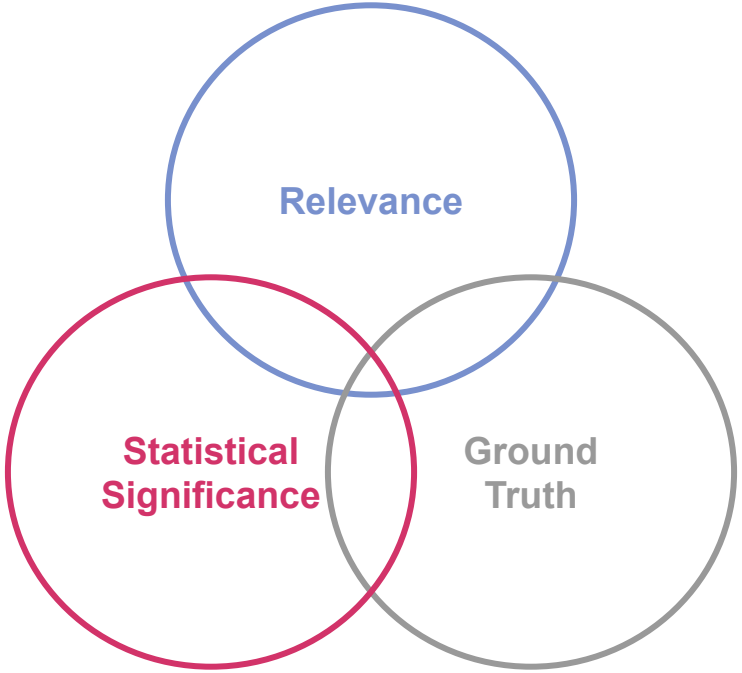
Bug Injection in SATE VI

SATE VI Workshop - September 19, 2019 - MITRE, McLean VA

https://samate.nist.gov/SATE.html

Aurelien Delaitre
Lead, SATE VI Classic Track
Prometheus Computing LLC

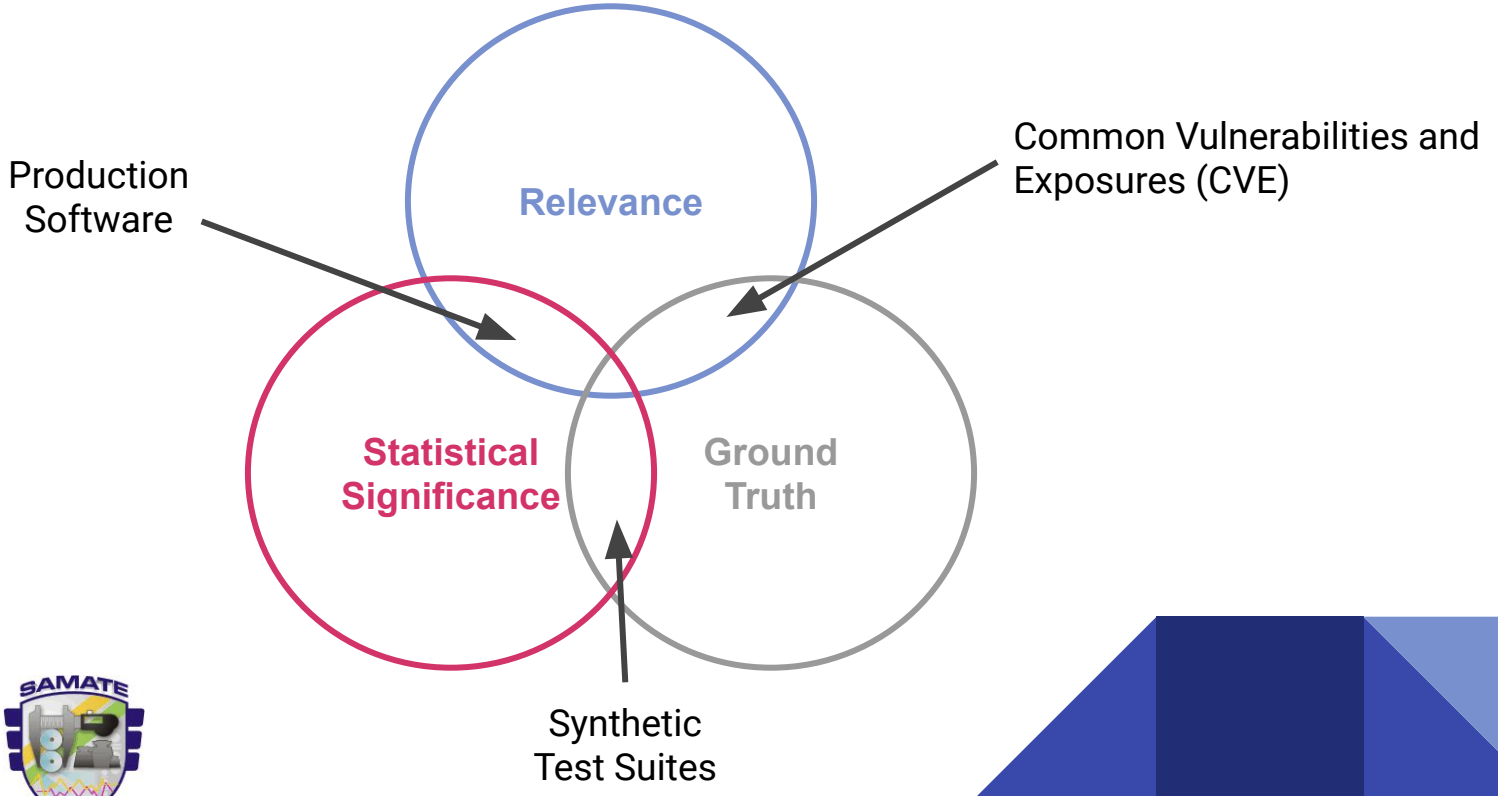
Why Bug Injection?



NIST National Institute of Standards and Technology U.S. Department of Commerce

SAMATE

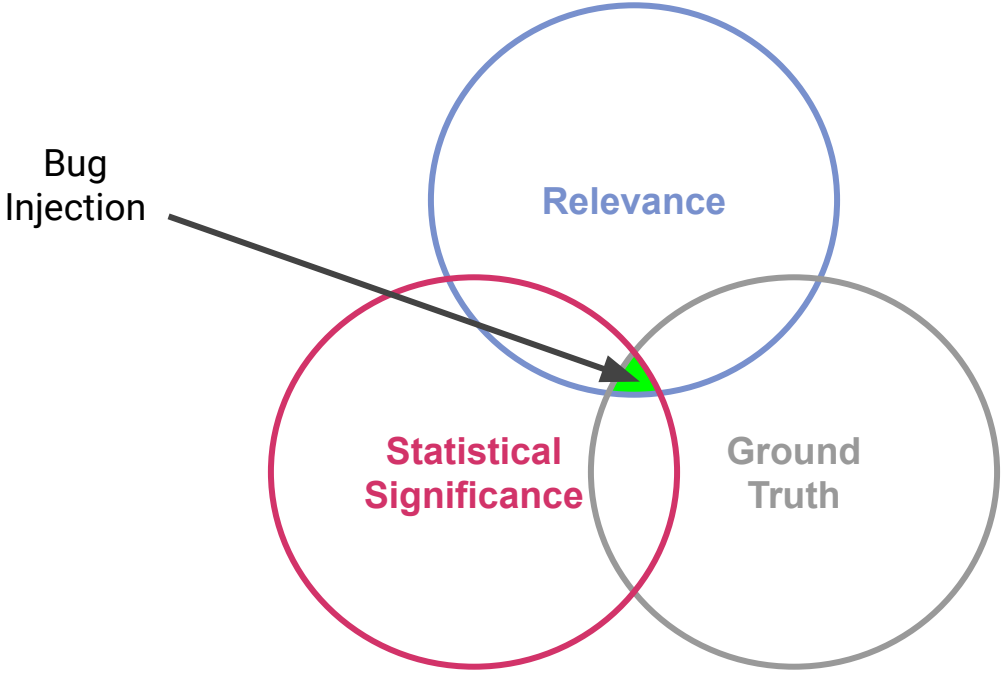
Why Bug Injection?



NIST National Institute of Standards and Technology U.S. Department of Commerce

SAMATE

Why Bug Injection?



National Institute of Standards and Technology
U.S. Department of Commerce

SAMATE

Ways to “Get” Bugs

- Bug Injectors
- Manual & Semi-Automated Injection
- Specifically Developed Test Suites
- Existing Bugs
  - Discovered
  - Undiscovered

Bug Types in SATE VI

C: Undefined Behavior

- Pointers
- Buffers
- Initialization

Java: Code Injection

- Cross-Site Scripting (XSS)
- SQL Injection

▶️ High-Impact
▶️ Easy to Prove

Proof of Vulnerability (PoV)

Why?

- Proves Bug Matters
- Retrieve Bug Trace

How?

- Fuzzing
- Bug Tracker
- Manual

Bug Traces

- Based on PoVs
  - C: GDB / Valgrind / ASAN
  - Java: Flow

- Manual Analysis
  - Doc Review
  - Code Review

What Went Wrong?

Cheap but Hard Bugs

packet-arp.c

```
mac = tvb_get_ptr(tvb, tha_offset, 6) + spa_offset * 64;
```

- Almost Never Found by Tools

Asymmetrical Bug/Fix Pairs

SimplePageBean.java

```
String query = "";
query = "INSERT INTO FORUM_SATE (ID, TITLE, BODY) VALUES('" + uuid + "', '" + title + "', 'body')";
statement = connection.createStatement();
int i = statement.executeUpdate(query);
connection.commit();
```

Buggy

```
query = "INSERT INTO FORUM_SATE (ID, TITLE, BODY) VALUES(?, ?, ?)";
statement = connection.prepareStatement(query);
statement.setString(1, uuid);
statement.setString(2, title);
statement.setString(3, "body");

statement.executeUpdate();
connection.commit();
```

Fixed

Buggy Bugs

global.c

```
int sqlite3PendingByte = 0x40000000;
```

fts3_write.c

```
/* from input (./gt-harness.sh test BIN 4039 1) */
printf("from input 4039");
/* POTENTIAL FLAW */vfsList = malloc(sqlite3PendingByte * sizeof(int));
```

- Tainted Data Questionable
- Unintended Bug Type
- Implementation-dependent
- Unknown Sink

Buggy Fixes

global.c

```
int sqlite3PendingByte = 0x40000000;
```

fts3_write.c

```
/* from input (./gt-harness.sh test BIN 4039 1) */
printf("from input 4039");
if (sqlite3PendingByte < 2147483647 / sizeof(int)) {
  vfsList = malloc(sqlite3PendingByte * sizeof(int));
}
```

- Tainted Data Questionable
- Condition Always False

National Institute of Standards and Technology
U.S. Department of Commerce

Buggy Bugs

global.c

```
int sqlite3PendingByte = 0x40000000;
```

pragma.c

```
/* from input (./gt-harness.sh test BIN 9300 1) */
printf("from input 9300");
int buf[100];
if (sqlite3PendingByte > 99) {
    /* POTENTIAL FLAW */
    buf[sqlite3PendingByte] = 1;
}
```

Tainted Data Questionable

Condition Always True

Buggy Fixes

global.c

```
int sqlite3PendingByte = 0x40000000;
```

pragma.c

```
/* from input (./gt-harness.sh test BIN 9300 1) */
printf("from input 9300");
int buf[100];
if (sqlite3PendingByte < 100) {
    /* POTENTIAL FLAW */
    buf[sqlite3PendingByte] = 1;
}
```

Tainted Data Questionable

Condition Always False

National Institute of Standards and Technology
U.S. Department of Commerce

Buggy Fixes

global.c

```
int sqlite3PendingByte = 0x40000000;
```

fts3_tokenize_vtab.c

```
/* from input (./gt-harness.sh test BIN 378 4) */
printf("from input 378");
if (pInfo != NULL && sqlite3_temp_directory != NULL && sizeof(pInfo) > sqlite3PendingByte
            && sizeof(sqlite3_temp_directory) > sqlite3PendingByte) {
  memcpy(pInfo, sqlite3_temp_directory, sqlite3PendingByte);
}
```

- Tainted Data Questionable
- Condition Always False Due to Programming Error

National Institute of Standards and Technology
U.S. Department of Commerce

Sink Separation

date.c

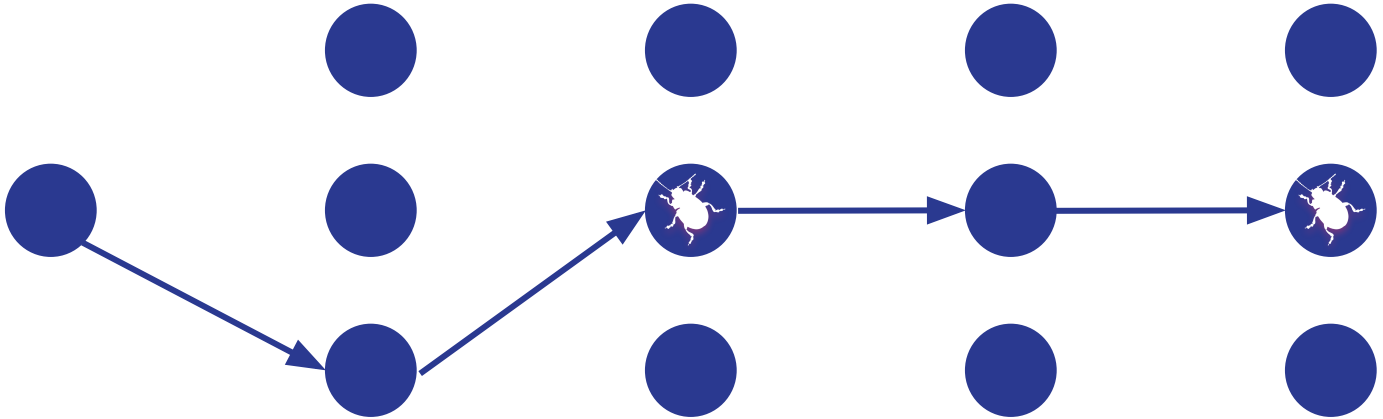
```
/* from input (./gt-harness.sh test BIN 9643 1) */
printf("from input 9643");
pRc = (int *)malloc(9*sizeof(int));
if (pRc != NULL) {
    int i;
    for (i = 0; i < 9; i+=1) {
        set_i(pRc, i);
        /* POTENTIAL FLAW */
        set_i(pRc, i + 1);
    }
}
```

insert.c

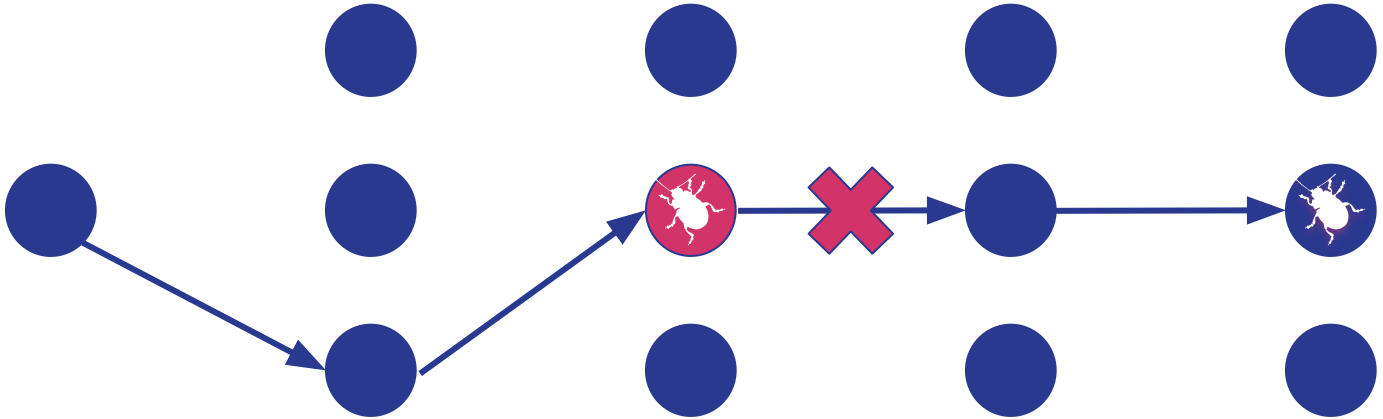
```
/* from input (./gt-harness.sh test BIN 8865 1) */
printf("from input 8865");
aiChng = (int *)malloc(9*sizeof(int));
if (aiChng != NULL) {
    int i;
    for (i = 0; i < 9; i+=1) {
        set_i(aiChng, i);
        /* POTENTIAL FLAW */
        set_i(aiChng, i + 1);
    }
}
```

```
void set_i(int *arr, int index) {
  arr[index] = 0;
}
```

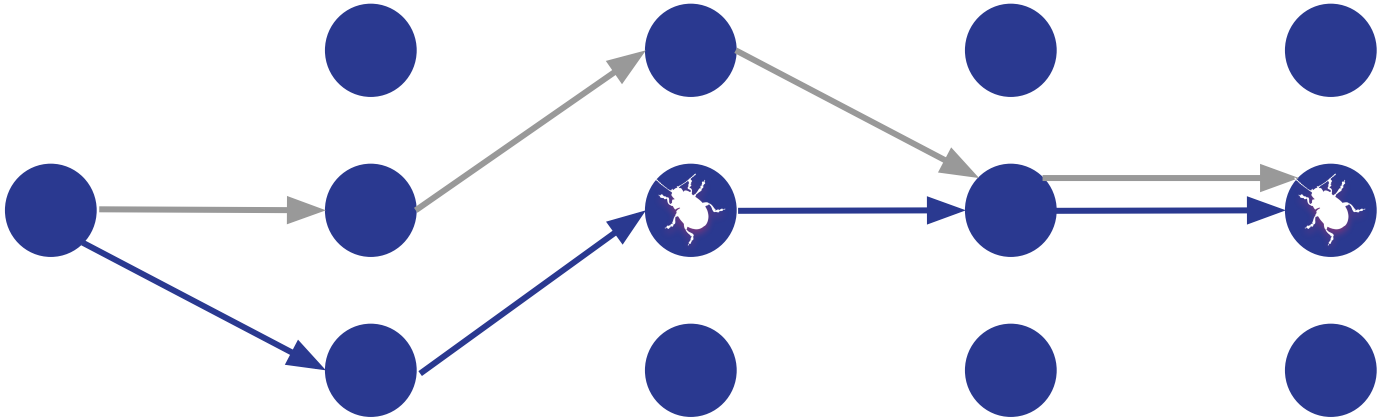
Shadowing



Shadowing



Shadowing



Take Away

National Institute of Standards and Technology
U.S. Department of Commerce