@amirmc

Unikernels!

Amir Chaudhry

… on behalf of a merry crew: Anil Madhavapeddy, Thomas Gazagnaire, David Scott, Thomas Leonard, Richard Mortier, Magnus Skjegstad, David Sheets, Balraj Singh, Jon Crowcroft, Mindy Preston, *and many others!*

PolyConf
July 2015

UNIVERSITY OF CAMBRIDGE

Overview

Intro

Unikernels

- What they are
- Why they matter
- Example scenario (static websites)

Unikernels

Deployments / Other work

Why I care

Overview

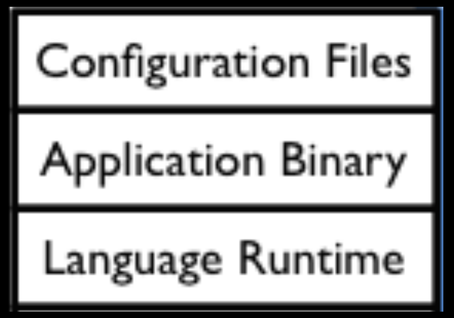
PM in OCaml Labs (I herd cats)

Community for MirageOS (moar cats)

I like systems stuff!

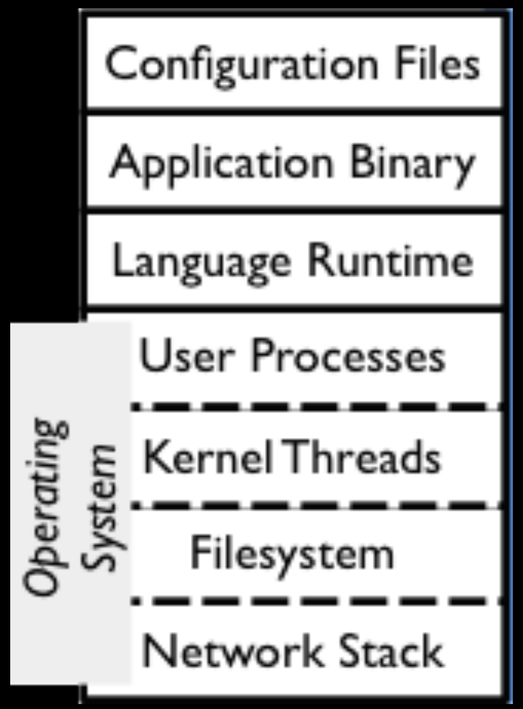
Background: Physicist, Neuroscientist, CompSci (ish), Startups, BigCo.

Software today…



…is an **application** …

Software today…



…is an **application** …

… on top of an **Operating System**.

Software today…

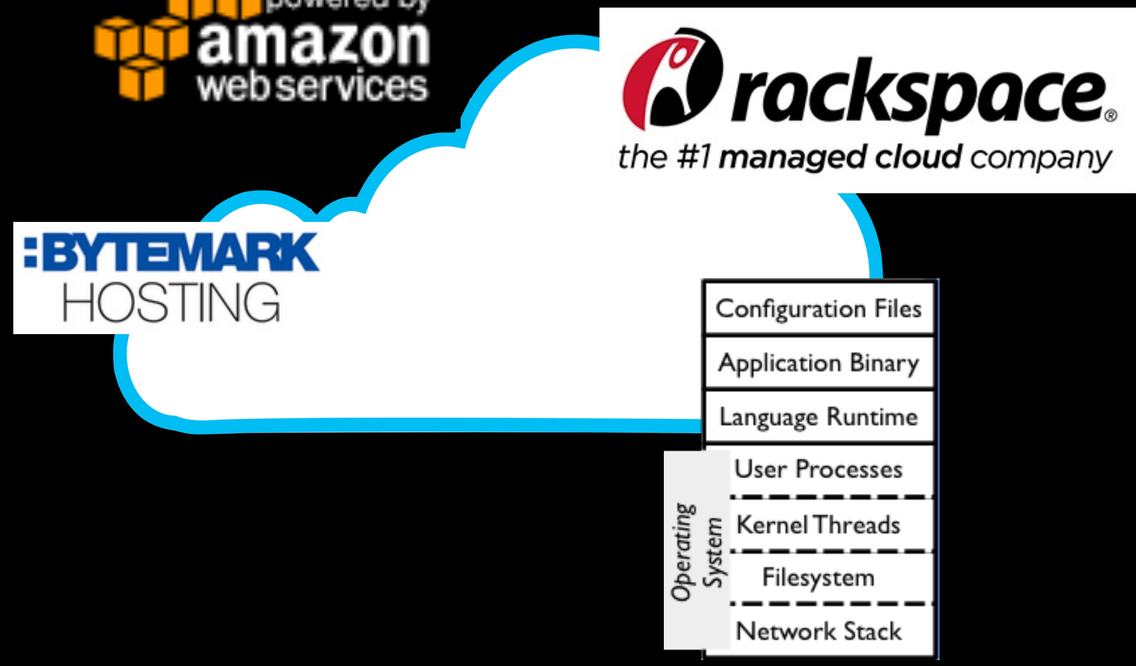
… is **built locally**…

@amirmc

Software today…

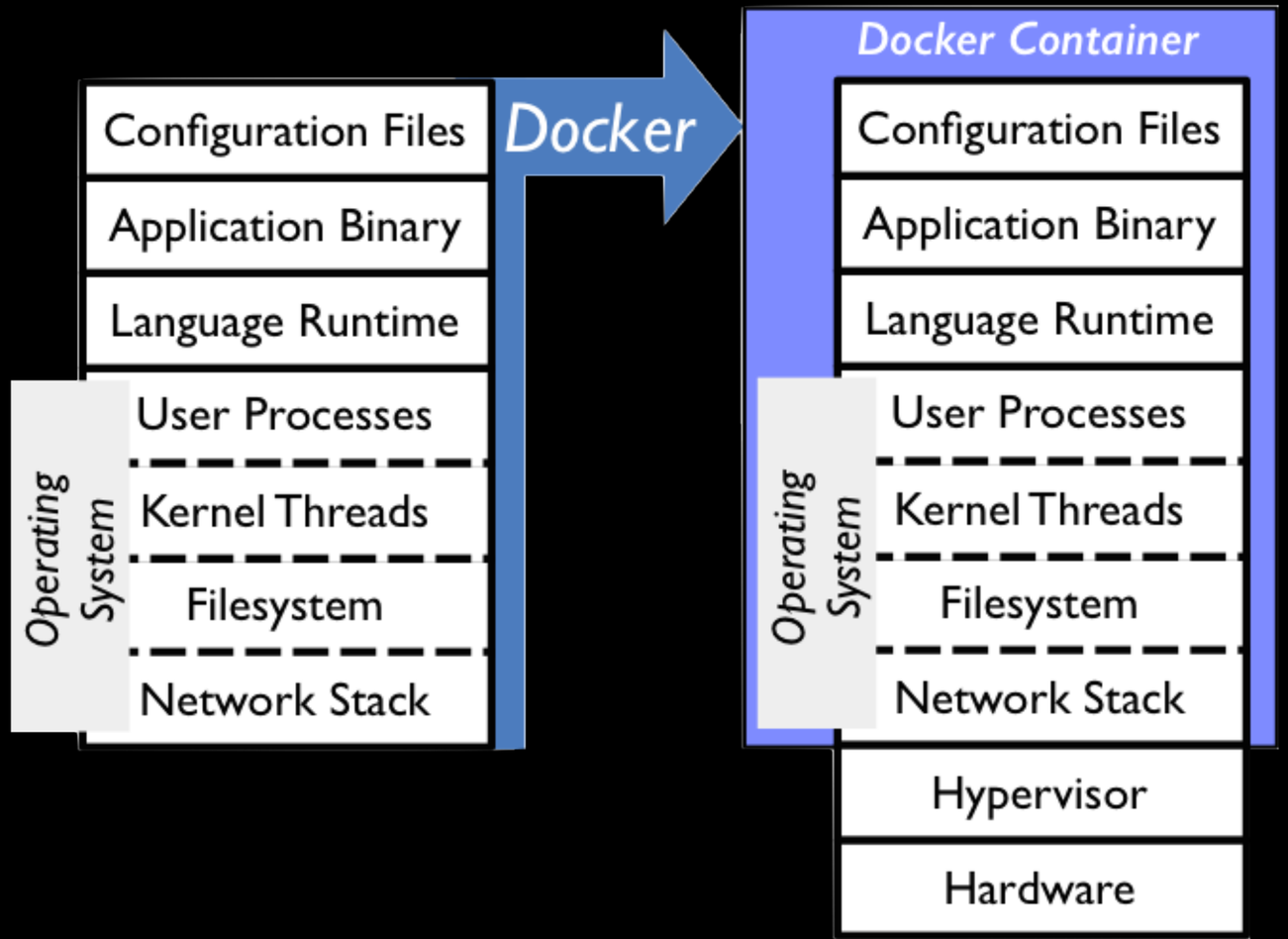
… is **built locally**…

… but **deployed remotely**.



Software today…

… needing **more tools**.

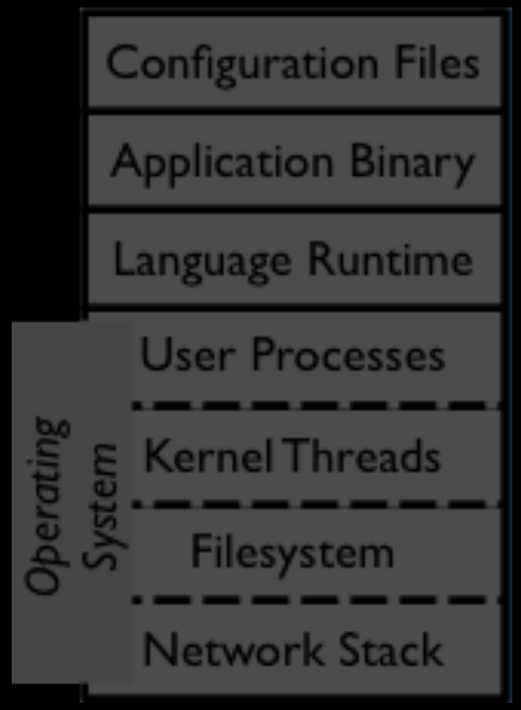


Software today…

…**is complex!**

Even though most apps are single-purpose

Complexity is the enemy…



More layers -> tricky **config**

Duplication -> **inefficiency**

Large sizes -> **long boot times**

More stuff -> **larger attack surface**

Why build for **clouds**
as we do for **desktops**?

Can we do **better**?

Can we do **better**?

Disentangle applications from the OS

Break up OS functionality into modular libraries

Link only the system functionality your app needs

Target alternative platforms from a single codebase

The Rise of the Unikernel

Unikernels are **specialised** virtual machine images built from a **modular** stack adding system libraries and configuration to application code

Every application is compiled into its own specialised OS that runs on the cloud or embedded devices

MirageOS

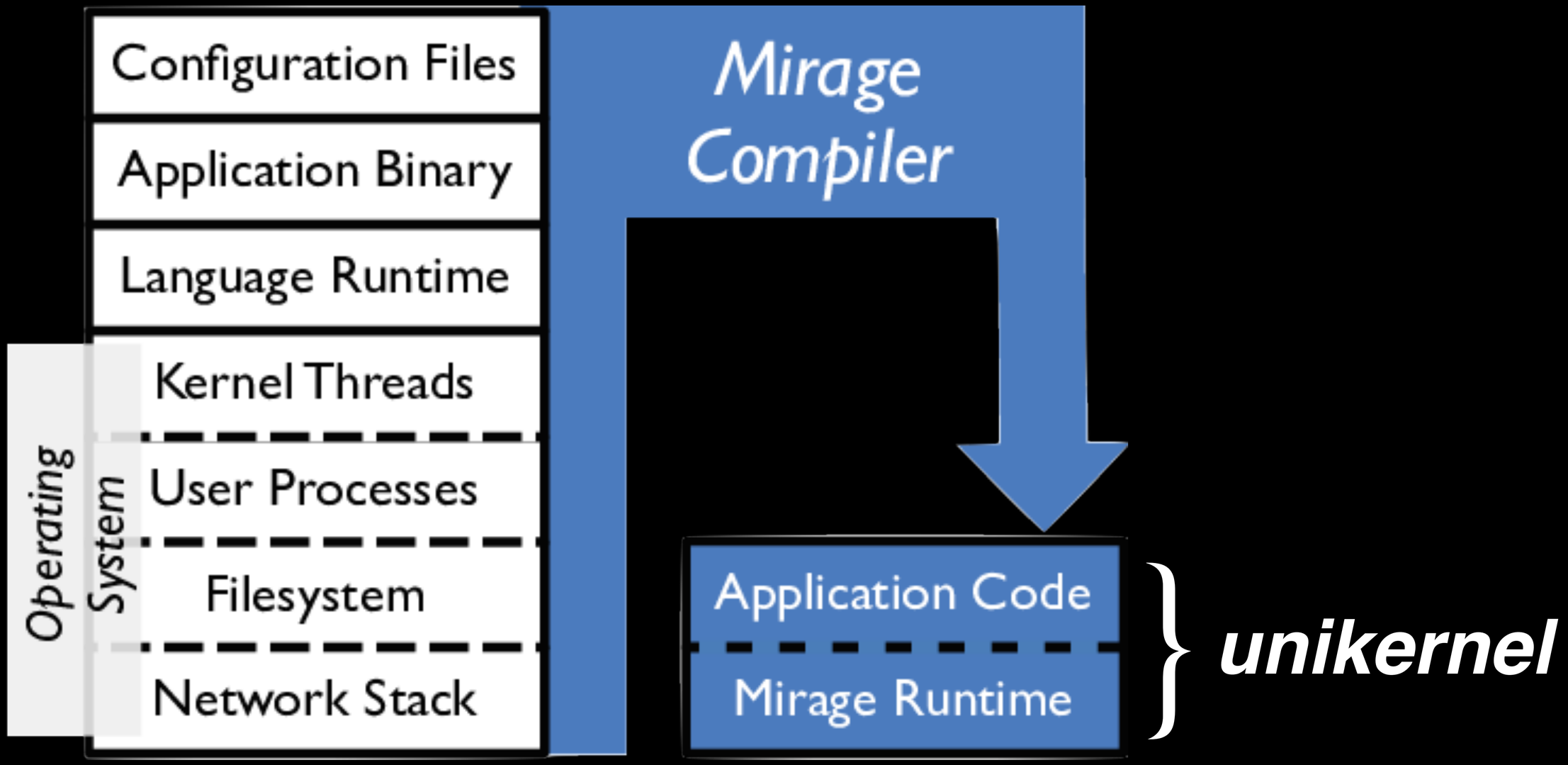
@amirmc

MirageOS

OCaml

@amirmc

MirageOS



MirageOS

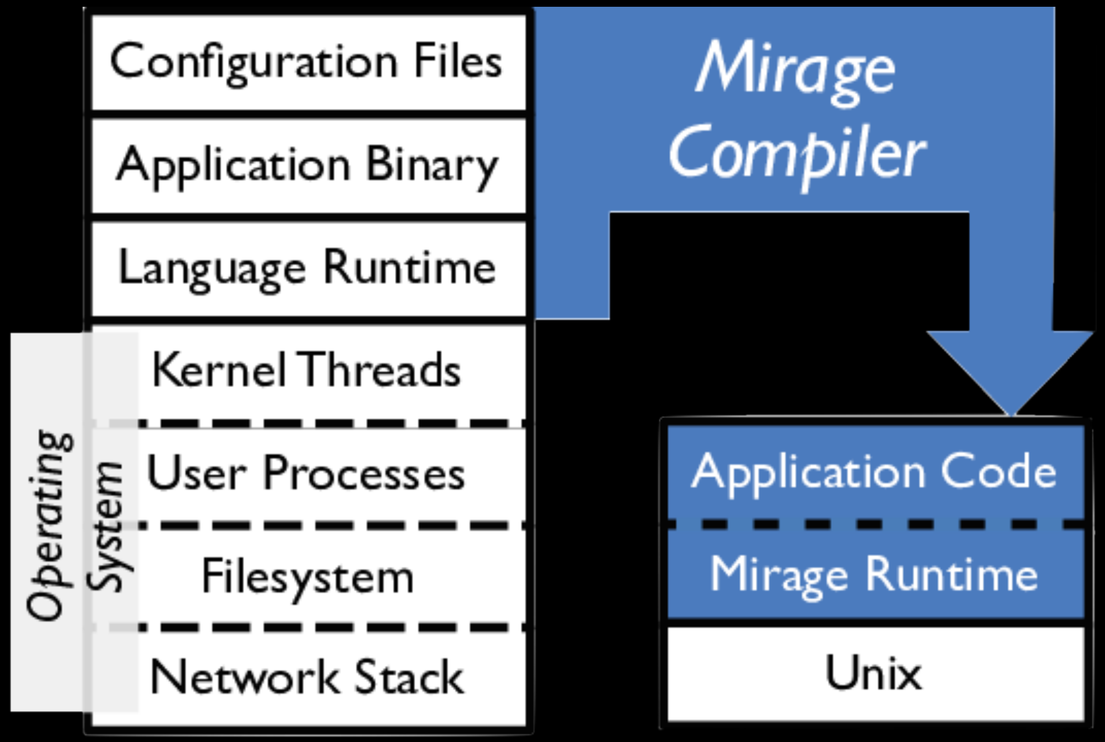
Familiar development cycle

Broad deployment scenarios

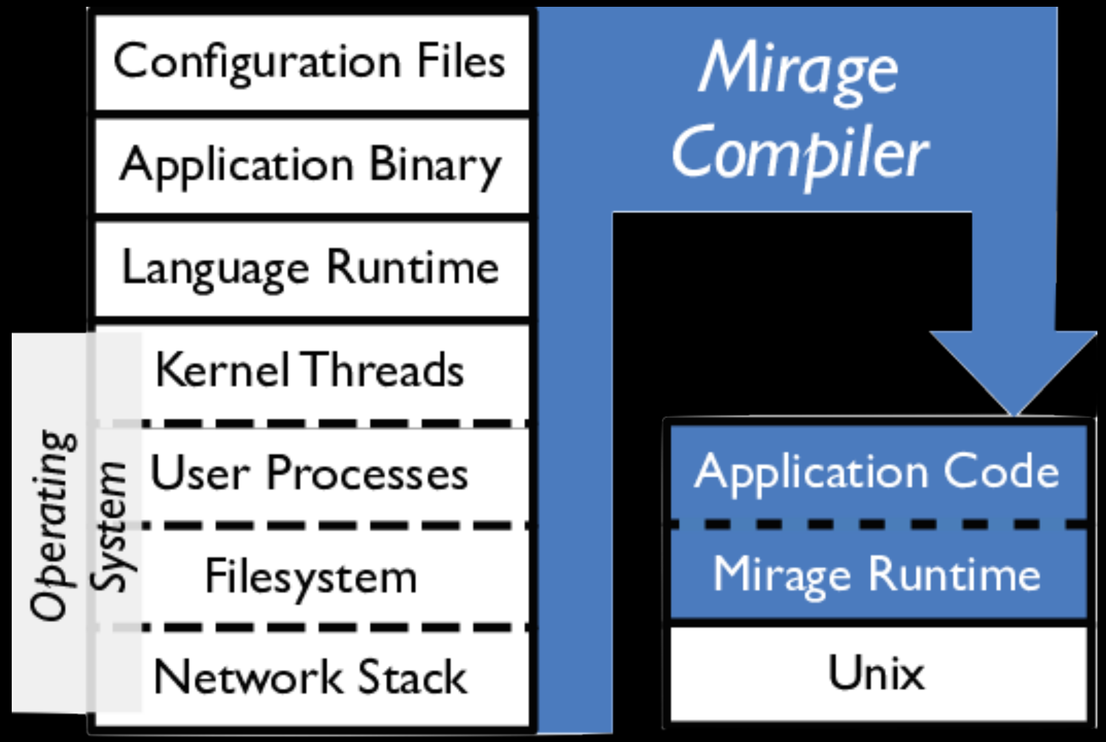
@amirmc

MirageOS

Unix



Develop logic

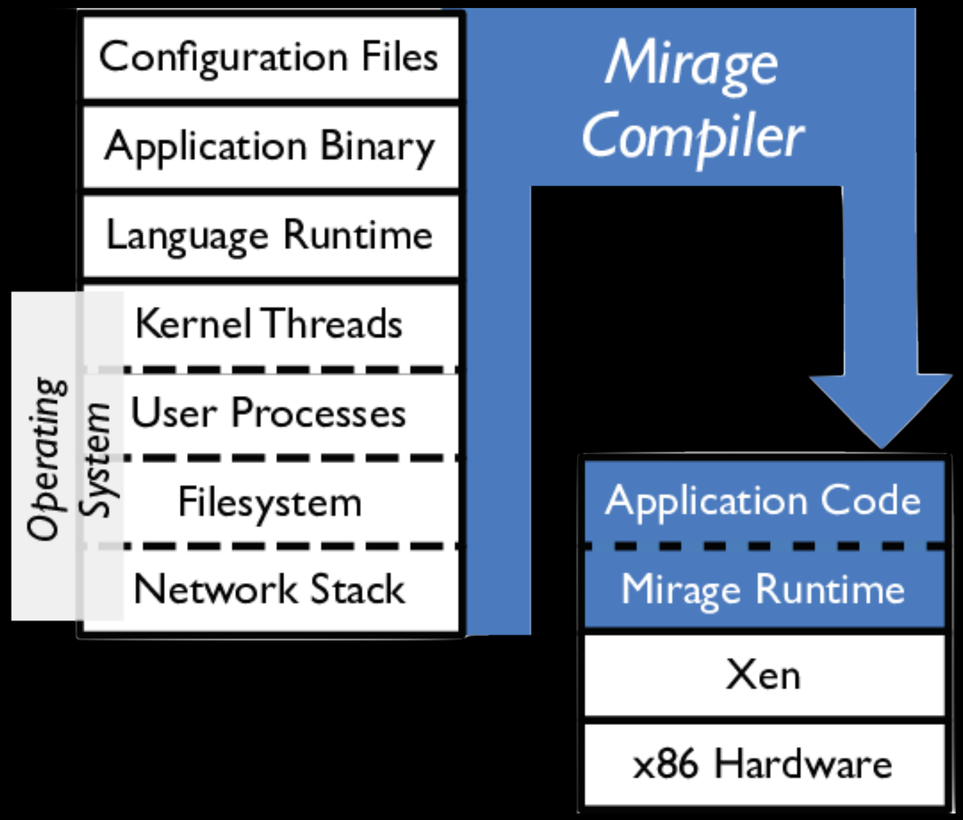


MirageOS System Libs

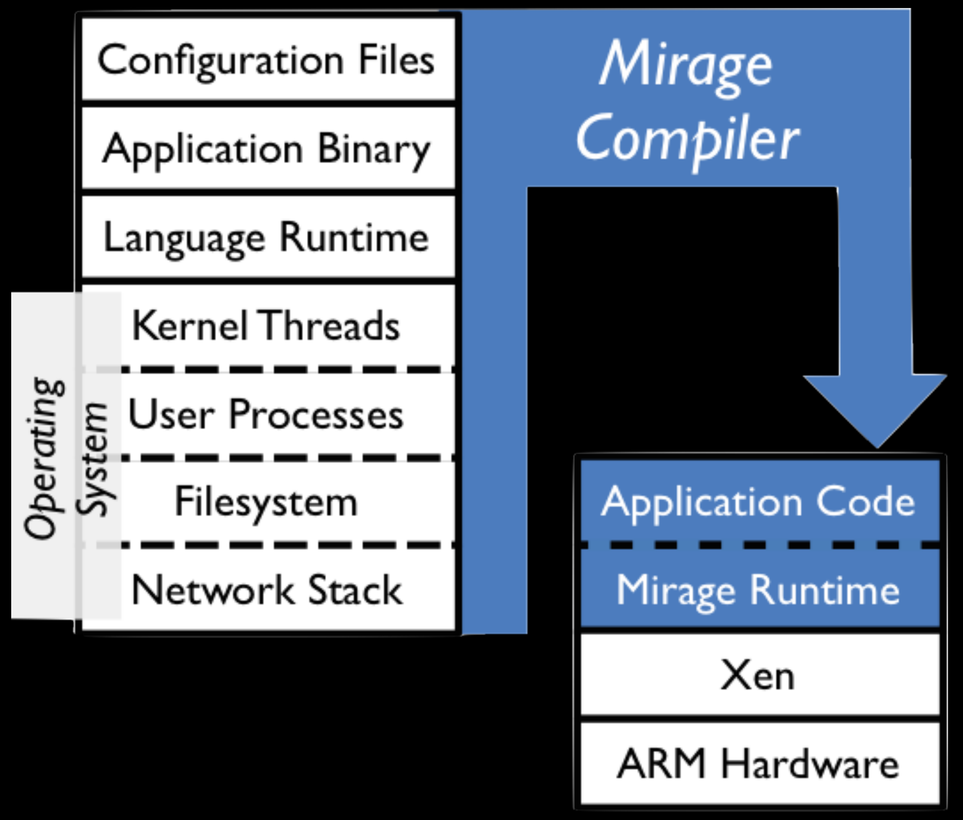
@amirmc

MirageOS

Xen



Specialise for deploy…



… to multiple environments

So what?

Example: Static websites

(though applicable to any application)

Unikernels at PolyConf! | A

amirchaudhry.com/unikernels-polyconf-2015/

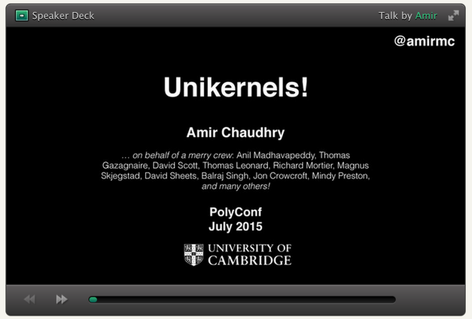
Amir Chaudhry

thoughts, comments & general ramblings

home journal free/busy about

4 July 2015

Unikernels at PolyConf!



Above are my slides from a talk at PolyConf this year. I was originally going to talk about the MISO tool stack and personal clouds (i.e. how we'll build towards Nymote) but after some informal conversations with other

Lead at Nymote.org
Wrangler at OCaml Labs
Community at Mirage OS
Post-doc at Computer Lab
Member of Darwin College
Cofounder of Springboard
Advisor to CUEntrepreneurs

Search

Filed under

irmin
mirage
ocaml
ocamllabs

MirageOS

https://mirage.io

MIRAGE OS Blog Docs API Changes Community

A programming framework for building type-safe, modular systems

MirageOS is a library operating system that constructs unikernels for secure, high-performance network applications across a variety of cloud computing and mobile platforms. Code can be developed on a normal OS such as Linux or MacOS X, and then compiled into a fully-standalone, specialised unikernel that runs under the Xen hypervisor.

Since Xen powers most public cloud computing infrastructure such as Amazon EC2 or Rackspace, this lets your servers run more cheaply, securely and with finer control than with a full software stack.

MirageOS uses the OCaml language, with libraries that provide networking, storage and concurrency support that work under Unix during development, but become operating system drivers when being compiled for production deployment. The framework is fully event-driven, with no support for preemptive threading.

MirageOS 1.0 was released in December 2013, followed by MirageOS 2.0 in July 2014. All the infrastructure you see here is self-hosted. Check out the documentation, compile your hello world unikernel, get started with the public cloud, watch the talks, or see the slides.

Recent Updates (all)

- Reviewing the Bitcoin Pinata (29 Jun 2015)
- MirageOS v2.5 with full TLS support (26 Jun 2015)
- Why OCaml-TLS? (26 Jun 2015)
- Unix TLS Tools (26 Jun 2015)
- Breaking changes (18 Jun 2015)
- Weekly Meeting: 2015-06-03 (3 Jun 2015)
- Weekly Meeting: 2015-05-20 (20 May 2015)
- Weekly Meeting: 2015-05-06 (6 May 2015)
- Weekly Meeting: 2015-04-23 (23 Apr 2015)
- Weekly Meeting: 2015-04-09 (9 Apr 2015)
- Weekly Meeting: 2015-03-25 (25 Mar 2015)
- Weekly Meeting: 2015-03-11 (11 Mar 2015)
- Weekly Meeting: 2015-02-25 (25 Feb 2015)

MirageOS is a Xen and Linux Foundation incubator project.

Some Random Idiot

somerandomidiot.com

Some Random Idiot

yet another blog by someone with somethin' to say

Blog Archives Contact Me Find Kitten

Search

APR 24TH, 2015

What a Distributed, Version-Controlled ARP Cache Gets You

`git` (and its distributed version control system friends `hg` and `darcs`) have some great properties. Not only do you get a full history of changes on objects stored in them, you can get comments on changes, as well as branching and merging, which lets you do intermediate changes without messing up state for other entities which want to work with the repository.

That's all pretty cool. I actually want that for some of my data structures, come to think of it. Say, for example, a boring ol' key-value store which can be updated from a few different threads — in this case, a cache that stores values it gets from the network and the querying/timeout code around it. It would be nice if each thread could make a new branch, make its changes, then merge them into the primary branch once it's done.

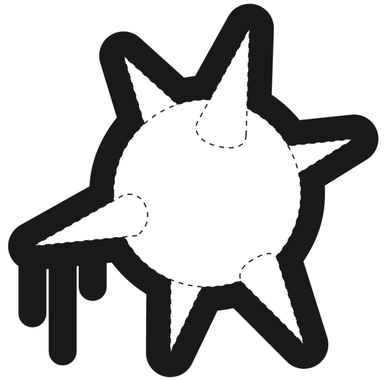
Recent Posts

- What a Distributed, Version-Controlled ARP Cache Gets You
- Let's Play Network Address Translation: The Home Game
- Things Routers Do: Network Address Translation
- Some Random Idiot
- Virtualization: WTF
- OPW FIN
- I Am Unikernel (and So Can You!)
- My Content Is Mine: Why I Unikernel, Part 2
- Attack Surface: Why I Unikernel, Part 1

BTC Piñata

ownme.ipredator.se

You have reached the BTC Piñata.



BTC Piñata knows the private key to the bitcoin address **183XuXTTgnfYfKcHbJ4sZeF46a49Fnihdh**. If you break the Piñata, you get to keep what's inside.

Here are the rules of the game:

- You can connect to port 10000 using TLS. Piñata will send the key and hang up.
- You can connect to port 10001 using TCP. Piñata will immediately close the connection and connect back over TLS to port 40001 on the initiating host, send the key, and hang up.
- You can connect to port 10002 using TCP. Piñata will initiate a TLS handshake over that channel serving as a client, send the key over TLS, and hang up.

And here's the kicker: in both the client and server roles, Piñata requires the other end to present a certificate. Authentication is performed using standard **path validation** with a single certificate as the trust anchor. And no, you can't have the certificate key.

It follows that it should be impossible to successfully establish a TLS connection as long as Piñata is working properly. To get the spoils, you have to smash it.

Before you ask: yes, Piñata will talk to itself and you can enjoy watching it do so.

@amirmc


BTC Piñata
ownme.ipredator.se
You have reached the BTC Piñata.
BTC Piñata knows the private key to the bitcoin address 183XuXTTgnfYfKcHbJ4sZeF46a49Fnihdh. If you break the Piñata, you get to keep what's inside.
Here are the rules of the game:
• You can connect to port 10000 using TLS. Piñata will send the key and hang up.
• You can connect to port 10001 using TCP. Piñata will immediately close the connection and connect back over TLS to port 40001 on the initiating host, send the key, and hang up.
• You can connect to port 10002 using TCP. Piñata will initiate a TLS handshake over that channel serving as a client, send the key over TLS, and hang up.
And here's the kicker: in both the client and server roles, Piñata requires the other end to present a certificate. Authentication is performed using standard path validation with a single certificate as the trust anchor. And no, you can't have the certificate key.
It follows that it should be impossible to successfully establish a TLS connection as long as Piñata is working properly. To get the spoils, you have to smash it.
Before you ask: yes, Piñata will talk to itself and you can enjoy watching it do so.

@amirmc

SMALL!

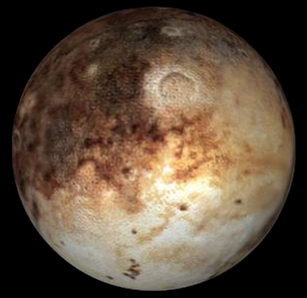
8.2MB
102 kloc

No extra stuff!

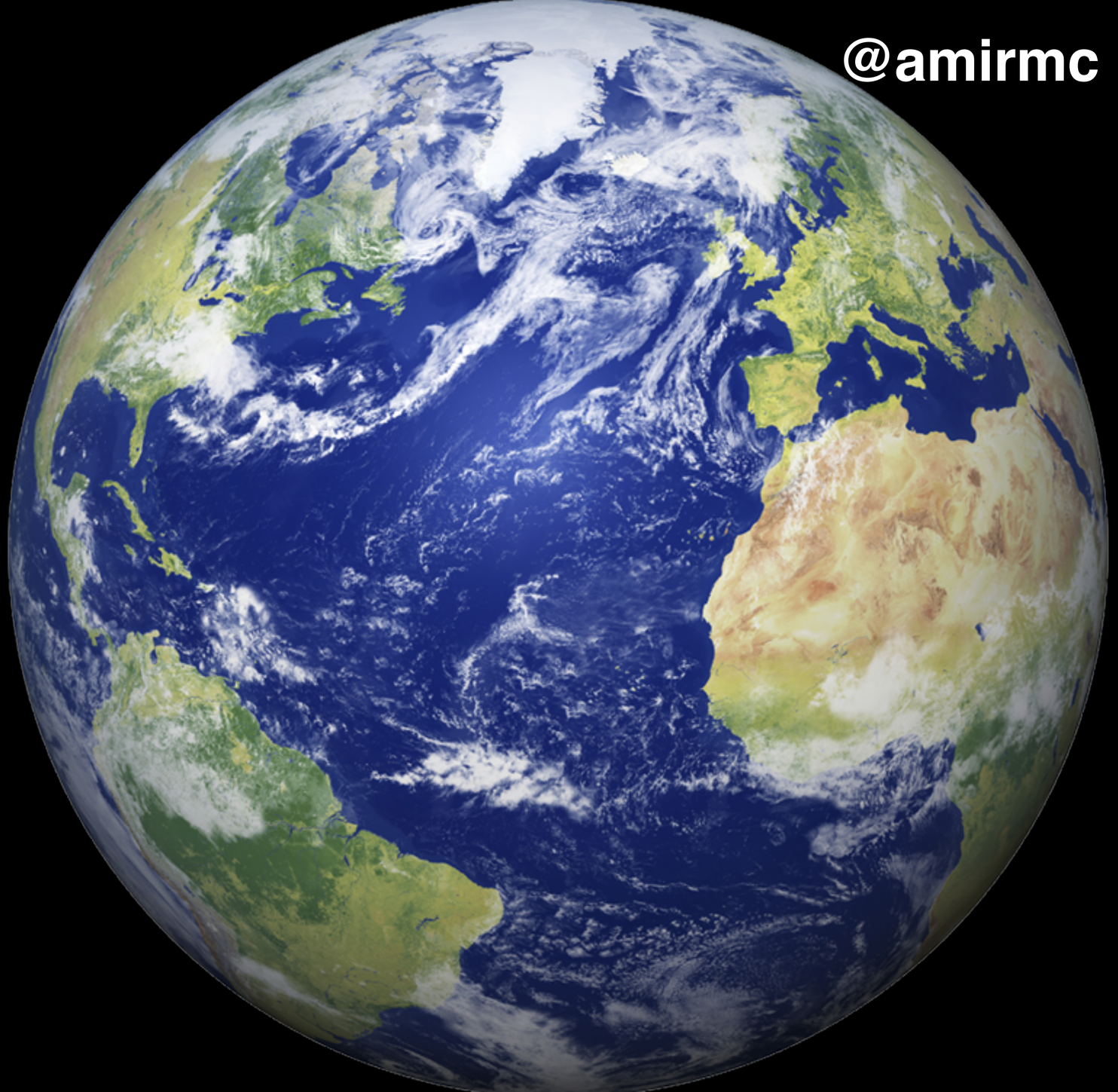
~200MB
2560 kloc

@amirmc

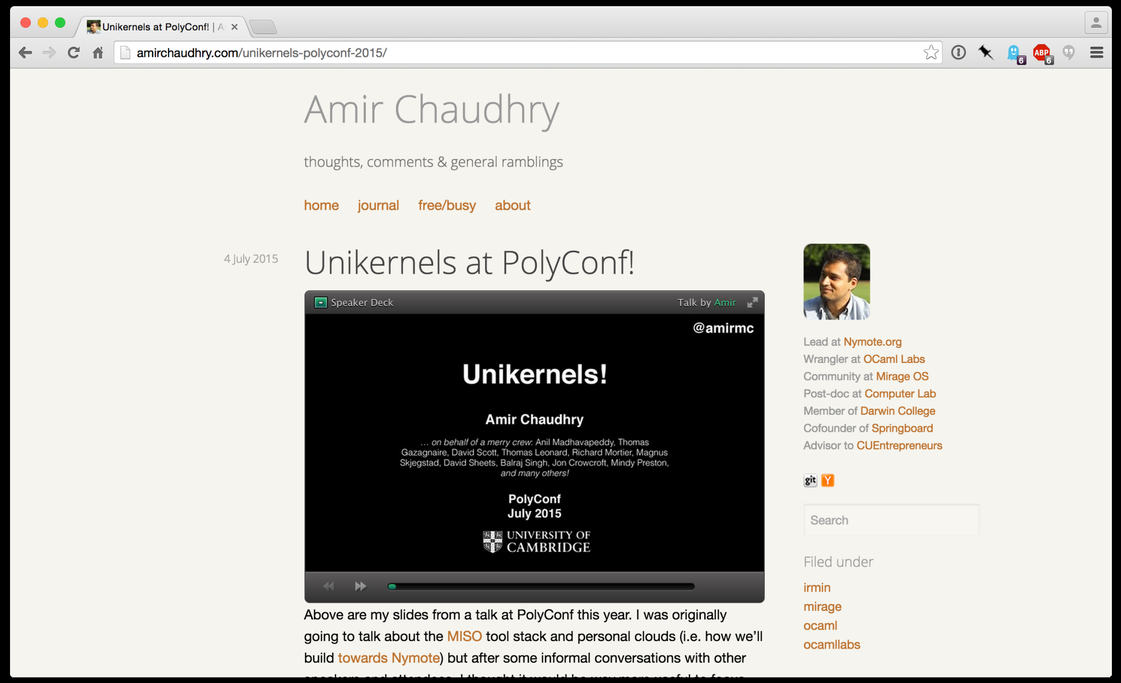
@amirmc

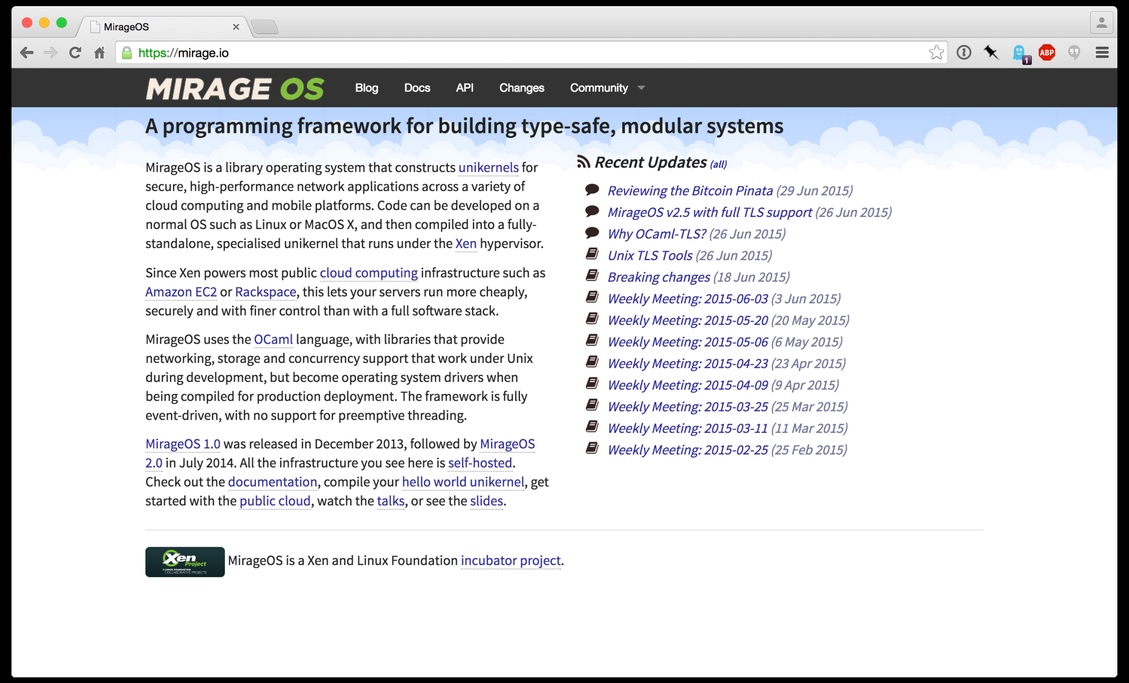


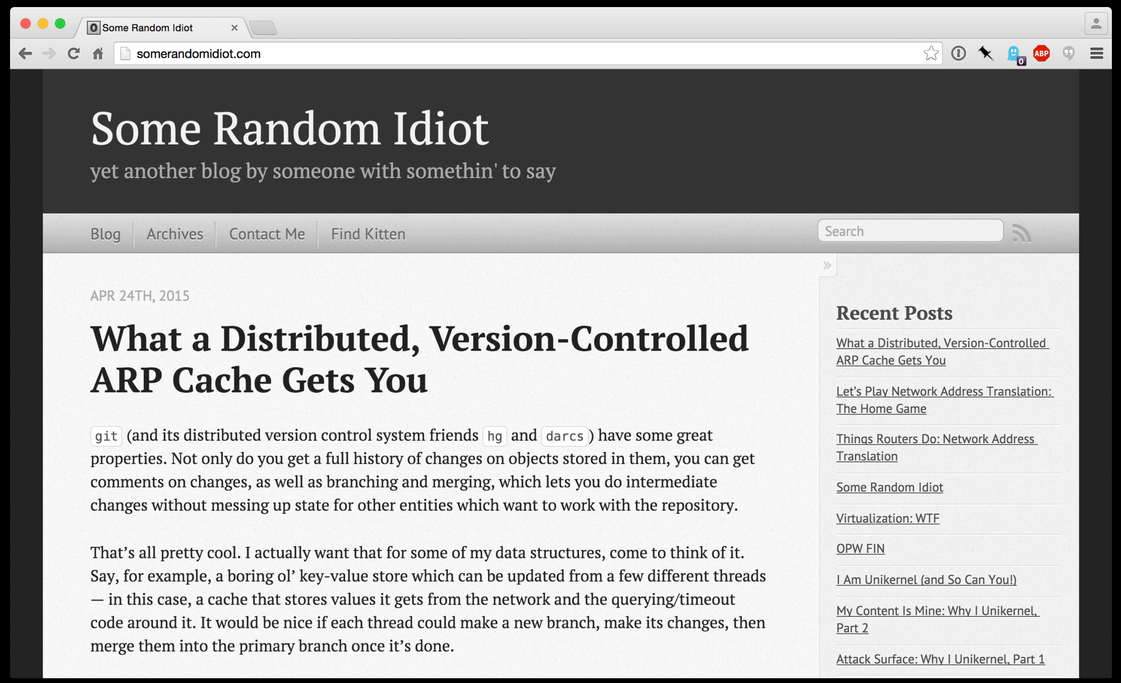
Pluto



just considering areas of the circles :)







Easy deployment

@amirmc

Heroku for Unikernels

…in ~100 lines of code

Walkthrough

Walkthrough

@amirmc

Heroku for Unikernels

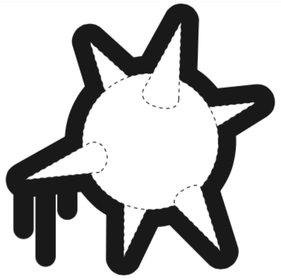
…in ~100 lines of code

General workflow

@amirmc

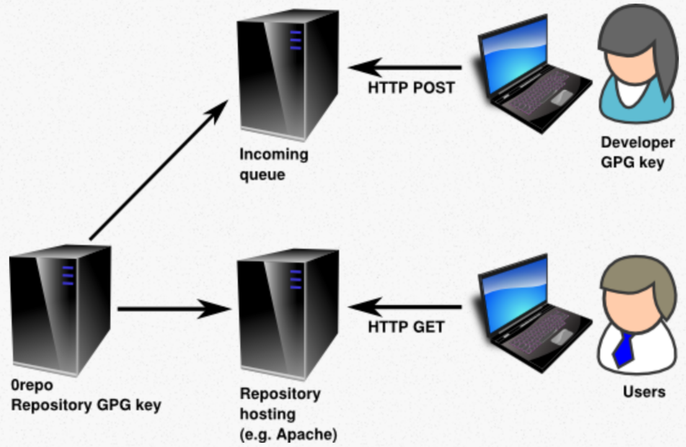
Deployments

You have reached the BTC Piñata.



My First Unikernel

I wanted to make a simple REST service for queuing file uploads, deployable as a virtual machine. The traditional way to do this is to download a Linux cloud image, install the software inside it, and deploy that. Instead I decided to try a unikernel.



Magnus Skjegstad

Blog | Publications | Software | About

WED 25 MARCH 2015

A unikernel experiment: A VM for every URL

I recently wrote a DNS server that can boot unikernels on demand called Jitsu. The following diagram shows a simplified version of how Jitsu works. The client sends a DNS query to a DNS server (Jitsu). The DNS server starts a unikernel and sends a DNS response back to the client while the unikernel is booting. When the client receives the DNS response it opens a TCP connection to the unikernel, which now has completed booting and is ready to respond to the TCP connection.

Thomas Leonard's blog

Blog | Archives | About Me

2015-04-28

CueKeeper: Gitting Things Done in the Browser

Git repositories store data with history, supporting replication, merging and revocation. The Irmin library lets applications use Git-style storage for their data. To try it out, I've written a GTD-based action tracker that runs entirely client-side in the browser.

Process | Work | Contact | Schedule | Review | Add or search

Job | Personal | Export | Show history | Close all

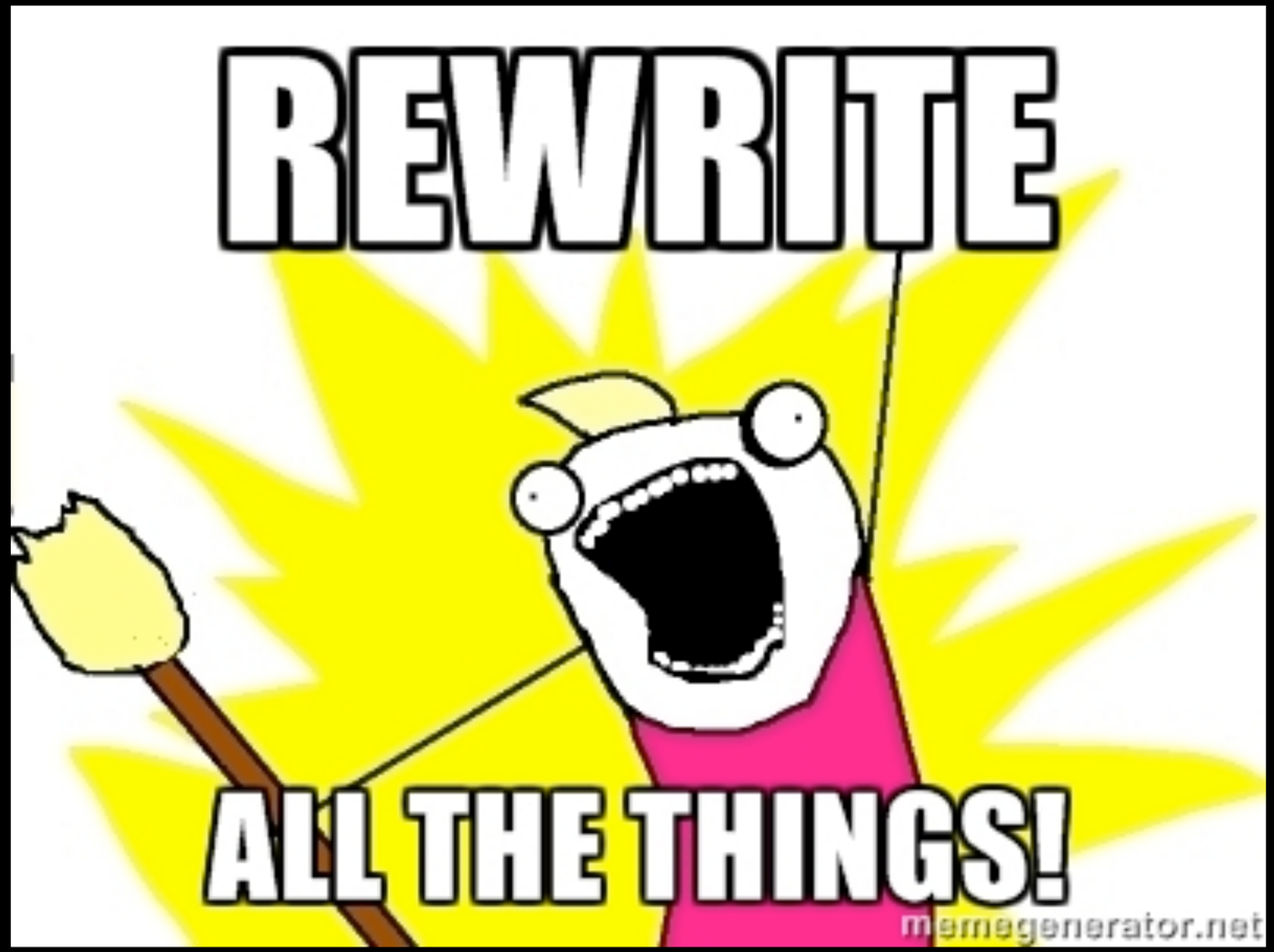
Next actions +
Email +
Make a Mirage unikernel +
Subscribe to Mirage list ✗
Reading +
Learn OCaml +
Read "Real World OCaml" ✗
Try OCaml tutorials ✗
Make a Mirage unikernel +
Follow Mirage tutorial ✗
Read "My First Unikernel" ✗
Start using CueKeeper +
Read wikipedia page on GTD ✗

Recently completed
~~Learn to use Git~~ ✗

Make a Mirage unikernel
A project in Hobbies (show)
Contact: (no contact)
Next actions +
Follow Mirage tutorial ✗
Read "My First Unikernel" ✗
Subscribe to Mirage list ✗
+sub-project +action
(add log entry) (edit)
Created 2015-04-20 19:40 (Mon) (delete)

Follow Mirage tutorial
An action in Make a Mirage unikernel (show)
Context: Reading (show)
Contact: (no contact)
Repeats: (never)
The official tutorial works on Linux or OS X: http://openmirage.org/wiki/install
(add log entry) (edit)
Created 2015-04-20 19:40 (Mon) (delete)

Trade-off (for now)



Why I care

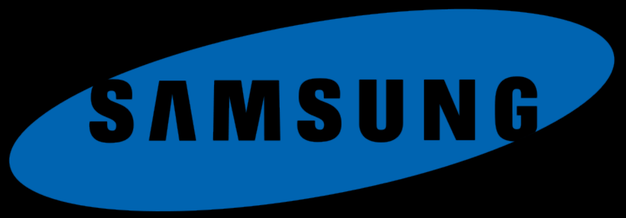
Empower individuals

Distributed personal clouds

Resilient, scalable systems


Google

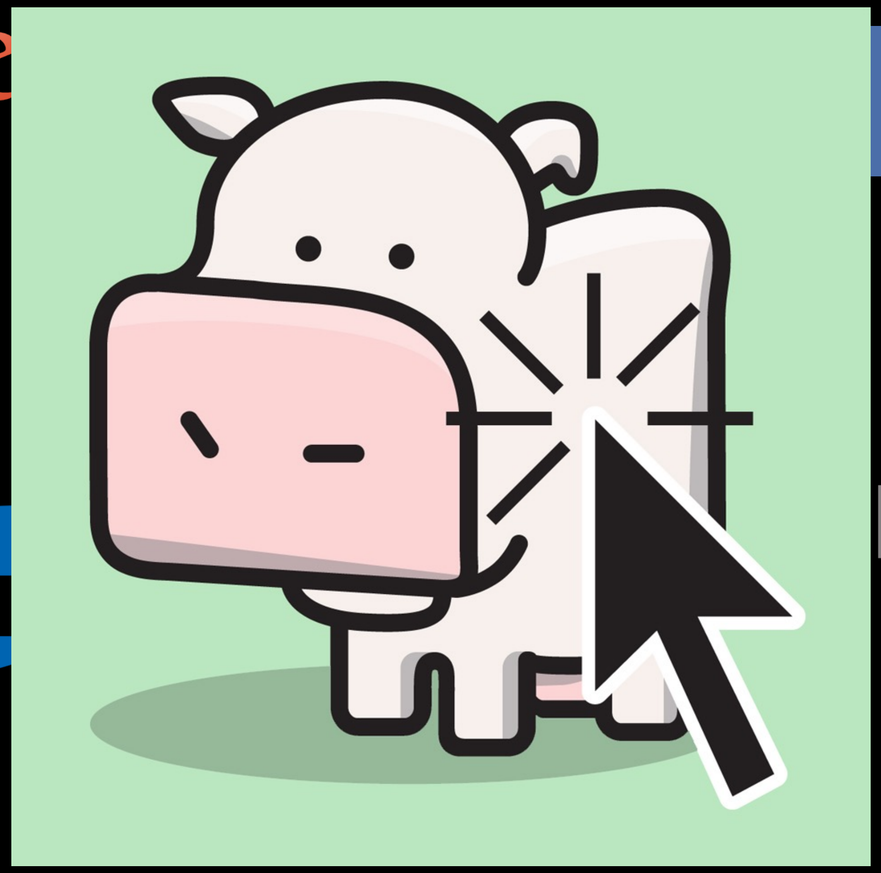

facebook


SAMSUNG


Microsoft

~~Cloud~~ *Feudal* Computing

Google



facebook

SAMSUNG

Microsoft

@amirmc


MirageOS (OS/application)


Irmin (Storage/Sync)

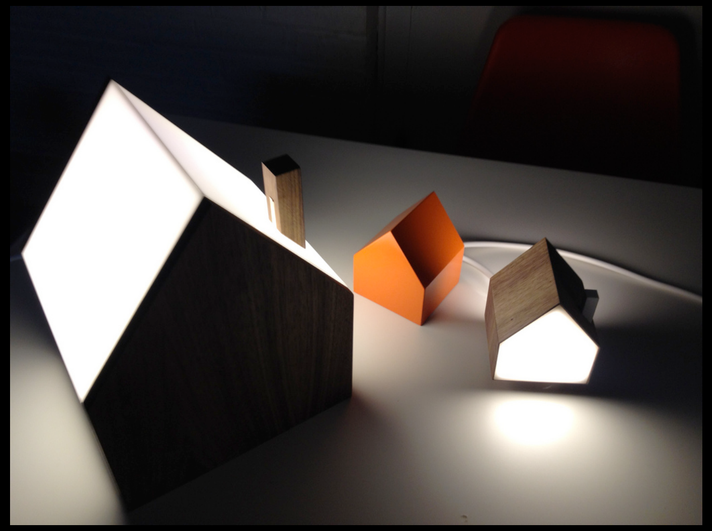

Signpost (Identity/Connectivity)


OCaml (Safety/Modularity)


Mail


Contacts


Calendar




nest
18
IN 20 MIN
22


73

Marco Arment
@marcoarment

Follow

Just had to accept a new license agreement to continue using my thermostat.

Yeah, Nest is awesome, not creepy at all since the Google buy…

RETWEETS 96 FAVORITES 101

5:17 AM - 3 Jul 2015

Contribute!

https://mirage.io

http://nymote.org

http://ocaml.org