**Wi-Fi Aware**

The [Wi-Fi Aware](https://developer.android.com/guide/topics/connectivity/wifi-aware.html) feature added in Android 8.0 enables supporting devices to discover, connect, and range (added in Android 9) to one another directly using the Wi-Fi Aware protocol without internet or cellular network access. This feature, built upon the [Wi-Fi Alliance](https://www.wi-fi.org/) (WFA) [Wi-Fi Aware specification](https://www.wi-fi.org/discover-wi-fi/wi-fi-aware) (versions 2.0 and 3.0), allows easy sharing of high-throughput data among trusted devices and apps that are otherwise off-network.

**Examples and source**

To use this feature, device manufacturers should implement the Wi-Fi [Hardware Interface Design Language (HIDL)](https://source.android.com/devices/architecture/hidl) provided in the Android Open Source Project (AOSP). HIDL replaces the previous [Hardware Abstraction Layer (HAL)](https://source.android.com/devices/architecture/hal) structure used to streamline implementations by specifying types and method calls collected into interfaces and packages.

Follow the Wi-Fi HIDL to employ the Wi-Fi Aware feature: hardware/interfaces/wifi/1.2 or higher. The Wi-Fi Aware HAL surface is very large; the [hardware/interfaces/wifi/1.2/README-NAN.md](https://android.googlesource.com/platform/hardware/interfaces/+/master/wifi/1.2/README-NAN.md) file describes the subset that is currently in use by the framework.

You can reference the legacy Wi-Fi HAL to see how it correlates with the new HIDL interface: [hardware/libhardware\_legacy/+/master/include/hardware\_legacy/wifi\_nan.h](https://android.googlesource.com/platform/hardware/libhardware_legacy/+/master/include/hardware_legacy/wifi_nan.h).

**Implementation**

Device manufacturers need to provide both framework and HAL/firmware support:

* Framework:
  + AOSP code
  + Enable Aware: Requires both a feature flag and an HIDL build flag
* Wi-Fi Aware (NAN) HAL support (which implies firmware support)

To implement this feature, device manufacturers implement the Wi-Fi HIDL and enable two feature flags:

* In BoardConfig.mk or BoardConfig-common.mk located in device/<oem>/<device>, add the following flag:

WIFI\_HIDL\_FEATURE\_AWARE := true

* In device.mk located in device/<oem>/<device>, modify the PRODUCT\_COPY\_FILES environment variable to include support for the Wi-Fi Aware feature:

PRODUCT\_COPY\_FILES +=  
frameworks/native/data/etc/android.hardware.wifi.aware.xml:$(TARGET\_COPY\_OUT\_VENDOR)/etc/permissions/android.hardware.wifi.aware.xml

Wi-Fi Aware includes ranging to peer devices using the IEEE 802.11mc protocol, also known as Round Trip Time (RTT). This sub-feature of Wi-Fi Aware is conditional on the device supporting the Wi-Fi RTT feature, that is, it requires the device to support both Wi-Fi Aware and Wi-Fi RTT. For more details, see [Wi-Fi RTT](https://source.android.com/devices/tech/connect/wifi-rtt).

Otherwise, everything required for this feature is included in AOSP.

The WIFI\_HIDL\_FEATURE\_AWARE flag is ignored if the WIFI\_HAL\_INTERFACE\_COMBINATIONS flag is specified. For more information, see [Wi-Fi multi-interface concurrency](https://source.android.com/devices/tech/connect/wifi-hal#wi-fi_multi-interface_concurrency).

**MAC randomization**

Android requires the MAC address of the Wi-Fi Aware discovery (NMI) and data interfaces (NDPs) to be randomized and not be identical to the true MAC address of the device. The MAC addresses must be:

* Randomized whenever Wi-Fi Aware is enabled or re-enabled.
* When Wi-Fi Aware is enabled, the MAC address must be randomized at a regular interval configured by the NanConfigRequest.macAddressRandomizationIntervalSec HIDL parameter. This is configured by the framework by default to be 30 minutes.

**Note:** Per the Wi-Fi Aware spec, randomization may be suspended while an NDP is configured. Suspension does not imply that the factory MAC address is used (it must never be used for Wi-Fi Aware), but that the MAC address is re-randomized less frequently than otherwise required.

**Validation**

Android provides a set of unit tests, integration tests (ACTS), [Compatibility Test Suite (CTS)](https://source.android.com/compatibility/cts) tests, and [CTS Verifier](https://source.android.com/compatibility/cts/verifier) tests to validate the Wi-Fi Aware feature. Wi-Fi Aware can also be tested using the [Vendor Test Suite (VTS)](https://source.android.com/devices/tech/test_infra/tradefed/fundamentals/vts).

**Unit tests**

The Wi-Fi Aware package tests are executed using:

Service tests:

% ./frameworks/opt/net/wifi/tests/wifitests/runtests.sh -e package  
com.android.server.wifi.aware

Manager tests:

% ./frameworks/base/wifi/tests/runtests.sh -e package android.net.wifi.aware

**Integration tests (ACTS)**

The acts/sl4a test suite, described in tools/test/connectivity/acts/tests/google/wifi/aware/README.md, provides functional, performance, and stress tests.

**Compatibility Test Suite (CTS) tests**

Use CTS tests to validate the Wi-Fi Aware feature. CTS detects when the feature is enabled and automatically includes the associated tests.

The CTS tests can be triggered using:

% atest SingleDeviceTest

**CTS Verifier tests**

CTS Verifier tests validate Wi-Fi Aware behavior using two devices: a test device and a *known good* device. To run the tests, open CTS Verifier and navigate to the section titled Wi-Fi Aware Tests.