Fedora CoreOS News & How Columbia University Uses Fedora CoreOS

Dusty Mabe
Principal Engineer at Red Hat

Mark Pusey
Chief of Software Engineering & Architecture at Columbia University
Agenda

- What is Fedora CoreOS
- Fedora CoreOS News
- How FCOS is used to build FCOS
- How Columbia University uses Fedora CoreOS
What is Fedora CoreOS?
Components of Fedora CoreOS

- Automatic updates
- Automated provisioning
- “Immutable” OS with transactional updates
- Your software runs in containers
None of this happens without...

Passing Tests!

<table>
<thead>
<tr>
<th>Repository</th>
<th>Time</th>
<th>Issue #</th>
<th>Time</th>
<th>Issue #</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>kola-aws</td>
<td>13 hr</td>
<td>#325</td>
<td>4 days</td>
<td>#312</td>
<td>1 hr 11 min</td>
</tr>
<tr>
<td>kola-azure</td>
<td>13 hr</td>
<td>#80</td>
<td>12 days</td>
<td>#57</td>
<td>45 min</td>
</tr>
<tr>
<td>kola-gcp</td>
<td>13 hr</td>
<td>#169</td>
<td>3 days</td>
<td>#164</td>
<td>15 min</td>
</tr>
<tr>
<td>kola-kubernetes</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
<tr>
<td>kola-openstack</td>
<td>13 hr</td>
<td>#340</td>
<td>1 day 10 hr</td>
<td>#337</td>
<td>45 min</td>
</tr>
</tbody>
</table>
Multiple Update Streams

Fedora CoreOS is available across 3 different release streams:

**Stable**
- v 35.20220424.3.0
- JSON — 2 days ago

The Stable stream is the most reliable version of Fedora CoreOS. Releases are battle-tested within the Testing stream before being promoted.

**Testing**
- v 36.20220505.2.0
- JSON — 2 days ago

The Testing stream contains the next Stable release. Mix a few Testing machines into your cluster to catch any bugs specific to your hardware or configuration.

**Next**
- v 36.20220507.1.0
- JSON — 2 days ago

The Next stream represents the future. It provides early access to new features and to the next major version of Fedora. Run a few Next machines in your cluster, or in staging, to help find problems.
Supported platforms and architectures

- Available for a plethora of **cloud/virt platforms**
  - Alibaba, AWS, Azure, DigitalOcean, Exoscale, GCP, IBM Cloud, OpenStack, Vultr, VMware, QEMU/KVM
  - Directly launchable on AWS & GCP

- Several options for **Bare Metal**
  - Live ISO (automated or interactive installations)
  - PXE (network) boot
  - Raw and 4K native disk images

- Multiple Architectures (**x86_64, aarch64, s390x**)
  - **ppc64le** support coming soon
What’s new in Fedora CoreOS?
New Website

The container optimized OS
A minimal OS with automatic updates. Scalable and secure.
The container optimized OS

A minimal OS with automatic updates. Scalable and secure.
CountMe Stats - All Nodes

![Graph showing transient and static node statistics over time from January 1, 2023, to May 1, 2023.]
CountMe Stats - Fedora Release

All Nodes

- 34
- 35
- 36
- 37
- 38
CountMe Stats - Fedora Release

Static Nodes
- 34
- 35
- 36
- 37
- 38

Transient Nodes
- 34
- 35
- 36
- 37
- 38
CountMe Stats - Architecture (2022/08)

- ppc64le
- s390x
- aarch64
- x86_64
All Nodes
- ppc64le
- s390x
- aarch64
- x86_64
Kubevirt Platform

Repositories

fedora / fedora-coreos-kubevirt

Repository Tags

<table>
<thead>
<tr>
<th>TAG</th>
<th>LAST MODIFIED</th>
<th>SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>next</td>
<td>a day ago</td>
<td>N/A</td>
</tr>
<tr>
<td>testing</td>
<td>a day ago</td>
<td>N/A</td>
</tr>
<tr>
<td>stable</td>
<td>a day ago</td>
<td>N/A</td>
</tr>
</tbody>
</table>
KubeVirt Platform

Project: fedora-coreos-pipeline

VirtualMachineInstances  ➔ VirtualMachineInstance details

VMI my-fcos ▶ Running

Details  YAML  Scheduling  Events  Console  Network interfaces  Disks

Guest login credentials ➔

VNC console  ➔  Send key  ➔

Fedora CoreOS 38.20230514.3.0
Kernel 6.2.15-300.fc38.x86_64 on an x86_64 (tty1)

SSH host key: SHA256:RXV/duVcPeJoaQN14Cq6YcKo+rczn9eck5P9WUBsPM (ECDSA)
SSH host key: SHA256:HybOk91C/LwMh6OJv2rXq1FjJa5m03JeMbCccs (ED25519)
SSH host key: SHA256:MyJvyGBQH5wUmXbKJL0HtWfNAGHNTtTgdxAkiU (RSA)

enp1s0: 10.0.0.2.2 fe80::1c55:8521:e370:3fa
Ignition: ran on 2023-06-01 16:22:34 UTC (this boot)
Ignition: user-provided config was applied
Ignition: wrote ssh authorized keys file for user: core
my-fcos login: _
CoreOS Layering / OSTree Native Containers

- Fedora CoreOS OSTree is additionally offered as a container
- Customize Fedora CoreOS by performing a container build
  - FROM: quay.io/fedora/fedora-coreos:stable
- Makes individual derivation and distribution easier
  - Dockerfile & Container registry
# This is like https://tailscale.com/download/linux/fedora
# except it happens as part of a container build! You then need to do
# `tailscale up` via some other mechanism.
FROM quay.io/fedora/fedora-coreos:stable
    rpm-ostree install tailscale && \
    systemctl enable tailscaled && \
    ostree container commit

- https://github.com/coreos/coreos-layering-examples
CoreOS Layering

- `podman build -t myfcos:latest .`
- `podman push myfcos:latest quay.io/$USER/myfcos:latest`
- `rpm-ostree rebase --experimental \ ostree-unverified-registry:quay.io/$USER/myfcos:latest`
More Platform Enablement

- Added aarch64 Azure images
  - [https://github.com/coreos/fedora-coreos-pipeline/pull/694](https://github.com/coreos/fedora-coreos-pipeline/pull/694)
- Added SEV support for GCP images
  - [https://github.com/coreos/fedora-coreos-tracker/issues/1202](https://github.com/coreos/fedora-coreos-tracker/issues/1202)
And.. A lot of boring stuff

- Time Spent...
  - Unifying upstream and downstream build pipelines
  - Automation
  - CI
How does Fedora CoreOS use Fedora CoreOS?
Helper Services

- The **Archive Repo Manager**
  - Watches Bodhi, stores updates in an Archive Repo
  - uses FUSE, so we run the service in a VM
    - needs extra configuration inside Kubernetes
  - We use FCOS for the VM
    - Easy to provision from scratch ([Butane/Ignition](Butane/Ignition))
    - Automatically updates itself
OpenShift/RHCOS

- The Fedora CoreOS Build pipeline runs on OpenShift
  - Offered by Fedora Infrastructure
- RHEL based RHCOS is the foundation of OpenShift
  - downstream of Fedora CoreOS
Multi-Arch Builders

- Fedora Infra OpenShift is x86_64 only
- Need to do native aarch64, ppc64le, s390x builds
- 👉 Use Fedora CoreOS
  - Jenkins from x86_64 pipeline
  - Talks to FCOS aarch64, ppc64le, s390x builders
    - Uses `podman --remote` for the builds
    - Automatically update on a schedule
    - Easy to provision via Ignition/Butane
How does Columbia University use Fedora CoreOS?
Get involved!

- Web: [https://getfedora.org/coreos](https://getfedora.org/coreos)
- Issues: [https://github.com/coreos/fedora-coreos-tracker/issues](https://github.com/coreos/fedora-coreos-tracker/issues)
- Forum: [https://discussion.fedoraproject.org/tag/coreos](https://discussion.fedoraproject.org/tag/coreos)
- Mailing list: coreos@lists.fedoraproject.org
- IRC: Libera.chat #fedora-coreos
- Other talks to get started:
  - [Fedora CoreOS Introduction (Jul 13, 2020)](https://example.com)
  - [Getting Started with Fedora CoreOS (Mar 17, 2021)](https://example.com)
PRECISION DENTAL MEDICINE

Columbia University Irving Medical Center
College of Dental Medicine
Agenda

• Business Domain Overview (10 mins)
• Tech Domain Overview (15 mins)
• Demo: Provisioning $\texttt{fedora COREOSes}$ via terraform (10 mins)
• Summing Up & Questions (10 mins)
Business Domain Objectives

• Providing best-in-class quality dental care to our local community patients (The Heights in Manhattan, NYC)
• Graduation of highly skilled dentists, trained in a joint medical/dental electronic health record environment, and prepared for a lifetime of learning.
• Support research by faculty and students into dental health conditions with the view to improving dental care globally.
Connected Dental Chairs - Features

Wide angle camera for viewing and recording operatory environment.

Patient biometrics port for logging of interchangeable device outputs.

RFID tracking at patient head to record utilization patterns of non-tethered RFID-tagged instruments and supplies.

Light head-integrated close angle camera for viewing and recording treatment session.

Millisecond logging of tethered hand piece utilization.

Logged identification of patient and provider.

Chair sensor to determine time of patient seating and departure.
Capabilities – Data Environment

- Dashboards allow review of in-progress or completed dental procedures.
- Electronic health record, dental chair data and video data is merged together.
- Drill down on machine data with synchronized video clips allows student dental techniques to be reviewed.
- **Open-Source Technology:** Keycloak, Grafana, Prometheus, Graphite, Percona XtraDB MySQL Cluster, RabbitMQ & TensorFlow ModelServer all running on Fedora CoreOS largely on on-premises vSphere.
Capabilities – Video-based Faculty Oversight

- Faculty touchdown stations provide video overviews of 48 operatory workstations.
- Widefield and in-mouth video camera views are available (96 separate 4K streams).
- Real time text overlays (in blue) summarize key information like provider (student) name + drill / curing light usage.
- In the future we expect to alert on poor student posture based on AI inference results.
- **Technology**: Synology Surveillance Station with Annotations added by custom microservices running on *fedora* COREOS
Capabilities – Data Science Education

• In 2022 for the first time 250 first-year dentists/medics studied data science as part of their core curriculum.

• Open-Source Technology: JupyterHub on a Typhoon Kubernetes cluster powered by fedora COREOS on AWS.
Capabilities – Research / Data Science

- JupyterHub environments are supported by Spark clusters facilitating research and data science.
- A data lake combines dental encounter clinical instrument/video data & select anonymized data.
- Many hypotheses to be tested, but automated educational recommendations for individual students could be an end goal.
Technology Domain Overview

- **Objective:**
  - Deliver application solutions for our business domain

- **Scale (Small):**
  - 3 person DevSecOps team deploying apps and infrastructure via ‘gitops’ pipelines
  - ~40 Fedora CoreOS servers running during business hours (~30 on prem, ~10 AWS)
  - ~40 git app projects deploying as docker services / podman units (+ ~10 shared libraries)
  - ~12 git projects that result in deployed infrastructure (and another ~4 libraries)
  - Started in Q4 2018 on CoreOS and have long since migrated to Fedora CoreOS

- **Methodology:**
  - Iterative and embracing / leaning into change. Running Fedora CoreOS next stream in dev.
  - Mitigating risk by running dev/test/prod parity and continuous canary-based monitoring.
  - Application pipelines re-build/patch/deploy to test nightly.
  - Some infra auto-redeploys too - e.g. Fedora CoreOS GPU Node AWS ASGs.
Software Architecture &

Sources
- Docker
- RFID Software
- MySQL
- Digital Chair Software
- Digital Chair Database
- Network Storage

Protocols
- AMQP Messaging
- HTTP WebAPI

Ingestion
- Docker Swarm
- 20+ custom microservices + apps
- 8+ open source components*
- 5+ open source observability components**

Storage
- Network Storage
- EAR DB
- CDM Data Lake (Columnar Objects)

Analytics
- Power BI Dashboarding

* Keycloak, HCP Vault, Traefik, RabbitMQ, MariaDB, Percona MySQL, Spark, JupyterHub, Tensorflow ModelServer, Triton ModelServer
** Grafana, Graphite, Prometheus, AlertManager, Loki, Various Exporters
Developers iterate on code rapidly locally receiving feedback from linting, code complexity scanning and unit tests. Code is then deployed to local containers on single node swarms or via development clusters.

A commit, or daily schedule, triggers a GitLab Runner to build new docker image(s), pulling new security updates, and tagging build with Git commit identifier and image SHA ID.

The built docker image is pushed to the registry and static container security scanning using trivy/grype is performed.

The GitLab Runner automatically deploys zero vulnerability images built from specific branches to specific docker swarms. Secret injection is via Vault.

Developers iterate on code rapidly locally receiving feedback from linting, code complexity scanning and unit tests. Code is then deployed to local containers on single node swarms or via development clusters.

The built docker image is pushed to the registry and static container security scanning using trivy/grype is performed.

The GitLab Runner automatically deploys zero vulnerability images built from specific branches to specific docker swarms. Secret injection is via Vault.
• Provision node via Terraform on vSphere
  - Deploy node using common org-specific module (e.g. configures for certificated SSH)
  - Deploy node again (no terraform change detected to node – no changes made)
  - Make an ignition change & redeploy (node is torn down and replaced)

• Terraform Details:
  - The poseidon/ct provider is used to render templated butane config into ignition files.
  - Templates are parameterized to abstract environment specifics.
• Platform (👍): stable, self-updating, minimal (more secure) and container-focused with strong support for Infrastructure as Code through ignition.

• Community (❤️): Engagement has been almost universally positive. Super smart people eager to help. And just like with my kids’ soccer I’ve got way more out of it than I ever expected...
Heavily Influenced By…

And…
Appendix A: Software/Infra Deployment

Developer can build/deploy scratch builds to development

Developer push to development branch targets dev env.

Developer push to test branch targets test environment

Developer push to main branch targets prod environment
• A canary process runs continuously performing a set number of health checks on any of our lcl, dev, tst or prod cluster environments.
• These health checks output issues to stdout but also control Luxafor flags to act as a visible health monitor.
• This is a key feedback loop mechanism to improve alerting bringing problem discovery temporally closer to root cause - a key objective of DevOps.
• Health check selection is key. Measuring a final output is better than trying to measure intermediate steps.
• Locating the canary is key. It needs to be unobtrusively in the line of sight.
• No, a process that emails you is not a canary. The canary is a publicly visible ‘information radiator’ that self-corrects when issues are resolved.