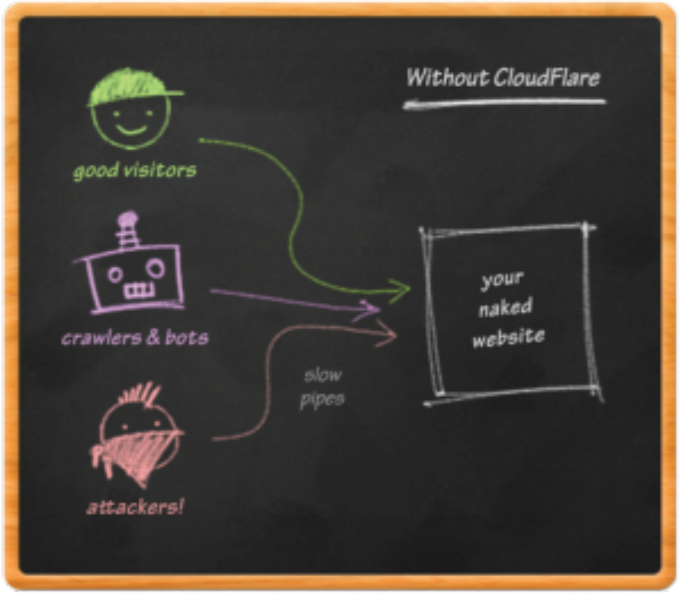
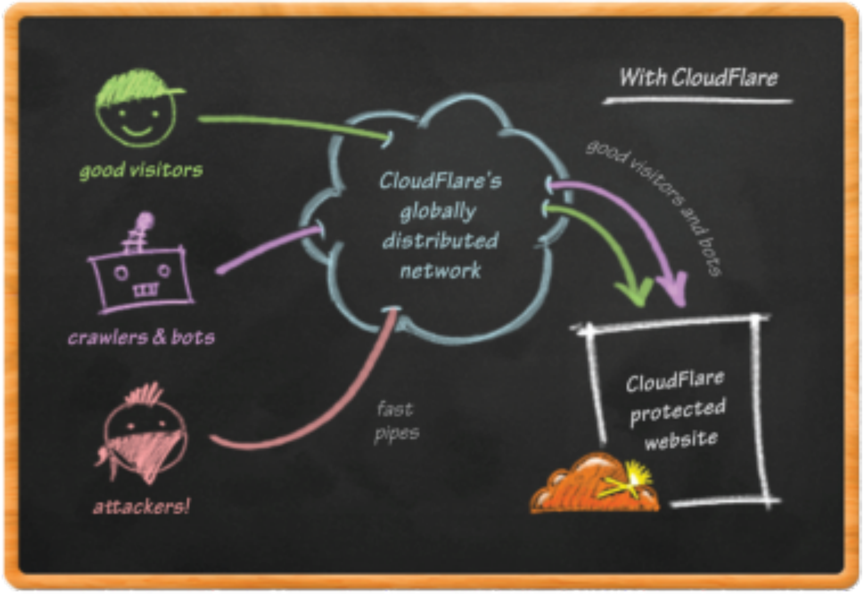
Gophers & Gremlins
An Introduction to Go

Ray Bejjani
Engineer
CloudFlare

What is CloudFlare?

CloudFlare makes websites faster, safer and smarter using our distributed network to deliver essential services to any website:





- Performance
- Content Optimization
- Security

- Analytics
- Third party services
- Monetization

What is CloudFlare?

- Billions of requests per day
- Throughput oriented work profile

  IO bound: disk and network
- Highly parallel

  Small changes accumulate

Go Resources

- Search for "golang" instead of "go"
- golang.com supports opensearch

  golang.com <tab> query === magic!
- golang-nuts mailing list

The Good

- Concurrency

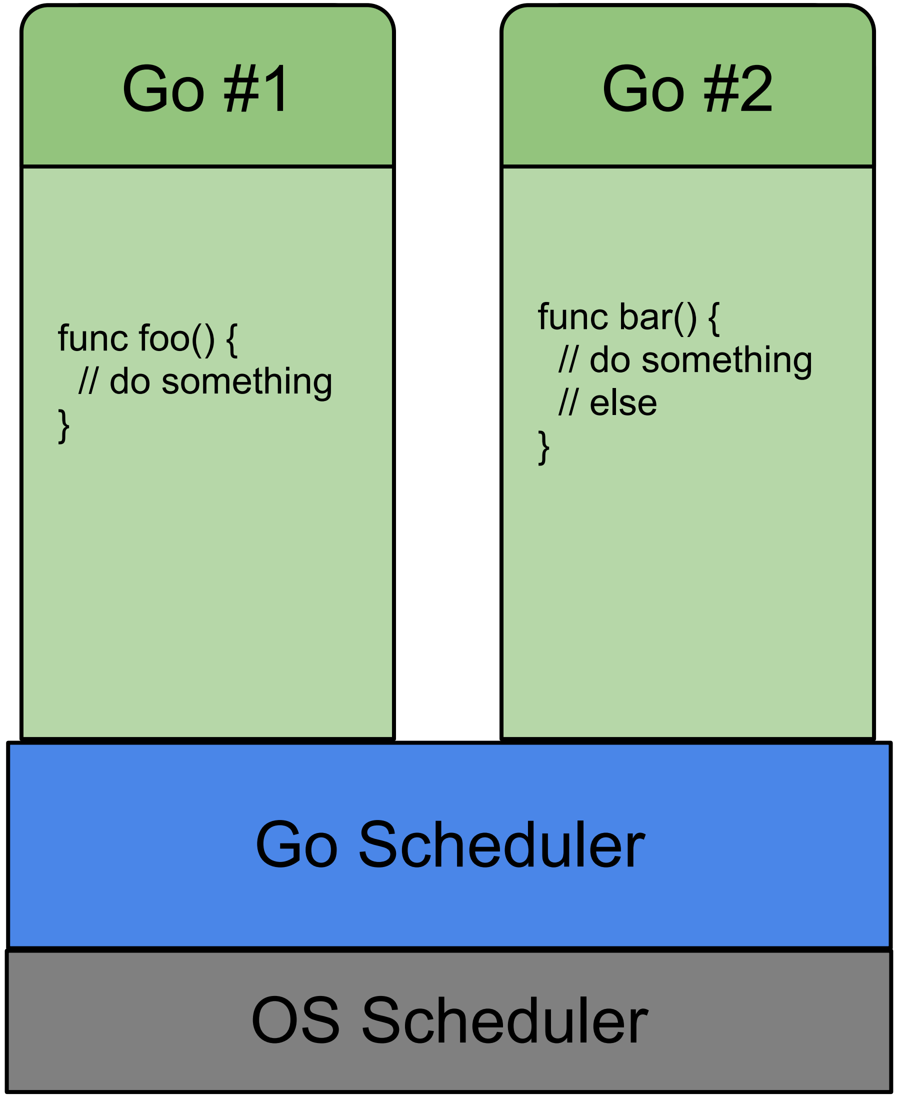
  goroutines & channels

- Type System

  Interfaces & slices (& maps)

- Garbage Collected

First Class Concurrency: Goroutines



```
func foo() {
  // do something
}
```

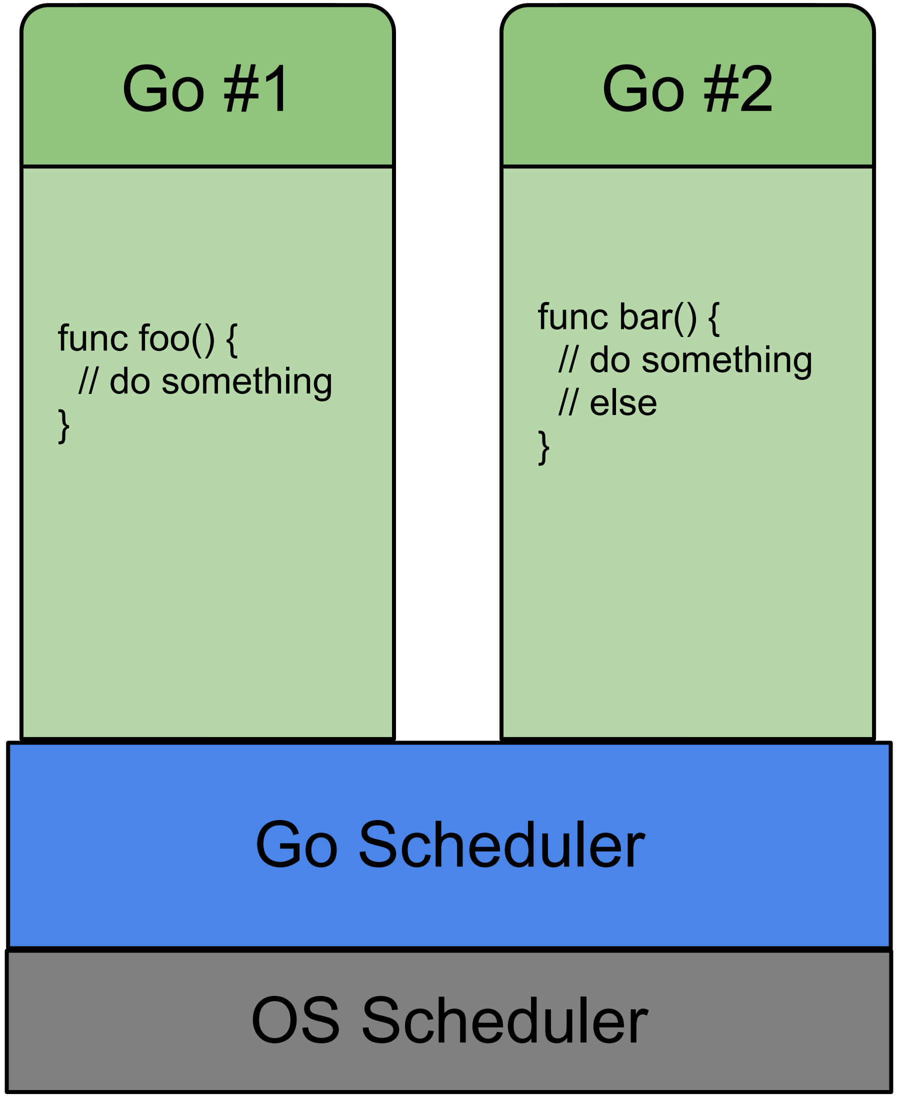
```
func bar() {
  // do something
  // else
}
```

- Zero cost logical parallelism
- Separate stacks

  https://gist.github.com/jgrahamc/5253020/
- Scheduling done for you
- Blocking IO yields routine

  Doesn't block process

First Class Concurrency: Goroutines

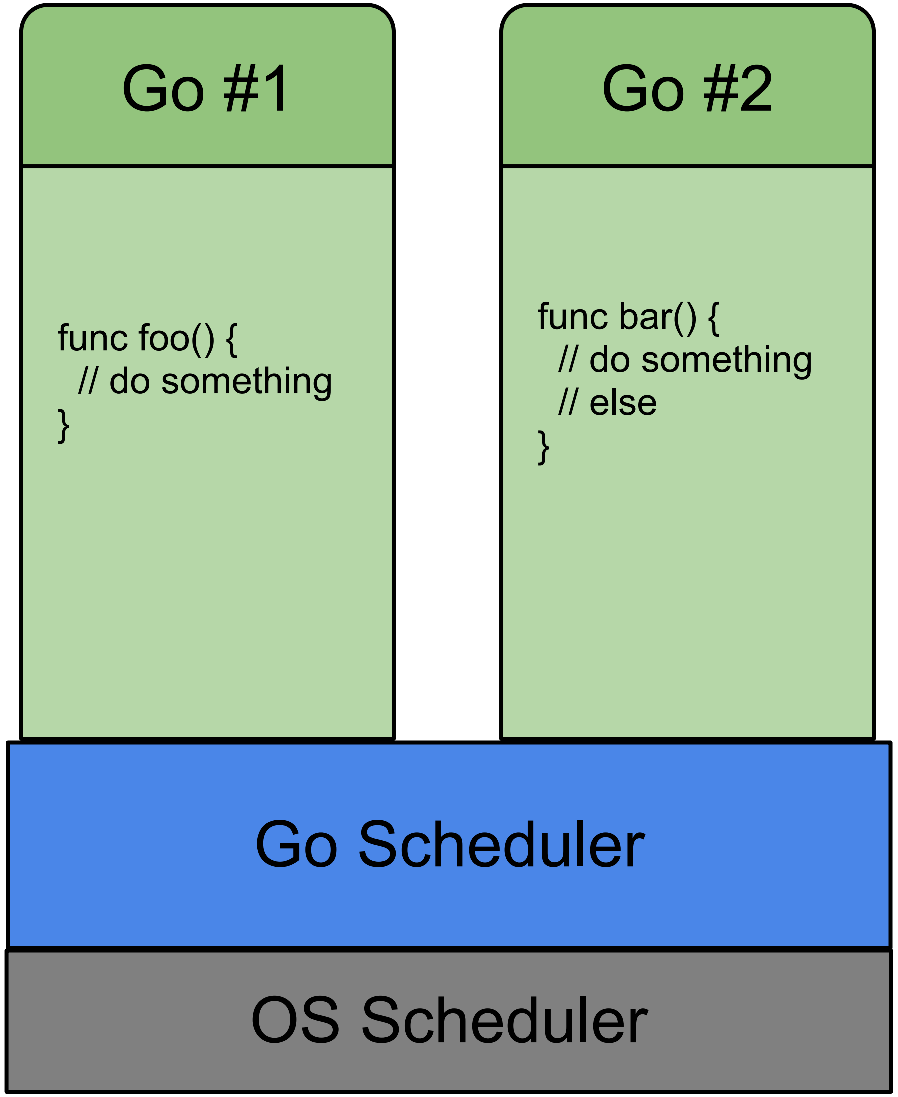


```
func foo() {
  // do something
}
```

```
func bar() {
  // do something
  // else
}
```

Let's write a url-grabber

First Class Concurrency: Goroutines



Go #1

```
func foo() {
  // do something
}
```

Go #2

```
func bar() {
  // do something
  // else
}
```

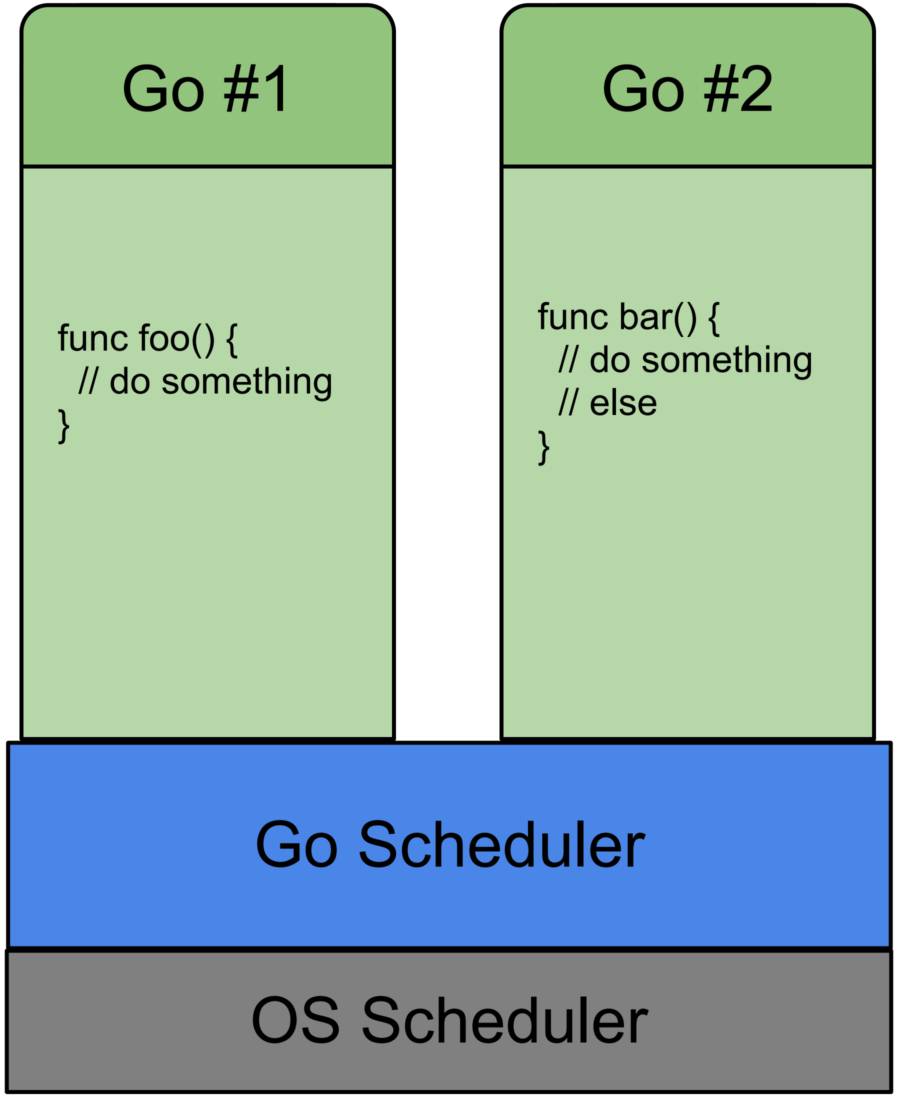
Go Scheduler

OS Scheduler

Let's write a url-grabber

because that's hip...

First Class Concurrency: Goroutines



```
func foo() {
  // do something
}
```

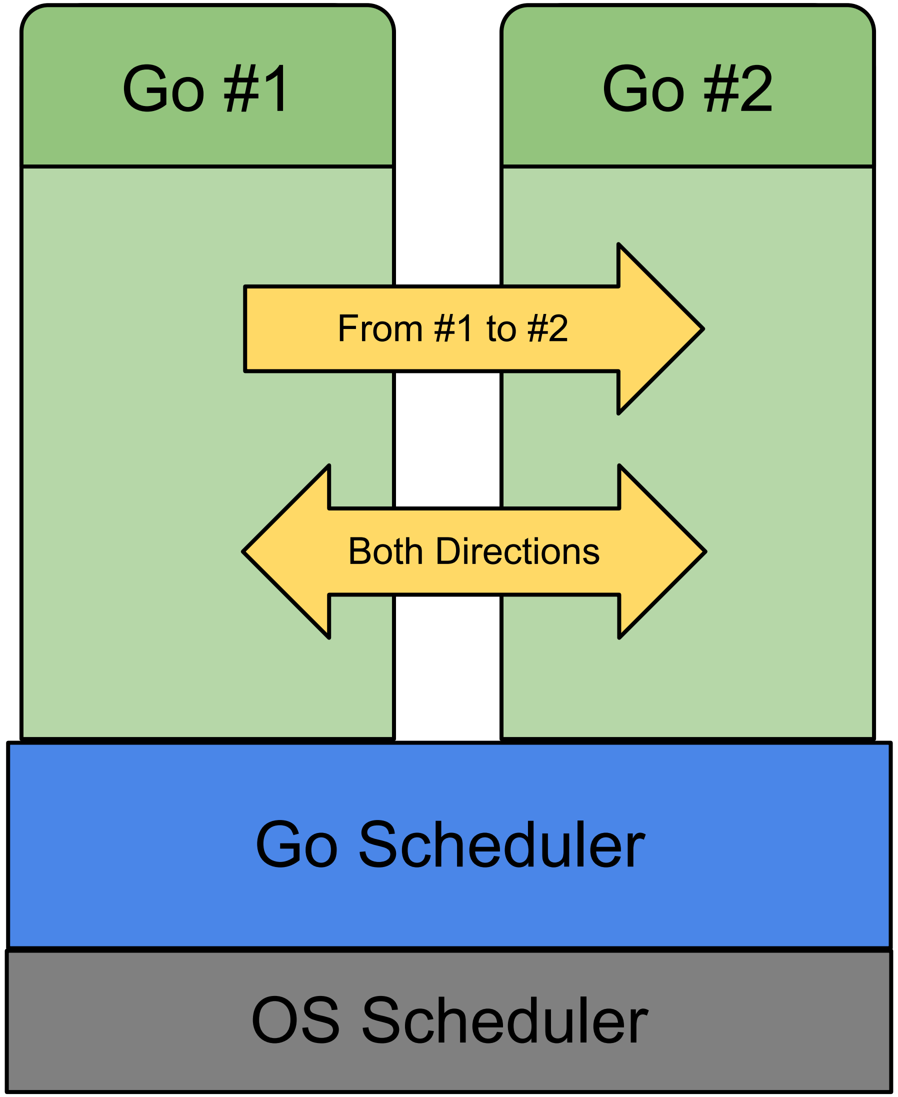
```
func bar() {
  // do something
  // else
}
```

Let's write a url-grabber

- Given a list of urls, get them
- Track durations of gets

https://github.com/raybejjani/ccsf-gophers/blob/84c4dd026cb81b463591face6db8b00bac0115e3/src/crawler/crawl.go

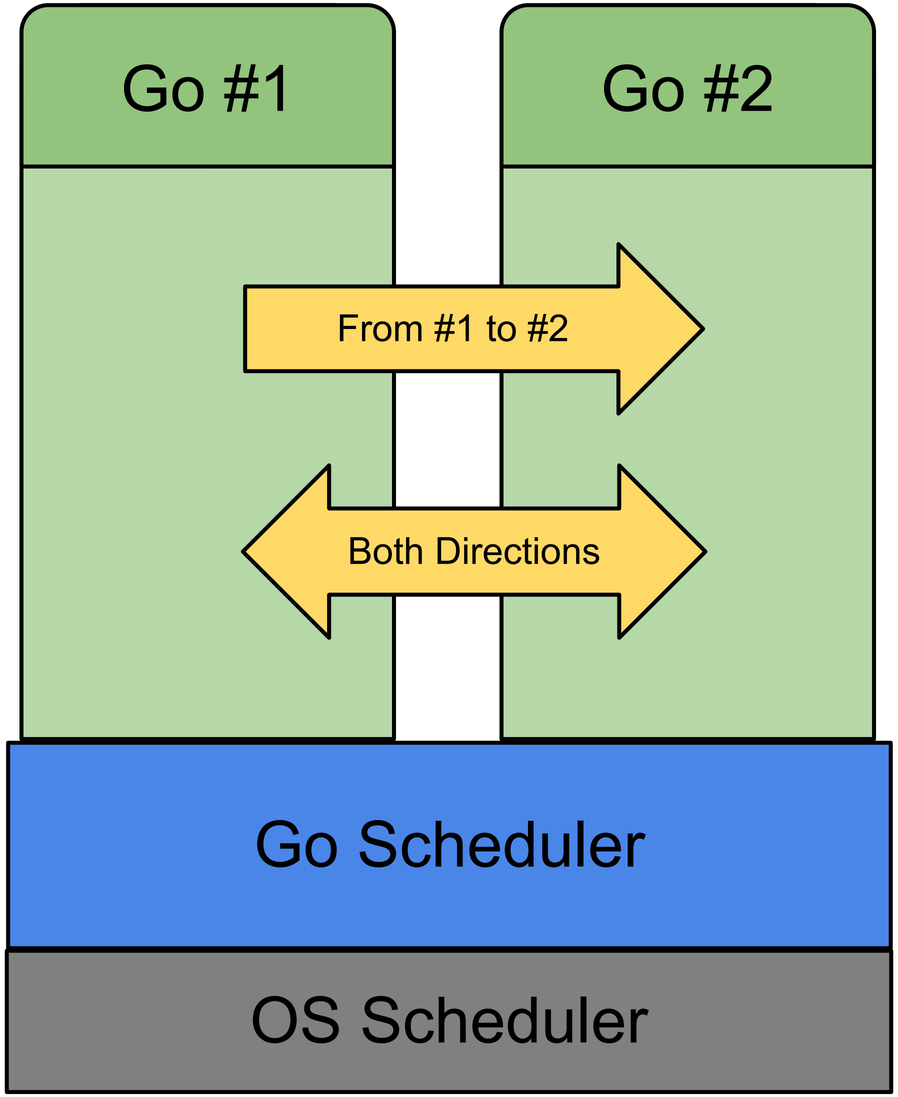
First Class Concurrency: Channels



- Data passing
- Synchronisation via "happens before" partial ordering

  Close operation is a signal
- Uni/Bi-directional
- Handy as FIFO queues

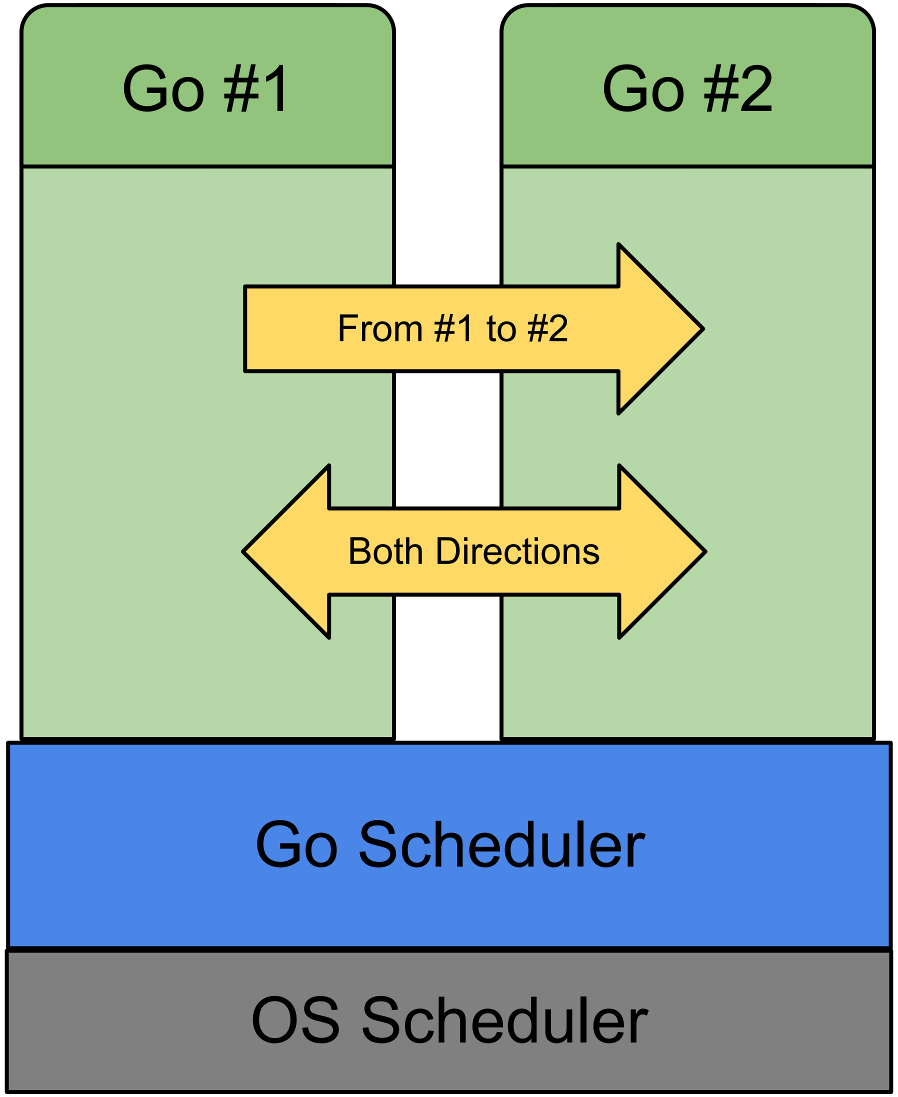
First Class Concurrency: Channels



Let's write a url-grabber

fix it so it actually works...

First Class Concurrency: Channels



Let's write a url-grabber

- Use channels to ensure sequencing

https://github.com/raybejjani/ccsf-gophers/blob/c2de1bac9f0b75a4b654c7e4f82756836f66b3da/src/crawler/crawl.go

Type System: Primitives

Integers, Floats

0, -1, 1.234

- 8, 16, 32 and 64 bit ints
- Signed and unsigned
- 32 and 64 bit floats

Strings, Arrays

"I am a string\n"

[3]byte{0,1,255}

- Strings are interned

  i.e. Immutable
- String length included
- Length part of array type

Type System: Reference Types

Slices

```
[]SomeType{...}
```

- pointer to beginning of data
- size of data
- capacity of data

- Can be resized

Maps

```
map[string]int {
    "Key": Value,
    "KeyTwo": OtherValue,
}
```

- Resizes automatically
- Any type for key and value

Type System: Reference Types

Slices

```
[]SomeType{...}
```

- pointer to beginning of data
- size of data
- capacity of data

- Can be resized

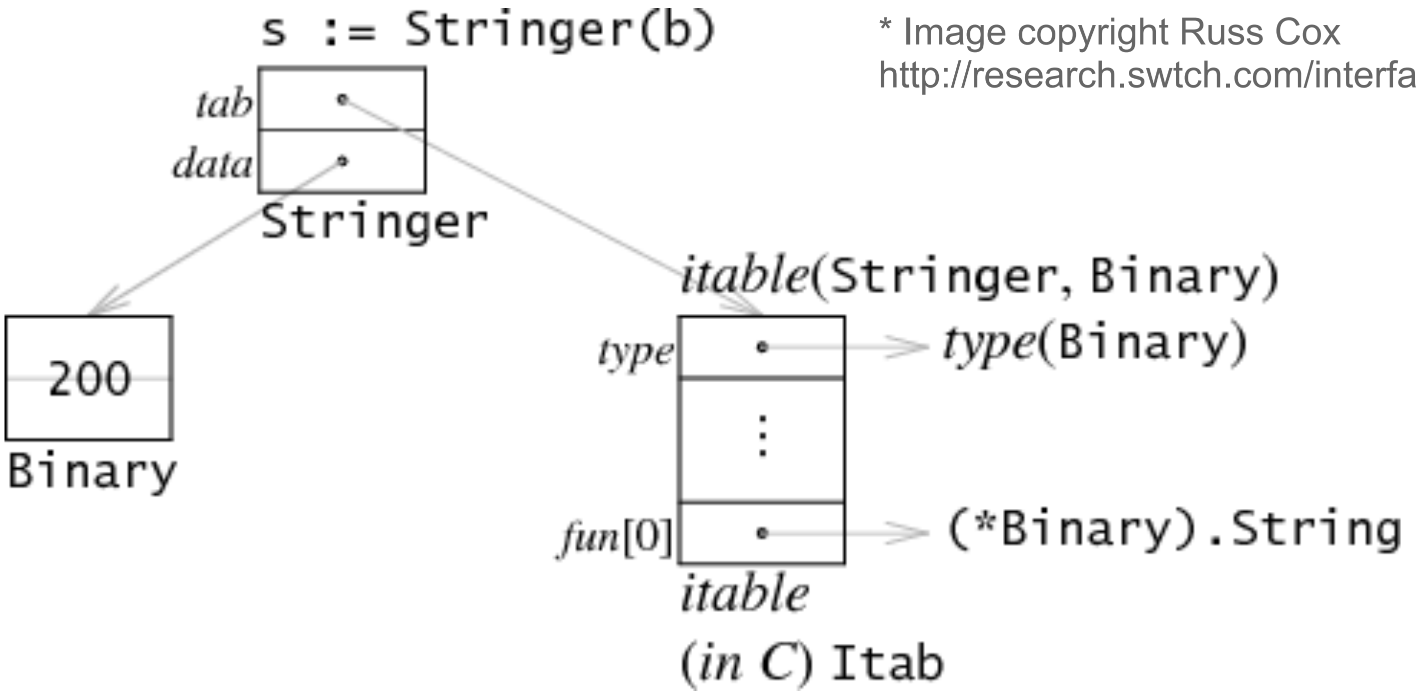
Maps

```
map[string]int {
    "Key": Value,
    "KeyTwo": OtherValue,
}
```

- Resizes automatically
- Any type for key and value

Channels are a reference type too

Type System: Interface Types



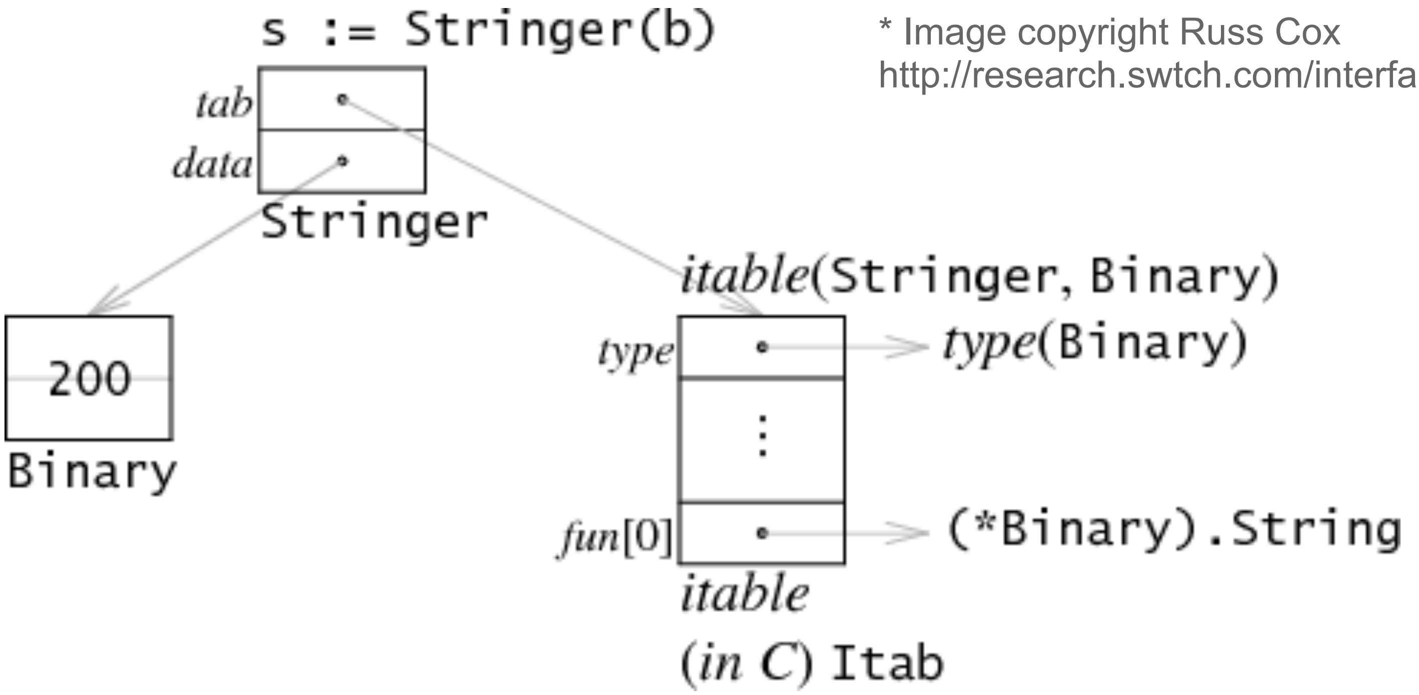
* Image copyright Russ Cox
http://research.swtch.com/interfaces

- Allow Type composition

  Mixins but keep concrete type info

- Cheap, effective type inference

Type System: Interface Types

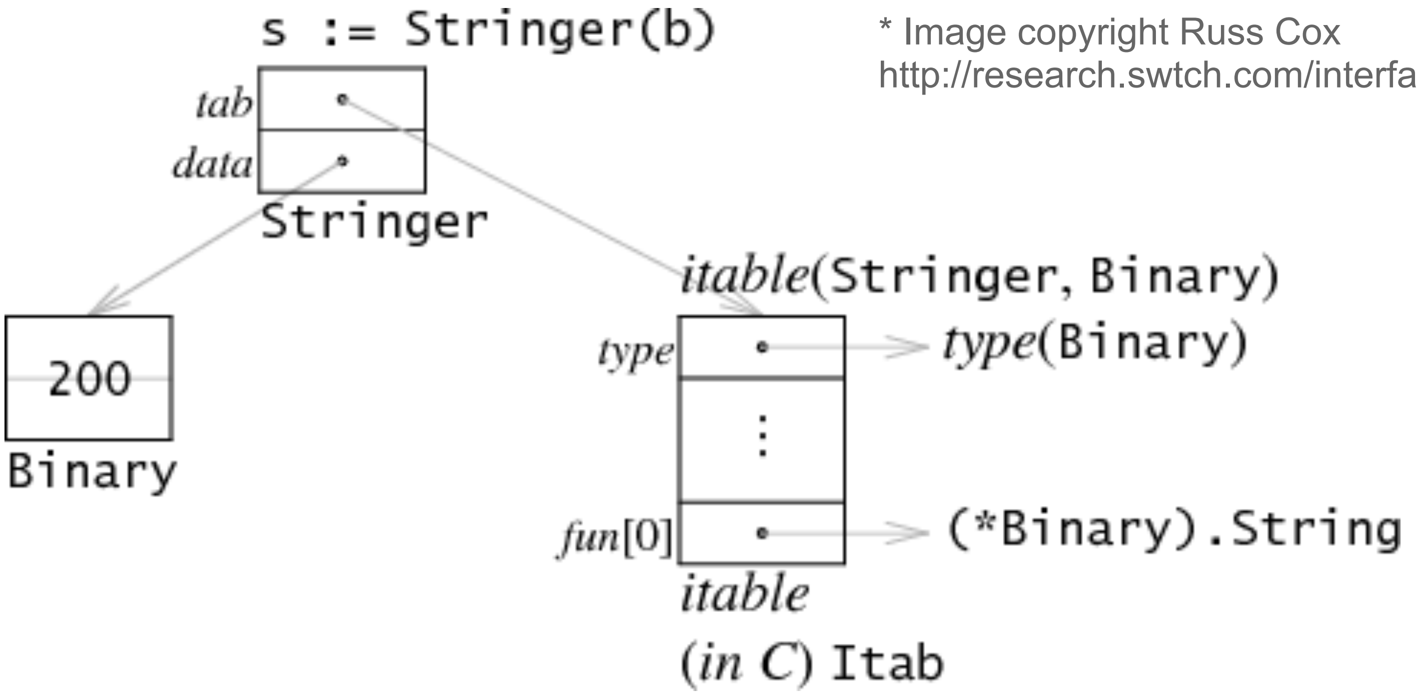


* Image copyright Russ Cox
http://research.swtch.com/interfaces

- Types implicitly meet an interface

  Can use them like Groups in Objective-C

Type System: Interface Types

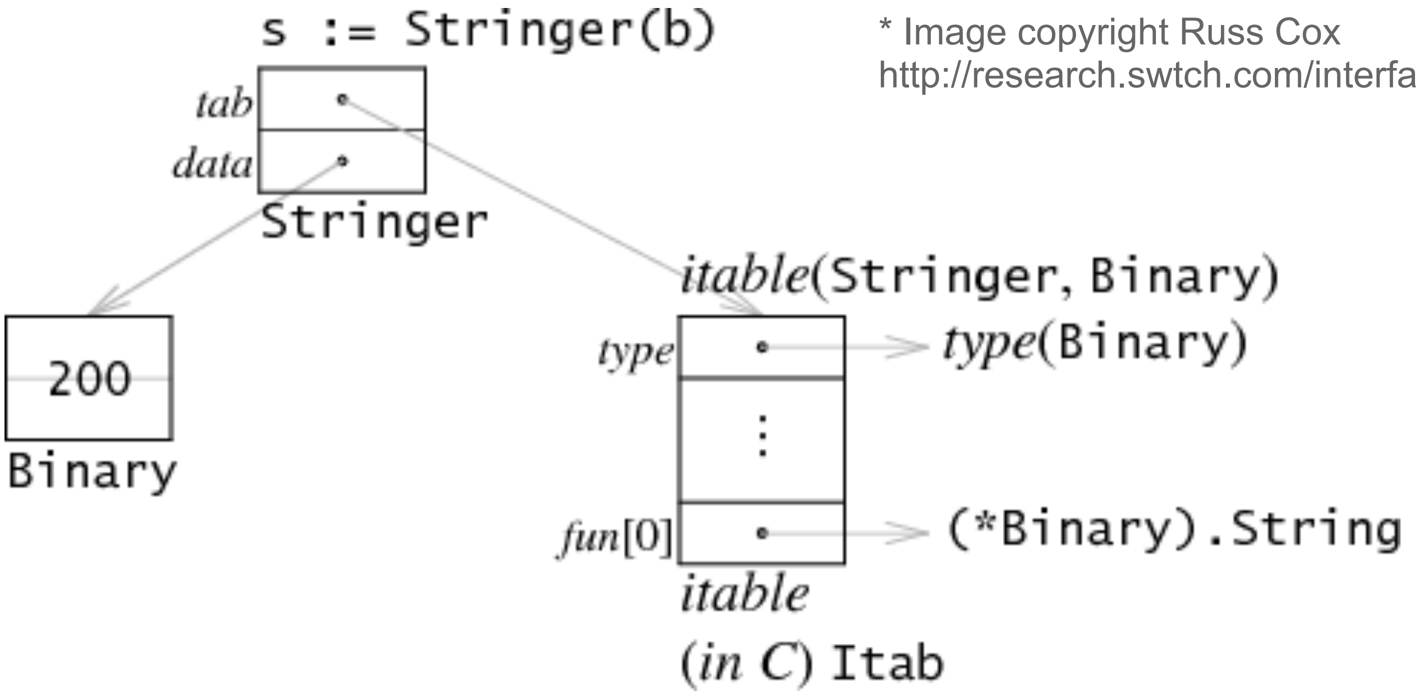


* Image copyright Russ Cox
http://research.swtch.com/interfaces

Let's write a url-grabber

add some bling...

Type System: Interface Types



* Image copyright Russ Cox
http://research.swtch.com/interfaces

Let's write a url-grabber

- Track the Vary header

https://github.com/raybejjani/ccsf-gophers/blob/a4267256a6f1fc72761327578f18ae66d7da41bf/src/crawler/crawl.go

Garbage Collected

- Mark-and-Sweep
- Stop-the-World
- Can attach Finalizer to objects

  Gets called instead of object being freed
- Escape analysis avoids heap allocation

The Bad

- Build system
- Threading model
- Developing Tools & Libraries

Build System: No Hooks, No Preprocessor

Tag based system to include/exclude files

```
go install -tags 'release production'
```

Example: How to version a binary?

https://github.com/raybejjani/ccsf-gophers/tree/9caeab29acfc7db15a4aef693a54f9554b1b445a/src/crawler

Alternatively,

```
go run -ldflags "-X main.xyz abc" main.go
```

Threading Model

Goroutines can spawned before main.

- Daemonizing and user changing may not work
- fork-exec a child with a different user

Start via runners (circus etc.) that handle the privilege drop for you.

Developing Tools & Libraries

Analysis of goroutine parent-child relationships and lifetime is hard

- panic trace
- profiler graph for memory and CPU

Developing Tools & Libraries

Analysis of goroutine parent-child relationships and lifetime is hard

- Race Detector in Go 1.1

The Reality: Three projects in Go

- SSL certificate service
- Railgun
- DNS infrastructure

CLOUDFLARE®

Thank You
Questions?

Ray Bejjani
Engineer
CloudFlare