Speeding up DNF and RPM using Copy on Write

CentOS Dojo, FOSDEM, 2021

Matthew Almond

Production Engineer

Operating Systems

FACEBOOK

Agenda

1. Context
2. DNF and RPM with Copy on Write

Presnet

3. Reuse Local Extents
4. Packed Object Repositories
5. Better Images

Future

1. But Why?

USE THE CLOUD

THERE IS NO CLOUD

THERE IS A CLOUD

AND IT IS ON OUR COMPUTER

What is on our computer?

Services running in containers

Per physical host software

- Container Runtime
- Log Aggregation
- Hardware Monitoring
- Service Selection
- ...

Underneath everything is CentOS

Mutable, Managed Operating System

- (Fairly) minimal CentOS Server installation
- Periodic convergence using Chef
  - Install/Upgrade/Remove RPM packages
  - Write configuration files
  - (Re)start services
  - ...

Contrast: Immutable Operating Systems?

- Multiple concurrent release processes
  - Immutable forces a small number of states, e.g. “stable” and “release candidate”
  - **There’s no such thing as “stable”**
  - **Many small changes**
- Minimise service downtime
  - Read only Operating Systems force “reboot” as a strategy
  - **Services benefit from caches in RAM**
- Consider CVE-2021-3156 “Sudo before 1.9.5p2 has a Heap-based Buffer Overflow, allowing privilege escalation to root via "sudoedit -s" and a command-line argument that ends with a single backslash character.”
  - Reboot the world?
  - **`dnf upgrade sudo`**

Using DNF in Production

Orchestrated using Chef while primary services are running

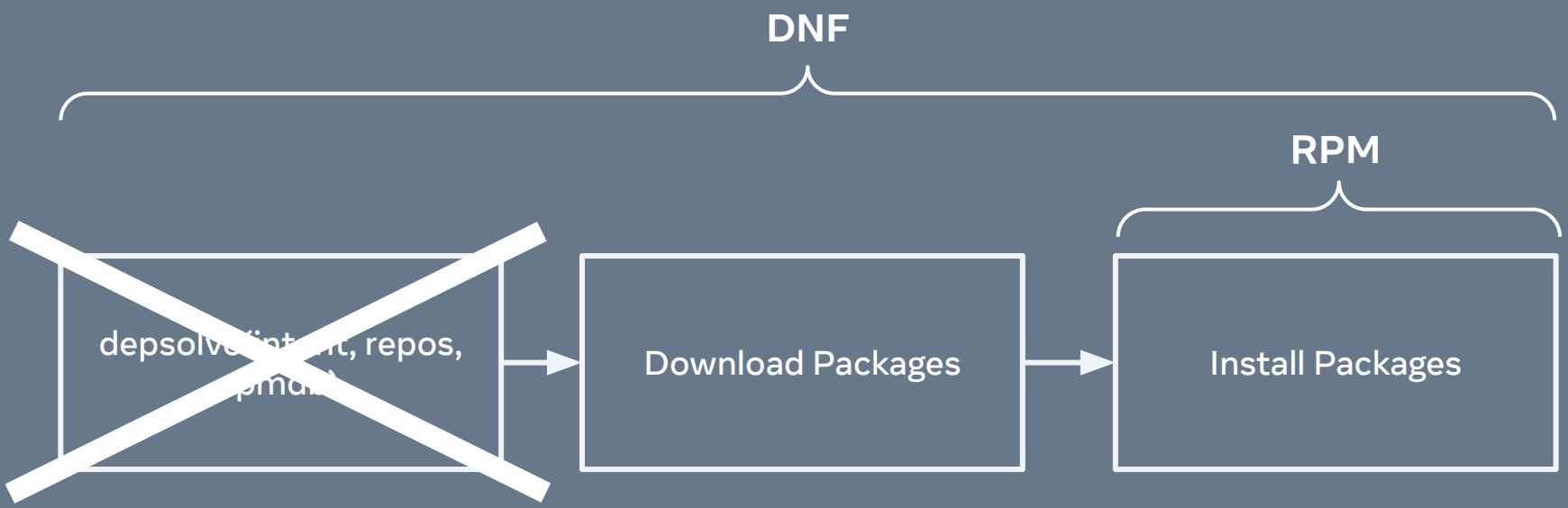
- I/O Contention!
- Deadlines!

2. DNF and RPM with Copy on Write

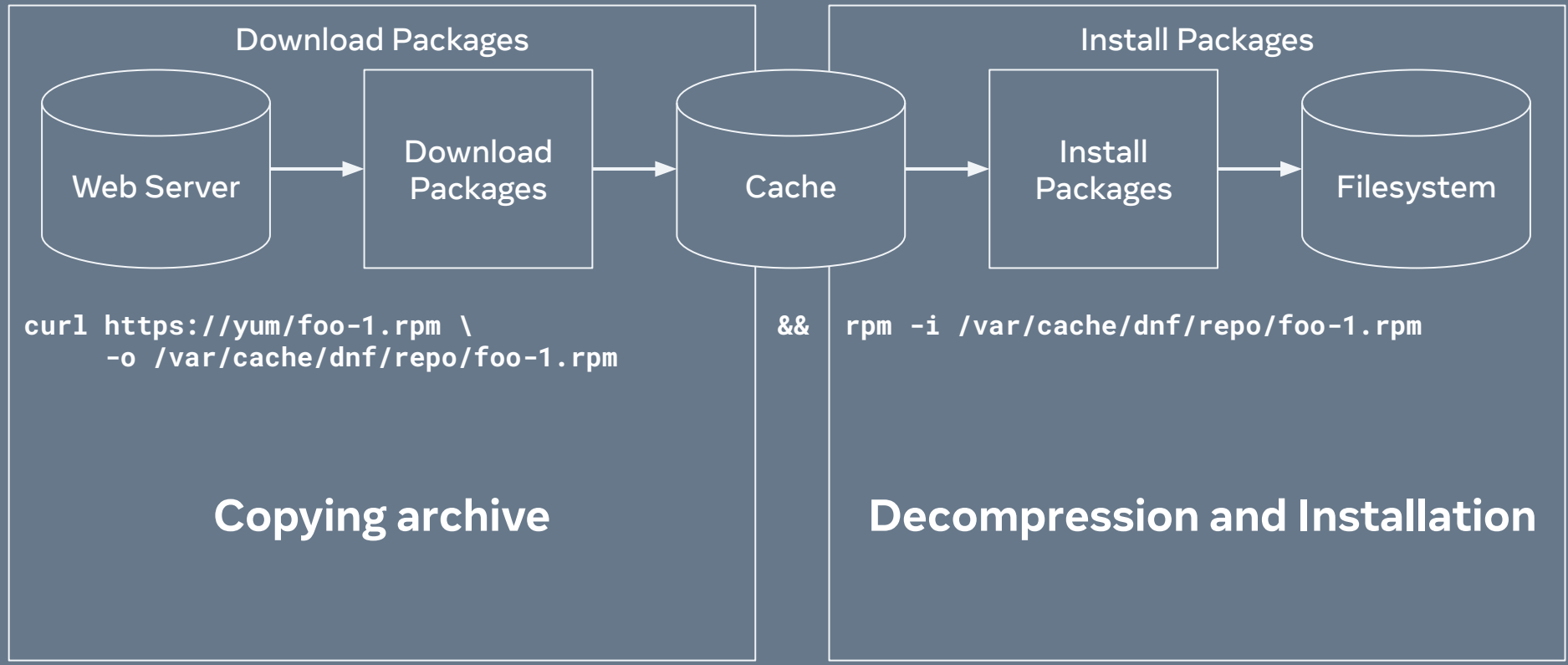
A 3,048 meter view of package installation and Copy on Write (CoW)

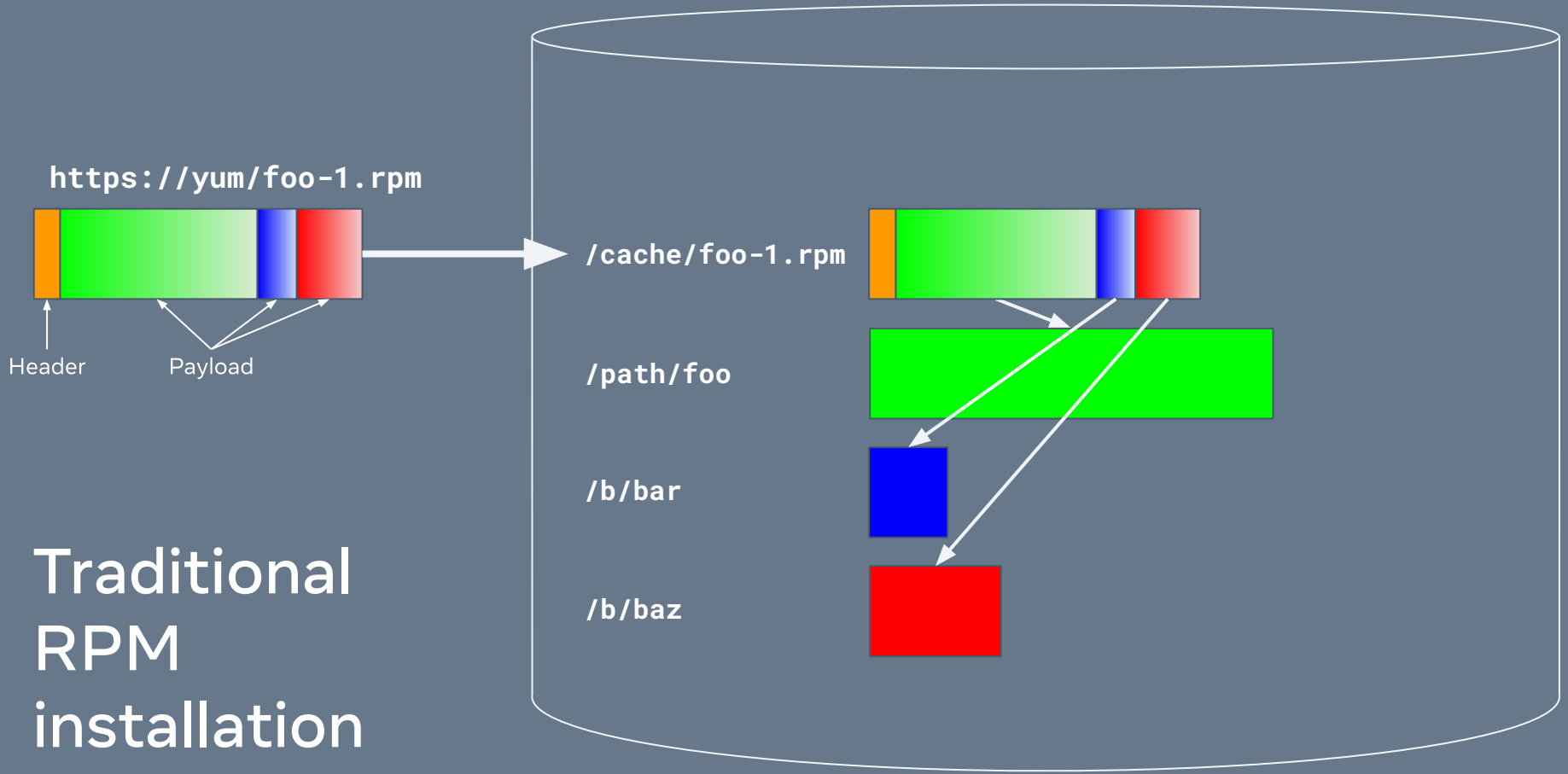
dnf install hello

DNF & RPM, the 3,048 meter view



DNF & RPM, the 1,524 meter view



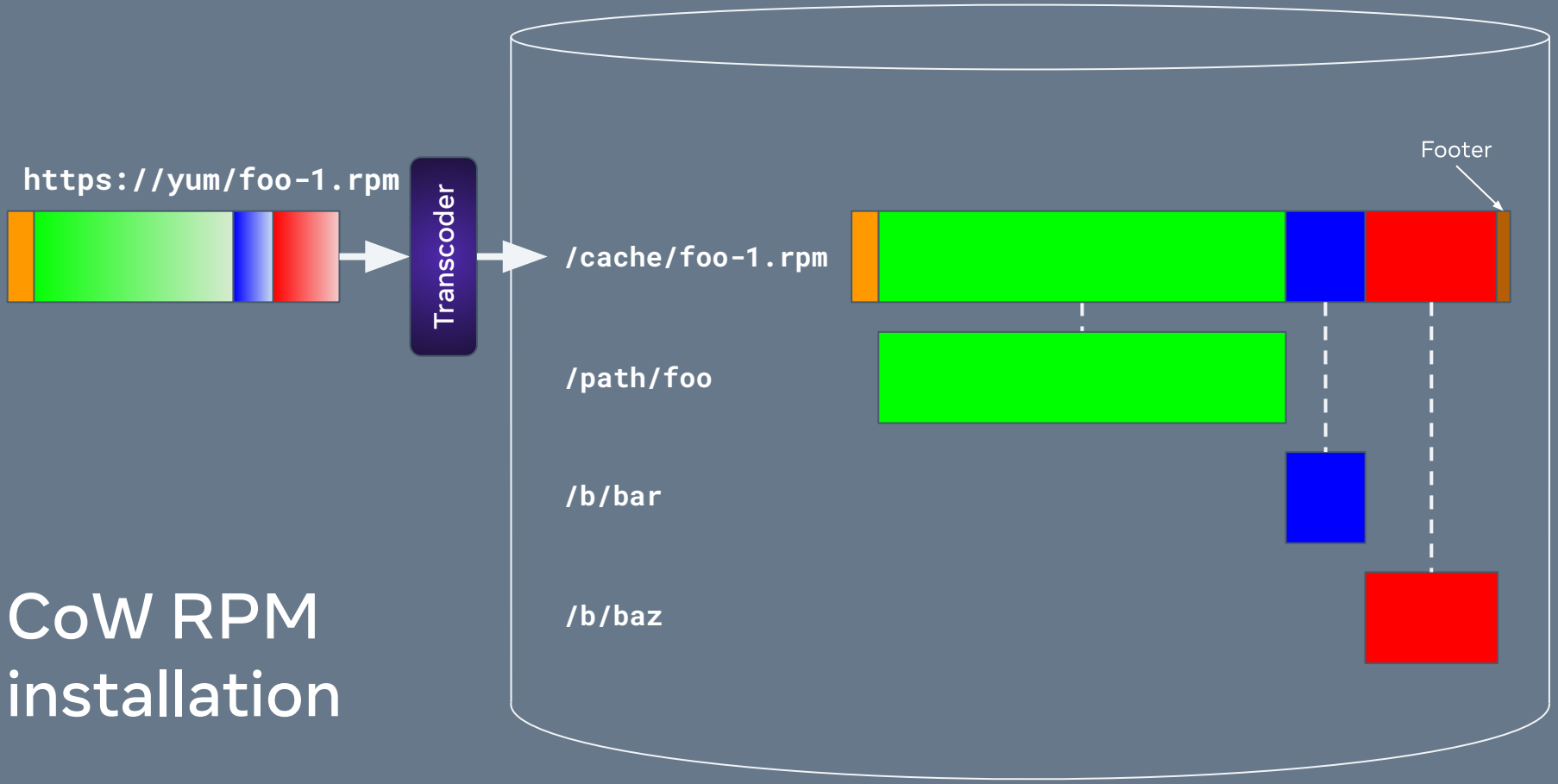


Traditional RPM installation

A note on duplication

	Cost	Granularity	So what?
Copy Octets	O(n)	Any	Expensive
Symlink	O(1)	File	Weak Reference Changes propagate Shares mode/owner
Hardlink	O(1)	File	Same filesystem Changes propagate Shares mode/owner
Reflink	O(log(n))	FS Block size	Same filesystem Filesystem support Page alignment

CoW RPM installation



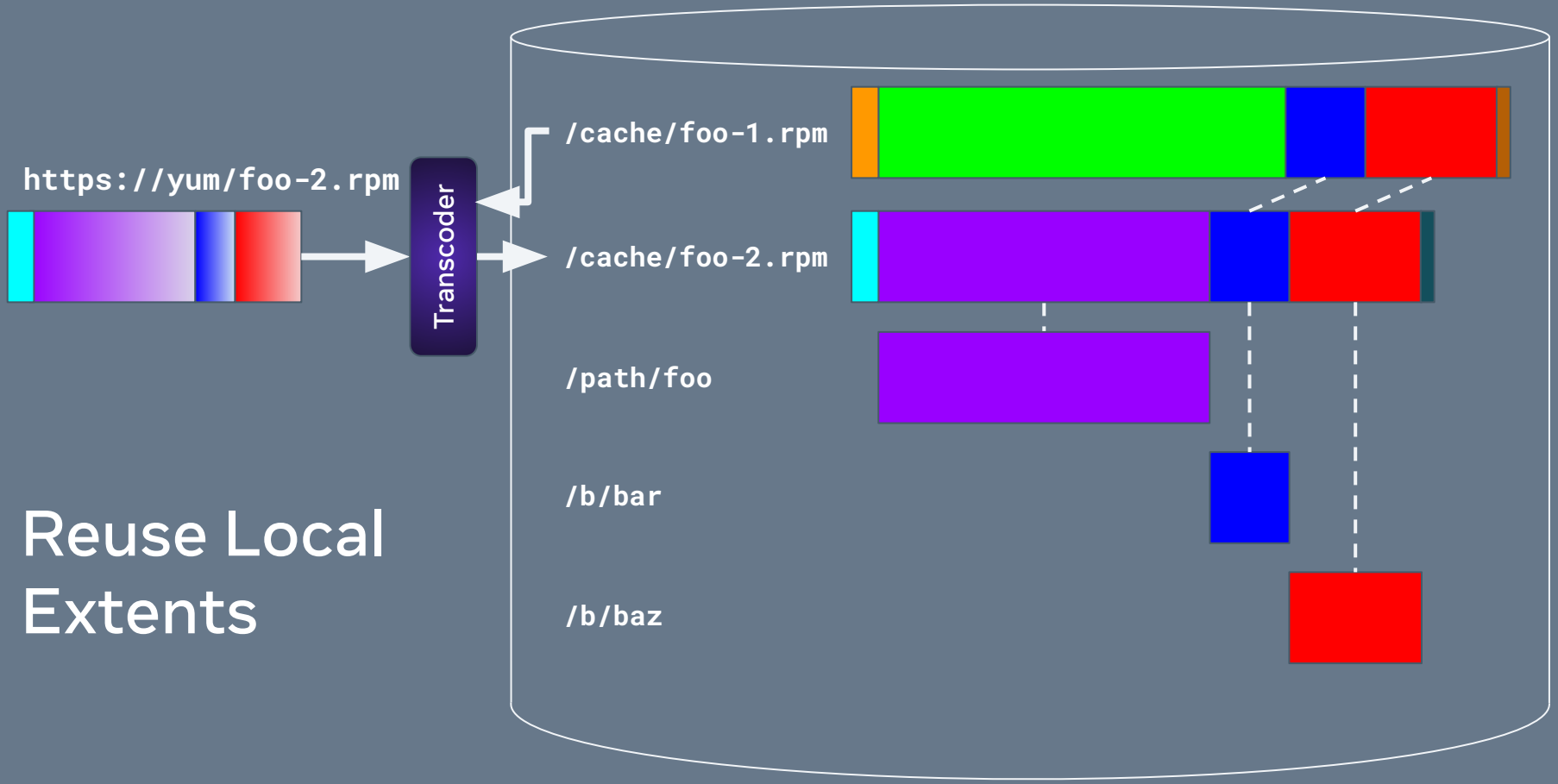
DNF with Copy on Write

- Packages are decompressed during download
  - Can be parallelized
- Footer contains
  - MAGIC value to identify transcoded data
  - Calculation of original file digest(s) to verify downloads
  - Sorted table of content digest→offset
- Contents are reflinked: existing data is referenced
  - Content is aligned to page boundaries/padded
  - `ioctl(dst, FICLONERANGE, &fcr)`
  - Fall back to regular file copy, e.g. `/boot`

All of this exists today

Looking forwards 

3. Reuse Local Extents



Reuse Local Extents

Reuse Local Extents

- `keepcache=True`
- Package Cache contains transcoded packages
- “digest addressable filesystem”
- Files from existing package can reused (reflinked) into new package, and into final destination
- Deduplication
- Similar intent to delta rpms, less expensive
- Saves writes
- Still costs network bandwidth and CPU for decompression

4. Packed Object Repositories

What's in a repo?

https://yum/foo-1.rpm

https://yum/foo-2.rpm

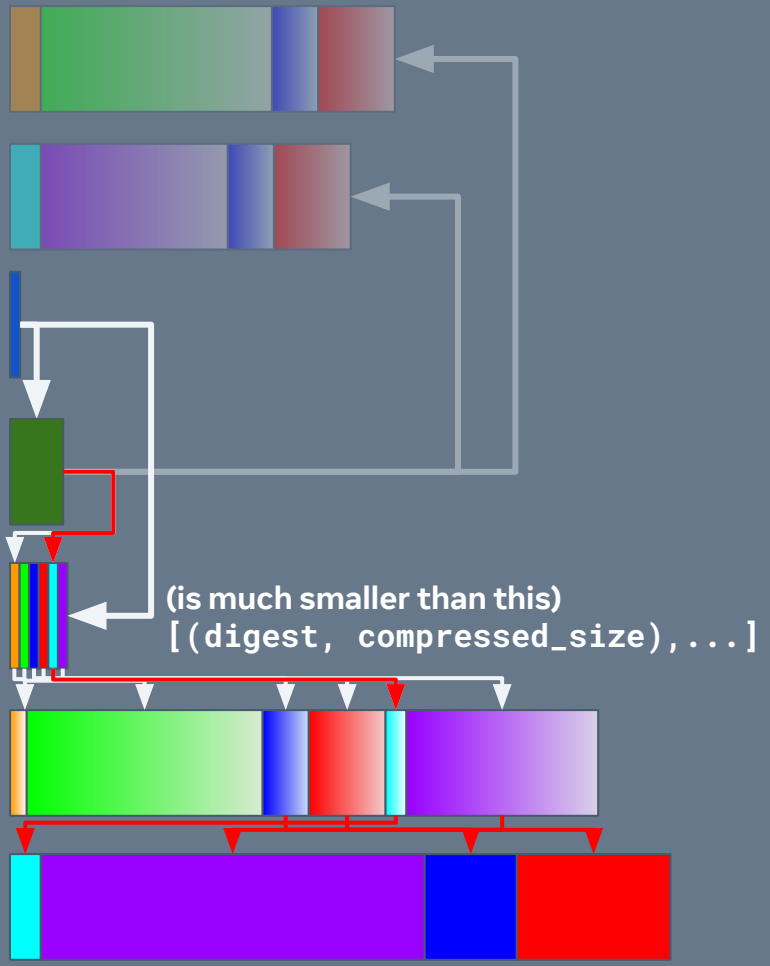
https://yum/repodata/repomd.xml

https://yum/repodata/15c..1a2-primary.xml.gz

https://yum/repodata/1-sha256.index

https://yum/repodata/1-sha256.data

/cache/foo-2.rpm



Packed Objects Repositories

- Changes Organization from Package to File oriented
- Pack index + data per digest type
- Clients maintain copy of indexes which grow over time
- Two step download
  1. Get headers for packages
  2. Follow digests in headers to reuse local extents, get data via http(s)
- You only download and decompress what you’re missing
- Parallelize even on single packages

5. Use case: Better Images

RPM CoW + Local Extent Reuse + Packed Repositories

Produce Image

1. Build image in `/var/images/${name}`
2. Package as single, (large?) rpm
3. Upload to packed repository

Consume for Container images

1. Install RPM

Consume for Operating System Images

1. Install RPM in existing OS or ramdisk, then on first boot:
2. `mv /* /old`

   `mv /old/var/images/${name}/* /`

   `mv /old/var/cache/dnf \`
   `/var/cache/dnf`
3. Continue booting
4. rm /old -r

Summary

- Today:
  - Time is proportional to sum(file sizes) + number of files
  - Churn on storage: reprovisioning ends up re-writing some GB each time
- Future
  - Time is proportional to delta of sum(file sizes) + number of files
  - Storage / distribution / download is delta based
  - Order of package operations / updates is not fixed. Contrast to sendstreams:
    - Sendstreams deltas only go from point A to B, exactly
    - Sendstreams are subvolume / filesystem not “package” level
  - Benefits to “image” and normal package installation/update flows

Status

1. DNF and RPM with Copy on Write
    - In production at Facebook
    - CentOS 8 version in Hyperscale SIG soon
    - Proposed for Fedora 34
      https://fedoraproject.org/wiki/Changes/RPMCoW
    - Refactoring code
    - Address package verification concerns
2. Reuse Local Extents + Packed Object Repositories
    - Next!

Let’s talk!

- **Now: Q&A**
- Later
  - malmond@fb.com
  - Freenode: malmond
  - https://fedoraproject.org/wiki/Changes/RPMCoW
  - Top level project: https://github.com/facebookincubator/dnf-plugin-cow/