**Wi-Fi RTT (IEEE 802.11mc)**

The [Wi-Fi Round Trip Time (RTT)](https://developer.android.com/guide/topics/connectivity/wifi-rtt) feature in Android 9 enables supporting devices to measure a distance to other supporting devices: whether they are Access Points (APs) or Wi-Fi Aware peers (if [Wi-Fi Aware](https://source.android.com/devices/tech/connect/wifi-aware) is supported on the device). This feature, built upon the IEEE 802.11mc protocol, enables apps to use enhanced location accuracy and awareness.

**Examples and source**

To use this feature, implement the Wi-Fi Hardware Interface Design Language (HIDL) provided in the Android Open Source Project (AOSP). In Android 8.0, HIDL replaces the previous Hardware Abstraction Layer (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 RTT feature: hardware/interfaces/wifi/1.0 or later.

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

**Implementation**

To implement Wi-Fi RTT, you must provide both framework and HAL/firmware support:

* Framework:
  + AOSP code
  + Enable Wi-Fi RTT: requires a feature flag
* Wi-Fi RTT (IEEE 802.11mc) HAL support (which implies firmware support)

To implement this feature, implement the Wi-Fi HIDL and enable the feature flag:

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

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

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

**MAC randomization**

To enhance privacy, the MAC address used during Wi-Fi RTT transactions must be randomized, i.e., it must not match the native MAC address of the Wi-Fi interface. However, as an exception, when a device is associated with an AP, it may use the MAC address with which it is associated for any RTT transactions with that AP or with other APs.

**Validation**

Android Compatibility Test Suite (CTS) tests exist for this feature. CTS detects when the feature is enabled and automatically includes the associated tests. This feature can also be tested using the [Vendor Test Suite (VTS)](https://source.android.com/compatibility/vts) and [acts/sl4a](https://android.googlesource.com/platform/tools/test/connectivity/+/master/acts/tests/google/wifi/), a test suite that conducts extensive integration testing.

**Unit tests**

The Wi-Fi RTT package tests are executed using:

Service tests:

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

Manager tests:

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

**Integration (ACTS) tests**

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

**CTS**

Android Compatibility Test Suite (CTS) tests exist for this feature. CTS detects when the feature is enabled and automatically includes the associated tests. An Access Point which supports Wi-Fi RTT (IEEE 802.11mc) must be within range of the device-under-test.

The CTS tests can be triggered using:

% atest WifiRttTest

**Calibration**

For Wi-Fi RTT to perform well, the ranges returned in the 802.11mc protocol are ideally accurate within the Key Performance Indicator (KPI). For the 90% CDF error, at the bandwidths listed, the recommended KPI for a range estimate is expected to have the following tolerances:

* 80MHz: 2 meter
* 40MHz: 4 meters
* 20MHz: 8 meters

To ensure the implementation of the feature is working correctly, calibration testing is necessary.

This can be achieved by comparing a ground truth range against the RTT estimated range at increasing distances. For basic conformance, you should validate your solution against a device known to be RTT calibrated. Range calibration should be tested under the following conditions:

1. A large open laboratory, or a corridor that does not have a lot of metal objects that may result in unusually high occurrences of multi-path.
2. At least a Line-Of-Sight (LOS) track/path extending for 25m.
3. Markers of 0.5 meter increments from one end of the track to the other.
4. A place to secure an RTT capable access point at one end of the track mounted 20cm above the floor, and a movable mount for an Android phone (or other Android mobile device under test) that can be moved along the track, and aligned with the 0.5m markers, also at 20cm above the floor. Note: This repetitive task can be performed by a small robot, but a human operator is also fine.
5. 50 ranging results should be recorded at each marker, along with the distance from the access point. Statistics, such as range mean and variance, should be calculated for each marker position.

From the results in step 5, a chart can be drawn for ground truth (x-axis) against estimated range (y-axis) and a best fit regression line estimated. Ideal device calibration will result in a line of gradient 1.0, with offset 0.0m on the y-axis. Deviations from these values are acceptable if they are within the KPI for the corresponding bandwidth. If the results are outside of the KPI, the device feature should be recalibrated to bring the results within the KPI specification.