WHAT IS IT WE WANT IN CONTAINERS ANYWAYS?

Vincent Batts @vbatts

bit.ly/vbatts-containers-anyways

```
$> finger $(whoami)
```

Login: vbatts   **Name:** Vincent Batt

Directory: /home/vbatts   **Shell:** /bin/bash

Such mail.

Plan:

OHMAN
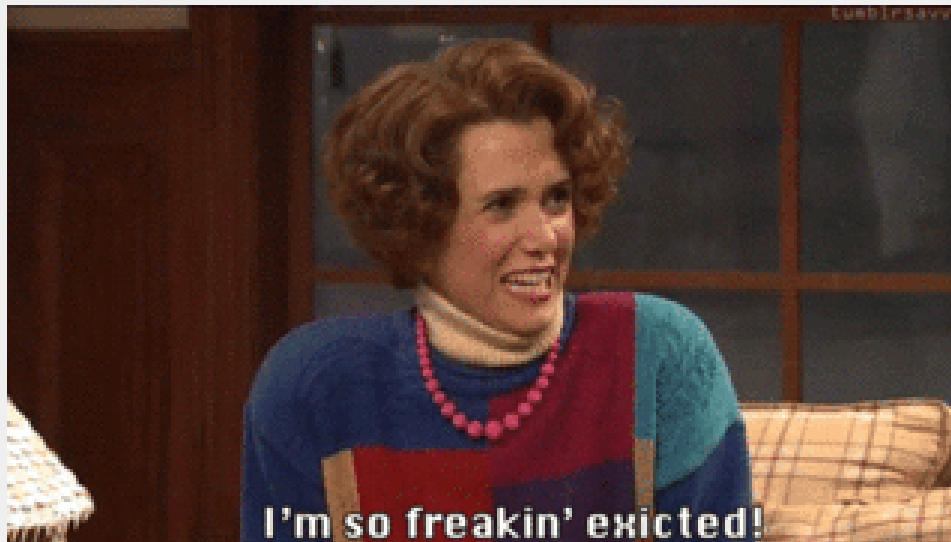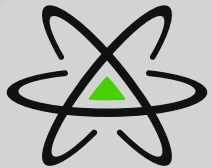
SO,
WHY, CONTAINERS?

SO,
WHY, CONTAINERS?

Single Application

SO,
WHY, CONTAINERS?

```
Single Application
     Full System
```

SO,
WHY, CONTAINERS?

Single Application
    Full System
       But Not a VM

SO,
WHY, CONTAINERS?

```
Single Application
    Full System
        But Not a VM
            Except Maybe a VM
```

SO,
WHY, CONTAINERS?

```
Single Application
    Full System
        But Not a VM
            Except Maybe a VM
        Pods of applications
```

SO,
WHY, CONTAINERS?

```
Single Application
    Full System
        But Not a VM
            Except Maybe a VM
        Pods of applications
    Labels of services
```

SO,
WHY, CONTAINERS?

```
Single Application
      Full System
            But Not a VM
                  Except Maybe a VM
            Pods of applications
      Labels of services
Non-root
```

SO,
WHY, CONTAINERS?

```
Single Application
        Full System
                But Not a VM
                        Except Maybe a VM
                Pods of applications
        Labels of services
Non-root
        Desktop Applications
```

SO,
WHY, CONTAINERS?

```
Single Application
      Full System
            But Not a VM
                  Except Maybe a VM
            Pods of applications
      Labels of services
Non-root
      Desktop Applications
            OMG AND CATS
```

SO,
WHY, CONTAINERS?

```
Single Application
    Full System
        But Not a VM
            Except Maybe a VM
        Pods of applications
    Labels of services
Non-root
    Desktop Applications
        OMG AND CATS
```

But wait,
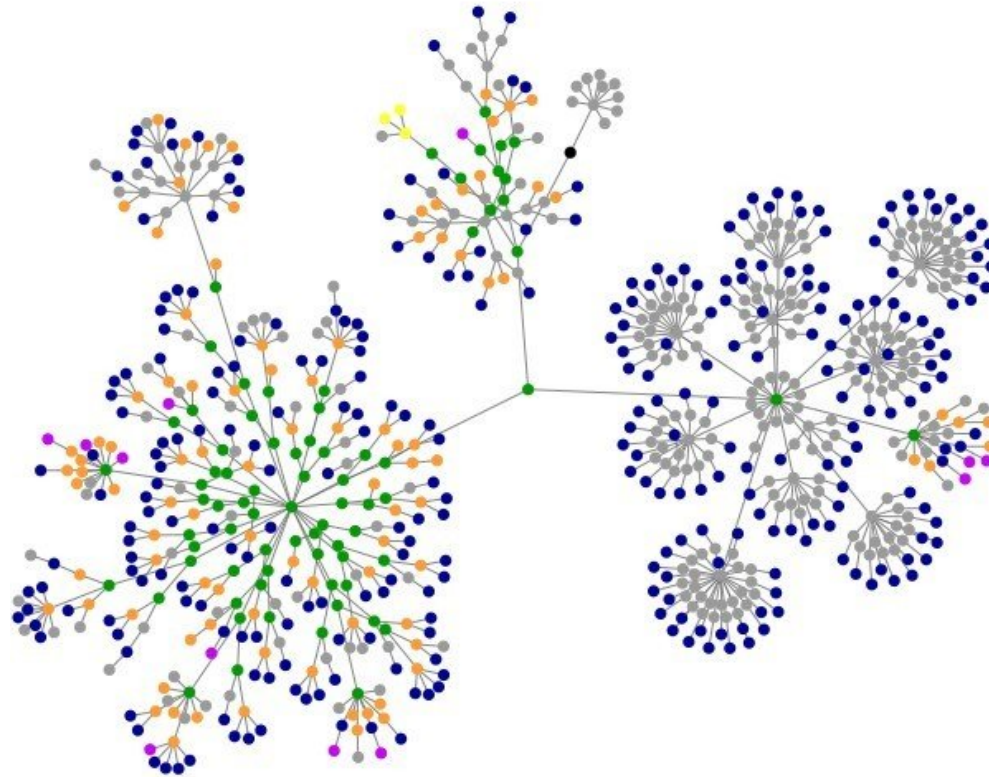What does "container" mean to you?

But wait,
   What does "container" mean to you?

But wait,
   What does "container" mean to you?

But wait,
    What does "container" mean to you?

USE-CASE

Reproducibility

USE-CASE

Reproducibility

USE-CASE

Ephemeral Environments

USE-CASE

Ephemeral Environments

USE-CASE

Freedom from host restrictions
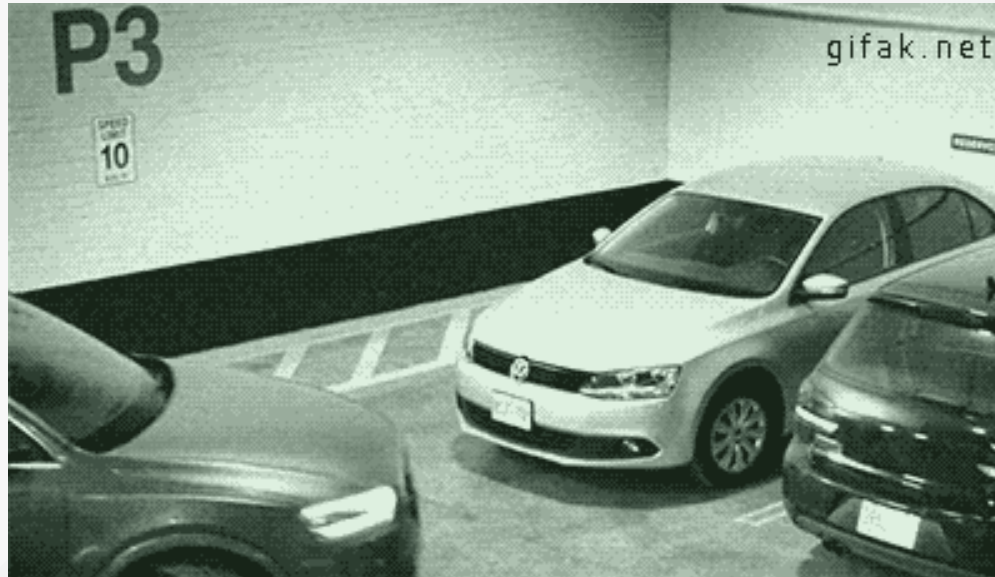
USE-CASE

Freedom from host restrictions

USE-CASE

Easy delivery

USE-CASE

Easy delivery

USE-CASE

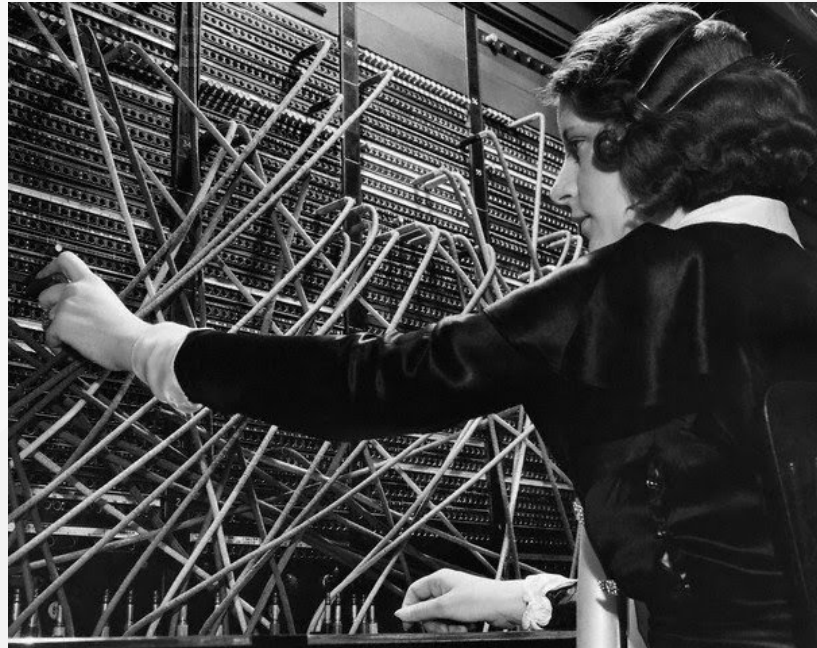Integrate to existing process

USE-CASE

Integrate to existing process

USE-CASE

Controls and knobs

USE-CASE

Controls and knobs

BUILD

BUILD

Dockerfile

BUILD

Dockerfile

Source to Image (github.com/OpenShift/source-to-image)

BUILD

Dockerfile

Source to Image (github.com/OpenShift/source-to-image)

appc/acbuild (github.com/appc/acbuild)

BUILD

Dockerfile

Source to Image (github.com/OpenShift/source-to-image)

appc/acbuild (github.com/appc/acbuild)

Your own Makefiles?

SHARING

SHARING

registry (i.e. docker-registry, dockyard)

SHARING

registry (i.e. docker-registry, dockyard)

Host it yourself (i.e. tarballs on an http server)

SHARING

registry (i.e. docker-registry, dockyard)

Host it yourself (i.e. tarballs on an http server)

Share recipes!

TOOLS

TOOLS

[LXC / LXD](#)

TOOLS

LXC / LXD

lmctfy

TOOLS

LXC / LXD

lmctfy

systemd-nspawn

TOOLS

LXC / LXD

lmctfy

Docker

systemd-nspawn

TOOLS

LXC / LXD

lmctfy

Docker

systemd-nspawn

runC

TOOLS

LXC / LXD

lmctfy

Docker

systemd-nspawn

runC

bubblewrap

TECH

TECH

- Namespaces

TECH

- Namespaces
- Resource Controls

TECH

- Namespaces
- Resource Controls
- Security and Isolation

SYSTEMD

SYSTEMD

[systemd directives](#)

SYSTEMD

systemd directives

Security Slides (CoreOS Fest2016)

SHELL

SHELL

unshare(1)

SHELL

unshare(1)

mount(8) shared subtrees

SHELL

unshare(1)

mount(8) shared subtrees

chroot(1)

SHELL

unshare(1)

mount(8) shared subtrees

chroot(1)

procfs, sysfs, tmpfs

SHELL

unshare(1)

mount(8) shared subtrees

chroot(1)

procfs, sysfs, tmpfs

cgroup filesystem (not for the faint of heart)

STANDARDS!

STANDARDS!

Standard

/ˈstandəd/

noun

something used as a measure, norm, or model in comparative evaluations

STANDARDS!

Areas to Standardize:
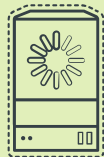
STANDARDS!

Areas to Standardize:

- Packaging

STANDARDS!

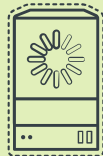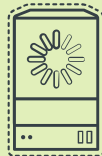**Areas to Standardize:**

- Packaging

- Runtime

STANDARDS!

Areas to Standardize:

- Packaging

- Runtime

- Networking

STANDARDS!

Areas to Standardize:

- Packaging
- Runtime
- Networking
- Cloud

Call to Action!

Call to Action!

Define <u>your</u> use-cases **first**

Call to Action!

Define <u>your</u> use-cases **first**

[Get involved in the conversations](#)

Call to Action!

Define <u>your</u> use-cases **first**

[Get involved in the conversations](#)

Ensure your container integration touchpoint stay generic,
to avoid lock-in to a particular platform.

Call to Action!

Define <u>your</u> use-cases **first**

[Get involved in the conversations](#)

Ensure your container integration touchpoint stay generic,
to avoid lock-in to a particular platform.

PoC tooling for your integration

Call to Action!

Define <u>your</u> use-cases **first**

[Get involved in the conversations](#)

Ensure your container integration touchpoint stay generic,
to avoid lock-in to a particular platform.

PoC tooling for your integration

VINCENT BATTS

[@VBATTS](#)| [VBATTS@REDHAT.COM](#)

THANKS!