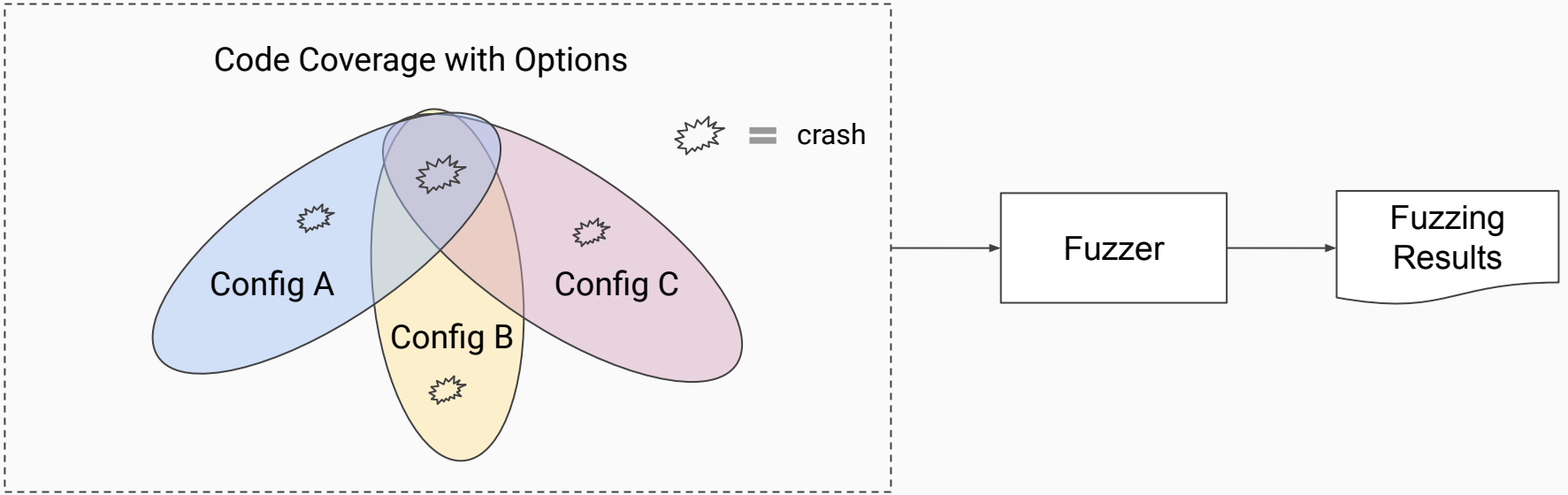
Fuzzing Configurations of Program Options

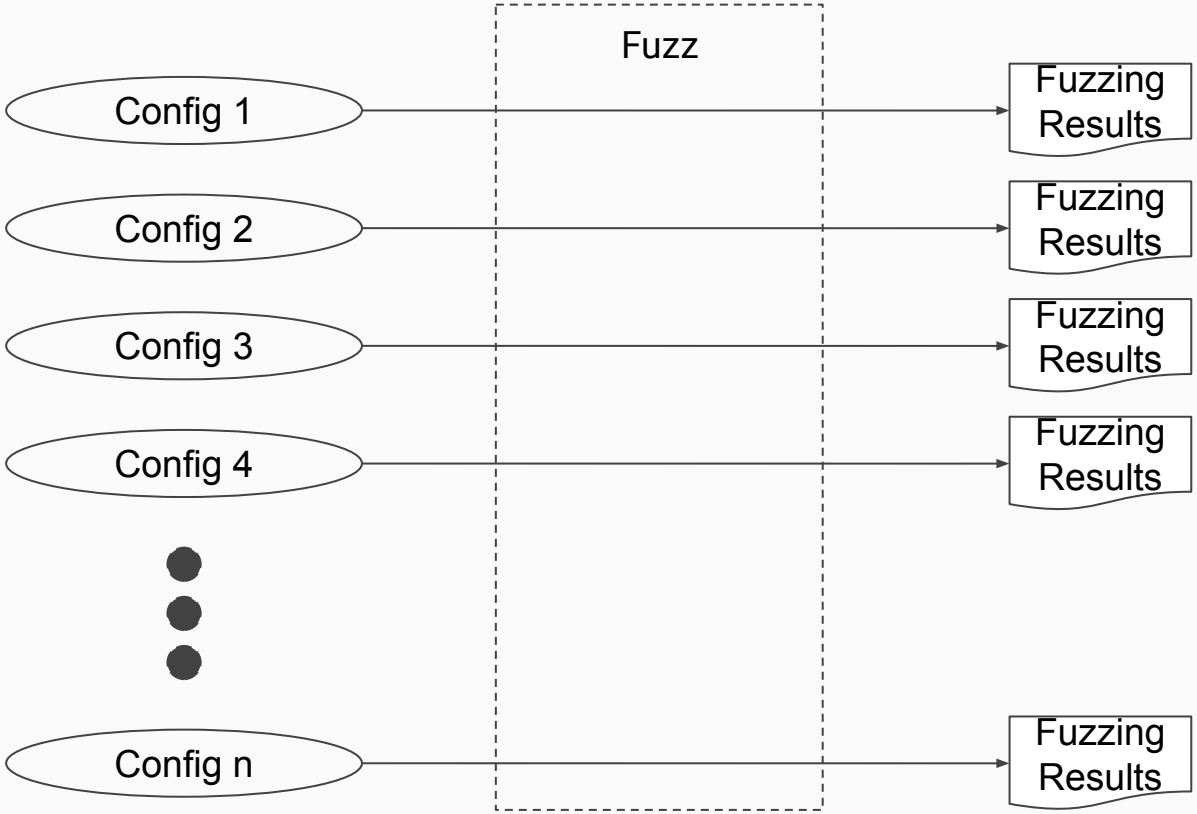
Zenong Zhang[*], George Klee[†], Eric Wang[‡], Michael Hicks[†] and Shiyi Wei[*]

[*]University of Texas at Dallas, [†]University of Maryland, [‡]Poolesville High School

Fuzzing Programs with Configurations



Existing Configuration Testing Approaches



Contributions of Our Work

- Empirical Study: How configurations affect fuzzing performance?
- ConfigFuzz: Expands input space to include configuration and allows fuzzers to mutate expanded input
- Evaluation: Compare ConfigFuzz to
    1. Fuzzing default configuration
    2. Fuzzing sample configurations with combinatorial testing

How Configurations Affect Fuzzing Performance?

Study setup:

- Target programs: FFmpeg, nm, gif2png
- Metric: line coverage
- Fuzzer setup: AFL, 3 trials, 24 hours
- Configurations: manually constructed

How Configurations Affect Fuzzing Performance?

Table 1: Total and unique line coverage for FFmpeg configurations.

	Default	-f flv	-f mpeg	-f h264	-f mp4	-f webm	**all configs**
# of all lines	17496	13511	12916	9344	1836	1782	26919
# of unique lines	4954	2073	1707	3130	149	109	-
% of unique lines	28%	15%	13%	33%	8%	6%	-

How Configurations Affect Fuzzing Performance?

Table 2: Total and unique line coverage for nm configurations.

	-l	–synthetic	-s	–defined-only	–with-symbol-versions	-f posix
# of all lines	4060	3913	3844	3827	3809	3788
# of unique lines	428	41	5	3	14	19
% of unique lines	11%	1%	0%	0%	0%	0%
	-a	-f bsd	-r	-g	-A	-n
# of all lines	3780	3762	3732	3719	3676	3655
# of unique lines	8	3	5	6	6	19
% of unique lines	0%	0%	0%	0%	0%	0%
	–special-syms	-u	-f sysv	–size-sort	-D	**all configs**
# of all lines	3591	2797	2728	2715	2486	4676
# of unique lines	0	4	38	96	7	-
% of unique lines	0%	0%	1%	4%	0%	-

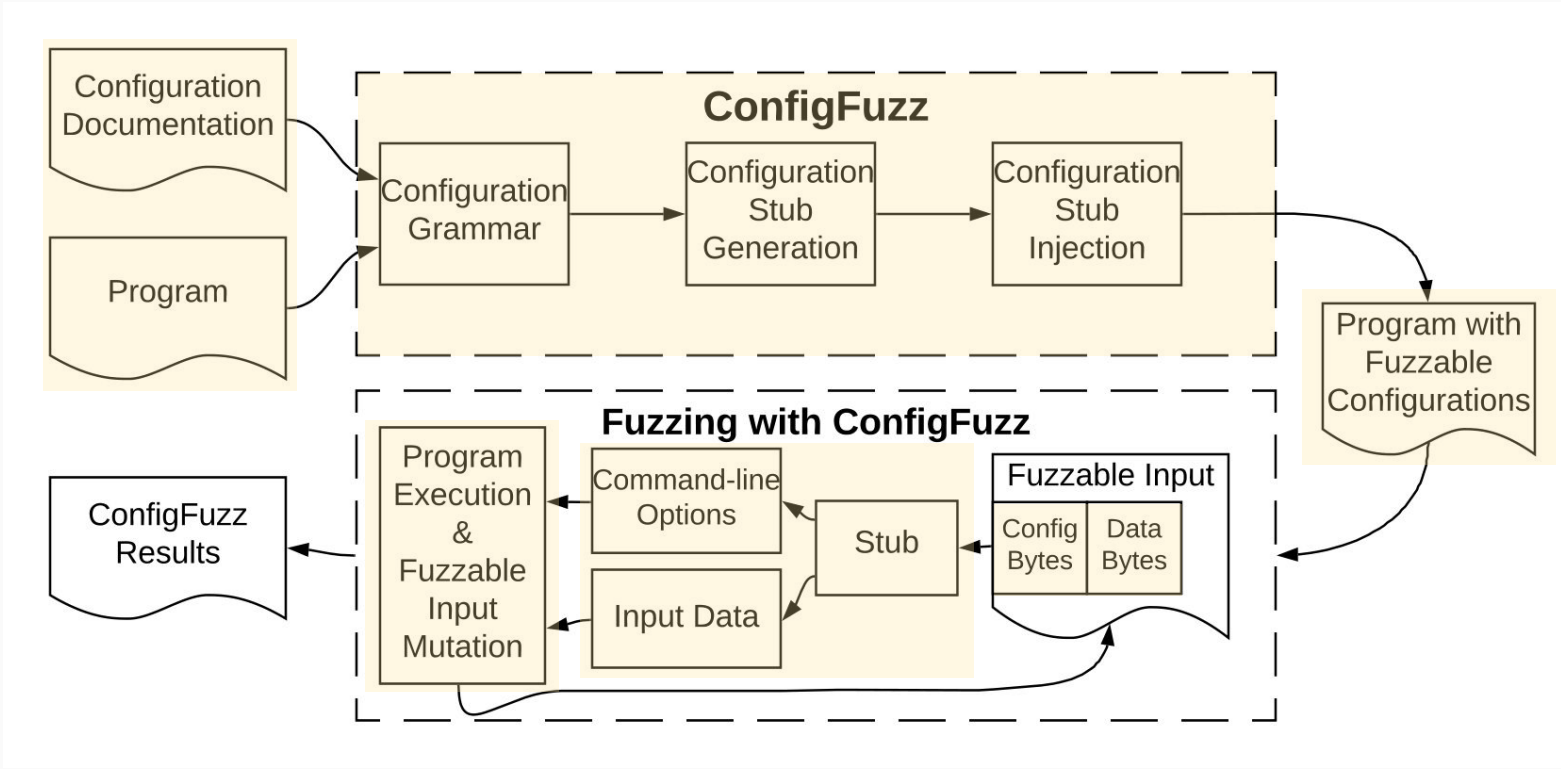
Configurations:

- Reach unique code locations
- Contribute disproportionately to fuzzing performance
- Also depend on programs

Key ideas:

- Expand input with configurations encoding
- Smartly fuzz configurations when fuzzers mutate expanded input

ConfigFuzz


Configuration Documentation
Program
ConfigFuzz
Configuration Grammar
Configuration Stub Generation
Configuration Stub Injection
Program with Fuzzable Configurations
Fuzzing with ConfigFuzz
Program Execution & Fuzzable Input Mutation
Command-line Options
Input Data
Stub
Fuzzable Input
Config Bytes
Data Bytes
ConfigFuzz Results

Configuration Grammar

Types: bool, choice, numeric, string

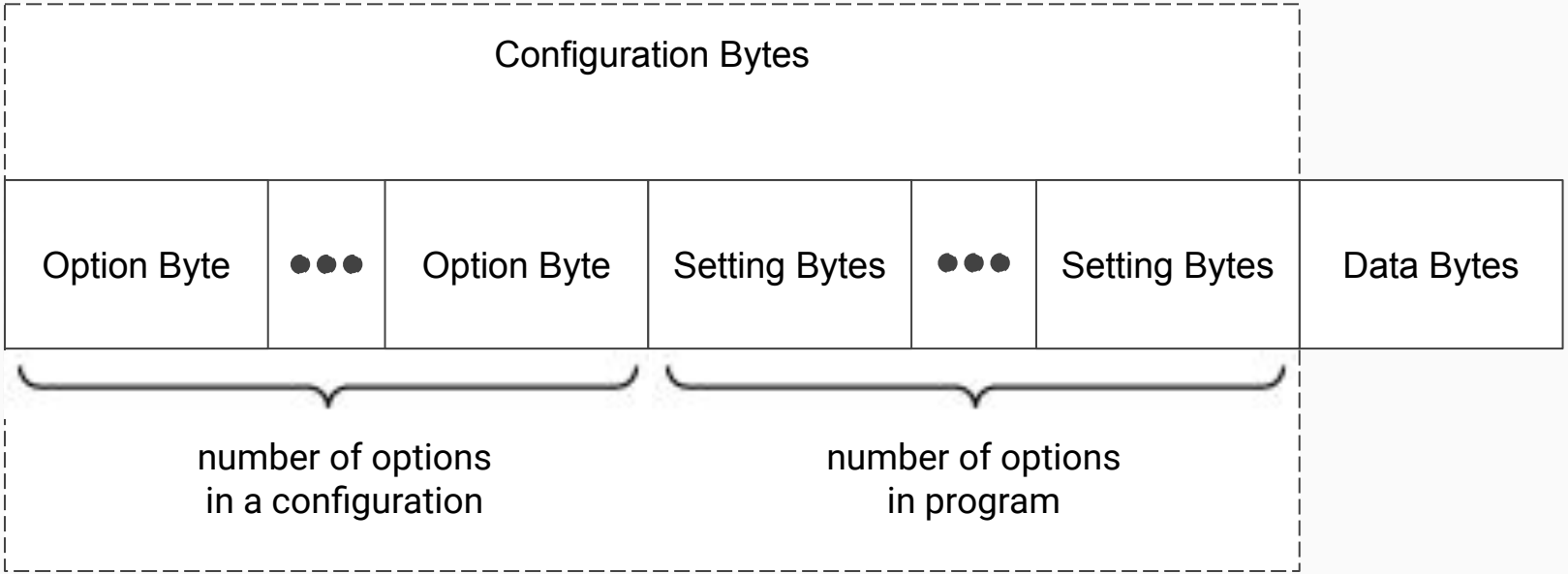
Input options: the way a program takes input file

Maxopts: number of options generated by ConfigFuzz in a configuration

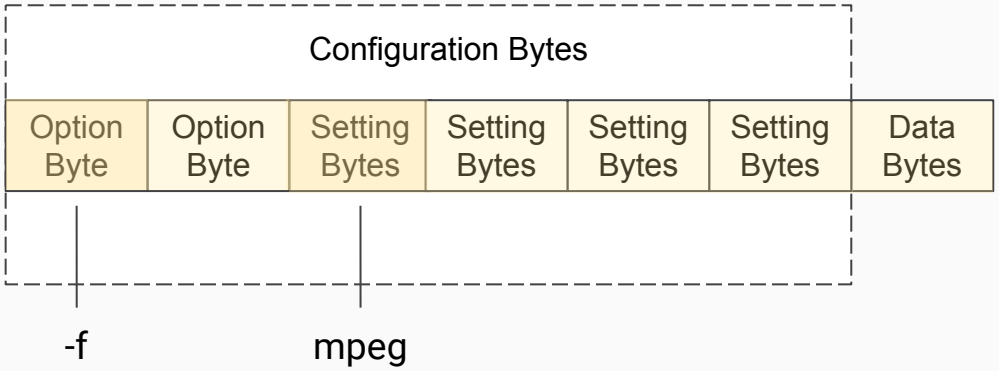
```
{
    "input options": ["-i"],
    "options": [{
            "id": 0,
            "opt": "-f",
            "type": "choice",
            "choices": ["mp4", "mpeg", "webm", "h264", "flv"]
        },
        {
            "id": 1,
            "opt": "-vframes",
            "type": "numeric",
            "range": [0, 432000]
        },
        {
            "id": 2,
            "opt": "-vn",
            "type": "bool"
        },
        {
            "id": 3,
            "opt": "-filter",
            "type": "string"
        }
    ],
    "dependence": [],
    "conflict": [
        ["-vn", "-vframes"]
    ],
    "strmax": 19,
    "maxopts": 2
}
```

An excerpt of configuration grammar of FFmpeg

Expanded Input Organization



Configuration Encoding



argv: ffmpeg -f mpeg -vn -i tmpfile

```
{
    "input options": ["-i"],
    "options": [{
            "id": 0,
            "opt": "-f",
            "type": "choice",
            "choices": ["mp4", "mpeg", "webm", "h264", "flv"]
        },
        {
            "id": 1,
            "opt": "-vframes",
            "type": "numeric",
            "range": [0, 432000]
        },
        {
            "id": 2,
            "opt": "-vn",
            "type": "bool"
        },
        {
            "id": 3,
            "opt": "-filter",
            "type": "string"
        }
    ],
    "dependence": [],
    "conflict": [
        ["-vn", "-vframes"]
    ],
    "strmax": 19,
    "maxopts": 2
}
```

An excerpt of configuration grammar of FFmpeg

How ConfigFuzz performs?

Two settings of ConfigFuzz:

- ConfigFuzz-k: the fuzzed configurations contain at most k explicitly set options

Setups:

- AFL, AFL++, 5 trials, 24 hours
- Target program: xmllint, objdump and cxxfilt
- No string option

Two baselines:

- Baseline-def: the default configuration
- Baseline-cov: sample of configurations generated by two-way covering arrays; each configuration is fuzzed for [24h / # of samples]

2-Way Covering Array

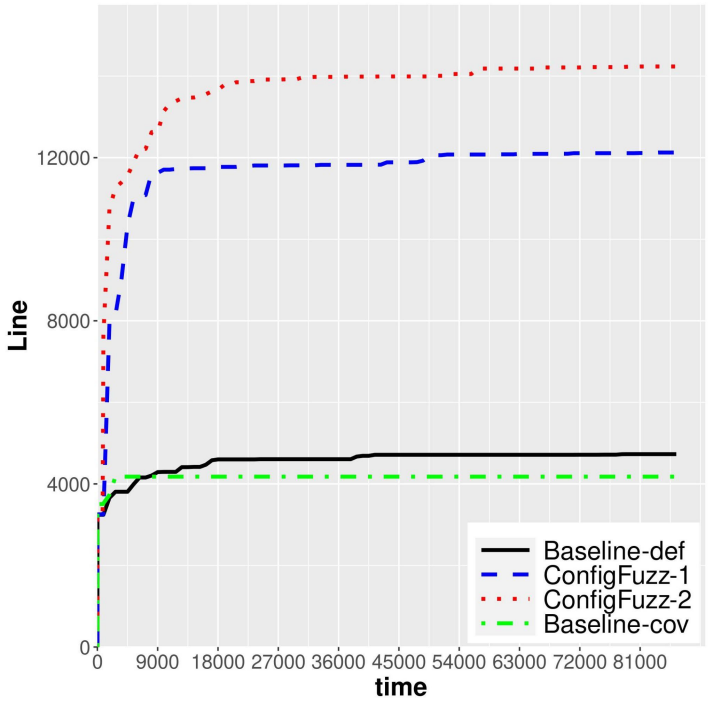
-_	-n	-p	-t
ON	ON	ON	ON
ON	OFF	OFF	OFF
OFF	ON	OFF	OFF
OFF	OFF	ON	OFF
OFF	OFF	OFF	ON

2-way Covering Array on cxxfilt Sample Options

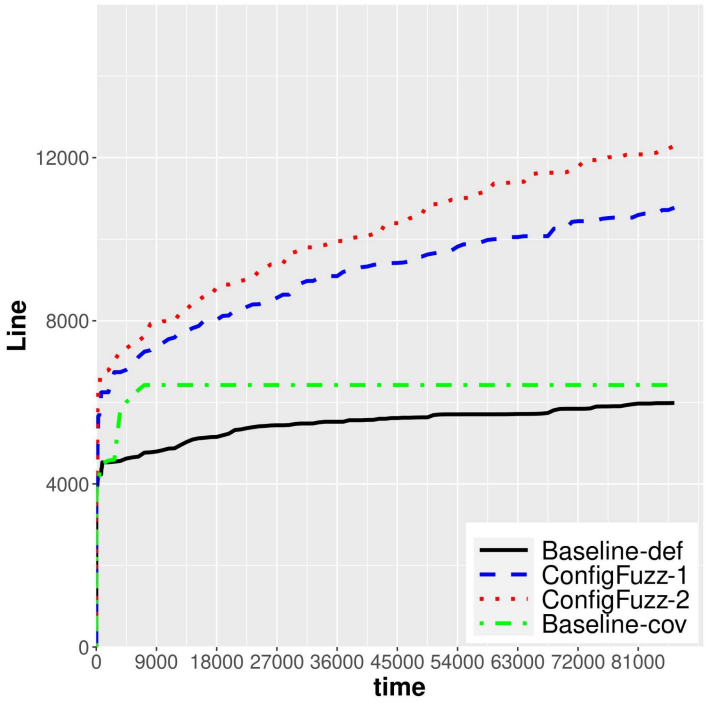
How ConfigFuzz performs?

- RQ1: Does ConfigFuzz outperform baselines?
- RQ2: How ConfigFuzz-1 and ConfigFuzz-2 compare?

Preliminary Results



AFL - xmllint

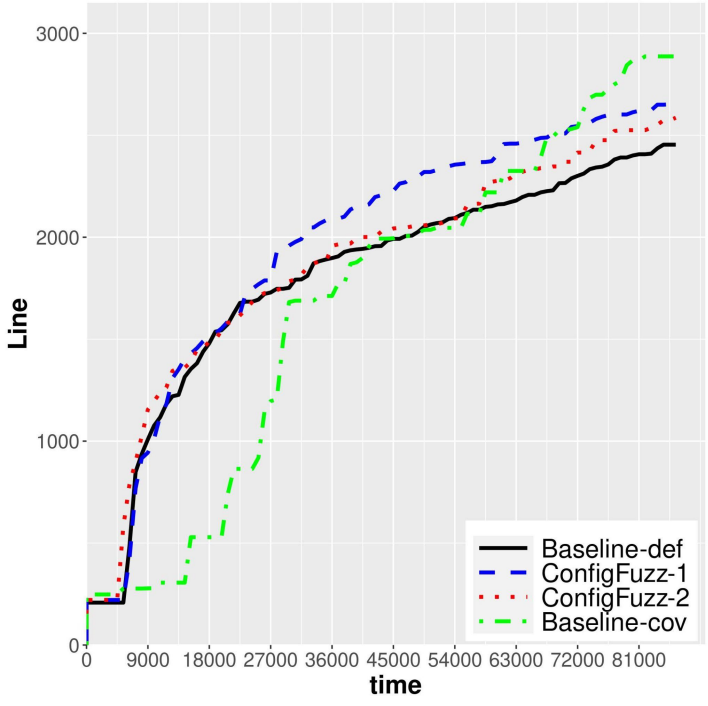


AFL++ - xmllint

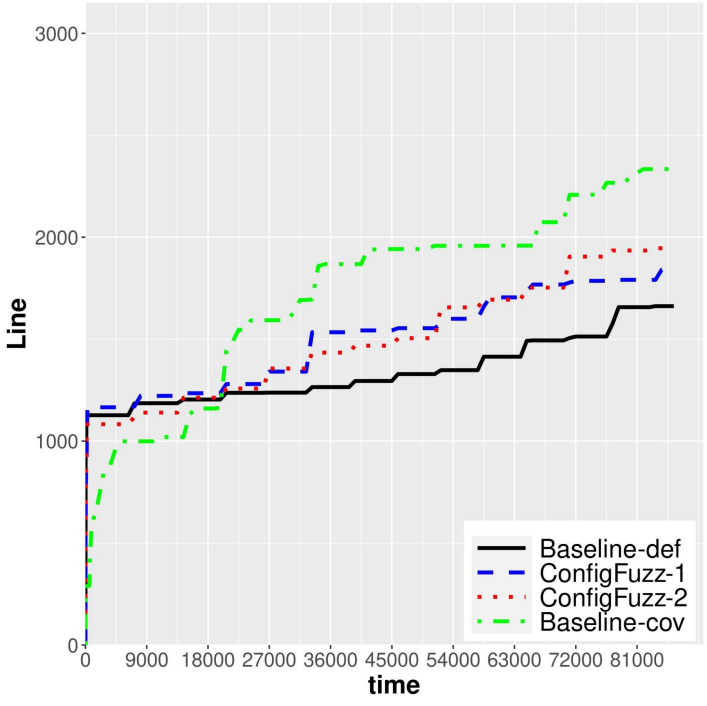
xmllint:

- ConfigFuzz outperformed two baselines
- ConfigFuzz- 2 outperformed ConfigFuzz -1

Preliminary Results



AFL - cxxfilt



AFL++ - cxxfilt

cxxfilt:

- Covering arrays performed the best
- ConfigFuzz-1 and ConfigFuzz-2 results did not show statistical significance

Planned Additional Study and Evaluation

- Deeper investigation of the experimental results
  - Which configurations are fuzzed over time
  - How configurations contribute to fuzzing performance
- Evaluate ConfigFuzz with more programs and fuzzers
  - The same set of programs in empirical study and preliminary evaluation
  - Search for more programs with well-documented options
- Experiment with other variants of ConfigFuzz
  - More option interactions
  - String options

Conclusions

- Configurations (1) cover unique codes during fuzzing and (2) contribute to code coverage disproportionately
- ConfigFuzz encodes configurations into expanded input and enables fuzzers to smartly fuzz configurations
- Deeper investigation is proposed to gain insights from evaluation results