Funktional, sicher, einfach zu nutzen – müssen Software-Entwickler Alleskönner sein?

M. ANGELA SASSE, RUHR-UNIVERSITÄT BOCHUM

RUHR
UNIVERSITÄT
BOCHUM

RUB

“Fix the Human” Approach

security awareness, education, training

Billion $ industry – $1bn alone spent on ”anti-phishing training”

But: training can’t fix human limitations

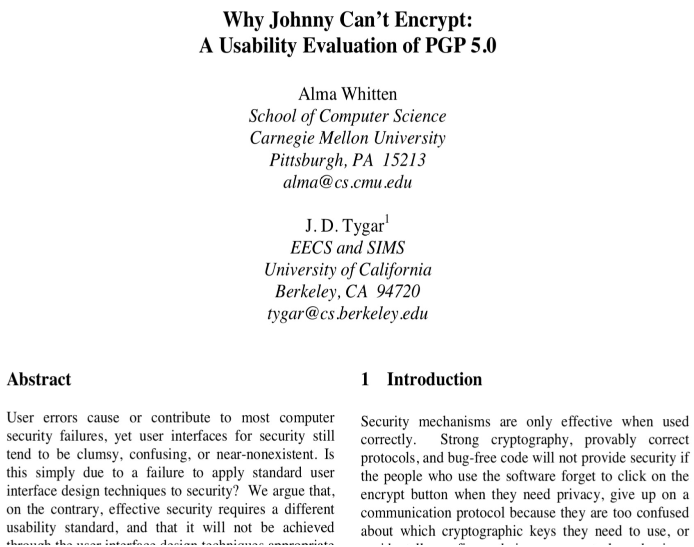
Usable security = make it easy to chose security

“If security doesn’t work for people, it doesn’t work.”

UK National Cyber Security Centre

Examples: Impossible memory tasks, unspecific warnings, CAPTCHAs …

1999 – Birth of Usable Security



Why Johnny Can't Encrypt:
A Usability Evaluation of PGP 5.0

Alma Whitten
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
alma@cs.cmu.edu

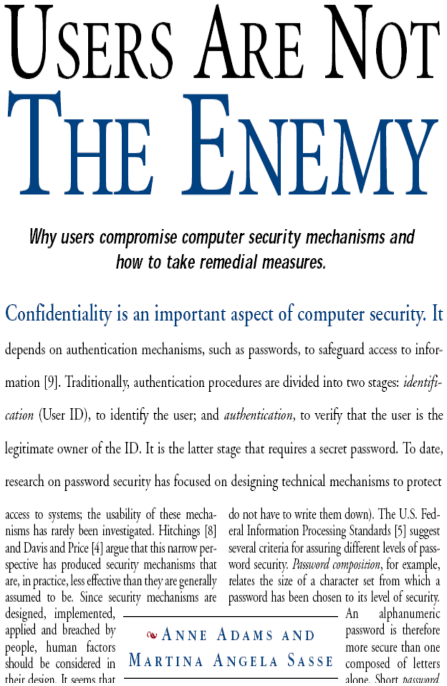
J. D. Tygar[1]
EECS and SIMS
University of California
Berkeley, CA 94720
tygar@cs.berkeley.edu

Abstract

User errors cause or contribute to most computer security failures, yet user interfaces for security still tend to be clumsy, confusing, or near-nonexistent. Is this simply due to a failure to apply standard user interface design techniques to security? We argue that, on the contrary, effective security requires a different usability standard, and that it will not be achieved through the user interface design techniques appropriate

1 Introduction

Security mechanisms are only effective when used correctly. Strong cryptography, provably correct protocols, and bug-free code will not provide security if the people who use the software forget to click on the encrypt button when they need privacy, give up on a communication protocol because they are too confused about which cryptographic keys they need to use, or



USERS ARE NOT THE ENEMY

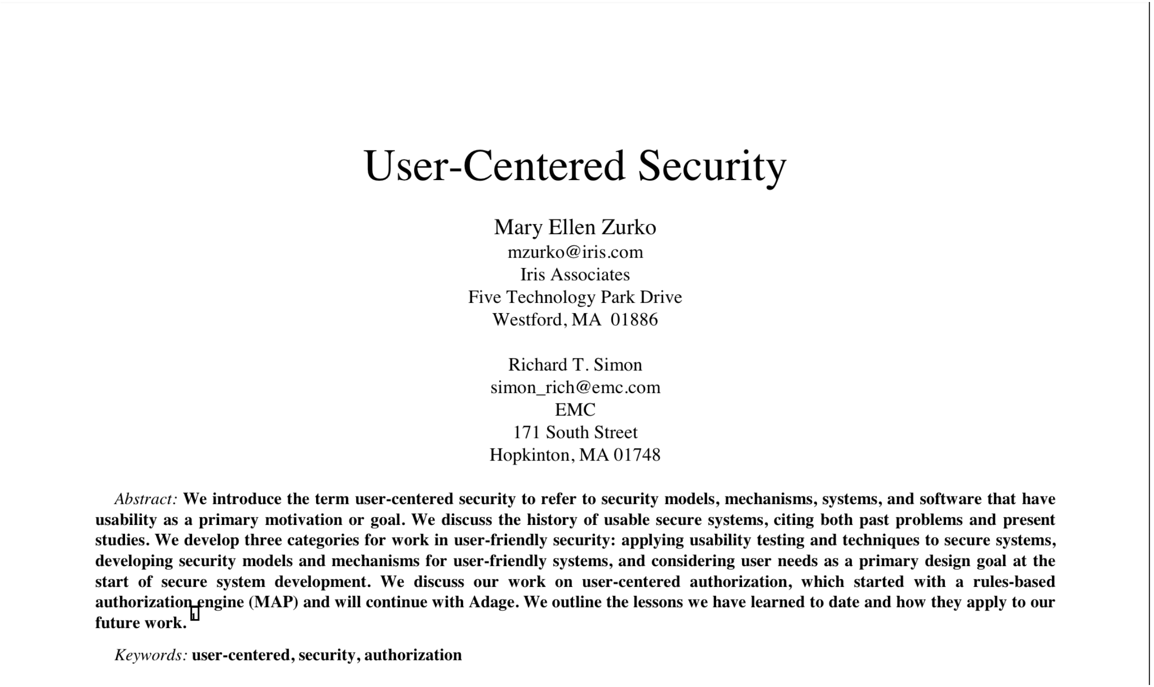
Why users compromise computer security mechanisms and how to take remedial measures.

Confidentiality is an important aspect of computer security. It depends on authentication mechanisms, such as passwords, to safeguard access to information [9]. Traditionally, authentication procedures are divided into two stages: *identification* (User ID), to identify the user; and *authentication*, to verify that the user is the legitimate owner of the ID. It is the latter stage that requires a secret password. To date, research on password security has focused on designing technical mechanisms to protect access to systems; the usability of these mechanisms has rarely been investigated. Hitchings [8] and Davis and Price [4] argue that this narrow perspective has produced security mechanisms that are, in practice, less effective than they are generally assumed to be. Since security mechanisms are designed, implemented, applied and breached by people, human factors should be considered in their design. It seems that

ANNE ADAMS AND MARTINA ANGELA SASSE

do not have to write them down). The U.S. Federal Information Processing Standards [5] suggest several criteria for assuring different levels of password security. *Password composition*, for example, relates the size of a character set from which a password has been chosen to its level of security. An alphanumeric password is therefore more secure than one composed of letters alone. Short *password*

Conception - 1996



User-Centered Security

Mary Ellen Zurko
mzurko@iris.com
Iris Associates
Five Technology Park Drive
Westford, MA 01886

Richard T. Simon
simon_rich@emc.com
EMC
171 South Street
Hopkinton, MA 01748

Abstract: **We introduce the term user-centered security to refer to security models, mechanisms, systems, and software that have usability as a primary motivation or goal. We discuss the history of usable secure systems, citing both past problems and present studies. We develop three categories for work in user-friendly security: applying usability testing and techniques to secure systems, developing security models and mechanisms for user-friendly systems, and considering user needs as a primary design goal at the start of secure system development. We discuss our work on user-centered authorization, which started with a rules-based authorization engine (MAP) and will continue with Adage. We outline the lessons we have learned to date and how they apply to our future work.**

Keywords: **user-centered, security, authorization**

Usable security for ***developers*** *and* ***sysadmins*** *as well as users*

Studies on developers

- Much security code is copied – Stackoverflow, Github
- ”Fix at source”, provide secure example code/patterns
- Balancing security and productivity is tricky

NSA Best Science of Cybersecurity paper 2017

2016 IEEE Symposium on Security and Privacy

You Get Where You’re Looking For
The Impact of Information Sources on Code Security

Yasemin Acar, Michael Backes, Sascha Fahl, Doowon Kim[†], Michelle L. Mazurek[†], Christian Stransky
CISPA, Saarland University; [†]University of Maryland, College Park

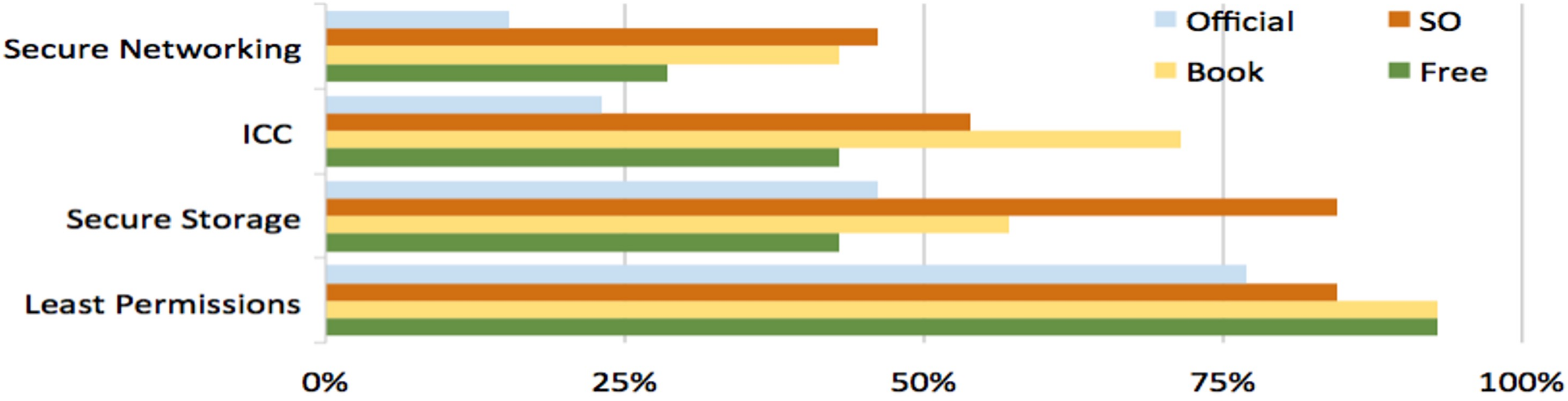
***Abstract*—Vulnerabilities in Android code – including but not limited to insecure data storage, unprotected inter-component communication, broken TLS implementations, and violations of least privilege – have enabled real-world privacy leaks and motivated research cataloguing their prevalence and impact. Researchers have speculated that appification promotes security problems, as it increasingly allows inexperienced laymen to develop complex and sensitive apps. Anecdotally, Internet resources such as Stack Overflow are blamed for promoting insecure solutions that are naively copy-pasted by inexperienced developers.**

In this paper, we for the first time systematically analyzed how the use of information resources impacts code security. We first surveyed 295 app developers who have published in the Google Play market concerning how they use resources to solve security-related problems. Based on the survey results, we

[29], [31], [33], [34], [36], [43], [44], [46]. Developers tend to request more permissions than actually needed, do not use TLS or cryptographic APIs correctly, often use insecure options for Inter Component Communication (ICC), and fail to store sensitive information in private areas.

Some previous work attempts to assess root causes for these programming errors. A frequent conclusion is that APIs are too complicated or insufficiently documented. Anecdotal reports indicate that developers use a search engine for help when they encounter an unfamiliar security issue. The search results often lead to official API documentation, blog posts, or Q&A forums such as Stack Overflow[1]. For example, Fahl et al. [16]–[18] interviewed developers whose use of pasted code snippets

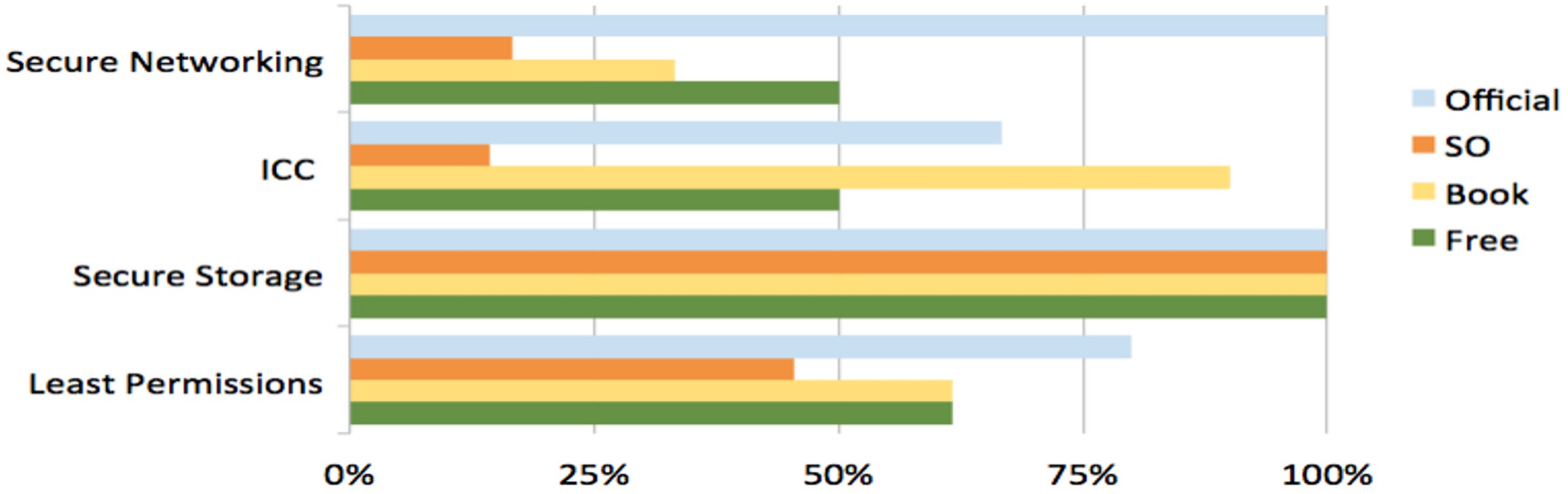
Functional correctness?



- SO (67%) and Book (66%) performed best
- Official (40%) performed worst, significantly worse than SO

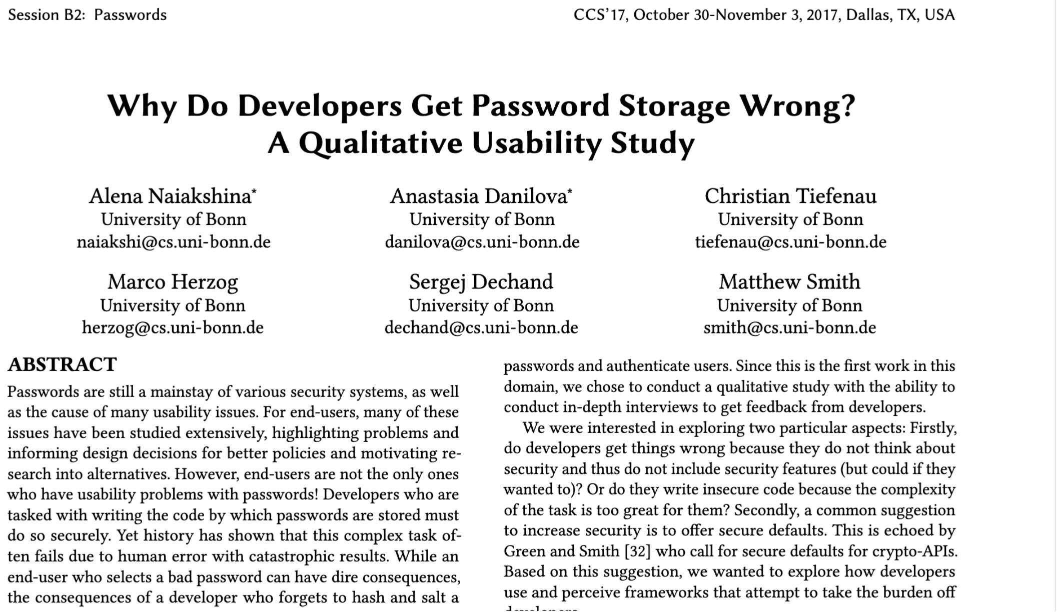
Security?

Percent of functional participants



SO worst (51%), Official best (86%) (significant)

- Giving developers choices they shouldn’t have – e.g. to chose outdated crypto
- OWASP advice is good advice: *“use bcrypt, unless you have a very good reason not to”*



Session B2: Passwords CCS’17, October 30-November 3, 2017, Dallas, TX, USA

Why Do Developers Get Password Storage Wrong? A Qualitative Usability Study

Alena Naiakshina*
University of Bonn
naiakshi@cs.uni-bonn.de

Anastasia Danilova*
University of Bonn
danilova@cs.uni-bonn.de

Christian Tiefenau
University of Bonn
tiefenau@cs.uni-bonn.de

Marco Herzog
University of Bonn
herzog@cs.uni-bonn.de

Sergej Dechand
University of Bonn
dechand@cs.uni-bonn.de

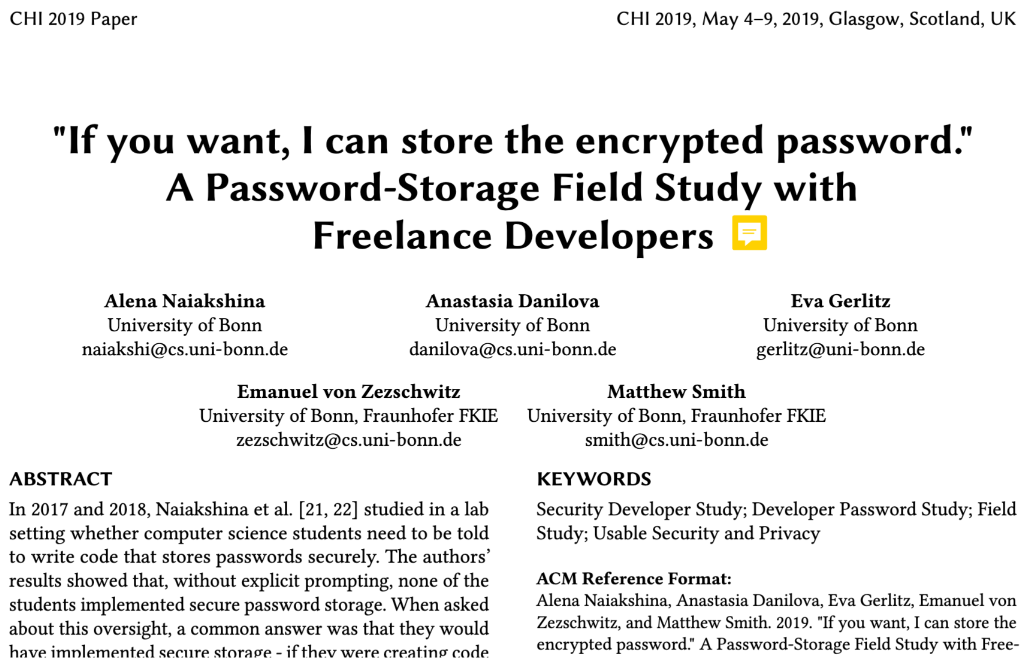
Matthew Smith
University of Bonn
smith@cs.uni-bonn.de

ABSTRACT

Passwords are still a mainstay of various security systems, as well as the cause of many usability issues. For end-users, many of these issues have been studied extensively, highlighting problems and informing design decisions for better policies and motivating research into alternatives. However, end-users are not the only ones who have usability problems with passwords! Developers who are tasked with writing the code by which passwords are stored must do so securely. Yet history has shown that this complex task often fails due to human error with catastrophic results. While an end-user who selects a bad password can have dire consequences, the consequences of a developer who forgets to hash and salt a passwords and authenticate users. Since this is the first work in this domain, we chose to conduct a qualitative study with the ability to conduct in-depth interviews to get feedback from developers.

We were interested in exploring two particular aspects: Firstly, do developers get things wrong because they do not think about security and thus do not include security features (but could if they wanted to)? Or do they write insecure code because the complexity of the task is too great for them? Secondly, a common suggestion to increase security is to offer secure defaults. This is echoed by Green and Smith [32] who call for secure defaults for crypto-APIs. Based on this suggestion, we wanted to explore how developers use and perceive frameworks that attempt to take the burden off

- If you want security, ask for it!
- The more specific security requirements developers are given, the better more secure the product



CHI 2019 Paper CHI 2019, May 4–9, 2019, Glasgow, Scotland, UK

"If you want, I can store the encrypted password." A Password-Storage Field Study with Freelance Developers

Alena Naiakshina
University of Bonn
naiakshi@cs.uni-bonn.de

Anastasia Danilova
University of Bonn
danilova@cs.uni-bonn.de

Eva Gerlitz
University of Bonn
gerlitz@uni-bonn.de

Emanuel von Zezschwitz
University of Bonn, Fraunhofer FKIE
zezschwitz@cs.uni-bonn.de

Matthew Smith
University of Bonn, Fraunhofer FKIE
smith@cs.uni-bonn.de

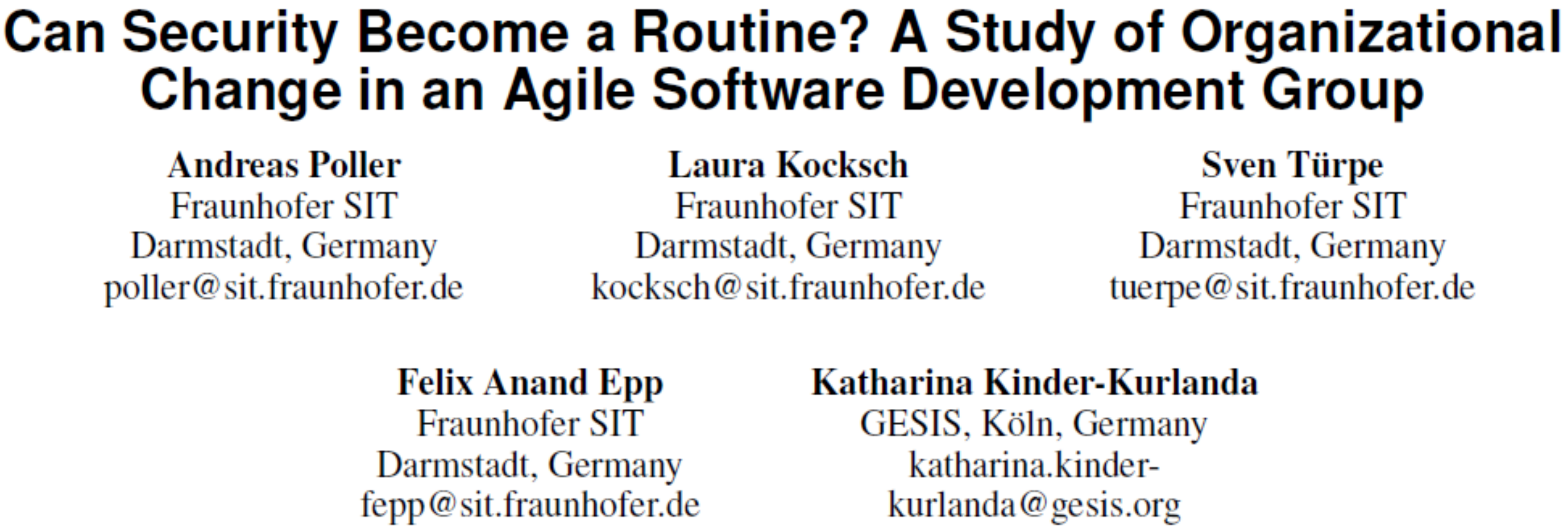
ABSTRACT

In 2017 and 2018, Naiakshina et al. [21, 22] studied in a lab setting whether computer science students need to be told to write code that stores passwords securely. The authors' results showed that, without explicit prompting, none of the students implemented secure password storage. When asked about this oversight, a common answer was that they would have implemented secure storage - if they were creating code

KEYWORDS

Security Developer Study; Developer Password Study; Field Study; Usable Security and Privacy

ACM Reference Format:
Alena Naiakshina, Anastasia Danilova, Eva Gerlitz, Emanuel von Zezschwitz, and Matthew Smith. 2019. "If you want, I can store the encrypted password." A Password-Storage Field Study with Free-



Can Security Become a Routine? A Study of Organizational Change in an Agile Software Development Group

Andreas Poller
Fraunhofer SIT
Darmstadt, Germany
poller@sit.fraunhofer.de

Laura Kocksch
Fraunhofer SIT
Darmstadt, Germany
kocksch@sit.fraunhofer.de

Sven Türpe
Fraunhofer SIT
Darmstadt, Germany
tuerpe@sit.fraunhofer.de

Felix Anand Epp
Fraunhofer SIT
Darmstadt, Germany
fepp@sit.fraunhofer.de

Katharina Kinder-Kurlanda
GESIS, Köln, Germany
katharina.kinder-kurlanda@gesis.org

Case Study: Initial Situation

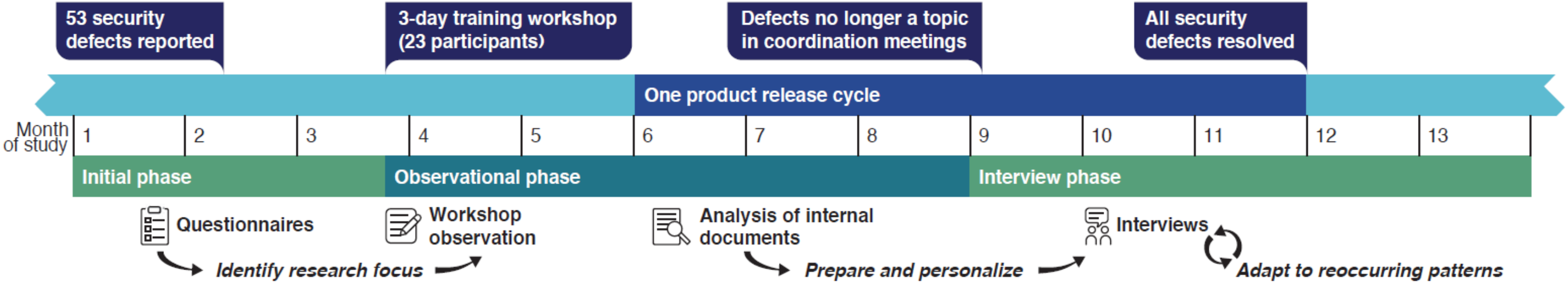
Globesoft Corp.

- Multinational software developing enterprise (3000+ employees)
- Agile development (Scrum & Kanban) for 10 years
- Development infrastructure centrally managed (defect tracker, source code version control systems, automated build and test systems)
- Product under investigation: Web dashboard for business data visualization
    - 37 developers (5 teams with each a Scrum Master + Product Owner)
    - R&D Management + Product Management
    - Distributed among Europe, North America & India (Software Testing)

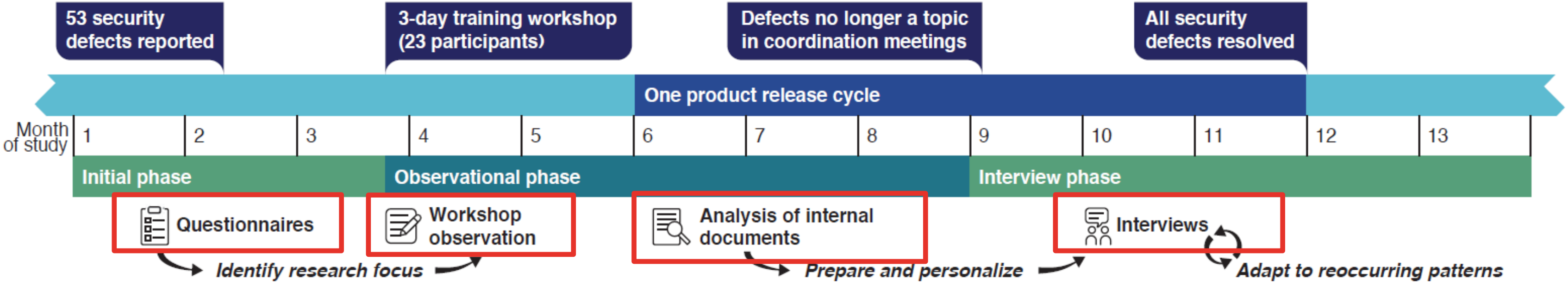
Case Study: Initial Situation - Security

- Security audits performed as part of their internal security initiative
- Central Security Team (limited resources)
  - Provided automated testing tools, gathering reports, awareness & guidelines
- No security “disasters” in the past
- Security is not a selling point

Case Study: Activities of the product group


53 security defects reported
3-day training workshop (23 participants)
Defects no longer a topic in coordination meetings
All security defects resolved
One product release cycle
Month of study
1
2
3
4
5
6
7
8
9
10
11
12
13
Initial phase
Observational phase
Interview phase
Questionnaires
Workshop observation
Analysis of internal documents
Interviews
Identify research focus
Prepare and personalize
Adapt to reoccurring patterns

Case Study: Activities of the product group



53 security defects reported

3-day training workshop (23 participants)

Defects no longer a topic in coordination meetings

All security defects resolved

One product release cycle

Month of study

1 2 3 4 5 6 7 8 9 10 11 12 13

Initial phase

Observational phase

Interview phase

Questionnaires

Workshop observation

Analysis of internal documents

Interviews

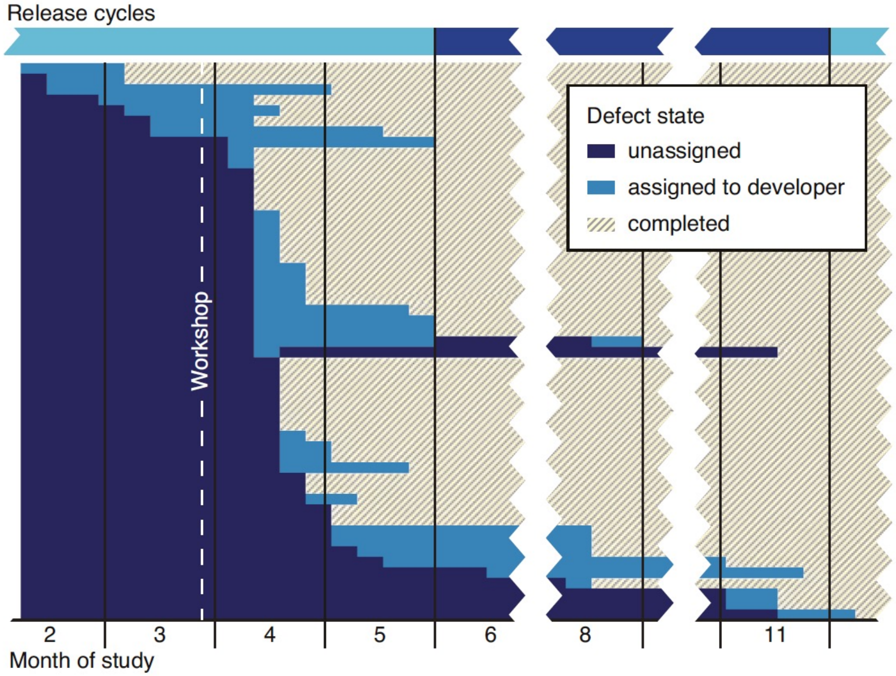
Identify research focus

Prepare and personalize

Adapt to reoccurring patterns

Field diary >100 pages + 14 hours of interviews

Defect Report



1. Defects were distributed to teams, or teams picked this by themselves
2. Reports were assigned to individual developers
3. Developers resolved defects and the code was then tested if something broke

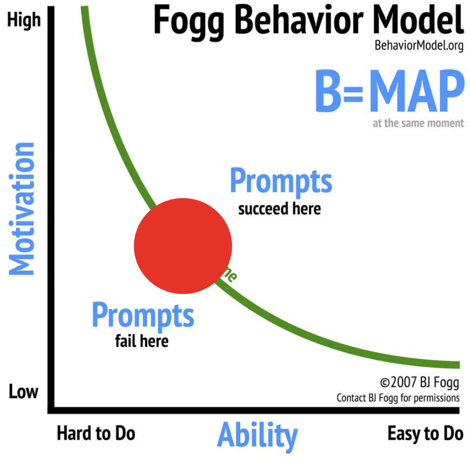
“The consultant himself explained that he **had not been contracted to change the processes in the teams.**

But this was obviously a topic for the developers, as we observed how discussions emerged during the workshop concerning collaboration and coordination in development teams. However, the **consultant did not follow up on them.**”

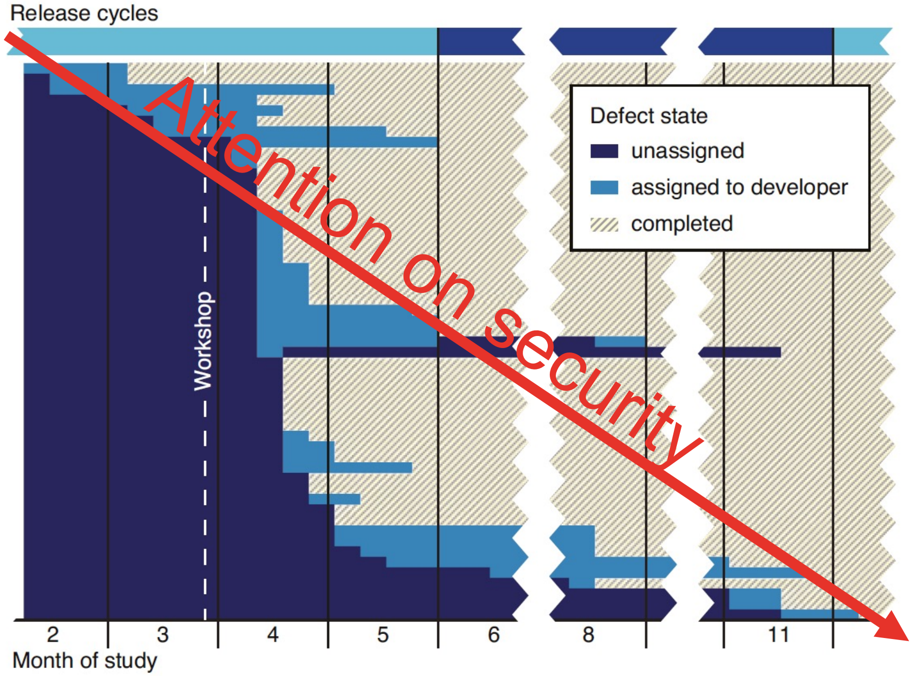
Effects of the workshop

- The workshop triggered engagement with security
- Developers felt empowered to fix security issues

“Sure, there was euphoria because of the training: We have these security holes – let’s tackle them.”



Attention on security – amount of defects



- Working through security errors became part of everyday life (at least for a short period)

"getting the counter down"
R&D Manager

"Security aspects are so far no special topic [at the coordination meetings], moreover it was one among many other work packages." (I6)

Dealing with defects (A thought experiment)

How would you assign defects and how would you deal with them in an agile software team?

What happened in the case?

Who had worked on the component with the defect?

→ **“take the defect”**

The team decides

Dealing with defects (A thought experiment)

How can we do better?

A team needs room (and guidance) to improve!!

1. **Try to understand why this is a security risk (if you do not understand this, consult someone who might know)**
2. **After fixing a defect let it review from someone (preferred someone with security expertise)**
3. **If this is something which may occur in future, write it into the internal wiki**
4. **Share it with the team (e.g. in the next “Review”)**

~~Long~~ short-term effects

- **Security conceptualized as a quality attribute (“stabilization routine”)**
- **Near everyone was motivated through the workshop (“Eye opener”)**
- **No new routines were established**

➡ **Everything went back to “normal” (insecure)**

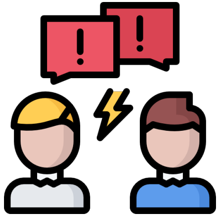
“It appeared as if developers were in a state of watchfulness for security problems after the consultancy.”

Management statement…

"... what the developers are saying, we actually need to have more time [for security], is exactly the same I'm trying to explain: That would be a [higher] management decision – we are building fewer features and focus on something else. From my perspective this is currently not considered." (16)

Pressure

Determining



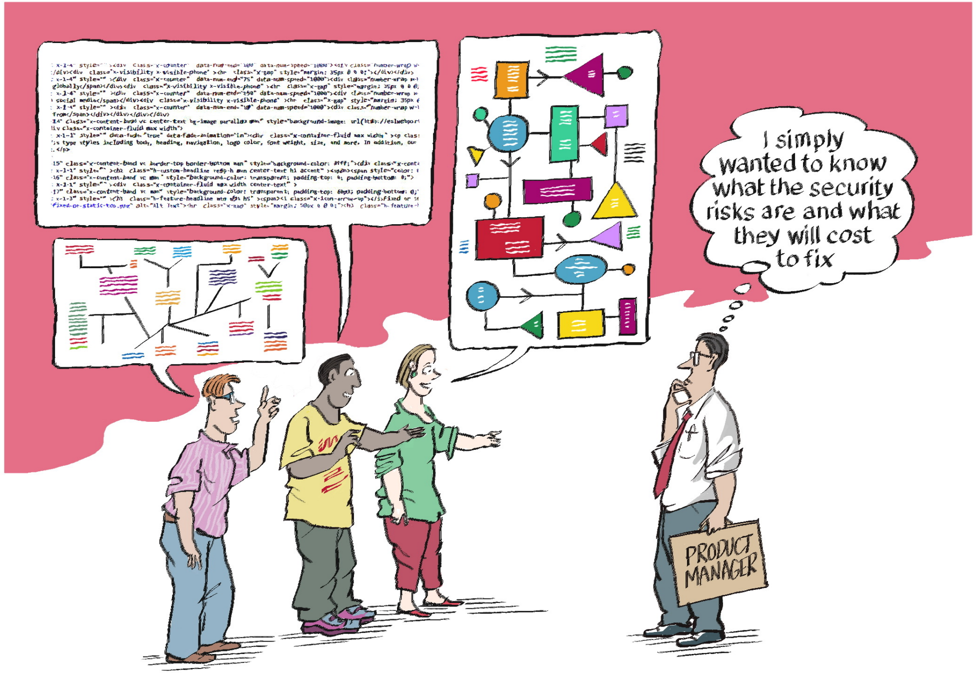
Agile

Team builds its own processes

Self-organized

Security experts often overdo it …

- *“The perfect is the enemy of good”*
- Futility is the last thing we want to induce


I simply wanted to know what the security risks are and what they will cost to fix
PRODUCT MANAGER

From https://www.securedevelopment.org Thank you Charles Weir and Noel Ford

Soft skills for security experts, so they can work with developers …

Security Dialogues:

Building Better Relationships between Security and Business

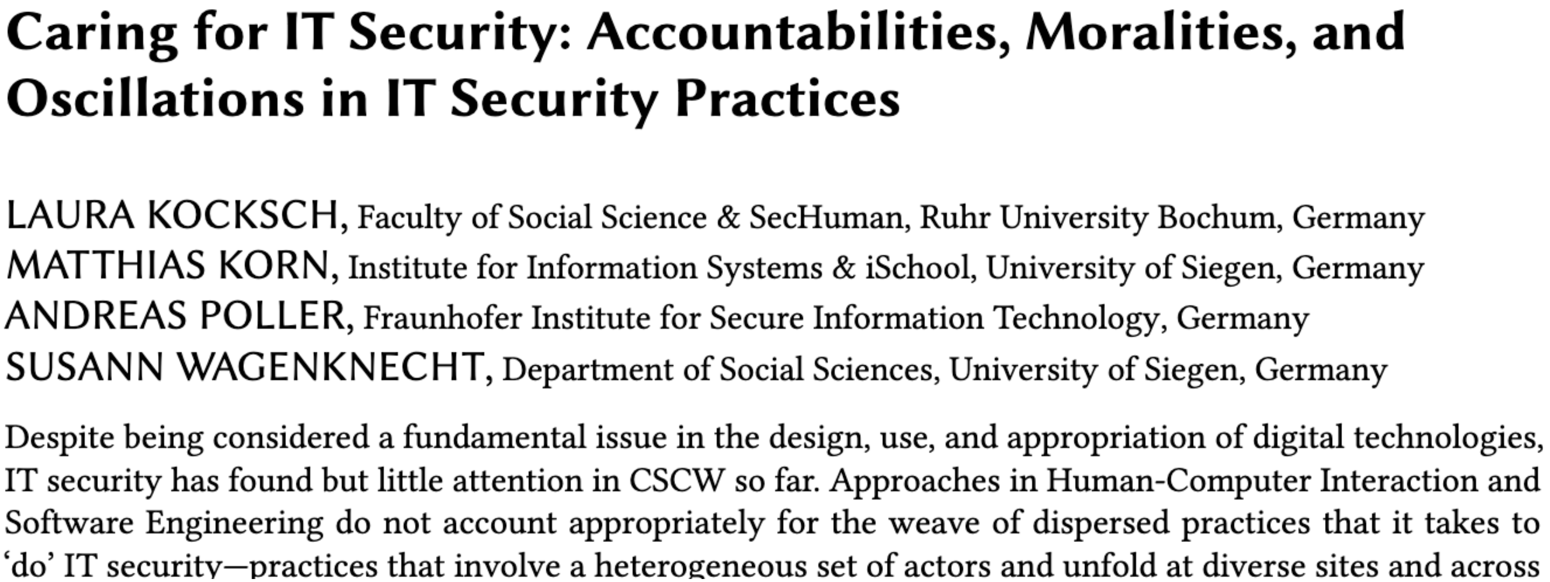
Debi Ashenden and Darren Lawrence | Cranfield University at the Defence Academy of the United Kingdom

A police officer sees a drunk man searching for something under a streetlight and asks what he's lost.

if successful, we might also improve security processes and contribute to the development of a stron-

exacerbated by their failure to take a “participative approach” to solving security problems.[6]

Deeper reason than routine …



Caring for IT Security: Accountabilities, Moralities, and Oscillations in IT Security Practices

LAURA KOCKSCH, Faculty of Social Science & SecHuman, Ruhr University Bochum, Germany
MATTHIAS KORN, Institute for Information Systems & iSchool, University of Siegen, Germany
ANDREAS POLLER, Fraunhofer Institute for Secure Information Technology, Germany
SUSANN WAGENKNECHT, Department of Social Sciences, University of Siegen, Germany

Despite being considered a fundamental issue in the design, use, and appropriation of digital technologies, IT security has found but little attention in CSCW so far. Approaches in Human-Computer Interaction and Software Engineering do not account appropriately for the weave of dispersed practices that it takes to 'do' IT security—practices that involve a heterogeneous set of actors and unfold at diverse sites and across

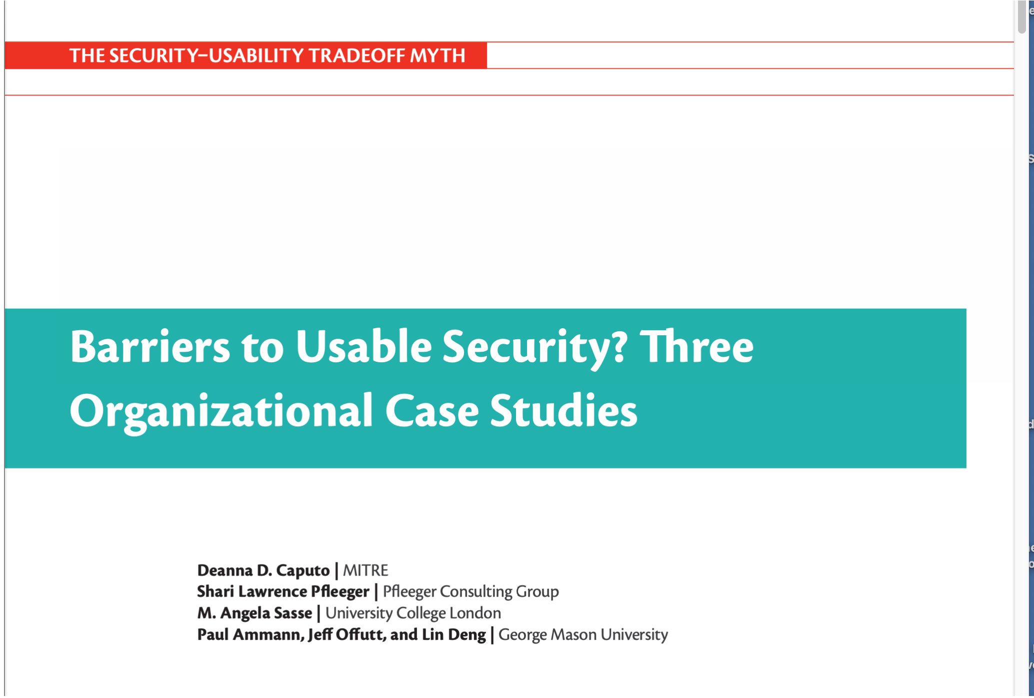
Security is not attractive to developers

1. Fear of failure: Why sacrifice productivity when your security gets broken anyway?
2. Security is seen as “caring” - not a desirable trait

Which might explain why usability doesn’t get a look in …

And what about usable security?

- 17 Interviews in 3 major companies that said they produced ”usable security”



THE SECURITY–USABILITY TRADEOFF MYTH

Barriers to Usable Security? Three Organizational Case Studies

Deanna D. Caputo | MITRE
Shari Lawrence Pfleeger | Pfleeger Consulting Group
M. Angela Sasse | University College London
Paul Ammann, Jeff Offutt, and Lin Deng | George Mason University

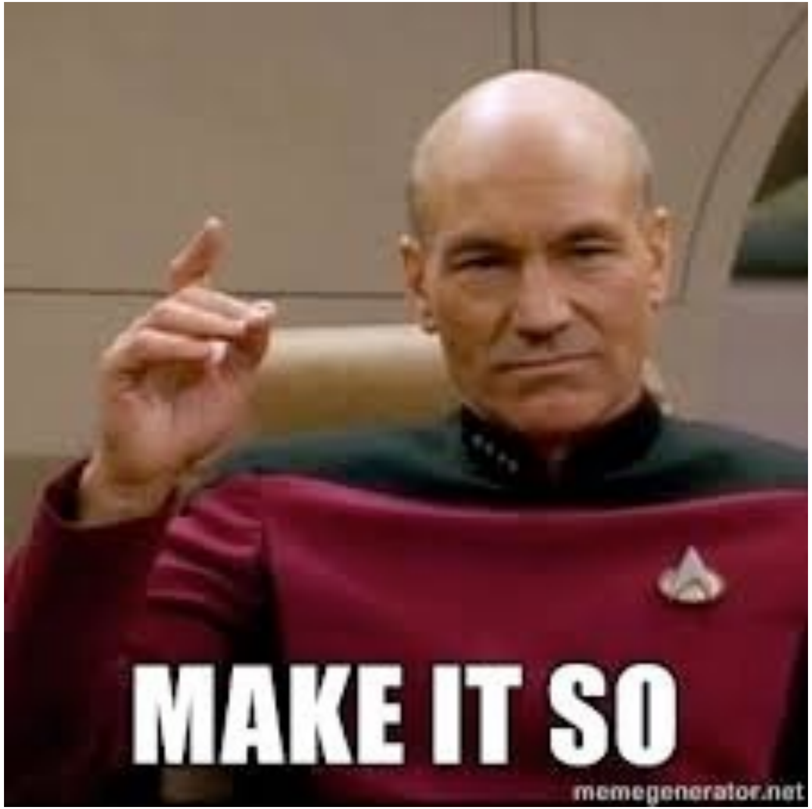
“Because those who deliver secure applications with poor usability generally don’t bear the resulting cost, complaints about unusable security are relayed to developers and then often ignored. Moreover, additional budget isn’t allocated to development for usability unless it affects the organization in a big way. “

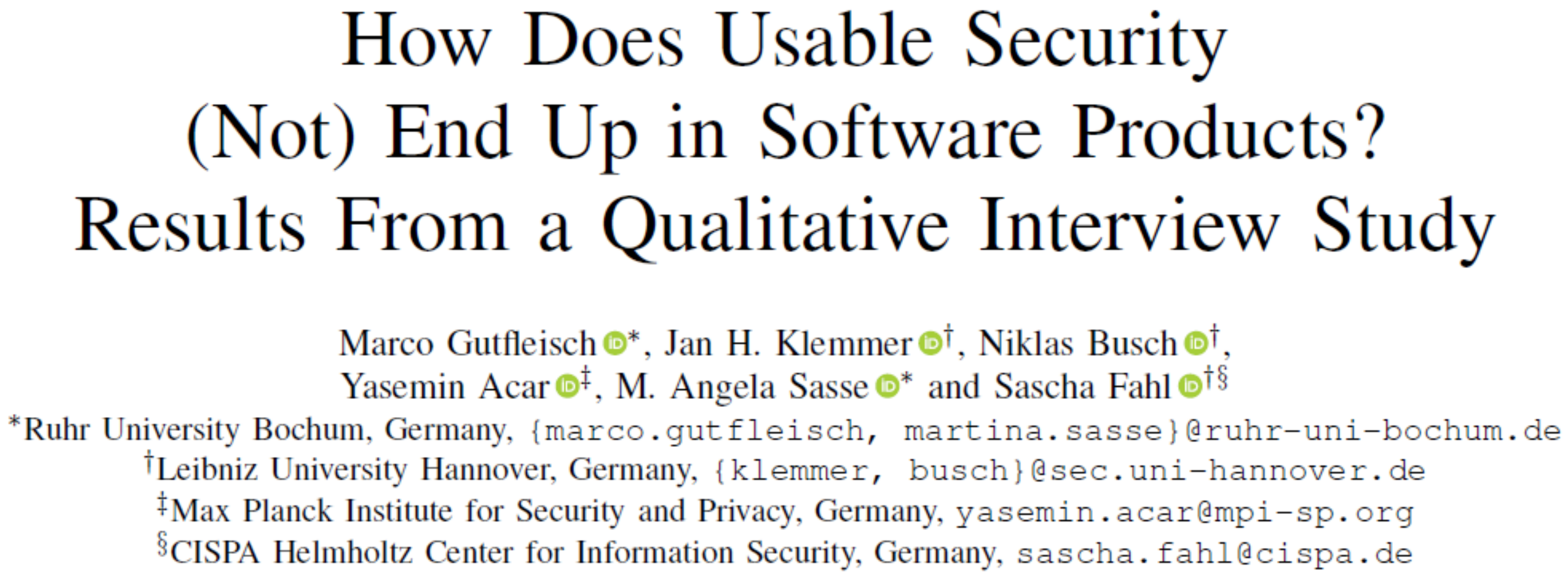
Lessons learned

- Security is to some extent ”policed”, usability is not
- Few criteria for measuring security, none for usability - except: support desk overload
- Lots of stereotyping
- Developers think they know best

- Managers assum they can just ‘order’ that security should be usable

  But not provide resources/support for it …





How Does Usable Security (Not) End Up in Software Products? Results From a Qualitative Interview Study

Marco Gutfleisch[*], Jan H. Klemmer[†], Niklas Busch[†],
Yasemin Acar[‡], M. Angela Sasse[*] and Sascha Fahl[†§]

[*]Ruhr University Bochum, Germany, {marco.gutfleisch, martina.sasse}@ruhr-uni-bochum.de
[†]Leibniz University Hannover, Germany, {klemmer, busch}@sec.uni-hannover.de
[‡]Max Planck Institute for Security and Privacy, Germany, yasemin.acar@mpi-sp.org
[§]CISPA Helmholtz Center for Information Security, Germany, sascha.fahl@cispa.de

Research Approach

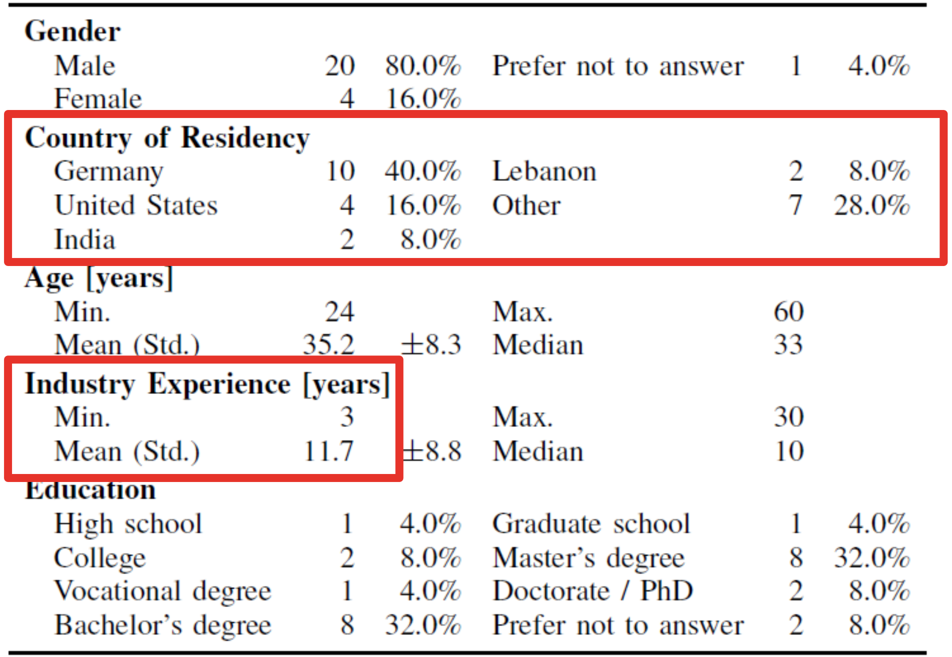
- ☑ Developer
- ☑ C-Roles
- ☑ Architects
- ☑ UI-/UX-Experts
- ☑ Designer

Get insights in **different** software **development teams** in different companies

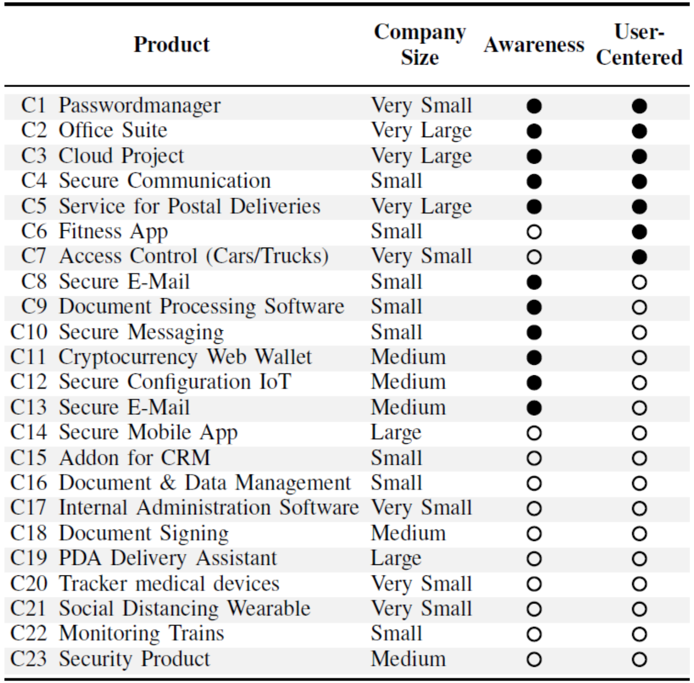
Talk to those who are **in the center of** the software development **process**

25x

Results: Demographics and Products



Gender					
Male	20	80.0%	Prefer not to answer	1	4.0%
Female	4	16.0%			
Country of Residency					
Germany	10	40.0%	Lebanon	2	8.0%
United States	4	16.0%	Other	7	28.0%
India	2	8.0%			
Age [years]					
Min.	24		Max.	60	
Mean (Std.)	35.2	±8.3	Median	33	
Industry Experience [years]					
Min.	3		Max.	30	
Mean (Std.)	11.7	±8.8	Median	10	
Education					
High school	1	4.0%	Graduate school	1	4.0%
College	2	8.0%	Master's degree	8	32.0%
Vocational degree	1	4.0%	Doctorate / PhD	2	8.0%
Bachelor's degree	8	32.0%	Prefer not to answer	2	8.0%



	Product	**Company Size**	**Awareness**	**User-Centered**
C1	Passwordmanager	Very Small	●	●
C2	Office Suite	Very Large	●	●
C3	Cloud Project	Very Large	●	●
C4	Secure Communication	Small	●	●
C5	Service for Postal Deliveries	Very Large	●	●
C6	Fitness App	Small	○	●
C7	Access Control (Cars/Trucks)	Very Small	○	●
C8	Secure E-Mail	Small	●	○
C9	Document Processing Software	Small	●	○
C10	Secure Messaging	Small	●	○
C11	Cryptocurrency Web Wallet	Medium	●	○
C12	Secure Configuration IoT	Medium	●	○
C13	Secure E-Mail	Medium	●	○
C14	Secure Mobile App	Large	○	○
C15	Addon for CRM	Small	○	○
C16	Document & Data Management	Small	○	○
C17	Internal Administration Software	Very Small	○	○
C18	Document Signing	Medium	○	○
C19	PDA Delivery Assistant	Large	○	○
C20	Tracker medical devices	Very Small	○	○
C21	Social Distancing Wearable	Very Small	○	○
C22	Monitoring Trains	Small	○	○
C23	Security Product	Medium	○	○

Factors against usable security

Limited Resources!

- “Functionality first”
- Customers and business goals do not include usability or security

“But in many cases, if the customer doesn’t have enough budget for development, you can’t set up that kind of security. [. . .] They have budget for main functionality but not for security or usability.” (P18)

Factors that hinder usable security

Limited Resources

- Usability Requirements were vague and rarely, if ever, written down.

Requirements, Guidelines, Compliance

- Usable Security Requirements did not emerge from guidelines / standards

“Actually, they came pretty naturally.” (P6)

RUHR
UNIVERSITÄT
BOCHUM
RUB

Factors that hinder usable security

Limited Resources

Requirements, Guidelines, Compliance

Misconceptions

- User blaming:

“[This is] not related to usability, mostly it’s related to lack of technology skills. [. . .] we can’t do anything about [authentication]” (P21).

- Misunderstanding of usable security:

*“But otherwise, I think we really don’t have [usable security]. **Because the login happens [. . .] [transparently for users]** and what we do there in terms of security things has no influence on how the normal user uses it.” (P9)*

Factors that hinder usable security

- Limited Resources
- Requirements, Guidelines, Compliance
- Misconceptions
- Communication Barriers

Designers & UX Experts

Developers & Security Experts

“I think they really put a lot of effort into it already. But what you wonder is if the designer was even able to grasp the front-end developer” (P10).

Structures that enable usable security

Communication Pivot

- Someone who acts as a communication bridge between two worlds (security-usability)
- Actively involvement of subject matter experts (e.g. in one case: designers were part of threat modeling)
- We observed that rather domain knowledge in only one of the areas

Structures that contribute usable security

Communication Pivot

Open Attitude and Commitment Towards Usability

- Usability was accepted and demanded by the companies
- If it is explicit part of the business goals

“The main and the most important request from the management was: they need an easy-to-use software or app or interface to compete with other competitors” (P21)

Structures that contribute usable security

Communication Pivot

Open Attitude and Commitment Towards Usability

Access to Real Users and Feedback

- Sometimes it could be difficult to get access to the actual users or to get fast feedback (e.g. in high confidential areas)
- Understanding users’ goals & problems requires involvement with users
- User Communities as an example of an effective way to get feedback

Structures that contribute usable security

- Communication Pivot
- Open Attitude and Commitment Towards Usability
- Access to Real Users and Feedback
- Knowledge About User-centered Methods and Usable Security

- Even if awareness is sufficient and access to the end users is available
  - At least a basic understanding of user-centred methods is needed
  - Processes need to be adapted

Summary

- Communication Pivot
- Open Attitude and Commitment Towards Usability
- Access to Real Users and Feedback
- Knowledge About User-centered Methods and Usable Security

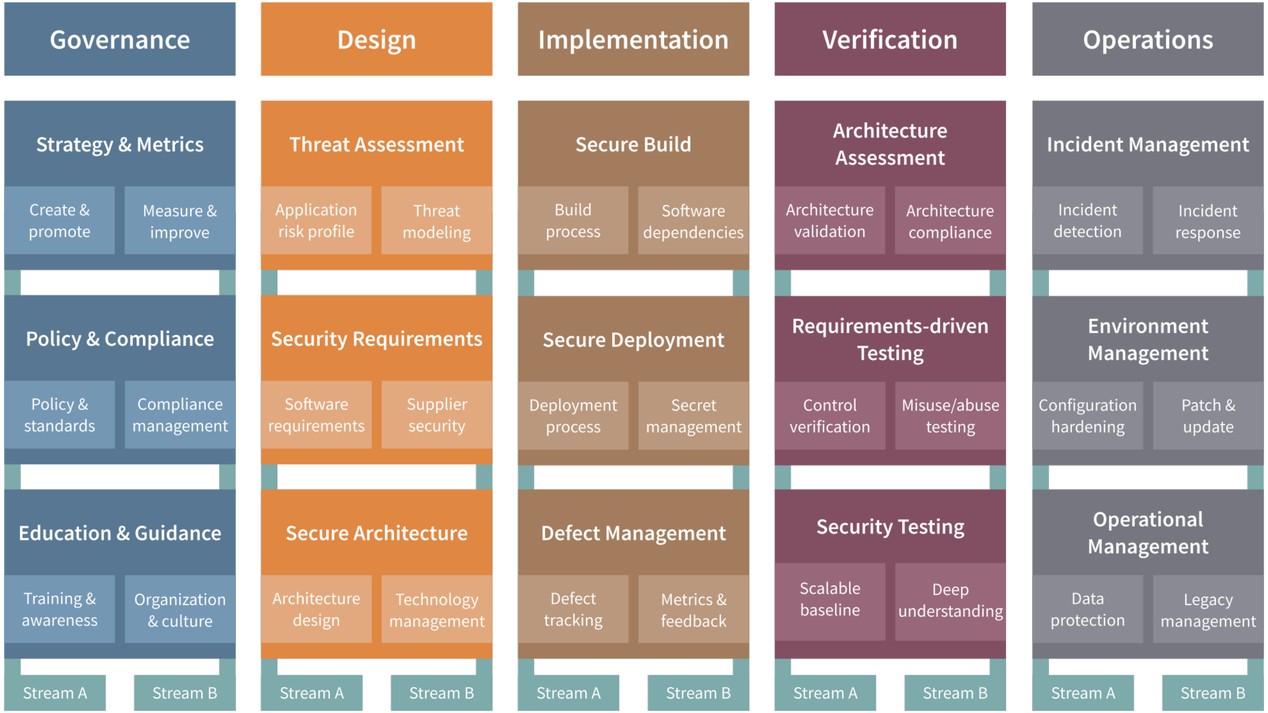
- Communication Barriers
- Limited Resources
- Requirements, Guidelines, Compliance
- Misconceptions

OWASP Software Assurance Maturity Model

“Measuring the extent of security activities as an approximation for organizational maturity to develop secure software”

- Open Source
- *“The solution details are easy enough to follow even for non-security personnel”*
- Flexibility to apply in small, medium or large organizations
- Desired maturity level depends on the organization’s needs

OWASP SAMM: Where’s usable security?



Governance

- Strategy & Metrics
  - Create & promote
  - Measure & improve
- Policy & Compliance
  - Policy & standards
  - Compliance management
- Education & Guidance
  - Training & awareness
  - Organization & culture

Design

- Threat Assessment
  - Application risk profile
  - Threat modeling
- Security Requirements
  - Software requirements
  - Supplier security
- Secure Architecture
  - Architecture design
  - Technology management

Implementation

- Secure Build
  - Build process
  - Software dependencies
- Secure Deployment
  - Deployment process
  - Secret management
- Defect Management
  - Defect tracking
  - Metrics & feedback

Verification

- Architecture Assessment
  - Architecture validation
  - Architecture compliance
- Requirements-driven Testing
  - Control verification
  - Misuse/abuse testing
- Security Testing
  - Scalable baseline
  - Deep understanding

Operations

- Incident Management
  - Incident detection
  - Incident response
- Environment Management
  - Configuration hardening
  - Patch & update
- Operational Management
  - Data protection
  - Legacy management

Stream A | Stream B (under each business function)

Conclusions (1)

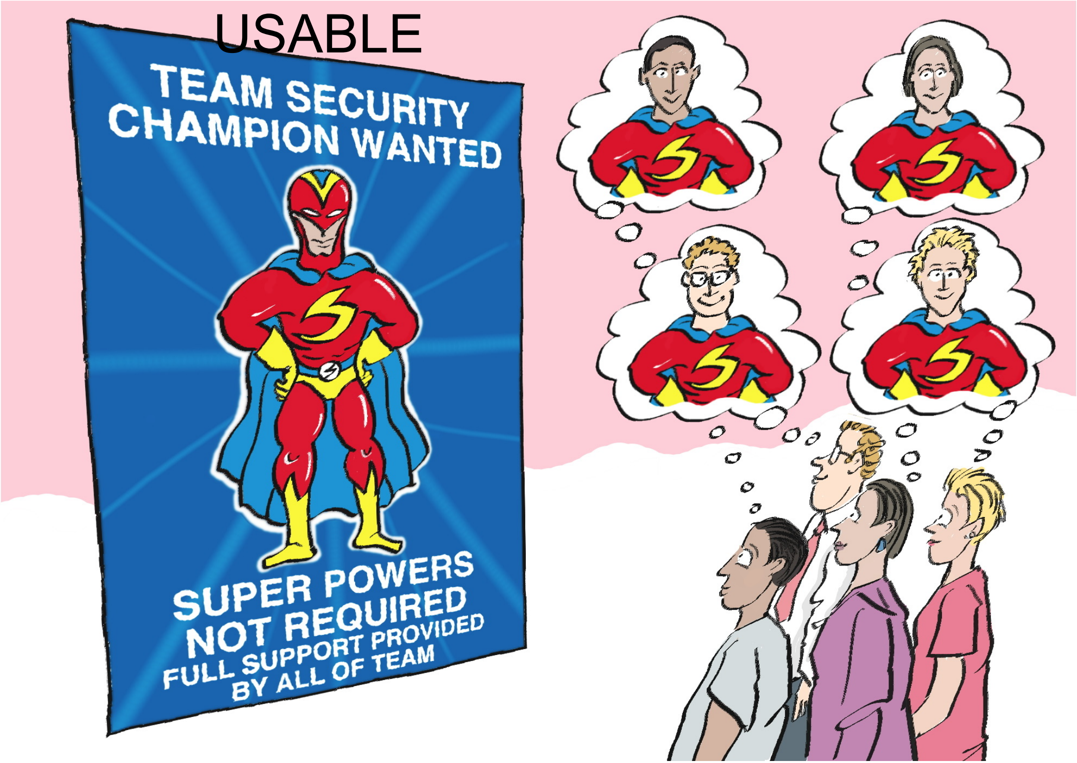
1. Business:

- If you don’t ask, you don’t get: elicit explicit requirements for security and usability!
- Support developers: with personas, scenarios, use cases. Time for heuristic evaluation, and learning from it.
- Reviewing and fixing security and usability need to become a routine – part of agile development process, represented by champions
- Lead: identify resources and synergies, broker collaboration

Security experts want to be their mini-me.

Tell them you have another job – they need to make it easy for people (users, developers) to do the right thing.

Usability is not rocket science


USABLE
TEAM SECURITY CHAMPION WANTED
SUPER POWERS NOT REQUIRED
FULL SUPPORT PROVIDED BY ALL OF TEAM

From https://www.securedevelopment.org Thank you Charles Weir and Noel Ford

Conclusions (2)

CASA “Hearts & Minds” program

- provide security and usability knowledge
- Examples for putting into practice in agile development cycle
- Transform attitudes

You can sign up at https://survey.hcs-rub.de/index.php/889189?lang=de