go-fuzz

randomized testing for Go

24 September 2015

Marty Schoch

Fuzz Testing?

- Automated
- Provide invalid, unexpected, or random input
- Monitor for crash, excessive memory allocation, or hang

go-fuzz

- Dmitry Vyukov and 36 other contributors
- Write one Fuzz() function
- Provide initial seed data corpus
- Testing proceeds by mutating inputs, guided by code coverage
- Mulitple slaves operate in parallel, saves progress to disk, can resume

github.com/dvyukov/go-fuzz (https://github.com/dvyukov/go-fuzz)

Step 1

Install go-fuzz

```
go get github.com/dvyukov/go-fuzz/go-fuzz
go get github.com/dvyukov/go-fuzz/go-fuzz-build
```

Step 2

Add a Fuzz() fuction, the entry point for fuzz testing your application.

```
func Fuzz(data []byte) int {
    if _, err := YourAppFunction(data); err != nil {
        return 0
    }
    return 1
}
```

- One argument, []byte to be tested
- Return 0 on error
- Return 1 on success

Step 3

Generate an initial data corpus

- Need not be large, but helps yield interesting results sooner
- Often you can generate this from existing unit tests
- We put these in:

```
workdir/corpus
```

Step 4

Build your package with go-fuzz-build

```
go-fuzz-build github.com/yourorg/yourapp
```

This is where go-fuzz builds a version of your program with instrumentation to detect crashes and measure code coverage. This generates a zip file containing everything go-fuzz needs.

```
yourapp-fuzz.zip
```

Step 5

Run go-fuzz:

```
$ go-fuzz -bin=yourapp-fuzz.zip -workdir=workdir
2015/09/08 14:19:19 slaves: 8, corpus: 1859 (0s ago), crashers: 0, restarts: 1/0, execs: 0 (0/sec
2015/09/08 14:19:22 slaves: 8, corpus: 2486 (0s ago), crashers: 0, restarts: 1/0, execs: 0 (0/sec
2015/09/08 14:19:25 slaves: 8, corpus: 2486 (3s ago), crashers: 0, restarts: 1/6850, execs: 54804
... data omitted ...
2015/09/08 15:04:10 slaves: 8, corpus: 7255 (1m6s ago), crashers: 0, restarts: 1/9982, execs: 672
2015/09/08 15:04:13 slaves: 8, corpus: 7255 (1m9s ago), crashers: 0, restarts: 1/9985, execs: 672
2015/09/08 15:04:16 slaves: 8, corpus: 7265 (0s ago), crashers: 0, restarts: 1/9984, execs: 67354
```

Live Demo

Summary

- Has been used successfully to find bugs in Go standard library
- Project README has list of 'Trophies' found by the tool
- Yes, they have used go-fuzz to test go-fuzz :)
- At Couchbase we've found bugs in our N1QL parser and wire protocol parsing
- Bleve search uses it on our Unicode segmentation library

github.com/dvyukov/go-fuzz (https://github.com/dvyukov/go-fuzz)

Thank you

Marty Schoch
marty@couchbase.com (mailto:marty@couchbase.com)
@mschoch (http://twitter.com/mschoch)