Know your audience: terminology

- When technical terminology makes your point clearly, use it.
- But provide definitions if a concept might be new to many readers.
  - Be careful to provide the right information in the definition.
  - Define at the first instead of a later use.
- On other hand, avoid introducing too many new terms.
  - Keep the same term when referring to the same concept.

Precise explanations

- Don't say “we” do something when it's the computer that does it.
  - And avoid passive constructions.
- Don't anthropomorphize (computers don't “know”).
- Use singular by default so plural provides a distinction:
  - The students take tests
  - Each student takes a test
  - Each student takes multiple tests

Provide structure

- Use plenty of sections and sub-sections.
- It's OK to have some redundancy in previewing structure.
- Limit each paragraph to one concept, and not too long.
  - Start with a clear topic sentence.
  - Split long, complex sentences into separate ones.

Plagiarism and citations

- Never use someone else's writing to make it look like your own.
  - Overlaps with but different than cheating.
- Give proper credit for ideas that you get from somewhere else.
  - For 4271, mostly don't need to credit course resources.
  - We have no specific requirements about citation format.

Know your audience: Project

- For projects in this course, assume your audience is another student who already understands general course concepts.
  - Up to the current point in the course.
  - I.e., don't need to define “buffer overflow” from scratch.
- But you need to explain specifics of bcimgview.
  - Make clear what part of the program you're referring to.
  - Explain all the specific details of a vulnerability.

Inclusive language

- Avoid words and grammar that implies relevant people are male.
- My opinion: avoid using he/him pronouns for unknown people.
- Some possible alternatives:
  - “he/she”
  - Alternating genders.
  - Rewrite to plural and use “they” (may be less clear).
  - Singular “they” (least traditional, but spreading).
Per-website authentication

Many web sites implement their own login systems
+ If users pick unique passwords, little systemic risk
  - Inconvenient, many will reuse passwords
  - Lots of functionality each site must implement correctly
  - Without enough framework support, many possible pitfalls

Building a session

HTTP was originally stateless, but many sites want stateful login sessions
- Built by tying requests together with a shared session ID
- Must protect confidentiality and integrity

Session ID: what

- Must not be predictable
  - Not a sequential counter
- Should ensure freshness
  - E.g., limited validity window
- If encoding data in ID, must be unforgeable
  - E.g., data with properly used MAC
  - Negative example: crypt(username || server secret)

Session ID: where

- Session IDs in URLs are prone to leaking
  - Including via user cut-and-paste
- Usual choice: non-persistent cookie
  - Against network attacker, must send only under HTTPS
  - Because of CSRF, should also have a non-cookie unique ID

Session management

- Create new session ID on each login
- Invalidate session on logout
- Invalidate after timeout
  - Usability / security tradeoff
  - Needed to protect users who fail to log out from public browsers

Account management

- Limitations on account creation
  - CAPTCHA? Outside email address?
- See previous discussion on hashed password storage
- Automated password recovery
  - Usually a weak spot
  - But, practically required for large system

Client and server checks

- For usability, interface should show what’s possible
- But must not rely on client to perform checks
- Attackers can read/modify anything on the client side
- Easy example: item price in hidden field
Direct object references
- Seems convenient: query parameter names resource directly
  - E.g., database key, filename (path traversal)
- Easy to forget to validate on each use
- Alternative: indirect reference like per-session table
  - Not fundamentally more secure, but harder to forget check

Function-level access control
- E.g. pages accessed by URLs or interface buttons
- Must check each time that user is authorized
  - Attack: find URL when authorized, reuse when logged off
- Helped by consistent structure in code

Outline
Good technical writing (cont’d)
Web authentication
Announcements break
Names and identities
Usability and security

Supplementary office hour
Prof. McCamant on Friday, 3:30-4:30pm
Same Zoom room as regular office hours

Accounts versus identities
- "Identity" is a broad term that can refer to a personal conception or an automated system
- "Name" is also ambiguous in this way
- "Account" and "authentication" refer unambiguously to institutional/computer abstractions
- Any account system is only an approximation of the real world

Project report pre-submission
Available now, due date Friday night
Optional, not graded, feedback only on writing and presentation style

ROC space revisited
B return REJECT;
E return ACCEPT;
F if (rand() & 1) return ACCEPT; else return REJECT;
G if (pitch()) return ACCEPT; else return REJECT;
C if (iris()) return ACCEPT; else return REJECT;
A if (iris()) return REJECT; else return ACCEPT;
D if (iris() && pitch()) return ACCEPT; else return REJECT;
H if (iris() || pitch()) return ACCEPT; else return REJECT;
Real human names are messy
- Most assumptions your code might make will fail for someone
  - ASCII, length limit, uniqueness, unchanging, etc.
- So, don’t design in assumptions about real names
- Use something more computer-friendly as the core identifier
  - Make “real” names or nicknames a presentation aspect

Zooko’s triangle
- Claims (2001) it is hard/impossible for a naming scheme to be simultaneously:
  - Human-meaningful
  - Secure
  - Decentralized
- Too imprecise to be definitively proven/refuted
  - Blockchain-based name systems are highest-profile claimed counterexamples
- A useful heuristic for seeing design tensions

Identity documents: mostly unhelpful
- “Send us a scan of your driver’s license”
  - Sometimes called for by specific regulations
  - Unnecessary storage is a disclosure risk
  - Fake IDs are very common

Identity numbers: mostly unhelpful
- Common US example: social security number
- Variously used as an identifier or an authenticator
  - Dual use is itself a cause for concern
- Known by many third parties (e.g., banks)
- No checksum, guessing risks
- Published soon after a person dies

“Identity theft”
- The first-order crime is impersonation fraud between two other parties
  - E.g., criminal trying to get money from a bank under false pretenses
- The impersonated “victim” is effectively victimized by follow-on false statements
  - E.g., by credit reporting agencies
  - These costs are arguably the result of poor regulatory choices
- Be careful w/ negative info from 3rd parties

Backup auth suggestion: use time
- Need for backup often comes for infrequently-used accounts
- May be acceptable to slow down recovery if it reduces attack risk
  - Account recovery is a hassle anyway
- Time can allow legitimate owner to notice malicious request

Outline
Good technical writing (cont’d)
Web authentication
Announcements break
Names and identities
Usability and security

Users are not ‘ideal components’
- Frustrates engineers: cannot give users instructions like a computer
  - Closest approximation: military
- Unrealistic expectations are bad for security
Most users are benign and sensible
- On the other hand, you can't just treat users as adversaries
  - Some level of trust is inevitable
  - Your institution is not a prison
- Also need to take advantage of user common sense and expertise
  - A resource you can't afford to pass up

Don't blame users
- "User error" can be the end of a discussion
- This is a poor excuse
- Almost any "user error" could be avoidable with better systems and procedures

Users as rational
- Economic perspective: users have goals and pursue them
  - They're just not necessarily aligned with security
- Ignoring a security practice can be rational if the rewards is greater than the risk

Perspectives from psychology
- Users become habituated to experiences and processes
  - Learn "skill" of clicking OK in dialog boxes
- Heuristic factors affect perception of risk
  - Level of control, salience of examples
- Social pressures can override security rules
  - "Social engineering" attacks

User attention is a resource
- Users have limited attention to devote to security
  - Exaggeration: treat as fixed
- If you waste attention on unimportant things, it won't be available when you need it
- Fable of the boy who cried wolf

Research: ecological validity
- User behavior with respect to security is hard to study
- Experimental settings are not like real situations
- Subjects often:
  - Have little really at stake
  - Expect experimenters will protect them
  - Do what seems socially acceptable
  - Do what they think the experimenters want

Research: deception and ethics
- Have to be very careful about ethics of experiments with human subjects
  - Enforced by institutional review systems
- When is it acceptable to deceive subjects?
  - Many security problems naturally include deception