Lecture 5

USABILITY OF PLATFORM SECURITY
You will be learning:

- Can usability of app authorization be improved?
- What other problems require balancing usability and security?
Usability of security?

Lack of security usability

- Harms security, eventually
- Lowers attractiveness of the device/service, eventually
- Costs money!
Outline

- Challenges in permission granting
- Why is usable mobile security different
- Examples of usable mobile security problem instances
Challenges in permission granting
Granting permissions to apps

Punt to user (mostly)

Decide centrally (mostly)

iOS, Windows Phone, (late) Symbian

Android
Granting permissions to apps

Punt to user
- Personalized
- ...
- Hard-to-use
- Ill-informed decisions
- Habituation
- ...

Decide centrally
- Ease-of-use
- ...
- Not personalized
- Potential liability
- ...

7
Improving usability

1. Provide more context in prompts
   Annotations with useful information
2. **Time** of granting: Install time vs. Run time
3. Implicit granting via trusted UIs
4. **Automatic granting** + auditability

Porter Felt et al, HotSec '12
1. Annotations

- Users don’t have enough signals to make informed decisions
  
  Chia et al, “Is this app safe?: a large scale study on application permissions and risk signals.”, WWW 2012

- Analyze app; show results to user

- Social navigation
  - Experts
  - Crowdsourcing
Annotations from analysis

- **Problem:** privacy risk depends on context
  - E.g., “Location”: ok for maps, not for flashlight
  - Privacy at risk if user’s expectations not met

AnnoMents from analysis

- Idea:
  - *Training*: Tell users what app does & ask if it matches their expectations
  - *Use*: Annotate permission prompt with results
Training: Get annotation info

- **Step #1:** Get permissions from manifests
- **Step #2:** Figure out how data is used
  - Analyse using [TaintDroid](https://taintdroid.org) (tracks where data goes)
  - Categorize uses: core functionality / secondary (e.g. tagging, sharing) / targeted ads
Training: Get annotation info

- Step #3: Check user reactions
  - Do you expect this app to use ...
  - Are you uncomfortable with it using X to support Y
  - Participants recruited on Amazon Mechanical Turk
Use: Show cues to users

Example permission UI from Lin et al, 2012
Crowdsourcing ratings

- Another example: Web of Trust
Concerns in centralized rating

- Who decides if a website/app/… is “bad”?
- How to incentivize participation?
Concerns in ratings by people

- How to improve coverage?

Web of Trust (http://mywot.net) ratings for popular web pages
Addressing concerns

- **Groupsourcing?**
  - Feedback from social circles, rather than the crowd as whole

See: “Groupsourcing: nudging users away from unsafe content”, NordiCHI 2014

- **Machine learning?**
  - Predict likely rating using model trained on sample ratings

2. Time of granting

Install time vs. Run time

- more time to think
- less disruptive
- no contextual info.

- contextual info.
- more fine-grained
- intrusive
3. Trusted UI

- Trusted path to user
  - Trusted widgets
  - E.g. PIN/login input screen
- Not forgeable or obscurable by REE apps
  - Hardware/OS support needed
- Other application areas:
  - User authentication
  - Transaction confirmation
  - Provisioning

Example: Dedicated Trusted UI (Global Platform)
Trusted permission widgets

- **Goal:** Permission requests should be
  - In context – informed decisions
  - Least-privilege – not “take photos at any time”
  - Supporting user task – not interrupting it
Trusted permission widgets

- Idea: trusted widget for action + permission
  - “Camera trigger”
  - “Record button”
  - access control gadget

Permission widgets: How?

- Grant: once, session, scheduled, permanent...
- Convey semantics clearly to user
- Identifiability vs. customizability?
How to realize permission widgets?

- How to make them unforgeable and unobscuable?
- What can be done without OS support?

4. Automatic granting

Grant requested permissions

- ... for low risk and reversible permissions

- ... but allow for auditability
  - Letting user figure out if app abuses permission

Thompson et al, “When it’s better to ask forgiveness than get permission: attribution mechanisms for smartphone resources”, SOUPS 2013
Allowing for auditability

Show who was responsible for a change (e.g., notification) e.g., notification shows which app is vibrating phone
Allowing for auditability

Show who was responsible for a change (e.g., notification)
e.g., notification shows which app changed wallpaper
Is attribution effective?

- Will users notice attribution indicators?
- Will they identify the apps responsible?

- Controlled laboratory study
Testing effectiveness

- What to test?
  - Pilot study, questionnaires, ...

- Experiment design
  - Avoid influence of other factors
    - E.g., only one app with wallpaper permission
  - Control condition vs. experiment condition

Thompson et al, “When it’s better to ask forgiveness than get permission: attribution mechanisms for smartphone resources”, SOUPS 2013
Testing security usability

- Moral hazard
  - Taking risks because of lack of consequences
  - E.g., lending test devices to participants
- Priming and self-reporting
  - Saying/doing what is expected
  - Example priming: saying “we are testing whether people choose strong passwords”

Ecological invalidity: test vs real life mismatch
Exercise: self-reporting

You want to find out the rate of mobile malware payloads delivered via adult websites. For this you need to know what proportion of infected users visited adult websites. For privacy reasons, you cannot automatically collect data about websites visited by users. You are only allowed to ask them (i.e., “self-reporting”)

How will you formulate your question in order to get an accurate measure for fraction of users visiting adult websites?
Mordac, the Preventer of Information Services.

Security is more important than usability.

In a perfect world, no one would be able to use anything.

To complete the log-in procedure, stare directly at the sun.
Exercise: self-reporting

You want to find out the rate of mobile malware payloads delivered via adult websites. For this you need to know what proportion of infected users visited adult websites. For privacy reasons, you cannot automatically collect data about websites visited by users. You are only allowed to ask them (i.e., “self-reporting”)

How will you formulate your question in order to get an accurate measure for fraction of users visiting adult websites?
Exercise: self-reporting

- How to extract true statistics from self-reported responses to sensitive questions?
- Hint 1: ask a more general question
- Hint 2: divide your sample into two groups; ask each group a different general question
Improving usability

1. Provide more context in prompts
   Annotations with useful information

2. Time of granting: Install time vs. Run time

3. Implicit granting via trusted UIs

4. Automatic granting + auditability
Choosing granting mechanism (1/3)

Revertible? (can action be undone easily?)

- Yes
- No

Not severe? (not abuse, just annoyance?)

- Yes
- No

Automatic grant + Auditability

Adapted from Porter Felt et al., HotSec '12
Choosing granting mechanism (2/3)

User Initiated? (did user initiate?)

No

Alterable? (can user change parameters?)

No

Yes

Yes

Trusted UI

Adapted from Porter Felt et al., HotSec ‘12
Choosing granting mechanism (3/3)

Transparent?
(does action need to work without immediate user involvement?)

No ➔ Runtime confirmation

Yes ➔ Install-time granting

Adapted from Porter Felt et al., HotSec '12
Why is usable mobile security different?
Your mobile phone: Not a smaller version of your PC
Your mobile phone: Not a smaller version of your PC

Mobile phone applications have different requirements due to

1. Smaller physical screen size
   → Less room for security indicators, notifications etc.
Your mobile phone: Not a smaller version of your PC

Mobile phone applications have different requirements due to
1. Smaller physical screen size
2. Different input mechanisms

- Directional pad + keyboard
- Touch screen
- Keyboard + mouse + ...
Your mobile phone: Not a smaller version of your PC

Mobile phone applications have different requirements due to

1. Smaller physical screen size
2. Different input mechanisms
3. Limited battery life
4. More prone to theft/loss
5. Slower and less reliable network connectivity
6. (Comparatively) limited computational power
Other usable security problems
Local user authentication

Need alternatives that are:
• Faster
• More enjoyable
• Secure enough

Biometrics
Wearables

Cost: users avoid using apps that mandate local authentication (work e-mail!)

Cost: weak PINs

Dunphy et al, “Shoulder-surfing resistance of authentication based on image recognition”, SOUPS 2010
Local user authentication: a cautionary tale

@koush Nope. Give us some credit.

http://youtu.be/BwfYSR7HttA
CAPTCHA on mobile devices

Cost:
Estimated 15% drop-off rate when encountering a CAPTCHA on mobile devices

http://antigate.com (not active anymore!)
CAPTCHA Alternatives

- The problem is real
- Avoid CAPTCHA?
  - Device authentication
  - reCAPTCHA

https://support.google.com/recaptcha/?hl=en
Other problem instances

- (Permission granting to apps)
- Local user authentication
- CAPTCHA
- Secure First Connect
- Context-specific access control
- ...?
Mobility helps security

- Mobility/portability can help in surprising ways: e.g.,
  - PayPal Bump
  - Čapkun et al, "Mobility helps security in ad hoc networks", MobiHoc ’03
  - ...

- Mobiles sense location, motion, light/sound, ...
  - Use cues from context/history to set sensible access policies? ("Contextual Security")
An example: Device Lock

- Intended for theft protection
- Example of one-size-fits-all
  - Lock always kicks in
- Can be annoying in
  - Freezing weather
  - Groggy mornings
  - ...


Better Device Lock via Context Profiling

Timeout and unlocking method adjusted based on estimated familiarity/safety of current context

Long timeout

Medium timeout

Short timeout

Home

Work Cafeteria

Unknown
Familiarity of people, things & places

Devices are proxies for people
Detect nearby devices & keep track of encounters
Identify places (“contexts”) meaningful to user

M. Miettinen et al, “ConXsense: automated context classification for context-aware access control” ACM ASIACCS ’14
Familiarity of people, things & places

Estimate familiarity of a device in a context

Estimate context familiarity based on who/what is nearby

M. Miettinen et al, “ConXsense: automated context classification for context-aware access control” ACM ASIACCS ’14
Familiarity of people, things & places

Estimate familiarity of a device in a context

Estimate context familiarity based on who is nearby

How to estimate safety?

M. Miettinen et al, “ConXsense: automated context classification for context-aware access control” ACM ASIACCS ‘14
Did you learn:

- Improving usability of app authorization
- Other problem instances of usable mobile security
Plan for the course

- Lecture 1: Platform security basics
- Lecture 2: Case study – Android
- Lecture 3: Mobile platform security
- Lecture 4: Hardware security enablers
- Lecture 5: Usability of platform security
- Extra lecture: IoT Security
- Invited lecture: SE Android policies
- Extra lecture: Machine learning and security
- Lecture 6: Summary and outlook