Microsoft

Reinventing Container Linux for the Wasm Era (and More) with System Extensions

Andrew Randall
Principal PM Manager
Azure Core Linux

So you're about to provision a new Linux server...

The Linux distro dichotomy

Tens of thousands of additional optional packages

Thousands of included packages

Kernel + systemd

Fully mutable filesystem

General Purpose Linux

Flexible, works for just about any application

The Linux distro dichotomy

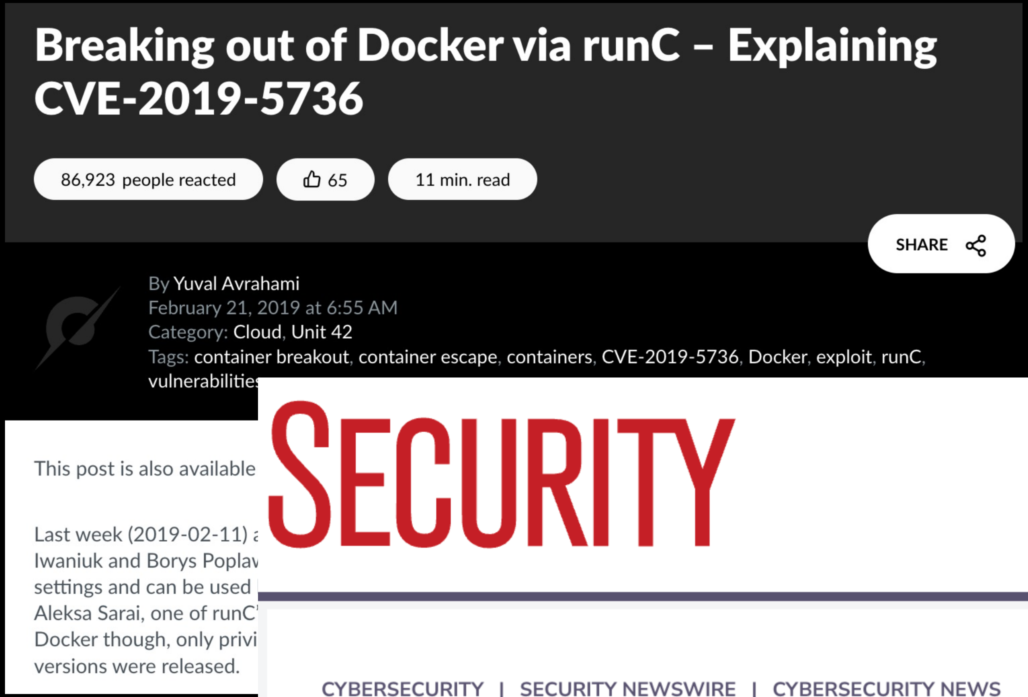
Tens of thousands of additional optional packages

Thousands of included packages

Kernel + systemd

Fully mutable filesystem

General Purpose Linux



Breaking out of Docker via runC – Explaining CVE-2019-5736

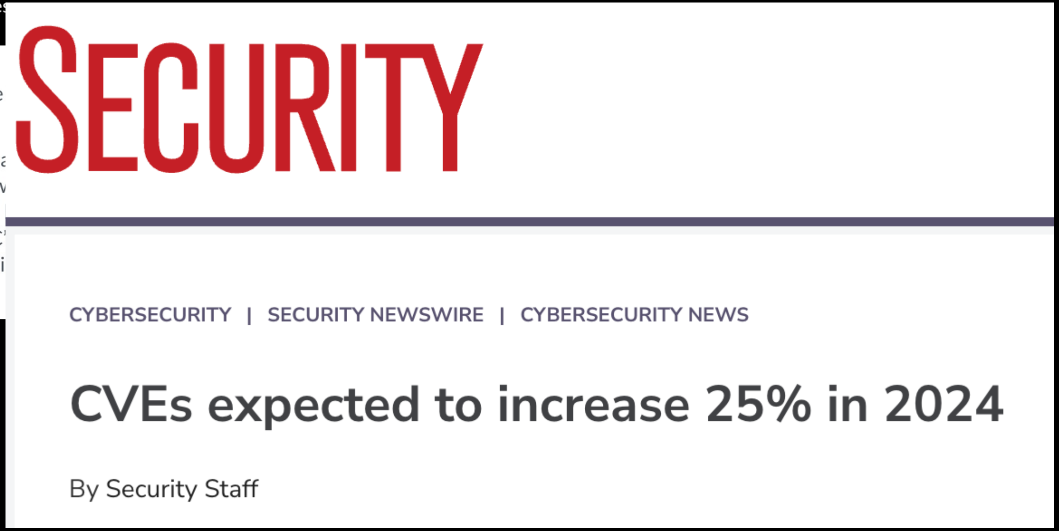
86,923 people reacted | 65 | 11 min. read

SHARE

By Yuval Avrahami
February 21, 2019 at 6:55 AM
Category: Cloud, Unit 42
Tags: container breakout, container escape, containers, CVE-2019-5736, Docker, exploit, runC, vulnerabilities

This post is also available

Last week (2019-02-11) a
Iwaniuk and Borys Poplav
settings and can be used
Aleksa Sarai, one of runC'
Docker though, only privi
versions were released.



SECURITY

CYBERSECURITY | SECURITY NEWSWIRE | CYBERSECURITY NEWS

CVEs expected to increase 25% in 2024

By Security Staff

Sources: Palo Alto Neworks, Security Magazine

The Linux distro dichotomy

Tens of thousands of additional optional packages

Thousands of included packages

Kernel + systemd

Fully mutable filesystem

General Purpose Linux

Flexible, works for just about any application

Large attack surface area

Manageability

Snowflakes / config drift

The Linux distro dichotomy

General Purpose Linux	Special Purpose Linux
Tens of thousands of additional optional packages	Container workloads loaded at runtime
Thousands of included packages	Minimal (10s/100s) collection of packages
Kernel + systemd	Kernel + systemd
Fully mutable filesystem	Immutable filesystem

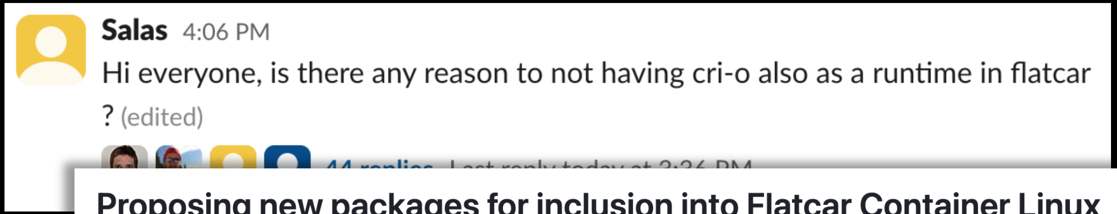
Minimal attack surface area

Manageability at scale

Repeatable deployments

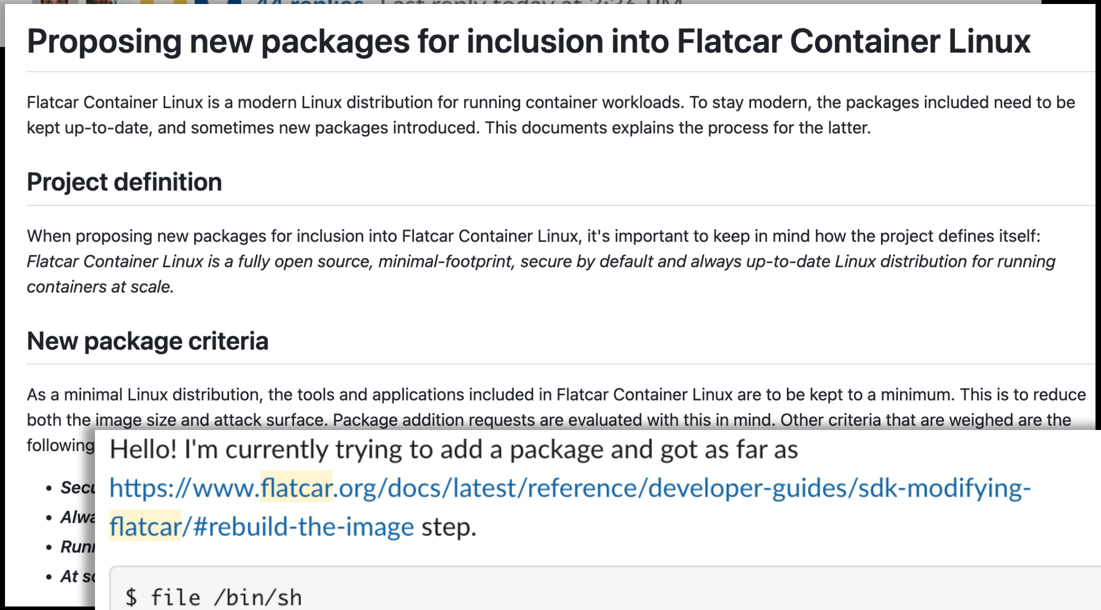
The Linux distro dichotomy

General Purpose Linux	Special Purpose Linux
Tens of thousands of additional optional packages	Container workloads loaded at runtime
Thousands of included packages	Minimal (10s/100s) collection of packages
Kernel + systemd	Kernel + systemd
Fully mutable filesystem	Immutable filesystem



Salas 4:06 PM
Hi everyone, is there any reason to not having cri-o also as a runtime in flatcar ? (edited)

44 replies Last reply today at 2:36 PM



Proposing new packages for inclusion into Flatcar Container Linux

Flatcar Container Linux is a modern Linux distribution for running container workloads. To stay modern, the packages included need to be kept up-to-date, and sometimes new packages introduced. This documents explains the process for the latter.

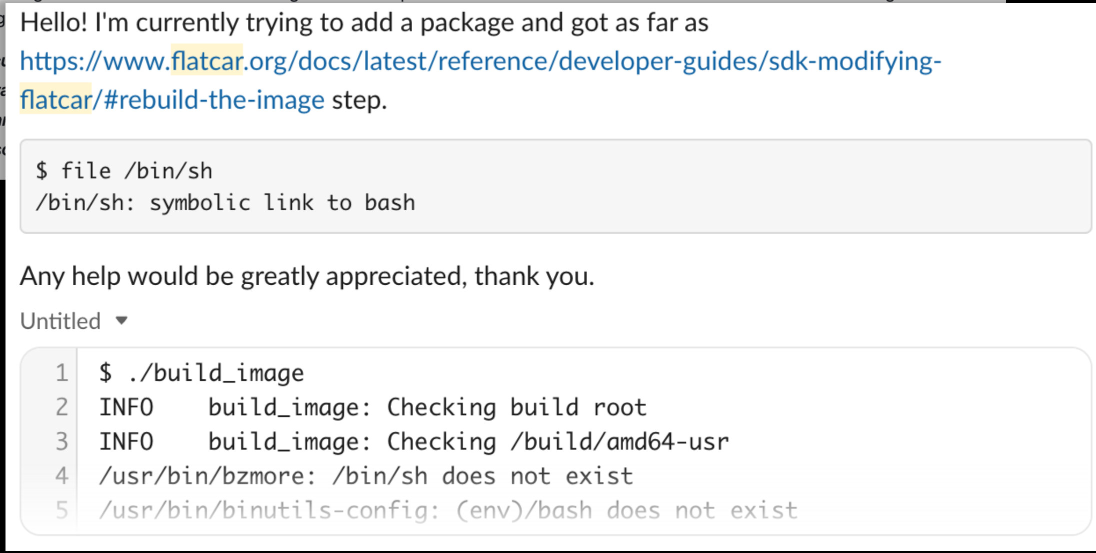
Project definition

When proposing new packages for inclusion into Flatcar Container Linux, it's important to keep in mind how the project defines itself: *Flatcar Container Linux is a fully open source, minimal-footprint, secure by default and always up-to-date Linux distribution for running containers at scale.*

New package criteria

As a minimal Linux distribution, the tools and applications included in Flatcar Container Linux are to be kept to a minimum. This is to reduce both the image size and attack surface. Package addition requests are evaluated with this in mind. Other criteria that are weighed are the following

- *Secu*
- *Alwa*
- *Runi*
- *At sc*



Hello! I'm currently trying to add a package and got as far as https://www.flatcar.org/docs/latest/reference/developer-guides/sdk-modifying-flatcar/#rebuild-the-image step.

```
$ file /bin/sh
/bin/sh: symbolic link to bash
```

Any help would be greatly appreciated, thank you.

Untitled

```
$ ./build_image
INFO    build_image: Checking build root
INFO    build_image: Checking /build/amd64-usr
/usr/bin/bzmore: /bin/sh does not exist
/usr/bin/binutils-config: (env)/bash does not exist
```

The Linux distro dichotomy

General Purpose Linux	Special Purpose Linux
Tens of thousands of additional optional packages	Container workloads loaded at runtime
Thousands of included packages	Minimal (10s/100s) collection of packages
Kernel + systemd	Kernel + systemd
Fully mutable filesystem	Immutable filesystem

Minimal attack surface area

Manageability at scale

Repeatable deployments

Inflexible - advanced knowledge required to modify base image

What if there were a better way…

Composable (Image-based) Linux

General Purpose Linux	Special Purpose Linux	Composable Linux
Tens of thousands of additional optional packages	Container workloads loaded at runtime	Container, Wasm modules, etc., loaded at runtime
Thousands of included packages	Minimal (10s/100s) collection of packages	OS extension layers loaded at boot time
Kernel + systemd	Kernel + systemd	Kernel + systemd
Fully mutable filesystem	Immutable filesystem	Immutable filesystem

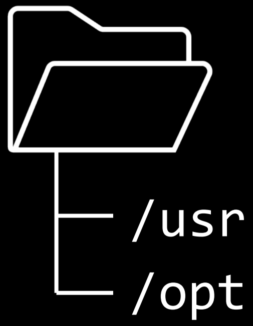
Minimal attack surface area

Manageability at scale

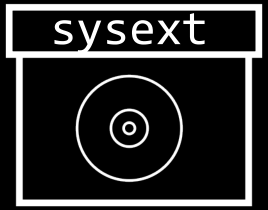
Repeatable deployments

Easy to create custom OS flavors from composable *system extension* layers

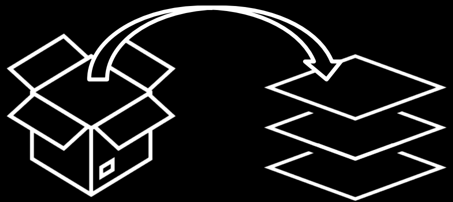
Anatomy of a System Extension (sysext)



An overlay file system containing /usr & /opt



Packaged as a disk image*



Loaded at boot time by systemd-sysext

* *typically*; can also be plain directory or btrfs subvolume

https://www.freedesktop.org/software/systemd/man/latest/systemd-sysext.html

Flatcar has embraced sysext

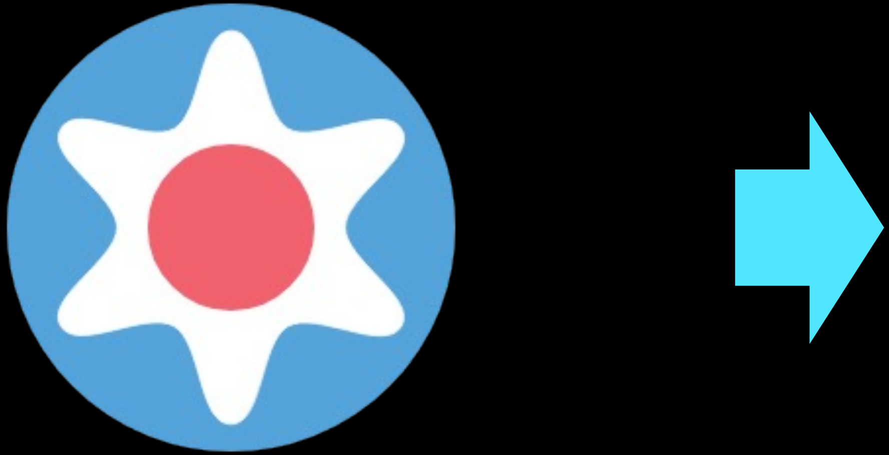
/oem

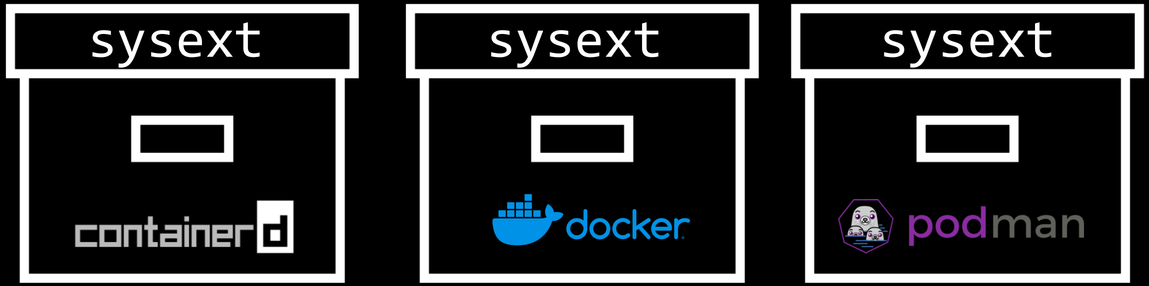
Torcx Replacement / Custom Container Runtimes

OEM Partition

Cluster API

Recent Applications in Flatcar Container Linux: 1) Torcx Replacement / Custom Container Runtimes

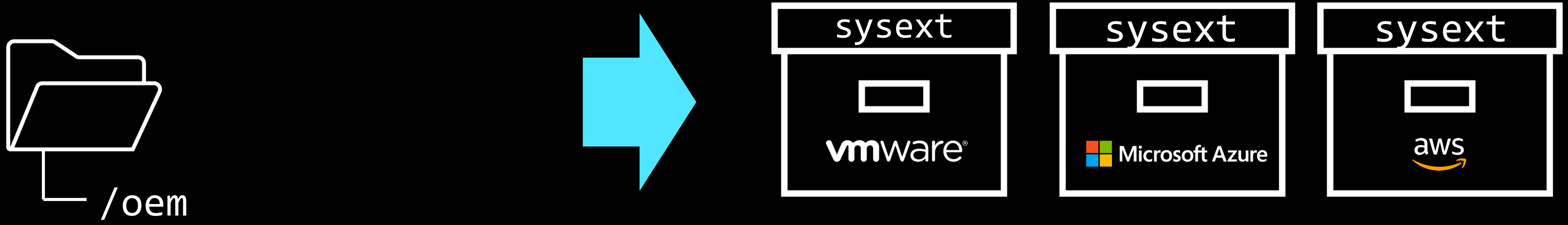




- torcx (from CoreOS)
- custom, tarball-based, complex, inflexible

- No behavior change for default (e.g. Docker, containerd)
- Easily add new runtimes (e.g. Podman) alongside or replacing standard ones

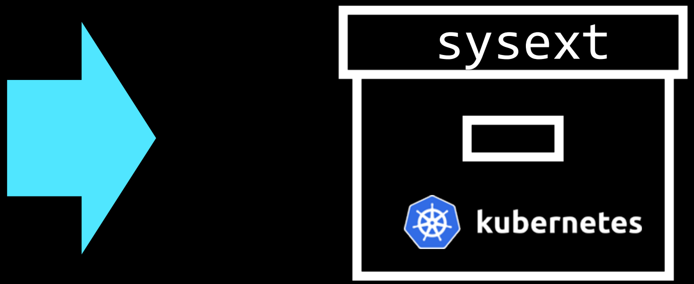
Recent Applications in Flatcar Container Linux: 2) OEM Partition



- Separate partition fixed at build time for platform-specific tools/agents
- Not upgradeable without reprovisioning entire node

- Sysext for each target platform
- In-place upgrades

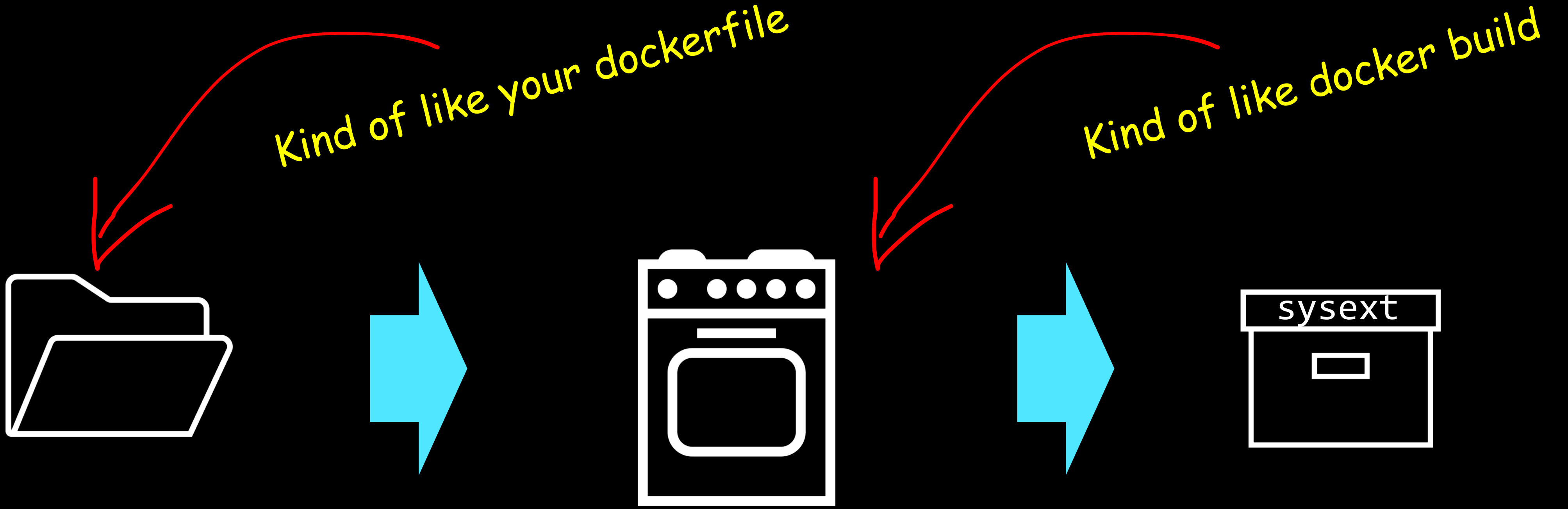
Recent Applications in Flatcar Container Linux: 3) Cluster API



- Custom worker node images combine OS + K8s control plane
- K8s + OS versions tied
- No in-place updates

- K8s control plane as sysext
- Stock distro images
- OS + K8s versions decoupled
- In-place updates

Creating Sysexts: the Flatcar Sysext Bakery



files to bake
+ config
+ metadata

bake.sh

sysext image (.raw)

https://github.com/flatcar/sysext-bakery/blob/main/README.md

Publishing sysexts

Kind of like docker push

CONF

Create checksum

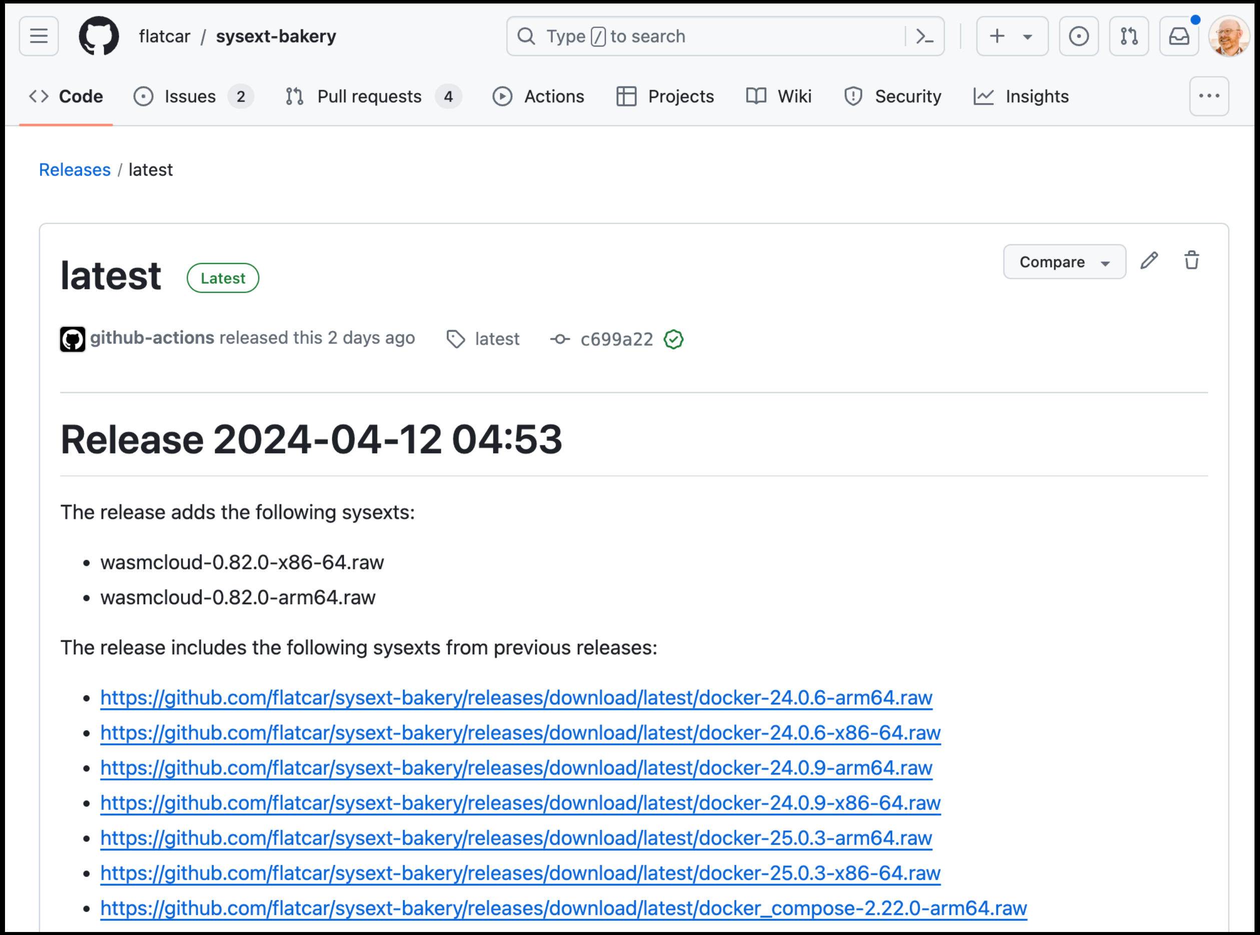
```
sha256sum *.raw > SHA256SUMS
```

Create update conf file (optional)

upload sysext image + checksum + update conf to http endpoint (e.g. GitHub as part of build pipeline)

Baked Goods, Ready to Consume

https://github.com/flatcar/sysext-bakery/releases/tag/latest



Releases / latest

latest Latest

github-actions released this 2 days ago latest c699a22

Release 2024-04-12 04:53

The release adds the following sysexts:

- wasmcloud-0.82.0-x86-64.raw
- wasmcloud-0.82.0-arm64.raw

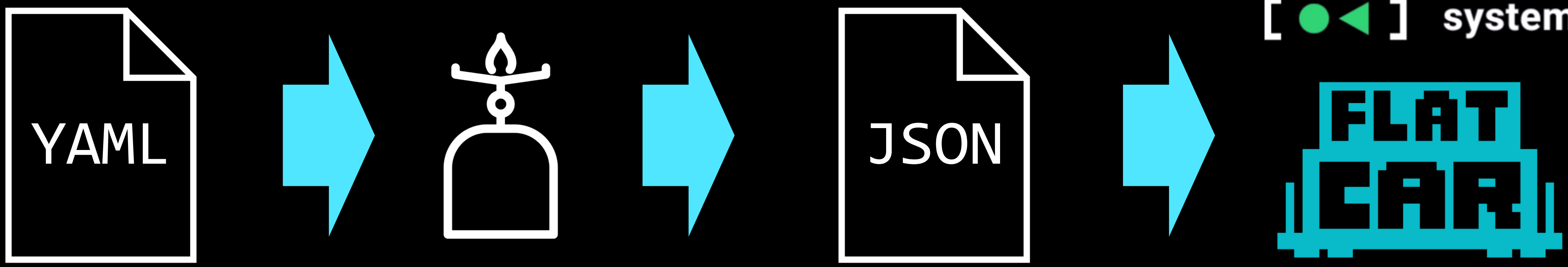
The release includes the following sysexts from previous releases:

- https://github.com/flatcar/sysext-bakery/releases/download/latest/docker-24.0.6-arm64.raw
- https://github.com/flatcar/sysext-bakery/releases/download/latest/docker-24.0.6-x86-64.raw
- https://github.com/flatcar/sysext-bakery/releases/download/latest/docker-24.0.9-arm64.raw
- https://github.com/flatcar/sysext-bakery/releases/download/latest/docker-24.0.9-x86-64.raw
- https://github.com/flatcar/sysext-bakery/releases/download/latest/docker-25.0.3-arm64.raw
- https://github.com/flatcar/sysext-bakery/releases/download/latest/docker-25.0.3-x86-64.raw
- https://github.com/flatcar/sysext-bakery/releases/download/latest/docker_compose-2.22.0-arm64.raw

- docker
- docker-compose
- kubernetes
- wasmcloud*
- wasmtime*
- cri-o (PR in progress)
- k3s (PR in progress)

* we'll come to these later

A Brief Detour into Flatcar provisioning



Butane config
(human readable)

Butane
transpiler

Ignition config
(machine readable)

This is where we want to specify
the sysext(s) to use

https://coreos.github.io/butane/

Provisioning Flatcar with a Sysext

```yaml
variant: flatcar
version: 1.0.0
storage:
  files:
    - path: /opt/extensions/wasmtime/wasmtime-17.0.1-x86-64.raw
      contents:
        source: https://github.com/flatcar/sysext-bakery/releases/download/latest/wasmtime-17.0.1-x86-64.raw
  links:
    - target: /opt/extensions/wasmtime/wasmtime-17.0.1-x86-64.raw
      path: /etc/extensions/wasmtime.raw
      hard: false
```

What about updates?

OS-independent sysexts

- E.g. standalone go binary, no OS dependencies
- systemd-sysupdate
- simple semver based mechanism over https

OS-dependent sysexts

- Needs to update in lockstep with OS due to dependencies
- Use OS update mechanism
- Flatcar update server (Nebraska) extended to support sysexts

OS images

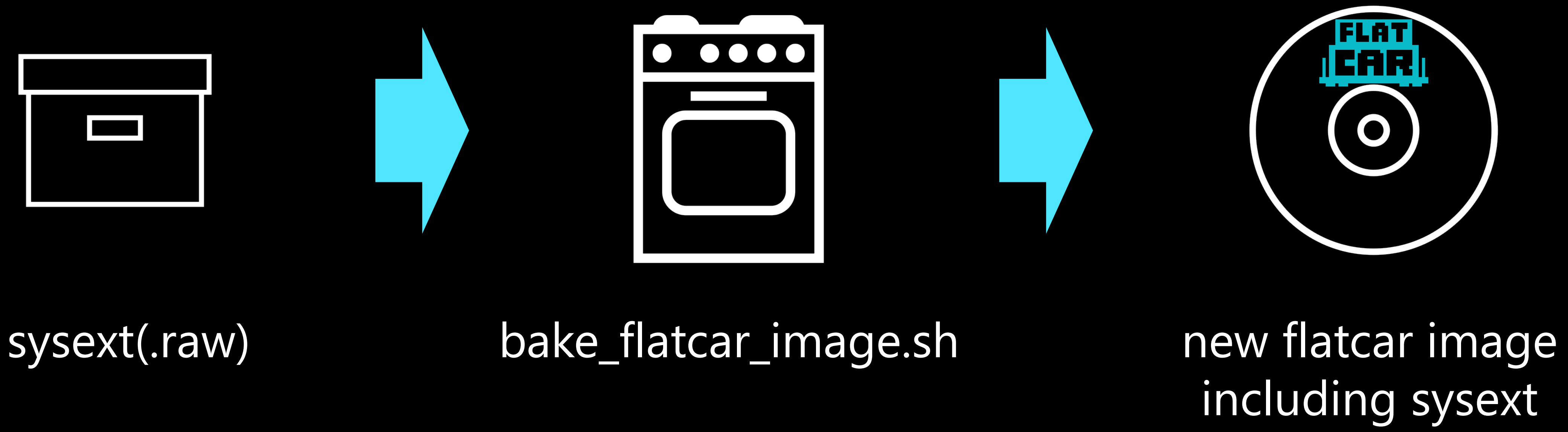
- Sysexts part of OS image → updated as part of OS update

Configuring for Updates of OS-independent Sysexts

YAML

```yaml
variant: flatcar
version: 1.0.0
storage:
  files:
    - path: /opt/extensions/wasmtime/wasmtime-17.0.1-x86-64.raw
      contents:
        source: https://github.com/flatcar/sysext-bakery/releases/download/latest/wasmtime-17.0.1-x86-64.raw
    - path: /etc/sysupdate.wasmtime.d/wasmtime.conf
      contents:
        source: https://github.com/flatcar/sysext-bakery/releases/download/latest/wasmtime.conf
  links:
    - target: /opt/extensions/wasmtime/wasmtime-17.0.1-x86-64.raw
      path: /etc/extensions/wasmtime.raw
      hard: false
systemd:
  units:
    - name: systemd-sysupdate.timer
      enabled: true
    - name: systemd-sysupdate.service
      dropins:
        - name: wasmtime.conf
          contents: |
            [Service]
            ExecStartPre=/usr/lib/systemd/systemd-sysupdate -C wasmtime update
    - name: sysext.conf
          contents: |
            [Service]
            ExecStartPost=systemctl restart systemd-sysext
```

What if I don’t want to pull the image at runtime?



```
// Create a qemu image (latest stable) with pre-baked wasmcloud
bake_flatcar_image.sh --fetch --vendor qemu_uefi wasmcloud:wasmcloud-0.82.0-x86-64.raw
```

https://github.com/flatcar/sysext-bakery?tab=readme-ov-file#baking-sysexts-into-flatcar-os-images

Putting it all together: Wasm-Optimized Linux

Wasm = Web Assembly

By default, provably secure sandbox

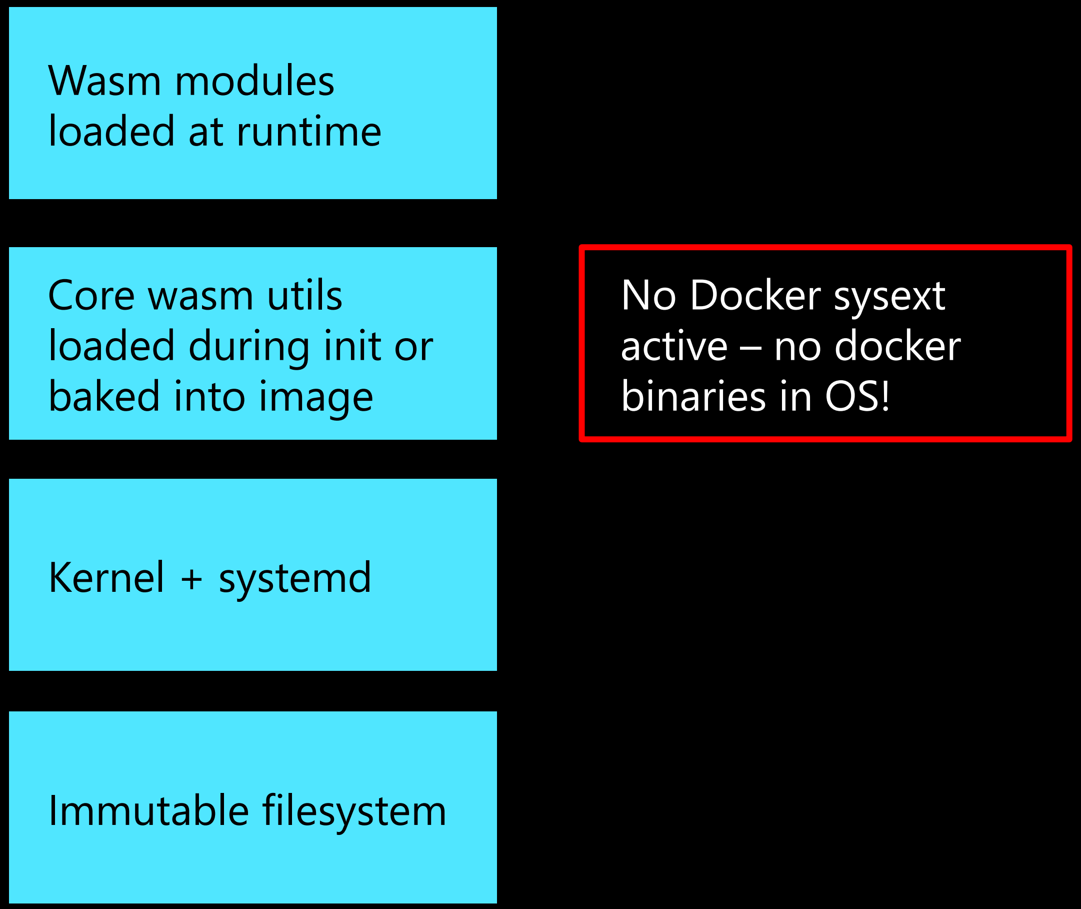
Most languages compile to it

Runs on most OSes, architectures

VERY small size, super fast start

Wasm *modules* run in a Wasm *runtime*

Putting it all together: Wasm-Optimized Linux


Wasm modules loaded at runtime
Core wasm utils loaded during init or baked into image
Kernel + systemd
Immutable filesystem
No Docker sysext active – no docker binaries in OS!

Worth noting that if you disable a sysext, the binaries disappear from the OS file system. Might be important for e.g. compliance.

Wasm-Optimized Linux

So many Wasm runtimes and tools to choose from

Lunatic

Modsurfer

nerdctl

runwasi

SpiderLightning (slight)

Spin

wamr (wasm-micro-runtime)

wasm3

WasmCloud

WasmEdge Runtime

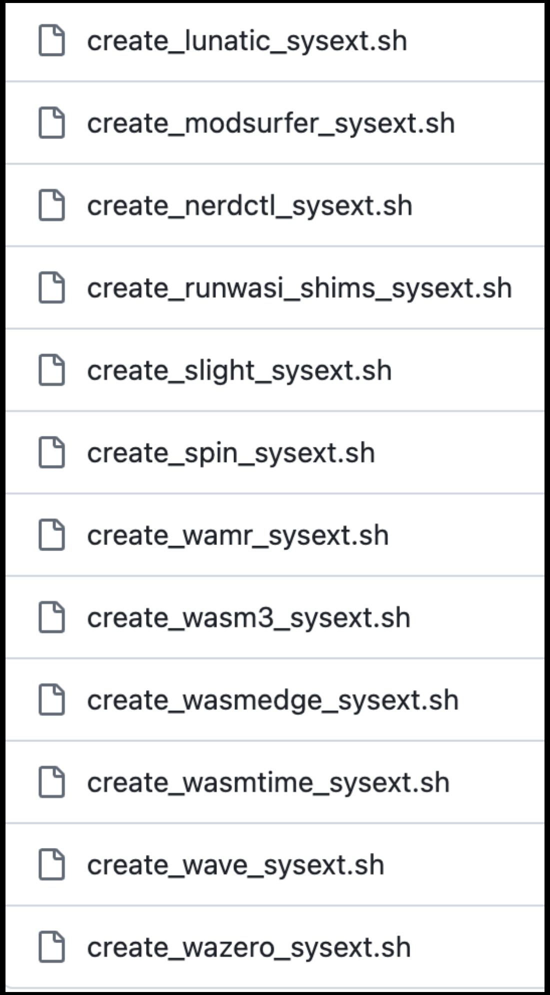
wasmtime

WaVe

WaZero

More Wasm Goodness in Ralph's Bakery

https://github.com/squillace/sysext-bakery



create_lunatic_sysext.sh
create_modsurfer_sysext.sh
create_nerdctl_sysext.sh
create_runwasi_shims_sysext.sh
create_slight_sysext.sh
create_spin_sysext.sh
create_wamr_sysext.sh
create_wasm3_sysext.sh
create_wasmedge_sysext.sh
create_wasmtime_sysext.sh
create_wave_sysext.sh
create_wazero_sysext.sh

This is a great playground for all: feel free to submit additional sysexts here or upstream Flatcar sysext-bakery for mature projects

Takeaways

[● ◀] systemd

- systemd-sysext is a promising new way to compose custom Linux distros
- Immutable + minimal (with all the benefits that brings), but also flexible

FLAT CAR

- Flatcar has already embraced as the way forward for enabling flexible deployments and customization
- Great platform for production environments for Wasm and more

Building an entire Flatcar release from scratch, just to add a new package like a different container runtime
Extend /usr with sysext images
imgflip.com

Get involved!

The Linux Userspace API (UAPI) Group

uapi-group.org

CNCF Special Purpose OS Working Group

tag-runtime.cncf.io/wgs/spos

Flatcar Container Linux Project

github.com/flatcar/Flatcar