Security Fictions: Bridging Speculative Design and Computer Security

Nick Merrill
University of California, Berkeley
Berkeley, California
ffff@berkeley.edu

ABSTRACT
This paper begins with an observation: that threat identification is an intrinsically speculative practice. It requires imagining possible futures. Drawing on methods from speculative design, this paper presents an improvisational role-playing game designed to help software developers identify security threats. It deploys this game with seven software developers, who used the game to successfully identify diverse threats in their software. The insights from this deployment motivate future work on both the game itself and on organizational accounts of security. I call on the design research community to continue to apply its methods and perspectives to computer security, locating threat identification itself, like all speculation, as a site of social and political power.

Author Keywords
security; design research; speculative design

CCS Concepts
•Security and privacy → Social aspects of security and privacy; •Human-centered computing → Empirical studies in HCI;

INTRODUCTION
Securing software requires, among other things, identifying threats that have not yet been exploited. This process, known in computer security as *threat modeling*, is famously tricky [32]. Existing practices are time-consuming for non-specialists to learn [35, 10]. In addition, they may systematically miss certain types of threats, especially those that apply differentially [28] across relational characteristics (e.g., gender, race, age, disability) [16, 27, 21, 19, 33, 8].

In addressing the shortcomings of existing threat modeling practices, this paper builds a bridge between speculative traditions in design research and threat identification in computer security. My core observation is that threat identification is an intrinsically speculative practice: it requires imagining possible futures. Motivated by this observation, I apply speculative techniques from design research to threat identification, contributing a novel practice: an improvisational role-playing game called Security Fictions. My primary design goal was to produce an easier practice for non-specialists to learn, expanding and diversifying participation in the practice of threat identification while sensitizing practitioners to sociotechnical security risks.

This paper motivates and describes Security Fictions. It reports on an exploratory deployment of the game among seven software developers, five of whom did not specialize in security. This deployment, which sought primarily to validate the game's design, surfaced a few, key insights. First, developers enjoyed playing the game, and used it to come up with realistic threats. Second, some of the threats non-security specialists devised were not strictly technical (for example, a UX designer discovered a social attack that relied on a misleading feature in a user interface). Third, developers did not always know to whom they could go with security threats: limited visibility into the organization's practice of security hindered developers' ability to raise security concerns. I discuss several avenues for improving Security Fictions and for mobilizing the game in future work (for example, investigations on how organizations split up the work of software development into "security" and "non-security" tasks).

This preliminary investigation aims to invite design researchers into the fold of computer security: to further mobilize design research methods to produce workable security interventions. Doing so may better include diverse perspectives in the practice of security. At the same time, the design research community should continue to investigate how threat identification, like other speculative practices, configures its practitioners into positions of social and political power [34]. Expanding what "counts" as a security threat may help tech workers raise dissent under a broadened umbrella of security concerns.

BACKGROUND

Threat modeling
Given the diverse goals of software and the diverse contexts in which software tools are used, no standardized "security checklist" can adequately cover all relevant risks. As a result, *threat modeling* seeks to imagine possible threats and describe their

impact through open-ended speculative processes [32]. Systems such as STRIDE and OCTAVE help developers search for attacks in pre-defined categories (e.g. spoofing, tampering with data, etc). Board and card games sometimes assist in the speculative process of threat modeling [31, 10, 18]. For example, the game *Elevation of Privilege* stemmed from Microsoft's development of the STRIDE methodology [31]. Even the US Central Intelligence Agency (CIA) developed a (now-declassified) game for helping developers generate threat models [25].

However, existing practices for threat modeling fall short in two, key respects:

1. **They are difficult for non-specialists to learn** [35, 10]. Software developers who are not specifically trained in computer security might write more security software if they could better participate in threat identification. However, existing practices are aimed at security specialists.

2. **They may systematically fail to identify particular types of threats**, particularly those that arise from social factors or relational characteristics (e.g., gender, race, age, disability) [28, 16, 27, 21, 19, 33, 8].

In one chilling example of the latter limitation, Freed et al found that many applications allow low-tech or "UI-bound" attacks commonly deployed by domestic abusers. Threat models from traditional practices tend to focus on anonymous, remote attackers without personal relationships to the victim, missing the technically unsophisticated threat posed by attackers who know the victim well [16].

Complicating matters further, cybersecurity jobs are growing more rapidly than the number of security specialists who can take these roles [4, 26]. Designing more inclusive threat identification practices represents a step toward not only capturing more diverse threats, but also filling this critical gap in the security talent pipeline.

Drawing on speculative practices: Enacting speculations

How can we overcome the limitations of threat modeling? Specifically, what kind of practice could help non-specialists identify security threats more readily, while simultaneously surfacing the sorts of sociotechnical threats that existing practices miss? To help answer these questions, I look to speculative practices in design research for inspiration in addressing these concerns. Motivating this move is the core observation that threat modeling is an intrinsically speculative practice. It shares with speculative practices in design a fundamental outlook toward the utility of imagining possible futures. To quote Dunne and Raby, "by speculating more... we can help set in place today factors that will increase the probability of more desirable futures happening. And equally, factors that may lead to undesirable futures can be spotted early on and addressed or at least limited" [12, p. 6]. This assertion about design applies to threat modeling as well: threat modeling generates possible futures such that action can be taken in the present to push toward futures that are desirable.

In order to broaden the speculations that threat modeling produces, design research has produced its own interventions into security practice [9, 10, 19, 5, 1]. For example, Friedman and Hendry's *Security and Privacy Threat Discovery Cards* aimed to broaden security's concerns to include sociotechnical considerations such as political structures, social relationships, and environmental concerns [19]. Past work in design research has also used collaborative storytelling practice to produce security "premoretems" [15], or by outline "value scenarios," speculative artifacts for identifying unforeseen risks [24]. Akmal and Coulton's (2019) work applies Foucauldian notions of space and rhetoric to Internet of Things privacy through a board game. These practices effectively broaden the concerns of past threat modeling games, and engage a broad variety of software developers.

However, one key limitation of these studies is that their play is abstract, lacking characters to make play memorable, and lacking enacted practices to engage stakeholders actively in threat identification. These properties have also also been criticized within security as they apply to threat modeling, a critique partially addressed by the practice of *redteaming* [38]. However, like threat modeling, redteaming requires security-specific expertise, a barrier to entry for many developers. It also focuses on technical threats alone, missing a primary benefit of the design research interventions described above.

To synthesize this work, I draw the threat modeling practice of redteaming together with the design research practice of speculative enactments. Elsden et al.'s Speculative Enactments engage research participants in improvisational play with real-world consequences [13]. As such, it provides a useful starting point for producing a novel, more sociotechnical take on redteaming and other threat identification practices. Merging redteaming and speculative enactments, I draw on the active, enacted, improvisational nature of both practices. I also look to the easy-to-learn, participatory style of speculative enactments as inspiration for lowering barriers of entry to non-specialists. Finally, I look toward the broad, sociotechnical frame of speculative enactments and other, related works from DIS, CHI and beyond [9, 10, 19, 5] to expand threat modeling's existing concerns. Ultimately, I synthesized these requirements into the design of a game, which I describe in the following section.

DESIGNING THE GAME: SECURITY FICTIONS

Building on redteaming and speculative enactment, the game needed to carry real-world stakes: finding security threats in real software. These heightened consequences should drive engagement [14, 38]. Expanding on redteaming, I believe that speculative enactments can broaden speculative artifacts beyond the technical. As such, I decided that the game should produce not threat models *per se*, but rather narrative scenarios similar to design fictions [2]. Thus, I name our enacted game *Security Fictions*, a reference to the narrative, sociotechnical speculation the game seeks to engage. As far as I am aware, this is the first paper to bring speculative enactments into the workplace as a professional practice.

For an initial prototype, I decided to eschew specific props such as cards, dice, or board games. Doing so allowed me to focus on the speculative enactment itself. I "staged" *Security Fictions* as a one-on-one conversation between the researcher

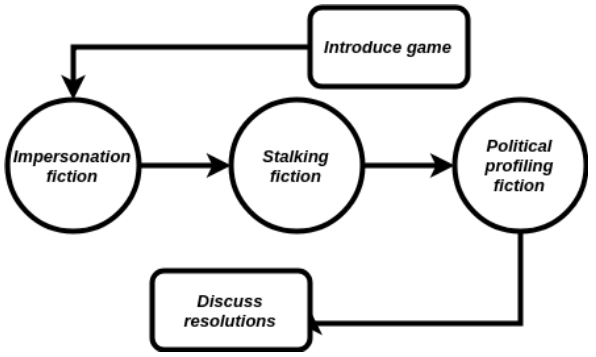

Figure 1. The flow of *Security Fictions* gameplay. Developers attempt to identify security vulnerabilities pertaining to three fictions in their conversation with the researcher.

and the developer. Following Elsden et al., I assured that the developers were fully "in" on the enactment, understanding it as fictional [13]: the researcher asked developers to partake in a role-playing game, and that their play would be used to study how they think about security with respect to the systems they build and maintain. Developers played the game by simply discussing their solutions with the researcher. They were invited to take notes, draw, or use software to demonstrate concepts as appropriate.

Researchers staged the intervention by sitting with the participant in a quiet room as if to tell them something in confidence. The researcher would tell the developer that s/he had a "client," who wished to perform some nefarious tasks with the platform. This client was fictional, and referred to no participant in the study: I used this term to anchor speculation in the motivations of a hypothetical actor. In the interest of staging the interaction faithfully, the researcher remained "in character" throughout the gameplay. To begin the game, the researcher used the following prompt:

> I am representing a client, who would like to perform some tasks with your platform. Leave your ethics at the door and tell me how my client can achieve these tasks, on their own, without your help.

A few key considerations guided my design for this prompt. My request to "leave [their] ethics at the door" was meant to signal that the scenarios may involve malicious actions. Specifying that the client would need to achieve tasks "on their own" signaled that threats should focus on outsiders (i.e., attackers should not require insider access to code or data). Finally, my prompt did not require developers to roleplay as a different character or person; they would play the person they really are, knowing the things they really know, and would use that knowledge to help resolve the client's request.

I asked developers to provide "resolutions" to the fictions: any attack the participant was reasonably certain would satisfy the fictional client's request. Rather than describing "solutions," the word "resolution" aimed to communicate to participants that there may exist multiple possible ways of satisfying the fiction; that the prompt is not a problem to be solved, but an open-ended invitation for play. Since the companies from which I recruited developers (and the software they produced) were highly diverse in their purpose, scope, and target users, imagined attacks and contextual their significance vary greatly; I allowed the developers themselves to be the ultimate arbiters of what resolutions "counted" (i.e., seemed practical and impactful). I counted as a resolution any response that the developer believed would satisfy the fictional client request.

The fictions

The game centers around several key "fictions:" speculative scenarios to prompt the game's roleplay. In the future, I hope to provide developers with tools to generate their own scenarios. However, to test the game's core mechanic of roleplay, I crafted pre-made scenarios to spur play. (I return to developer-generated scenarios in Section 6.2.3).

In crafting scenarios for the game, I looked toward recent headlines and prior work from CHI in which security issues went undetected with serious human consequences. In my review of recent news and recent literature, three themes emerged: impersonation [3, 16], political profiling of users on Internet platforms [37], and stalking mediated by digital tools [16, 23]. Following this work, the game centered around three fictions: one on impersonating a user, one on profiling the political beliefs of users, and one on finding a user's geographic location. I designed these fictions to span (social, political) concerns, drawing from contemporary events, and prior research on limits to security methods. Each aimed to reflect different notions of computer security, which in turn reflects security as an essentially contested concept (as discussed in the Background section, above).

Threat Fiction 1: Impersonation

> *"Our client wants to impersonate another user."*

This scenario maps to a traditional, computer-science notion of "Bob-and-Alice" security [11, 33]. In cryptographic systems, users should be able to verify the provenance of a message. However, cryptographic threats are not the only ways this scenario can play out—recent incidents have seen non-cryptographic forms of impersonation for trolling, doxxing and harassment [7].

Threat Fiction 2: Stalking

> *"Our client wants to find the physical location of a particular user."*

This scenario is inspired by Freed et al (2018)'s work on physical stalking perpetuated by domestic abusers [16], a security threat which, their study found, is often not considered in traditional security audits. Although Freed et al. found domestic abusers tend to employ unsophisticated, "UI-bound" attacks, I allowed participants to come up with any conceivable response to the scenario. (Some, but not all of the responses were UI-bound, as I discuss below).

Threat Fiction 3: Political profiling

> *"Our client wants to find all of the politically conservative users."*

This scenario was inspired by political events recent to the writing of this work: Russian intelligence's use of Facebook ads, which allowed them to use users' algorithmically-derived political interests to target political messages (which were not labeled as such) [37]. Again, such threats may evade institutions' traditional definitions of security, while still threatening highly consequential harms if exploited. Since the companies collected various data about users in a range of contexts, I chose the ambiguous term "conservative" to allow participants flexibility in interpreting how the prompt may apply to their particular situation.

PRELIMINARY DEPLOYMENT

This section reports on our preliminary deployment of the game among software developers. My main goal was to understand how players engaged with the game, and whether the game might allow developers to identify threats. Below, I describe my strategy for recruitment, how I ran gameplay sessions and how I analyzed participants' data.

Researcher positionality

My approach to this research is shaped by my personal experience. As a Californian, the industry around computation has reshaped ("disrupted") the regional economy over my lifetime, a transformation in which I was able to participate due to, among other factors, my educational opportunities, my age, being a white male, and being a native speaker of California English. I have worked in software development in both commercial and activist capacities. The various privileges of my positionality, along with the privileges afforded by my institutional affiliation as an academic, helped to award me access to technology companies in the San Francisco Bay Area, and to their employees. While these developers likely do not represent all software practitioners globally, they have historically enjoyed a relatively large impact on software developed for mass consumption [30], and are a group I am personally well-equipped to reach as a perceived "insider."

Recruitment

As security concerns vary depending on what software is meant to do, I recruited software developers from a diverse set of software companies. For consistency, I wanted to use the same game with all of them. To address the tension between my need for diverse software and my need for consistent methods, I applied one criterion to the companies I recruited: all companies' software should have users about whom data is collected and stored. This criterion also motivated the design of the game, as described in Section 3.

Within this scoping criterion, I sought diversity along two categories of interest. First, I sought both large and small companies (measured by number of employees). I expected the size of companies to dictate their ability to donate resources to security, perhaps via specialized security teams. I expected larger companies to have overall better security practices, and their developers to be less likely to find convincing responses for the fictions. Second, I sought companies that did, and did not operate in markets with a high degree of regulation (e.g. healthcare, education, law, finance). I expected that companies operating in highly-regulated sectors would have stronger security practices due to increased regulatory scrutiny, and thus be less likely to identify vulnerabilities from the fictions.

Company	Size	Regulated?	Developer specialty
A	7	No	UX
B	1000+	No	Mobile
C	1000+	Yes	Infrastructure
D	2	Yes	Infrastructure
E	1000+	No	UX

Table 1. Profile of companies in our sample, along with the specialty of the developer at each. (For Companies A and C, our sample included an additional security speciaist). I sought a diverse set of companies across two dimensions. First, how large is the company (in number of employees)? Second, does the company operate in a sector with a high degree of legal regulation?

I recruited developers through local connections in the software development community. I aimed to recruit at least one developer per company. When speaking to developers, I asked them if security was their primary job responsibility, *and* who had received training in security. If so, I asked them to refer me to another developer. The developers I recruited had a variety of specialties. All together, the developers in our pool represented UX programming, backend infrastructure and mobile development as their primary job responsibilities.

In the end, each company was represented by at least one developer who had received no formal training in security, and for whom security was not among their job responsibilities. In addition, I recruited security specialists from Company A and Company C. In total, I recruited seven software developers from five companies, two who identified as women and five who identified as men, with ages ranging from 27 to 37 (Table 1). I anonymized all participant and company names. I aimed to assure that vulnerabilities cannot be traced back to particular companies.

Sessions and analysis

Gameplay sessions occurred in developers' place of work and ran for about one hour. When I first sat down with developers, I began with a brief, semi-structured interview asking about their role in the company along with basic questions about the company's structure. I was particularly interested in whether the company had a dedicated security team; if so, how often a software developer interacted with them and, if not, who was responsible for assuring security. I also asked them about the company's data collection practices, in particular, how the organization weighed the risks of data with its benefits. After the interviews, I transitioned into playing *Security Fictions* with developers, a game I describe further in the following section.

The researcher audio recorded interviews and gameplay sessions for later analysis. After gameplay, the researchers transcribed audio recordings of gameplay sessions. I analyzed transcripts using a grounded theory approach [6], using the constant comparative method of joint coding and analysis to reveal common reflections and themes. I created audio recordings of interactions with participants, including gameplay sessions. I supplemented these audio recordings with photos when necessary (e.g., when participants took notes or demonstrated attacks in their apps). I transcribed audio from interviews, deleting original recordings.

Company	Threat fiction	Threat identified?
A	Impersonation	•
	Stalking	•
	Political	•
B	Impersonation	•
	Stalking	•
	Political	
C	Impersonation	
	Stalking	
	Political	
D	Impersonation	
	Stalking	
	Political	
E	Impersonation	•
	Stalking	•
	Political	•

Table 2. Fictions for which vulnerabilities were identified, categorized by company.

DEPLOYMENT RESULTS

Playing the game

The developers in our sample found a variety of solutions to the security fictions (Table 2). Developers picked up the game quickly, with no training outside of the researcher's explanation of the game through the prompt described above. Resolutions to fictions varied somewhat in detail, but all resolutions gave enough information for the researcher to independently test the proposed attack, if required. Companies A, B and E found at least one vulnerability.

All developers engaged actively in finding resolutions to the fictions. A few became frustrated when they were unable to find an attack for a particular fiction. When posed the political profiling fiction, Company E's developer said, "Crap, I don't think we collect any data that could be used for that." After finding successful resolutions, developers often expressed personal satisfaction. For example, after resolving the Political Profiling fiction, Company A's developer reflected, "That's nice and elegant. That's simple. It would be hard for us to stop that." (I discuss his resolution below). These developers drew on their notions of engineering "elegance" in their solutions. I take these observation as evidence that developers engaged their "professional vision" [20], actively engaged in the speculative process, and motivated to find threats even when they bring possible risks (and labor to fix).

The resolution developers discovered typically did not rely on sophisticated attacks on which threat models tend to focus (for example, credential stealing or elevation of privilege). Instead, attacks emerged in two categories, social and UI-bound [16], discussed below.

UI-bound attacks

The developer at Company A proposed that the attacker simply change their display name, bio, and avatar to match those of the impersonation target. This developer, a UX designer, was able to show in the company's app that the attacker would be indistinguishable from the impersonation target. The developer tried the attack out on his personal smartphone, showing the researcher the feasibility of the attack. In only a few minutes, he was able to impersonate his coworker, and post a (mildly) embarrassing message on a public channel.

A similar pattern emerged at Company E. Company E's developer was confident that their platform did not collect information about location or political affiliation. However, that same developer proposed a method for using the company's developer-facing tools to collect such information, designing misleading user interfaces and deploying them through the company's developer-facing platform.

These episodes corroborate past studies that "UI-bound" vulnerabilities often go unnoticed [16]. UX designers were often well-posed to identify these attacks, even though the UX designers in our study reported that they have never been asked to partake in any security practices. I return to this point in the Discussion.

Social resolutions

Alongside the UI-bound attacks, non-security developers also identified exploits that relied on social engineering rather than technical approaches. At Company B, the security developer participant could not think of a way to stalk users on the company's app. However, an iOS developer invented a simple, social engineering attack. The developer even admitted that she had used this attack before, albeit unwittingly, and without thinking of it as a security issue. She described seeing a woman on the street who had affixed to her backpack "an outlandish number" of what the developer described as "high-fashion pom-pom balls—very recognizable." By noticing this distinctive purchasing habit, the developer was able to locate this user's physical location using data publicly available on her company's platform.

At Company A, a similarly social resolution emerged around political profiling. Company A's security developer did not believe sufficient data existed to profile the political beliefs of users. However, the UX developer I spoke to also produced a social-engineering solution. His solution involved not data scraping (as the security developer's did), but encouraging users to join groups that indicate particular beliefs or interests. Such an attack could easily be leveraged by one user against another, and without violating the application's terms of service.

What do we do now?

After the session, I asked developers to raise any security issues they discovered within their companies. To whom in their organization could they go with the concerns or security issues they surfaced? What could they personally do to help ameliorate the issues they discovered? Developers in Company C, who discovered no attacks, knew exactly to whom they could have reported vulnerabilities they uncovered. Company C's developers reported that their company's executives have long prioritized security and built security teams early. As a result, clear protocols existed for reporting concerns to particular teams.

Outside of Company C, developers at the smaller companies knew whom to ask about particular security concerns, however informal that person's role may have been. For example, Company D had only two employees, but one took personal responsibility for any threats they uncovered. However, he emphasized that he had limited time for finding these issues in the first place, and no system or practice in place to do so. In lieu of a systematic method for identifying or resolving concerns, the security-oriented developer paraphrased Notorious B.I.G. in explaining the company's informal policy: "more data, more problems." He believed that he could save time and bolster security by collecting less data, and adding more collection features when the company had the resources to hire security-specific personnel.

In contrast, at larger Companies A and B, developers outside of security roles did not know to whom to go with their concerns. At Company B, a mobile developer had to use a search engine to check whether the company *had* a security team. She was surprised to find that there were a number of security teams, all with different responsibilities. She described her interactions with security at the company as institutional and impersonal.

> Every quarter, posters go up, "if you see something, say something" type of deal. Once a year, there's a security week, they promote bug bounties. Once every 6 months, send out a thing saying "don't engage with phishing emails." *(Mobile developer, Company B)*

Her interactions with security teams revealed little about how security at the organization was structured, and did not provide her with opportunities to partake in security outside of bug bounties. At two of the three larger companies, however, developers reported that they have never been consulted about security, and felt less responsibility for security than did non-security developers at smaller companies.

> We don't really think about [security] that much, our team doesn't engage with it a lot [...] it's something we're reasonably considered about, but there's a solution already in place. *(UX developer, Company E)*

This developer felt security was someone else's responsibility, even though she did not know which security teams were in place. Her experiences gesture toward a general opacity around security in her company. Not only is unclear how the company does security work, it is unclear *who* does it, and how (or if) other members of the organization are meant to think of it. Bug bounties, which aim to act as incentives, do not give any hints as to where developers should look, or what types of issues might count as bugs.

At Firm A, the employee tasked with managing security balanced the threats discovered against pragmatic concerns. Discussing the threats, he said, "we would love to have that problem, that we were popular enough for people to be fucking with us." He described Company A's main focus as growing its userbase. Security was both secondary to this goal, and perceived as unimportant to its target demographic. The security engineer described testing their application with users, who were prompted to accept or decline various application permissions requests (e.g., microphone or location access).

> In markets we were testing with, they do not give a shit. They don't even read [permissions requests]. They just hit "accept." (Security, Firm A)

While this software developer lamented users' lack of concern for their privacy, his explanation points to structural issues that cause companies to de-prioritize threat identification. These structural issues prevent the implementation of security not only at the organizational level, but also in broader contexts of business goals and perceived market demands. His anecdote suggests that the user-centered design process may itself undermine security goals, indicating to designers and management that security is not a user need and therefore does not merit developer time..

DISCUSSION

The present deployment was highly preliminary. I intend to follow this work with more in-depth studies, most likely expanding the game to allow players to generate their own prompts and threat scenarios. However, this preliminary study does validate the game's core mechanic, providing compelling evidence that developers engage in the game even if they do not have specific training in computer security. Players identified threats beyond the technical, and did so with no prior training beyond the three or four minute introduction given by the researcher.

Two major caveats mark this work. First, while this preliminary study implies that speculative enactments hold the potential to act as "light-touch" security interventions [35], we must ask whether this intervention would be "light-touch" in practice. Generating scenarios, playing Security Fictions, then integrating requisite security fixes into software may be much more time-intensive than our deployment indicates. Future work should investigate the true time-cost of this intervention. Second, were not able to determine from this deployment whether the security vulnerabilities that non-specialist developers generated were truly novel to their companies. Due to my reluctance to share study data across participants, I did not check with security specialists to see if non-specialists' solutions were known or unknown to their companies.

These limitations, along with the results of this preliminary deployment, motivate much future work at the intersection of security and design research, which I discuss in this section. In general, chief among the reflections the game prompted were about the organizational practices that surrounded security at individual companies. For us, these reflections generate questions about the social practice of software security—how software work is divided into "security" and "non-security" components, how threat identification is shaped by relational characteristics, and how threat identification relates to social power.

Organizational stories of security

Large organizations sliced the work of software neatly between "security" and "non-security" components. Non-security specialists reported confidently that security was not their job, and was instead the sole domain of security a specialist or team(s) of specialists. This strict division between security and non-security specialists also hindered developers' ability to determine whether the issue they discovered was novel, or already known to their company. Known security vulnerabilities may be tightly-guarded by security specialists. Beyond formal rules, in many cases (e.g. at a large company B), the non-security specialist did not know any member of the security team personally. At many large companies, security teams may work separately from product teams, with little opportunity to interact or share insights. In some cases, these organizational divisions hindered developers ability to raise security issues at all, a point to which I return below.

These findings motivate work into the "edges" of security's occupational boundaries. How is the work of security divided into "security" and "non-security" domains? Interrogating this question may help researchers understand who is left out of the security process—and how their perspectives, and unique ability to identify threats, might be better-leveraged to create safer tools and products. This work would be well-suited to future qualitative research at DIS, CHI, CSCW and beyond.

Threat identification and social power

This preliminary work confirms that not all software engineers participate equally in the work of threat identification. This, in turn, begs the question: who *gets* to identify threats? What are the privileges that come along with threat identification, and what does it mean for users that only certain people are entrusted with this responsibility? As prior work in HCI has investigated the social power associated with speculative practices [34], we (as a community) must continue to investigate how threat identification configures its practitioners into positions of social and political power. The ability to refer to something as a "security threat" may itself embed, challenge, or maintain normative structures of property, familial or social relationships and political ideology. Future work should dig deeply into how the practice of security itself embeds power dynamics between designers and users, and design possible interventions to both work with and challenge these dynamics.

Another, related question is: how does a person's social positionality relate to the type of threats they are able to identify? Pierce et al. coin the term *differential vulnerability* to describe how social position affects the security threats to which one is vulnerable [28]. It is equally possible that there exists a "differential visibility," in which one's social positionality affects the security threats one is likely to surface. If so, such differences could add a new dimension to discussions about diversity in software development: people with different relational characteristics may be able to identify security threats that would not otherwise be noticed. If true, this would give no urgency to prior pushes to integrate users' considerations into the security process [29].

Who beyond software developers might participate in the practice of security? Designers, managers, primary, and secondary users may all provide perspectives. Designers pose a particularly interesting group to explore. Designers, who tend to work upstream of project deployment, have the ability to identify security threats before engineering takes place. Future work may be able to draw on existing UX design practices to imagine security risks in real-life context. For example, the UX practice of "user personas" helps designers imagine the lived experiences of users. Could a similar practice be applied to the lived experiences of possible attackers? Such adversaries could help in generating scenarios, which could in turn be used to play Security Fictions. Future work should examine this issue more deeply, looking beyond software developers and customers toward activists and other marginalized groups.

Inviting design research into the fold of computer security

Methods from design research stand to bring potentially great value to the practice of computer security. Prior work in design research and related fields has already broadened the concerns of security considerably, making threats "count" beyond Alice-and-Bob style scenarios of anonymous, impersonal attackers [17, 23, 33, 11]. This work calls for the design research community to further mobilize design research methods, speculative practices and beyond, to produce workable security interventions. Such methods continue to show success in surfacing the sorts of sociotechnical threats that traditional threat identification may miss. When design researchers place these tools in the hands of developers, those tools may find a new life in the realm of technical practice. In the remainder of this section, I provide various avenues for design research to offer its methods to aid the work of computer security.

Raising (and fostering) dissent

Participants did not always know where to go with their observations. In some cases, the study left participants with issues they did not know how to fix or pursue. Leaving participants in this predicament raises ethical issues for researchers who develop threat identification practices; future work should join HCI's existing efforts to involve research subjects more fundamentally in the outcomes and benefits of research paradigms [22].

In the meantime, while our work already motivates work on shifting occupational boundaries to expand practice in threat identification, future work should look more closely at what can or should happen *after* threats are identified. What channels exist to allow developers who identify threats to come forward with them? How do those channels work, and when or why do they fail?

More broadly, future work should investigate channels of dissent or concern among tech workers. As emerging harms of technology take center-stage in the public discourse—from disinformation on social media to participation in state- and non-state surveillance systems—these questions hang heavy over those who seek to reform organizational practices in technology companies with purportedly "positive" social visions. In the meantime, our study flags this issue, providing preliminary evidence that appropriate channels may not exist in technology companies currently. Design research interventions such as games and speculative enactments may foster sanctuaries within which dissent can occur. As we, the design researcher community, expand what "counts" as a security threat, we may have greater latitude to foster what is now considered political dissent under a broader umbrella of "security." This discursive move may make dissent more palatable to stakeholders, such as managers or shareholders.

Speculative enactments in the workplace

As far as I am aware, this study is the first to use speculative enactments in a workplace setting. Developers engaged actively with this method, understanding and embracing the spirit of play and speculation. Since this method is oriented toward engaging real-life consequences, it is, in retrospect, unsurprising that speculative enactments would succeed in work environments. Future work should investigate whether such games are legible or appealing to those in management positions, and also what work domains beyond security might benefit from the integration of speculative enactments. How do managers, C-level executives and others conceive of security? Involving them in the speculative practice of threat identification may reveal their biases, preferences and beliefs, perhaps also revealing further organizational dimensions to security's work.

Generating threat scenarios

This study did not draw on material artifacts or visual design. Both present rich avenues to explore in future work. Artifacts may assist in imagining scenarios (for example, design workbooks [36]) or in generating them (for example, card games to assist in role-playing). A few prior projects from DIS and beyond may assist in the design of such a practice [19, 9, 1]. Such a practice would help overcome one of the main limitations of our current design: the researcher-prescribed prompts. Putting prompt generation in the hands of players could help foster more diverse perspectives, contributing to questions above around the potential "differential visibility" of security threats.

In addition to requiring developers to generate their own prompts, a future version of the game could require developers to build the vulnerabilities they describe. These could extend speculative enactment further, adding another level of engagement and real-life consequence to speculative practices. While this may raise technical barriers of entry for certain types of threats, others (such as social engineering attacks) could create accessible enactments for a broad variety of developers and non-developers.

CONCLUSION

This work began with the observation that threat identification is an intrinsically speculative practice. It requires imagining possible futures. I echo prior work: threat identification is a socially-situated practice [11]. Multiple stakeholders and values collide with business imperatives to produce a socially-contingent set of threats deemed relevant. These observations call design research to deepen its attention on computer security, contributing practices and perspectives to help make sense of how software views its threats and how labor figures in the process.

ACKNOWLEDGMENTS

This work was made possible by funding from the William and Flora Hewlett Foundation. Many thanks to Sarah Fox, Richmond Y. Wong, Noura Howell, Lilly Irani, Dorothy Howard and reviewers at CHI and DIS for their generous feedback.

REFERENCES

[1] Stephanie Ballard, Karen M. Chappell, and Kristen Kennedy. 2019. Judgment Call the Game: Using Value Sensitive Design and Design Fiction to Surface Ethical Concerns Related to Technology. In *Proceedings of the 2019 on Designing Interactive Systems Conference (DIS '19)*. Association for Computing Machinery, New York, NY, USA, 421–433. DOI: http://dx.doi.org/10.1145/3322276.3323697

[2] Mark Blythe. 2014. Research Through Design Fiction: Narrative in Real and Imaginary Abstracts. *Proceedings of the 2014 Conference on Human Factors in Computing Systems (CHI '14)* (2014), 10. DOI: http://dx.doi.org/10.1145/2556288.2557098

[3] Sean Brooks. 2018. *Defending Politically Vulnerable Organizations Online*. Technical Report. Center for Long-Term Cybersecurity, Berkeley, CA, USA.

[4] U.S. Department of Labor Bureau of Labor Statistics. 2019. Information Security Analysts. (2019). https://www.bls.gov/ooh/computer-and-information-technology/information-security-analysts.htm

[5] Stuart Candy. 2018. Gaming futures literacy : The Thing From The Future. April (2018).

[6] Kathy Charmaz. 2006. *Constructing grounded theory: A practical guide through qualitative analysis*. sage.

[7] Danielle Keats Citron. 2014. *Hate crimes in cyberspace*. Harvard University Press.

[8] Lizzie Coles-Kemp. 2009. Information security management: An entangled research challenge. *Information Security Technical Report* (2009). DOI: http://dx.doi.org/10.1016/j.istr.2010.04.005

[9] Tamara Denning, Batya Friedman, and Tadayoshi Kohno. 2013a. The Security Cards. (2013).

[10] Tamara Denning, Adam Lerner, Adam Shostack, and Tadayoshi Kohno. 2013b. Control-Alt-Hack: the design and evaluation of a card game for computer security awareness and education. *CCS '13: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security* (2013), 915–928. DOI: http://dx.doi.org/10.1145/2508859.2516753

[11] Paul Dourish and Ken Anderson. 2006. Collective information practice: Exploring privacy and security as social and cultural phenomena. *Human-Computer Interaction* 21, 3 (2006), 319–342. DOI: http://dx.doi.org/10.1207/s15327051hci2103_2

[12] Anthony Dunne and F Raby. 2013. Speculative Everything. *Design, Fiction and Social Dreaming* (2013), 1–10. https://mitpress.mit.edu/books/speculative-everything

[13] Chris Elsden, David Chatting, Abigail C. Durrant, Andrew Garbett, Bettina Nissen, John Vines, and David S. Kirk. 2017. On Speculative Enactments. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)* (2017), 5386–5399. DOI: http://dx.doi.org/10.1145/3025453.3025503

[14] Chris Elsden, Bettina Nissen, Andrew Garbett, David Chatting, David Kirk, and John Vines. 2016. Metadating. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16* (2016), 685–698. DOI: http://dx.doi.org/10.1145/2858036.2858173

[15] Shamal Faily, Simon Parkin, and John Lyle. *Secure System? Challenge Accepted: Finding and Resolving Security Failures Using Security Premortems*. Technical Report.

[16] Diana Freed, Jackeline Palmer, Diana Minchala, Karen Levy, Thomas Ristenpart, and Nicola Dell. 2018. A Stalker's Paradise: How Intimate Partner Abusers Exploit Technology. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)* (2018), 1–13. DOI: http://dx.doi.org/10.1145/3173574.3174241

[17] Michael Freed, Jaime Carbonell, Geoff Gordon, Jordan Hayes, Brad Myers, Daniel Siewiorek, Stephen Smith, Aaron Steinfeld, and Anthony Tomasic. 2008. RADAR : A Personal Assistant that Learns to Reduce Email Overload. In *Twenty-Third AAAI Conference on Artificial Intelligence*. 1287–1293. DOI: http://dx.doi.org/10.1093/acprof:oso/9780199606375.003.0001

[18] S. Frey, A. Rashid, P. Anthonysamy, M. Pinto-Albuquerque, and S. A. Naqvi. 2019. The Good, the Bad and the Ugly: A Study of Security Decisions in a Cyber-Physical Systems Game. *IEEE Transactions on Software Engineering* 45, 5 (2019), 521–536.

[19] Batya Friedman and David G. Hendry. 2012. The envisioning cards: A toolkit for catalyzing humanistic and technical imaginations. *Proceedings of the 2012 ACM Annual Conference on Human Factors in Computing Systems (CHI '12)* (2012), 1145–1148. DOI: http://dx.doi.org/10.1145/2207676.2208562

[20] Charles Goodwin. 1994. Professional Vision. *American Anthropologist* 96, 3 (1994), 606–633. DOI: http://dx.doi.org/10.1525/aa.1994.96.3.02a00100

[21] Jason I. Hong, Jennifer D. Ng, Scott Lederer, and James A. Landay. 2004. Privacy risk models for designing privacy-sensitive ubiquitous computing systems. *Proceedings of the 2004 conference on Designing interactive systems processes, practices, methods, and techniques (DIS '04)* (2004), 91. DOI: http://dx.doi.org/10.1145/1013115.1013129

[22] Dorothy Howard and Lilly Irani. 2019. Ways of Knowing When Research Subjects Care. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–16.

[23] Roxanne Leitão. 2019. Anticipating Smart Home Security and Privacy Threats with Survivors of Intimate Partner Abuse. In *Proceedings of the 2019 on Designing Interactive Systems Conference (DIS '19)*. ACM, New York, NY, USA, 527–539. DOI: http://dx.doi.org/10.1145/3322276.3322366

[24] Abstract P Lisa Nathan, Predrag V Klasnja, and Batya Friedman. 2007. *Value Scenarios: A Technique for Envisioning Systemic Effects of New Technologies*.

[25] Mike Masnick. 2018. CIA: A competitive card game based on the CIA's declassified training game, Collection Deck. (2018). https://www.kickstarter.com/projects/mmasnick/cia-collect-it-all

[26] William Newhouse, Stephanie Keith, Benjamin Scribner, and Greg Witte. 2017. National initiative for cybersecurity education (NICE) cybersecurity workforce framework. *NIST Special Publication* 800 (2017), 181.

[27] Helen Nissenbaum. 2005. Where computer security meets national security. *Ethics and Information Technology* 7, 2 (2005), 61–73. DOI: http://dx.doi.org/10.1007/s10676-005-4582-3

[28] James Pierce, Nick Merrill, Richmond Y Wong, Sarah Fox, and Eric Paulos. 2018. An Interface without A User : An Exploratory Design Study of Online Privacy Policies and Digital Legalese. (2018).

[29] Lena Reinfelder, Robert Landwirth, and Zinaida Benenson. 2019. Security Managers Are Not The Enemy Either. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19* (2019), 1–7. DOI:http://dx.doi.org/10.1145/3290605.3300663

[30] AnnaLee Saxenian. 1996. *Regional advantage*. Harvard University Press.

[31] Adam Shostack. 2014a. Elevation of Privilege: Drawing Developers into Threat Modeling. *USENIX Summit on Gaming, Games, and Gamification in Security Education* (2014), 1–15.

[32] Adam Shostack. 2014b. *Threat modeling: Designing for security*. John Wiley & Sons.

[33] Lucy Suchman, Karolina Follis, and Jutta Weber. 2017. Tracking and Targeting : Sociotechnologies of (In)security. 42, 6 (2017), 983–1002.

[34] Jasper Tran O'Leary, Sara Zewde, Jennifer Mankoff, and Daniela K Rosner. 2019. Who Gets to Future? Race, Representation, and Design Methods in Africatown. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–13.

[35] Charles Weir, Lynne Blair, Ingolf Becker, M Angela Sasse, and James Noble. 2018. Light-touch Interventions to Improve Software Development Security. *IEEE Cybersecurity Development Conference* (2018). DOI: http://dx.doi.org/10.1109/SecDev.2018.00019

[36] Richmond Y Wong, Ellen Van Wyk, and James Pierce. 2017. Real-fictional entanglements: Using science fiction and design fiction to interrogate sensing technologies. In *Proceedings of the 2017 Conference on Designing Interactive Systems*. 567–579.

[37] Christopher Wylie. 2018. Cambridge Analytica and Data Privacy. (May 2018). https://cs.pn/2WC4Oz4

[38] Micah Zenko. 2015. *Red Team : How to Succeed by Thinking Like the Enemy.* Basic Books. http://search.ebscohost.com/login.aspx?direct=true