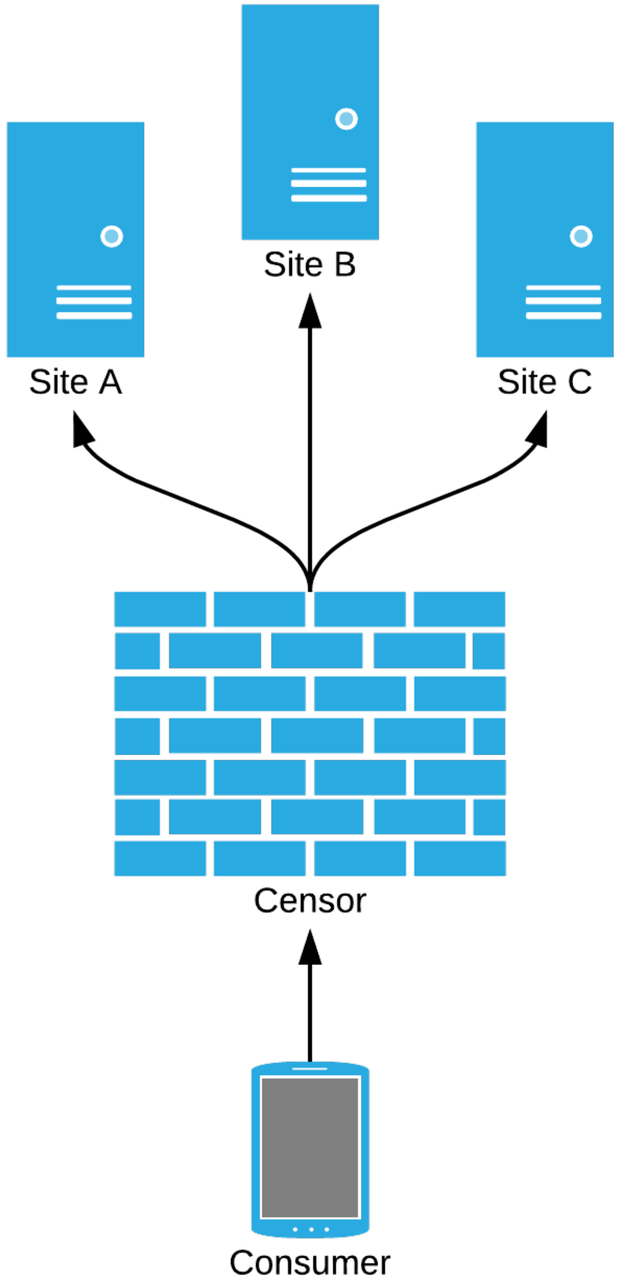
Go as the backbone of Lantern for Android

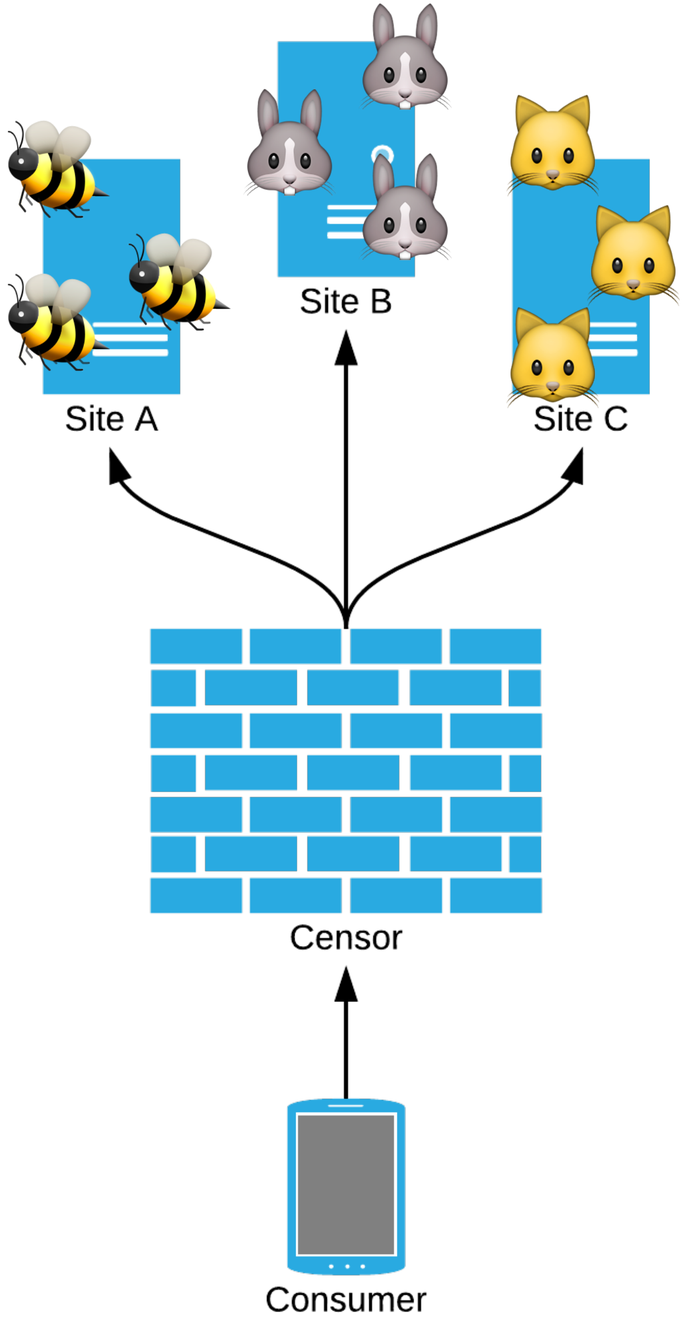
github.com/xiam
@xiam

Meet Lantern

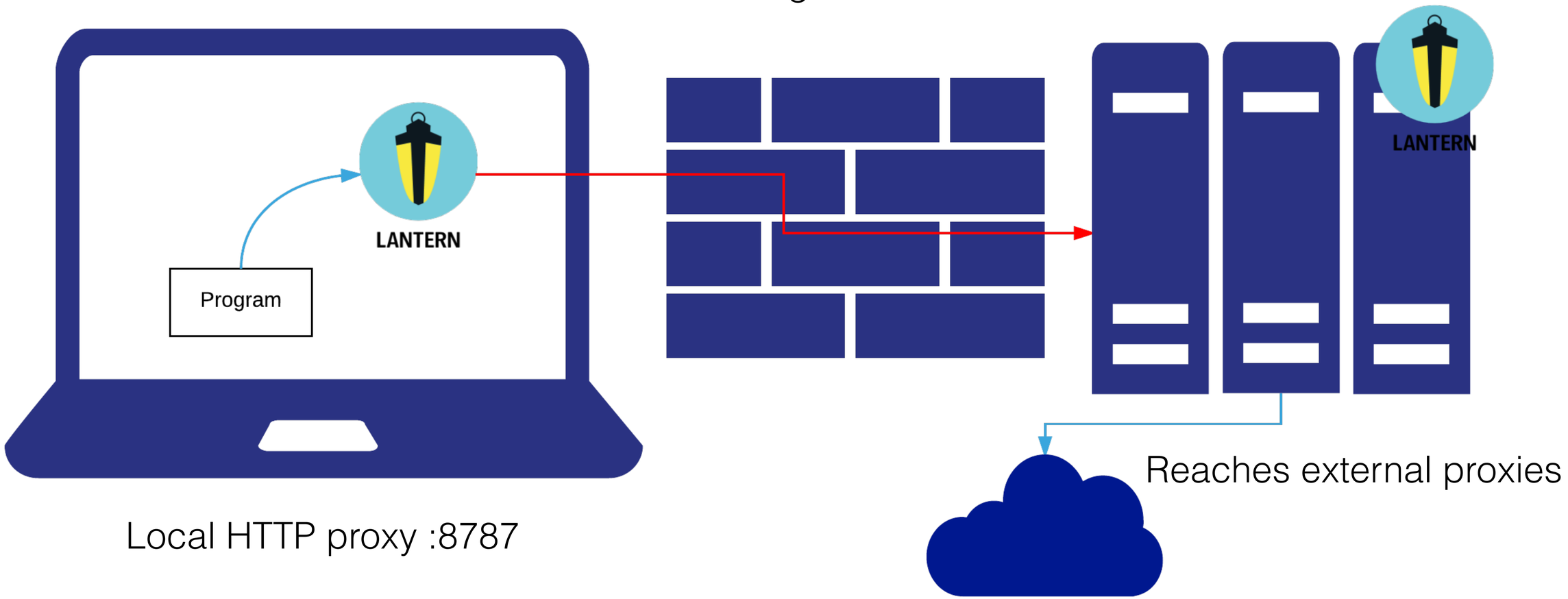
Lantern is an application for Windows, Mac and Linux that allows people to access the Open Internet

https://www.getlantern.org


Site A
Site B
Site C
Censor
Consumer

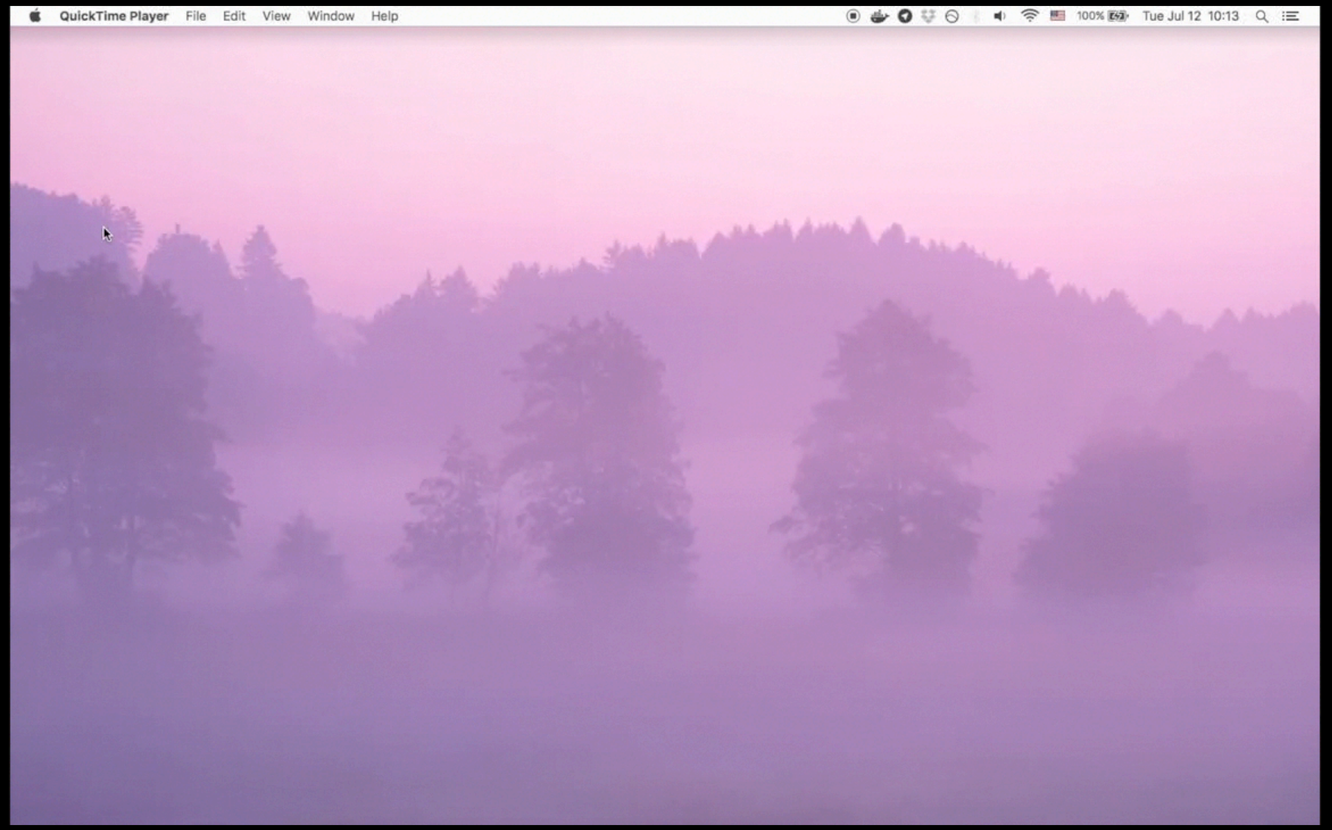

Site A
Site B
Site C
Censor
Consumer

Passes through censor's Firewall



Local HTTP proxy :8787

Reaches external proxies


QuickTime Player File Edit View Window Help
100% Tue Jul 12 10:13

451

Unavailable for legal reasons

200 OK

```
$ curl -x 127.0.0.1:8787 http://www.google.com/humans.txt
```

```
GET http://www.google.com/humans.txt HTTP/1.1
Host: www.google.com
```

```
$ curl -x 127.0.0.1:8787 https://www.google.com/humans.txt
```

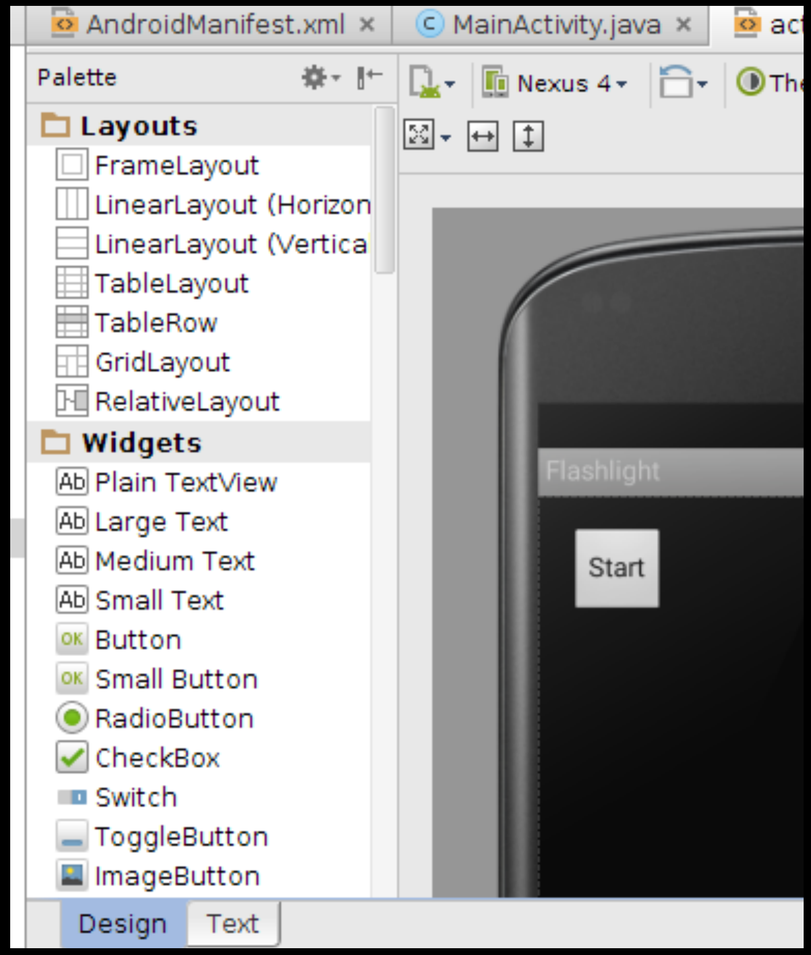
```
CONNECT www.google.com:443 HTTP/1.1
Host: www.google.com:443
```

Lantern and Go

- Go is part of a multilingual ecosystem.
- The Lantern client and proxies are written in Go.
- The auto-update server is also written in Go.
- Lantern for Android is powered by Go Mobile.

Early 2015

- go1.4, required special setup for cross compilation.
- `x/mobile` provided a docker image.
- Fortunately, that was enough to start getting somewhere.



```
$ CGO_ENABLED=1 CC=arm-linux-androideabi-gcc \
> GOOS=android GOARCH=arm GOARM=7 \
> go build \
> -o lantern -tags headless \
> github.com/getlantern/flashlight/main
```

```
$ file lantern
lantern: ELF 32-bit LSB shared object, ARM, EABI5
version 1 (SYSV), dynamically linked, interpreter /
system/bin/linker, not stripped
```

```
$ adb push lantern /data/local/tmp/lantern
$ adb shell
```

```
$ cd /data/local/tmp
$ chmod 755 lantern
$ ./lantern -configdir . -addr 0.0.0.0:8787
```

```
Jul 04 21:57:50.650 - 0m3s DEBUG flashlight.client: client.go:179
Jul 04 21:57:50.651 - 0m3s DEBUG flashlight.client: balancer.go:3
Jul 04 21:57:50.653 - 0m3s DEBUG flashlight.client: balancer.go:4
443
Jul 04 21:57:50.654 - 0m3s DEBUG flashlight.proxiedsites: proxied
Jul 04 21:57:50.681 - 0m3s DEBUG fronted: cache.go:60 Cache close
Jul 04 21:57:50.682 - 0m3s DEBUG flashlight.client: balancer.go:5
Jul 04 21:57:50.778 - 0m3s DEBUG proxiedsites: proxiedsites.go:11
Jul 04 21:57:50.779 - 0m3s DEBUG flashlight: flashlight.go:113 Ap
```

```
$ curl -x 10.0.0.45:8881 https://www.google.com/humans.txt
```

FireTweet

Powered by Lantern

FireTweet

- Forked from an Open Source client (Twidere).
- We built Lantern as a library and embedded it into FireTweet.
- We made FireTweet initialize and use the Lantern client as an HTTP proxy.

```go
package flashlight

func Start(addr string) error {
    // ...
}

func Stop() error {
    // ...
}
```

```
import org.lantern.flashlight;

private static void startProxy()
{
    flashlight.Start(LANTERN_ADDR + ":" + LANTERN_PORT)

    System.setProperty("http.proxyHost", LANTERN_ADDR)
    System.setProperty("http.proxyPort", LANTERN_PORT)

    System.setProperty("https.proxyHost", LANTERN_ADDR)
    System.setProperty("https.proxyPort", LANTERN_PORT)
}
```

Mid 2015

- Hana Kim gave a talk on the `gomobile` command.
- go1.5 didn't require manual bootstrapping anymore.
- Everything got easier!

The `gomobile` command

```
$ go get golang.org/x/mobile/cmd/gomobile
```

The `gomobile` command

- Initialize a local development environment with `gomobile init`.
- Create libraries to be used from Java or Objective C with `gomobile bind`.
- Compile and package standalone apps (games?) with `gomobile build`.

Back to FireTweet

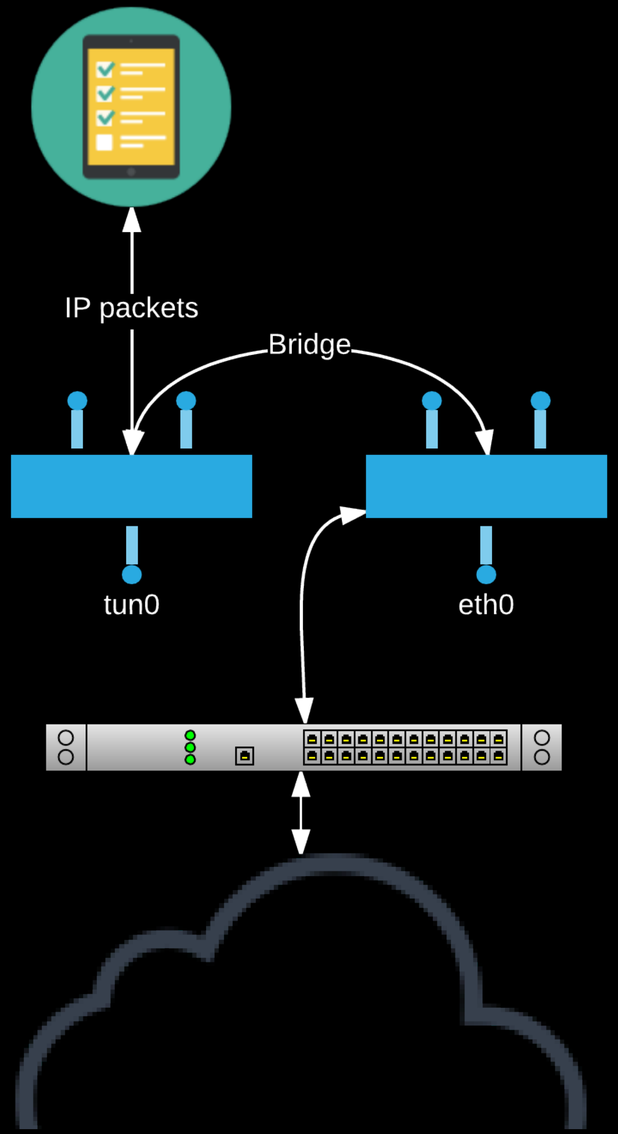
- We wanted to proxy the whole device.
- How was Lantern going to proxy traffic from all the other apps?

Time for a different approach!

Android's `VpnService` API

- The `VpnService` API creates a virtual network interface (TUN device) and sets up routing rules.
- A TUN device can intercept/inject IP packets from/into an user space application.

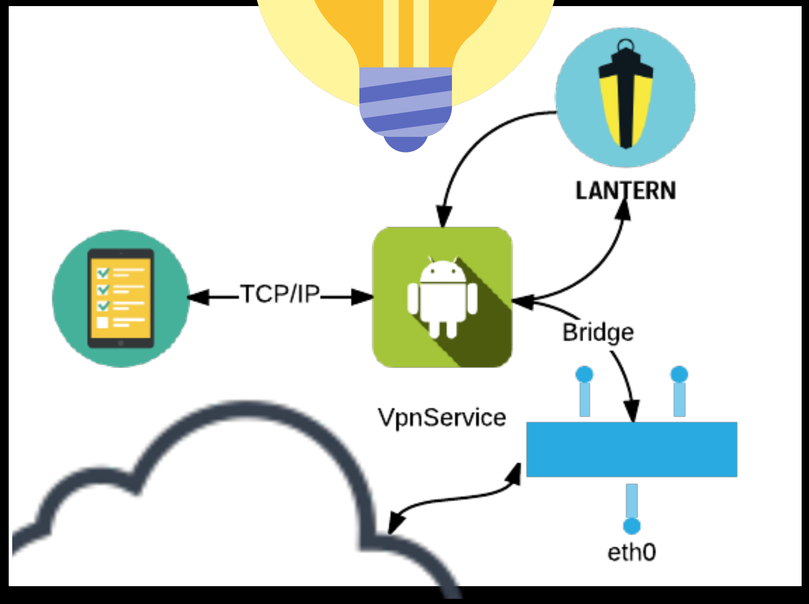
https://developer.android.com/reference/android/net/VpnService.html

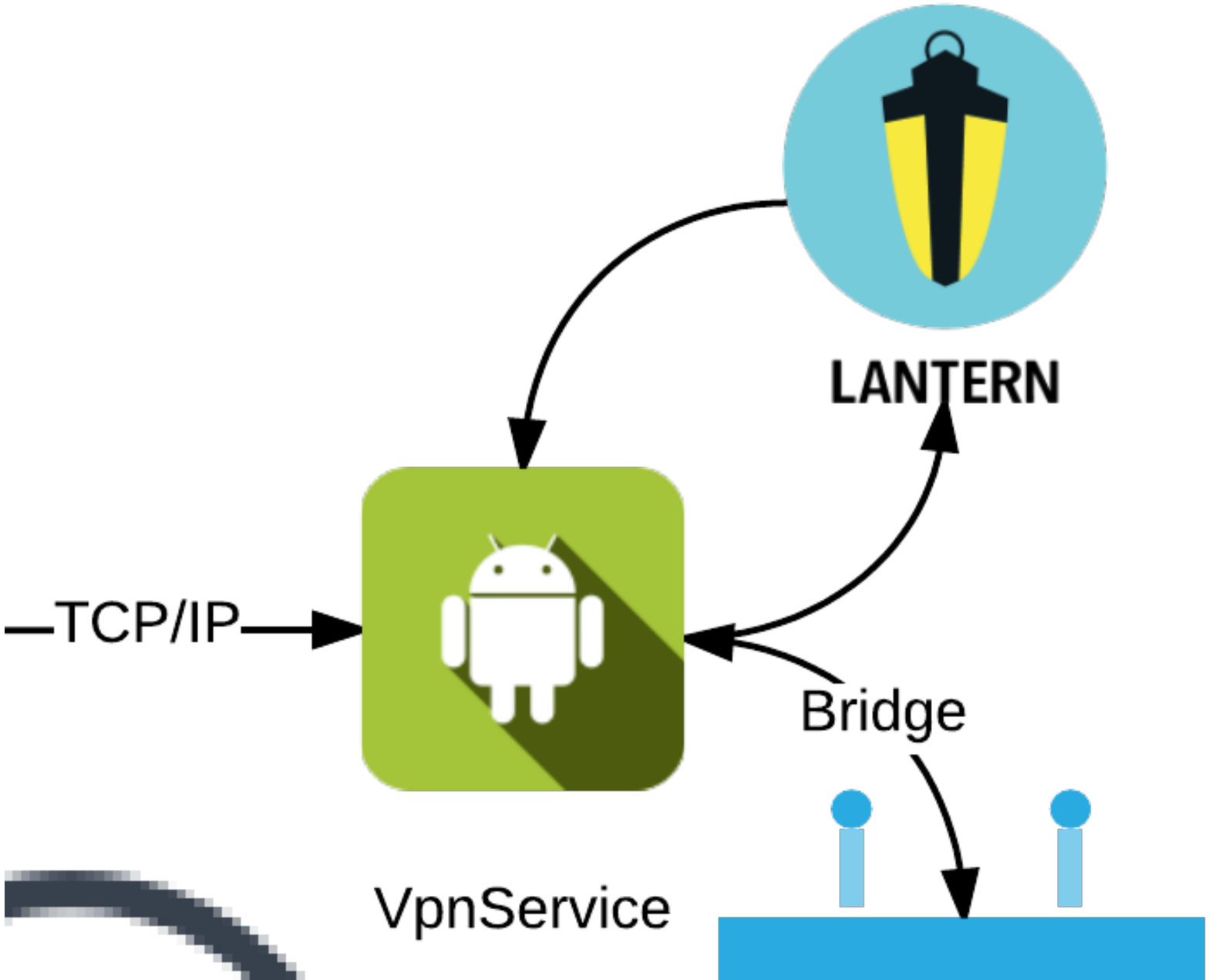


VpnService

```
public class VpnService
extends Service
```

java.lang.Object
↳ android.content.Context
↳ android.content.ContextWrapper
↳ android.app.Service
↳ VpnService



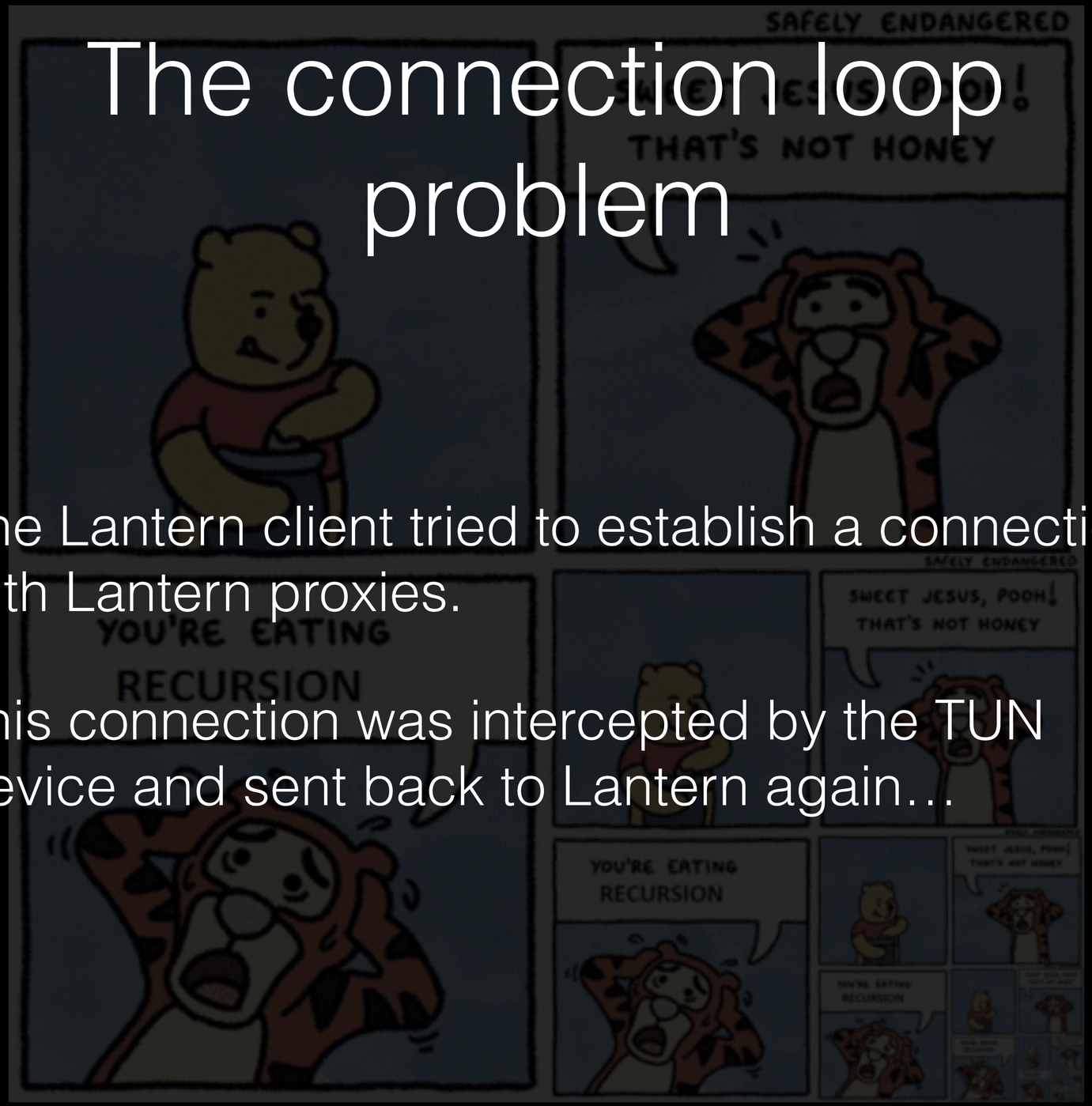

LANTERN
TCP/IP
Bridge
VpnService

How do we make Lantern work with `VpnService`?

tun2socks

- Written in C.
- Creates a tunnel between a TUN device and a SOCKS server.
- Provides support for UDP packets (DNS queries, etc.)
- Old and stable solution already used by Orbot and others.

The connection loop problem



- The Lantern client tried to establish a connection with Lantern proxies.
- This connection was intercepted by the TUN device and sent back to Lantern again…

Protecting outbound connections

```
boolean protect(int socket)

// Protect a socket from VPN connections. After protecting, data
// sent through this socket will go directly to the underlying
// network, so its traffic will not be forwarded through the VPN
```

https://developer.android.com/reference/android/net/VpnService.html#protect(java.net.Socket)

Protecting outbound connections

```
type SocketProtector interface {
    Protect(socket int) error
}

func UseProtector(protector SocketProtector) {
    // ...
}
```

Protecting outbound connections

```
// ...
Lantern.UseProtector(
  new Lantern.Protector() {
    public void Protect(long socket) throws Exception {
      if (!VpnService.protect((int)socket)) {
        throw new Exception("could not protect socket");
      }
    }
  }
);
// ...
```

Protecting outbound connections

```
func (p *Protector) dial(network, addr string) (net.Conn, error) {
    // ...
    socketFd, err := syscall.Socket(syscall.AF_INET,
    syscall.SOCK_STREAM, 0)
    if err != nil {
        return nil, errors.New("Could not create socket: %v", err)
    }
    conn.socketFd = socketFd

    // ...
    return nil, p.Protect(conn.socketFd)
}
```

Meet Lantern for Android


10:58
Google

Is Go Mobile ready for production?

The good parts

- ~~We <3 Go.~~ Yeah, but this isn't a valid reason.
- Share common logic.
- Concurrency and network programming.
- Go code can be tested easily apart from Java or Objective C.
- You can provide binary SDKs for third parties.
- Great stdlib!

The bad parts?

- According to the Go Mobile developers, crossing language boundaries between Go and another one has a performance overhead.
- No native UI (yet).
- Android is fine, but iOS still needs some work.
- Some parts of the standard library may not be available or may behave differently.

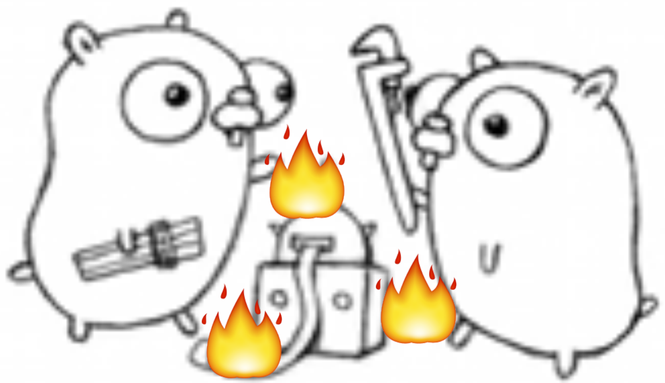
Summary

- `gomobile` is a wrapper around Go that makes it easy to compile for mobile architectures.
- Go Mobile is experimental, but it already can be used to get things done.
- You can use Go to provide services to other programs (a long-running server, a REST API wrapper, an embedded database, etc).
- Don't be afraid to experiment!

Our case

- 1.7 M monthly users.
- 4.5 M all time unique users.
- 258k daily active users.

⚠ CAUTION



Gophers
are
doing
experiments

Try Go Mobile!

- **Wiki**: golang.org/wiki/Mobile
- **Github Repository**: github.com/golang/mobile

¡Gracias!

¡Fiesta!

¡Órale!

Mexican Gopher by @francesc

https://twitter.com/francesc/status/586212072700125184