Industrial supervisor: Mr. Vadim OKUN
Academic supervisor: Mr. Charles DESPRES

How to inject quality bugs for Static Analysis Tool Exposition's test cases

Guillaume HABEN's Oral Defense

December 1, 2017

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

Lorraine INP

TELECOM nancy
Ingénieurs du numérique • Inspiring your digital future

"How do you change the world? Always work on something uncomfortably exciting"

– Larry Page, Alphabet CEO

Outline

Introduction

One of the biggest data breaches of all time

145 millions Americans affected

$59.5 billion annually

1. Context & environment

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

Software Assurance Metrics And Tool Evaluation

- Improving software assurance
- Measuring the effectiveness of tools

Static analysis

Input → Tool analysis → Output

Source code

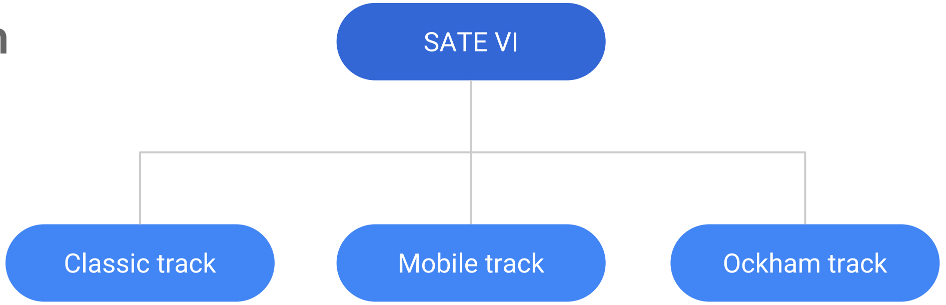
Warning reports

Limits

Complexity of real world software $\longrightarrow$ Use of approximations

Static Analysis Tool Exposition

- Encourage improvement of tools
- Speed tool adoption
- Enable empirical research

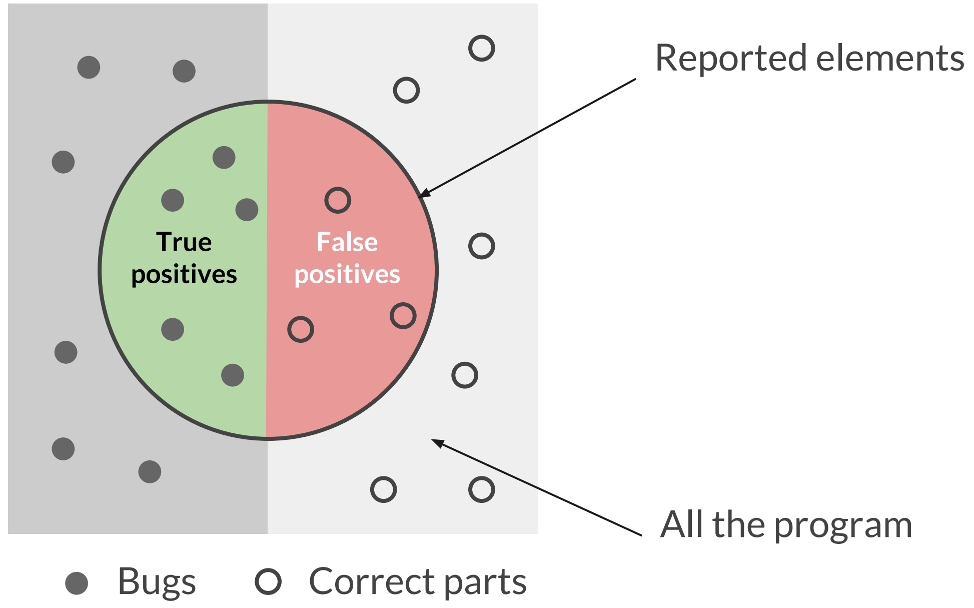


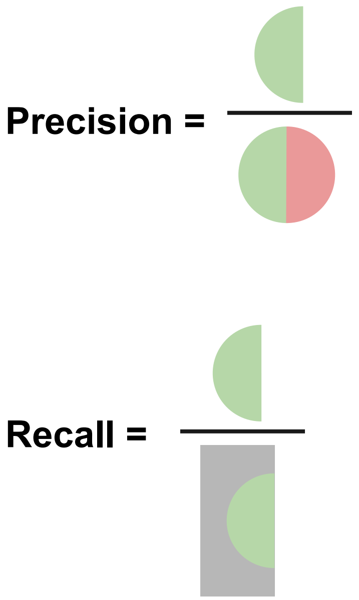
2. Presentation of the project

What do we want to know ?

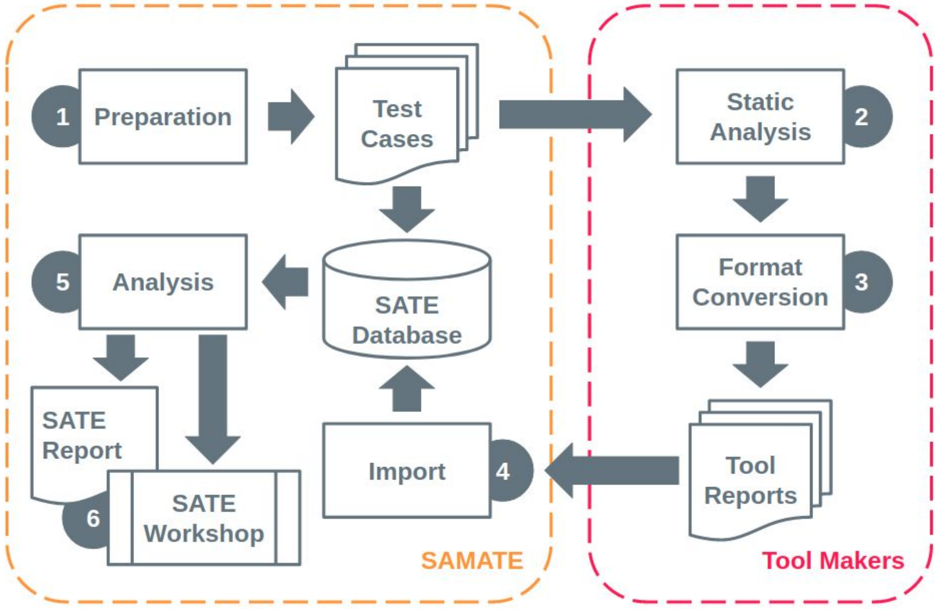
1. What proportion of defects can a tool find?
   Recall and coverage
2. How noisy is a tool?
   Precision and discrimination
3. How similar are unrelated tools?
   Overlap

Precision & Recall

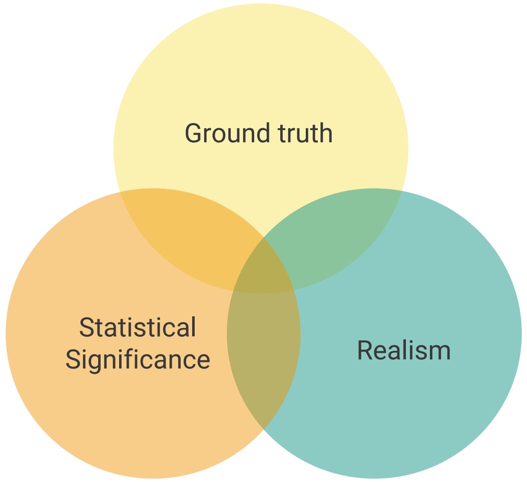




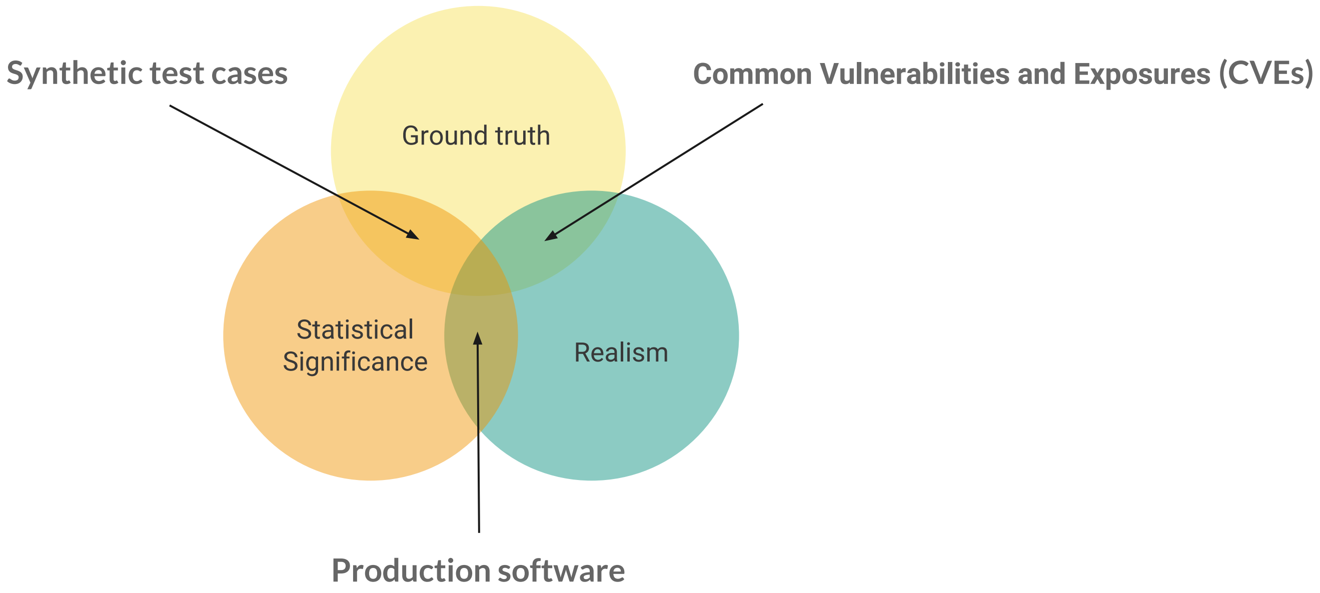
How to assess static analyzers ?



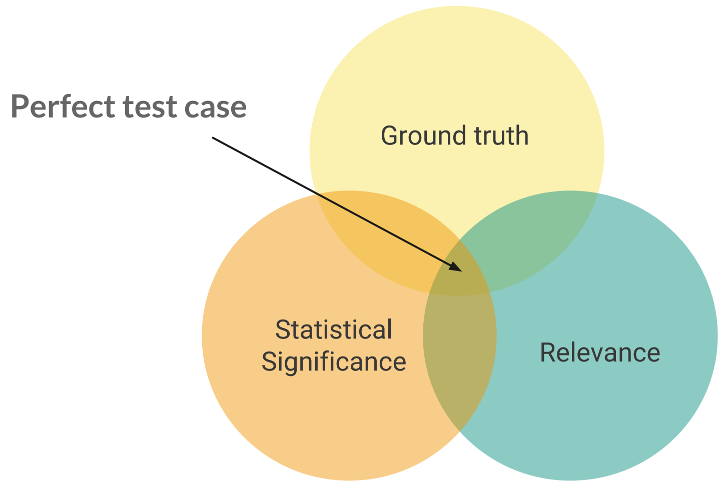
Test case's characteristics



Existing test cases



Benefits of bug injection in Production Software



3. Design of the solution

How to inject quality bugs in Production Software?

Requirements

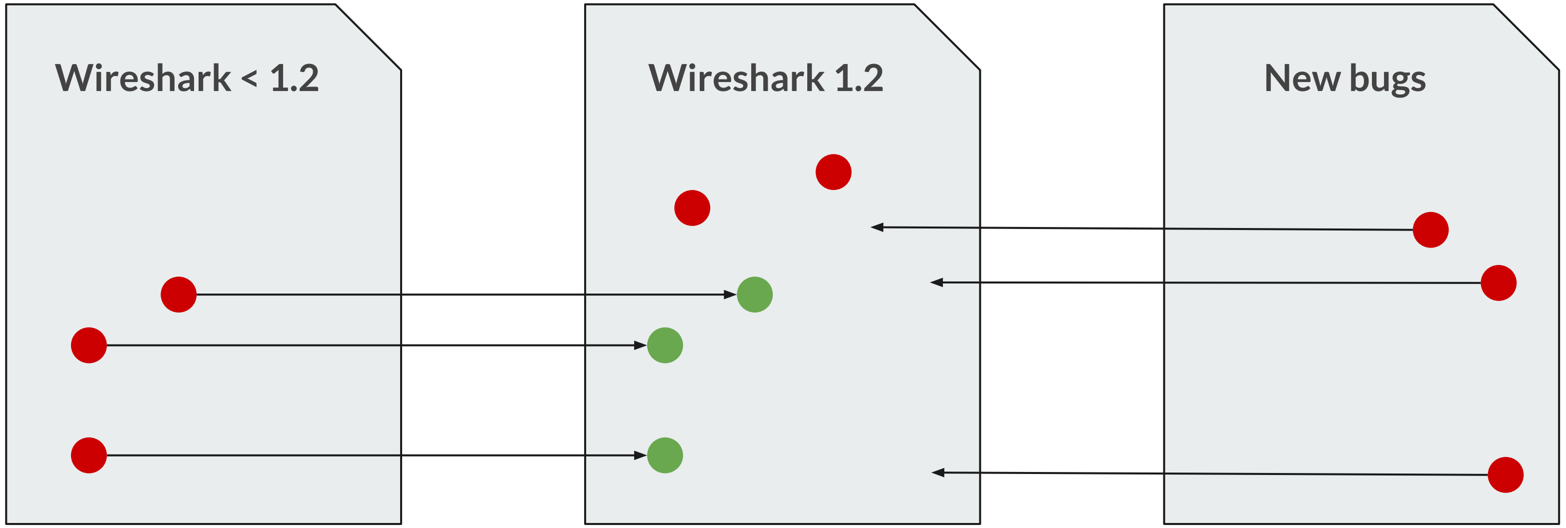
WIRESHARK A program

● Bugs

● Fixes

⚡ Triggering inputs

Different ways to inject bugs



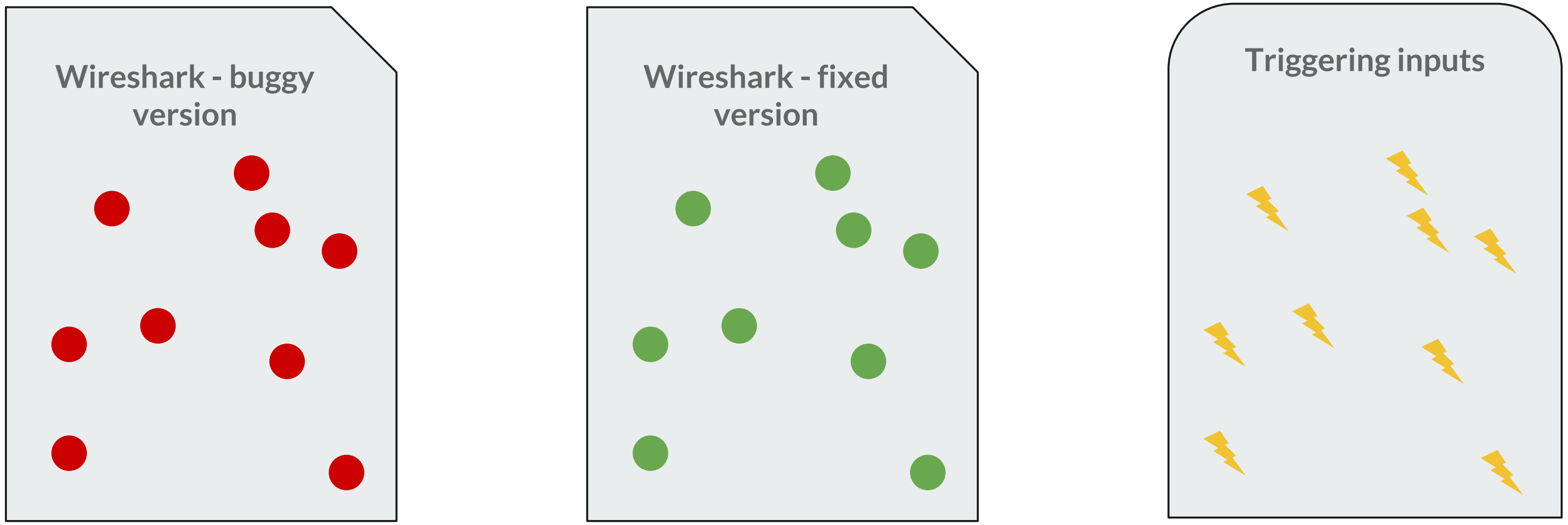
Different ways to inject bugs

	Pros	Cons
Existing bugs (reported)	● Real by definition ● Easy to add ● Come with triggering inputs & fixes	● Only a small amount existing
Injected bugs	● Choice of the category ● We can inject a lot of them	● Creating a bug, its fix and its triggering input is time-consuming

Suggested criteria for bug's quality

- Reflect a programmer's way of coding
- Bug complexity
- Span the execution lifetime of a program
- Come with an input that serves as an existence proof
- Manifest for a very small fraction of possible inputs

The prepared test case



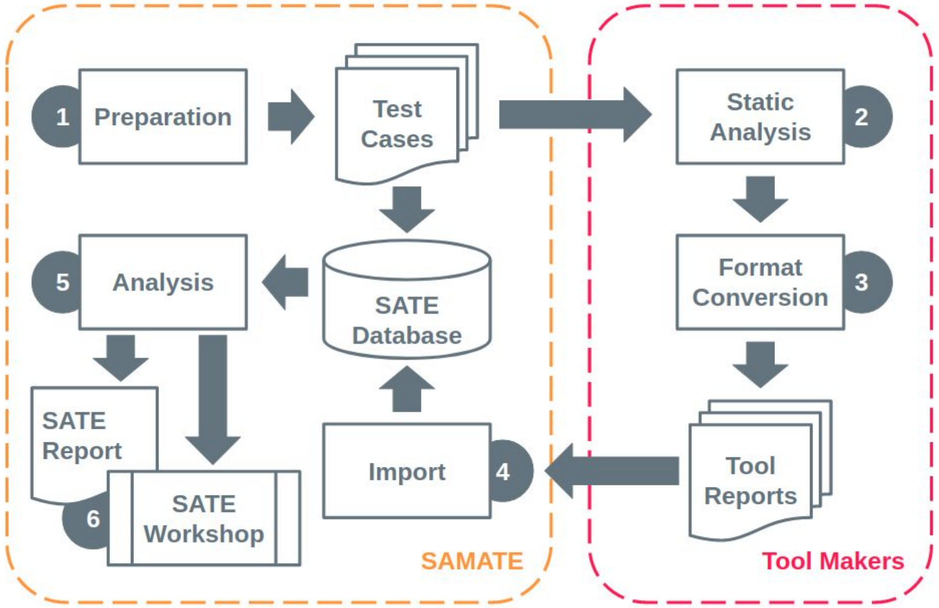
4. Results & future outlook

Results

New approach for assessing static analyzers in SATE VI.

~ 50 quality bugs injected

What now?


1 Preparation
Test Cases
Static Analysis 2
5 Analysis
SATE Database
Format Conversion 3
SATE Report
6 SATE Workshop
Import 4
Tool Reports
SAMATE
Tool Makers

Conclusion

- Awareness on software security
- Versatility
- Great experience at NIST

- SATE VI test cases ready
- 8 months of training so far

Any Questions?

Bug example

```
nresp = packet_get_int();

#if defined(BUG_7DD70701) // Compiling the version with the bug
if (nresp > 0 && nresp <  1048576) {
#else // Compiling the correct version
if (nresp > 0) {
#endif

response = malloc(nresp * sizeof(char*));

for (i = 0; i < nresp; i++)
        response[i] = packet_get_string(NULL);
}
```