# Breaking New Ground for Researching Secure Software Development with Social Theory

**Laura Kocksch**
Ruhr University Bochum
Universitätsstrasse 150,
44801 Bochum, Germany
laura.kocksch@rub.de

**Andreas Poller**
Fraunhofer Institute for Secure Information Technology (SIT)
Rheinstrasse 75, 64295
Darmstadt, Germany
poller@sit.fhg.de


CSCW '17 "Theory transfers? Social theory & CSCW research" Workshop, February 25, 2017, Portland, Oregon, USA.

## Abstract
It is a difficult challenge to improve the security of software products and to support the emergence of security practices in software-developing organizations. While current research work in software engineering and *usable security* takes up this challenge, it rarely considers social theory aspects. Instead, studies often employ behaviorist accounts, or monocausal explanatory models starting with effects of human or organizational factors. Both views fall short of describing the entanglement between technology, organizational structures and practices that is advantageous to map out the perspectives for design and intervention in software-developing organizations. Along these lines, we argue for the unique role CSCW researchers can play in the security discourse in both research and public.

## Author Keywords
Social theory; sociomateriality; organizational theory; software security; secure software development; software development; software engineering

## ACM Classification Keywords
H.5.3 [Group and Organization Interfaces]: Theory and models

## Introduction

Improving software products to respond to emerging cybersecurity threats is a continuous challenge in software development. While computer security is an acknowledged area for scientific research with a longstanding tradition, only since the end of the 90s-era it became a broader topic for research on software development and engineering, as well as for industry practitioners. And still there are difficulties to make security work a regular activity in software development processes. These problems repeatedly result in vulnerabilities in software products exposing sensitive information assets.

In order to tackle the question how more secure software can be created, researchers from various disciplines investigate into the human and organizational factors shaping software development, and propose new or enhance existing software development tools, methods, and procedures. We argue that current research efforts can benefit from social theory, in particular organizational theory and sociomaterial perspectives. We suggest that the CSCW community has a unique capability for secure software development research that is bringing together theorists, technical security experts, and field researchers with an interest in software development settings.

## Security Challenges Technology and Design

Technical papers on software security issues focus on proposing countermeasures for defending adversary's attack techniques. The researchers pay specific attention to cryptographic methods, detection and prevention of exploitable software code, incorporating security into software designs and architectures, and so forth [1]. For developing security technology, human factors are considered when designing interfaces between humans and software security functions - widely referred to as *usable security* [9].

However, current *usable security* work rarely accounts for the users as being on par with technology. Instead, users' behavior is predominantly conceptualized as a source of mistakes or as causing dysfunctionality thereby hampering the achievement of stated goals of security technology [9]. Accounts of users as being the flip side of sociomaterial constitutions, where technology design and user practices are entangled and mutually explicative, are rare and seldom explicated. Instead the goal is to foster security-supportive behavior with technology design. In this regard usable security falls behind sociomaterial understandings of design and workplaces in software development in CSCW [3, 7].

## Security as an Organizational Challenge

While security as a matter of software technology attracts much attention in scientific research, the creation of secure software as an organizational achievement of software-developing enterprises is not nearly as popular as a research area. Whereas the question how technology can be improved to address cybersecurity threats is continuously asked, the question how software vendors as organizations can improve to better account for security is rarely investigated systematically.

In software engineering, empirical investigations are quite common, but only few studies are field studies informed by ethnographic methods such as observations or qualitative interviews. Taking the International Conference on Software Engineering (ICSE) as a case in point, a search for the author keywords "qualitative", "workplace", "ethnography", and "case study" yields 42 hits for a period since conference inception. While this number is already low considering a time span of over 30 years, it yet includes several false positives such as experiments with tool prototypes. The more reliable numbers Zannier et al. presented for a period up to 2005 confirm this general trend that investigations into soft-

ware development settings are difficult to find in software engineering research [11].

And even if organizational settings are investigated empirically in software engineering, social theory is often marginalized, e.g., by applying monocausal explanatory models to describe how organizational factors impact particular software development outcomes like in [6] and [4]. Given this state of the art of empiricism in software engineering it is not surprising that software security as an organizational matter has also not received much attention in this community.

There is, however, a stronger focus on proposing software tools to support developers also incorporating empirical work within experimental settings and prototype evaluations. But similar to concepts of the user in *usable security*, developers are viewed merely as a source of insecurity as they might introduce security weaknesses into the software. Accordingly, the conclusion is to prevent this behavior through (technological) interventions rather than attempting to understand the broader context in which developers act and interact, as agents in organizational settings whose structure they produce and reproduce through their actions.

## The Potential Role of CSCW Researchers

Software engineers continuously advocate forwarding ever more sophisticated tools, process templates, and education measures to those who design, engineer and implement software. Security and privacy researchers create a steady stream of findings about insecure software, and new technological suggestions on how software could be changed to better defend cybersecurity attacks. However, these manifold solution proposals leave out the practices leading to insecure software, and in particular their sociomateriality and reciprocal entanglement with organizational structures. And, on top of that, we yet lack a comprehensive understanding why particular tools and procedures in software development succeed in engendering secure development practices while others do not.

In order to evolve our approaches for design or intervention into software development to support the emergence of security practices, it is required to unpack the enactment of materiality and organizational structures through practice. The goal could be to develop principles for secure software development grounded in both empirical evidence as well as organizational theory and sociomaterial perspectives. To fill this conceptual and empirical gap is virtually by definition a task for the CSCW community as it requires collaboration between security technologists, software engineers, field researchers and social theorists.

## Case Examples

An example for a CSCW perspective on secure software development is provided by Xiao et al. in their 2014 paper about security tool adoption in software-developing companies. Xiao et al. use the diffusion-of-innovation theory developed by sociologists Everett Rogers to explain how and why security tools diffuse into software development practices [10]. Applying the diffusion-of-innovation theory the authors argue for a stronger focus on social aspects of security tool adoption. For instance, the authors stress that security tools as preventive innovations face particular challenges for being adopted in organizational settings.

Interestingly, Xiao et al. also found sociomaterial aspects to shape processes of security tool adoption, though they do not explicitly address them as such: The authors emphasize that tool adoption as a collaborative practice is enacted in material artifacts as well as communication activities,

hence security tool designers should pay attention to aspects of social computing.

We also want to suggest one of our studies as an example [8]: We explored into software development practices and changes thereof after a security training for developers. We understand software development work as a deeply social activity [2] that is organized collaboratively along practices reciprocal to organizational structures [5]. We showed how the interaction between actions taken by developers and structures that result of past actions lead to two major obstacles for adopting security work practices: First, developers framed security in a way that did not make it visible as a value of the product they develop. Second, developers were not taught how to deal with security as a collaborative effort with visible goals, though it was their preferred way to work together in teams.

While our theoretical framing was influenced primarily by organizational science theories, we also found starting points where other analytical lenses could be helpful: For example, we suggest the concept of boundary objects as useful to explain how development artifacts shape practices of product development teams in terms of paying attention to or disregarding security aspects. In particular, we encountered that artifacts seem to serve a specific purpose for software development stakeholders to continually translate requirements and goals between different areas in the organization, e.g., between sales, development managers, and development teams. For our setting we found difficulties to account for security in this translation process.

As shown with these two examples, social theory adds a unique benefit to research on secure software development. Also, incorporating social theory perspectives yields further interesting challenges for CSCW research on secure software development: How can we design tools that give sufficient structure but enable flexibility and the emergence of security work practices? How can organizations master the balance between flexible structures imposed by agile development paradigms, and more fixed structures of support and guidance by their security technologists? What are our design perspectives for security tools in software development that acknowledge the embeddedness of security work within other activities, e.g., selling a product? What distinguishes organizations that suceed with security efforts from those which fail.

## Conclusion and Outlook

This paper provides insights into the benefits of a sociomateriality and organizational theory informed perspective on practices in developing secure software. We advocate the CSCW community to pay attention to this area. We suggest that a broader interdisciplinary collaboration of technologists, field researchers and social theorist can provide us a more comprehensive picture of the challenges for security in software development, and the opportunities for design and intervention that can help security practices to emerge.

Software security is a topic of broad public reception and relevance. Recent events around the 2016 presidential election in the US, but also policy makers increasing efforts to impose new cybersecurity regulations show that there is a need for a differentiated view on security in software development. CSCW researchers with their blend of technology knowledge, empiricism and theory expertise can bring a unique perspective to this public discourse.

## REFERENCES


1. A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr. 2004. Basic concepts and taxonomy of dependable and secure computing. *IEEE TDSC* 1, 1 (2004), 11–33.
2. C. D. Bates and S. Yates. 2008. Scrum down: a

software engineer and a sociologist explore the implementation of an agile method. In *Proc. CHASE '08*. ACM, 13–16.

3. P. Bjørn. 2014. Sociomaterial-Design in Global Software Development. (2014).
4. T. Dybå. 2003. Factors of software process improvement success in small and large organizations: an empirical study in the scandinavian context. In *ACM SIGSOFT Software Engineering Notes*, Vol. 28. ACM, 148–157.
5. M. S. Feldman and B. T. Pentland. 2003. Reconceptualizing Organizational Routines as a Source of Flexibility and Change. *Administrative Science Quarterly* 48, 1 (2003), 94–118.
6. M. Lavallée and P. N. Robillard. 2015. Why Good Developers Write Bad Code: An Observational Case Study of the Impacts of Organizational Factors on Software Quality. In *Proc. ICSE '15*. IEEE, 677–687.
7. W. J. Orlikowski. 2009. The sociomateriality of organisational life: considering technology in management research. *Cambridge Journal of Economics* (2009).
8. A. Poller, L. Kocksch, S. Türpe, F. A. Epp, and K. Kinder-Kurlanda. 2017. Can Security Become a Routine? A Study of Organizational Change in an Agile Software Development Group. In *Proc. CSCW'17*. ACM. to appear.
9. M. A. Sasse and I. Flechais. 2005. Usable security: Why do we need it? How do we get it? (2005).
10. S. Xiao, J. Witschey, and E. Murphy-Hill. 2014. Social Influences on Secure Development Tool Adoption: Why Security Tools Spread. In *Proc. CSCW '14*. ACM, 1095–1106.
11. C. Zannier, G. Melnik, and F. Maurer. 2006. On the success of empirical studies in the international conference on software engineering. In *Proc. ICSE '06*. ACM, 341–350.