Historical overview of Android and its kernel

Park Ju Hyung (arter97)
qkrwngud825@gmail.com

About me

Park Ju Hyung
(arter97)

- Hobbyist Android / kernel developer since 2012
- SW Maestro 8th
- Community developer @ OnePlus
- Inha University undergraduate junior
- Scheduled for Ph.D. over at DGIST
- Interests: Anything that can improve Android / Linux!

About me

“Paranoid Android”
- Full-stack optimizations based on Android 7.1
- libc routines, jemalloc, sqlite, LTO, mm, f2fs, zlib, QPerformance, iop optimizations

“arter97 kernel”
- Custom kernel for Android on various devices
- Amalgamations of optimizations from various OEMs (with a few of my own)
- 500k downloads

“arkTube”
- YouTube Downloader for Android
- Powered by Python
- 8k downloads

- (Sent a patch upstream to fix reboot noise with 2017/2018 LG Gram laptops)

In this presentation...

- Light taste with each changes, no deep-dives

- With each major Android versions:
  - New notable system-level features / changes
  - My personal thoughts(discussions welcome)

- What other OEMs were doing

- What other communities were doing

- Feel free to interrupt and ask questions!

Index

2012	Android 4.3	Jellybean		fstrim zram f2fs - Flash Friendly File System
2013	Android 4.4	KitKat		dm-verity ksm SELinux
2014	Android 5.0	Lollipop		jemalloc memory allocator 64-bit support
2015	Android 6.0	Marshmallow	End of CPU cores hotplugging CONFIG_HZ=300 cpuset	
2016	Android 7.0	Nougat	Greg KH joins Google LLVM compiler	sdcardfs A/B partition layout EAS
2017	Android 8.0	Oreo	Regular LTS merges LLVM compiler for Android kernels	SECCOMP Project Treble
2018	Android 9.0	Pie	Android kernels + LTO & CFI Speculative Page Faults	HPB - Host Performance Booster erofs
2019	Android Q	?	APEX PSI & lmkd	Adiantum encryption Scoped Storage

Android 4.x (2012)

fstrim

Rotational storage (e.g. HDD)

READ, WRITE

Non-rotational storage (e.g. SSD, UFS, eMMC)

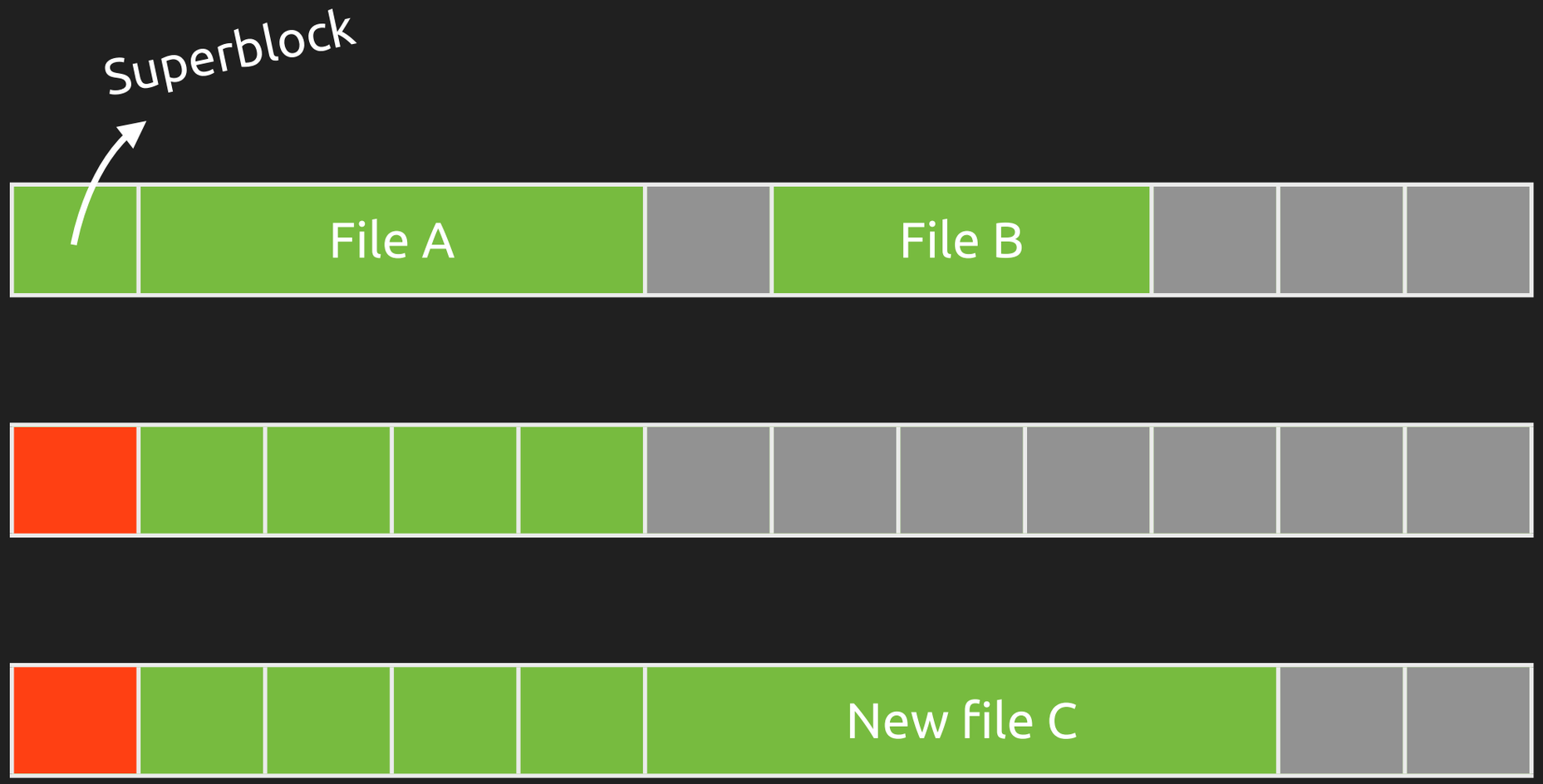
READ, PROGRAM, ERASE

ERASE must be done prior to PROGRAM

Android 4.x (2012)

fstrim

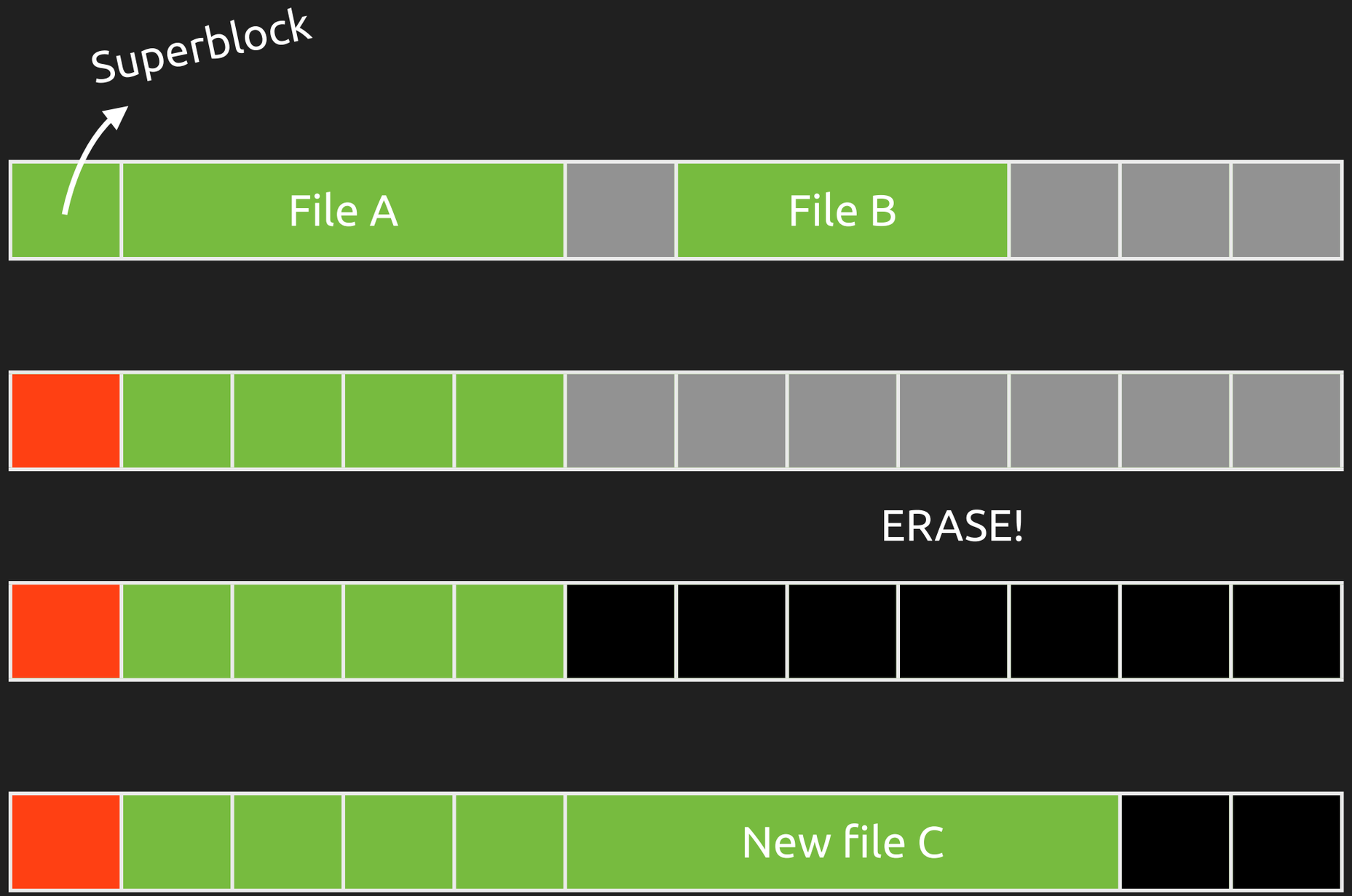
Deleting “File B” in HDD



Android 4.x (2012)

fstrim

Deleting “File B” in SSD

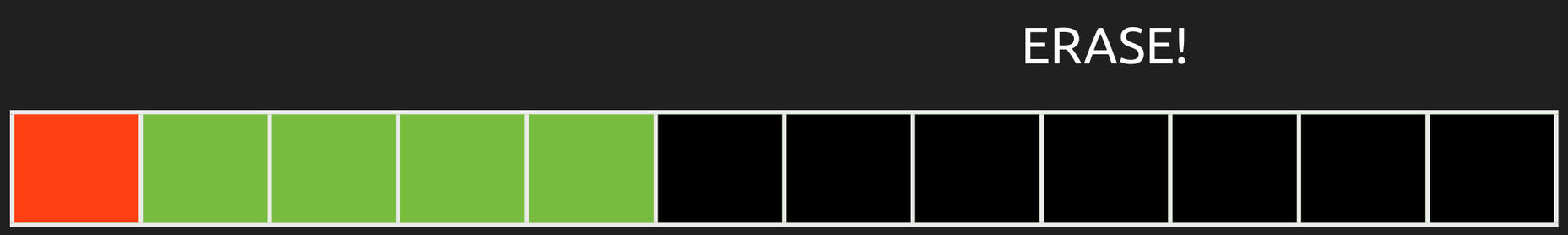


Android 4.x (2012)

fstrim

NAND flash chip doesn’t know which blocks are unused

It needs to be hinted from the kernel

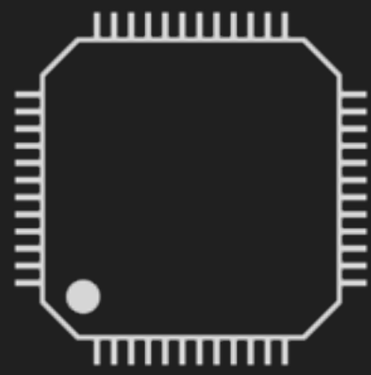


Android 4.x (2012)

fstrim

Enter TRIM command

Hey, block 4~10
are now unused



Ok, I'll ERASE those

ERASE!

Android 4.x (2012)

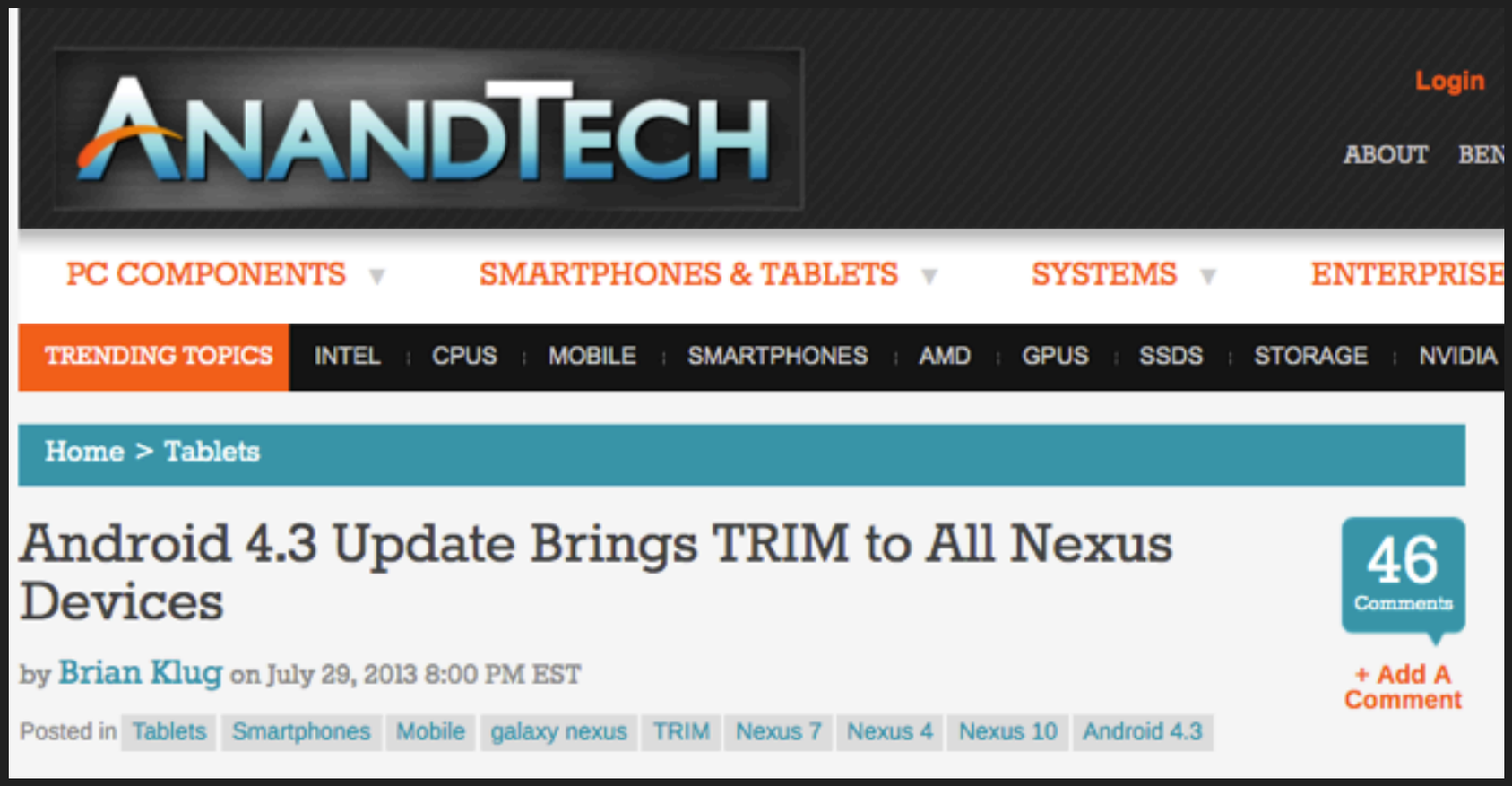
fstrim

	Time
READ	50 µs
PROGRAM	1000 µs
ERASE	3000 µs

* data not representative for modern NAND

Android 4.x (2012)

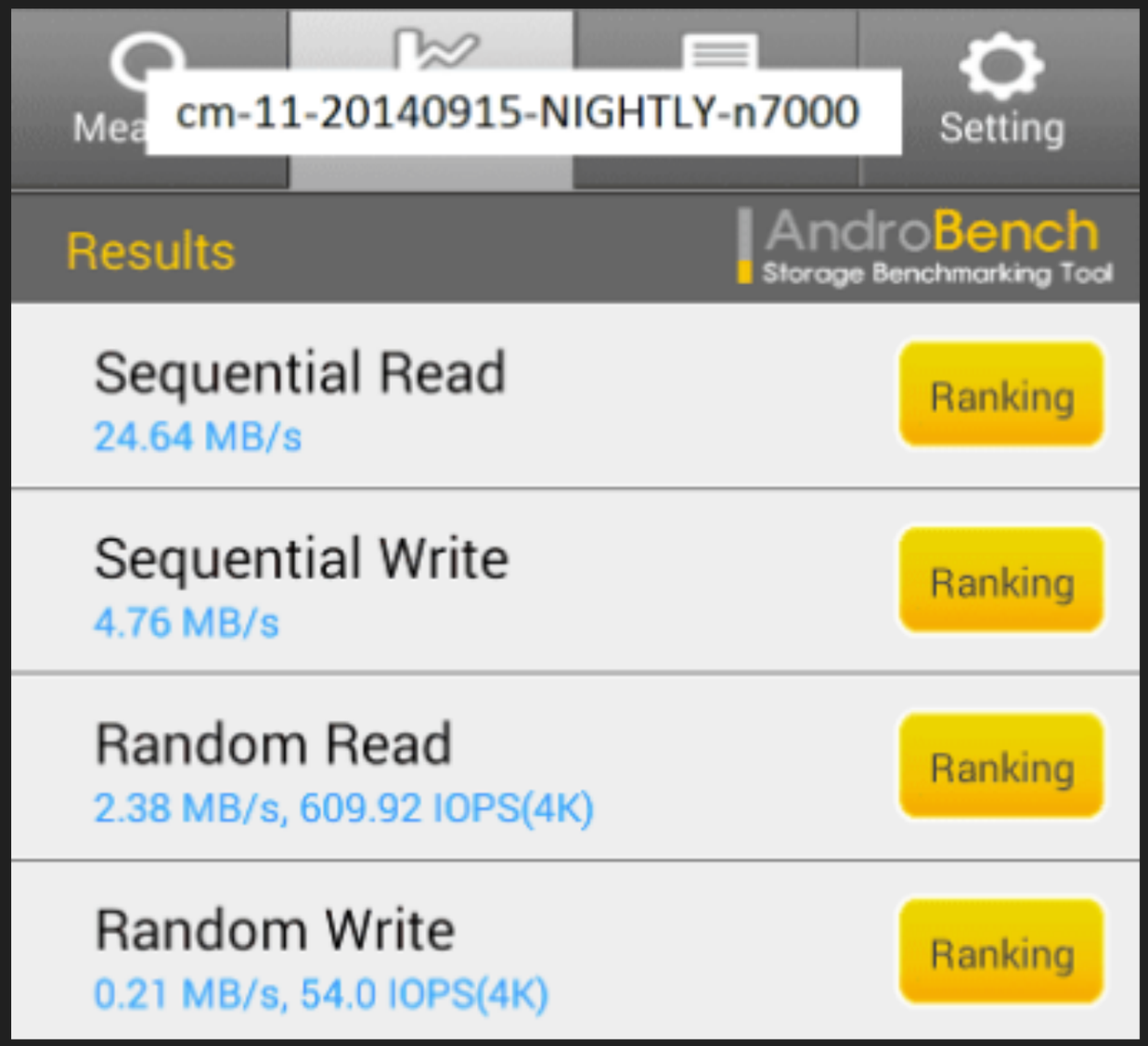
fstrim



ERASE!

Android 4.x (2012)

zram



Traditional swap doesn’t make sense on Android

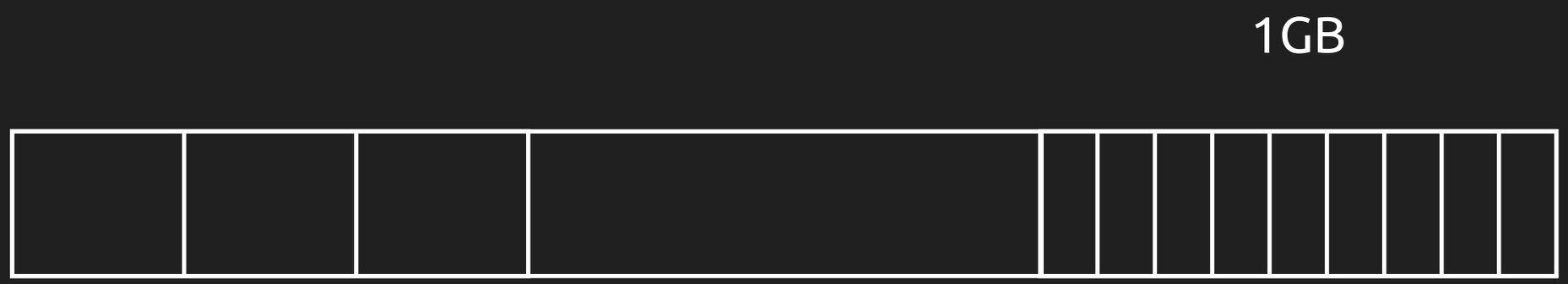
Android 4.x (2012)

zram

3GB

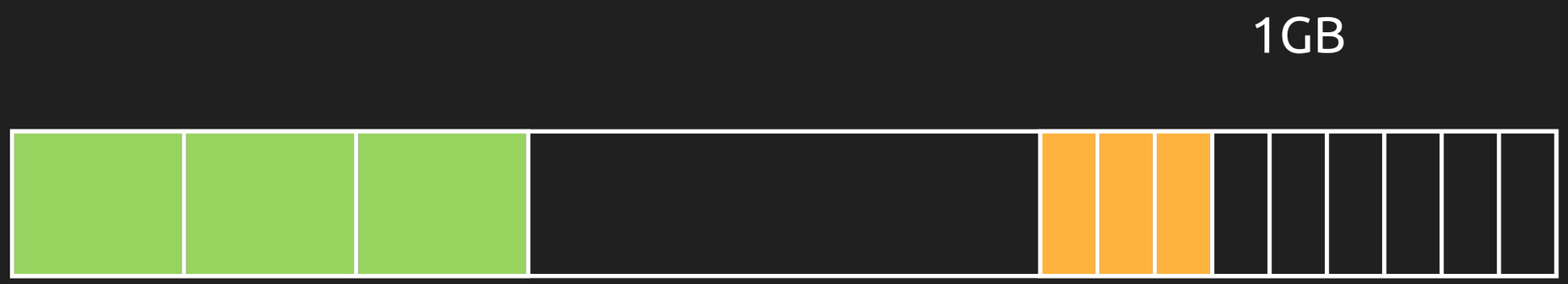
Android 4.x (2012)

zram



Android 4.x (2012)

zram



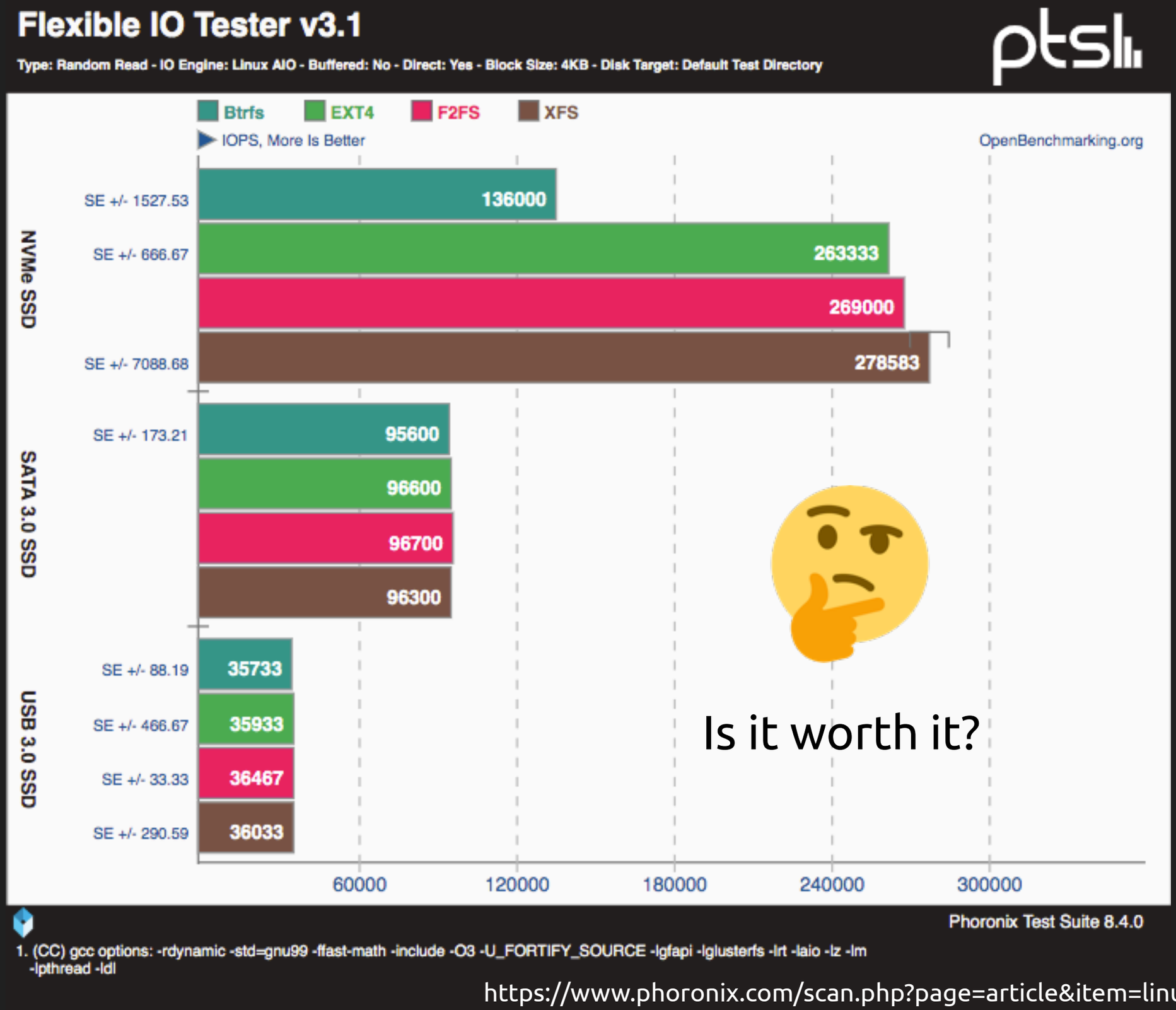
New Filesystem for Android

f2fs

- Optimized for devices with FTL(Flash Translation Layer)
- Log-structured filesystem

New Filesystem for Android

f2fs



https://www.phoronix.com/scan.php?page=article&item=linux-50-filesystems

New Filesystem for Android

f2fs

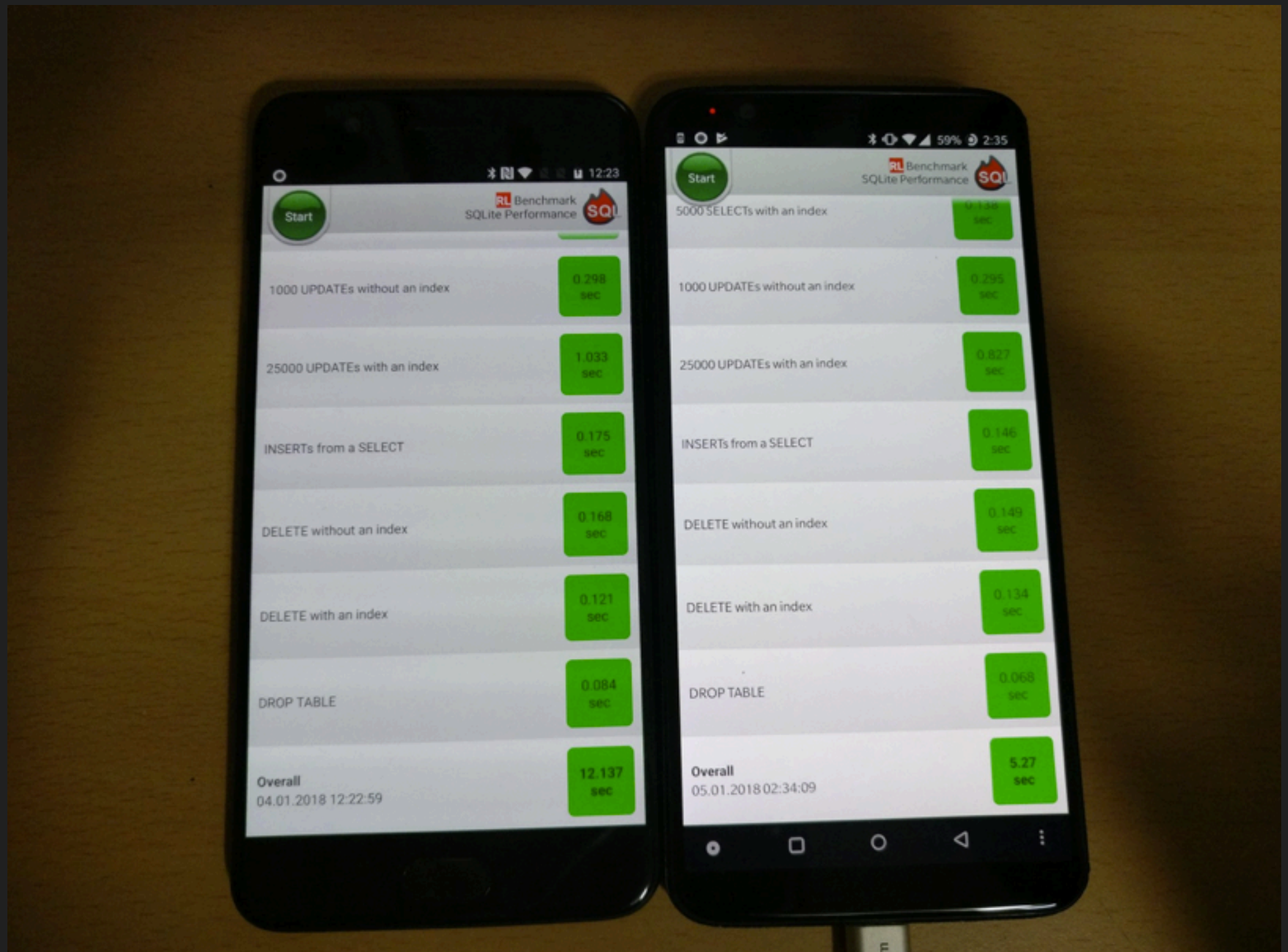
Yes !

- Atomic writing for SQLite
- Less wear & tear
- Less performance impact with high utilization
- nobarrier fsync() without hurting filesystem integrity
- Asynchronous discard(trim)s

- Included in the Pixel 3
- Mandatory for Android Go devices

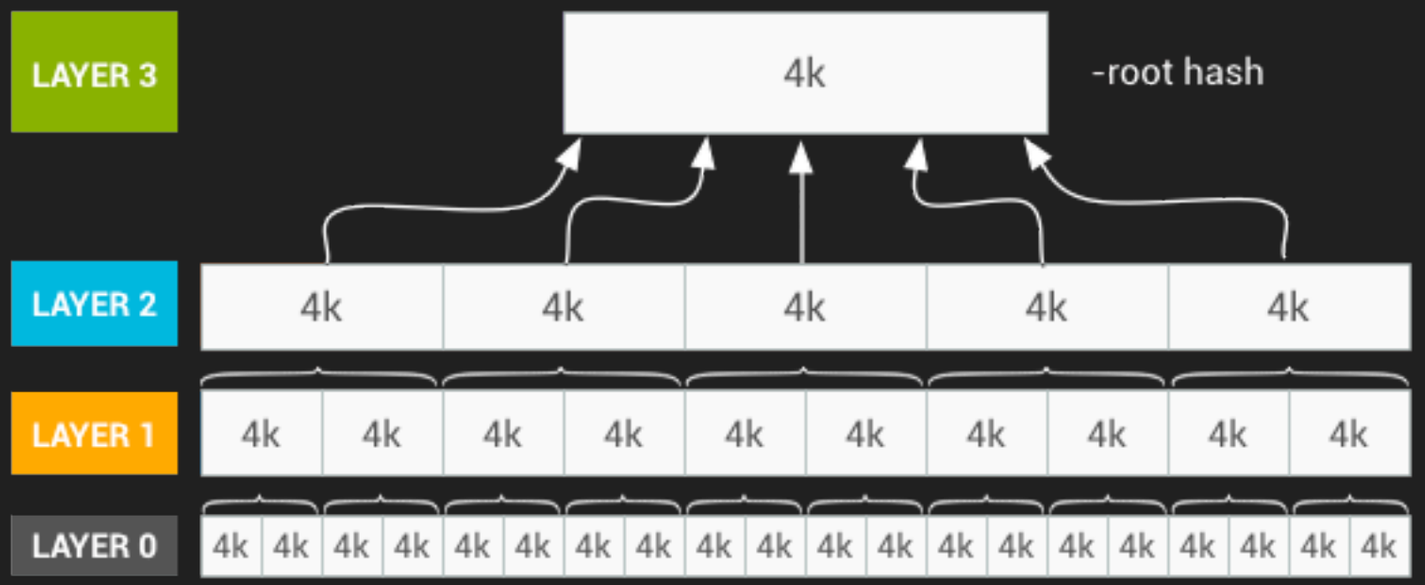
New Filesystem for Android

f2fs



Android 4.4 (2013)

dm-verity (not dm-verify)



Part of Verified Boot

Protection against system image temperament

https://source.android.com/security/verifiedboot/dm-verity

Android 7.0 (2016)

dm-verity + FEC(Forward Error Correction)

author	Sami Tolvanen <samitolvanen@google.com>	2015-12-03 14:26:30 +0000
committer	Mike Snitzer <snitzer@redhat.com>	2015-12-10 10:39:03 -0500
commit	a739ff3f543afbb4a041c16cd0182c8e8d366e70 (patch)	
tree	2613a382283548a1b88e6f2ba589252312d05fff	
parent	bb4d73ac5e4f0a6c4853f35824f6cb2d396a2f9c (diff)	
download	linux-a739ff3f543afbb4a041c16cd0182c8e8d366e70.tar.gz	

dm verity: add support for forward error correction

Add support for correcting corrupted blocks using Reed-Solomon.

Reed-Solomon error-correcting code

Can recover up-to 16-24MB of data on a typical 2-3GB system image

Verified Boot

dm-verity - My thoughts

android / kernel / common / **a73c9bca682673630cd95a7fa55190f53bab73cf**

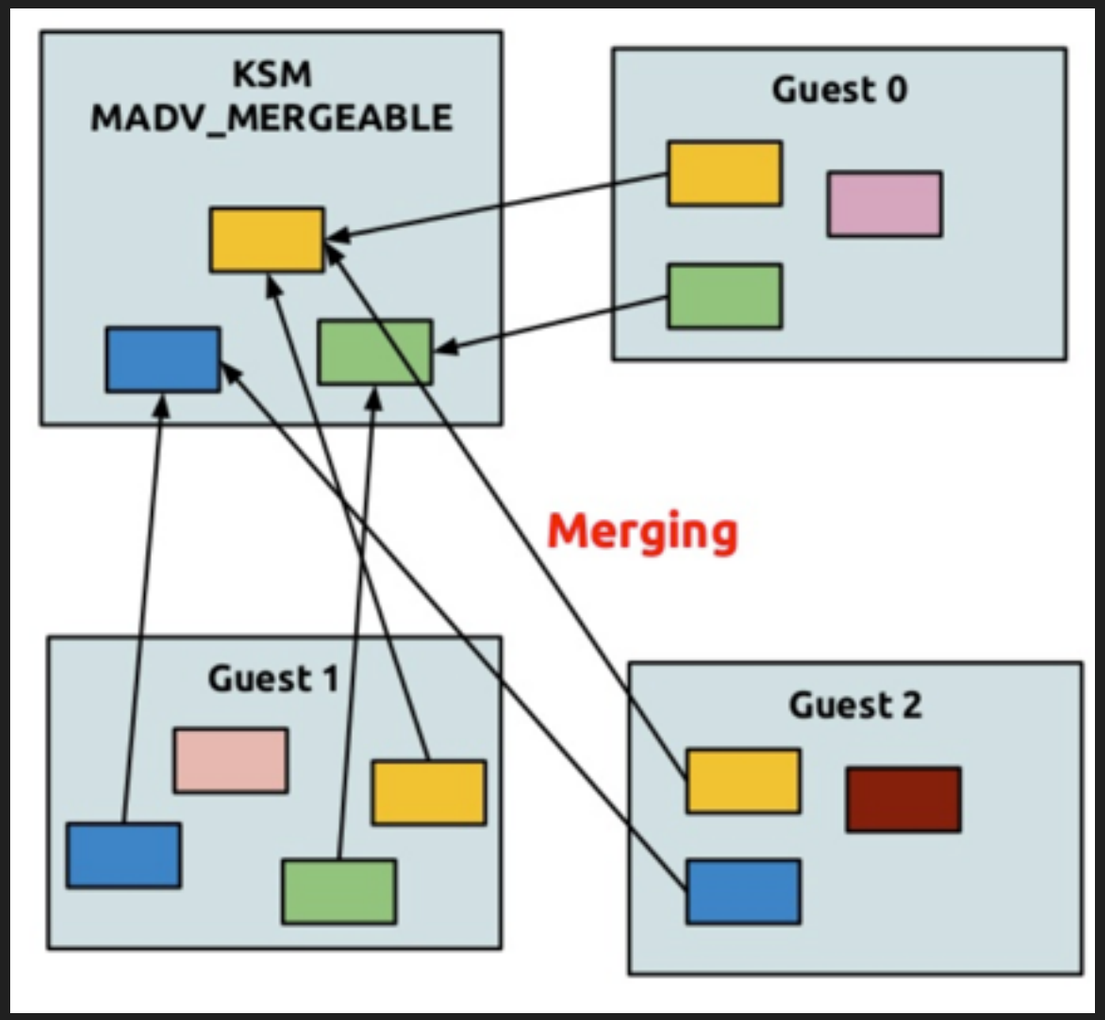
```
   commit  a73c9bca682673630cd95a7fa55190f53bab73cf        [log] [tgz]
   author  Patrik Torstensson <totte@google.com>          Thu Mar 22 18:18:04 2018 -0700
committer  Greg Kroah-Hartman <gregkh@google.com>         Mon Apr 23 14:36:05 2018 +0000
     tree  5cc53651b2d6af5f8e07743a8b848d7f5be37603
   parent  b84f60104105155e456a2341ce19337021c12c5e [diff]
```

```
BACKPORT: dm verity: add 'check_at_most_once' option to only validate hashes once
```

- Calculates hash on every 4K reads
- Offers a relaxed solution that “only validate hashes once”
- Why not just make it R/O from hardware?
- FEC: Your storage shouldn’t be trusted at all if corruptions happened from cold, stale blocks

Android 4.4 (2013)

ksm - Kernel Same-page Merging



https://www.slideshare.net/janghoonsim/kvm-performance-optimization-for-ubuntu

Android 4.4 (2013)

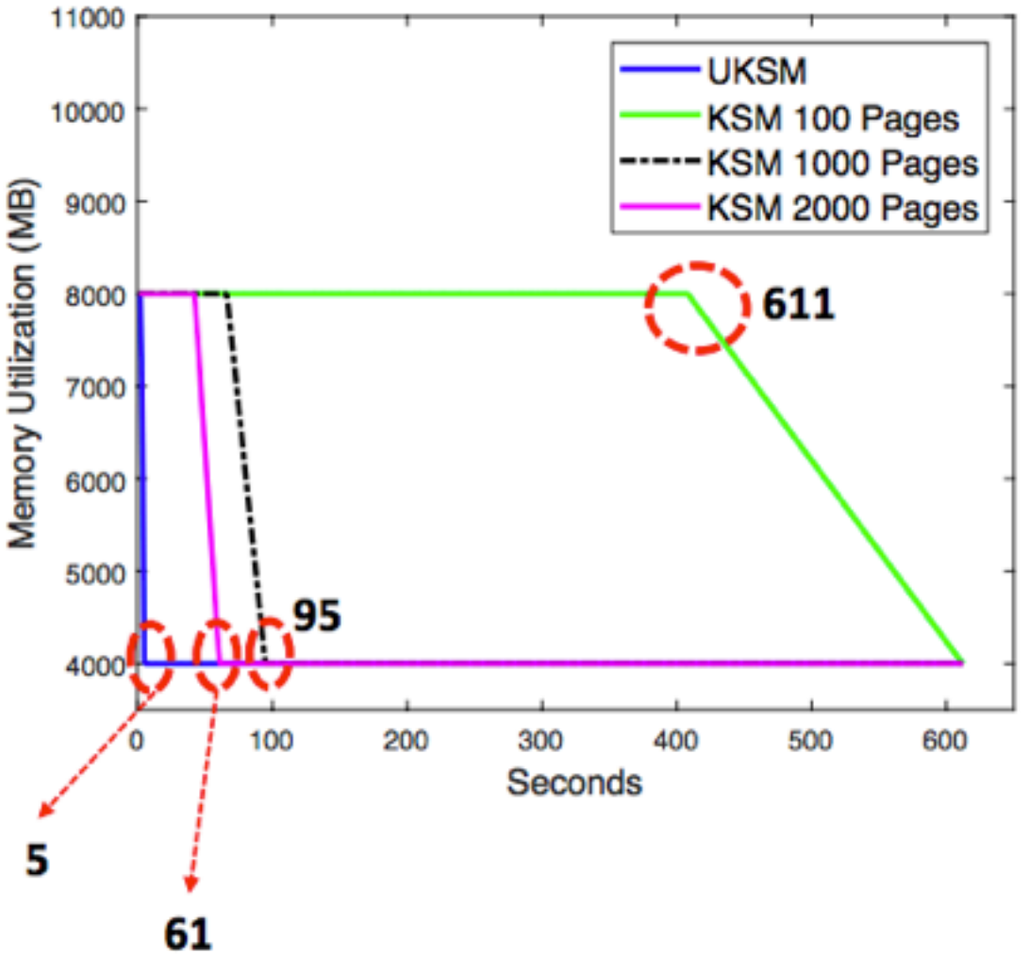
ksm - Kernel Same-page Merging

- Stale development
- Huge overheads from constantly scanning memory pages
- Insignificant savings, inferior compared to zram

From the community

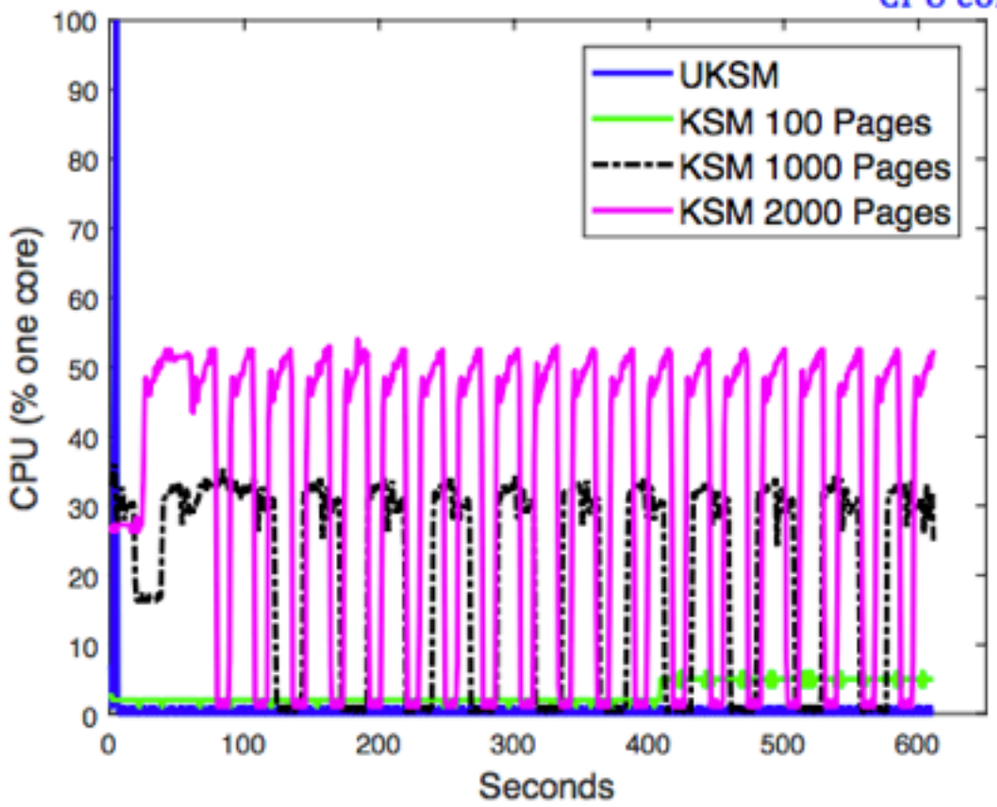
uksm - Ultra Kernel Same-page Merging

Responsiveness Analysis



Setting: Two processes, each with 4GB memory. One contains identical pages while the other random ones.

$$\text{Efficiency} = \frac{\text{memory saving}}{\text{CPU consumption}}$$



UKSM is 8.3×, 12.6×, 11.5× more efficient than KSM at scan speed of 100, 1000, 2000 pages.

UMASS LOWELL
Learning with Purpose

FAST'18

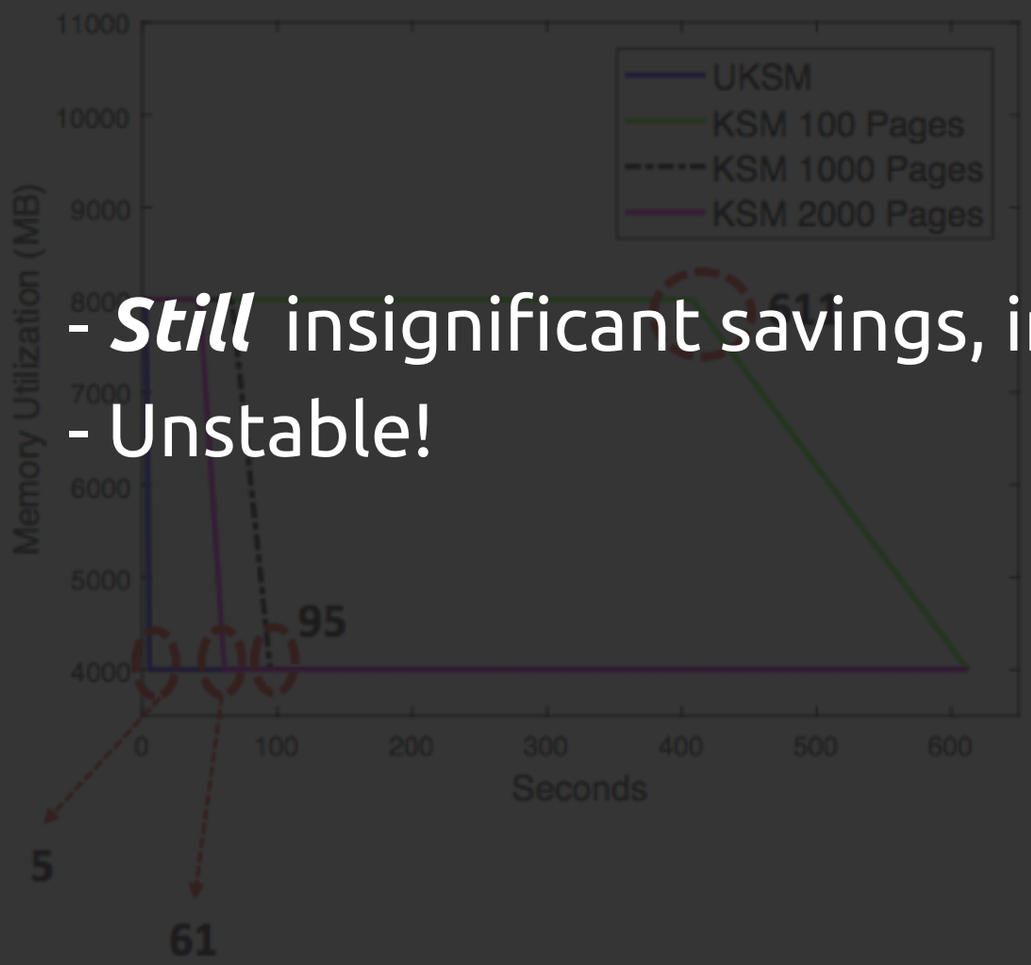
https://www.usenix.org/sites/default/files/conference/protected-files/fast18_slides_xia.pdf

From the community

uksm - Ultra Kernel Same-page Merging

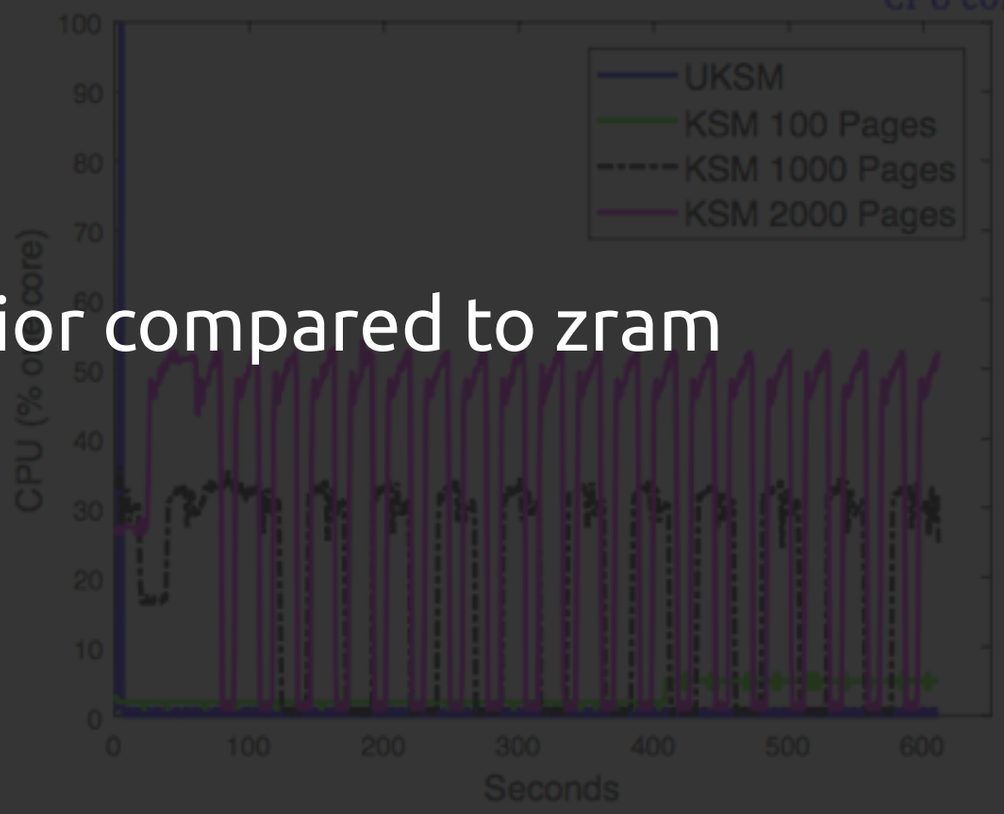
Setting: Two processes, each with 4GB memory. One contains identical pages while the other random ones.

Responsiveness Analysis



- ***Still*** insignificant savings, inferior compared to zram
- Unstable!

$$\text{Efficiency} = \frac{\text{memory saving}}{\text{CPU consumption}}$$

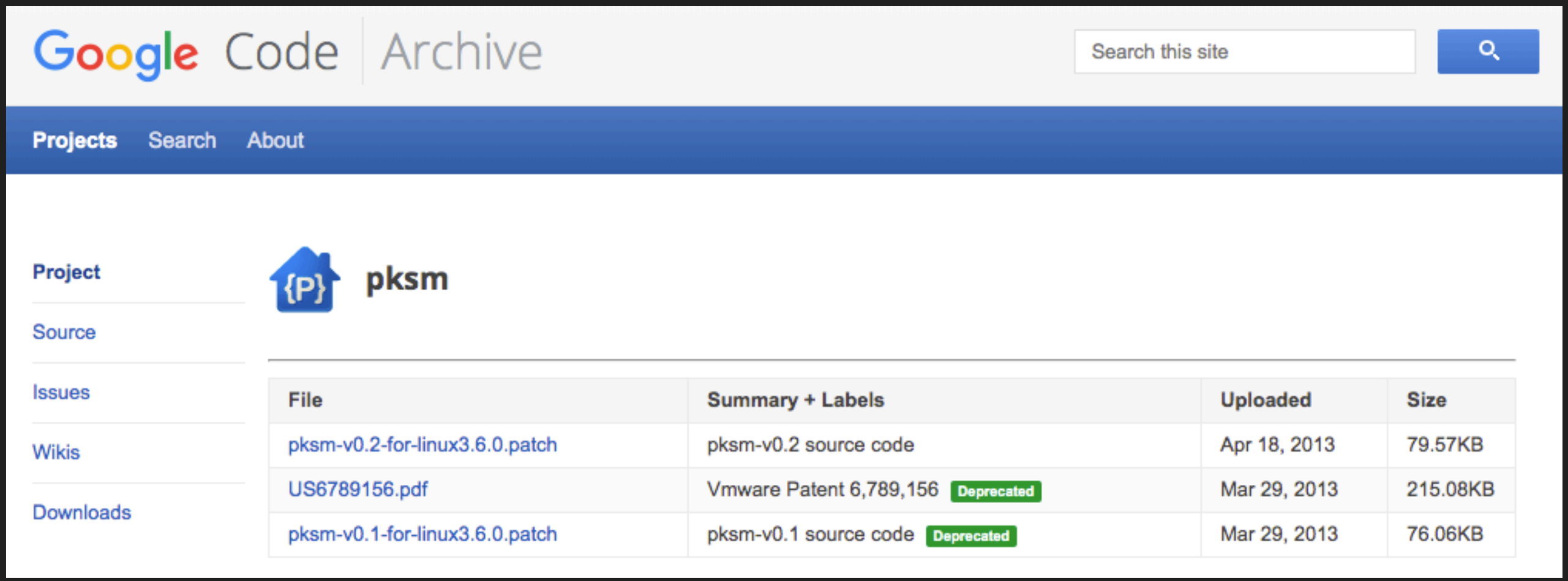


UKSM is 8.3×, 12.6×, 11.5× more efficient than KSM at scan speed of 100, 1000, 2000 pages.

https://www.usenix.org/sites/default/files/conference/protected-files/fast18_slides_xia.pdf

From the community

pksm



https://code.google.com/archive/p/pksm

From the community

pksm

```
fs/proc/meminfo.c          |    6 +
include/linux/ksm.h        |   47 +-
include/linux/mm.h         |    4 +
include/linux/mm_types.h   |    3 +
include/linux/mmzone.h     |    3 +
include/linux/page-flags.h |    9 +
include/linux/pksm.h       |   34 +
mm/Kconfig                 |   22 +
mm/Makefile                |    3 +-
mm/internal.h              |    6 +
mm/memory.c                |   95 ++-
mm/page_alloc.c            |    9 +-
mm/pksm.c                  | 2580 ++++++++++++++++++++++++++++++++++++++++++++++
mm/swap.c                  |    6 +-
mm/vmstat.c                |    3 +
15 files changed, 2807 insertions(+), 23 deletions(-)
```

From the community

pksm

mm/Kconfig

```
config PKSM
    bool "AnonPage-KSM for page merging"
    depends on KSM
help
PKSM is inspired by the Linux kernel project -KSM,
but with a fundamentally rewritten core algorithm. With
an advanced algorithm, PKSM now can transparently scans all anonymously
mapped pages. Since KVM is friendly to KSM, KVM can also benefit from
PKSM.
```

From the community

pksm - My thoughts

- Finally a decent KSM for Android
- But no apparent advantages over zram
- Higher CPU consumption


WHY NOT
BOTH
memegenerator.net

zswap + deduplication

```
From: Srividya Desireddy <srividya.dr@samsung.com>
Date: Wed, 17 Aug 2016 14:31:01 +0530
Subject: [PATCH 1/4] zswap: Share zpool memory of duplicate pages

This patch shares the compressed pool memory of duplicate pages and reduces
compressed pool memory utilized by zswap.

For each page requested for swap-out to zswap, calculate 32-bit checksum of
the page. Search for duplicate pages by comparing the checksum of the new
page with existing pages. Compare the contents of the pages if checksum
matches. If the contents also match, then share the compressed data of the
existing page with the new page. Increment the reference count to check
the number of pages sharing the compressed page in zpool.

If a duplicate page is not found then treat the new page as a 'unique' page
in zswap. Compress the new page and store the compressed data in the zpool.
Insert the unique page in the Red-Black Tree which is balanced based on
32-bit checksum value of the page.

Signed-off-by: Srividya Desireddy <srividya.dr@samsung.com>
---
 mm/zswap.c |  265 ++++++++++++++++++++++++++++++++++++++++++++++++++++++----
 1 file changed, 248 insertions(+), 17 deletions(-)
```

https://lore.kernel.org/patchwork/patch/708326

zram + deduplication

From: Joonsoo Kim <iamjoonsoo.kim@lge.com>
Date: May 12, 2017, 2:30 a.m. UTC
Subject: zram: implement deduplication in zram

This patchset implements deduplication feature in zram. Motivation
is to save memory usage by zram. There are detailed description
about motivation and experimental results on patch #2 so please
refer it.

Thanks.

```
 Documentation/ABI/testing/sysfs-block-zram |  10 ++
 Documentation/blockdev/zram.txt            |   3 +
 drivers/block/zram/Kconfig                 |  14 ++
 drivers/block/zram/Makefile                |   3 +-
 drivers/block/zram/zram_dedup.c            | 254 ++++++++++++++++++++++++++++++
 drivers/block/zram/zram_dedup.h            |  45 +++++
 drivers/block/zram/zram_drv.c              | 184 ++++++++++++++++++---
 drivers/block/zram/zram_drv.h              |  32 +++-
 8 files changed, 507 insertions(+), 38 deletions(-)
 create mode 100644 drivers/block/zram/zram_dedup.c
 create mode 100644 drivers/block/zram/zram_dedup.h
```

https://lore.kernel.org/patchwork/cover/787160

Android 4.4 (2013)

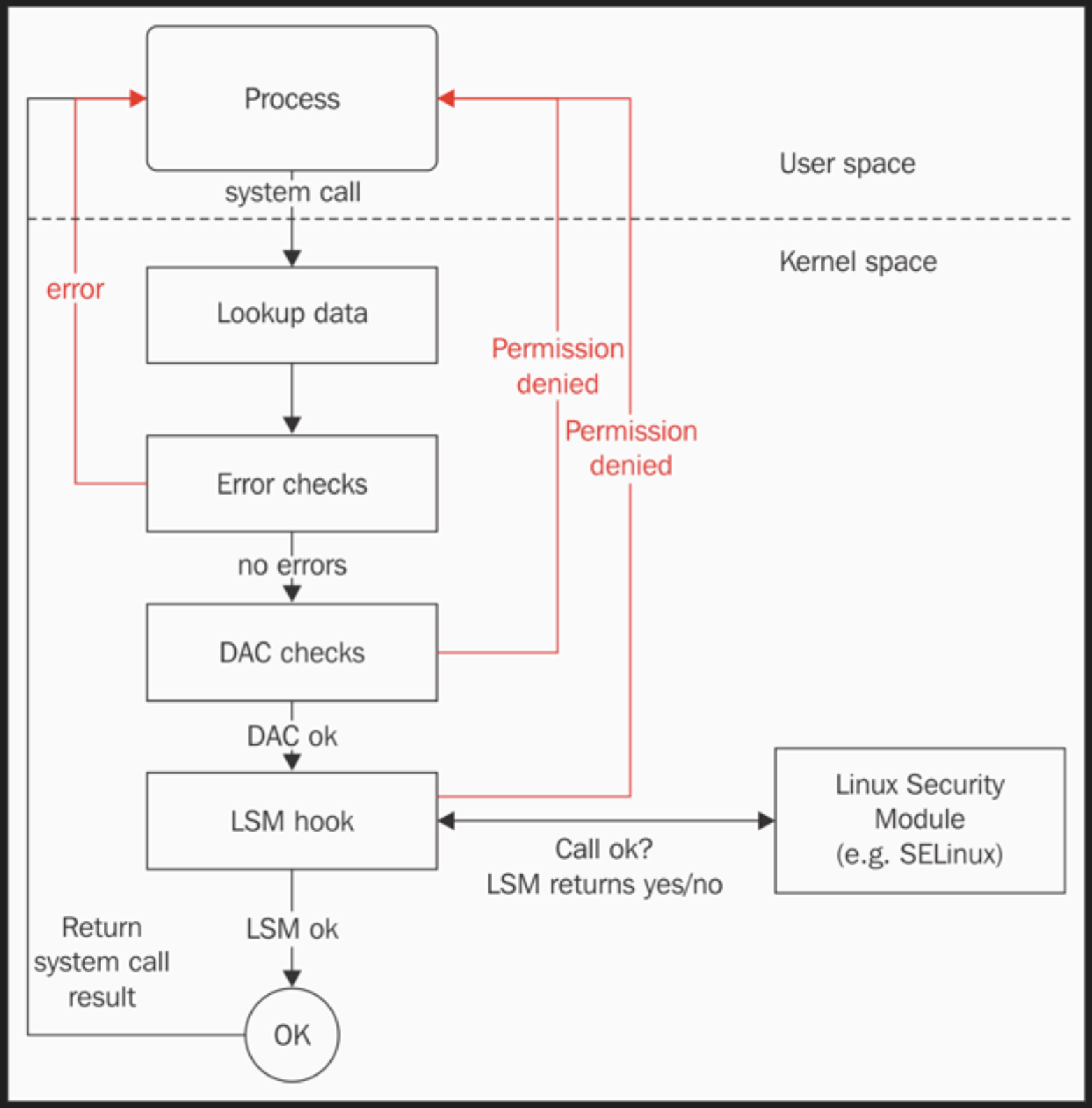
SELinux

DAC(Discretionary Access Control) mechanism

MAC(Mandatory Access Control) mechanism

Android 4.4 (2013)

SELinux



Android 4.4 (2013)

SELinux

```
$ su
# id
uid=0(root) gid=0(root)
# cat /etc/shadow
cat: /etc/shadow: Permission denied
# chmod a+r /etc/shadow
chmod: changing permissions of '/etc/shadow': Permission denied
```

SELinux

Android 4.3	Permissive, testing
Android 4.4	Partial enforcement
Android 5.0	Full enforcement
Android 6.0	ioctl filterings, extremely limited /proc access
Android 7.0	App sandbox, media stack breakdown
Android 8.0	Project Treble compatibility
Android Q	Prevent dynamic code

https://source.android.com/security/selinux

SELinux

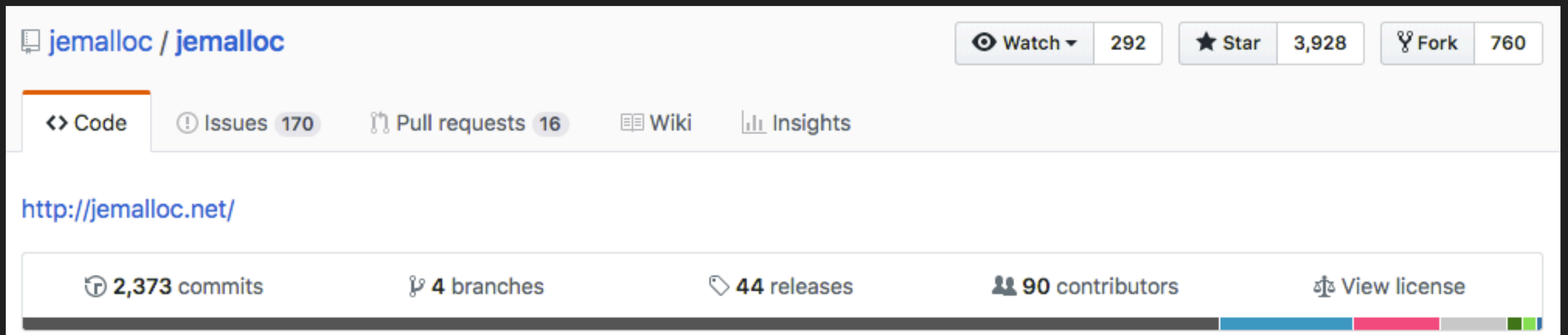
```
OnePlus6:/ $ ls /system/etc/selinux/ -al
system/etc/selinux/:
total 1164
drwxr-xr-x  3 root root    4096 2009-01-01 01:00 .
drwxr-xr-x 23 root root    4096 2019-04-21 14:36 ..
drwxr-xr-x  2 root root    4096 2009-01-01 01:00 mapping
-rw-r--r--  1 root root      65 2009-01-01 01:00 plat_and_mapping_sepolicy.cil.sha256
-rw-r--r--  1 root root   25377 2009-01-01 01:00 plat_file_contexts
-rw-r--r--  1 root root    7212 2009-01-01 01:00 plat_hwservice_contexts
-rw-r--r--  1 root root    4214 2009-01-01 01:00 plat_mac_permissions.xml
-rw-r--r--  1 root root   30422 2009-01-01 01:00 plat_property_contexts
-rw-r--r--  1 root root    1411 2009-01-01 01:00 plat_seapp_contexts
-rw-r--r--  1 root root 1074451 2009-01-01 01:00 plat_sepolicy.cil
-rw-r--r--  1 root root   16325 2009-01-01 01:00 plat_service_contexts
```

“The other mistake is to think that *SELinux* is sane, or should be the default. It's a complex disaster, and makes performance plummet on some things. I turn it off, and I know lots of other sane people do too.”

- Linus Torvalds(2009)

Android 5.0 (2014)

jemalloc memory allocator



Memory allocator from Facebook

Android 5.0 (2014)

jemalloc memory allocator

```
// Default to a single arena for svelte configurations to minimize
// PSS consumed by jemalloc.
common_cflags += [
    "-DANDROID_MAX_ARENAS=1",
    "-DANDROID_LG_TCACHE_MAXCLASS_DEFAULT=16",
]

common_c_local_includes = [
    "src",
    "include",
]

common_product_variables = {
    // Only enable the tcache on non-svelte configurations, to save PSS.
    malloc_not_svelte: {
        cflags: [
            "-UANDROID_MAX_ARENAS",
            "-DANDROID_MAX_ARENAS=2",
            "-DJEMALLOC_TCACHE",
            "-DANDROID_TCACHE_NSLOTS_SMALL_MAX=8",
            "-DANDROID_TCACHE_NSLOTS_LARGE=16",
        ],
    },
}
```

https://android.googlesource.com/platform/external/jemalloc/+/refs/tags/android-cts-9.0_r8/Android.bp

Android 5.0 (2014)

64-bit support

- Android RunTime
- Android NDK
- zygote32/zygote64

When will we get rid of arm32?

Android 5.0 (2014)

Killing arm32 - My thoughts

Apple removed arm32 from iOS 11, subsequent chip designs removed AArch32 decoder and used its die space for improving AArch64 performance.

Processors starting from Apple A9 (iPhone 6s) uses 16kB page size.

“Starting August 1, 2019, your apps published on Google Play will need to support 64-bit architectures.”

Long overdue transition

Android 6.0 (2015)

End of CPU cores hotplugging

Arm	30 ms
x86	110 ms

CPU cores hotplugging latency

* data not representative for modern core designs

Android 6.0 (2015)

End of CPU cores hotplugging

CPU cores hotplugging causes delays on both hardware and software:

- hw: physical architectural hotplugging delays
- sw: expensive Linux CPU hotplugging code paths

Hotplugging isn’t fast enough, hence causing janks.

Disable hotplugging?
Arm hardware LPM and cpuidle got a lot better.
(Contributions by Intel, Linaro and Qualcomm)

Android 6.0 (2015)

CONFIG_HZ=300

```
kernel/Kconfig.hz:

choice
        prompt "Timer frequency"
        default HZ_250
help
 Allows the configuration of the timer frequency. It is customary
 to have the timer interrupt run at 1000 Hz but 100 Hz may be more
 beneficial for servers and NUMA systems that do not need to have
 a fast response for user interaction and that may experience bus
 contention and cacheline bounces as a result of timer interrupts.
 Note that the timer interrupt occurs on each processor in an SMP
 environment leading to NR_CPUS * HZ number of timer interrupts
 per second.
```

Android 6.0 (2015)

CONFIG_HZ=300

CONFIG_HZ	One jiffy	Load balancing
100	10 ms	**20 ms**
250	4 ms	8 ms
300	3.33 ms	6.67 ms
1000	1 ms	2 ms

vsync = 1 / 60 = 16.67 ms

https://source.android.com/devices/tech/debug/jank_jitter

Android 6.0 (2015)

CONFIG_HZ=300

4 arch/arm64/Kconfig

```
@@ -687,9 +687,7 @@ config ARCH_NR_GPIO
687  687
688  688      source kernel/Konfig.preempt
689  689
690       -  config HZ
691       -          int
692       -          default 100
     690  +  source kernel/Konfig.hz
693  691
694  692      config ARCH_HAS_HOLES_MEMORYMODEL
695  693              def_bool y if SPARSEMEM
```

Android 6.0 (2015)

CONFIG_HZ=300

```
1190  1190          bq27541_set_allow_reading(false);
1191  1191          bq27541_temperature_thrshold_update(temp);
1192  1192          if (!bq27541_di->already_modify_smooth)
1193        -               schedule_delayed_work(
1194        -               &bq27541_di->modify_soc_smooth_parameter, 1000);
1195        -       schedule_time =
1196        -               vbat < 3600 ? LOW_BAT_SOC_UPDATE_MS : BATTERY_SOC_UPDATE_MS;
      1193  +               schedule_delayed_work(&bq27541_di->modify_soc_smooth_parameter,
      1194  +                       msecs_to_jiffies(10000));
      1195  +
1197  1196          schedule_delayed_work(&bq27541_di->battery_soc_work,
1198        -                       msecs_to_jiffies(schedule_time));
      1197  +                       msecs_to_jiffies(vbat < 3600 ?
      1198  +                               LOW_BAT_SOC_UPDATE_MS : BATTERY_SOC_UPDATE_MS));
1199  1199      }
```

https://github.com/arter97/android_kernel_oneplus_sdm845/commit/89de7f275

Android 7.0 (2016)

Greg Kroah-Hartman joins Google

Second to Linus Torvalds
linux-stable.git

VTS(Vendor Test Suite) started to check for kernel versions

Android 7.0 (2016)

Android adopts clang(LLVM)

- December 2014 First side-by-side (mostly) Clang build for Nexus 5
- January 2016 Android Platform defaults to Clang
- April 2016 99% Android Platform Clang (valgrind was the last!)
- August 2016 Forbid non-Clang builds (AOSP / Gitiles).
- October 2016 100% Clang userland for Google Pixel.

https://llvm.org/devmtg/2017-10/slides/Hines-CompilingAndroidKeynote.pdf

Android 7.0 (2016)

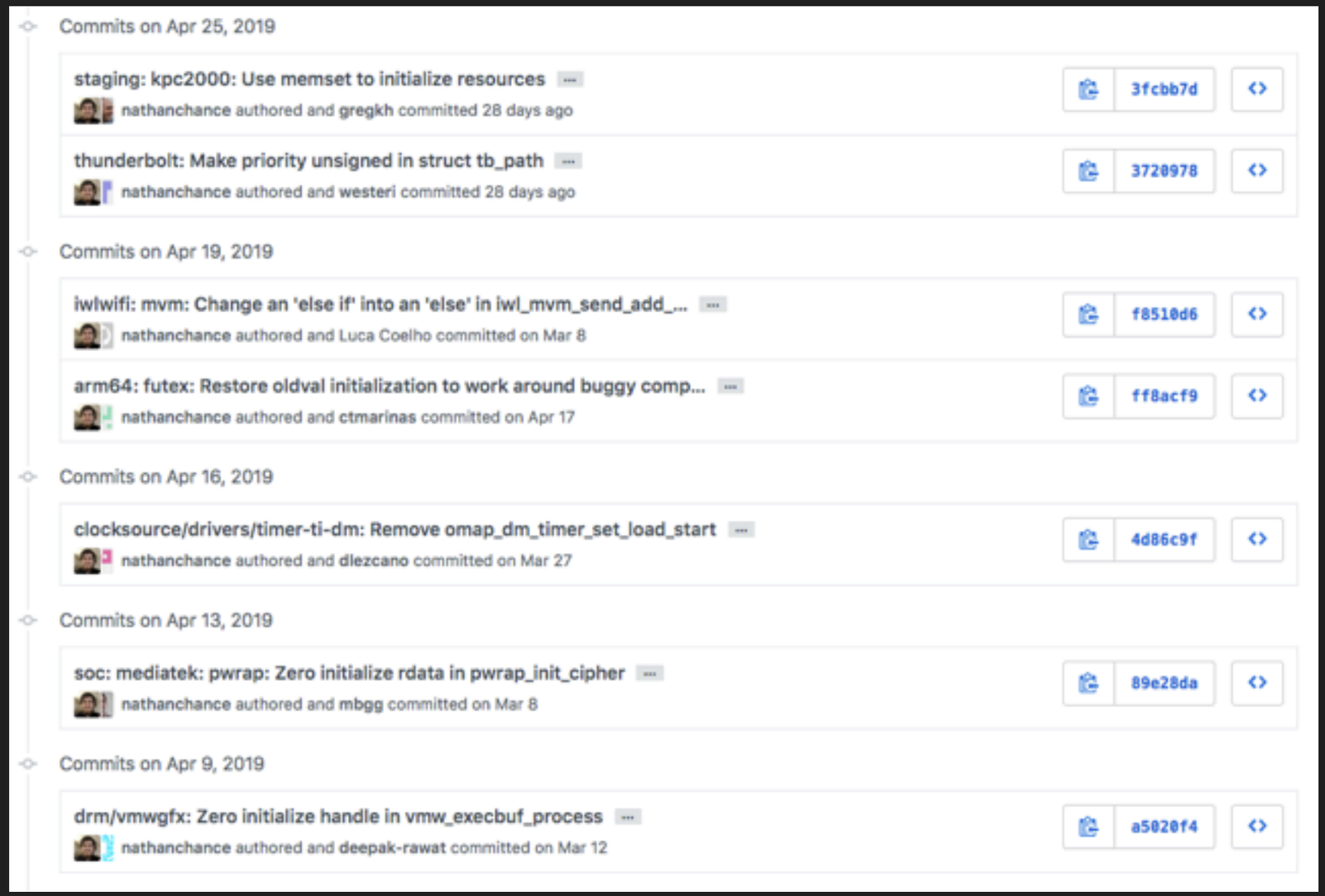
Android adopts clang(LLVM)

- Advanced static analysis
- Better linting
- Address sanitizer
- Integration to non-GPL projects
- Easier LTO(Link-Time Optimizations) + PGO(Profile-Guided Optimizations)

https://events.static.linuxfound.org/sites/events/files/slides/2014-ABS-LLVMLinux.pdf

Android 8.0 (2017)

LLVMLinux on Android kernels



Picks up more coding mistakes

https://github.com/torvalds/linux/commits?author=nathanchance

Android 8.0 (2017)

LLVMLinux on Android kernels

A	B	C	D	E	F	G
Hackbench	**GCC 9**	**Arm GCC 8 + Graphite**	**Korg GCC 8 + Graphite**	**Arm GCC 8**	**Clang 8 + LTO**	**Kernel.org GCC 8**
Run 1 (seconds)	14.707	14.128	17.211	16.235	16.726	16.052
Run 2 (seconds)	16.73	16.98	16.626	15.936	16.099	17.267
Run 3 (seconds)	15.467	16.777	14.672	17.439	16.898	16.509
Average (seconds)	15.63	15.96	16.17	16.54	16.57	16.61
Performance Index	**937**	**904**	**883**	**846**	**843**	**839**
Build time	4m37s	5m40s	5m27s	5m53s	10m38s	5m32s

H	I	J	K	L	M	N
GCC 9 + Graphite	**SDClang 8**	**Clang 9 + LTO**	**SDClang 8 + LTO**	**Clang 8**	**Clang 9**	**GCC 4.9** (no tune)
16.804	16.631	16.7	16.14	17.537	16.826	17.677
17.227	17.104	17.089	17.377	17.114	16.746	17.838
16.117	16.665	16.964	17.282	16.165	17.475	17.145
16.72	16.8	16.92	16.93	16.94	17.02	17.55
828	**820**	**808**	**807**	**806**	**798**	**745**
5m29s	7m22s	9m40s	10m50s	6m48s	6m5s	6m4s

However..

https://github.com/kdrag0n/proton_bluecross/commit/38d2c436bebc

Android 7.0 (2016)

sdcardfs replacing FUSE

Shared storage(/sdcard): vfat style

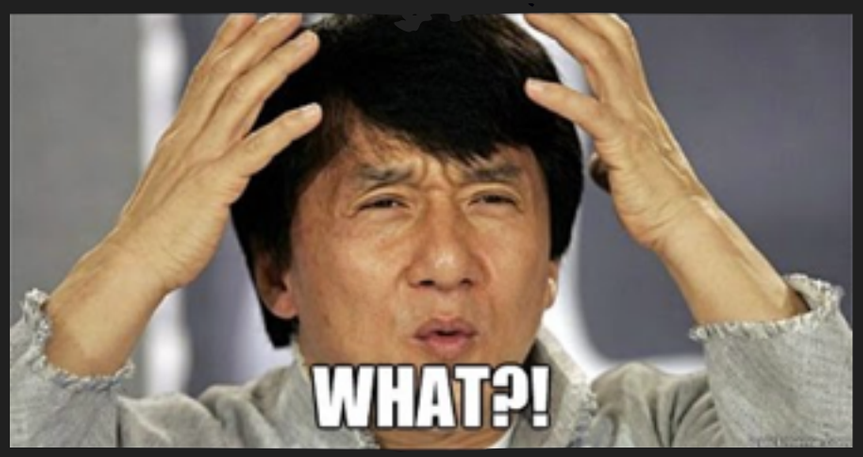
FUSE
(Filesystem in userspace)
to the rescue!

Android 7.0 (2016)

sdcardfs replacing FUSE

Problems with FUSE

- 17% slower sequential operations
- 3.6x slower random operations
- Double caching
  (reading 10MB will consume 20MB of caches)



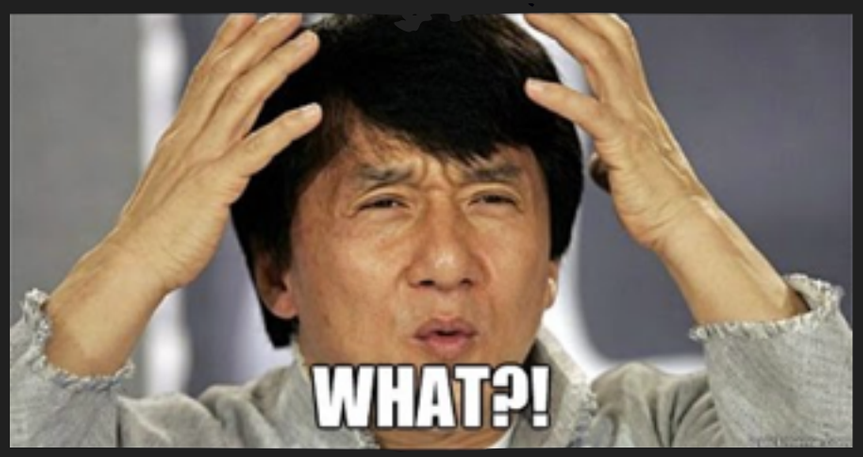
http://fixbugfix.blogspot.com/2015/11/why-does-fuse-on-android-suck.html

Android 7.0 (2016)

sdcardfs replacing FUSE

Problems with FUSE

1. Userspace application issues system call that will be handled by FUSE driver in kernel
2. FUSE driver in kernel notifies userspace daemon (/system/bin/sdcard) about the new request
3. Userspace daemon reads /dev/fuse
4. Userspace daemon parses command and recognizes file operation (e.g. open())
5. Userspace daemon issues system call to the actual filesystem (EXT4)
6. Kernel handles physical data access and sends data back to the userspace
7. Userspace modifies (or not) data and passes it through /dev/fuse to kernel again
8. Kernel completes original system call and moves data to the actual userspace application



http://fixbugfix.blogspot.com/2015/11/why-does-fuse-on-android-suck.html

Samsung took actions

sdcardfs replacing FUSE

Modify wrapfs to replace /system/bin/sdcard(FUSE daemon): sdcardfs

```
 fs/Kconfig                  |   1 +
 fs/Makefile                 |   5 +-
 fs/sdcardfs/Kconfig         |  18 +
 fs/sdcardfs/Makefile        |   7 +
 fs/sdcardfs/dentry.c        | 182 +++++++
 fs/sdcardfs/derived_perm.c  | 290 ++++++++++
 fs/sdcardfs/file.c          | 357 ++++++++++++
 fs/sdcardfs/hashtable.h     | 190 ++++++++
 fs/sdcardfs/inode.c         | 886 ++++++++++++++++++++++++++++++++
 fs/sdcardfs/lookup.c        | 386 ++++++++++++++
 fs/sdcardfs/main.c          | 425 +++++++++++++++
 fs/sdcardfs/mmap.c          |  82 ++++
 fs/sdcardfs/multiuser.h     |  37 ++
 fs/sdcardfs/packagelist.c   | 458 +++++++++++++++++
 fs/sdcardfs/sdcardfs.h      | 493 ++++++++++++++++++
 fs/sdcardfs/strtok.h        |  75 +++
 fs/sdcardfs/super.c         | 229 +++++++++
 include/linux/namei.h       |   3 +
 include/uapi/linux/magic.h |   2 +
 19 files changed, 4124 insertions(+), 2 deletions(-)
```

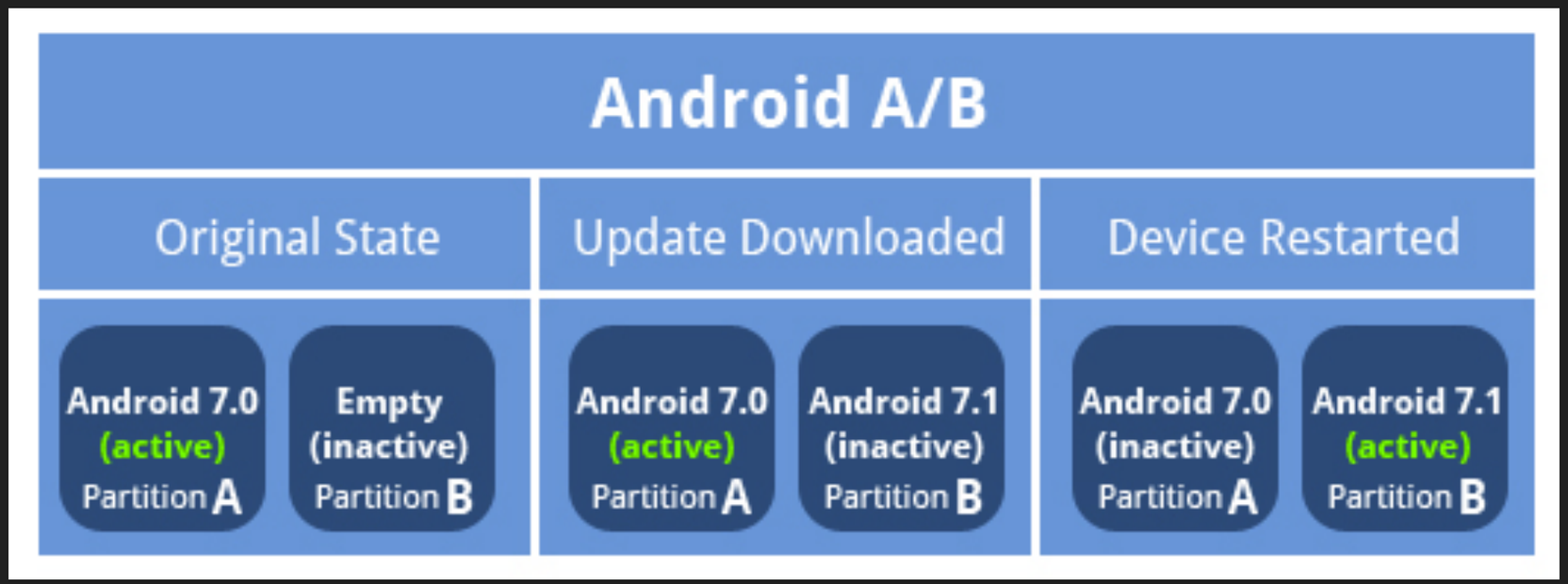
AOSP merged sdcardfs, enabled by default in Oreo

sdcardfs replacing FUSE

```
OnePlus6:/ $ mount | grep -i sdcardfs
/data/media on /mnt/runtime/default/emulated type sdcardfs
(rw,nosuid,nodev,noexec,noatime,fsuid=1023,fsgid=1023,gid=1015,multiuser,mask=6,derive_gid,default_normal)
/data/media on /storage/emulated type sdcardfs
(rw,nosuid,nodev,noexec,noatime,fsuid=1023,fsgid=1023,gid=1015,multiuser,mask=6,derive_gid,default_normal)
/data/media on /mnt/runtime/read/emulated type sdcardfs
(rw,nosuid,nodev,noexec,noatime,fsuid=1023,fsgid=1023,gid=9997,multiuser,mask=23,derive_gid,default_normal)
/data/media on /mnt/runtime/write/emulated type sdcardfs
(rw,nosuid,nodev,noexec,noatime,fsuid=1023,fsgid=1023,gid=9997,multiuser,mask=7,derive_gid,default_normal)
```

Android 7.0 (2016)

A/B partition layout(seamless updates)



Android 7.0 (2016)

A/B partition layout(seamless updates)

```
root@localhost:/# ls /dev/block/sd? | while read node; do
echo -e 'unit GB\nprint' | parted $node; done 2>&1 | grep '_a\|_b' |
sort -k4 -g | tail -n10
Number  Start   End     Size    File system  Name          Flags
39      1.60GB  1.67GB  0.07GB               boot_b
58      2.89GB  2.96GB  0.07GB               boot_a
15      6.35GB  6.45GB  0.10GB  ext4         odm_a
16      6.45GB  6.56GB  0.10GB  ext4         odm_b
32      1.40GB  1.52GB  0.13GB               modem_b       msftdata
 4      0.00GB  0.13GB  0.13GB               modem_a
16      0.28GB  1.35GB  1.07GB  ext2         vendor_a
44      1.67GB  2.75GB  1.07GB  ext2         vendor_b
13      0.35GB  3.35GB  3.00GB  ext2         system_a
14      3.35GB  6.35GB  3.00GB  ext2         system_b
```

```
# fastboot set_active b
```

Android 8.0 (2017)

SECCOMP - SECure COMPuting mode

Some syscalls may be dangerous even from non-privileged permissions

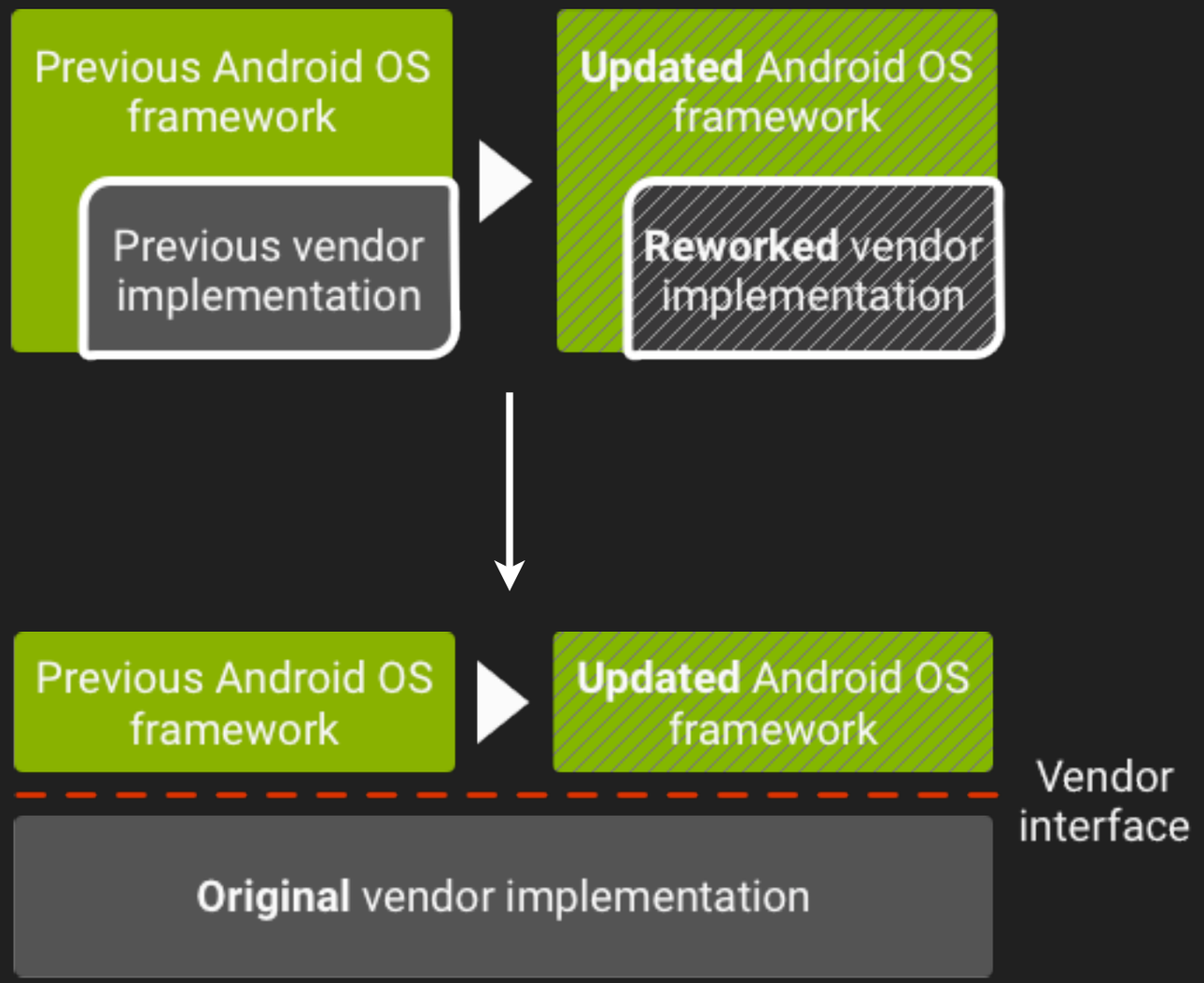
```
03-09 16:39:32.122 15107 15107 I crash_dump32: performing dump of process 14942 (target tid = 14971)
03-09 16:39:32.127 15107 15107 F DEBUG   : *** *** *** *** *** *** *** *** *** *** *** *** *** *** ***
03-09 16:39:32.127 15107 15107 F DEBUG   : Build fingerprint: 'google/sailfish/sailfish:O/
OPP1.170223.013/3795621:userdebug/dev-keys'
03-09 16:39:32.127 15107 15107 F DEBUG   : Revision: '0'
03-09 16:39:32.127 15107 15107 F DEBUG   : ABI: 'arm'
03-09 16:39:32.127 15107 15107 F DEBUG   : pid: 14942, tid: 14971, name: WorkHandler  >>> com.redacted <<<
03-09 16:39:32.127 15107 15107 F DEBUG   : signal 31 (SIGSYS), code 1 (SYS_SECCOMP), fault addr --------
03-09 16:39:32.127 15107 15107 F DEBUG   : Cause: seccomp prevented call to disallowed system call 55
03-09 16:39:32.127 15107 15107 F DEBUG   :     r0 00000091  r1 00000007  r2 ccd8c008  r3 00000001
03-09 16:39:32.127 15107 15107 F DEBUG   :     r4 00000000  r5 00000000  r6 00000000  r7 00000037
```

Getting more aggressive as Android develops,
be careful with using non-exposed system calls

https://android-developers.googleblog.com/2017/07/seccomp-filter-in-android-o.html

Android 8.0 (2017)

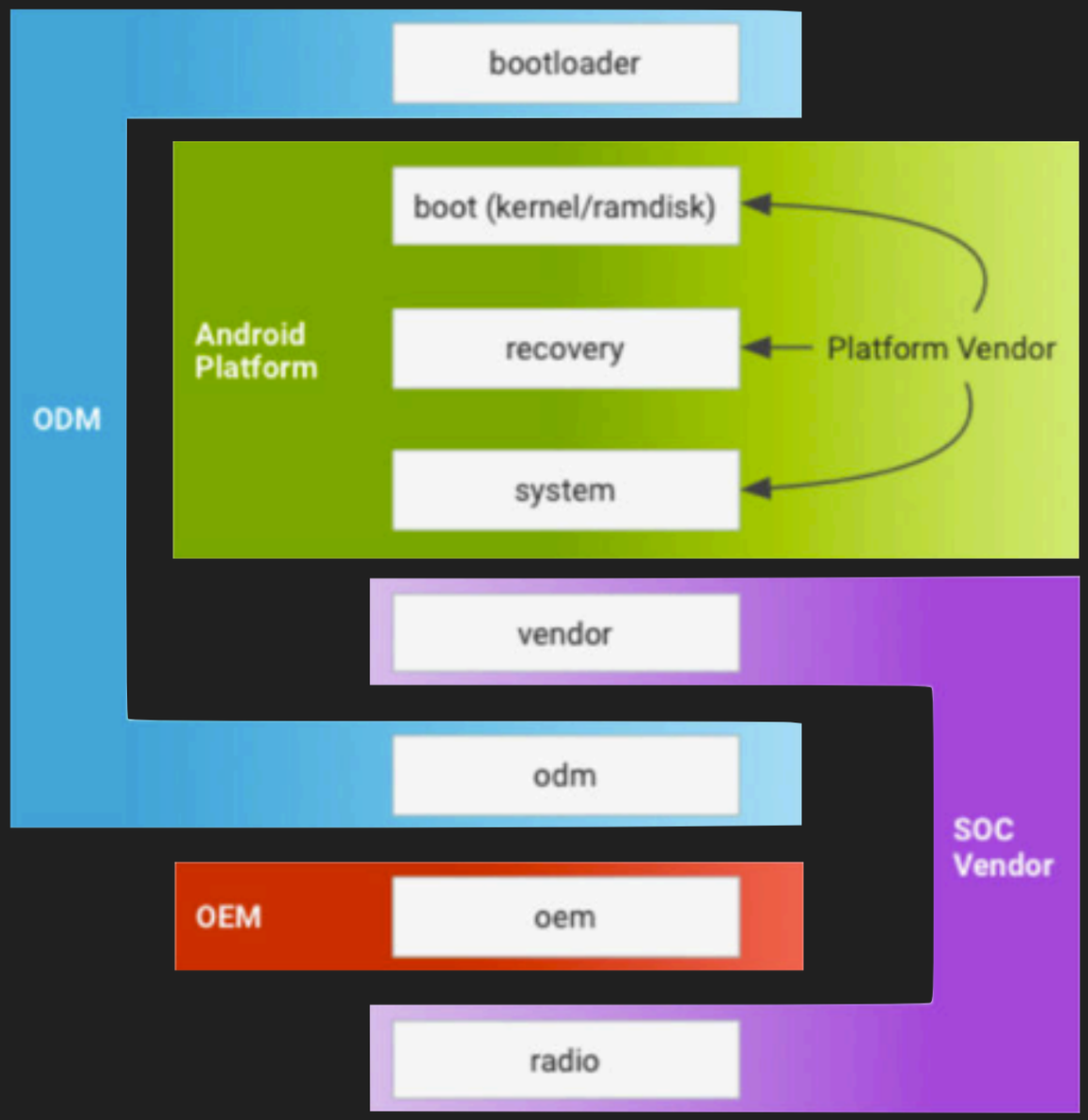
Project Treble - Brain-transplant



https://source.android.com/devices/architecture

Android 8.0 (2017)

Project Treble - Brain-transplant



https://arstechnica.com/gadgets/2017/09/android-8-0-oreo-thoroughly-reviewed/2/#h4

Android 8.0 (2017)

Project Treble - Brain-transplant

1. Standardized HAL
2. Standardized namespace
3. Loadable kernel modules

“Android's biggest re-architecture, ever”

- Dave Burke(VP Engineering, Android)

https://source.android.com/devices/architecture

Android 8.0 (2017)

Project Treble - Brain-transplant

```
OnePlus6:/ # ls /system/lib/hw
android.hardware.tests.libhwbinder@1.0-impl.so android.hidl.memory@1.0-impl.so audio.a2dp.default.so

OnePlus6:/ # ls /vendor/lib/hw
android.hardware.audio.effect@2.0-impl.so
android.hardware.audio.effect@4.0-impl.so
android.hardware.audio@2.0-impl.so
android.hardware.audio@4.0-impl.so
android.hardware.bluetooth@1.0-impl-qti.so
android.hardware.boot@1.0-impl.so
android.hardware.camera.provider@2.4-impl.so
android.hardware.contexthub@1.0-impl.generic.so
android.hardware.drm@1.0-impl.so
android.hardware.gatekeeper@1.0-impl-qti.so
android.hardware.gnss@1.0-impl-qti.so
android.hardware.graphics.composer@2.1-impl.so
android.hardware.graphics.mapper@2.0-impl-qti-display.so
android.hardware.keymaster@3.0-impl-qti.so
android.hardware.light@2.0-impl.so
android.hardware.memtrack@1.0-impl.so
android.hardware.power@1.0-impl.so
android.hardware.renderscript@1.0-impl.so
android.hardware.sensors@1.0-impl.so
android.hardware.soundtrigger@2.1-impl.so
android.hardware.thermal@1.0-impl.so
android.hardware.vibrator@1.0-impl.so
android.hardware.vr@1.0-impl.so
audio.primary.default.so
audio.primary.sdm845.so
audio.r_submix.default.so
audio.usb.default.so
bootctrl.sdm845.so
camera.qcom.so
com.qti.chi.override.so
com.qualcomm.qti.ant@1.0-impl.so
com.qualcomm.qti.bluetooth_audio@1.0-impl.so
fingerprint.sdm845.so
goodix.fingerprint.sdm845.so
goodix.fod.sdm845.so
gralloc.default.so
gralloc.sdm845.so
hwcomposer.sdm845.so
lights.sdm845.so
local_time.default.so
memtrack.sdm845.so
power.default.so
power.qcom.so
silead.fod.sdm845.so
sound_trigger.primary.sdm845.so
thermal.sdm845.so
vendor.qti.esepowermanager@1.0-impl.so
vendor.qti.gnss@2.0-impl.so
vendor.qti.hardware.alarm@1.0-impl.so
vendor.qti.hardware.fm@1.0-impl.so
vendor.qti.hardware.iop@2.0-impl.so
vendor.qti.hardware.qteeconnector@1.0-impl.so
vendor.qti.hardware.sensorscalibrate@1.0-impl.so
vibrator.default.so
vr.sdm845.so
vulkan.sdm845.so
```

Android 9.0 (2018)

Linux kernel + CFI & LTO

86% of Android vulnerabilities are memory safety related.

Most vulnerabilities are exploited by changing the normal control flow to perform malicious activities.

CFI(Control Flow Integrity) is a security mechanism that disallows changes to the original control flow graph of a compiled binary.

Android 8.1: CFI in media stack
Android 9.0: CFI in more components & kernel

https://source.android.com/devices/tech/debug/cfi

Android 9.0 (2018)

Linux kernel + CFI & LTO

LLVM CFI requires LTO(Link-Time Optimizations).

Rather than optimizing on a per-file basis, LTO optimizes the whole program.
LTO allows a whole range of optimizations, most notably cross-file inlining.

* SQLite provides an “amalgamated” source code for the same effect.

https://source.android.com/devices/tech/debug/cfi

Qualcomm - 2018

Speculative Page Faults

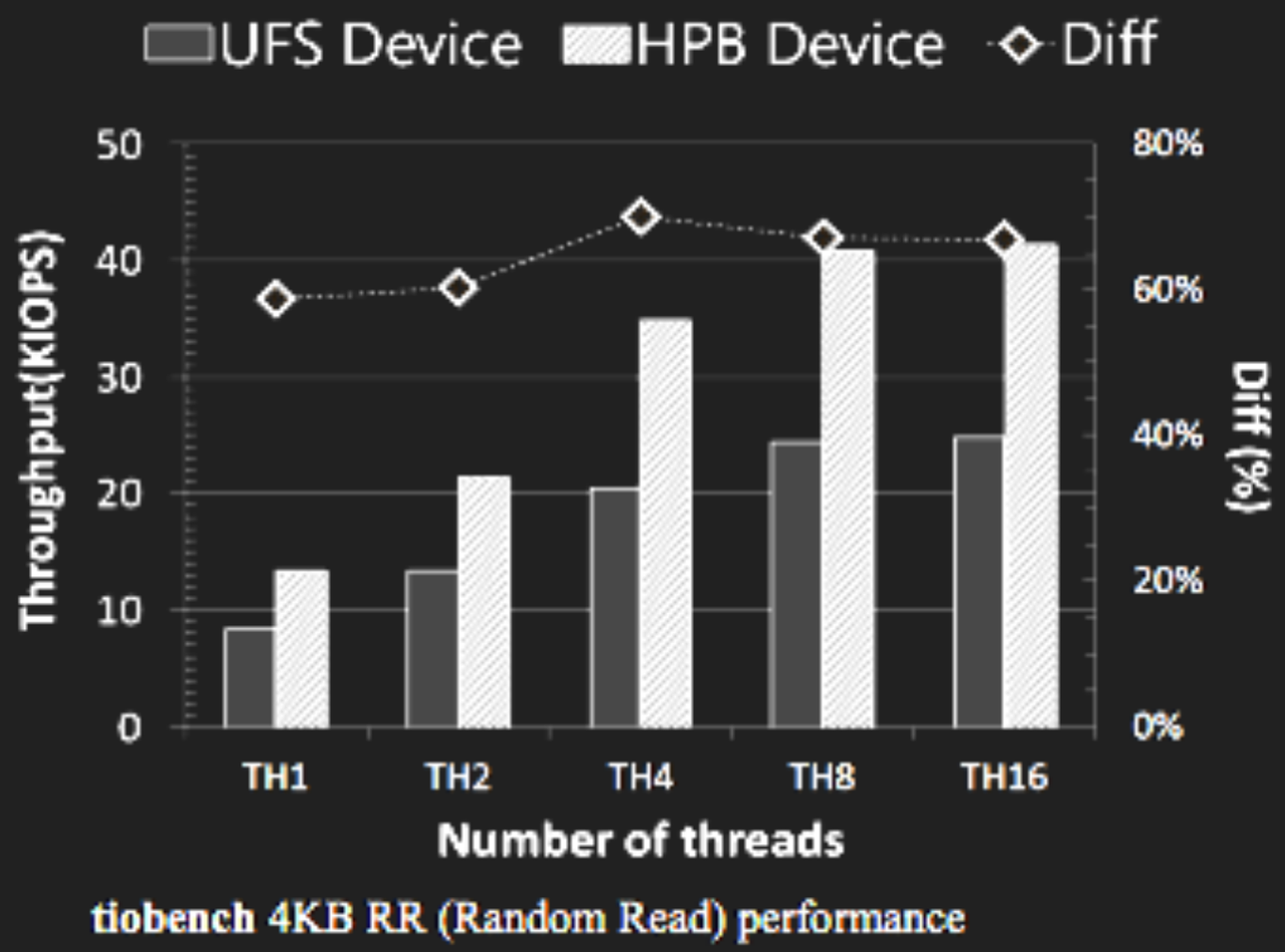
Avoid holding mmap_sem during page faults

Backported to Linux 4.9(sdm845) by Qualcomm

https://lwn.net/Articles/725607
https://lwn.net/Articles/730531

Samsung & Google - 2018

HPB - Host Performance Booster



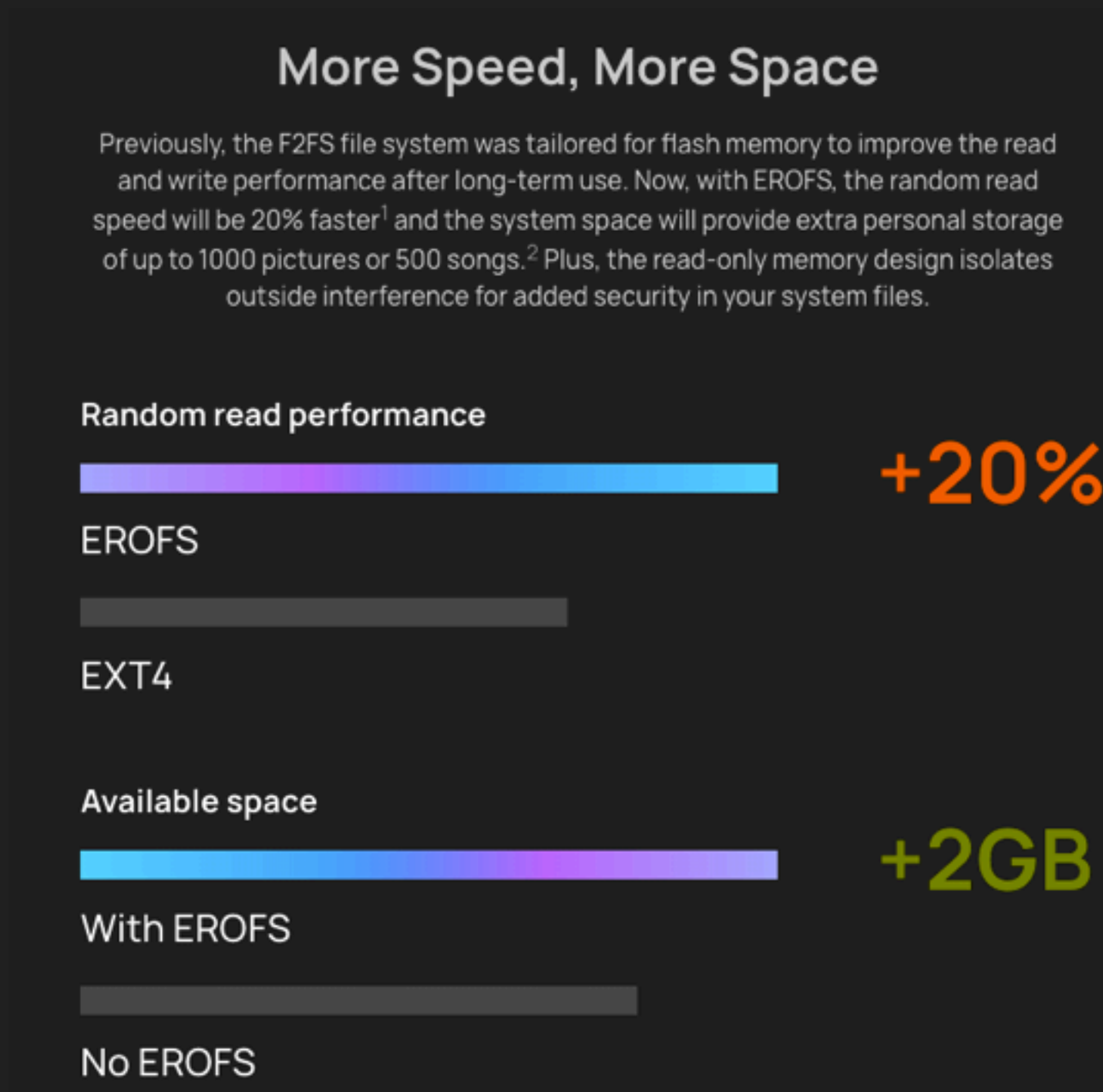
tiobench 4KB RR (Random Read) performance

Caching L2P(Logical-to-Physical) mapping table to host RAM
First appeared in the Pixel 3

https://www.usenix.org/conference/hotstorage17/program/presentation/jeong

Huawei - 2018/2019

erofs - Extendable Read-Only File-System



Read-only filesystem by Huawei

- Transparent compression
- Optimized for random reads
- EMUI 9.1(Mate 20, P20)

https://www.phoronix.com/scan.php?page=news_item&px=EROFS-File-System
https://consumer.huawei.com/en/emui

Android Q (2019)

APEX

Modular, less fragmented updates

Android O	Modular GPU drivers updates Project Treble
Android Q	APEX for updating shared system libraries

Android Q (2019)
APEX

New usecase for loopback(/dev/loop0) devices on Android

1. The APEX manager verifies the APEX file.
2. If the APEX file is verified, the internal database of the APEX manager is updated to reflect that the APEX file will be activated at next boot.
3. The requestor of the install receives a broadcast upon successful verification of the package.
4. To continue the installation, the system automatically reboots the device.
5. At reboot, the APEX manager starts, reads the internal database, and does the following for each APEX file listed:
    i. Verifies the APEX file.
    ii. Creates a loopback device from the APEX file.
    iii. Creates a device mapper block device on top of the loopback device.
    iv. Mounts the device mapper block device onto a unique path (for example, /apex/name@ver).

When all APEX files listed in the internal database are mounted, the APEX manager provides a binder service for other system components to query information about the installed APEX files.

Android Q (2019)

Adiantum encryption

arm32 lacks AES hardware instructions

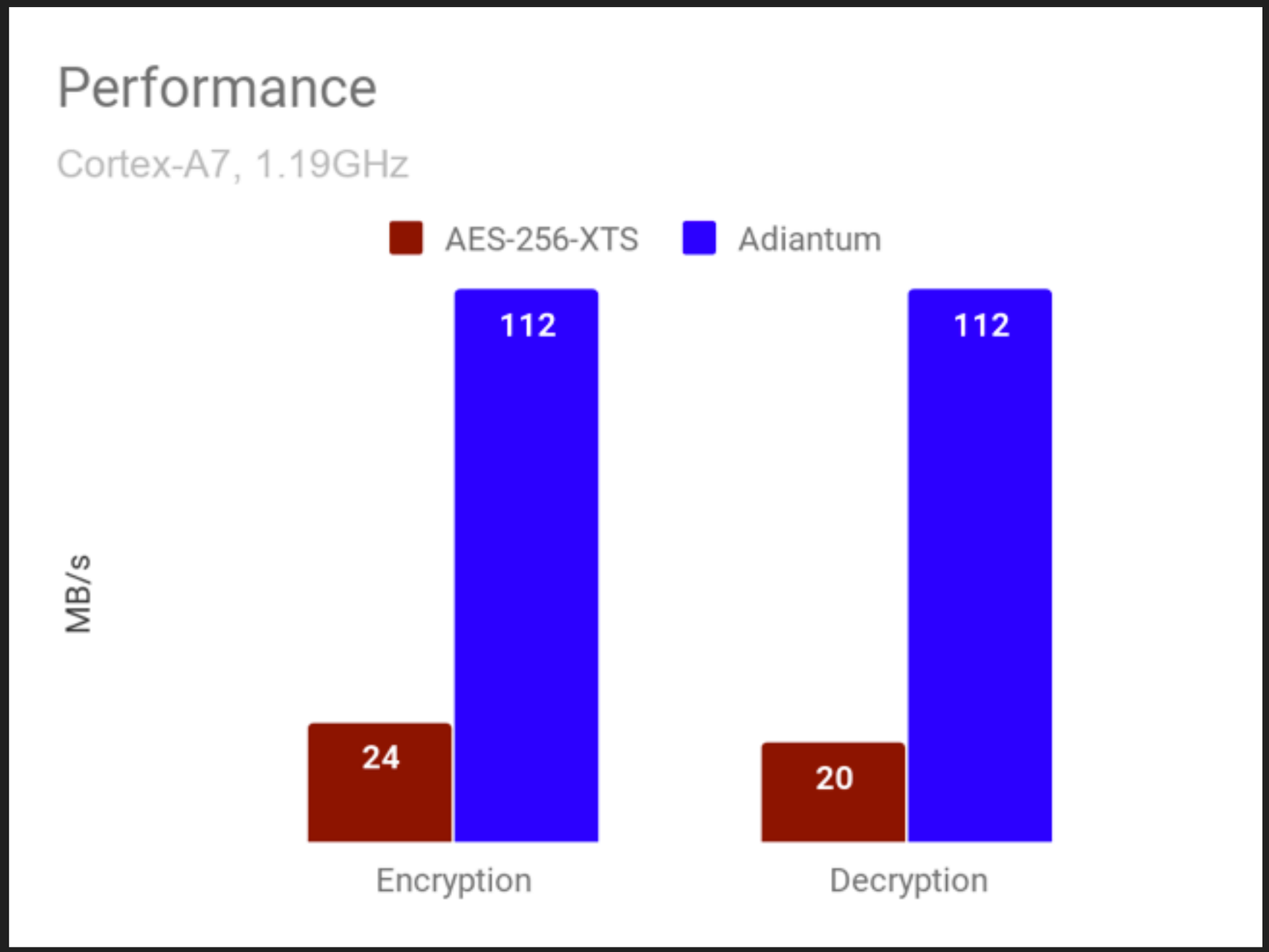
Speck > Linux 4.17
NSA controversy

Removed from Linux 4.19
(Backported to linux-stable!)

Adiantum > Linux 5.0

Android Q (2019)

Adiantum encryption



Android Q (2019)

Scoped Storage

https://cafe.naver.com/develoid/867488

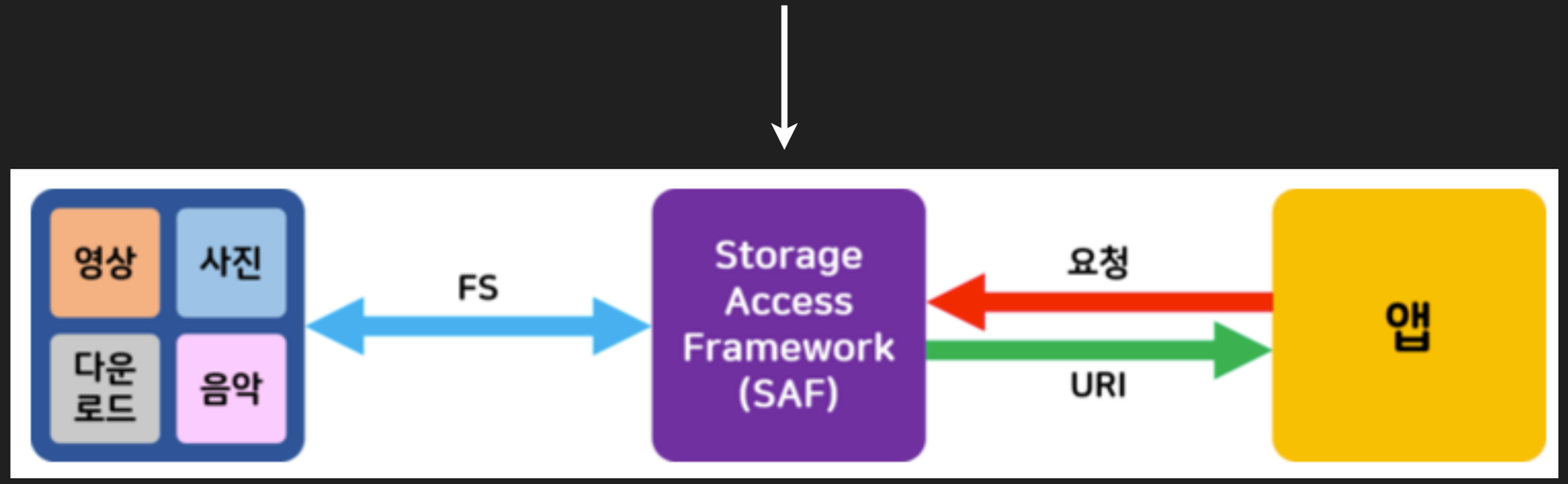
Android Q (2019)

Scoped Storage

https://cafe.naver.com/develoid/867488

Android Q (2019)

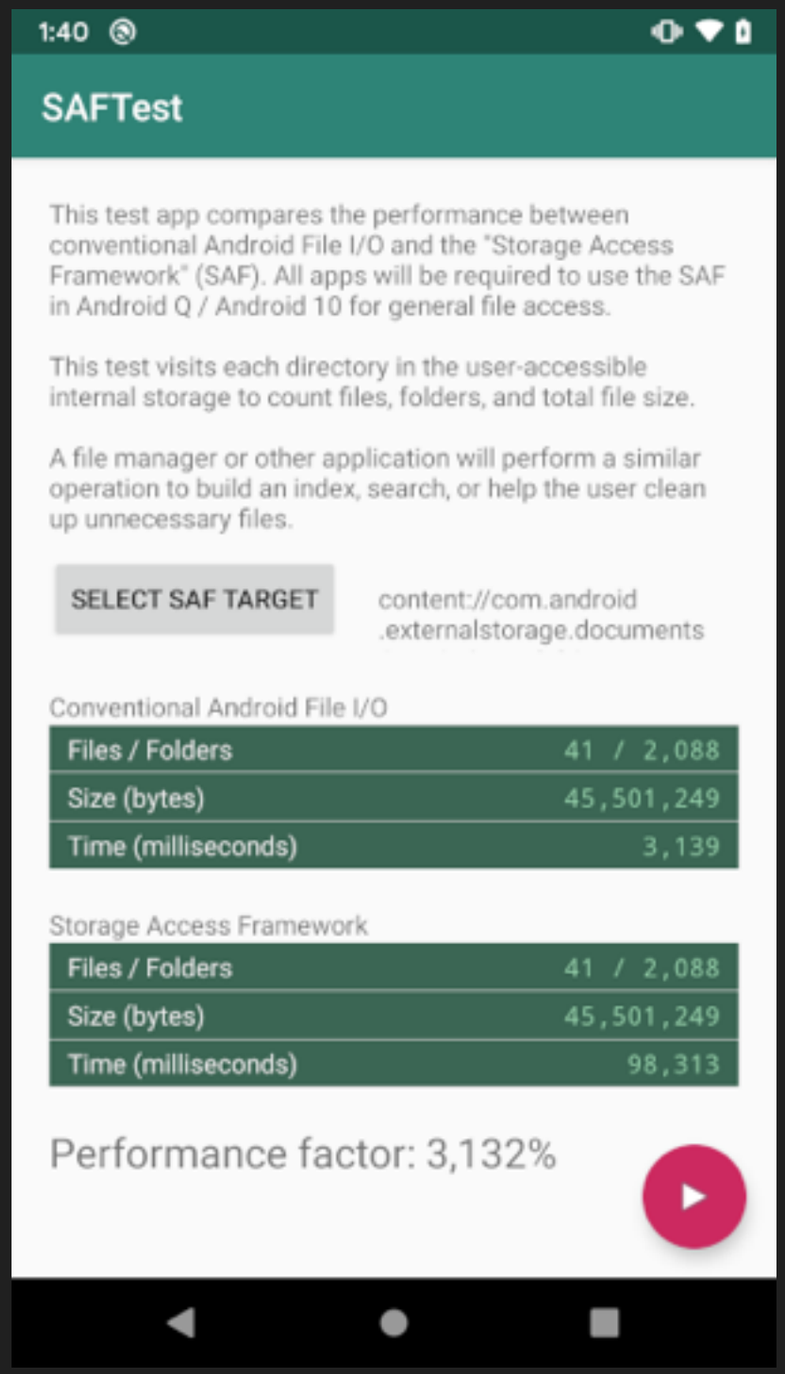
Scoped Storage



https://cafe.naver.com/develoid/867488

Android Q (2019)

SAF, a.k.a. Overhead fest



https://github.com/tliebeck/saftest

Thank You !