CSci 4271W
Development of Secure Software Systems
Day 24: Human factors part 2: general best practices

Stephen McCamant (he/him)
University of Minnesota, Computer Science & Engineering

Based in large part on slides originally by Prof. Nick Hopper
Licensed under Creative Commons Attribution-ShareAlike 4.0

Human factors

Ultimately, most computing systems will involve people at some point. How do we design security mechanisms that take the needs, abilities and goals of people into account?



Photo credits via freepik.com: rawpixel.com, wayhomestudio, rawpixel.com

What are we building? (1)

Three primary kinds of interactions occur in user interactions for security:

- Authentications prove that a person can access a computer, application, or resource
- Warnings inform a person that an action will or could have security consequences
- Configurations allow a person to make decisions about the security policy of a system

What are we building? (2)

Configurations can include:

- Configuration of software settings
- Consenting to terms
- Authorization of permission settings
- Verification of settings or claims
- Auditing the state of the system

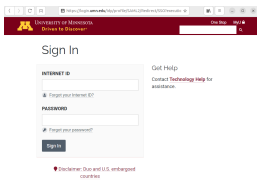
Challenges with users

Challenges with users

Conditioning: people learn to respond to frequent stimuli

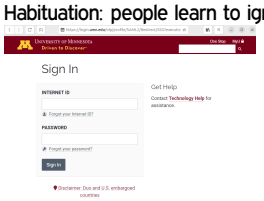
Challenges with users

Conditioning: people learn to respond to frequent stimuli



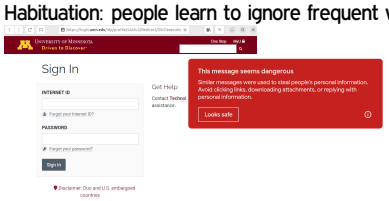
Challenges with users

Conditioning: people learn to respond to frequent stimuli
Habituation: people learn to ignore frequent warnings



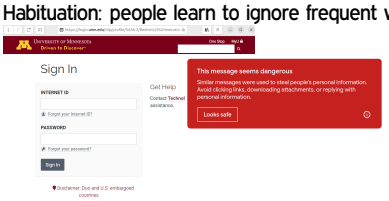
Challenges with users

Conditioning: people learn to respond to frequent stimuli
Habituation: people learn to ignore frequent warnings



Challenges with users

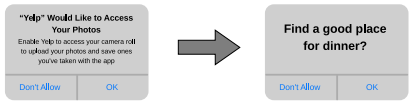
Conditioning: people learn to respond to frequent stimuli
Habituation: people learn to ignore frequent warnings



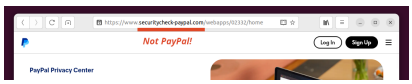
Wicked environment: no way to tell when a decision was bad

User behavior

Goal Orientation: when people are using computers, they are trying to achieve a task.



Confirmation Bias: people look for information that confirms their expectations.



What can go wrong

Channel of contact	Thing spoofed	Persuasion to interact	Human act exploited	Technical spoofing
Email	UI element	Greed	Open doc	System dialog
Website	Product or service	Fear	Click link	Filename
Social Network	Person you know	Social relationship	Attach device	File type
IM	Organization	Business relationship	Run program	Icon
Physical	Person you don't know	Curiosity	Enter credentials	Filename (multilingual)
	An authority	Lust	Establish relationship	

Outline

Review: what we're building

Announcements intermission

General best practices

Bonus: DNSSEC and ceremonies

Upcoming activities

- Homework 6 is due Tuesday night 4/29
- Project part 3 materials and assignments posted
  - One section draft due Thursday 5/1
  - Final report due Monday 5/5, no extensions
- Final exam Saturday 5/10

Outline

Review: what we're building

Announcements intermission

General best practices

Bonus: DNSSEC and ceremonies

Threat elicitation

How do we find out what can go wrong in our system?

Ceremonies

Explicitly model users' role(s) in protocols

Can be used to expose potential attacks due to flows involving humans. Threats to consider:

- Missing information
- Distracting information
- Underspecified elements
- Fuzzy comparison: Quick, make sure that

`f51c0904ec22a44453dd1651 == f51c0904ec22a24453dd1651`

Ceremonies

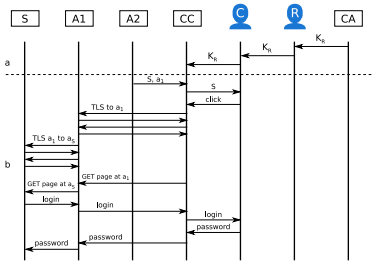
Explicitly model users' role(s) in protocols

Can be used to expose potential attacks due to flows involving humans. Threats to consider:

- Missing information
- Distracting information
- Underspecified elements

`f51c0904ec22a44453dd1651 ==`
`f51c0904ec22a24453dd1651`

Ceremony example (Ellison)

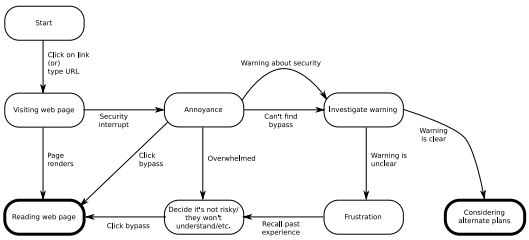


Source: [Ellison, 2007] via Shostack

Human models

- A mental/emotional state machine can help explain user reactions.
- Related to "Cognitive Walkthrough" technique for pre-user study evaluation of interfaces (soon)

Human model (example)



Source: Shostack Figure 15-4

Cognitive walkthrough

"Walk through" a task, at each step attempting to answer the questions:

- "Can a user identify the right next step?"
- "How will the user know they did the right thing?"

Record reactions at each step of the process.

Cognitive walkthrough

"Walk through" a task, at each step attempting to answer the questions:

- "Can a user identify the right next step?"
- "How will the user know they did the right thing?"

Record reactions at each step of the process.

Example: block a site from using JavaScript in Chrome.

Building effective mitigations

Non-productive myths:

Building effective mitigations

Non-productive myths:

- "People will choose dancing pigs every time."

Building effective mitigations

Non-productive myths:

- "People will choose dancing pigs every time."
- "People don't care about security"

Building effective mitigations

Non-productive myths:

- "People will choose dancing pigs every time."
- "People don't care about security"
- "People just don't listen"

Building effective mitigations

Non-productive myths:

- "People will choose dancing pigs every time."
- "People don't care about security"
- "People just don't listen"
- "My mom couldn't understand that"

Good decisions: design patterns (1)

Minimize what you ask of people

- Make a list: what do they need to know? How will they find out?
- Be consistent: treat similar situations with similar requests, use the same interface, etc.

Good decisions: design patterns (1)

Minimize what you ask of people

- Make a list: what do they need to know? How will they find out?
- Be consistent: treat similar situations with similar requests, use the same interface, etc.

Force people to complete important steps

- Like stepping on the brake to start a car; make it easier to do the safe thing than avoid it altogether.

Good decisions: design patterns (2)

Avoid urgency

- No links in emails, always ask to use the app or a bookmark.
- Never require users to "opt out" of a change.

Good decisions: design patterns (2)

Avoid urgency

- No links in emails, always ask to use the app or a bookmark.
- Never require users to "opt out" of a change.

Easy path to safety

- Make all communication available through the app or account
- Remind users about this at login and in emails

Kind(er) environment

No "scamicry"

- Communicate through known, trusted channels only
- No urgent callbacks, no emails with links
- Never start with a request for personal data

Kind(er) environment

No "scamicry"

- Communicate through known, trusted channels only
- No urgent callbacks, no emails with links
- Never start with a request for personal data

Give good advice

- Realistic
- Memorable
- Concise
- Durable
- Proved effective
- Consistent

Outline

Review: what we're building

Announcements intermission

General best practices

Bonus: DNSSEC and ceremonies

DNSSEC goals and non-goals

+ Authenticity of positive replies
+ Authenticity of negative replies
+ Integrity
– Confidentiality
– Availability

First cut: signatures and certificates

- Each resource record gets an `RRSIG` signature
  - E.g., `A` record for one name→address mapping
  - Observe: signature often larger than data
- Signature validation keys in `DNSKEY` RRs
- Recursive chain up to the root (or other "anchor")

Add more indirection

- DNS needs to scale to very large flat domains like `.com`
- Facilitated by having single `DS` RR in parent indicating delegation
- Chain to root now includes `DS`es as well

Negative answers

- Also don't want attackers to spoof non-existence
  - Gratuitous denial of service, force fallback, etc.
- But don't want to sign "`x` does not exist" for all `x`
- Solution 1, `NSEC`: "there is no name between `acacia` and `baobab`"

Preventing zone enumeration

- Many domains would not like people enumerating all their entries
- DNS is public, but "not that public"
- Unfortunately `NSEC` makes this trivial
- Compromise: `NSEC3` uses password-like salt and repeated hash, allows opt-out

DANE: linking TLS to DNSSEC

- "DNS-based Authentication of Named Entities"
- DNS contains hash of TLS cert, don't need CAs
- How is DNSSEC's tree of certs better than TLS's?

Deployment

- Standard deployment problem: all cost and no benefit to being first mover
- Servers working on it, mostly top-down
- Clients: still less than 40%
- Will probably be common for a while: insecure connection to secure resolver

Signing the root

- Political problem: many already distrust US-centered nature of DNS infrastructure
- Practical problem: must be very secure with no single point of failure
- Finally accomplished in 2010
  - Solution involves key ceremonies, international committees, smart cards, safe deposit boxes, etc.

Current state of root signing

- Carefully designed ceremony managed by ICANN with community participants
- Happens quarterly in LA-area or Virginia data centers
- Key goals are transparency and avoiding single points of failure or attack
- Livestreamed on YouTube, coincidentally today