Poster: A Tight Integration of Symbolic Execution and Fuzzing

Sébastien Bardin, Yaëlle Vinçont, Michaël Marcozzi

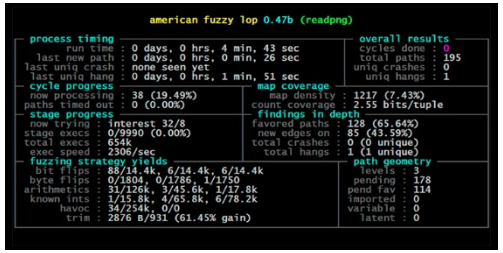
September 15-16, 2022
London, UK

About Fuzzing

- Automatic test input generation
- Black-box fuzzing = random generation (usually specification-based)

2000+ bugs in GCC/LLVM
1000+ bugs in Z3/CVC4
400+ bugs in SQL DBMS

- Grey-box fuzzing = semi-random generation (code coverage feedback)



Many CVEs and bugs in many apps:
iOS, Firefox, dpkg, OpenSSH, etc.

Fuzzing vs Symbolic Execution

- Symbolic execution = white-box fuzzing (path constraints solving)
- Grey-box and white-box fuzzing seem complementary
  - Grey-box fuzzing: no *path explosion,* nor *complex path constraints*
  - Symbolic execution: easily penetrates *paths guarded by an infrequent condition*
- Several tools aim at hybrid fuzzing (best of both worlds)

DRILLER Qsym Pangolin Angora

MATRYOSHKA Eclipser

Confuzz: Goals and Principles

	Analysis of path constraints				Fuzzing			
	Symbolic	Cheap	Targeted	Correct	Efficient	Constraints	Well-integrated components	
Fuzzing	-	-	-	-	✓	×	-	
SE	✓	×	×	✓	-	-	-	
Driller	✓	×	✓	✓	✓	×	×	
Qsym	✓	✓	✓	×	✓	×	×	
Pangolin	✓	✓	✓	×	✓	✓	✓	
Angora	×	✓	✓	×	~	✓	ok	
Matryoshka	×	✓	✓	×	~	✓	ok	
Eclipser	×	✓	×	×	✓	×	×	
ConFuzz	✓	✓	✓	✓	✓	✓	✓	

- Confuzz = yet another hybrid fuzzing tool
- Goal: be better than the others ☺
- Based on two main principles:
  - Rely on grey-box fuzzing to explore path space
  - For paths beyond infrequent conditions:
    - Create *easy-enumerable* version of prefix constraint
    - Use the grey-box fuzzer to *find correct solutions*

Preliminary Evaluation and Future Work

- Good **preliminary results** vs AFL++ and KLEE, on 3 LAVA-M programs

		AFL	AFL++	KLEE	ConFuzz
base64	**Avg**	0	0.2	10.0	38.8
3kloc	**Dev** (σ)	0	0.4	1.3	0.4
44 bugs	$\hat{A}_{12}$	1.0	1.0	1.0	-
md5sum	**Avg**	0	0	0	9
3kloc	**Dev** (σ)	0	0	0	1.7
57 bugs	$\hat{A}_{12}$	1.0	1.0	1.0	-
uniq	**Avg**	0	0.4	5	26.9
3kloc	**Dev** (σ)	0	0.5	0	3.6
28 bugs	$\hat{A}_{12}$	1.0	1.0	1.0	-

- TODO: **extend** constraint language, **evaluate** at scale vs s.o.t.a. tools