contents

Deprecated QMI

Compare to MDM6200/MDM9215 chips, on MDM9x07/MDM9x40/SDX20 and later chips,

Some QMI are deprecated by Qualcomm. So the customer should not use these QMI.

Instead Qualcomm develop new QMI or enhance old QMI to replace deprecated QMI.

Next table list the deprecated QMI and the new QMI.

deprecated	new
QMINAS_GET_SERVING_SYSTEM_REQ	QMINAS_GET_SYS_INFO_REQ
QMIDMS_UIM_GET_PIN_STATUS_REQ	QMIUIM_GET_CARD_STATUS_REQ
QMIDMS_UIM_VERIFY_PIN_REQ	QMIUIM_VERIFY_PIN_REQ
QMI_DMS_UIM_GET_ICCID	QMIUIM_READ_TRANSPARENT_REQ
QMI_DMS_UIM_GET_IMSI	QMIUIM_READ_TRANSPARENT_REQ

Note: this table not list all deprecated QMI, only used by quectel-CM.

Quectel's EC21/EC25/EG06/E12/EG16/EG18/EG12 do not support deprecated QMI.

libqmi for Quectel modules

This chapter list the common used qmicli commands. If one command use deprecated QMI , will tell what correct qmicli commands should to use, and support on what libqmi version.

For Quectel modules have a firmware bug, and fixed by next commit, so libqmi V1.17.0 is mini version.

```
commit 4d373376b65d77cf30f6d534fa9b059a43ec6539
Author: Dan Williams <dcbw@redhat.com>
Date:   Fri Sep 30 09:25:01 2016 -0500

    qmi-device: assume reasonable DMS version if WDS is high enough

    Some devices (Quectel EC21) lie about their supported DMS version,
    so assume a reasonable DMS version if the WDS version is high enough.
```

https://cgit.freedesktop.org/libqmi/commit/?id=4d373376b65d77cf30f6d534fa9b059a43ec6539

qmicli command	correct command	libqmi version
--dms-get-activation-state		V1.17.0
--dms-get-capabilities		V1.17.0
--dms-get-hardware-revision		V1.17.0
--dms-get-ids		V1.17.0
--dms-get-model		V1.17.0
--dms-get-msisdn		V1.17.0
--dms-get-operating-mode		V1.17.0
--dms-get-prl-version		V1.17.0
--dms-get-revision		V1.17.0
--dms-set-operating-mode		V1.17.0
--dms-uim-get-iccid	--uim-read-transparent= 0x3f00,0x2fe2	V1.17.0
--dms-uim-get-imsi	--uim-read-transparent=0x3f00,0x7fff,0x6f07	V1.17.0
--dms-uim-get-pin-status	--uim-get-card-status	V1.17.0
--dms-uim-get-state	--uim-get-card-status	V1.17.0
--nas-get-cell-location-info		V1.17.0
--nas-get-rf-band-info		V1.17.0
--nas-get-serving-system	--nas-get-system-info	V1.17.0
--nas-get-system-info		V1.17.0
--nas-get-signal-info		V1.17.0
--nas-set-system-selection-preference		V1.17.0
--wds-get-packet-service-status		V1.17.0
--wds-get-profile-list		V1.17.0
--wds-bind-mux-data-port		V1.18.0
--wds-set-ip-family		**V1.19.0**
--wds-start-network		V1.17.0
--wds-stop-network		V1.17.0
--wds-get-current-data-bearer-technology		V1.17.0
--wda-set-data-format		**V1.18.0**
--wda-get-data-format		**V1.18.0**

What is QMI?

QMI is a binary protocol designed to replace the AT command based communication with modems, and is available in devices with Qualcomm chipsets from multiple vendors (Novatel, Huawei, Sierra Wireless, ZTE… and of course Qualcomm itself).

The protocol defines different ‘services‘, each of them related to different actions that may be requested to the modem. For example, the ‘DMS’ (Device Management) service provides actions to load device information; while the ‘NAS’ (Network Access) service provides actions to register in the network. Similarly, other services will allow the user to request data connections (WDS), setup GPS location reporting (PDS), or manage internals of the user

identity module (UIM service). The user needs to handle the creation of 'clients' for those services by allocating/deallocating 'client IDs' using the generic always-on 'control' (CTL) service.
Each service in the protocol defines 'Request and Responses' as well as 'Indications'. Each pair of request/response has a maching ID which lets the user concatenate multiple requests and get out-of-order responses that can be afterwards matched through the common ID. Indications arrive as unsolicited messages, sent either to a specific client or as a broadcast message to all the clients of a given service. Usually the user needs to request the enabling of the indications to receive via some request/response.
Finally, each message in the protocol defines a set of input (in requests) and output (in responses and indications) arguments, which we name as TLVs. Some of these are defined to be mandatory, others are optional, and others are only mandatory if some other argument is available and has a given value. For example, some output arguments in response messages are only mandatory if the result of the response (given as a TLV) is SUCCESS.

Using the QMI protocol

This protocol is easily accessible in recent enough Linux kernels (>= 3.4), through the cdc-wdm and qmi_wwan drivers. Once these drivers are in place and the modem gets plugged in, the kernel will expose a new /dev/cdc-wdm device which can talk QMI, along with a wwan interface associated to each QMI port.

libqmi

libqmi is an on-going effort to get a library providing easy access to Qualcomm's 'QMI' protocol. The current libqmi available is based on the GLib/GObject/GIO-powered 'libqmi-glib'. libqmi tries to ease the use of the protocol by providing:

- A 'QmiDevice' object to control the access to the /dev/cdc-wdm device. This object allows creating new per-service 'QmiClient's. It also hides the use of the implicit 'CTL' service client.
- The 'QmiClient' object provides an interface to the set of request/responses of a given service. Users of the library will just need to execute a GIO-asynchronous method to send the request and get the processed response.
- The 'QmiClient' object also provides signals for each of the indications given in the service.
- The input and output arguments needed in requests, responses and indications are compiled into 'bundles'. These bundles are opaque structs with accessors for their elements, which allow us to add new TLVs to a given message in the future without breaking API/ABI.
- The protocol is written in the library as a JSON dictionary of rules, and most of the code to handle the protocol in the library is autogenerated during compilation. If you want to take a look at the currently available interface, check the gtk-doc documentation at: http://www.freedesktop.org/software/libqmi/libqmi-glib/latest/