PRESENTER
Frank Busse

Motivation

Static analysis is fast but imprecise and hence creates many false positives.

Dynamic symbolic execution is precise but often gets lost in the many program paths.

We **combine** two off-the-shelf static analysers with a symbolic execution engine to **confirm true positives** and provide developers with concrete test cases.

Approach

For each static analysis error report we use its trace to instrument the original program and **use the instrumented locations**

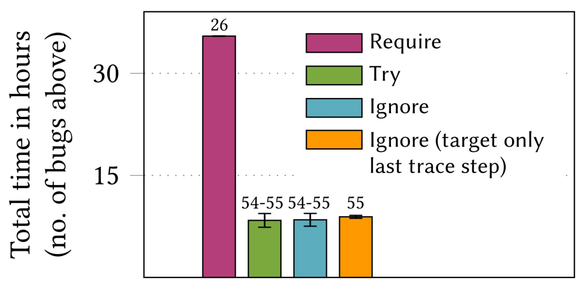
1. as target for symbolic execution
2. to constrain the path condition

We provide **three different strictness levels** to handle constraints that should hold according to the error trace:

Ignore noop
Try add constraint if feasible
Require constraint has to hold

Results

Guess what? Neither the intermediate step locations nor their constraints are beneficial in most cases (GNU Coreutils fault injection study based on Clang SA and Infer traces).



Static analysis **error traces do not add any benefits** when combined with targeted symbolic execution.

Combining Static Analysis Error Traces with Dynamic Symbolic Execution (Experience Paper)
Frank Busse, Pritam Gharat, Cristian Cadar, Alastair Donaldson
ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2022)

https://srg.doc.ic.ac.uk/projects/klee-sa/