Open Mobile API Test Specification

Version 1.0 - Errata Note

Published by simalliance now Trusted Connectivity Alliance

July 2014

Table of Contents

1. Introduction

During the review of the SIMalliance Test Specification v1.0 by the Global Certification Forum and GlobalPlatform, several points were raised which required either an update of the test cases or additional clarification on the expected behaviour from the devices or from the test tools implementing the test specification.

This document summarises all the changes; the changes will also be integrated into the next SIMalliance Test Specification document. The document shows revision marks to indicate the new applicable content (and only the new content).

2. List of modifications

2.1 Chapter 1.2. Terms

Additional definition for the meaning of specific terms used in the description of test cases.

No APDU	When "No APDU" is mentioned in a test case, it means that the device shall not send any command to the Secure Element while processing the called API method. For a test tool, it means that no APDU (except STATUS command generated by the device telecom stack in case of UICC) shall be sent on the device-SE interface on any logical channel.
none	When "none" is mentioned in a test case, it means that it is not relevant if APDUs are sent.
No selection	No select by DF name command is sent.

2.2 Chapter 2. Informative References

Fixing of a typo error in the SIMalliance Open Mobile API Specification reference.

Specification	Description
[1] OMAPI V2.05	SIMalliance Open Mobile API specification V2.05

2.3 Chapter 4.2. Table of DUT Options

Many devices do not allowing the use of a NULL AID (implicit selection) when opening a logical channel on the UICC. This mechanism is intended to block the selection of Telecom NAA over a logical channel as a side effect. It also reinforces the protection brought by GP SE AC as the mobile application will have to specify the AID of the applet targeted and to have the access granted by a rule.

A new option "OP_010, Access to the default applet is blocked by the DUT" was created to reflect this behaviour which is acceptable from a device point of view.

Two new options were also introduced to declare if the device is managing the maximum number of logical channels internally (using the information in the SE's ATR) or not. These options are used to split the previous test case openLogicalChannel ID5 in three separated test cases.

Table 4: DUT Options

Item	Option	Status	Optional item
10	Access to the default applet is allowed by the DUT	OP	OP-010
11	RFU	OP	OP-011
12	DUT knows when all SE logical channels are already opened	OP	OP-012
13	DUT relies on SE to know if all logical channels are already opened and check access control rules on openSession	OP	OP-013
14	DUT relies on SE to know if all logical channels are already opened and check access control rules on openLogicalChannel	OP	OP-014

2.4 Chapter 4.3. Applicability Table

Updates to the applicability table reflect the new option above and also some previous options not correctly set according to a device's capabilities.

Clause	Test case number and description	SUE	RSE		
			UICC	eSE	mSD
6.4.6	Method: Channel openBasicChannel ID10	OP-003	OP-003	N/A	N/A
6.4.7	Method: Channel openLogicalChannel ID1, ID2, ID6, ID7,ID09 – ID17	M	M	M	M
6.4.7	Method: Channel openLogicalChannel ID3 – ID4	OP-010	OP010	M	M
6.4.7	Method: Channel openLogicalChannel ID5a	OP-012	OP-012	OP-012	OP-012
6.4.7	Method: Channel openLogicalChannel ID5b	OP-013	OP-013	OP-013	OP-013
6.4.7	Method: Channel openLogicalChannel ID5c	OP-014	OP-014	OP-014	OP-014
6.5.1	Method: void close() ID2	OP-003	OP-003	OP-003	OP-003
6.5.4	Method: byte[] getSelectResponse() ID1,2,4,5,7,8	OP-008	OP-008	OP-008	OP-008
6.5.4	Method: byte[] getSelectResponse() ID 3	OP-008 and OP-010	OP-008 and OP-010	OP-008	OP-008
6.5.6	Method: byte[] transmit(byte[] command) ID2 – ID7; ID9 – ID11,ID15 - ID20	M	M	M	M
6.5.6	Method: byte[] transmit(byte[] command) ID13, 21	OP-006	OP-006	NA	NA
6.5.7	Method: Boolean[] selectNext()ID1 –ID2, ID7	M	M	M	M
6.5.7	Method: Boolean[] selectNext() ID3-ID4, ID8-ID9	OP-008	OP-008	OP-008	OP-008
6.5.7	Method: Boolean[] selectNext() ID5, ID6	M	NA	NA	NA

Table 6: Applicability of Tests

Note: For devices preventing the use of basic channel, the test case close() ID 2 is also not applicable.

Note: test transmit() ID 21 is based on a sequence involving SW = 0x61xx returned by the UICC and then a GET RESPONSE from low level layers of the device so this test is only applicable for T=0 protocol.

2.5 Chapter 5.2.3 Test Controller & Chapter 5.3.3 Test Procedure

Update of the requirements associated to the test controller and to the test procedure to better fit with test strategy. The test specification is protocol agnostic and test cases shall, as much as possible, remain independent of execution result from the UICC/UICC Simulator.

5.2.3 Test controller

The following requirements shall be provided by the test controller:

- the APDU exchange must be made visible by the test tool when they are available. For example in the case of UICC, or UICC simulator.
- the API commands must be made visible by the test tool.
- shall provide the test setup prior to the execution of the test, i.e. install the related application on the mobile and do any further actions required to run the test.
- shall provide results of the tests
- shall check that the correct C-APDU is sent by the terminal on the interface with the Secure Element/ UICC / UICC Simulator (as specified in the ISO Command Expectation column).
- shall check that the correct R-APDU is received by the mobile application as the return value to the transmit() method (as specified in the API Expectation column).
- may check the R-APDU sent on the Secure Element/ UICC / UICC Simulator interface.
- should be able to automatically execute the tests

5.3.3 Test procedure

Each test procedure contains a table of a number of test cases, each of these tests are specified as follows:

Test case					
ID	**API Description**	**ISO Command Expectation DUT → UICC Simulator / SE**	**ISO Response UICC Simulator / SE → DUT**	**API Expectation**	**CRR**
The ID of the test case.	The name of the OMAPI method called by the test application.	The expected ISO command (C-APDU) received by the UICC Simulator / SE. It is sent by the DUT to UICC Simulator / SE as a result of the OMAPI method call.	The ISO response (R-APDU) sent by UICC Simulator / SE to the DUT as a response to the received ISO command.	The expected result of the OMAPI method called. E.g.: 'true' is returned.	The list of the Conformity Requirements which is the scope of the test case.

General notes regarding the ISO Command Expectation and ISO Response columns:

- Test cases test the implementation of the SIMalliance Open Mobile API implementation and not the behaviour of the Secure Elements. However to make sure the API is correctly implemented by the device, test cases verify command exchanges between the device and the SE/UICC Simulator as well as data and the result provided by API methods.
- The ISO Command Expectation is checked to validate if the OMAPI implementation sends the expected commands to the SE/UICC Simulator.
- The ISO Response is provided for information on the UICC Simulator /SE behaviour.
- The test procedure description contains APDUs. The TPDUs are not in the scope of the test specification so they are not listed in the test procedure descriptions.
- The APDUs exchanged during the access control procedure are out of scope of the test procedure description and shall be not considered as ISO command expectation or ISO response.
- Except for specific test cases aimed at checking the correct behaviour of the underlying transport protocol, all test cases are protocol agnostic.

Meaning of "No APDU", "none" and "no selection" is defined under terms.

2.6 Chapter 5.7.1 Test APDU Interface

In the case where an openLogicalChannel command is called, the test plan expects (nominal case) a MANAGE CHANNEL command followed by a SELECT command. In the check of the SELECT command, the test plan expects a Le present and set to 00. However, some devices do not send any Le and it might be due to the fact that they do not support the option "The selection response can be retrieved by the reader implementation" or the device can manage a GET RESPONSE based on the SW return by the card.
.
Currently, a device not sending the Le = 00 fails many tests while it can successfully manage the selection of the applet and – if the option is supported – the availability of the getSelectResponse. Le was therefore made optional.

APDU_SELECT_BY_FID	0x	A4	00	00	02	3F00	00

							or empty
APDU_SELECT_BY_DF	0x	A4	04	00	XX	'AID'	00 or empty

2.7 Management of Exceptions

During the GlobalPlatform TestFest, several test cases were failing across the different devices due to differences between exceptions raised and those expected in the test specification.

Several devices send a "NullPointerError" instead of an "IllegalParameterError" when a Null pointer is used. This exception is widely used by developers and is appropriate in such cases so it was authorised. The core specification will also be updated to reflect this change.

Some test cases were also not in line with the core specification, expecting an IllegalParameterError instead of a SecurityError. The test cases were updated.

The following chapters were modified according to the above points:

2.7.1 Chapter 6.1.1 Constructor: SEService(Context context, SEService.CallBack listener)

Parameter errors
CRP1: IllegalParameterError or NullPointerError – if the parameter "context" is null.

3	**SEService Constructor with missing Context**				
	Constructor: SEService(null, listener)	none	none	`IllegalParameterError or NullPointerError expected`	CRP1
5	**SEService Constructor without any parameters**				
	Constructor: SEService(null, null)	none	none	`IllegalParameterError or NullPointerError expected`	CRP1

2.7.2 Chapter 6.5.6 Method: byte[] transmit(byte[] command)

Parameter errors
CRP1: IllegalParameterError or NullPointerError – if the parameter "context" is null.

5	**Null parameter command**				
	1. Channel.transmit(null);	CMD 1-1: No APDU	RESP 1-1: None	`1.IllegalParameterError or NullPointerError`	CRP1

6	**MANAGE CHANNEL_OPEN as parameter command**				
	1. Channel.transmit(APDU_MANAGE_CH_OPEN);	CMD 1-1: No APDU	RESP 1-1: None	1. SecurityError	CRP2
7	**SELECT BY DF NAME as parameter command**				
	1. Channel.transmit(APDU_SELECT_BY_DF(AID_TestApp));	CMD 1-1: No APDU	RESP 1-1: None	1. SecurityError	CRP3
19	**MANAGE CHANNEL CLOSE as parameter command**				
	1. Channel.transmit(APDU_MANAGE_CH_CLOSE);	CMD 1-1: No APDU	RESP 1-1: None	1. SecurityError	CRP2

2.8 Value of Class Byte in Several Test Cases

In the column 'ISO Command Expectation DUT UICC Simulator/SE', in many test cases, the CLA is specified in the APDU command sent and must be checked. However, the device may open a channel on some NAA or other applications (e.g. PKCS#15 or ARA) and the value of this class byte may be different than the one specified due to internal device management of the logical channels/applications.

CLA byte value was therefore updated and the exact value was replaced by a generic “XX”.

The following test cases were updated:

- Chapter 6.1.4: method shutdown ID3
- Chapter 6.4.7: method openLogicalChannel ID2, 4, 14, 15, 16, 17
- Chapter 6.5.1: method close ID4
- Chapter 6.5.6: method transmit ID2, 4, 8, 12, 13, 14, 15, 16, 17, 18

2.9 New Expected Result “No APDU”

Several test cases were updated to change an expected result “none” into a “No APDU”. The objective is to clarify the expectation on the device behaviour in such cases.

The following test cases were updated:

- Chapter 6.4.5: method close ID2
- Chapter 6.4.6: method openBasicChannel ID4, 5, 6, 7, 8, 9, 12
- Chapter 6.4.7: method openLogicalChannel ID6, 7, 11
- Chapter 6.5.1: method close ID2, 3
- Chapter 6.5.6: method transmit ID5, 6, 7, 9, 10, 19
- Chapter 6.5.7: method selectNext ID7

2.10 Chapter 6.4.6 openBasicChannel

Clarifications of the initial conditions for several test cases:
Test case ID 3, ID4: AID_TestApp is installed as the default selected applet.
Test case ID 7: DUT should not be connected to Telecom network to avoid unexpected APDU commands. DUT may still send STATUS command.

2.11 Chapter 6.4.6 openBasicChannel – ID 3 & 13

The test specification was updated to clarify what is expected in terms of test tool checking when a test case specifies “No APDU”, “none” or “no selection”. Several test cases were also updated to better reflect what exactly needs to be checked in a given test case, taking into account the Access Control mechanism which can trigger the sending of several different APDU commands, depending on what the device is supporting (ARA and/or ARF). It is the case for the two tests below where the “none” check was replaced with “no selection” check.

3	**Open a basic channel with the default SE applet**				
	1. session.openBasicChannel (null);	no selection <on basic channel>	None	1. Returned Channel object is not null. No exception is expected.	CRN2

13	**The application opening the basic channel has no access to the selected SE applet**				
	session.openBasicChannel (AID_accessdenied);	no selection <for AID_accessdenied>	None.	SecurityError is expected.	CRC4

2.12 Chapter 6.4.6 openBasicChannel – ID 4

Depending on the Secure Element, it may not be possible to make a test applet the “default” application. This is, for example, the case for a UICC where the Telecom NAA is always the default application.

In such cases, the default application (e.g. the USIM) does not accept the test APDU sent by the test application and the test fails while the device may behave correctly. Additional expected results were added to match with these cases where UICC may answer with SW1 SW2=0x6D 00 or SW1 SW2=0x6E 00.

In addition the test case was also updated to better specify the expected checks on the tool side, using a check based on “no selection”.

4	**Open a basic channel with the default SE applet and check, if the applet answers**				
	1. session.openBasicChannel (null);	No APDU	None	1. Returned Channel object is not null. No exception is expected.	CRN2 CRN3
	2. channel.transmit(Test_APDU1);	CMD 2: C-APDU (’00 10 01 00 04 01 02 03 04 00’)	RESP 2: R-APDU - Data = ’01 02 03 04’; SW ‘90 00’(if the test applet can be made as the default SE Applet) Or SW ‘6D 00’ or SW ‘6E 00’ (if the default applet is different from the test applet, e.g. USIM on a UICC SE, etc.)	2. Returned Response equals to ‘R-APDU’ - Data = ‘01 02 03 04’; SW ‘90 00’. Or SW ‘6D 00’ or SW ‘6E 00’ No exception is expected.	
	3. session.openBasicChannel (null);	no selection <on basic channel>	None	3 Returned Channel object is null. No exception is	

				expected.	

2.13 Chapter 6.4.7 openLogicalChannel

Clarifications of the initial conditions for several test cases:
Test case ID 3 and ID 4: AID_TestApp is installed as the default selected applet.

2.14 Chapter 6.4.7 openLogicalChannel, ID 4

The same applies as for the test on openBasicChannel ID 4:

4	**Open a logical channel with the default SE applet and check, if the applet answers**				
	1. session.openLogicalChannel (null);	CMD 1: APDU_MANAGE_CH_OPEN	RESP 1: R-APDU - Data: Channel Number; SW '90 00'	1. Returned Channel object is not null. No exception is expected.	CRN2
	2. channel.transmit(Test_APDU1);	CMD 2: C-APDU ('XX 10 01 00 04 01 02 03 04 00')	RESP 2: R-APDU - Data = '01 02 03 04'; SW '90 00' (if the test applet can be made as the default SE Applet) Or SW '6D 00' or SW '6E 00' (if the default applet is different from the test applet, e.g. USIM on a UICC SE, etc.)	2. Returned Response equals to 'R-APDU' - Data = '01 02 03 04 '; SW '90 00'. Or SW '6D 00' or SW '6E 00' No exception is expected.	

2.15 Chapter 6.4.7 openLogicalChannel, ID5

While opening all the logical Channels, the test expects an error Status Word returned by MANAGE_CHANNEL (meaning the error is managed by the card). However the DUT may already know the limitation in terms of the number of channels and then DUT may manage the error itself without having the card performing a MANAGE_CHANNEL. Furthermore, if the device is trying to check Access Control rules (e.g. Refresh Tag) on the openLogicalChannel method itself, access to the access control mechanism over a logical channel will not be possible (as all channels are used) and will lead to a security error "Access Denied" coming from the access control enforcer. All behaviours are acceptable.

In addition, the SW=0x6A 81 was also added in case there is no more channels available on the SE side.

5a	**Open a logical channel, when no new logical channel is available, device manages maximum number of logical channels**				
	1. session.openLogicalChannel (AID_TestApp);	~~CMD 1: APDU_MANAGE_CH_OPEN~~ ~~or~~ No APDU command send in case the device knows and handles the maximum number of available logical channels	~~RESP 1: R-APDU – SW '68 81'~~ ~~or~~ none	1. Returned Channel object is null. No exception is expected.	CRN4

5b	Open a logical channel, when no new logical channel is available, device does not manage maximum number of logical channels and access control Refresh Tag is checked on openSession method			
	1. session.openLogicalChannel (AID_TestApp);	CMD 1: APDU_MANAGE_CH_OPEN	RESP 1: R-APDU – SW '68 81' or '6A 81'	1. Returned Channel object is null. No exception is expected.	CRN4
5c	**Open a logical channel, when no new logical channel is available, device does not manage maximum number of logical channels and access control Refresh Tag is checked on openLogicalChannel method**				
	1. session.openLogicalChannel (AID_TestApp);	no selection <for AID_TestApp>	None	SecurityError is expected.	CRN4

2.16 Chapter 6.4.7 openLogicalChannel, ID10

Depending on the UICC and test applet, the error code sent back on the test case below can be SW=0x6985. In this case, the device may behave correctly but the test may fail due to a different error code sent by the UICC. The test has therefore been updated to add this error code as a valid result.

10	**A logical channel is already open to the non-multiselectable SE Applet**				
	1. session.openLogicalChannel (AID_TestApp);	CMD 1-1: APDU_MANAGE_CH_OPEN CMD 1-2: APDU_SELECT_BY_DF – CLA contains the Channel Number returned by the card in RESP 1-1;; Data = 'AID_TestApp'	RESP 1-1: R-APDU - Data: Channel Number; SW '90 00' RESP 1-2: R-APDU - SW '90 00'	1. No exception is expected.	CRC2
	2. session.openLogicalChannel (AID_TestApp);	CMD 2-1: APDU_MANAGE_CH_OPEN CMD 2-2: APDU_SELECT_BY_DF – CLA contains the Channel Number returned by the card in RESP 2-1; Data = 'AID_TestApp CMD 2-3 MANAGE CHANNEL (P1='80')	RESP 2-1: R-APDU - Data: Channel Number; SW '90 00' RESP 2-2: R-APDU - SW '6A 82' or '69 99' or '69 85' RESP 2-3: R-APDU - SW '90 00'	2. NoSuchElementError is expected.	

2.17 Chapter 6.4.6 openLogicalChannel – ID12

The same applies as for the test cases on openBasicChannel ID 3 and 13:

12	**The application opening the logical channel has no access to the selected SE applet**			
	1. session.openLogicalChannel (AID_accessdenied);	no selection <for AID_accessdenied>	None	SecurityError is expected.	CRC4

2.18 Chapter 6.4.7 openLogicalChannel() ID 14, 15, 16 & 17

For test cases ID 14 to 17, the v1.0 specification uses a set of test applets, AID_TestApp_SWxxxx, which do not send back any data to the device after the selection. Upon reception of the warning code (0x62xx or 0x63xx), the device issues a GET RESPONSE command. As the applets do not have any data to send back to the device, they answer with an error code (0x6D00) to the GET RESPONSE command and this error code is considered as a failure by the device.

In order to accurately check the device's behaviour, test cases were updated to use applets providing an answer to selectResponse().

14	Application selection returns a warning code 6283 (specified in ISO7816-4) – channel shall be opened				
	1. Session.openLogicalChannet(AID_TestAppl_SW6283_selectresponse) 2. channel.transmit(Test_APDU1);	CMD 1-1: APDU_MANAGE_CH_OPEN CMD 1-2: APDU_SELECT_BY_DF – CLA contains the Channel Number returned by the card in RESP 1-1; Data = 'AID_TestApp_SW6283_select response ' CMD 2: C-APDU ('01 10 01 00 04 01 02 03 04 00')	RESP 1-1: R-APDU - Data: Channel Number; SW '90 00' RESP 1-2: R-APDU -'DE AD C0 DE 62 83' RESP 2: R-APDU - Data = '01 02 03 04'; SW '90 00'	1. Returned Channel object is not null. No exception is expected. 2. Returned Response equals to 'R-APDU' - Data = '01 02 03 04'; SW '90 00'. No exception is expected.	CRN6
15	Application selection returns a warning code 6280 (not specified in ISO 7816-4) – channel shall be opened				
	1. Session.openLogicalChannet(AID_TestApp_SW6280_selectresponse_selectresponse) 2. channel.transmit(Test_APDU1);)	CMD 1-1: APDU_MANAGE_CH_OPEN CMD 1-2: APDU_SELECT_BY_DF – CLA contains the Channel Number returned by the card in RESP 1-1; Data = 'AID_TestApp_SW6280_select response ' CMD 2: C-APDU ('01 10 01 00 04 01 02 03 04 00')	RESP 1-1: R-APDU - Data: Channel Number; SW '90 00' RESP 1-2: R-APDU - 'DE AD C0 DE 62 80' RESP 2: R-APDU - Data = '01 02 03 04'; SW '90 00'	1. Returned Channel object is not null. No exception is expected. 2. Returned Response equals to 'R-APDU' - Data = '01 02 03 04'; SW '90 00'. No exception is expected.	CRN6
16	Application selection returns a warning code 6310 (not specified in ISO 7816-4) – channel shall be opened				
	1. Session.openLogicalChannet(AID_TestApp_SW6310_selectresponse_selectresponse) 2. channel.transmit(Test	CMD 1-1: APDU_MANAGE_CH_OPEN CMD 1-2: APDU_SELECT_BY_DF – CLA contains the Channel Number returned by the card in RESP 1-; Data = 'AID_TestApp_SW6310_select response ' CMD 2: C-APDU ('01 10 01 00 04 01 02 03 04 00')	RESP 1-1: R-APDU - Data: Channel Number; SW '90 00' RESP 1-2: R-APDU - 'DE AD C0 DE 63 10' RESP 2: R-APDU - Data = '01 02 03 04'; SW '90 00'	1. Returned Channel object is not null. No exception is expected. 2. Returned Response equals to 'R-APDU' - Data =	CRN6

	_APDU1);)			'01 02 03 04'; SW '90 00'. No exception is expected.	
17	Application selection returns a warning code 63C1 (specified in ISO7816-4) – channel shall be opened				
	1. Session.openLogicalChannet(AID_TestAppl_SW63C1_selectresponse_selectresponse) 2. channel.transmit(Test_APDU1);	CMD 1-1: APDU_MANAGE_CH_OPEN CMD 1-2: APDU_SELECT_BY_DF – CLA contains the Channel Number returned by the card in RESP 1-1; Data = 'AID_TestApp_SW63C1_selectresponse ' CMD 2: C-APDU ('01 10 01 00 04 01 02 03 04 00')	RESP 1-1: R-APDU - Data: Channel Number; SW '90 00' RESP 1-2: R-APDU - 'DE AD C0 DE 63 C1' RESP 2: R-APDU - Data = '01 02 03 04'; SW '90 00'	1. Returned Channel object is not null. No exception is expected. 2. Returned Response equals to 'R-APDU' - Data = '01 02 03 04'; SW '90 00'. No exception is expected.	CRN6

2.19 Chapter 6.5.6 transmit

Clarifications of the Initial Conditions and test procedure for several test cases:

(a) *Initial Conditions*

SEService Object has been created and the isConnected() method has been called and has returned true.
A Reader instance "reader" is selected and a Session instance "session" is opened with the selected "reader".
Test case ID1: A basic channel with "AID_TestApp" is already open.
Test cases ID2, ID5 to ID14 and ID16, ID19, ID20: A logical channel with "AID_TestApp" is already open.

Test cases ID3: A logical channel with "AID_TestApp" is already open. SE shall return logical channel number 1.

Test case ID4: Three channels are created in three different sessions.
Test case ID15: The two channels are created in two different sessions, each one created in a different SEService. (e.g. channel1 created by session1 created by seService1 created in thread1).
Test case ID17: A logical channel with "AID_TestApp_p1p2" is already open.
Test case ID18: A logical channel with "AID_TestApp_clains" is already open.
Test case ID13: UICC Simulator/UICC must only support T=0, a logical channel with "AID_TestApp" is already open.
Test case ID14: UICC Simulator/UICC must only support T=1, a logical channel with "AID_TestApp" is already open.
Test case ID 21: A logical channel with "AID_TestApp_SW61xx" is already open.

(b) *Test Procedure*

In case of T=0 protocol the case 2 type APDUs sent to the SE/UICC Simulator with wrong length are resent with correct length. The test procedure description only contains the APDUs sent first (with wrong length) and does not contain the APDUs resent with correct length.

For test case ID18, the scope of this test case is to check that the API implementation is not blocking any CLA/INS pairs except those mentioned in the specification. However, as some CLA/INS pairs are invalid, SE or UICC Simulator may send different R-APDU depending on their internal implementation. This behaviour is normal but it is impossible to specify accurately the "API expectation". As long as the API implementation returns the response of the Secure Element, whatever it is, the test shall be considered successful.
It means that the "API Expectation" is that the transmit method shall always return the R-APDU sent by the SE/UICC Simulator as a response to the C-APDU, whatever it is.

2.20 Update of the access control rules

Update of the initial access control rules to authorise the sending of additional APDU commands and to make sure that security errors come from the OMAPI internal implementation and not from the access control enforcer denying an access.
Test cases impacted are:

2.20.1 Annex B – Access Control ARA

Incoming APDU	Mask	Expected result
Test_APDU1	*FE FF FF FF*	*00 10 01 00*
Test_APDU2 (CLA = 00, 01)	*FE FF FF FF*	*00 10 02 00*
Test_APDU2 (CLA = 02)	*FF FF FF FF*	*02 10 02 00*
Test_APDU4	*FE FF FF FF*	*00 30 00 00*
Test_APDU5 and Test_APDU6	*FE EF FF FF*	*00 40 00 00*
APDU_SELECT_BY_FID	*FE FF FB FF*	*00 A4 00 00*
APDU_MANAGE_CHANNEL	*FE FF 7F E0*	*00 70 00 00*

2.20.1 Annex B – Access Control File System ARF

```
In addition to the common point for both ARA and ARF, fixing of an encoding issue
on the length of Tag for ARF.

ACConditions2:
30 53
04 00
A0 4F
A0 48
A1 46
04 08 00 10 01 00 FE FF FF FF
04 08 00 10 02 00 FE FF FF FF
04 08 02 10 02 00 FF FF FF FF
04 08 00 30 00 00 FE FF FF FF
04 08 00 40 00 00 FE EF FF FF
04 08 00 A4 00 00 FE FF FB FF
04 08 00 70 00 00 FE FF 7F E0
A1 03
80 01 00
```