Computer Security and Privacy where Human Factors meet Engineering

Franziska Roesner

Assistant Professor
Computer Science & Engineering
University of Washington
New technologies bring new benefits...

... but also new risks.
Surveillance Society: Wearable fitness devices often carry security risks.

... but also new risks.
Improving Security & Privacy

Security and privacy challenges often arise when user expectations don’t match real system properties.
Improving Security & Privacy

Security and privacy challenges often arise when user expectations don’t match real system properties.

Educate, design better UIs, increase transparency.

Build systems that better match user expectations.
Why Johnny Can’t Encrypt:
A Usability Evaluation of PGP 5.0

Nikita Borisov @nikitab · Aug 12

"Why Johnny Can't Encrypt" wins the USENIX Security test of time award since Johnny still can't encrypt. #sec15 (congratulations Alma&Doug!)

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 to other types of consumer software.

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 accidentally configure their access control mechanisms to make their private data world-readable. Problems
Usable Security and Privacy

Symposium On Usable Privacy and Security

In-cooperation with USENIX

The eleventh Symposium on Usable Privacy and Security (SOUPS) will be held July 22-24, 2015 at Carleton University in Ottawa, Canada. This symposium will bring together an interdisciplinary group of researchers and practitioners in human computer interaction, security, and privacy. The program features technical papers, workshops and tutorials, a poster session, panels and invited talks, and lightning talks. SOUPS 2015 will be held in cooperation with USENIX and ACM SIGCHI.

NDSS Workshop on Usable Security 2015

Co-located with Network and Distributed System Security (NDSS) Symposium 2015

Workshop date: February 8, 2015

Workshop on Usable Privacy & Security for wearable and domestic ubiquitous DEvices (UPSIDE)

The UPSIDE workshop is an opportunity for researchers and practitioners to discuss research challenges and experiences around the usable privacy and security of wearable devices and other consumer electronics and domestic devices (e.g., home automation systems; smart appliances in the home; smart healthcare devices). The workshop was held on September 14, 2014 in conjunction with the conference in Seattle, WA, USA (41+ attendees). The workshop proceedings are available online.

Second Workshop on Usable Privacy & Security for Mobile Devices (U-PriSM 2)

August 27, 2013 Munich, Germany

co-located with MobileHCI 2013

Workshop on Usable Security and Privacy Education

THE 1ST SOFTWARE AND USABLE SECURITY ALIGNED FOR GOOD ENGINEERING (SAUSAGE) WORKSHOP

National Institute of Standards and Technology
Gaithersburg, MD USA
April 5-6, 2011
Example: Prompting Users for Security

"Our results show that 69% of participants did not apply the UAC (User Account Control) approach correctly."

[Motiee et al., SOUPS 2010]
Example: Prompting Users for Security

“[These designs] reduced the proportion of participants choosing the less safe option by up to 50%.”

[Bravo-Lillo et al., SOUPS 2013]
Warning Design Matters!  [Felt et al., CHI 2014]

~30% adherence
Warning Design Matters!  

[Felt et al., CHI 2014]

```
~60% adherence

Your connection is not private
Attackers might be trying to steal your information from www.example.com (for example, passwords, messages, or credit cards).

Advanced

Back to safety
```

```
~30% adherence
```

"opinionated design"
Improving Security & Privacy

Security and privacy challenges often arise when user expectations don’t match real system properties.

Educate, design better UIs, increase transparency.

Build systems that better match user expectations.
Improving Security & Privacy

Security and privacy challenges often arise when user expectations don’t match real system properties.

Educate, design better UIs, increase transparency.

Build systems that better match user expectations.
Case Study: Permission Granting in Modern Operating Systems  

[Roesner et al.]
Smartphone (In)Security

Users accidentally install malicious applications.

Over 60% of Android malware steals your money via premium SMS, hides in fake forms of popular apps

By Emil Protalinski, Friday, 5 Oct ’12, 05:50pm
Users accidentally install malicious applications.

Even legitimate applications exhibit questionable behavior.

Hornyack et al.: 43 of 110 Android applications sent location or phone ID to third-party advertising/analytics servers.
Smartphones (and other modern OSes) try to prevent such attacks by limiting applications’ access to:

- System Resources (clipboard, file system).
- Devices (camera, GPS, phone, ...).

How should operating system grant permissions to applications?

Standard approach: Ask the user.
State of the Art

Prompts (time-of-use)
State of the Art

Prompts (time-of-use)

Disruptive, which leads to prompt-fatigue.

Manifests (install-time)

- Storage: Modify/delete SD card contents
- System tools: Prevent phone from sleeping, write sync settings
- Your location: Fine (GPS) location
- Network communication
State of the Art

Prompts (time-of-use)

Disruptive, which leads to prompt-fatigue.

In practice, both are overly permissive: Once granted permissions, apps can misuse them.

Manifests (install-time)

Out of context; not understood by users.

[Felt et al.]
Goals for Permission Granting

1. **Least-Privilege:** Applications should receive the minimum necessary access.

2. **Usable:**
   - Not disruptive to users.
   - Matches user expectations.
   - Doesn’t require constant comprehension/management.

3. **Generalizable:** Easily extended to new resources.

("magically" grants exactly those permissions expected by the user)
Our Work: User-Driven Access Control

Let this application access my location now.

Insight:
A user’s natural UI actions within an application implicitly carry permission-granting semantics.
Our Work: User-Driven Access Control

Let this application access my location now.

Our study shows:
Many users already believe (52% of 186) – and/or desire (68%) – that resource access follows the user-driven access control model.
Resource-Related UIs Today

User’s View

- Photo Editor App

Operating System’s View

- Photo Editor App
  - Permissions: CAMERA, LOCATION

1. User clicks on camera button
2. Access camera APIs

Franziska Roesner

2015 US Frontiers of Engineering Symposium
Resource-Related UIs Today

**Problem:** OS can’t understand user’s interaction with application → can’t link permission use to user intent.

**Challenge:**
Can the system extract access control decisions from user actions in a general, application-agnostic way?

Prior approaches are hard to generalize:
EWS [SVNC ’04], NitPicker [FH ’05], CapDesk [M ’06], Qubes, Polaris [SKYCM ’06], UIBAC [SE ’08], BLADE [LYPL ’10]
New OS Primitive: Access Control Gadgets (ACGs)

Approach: Make resource-related UI elements first-class operating system objects (access control gadgets).

• To receive resource access, applications must embed a system-provided ACG.
• ACGs allow the OS to capture the user’s permission granting intent in application-agnostic way.
Access Control Gadgets (ACGs) in Action

**User’s View**

- Photo Editor App
- Camera ACG

(1) User clicks on camera ACG

**Operating System’s View**

- Camera
- Resource Monitor
- Isolation container
- Photo Editor App

<object src="rm://camera/takePicture"/>

(2) Take picture
(3) Receive picture

Franziska Roesner
2015 US Frontiers of Engineering Symposium
Access Control Gadgets (ACGs) in Action

User’s View

Photo Editor App

Operating System’s View

Camera
Resource Monitor

Isolation
container

ACG

OS

User clicks on camera ACG

(1) User clicks on camera ACG

(2) Take picture

(2) Take picture

(3) Receive picture

(3) Receive picture

User-driven access control improves security and better matches user expectations.
Other Examples

**Sign Up to see what your friends like.**

[ShareMeNot, Roesner et al.]

---

[ShadowCrypt, He et al.]

---

[PhoneAuth, Czeskis et al.]

---

[Gyrus, Jang et al.]
Challenges for the Future

New technologies are appearing rapidly, while we have still not sufficiently addressed these issues in existing technologies!
Conclusion: *Computer Security and Privacy where Human Factors meet Engineering*

As our world becomes more computerized and interconnected, (usable!) computer security and privacy are increasingly critical – for both traditional and emerging technologies.

- Educate, design better UIs, increase transparency.
- Build systems that better match user expectations.