Android 10 release notes

Archived: 2026-04-05 21:17:36 UTC

This page summarizes the major features in the Android 10 release, and provides links to additional information. These feature summaries are organized according to the feature's documentation location on this site.

Build

java_sdk_library

Android 10 introduces `java_sdk_library`, a new build rule to fix compatibility issues for shared Java libraries. Device manufacturers can use this mechanism for their own shared Java libraries to maintain backward compatibility for their APIs.

Architecture

Modular system components

Android 10 modularizes some Android system components and enables them to be updated outside of the normal Android release cycle. Some modules include:

- Android Runtime
- Conscrypt
- DNS Resolver
- DocumentsUI
- ExtServices
- Media
- ModuleMetadata
- Networking
- PermissionController
- Time Zone Data

Hardware abstraction layer (HAL)

Android 10 adds support for HALs to shut down automatically when they have no clients.

Kernel

ABI

Android 10 includes support for new ABI monitoring utilities to help with comparing, tracking, and mitigating kernel ABI changes that affect compatibility with kernel modules.

Android 10 also introduces a symbol-based ABI usages checker. The checker can detect outdated prebuilt binaries at build time, so that shared library developers can know which prebuilt binaries might be broken by their change and which prebuilt binaries must be rebuilt.

Android Live-Lock Daemon

Android 10 includes the Android Live-Lock Daemon (llkd), which is designed to catch and mitigate kernel deadlocks.

vDSO32 on ARM64

Android 10 supports using vDSO32 on 64-bit kernels, which provides a 0.4 percent increase in battery life and other performance improvements.

fstab entries for early mounted partitions

Android 10 requires devices to specify `fstab` entries for early mounted partitions using an `fstab` file in the first stage ramdisk.

HIDL

Offload BroadcastQueue

Android 10 includes a new *offload* `BroadcastQueue` to the existing *background* and *foreground* queues. The offload queue has the same priority and timeout behavior as the background queue. To prevent blocking the background queue, where more interesting or user-visible broadcasts can happen, the offload queue handles the `BOOT_COMPLETED` broadcast, which many apps listen to and can take a long time to complete. The offload queue currently only handles the `BOOT_COMPLETED` broadcast, but can potentially handle other long broadcasts.

SystemSuspend service

Android 10 replaces the thread in `libsuspend` responsible for initiating system suspend with the SystemSuspend HIDL service. This implementation offers equivalent functionality to previous versions while leveraging benefits from the Android HIDL infrastructure.

safe_union in HIDL

Android 10 introduces `safe_union` , an explicitly tagged union type, in HIDL.

Configuration

ConfigStore HAL

Android 10 deprecates the ConfigStore HAL due to high memory consumption and difficult usage, and replaces the HAL with system properties.

Config File Schema API

The Android platform contains a large number of XML files for storing config data. Many of the XML files are in the `vendor` partition, but they're read in the `system` partition. In this case, the schema of the XML file serves as the interface across the two partitions, and therefore the schema must be explicitly specified and must evolve in a backward-compatible manner. Before Android 10, the platform didn't provide mechanisms to require specifying and using the XML schema, or to prevent incompatible changes in the schema. Android 10 provides this mechanism, called the Config File Schema API.

System properties as APIs

System properties accessed across partitions are schematized into `sysprop` description files, and APIs to access properties are generated as concrete functions for C++ and classes for Java.

Vendor interface (VINTF) object

VINTF

Changes to VINTF in Android 10 include:

- Deprecating AVB version tags
- Adding kernel information in OTA packages
- Building ODM manifests officially
- Adding a product compatibility matrix
- Associating a manifest entry with a HAL module in the build system

Bootloader

Ramdisk

In Android 10, the root file system is no longer included in `ramdisk.img` and is instead merged into `system.img`.

Build ODM partitions

Android 10 includes support for building `odm` partitions using the Android build system. You can use a separate `/odm` partition for customizations, which enables you to use a single vendor image for multiple hardware SKUs. This enables original design manufacturers (ODMs) to customize system-on-chip (SoC) vendor board-support packages (BSPs) to their specific devices (their boards). They can implement kernel modules for board-specific components, board-specific daemons, or their own features on hardware abstraction layers (HALs). They may also replace or customize SoC components.

Android 10 updates the boot image header to version 2, which includes a section to store the device tree blob (DTB) image. Android 10 VTS tests verify that all devices launching with Android 10 use boot image header version 2 and include a valid DTB image as part of the boot/recovery images.

Recovery images for non-A/B devices

In Android 9 and higher, a device's recovery image must contain information from the overlay image. Device manufacturers can use DeviceTree or Advanced Configuration and Power Interface (ACPI) to describe all nondiscoverable devices. Android 10 and higher includes support for architectures that use ACPI instead of DeviceTree blob for overlay (DTBO).

Stable AIDL

Android 10 adds support for stable Android Interface Definition Language (AIDL), a new way to keep track of the application program interface (API)/application binary interface (ABI) provided by AIDL interfaces.

Move fastboot to user space

Android 10 adds support for resizable partitions by relocating the fastboot implementation from bootloader to user space.

Display

HDR video playback

Android 10 supports HDR10, VP9, and HDR10+ playback.

Text classification

Text classification uses machine learning techniques to help developers classify text. Android 10 introduces two methods to the TextClassifier API: `suggestConversationActions` and `detectLanguage`. The `suggestConversationActions` method generates suggested replies and actions from a given conversation and the `detectLanguage` method detects the language of the text.

Support for Zawgyi font rendering

Zawgyi is the most popular font in Myanmar. Android 9 and lower didn't support rendering Zawgyi because it isn't Unicode compliant. Android 10 addresses this by including a Unicode font capable of rendering both Unicode Burmese and Zawgyi together. No implementation work is needed to support Zawgyi font rendering on devices launching with Android 10. If your devices have a custom implementation to support Zawgyi, you can:

- Revert those changes and use the platform-supported method.
- Keep the common Zawgyi font in your system and use the locale code `my-qaag` in your `fonts.xml`. For more information, see the Unicode CLDR release notes on Zawgyi (Qaag).

Limitations to hiding app icons

Android 10 limits the ability for apps to hide their launcher icons. If an app doesn't have a launcher activity enabled, the system displays a *synthesized activity* in the launcher; this synthesized activity represents the app's details page within system settings.

For more information about the logic used to show app icons, including the types of apps whose app icons aren't shown, see the documentation for `getActivityList()` in the API reference.

Settings

To improve accessibility, Android 10 includes user-customizable timeout settings. The API and Settings changes come with Android 10. If you customize your Settings, make sure that this feature is supported. If you have UI elements that time out on your device, use the timeouts API on those. For more information, see the Android developer accessibility guidelines.

Compatibility

Android Compatibility Definition Document (CDD)

The Android 10 Compatibility Definition Document iterates upon previous versions with updates for new features and changes to requirements for previously released functionality.

Tests

Compatibility Test Suite (CTS)

Android CTS has a separate release notes page that lists many important changes for Android 10.

CTS downloads

CTS packages supporting Android 10 are available on the CTS Downloads page. The source code for the included tests can be synced with the `android-cts-10_r1` tag in the open-source tree.

CTS shim APEX

Android 10 introduces a package called `CtsShimApex`, which must be preinstalled on a device to write CTS tests for APEX management.

Test harness mode

CTS test harness mode helps developers automate testing for a device or a fleet of devices.

Instant Apps mode

Starting in Android 10, CTS runs in Instant Apps mode, which means installing the test APK as an Instant App and running the tests.

In addition to a CTS mode for Instant Apps, Android 10 includes CTS Verifier for Instant Apps

CTS Verifier pro audio test

Android 10 adds a CTS Verifier test for Pro Audio compliance.

CTS Verifier MIDI tests

In Android 10, the CTS Verifier MIDI test tests MIDI functionality with USB MIDI interfaces, Bluetooth MIDI interfaces, and a virtual MIDI device path.

CTS test interpretation

Android 10 updates the mechanism for interpreting CTS results.

Vendor Test Suite (VTS)

VTS testing with debug ramdisk

In Android 10, the generic system image (GSI) used to run CTS-on-GSI/VTS compliance testing changes from userdebug to user build type, because GSI is release signed. However, the `adb root` command that gives a host root permissions to the Android device under test isn't available in a user build. This is a problem because VTS requires `adb root` to run.

The debug ramdisk is introduced to make `adb root` possible, if the device is unlocked. This simplifies the testing flow by reusing the same user build `system.img` (either GSI or the OEM's `system.img`).

Hardware Composer validation

Android 10 adds a new VTS test class for Hardware Composer validation through the `readback` interface in `IComposerClient.hal`. If vendors don't implement `readback`, tests pass automatically.

Debugging

Load shared libraries with different class loaders

In Android 9 and lower, apps loaded their linked Java shared libraries in the app's class loader. In Android 10, the framework uses a different class loader than the app's class loader to load Java shared libraries linked through `uses-library` or `uses-static-library`.

In general, apps shouldn't rely on using a specific class loader, so this change shouldn't break app behavior. However, if an app relies on using a single class loader, that behavior is broken. Additionally, package-private visibility of classes in the same package is still supported, but isn't supported in shared libraries.

Device manufacturers might see app compatibility issues as they test their devices running Android 10.

Security features

For a more complete list of enhancements related to security and privacy only, see the Android 10 security and privacy enhancements page.

Face authentication

Face authentication allows users to unlock their device simply by looking at the front of their device. Android 10 adds support for a new face authentication stack that can securely process camera frames, preserving security and privacy during face authentication on supported hardware. Android 10 also provides an easy way for security-compliant implementations to enable app integration for transactions such as online banking or other services.

Extended access

Trust agents, the underlying mechanism used by tertiary authentication mechanisms such as Smart Lock, can only extend unlock in Android 10. Trust agents can no longer unlock a locked device and can only keep a device unlocked for a maximum of four hours.

Encryption

OEMCrypto

Android 10 uses OEMCrypto API version 15.

Testing

BoundsSanitizer

Android 10 deploys BoundsSanitizer (BoundSan) in Bluetooth and codecs. BoundSan uses UBSan's bounds sanitizer. This mitigation is enabled on a per-module level. It helps keep critical components of Android secure and shouldn't be disabled. BoundSan is enabled in the following codecs:

- `libFLAC`
- `libavcdec`
- `libavcenc`
- `libhevcdec`
- `libmpeg2`
- `libopus`
- `libvpx`
- `libspeexresampler`
- `libvorbisidec`
- `libaac`
- `libxaac`

Integer Overflow Sanitization

Android 10 enables Integer Overflow Sanitization (IntSan) in software codecs. Ensure that playback performance is acceptable for any codecs that aren't supported in the device's hardware. IntSan is enabled in the following codecs:

- `libFLAC`
- `libavcdec`
- `libavcenc`
- `libhevcdec`
- `libmpeg2`
- `libopus`
- `libvpx`
- `libspeexresampler`
- `libvorbisidec`

Execute-only memory

By default, executable code sections for AArch64 system binaries are marked execute-only (nonreadable) as a hardening mitigation against just-in-time code reuse attacks. Code that mixes data and code together and code that purposefully inspects these sections (without first remapping the memory segments as readable) no longer functions. Apps with a target SDK of Android 10 (API level 29 or higher) are impacted if the app attempts to read code sections of execute-only memory (XOM) enabled system libraries in memory without first marking the section as readable.

Scudo

Scudo is a dynamic user-mode memory allocator designed to be more resilient against heap-related vulnerabilities. It provides the standard C allocation and deallocation primitives, as well as the C++ primitives.

ShadowCallStack

`ShadowCallStack` (SCS) is an LLVM instrumentation mode that protects against return address overwrites (like stack buffer overflows) by saving a function's return address to a separately allocated `ShadowCallStack` instance in the function prolog of nonleaf functions and loading the return address from the `ShadowCallStack` instance in the function epilog.

Audio

Audio HAL

Android 10 includes the following new capatibilties for audio HAL.

- `AudioSource`
- `AudioFormat`
- `AudioChannelMask`

Additional requirements are added for audio HAL and subsystem implementation.

Preprocessing effects

Android provides preprocessing effects, such as acoustic echo cancellation, automatic gain control, and noise suppression. Android 10 includes new requirements for capturing with `VOICE_COMMUNICATION`.

Audio policy manager

Android 10 includes a significant refactoring of the audio policy manager to provide more flexibility to support complex automotive use cases.

High-resolution audio

Android 10 includes the following improvements for high-resolution audio.

- Float support
- 192 KHz frequency support
- Eight-channel support
- Inclusion of timing information

Concurrent capture

Android 10 improves the concurrent capture user experience that requires more than one active audio capture to happen simultaneously.

AudioPlaybackCapture

Android 10 contains a new API called `AudioPlaybackCapture`, which gives apps the ability to copy the audio being played by other apps. This feature is similar to screen capture, but for audio. The primary use case is to enable streaming apps to capture the audio being played by games.

The capture API doesn't affect the latency of the app whose audio is being captured.

MIDI

Android 10 makes it easier to port professional audio apps using MIDI to the Android platform using the `AMidi` NDK API.

Camera

For a summary of the changes to the camera API, camera HAL, and camera module introduced in Android 10, see Android 10 camera updates.

Camera framework privacy improvements

Android 10 introduces privacy enhancements to the camera framework. To avoid exposing potentially sensitive static camera information in `CameraCharacteristics` without user consent, apps must obtain the `CAMERA` permission to retrieve static metadata with a privacy-sensitive tag using the `getCameraCharacteristics` method.

To get a list of the camera characteristic keys that require the `CAMERA` permission, call the `getKeysNeedingPermission` method.

Session reconfiguration query

Android 10 adds a session reconfiguration query feature, which allows for improved performance through more control over the internal session parameter reconfiguration logic.

Camera HAL3 buffer management APIs

Android 10 introduces optional camera HAL3 buffer management APIs that allow you to implement buffer management logic to achieve different memory and capture latency tradeoffs in camera HAL implementations.

Camera HAL dynamic physical camera switch

Android 10 introduces a dynamic metadata tag, `ANDROID_LOGICAL_MULTI_CAMERA_ACTIVE_PHYSICAL_ID`, which indicates the active underlying physical camera of a logical camera device. For more information, see Multi-Camera Support.

Support for hiding physical cameras

In Android 10, the camera HAL can reduce the number of physical cameras that can be directly opened by an app. For more details, see Multi-Camera Support.

Camera2 VNDK API

In Android 10, vendor modules can access and control camera devices through two new standard HIDL interfaces, `android.frameworks.cameraservice.service@2.0` and `android.frameworks.cameraservice.device@2.0`. To make using the HIDL interfaces more convenient, Android 10 also introduces a vendor-available library, `libcamera2_vendor`. This library is similar to the Camera NDK library, with a few minor modifications.

Stream configurations

Android 10 adds features that allow camera vendors to advertise recommended camera streams to camera clients and to support an API to query stream combinations.

Camera stream combination requirements

Devices running Android 10 are no longer required to support stream combinations with physical subcamera streams. However, devices running Android 10 with the camera HAL device version 3.5 must support `isStreamCombinationSupported()` to allow apps to query whether a stream combination containing physical streams is supported.

For more information, see Multi-Camera Support.

HEIF imaging

Android 10 provides native camera support for high efficiency image file format (HEIF) images, which offer improved image quality and smaller sizes over JPEG images. Devices must have an HEIC or HEVC encoder to support HEIF images.

Monochrome cameras

Android 10 provides additional support for the Y8 stream format, monochrome and near-infrared (NIR) color filter array static metadata, and `DngCreator` functions for monochrome cameras.

Connectivity

Calling and messaging

Emergency numbers and emergency calling

Android 10 provides improved support for emergency calling. In an emergency, devices with support for IRadio HAL v1.4 can initiate an emergency call using emergency numbers retrieved from a source such as a SIM card, the network signal, or the Android database. Numbers can be categorized based on emergency service categories such as police, fire, and ambulance.

Group call APIs

The group call APIs are an extension of the eMBMS APIs added in Android 9. The new APIs define a standard for apps to join and broadcast on cell-broadcast group calls by interacting with eMBMS middleware packages. Group calls require support from the chipset vendor, middleware vendor, and the cell carrier to function properly. Developer documentation is located at developer.google.com.

Remote SIM capabilities

Android 10 introduces remote SIM capabilities that allow messaging apps on an Android host device to send SMS messages through phones using mechanisms such as Bluetooth. For more information, see the reference documentation for the `getSubscriptionType` method and the `SUBSCRIPTION_TYPE_REMOTE_SIM` constant.

Multiple eSIMs

In Android 10, the `EuiccManager` class supports devices with multiple embedded SIMs (eSIMs), or eUICCs.

eSIM updates

For devices running Android 10 that support eSIMs, a nonremovable eUICC slot ID array must be defined. Devices must also support IRadio HAL v1.4 and IRadioConfig HAL v1.2. For more information, see Implementing eSIM and HAL Requirements.

5G Non-Standalone (NSA)

Android 10 adds support for 5G non-standalone (NSA). 5G NSA is a solution for 5G networks where the network is supported by an existing 4G infrastructure. On Android 10, devices can display a 5G icon on the status bar when a device connects to a 5G network.

Phone account suggestion

Android 10 introduces the phone account suggestion service, which allows suggestions for phone accounts to be shown to users when making a call.

Carrier

Migrate Mobile Network settings

Android 10 rearchitected the Mobile Network settings UI code and moved it from the Telephony stack to the Settings stack. To support the migrated code, change the following Mobile Network settings configuration values from Android resources to `CarrierConfig` resources:

`config_world_mode` -> `CarrierConfigManager#KEY_WORLD_MODE_ENABLED_BOOL`

`config_support_tdscdma` -> `CarrierConfigManager#KEY_SUPPORT_TDSCDMA_BOOL`

`config_support_tdscdma_roaming_on_networks` -> `CarrierConfigManager#KEY_SUPPORT_TDSCDMA_ROAMING_NETWORKS_STRING_ARRAY`

`config_enabled_lte` -> `CarrierConfigManager#KEY_LTE_ENABLED_BOOL`

Device identifiers

Persistent device identifiers (IMEI/MEID, IMSI, and build serial) are guarded by a privileged permission with access also granted to device and profile owner apps. Because the IMSI and SIM serial number are carrier provided, access to these identifiers is granted to packages with carrier privileges.

Wi-Fi

Network selection

Android continuously evaluates the quality of the connected network and assesses the quality of available networks. Android 10 has updated algorithms and procedures for selecting and switching between Wi-Fi networks.

Wi-Fi preferred network offload scanning

Android 10 introduces an optional API method named `setDeviceMobilityState()` in `WifiManager` that increases the interval between preferred network offload (PNO) scans when the device is stationary to reduce power usage.

Carrier Wi-Fi

In Android 10, devices with the carrier Wi-Fi feature automatically connect to configured carrier Wi-Fi networks (networks with public key certificates).

Wi-Fi Easy Connect

In Android 10, devices can use Wi-Fi Easy Connect, which uses the device provisioning protocol (DPP) introduced by the Wi-Fi Alliance (WFA), to provision and configure Wi-Fi devices.

Wi-Fi low-latency mode

Android 10 introduces a Wi-Fi low-latency mode, which configures the Wi-Fi chip to reduce latency.

Updated DHCP server

As part of the formation of an "IP Server" service umbrella, `dnsmasq` is being deleted. Android 10 replaces its DHCPv4 server functional use with a separate component, primarily written in Java to better integrate with the Java framework control plane. This improves security and updatability for the DHCP server. For more details, see `packages/modules/NetworkStack/src/android/net/dhcp/DhcpServer.java`.

No action is required to implement this change: all devices releasing and upgrading to Android 10 use `DhcpServer` by default. If you have customizations to the DHCP server, you can revert to Android 9 behavior by setting the global setting `tether_enable_legacy_dhcp_server=1`. The new `DhcpServer` is included in the networking components module, so any customization to DHCP server functionality should be upstreamed.

WPA3 and Wi-Fi Enhanced Open

Android 10 adds support for the Wi-Fi Protected Access 3 (WPA3) and Wi-Fi Enhanced Open security standards to provide better privacy and robustness against known attacks.

Wi-Fi Direct

Wi-Fi Direct, also known as Wi-Fi P2P, allows supporting devices to discover and connect to one another directly using the Wi-Fi Direct protocol without internet or cellular network access.

MAC randomization enhancements

From Android 10, MAC randomization is enabled by default for client mode, SoftAp, and Wi-Fi Direct. Devices must provide an option to enable or disable MAC randomization for each SSID in the system UI.

Passpoint R2

Android 10 introduces support for Passpoint R2 features. Passpoint R2 implements online sign up (OSU), a standard method to provision new Passpoint profiles. Android 10 supports the provisioning of EAP-TTLS profiles

using SOAP-XML.

NFC

Secure NFC

Secure NFC allows off-host NFC card emulation to be enabled only when the device's screen is unlocked. Implementing this feature gives users the option to enable Secure NFC for improved security.

Android Beam deprecated

In Android 10, Android Beam is no longer required and the following interfaces and methods have been deprecated.

Interfaces:

- `NfcAdapter.CreateBeamUrisCallback`
- `NfcAdapter.CreateNdefMessageCallback`
- `NfcAdapter.OnNdefPushCompleteCallback`

Methods:

- `createBeamUris`
- `invokeBeam`
- `isNdefPushEnabled`
- `setBeamPushUris`
- `setBeamPushUrisCallback`
- `setNdefPushMessage`
- `setNdefPushMessageCallback`
- `setOnNdefPushCompleteCallback`
- `createNdefMessageCallback`
- `onNdefPushCompleteCallback`

To use Android Beam, report the `android.sofware.nfc.beam` feature constant.

Graphics

ASurfaceControl

Android 10 adds `ASurfaceControl`, a new way for `SurfaceFlinger` to accept buffers.

Graphics implementation

OpenGL ES layers

Android 10 introduces a layering system for GLES.

EGL 1.5

Android 10 implements the EGL 1.5 interface. For information on new features in EGL 1.5, view the Khronos Releases EGL 1.5 Specification.

Vulkan

Android 10 includes support for Vulkan 1.1 graphics. The platform also supports `VK_KHR_swapchain` v70, so the Vulkan app is able to create a `VkImage` backed by swapchain memory.

Performance refresh rate

Android 10 adds support for a performance refresh rate. This feature is turned off by default.

Interaction

Automotive

Automotive audio

In Android 10, Audio HAL context maps to `AudioAttributes.usage` to identify sounds. Android supports one `AUDIO_DEVICE_OUT_BUS` instance per context. `IAudioControl` HAL provides vehicle-specific extensions to the Audio HAL.

Gestural navigation

Android 10 introduces an option for a fully gestural system navigation. For information about how to prepare apps to use this feature, see the Gestural navigation page on the Android Developer site.

Neural networks

Android 10 introduces updates to the Neural Networks API and the Neural Networks HAL. For a summary of the changes, see Neural networks.

New and updated Neural Networks documentation for Android 10:

- Overview
- AHardwareBuffer
- Burst Executions and Fast Message Queues
- Compilation Caching
- Device Discovery and Assignment
- Vendor Extensions

Sensors

Sensors HAL 2.0

Sensors HAL 2.0 supports using fast message queues (FMQs) to send sensor events from the HAL into the Android Sensors Framework.

Sensors off

Android 10 includes a developer setting to shut off all sensors on a device. This feature helps developers test their app's functionality in situations where those sensors become unavailable, and also gives users a way to control the sensors in their device.

If your devices use the default implementation of `SensorService`, `CameraService`, and `AudioPolicyService`, then no additional customization is needed to the reference design. If you have other sensors, see Customization for more details about supporting this feature.

Updatable media components

Android 10 provides updatable media components that enable updating media-related modular system components through the Google Play Store infrastructure or through a partner-provided over-the-air (OTA) mechanism.

Media DRM

Android 10 improves the utility and usability of the `MediaDrm` Java and NDK APIs.

Decoding

Android 10 supports AV1 SW decoding.

Permissions

Android 10 provides additional permission configurations for transparency and user privacy.

Contacts provider and affinities information

Starting in Android 10, contacts-affinity-related data, managed by the Contacts Provider component, is accessed differently than in Android 9 and lower. These changes regarding data accessibility improve user privacy in all Android 10 devices that use the Contacts Provider component. The underlying database no longer contains contact affinities data. Therefore apps can't write to or read from it.

The changes in Android 10 are expected to have a large impact on APIs. If your apps rely on the deprecated features mentioned in Contacts Provider and Affinities Information, you may want to update your apps to compensate for any changes. Additionally, if you use a forked version of the Contacts Provider, you must update your Contacts Provider.

Tristate location permissions

Tristate location permissions in Android 10 give users more control over how apps access their device locations.

Background location access reminder

Android 10 features a background access location reminder, which increases transparency into how much access apps have to a device's location and helps users maintain control over such access.

Restrict opportunistic locations

When an app requests a device's location, the app can either wait for the request response or, by using active location listeners, get an opportunistic location update. Starting in Android 10, to get opportunistic location updates, developers must specify that they need passive location updates from the `FusedLocationProviderClient` class.

Background apps launching

In Android 10, nonprivileged apps without a visible window can't launch themselves automatically to the foreground. This change suppresses ad popups and malicious takeovers. No action is required to enable this.

App sandboxing

In Android 10, apps have a limited raw view of the file system, with no direct access to paths like `/sdcard/DCIM`. However, apps retain full raw access to their package-specific paths, as returned by any applicable methods such as `Context.getExternalFilesDir()`. Apps still have full raw access to their package-specific paths.

Use the app sandbox guidelines for sharing files to provide appropriate data-sharing granularity.

Restrict app clipboard access

In Android 10, clipboard access has changed so that clipboard content can't be watched by calling `ClipboardManager.getPrimaryClip` or by adding an `onPrimaryClipChangedListener` listener for notification when the clipboard changes. This increases user privacy and disables malvertizing apps from modifying the clipboard.

In Android 10, read access is only allowed to either the current app with input focus, or to the current keyboard. The `ClipboardManager.onPrimaryClipChanged()` listener call now only fires for apps that meet such restrictions. `ClipboardManager.getPrimaryClip` and `ClipboardManager.getPrimaryClipDescription` return `null` if the requesting app either isn't the default input method editor (IME), or doesn't have input focus.

Runtime permissions include activity recognition

Users now see an activity recognition dialog when an app accesses device location in the background. Hard-restricted runtime permissions must be properly whitelisted in Android 10.

MANAGE_DEVICE_ADMINS permission

Android 10 changes the `MANAGE_DEVICE_ADMINS` permission from signature or privileged to signature only. This means that only platform-signed apps can set other apps as a device admin.

Sharing API improvements

Android 10 provides a number of new Android Platform API features related to sharing. If you've modified the Share Sheet code in your implementation, ensure that your implementation supports these new features. If you haven't modified the Share Sheet code in your implementation, you don’t need to do anything to support these new features.

Android Runtime (ART)

Signed Config

The Signed Config feature allows embedding configuration of non-SDK interface restrictions in APKs. This enables removing specific non-SDK interfaces from the blacklist, so that AndroidX can safely use them. With this change, AndroidX can add support for new features on older Android versions.

Performance

Cgroup abstraction layer

Android 10 includes a cgroup abstraction layer and task profiles, which developers can use to describe a set of restrictions to apply to a thread or a process.

Low Memory Killer Daemon (lmkd)

Android 10 supports a new `lmkd` mode that uses kernel pressure stall information (PSI) monitors for memory pressure detection.

Power

Platform power management

In Android 10, Doze mode can be enabled on always-on devices as well as on battery-powered devices.

Routine battery saver

Android 10 introduces a new battery saver schedule option called **based on routine**. Routine battery saver allows an app chosen by the OEM to provide signals to the system for more intelligent battery saver scheduling. This option requires configuration, and is optional to implement.

Power stats HAL

In Android 10, `IPowerStats.hal` replaces the power stats collection APIs in `IPower.hal`. While the power HAL still supports the APIs, they'll be migrated exclusively to the power stats HAL in the future.

The power stats HAL includes new APIs to support the collection of data from on-device power measurement, for supported devices. The existing power stats collection APIs are also updated to improve flexibility. The power

hinting APIs remain in power HAL and aren't changing.

Thermal mitigation

The thermal framework in Android 10 abstracts device interfaces for the thermal subsystem temperature sensor, including CPU, GPU, battery, skin, and cooling device. The framework introduces a polling interface to query thermal status to initiate throttling, and a callback interface to send a message to the user when a threshold is exceeded.

Android 10 provides the new data types through the `IThermalService` interface using these three new methods:

- `getCurrentThermalStatus()` returns the current thermal status of the device as an integer, unless the device is undergoing throttling.
- `addThermalStatusListener()` adds a listener.
- `removeThermalStatusListener()` removes a previously added listener.

Apps add and remove listeners and access temperature status in the `PowerManager` class. Only a trusted system service such as an Android API or device manufacturer API can access information about associated causal events. Device manufacturers or SoC makers must implement `thermal HAL 2.0` to enable the full functionality of the new thermal framework.

For a thermal mitigation implementation example, see the Reference implementation.

Updates

APEX file format

Android Pony EXpress (APEX) is a new container format used in the install flow for modular system components.

Dynamic partitions

Dynamic partitions introduce a userspace partitioning system to Android, allowing partitions to be created, resized, or destroyed during OTA updates. Device makers don't have to worry about the individual sizes of partitions, such as `system`, `vendor`, and `product`. Instead, one big `super` partition is allocated, and subpartitions can be sized dynamically within it.

Dynamic system updates

Dynamic system updates (DSU) allows you to make an Android system image that users can download from the internet and try out without the risk of corrupting the current system image.

Multiuser backup and restore

Android 10 supports backup and restore functionality for all users on a device. Previously, backup and restore was only available for the system user. Backup and restore for nonsystem users is turned off by default as it has only

partial coverage for settings, wallpaper, and system components.

Overlayfs

Users working with `userdebug` or `eng` builds expect to be able to efficiently remount the system partition as read-write and then add or modify any number of files without reflashing the system image. You can use `Overlayfs`, which automatically sets up backing storage for a writable file system as an upper reference, and mounts over the lower. These actions happen in the `adb disable-verity` and `adb remount` requests. For more details, see the Overlayfs README in AOSP.

Shared library support in recovery mode

In Android 10, shared libraries are available in the recovery partition, which eliminates the need for all recovery mode executables to be static. The shared libraries are located under the `/system/lib` (or `/system/lib64` for 64-bit devices) directory in the partition.

To add a new shared library to the recovery partition, add `recovery_available: true` or `recovery: true` to `Android.bp` of the shared library. The former installs the library to both the system and recovery partitions, while the latter installs it only to the recovery partition.

Shared library support can't be built with Android's make-based build system. To convert an existing static executable for the recovery mode to a dynamic one, remove `LOCAL_FORCE_STATIC_EXECUTABLE := true` in `Android.mk` or `static_executable: true` (in `Android.bp`).

User Data Checkpoint (UDC)

Android 10 introduces the User Data Checkpoint (UDC) feature, which allows Android to roll back to its previous state when an Android over-the-air (OTA) update fails.
