FuzzOps: keep your computers busy and your pager quiet

wallet name: Mindy Preston
e-mail: mindy@identity-function.com
mastodon: @yomimono@witches.town
github: @yomimono
prounouns: she/her

non-controversial things

bugs are bad

non-controversial things

writing bugs is easy

non-controversial things

finding bugs earlier is better

non-controversial things

tests can expose bugs

non-controversial things

bugs are bad

writing bugs is easy

finding bugs earlier is better

tests can expose bugs

a spooky story

test **all** the things

a spooky story

input → computation → output

a spooky story

known input → computation → known output?

a spooky story

```
import unittest

class TestStringMethods(unittest.TestCase):

    def test_upper(self):
        self.assertEqual('foo'.upper(), 'FOO')
```

a spooky story

human failure

a spooky story

robot gatekeepers

a spooky story

unit testing

- expect (tcl/shell)
- cUnit, jUnit, jsUnit, csUnit, oUnit...
- test::Unit (ruby)
- unittest (python)

CI/CD

- Jenkins
- your favorite cloud-based CI (Travis, Circle...)

a spooky story

human: write lots of tests
computer: make decisions

it came from PL research

~~known input → computation → known output~~

it came from PL research

input → computation → property?

it came from PL research

property-based testing
(generative testing)

it came from PL research

human: state the property
computer: generate inputs, find counterexamples

it came from PL research

```
from hypothesis import given
from hypothesis.strategies import lists, floats

@given(lists(floats(allow_nan=False,
allow_infinity=False), min_size=1))
def test_mean_is_within_reasonable_bounds(ls):
    assert min(ls) <= mean(ls) <= max(ls)
```

it came from PL research

generative testing

- hypothesis (python)
- eris (php)
- theft (c)
- jsverify (javascript)
- quickcheck (haskell + friends)

CI/CD

- Jenkins
- your favorite cloud-based CI (Travis, Circle...)

it came from PL research

many inputs are possible
life (and test cycles) are finite

it came from PL research

nondeterminism is a mixed blessing

it came from PL research

less exhaustive exploration
=
less deterministic results

it came from security research

the world of fuzzers
(“doesn't crash” is so a property!)

it came from security research

examine inputs that influence the computation

it came from security research

fantastically effective at finding crashes

it came from security research

[let's see one in action]

it came from security research

interesting inputs → computation → property?

monster mash

human: define properties
computer: generate *interesting* inputs, find counterexamples

a dash of mad science

properties besides “doesn't crash”?

a dash of mad science

reproducibility?

a dash of mad science

runs in cloud CI?

monster mash

fuzzers

- afl-fuzz
- ossfuzz, ...

test harness

- your name here?
- crowbar (ocaml)

CI/CD

- Jenkins
- your favorite cloud-based CI (Travis, Circle...)

non-controversial things

bugs are bad

writing bugs is easy

finding bugs earlier is better

tests can expose bugs

a spooky story

test **all** the things

monster mash

generative testing
+ instrumentation-guided fuzzing
+ CI
= tests you didn't have to write, run every time

spooky resources

https://github.com/yomimono/talks/blob/primary/fuzzops.pdf
http://lcamtuf.coredump.cx/afl
https://github.com/HypothesisWorks/hypothesis-python/
https://zubu.re/blog/fuzzing-automation-with-afl-and-jenkins/
http://www.cse.chalmers.se/~rjmh/QuickCheck/

@yomimono@witches.town
https://somerandomidiot.com