QMI AUTH 1.4 for MPSS.NI.5.0.x

QMI Authentication Svc Spec

80-ND591-21 A

November 30, 2012

Submit technical questions at:

https://support.cdmatech.com

Confidential and Proprietary - Qualcomm Technologies, Inc.

Restricted Distribution. Not to be distributed to anyone who is not an employee of either Qualcomm or its subsidiaries without the express approval of Qualcomm's Configuration Management.

Not to be used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm Technologies, Inc.

Qualcomm reserves the right to make changes to the product(s) or information contained herein without notice. No liability is assumed for any damages arising directly or indirectly by their use or application. The information provided in this document is provided on an "as is" basis.

This document contains confidential and proprietary information and must be shredded when discarded.

Qualcomm and MSM are trademarks of QUALCOMM Incorporated, registered in the United States and other countries. All QUALCOMM Incorporated trademarks are used with permission. Other product and brand names may be trademarks or registered trademarks of their respective owners.

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Contents

List of Figures

List of Tables

Revision History

Revision	Date	Description
A	Nov 2012	Initial release. Created from 80-VB816-21 D. Updates for this revision include minor version 3 and minor version 4. Updated Section 2.3.1. Updated the EAP Method Mask TLV (Section 3.2.1) Updated the description of the EAP SIM AKA Algorithm TLV (Section 3.2.1) Updated the description of QMI_AUTH_START_EAP_SESSION REQ/RESP (Section 3.2.3)

1 Introduction

1.1 Purpose

This specification documents Major Version 1 of the Qualcomm Messaging Interface (QMI) for the Authentication Service (QMI_AUTH).

QMI_AUTH provides a command set to interface to a wireless mobile station to access some authentication services. QMI_AUTH is a QMI service within the QMI framework defined in [Q1].

1.2 Scope

This document is intended for QMI clients to perform authentication-related operations with Qualcomm MSM® devices from a host processor.

This document provides the following details about QMI_AUTH:

- Theory of operation – Chapter 2 provides the theory of operation of QMI_AUTH. The chapter includes messaging conventions, assigned QMI service type, fundamental service concepts, and state variables related to the service.
- Message formats, syntax, and semantics – Chapter 3 provides the specific syntax and semantics of messages included in this version of the QMI_AUTH specification.

1.3 Conventions

Function declarations, function names, type declarations, and code samples appear in a different font. For example, `#include`.

Parameter types are indicated by arrows:

→ Designates an input parameter

← Designates an output parameter

↔ Designates a parameter used for both input and output

1.4 References

Reference documents are listed in Table 1-1. Reference documents that are no longer applicable are deleted from this table; therefore, reference numbers may not be sequential.

Table 1-1 Reference documents and standards

Ref.	Document	
Qualcomm Technologies		
Q1	*Application Note: Software Glossary for Customers*	CL93-V3077-1
Q2	*Qualcomm MSM Interface (QMI) Architecture*	80-VB816-1
Standards		
S1	*Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)*	RFC 4187
S2	*Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)*	RFC 4186

1.5 Technical Assistance

For assistance or clarification on information in this document, submit a case to Qualcomm Technologies at https://support.cdmatech.com.

If you do not have access to the CDMATech Support website, register for access or send email to support.cdmatech@qti.qualcomm.com.

1.6 Acronyms

For definitions of terms and abbreviations, refer to [Q1]. Table 1-2 lists terms that are specific to this document.

Table 1-2 Acronyms

Acronym	Definition
AKA	Authentication and Key Agreement
EAP	Extensible Authentication Protocol
GSM	global system for mobile communications
QMI	Qualcomm messaging interface
R-UIM	removable user identity module
SIM	subscriber identity module
TLV	type-length-value
UMTS	universal mobile telecommunications system
USIM	UMTS SIM

2 Theory of Operation

2.1 Generalized QMI Service Compliance

The QMI_AUTH service complies with the generalized QMI service specification, including the rules for messages, indications and responses, byte ordering, arbitration, constants, result, and error code values described in [Q2]. Extensions to the generalized QMI service theory of operation are noted in subsequent sections of this chapter.

2.2 AUTH Service Type

AUTH is assigned QMI service type 0x07.

2.3 Message Definition Template

2.3.1 Response Message Result TLV

This Type-Length-Value (TLV) is present in all Response messages defined in this document. It is not present in the Indication messages.

Name	Version introduced	Version last modified
Result Code	Corresponding command's *Version introduced*	N/A

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x02			1	Result Code
Length	4			2	
Value	→	uint16	qmi_result	2	Result code • QMI_RESULT_SUCCESS • QMI_RESULT_FAILURE
		uint16	qmi_error	2	Error code – Possible error code values are described in the error codes section of each message definition

2.4 QMI_AUTH Fundamental Concepts

The QMI_AUTH service provides authentication and session key distribution using the Extensible Authentication Protocol (EAP) mechanism.

2.4.1 EAP-AKA

EAP is a mechanism for authentication and session key distribution that uses the Authentication and Key Agreement (AKA) mechanism. AKA is used in the third generation mobile networks Universal Mobile Telecommunications System (UMTS) and cdma2000®. AKA is based on symmetric keys and typically runs in a Subscriber Identity Module (SIM), which is a UMTS Subscriber Identity Module (USIM) or a Removable User Identity Module (R-UIM), similar to a smart card.

2.4.2 EAP-SIM

EAP is also a mechanism for authentication and session key distribution using the Global System for Mobile Communications (GSM) SIM. GSM is a second generation mobile network standard. The EAP-SIM mechanism specifies enhancements to GSM authentication and key agreement whereby multiple authentication triplets can be combined to create authentication responses and session keys of greater strength than the individual GSM triplets. The mechanism also includes network authentication, user anonymity support, result indications, and a fast re-authentication procedure.

2.4.3 EAP Support

QMI_AUTH service enables clients to use the wireless mobile station for EAP authentication. Figure 2-1 illustrates a sample QMI_AUTH call flow.

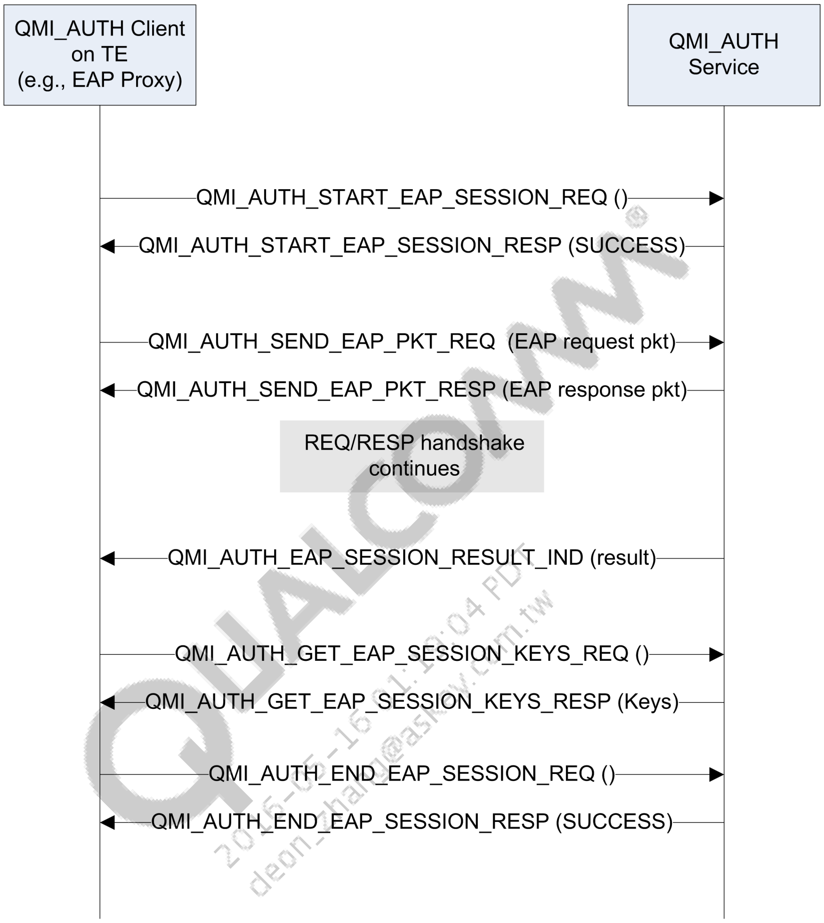
QMI_AUTH Client on TE (e.g., EAP Proxy)
QMI_AUTH Service
QMI_AUTH_START_EAP_SESSION_REQ ()
QMI_AUTH_START_EAP_SESSION_RESP (SUCCESS)
QMI_AUTH_SEND_EAP_PKT_REQ (EAP request pkt)
QMI_AUTH_SEND_EAP_PKT_RESP (EAP response pkt)
REQ/RESP handshake continues
QMI_AUTH_EAP_SESSION_RESULT_IND (result)
QMI_AUTH_GET_EAP_SESSION_KEYS_REQ ()
QMI_AUTH_GET_EAP_SESSION_KEYS_RESP (Keys)
QMI_AUTH_END_EAP_SESSION_REQ ()
QMI_AUTH_END_EAP_SESSION_RESP (SUCCESS)

Figure 2-1 QMI AUTH sample call flow

2.5 EAP Session Handle

The QMI_AUTH_START_EAP_SESSION_REQ message creates an EAP instance and stores the EAP handle. The same handle is used internally when the QMI_AUTH_SEND_EAP_PACKET command is issued. The EAP handle is deleted when the QMI_AUTH_END_EAP_SESSION command is issued.

3 QMI_AUTH Messages

Table 3-1 QMI_AUTH messages

Command	ID	Description
QMI_AUTH_RESET	0x0000	Resets the client.
QMI_AUTH_START_EAP_SESSION	0x0020	Starts the EAP session.
QMI_AUTH_SEND_EAP_PACKET	0x0021	Sends and receives EAP packets.
QMI_AUTH_EAP_SESSION_RESULT_IND	0x0022	Communicates the result of the EAP session.
QMI_AUTH_GET_EAP_SESSION_KEYS	0x0023	Queries the EAP session keys.
QMI_AUTH_END_EAP_SESSION	0x0024	Ends the EAP session.
QMI_AUTH_RUN_AKA_ALGO	0x0025	Runs the AKA algorithm.
QMI_AUTH_AKA_ALGO_RESULT_IND	0x0026	Communicates the result of the AKA algorithm.

3.1 QMI_AUTH_RESET

Resets the client.

AUTH message ID

0x0000

Version introduced

Major - 1, Minor - 0

3.1.1 Request - QMI_AUTH_RESET_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

None

3.1.2 Response - QMI_AUTH_RESET_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.1) is always present in the response.

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request

3.1.3 Description of QMI_AUTH_RESET REQ/RESP

This command resets the state of the requesting control point. The command clears all the resources that were set up for the EAP session started by the control point.

3.2 QMI_AUTH_START_EAP_SESSION

Starts the EAP session.

AUTH message ID

0x0020

Version introduced

Major - 1, Minor - 0

3.2.1 Request - QMI_AUTH_START_EAP_SESSION_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

Name	Version introduced	Version last modified
EAP Method Mask	1.0	1.4
User ID	1.2	1.2
EAP Meta Identity	1.2	1.2
EAP SIM AKA Algorithm	1.2	1.2

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	EAP Method Mask
Length	4			2	
Value	→	mask32	eap_method_mask	4	Bitmask. The bits corresponding to the methods to be supported must be set to 1. Bit values: • 0 – EAP-SIM • 1 – EAP-AKA • 2 – EAP-AKA′
Type	0x11			1	User ID
Length	Var			2	
Value	→	uint8	user_id_len	1	Number of sets of the following elements: • user_id
		uint8	user_id	Var	Buffer containing the EAP identity.

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x12			1	EAP Meta Identity
Length	Var			2	
Value	→	uint8	eap_meta_identity_len	1	Number of sets of the following elements: • eap_meta_identity
		uint8	eap_meta_identity	Var	Buffer containing the EAP meta identity.
Type	0x13			1	EAP SIM AKA Algorithm
Length	4			2	
Value	→	enum	eap_sim_aka_algo	4	EAP AKA algorithm. Values: • 0x0000 – EAP AKA none • 0x0001 – EAP AKA SHA-1 • 0x0002 – EAP AKA MILENAGE • 0x0003 – EAP AKA CAVE • 0x0004 – EAP SIM GSM • 0x0005 – EAP SIM USIM GSM If only the eap_sim_aka_algo parameter is specified, the EAP method type associated with the current EAP session is set as follows: • 0x0001, 0x0003 – EAP method is QMI_AUTH_EAP_METHOD_MASK_AKA • 0x0004, 0x0005 – EAP method is QMI_AUTH_EAP_METHOD_MASK_SIM • 0x0002 – EAP method is set to AKA|AKA′. Depending on which packets are sent, the EAP module uses either AKA or AKA′ at a later time • 0x0000 – All EAP methods are supported

3.2.2 Response - QMI_AUTH_START_EAP_SESSION_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.1) is always present in the response.

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_INVALID_ARG	Value field of one or more TLVs in the request message contains an invalid value
QMI_ERR_INVALID_OPERATION	Operation is invalid in the current state

3.2.3 Description of QMI_AUTH_START_EAP_SESSION REQ/RESP

This command starts an Extensible Authentication Protocol (EAP) session, after which control points can send EAP packets using the QMI_AUTH_SEND_EAP_PACKET command. The optional EAP Method Mask TLV can be used to set the EAP authentication method to either SIM (Subscriber Identity Module) or AKA (Authentication and Key Agreement).

This command creates the required handle with the required authentication method to allow control points to send the EAP packets using the QMI_AUTH_SEND_EAP_PACKET_REQ message.

If both the EAP Method Mask and EAP SIM AKA Algorithm TLVs are provided, the values must be consistent (e.g., an eap_method_mask value of 0x01 with an eap_sim_aka_algo value of 0x0001 results in QMI_ERR_INVALID_ARG. Also, an eap_method_mask value of 0x04 is to have an eap_sim_aka_algo value of 0x0002 or 0x0000; otherwise, the value results in QMI_ERR_INVALID_ARG).

All EAP methods are supported if neither eap_method_mask nor eap_sim_aka_algo is provided.

3.3 QMI_AUTH_SEND_EAP_PACKET

Sends and receives EAP packets.

AUTH message ID

0x0021

Version introduced

Major - 1, Minor - 0

3.3.1 Request - QMI_AUTH_SEND_EAP_PACKET_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
EAP Request Packet	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	EAP Request Packet
Length	Var			2	
Value	→	uint8	eap_request_pkt	Var	Buffer containing the EAP request packet.

Optional TLVs

None

3.3.2 Response - QMI_AUTH_SEND_EAP_PACKET_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.1) is always present in the response. The following mandatory TLV is present if the result code is QMI_RESULT_SUCCESS.

Name	Version introduced	Version last modified
EAP Response Packet	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	EAP Response Packet
Length	Var			2	
Value	→	uint8	eap_response_pkt	Var	Buffer containing the EAP response packet.

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_MISSING_ARG	TLV was missing in the request
QMI_ERR_INVALID_OPERATION	Operation is invalid in the current state

3.3.3 Description of QMI_AUTH_SEND_EAP_PACKET REQ/RESP

This command is used by a control point to send an EAP packet after an EAP session is started. The response EAP packet for the request packet is returned in the QMI_AUTH_SEND_EAP_PACKET_RESP message. The EAP packet details are found in [S1] and [S2].

3.4 QMI_AUTH_EAP_SESSION_RESULT_IND

Communicates the result of the EAP session.

AUTH message ID

0x0022

Version introduced

Major - 1, Minor - 0

3.4.1 Indication - QMI_AUTH_SESSION_RESULT_IND

Message type

Indication

Sender

Service

Indication scope

Unicast

Mandatory TLVs

Name	Version introduced	Version last modified
EAP Result	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	EAP Result
Length	1			2	
Value	→	boolean	eap_result	1	Values: • 0 – SUCCESS • 1 – FAILURE

Optional TLVs

None

3.4.2 Description of QMI_AUTH_EAP_SESSION_RESULT_IND

This indication communicates the result of the current EAP session to the control point that started the EAP session.

If the result is SUCCESS, the keys are available and the client must use the QMI_AUTH_GET_EAP_SESSION_KEYS command to query the keys.

The client can later end the current EAP session using the QMI_AUTH_END_EAP_SESSION command.

3.5 QMI_AUTH_GET_EAP_SESSION_KEYS

Queries the EAP session keys.

AUTH message ID

0x0023

Version introduced

Major - 1, Minor - 0

3.5.1 Request - QMI_AUTH_GET_EAP_SESSION_KEYS_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

None

3.5.2 Response - QMI_AUTH_GET_EAP_SESSION_KEYS_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.1) is always present in the response. The following mandatory TLV is present if the result code is QMI_RESULT_SUCCESS.

Name	Version introduced	Version last modified
Key	1.0	1.0

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	Key
Length	Var			2	
Value	→	uint8	session_key	Var	Session key.

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_INVALID_OPERATION	Operation is invalid in the current state

3.5.3 Description of QMI_AUTH_GET_EAP_SESSION_KEYS REQ/RESP

To extract the session keys:

1. The control point issues the QMI_AUTH_START_EAP_SESSION_REQ message.
2. If the start EAP session is successful, a QMI_AUTH_EAP_SESSION_RESULT_IND indication is sent to the control point to communicate that the session keys are available.
3. The control point uses the QMI_AUTH_GET_EAP_SESSION_KEYS_REQ message to retrieve the session keys.

3.6 QMI_AUTH_END_EAP_SESSION

Ends the EAP session.

AUTH message ID

0x0024

Version introduced

Major - 1, Minor - 0

3.6.1 Request - QMI_AUTH_END_EAP_SESSION_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

None

Optional TLVs

None

3.6.2 Response - QMI_AUTH_END_EAP_SESSION_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.1) is always present in the response.

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_INVALID_OPERATION	Operation is invalid in the current state

3.6.3 Description of QMI_AUTH_END_EAP_SESSION REQ/RESP

This command is used by a control point to end an EAP session that it started. The EAP session must be ended after the authentication process.

3.7 QMI_AUTH_RUN_AKA_ALGO

Runs the AKA algorithm.

AUTH message ID

0x0025

Version introduced

Major - 1, Minor - 1

3.7.1 Request - QMI_AUTH_RUN_AKA_ALGO_REQ

Message type

Request

Sender

Control point

Mandatory TLVs

Name	Version introduced	Version last modified
AKA Version	1.1	1.1

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	AKA Version
Length	1			2	
Value	→	enum8	aka_ver	1	AKA version the algorithm must use: • 0 – AKA_V1 • 1 – AKA_V2 All other values are reserved for future use.

Optional TLVs

Name	Version introduced	Version last modified
AKA_V1/V2 Authentication Parameters	1.1	1.1

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	AKA_V1/V2 Authentication Parameters
Length	Var			2	

Field	Field value	Field type	Parameter	Size (byte)	Description
Value	→	uint8	rand_len	1	Number of sets of the following elements: • rand
		uint8	rand	Var	Buffer containing the random challenge value.
		uint8	autn_len	1	Number of sets of the following elements: • autn
		uint8	autn	Var	Buffer containing the authentication token.

3.7.2 Response - QMI_AUTH_RUN_AKA_ALGO_RESP

Message type

Response

Sender

Service

Mandatory TLVs

The Result Code TLV (defined in Section 2.3.1) is always present in the response. The following mandatory TLV is present if the result code is QMI_RESULT_SUCCESS.

Name	Version introduced	Version last modified
AKA Handle	1.1	1.1

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	AKA Handle
Length	4			2	
Value	→	uint32	aka_handle	4	AKA handle to identify the AKA request.

Optional TLVs

None

Error codes

QMI_ERR_NONE	No error in the request
QMI_ERR_INTERNAL	Unexpected error occurred during processing
QMI_ERR_MALFORMED_MSG	Message was not formulated correctly by the control point, or the message was corrupted during transmission
QMI_ERR_NO_MEMORY	Device could not allocate memory to formulate a response
QMI_ERR_INVALID_ARG	Value field of one or more TLVs in the request message contains an invalid value
QMI_ERR_MISSING_ARG	TLV was missing in the request

3.7.3 Description of QMI_AUTH_RUN_AKA_ALGO REQ/RESP

The control point uses this command to initiate the AKA algorithm (refer to [S1] and [S2]) to generate the digest and AKA data.

When AKA_V1 or AKA_V2 is specified in the AKA Version TLV, the optional AKA_V1/V2 Authentication Parameters TLV must be present.

A success in the QMI_AUTH_RUN_AKA_ALGO_RESP message does not imply the algorithm completed successfully. The control point must process the QMI_AUTH_AKA_RESULT_IND indication to determine the outcome.

3.8 QMI_AUTH_AKA_ALGO_RESULT_IND

Communicates the result of the AKA algorithm.

AUTH message ID

0x0026

Version introduced

Major - 1, Minor - 1

3.8.1 Indication - QMI_AUTH_AKA_ALGO_RESULT_IND

Message type

Indication

Sender

Service

Indication scope

Unicast

Mandatory TLVs

Name	Version introduced	Version last modified
AKA Algorithm Result Indication	1.1	1.1

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x01			1	AKA Algorithm Result Indication
Length	5			2	
Value	→	uint32	aka_handle	4	AKA handle to identify the AKA request.
		enum8	aka_status	1	Result of the AKA Request algorithm: • 0 – AKA_SUCCESS • 1 – AKA_SYNC_FAILURE • 2 – AKA_FAILURE All other values are reserved for future use.

Optional TLVs

The following TLV is present only if the mandatory status parameter is returned as AKA_SUCCESS.

Name	Version introduced	Version last modified
AKA_V1/V2 Response Data	1.1	1.1

Field	Field value	Field type	Parameter	Size (byte)	Description
Type	0x10			1	AKA_V1/V2 Response Data
Length	Var			2	
Value	→	uint8	digest_len	1	Number of sets of the following elements: • digest
		uint8	digest	Var	Buffer containing the digest response.
		uint8	aka_data_len	1	Number of sets of the following elements: • aka_data
		uint8	aka_data	Var	Buffer containing the AKA response data.

3.8.2 Description of QMI_AUTH_AKA_ALGO_RESULT_IND

This indication communicates the result of the AKA algorithm request sent as part of the QMI_AUTH_RUN_AKA_ALGO_REQ message. If the result is AKA_SUCCESS, the message also contains the optional AKA_V1/V2 Response Data TLV.