/stevens*