

Backing Chain Management in libvirt and qemu

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

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

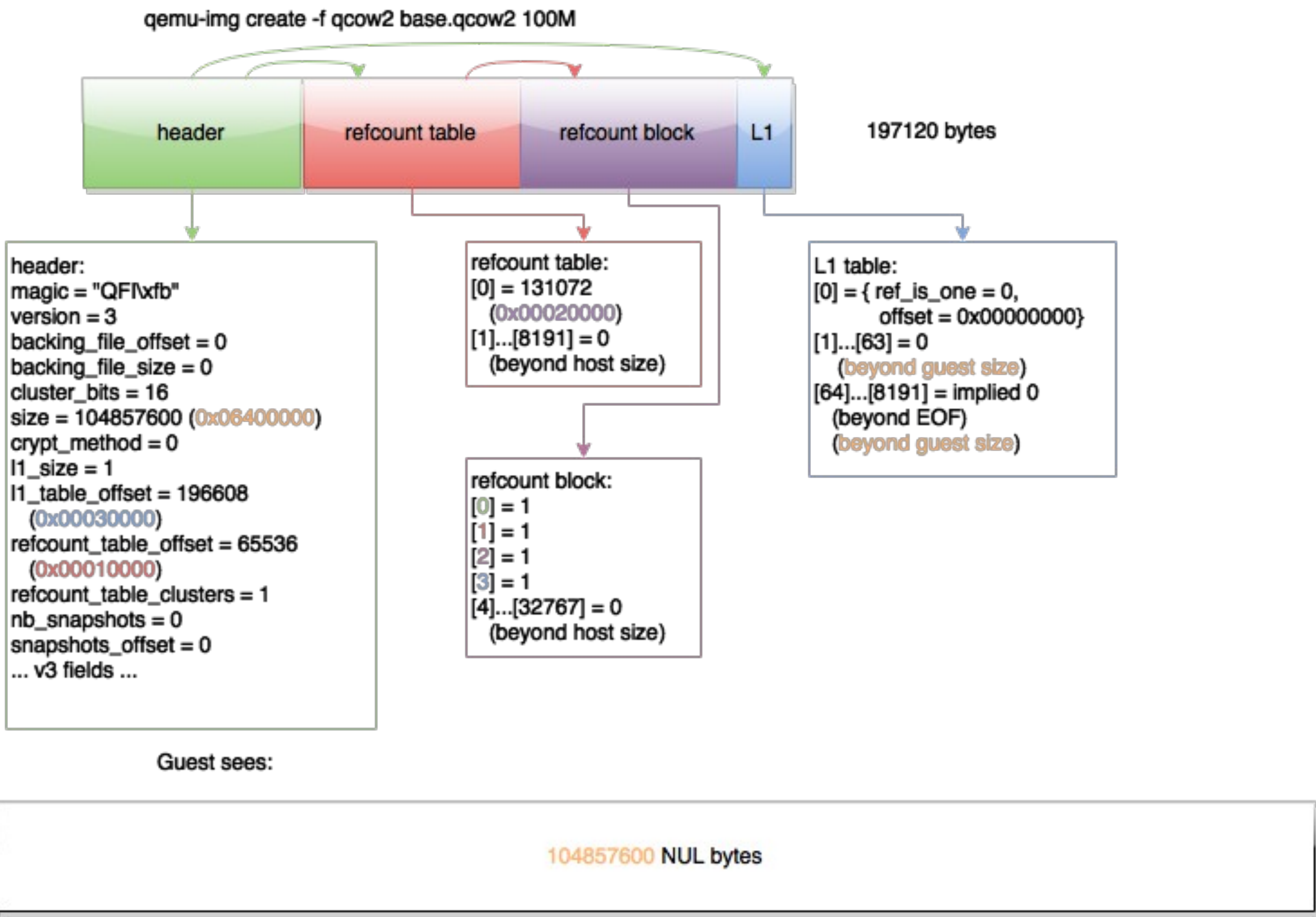
Let's look under the hood

- 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

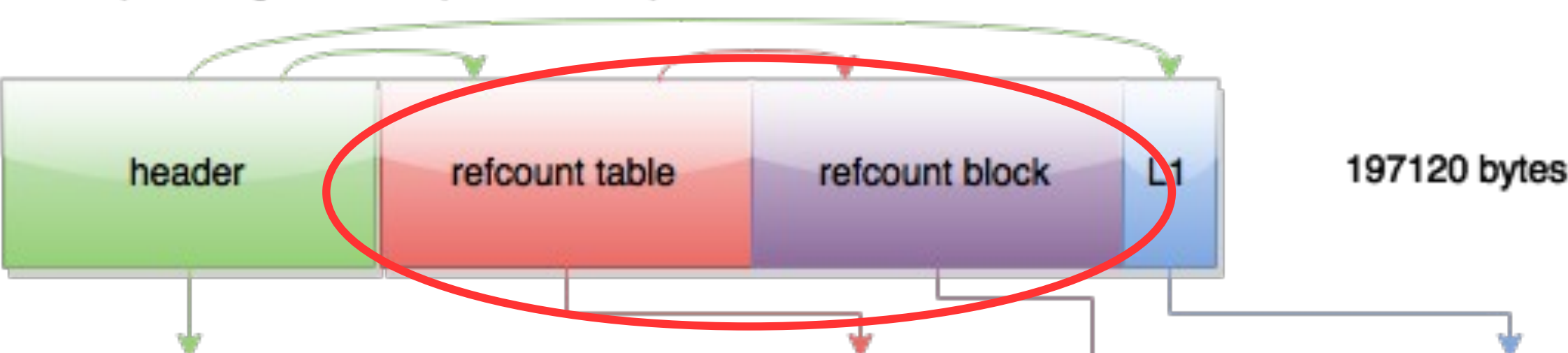
```
qemu-img create -f qcow2 base.qcow2 100M
```


Create a new file



Create a new file

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



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

```
cluster_size = 1048576  
size = 104857600 (0x06400000)  
crypt_method = 0  
l1_size = 1  
l1_table_offset = 196608  
    (0x00030000)  
refcount_table_offset = 65536  
    (0x00010000)  
refcount_table_clusters = 1  
nb_snapshots = 0  
snapshots_offset = 0  
... v3 fields ...
```

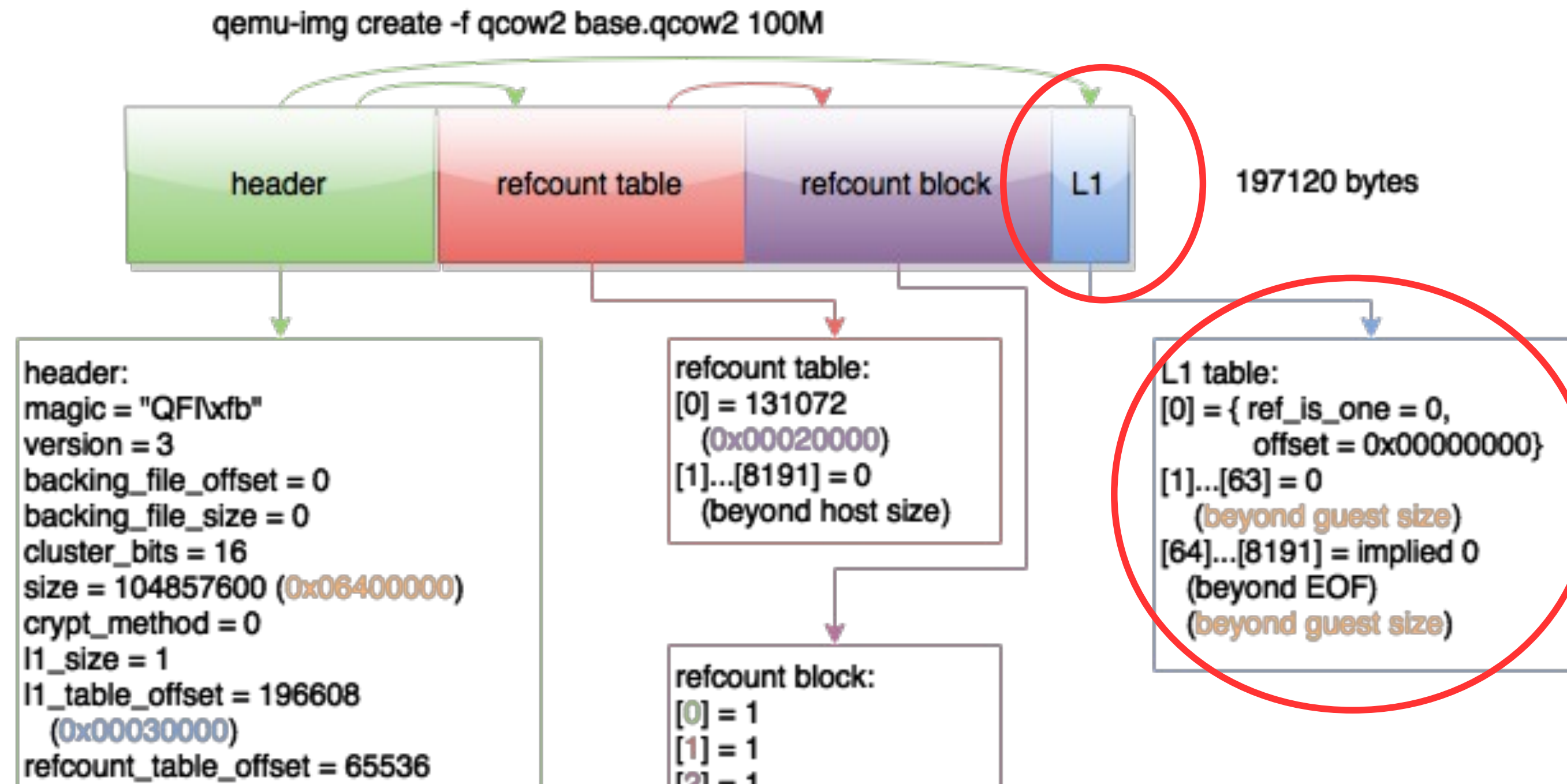
Guest sees:

```
refcount block:  
[0] = 1  
[1] = 1  
[2] = 1  
[3] = 1  
[4]...[32767] = 0  
    (beyond host size)
```

[0]...[32767] = implied 0
 (beyond EOF)
 (beyond guest size)

104857600 NUL bytes

Create a new file



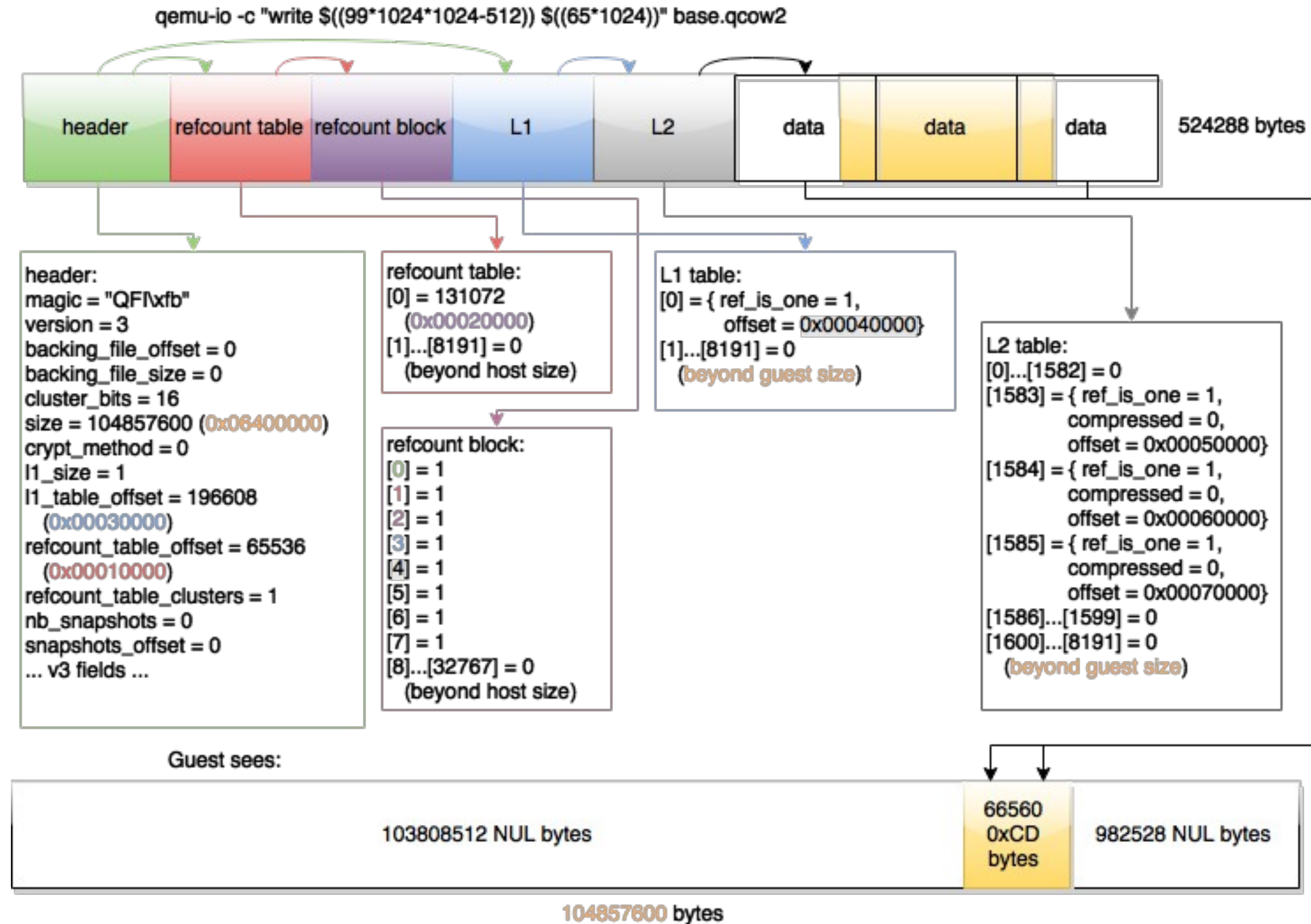
All images have an L1/L2 table, describing the mapping of each guest cluster (but with no data mapped, L2 is omitted)

104857600 NUL bytes

Write some guest data

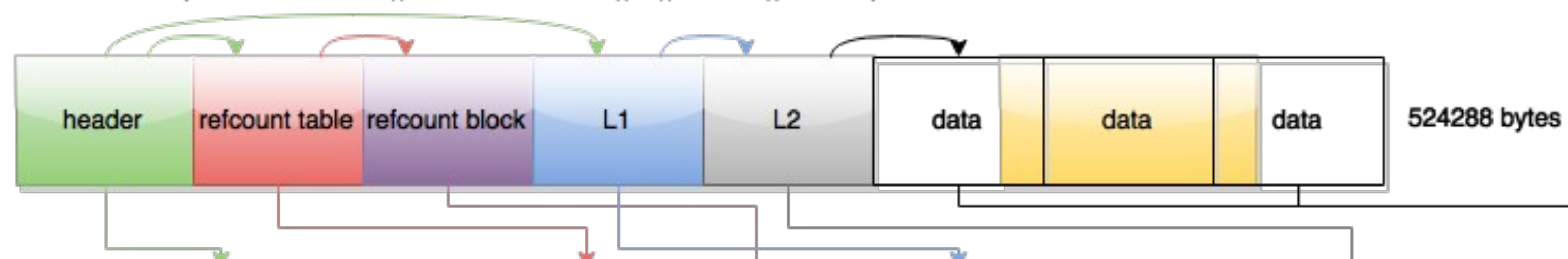
```
qemu-io -c "write $((99*1024*1024-512)) $((65*1024))" base.qcow2
```

Write some guest data



Write some guest data

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



Refcount table tracks additional clusters

backing_file_offset = 0
backing_file_size = 0
cluster_bits = 16
size = 104857600 (0x06400000)
crypt_method = 0
l1_size = 1
l1_table_offset = 196608
(0x00030000)
refcount_table_offset = 65536
(0x00010000)
refcount_table_clusters = 1
nb_snapshots = 0
snapshots_offset = 0
... v3 fields ...

[1]...[8191] = 0
(beyond host size)

refcount block:
[0] = 1
[1] = 1
[2] = 1
[3] = 1
[4] = 1
[5] = 1
[6] = 1
[7] = 1
[8]...[32767] = 0
(beyond host size)

[1]...[8191] = 0
(beyond guest size)

L2 table:
[0]...[1582] = 0
[1583] = { ref_is_one = 1,
compressed = 0,
offset = 0x00050000}
[1584] = { ref_is_one = 1,
compressed = 0,
offset = 0x00060000}
[1585] = { ref_is_one = 1,
compressed = 0,
offset = 0x00070000}
[1586]...[1599] = 0
[1600]...[8191] = 0
(beyond guest size)

Guest sees:



104857600 bytes

Write some guest data

