# Parallel Fuzzing

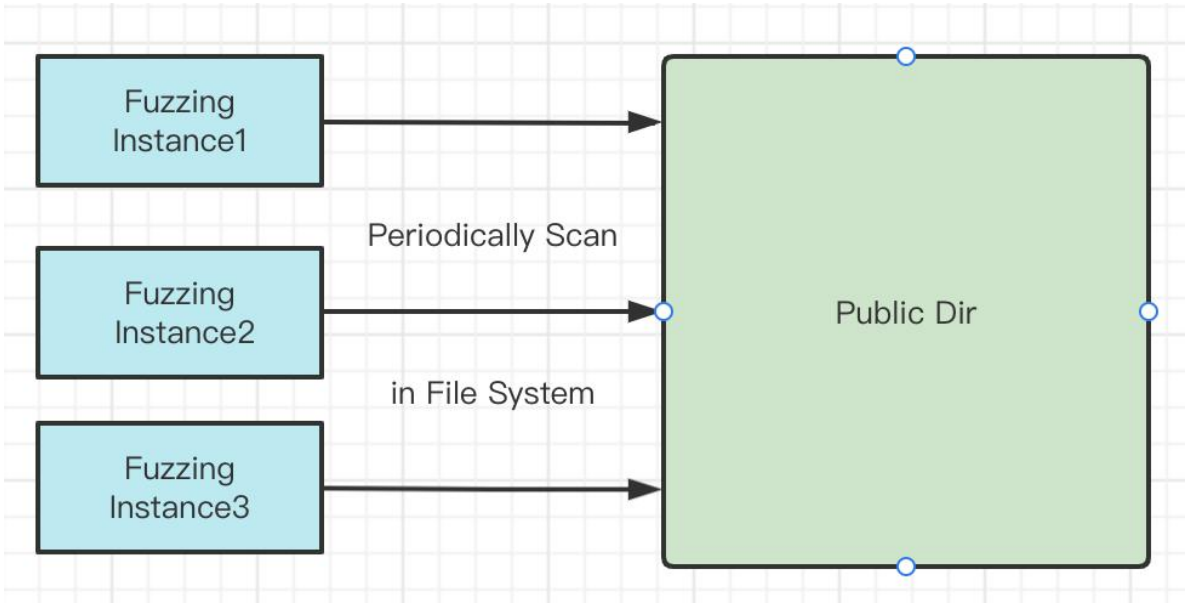
An Overview of Recent Works

Yifan Xia

# Introduction

# Introduction

- Large-scale fuzzing is gaining popularity, Espeically in industry
  - Google's OSSFuzz  powered by ClusterFuzz
  - Microsoft Springfield
- Illustration of AFL Parallel Mode

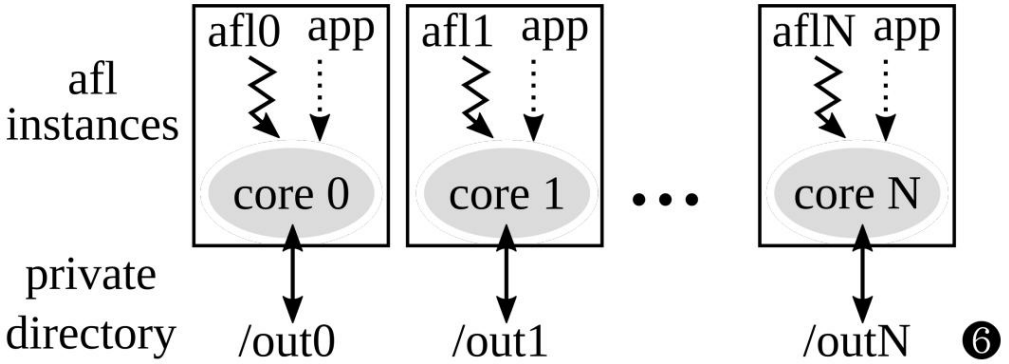



**When it comes to multiple-system, AFL uses SSH to synchronize.**

# AFL explained – parallel fuzzing

*Syncing phase*

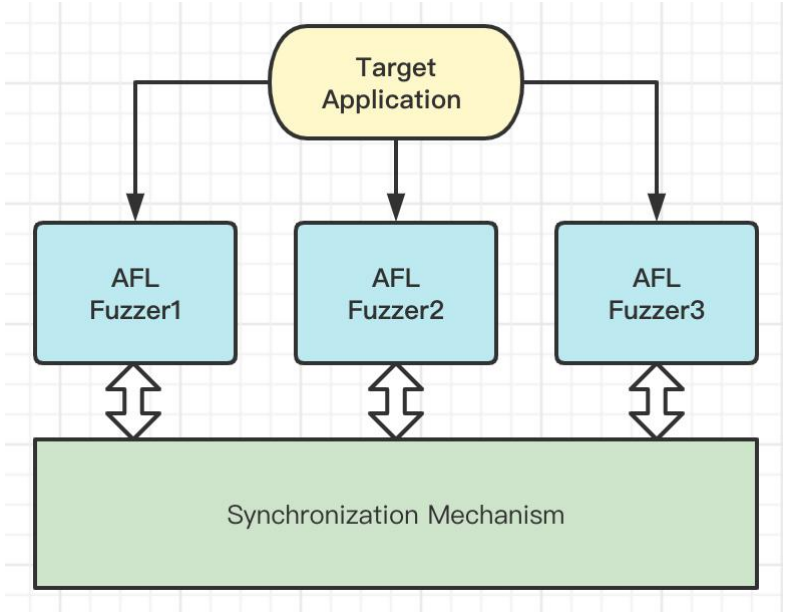
(1) Scanning the private directories of other fuzzer instances
(2) Executing unseen test cases
(3) Copying to own directory if interesting



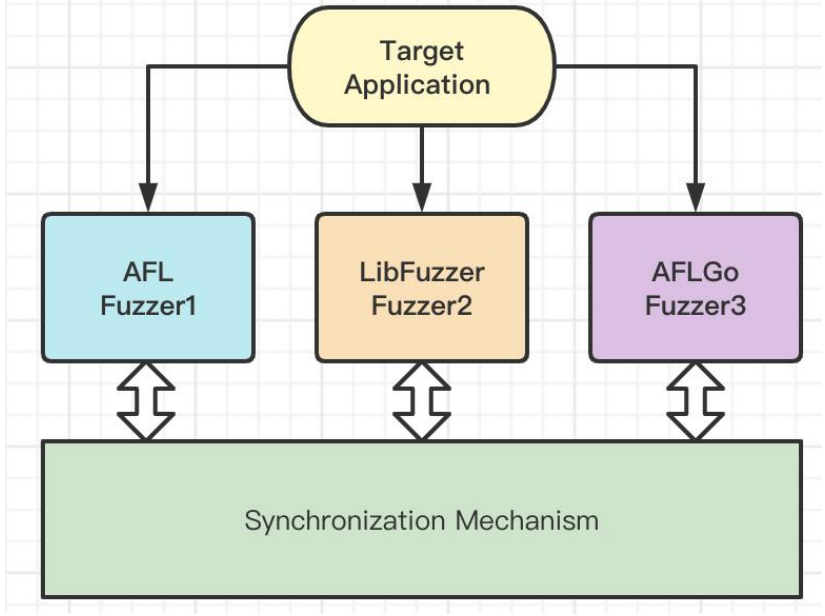
# Scope

# Scope

- Existing work can be categorized by two features
  - Single-Sysyem or Multiple-System ?
  - Single Fuzzer-Type or Multiple Fuzzer-Type ?




Single Fuzzer-Type




Multiple Fuzzer-Type

# Scope

- And two main challenges
  - Synchronizing guiding information (with lower overhead)

    For multiple-system, distributed architecture may be required.
  - Task Scheduling

    For multiple-system, different computing capability should be taken into consideration

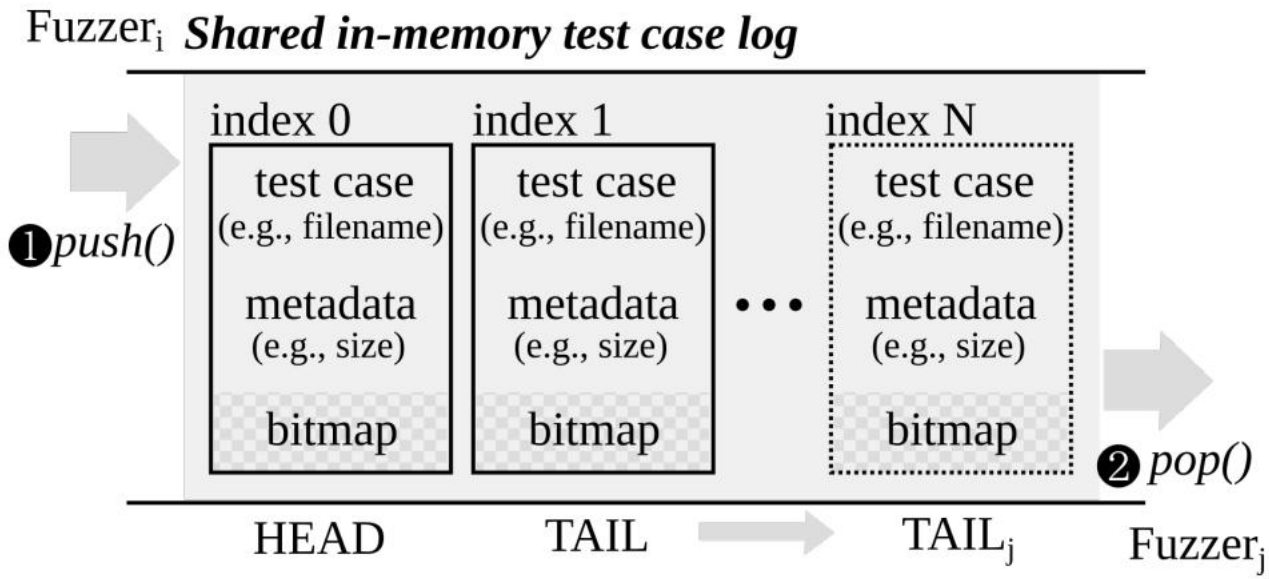
The origin AFL parallel mode is simple
( with high overhead synchronization)

# Designing New Operating Primitives to Improve Fuzzing Performance ( CCS' 17 )

## Shared in-memory test case log

- No directory enumeration:
  pop() to examine
  test cases from neighbors

- No test case re-execution:
  direct reference on the bitmap



Parallel mode is great
But not all fuzzers support that

# PAFL ( FSE'18 )

## Extend Fuzzing Optimizations of Single Mode to Industrial Parallel Mode

Yu Jiang KLISS, Tsinghua University, China

- Motivation

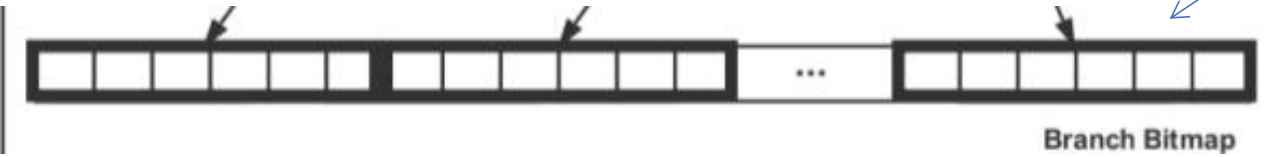
  The performance of some optimizations(i.e. FairFuzz, AFLFast) degrade in parallel mode due to the defect of design.

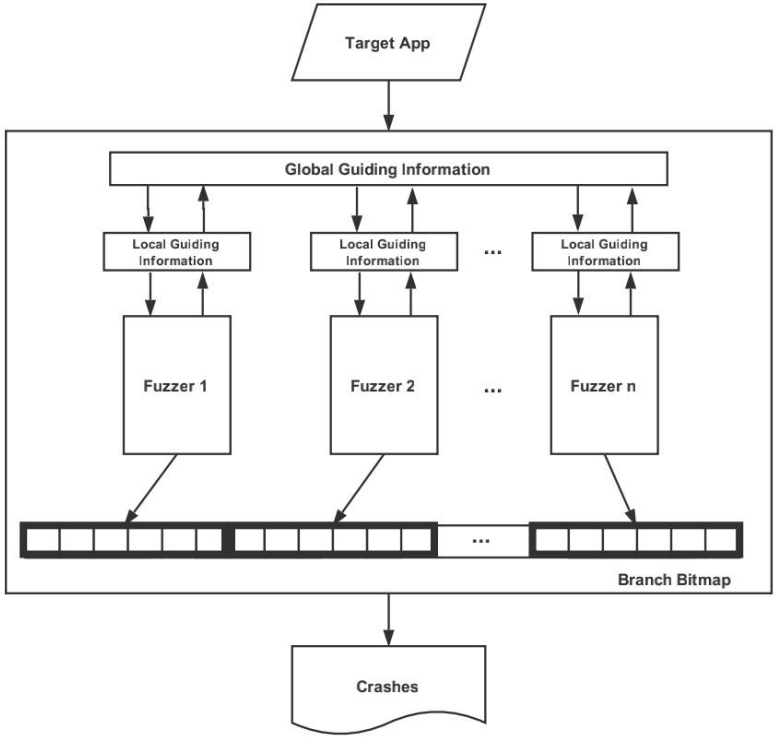
- Two Problems

  1. Guiding Information Loss
  2. Task Conflicts

- Two Contributions

  1. Global Guiding Information
  2. Task Division Mechanism



Since instances with same type collabrate, what about ensembling diverse fuzzers?

# EnFuzz (USENIX Security '19)

Ensemble Fuzzing with Seed Synchronization among Diverse Fuzzers

Yu Jiang KLISS, Tsinghua University, China

- Motivation

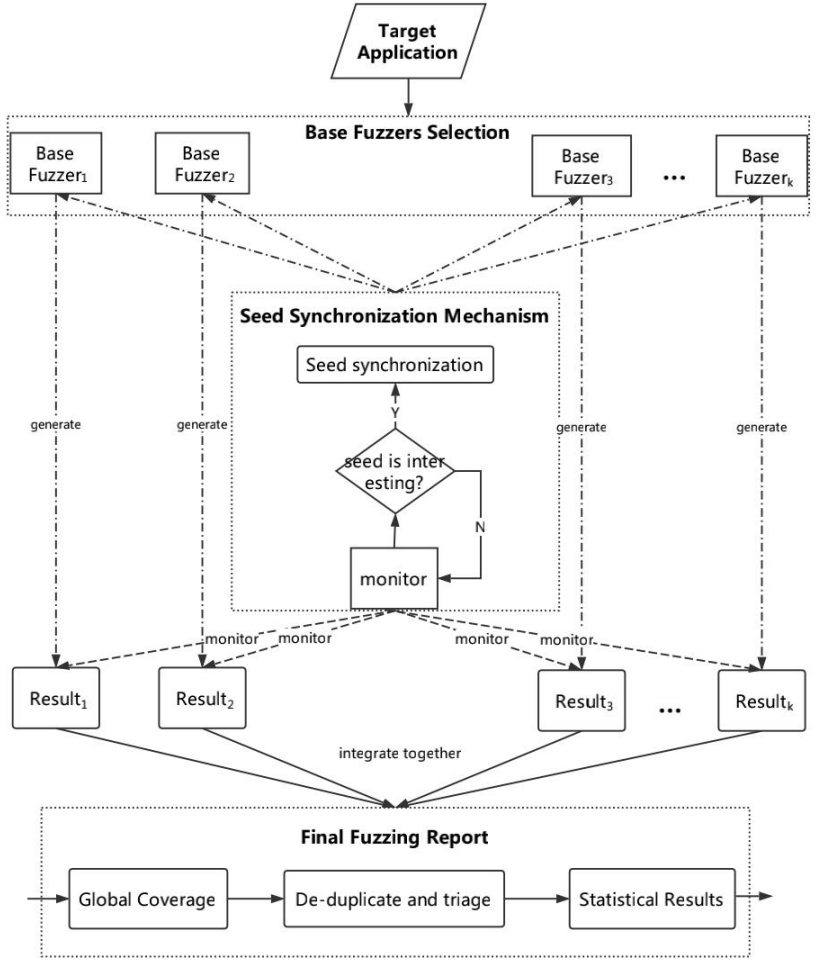
  Well-designed fuzzing strategies perform weakly in industrial practice. (due to the complexity)

- Insight

  Collaborate advantages of diverse fuzzers

- Contribution

  1. a heuristic method for diverse fuzzers choosing
  2. a GALS seed synchronization mechanism



Task allocation is important,
while exsiting method is inefficient

# AFLTeam (ASE' 21)

Towards Systematic and Dynamic Task Allocation for Collaborative Parallel Fuzzing

University of Melbourne

- Motivation

  existing task-dividing algorithms might be ineffective

  due to the lack of structural information of the input program. (i. bitmap is random. ii. static allocation )

- Insight

  1. Obtaining structural information from

     attributed call graph.

     graph => sub-graphs <=> tasks
  2. Filtering seeds which can't reach target functions.
  3. dynamic updating with execution information.

*A. Framework Overview*

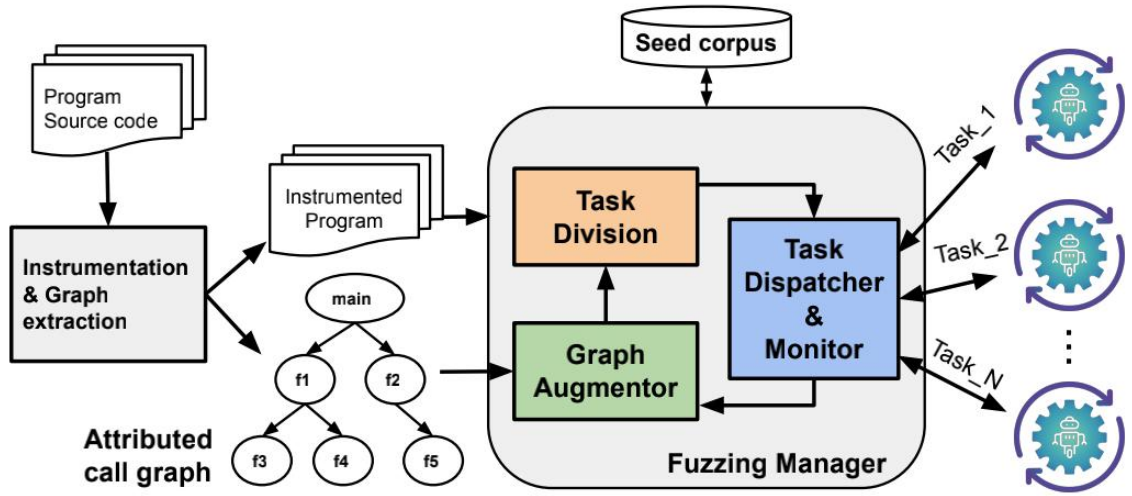



**Figure 3.** AFLTeam's workflow.

## How about Multiple-System?

No representative work yet,
but researchers are exploring.

# UniFuzz (NUDT)

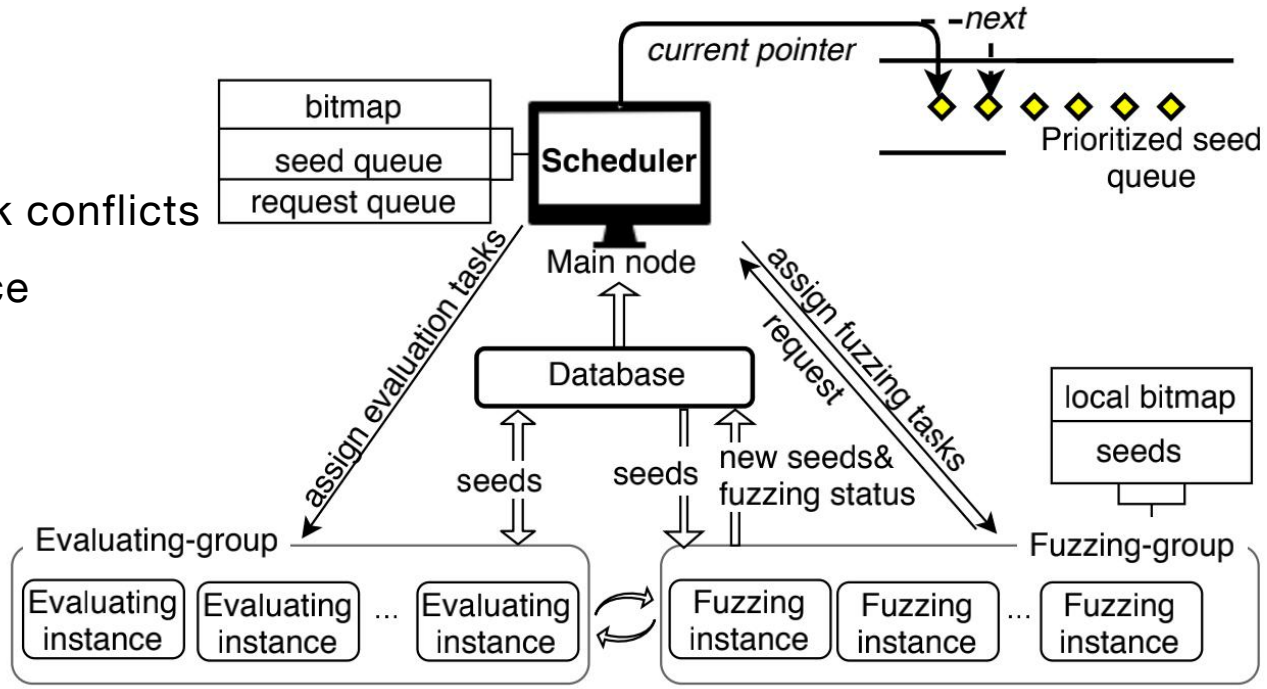
Optimizing Distributed Fuzzing via Dynamic Centralized Task Scheduling

- Motivation
  1. Challenges mentioned before (information synchronization, task conflicts)
  2. Difficulties in distributed environment (workload balance, synchronization overhead)

- Insight

  Distributed architecture to solve three challenges

1. a main scheduler which arranges fuzzing tasks for task conflicts
2. a request-response machanism for workload imbalance
3. a global information database for synchronization



# Future Direction

# Future Direction

- Single-System & Single Type
  1. Better Task-Allocation Algorithm
  2. Task-Aware Mutation
- Single-System & Multiple Type
  1. Heuristics for choosing diverse fuzzers

     (ACSAC 20) Cupid: Automatic Fuzzer Selection for Collaborative Fuzzing
  2. Improvement of the ensemble architecture
  3. Intelligent resource allocation
- Multiple-System

  Distributed Architecture

# THANKS