

Fighting Software Bugs

Shan Lu
University of Chicago

1

你好，我叫卢山

2

UIUC



3

University of Wisconsin -- Madison



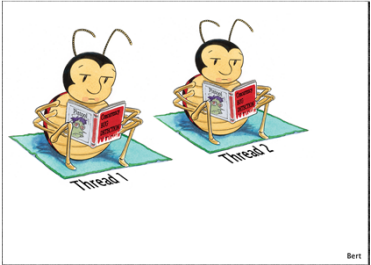
4

University of Chicago



5

My research is all about bugs




Bert

6


Software bugs

- How many of you have never written bugs?
- How many of you have been bothered by bugs?



7

I like interaction!



8

Outline

- Software dependability
- Software bugs
- Tackling memory bugs
 - Memory bug detection
 - Memory bug tolerance/survival

9

Software Dependability

10

Terminology 1: the metrics

- Reliability
- Availability
- Dependability

11

Terminology 1: the metrics

- Reliability
 - How often does the system fail?
 - MTF
- Availability
 - How often is the system available?
 - Available Time / Total Time
- Dependability
 - Availability + Reliability + Security

12

Question

- How to improve availability with fixed reliability?

13

Terminology 2: why does my PC stop?

14

Terminology 2: why does my PC stop?

- Hardware problems
- Software problems
- Operator/configuration problems

15

Terminology 2: why does my PC stop?

- Hardware problems
 - Types
 - Models
- Software problems
 - Types
 - Models
- Operator/configuration problems

16

Question

- How to compute system availability based on components' availability?

17

Terminology 3: different stages

- Fault
- Error
- Failure

18

Software Bugs

19

Software bugs

- Some more examples of software bugs?

20

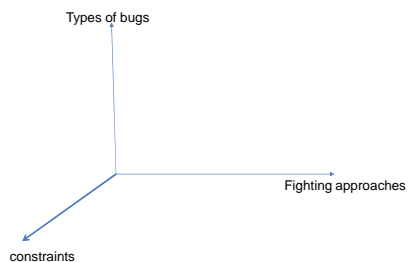
Fighting software bugs is crucial

- Software is everywhere
 - http://en.wikipedia.org/wiki/List_of_software_bugs
- Software bugs are widespread and costly
 - Lead to 40% system down time [Blueprints 2000]
 - Cost 312 Billion lost per year [Cambridge 2013]



22

The space of bug fighting



22

Different types of bugs

- What types of bugs do you know?

Faults in linux: ten years later. ASPLOS'11
Bug Characteristics in Open Source Software. EMSE '13

23

How do we know what are real-world bugs?

<https://bugzilla.mozilla.org/>

24

Slide 21

SL1 add new figures. use new citations
Shan Lu, 2014-1-7

Different types of bugs & examples

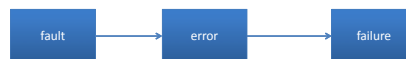
- Memory bugs
- Semantic bugs
- Concurrency bugs
- Performance/energy bugs

Faults in linux: ten years later. ASPLOS'11
Bug Characteristics in Open Source Software. EMSE'13

25

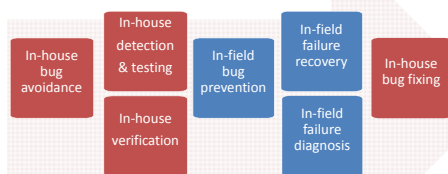
Approaches to tackling bugs

- What are the approaches?



26

Different aspects of fighting bugs



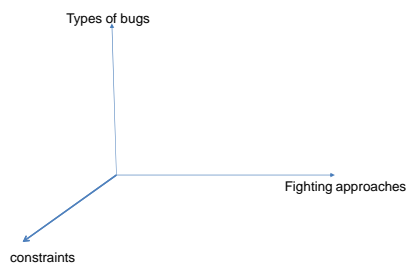
27

Questions

- What are the challenges/goals for designing each approach?

28

The space of bug fighting



29

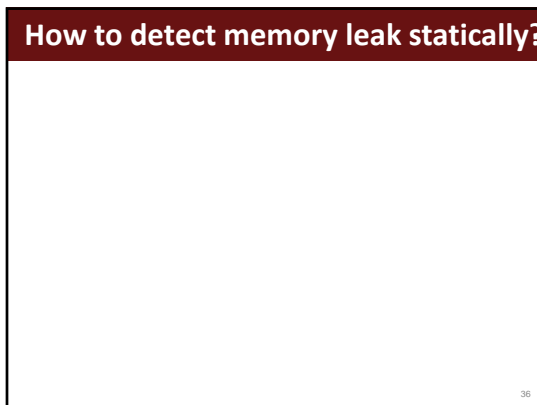
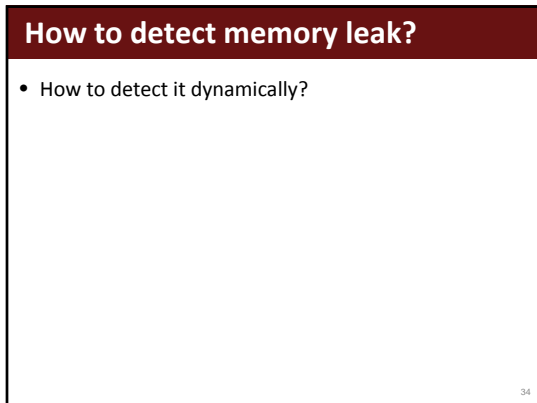
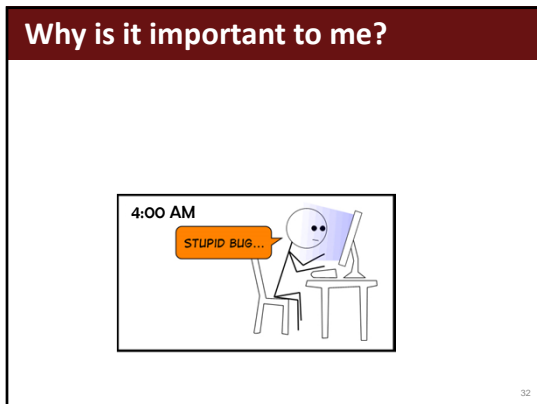
Tackling memory bugs (mainly C/C++)

- Why it is important to fight memory bugs
- How to detect memory bugs
- How to tolerate memory bugs at run-time?

30

Slide 27

SL3 ideally, this should be a cycle, but ...
Shan Lu, 2014-1-7



Static program analysis

- Program Dependency Graph
- Pointer-alias analysis
- Path sensitivity
- Inter-procedural vs. intra-procedural
- Advantage?
- Limitations?

37

Static vs. Dynamic Bug Detection

- Static analysis
 - Advantage: ??
 - Disadvantage: ??
- Dynamic analysis
 - Advantage: ??
 - Disadvantage: ??

38

How to do better?

- (will discuss later)

39

How to detect buffer overflow?

- How to detect it dynamically?

```
chat* p = malloc (10);
p[11] = 'A';
```

40

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?

41

How to detect other mem. bugs?

- Uninitialized reads
- Dangling pointers
- Double free
- Null pointer dereference (hmm)

42

What are the problems?

- Accuracy?
 - False positives
- Coverage?
 - False negatives
- Performance?
 - Overhead

43

How to do better?

- Hybrid analysis
- Hardware/OS support
 - How can I know whether a memory variable V is accessed at run time with low overhead?

44

How to do better?

- Hybrid analysis
- Hardware/OS support
 - How can I know whether a memory variable V is accessed at run time with low overhead?
 - Using page fault
 - Using ECC bits (my first project idea, yeah~~)
 - New hardware
 - iWatcher
 - [Intel memory protection extensions \(MPX\)](#)



45

Memory bug survival

46

How to survive/tolerate mem. bugs?

- Suppose we can detect buffer overflows at run time, what can I do to avoid security attack or crash? [Failure Oblivious Computing]

47

How to survive/tolerate mem. bugs?

- Can we improve failure oblivious computing to maintain program semantics? [Rx]
 - No need to do bug detection

48

What is next for F.O. and Rx?

- Approximate computing
- [Diehard](#)

49

Summary

- Dependability terminologies
- What are bugs?
- Basic techniques
 - Static/dynamic program analysis
 - Hardware/OS support
 - ML-ish techniques (not discussed yet)
- How to detect & survive memory bugs

50

Backup

51

Backup topics

- How did people solve stack overflows?
- Can heap overflow be exploit?
- Can we automatically fix memory bugs?
- How to detect semantic bugs?
- Background about threads & concurrency

52