


Who Are “We”?

Power Centers in Threat Modeling

Adam Shostack¹ 
shostack@uw.edu

University of Washington and Shostack + Associates

Abstract. I examine threat modeling techniques and questions of power dynamics in the systems in which they’re used. I compare techniques that can be used by system creators to those used by those who are not involved in creating the system. That second set of analysts might be scientists doing research, consumers comparing products, or those trying to analyze a new system being deployed by a government. Their access to information, skills and choices are different. I examine the impact of those difference on threat modeling methods.

1 Introduction

Threat modeling is a collection of techniques for proactive security analysis of systems. The consensus industry methods are based on Shostack’s Four Question Framework (“What are we working on, what can go wrong, what are we going to do about it, did we do a good job?” [12]) This paper builds on work by feminist scholars and activists to look at the influence of the target users on industry methods. In other words, the use of ‘we’ in the framework was a choice that ignored power dynamics. I suggest a threat modeling approach designed to helping people analyze a system they were not involved in creating. (Terms like ‘customer’ or ‘user’ are not broad enough. Systems are often imposed, such resume scanners, traffic cameras or border security.) This paper is written in the first person singular, and avoids the authorial ‘we’ convention for clarity.

1.1 Background

There are two main senses in which the term *threat model* is used. The earlier is ‘What’s your threat model, and ‘random oracle’, or ‘a network attacker,’ could be complete answers. The term was adopted into ‘a model of threats,’ in the sense of an abstraction of possible future harms (spoofing, tampering, etc) as applied to a system under development [5], and was deployed in informal practices such as whiteboard discussions about system security. These were adopted by [5], [6], [16]and others into increasingly structured methodologies. The first sense is answered by a few words, the second sense is often answered with a set of diagrams, lists of threats and mititagations and tables interlinking them.

These methods were used to develop products which were ‘secure by design.’ The analytic techniques were designed to work quickly, with limited training, in

an environment where there was no requirement for mathematical analysis or proof.

Between 2006 and 2014, the author created a set of questions to help frame and organize threat modeling. Originally framed with ‘you’, they were later rewritten in the ‘we’ form. That form explicitly centers the perspective of the team, division and company doing threat modeling. As the work was funded by Microsoft to improve Microsoft products¹ this was not seen as an issue by the author at the time.

I’ll refer to these approaches as ‘analyst’ threat modeling and ‘creator’ threat modeling, respectively. The first helps us understand the relevance of an attack or analysis, the second helps anticipate and thus prevent them. Interestingly, the question ‘what are we working on’ can be applied in either, while the techniques for answering it change. In analyst modeling the analyst will start identifying components, data flows, and scope from a purely observational perspective. In creator modeling, documentation, source code, and even conversations with decision makers are normal.²

2 Critiques

Sets of scholars and practitioners sought to bring creator threat modeling techniques to the analyst perspective. These included those writing under an umbrella of feminist cybersecurity and others focused on the needs of activists. In doing so, they exposed biases and limits of the techniques. Others lacked either access to the developers, or technical knowledge of software creation or operations.

2.1 Survey of Critiques

Freed et al consider the case of ‘interface-bound attackers,’ who use the product as intended and cause harm[2]. Spammers, bullies, trolls, phishers, fake reviewers, posters of deepfakes are not breaking the traditional rules of computer security (A professional society and workshop on trust and safety has arisen to support this community; Stamos points out that these sorts of attacks caused the vast majority of harm when he was CSO at Facebook[15].)

Slupska et al attempted to threat model a smart lock, and in particular analyze it for issues of intimate partner violence (IPV) [10]. The project exposed first, that creator perspective is limited, and second, that the techniques of creator threat modeling don’t help an end user understand the problem.

Typical creator techniques assume the existence of a TCB and a trustworthy administrator. IPV perpetrators often take control of a user session, and may closely monitor systems for administrative changes. I’ll call the the administrator of the smart lock Alice, and aperson abusing her, Bob. Bob may (a) have her

¹ And incidentally but intentionally the third party products that interact with them.

² A distinction that I failed to note in a recent corporate whitepaper [?].

password, (b) demand administrative access. If Alice tries to restrict Bob’s access to the lock, he may be notified. If Bob is the administrator and Alice uses physical access to the lock to reset it, Bob may be notified or asked to approve the change. So how should the lock company design an access control matrix? They can focus only on the administrator who can create accounts or change their permissions, and regular users who can lock or unlock the door. But the use case of two users with the administrative password is unusual for computer security, and our normal response of ‘set an acceptable policy’ may lead to a literal slap in the face. The complexity and effort of enumerating attacks may inhibit creators from investigating or recording them. If they are analyzed, the complexity of addressing them may be declared to be an ‘edge case’ or otherwise de-prioritized.

Additionally, creator threat modeling methods like STRIDE or kill chains don’t help Alice (as an analyst) discover or reason about these problems.

Space limits our ability to discuss other critiques such as [7], [1], [14] or [9].

2.2 Analysis

We can consider possible threat modelers in a space defined by technical knowledge and system knowledge.

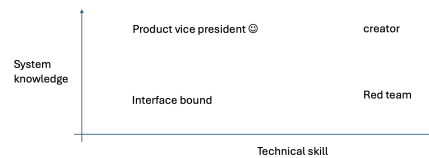


Fig. 1: A threat modeling space

Social Milieu Creator practices were simplified to achieve scalability, which is to say ‘possible to achieve by those with an hour or two of training.’ The company recognized that design choices were being made unknowingly by developers and wanted them to be able to perform analysis. (There were several downsides to this, including perhaps insufficient recognition of the quality tradeoffs between experts, and a focus on reviews and documents over skills and engagement.) These circumstances informed the creation of threat modeling methodologies appropriate for use by technical experts to analyze systems with which they were highly familiar, or where they had access to the developers or code.³ Early versions of the Four Question Framework used ‘you,’ as in “What are you working on?”, and that was intentionally changed to ‘we’ to be more collaborative.⁴

This approach can be (and was!) contrasted with Anderson’s educational approach. Colleagues argued “We can’t require people to get a PhD in security,” or

³ It is tempting to say easy access, but that ignores the sometimes contentious inter-team relationships.

⁴ Other important work included that of Kohnfelder and Garg and Swiderski and Snyder. A slightly fuller history is available at [4].

“read a 500 page book.”⁵ Anderson expected people to think critically and well, Microsoft needed to provide a process or methodological set of steps they could follow. The focus on process was seen as a requirement for scaling, supported auditability, and was a response to a frequently expressed “just tell me what you want me to do.”

The approach can also be contrasted to the sorts of threat modeling done by spies, attackers, bug bounty participants, or even academics who start with limited knowledge of a system, but a great deal of technical knowledge, possibly including security knowledge. They may be willing to dedicate more time, or they may see a single bug as a sufficient result. (The ‘single bug’ goal can be contrasted with the need for creators to build a secure system.) Their technique choices and investment of energy will be shaped by those circumstances.

Technical Knowledge Microsoft hires quite selectively into the broad category of ‘software engineering’ roles, historically including development, test, and program management roles. The least technical of these, program management, requires deep technical and domain knowledge. As a result, threat modeling methodologies did not need to account for participants without technical skills.

Knowledge of system Threat modeling methodologies were developed for internal use by Microsoft product teams who were asked to engage with product security experts. Cost and effort of knowledge transfer less important because these experts would often embed for periods between weeks and years. Even so, those experts might not be briefed on features for many reasons. Those could include people doing feature work didn’t see security implications, or a desire to avoid security so an insecure feature could ship. Reviews were also conducted by highly skilled experts, and likely closer to what’s called product red teaming.

3 Threat Modeling ‘for the rest of us’

This section presents a simpler approach to threat modeling, designed for use by those with lower technical skill and less knowledge of a system. (The term is used for clarity, not as a judgement.)

The Framework:

1. What have they delivered?
2. How will it hurt us?
3. Can we protect ourselves?
4. Should we even use it?

⁵ Noting that the first edition of *Writing Secure Code* was 501 pages including introduction, and had a quote from Bill Gates, ‘Required reading at Microsoft’ on the front cover.

These questions are designed to be answerable (although finding the answers may require uncommon skills) and aligned with the Four Question Framework. The alignment is intended to make the questions easily remembered. Next, I present explanations of each question and structured approaches to them.

3.1 What have they delivered?

The question of what a software package is has become more complex since the days of software delivered on floppy disk. Much software, including IoT and mobile apps are delivered as ‘web apps’ (with associated back ends).

Assuming that people have a general model of software that runs on their device, allows us to use a simple model of ‘local’ and ‘cloud.’ People believe that data on their device is private and more secure, a belief created or reinforced by both intuition, and marketing like “Your fingerprint never leaves your device.” Questions that can be asked by those with low technical skill might include:⁶

- Does it work without internet access?
- Can you use it without creating an account?
- Can you use it without giving it a working email address?
- What does the privacy policy tell us?

Reading and summarizing privacy policies requires determination, and may require some skill, and such analysis can result in highly accessible lessons, such as “We share data with our 1400 partners.”

Those with more technical skill can consider use of browser plugins like No-script or tools like Wireshark, and going deeper, analyst methods start to resemble those used by security researchers, rising to enumerating libraries, using a debugger or even logic probes or electron microscopy to analyze a chip or device. Firmware and mobile apps can be downloaded and prised open, and freely available code even provides the permissions the library uses.[17]

3.2 How will it hurt us

Creator-oriented threat modeling may draw on frameworks like STRIDE to structure an analysis, but that requires technical skills.[11]. A simpler set of threats, such as what does it learn and where does it send it may be helpful, but even local processing may be against the interests of a user. For example, does it show ads? Will it change function on update?

3.3 Can we protect ourselves and Should we even use it?

The history of general-purpose computing is a history of modifying software to serve local needs, including security. Three of the top ten Chrome extensions are adblockers [18]. The rise of restricted computers (phones, iot) has made people more secure against malware and less able to take control of their experience. In this situation, the question of should we even use it becomes more difficult.

⁶ Usability testing these ideas is obviously important.

4 Conclusion

The author regrets implying that threat modeling techniques are universal. Both people's depth of technical skills and their involvement in the creation of a system influence how they may threat model.

Acknowledgements

Julia Slupska and Leonie Tanczer helped me understand the problem they were grappling with. Loren Kohnfelder and Kim Wuyts provided helpful feedback on drafts. Over decades, Ross Anderson's writings have profoundly influenced my own. I mourn his loss and hope to contribute this small bit to the celebration of his legacy.

References

1. Electronic Frontier Foundation, *Risk Assessment (Threat Modeling)*, June 24, 2019, <https://www.eff.org/files/2020/01/06/threatmodeling-onepager.pdf>
2. D. Freed, J. Palmer, D. Minchala, K. Levy, T. Ristenpart, and N. Dell, *A Stalker’s Paradise: How Intimate Partner Abusers Exploit Technology*, in Proceedings of the 2018 ACM Conference on Computer Supported Cooperative Work and Social Computing, 2018, pp. 163-177.
3. Farmer, W. R. and Venema, A., *Improving the Security of Your Site by Breaking Into It*, message to comp.security.unix, December 1993, <https://cyberwar.nl/d/1993-FarmerVenema-comp.security.unix-Improving-the-Security-of-Your-Site-by-Breaking-Into-It.pdf>
4. A. Shostack, *Understanding the Four-Question Framework for Threat Modeling*, whitepaper, November, 2024, <https://shostack.org/whitepapers>
5. L. Kohnfelder and P. Garg, *The threats to our products*, Microsoft Interface, Microsoft Corporation, vol. 33, Apr. 1999.
6. M. Howard and D. LeBlanc, *Writing Secure Code*, 1st ed. Redmond, WA: Microsoft Press, 1999.
7. B. Kazansky, ‘It depends on your threat model’: the anticipatory dimensions of resistance to data-driven surveillance, *Big Data and Society*, January 29, 2021, DOI: 10.1177/2053951720985557
8. Schneier, B., *Attack Trees: Modeling Security Threats*, Dr. Dobb’s Journal, December 1999.
9. Levy, K. and Schneier, B., “Privacy Threats in Intimate Relationships”, *Journal of Cybersecurity*, Vol 6, Issue 1, 2020, available at: <https://academic.oup.com/cybersecurity/article/6/1/tyaa006/5849222>.
10. J. Slupska and L. M. Tanczer, *Threat Modeling Intimate Partner Violence: Tech Abuse as a Cybersecurity Challenge in the Internet of Things*, in *The Emerald International Handbook of Technology Facilitated Violence and Abuse*, 2021, pp. 663–688.
11. Shostack, A., *Threats: What Every Engineer Should Learn From Star Wars* (Wiley, 2023).
12. Shostack, A., *Threat Modeling: Designing for Security* (Wiley, 2014).
13. Shostack, A. *Understanding The Four Question Framework for Threat Modeling*, corporate whitepaper, <https://shostack.org/whitepapers>, 2024.
14. L. Sterling, *Practitioners of Civil Resistance: Assess Your Cybersecurity through Threat Modeling* March 22, 2018, International Center for Non-violent Conflict, https://www.nonviolent-conflict.org/blog_post/practitioners-civil-resistance-assess-cybersecurity-threat-modeling/
15. A. Stamos, *Stepping Up Our Game: Re-focusing the Security Community on Defense and Making Security Work for Everyone*, YouTube, 2017. Available: <https://www.youtube.com/watch?v=YJOMTAREftY>
16. F. Swiderski and W. Snyder, *Threat Modeling*, Microsoft Press, 1st ed., 2004.
17. J. Xin, *app-third-party-library*, GitHub, 2024. Available: <https://github.com/xinjin95/app-third-party-library> [Accessed: 22 Nov 2024].
18. M. Zeunert, *Chrome Extension Statistics: Data From 2024*, blog post August 29, 2024, <https://www.debugbear.com/blog/chrome-extension-statistics>