Linux
Plumbers
Conference 2022

>> Dublin, Ireland / September 12-14, 2022

io_uring in Android OTA

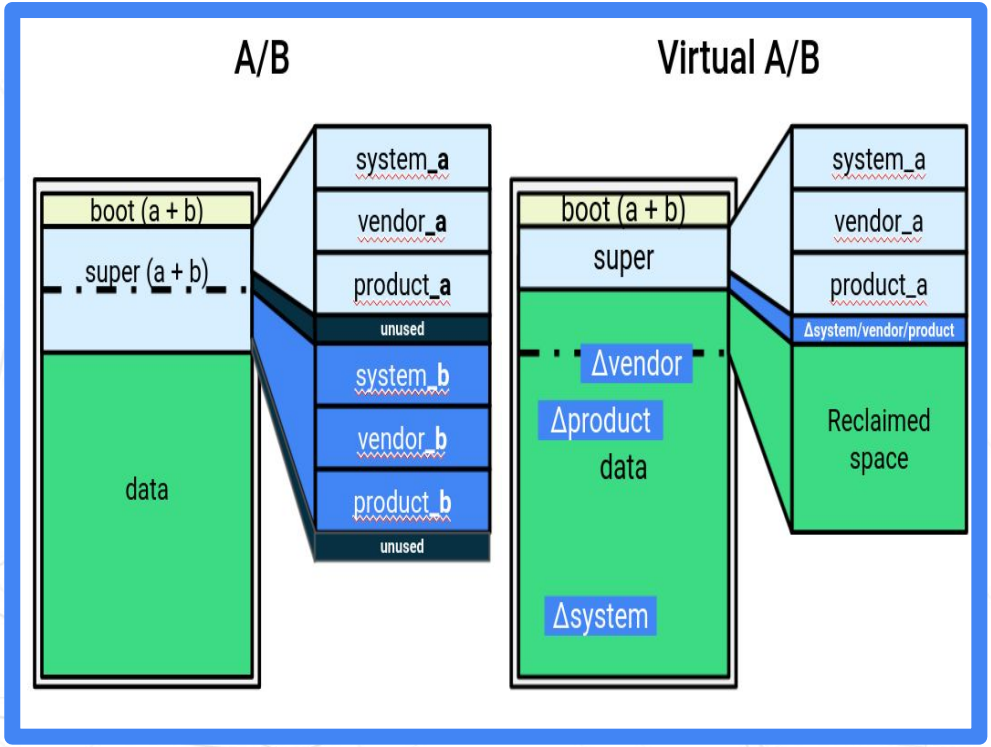
Akilesh Kailash (akailash@google.com)

Agenda

- Overview of Android OTA and Virtual A/B
- Snapshot-merge performance with io_uring
- Overview of dm-user in Virtual A/B
- ublk - userspace block driver and integration with Android OTA

Virtual A/B

- **Delta size** depends on the **update size** (copy-on-write) and can be computed in advance
- Space for **deltas** is **dynamically allocated** during an update (use free space in **super** and files in **/data**)



Android Copy on Write (COW) format

- Encodes four block-level operations:
  - ZERO: The destination block is zeroed.
  - COPY: The destination block is copied from a pre-existing block.
  - REPLACE: The destination block is replaced with new data, gz-compressed into the COW.
  - XOR: The destination block is an XOR from a pre-existing block with the changed content stored in COW.

Snapshot Merge during OTA

OTA operation:

- Operation: COPY
  - BLOCK X -> BLOCK Y
    - Read BLOCK X to buffer (syscall)
    - Write BLOCK Y from buffer (syscall)
    - Fsync (syscall)

- An Incremental OTA of ~200MB has ~150k COPY operations on SYSTEM partition and ~350k COPY operations on PRODUCT partition
- Total syscall = 150k + 350k = 500k COPY ops * 3 = 1.5M syscalls
- Potentially another 500k+ when SYSTEM_EXT, VENDOR partitions are considered

Snapshot-merge COPY operation

TOTAL	%	SELF	%	LOCATION
14,214,485,212 cpu-cycles	99.97%	0 cpu-cycles	0%	▾ __start_thread
14,214,485,212 cpu-cycles	99.97%	0 cpu-cycles	0%	▾ __pthread_start(void*)
7,326,820,161 cpu-cycles	51.53%	0 cpu-cycles	0%	▾ void* std::__1::__thread_proxy<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__
7,326,820,161 cpu-cycles	51.53%	0 cpu-cycles	0%	▾ std::__1::__async_assoc_state<bool, std::__1::__async_func<bool (android::snapshot::ReadAhead:
7,326,820,161 cpu-cycles	51.53%	0 cpu-cycles	0%	▾ android::snapshot::ReadAhead::RunThread()
7,326,392,059 cpu-cycles	51.52%	10,166,584 cpu-cycles	0.071%	▾ android::snapshot::ReadAhead::ReadAheadIOStart()
6,624,203,352 cpu-cycles	46.59%	45,460,799 cpu-cycles	0.320%	▾ android::snapshot::ReadAhead::ReadAheadSyncIO()
5,862,696,041 cpu-cycles	41.23%	21,418,292 cpu-cycles	0.151%	▾ android::base::ReadFullyAtOffset(android::base::borrowed_fd, void*, unsigned long, long)
5,835,200,469 cpu-cycles	41.04%	40,987,855 cpu-cycles	0.288%	▾ pread64
5,714,641,949 cpu-cycles	40.19%	0 cpu-cycles	0%	▾ el0_sync
5,714,641,949 cpu-cycles	40.19%	0 cpu-cycles	0%	▾ el0_sync_handler
5,714,641,949 cpu-cycles	40.19%	0 cpu-cycles	0%	▾ el0_svc
5,710,857,963 cpu-cycles	40.16%	76,245,511 cpu-cycles	0.536%	▾ el0_svc_common
5,613,168,931 cpu-cycles	39.48%	12,419,850 cpu-cycles	0.087%	▾ __arm64_sys_pread64
5,566,005,030 cpu-cycles	39.14%	57,659,233 cpu-cycles	0.406%	▸ vfs_read

TOTAL	%	SELF	%	LOCATION
14,214,485,212 cpu-cycles	99.97%	0 cpu-cycles	0%	▾ __start_thread
14,214,485,212 cpu-cycles	99.97%	0 cpu-cycles	0%	▾ __pthread_start(void*)
7,326,820,161 cpu-cycles	51.53%	0 cpu-cycles	0%	▸ void* std::__1::__thread_proxy<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete<std::__1::__thread_struct> >, void (std::__1::__
6,803,569,548 cpu-cycles	47.85%	0 cpu-cycles	0%	▾ void* std::__1::__thread_proxy<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete<std::__1::__thread_struct> >, void (std::__1::__
6,803,569,548 cpu-cycles	47.85%	0 cpu-cycles	0%	▾ std::__1::__async_assoc_state<bool, std::__1::__async_func<bool (android::snapshot::ReadAhead::*)(), android::snapshot::ReadAhead*> >::__execute()
6,538,236,651 cpu-cycles	45.98%	0 cpu-cycles	0%	▾ android::snapshot::Worker::RunMergeThread()
6,538,236,651 cpu-cycles	45.98%	0 cpu-cycles	0%	▾ android::snapshot::Worker::Merge()
4,185,569,881 cpu-cycles	29.44%	2,476,117 cpu-cycles	0.017%	▸ android::snapshot::Worker::MergeReplaceZeroOps()
2,352,666,770 cpu-cycles	16.55%	0 cpu-cycles	0%	▾ android::snapshot::Worker::SyncMerge()
2,352,666,770 cpu-cycles	16.55%	2,422,373 cpu-cycles	0.017%	▾ android::snapshot::Worker::MergeOrderedOps()
1,178,822,451 cpu-cycles	8.290%	10,792,717 cpu-cycles	0.076%	▾ pwrite64
1,165,987,233 cpu-cycles	8.200%	0 cpu-cycles	0%	▸ el0_sync
2,042,501 cpu-cycles	0.014%	0 cpu-cycles	0%	▸ work_pending
987,117,026 cpu-cycles	6.942%	0 cpu-cycles	0%	▾ fsync
987,117,026 cpu-cycles	6.942%	0 cpu-cycles	0%	▾ el0_sync
987,117,026 cpu-cycles	6.942%	0 cpu-cycles	0%	▸ el0_sync_handler

- ~40% CPU cycles spent in READ syscall - Reading the block device
- 8% CPU cycles spent in WRITE syscall - Writing to block device
- 6% CPU cycles in FSYNC

Data from Pixel 6 running Android T

io_uring during snapshot-merge COPY operation

TOTAL	%	SELF	%	LOCATION
9,353,174,621 cpu-cycles	99.97%	0 cpu-cycles	0%	▾ __start_thread
9,353,174,621 cpu-cycles	99.97%	0 cpu-cycles	0%	▾ __pthread_start(void*)
6,725,046,995 cpu-cycles	71.88%	0 cpu-cycles	0%	▸ void* std::__1::__thread_proxy<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete<std::__1::__thread_struct> >, void
2,551,959,132 cpu-cycles	27.28%	0 cpu-cycles	0%	▾ void* std::__1::__thread_proxy<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete<std::__1::__thread_struct> >, void
2,551,959,132 cpu-cycles	27.28%	0 cpu-cycles	0%	▾ std::__1::__async_assoc_state<bool, std::__1::__async_func<bool (android::snapshot::ReadAhead::*)(), android::snapshot::ReadAhead*> >::__execute
2,551,959,132 cpu-cycles	27.28%	2,736,394 cpu-cycles	0.029%	▾ android::snapshot::ReadAhead::RunThread()
2,547,779,500 cpu-cycles	27.23%	16,923,590 cpu-cycles	0.181%	▾ android::snapshot::ReadAhead::ReadAheadIOStart()
1,456,737,914 cpu-cycles	15.57%	37,565,452 cpu-cycles	0.402%	▾ android::snapshot::ReadAhead::ReadAheadAsyncIO()
449,092,962 cpu-cycles	4.800%	23,426,968 cpu-cycles	0.250%	▸ android::snapshot::ReadAhead::PrepareNextReadAhead(unsigned long*, int*, std::__1::vector<unsigned long, std::__1::allocator<unsigned long>
442,973,369 cpu-cycles	4.735%	5,364,494 cpu-cycles	0.057%	▾ android::snapshot::ReadAhead::ReapIoCompletions(int)
436,827,701 cpu-cycles	4.669%	7,399,298 cpu-cycles	0.079%	▸ __io_uring_get_cqe
442,904 cpu-cycles	0.005%	0 cpu-cycles	0%	▸ el0_irq_naked
338,270 cpu-cycles	0.004%	338,270 cpu-cycles	0.004%	- _io_uring_get_cqe
391,857,795 cpu-cycles	4.188%	126,195 cpu-cycles	0.001%	▸ __io_uring_submit_and_wait
58,991,572 cpu-cycles	0.631%	19,509,233 cpu-cycles	0.209%	▸ scudo::Allocator<scudo::AndroidConfig, &(scudo_malloc_postinit)>::deallocate(void*, scudo::Chunk::Origin, unsigned long, unsigned long)

TOTAL	%	SELF	%	LOCATION
9,353,174,621 cpu-cycles	99.97%	0 cpu-cycles	0%	▾ __start_thread
9,353,174,621 cpu-cycles	99.97%	0 cpu-cycles	0%	▾ __pthread_start(void*)
6,725,046,995 cpu-cycles	71.88%	0 cpu-cycles	0%	▾ void* std::__1::__thread_proxy<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete<std::__1::__thread_struct> >, void (std::__1
6,725,046,995 cpu-cycles	71.88%	0 cpu-cycles	0%	▾ std::__1::__async_assoc_state<bool, std::__1::__async_func<bool (android::snapshot::ReadAhead::*)(), android::snapshot::ReadAhead*> >::__execute()
6,494,705,917 cpu-cycles	69.42%	0 cpu-cycles	0%	▾ android::snapshot::Worker::RunMergeThread()
6,493,793,533 cpu-cycles	69.41%	0 cpu-cycles	0%	▾ android::snapshot::Worker::Merge()
5,778,999,328 cpu-cycles	61.77%	2,470,899 cpu-cycles	0.026%	▸ android::snapshot::Worker::MergeReplaceZeroOps()
714,794,205 cpu-cycles	7.640%	0 cpu-cycles	0%	▾ android::snapshot::Worker::AsyncMerge()
714,794,205 cpu-cycles	7.640%	7,189,155 cpu-cycles	0.077%	▾ android::snapshot::Worker::MergeOrderedOpsAsync()
273,365,202 cpu-cycles	2.922%	1,365,770 cpu-cycles	0.015%	▸ __io_uring_submit_and_wait
247,425,610 cpu-cycles	2.645%	4,889,293 cpu-cycles	0.052%	▸ __io_uring_get_cqe

- ~4% CPU cycles spent in READ - Reading the block device
- ~7% CPU cycles spent in WRITE + FSYNC

Data from Pixel 6 running Android T

Snapshot Merge time

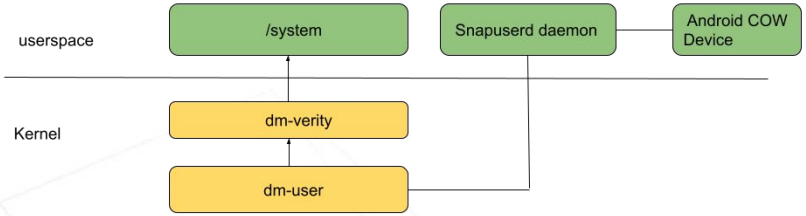
- **With io_uring, snapshot merge cuts down ~40%**.
  - On Pixel 6 running Android T, snapshot merge for an incremental OTA completes ~60-75 seconds with io_uring.
  - Merge time varies based on OTA configuration. Without io_uring, merge time varies between 120-180 seconds.

- Cut down on CPU cycles and number of threads to merge.

- Faster snapshot merge is important as partitions are mounted off dm-user
  - I/O from root filesystem will have to be served from user-space daemon until merge is completed.

dm-user kernel driver in Virtual A/B

- dm-user - Kernel module, like FUSE but userspace block device
- Out of tree patch maintained on 4.14, 4.19, 5.4, 5.10, 5.15+ android kernels.
- ~10% CPU cycles spent when dm-user is used as a loop-back device.



cpu-cycles | cpu-cycles_samples | Search for a function/file..

TOTAL	%	SELF	%	LOCATION
1,288,568,601 cpu-cycles	30.40%	0 cpu-cycles	0%	▾ __start_thread
1,288,568,601 cpu-cycles	30.40%	0 cpu-cycles	0%	▾ __pthread_start(void*)
1,288,568,601 cpu-cycles	30.40%	0 cpu-cycles	0%	▾ void* std::__1::__thread_proxy<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::defa
1,288,568,601 cpu-cycles	30.40%	0 cpu-cycles	0%	▾ std::__1::__async_assoc_state<bool, std::__1::__async_func<bool (android::snapshot::ReadAhead::*)(), ar
1,288,568,601 cpu-cycles	30.40%	0 cpu-cycles	0%	▾ android::snapshot::Worker::RunThread()
1,288,568,601 cpu-cycles	30.40%	0 cpu-cycles	0%	▾ android::snapshot::Worker::ProcessIORequest()
1,202,513,169 cpu-cycles	28.37%	25,681,279 cpu-cycles	0.606%	▾ android::snapshot::Worker::ReadAlignedSector(unsigned long long, unsigned long, bool)
1,002,938,741 cpu-cycles	23.66%	2,152,158 cpu-cycles	0.051%	▾ android::snapshot::Worker::ProcessCowOp(android::snapshot::CowOperation const*)
619,104,825 cpu-cycles	14.61%	979,853 cpu-cycles	0.023%	▸ android::snapshot::Worker::ProcessReplaceOp(android::snapshot::CowOperation const*)
357,997,252 cpu-cycles	8.447%	5,058,691 cpu-cycles	0.119%	▾ android::snapshot::Worker::ProcessOrderedOp(android::snapshot::CowOperation const*)
283,780,451 cpu-cycles	6.695%	612,354 cpu-cycles	0.014%	▾ android::snapshot::Worker::ReadFromSourceDevice(android::snapshot::CowOperation const*)
278,697,597 cpu-cycles	6.576%	3,874,314 cpu-cycles	0.091%	▾ android::base::ReadFullyAtOffset(android::base::borrowed_fd, void*, unsigned long, long)
274,823,283 cpu-cycles	6.484%	9,050,772 cpu-cycles	0.214%	▾ pread64
258,193,313 cpu-cycles	6.092%	0 cpu-cycles	0%	▸ el0_sync
4,904,731 cpu-cycles	0.116%	0 cpu-cycles	0%	▸ work_pending
2,674,467 cpu-cycles	0.063%	2,674,467 cpu-cycles	0.063%	- __blk_rq_map_sg
4,470,500 cpu-cycles	0.105%	0 cpu-cycles	0%	▸ android::base::ShouldLog(android::base::LogSeverity, char const*)
38,165,963 cpu-cycles	0.900%	15,357,490 cpu-cycles	0.362%	▸ android::snapshot::SnapshotHandler::ProcessMergingBlock(unsigned long, void*)
22,196,361 cpu-cycles	0.524%	0 cpu-cycles	0%	▸ android::snapshot::Worker::ReadDataFromBaseDevice(unsigned long long, unsigned long)
7,605,318 cpu-cycles	0.179%	107,891 cpu-cycles	0.003%	▸ android::snapshot::SnapshotHandler::NotifyIOCompletion(unsigned long)
752,441 cpu-cycles	0.018%	752,441 cpu-cycles	0.018%	- android::base::ReadFullyAtOffset(android::base::borrowed_fd, void*, unsigned long, long)
438,027 cpu-cycles	0.010%	0 cpu-cycles	0%	▸ el0_sync
23,661,165 cpu-cycles	0.558%	0 cpu-cycles	0%	▸ el0_sync
23,341 cpu-cycles	0.001%	0 cpu-cycles	0%	▸ work_pending
120,218,857 cpu-cycles	2.836%	0 cpu-cycles	0%	▾ android::snapshot::Worker::WriteDmUserPayload(unsigned long, bool)
120,218,857 cpu-cycles	2.836%	184,038 cpu-cycles	0.004%	▾ android::base::WriteFully(android::base::borrowed_fd, void const*, unsigned long)
120,034,819 cpu-cycles	2.832%	0 cpu-cycles	0%	▸ write

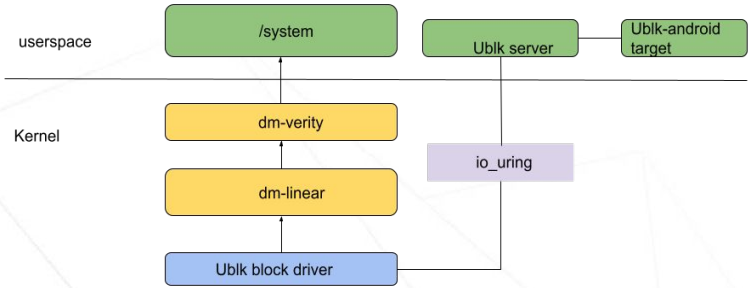
Data from Pixel 6 running Android T

ublk - userspace block driver

- ublk - Userspace block driver available upstream from 5.20
- It is io_uring based: i/o request is delivered to userspace via the newly added io_uring command (IORING_OP_URING_CMD).
- Supports multiple queues.
- Mmap ublk daemon VM space for re-mapping block I/O request pages.
- Libublksrv: userspace library available to integrate new ublk targets.

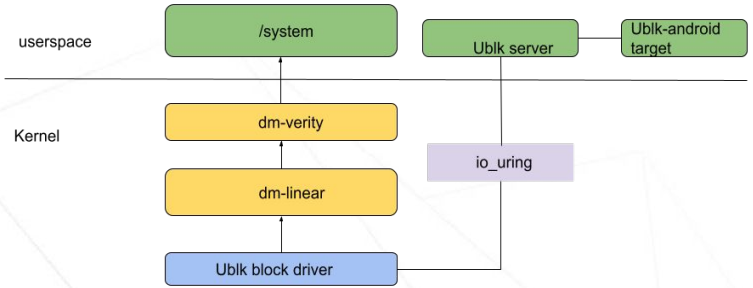
ublk - Integration with Android OTA

- Add new ublk-android target - A variation of loop target. Handle I/O request from ublk-server.
- ublk-loop target prototype completed on **Pixel 6 running android-mainline 6.0-rc1**.
- Some changes required in ublk server to support android specific target.
- No more out of tree kernel patch.
- Perf improvements - io_uring instance can be used for loop back during COPY ota operations.



ublk - Integration with Android OTA. Questions ?

- Additional dm-linear device-mapper target is required as I/O needs to be suspended during init first stage and selinux transition
- Post snapshot-merge, ublk driver has to be removed
- Device mapper has the suspend/resume support
  - Extend ublk driver to support it ?
- ublkserver needs trivial changes
  - No c++20 support in Android



Thank you