Fuzzing

Announcements

- HW2 deadline extended
- Lab5 deadline extended (3/17)
  - Fix is to use java version 11.0.11
  - Lab instructions updated
  - Due after spring break
- Midterm check-in survey
  - https://docs.google.com/forms/d/e/1FAIpQLSe0h2xf-pr8mau3iO-iJea3hsshnqk-nud1T230IlvFIgiZtA/viewform
- Please submit group suggestions by 3/17

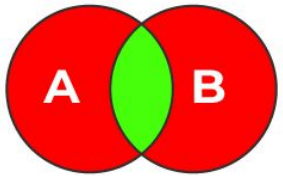
Overview

- Homework review
- Fuzzing
- Fuzzing class activity

Similarity Metric - Jaccard

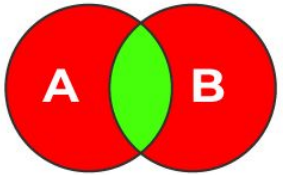
- Measures similarity between two sets
- Values range from o (no similarity) to 1 (identical sets)

$$J(X, Y) = |X \cap Y| / |X \cup Y|$$



$$\text{Jaccard} = \frac{\text{Intersection (A, B)}}{\text{Union (A, B)}}$$

Jaccard Coefficient



$$\text{Jaccard} = \frac{\text{Intersection (A, B)}}{\text{Union (A, B)}}$$

D1:

"Information Retrieval is useful"

D2:

"Retrieval of information is important"

{ information, retrieval, is useful }

{ retrieval, of, information, is, important }

$(A \cap B) = 3$

$(A \cup B) = 6$

$J(D1, D2) = 3 / 6 = 0.5$

Fuzzing

- Automated software testing technique
- Involves providing unexpected or invalid inputs and monitoring the programs behavior
- Typically refers to testing C programs

How is this different from randomized testing of Java programs?

```
public class Number {
    private int num;
    public Number(int num) {
        this.num = num;
    }
    public boolean isPrime() {
        if (num <= 1) return 0;

        for (int i = 2; i <= num; i++) {
            if (num % i == 0)
                return false;
        }

        return true;
    }
}
```

```
Scanner scanner = new Scanner(System.in);

int number = scanner.nextInt();

Number  n = new Number(number));

n.isPrime()
```

How is this different from randomized testing of Java programs?

```
int is_prime(int num) {
        if (num <= 1) return 0;
        for (int i = 2; i <= num; i++) {
                if (num % i == 0)
                        return 0;
        }
        return 1;
}
```

```
fgets(input, sizeof(input), stdin);
num = atoi(input);
is_prime(num)
```

The First Fuzzing Study

- Conducted by Barton Miller @ Univ of Wisconsin
- 1990: Command-line fuzzer, testing reliability of UNIX programs
    - Bombards utilities with random data
- 1995: Expanded to network protocols, file systems etc.
- Later: Expanded to GUI-based Windows programs, OS X apps, Android apps, object oriented programs

Fuzzing UNIX Utilities: Aftermath

- 1990: Caused 25-33% of UNIX utility programs to crash (dump state) or hang (loop indefinitely)

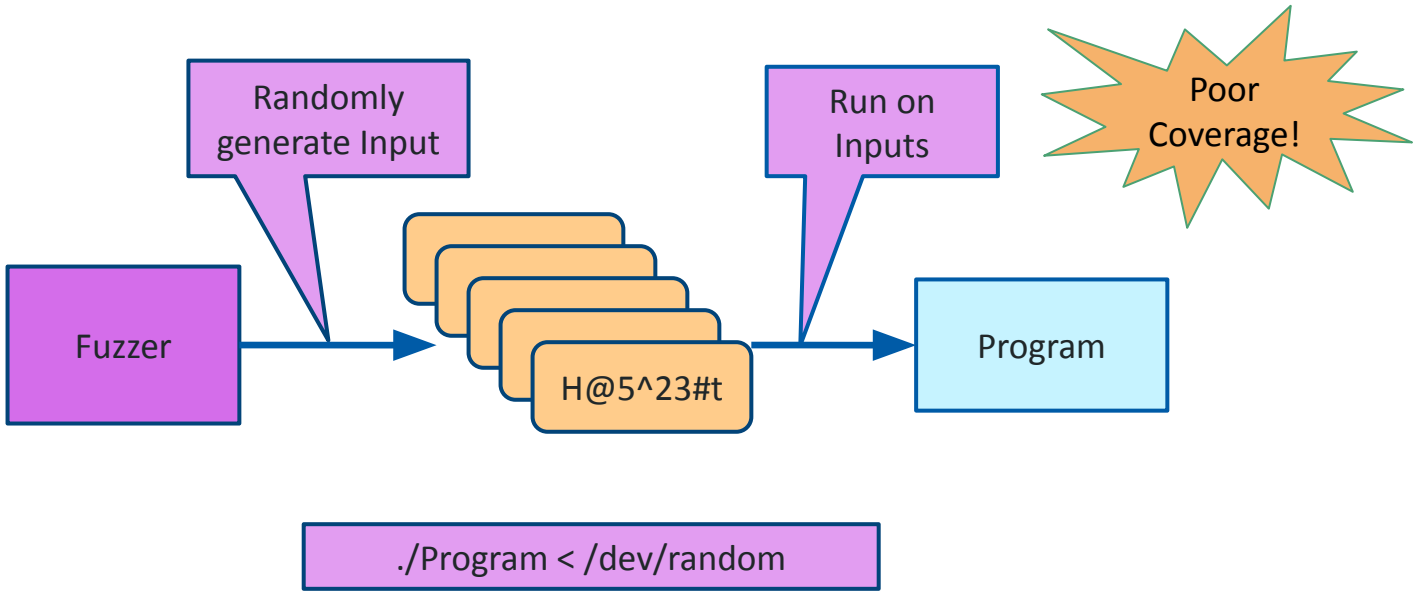
- 1995: Systems got better... but not by much!

  “Even worse is that many of the same bugs that we reported in 1990 are still present in the code releases of 1995.”

3 Generations of Fuzzers

1. Entirely Random
2. Smart Seed Inputs
3. Coverage Feedback

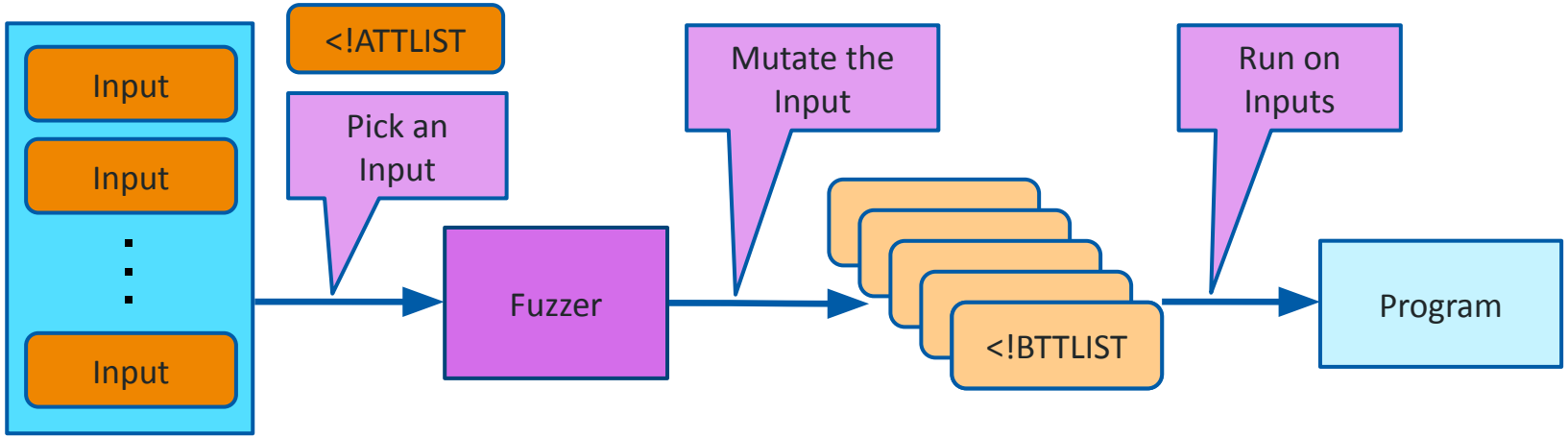
The First Generation



The Second Generation

```
JsonObject parse_json(String input) {

    If (!well_formed(input)) {
        return null;
    }

    if (has_a_list(input)) {
        //BUG
    }

    return JsonObject(input); //exits normally

}
```

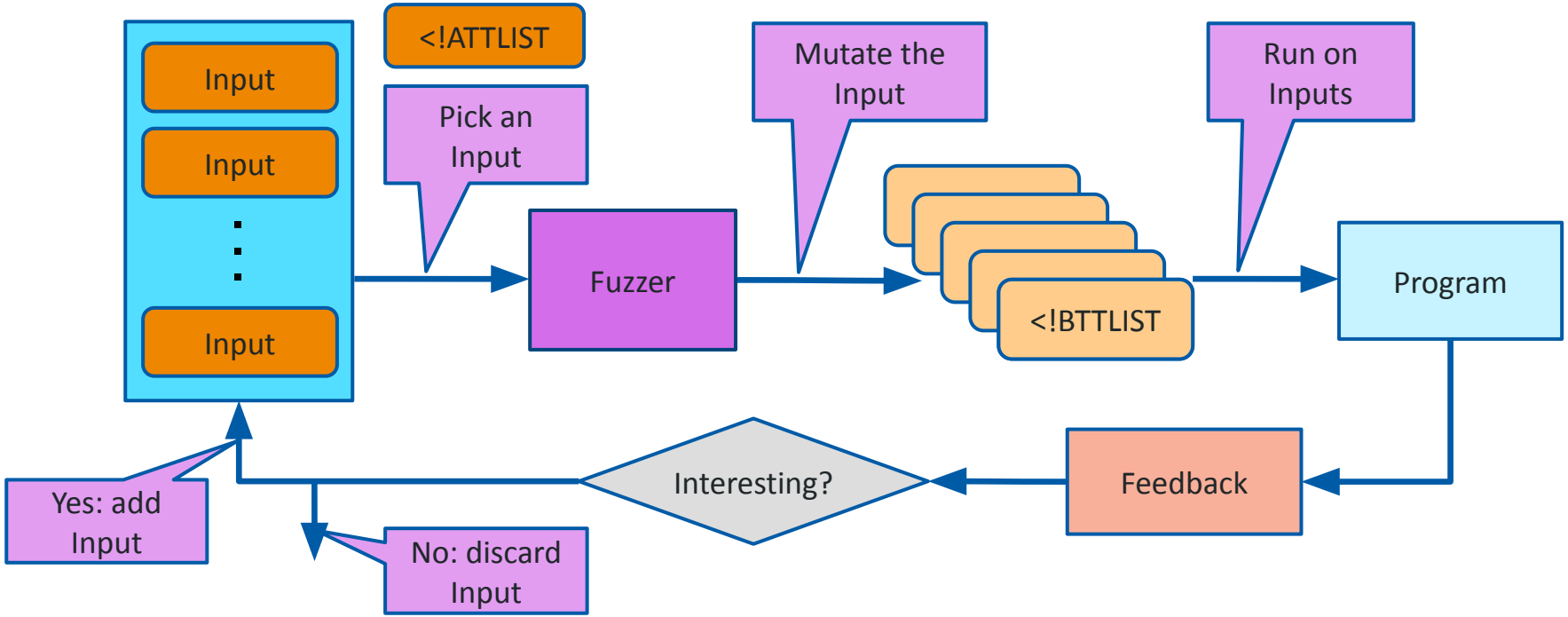
The Second Generation



The Second Generation

```
JsonObject parse_json(String input) {

    If (!well_formed(input)) {
        return null;
    }

    if (has_a_list(input)) {
        //BUG
    }

    return JsonObject(input); //exits normally

}
```

The Third Generation



What Kinds of Bugs can Fuzzing Find?

C Programs

- SegFault
- DivByZero
- Other memory safety violations…..
- Assertion violations

java

- Null Pointer Exception
- Divide by Zero
  - Assertion violations
- …

AFL finds real bugs!

Trophies

- VLC
  - CVE-2019-14437 CVE-2019-14438 CVE-2019-14498 CVE-2019-14533 CVE-2019-14534 CVE-2019-14535 CVE-2019-14776 CVE-2019-14777 CVE-2019-14778 CVE-2019-14779 CVE-2019-14970 by Antonio Morales (GitHub Security Lab)
- Sqlite
  - CVE-2019-16168 by Xingwei Lin (Ant-Financial Light-Year Security Lab)
- Vim
  - CVE-2019-20079 by Dhiraj (blog)
- Pure-FTPd
  - CVE-2019-20176 CVE-2020-9274 CVE-2020-9365 by Antonio Morales (GitHub Security Lab)
- Bftpd
  - CVE-2020-6162 CVE-2020-6835 by Antonio Morales (GitHub Security Lab)
- Tcpdump
  - CVE-2020-8036 by Reza Mirzazade
- ProFTPd
  - CVE-2020-9272 CVE-2020-9273 by Antonio Morales (GitHub Security Lab)
- Gifsicle
  - Issue 130 by Ashish Kunwar
- FFmpeg
  - Ticket 8592 Ticket 8593 Ticket 8594 Ticket 8596 by Andrea Fioraldi
  - Ticket 9099 by Qiuhao Li
- Glibc
  - Bug 25933 by David Mendenhall
- FreeRDP
  - CVE-2020-11095 CVE-2020-11096 CVE-2020-11097 CVE-2020-11098 CVE-2020-11099 CVE-2020-13397 CVE-2020-13398 CVE-2020-4030 CVE-2020-4031 CVE-2020-4032 CVE-2020-4033 by Antonio Morales (GitHub Security Lab)
- GNOME
  - Libxps issue 3 by Qiuhao Li
- QEMU
  - CVE-2020-29129 CVE-2020-29130 by Qiuhao Li
- GNU coreutils
  - Bug 1919775 by Qiuhao Li
- PostgreSQL
  - Crash while parsing zero-symbols in jsonb string by Nikolay Shaplov (Postgres Professional)

Activity

Running AFL on C programs with div-by-zero errors

Python is your friend!

Remember that newline counts as a character

Summary

- AFL is a practical tool for finding bugs in C programs
- Coverage guided

- HW2 due friday
- Please submit group member suggestions by 3/17