Backing Chain Management in libvirt and qemu

Eric Blake <eblake@redhat.com>
KVM Forum, August 2015

redhat.

In this presentation

- How does the qcow2 format track point-in-time snapshots
- What are the qemu building blocks for managing backing chains
- How are these building blocks used together in libvirt

Part I

Understanding qcow2

qcow2 history

- qcow format (QEMU Copy On Write) documented in 2006
- qcow2 created in 2008, adding things like:
  - Internal snapshots with reference counting
- Hacky addition in 2009 to add header extensions
  - Backing file format, to avoid format probing CVEs
- qcow2v3 created in April 2012, adding things like:
  - Feature bits (extension is easier!)
  - Efficient zero cluster management

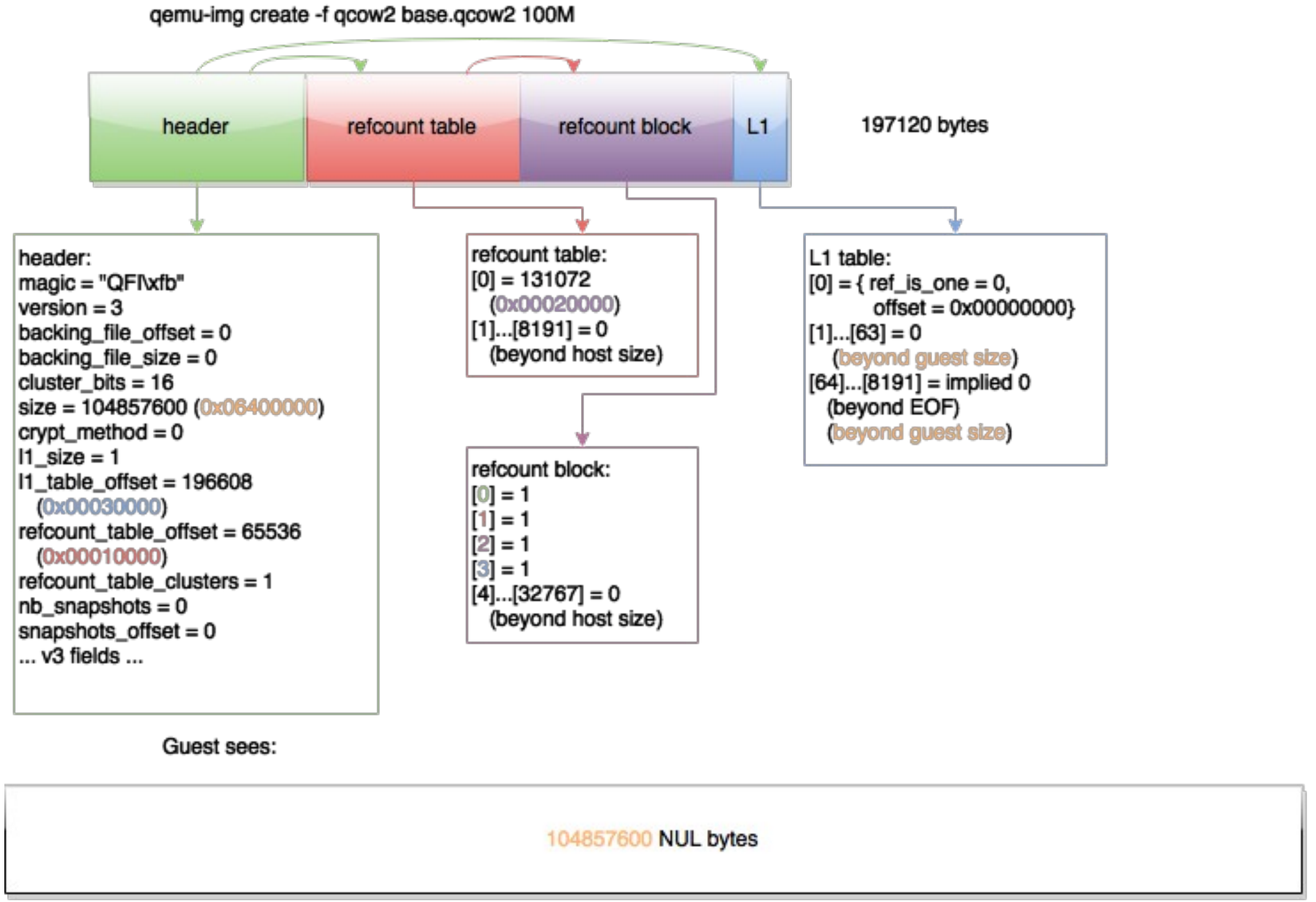
Let's look under the hood

- Create a new file
- Write some guest data
- Create an internal snapshot
- Write more guest data
- Create an external snapshot
- Write even more guest data

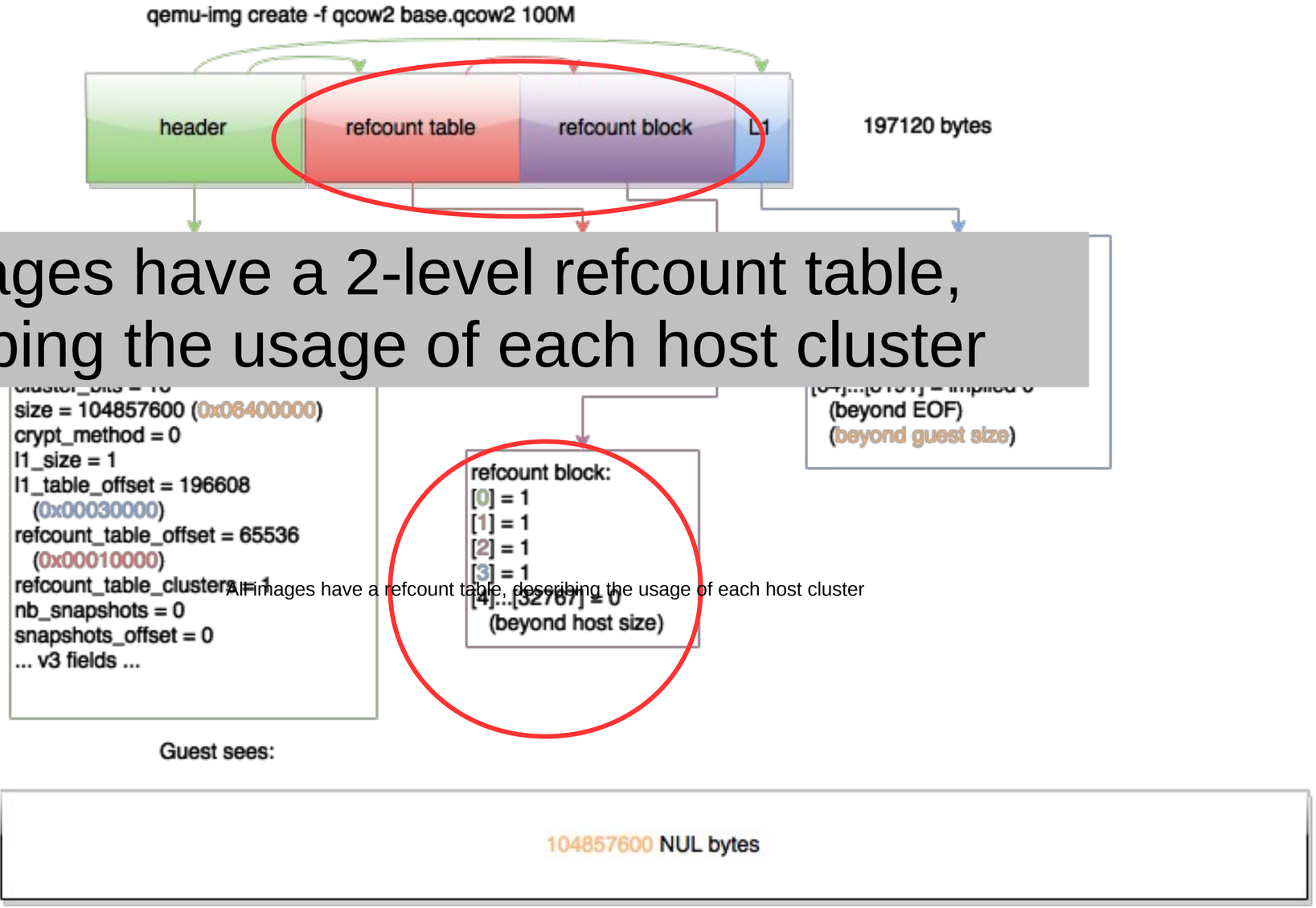
Create a new file

qemu-img create -f qcow2 base.qcow2 100M

Create a new file

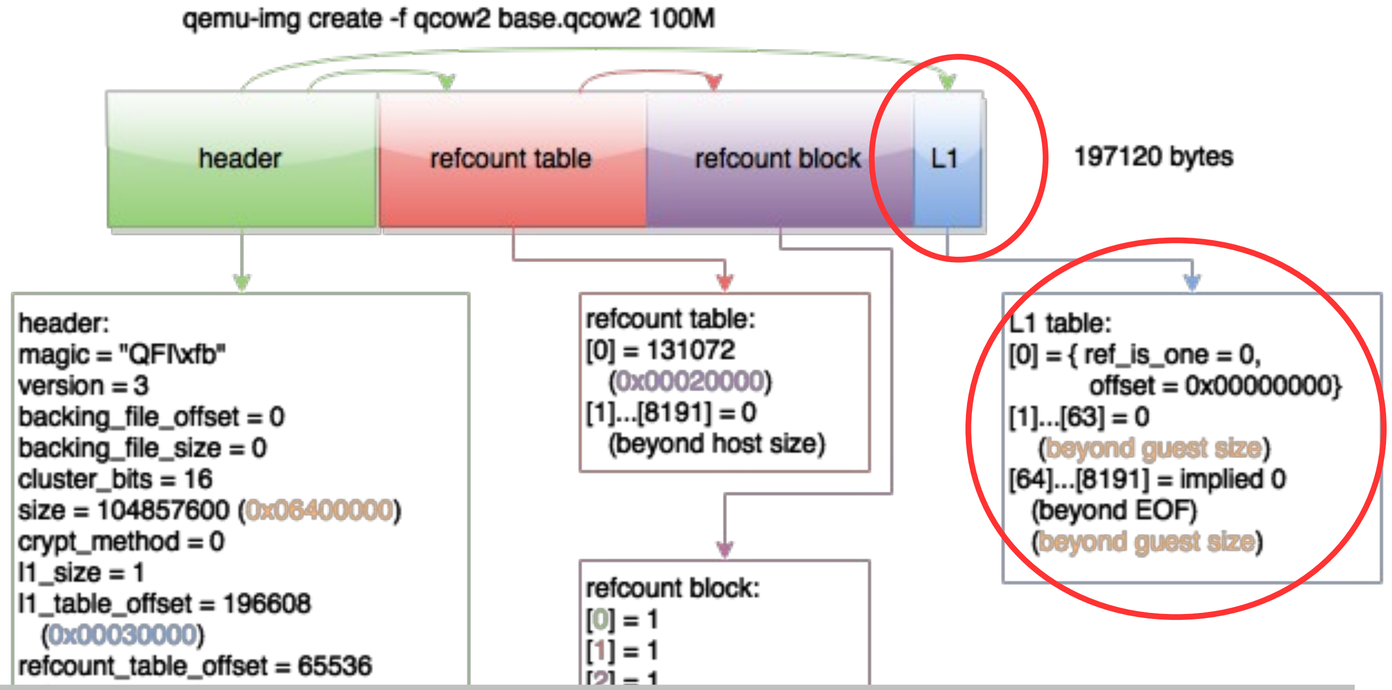


Create a new file



All images have a 2-level refcount table, describing the usage of each host cluster

Create a new file

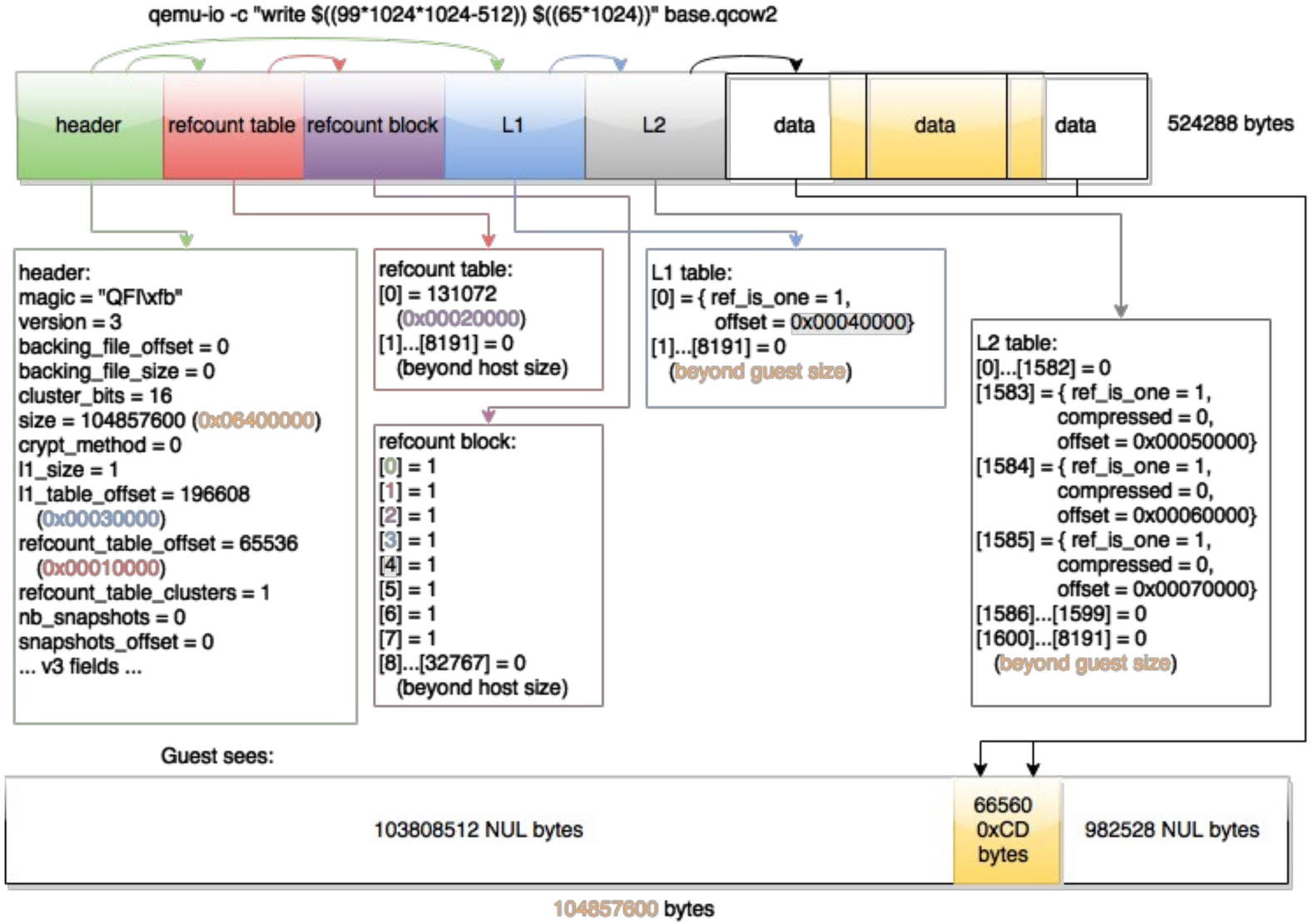


All images have an L1/L2 table,
describing the mapping of each guest cluster
(but with no data mapped, L2 is omitted)

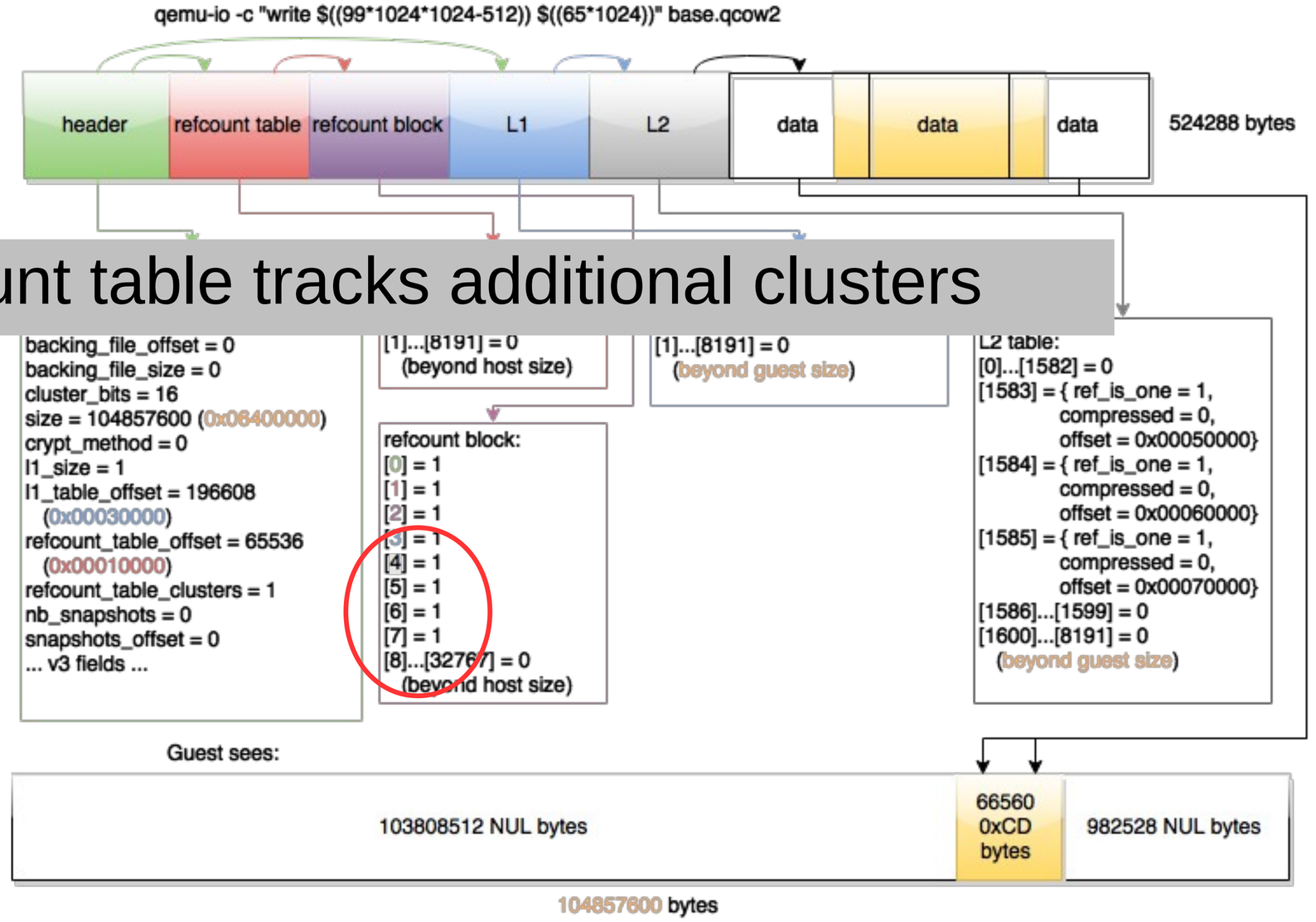
Write some guest data

qemu-io -c “write $((99*1024*1024-512)) $((65*1024))” base.qcow2

Write some guest data

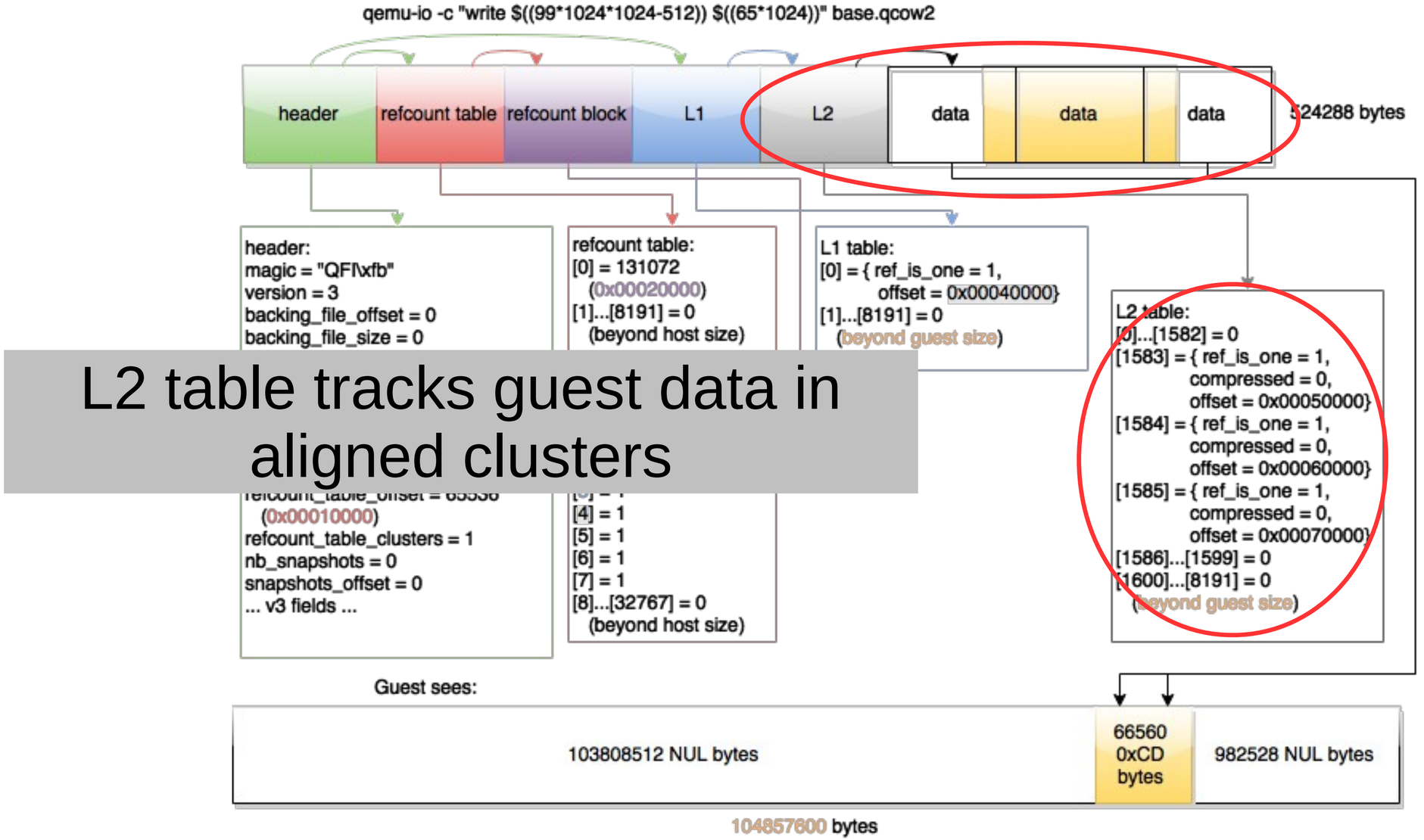


Write some guest data



Refcount table tracks additional clusters

Write some guest data



L2 table tracks guest data in aligned clusters