CS220

Software Bugs
Agenda for Today

• Quiz 6
• Technical content
  – Software bugs
  – Auto Bug Detection
Terminology 1: the metrics

- Reliability
- Availability
- Dependability
Terminology 1: the metrics

• Reliability
  – How often does the system fail?
  – Mean time to failure
    • Average time between two failures

• Availability
  – How often is the system available?
  – Available Time / Total Time

• Dependability
  – Availability + Reliability + Security
Question

• How to improve availability with fixed reliability?
  – If we can fix a system faster, its availability would increase although its reliability is unchanged
Terminology 2: why does my PC stop?
Term. 2: why does my PC stop?

- Hardware problems
- Software problems
- Operator/configuration problems
Question

• How to compute system availability based on components’ availability?

(not discussed in lecture; won’t appear in exams)
Terminology 3: different stages

• Fault
• Error
• Failure

(not discussed in lecture; won’t appear in exams)
Software bugs

• What bugs have bothered you?
Fighting software bugs is crucial

• Software is everywhere

• Software bugs are widespread and costly
  – Lead to 40% system down time [Blueprints 2000]
  – Cost 312 Billion lost per year [Cambridge 2013]
Different types of bugs

• What types of bugs do you know?
How do we know what are real-world bugs?

https://bugzilla.mozilla.org/
Different types of bugs & examples

- Memory bugs
- Semantic bugs
- Concurrency bugs
- Performance/energy bugs
Memory bugs

• Buffer overflow
• Uninitialized read
• Dangling pointers
• Memory leaks
Examples
What are the impact of memory bugs?

• Buffer overflow
  – Data corruption
  – Security attack, particularly code injection
  – Program crash

• Uninitialized read & dangling pointer
  – Data corruption, program crash

• Memory leak
  – High memory usage → bad performance, crash
How to detect buffer overflow?

• The basic algorithm
  – monitor every memory allocation, pointer arithmetic, memory accesses
  – Maintain a big hash table ...

• A faster algorithm
  – Add padding around heap buffers
  – Demo: valgrind – tool = memcheck ...
  – How about stack buffer overflow detection?
    • Using stack canary to check whether return address has been changed or not
How to detect other mem. bugs?

• Uninitialized reads

• Dangling pointers

• Double free

• Memory leak

• Null pointer dereference (hmm)
What are the problems?

• Accuracy?
  – False positives

• Coverage?
  – False negatives

• Performance?
  – Overhead
Another fun bug story