Block-Based OTAs

You can enable block-based over-the-air (OTA) updates for new devices running Android 5.0. OTA is the mechanism by which OEMs remotely update the system partition of a device:

- **Android 5.0** and later versions use block OTA updates to ensure that each device uses the exact same partition. Instead of comparing individual files and computing binary patches, block OTA handles the entire partition as one file and computes a single binary patch, ensuring the resultant partition contains exactly the intended bits. This allows the device system image to achieve the same state via fastboot or OTA.
- **Android 4.4** and earlier versions used file OTA updates, which ensured devices contained similar file contents, permissions, and modes, but allowed metadata such as timestamps and the layout of the underlying storage to vary between devices based on the update method.

Because block OTA ensures that each device uses the same partition, it enables the use of dm-verity to cryptographically sign the system partition. For details on dm-verity, see Verified Boot (https://source.android.com/security/verifiedboot/index.html).

Note: You must have a working block OTA system before using dm-verity.

Recommendations

For devices launching with Android 5.0 or later, use block OTA updates in the factory ROM. To generate a block-based OTA for subsequent updates, pass the `--block` option to `ota_from_target_files`.

For devices that launched with Android 4.4 or earlier, use file OTA updates. While it is possible to transition devices by sending a full block OTA of Android 5.0 or later, it requires sending out a full OTA that is significantly larger than an incremental OTA (and is therefore discouraged).

Because dm-verity requires bootloader support found only in new devices shipping with Android 5.0 or later, you *cannot* enable dm-verity for existing devices.

Developers working on the Android OTA system (the recovery image and the scripts that generate OTAs) can keep up with changes by subscribing to the android-ota@googlegroups.com (https://groups.google.com/forum/#!forum/android-ota) mailing list.

File vs. Block OTAs

During a file-based OTA, Android attempts to change the contents of the system partition at the filesystem layer (on a file-by-file basis). The update is not guaranteed to write files in a consistent order, have a consistent last modified time or superblock, or even place the blocks in the same location on the block device. For this reason, file-based OTAs fail on a dm-verity-enabled device; after the OTA attempt, the device does not boot.

During a block-based OTA, Android serves the device the difference between the two block images (rather than two sets of files). The update checks a device build against the corresponding build server at the block level (below the filesystem) using one of the following methods:

- **Full update**. Copying the full system image is simple and makes patch generation easy but also generates large images that can make applying patches expensive.
- **Incremental update**. Using a binary differ tool generates smaller images and makes patch application easy, but is memory-intensive when generating the patch itself.

Note: `adb fastboot` places the exact same bits on the device as a full OTA, so flashing is compatible with block OTA.

Updating unmodified systems

For devices with *unmodified* system partitions running Android 5.0, the download and install process for a block OTA remains the same as for a file OTA. However, the OTA update itself might include one or more of the following differences:

- **Download size**. Full block OTA updates are approximately the same size as full file OTA updates, and incremental updates can be just a few megabytes larger.

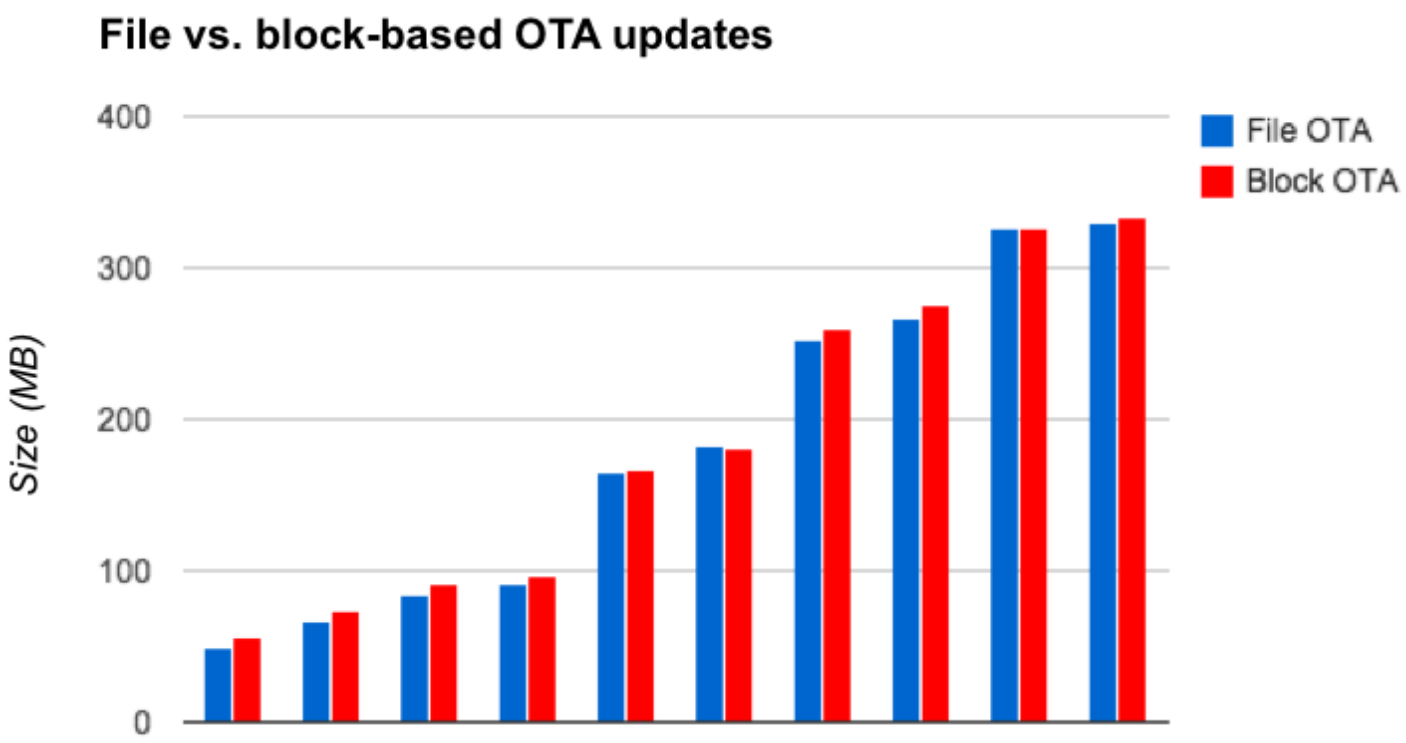

Figure 1. Compare Nexus 6 OTA sizes between Android 5.0 and Android 5.1 releases (varying target build changes)

In general, incremental block OTA updates are larger than incremental file OTA updates due to:

  - *Data preservation*. Block-based OTAs preserve more data (file metadata, dm-verity data, ext4 layout, etc.) than file-based OTA.
  - *Computation algorithm differences*. In a file OTA update, if a file path is identical in both builds, the OTA package contains no data for that file. In a block OTA update, determining little or no change in a file depends on the quality of the patch computation algorithm and layout of file data in both source and target system.
- **Sensitivity to faulty flash and RAM**. If a file is corrupted, a file OTA succeeds as long as it doesn't touch the corrupted file, but a block OTA fails if it detects any corruption on the system partition.

Updating modified systems

For devices with *modified* system partitions running Android 5.0:

- **Incremental block OTA updates fail**. A system partition might be modified during an `adb remount` or as a result of malware. File OTA tolerates some changes to the partition, such as the addition of files that are not part of the source or target build. However, block OTA does not tolerate additions to the partition, so users will need to install a full OTA overwriting any system partition modifications) or flash a new system image to enable future OTAs.
- **Attempts to change modified files cause update failure**. For both file and block OTA updates, if the OTA attempts to change a file that has been modified, the OTA update fails.
- **Attempts to access modified files generate errors** *(dm-verity only)*. For both file and block OTA updates, if dm-verity is enabled and the OTA attempts to access modified parts of the system filesystem, the OTA generates an error.

Last updated March 27, 2017.