Università di Catania

Fuzzing: the art of discovering vulnerabilities in an automated way

Marcello Maugeri

Hardening 7

Agenda

1. Introduction to Fuzz Testing
2. Input-based Taxonomy
3. Knowledge-based Taxonomy
4. Demo
5. Reflections

Agenda

1. Introduction to Fuzz Testing
2. Input-based Taxonomy
3. Knowledge-based Taxonomy
4. Demo
5. Reflections

Functional Testing VS Non-Functional Testing

User view

- Unit Testing
- Integration Testing
- System Testing
- ...

Attacker view

- Performance Testing
- Compliance Testing
- Security Testing
- ...

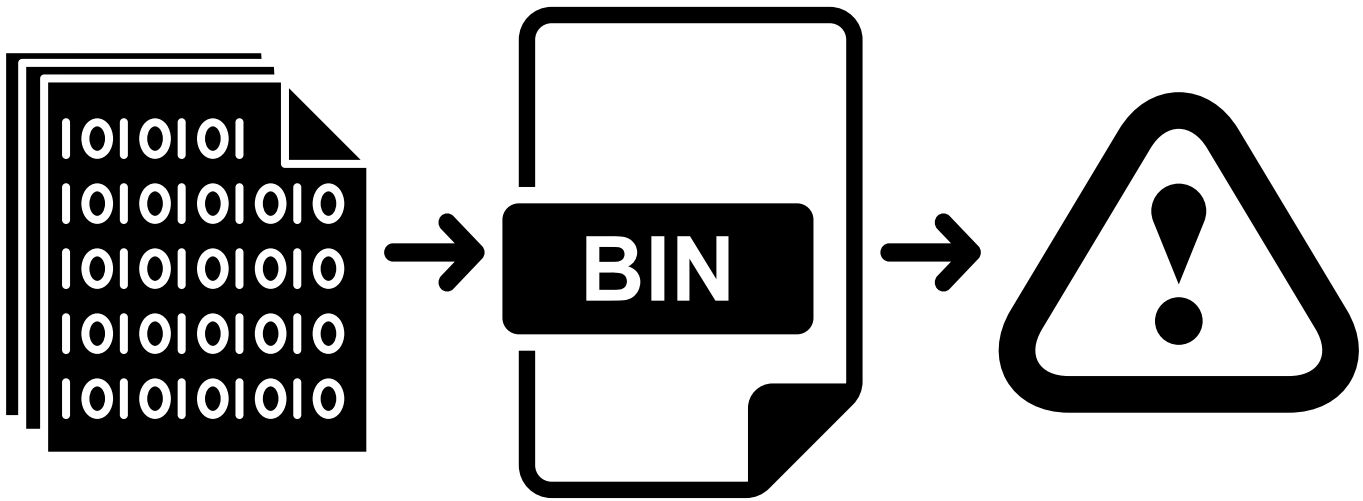
Security Testing

- Static Analysis
- Manual Code Inspection
- Dynamic Analysis

Patrice Godefroid. 2020. Fuzzing: hack, art, and science.

Fuzz testing

Fuzz Testing, or fuzzing, is a software testing technique that involves providing invalid, unexpected, or random test inputs to the software system under test. The system is then monitored for crashes and other undesirable behaviour.[1]



[1] Okun, V. and Fong, E. (2015), Fuzz Testing for Software Assurance

History of Fuzzing

The name was given by Miller et al.[2] to denote the *fuzz* utility.

[2] Barton P. Miller, Lars Fredriksen, and Bryan So. 1990. An empirical study of the reliability of UNIX utilities.

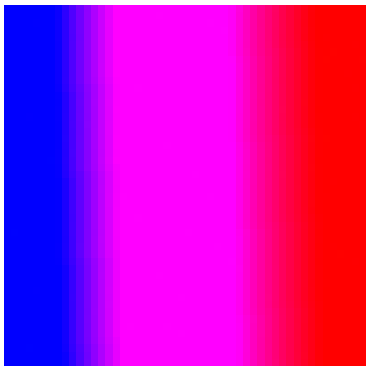
Some taxonomies

- Input-based
  1. Generation-based
  2. Mutation-based
  3. Structure-aware
- Knowledge-based
  1. Black-box fuzzing
  2. White-box fuzzing
  3. Gray-box fuzzing

Agenda

1. Introduction to Fuzz Testing
2. Input-based Taxonomy
3. Knowledge-based Taxonomy
4. Demo
5. Reflections

Generation-based



	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	
0x000	89	50	4E	47	0D	0A	1A	0A	00	00	00	0D	49	48	44	52	00	00	00	32	00	00	00	32	08	06	00	00	00	1E	3F	88	B1	00	.PNG . ... IHDR...2...2..... ?.±.
0x022	00	00	09	70	48	59	73	00	00	2E	23	00	00	2E	23	01	78	A5	3F	76	00	00	03	17	49	44	41	54	68	DE	CD	5A	CB	8E	.. pHYs...#...#.x¥?v....IDAThÞÍZÉ.
0x044	14	31	0C	AC	F2	4C	73	E1	07	38	80	10	E2	FF	3F	6B	11	68	05	88	13	CC	EC	CE	78	0F	A4	D9	6C	4F	1E	8E	E3	06	.1 ¬òLsá.8..âÿ?k.h...ÌìÎx.¤ÙlO .ã.
0x066	22	45	E9	C3	74	62	BB	FC	A8	B8	87	80	2A	2A	E3	35	80	0F	00	3E	A6	F5	7D	9A	EF	00	BC	05	F0	06	B3	E3	27	C0	"EéÃtb»ü¨¸..**ã5...>¦õ}.ï.¼.ð.³ã'À
0x088	3B	40	EF	00	7C	02	F0	19	C0	17	80	F7	80	DE	03	F8	0A	E0	1B	80	EF	00	7E	00	F8	55	DD	49	7A	47	B1	F2	6C	7E	;@ï.|.ð.À..÷.Þ.ø à..ï.~.øUÝIzG±òl~
0x0AA	A9	37	4A	76	2C	9A	56	9B	DB	48	4F	70	F5	C8	AA	A3	4A	6A	E3	45	ED	2A	51	54	44	AD	86	F4	7A	93	B6	14	62	65	©7Jv,.V.ÛHOpõÈªÉJjãEí*QTD-.ôz.¶.be
0x0CC	77	76	61	16	EB	D9	8A	E0	A1	3D	98	B4	E2	17	1C	57	44	CD	F6	08	D4	8A	5B	14	08	E0	98	89	AA	45	14	05	FF	C3	wva.ëÙ.à¡=.´â. WDÍö.Ô.[..à..ªE..ÿÃ
0x0EE	60	26	B4	1E	00	2C	49	F8	43	9A	7C	69	4E	DE	1A	E1	DF	2A	C2	E5	B7	D0	BA	6C	84	6F	F8	00	CB	A0	B8	14	09	73	`&´ .,IøC.|iNÞ.áß*Âå·Ðºl.oø.Ë ,. s
0x110	33	95	64	4B	DA	53	49	25	B1	C9	74	9C	EE	03	D5	F0	EF	64	74	CB	6B	9A	97	B4	EE	A3	80	35	4F	AA	4D	91	55	E0	3.dKÚSI%±Ét.î.Õðïdtëk..´î£.5OªM.Uà
0x132	0B	80	73	36	1F	00	3C	6E	15	61	84	12	9E	3C	69	C8	5A	8F	00	4E	49	F8	2E	02	1A	85	88	BA	1C	5D	AC	B2	B1	B2	.s6 .<n.a...<iÈZ..NIø.....º]¬²±²
0x154	C6	D6	12	2D	EF	4C	07	45	A9	79	8B	16	56	EE	19	E4	2F	0E	67	17	76	E9	B9	39	1B	A0	6B	18	12	85	D3	8B	B7	0B	ÆÖ.-ïL.E©y..Vî.ä/.g.vé¹9. k...Ó.·
0x176	DA	62	44	3B	6C	87	DC	A1	96	34	89	A2	D5	10	B4	93	C6	D5	48	FB	70	AE	96	09	65	43	55	CA	EF	8B	D5	60	2C	DA	ÚbD;l.Ü¡.4.¢Õ.´.ÆÕHûp®. eCUÊï.Õ`,Ú
0x198	21	12	95	B5	C2	E7	5C	6B	25	8B	F2	EC	0E	F4	D2	78	DD	C4	5B	68	71	96	24	74	CE	B5	2A	94	45	DB	81	D9	CD	5A	!..µÂç\k%.òì.ôÒxÝÄ[hq.$tÎµ*.EÛ.ÙÍZ
0x1BA	9A	1E	D6	0C	A8	E8	EE	39	98	09	98	89	41	77	82	17	93	4B	69	F9	BE	C6	D0	60	87	21	E8	D5	A1	88	DE	32	9E	15	. Ö ¨èî9. ..Aw...Kiù¾ÆÐ`.!èÕ¡.Þ2..
0x1DC	A1	6B	46	5B	E2	83	BD	A2	18	DB	28	89	E5	88	4B	A2	2A	A7	C4	B3	D6	A9	61	31	C3	8A	E0	76	57	ED	92	C6	53	26	¡kF[â.¾¢.Û(.â.K¢*§Ä³Ö©a1Ã.àvWí.ÆS&
0x1FE	B4	CE	DC	DF	75	10	11	1D	B3	8A	CC	80	1F	57	D9	D9	48	21	34	59	45	66	8E	A6	A7	C4	B3	45	82	2C	50	D2	17	23	´ÎÜßu.. ³.Ì. WÙÙH!4YEf.¦§Ä³E.,PÒ.#
0x220	2D	56	A4	1E	58	D4	79	1A	DB	1B	C8	AC	43	C4	D3	94	0A	31	9C	09	F6	21	23	32	22	4E	50	B8	25	32	A3	2E	32	A7	-V¤ XÔy.Û.È¬CÄÓ. 1. ö!#2"NP¸%2£.2§
0x242	08	43	5D	A6	76	42	2E	F0	31	9B	87	7E	8B	E8	8F	22	B4	1F	E7	4A	BF	CD	FD	97	E7	FE	16	96	8A	D0	7D	52	24	96	.C]¦vB.ð1..~.è."´ çJ¿Íý.çþ...Ð}R$.
0x264	36	92	16	6C	E7	8A	DB	9A	08	CA	69	8B	88	05	70	6C	2E	55	D3	F5	83	F1	09	C1	5E	10	37	EC	F7	9A	2A	7E	C8	3D	6..lç.Û..Êi...pl.UÓõ.ñ Á^.7ì÷.*~È=
0x286	6A	B8	21	77	7B	C3	EB	76	51	AE	85	BE	D6	D9	C3	B5	AA	72	8D	44	9D	16	1E	0D	7D	AD	9C	6B	AD	1D	C6	D0	76	E9	j,!w{ÃëvQ®.%ÖÙÃµªr.D.. }-.k- ÆÐvé
0x2A8	B6	11	D0	45	88	BE	BE	96	DB	C5	39	E3	66	BE	B1	4F	13	3B	BC	CB	CD	4E	E5	EF	7C	0C	DD	AD	90	87	54	FE	8E	22	¶.ÐE.%%.ÛÅ9ãf¾±O.;¼ËÍNåï| Ý-..Tþ."
0x2CA	3A	E4	99	3B	68	CC	1D	5D	4B	67	0E	1C	25	6A	9A	23	20	D9	2A	F3	5C	AB	C8	78	C3	D3	D7	2A	F4	31	A3	2A	87	8C	:ä.;hÌ]Kg. %j.# Ù*ó\«ÈxÃÓ×*ô1£*..
0x2EC	6F	49	5C	B0	73	8F	38	C6	02	E0	55	C6	B7	C4	45	84	5C	17	2B	93	9B	71	C4	BB	D9	F8	4B	4C	3F	63	B9	10	31	6B	oI\°s.8Æ.àUÆ·ÄE.\.+..qÄ»ÙøKL?c¹.1k
0x30E	A8	8E	E0	25	DC	9D	81	61	44	34	AB	F2	0F	61	C1	5E	BA	3B	97	3E	FC	58	1B	74	2C	D3	94	4B	C6	AF	CE	59	5F	2B	¨.à%Ü..aD4«ò.aÁ^º;.>üX.t,Ó.KÆ¯ÎY_+
0x330	EE	63	28	1B	CF	B6	4B	B5	F4	AC	97	0B	7D	F9	2B	05	AF	A0	24	83	5D	6B	9F	AC	55	B0	22	6B	B5	A5	3E	9E	00	8B	îc(.Ï¶KµôÂ¬. }ù+.¯ $.]k.¬U°"kµ¥>...
0x352	31	02	6D	7C	E6	8B	2C	00	00	00	00	49	45	4E	44	AE	42	60	82																1.m|æ.,....IEND®B`.

Mutation-based

http://www.google.com/search?q=fuzzing

Dopo 1 mutazione: ’http://www.google.com/search?q=fuz:ing’
Dopo 5 mutazioni: ’http:/L/www.googlej.com/seaRchq=fuz:ing’
Dopo 10 mutazioni: ’http:/L/www.ggoWglej.com/seaRchqfu:in’
Dopo 15 mutazioni: ’http:/L/wwggoWglej.com/seaR3hqf,u:in’
Dopo 20 mutazioni: ’htt://wwggoVgle”j.som/seaR3hqf,u:in’

Structure-Aware

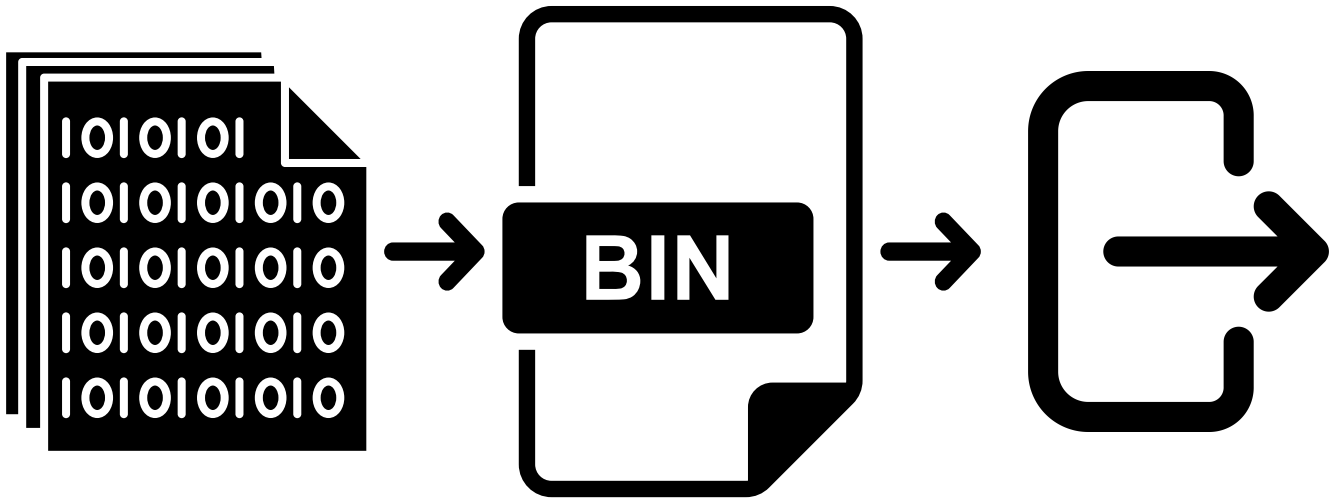
```
message SQLQueries {
    repeated CreateTable queries = 1;
}

message CreateTable {
    optional TempModifier temp_table = 1;
    required Table table = 2;
    required ColumnDef col_def = 3;
    repeated ColumnDef extra_col_defs = 4;
    repeated TableConstraint table_constraints = 5;
    required bool without_rowid = 6;
}
```

Agenda

1. Introduction to Fuzz Testing
2. Input-based Taxonomy
3. Knowledge-based Taxonomy
4. Demo
5. Reflections

Black-box Fuzzing



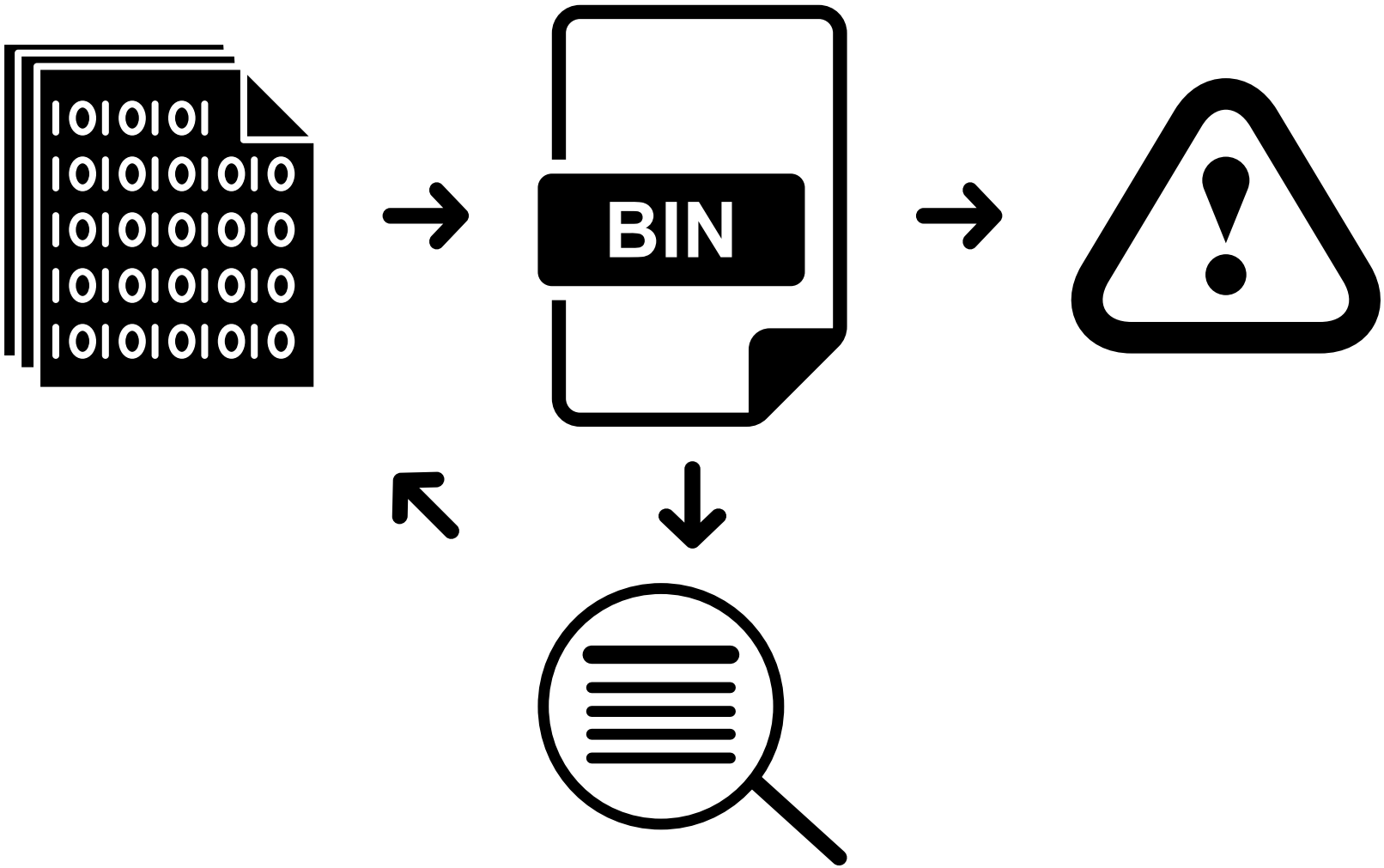
White-box Fuzzing (Symbolic Execution)

Steps: [3]

1. Run the code with some initial well-formed input
2. Collect constraints on input with symbolic execution
3. Generate new constraints (by negating constraints one by one)
4. Solve constraints with a constraint solver
5. Synthesize new inputs

[3] Patrice Godefroid. 2008. Automated Whitebox Fuzz Testing

Gray-box Fuzzing (Coverage-Guided Fuzzing)



AFL++

Steps:

1. Instrument the target
2. Collect input seed
3. Start Fuzzing!

Agenda

1. Introduction to Fuzz Testing
2. Input-based Taxonomy
3. Knowledge-based Taxonomy
4. Demo
5. Reflections

Demo time

- First example: a simple crashing program
- Second example: CVE-2021-3156 on sudo
- Both examples available on `https://github.com/marcellomaugeri/Hardening-7-Fuzzing`

Agenda

1. Introduction to Fuzz Testing
2. Input-based Taxonomy
3. Knowledge-based Taxonomy
4. Demo
5. Reflections

What else?


THE #1 FUZZ TESTER EXCUSE FOR LEGITIMATELY SLACKING OFF:
"MY CODE'S FUZZING!"
HEY! GET BACK TO WORK!
FUZZING!
OH. CARRY ON.

Automation Challenges

1. How can we fuzz efficiently more types of software systems?
2. How can the fuzzer identify more types of vulnerabilities?
3. How can we improve the usability of fuzzing tools?

Böhme, Marcel, Cristian Cadar, and Abhik Roychoudhury. 2021. ”Fuzzing: Challenges and Reflections.”

Current Research

- Fork-Awareness
- Blackbox fuzzing of API

Visit *marcellomaugeri.github.io* for all my contacts

Summary

- Fuzzing is an effective technique to discover bugs
- It is automated, but not yet automatic
- Several open challenges to be faced

References

M. Boehme, C. Cadar, and A. Roychoudhury.
Fuzzing: Challenges and reflections.
IEEE Softw., 38(3):79–86, 2021.

P. Godefroid, M. Y. Levin, D. A. Molnar, et al.
Automated whitebox fuzz testing.
In *NDSS*, volume 8, pages 151–166, 2008.

B. P. Miller, L. Fredriksen, and B. So.
An empirical study of the reliability of unix utilities.
Communications of the ACM, 33(12):32–44, 1990.

V. Okun and E. N. Fong.
Fuzz testing for software assurance.
2015.

Thank you for the attention

Q&A