DiagnosticsAgent Porting Notes

Table of Contents

1. Testing

1.1. Simulating Analytics events

The RCN_AnalyticsUtil command line utility can be used to send analytics events using the libAnalytics interface.

The utility has the following options for defining the event, attributes, how many events to send, etc.

```
 RCN_AnalyticsUtil - Arguments:
        -event_type <param>      - event type 0-structured event 1-unstructured event 2-large event 3-UCF
event. Eg -event_type 0 (Required)
        -category <param>        - category of the event .eg Eg -category button (Optional)
        -action <param>  - Events action .eg Eg -action clicked (Optional)
        -attr <param>    - event attributes. The attributes must be specified as a JSON map object with string
key and value types.
                Eg -attr {\"a\":\"apple\",\"b\":\"banana\",\"c\":\"car\"}
                Note double quotes must be escaped on the command line. (Optional)
        -sd_attr <param>         - sensitive data attributes with sd level.
                Eg -sd_attr [{\"k\":\"Key1\",\"v\":\"value1\",\"l\":2},
                {\"k\":\"Key2\",\"v\":\"value2\",\"l\": 2} ] (Optional)
        -sd_level <param>        - sensitive atrributes level(0-SD0 1-SD1 2-SD2 3-SD3 4-SDlocation 5-
FnvaSDNoUpload). Eg -sd_level 2 (Optional)
        -count <param>   - number of events to be sent.Eg -count 10 (Optional)
        -duration <param>        - duration of the test for structured events in seconds.Eg -duration 10
(Optional)
        -label <param>   - labels to the event.Eg -label \" a\",\"b\",\"c\" (Optional)
        -wait_time <param>       - time to wait for response in seconds. Eg -wait_time 10 (Optional)
        -description <param>     - description of the event.Eg -description Test_description (Optional)
        -start_timer <param>     - start timer for the event with string id.Eg -start_timer testTimer1
(Optional)
        -stop_timer <param>      - stop a timer that was started after a given period of time(seconds).Eg -
stop_timer 10 (Optional)
        -reset_timer <param>     - reset timer that was started earlier after a give time. Eg -reset_timer 10
(Optional)
        -security <param>        - send a SecurityLogEvent with the specified priority, instead of an
AnalyticsEvent. Eg. -security 2 to send an event with security priority FnvaSecLogPriHigh (Optional)
        -bluezone <param>        - send a BlueZoneEvent instead of an AnalyticsEvent. (Optional)
        -adc <param>     - send advance data collection event instead of an AnalyticsEvent. (Optional)
        -h <param>       - Show this message (Optional)
```

A simple example of this for testing a single event is:

```
RCN_AnalyticsUtil  -event_type 0 -category "test" -action "run" -attr "{\"a\":\"apple\",\"b\":\"banana\",\"c\":
\"car\"}"
```

This will send a single event with category "test", action "run" and three attributes. Optionally use -count to send multiple events in a loop.

Permissions Requirement:

- root user will be permitted
- users that belong to the analytics-client group (currently refers to "nobody" group on avdxp)

1.2. Watching for analytics events arriving at AnalyticsAgent.

Here is a quick way to verify that events are arriving at the AnalyticsAgent. The AnalyticsAgent logs the JSON content of all events to rotating file on disk as long as they are not marked as sensitive events. The files can be found on the ECG at:

```
/data/tmp/RCN_AnalyticsAgent/log_general/0.txt
```

The agent will begin writing events at 0.txt. Once the file reaches a certain size it will move, and create a new 0.txt. To watch for incoming events use the following command.

```
$ tail -f /data/tmp/RCN_AnalyticsAgent/log_general/0.txt
```

There are alot of events getting written to this file, so the output can be hard to follow. Pipe through grep to filter the output. Each event has a "batch" property which contains the ECU name. This a useful filter.

```
$ tail -f /data/tmp/RCN_AnalyticsAgent/log_general/0.txt | grep "AVDXP"
```

2. Configuration

DiagnosticsAgent requires a configuration file stored on the device to start. A path to a config file is expected via the -p parameter.

The config file defines paths to database files. The agent will require a writable filesystem to start. Items in the sample below that require a writable path are marked with TODO. It is recommended to create a directory specifically for diagnostics. E.g.: /path/to/writeable/diagnostics

Most of the file paths in the config file are not related to analytics, but the agent will likely not behave well if they are not defined.

```
ECU_ID:DUEROS
STANDALONE_ENABLED:1
NUM_LOG_COLLECTION_JOBS:3
LOG_CAPTURE_COMMAND:/TODO/log_capture_sync_plus.sh
DIAGNOSTICS_SLAVE_DB:TODO/db/diagnostics_slave.sqlite
UNSENT_ANALYTICS_EVENTS_DB:TODO/db/slave_unsent_analytics_events.sqlite
UNSENT_ANALYTICS_EVENTS_RAMDISK_DB:TODO/db/slave_unsent_analytics_events_ramdisk.sqlite
DIMENSIONS_DB:TODO/db/dimension_db.sqlite
DS_DB_ROOT_PATH:TODO
CONSENT_CACHE_FILE_PATH:TODO/consentCache
RWDATA_PATH:TODO
TMPFOLDER_PATH:TODO
DIAG_PERSIST_PATH:TODO/diagnostics
DIAG_SFTP_KEY_PATH:TODO/diagnostics/.ssh
DS_STORAGE_HIGH_THRESHOLD_MB:180
DS_STORAGE_HIGH_THRESHOLD_MB_DEV:360
DS_STORAGE_LOW_THRESHOLD_MB:140
DS_STORAGE_LOW_THRESHOLD_MB_DEV:300
DS_EVENT_HIGH_THRESHOLD:300
DS_EVENT_LOW_THRESHOLD:250
DS_HISTORY_HIGH_THRESHOLD:500
DS_HISTORY_LOW_THRESHOLD:400
PERSISTED_ANALYTICS_EVENT_DB_LIMIT:1000
DS_EVENT_THROTTLE_TYPE:0
DS_EVENT_THROTTLE_DURATION:14400
DS_EVENT_THROTTLE_SPAM_REQUEST_DURATION:60
DS_EVENT_THROTTLE_MAX_DUPLICATES:5
DS_EVENT_THROTTLE_MAX_CAPTURES:2
DS_OTA_CONFIG_DISABLED:0
PROXY_THROTTLE_MAX_MATCHES:60
PROXY_THROTTLE_WINDOW_DURATION_SEC:60
PROXY_THROTTLE_MAX_TRACKED_EVENTS:180
```

3. Starting the agent

The agent shall not start from a root shell. The user should be a member of its clients groups (eg analytics-client, bluezone-client) and a member of the soagateway group. For testing, use adb or nobody

Take these steps before starting the agent.

1. The agent will assume the existence of /var/run/diagnostics for IPC purposes. You will need to create this path before starting the agent. This is usually done by the startup script.

The agent can be started from the command line as follows:

```
$ RCN_DiagnosticsAgent -p /path/to/config_file
```

4. Send Diagnostics Events

TODO: - write log capture script

- have zip on target

```
$ RCN_DiagnosticsAgent  //without any parameters, send default TEST EVENT.
$ RCN_DiagnosticsAgent  -bug_report all //send a bug report
```