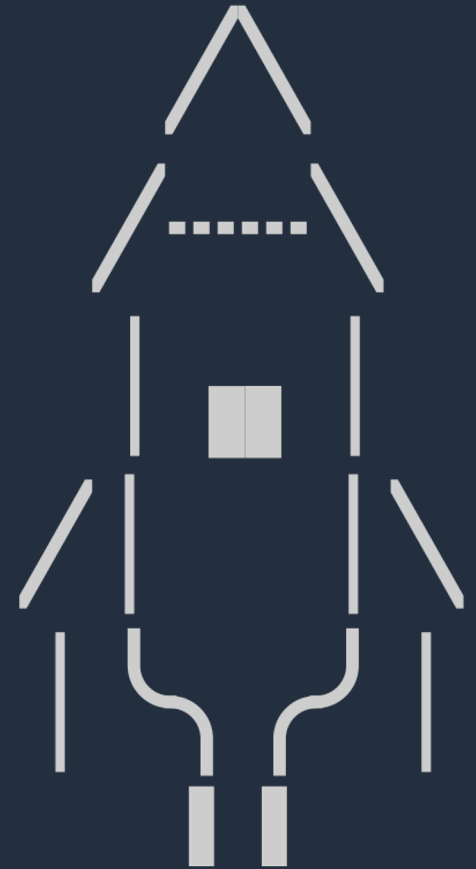


Bottlerocket

A container-optimized Linux.

Sean McGinnis - @smcginnis@fosstodon.org

<https://github.com/bottlerocket-os>



Agenda

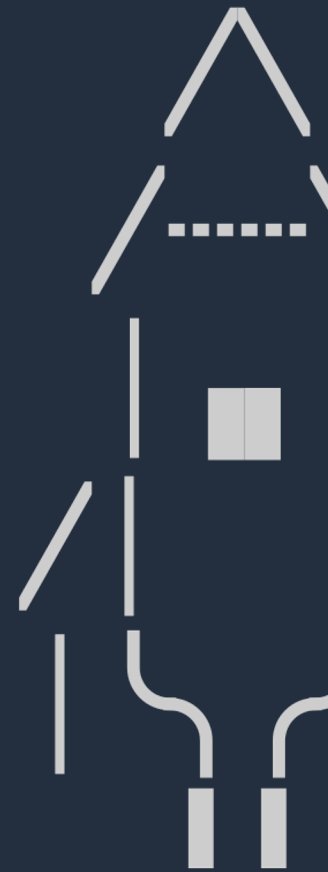
- What is a Container Optimized Linux
- Overview of Bottlerocket
- Bottlerocket Deep Dive
- Getting Involved
- Q&A

**Bottlerocket is a free and open source Linux-based
operating system meant for hosting containers**

<https://github.com/bottlerocket-os>

Container What?

<https://github.com/bottlerocket-os>



Challenges with General Purpose Distros



Mutable software configuration



Extra services not needed



Greater surface area – more security risk



Easy to become pets

Container Optimized

Only services needed to run containers

Reduced attack surface

Smaller footprint – faster boot times

Bottlerocket Goals

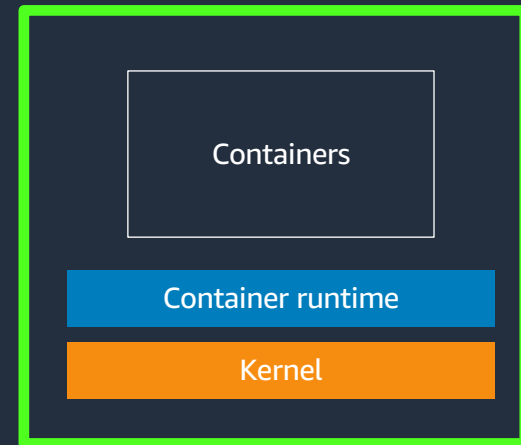
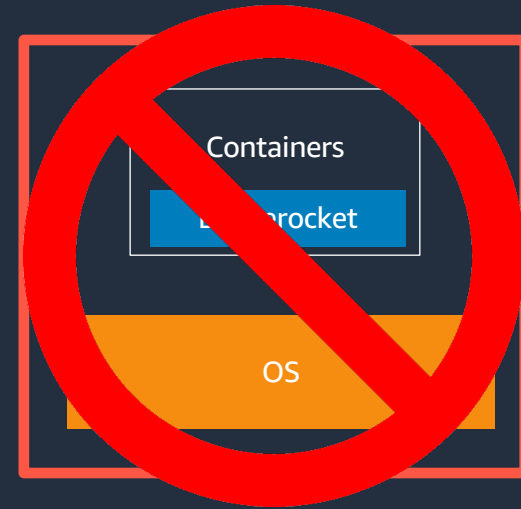
- Only essential software to run containers
- Small and fast with minimal bloat
- More secure by default
- Reduced maintenance overhead
- Open source

Bottlerocket Non-Goals

- General purpose, non-containerized computing
- An AWS-only solution
- Container base OS

Container Host OS

- Runs “outside” the container, not “inside”
- “Just enough” software to run containers
- Focus on scale – “cattle not pets”



Introducing Bottlerocket

AWS News Blog

Bottlerocket – Open Source OS for Container Hosting

by Jeff Barr | on 10 MAR 2020 | in [Containers](#), [Launch](#), [News](#) | [Permalink](#) | [Share](#)



Voiced by Amazon Polly

It is safe to say that our industry has decided that [containers](#) are now the chosen way to package and scale applications. Our customers are making great use of [Amazon Elastic Container Service \(Amazon ECS\)](#) and [Amazon EKS](#), with over 80% of all cloud-based containers running on AWS.

Container-based environments lend themselves to easy scale-out, and customers can run host environments that encompass hundreds or thousands of instances. At this scale, several challenges arise with the host operating system. For example:

Security – Installing extra packages simply to satisfy dependencies can increase the attack surface.

Updates – Traditional package-based update systems and mechanisms are complex and error prone, and can have issues with dependencies.

Overhead – Extra, unnecessary packages consume disk space and compute cycles, and also increase startup time.

Drift – Inconsistent packages and configurations can damage the integrity of a cluster over time.

Introducing Bottlerocket

Today I would like to tell you about [Bottlerocket](#), a new Linux-based open source operating system that we designed and optimized specifically for use as a container host.

Bottlerocket reflects much of what we have learned over the years. It includes only the packages

AWS Open Source Blog

Announcing the General Availability of Bottlerocket, an open source Linux distribution built to run containers

by Samartha Chandrashekar | on 31 AUG 2020 | in [Amazon EC2](#), [Announcements](#), [AWS Partner Network](#), [Compute](#), [Containers](#), [Open Source](#) | [Permalink](#) | [Comments](#) | [Share](#)

As our customers increasingly adopt containers to run their workloads, we saw a need for a Linux distribution designed from the ground up to run containers with a focus on security, operations, and manageability at scale. Customers needed an operating system that would give them the ability to manage thousands of hosts running containers with automation.

Meet Bottlerocket, a new open source Linux distribution that is built to run containers. Bottlerocket is designed to improve security and operations of your containerized infrastructure. Its built-in security hardening helps simplify security compliance, and its transactional update mechanism enables the use of container orchestrators to automate operating system (OS) updates and decrease operational costs.

Bottlerocket is developed as an open source project on [GitHub](#) with a public [roadmap](#). We're looking forward to building a community around Bottlerocket on GitHub and welcome your feature requests, bug reports, or contributions.

Bottlerocket technology

We began designing and building Bottlerocket based on the things we've learned from how customers use Amazon Linux to run containers and from running services such as AWS Fargate. At every step of the design process, we optimized Bottlerocket for security, speed, and ease of maintenance.

Announced – 10th March 2020

<https://aws.amazon.com/blogs/aws/bottlerocket-open-source-os-for-container-hosting/>

<https://github.com/bottlerocket-os>

GA – 31st August 2020

<https://aws.amazon.com/blogs/opensource/announcing-the-general-availability-of-bottlerocket-an-open-source-linux-distribution-purpose-built-to-run-containers/>

Bottlerocket Variants

To keep the footprint of Bottlerocket as small as possible for security and performance reasons, variants of Bottlerocket serve a particular use case, with the relevant software and API configuration.

[aws-k8s-
<1.21/1.22/1.23/1.24>](#)

Ships with Containerd and Kubelet.

Supports any Kubernetes platform running on AWS.

NVIDIA GPU variants.

[aws-ecs](#)

Ships with Docker Engine and ECS Agent

Supports Amazon ECS Container Instances.

NVIDIA GPU variants.

[vmware-k8s-
<1.21/1.22/1.23/1.24>](#)

Ships with Containerd and Kubelet

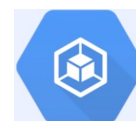
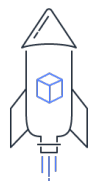
Supports Amazon EKS Anywhere running on VMware.

[metal-k8s
<1.21/1.22/1.23/1.24>](#)

Ships with Containerd and Kubelet

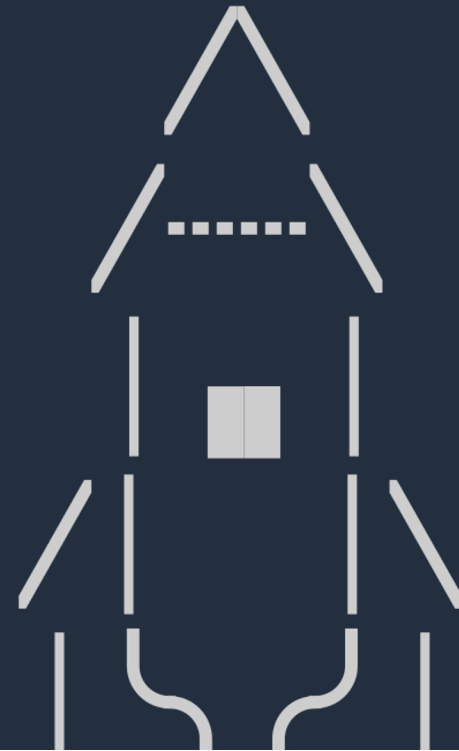
Supports Amazon EKS Anywhere running on Bare Metal.

Container Optimized Distros



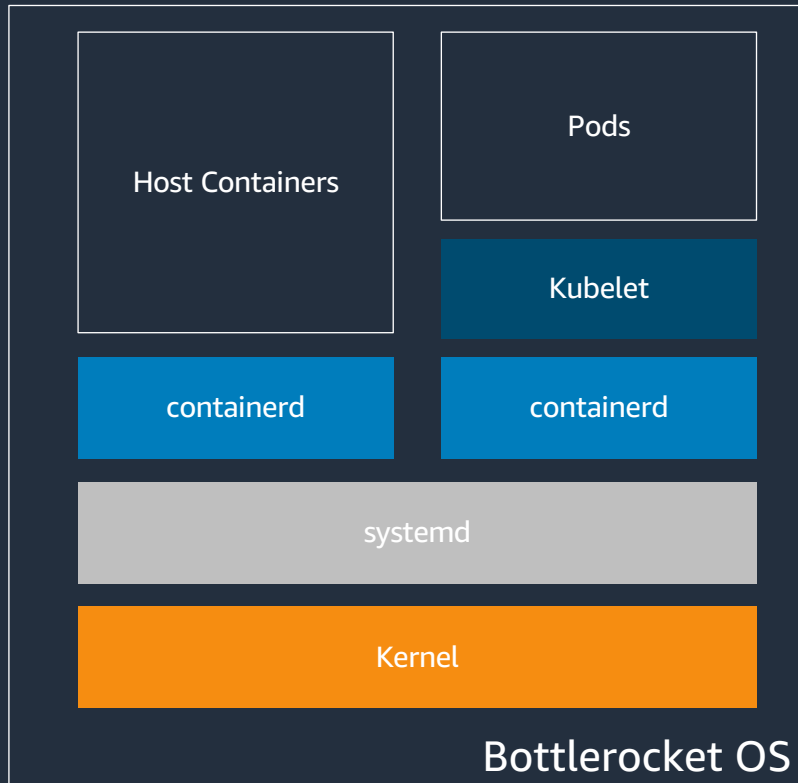
Bottlerocket	CoreOS	Flatcar	Container-Optimized Linux	Talos Linux
AWS	Fedora / Red Hat	Microsoft / Kinvolk	Google	Talos
Open Source	Open Source	Open Source	Open Source	Open Source
AWS VMware Bare Metal	AWS Azure GCP VMware OpenStack Bare Metal (more)	AWS Azure GCP VMware Bare Metal (more)	GCP	AWS Azure GCP VMware OpenStack Bare Metal (more)

Bottlerocket Architecture



<https://github.com/bottlerocket-os>

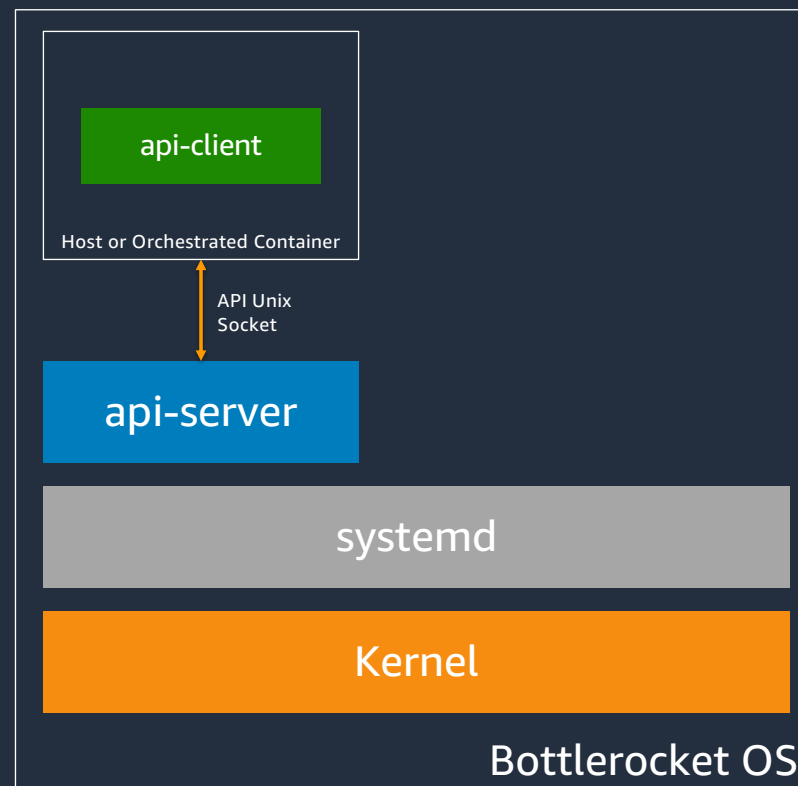
Host Containers



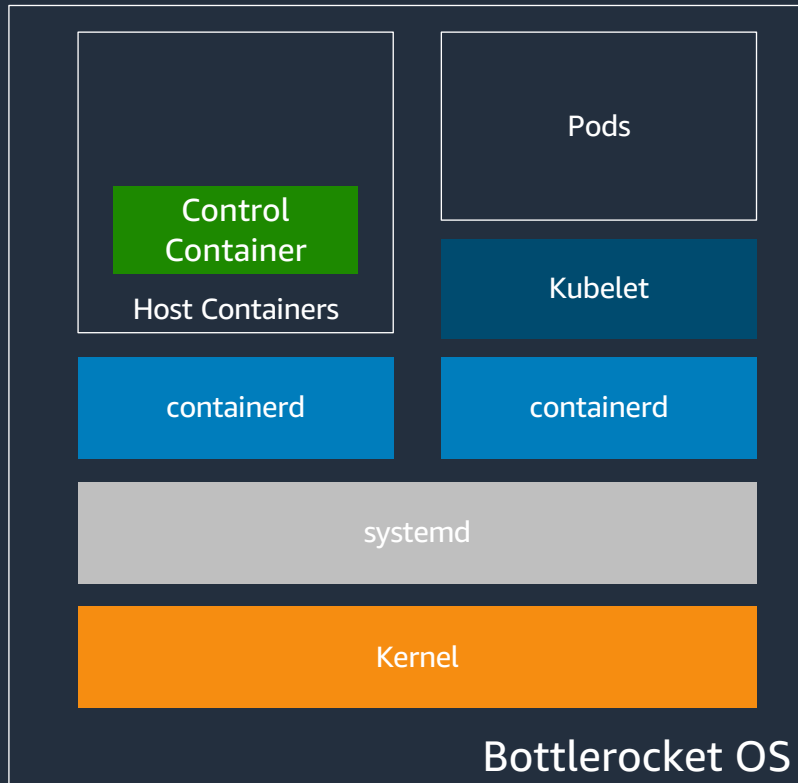
- Bottlerocket runs 2 completely separate container runtimes:
 - 1 for Orchestrator Containers (Kubernetes / ECS)
 - 1 for Host Containers (operational tasks)
- Providing Separate Fault Domains
- Different Security Profiles
- Could even be different runtimes

API Driven Configuration

- Bottlerocket is intended to be an API-first operating system - direct user interaction with Bottlerocket is usually through the API
- Boot Configuration (i.e. User Data) is loaded into the API Server.
- The API can make real time changes to the system configuration.

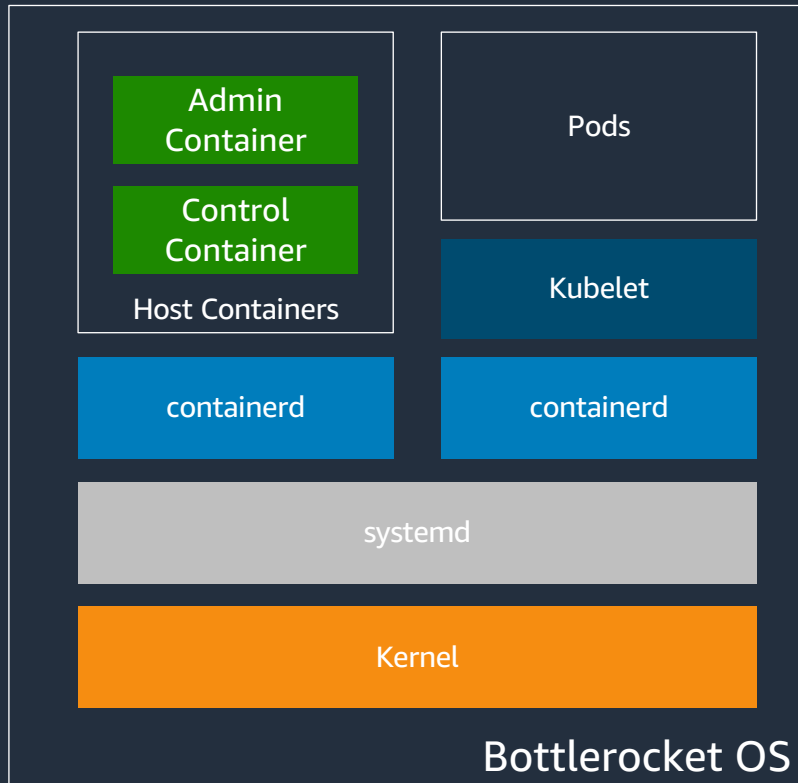


Host Containers – Control Container



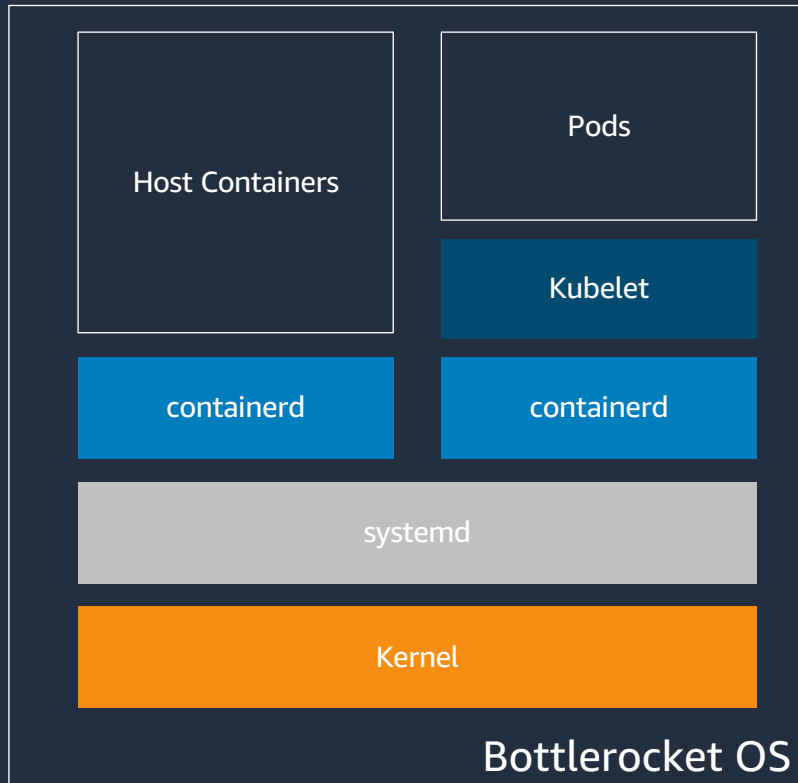
- The Control container is launched on boot
- The Control Container has the Bottlerocket API socket and API client mounted into the control container so you can configure the host

Host Containers – Admin Container



- Admin container is launched with additional privileges and can access the host process namespace
- Runs an SSH Server, access through the Hosts primary network interface
- Can break out of the Admin container to troubleshoot the host
- Should only be launched in exceptional circumstances to troubleshoot the host

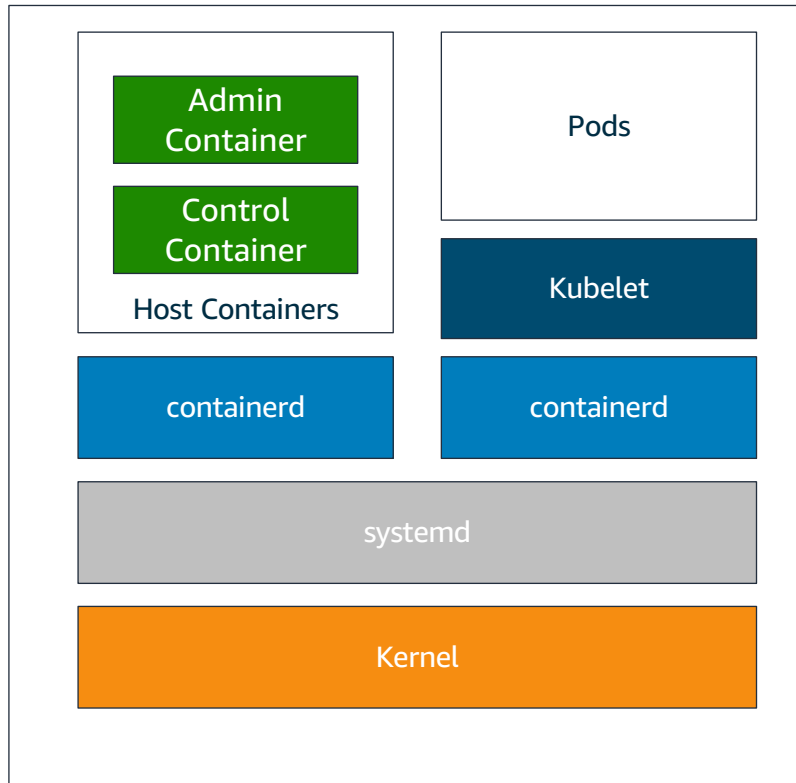
Bootstrap Containers



- Bootstrap containers are host containers that can be used to "bootstrap" the host before services like ECS Agent, Kubernetes, and Docker start
- Bootstrap containers have additional permissions, they have access to the underlying root filesystem and "CAP_SYS_ADMIN"
- Bootstrap containers can be set to run "always", "once" or "off"

Demo – System Access

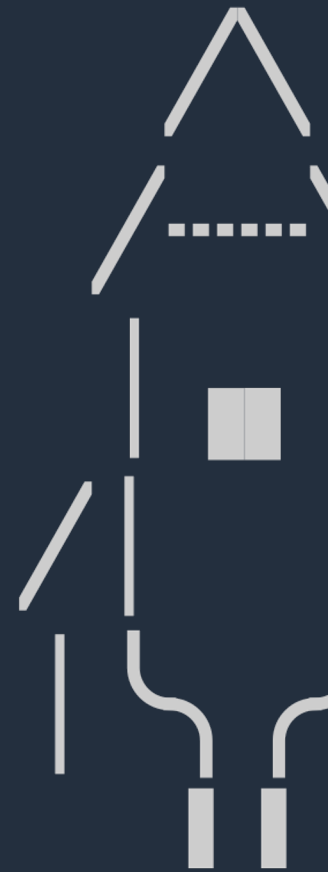
Demo - System Access



- Connection via console to the Control Container
- Access the Admin Container via `enter-admin-container`

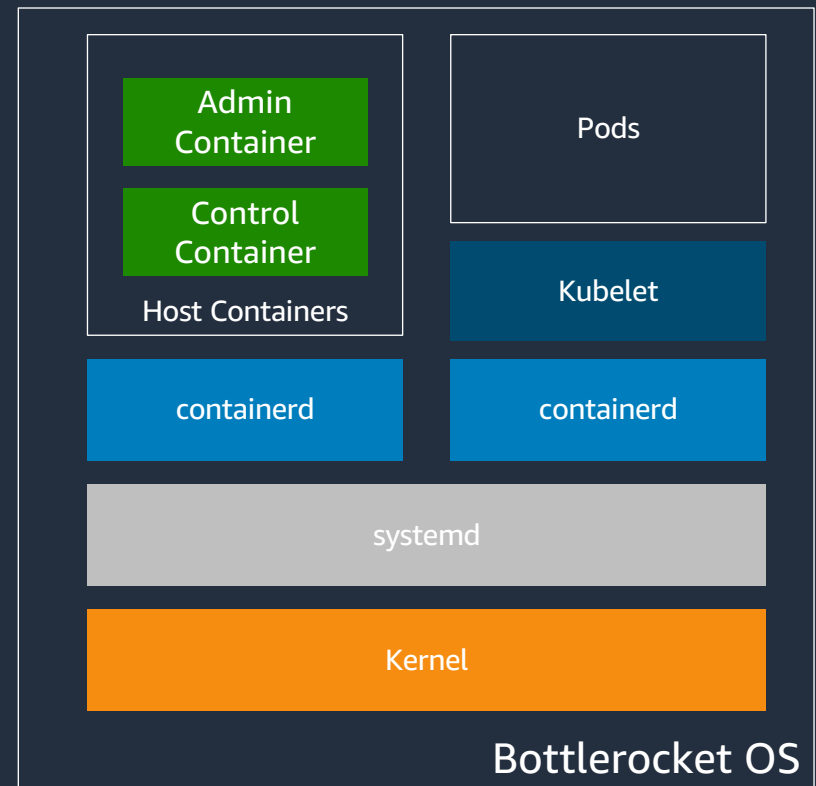
Securing Bottlerocket

<https://github.com/bottlerocket-os>



Bottlerocket's Minimal Footprint

- Bottlerocket has a very light footprint. It does not include a package manager, a shell, interpreters like python.
- If an attacker is able to escape the container they will have a lot less tools to work with.
- All additional debugging / troubleshooting tools or any additional monitoring tools should run in containers. Either Host Containers or Orchestrated Containers.



Read Only Root Filesystem

- Bottlerocket's root file system is read only
- The Bottlerocket OS image will be exactly the same every time it boots
- A separate User Partition is mounted for Container Images and Volumes
- /etc is backed by a tmpfs file system that is regenerated on boot

```
bash-5.0# findmnt /
TARGET SOURCE      FSTYPE OPTIONS
/         /dev/dm-0 ext4    ro,relatime,seclabel
bash-5.0# findmnt /etc/
TARGET SOURCE FSTYPE OPTIONS
/etc    tmpfs  tmpfs  rw,nosuid,nodev,noexec,noatime,context=system_u:object_r:etc_t:s0,mode=755
bash-5.0#
bash-5.0# touch /hello1
touch: cannot touch '/hello1': Read-only file system
bash-5.0#
```

Security Features

- dm-verity is used to check the integrity of the block devices.
 - Verifying that the read only filesystem is the same as the original hash tree of that file system. Checks that no one has been able to write data to that read only file system.
- Kernels boot in lockdown mode by default. This prevents the root user from modifying the kernel, increasing assurances that the running kernel corresponds to the booted kernel.
 - If you verify your boot chain but allow root to modify that kernel, e benefits of the verified boot chain are significantly reduced.

Enforced Permissions Boundary

Bottlerocket OS runs with SELinux in enforcing mode.

In the Container world SELinux can be used to protect files and directories from processes running in containers.

- Through the formal method of mounting files into containers
- Through informal methods of a container escape.



Compliance Features

Current

- CIS Benchmark available since August, 2022
- PCI DSS Compliance (FAQ has instructions on how to achieve)
- HIPAA-eligible feature authorized for use with regulated workloads for both Amazon EC2 and Amazon EKS

Roadmap

- FIPS certification
- FedRAMP compliance

Updating Bottlerocket

<https://github.com/bottlerocket-os>

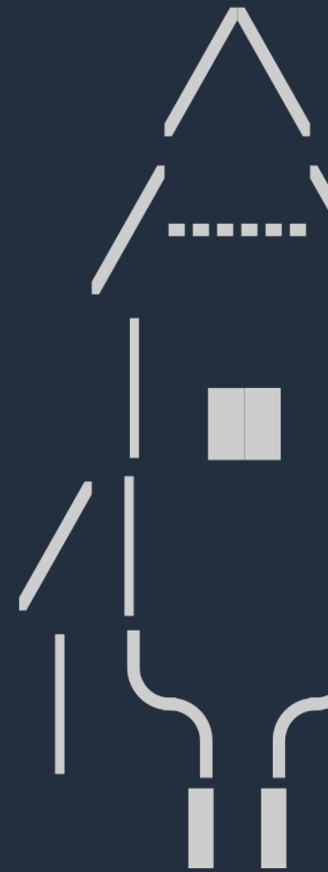
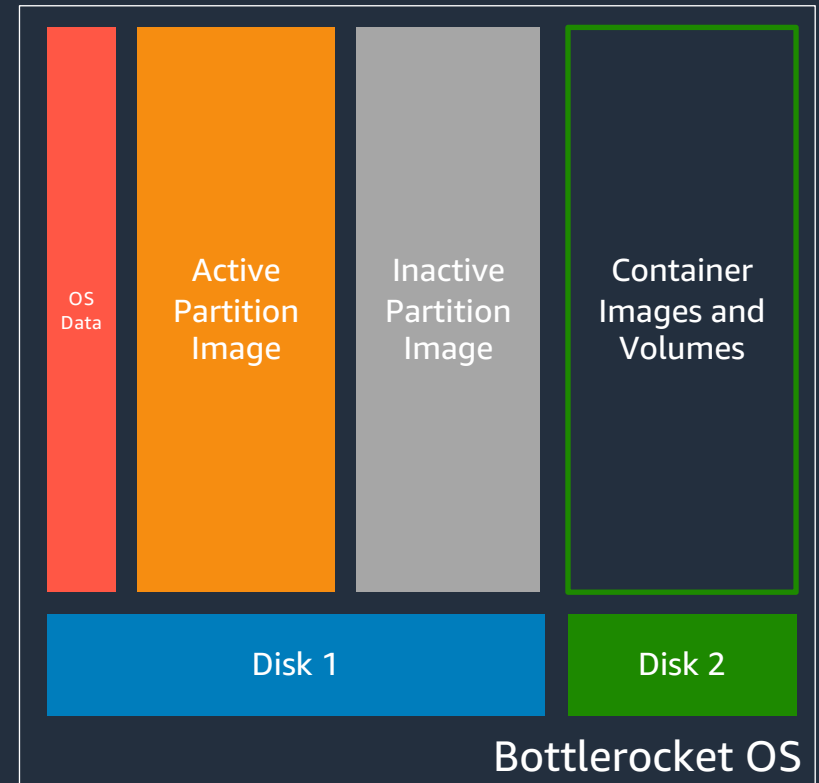


Image Based Updates

- Instead of a package update system, Bottlerocket uses an image-based model that allows for a rapid & complete rollback if necessary.
- All kernel, system packages, and container runtime packages will be stored inside the operating system image.
- All hosts in your fleet will be running a Bottlerocket image. No package drifts.



Active / Inactive Partitions

The first block device for Bottlerocket Hosts is an immutable device split into 2 halves:

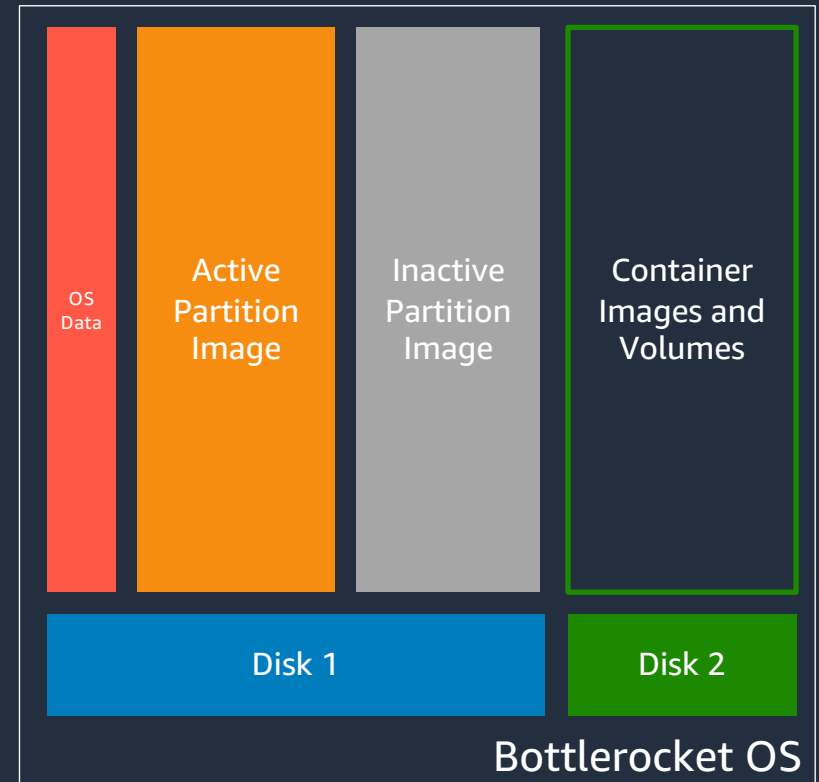
- An Active Partition
- An Inactive Partition

An upgraded Image is downloaded to the inactive partition.

Upon reboot the node will boot in to the previously inactive partition.

The previous image is still stored in the inactive partition for fast rollback if required.

OS Data stores settings and handles migrations.



Update Tooling

- apiclient
- updog
- Bottlerocket Update Operator ([brupop](#))
- Or just replace the nodes!

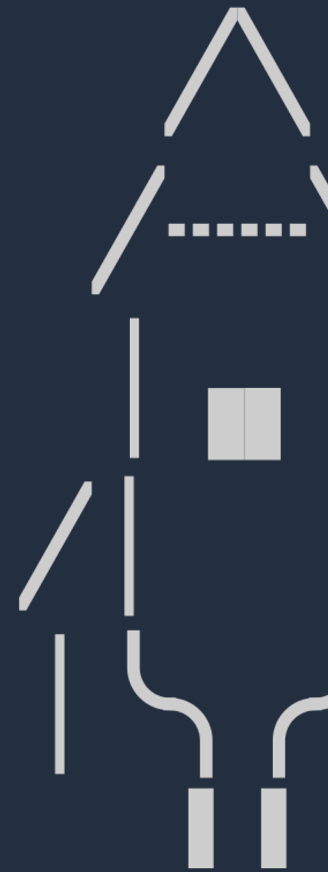
<https://github.com/bottlerocket-os/bottlerocket/blob/develop/sources/updater/README.md>

<https://github.com/bottlerocket-os>

Working With Bottlerocket

yaml and json and toml... oh my

<https://github.com/bottlerocket-os>



Configuration

YAML

```
bottlerocket:  
  settings:  
    kubernetes:  
      log-level: 2
```

JSON

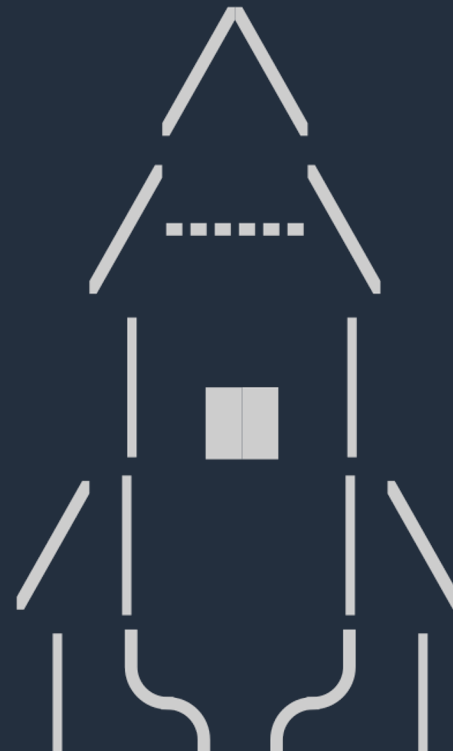
```
apiclient set -j '{"kubernetes": {"log-level": 2}'
```

TOML

```
[settings.kubernetes]  
"log-level": 2
```

<https://github.com/bottlerocket-os/bottlerocket#settings>

Thing to Consider



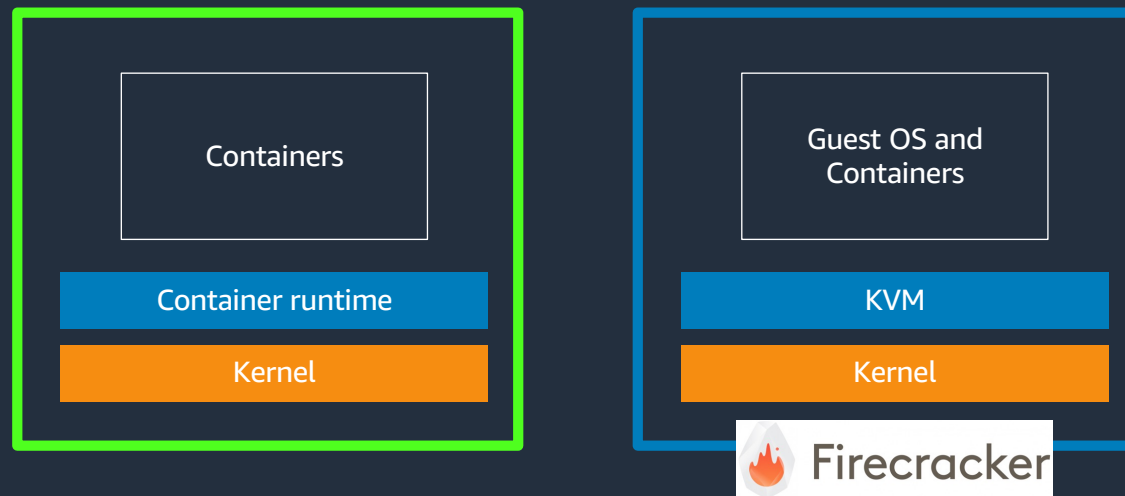
<https://github.com/bottlerocket-os>

Considerations when migrating from other distros

- All software running on Bottlerocket needs to be containerized
- No host agents can run on a Bottlerocket node
- Cannot install packages via package managers like yum or dnf
- Access mechanisms for a Bottlerocket host is different from traditional operating systems. You cannot directly SSH to a Bottlerocket worker node, but instead need to temporarily enable admin container to do so.

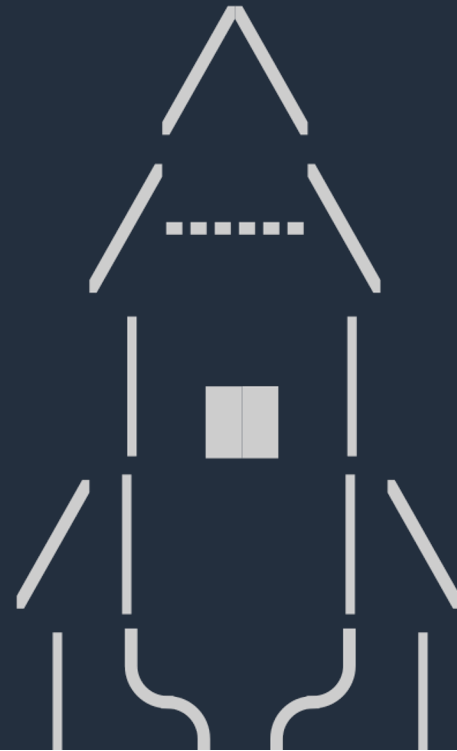
Bottlerocket versus Firecracker

- Both open source
- Different use cases and functionality



Getting Involved!

<https://github.com/bottlerocket-os>



Contributing to Bottlerocket

- All repos and source code* under <https://github.com/bottlerocket-os/>
- Source code dual licensed under Apache 2.0 and MIT
- Published roadmap - <https://github.com/orgs/bottlerocket-os/projects/18>

* some internal integration and release tooling are in private repos (for now!)

<https://github.com/bottlerocket-os>

Building Bottlerocket

Requirements

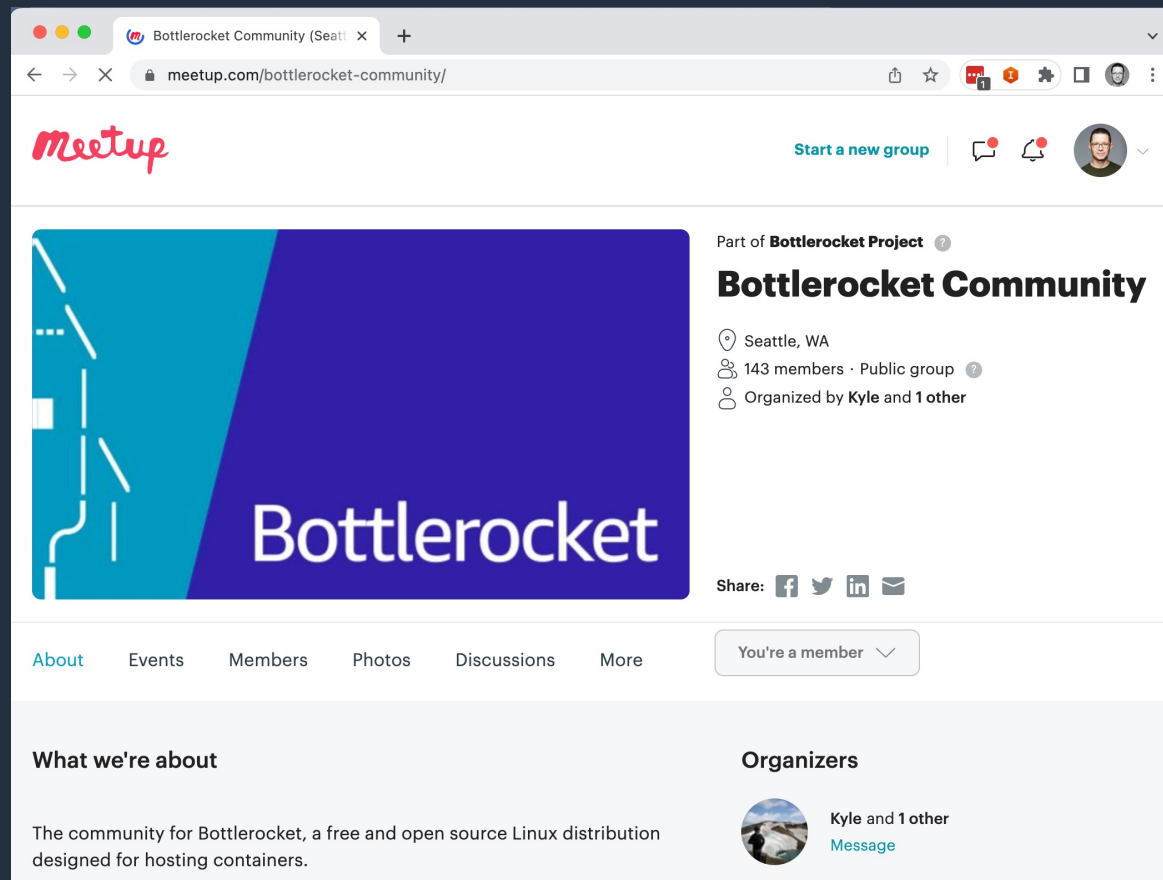
- Linux
- Rust (and cargo-make)
- Docker
- A “decent amount” of CPU, memory, and storage

<https://github.com/bottlerocket-os/BUILDING.md>

Building Bottlerocket Variants

The screenshot shows a web browser window displaying a GitHub issue page. The browser's address bar shows the URL `github.com/bottlerocket-os/bottlerocket/issues/2669`. The page header includes the GitHub logo, navigation links for Product, Solutions, Open Source, and Pricing, a search bar, and buttons for Sign in and Sign up. Below the header, the repository name `bottlerocket-os / bottlerocket` is shown as public, along with statistics for Notifications, Forks (391), and Stars (7.1k). A navigation bar highlights the Issues section with 215 items, alongside other repository features like Pull requests (6), Discussions, Actions, Projects (1), and Security (268). The main content area features the issue title `Enable out-of-tree Bottlerocket variants #2669` and a `New issue` button. A green `Open` badge indicates the issue status, with a note that `cbgbt` opened it on Dec 16, 2022, and it has 1 comment. A comment from `cbgbt` is visible, dated Dec 16, 2022, with a `Contributor` label. The comment text reads: `What I'd Like: A supported mechanism for creating custom Bottlerocket variants that live within their own git trees. These variants should support some mechanism for defining a dependency on Bottlerocket's core in a way that allows the custom variant to evolve`. To the right of the comment, an `Assignees` section lists `cbgbt` and `webern`. A `Labels` section is partially visible at the bottom.

Bi-Weekly Community Meetings



The screenshot shows a web browser window displaying the Bottlerocket Community page on Meetup.com. The browser's address bar shows the URL `meetup.com/bottlerocket-community/`. The Meetup logo is in the top left, and a "Start a new group" button is in the top right. The main content area features a large blue and purple graphic with the word "Bottlerocket" in white. To the right of the graphic, it says "Part of Bottlerocket Project", "Bottlerocket Community", "Seattle, WA", "143 members · Public group", and "Organized by Kyle and 1 other". Below this is a "Share:" section with icons for Facebook, Twitter, LinkedIn, and Email. A navigation bar includes links for "About", "Events", "Members", "Photos", "Discussions", and "More", along with a "You're a member" button. The "What we're about" section contains the text: "The community for Bottlerocket, a free and open source Linux distribution designed for hosting containers." The "Organizers" section shows a profile picture and the text "Kyle and 1 other" with a "Message" link.

<https://github.com/bottlerocket-os>

<https://www.meetup.com/bottlerocket-community/>

Thank You

Any Questions?

Sean McGinnis - @smcginnis@fosstodon.org

<https://github.com/bottlerocket-os>

