Writing in CS versus other writing

- Key goal is accurately conveying precise technical information
- More important: careful use of terminology, structured organization
- Less important: writer's personality, persuasion, appeals to emotion

Still important: concise expression

- Don't use long words or complicated expressions when simpler ones would convey the same meaning.
  Examples:
  - necessitate
  - utilize
  - due to the fact that
- Beneficial for both clarity and style

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

Know your audience: Project 1

- 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
  - 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)

Outline
- Good technical writing (pt. 1)
- Logistics announcements
- Program privileges with setuid
- Shell code injection and related threats
- More Unix permissions

Another supplemental office hour
- My last office hour before the project 1 submission will be 1-2pm on Friday
- Please also keep using Piazza

Feedback on Saugata’s TA performance
- Anonymous survey on how Saugata is doing as a TA
- Your feedback helps his development and the rest of the semester
- https://forms.gle/ANiy6hR1mdJmfULp8

Outline
- Good technical writing (pt. 1)
- Logistics announcements
- Program privileges with setuid
- Shell code injection and related threats
- More Unix permissions

Process UIDs and setuid(2)
- UID is inherited by child processes, and an unprivileged process can't change it
- But there are syscalls root can use to change the UID, starting with setuid
- E.g., login program, SSH server

Setuid programs, different UIDs
- If 04000 “setuid” bit set, newly exec'd process will take UID of its file owner
  - Other side conditions, like process not traced
- Specifically the effective UID is changed, while the real UID is unchanged
  - Shows who called you, allows switching back

What is setuid good for?
- Setuid allows a user’s privilege to be granted to a program
- Using a setuid program, users can do things they couldn't do directly
- The program is responsible for using the privilege correctly
Setuid and security risk

- Bugs in a setuid program are more likely to be security vulnerabilities
- Subverting a setuid program provides undeserved privilege
- Authors of setuid programs need to be very careful about secure programming

Outline

- Good technical writing (pt. 1)
- Logistics announcements
- Program privileges with setuid
- Shell code injection and related threats
- More Unix permissions

Two kinds of privilege escalation

- Local exploit: give higher privilege to a regular user
  - E.g., caused by bug in setuid program or OS kernel
- Remote exploit: give access to an external user who doesn't even have an account
  - E.g., caused by bug in network-facing server or client

Shell code injection

- The command shell is convenient to use, especially in scripts
  - In C: `system`, `popen`
- But it is bad to expose the shell's power to an attacker
- Key pitfall: assembling shell commands as strings
- Note: different from binary "shellcode"

Shell code injection example

- Benign: `system("cp $arg1 $arg2"), arg1 = "file1.txt"
- Attack: `arg1 = "a b; echo Gotcha"
- Command: "cp a b; echo Gotcha file2.txt"
- Not a complete solution: prohibit ‘;’

The structure problem

- What went wrong here?
- Basic mistake: assuming string concatenation will respect language grammar
  - E.g., that attacker supplied "filename" will be interpreted that way

Best fix: avoiding the shell

- Avoid letting untrusted data get near a shell
- For instance, call external programs with lower-level interfaces
  - E.g, `fork` and `exec` instead of `system`
- May constitute a security/flexibility trade-off

Less reliable: text processing

- Allow-list: known-good characters are allowed, others prohibited
  - E.g, username consists only of letters
- Deny-list: known-bad characters are prohibited, others allowed
  - Easy to miss some bad scenarios
- "Sanitization": transform bad characters into good
  - Same problem as deny-list, plus extra complexity
Terminology note
- Historically the most common terms for allow-list and deny-list have been "whitelist" and "blacklist" respectively.
- These terms have been criticized for a problematic "white = good", "black = bad" association.
- The push to avoid the terms got significant additional attention this summer, but is still somewhat politicized.

Different shells and multiple interpretation
- Complex Unix systems include shells at multiple levels, making these issues more complex.
  - Frequent example: `scp` runs a shell on the server, so filenames with whitespace need double escaping.
  - Other shell-like programs also have caveats with levels of interpretation.
  - Tcl before version 9 interpreted leading zeros as octal.

Related local dangers
- File names might contain any character except `/` or the null character.
- The `PATH` environment variable is user-controllable, so `cp` may not be the program you expect.
- Environment variables controlling the dynamic loader cause other code to be loaded.

IFS and why it was a problem
- In Unix, splitting a command line into words is the shell's job.
  - `String -> argv array`
  - `grep a b c vs. grep 'a b' c`
- Choice of separator characters (default space, tab, newline) is configurable.
- Exploit `system("/bin/uname")`
- In modern shells, improved by not taking from environment.

Outline
- Good technical writing (pt. 1)
- Logistics announcements
- Program privileges with `setuid`
- Shell code injection and related threats
- More Unix permissions

More different UIDs
- Two mechanisms for temporary switching:
  - Swap real UID and effective UID (BSD)
  - Remember saved UID, allow switching to it (System V)
- Modern systems support both mechanisms at the same time.

Setgid, games
- Setgid bit 02000 mostly analogous to `setuid`.
- But note no supergroup, so UID 0 is still special.
- Classic application: `setgid games` for managing high-score files.

Special case: `/tmp`
- We'd like to allow anyone to make files in `/tmp`.
- So, everyone should have write permission.
- But don't want Alice deleting Bob's files.
- Solution: "sticky bit" 01000.
### Special case: group inheritance

- When using group to manage permissions, want a whole tree to have a single group.
- When 02000 bit set, newly created entries with have the parent's group.
  - (Historic BSD behavior)
- Also, directories will themselves inherit 02000.

### Other permission rules

- Only file owner or root can change permissions.
- Only root can change file owner.
  - Former System V behavior: “give away chown.”
- Setuid/gid bits cleared on chown.
  - Set owner first, then enable setuid.