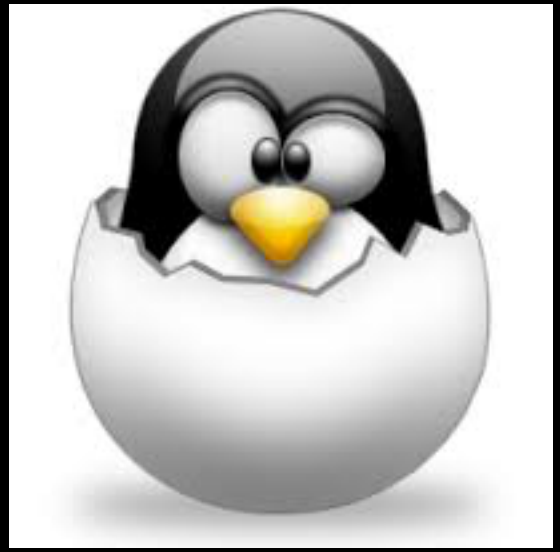


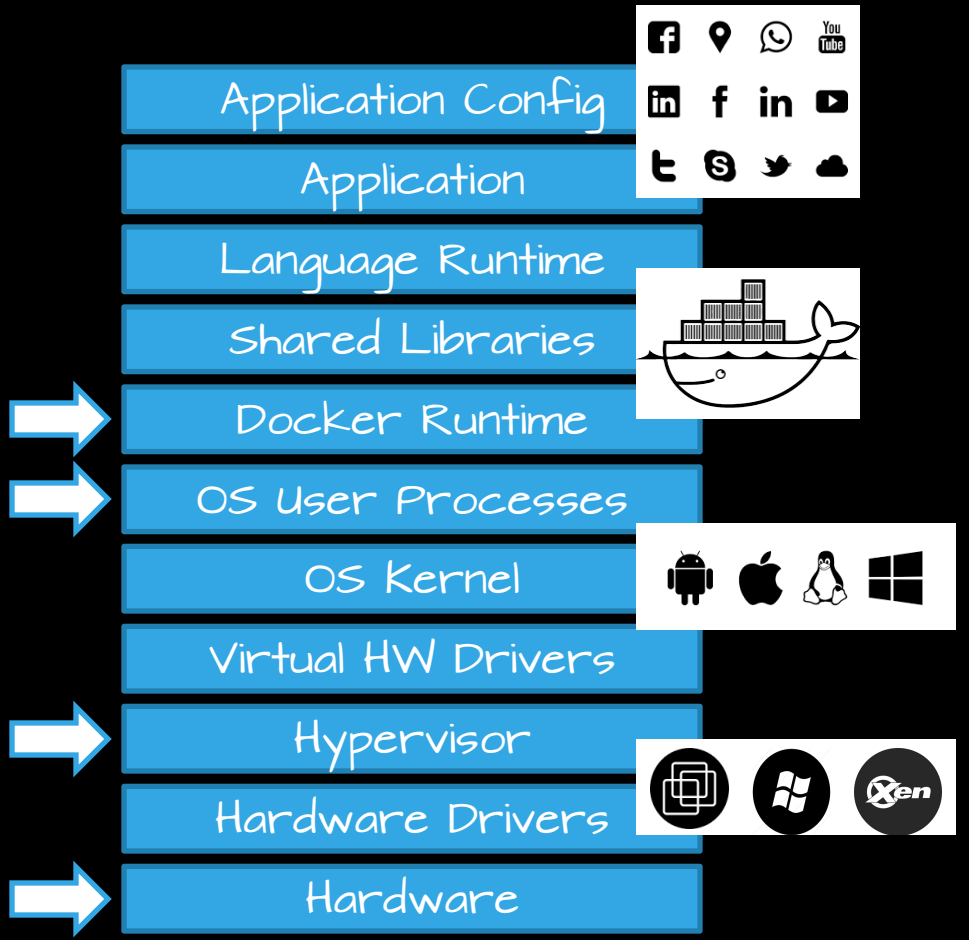
Unik

Idit Levine

Virtualization Stack

The aim is to run single Application with a single user on a single server

Redundancy in the stack - e.g. Isolation



Kernel Complexity - Protection

Application safe from application

User safe from user

Application safe from user

Inefficiency

- Needless permission check, it is hard and an updated model from time sharing computer from the 50s, 60s
- Microservices architecture duplicate what Linux did for us
- Kernel include a lot of unnecessary drivers that not being used: floppy
- Update and patches using yam bring a lot of unnecessary components

Security

- Very large attack surface
- A lot of exploits target linux.
  It is harder to attack hypervisor - not expose to the internet
- Microservices architecture
  Sharing - Kernel, Memory, filesystem, hardware
  The only thing make it safe is kernel extension like: cgroup

How did we get here ? Evolution !

Unix was supported us the entire way!

Decades of backwards compatibility

What can linux run on ?

Anything !

What can run on linux ?

Anything !

Trade Off

Compatibility

vs

Efficiency

Make it works.

Make it right.

Make it fast.

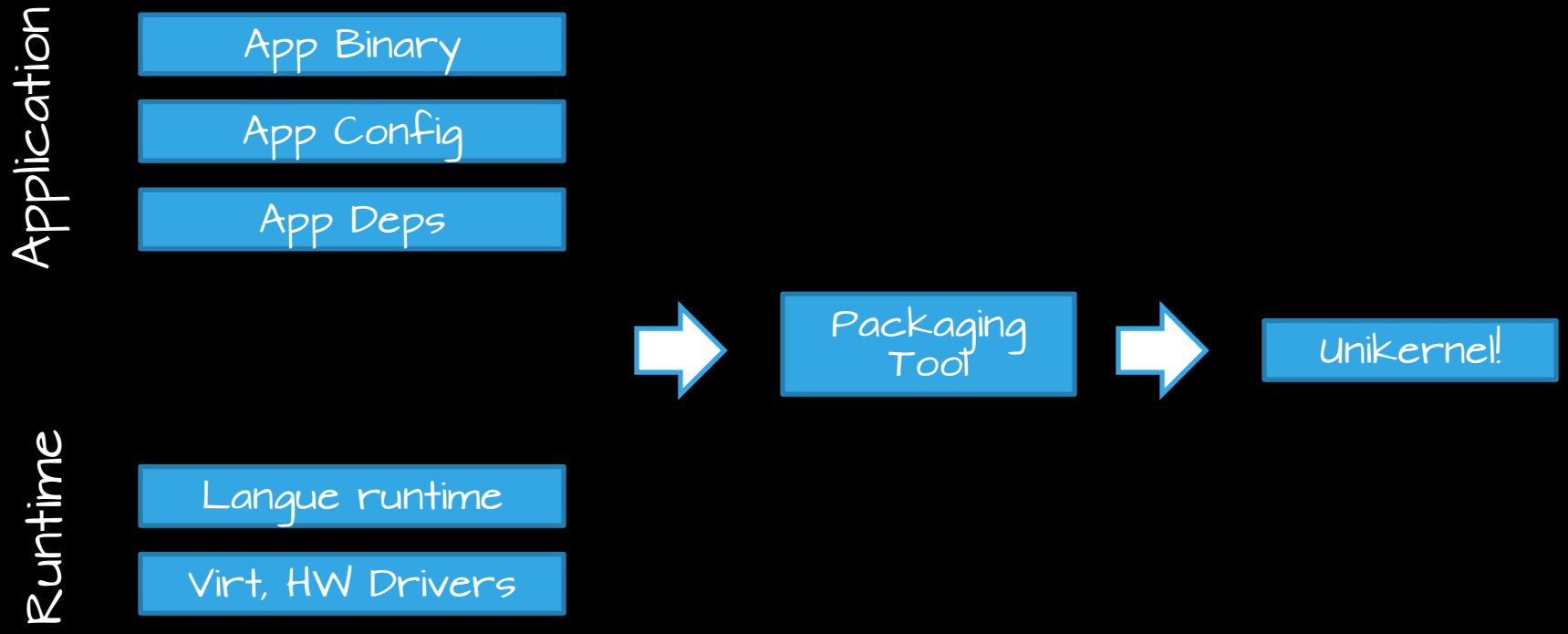
{uni-}

One; having or consisting of one.

{kernel}

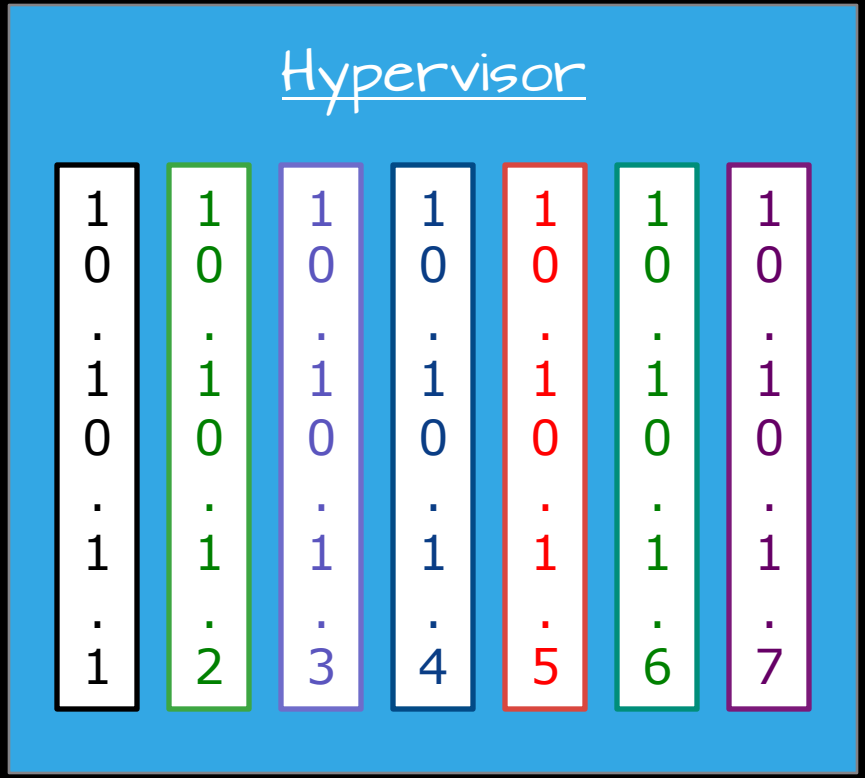
a bridge between applications and the actual data processing done at the hardware level.

Unikernel creation



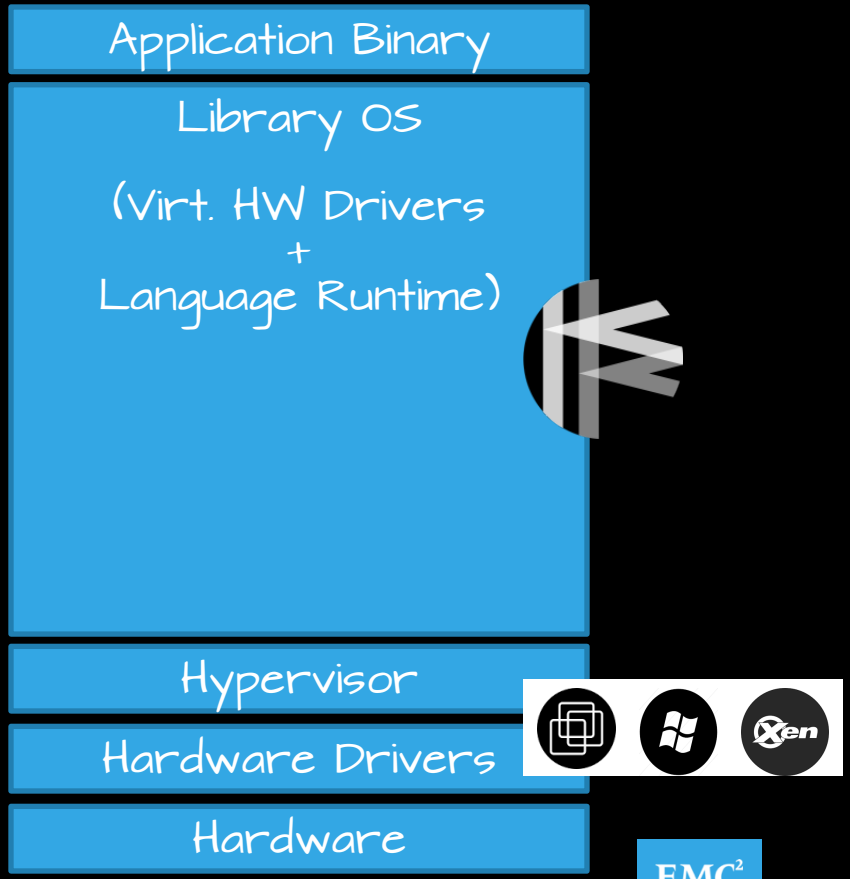
Unikernel Stack

- Unikernels deploy directly against the hypervisor
- Unikernels have their own network stack
- Unikernels have their own virtualize memory presented as hardware
- Unikernel are completely self contain & ideally immutable as well

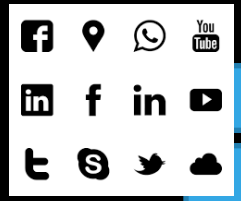


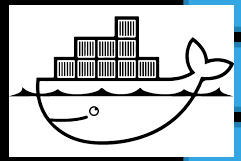
Unikernel Stack

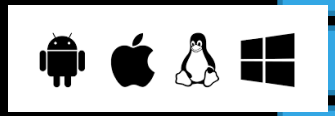
Less layer, less code, much simpler !



Docker Stack vs. Unikernel Stack







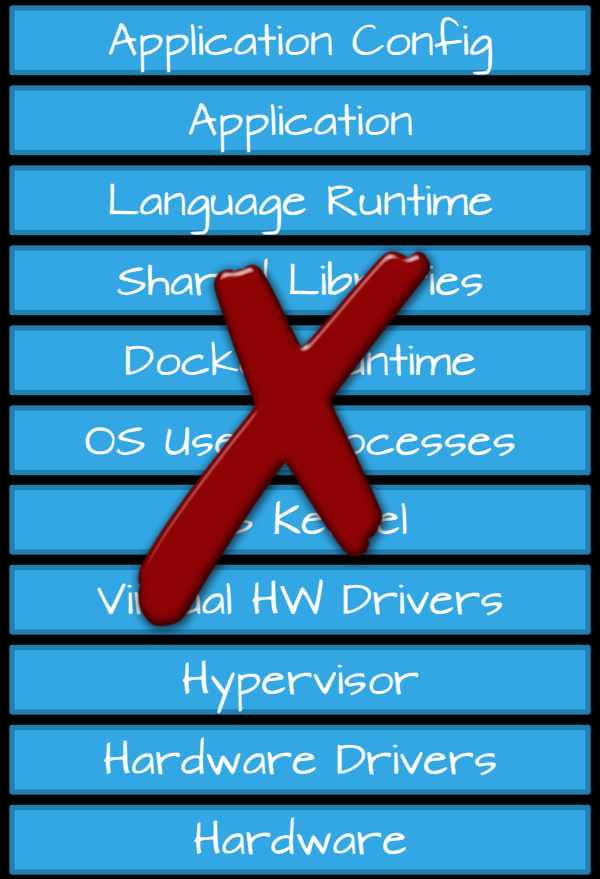
Docker Stack	Unikernel Stack
Application Config	Application Binary
Application	Library OS (Virt. HW Drivers + Language Runtime)
Language Runtime	
Shared Libraries	
Docker Runtime	
OS User Processes	
OS Kernel	
Virtual HW Drivers	
Hypervisor	Hypervisor
Hardware Drivers	Hardware Drivers
Hardware	Hardware

How can unikernels help address our problems?

Minimized layers of isolation and abstraction

Include only what we really need !

Less code, Less bug, easy to reason about



Unikernel advantages

- No other users, no multi users support
- No permission checks - you can utilis 100% of your hardware
- Isolation at the virtual hardware - only !
- Shared only hardware
- Minimum virtual machine ~1 gb in size, minimum unikernel is tiny kb in size
- Very fast boot time
- A tiny custom surface of attack, less likely to be effected by a public exploit

Backward compatibility

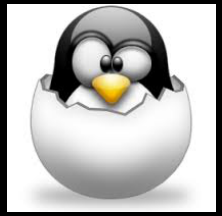
Forward compatibility

POSIX compliance

Language specifics

Unik

```
bash
Last login: Sun Feb 28 19:57:35 on ttys000
usxxlevinim3:~ levini$ unik
NAME:
   unik -

USAGE:
   unik [global options] command [command options] [arguments...]

VERSION:
   0.0.0

COMMANDS:
   delete, rm                   delete running instances
   delete-unikernel, rmu        delete compiled unikernel
   logs, l                      get stdout/stderr from a running unikernel
   ps                           list running instances
   push, p                      Push and push a new unikernel from the source code at PATH
   run, r                       run one or more instances of a unikernel
   target, t                    set unik cli endpoint or view current endpoint
   unikernels, u                list compiled unikernels
   list-volumes, lv             list unik-managed volumes
   create-volume, cv            create a unik-managed volume with NAME of SIZE GB
   delete-volume, rmv           delete volume
   attach-volume, av            attach volume to instance
   detach-volume, dv            detach volume from instance
   help, h                      Shows a list of commands or help for one command

GLOBAL OPTIONS:
   --verbose, -V        stream logs from the unik backend
   --help, -h           show help
   --version, -v        print the version

usxxlevinim3:~ levini$
```



Unik builds and runs unikernels on a variety of cloud providers through an easy-to-use REST API or a simple command-line tool


vagrant up -provider=aws
unik target 54.209.79.227
unik push unik-demo .
unik run unik-demo

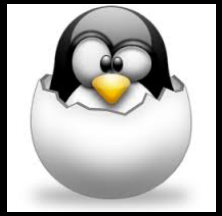
Unik is NOT opinionated !

Unikernel types

Processor architectures

Cloud providers

Unik hub



Unikernel hub: http://www.unikhub.tk

Unik integration with Docker

Docker API can be used to created unilkernel via Unik

bash bash

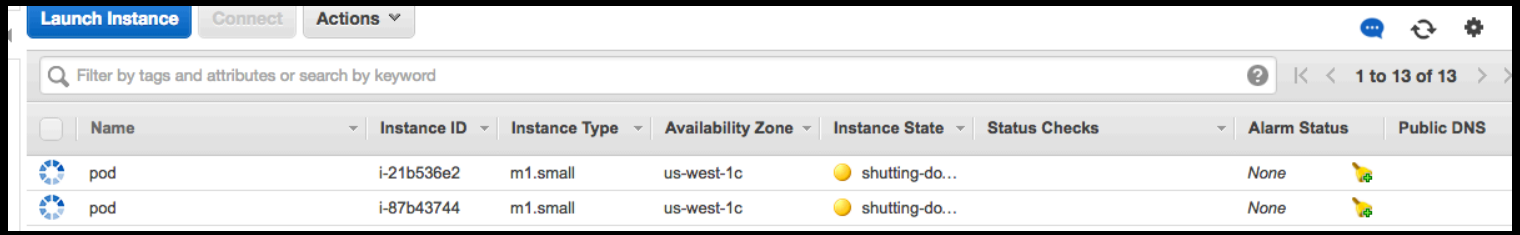
```
Last login: Sun Feb 28 19:58:15 on ttys005
usxxlevinim3:~ levini$ docker -H tcp://52.91.115.67:3000 images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
unikdemo            latest              ami-ae1624c4        292 years ago       1 kB
usxxlevinim3:~ levini$ docker -H tcp://52.91.115.67:3000 ps
CONTAINER ID        IMAGE               COMMAND             CREATED              STATUS              PORTS               NAMES
unikdemo_ce4        unikdemo            "go run main.go"    About a minute ago   running                                 nikdemo_ce440fd1-9fdf-4826-99d
9-8ebac5283615
usxxlevinim3:~ levini$ docker -H tcp://52.91.115.67:3000 run -d unikdemo
unikdemo_49ac9d1c-cd1e-4295-8519-ed4b675ce44c
usxxlevinim3:~ levini$ docker -H tcp://52.91.115.67:3000 ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
unikdemo_ce4        unikdemo            "go run main.go"    2 minutes ago       running                                 nikdemo_ce440fd1-9fdf-4826-99d9
-8ebac5283615
unikdemo_49a        unikdemo            "go run main.go"    17 seconds ago      pending                                 nikdemo_49ac9d1c-cd1e-4295-8519
-ed4b675ce44c
usxxlevinim3:~ levini$ docker -H tcp://52.91.115.67:3000 ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
unikdemo_ce4        unikdemo            "go run main.go"    2 minutes ago       running                                 nikdemo_ce440fd1-9fdf-4826-99d9
-8ebac5283615
unikdemo_49a        unikdemo            "go run main.go"    45 seconds ago      running                                 nikdemo_49ac9d1c-cd1e-4295-8519
-ed4b675ce44c
usxxlevinim3:~ levini$
```

Unik integration with kubernetes

Kubernetes support docker, rocket and now also unik !

```
ubuntu@ip-172-31-4-251:~/go/src/k8s.io/kubernetes$ kubectl run pod k8s-demo --image=exampleapp-unik-2
replicationcontroller "pod" created
```

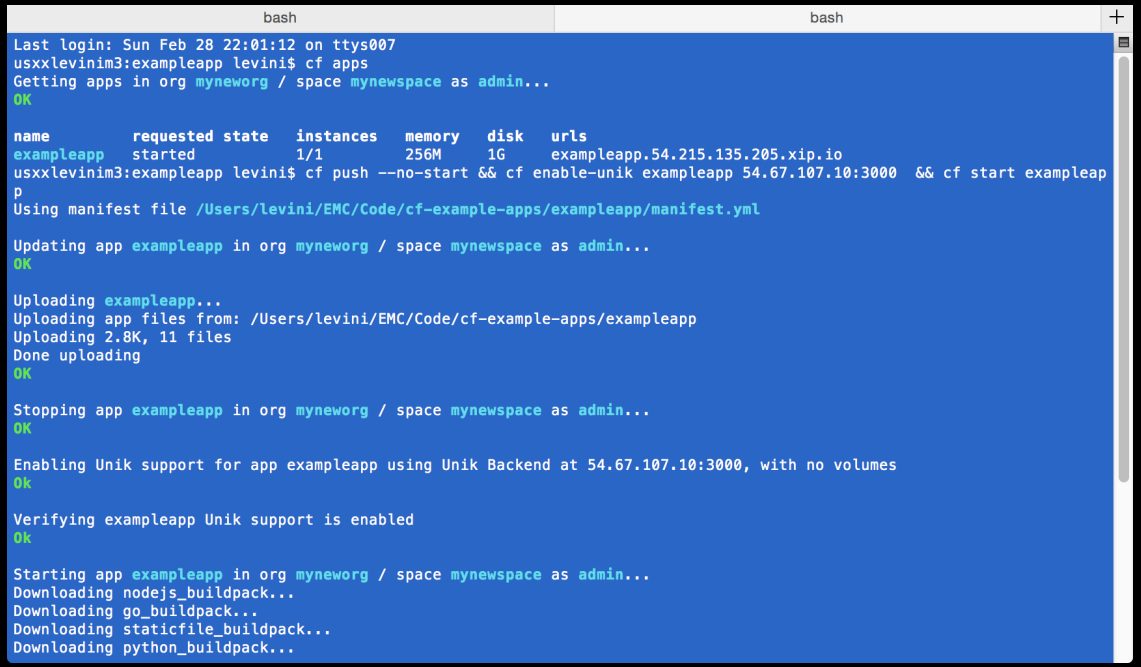
```
ubuntu@ip-172-31-4-251:~/go/src/k8s.io/kubernetes$ kubectl get pods
NAME        READY     STATUS    RESTARTS   AGE
pod-7i2mr   1/1       Running   1          1m
ubuntu@ip-172-31-4-251:~/go/src/k8s.io/kubernetes$ kubectl get rc
CONTROLLER   CONTAINER(S)   IMAGE(S)           SELECTOR   REPLICAS   AGE
pod          pod            exampleapp-unik-2   run=pod    1          2m
ubuntu@ip-172-31-4-251:~/go/src/k8s.io/kubernetes$ kubectl delete pod pod-7i2mr
pod "pod-7i2mr" deleted
ubuntu@ip-172-31-4-251:~/go/src/k8s.io/kubernetes$ kubectl delete rc pod
replicationcontroller "pod" deleted
```



Unik with Cloud Foundry

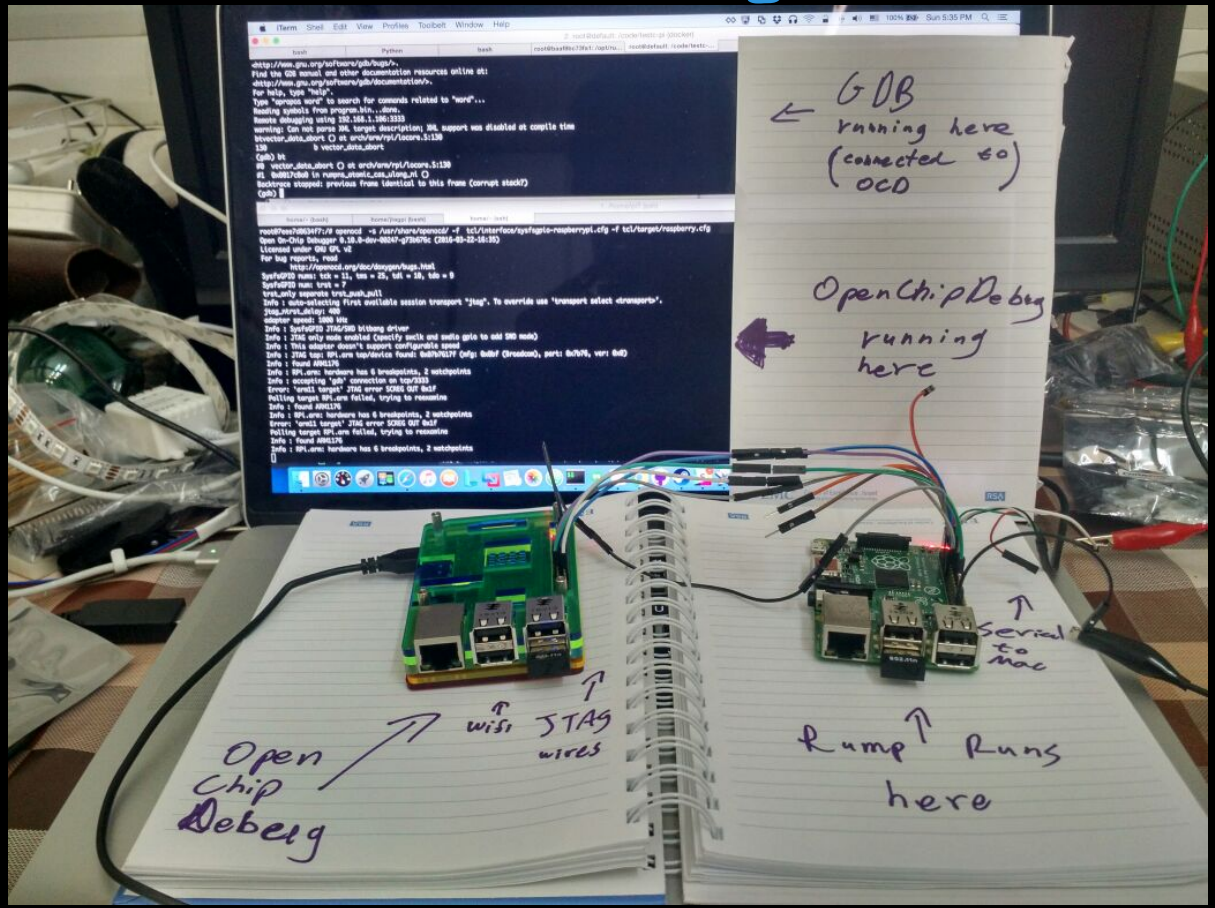
CLOUD FOUNDRY

To provide the user with a seamless PaaS experience, Unik is integrated as a backend to Cloud Foundry runtime.

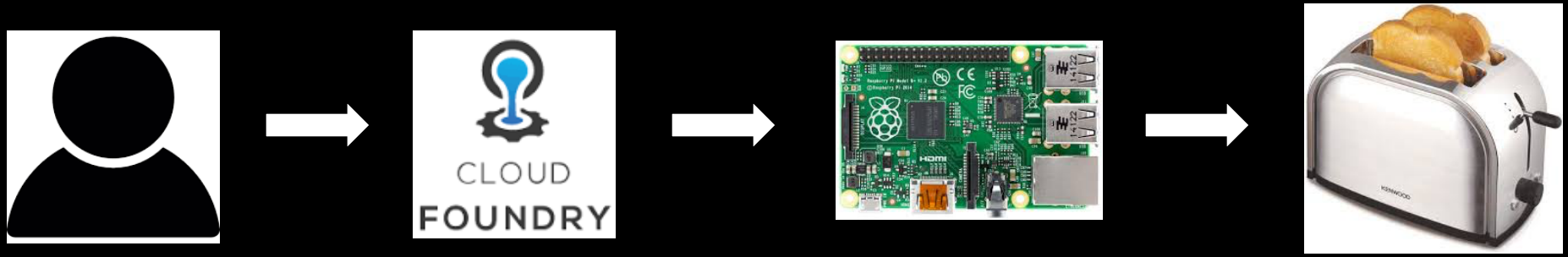


EMC²

Vision - Internet of Things



Vision - Internet of Things



A user push a unikernel application to cloud foundry. Cloud Foundry deploy the unikernel application on Raspberry PI.
The application talking to a toaster and make a toast for the user to eat. Classic use case of Internet of things.

EMC²

@Idit_Levine

Because the people who are crazy enough to think they can change the world are the ones who do.

R.I.P. Steve Jobs