



FishFuzz: Catch Deeper Bugs By Throwing Larger Nets

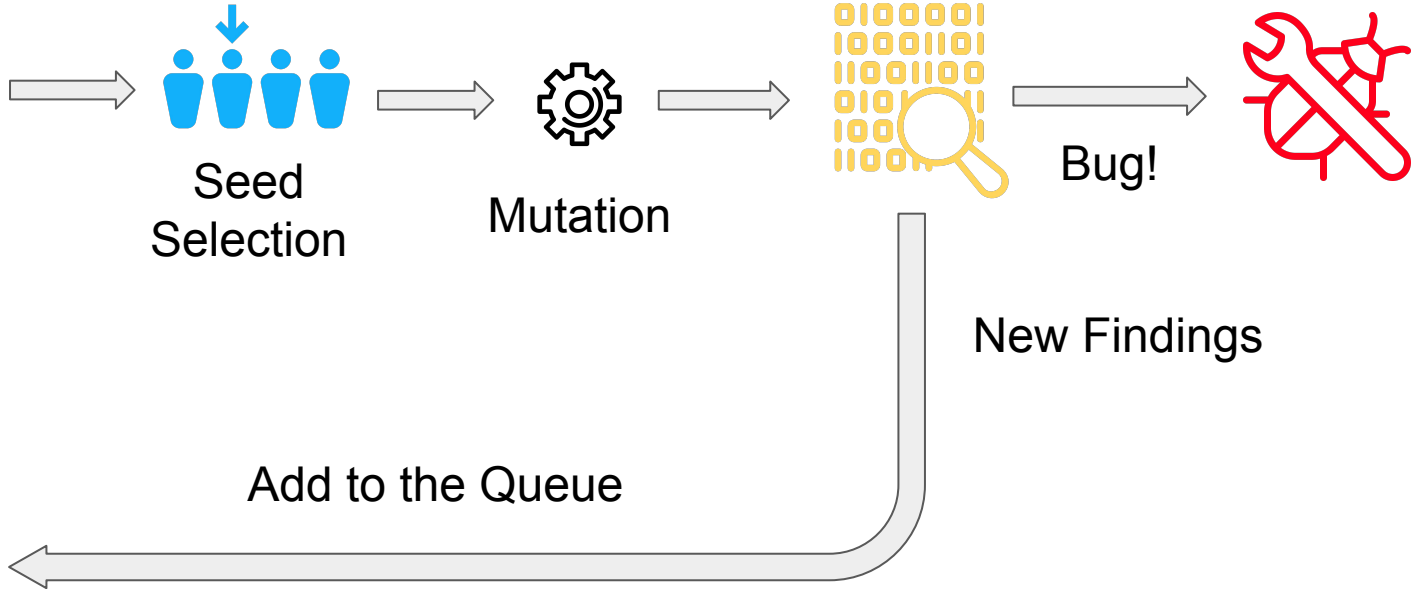
Han Zheng, Jiayuan Zhang, Yuhang Huang, Zezhong Ren, He Wang, Chunjie Cao,
Yuqing Zhang, Flavio Toffalini, Mathias Payer

EPFL

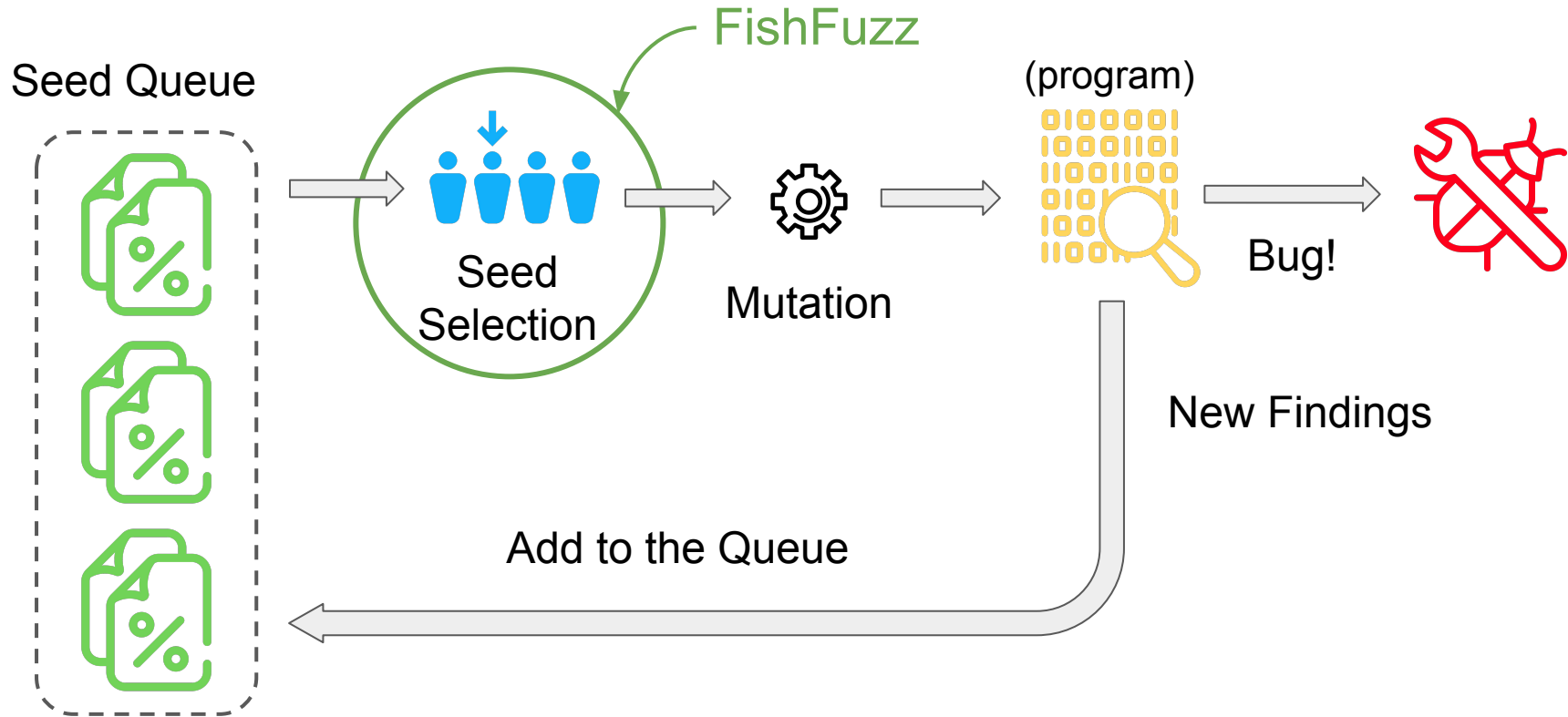


Introduction: Fuzzing

Seed Queue



Introduction: Fuzzing



Introduction: Classification

Coverage Guided Greybox Fuzzer

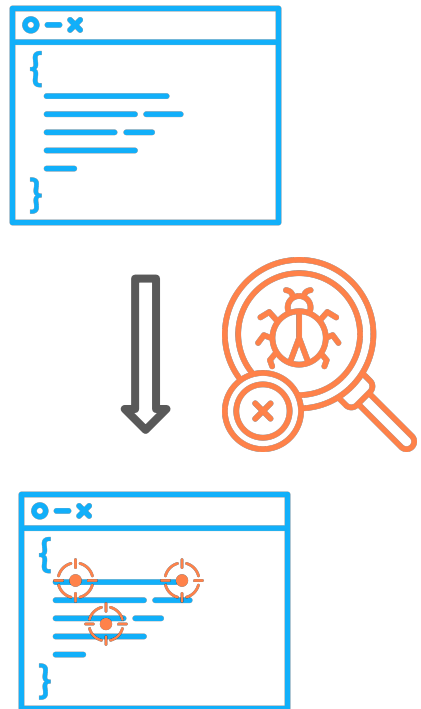
- Choose **arbitrary** locations as targets
- Every line of code is targeted

Directed Greybox Fuzzers

- Choose **few** locations as targets
- Manually selected targets

Sanitizer Guided Greybox Fuzzers (**SGFs**)

- Choose **many** locations as targets
- Sanitizer instrumented basic blocks are targeted



Sanitizer Guided Greybox Fuzzers

Sanitizer Guided Greybox Fuzzers (SGFs):

- Select a subset of sanitized lables as a target set
- Calculate the Seed-Target distance to minize the distance between seed and target set

Problem:

- The target set size is large (>10k)

Two Open Challenges:

- **Challenge 1:** how to correctly direct the testing
- **Challenge 2:** how to find interesting targets to test

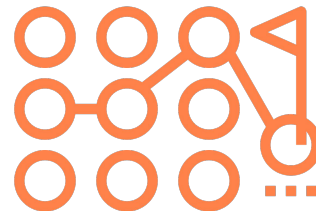


Challenge 1: Reaching Targets

Existing SGFs inherit distance metric from AFLGo

- Calculate seed-target distance
- Choose seed with shortest seed-targetset distance

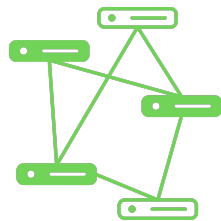
For large target sets, no seed is representative enough to cover all targets!



Challenge 2: Selecting Interesting Targets

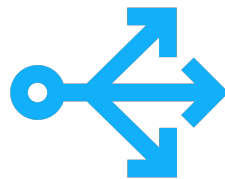
Existing SGFs notice that:

- Not every sanitizer target can be triggered
- Large target set can hinder the program testing



SGFs prune the target set before fuzzing campaign:

- Complexity-Based (ParmeSan)
- Profile-Based (ParmeSan)
- Reachability-Based (SAVIOR)

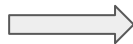


Static analysis may introduce false negatives!

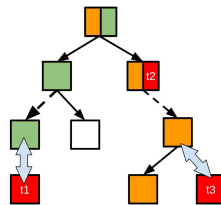
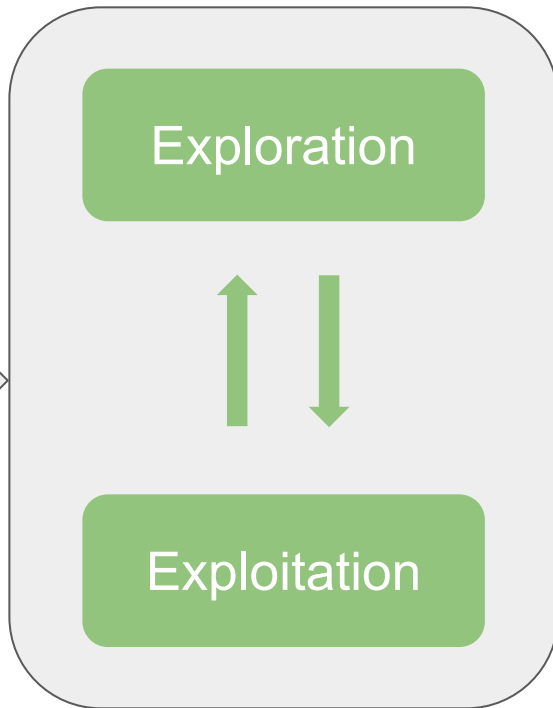


Queue Culling Algorithm

Seed Queue

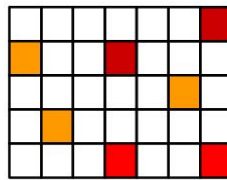


Seed Selection
(simplified)



A **novel distance metric** that calculates seed-target distance for each target

(contribution 1)



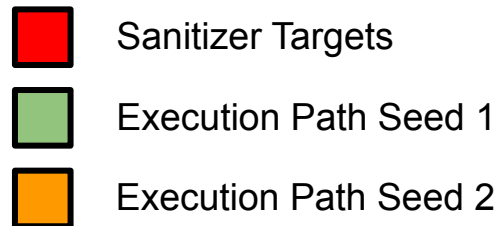
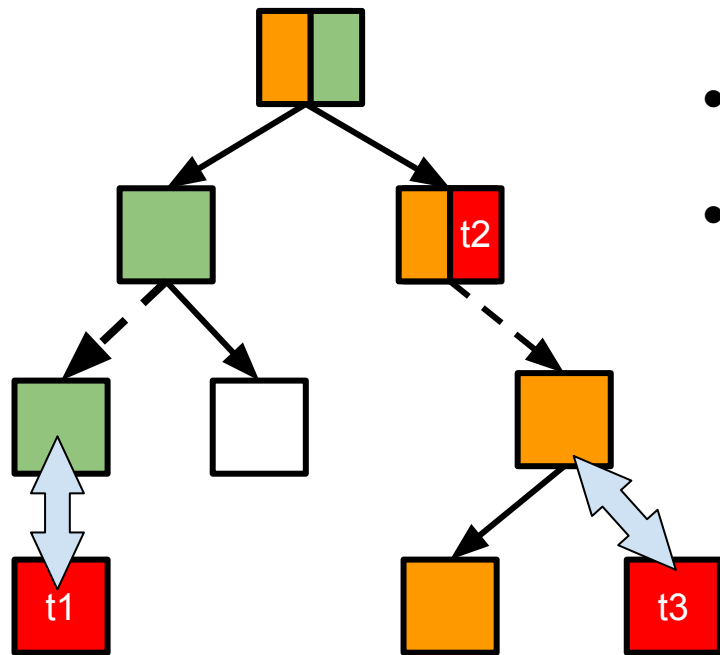
A **dynamic target ranking** algorithm that automatic discards exhaustively fuzzed targets

(contribution 2)

Contribution 1: Seed-Target Distance Metric (exploration)

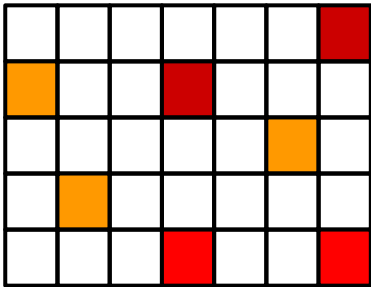
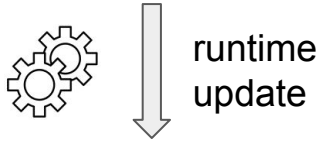
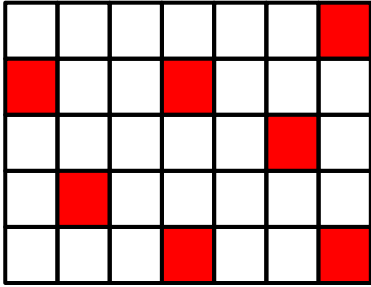
FishFuzz calculates distance by:

- Calculate $\text{Min}_{\text{path}_j \in \text{Queue}} D(\text{Path}_j, t_i)$
- For each target t_i , choose Path_j with shortest distance



(whole-program call-graph)

Contribution 2: Dynamic Target Ranking (exploitation)



(targets' priority)

Intuition:

The most visited targets are less likely to contain a bug
The less visited targets could still hide bug

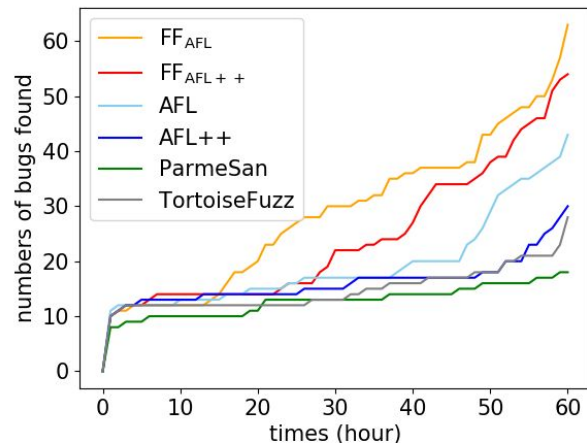
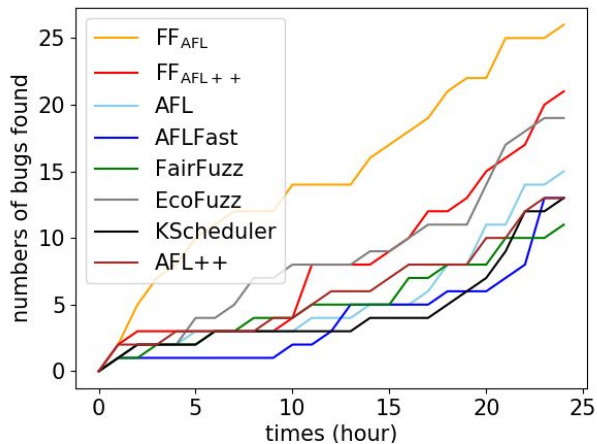
Two tactics:

Recording the frequency a target is visited
Selecting the seeds that hit the less explored targets

Unreachable and unexploitable targets get discarded

Evaluation

- Two prototypes based on AFL and AFL++
- FishFuzz finds up to 2.8x UNIQUE bugs compared to baseline
- FishFuzz finds 56 new bugs (38 CVEs assigned) in the latest version



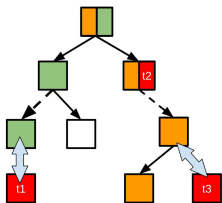
Conclusion

A queue culling algorithm that
orchestrates exploration and exploitation



56 unknown bugs discovered
and 38 CVEs assigned

A robust seed-target distance metric
that is independent of target size



Dynamically targets ranking to direct the
fuzzer towards promising locations

