# CONTAINING THE OS

## WHAT'S LEFT?

Vincent Batts @vbatts

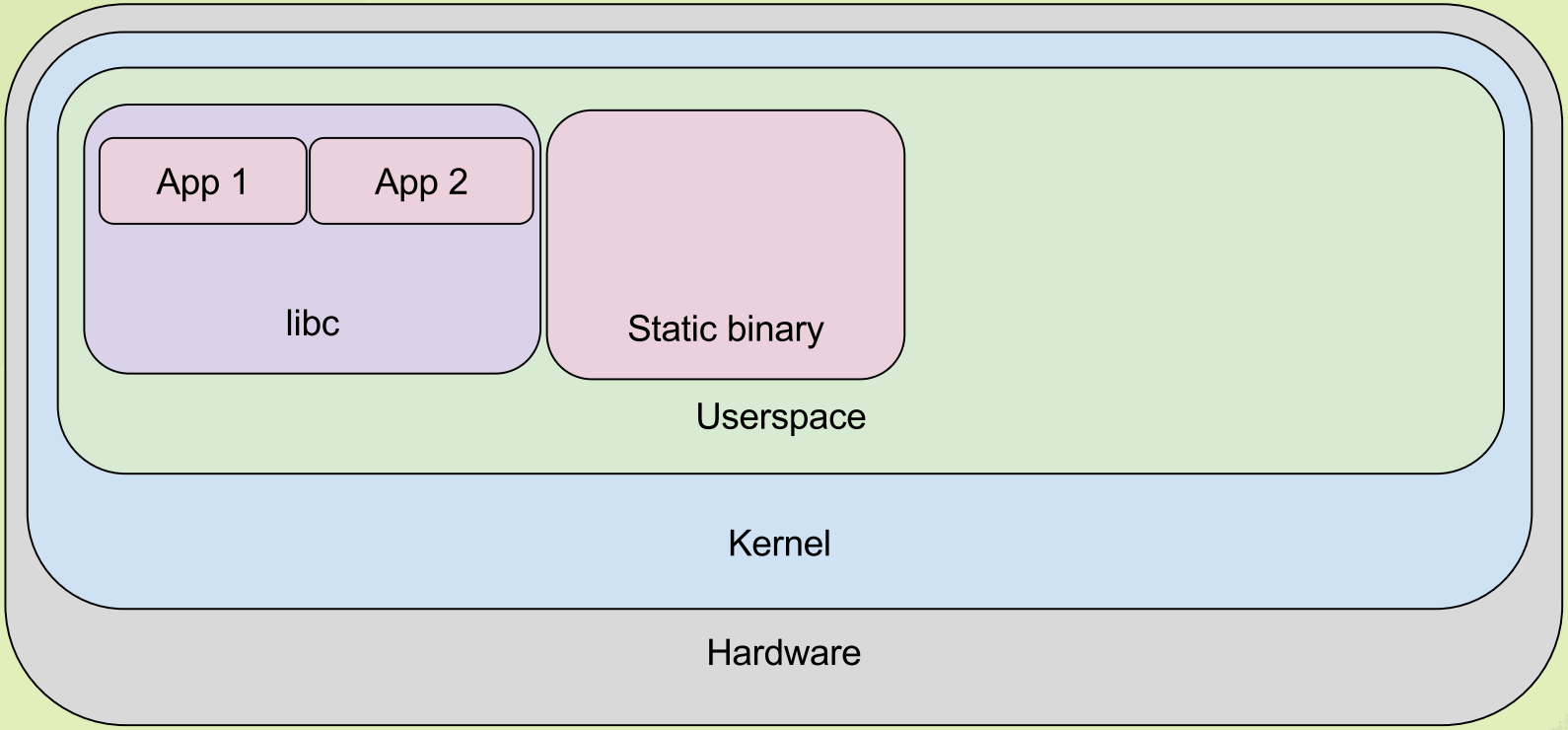
```
$> finger $(whoami)
Login: vbatts                           Name: Vincent Batts
Directory: /home/vbatts                 Shell: /bin/bash
Such mail.
Plan:
OHMAN
$> id -Gn
devel opencontainers docker appc redhat golang slackware
```

# CONTAINERS

(Cite: the internet)

# CONTAINERS


(Cite: The Internet)


App 1
App 2
libc
Static binary
Userspace
Kernel
Hardware

Kernel's Guarantee:

**DON'T BREAK USERSPACE**

Kernel's Guarantee:

# DON'T BREAK USERSPACE

But what is there to break?

Kernel's Guarantee:

# DON'T BREAK USERSPACE

But what is there to break?

- syscalls (open, read, write, close, exec, fork, mmap, mount, stat, etc.)

Kernel's Guarantee:

# DON'T BREAK USERSPACE

But what is there to break?

- syscalls (open, read, write, close, exec, fork, mmap, mount, stat, etc.)
- signals

Kernel's Guarantee:

# DON'T BREAK USERSPACE

But what is there to break?

- syscalls (open, read, write, close, exec, fork, mmap, mount, stat, etc.)
- signals
- ioctl's

Kernel's Guarantee:

# DON'T BREAK USERSPACE

But what is there to break?

- syscalls (open, read, write, close, exec, fork, mmap, mount, stat, etc.)
- signals
- ioctl's
- prctl's

Kernel's Guarantee:

# DON'T BREAK USERSPACE

But what is there to break?

- syscalls (open, read, write, close, exec, fork, mmap, mount, stat, etc.)
- signals
- ioctl's
- prctl's
- fcntl's

Kernel's Guarantee:

# DON'T BREAK USERSPACE

But what is there to break?

- syscalls (open, read, write, close, exec, fork, mmap, mount, stat, etc.)
- signals
- ioctl's
- prctl's
- fcntl's
- sysfs

Kernel's Guarantee:

# DON'T BREAK USERSPACE

But what is there to break?

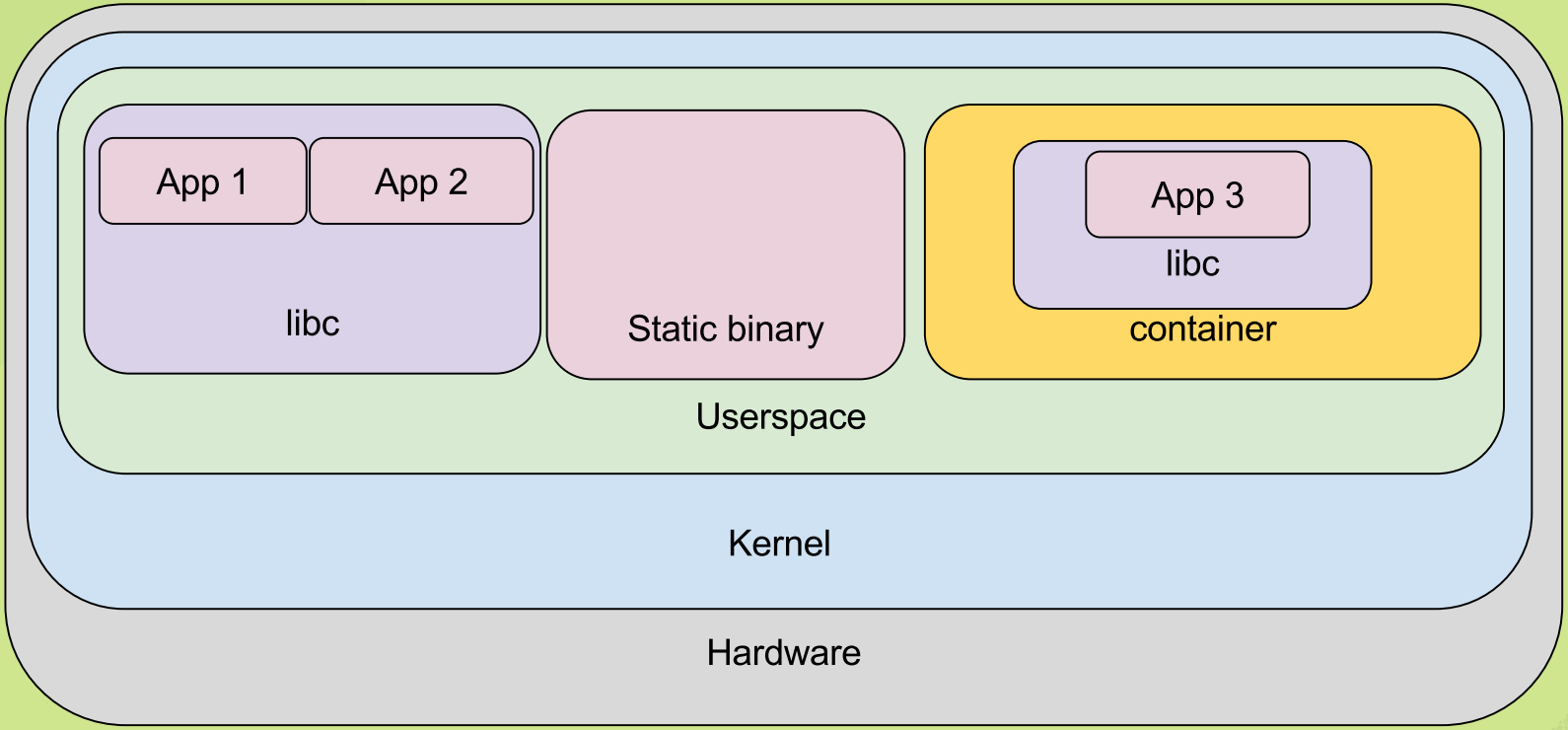
- syscalls (open, read, write, close, exec, fork, mmap, mount, stat, etc.)
- signals
- ioctl's
- prctl's
- fcntl's
- sysfs
- procfs

Kernel's Guarantee:

# DON'T BREAK USERSPACE

But what is there to break?

- syscalls (open, read, write, close, exec, fork, mmap, mount, stat, etc.)
- signals
- ioctl's
- prctl's
- fcntl's
- sysfs
- procfs
- and more, I'm sure


App 1
App 2
libc
Static binary
App 3
libc
container
Userspace
Kernel
Hardware

## CONTAINERS:

**CONTAINERS:**

Share the host's kernel

**CONTAINERS:**

Share the host's kernel

Crashes and Exploits alike

**CONTAINERS:**

Share the host's kernel

Crashes and Exploits alike

virtualizing by "namespacing" kernel resources and concepts

## CONTAINERS:

Share the host's kernel

Crashes and Exploits alike

virtualizing by "namespacing" kernel resources and concepts

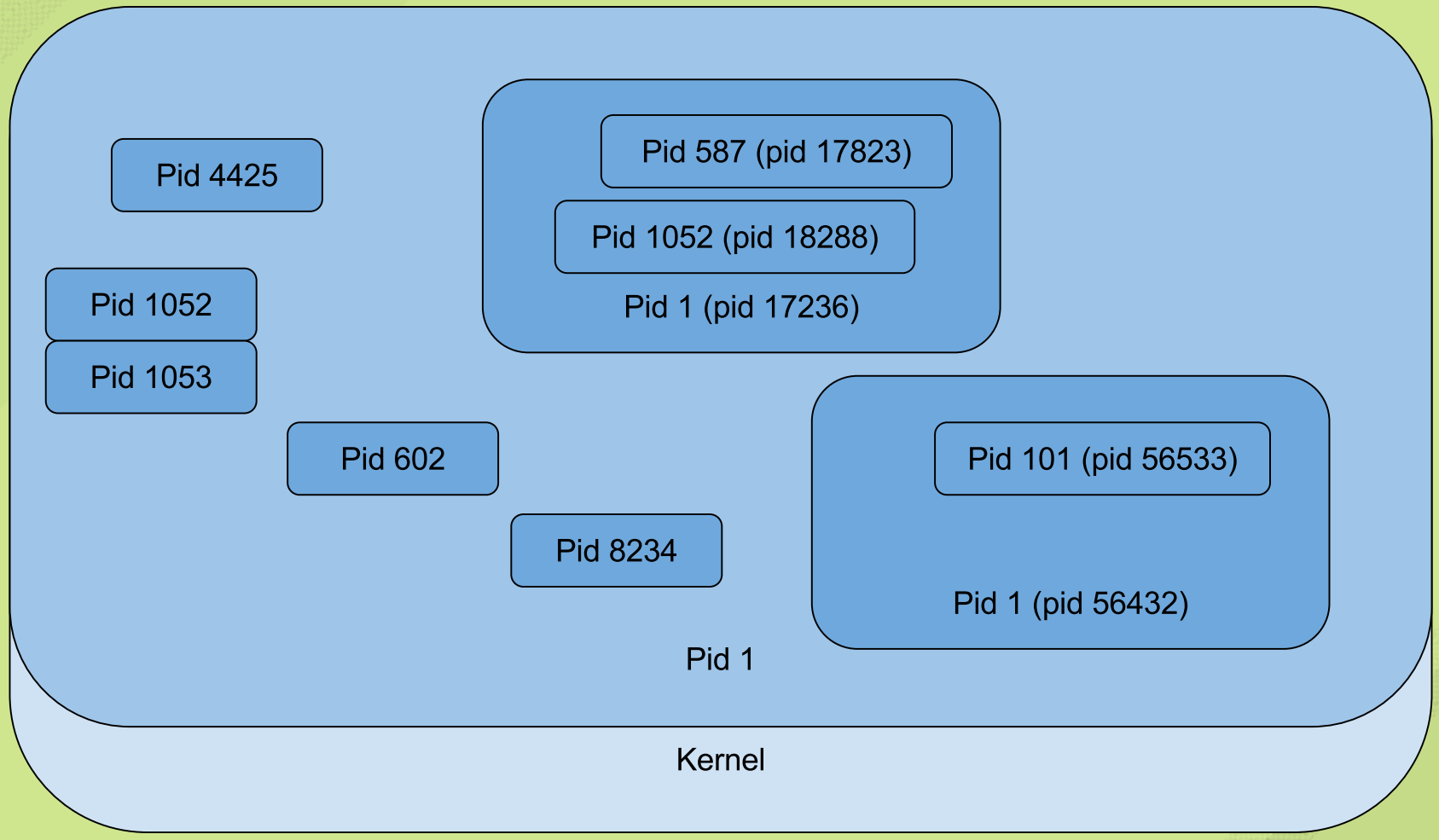
Isolation by control groups, syscall filtering, and Linux Security Modules (SELinux, apparmor, etc.)

## KERNEL NAMESPACES:

unshare() docs

- mount
- IPC (message queues, semaphores, shm)
- UTS (hostname)
- network
- PID
- cgroup
- user

## KERNEL NAMESPACES: PID



# CONTAINER DISTRIBUTION

## CONTAINER DISTRIBUTION

How many have attempted to configure *some* open source project?

## CONTAINER DISTRIBUTION

How many have attempted to configure *some* open source project?

Discovered it required *other* projects to be configured first

## CONTAINER DISTRIBUTION

How many have attempted to configure *some* open source project?

Discovered it required *other* projects to be configured first

Which required *still* more projects to be configured

## CONTAINER DISTRIBUTION

How many have attempted to configure *some* open source project?

Discovered it required *other* projects to be configured first

Which required *still* more projects to be configured

Only to find a fundamental incompatibility with the distro version

## CONTAINER DISTRIBUTION

How many have attempted to configure *some* open source project?

Discovered it required *other* projects to be configured first

Which required *still* more projects to be configured

Only to find a fundamental incompatibility with the distro version

# CONTAINER DISTRIBUTION

# CONTAINER DISTRIBUTION

Root ('/') File System

# CONTAINER DISTRIBUTION

Root ('/') File System

Approaches:

- Tar Archive
- Raw Image
- rsync
- ostree

## CONTAINER DISTRIBUTION

Root ('/') File System

Approaches:

- Tar Archive
- Raw Image
- rsync
- ostree

Standardize the formats (see Open Container Initiative)

# WHAT'S LEFT?

## WHAT'S LEFT?

Cloud Native application development (see CNCF)

## WHAT'S LEFT?

Cloud Native application development (see CNCF)

Rather than only shoving "legacy" code in new boxes

## WHAT'S LEFT?

Cloud Native application development (see CNCF)

Rather than only shoving "legacy" code in new boxes

Discoverable APIs (see OpenAPIs)

## WHAT'S LEFT?

Cloud Native application development (see CNCF)

Rather than only shoving "legacy" code in new boxes

Discoverable APIs (see OpenAPIs)

"Scheduled" functionality (see OpenShift and Kubernetes)

## WHAT'S LEFT?

Cloud Native application development (see CNCF)

Rather than only shoving "legacy" code in new boxes

Discoverable APIs (see OpenAPIs)

"Scheduled" functionality (see OpenShift and Kubernetes)

intercommunication (see gRPC)

## WHAT'S LEFT?

Cloud Native application development (see CNCF)

Rather than only shoving "legacy" code in new boxes

Discoverable APIs (see OpenAPIs)

"Scheduled" functionality (see OpenShift and Kubernetes)

intercommunication (see gRPC)

event driven functions (aka "serverless")

intelligent routing (istio and envoy)

trusted pipeline (CI/CD, grafeas, etc)

# CLOUD

(Cite: the internet)

VINCENT BATTS
@VBATTS| VBATTS@REDHAT.COM

THANKS!