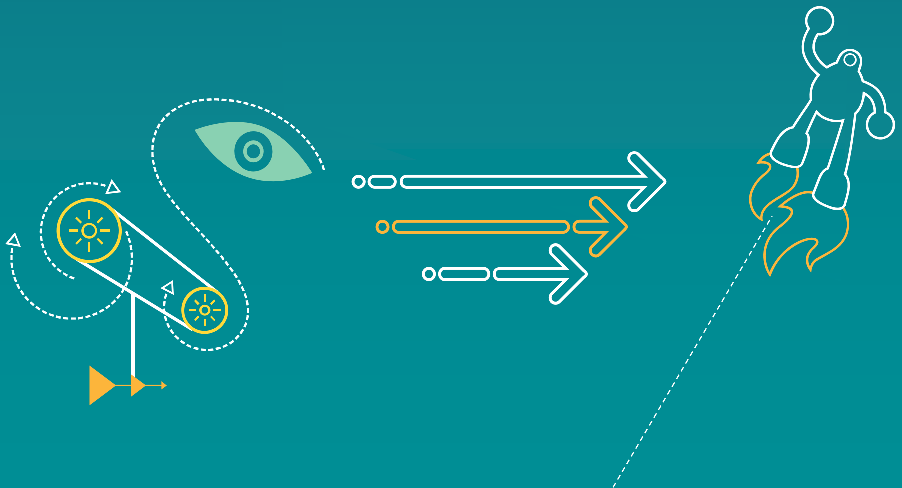
G-Scan Specification

Qualcomm Technologies, Inc.

80-Y8698-3 Rev. C



Revision History

Revision	Date	Description
A	August 2014	Initial release
B	January 2016	Added Configuration parameters. Updated Limitations. Added Android M enhancements: Exponential Backoff timer for G-Scan Introduce report_threshold_percent / report_threshold_percent_num_scans
C	August 2016	Added Android N enhancements: Changes in structure/function declaration Added WCN39xx in Introduction.

Agenda

Feature Introduction

Introduction (1 of 3)

- G-Scan is offloaded Wi-Fi scan used to collect data about device’s environment for positioning.
- Results can be used to opportunistically drive the existing firmware roam state machine and PNO.
- G-Scan addresses the following design flaws in Batch scan:
  - Consumes more power as it scans all the channels at the specific period.
  - It is not required to scan all the channels.
    - For example, the DFS channels might not be required to scan aggressively.
  - Prioritizes the channels to scan with aggressive periodicity by introducing the concepts of buckets.
  - Results can be leveraged and used as an input to roaming.

Introduction (2 of 3)

- As part of this feature, the following sub features are available:
  - G-Scans are used to collect data about device’s environment such as.
    - Per channel scanning frequency.
    - Scan history and hot lists.
  - Per channel scanning frequency:
    - Wi-Fi channels can be classified in to several buckets based on firmware capability and regulatory domain.
      - Example:

        bucket 1: 2.4 GHz 1,6,11

        bucket 2: If associated the current infrastructure channel

        bucket 3: 5 GHz 149 (a.k.a. NAN social channel)

        bucket 4: 2.4 GHz 2, 3, 4, 5, 7, 8, 9, 10, 12?, 13?

        bucket 5: Non-DFS 5 GHz channels (in US, lower and upper UNI)

        bucket 6: DFS 5 GHz channels

Introduction (3 of 3)

- This feature is implemented with a single firmware timer, ensure that scan frequencies for each bucket are multiple of a base frequency.

  For instance:

  bucket 1: Every 10 seconds (= base frequency)

  bucket 2: Every 30 seconds (3 times base frequency etc…)

  bucket 3: Every 30 seconds

  bucket 4: Every 60 seconds

  bucket 5: Every 60 seconds

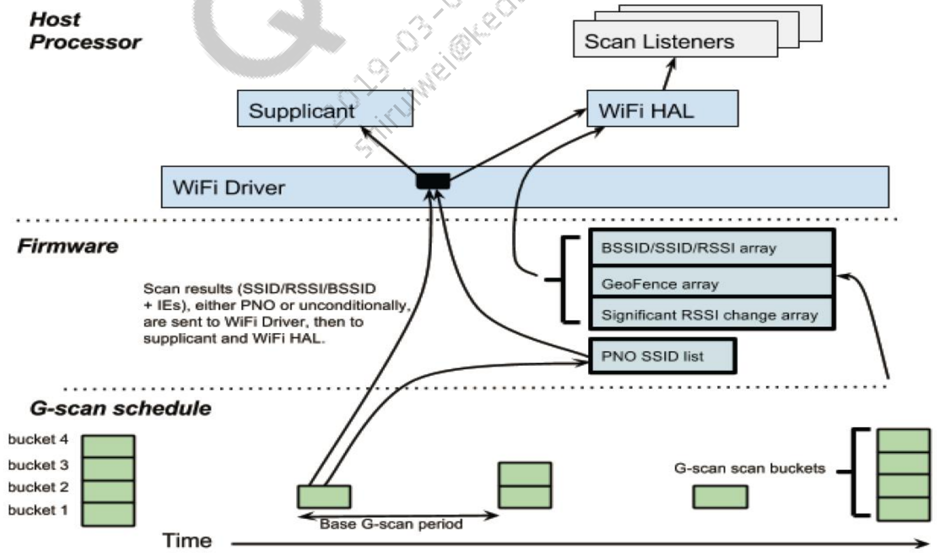
  bucket 6: Never

- Scan History and Hot Lists:
  - BSSID/SSID/RSSI history consists of:
    - List of AP's that are in the specified range.
    - List of AP's that are in close range in a sparse environment.
    - Few far away access points that may be present in a dense environment.
  - Significant RSSI changes:
    - BSSI list consists the list of configured BSSID's that have crossed the configured RSSI threshold, and the BSSID list that are not observed for a given amount of time.

Scan Coalescing

Scan Coalescing

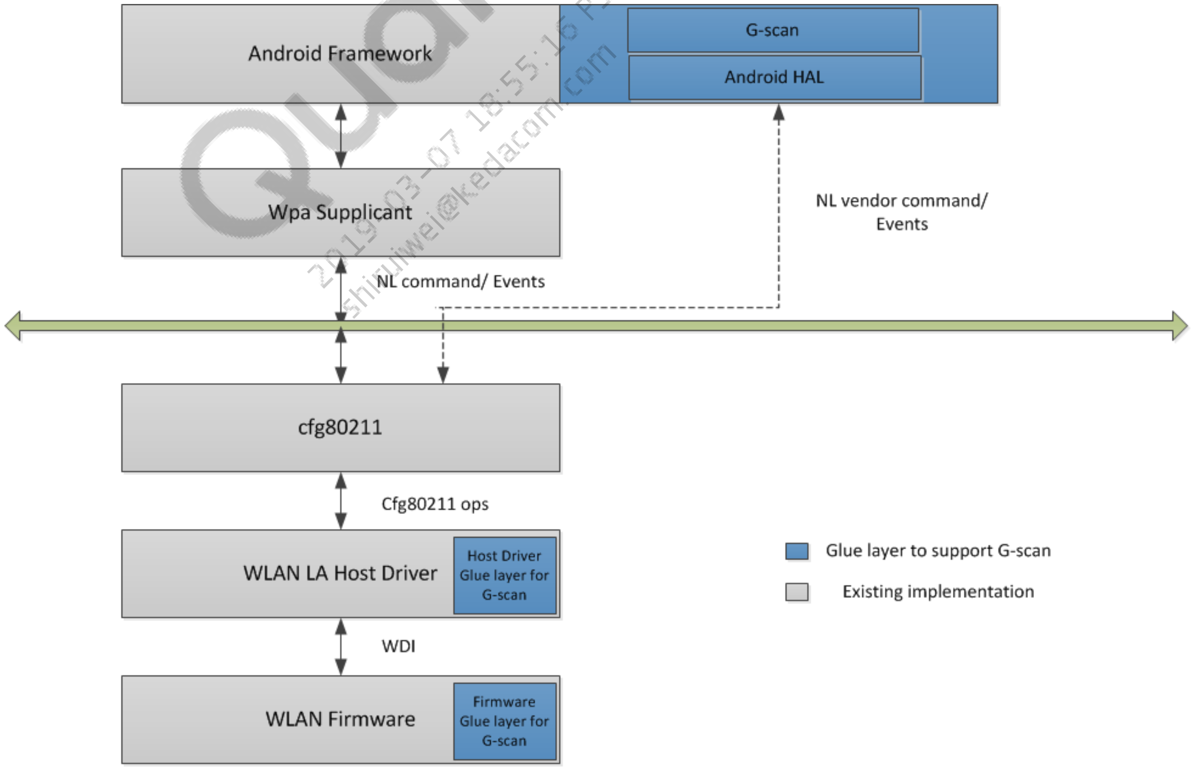
- The G-Scan scan results may be used for the following:
  - Populating the BSSID/SSID/RSSI array and matching against the GeoFence and significant RSSI change array.
  - Matching against the PNO list.
  - Forward to host processor, after each scan bucket is scanned.
- G-Scans are represented with four scan buckets in the following figure:



Architecture and Implementation

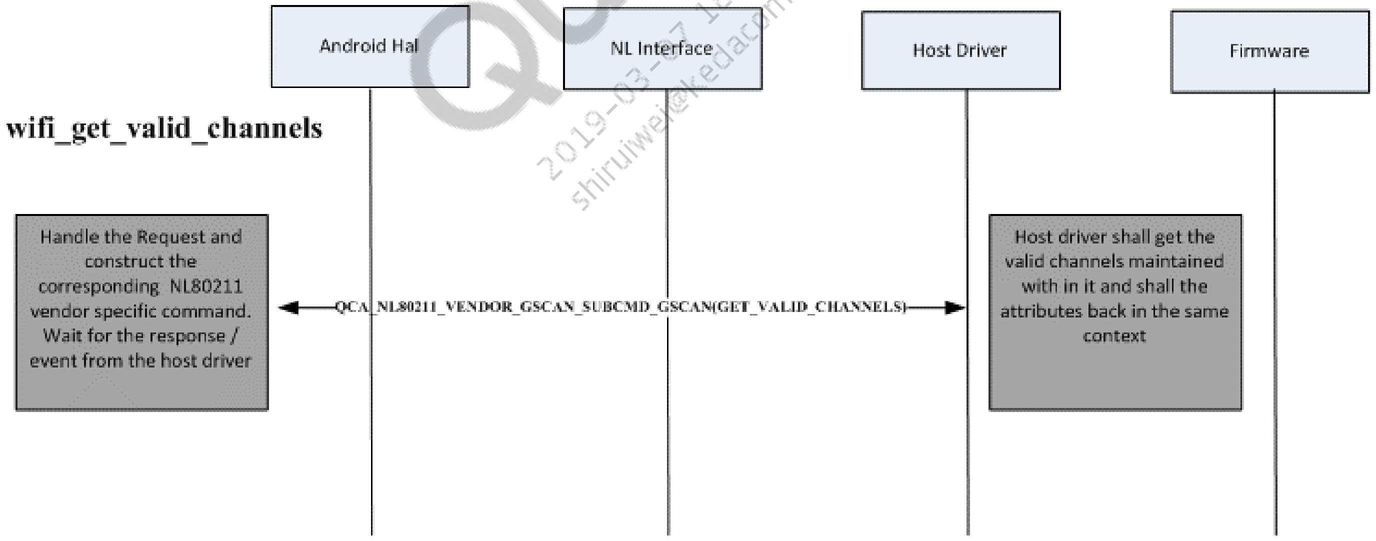
G-Scan Architecture and Implementation

- G-Scan interact with the Android HAL layer which is a new enhancement to the Android framework to interact with the host driver.
- Android HAL interacts with the NL 80211 layer through the NL vendor commands/events.



G-Scan Host API's (1 of 2)

- Get Valid Channels:
  - wlan_hdd_cfg80211_extscan_get_valid_channels()
  - Get the list of valid Wi-Fi channels for specified frequency band.
  - Input: Wi-Fi band.
  - Output: Number of channels and channel list.


Android Hal
NL Interface
Host Driver
Firmware
wifi_get_valid_channels
Handle the Request and construct the corresponding NL80211 vendor specific command. Wait for the response / event from the host driver
QCA_NL80211_VENDOR_GSCAN_SUBCMD_GSCAN(GET_VALID_CHANNELS)
Host driver shall get the valid channels maintained with in it and shall the attributes back in the same context

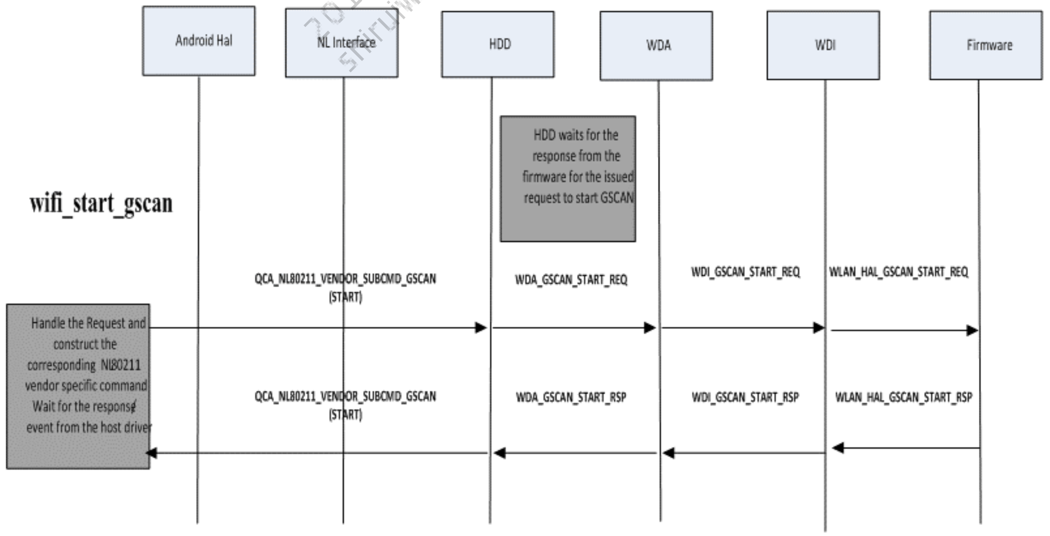
G-Scan Host API's (2 of 2)

- Get Capabilities:
  - wlan_hdd_cfg80211_extscan_get_capabilities()
  - Gets the G-Scan capabilities from the driver.
  - Configures the G-Scan parameters before issuing the start G-Scan command.
  - Input: G-Scan capabilities structure.
  - Output: The capabilities are obtained through the response to the issued command in the same context.
  - G-Scan capabilities structure:

```
typedef struct {
    int max_scan_cache_size;            // total space allocated for scan (in bytes)
    int max_scan_buckets;               // maximum number of channel buckets
    int max_ap_cache_per_scan;          // maximum number of APs that can be stored per scan
    int max_rssi_sample_size;           // number of RSSI samples used for averaging RSSI
    int max_scan_reporting_threshold;   // max possible report_threshold as described   in
                                         //wifi_scan_cmd_params
    int max_hotlist_bssids;              // maximum number of entries for hotlist BSSIDs
   int max_hotlist_ssids;                // maximum number of entries for hotlist SSIDs
  int max_bssid_history_entries;        // number of BSSID/RSSI entries that device can hold
} wifi_gscan_capabilities;
```

G-Scan start (1 of 3)

- wlan_hdd_cfg80211_extscan_start()
  - Start the G-Scan with the configured parameters.
  - Input: G-Scan configuration parameters per bucket (channels to be scanned, dwell times, reporting rate etc.)
  - Output: Success/failure of this API.
  - A call back registered to the Android HAL gets invoked based on the following configuration:
- wlan_hdd_cfg80211_extscan_start_rsp()
  - Android HAL callback functions structure are invoked when scan results are available.


Android Hal
NL Interface
HDD
WDA
WDI
Firmware
HDD waits for the response from the firmware for the issued request to start GSCAN
wifi_start_gscan
Handle the Request and construct the corresponding NL80211 vendor specific command Wait for the response event from the host driver
QCA_NL80211_VENDOR_SUBCMD_GSCAN (START)
WDA_GSCAN_START_REQ
WDI_GSCAN_START_REQ
WLAN_HAL_GSCAN_START_REQ
QCA_NL80211_VENDOR_SUBCMD_GSCAN (START)
WDA_GSCAN_START_RSP
WDI_GSCAN_START_RSP
WLAN_HAL_GSCAN_START_RSP

G-Scan start (2 of 3)

typedef struct {

int bucket; // bucket index, 0 based

wifi_band band; // when UNSPECIFIED, use channel list

int period; // desired period, in millisecond; if this is too low, the firmware should choose to generate results as fast as it can instead of failing the command. For exponential backoff bucket this is the min_period.

/ report_events semantics - This is a bit field; which defines following bits:*

report_events semantics	Description
REPORT_EVENTS_BUFFER_FULL	Report only when scan history is X% full, where X is the obtained buffer percentage.
REPORT_EVENTS_EACH_SCAN	Report a scan completion event
REPORT_EVENTS_FULL_RESULTS	Forward scan results (beacons/probe responses + IEs) in real time to HAL, in addition to completion events.
REPORT_EVENTS_NO_BATCH	Controls batching, 0–batching, 1–no batching

Note: *To keep backward compatibility, fire completion events regardless of REPORT_EVENTS_EACH_SCAN.*

byte report_events;

int max_period; // if max_period is non zero or different than period, then this bucket is an exponential backoff bucket and the scan period will grow exponentially as per formula: actual_period(N) = period ^ (N/(step_count+1)) to a maximum period of max_period

*int exponent; // for exponential back off bucket: multiplier: new_period=old_period*exponent*

int step_count; // for exponential back off bucket, number of scans performed at a given period and until the exponent is applied

int num_channels; // channels to scan; these may include DFS channels. Note that a given channel may appear in multiple buckets

wifi_scan_channel_spec channels [MAX_CHANNELS];

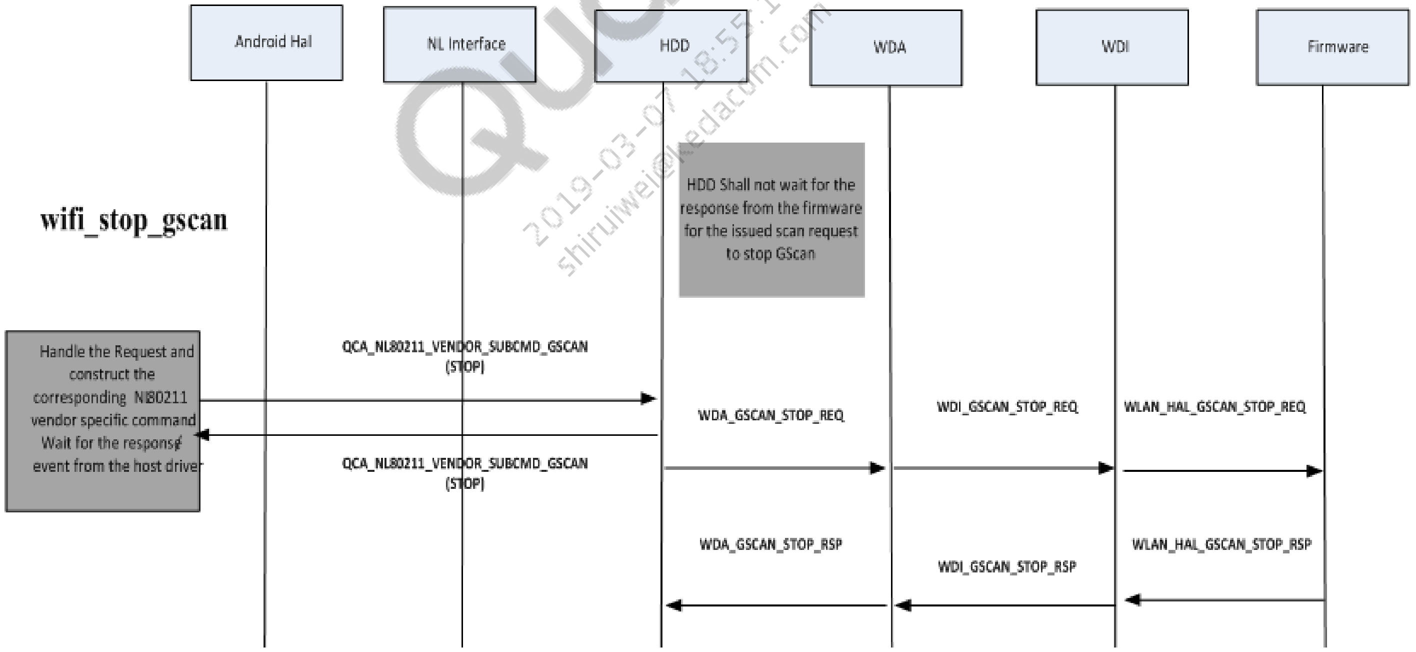
} wifi_scan_bucket_spec;

G-Scan start (3 of 3)

- An exponential scan bucket is defined by the following attributes in wifi_scan_bucket_spec:
  - min_period: For exponential backoff bucket this is the min_period
  - max_period: If max_period is non zero or different than period, then this bucket is an exponential backoff bucket and the scan period will grow exponentially as per formula:
    - actual_period(N) = period ^ (N/(step_count+1)) to a maximum period of max_period.
      - base: For exponential back off bucket: multiplier: new_period=old_period*base
      - step_count : For exponential back off bucket, number of scans performed at a given period and until the exponent is applied.
- The following are generated as a part of the G-Scan results:
  - Report_threshold_percent: This maps to REPORT_EVENTS_BUFFER_FULL event configuration.
  - Report_threshold_num_scans: The number of scans until G-Scan forwards the scan results to Host. This maps to REPORT_EVENTS_EACH_SCAN event configuration.

G-Scan stop

- wlan_hdd_cfg80211_extscan_stop()
  - Stop the G-Scan operation.
  - Input: Request ID assigned while invoking G-Scan start.
  - Output: Success/failure of this API.


Android Hal
NL Interface
HDD
WDA
WDI
Firmware
wifi_stop_gscan
HDD Shall not wait for the response from the firmware for the issued scan request to stop GScan
Handle the Request and construct the corresponding Nl80211 vendor specific command Wait for the response event from the host driver
QCA_NL80211_VENDOR_SUBCMD_GSCAN (STOP)
QCA_NL80211_VENDOR_SUBCMD_GSCAN (STOP)
WDA_GSCAN_STOP_REQ
WDI_GSCAN_STOP_REQ
WLAN_HAL_GSCAN_STOP_REQ
WDA_GSCAN_STOP_RSP
WDI_GSCAN_STOP_RSP
WLAN_HAL_GSCAN_STOP_RSP

G-Scan get cached results

- wlan_hdd_cfg80211_extscan_get_cached_results()
  - Get G-Scan cached results are categorized by the following report events.

Report events	Description
REPORT_EVENTS_EACH_SCAN(1)	Report a scan completion event after scan. If this is not set then scan completion events should be reported if report_threshold_percent or report_threshold_num_scans is reached
REPORT_EVENTS_FULL_RESULTS(2)	Forward scan results (beacons/probe responses + IEs) in real time to HAL, in addition to completion events.
REPORT_EVENTS_NO_BATCH(4)	Controls if scans for this bucket should be placed in the history buffer

Note: To keep backward compatibility, fire completion events regardless of REPORT_EVENTS_EACH_SCAN.

  - Input: Request ID assigned while invoking G-Scan start.
  - Output: Success/failure of this API.
  - A call back registered to the Android HAL gets invoked based on the configuration.

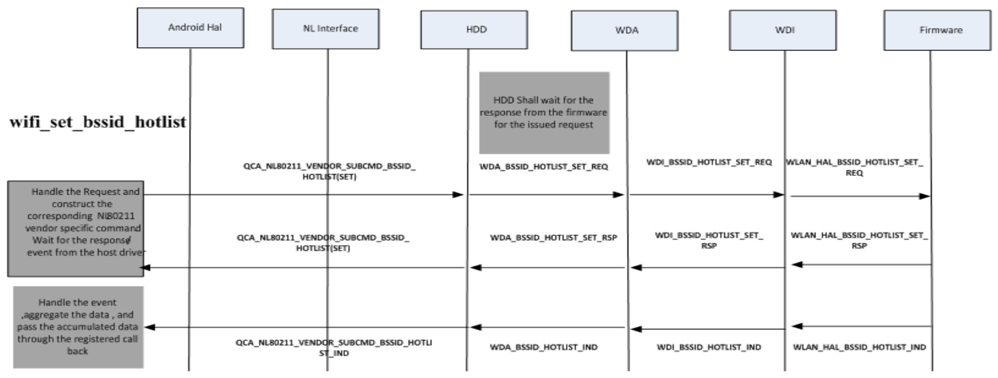
```
typedef struct {
    /* reported when each probe response is received, if report_events
    * enabled in wifi_scan_cmd_params. buckets_scanned is a bitset of the
    * buckets that are currently being scanned. See the buckets_scanned field
    * in the wifi_cached_scan_results struct for more details.     */
    void (*on_full_scan_result) (wifi_request_id id, wifi_scan_result *result, unsigned buckets_scanned);
    /* indicates progress of scanning statemachine */
    void (*on_scan_event) (wifi_request_id id, wifi_scan_event event);
    } wifi_scan_result_handler;
```

BSSID Hotlist

Set BSSID Hotlist

- wlan_hdd_cfg80211_extscan_set_bssid_hotlist()
  - Configures the BSSID’s to be monitored by the firmware for hotlist.
  - Expects an event from the firmware, whenever an AP from the list is found/lost.
  - Expects events only on state change (Found > Lost or Lost > Found but not Lost > Lost or Found > Found).
  - Input: Hotlist AP’s BSSID.
  - Output: Success/failure of this API.
  - A call back registered to the Android HAL getS invoked whenever hotlist BSSID state changes.

```
typedef struct {
void (*on_hotlist_ap_found)(wifi_request_id id, unsigned num_results, wifi_scan_result *results);
void (*on_hotlist_ap_lost)(wifi_request_id id, unsigned num_results, wifi_scan_result *results);
} wifi_hotlist_ap_found_handler;
```


Android Hal
NL Interface
HDD
WDA
WDI
Firmware
wifi_set_bssid_hotlist
HDD Shall wait for the response from the firmware for the issued request
Handle the Request and construct the corresponding Nl80211 vendor specific command Wait for the response event from the host driver
QCA_NL80211_VENDOR_SUBCMD_BSSID_HOTLIST(SET)
WDA_BSSID_HOTLIST_SET_REQ
WDI_BSSID_HOTLIST_SET_REQ
WLAN_HAL_BSSID_HOTLIST_SET_REQ
QCA_NL80211_VENDOR_SUBCMD_BSSID_HOTLIST(SET)
WDA_BSSID_HOTLIST_SET_RSP
WDI_BSSID_HOTLIST_SET_RSP
WLAN_HAL_BSSID_HOTLIST_SET_RSP
Handle the event ,aggregate the data , and pass the accumulated data through the registered call back
QCA_NL80211_VENDOR_SUBCMD_BSSID_HOTLIST_IND
WDA_BSSID_HOTLIST_IND
WDI_BSSID_HOTLIST_IND
WLAN_HAL_BSSID_HOTLIST_IND

Reset BSSID Hotlist

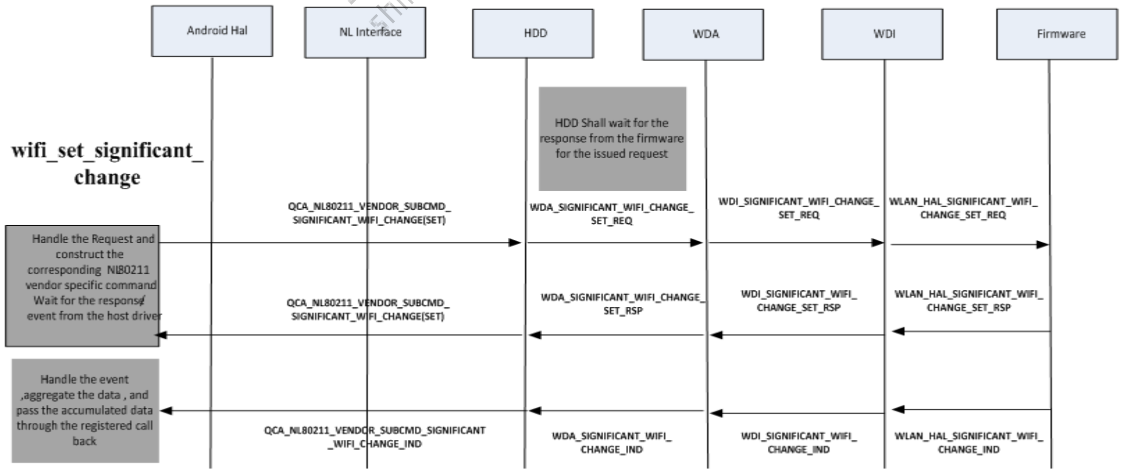
- wifi_reset_bssid_hotlist:
  - wlan_hdd_cfg80211_extscan_reset_bssid_hotlist()
  - Remove all the entries from the hotlist monitoring.
  - Input: Request ID assigned while invoking G-Scan start.
  - Output: Success/failure of this API.

Significant RSSI

Set Significant RSSI

- wifi_set_significant_change:
  - wlan_hdd_cfg80211_extscan_set_significant_change()
  - Configures the BSSID’s to be monitored by the firmware for RSSI changes.
  - Input: Configures the parameters for RSSI monitoring to the host driver/firmware.
  - Output: Success/failure of this API.
  - A call back registered to the Android HAL gets invoked whenever RSSI changes.

```
typedef struct {
  void (*on_significant_change)(wifi_request_id id, unsigned num_results, wifi_significant_change_result **results);
 } wifi_significant_change_handler;
```


Android Hal
NL Interface
HDD
WDA
WDI
Firmware
wifi_set_significant_change
HDD Shall wait for the response from the firmware for the issued request
Handle the Request and construct the corresponding NL80211 vendor specific command Wait for the response event from the host driver
QCA_NL80211_VENDOR_SUBCMD_SIGNIFICANT_WIFI_CHANGE(SET)
WDA_SIGNIFICANT_WIFI_CHANGE_SET_REQ
WDI_SIGNIFICANT_WIFI_CHANGE_SET_REQ
WLAN_HAL_SIGNIFICANT_WIFI_CHANGE_SET_REQ
QCA_NL80211_VENDOR_SUBCMD_SIGNIFICANT_WIFI_CHANGE(SET)
WDA_SIGNIFICANT_WIFI_CHANGE_SET_RSP
WDI_SIGNIFICANT_WIFI_CHANGE_SET_RSP
WLAN_HAL_SIGNIFICANT_WIFI_CHANGE_SET_RSP
Handle the event ,aggregate the data , and pass the accumulated data through the registered call back
QCA_NL80211_VENDOR_SUBCMD_SIGNIFICANT_WIFI_CHANGE_IND
WDA_SIGNIFICANT_WIFI_CHANGE_IND
WDI_SIGNIFICANT_WIFI_CHANGE_IND
WLAN_HAL_SIGNIFICANT_WIFI_CHANGE_IND

Reset Significant RSSI

- wifi_reset_significant_change:
  - wlan_hdd_cfg80211_extscan_reset_significant_change()
  - Resets the parameters configured with set significant and stops monitoring for the significant RSSI changes.
  - Input: Request ID assigned while invoking G-Scan start.
  - Output: Success/failure of this API.

High-level Test Setup (OTA)

High-level Test Setup (OTA)

- G-Scan test setup contains DUT with Wi-Fi on and a set of AP’s that are configured in different channels/band.
- G-Scan Implementation exercises the execution of the different functionalities (start G-Scan, stop G-Scan, etc).
- HAL Proxy Daemon is a proprietary application used to verify G-Scan functionalities.

HAL Proxy Daemon Application

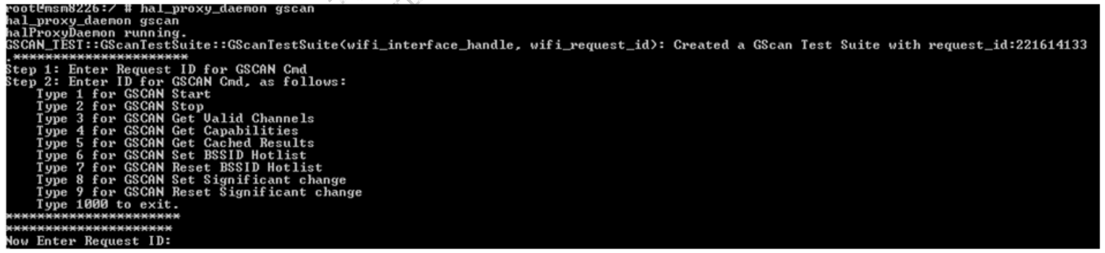
HAL proxy daemon Application

- HAL Proxy Daemon is a proprietary application developed by Qualcomm® to mimic the functionality of the Android framework to interact with the Android HAL layer.
- The following are the steps for HAL Proxy Daemon:
  - `adb shell`
  - `adb push libwifi-hal-qcom.so /system/lib`
  - `adb push hal_proxy_daemon /system/bin`
  - `chmod +x /system/bin/hal_proxy_daemon`
- G-Scan implementation exercises the execution of the following functionalities (API’s):
  - Get Valid Channels
  - Get Capabilities
  - Start G-Scan
  - Stop G-Scan
  - Get Cached Results
  - Set BSSID Hotlist
  - Reset BSSID Hotlist
  - Set Significant RSSI
  - Reset Significant RSSI

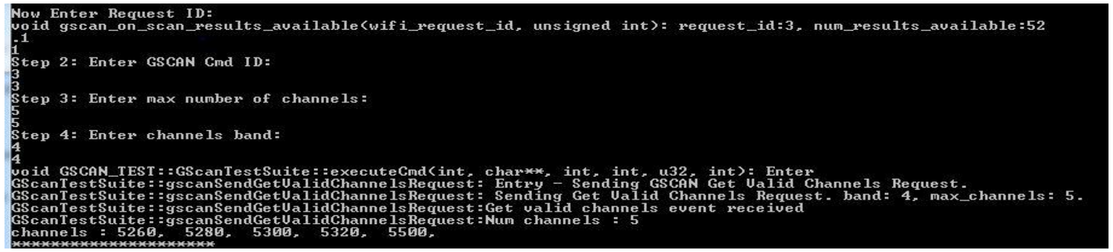
Test Cases

Test Cases (1 of 12)

- Get Valid Channels:
  - Turn on Wi-Fi.
  - Execute the command “adb shell hal_proxy_daemon gscan”.
  - A command line option to execute a specific command is prompted.
  - Select the corresponding options to get the valid channels.
    - Select the number of channels needed.
    - Select the band.
  - The Information of the valid channels are displayed on the CLI prompt.



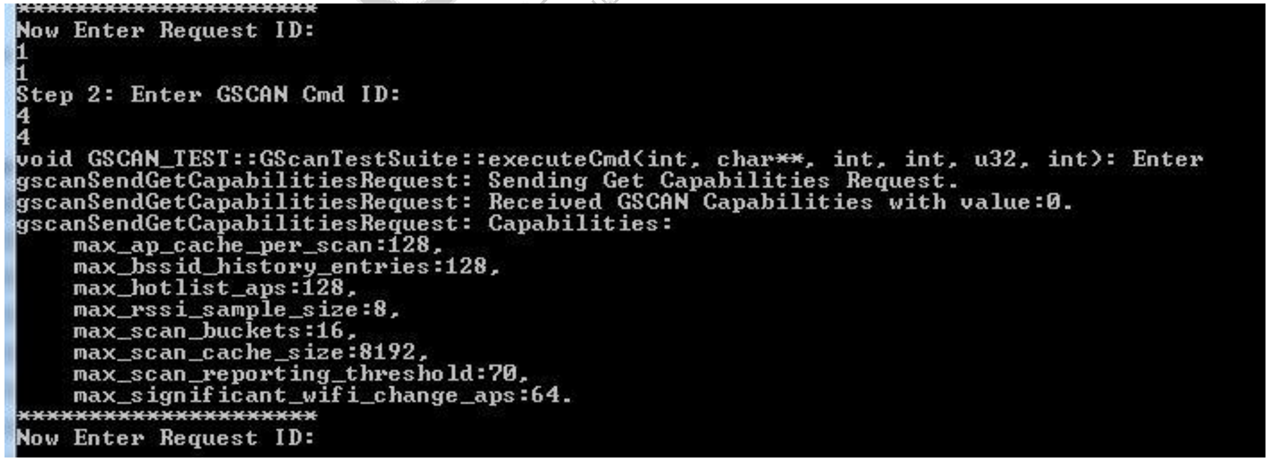
```
root@msm8226:/ # hal_proxy_daemon gscan
hal_proxy_daemon gscan
halProxyDaemon running.
GSCAN_TEST::GScanTestSuite::GScanTestSuite(wifi_interface_handle, wifi_request_id): Created a GScan Test Suite with request_id:221614133
.**********************
Step 1: Enter Request ID for GSCAN Cmd
Step 2: Enter ID for GSCAN Cmd, as follows:
    Type 1 for GSCAN Start
    Type 2 for GSCAN Stop
    Type 3 for GSCAN Get Valid Channels
    Type 4 for GSCAN Get Capabilities
    Type 5 for GSCAN Get Cached Results
    Type 6 for GSCAN Set BSSID Hotlist
    Type 7 for GSCAN Reset BSSID Hotlist
    Type 8 for GSCAN Set Significant change
    Type 9 for GSCAN Reset Significant change
    Type 1000 to exit.
**********************
*********************
Now Enter Request ID:
```



```
Now Enter Request ID:
void gscan_on_scan_results_available(wifi_request_id, unsigned int): request_id:3, num_results_available:52
.1
1
Step 2: Enter GSCAN Cmd ID:
3
3
Step 3: Enter max number of channels:
5
5
Step 4: Enter channels band:
4
4
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
GScanTestSuite::gscanSendGetValidChannelsRequest: Entry - Sending GSCAN Get Valid Channels Request.
GScanTestSuite::gscanSendGetValidChannelsRequest: Sending Get Valid Channels Request. band: 4, max_channels: 5.
GScanTestSuite::gscanSendGetValidChannelsRequest:Get valid channels event received
GScanTestSuite::gscanSendGetValidChannelsRequest:Num channels : 5
channels : 5260,  5280,  5300,  5320,  5500,
**************************
```

Test Cases (2 of 12)

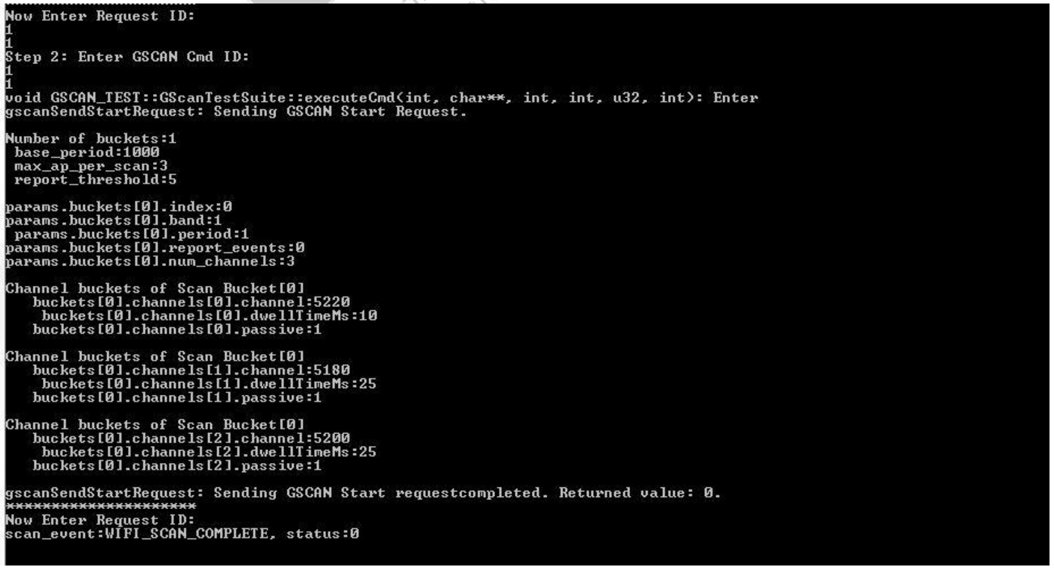
- Get G-Scan Capabilities:
  - Turn on Wi-Fi.
  - Execute the command “adb shell hal_proxy_daemon gscan”.
  - A command line option to execute a specific command is prompted.
  - Select the corresponding options to get the G-Scan Capabilities.
  - Information about G-Scan capabilities is displayed on the CLI prompt.



```
*********************
Now Enter Request ID:
1
1
Step 2: Enter GSCAN Cmd ID:
4
4
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
gscanSendGetCapabilitiesRequest: Sending Get Capabilities Request.
gscanSendGetCapabilitiesRequest: Received GSCAN Capabilities with value:0.
gscanSendGetCapabilitiesRequest: Capabilities:
    max_ap_cache_per_scan:128,
    max_bssid_history_entries:128,
    max_hotlist_aps:128,
    max_rssi_sample_size:8,
    max_scan_buckets:16,
    max_scan_cache_size:8192,
    max_scan_reporting_threshold:70,
    max_significant_wifi_change_aps:64.
*********************
Now Enter Request ID:
```

Test Cases (3 of 12)

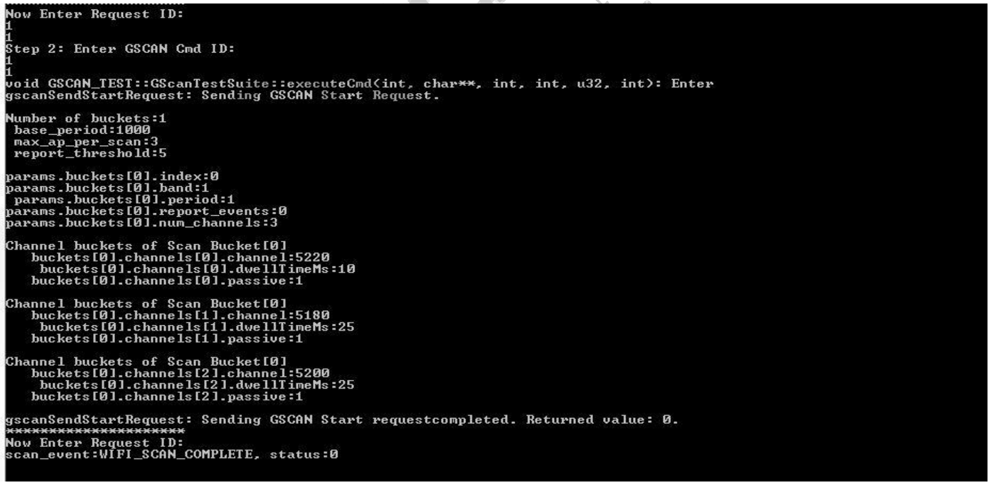
- Start G-Scan:
  - Turn on Wi-Fi.
  - Execute the command “adb shell hal_proxy_daemon gscan”.
  - A command line option to execute a specific command is prompted.
  - Select the corresponding option to start the G-Scan operation.
  - The command line options for the start of the G-Scan command can be input through the file gscan_start_params.txt.
  - Place the file at /etc/wifi/ in the device.



```
Now Enter Request ID:
1
1
Step 2: Enter GSCAN Cmd ID:
1
1
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
gscanSendStartRequest: Sending GSCAN Start Request.

Number of buckets:1
 base_period:1000
 max_ap_per_scan:3
 report_threshold:5

params.buckets[0].index:0
params.buckets[0].band:1
 params.buckets[0].period:1
params.buckets[0].report_events:0
params.buckets[0].num_channels:3

Channel buckets of Scan Bucket[0]
   buckets[0].channels[0].channel:5220
    buckets[0].channels[0].dwellTimeMs:10
   buckets[0].channels[0].passive:1

Channel buckets of Scan Bucket[0]
   buckets[0].channels[1].channel:5180
    buckets[0].channels[1].dwellTimeMs:25
   buckets[0].channels[1].passive:1

Channel buckets of Scan Bucket[0]
   buckets[0].channels[2].channel:5200
    buckets[0].channels[2].dwellTimeMs:25
   buckets[0].channels[2].passive:1

gscanSendStartRequest: Sending GSCAN Start requestcompleted. Returned value: 0.
*********************
Now Enter Request ID:
scan_event:WIFI_SCAN_COMPLETE, status:0
```

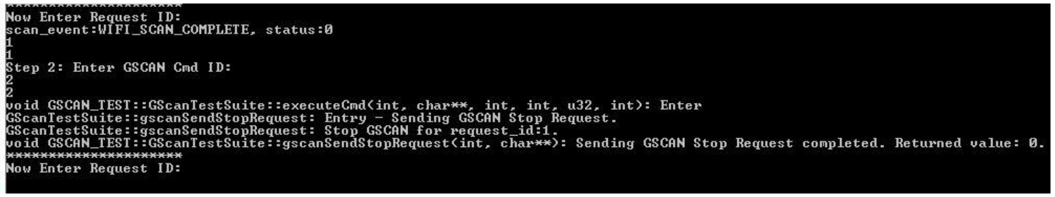
Test Cases (4 of 12)

- Stop G-Scan:
  - Turn on Wi-Fi
  - Execute the command “adb shell hal_proxy_daemon gscan”.
  - A command line option to execute a specific command is prompted.
  - Select the corresponding option to stop the G-Scan operation.



```
Now Enter Request ID:
1
1
Step 2: Enter GSCAN Cmd ID:
1
1
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
gscanSendStartRequest: Sending GSCAN Start Request.

Number of buckets:1
 base_period:1000
 max_ap_per_scan:3
 report_threshold:5

params.buckets[0].index:0
params.buckets[0].band:1
 params.buckets[0].period:1
params.buckets[0].report_events:0
params.buckets[0].num_channels:3

Channel buckets of Scan Bucket[0]
   buckets[0].channels[0].channel:5220
    buckets[0].channels[0].dwellTimeMs:10
   buckets[0].channels[0].passive:1

Channel buckets of Scan Bucket[0]
   buckets[0].channels[1].channel:5180
    buckets[0].channels[1].dwellTimeMs:25
   buckets[0].channels[1].passive:1

Channel buckets of Scan Bucket[0]
   buckets[0].channels[2].channel:5200
    buckets[0].channels[2].dwellTimeMs:25
   buckets[0].channels[2].passive:1

gscanSendStartRequest: Sending GSCAN Start requestcompleted. Returned value: 0.
*********************
Now Enter Request ID:
scan_event:WIFI_SCAN_COMPLETE, status:0
```

  - The following prompt points to the Stop G-Scan operation.



```
*********************
Now Enter Request ID:
scan_event:WIFI_SCAN_COMPLETE, status:0
1
1
Step 2: Enter GSCAN Cmd ID:
2
2
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
GScanTestSuite::gscanSendStopRequest: Entry - Sending GSCAN Stop Request.
GScanTestSuite::gscanSendStopRequest: Stop GSCAN for request_id:1.
void GSCAN_TEST::GScanTestSuite::gscanSendStopRequest(int, char**): Sending GSCAN Stop Request completed. Returned value: 0.
*********************
Now Enter Request ID:
```

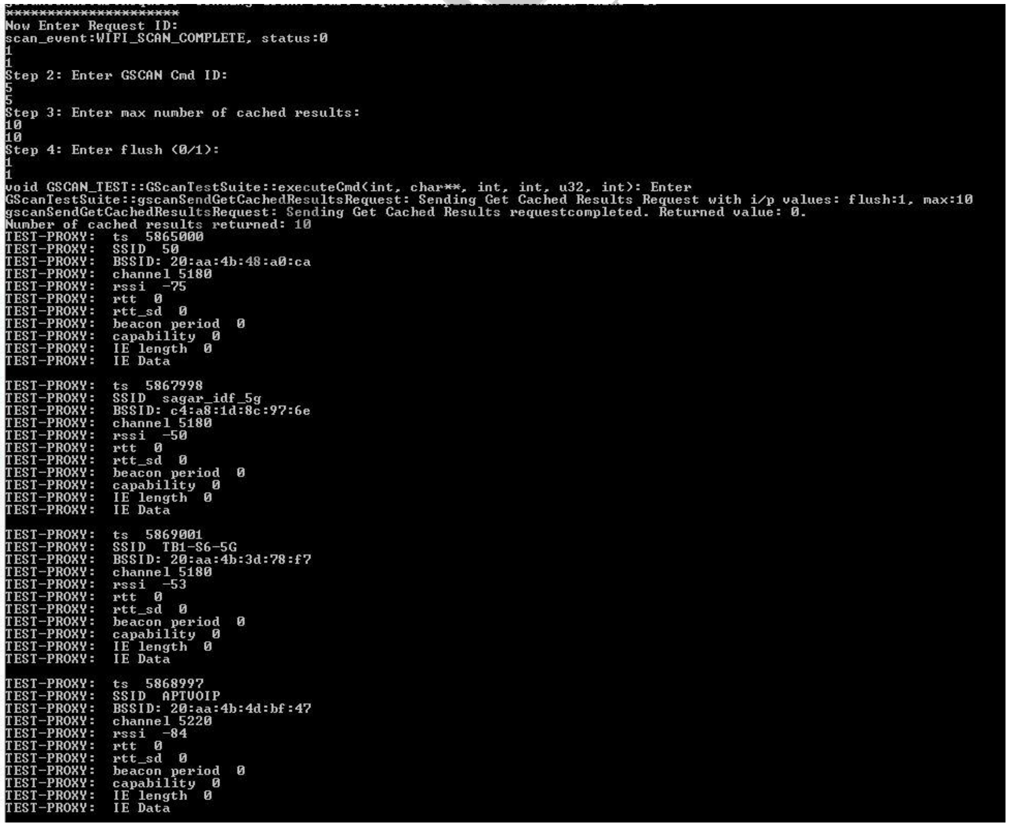
Test Cases (5 of 12)

- Get G-Scan Cached Results:
  - Turn on Wi-Fi.
  - Execute the command “adb shell hal_proxy_daemon gscan”.
  - A command line option to execute a specific command is prompted.
  - Select the corresponding option to start the Get G-Scan cache results.

Report event	Description
Report event 0	Only indication from the firmware shall be on buffer full. The results captured in the firmware shall be obtained on every ask by the user by get cached results command (“wifi_get_cached_gscan_results” API)
Report event 1	Similar to 0. Additionally, firmware sends “scan results available” indication to the application whenever the report threshold configured in the start G-Scan is reached.
Report event 2	Similar to 0 and 1. Additionally, firmware shall pass the beacons/probe responses obtained during the scan to the upper layers/application.

Test Cases (6 of 12)

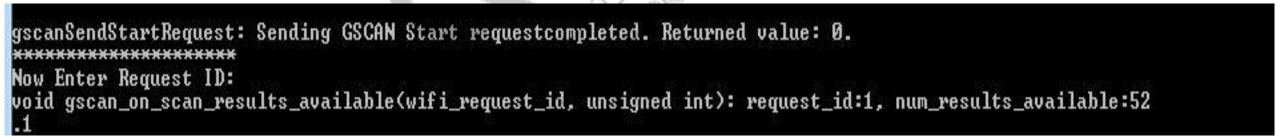
- Report Event 1:
  - Ensure that the buckets are configured with the correct settings for Report Event 1.
  - Refer to the steps for start G-Scan to trigger the same.
  - Select the corresponding option to get G-Scan cached results.



```
*********************
Now Enter Request ID:
scan_event:WIFI_SCAN_COMPLETE, status:0
1
1
Step 2: Enter GSCAN Cmd ID:
5
5
Step 3: Enter max number of cached results:
10
10
Step 4: Enter flush (0/1):
1
1
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
GScanTestSuite::gscanSendGetCachedResultsRequest: Sending Get Cached Results Request with i/p values: flush:1, max:10
gscanSendGetCachedResultsRequest: Sending Get Cached Results requestcompleted. Returned value: 0.
Number of cached results returned: 10
TEST-PROXY:  ts  5865000
TEST-PROXY:  SSID  50
TEST-PROXY:  BSSID: 20:aa:4b:48:a0:ca
TEST-PROXY:  channel 5180
TEST-PROXY:  rssi  -75
TEST-PROXY:  rtt  0
TEST-PROXY:  rtt_sd  0
TEST-PROXY:  beacon period  0
TEST-PROXY:  capability  0
TEST-PROXY:  IE length  0
TEST-PROXY:  IE Data

TEST-PROXY:  ts  5867998
TEST-PROXY:  SSID  sagar_idf_5g
TEST-PROXY:  BSSID: c4:a8:1d:8c:97:6e
TEST-PROXY:  channel 5180
TEST-PROXY:  rssi  -50
TEST-PROXY:  rtt  0
TEST-PROXY:  rtt_sd  0
TEST-PROXY:  beacon period  0
TEST-PROXY:  capability  0
TEST-PROXY:  IE length  0
TEST-PROXY:  IE Data

TEST-PROXY:  ts  5869001
TEST-PROXY:  SSID  TB1-S6-5G
TEST-PROXY:  BSSID: 20:aa:4b:3d:78:f7
TEST-PROXY:  channel 5180
TEST-PROXY:  rssi  -53
TEST-PROXY:  rtt  0
TEST-PROXY:  rtt_sd  0
TEST-PROXY:  beacon period  0
TEST-PROXY:  capability  0
TEST-PROXY:  IE length  0
TEST-PROXY:  IE Data

TEST-PROXY:  ts  5868997
TEST-PROXY:  SSID  APTVOIP
TEST-PROXY:  BSSID: 20:aa:4b:4d:bf:47
TEST-PROXY:  channel 5220
TEST-PROXY:  rssi  -84
TEST-PROXY:  rtt  0
TEST-PROXY:  rtt_sd  0
TEST-PROXY:  beacon period  0
TEST-PROXY:  capability  0
TEST-PROXY:  IE length  0
TEST-PROXY:  IE Data
```

Test Cases (7 of 12)

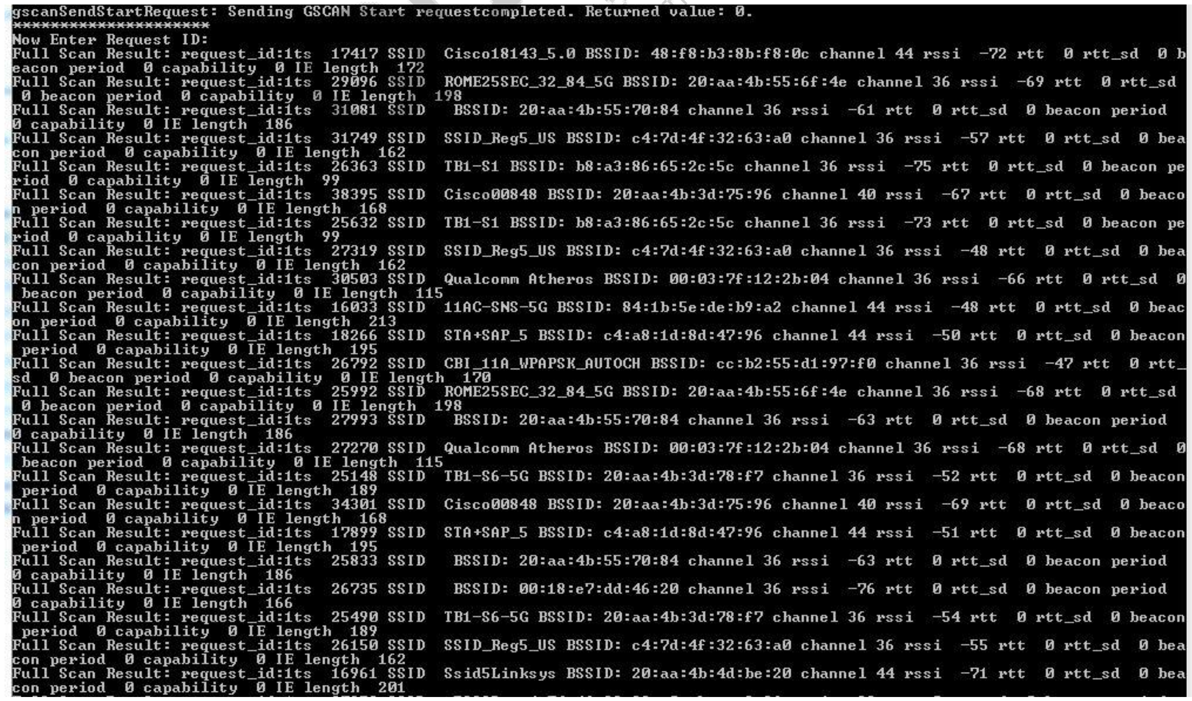
- Report Event 2:
  - Ensure that the buckets are configured with the correct settings for Report Event 2.
  - Refer to the steps for start G-Scan to trigger the same.
  - Select the corresponding option to get G-Scan cached results.
  - The output of G-Scan cached results are displayed on the CLI prompt.
  - The following indication indicates that the firmware has triggered the event ‘scan results available’. User can query for the scan results by giving the command to get cached results.



```
gscanSendStartRequest: Sending GSCAN Start requestcompleted. Returned value: 0.
**********************
Now Enter Request ID:
void gscan_on_scan_results_available(wifi_request_id, unsigned int): request_id:1, num_results_available:52
.1
```

Test Cases (8 of 12)

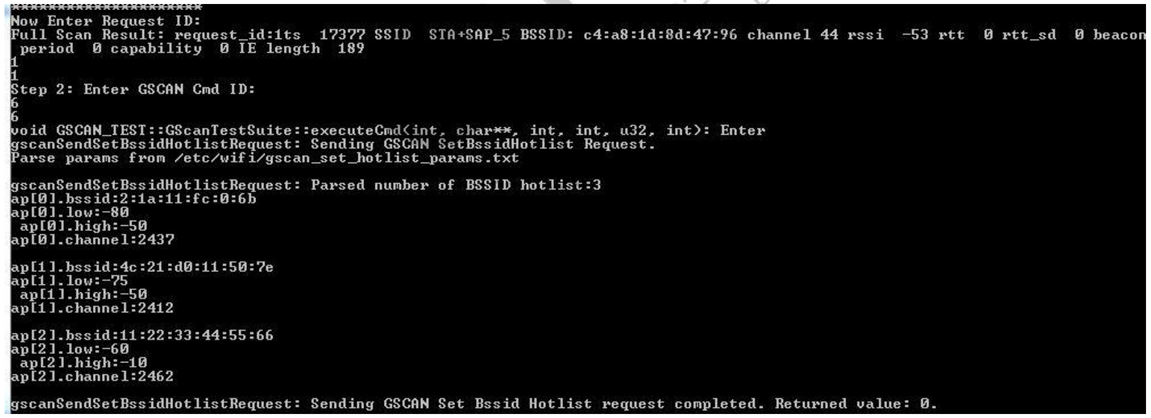
- Report Event 4:
  - Ensure that the buckets are configured with the correct settings for Report Event 4.
  - Refer to the steps for start G-Scan to trigger the same.
  - Select the corresponding option to get G-Scan cached results.
  - The output of G-Scan cached results obtained from the firmware are displayed on the CLI prompt.



```
gscanSendStartRequest: Sending GSCAN Start requestcompleted. Returned value: 0.
**********************
Now Enter Request ID:
Full Scan Result: request_id:1ts  17417 SSID  Cisco18143_5.0 BSSID: 48:f8:b3:8b:f8:0c channel 44 rssi  -72 rtt  0 rtt_sd  0 b
eacon period  0 capability  0 IE length  172
Full Scan Result: request_id:1ts  29096 SSID  ROME25SEC_32_84_5G BSSID: 20:aa:4b:55:6f:4e channel 36 rssi  -69 rtt  0 rtt_sd
 0 beacon period  0 capability  0 IE length  198
Full Scan Result: request_id:1ts  31081 SSID   BSSID: 20:aa:4b:55:70:84 channel 36 rssi  -61 rtt  0 rtt_sd  0 beacon period
0 capability  0 IE length  186
Full Scan Result: request_id:1ts  31749 SSID  SSID_Reg5_US BSSID: c4:7d:4f:32:63:a0 channel 36 rssi  -57 rtt  0 rtt_sd  0 bea
con period  0 capability  0 IE length  162
Full Scan Result: request_id:1ts  26363 SSID  TB1-S1 BSSID: b8:a3:86:65:2c:5c channel 36 rssi  -75 rtt  0 rtt_sd  0 beacon pe
riod  0 capability  0 IE length  99
Full Scan Result: request_id:1ts  38395 SSID  Cisco00848 BSSID: 20:aa:4b:3d:75:96 channel 40 rssi  -67 rtt  0 rtt_sd  0 beaco
n period  0 capability  0 IE length  168
Full Scan Result: request_id:1ts  25632 SSID  TB1-S1 BSSID: b8:a3:86:65:2c:5c channel 36 rssi  -73 rtt  0 rtt_sd  0 beacon pe
riod  0 capability  0 IE length  99
Full Scan Result: request_id:1ts  27319 SSID  SSID_Reg5_US BSSID: c4:7d:4f:32:63:a0 channel 36 rssi  -48 rtt  0 rtt_sd  0 bea
con period  0 capability  0 IE length  162
Full Scan Result: request_id:1ts  30503 SSID  Qualcomm Atheros BSSID: 00:03:7f:12:2b:04 channel 36 rssi  -66 rtt  0 rtt_sd  0
 beacon period  0 capability  0 IE length  115
Full Scan Result: request_id:1ts  16033 SSID  11AC-SNS-5G BSSID: 84:1b:5e:de:b9:a2 channel 44 rssi  -48 rtt  0 rtt_sd  0 beac
on period  0 capability  0 IE length  213
Full Scan Result: request_id:1ts  18266 SSID  STA+SAP_5 BSSID: c4:a8:1d:8d:47:96 channel 44 rssi  -50 rtt  0 rtt_sd  0 beacon
 period  0 capability  0 IE length  195
Full Scan Result: request_id:1ts  26792 SSID  CBI_11A_WPAPSK_AUTOCH BSSID: cc:b2:55:d1:97:f0 channel 36 rssi  -47 rtt  0 rtt_
sd  0 beacon period  0 capability  0 IE length  170
Full Scan Result: request_id:1ts  25992 SSID  ROME25SEC_32_84_5G BSSID: 20:aa:4b:55:6f:4e channel 36 rssi  -68 rtt  0 rtt_sd
 0 beacon period  0 capability  0 IE length  198
Full Scan Result: request_id:1ts  27993 SSID   BSSID: 20:aa:4b:55:70:84 channel 36 rssi  -63 rtt  0 rtt_sd  0 beacon period
0 capability  0 IE length  186
Full Scan Result: request_id:1ts  27270 SSID  Qualcomm Atheros BSSID: 00:03:7f:12:2b:04 channel 36 rssi  -68 rtt  0 rtt_sd  0
 beacon period  0 capability  0 IE length  115
Full Scan Result: request_id:1ts  25148 SSID  TB1-S6-5G BSSID: 20:aa:4b:3d:78:f7 channel 36 rssi  -52 rtt  0 rtt_sd  0 beacon
 period  0 capability  0 IE length  189
Full Scan Result: request_id:1ts  34301 SSID  Cisco00848 BSSID: 20:aa:4b:3d:75:96 channel 40 rssi  -69 rtt  0 rtt_sd  0 beaco
n period  0 capability  0 IE length  168
Full Scan Result: request_id:1ts  17899 SSID  STA+SAP_5 BSSID: c4:a8:1d:8d:47:96 channel 44 rssi  -51 rtt  0 rtt_sd  0 beacon
 period  0 capability  0 IE length  195
Full Scan Result: request_id:1ts  25833 SSID   BSSID: 20:aa:4b:55:70:84 channel 36 rssi  -63 rtt  0 rtt_sd  0 beacon period
0 capability  0 IE length  186
Full Scan Result: request_id:1ts  26735 SSID   BSSID: 00:18:e7:dd:46:20 channel 36 rssi  -76 rtt  0 rtt_sd  0 beacon period
0 capability  0 IE length  166
Full Scan Result: request_id:1ts  25490 SSID  TB1-S6-5G BSSID: 20:aa:4b:3d:78:f7 channel 36 rssi  -54 rtt  0 rtt_sd  0 beacon
 period  0 capability  0 IE length  189
Full Scan Result: request_id:1ts  26150 SSID  SSID_Reg5_US BSSID: c4:7d:4f:32:63:a0 channel 36 rssi  -55 rtt  0 rtt_sd  0 bea
con period  0 capability  0 IE length  162
Full Scan Result: request_id:1ts  16961 SSID  Ssid5Linksys BSSID: 20:aa:4b:4d:be:20 channel 44 rssi  -71 rtt  0 rtt_sd  0 bea
con period  0 capability  0 IE length  201
```

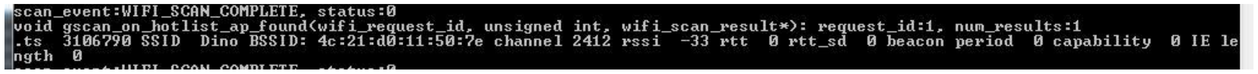
Test Cases (9 of 12)

- Set BSSID hotlist:
  - Start the G-Scan.
  - Enter the parameters for the set BSSID hotlist command in the file gscan_set_hotlist_params.txt.
  - Place the file in /etc/wifi location.
  - Select the corresponding command to set the BSSID hotlist option.
  - The output of the set BSSID hotlist displayed on the CLI prompt.



```
**********************
Now Enter Request ID:
Full Scan Result: request_id:1ts  17377 SSID  STA+SAP_5 BSSID: c4:a8:1d:8d:47:96 channel 44 rssi  -53 rtt  0 rtt_sd  0 beacon
 period  0 capability  0 IE length  189
1
1
Step 2: Enter GSCAN Cmd ID:
6
6
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
gscanSendSetBssidHotlistRequest: Sending GSCAN SetBssidHotlist Request.
Parse params from /etc/wifi/gscan_set_hotlist_params.txt

gscanSendSetBssidHotlistRequest: Parsed number of BSSID hotlist:3
ap[0].bssid:2:1a:11:fc:0:6b
ap[0].low:-80
 ap[0].high:-50
ap[0].channel:2437

ap[1].bssid:4c:21:d0:11:50:7e
ap[1].low:-75
 ap[1].high:-50
ap[1].channel:2412

ap[2].bssid:11:22:33:44:55:66
ap[2].low:-60
 ap[2].high:-10
ap[2].channel:2462

gscanSendSetBssidHotlistRequest: Sending GSCAN Set Bssid Hotlist request completed. Returned value: 0.
```

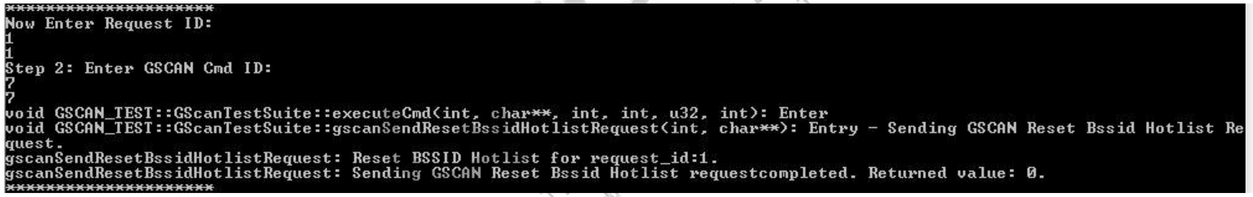
  - The following image indicates the BSSID configured as a part of the Set BSSID hotlist command corresponding to the AP find event:



```
scan_event:WIFI_SCAN_COMPLETE, status:0
void gscan_on_hotlist_ap_found(wifi_request_id, unsigned int, wifi_scan_result*): request_id:1, num_results:1
.ts  3106790 SSID  Dino BSSID: 4c:21:d0:11:50:7e channel 2412 rssi  -33 rtt  0 rtt_sd  0 beacon period  0 capability  0 IE le
ngth  0
```

Test Cases (10 of 12)

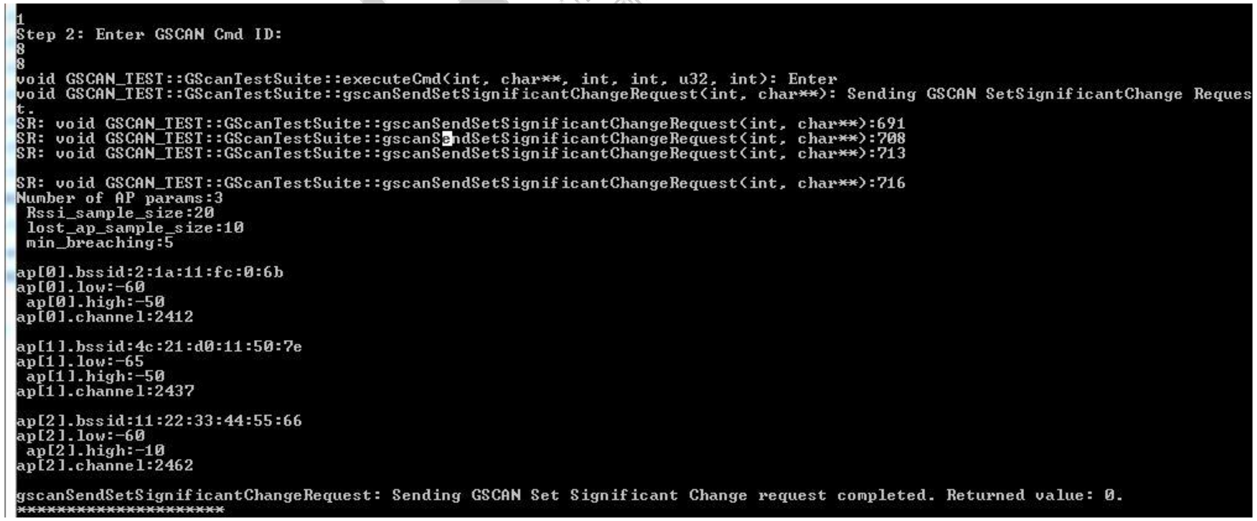
- Reset BSSID hotlist:
  - Ensure that BSSID hotlist option is set.
  - Chose the command line to reset BSSID hotlist option.
  - The output of the reset BSSID hotlist is displayed on the CLI prompt.



```
*********************
Now Enter Request ID:
1
1
Step 2: Enter GSCAN Cmd ID:
7
7
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
void GSCAN_TEST::GScanTestSuite::gscanSendResetBssidHotlistRequest(int, char**): Entry - Sending GSCAN Reset Bssid Hotlist Re
quest.
gscanSendResetBssidHotlistRequest: Reset BSSID Hotlist for request_id:1.
gscanSendResetBssidHotlistRequest: Sending GSCAN Reset Bssid Hotlist requestcompleted. Returned value: 0.
*********************
```

Test Cases (11 of 12)

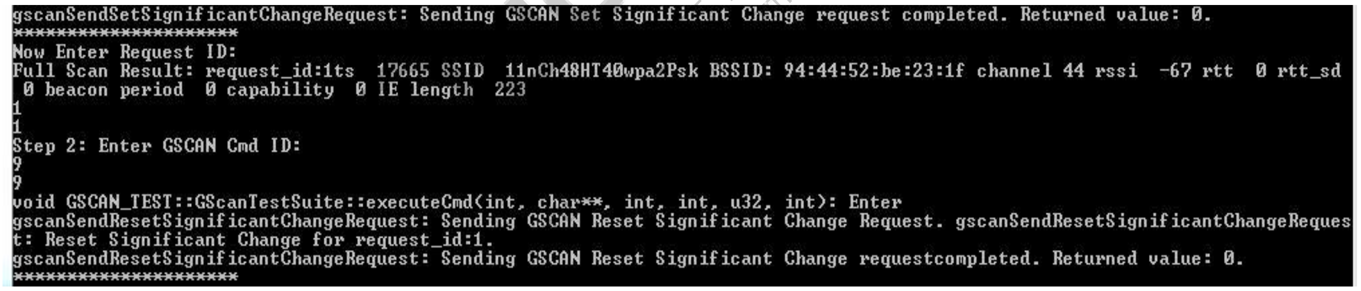
- Set significant RSSI:
  - Start the G-Scan.
  - Enter the parameters for the set significant RSSI command in the file gscan_set_significant_change_params.txt.
  - Placed the file in /etc/wifi location.
  - Select the corresponding command to set the significant RSSI option.
  - The output of this command is displayed on the CLI prompt.



```
1
Step 2: Enter GSCAN Cmd ID:
8
8
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
void GSCAN_TEST::GScanTestSuite::gscanSendSetSignificantChangeRequest(int, char**): Sending GSCAN SetSignificantChange Reques
t.
SR: void GSCAN_TEST::GScanTestSuite::gscanSendSetSignificantChangeRequest(int, char**):691
SR: void GSCAN_TEST::GScanTestSuite::gscanSendSetSignificantChangeRequest(int, char**):708
SR: void GSCAN_TEST::GScanTestSuite::gscanSendSetSignificantChangeRequest(int, char**):713

SR: void GSCAN_TEST::GScanTestSuite::gscanSendSetSignificantChangeRequest(int, char**):716
Number of AP params:3
 Rssi_sample_size:20
 lost_ap_sample_size:10
 min_breaching:5

ap[0].bssid:2:1a:11:fc:0:6b
ap[0].low:-60
 ap[0].high:-50
ap[0].channel:2412

ap[1].bssid:4c:21:d0:11:50:7e
ap[1].low:-65
 ap[1].high:-50
ap[1].channel:2437

ap[2].bssid:11:22:33:44:55:66
ap[2].low:-60
 ap[2].high:-10
ap[2].channel:2462

gscanSendSetSignificantChangeRequest: Sending GSCAN Set Significant Change request completed. Returned value: 0.
**********************
```

Test Cases (12 of 12)

- Reset significant RSSI:
  - Ensure that significant RSSI option is Set.
  - Chose the command line to Reset significant RSSI option.
  - The output of this command is displayed on the CLI prompt.



```
gscanSendSetSignificantChangeRequest: Sending GSCAN Set Significant Change request completed. Returned value: 0.
**********************
Now Enter Request ID:
Full Scan Result: request_id:1ts  17665 SSID  11nCh48HT40wpa2Psk BSSID: 94:44:52:be:23:1f channel 44 rssi  -67 rtt  0 rtt_sd
 0 beacon period  0 capability  0 IE length  223
1
1
Step 2: Enter GSCAN Cmd ID:
9
9
void GSCAN_TEST::GScanTestSuite::executeCmd(int, char**, int, int, u32, int): Enter
gscanSendResetSignificantChangeRequest: Sending GSCAN Reset Significant Change Request. gscanSendResetSignificantChangeReques
t: Reset Significant Change for request_id:1.
gscanSendResetSignificantChangeRequest: Sending GSCAN Reset Significant Change requestcompleted. Returned value: 0.
**********************
```

Configuration Parameters

Configuration parameters

- gEnableEXTScan:
  - 1 (Enable G-Scan) Default value
  - 0 (Disable G-Scan)
- gExtScanConcMode: G-Scan on WFD start/stop
  - 0 (Disable G-Scan) Default value
  - 1 (Use split scan)
  - 2 (Use full scan)

Limitations and Assumptions

Limitations and assumptions

- RSSI passed to the host is always a signed value.
- Reported BIT for the hot list BSSID should be cleared by the host. Currently, there is no host API for this.
- Sample size for the significant RSSI change indication is bucket interval containing that channel for Found BSSIDs.
- For lost/not found BSSIDs, it is same as the channel hint passed by the host.
- Adding the hot list BSSID into the significant RSSI BSSID is the responsibility of user app.
- Scan coalescing is not supported.
- G-Scan cached results with report event 2 are not supported.
- G-Scan is expected to support concurrency with BTC by disabling it.
- Disable G-Scan on WFD start with configuration parameter:

gExtScanConcMode:	0 (Disable G-Scan) Default
	1 (Use split scan)
	2 (Use Full scan)

References

References

DCN	Document Title
80-Y8698-3	G-scan Specifications
80-Y7938-2	Batched scan design document

Questions?

Thank You