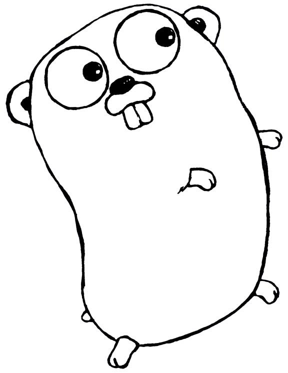
Fuzz Testing Made Easy

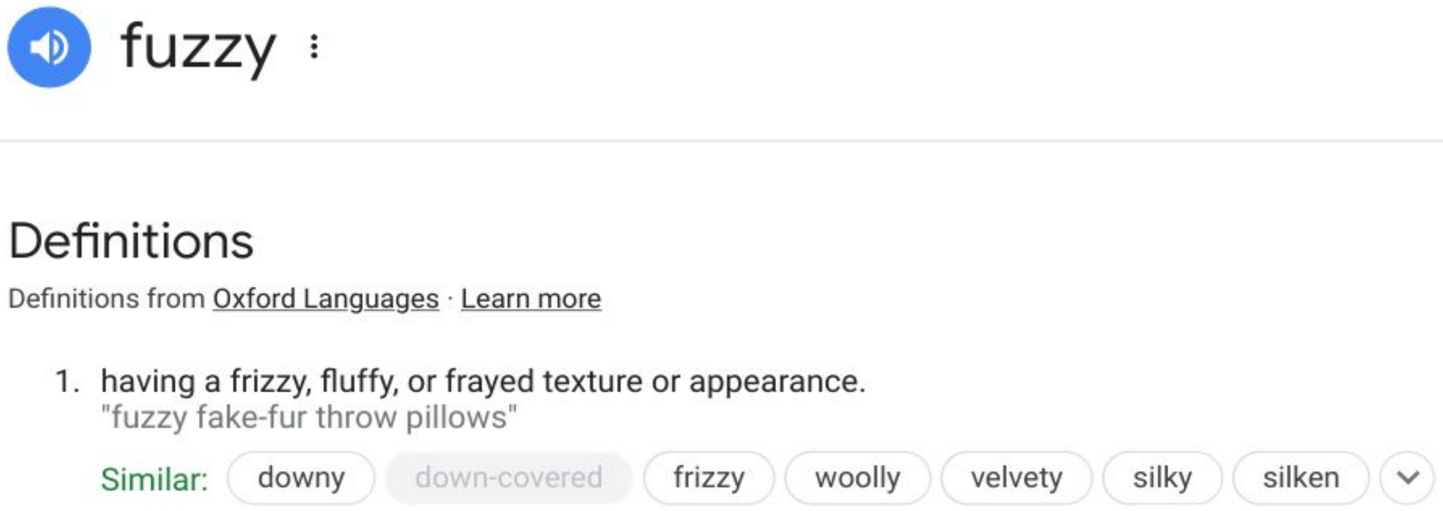
Katie Hockman, Datadog
APM Go Lead
github.com/katiehockman
@katie_hockman



fuzzy

Definitions

Definitions from Oxford Languages · Learn more



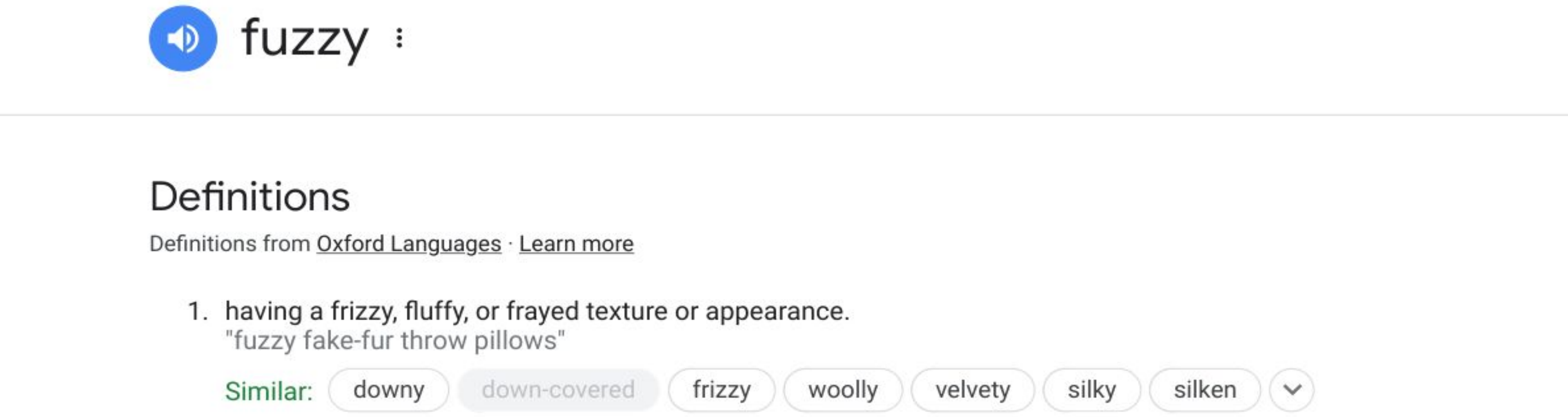
fuzzy

Definitions

Definitions from Oxford Languages · Learn more

1. having a frizzy, fluffy, or frayed texture or appearance.
   "fuzzy fake-fur throw pillows"

   Similar: downy down-covered frizzy woolly velvety silky silken



fuzzy

Definitions

Definitions from Oxford Languages · Learn more

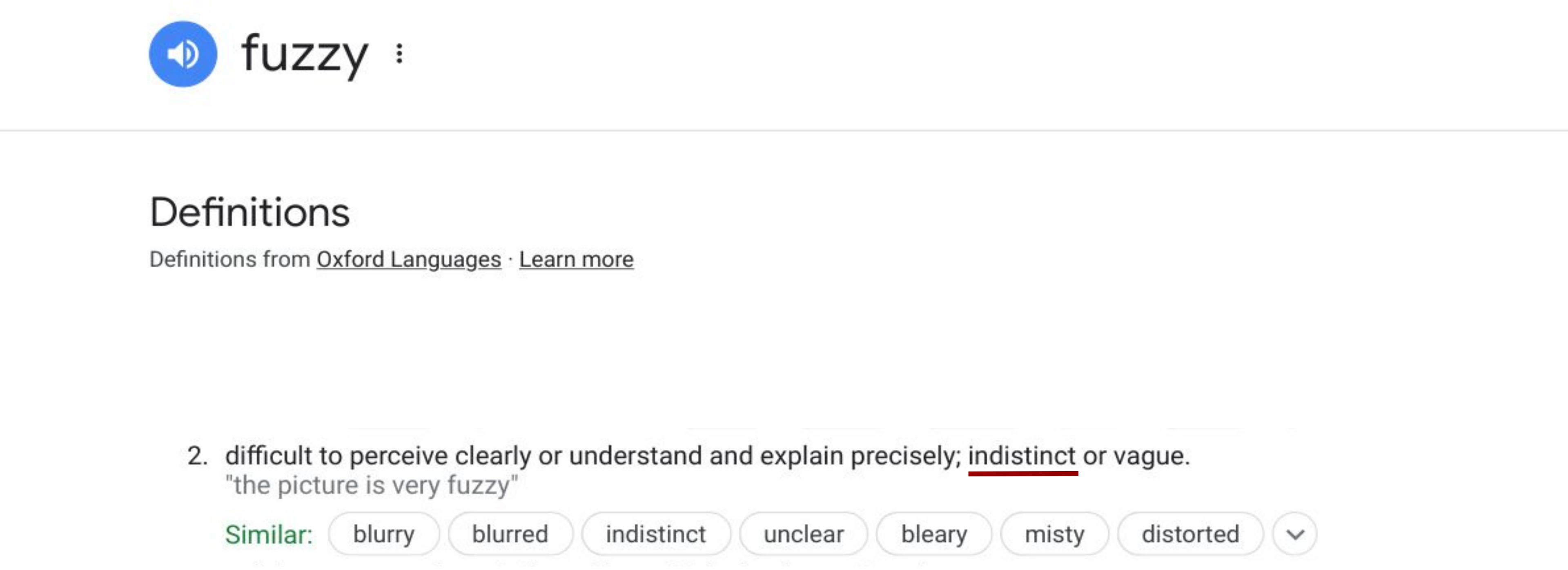
1. having a frizzy, fluffy, or frayed texture or appearance.
   "fuzzy fake-fur throw pillows"

   Similar: downy down-covered frizzy woolly velvety silky silken fuzzy

Definitions

Definitions from Oxford Languages · Learn more

2. difficult to perceive clearly or understand and explain precisely; indistinct or vague.
   "the picture is very fuzzy"
   Similar: blurry blurred indistinct unclear bleary misty distorted



fuzzy

Definitions

Definitions from Oxford Languages · Learn more

2. difficult to perceive clearly or understand and explain precisely; indistinct or vague.
"the picture is very fuzzy"

Similar: blurry · blurred · indistinct · unclear · bleary · misty · distorted

Fuzzing at a glance

Provide starting values (optional)

Start fuzzing!

Report bugs

Go Fuzzing API

Fuzz
test

```
func FuzzFoo(f *testing.F) {






}
```

*testing.F

`Similar API as *testing.T` **+**

- **`(*testing.F).Add`**
- **`(*testing.F).Fuzz`**

Fuzz test

```go
func FuzzFoo(f *testing.F) {




}
```

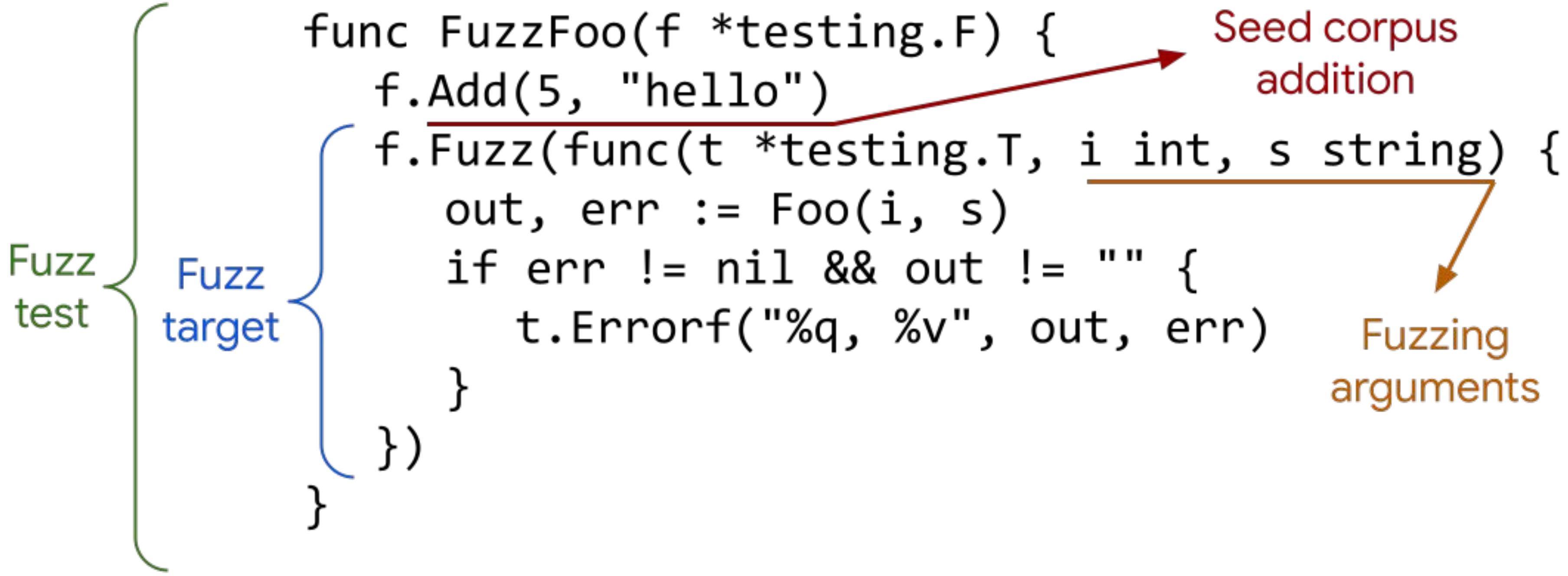
Fuzz test

```go
func FuzzFoo(f *testing.F) {
    f.Add(5, "hello")




}
```

Seed corpus addition



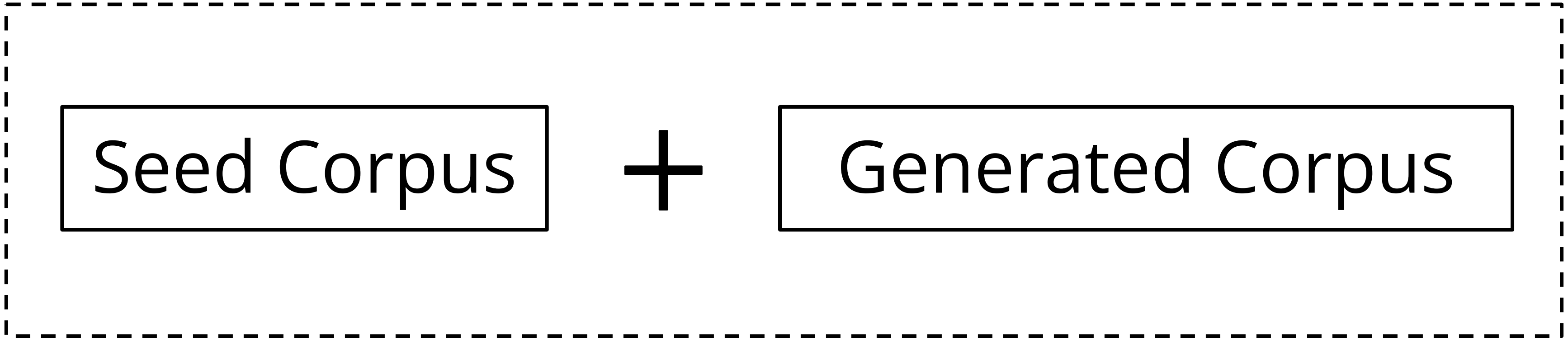
```go
func FuzzFoo(f *testing.F) {
	f.Add(5, "hello")
	f.Fuzz(func(t *testing.T, i int, s string) {
		out, err := Foo(i, s)
		if err != nil && out != "" {
			t.Errorf("%q, %v", out, err)
		}
	})
}
```

Go Fuzzing API

The corpus

Corpus

A collection of inputs that guide fuzzing.



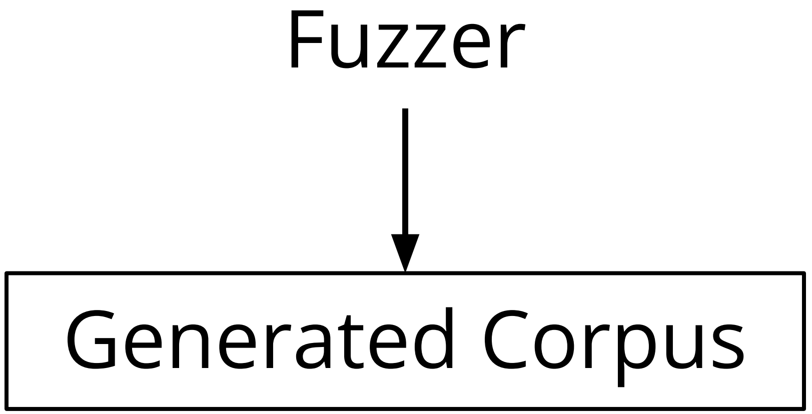
Seed Corpus

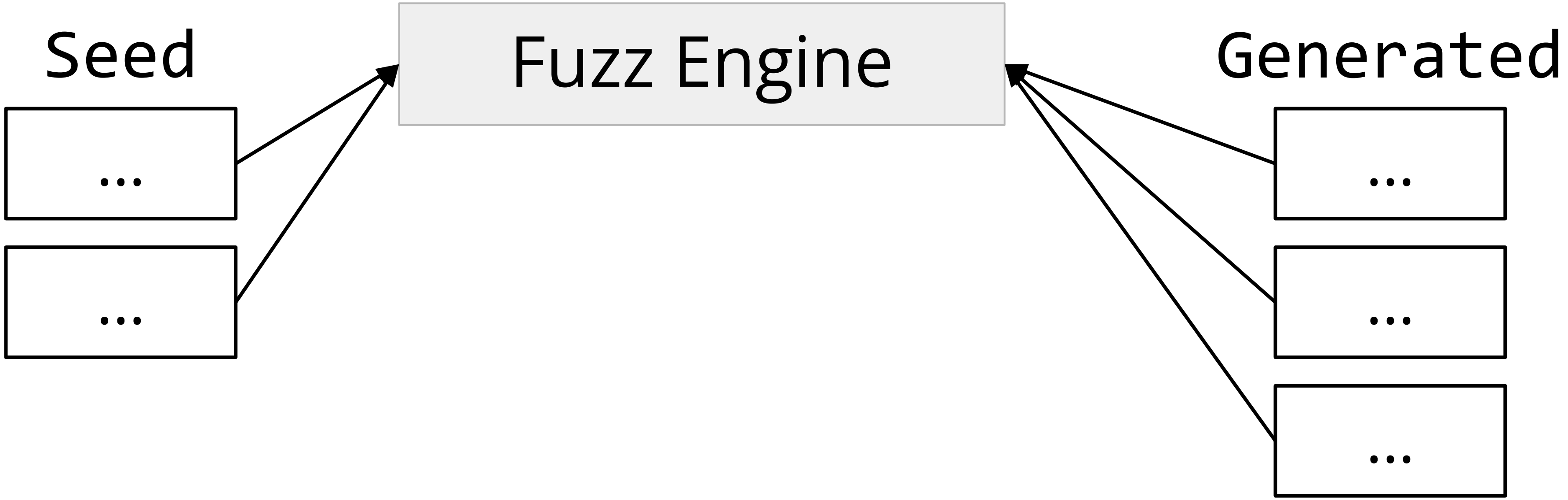
A **user provided** dataset

$$\begin{tikzcd}
\text{f.Add() calls} \arrow[dr] & & \text{testdata/fuzz/Fuzz*} \arrow[dl] \\
& \boxed{\text{Seed Corpus}} &
\end{tikzcd}$$

Generated Corpus

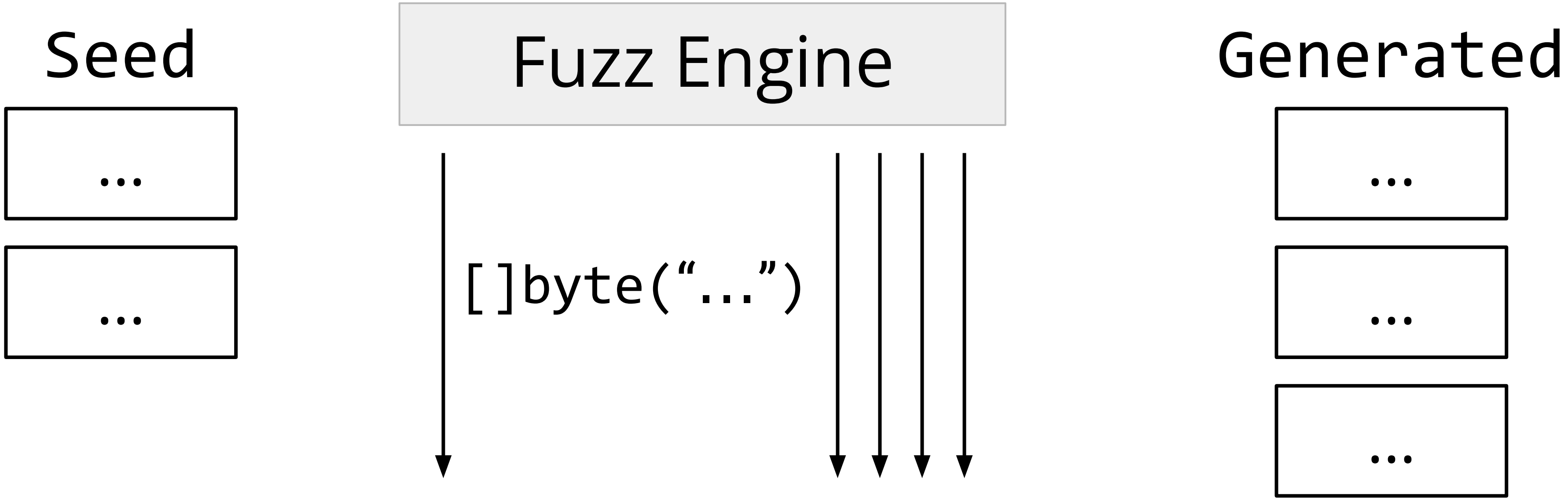
A **machine generated** dataset





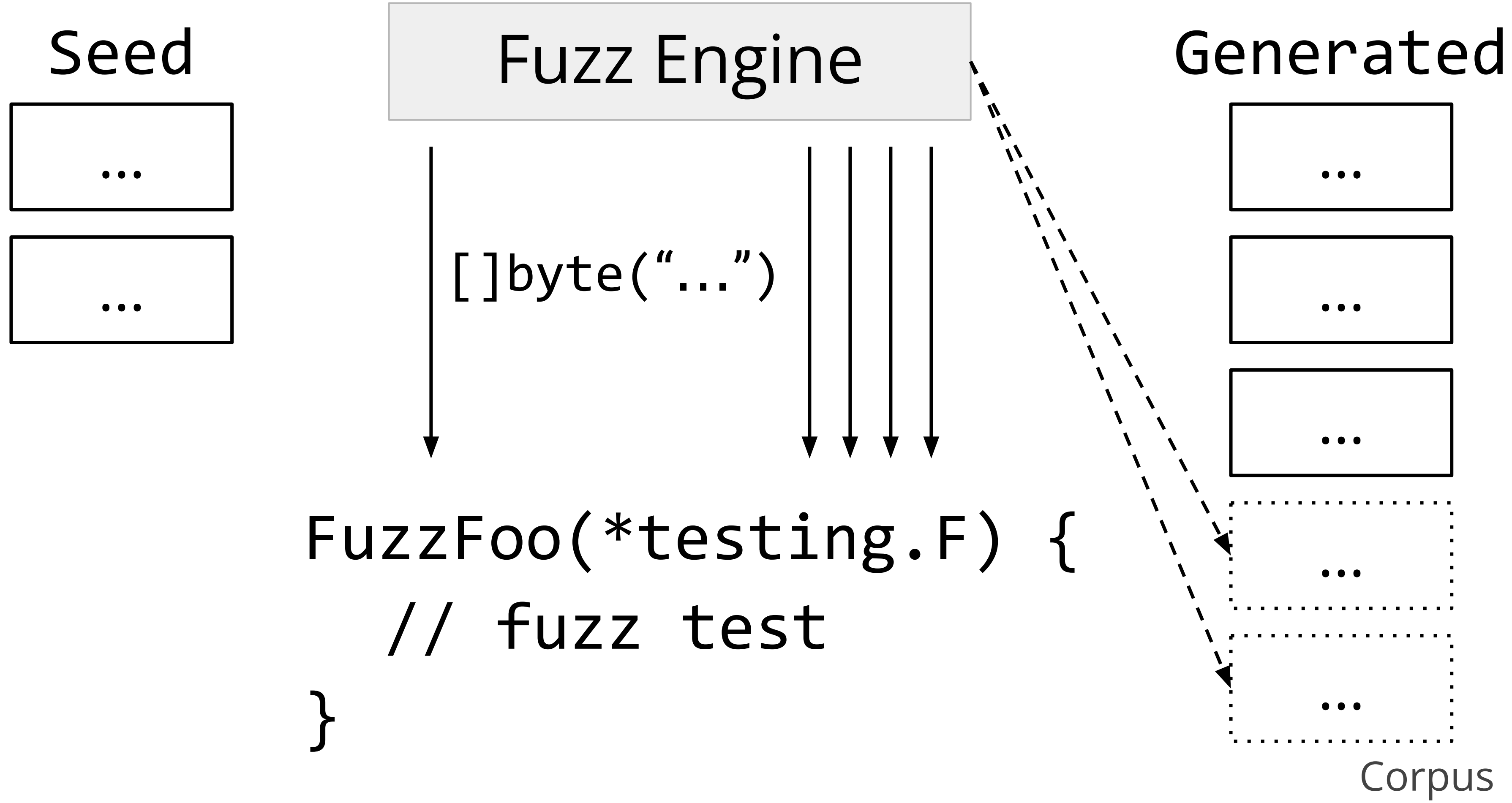
```
FuzzFoo(*testing.F) {
   // fuzz test
}
```

Corpus

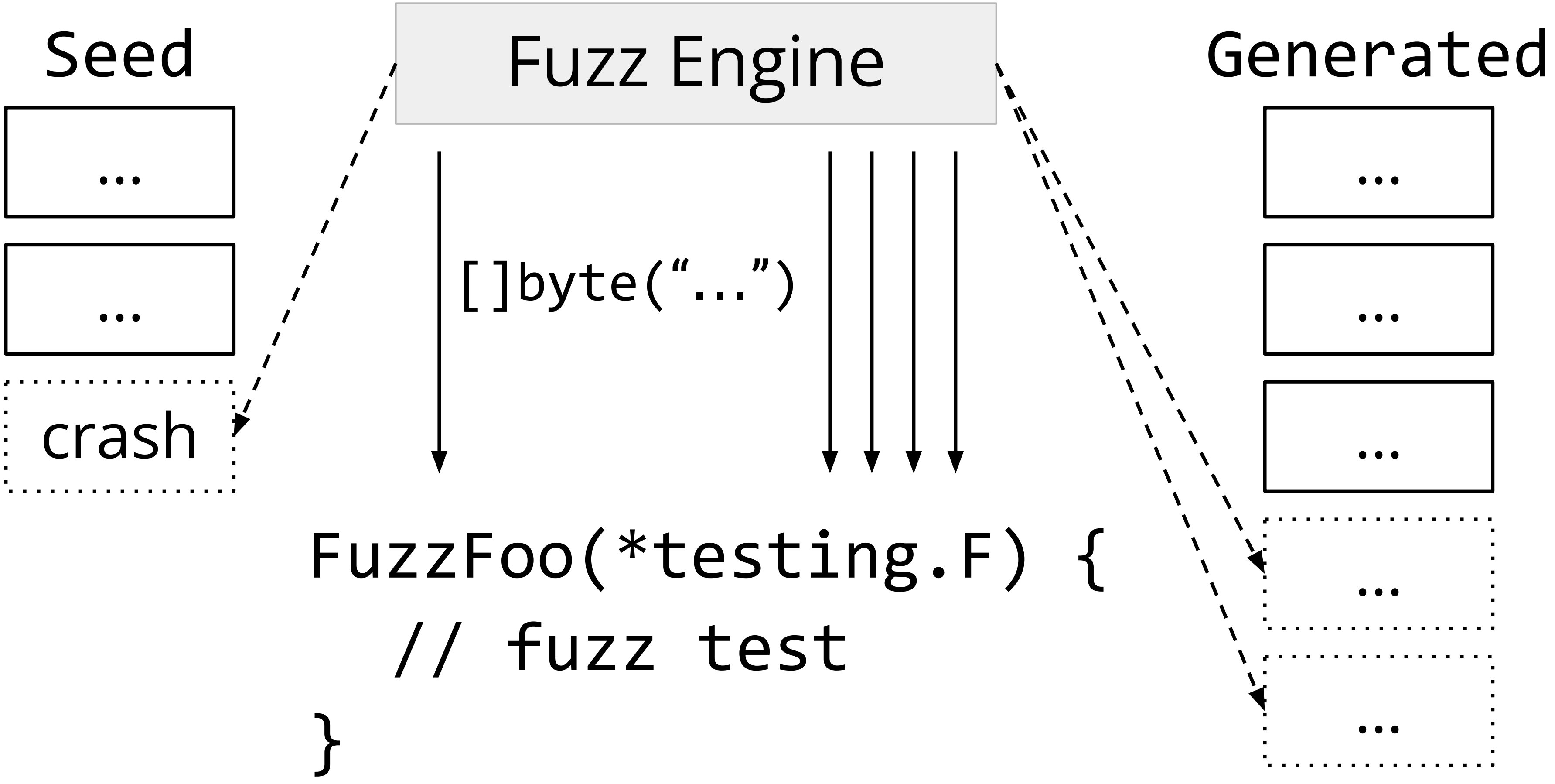


```
FuzzFoo(*testing.F) {
    // fuzz test
}
```

Corpus



```
FuzzFoo(*testing.F) {
   // fuzz test
}
```

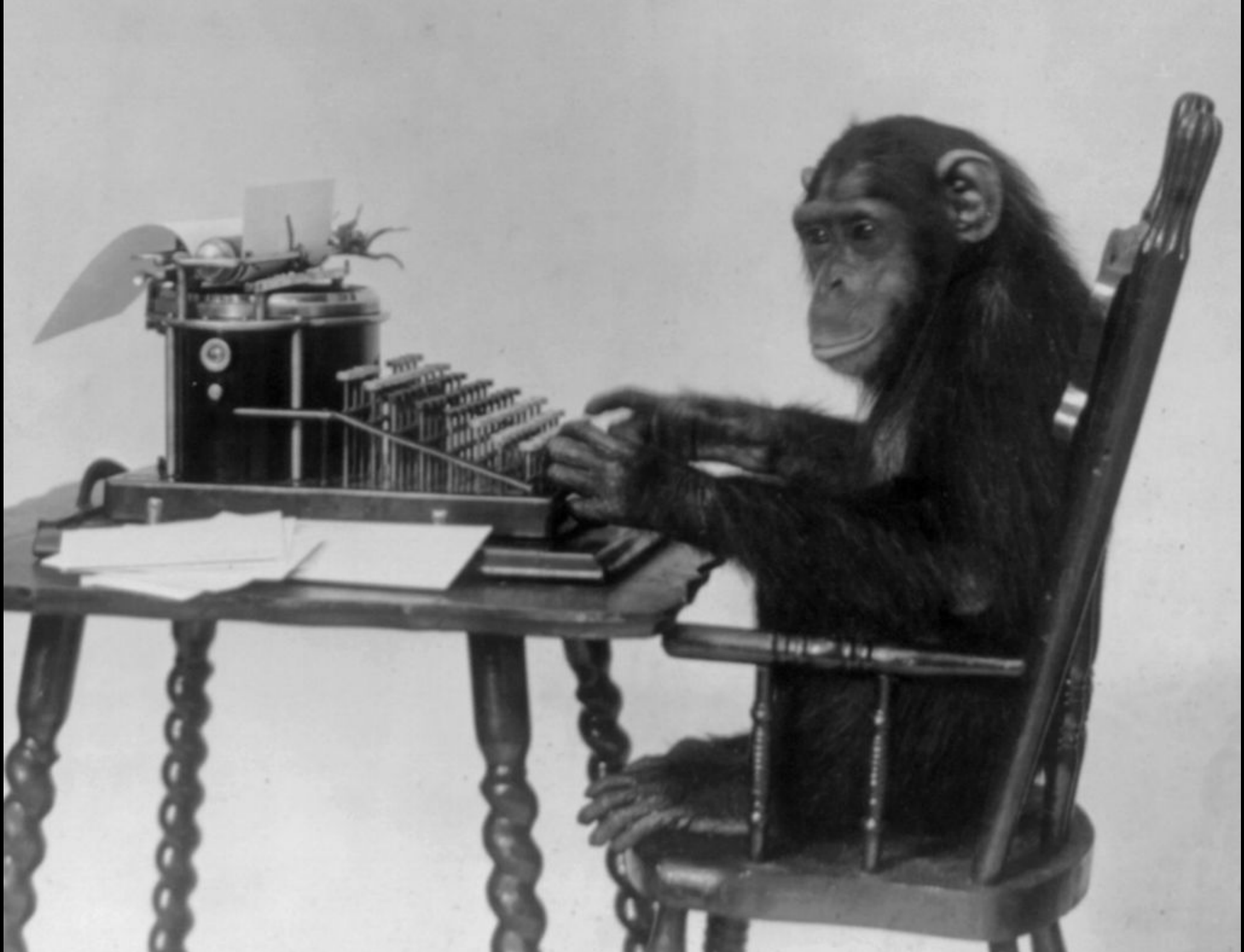


```
FuzzFoo(*testing.F) {
   // fuzz test
}
```

Corpus

Demo

Effective Fuzzing



What makes it naive?

- Inputs unrelated
- Ignores outcome
- Structure agnostic

What makes it naive?

- Inputs unrelated

1. abcde
2. uwozkfa3
3. 028zk8d
4. 23
5. EkciJx:1i3j

1. <html>abc</html>
2. <html><html>abc</html>
3. <a>htmlbc</html>
4. <>htA3cc<>html
5. <html>1bc/<html>

What makes it naive?

- Inputs unrelated
- Ignores outcome

1. abcde
2. uwozkfa3
3. 028zk8d
4. 23
5. EkciJx:1i3j

1. <html>abc</html>
2. <html><html>abc</html>
3. <a>htmlbc</html>
4. <>htA3cc<>html
5. <HTML>1bc/<html>

1. abcde
2. uwozkfa3
3. 028zk8d
4. 23
5. EkciJx:1i3j

1. <html>abc</html>
2. <html><html>abc</html>
3. **<a>htmlbc</html>**
4. <>htA3cc<>html
5. <HTML>1bc/<html>

What makes it naive?

- Inputs unrelated
- Ignores outcome
- Structure agnostic

```
Foo(i int, b byte, s string)
```

```go
func FuzzFoo(f *testing.F) {
  f.Fuzz(func(t *testing.T, in []byte) {
    if len(in) < 9 { return }
    i, err := strconv.Atoi(string(in[:8]))
    if err != nil { return }
    b, s := in[8], string(in[9:])
    Foo(i, b, s)
  })
}
```

```go
func FuzzFoo(f *testing.F) {
  f.Fuzz(func(t *testing.T, in []byte) {
    if len(in) < 9 { return }
    i, err := strconv.Atoi(string(in[:8]))
    if err != nil { return }
    b, s := in[8], string(in[9:])
    Foo(i, b, s)
  })
}
```

```go
func FuzzFoo(f *testing.F) {
  f.Fuzz(func(t *testing.T, i int, b byte,
        s string) {
    Foo(i, b, s)
  })
}
```

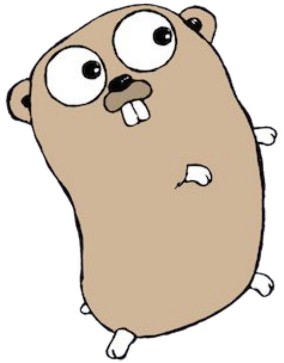
What makes it naive?

- Inputs unrelated
- Ignores outcome
- Structure agnostic

Let's make it better!

+ Mutation-based
+ Coverage guided
+ Structured inputs

Happy fuzzing!!



Katie Hockman

github.com/katiehockman

Twitter: @katie_hockman

#katie-hockman on Gophercon Discord

Resources

- If you find a bug with Go fuzzing, consider adding this to the trophy case on the wiki!
- Further reading:
    - Fuzzing Landing Page
    - Design Draft
    - Proposal
    - Tutorial
    - `testing` package docs