ETSI TR 103 200 V1.1.1 (2011-09)

**Methods for Testing and Specification (MTS);
ePassport Readers Interoperability Support;
Framework for Developing Conformance Test Specifications**

Reference

DTR/MTS-00126 ePassFwk

Keywords

conformance, interoperability, testing, TTCN

Contents

Intellectual Property Rights

Foreword

This Technical Report (TR) has been produced by ETSI Technical Committee Methods for Testing and Specification (MTS).

1 Scope

The present document provides the Test Suite Structure and Test Purposes (TSS&TP) for ePassport Isnspection System testing in compliance with the relevant requirements and in accordance with the relevant guidance given in ISO/IEC 9646-7 [i.9].

The ISO standard for the methodology of conformance testing (ISO/IEC 9646-1 [i.7] and ISO/IEC 9646-2 [i.8]) as well as the ETSI rules for conformance testing (ETS 300 406 [i.6]) are used as a basis for the test methodology.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the reference document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

2.1 Normative references

Non applicable.

2.2 Informative references

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI ES 201 873-1: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: TTCN-3 Core Language".

[i.2] BSI Technical Guideline TR-03110 1.11: "Advanced Security Mechanisms for Machine Readable Travel Documents - Extended Access Control (EAC)".

[i.3] BSI TR-03105-5 1.1: "ePassport Conformity Testing - Test plan for ICAO compliant inspection systems with EAC".

[i.4] ICAO Document 9303, Edition 6, Part 1, Part 2 and Part 3.

[i.5] AIS Version 1.1: "ICAO Compliant Inspection Systems With EAC Conformity Testing - Automatic Interface Specification".

[i.6] ETSI ETS 300 406: "Methods for Testing and Specification (MTS);Protocol and profile conformance testing specifications; Standardization methodology".

[i.7] ISO/IEC 9646-1: "Information technology -- Open Systems Interconnection -- Conformance testing methodology and framework -- Part 1: General concepts".

[i.8] ISO/IEC 9646-2: "Information technology -- Open Systems Interconnection -- Conformance testing methodology and framework -- Part 2: Abstract Test Suite specification".

[i.9] ISO/IEC 9646-7: "Information technology -- Open Systems Interconnection -- Conformance testing methodology and framework -- Part 7: Implementation Conformance Statements".

[i.10] ETSI ETR 266: "Methods for Testing and Specification (MTS); Test Purpose style guide".

[i.11]	ISO/IEC 9646-6 (1994): "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 6: Protocol profile test specification".
[i.12]	ISO/IEC 7816-4: "Identification cards - Integrated circuit cards: Organization, security and commands for interchange".
[i.13]	ISO/IEC 14443: "Cartes d'identification -- Cartes à circuit(s) intégré(s) sans contact -- Cartes de proximité".
[i.14]	ISO/IEC 9796-2: "Information technology -- Security techniques -- Digital signature schemes giving message recovery -- Part 2: Integer factorization based mechanisms".
[i.15]	ISO/IEC 15946: "Information technology -- Security techniques -- Cryptographic techniques based on elliptic curves".
[i.16]	ANSI X9.63: "Public Key Cryptography for the Financial Services Industry, Key Agreement and Key Transport Using Elliptic Curve Cryptography".
[i.17]	IETF RFC 3278: "Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ISO/IEC 7816-4 [i.12], BSI Technical Guideline TR-03110 1.1 [i.2], BSI TR-03105-5 1.1 [i.3] and ICAO 9303 [i.4], Part 1 Vol.2 apply.

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AA	Active Authentication
AIP	Advanced Inspection Procedure
APDU	Application Protocol Data Unit
ASP	Abstract Services Primitives
ATM	Abstract Test Method
ATS	Abstract Test Suite
BAC	Basic Access Control
BHT	Biometric Header Template
CA	Chip Authentication
C-APDU	Command APDU
CAR	Certificate Authority Reference
CCSCA	Country Signing Certificate Authority Certificate
CDS	Document Signer Certificate
CMS	Cryptographic Message Syntax
CRL	Certificate Revocation List
CSCA	Country Signing Certification Authority
CV	Card Verifiable
CVCA	Country Verifying Certification Authority
DG	Data Group
DH	Diffie-Hellman
DV	Document Verifier
DVCA	Document Verifying Certification Authority
EAC	Extended Access Control
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
IS	Inspection System
IUT	Implementation Under Test

LDS	Logical Data Structure
MAC	Message Authentication Code
MRP	Machine Readable Passport
MRZ	Machine Readable Zone
MTC	Main Test Component
OID	Object IDentifier
PA	Passive Authentication
PKI	Public Key Infrastructure
R-APDU	Response APDU
RSA	Rivest-Shamir-Adleman
SHA	Secure Hash Algorithm
SIP	Standard Inspection Procedure
SOD	Document Security Object
SSC	Send Sequence Counter
SUT	System Under Test
TA	Terminal Authentication
TC	Test Case
TSS	Test Suite Structure
WSQ	Wavelet Scalar Quantization

4 Electronic Passport Overview

4.1 Introduction

The difference between a traditional passport and an electronic passport (ePassport) is the embedded chip with contactless interface (and the electronic passport logo on the front cover). According to [i.4] the location of the contactless integrated circuit with its associated antenna in the MRP is at the discretion of the issuing State. States should be aware of the importance of the need for the contactless IC to be protected against physical tampering and casual damage including flexing and bending. (e.g. see Figure 1). The chip is a contactless smart card compliant to the ISO/IEC 14443 [i.13] standard (both variants - A and B - are allowed). Technology based on ISO/IEC 14443 [i.13] is designed to communicate over distance up to 10 cm and supports also relatively complex cryptographic chips and permanent memory of kilobytes or megabytes. Here it differs from many other RFID technologies that are capable to communicate over longer distances, but do not support more complicated operations other than sending a simple identification bitstring. Higher communication layer is based on classical smart card protocol ISO/IEC 7816-4 [i.12] (i.e. commands like SELECT AID, SELECT FILE and READ BINARY are used).

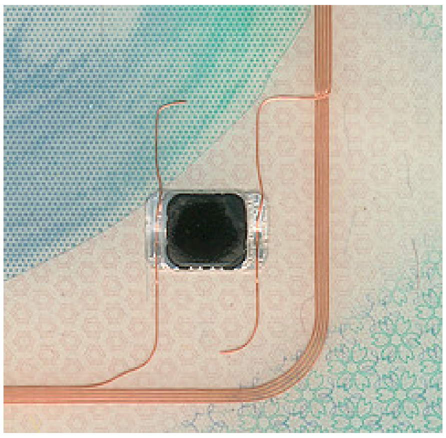


Figure 1: Directly visible contactless chip and antenna in UK passports

The data in electronic passports are stored as files (elementary files in the smart card terminology) in a single folder (dedicated file). Up to 16 data files named as DG1 to DG16 (DG for Data Group) can hold the data. DG1 contains the data from the machine-readable zone (i.e. nationality, first name, surname, passport number, issuing state, sex, birth date, validity date, and optional data - for example a personal number), DG2 contains the photo of the passport holder (in JPEG or JPEG2000 and some additional metadata). DG3 is dedicated for fingerprints, DG4 may contain iris image. Remaining data groups contain information about the holder, issuing institution or passport itself. Two or three additional files with metadata are also present. The file EF.COM contains a list of available data groups (and the information about versions used) and the file EF.SOD contains the digital signature of the data. EF.CVCA file may contain the name of the trustpoint used for the access control to sensitive biometric data. The files EF.COM, EF.SOD, DG1 and DG2 are mandatory for all electronic passports. The data groups DG3 and DG14 have been made mandatory in the EU countries after 28th June 2009. All other data groups are optional.

4.2 Data integrity (passive authentication)

Data integrity of the stored information is protected by a digital signature available in the EF.SOD file. The file uses the SignedData structure of the CMS (Cryptographic Message Syntax) standard. The PKI hierarchy has a single level. Each country establishes its own CSCA (Country Signing Certificate Authority), which certifies bodies responsible for issuing the passports (e.g. the state printers, embassies etc.). These bodies are called Document Signers. Data in the passport are then signed by one of these Document Signers.

To verify signatures, the CSCA certificate of the issuing country needs to be available and its integrity guaranteed. The certificate of the Document Signer is either directly stored in the passport (in the certificate part of the SignedData structure - and this is mandatory in the EU) or is obtained from other sources (the issuing country, the ICAO public key directory, etc.).

The signed data is a special structure containing hashes of all present datagroups in the passport. Integrity of each file can be verified separately (i.e. first the digital signature in EF.SOD is verified and then integrity of each file is checked by verifying its hash against the hash stored in the EF.SOD file).

The digital signature is one of the key security mechanisms of the electronic passports - if not the most important one. Every country chooses the signature scheme that best fits its needs from the implementation and security perspective (supported schemes are RSA PKCS#1 v1.5, RSA PSS, DSA and ECDSA in combination with SHA-1 or any of the SHA-2 hash functions). Every inspection system needs to support all these schemes to be able to verify any valid passport. The signature verification is a relatively simple process, yet complications may arise due to a relatively high number of signature schemes that have to be supported, availability of the root certificates (CSCA) of all countries and CRLs (each country is required to issue one at least every 90 days).

Digital signature alone cannot prevent from making identical copies of the passport content (including the EF.SOD file with digital signature) - so-called cloning. The inspection of the classical security features (security printing, watermarks, holograms, etc.) still makes sense and the correspondence between the printed data and the data stored on the chip should also be verified.

4.3 Active authentication (AA)

Cloning of passports can be prevented by using a combination of cryptographic techniques and reasonable tamper resistance. In such a case a passport-specific asymmetric key pair is stored in the chip. Whereas the public key is freely readable (stored in DG15 and its hash is digitally signed), the private key is not readable from the chip and its presence can be only verified using a challenge-response algorithm (based on ISO/IEC 9796-2 [i.14]). This protocol is called the Active Authentication (AA) and it is an optional security feature of electronic passports. Also for EU countries AA is an optional feature and indeed not all the countries implement active authentication mechanism.

The point of the active authentication is to verify whether the chip in the passport is authentic. The inspection system generates an 8-byte random challenge and using the INTERNAL AUTHENTICATE command asks the chip to authenticate. The chip generates its own random string and cryptographically hashes both parts together. The chip's random string and the hash of both parts (together with a header and a tail) are then signed by the chip's private key. The result is sent back to the inspection system, which verifies the digital signature. If the digital signature is correct the chip is considered to be authentic. Possible attacks might try to exploit weaknesses in the tamper resistance of the chip or can be based on the analysis of side-channels.

4.4 Basic Access Control (BAC)

Basic access control is a mechanism that prevents reading of the passport data before the authentication of the inspection system (i.e. prevents so-called skimming). The authentication keys are derived from data printed in the machine-readable zone of the data page. The document number, the birth date of the holder and the passport expiration date are used. All these items are printed in the second line of the machine readable zone and are protected with a check digit (the optical character recognition is error prone; hence the choice of data fields with check digits). These three entries are concatenated in an ASCII form (including their respective check digits) and are hashed using the SHA-1 function. The hash value is then used to derive two (112-bit 3DES) keys for encryption and MAC authentication. The command GET CHALLENGE is used to obtain the challenge from the chip and then the inspection system and the chip mutually authenticate using the MUTUAL AUTHENTICATE command. The session key is established and further communication is secured using Secure Messaging.

BAC is based on a standard mutual authentication technique, which is considered to be secure as long as the keys are kept secret. In the case of electronic passports the keys are not secret in the classical sense as they are derivable from the data printed in the passport, but even so could prevent the random remote reading. This is, however, slightly problematic as the data used to derive the key do not necessarily have much of entropy. Although the theoretical maximum is 58 bits and in case of alphanumerical document numbers even 74 bits, real values are significantly lower. Some analysis have shown that the total entropy can be reduced to approximately 30 bits to 40 bits in certain situations. The brute-force key search then can be then mounted against a significantly smaller number of possible keys.

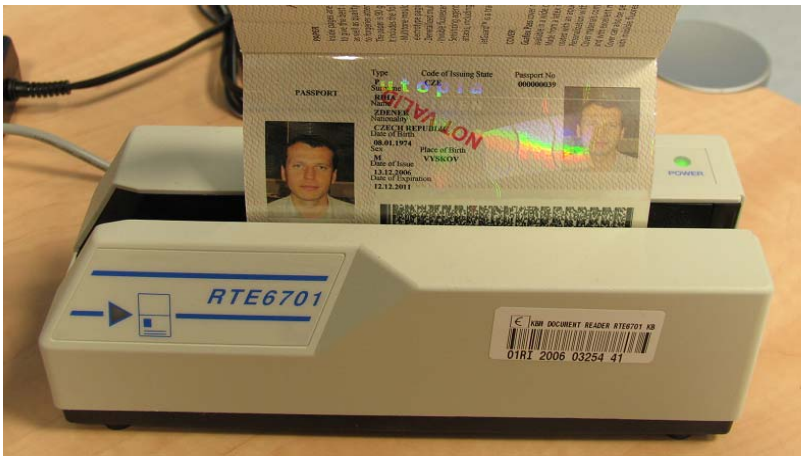


Figure 2: Scanning of the machine-readable zone data

4.5 Extended Access Control (EAC)

Electronic passports of so-called second generation store fingerprints as images in the WSQ format (lossy compression optimized for images of fingerprints). As fingerprints are considered to be more sensitive data than facial images (their recognition capabilities are much better), reading of DG3 is protected by an additional mechanism. This mechanism is called the Extended Access Control. In EU the Extended Access Control is based on asymmetric cryptography and PKI as defined in [i.2]. The European EAC consists of two protocols. The aim of the chip authentication is to verify authenticity of the passport chip (similarly as in AA) and replace low-entropy session keys for Secure Messaging with new session keys with high entropy to cope with the problem of communication eavesdropping. The role of the terminal authentication is to control access to sensitive biometric data (fingerprints, possibly also iris images).

4.5.1 Terminal authentication

Each country establishes a CVCA (Country Verifying Certification Authority) that decides which other countries will have the access to sensitive biometric data in their passports. A certificate of this authority is stored in passports (issued by that country) and it forms the starting trust point (root certificate) for the access control. Other countries wishing to access sensitive biometric data (no matter if in their own passports or in passports of other countries), have to establish a DVCA (Document Verifying Certification Authority). This authority will obtain the certificate from all countries willing to grant access to the data in their own passports. This DVCA will then issue the certificates to end-point entities actually accessing the biometric data - the inspection systems. See Figure 3.

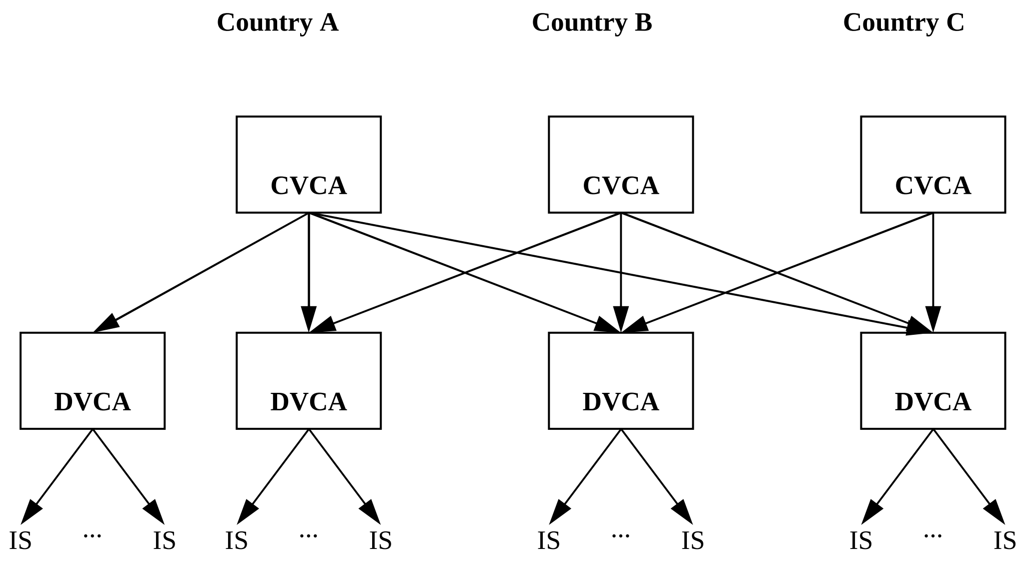


Figure 3: A simplified view of an EAC PKI hierarchy

Each passport stores a CVCA certificate of the issuing country (e.g. country C). In order to convince the passport that it is authorized to access sensitive biometric data, an inspection system (e.g. one of country S) needs to provide the DV certificate (issued by the country S in our case) signed by the issuing CVCA (of the country C) and its own IS certificate (for that particular IS) signed by the DV certification authority (i.e. of the country S in this case). After the passport verifies the whole certification chain it has to check whether the inspection system can access the corresponding private key. That is performed using a challenge-response protocol. If the authentication succeeds, the inspection system can access sensitive biometric data (the DG3 and/or DG4 files). This part of the EAC is called the Terminal Authentication (TA).

The above mentioned process can be slightly more complicated as the CVCA certificates are updated from time to time (by link certificates) and the bridging link certificates have to be provided (and verified by the passport) at first. The terminal authentication can be based on RSA (the PSS as well as PKCS#1 v1.5 padding is possible) or ECDSA, both in combination with SHA-1 or one of SHA-2 hash functions.

Certificates are sent by using commands Manage Security Environment - Set for verification - Digital Signature Template and Perform Security Operation - Verify Certificate. The certificate chain may contain also link certificates if necessary and (after their verification) the passport updates the CVCA certificate with a new one (due to a possible overlap of the validity periods of the CVCA certificates, there can be up to two certificates valid at the same time - in such case both are stored in the passport). Remaining certificates (the DV certificate issued by the CVCA and the DVCA certificate issued for IS) are stored only temporarily and used during the verification of the certificate chain. Once the chain verification succeeds, the passport obtains the public key of the IS and its access rights. Only two access rights are specified at the moment, these are reading access to DG3 (fingerprints) and to DG4 (iris image).

After obtaining the public key of an IS it has to be verified if the IS has also the access to the corresponding private key. This is done using a challenge-response protocol. At first, the inspection system gets an 8-byte long random challenge (using the GET CHALLENGE command), signs it (in fact the concatenation of the passport number, random challenge and the hash of the ephemeral DH key of the inspection system (from the previous chip authentication) is signed). The signature is then sent to the chip for verification using the EXTERNAL AUTHENTICATE command. If the verification runs correctly, the inspection system is authenticated and may access DG3 or DG4 according to the assigned rights. Terminal authentication is not a mandatory part of the communication with the electronic passport. The inspection system can skip the terminal authentication if there is no need to read the secondary biometric data from the chip.

As the computational power of smart cards is limited, simplified certificates (card verifiable certificates) are used instead of X.509 certificates. An interesting point is the verification of certificate validity. As the chip has no internal clock, the only available time-related information is the certificate issue date. If the chip successfully verifies the validity of given certificate issued on a particular day, then it knows that this date has already passed (or is today) and can update its own internal time estimate (if the value is newer than the one already stored). It is clear that if a CVCA or DVCA issues (either by a mistake, intentionally or as a result of an attack) a certificate with the issue date in a distant future, the passport will then be rejecting valid certificates and will become practically unusable. For that reason, only the CVCA (link certificates), DV and domestic IS certificates are used to update the internal date estimate.

4.5.1.1 Chip authentication

In addition to the terminal authentication, the European EAC also introduces the Chip Authentication (CA) protocol, which eliminates the low entropy of the BAC key and also may replace active authentication, as access to the private key on the chip is verified (the public key is stored in DG14 and is part of the passive authentication).

An inspection system reads the public part of the Diffie-Hellman (DH) key pair from the passport (supported are the classic DH described in PKCS #3 and DH based on elliptic curves (ECDH) according to ISO/IEC 15946 [i.15]), together with the domain parameters (stored in DG14). Then the inspection system generates its own ephemeral DH key pair (valid only for a single session) using the same domain parameters as the chip key and sends it to the chip (using the command Manage Security Environment - Set for Computation - Key Agreement Template). The chip as well as the IS can then derive the shared secret based on available information. This secret is used to construct two session keys (one for encryption and the other one for MAC) that will secure the subsequent communication by Secure Messaging (and SSC (Send Sequence Counter - the message counter value utilized for protection against replay attack) is reset to zero). Whether the chip authentication ran successfully or not is only clear after sending and receiving the next command correctly protected with the new session keys.

4.6 Inspection system definition

An Inspection system is a system used for inspecting (e)MRTDs by any public or private entity having the need to validate the (e)MRTD, and using the present document for identity verification, e.g. border control authorities, airlines and other transport operators, financial institutions.

In order to support the required functionality and the defined options that can be implemented on MRtds that will be offered, the inspection system will have to meet certain pre-conditions.

For MRtd Basic Access Control

Although the described Basic Access Control is OPTIONAL, inspection systems supporting it have to meet the following pre-conditions:

1) The inspection system is equipped with an MRZ reader or a form of manual input device (e.g.a keyboard) to derive the Document Basic Access Keys (KENC and KMAC) from the MRtd.

2) The inspection system's software supports the protocol described upper, in the case that an MRtd with Basic Access Control is offered to the system, including the encryption of the communication channel with Secure Messaging.

For Passive Authentication

To be able to perform a passive authentication of the data stored in the MRtd's contactless IC, the inspection system needs to have knowledge of key information of the issuing States:

1) Of each participating issuing State, the Country Signing Certificate Authority Certificate (CCSCA) is stored in the inspection system.

2) Of each participating issuing State, the Document Signer Certificate CDS is stored in the inspection system.

Before using a Document Signer Certificate for verification of a SOD, the inspection system verifies its digital signature, using the Country Signing CA Public Key (KPuCSCA).

For Active Authentication

Support of Active Authentication by inspection systems is OPTIONAL.

If the inspection system supports the OPTIONAL Active Authentication, it is REQUIRED that the inspection system have the ability to read the visual MRZ.

If the inspection system supports the OPTIONAL Active Authentication, the inspection system's software should support the Active Authentication protocol.

For Extended Access Control to additional biometrics

The implementation of the protection of the OPTIONAL additional biometrics depends on the State's internal specifications or the bilaterally agreed specifications between States sharing this information.

If the inspection system supports the OPTIONAL Extended Access Control, the inspection system's software should support the Chip and Terminal Authentication protocol.

4.7 Use Cases: Example of EAC Message flows

Auth	Step	Direction		Message	Comment
		MRTD	IS		
BAC	1	←		Select AID	IS selects Application
	2		→	OK 9000	MTRD ACK OK
	3	←		Select EF.COM	IS selects EF.COM to read
	4		→	NOK 6982	Security Status Not satisfied
	5	←		Get Challenge 84	IS requests a challenge
	6		→	Data	MRTD reply the RND.IFD.
					IS
	7	←		External Mutual Authenticate 82	IS Send its challenge RND.IFD
	8		→	Data	MRTD reply the RND.IFD.
					Calculation of Authentication Data: => KENC and KMAC are derived
SM					The messages flows is now encrypted by Secure Messaging
EF.COM	9	←		Select EF.COM	IS selects EF.COM to read
	10		→	OK 9000	MTRD ACK OK
	11	←		Read x bytes	IS requests to read the first x bytes of the file
	12		→	Data	MTRD provide the first x bytes of the file
		…		…	Read sequence until the end of the EF.COM file
CA	13	←		Select DG.14	IS selects DG.14 to read
	14		→	OK 9000	MTRD ACK OK
	15	←		Read x bytes	IS requests to read the first x bytes of the file
	16		→	Data	MTRD provide the first x bytes of the file
		…		…	Read sequence until the end of the DG.14 file
	17	←		MSE:Set KAT (Key Agreement Template)	IS initiates a Key Agreement operation
	18		→	OK 9000	MTRD ACK OK
					New KENC and KMAC derived The Message flow is now encrypted by new derived Secure Messaging
	19	←		Select EF.COM	IS selects EF.COM to read in order to validate the Chip Authentication
	20		→	OK 9000	MTRD ACK OK => CA successful
AA	21	←		Select DG.15	IS selects DG.15 to read
	22		→	OK 9000	MTRD ACK OK
	23	←		Read x bytes	IS requests to read the first x bytes of the file
	24		→	Data	MTRD provide the first x bytes of the file
		…		…	Read sequence until the end of the DG.15 file.
					IS extracts from received DG.15 the AA Public key of the MRTD
	25	←		Internal Authenticate 88	I requests a signature
	26		→	Data	MTRD signs the data and provides the signature. IS decrypts the signature: if OK, the Active Authentication is successful.

Auth	Step	Direction		Message	Comment
		MRTD	IS		
EF.COM	27	←		Select EF.SOD	IS selects EF.SOD to read
	28		→	OK 9000	MTRD ACK OK
	29	←		Read x bytes	IS requests to read the first x bytes of the file
	30		→	Data	MTRD provide the firstx bytes of the file
		...		...	Read sequence until the end of the EF.SOD file. IS checks the following info: SOD message digest OK Certificate Signature Certificate validity period Hash DG14 OK Hash DG15 OK EF.COM integrity check OK The Passive Authentication is successful
EF.DG1	31	←		Select EF.DG1	IS selects EF. DG1 to read
	32		→	OK 9000	MTRD ACK OK
	33	←		Read x bytes	IS requests to read the first x bytes of the file
	34		→	Data	MTRD provide the first x bytes of the file
		...		...	Read sequence until the end of the EF. DG1 file
EF.DG2	35	←		Select EF.DG2	IS selects EF. DG2 to read
	36		→	OK 9000	MTRD ACK OK
	37	←		Read x bytes	IS requests to read the first x bytes of the file
	38		→	Data	MTRD provide the first x bytes of the file
		...		...	Read sequence until the end of the EF. DG2 file
TA	39	←		Select EF.CVCA	IS selects EF. CVCA to read
	40		→	OK 9000	MTRD ACK OK
	41	←		Read x bytes	IS requests to read the first x bytes of the file
	42		→	Data	MTRD provide the first x bytes of the file
		...		...	Read sequence until the end of the EF. CVCA file
	43	←		MSE:Set DST (Key Reference)	IS initiates verifcation of Digital Signature procedure
	44		→	OK 9000	MTRD ACK OK
	45	←		PSO:Verify Certificate	IS Request to verify DV certificate
	46		→	OK 9000	MTRD ACK OK
	47	←		MSE:Set DST (Key Reference)	IS initiates verifcation of Digital Signature procedure
	48		→	OK 9000	MTRD ACK OK
	49	←		PSO:Verify Certificate	IS Request to verify IS certificate
	50		→	OK 9000	MTRD ACK OK
	51	←		MSE:Set AT (Key Reference)	IS initiates External Authentication Template
	52		→	OK 9000	MTRD ACK OK
	53	←		Get Challenge (84)	IS requests a challenge to the MRTD
	54		→	Data (8 bytes)	
	55	←		External Authenticate (82)	IS send s the challenge. This will prove that it has trhe private key corresponding to the public key of the IS certificate.
	56		→	OK 9000	MTRD ACK OK. Terminal Authentication Successful. **EAC established**
EF.DG3	57	←		Select EF.DG3	IS selects EF. DG3 to read
	58		→	OK 9000	MTRD ACK OK
	59	←		Read x bytes	IS requests to read the first x bytes of the file
	60		→	Data	MTRD provide the first x bytes of the file
		...		...	Read sequence until the end of the EF. DG3 file

4.8 Automatic Interface

The Automatic interface specification [i.5] defines a specific and easy-to-implement automatic mode for Inspection System testing with four objectives:

1) To distribute all certificates from test platform to Inspection System (TEST -> IS).

2) To provide MRZ information (line 1, 2 & 3) for BAC and SAC from test platform to Inspection System (TEST -> IS).

3) To trigger the inspection procedure start (TEST -> IS).

4) To collect the results from Inspection System to Test Platform (IS -> TEST).

These objectives will automate the test procedure of an Inspection System.

The specification is based on APDU exchanges between the Inspection System and Test Platform through the emulator probe.

The four objectives described above will be managed with GET DATA and PUT DATA APDU exchanges as shown below:

Objectives	APDU Reference	Status
Objective 1	APDU GET_DATA_CERTIFICATES_ID	Optional
Objective 2	APDU GET_DATA_MRZ_CAN	Optional
Objective 3	No APDU exchange needed	-
Objective 4	APDU PUT_DATA_RESULTS	Mandatory

The Automatic Interface specification defines also a different way to retrieve the complete certificates data instead of using APDU GET_DATA_CERTIFICATES_ID.

The idea is to provide the data in a specific Data Group loaded in the ePassport emulator, and not only the certificate ID. The Data Group data is structured as concatenation of DER encoding Data Object within the template tag 70h.

5 Test Suite Structure (TSS)

5.1 Structure for ePassport Inspection System tests

Table 1 shows the ePassport Inspection System Test Suite Structure (TSS) including its subgroups defined for conformance testing.

Table 1: TSS for ePassport Inspection System

Root	Group	Sub-Group
ePassport IS	Application protocol	
		Logical data structure
		Application selection
		Basic access control
		Reading binary files
		Chip authentication
		Terminal authentication
	Logical data structure	
		EF.COM
		EF.DG1
		EF.DG2
		EF.DG3
		EF.DG4
		EF.DG14
		EF.CVCA
		EF.SOD
		EF.DG15

The test suite is structured as a tree with the root defined as ePassport IS. The tree is of rank 2 with the first rank a Group and the second a Sub-group.

5.2 Test groups

The test suite has a total of four levels. The first level is the root. The second level separates the root into two groups, each one representing a protocol layer. The third level are the sub-functional areas.

5.2.1 Root

The root identify the ePassport Inspection System.

5.2.2 Groups

This level contains two protocol layers identified as: Application protocol (Layer 6), and Logical data structure (Layer 7).

6 Test Purposes (TP)

6.1 Introduction

6.1.1 TP definition conventions

The TP definition is built according to ETR 266 [i.10].

6.1.2 TP Identifier naming conventions

The identifier of the TP is built according to Table 2.

Table 2: TP naming convention

Identifier:	**TP/<gr>/<sgr>/<nn>**		
	<gr> = group	ISO7816	Application protocol
		LDS	Logical data structure
	<sgr> =sub- group	ISO7816/A	Application selection
		ISO7816/B	Basic access control
		ISO7816/C	Reading binary files
		ISO7816/D	Chip authentication
		ISO7816/E	Terminal authentication
		ISO7816/F	Active authentication
		LDS/A	EF.COM
		LDS/B	EF.DG1
		LDS/C	EF.DG2
		LDS/D	EF.DG3
		LDS/E	EF.DG4
		LDS/F	EF.DG14
		LDS/G	EF.CVCA
		LDS/H	EF.SOD
		LDS/I	EF.DG15
	<nn> = sequential number		01 to 99

6.1.3 Rules for the behaviour description

The description of the TP is built according to ETR 266 [i.10].

The base standards are not using finite state machine concept. As consequence, the test purposes use a generic "Intial State" that corresponds to a state where the IUT is ready for starting the test execution. Furthermore, the IUT will be left in this "Initial State", when the test is completed.

Being in the "Initial State", no pending actions, which could disturb the execution of following test purposes, are left in the IUT.

6.1.4 Sources of TP definitions

All TPs are specified according to BSI TR-03105 [i.3].

6.2 Test purposes for ePassport Inspection System

6.2.1 ISO7816_A

TP Id	TP/ISO7816/A/02
Test objective	Checks that IUT can successfully read a BAC protected ePassport
Reference	9003v2
Profile	SIP
Configuration	CFG.DFLT.BAC
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM passed
    }
}	

6.2.2 ISO7816_B

TP Id	TP/ISO7816/B/07
Test objective	Checks that IUT recognizes an incorrect R-APDU in first secure messaging command
Reference	9003v2
Profile	SIP
Configuration	CFG.DFLT.BAC

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives secured R-APDU not containing status bytes (tag 99)
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC failed and
        IUT indicates PA not performed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM failed
    }
}
```

TP Id	TP/ISO7816/B/08
Test objective	Checks that IUT recognizes an SM failure in the R-APDU
Reference	9003v2
Profile	SIP
Configuration	CFG.DFLT.BAC

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives secured R-APDU
            containing an incorrect MAC due to non incremented SSC
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC failed and
        IUT indicates PA not performed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM failed
    }
}
```

6.2.3 ISO7816_C

TP Id	TP/ISO7816/C/02
Test objective	Checks that IUT is capable of reading large binary files
Reference	9003v2
Profile	SIP
Configuration	CFG.BAC.ISO7816.C02
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives EF.DG2 file larger than 32 Kbytes
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM passed
    }
}	

TP Id	TP/ISO7816/C/03
Test objective	Checks that IUT recognizes the end of a binary file
Reference	9003v2
Profile	SIP
Configuration	CFG.BAC.ISO7816.C03
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an incomplete EF.DG2 file
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA not performed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM failed
    }
}	

6.2.4 ISO7816_D

c	TP/ISO7816/D/02
Test objective	Checks that IUT performs Chip Authentication successfully with Elliptic Curve Diffie-Hellman algorithm and no key reference in DG14
Reference	TR-03110
Profile	AIP
Configuration	CFG.DFLT.EAC
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}	

TP Id	TP/ISO7816/D/04
Test objective	Checks that IUT performs Chip Authentication successfully if there are two key references in DG14.
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.ISO7816.D04
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG14 file
            containing 2 key references
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}	

TP Id	TP/ISO7816/D/11
Test objective	Checks that IUT fails Chip Authentication if there is an invalid DH key specification in DG14
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.ISO7816.D11
Initial conditions	

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG14 file
            containing SubjectPublicKeyInfo indicating invalid OID
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA failed and
        IUT indicates COM passed
    }
}
```

6.2.5 ISO7816_E

TP Id	TP/ISO7816/E/06
Test objective	Checks that IUT performs Terminal Authenticationsuccessfully with ECDSA-SHA224 algorithm
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.ISO7816.E06
Initial conditions	

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.CVCA file
            containing CAR indicating Trust Point using TA-ECDSA-SHA-224 algorithm
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

TP Id	TP/ISO7816/E/08
Test objective	Checks that IUT shows correct behaviour if EF.CVCA stores a wrong CAR
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.ISO7816.E08
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.CVCA file
            containing CAR indicating a wrong value
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA failed and
        IUT indicates CA passed and
        IUT indicates COM passed
        IUT indicates EF.CVCA failed
    }
}	

TP Id	TP/ISO7816/E/11
Test objective	Checks that IUT shows correct behaviour if external authentication internally uses a wrong document number
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.ISO7816.E11
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT sends an External Authenticate C-APDU and
        IUT receives an R-APDU indicating authentication failure
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA failed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}	

TP Id	TP/ISO7816/E/12
Test objective	Checks that IUT shows correct behaviour if GET CHALLENGE command delivers only 7 bytes
Reference	TR-03110
Profile	AIP
Configuration	CFG.DFLT.EAC

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT sends a Get Challenge C-APDU during Terminal Authentication and
        IUT receives an R-APDU
            containing Challenge indicating 7 bytes
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA failed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

TP Id	TP/ISO7816/E/18
Test objective	Checks that IUT shows correct behaviour if IS certificate contains wrong signature
Reference	TR-03110
Profile	AIP
Configuration	CFG.DFLT.EAC

Initial conditions

```
with {
    IUT using IS certificate with wrong signature
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA failed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

TP Id	TP/ISO7816/E/28
Test objective	Checks that IUT performs Chip Authentication and Terminal Authentication successfully if there are same algorithms but different key sizes used in CA and TA
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.ISO7816.E28

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG14 file
            containing Key agreement algorithm indicating CA-ECDH-3DES-CBC-CBC and
            containing Key size indicating 224
}
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

6.2.6 ISO7816_F

TP Id	TP/ISO7816/F/05
Test objective	Checks that IUT performs Active Authentication with RSA-SHA256 algorithm in signature function
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.ISO7816.F05

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG15 file
            containing Signature algorithm indicating RSA SHA256
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA passed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

6.2.7 LDS_A

TP Id	TP/LDS/A/03
Test objective	Checks that IUT performs correctly if EF.COM is wrong (length byte of tag 60 is too big)
Reference	9303v2
Profile	SIP
Configuration	CFG.EAC.LDS.A03

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.COM file
            containing Tag 60 indicating length too big
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM failed and
        IUT indicates EF.COM failed
    }
}
```

TP Id	TP/LDS/A/04
Test objective	Checks that IUT performs correctly if EF.COM is wrong (incorrect LDS version)
Reference	9303v2
Profile	SIP
Configuration	CFG.EAC.LDS.A04

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.COM file
            containing LDS Version indicating version 3.0
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM passed and
        IUT indicates EF.COM failed
    }
}
```

6.2.8 LDS_B

TP Id	TP/LDS/B/11
Test objective	Checks that IUT performs correctly if EF.DG1 is wrong (name in DG1 and on data page are different)
Reference	9303v2
Profile	SIP
Configuration	CFG.EAC.LDS.B11

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG1 file
            containing Name indicating value not corresponding to Data Page MRZ
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM passed and
        IUT indicates EF.DG1 failed
    }
}
```

TP Id	TP/LDS/B/22
Test objective	Checks that IUT performs correctly if EF.DG1 is wrong (incorrect checksum of optional data)
Reference	9303v2
Profile	SIP
Configuration	CFG.EAC.LDS.B22

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG1 file
            containing optional data indicating wrong checksum
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM passed and
        IUT indicates EF.DG1 failed
    }
}
```

TP Id	TP/LDS/B/25
Test objective	Checks that IUT performs correctly if EF.DG1 stores an incomplete birth date (missing day)
Reference	9303v2
Profile	SIP
Configuration	CFG.EAC.LDS.B25

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG1 file
            containing birth date not indicating birth day
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM passed
    }
}
```

6.2.9 LDS_C

TP Id	TP/LDS/C/03
Test objective	Checks that IUT performs correctly if EF.DG2 contains an image in JPEG2000 format with additional facial feature points
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.C03

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG2 file
            containing JPG2000 image indicating additional facial feature points
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

TP Id	TP/LDS/C/09
Test objective	Checks that IUT performs correctly if EF.DG2 is wrong (BHT, missing format owner)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.C09

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG2 file
            containing BHT not indicating format owner
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.DG2 failed
    }
}
```

TP Id	TP/LDS/C/13
Test objective	Checks that IUT performs correctly if EF.DG2 is wrong (BHT, incorrect biometric type)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.C13

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG2 file
            containing BHT indicating incorrect biometric type
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.DG2 failed
    }
}
```

TP Id	TP/LDS/C/19
Test objective	Checks that IUT performs correctly if EF.DG2 is wrong (FIB, incorrect hair colour)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.C19
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG2 file
            containing FIB indicating incorrect hair colour
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.DG2 failed
    }
}	

6.2.10 LDS_D

TP Id	TP/LDS/D/03
Test objective	Checks that IUT performs correctly if EF.DG3 contains three fingerprints in WSQ format
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.D03
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG3 file
            containing 3 fingerprints in WSQ format
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}	

TP Id	TP/LDS/D/12
Test objective	Checks that IUT performs correctly if EF.DG3 is wrong (BHT, incorrect biometric subtype)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.D12
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG3 file
            containing BHT indicating incorrect biometric subtype
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.DG3 failed
    }
}	

TP Id	TP/LDS/D/15
Test objective	Checks that IUT performs correctly if EF.DG3 is wrong (Missing fingerprint image in instance two (Tag 5F2E))
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.D15
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG3 file
            containing Tag 5F2E not containing fingerprint image
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.DG3 failed
    }
}	

6.2.11 LDS_E

TP Id	TP/LDS/E/01
Test objective	Checks that IUT performs correctly if EF.DG4 contains two iris images in JPG2000 format
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.E01

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG4 file
            containing 2 iris images in JPG2000 format
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

TP Id	TP/LDS/E/09
Test objective	Checks that IUT performs correctly if EF.DG4 is wrong (BHT, not allowed format type)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.E09

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG4 file
            containing BHT indicating not allowed format type
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.DG4 failed
    }
}
```

6.2.12 LDS_F

TP Id	TP/LDS/F/04
Test objective	Checks that IUT performs correctly if EF.DG14 is wrong (not allowed chip authentication public key info OID in Security Infos)
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.LDS.F04

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG14 file
            containing SecurityInfos indicating not allowed chip authentication public key info OID
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA failed and
        IUT indicates COM passed and
        IUT indicates EF.DG14 failed
    }
}
```

TP Id	TP/LDS/F/06
Test objective	Checks that IUT performs correctly if EF.DG14 is wrong (incorrect version element in ChipAuthenticationInfo)
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.LDS.F06

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG14 file
            containing ChipAuthenticationInfo indicating incorrect version element
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA failed and
        IUT indicates COM passed and
        IUT indicates EF.DG14 failed
    }
}
```

6.2.13 LDS_G

TP Id	TP/LDS/G/01
Test objective	Checks that IUT performs correctly if EF.CVCA is wrong (first CAR is not encoded by tag 42)
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.LDS.G01

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.CVCA file
            containing CAR not indicating tag ID 42
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM passed and
        IUT indicates EF.CVCA failed
    }
}
```

TP Id	TP/LDS/G/02
Test objective	Checks that IUT performs correctly if EF.CVCA is wrong (missing CARs, file empty, no trust point implemented)
Reference	TR-03110
Profile	AIP
Configuration	CFG.EAC.LDS.G02

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.CVCA file
            containing no information
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA not performed and
        IUT indicates CA not performed and
        IUT indicates COM passed and
        IUT indicates EF.CVCA failed
    }
}
```

6.2.14 LDS_H

TP Id	TP/LDS/H/03
Test objective	Checks that IUT performs correctly if EF.SOD contains RSASSA-PSS with SHA256, SHA256 DG hash, DS stored inside SOD
Reference	9303v2
Profile	AIP
Configuration	CFG.DFLT.EAC
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing Signature algorithm indicating RSASSA-PSS with SHA256
            containing DG hashes indicating SHA256 algorithm
            containing DS
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}	

TP Id	TP/LDS/H/07
Test objective	Checks that IUT performs correctly if EF.SOD contains RSASSA-PKCS1_v15 with SHA512, SHA512 DG hash, DS stored inside SOD
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H07
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing Signature algorithm indicating RSASSA-PKCS1_v15 with SHA256
            containing DG hashes indicating SHA512 algorithm
            containing DS
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}	

TP Id	TP/LDS/H/22
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (SignedData with illegal digestAlgorithm)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H22

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing LDS security object digest algorithm indicating MD5
            containing Digest algorithm indicating MD5
            containing Signature algorithm indicating RSASSA-PSS with MD5
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}
```

TP Id	TP/LDS/H/24
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (SignedData incorrect content type OID for id-icao-ldsSecurityObject)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H24

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing SignedData
                containing id-icao-ldsSecurityObject
                    containing OID indicating incorrect content type
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}
```

TP Id	TP/LDS/H/31
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (SignerInfo, missing digestAlgorithm)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H31

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing SignerInfo not containing digestAlgorithm
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}
```

TP Id	TP/LDS/H/32
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (SignerInfo, incorrect messageDigest attribute value)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H32

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing SignerInfo
                containing messageDigest indicating incorrect value
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}
```

TP Id	TP/LDS/H/33
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (SignerInfo, missing messageDigest attribute)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H33
Initial conditions	
with { IUT being in the "initial state" }	
Expected behaviour	
ensure that { when { standard inspection procedure is started and IUT receives an EF.SOD file containing SignerInfo not containing messageDigest attribute } then { IUT indicates 'ePassport inspection procedure successful' and IUT indicates BAC passed and IUT indicates PA failed and IUT indicates AA not performed and IUT indicates TA passed and IUT indicates CA passed and IUT indicates COM passed and IUT indicates EF.SOD failed } }	

```
with {
    IUT being in the "initial state"
}
```

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing SignerInfo not containing messageDigest attribute
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}
```

TP Id	TP/LDS/H/37
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (SignerInfo, incorrect signature)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H37
Initial conditions	
Expected behaviour	

```
with {
    IUT being in the "initial state"
}
```

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing SignerInfo
                containing Signature indicating incorrect value
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}
```

TP Id	TP/LDS/H/45
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (LDS Security Object, additional DataGroup Hash value for DG10)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H45
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing additional Hash for DG10
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}	

TP Id	TP/LDS/H/51
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (DS certificate, signature element does not match SignatureAlgorithm)
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H51
Initial conditions	

with {
    IUT being in the "initial state"
}	

Expected behaviour	

ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing DS certificate
                containing signature element indicating value not matching SignatureAlgorithm
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}	

TP Id	TP/LDS/H/54
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (DS certificate, incorrect issuer element(does not match CSCA subject value))
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H54

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing DS certificate
                containing issuer element indicating value not matching CSCA subject value
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}
```

TP Id	TP/LDS/H/63
Test objective	Checks that IUT performs correctly if EF.SOD is wrong (DS certificate, incorrect keyUsage extension (digitalSignature bit not asserted))
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H63

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing DS certificate
                containing keyUsage
                    containing digitalSignature bit indicating 'not asserted'
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA failed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and
        IUT indicates EF.SOD failed
    }
}
```

TP Id	TP/LDS/H/71
Test objective	Checks that IUT performs correctly if EF.SOD contains ECDSA with SHA256 (ANSI OID), SHA256 DG hash, DS stored inside SOD
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H71

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing Signature algorithm indicating ECDSA with SHA256
            containing DG hashes indicating SHA256 algorithm
            containing DS
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

TP Id	TP/LDS/H/75
Test objective	Checks that IUT performs correctly if EF.SOD contains two different valid signature algorithms in SOD and DS
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H75

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing SOD
                containing SignatureAlgorithm indicating signature algorithm
            containing DS
                containing SignatureAlgorithm indicating different algorithm
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

TP Id	TP/LDS/H/83
Test objective	Checks that IUT performs correctly if EF.SOD contains RSASSA-PKCS1_v15 with SHA512, SHA512 DG hash, DS stored inside SOD
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.H83
Initial conditions	
with { IUT being in the "initial state" }	
Expected behaviour	
ensure that { when { standard inspection procedure is started and IUT receives an EF.SOD file containing Signature algorithm indicating RSASSA-PKCS1_v15 with SHA256 containing DG hashes indicating SHA512 algorithm containing DS } then { IUT indicates 'ePassport inspection procedure successful' and IUT indicates BAC passed and IUT indicates PA passed and IUT indicates AA not performed and IUT indicates TA passed and IUT indicates CA passed and IUT indicates COM passed } }	

```
with {
    IUT being in the "initial state"
}
```

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.SOD file
            containing Signature algorithm indicating RSASSA-PKCS1_v15 with SHA256
            containing DG hashes indicating SHA512 algorithm
            containing DS
    }
    then {
        IUT indicates 'ePassport inspection procedure successful' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA not performed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed
    }
}
```

6.2.15 LDS_I

TP Id	TP/LDS/I/01
Test objective	Checks that IUT performs Active Authentication with wrong tag in data group
Reference	9303v2
Profile	AIP
Configuration	CFG.EAC.LDS.I01

Initial conditions

```
with {
    IUT being in the "initial state"
}
```

Expected behaviour

```
ensure that {
    when {
        standard inspection procedure is started and
        IUT receives an EF.DG15 file
            containing tag indicating wrong value
    }
    then {
        IUT indicates 'ePassport inspection procedure failed' and
        IUT indicates BAC passed and
        IUT indicates PA passed and
        IUT indicates AA failed and
        IUT indicates TA passed and
        IUT indicates CA passed and
        IUT indicates COM passed and.
        IUT indicates EF.DG15 failed
    }
}
```

7 Abstract Test Method (ATM)

This clause describes the ATM used to test ePassport Inspections Systems.

7.1 Abstract protocol tester

The abstract protocol tester used by the Geonetworking test suite is described in Figure 4. The test system will simulate valid and invalid protocol behaviour, and will analyse the reaction of the IUT.

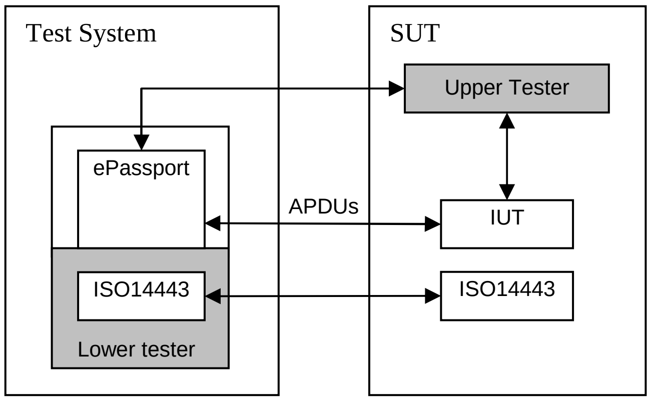


Figure 4: Abstract protocol tester - ePassport

7.2 Test configuration

This test suite uses a unique test configuration in order to cover the different test scenarios. In this configuration, the tester simulates an electronic passport placed in the communication range of the IUT.

7.3 Test architecture

Figure 5 shows the TTCN-3 test architecture used for the ePassport ATS.

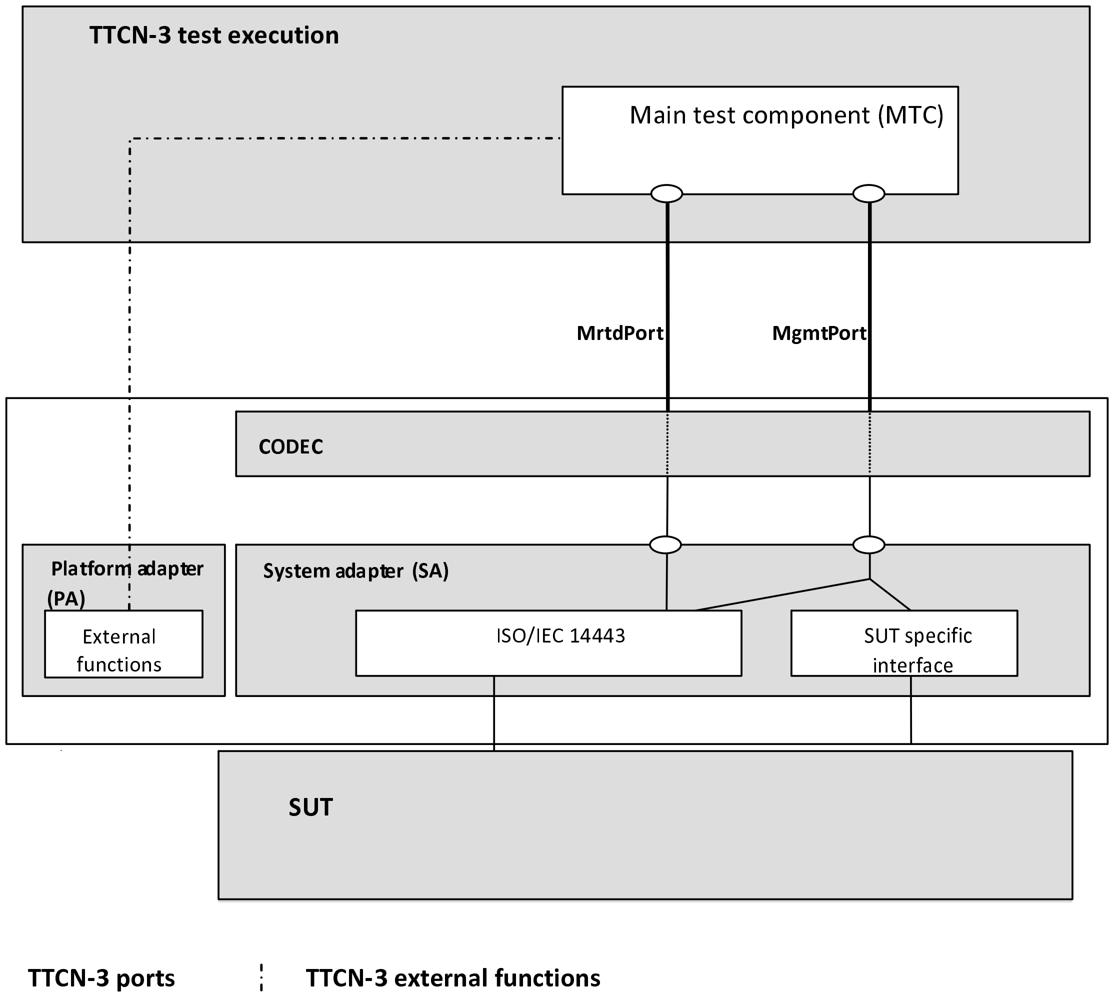


Figure 5: Test system architecture

The Test system platform software, including adaptation layer, codec, test management and security profiles needed for ePassport Readers conformance testing is publicly available as contribution **MTS(11)0044** on the ETSI portal. It can also be downloaded directly using the following link: http://docbox.etsi.org/MTS/MTS/05-CONTRIBUTIONS/2011/MTS(11)0044_DMIMTS-00127_ePassport_Prototype_Platform.zip.

7.3.1 Codec

The codec provides a translation mechanism between the abstract data structures used in the TTCN-3 code and the concrete octet representations to be exchanged on the hardware.

For the particular case of the ePassport test suite, the codec is also in charge of deciphering secured messages received from IUT so that they can be presented in their cleartext form to the TTCN-3 scripts as depicted in Figure 6, and enciphering cleartext R-APDU transmitted from TTCN-3 to IUT when necessary (Figure 7).

NOTE: The keying materials necessary for encryption/decryption and checksum computation will be set by TTCN-3 using the external function `fx_setKeysForSecureMessaging`.

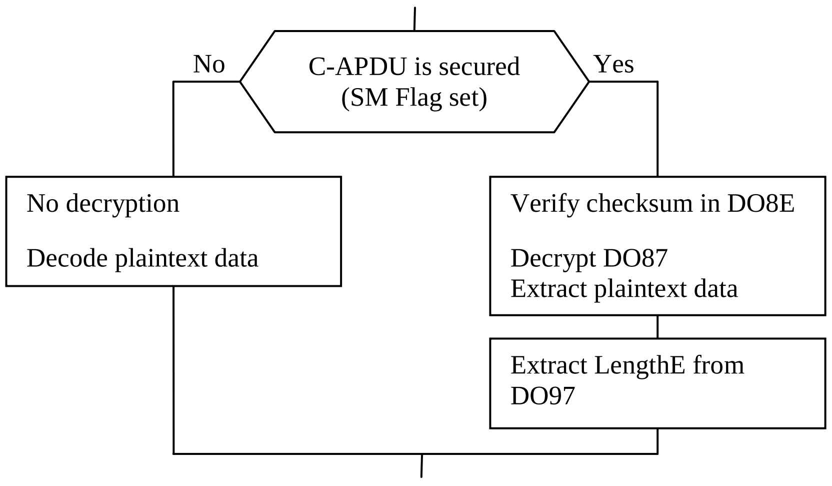


Figure 6: Decryption of C-APDU

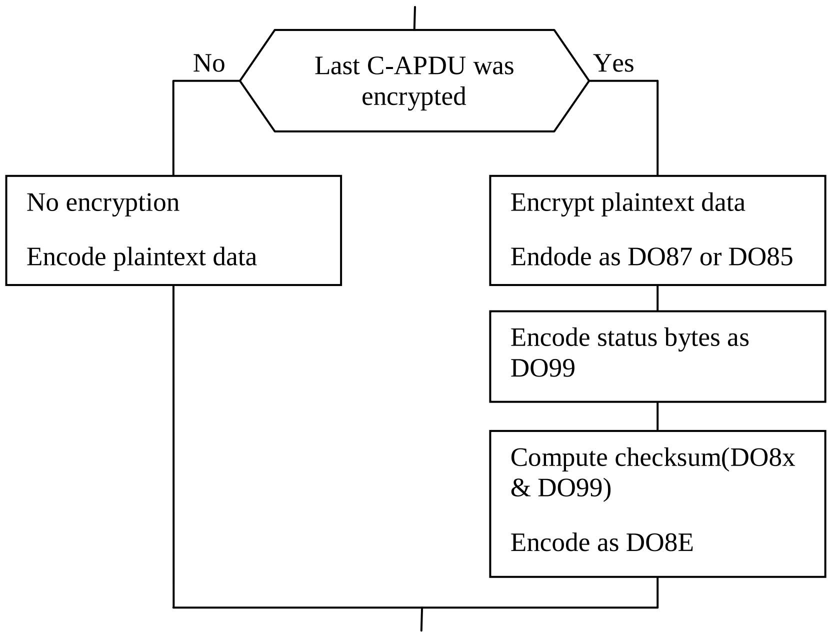


Figure 7: Encryption of R-APDU

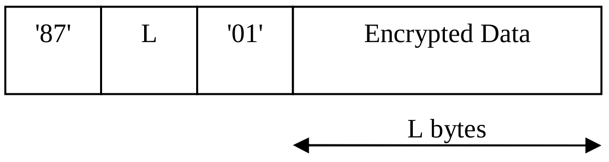


Figure 8: Encoding of DO87

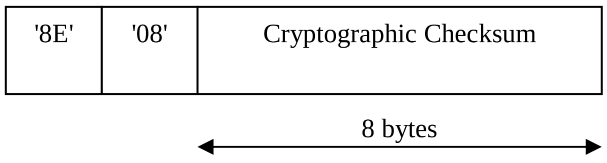


Figure 9: Encoding of DO8E

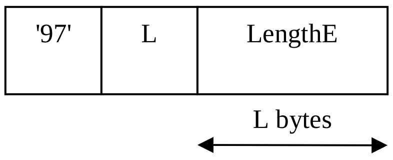


Figure 10: Encoding of DO97

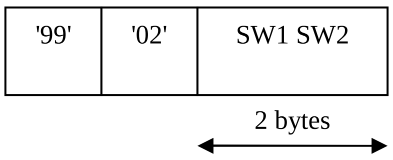


Figure 11: Encoding of DO99

7.3.2 Platform adapter

The platform adapter mainly provides external functions to TTCN-3 scripts. These functions are grouped in three categories:

- Management functions for driving the test hardware.
- Security functions for enciphering/deciphering, computing checksums, etc.
- File functions for managing passport element files.

7.3.2.1 Management external functions

Two external functions have been defined to respectively activate and deactivate the antenna of the passport simulator:

- fx_activateProbe.
- fx_deactivateProbe.

These functions do not have any parameters, and are usually called ant the beginning and at the end of each testcase.

7.3.2.2 Security external functions

Security functions represents the most critical part of the platform adapter. They can be regrouped in six categories:

- Secure Messaging related functions:
  - `fx_setKeysForSecureMessaging` - This function can be used for setting encryption and authentication keys for Secure Messaging.
  - `fx_setInitialSscForMessageAuthentication` - This function sets the initial value of the SSC.
  - `fx_setIncrementSsc` - This function function is called with parameter set to false to prevent incrementation of SSC counter after checksum computation (default behaviour is to increment it).
  - `fx_setIncludeStatusBytes` - This function, if called with parameter set to false, requires the Codec to behave incorrectly by not to including DO99 (secured status bytes) when encoding a secured R-APDU.
- Keying related functions:
  - `fx_deriveKeySeedFromMRZ` - This function is used to extract key seed from optical MRZ.
  - `fx_deriveBasicAccessKeysFromKeySeed` - This function derives Kenc (KencA+KencB) and Kmac (KmacA+KmacB) keys from key seed.
- Encryption related functions:
  - `fx_encrypt3Des` - This function encrypts provided data using 3DES algorithm.
  - `fx_decrypt3Des` - This function deciphers data using 3DES algorithm.
  - `fx_encrypt` - This function encrypts data using provided algorithm.
  - `fx_decrypt` - This function deciphers data using provided algorithm.
- Authentication related functions:
  - `fx_cryptographicChecksum` - This function computes a cryptographic checksum. In addition it maintains a persistent counter SSC and increment it for each call.
  - `fx_digest` - This function a message digest using specified hash algorithm.
- Certificate related functions:
  - `fx_setSignatureVerificationParameters` - This function sets the domain parameters for signature verification.
  - `fx_verifySignature` - This function verifies the digital signature of a message.
  - `fx_computeSignature` - This function computes the digital signature of a message.
  - `fx_verifyCertificate` - This function checks whether a given certificate is trusted by a CA.
  - `fx_extractPublicKey` - This function extracts the public key information of a certificate.
  - `fx_extractXcoordinateFromEcPublicKey` - This function extracts the X coordinate of an Elliptic-Curve public key.
  - `fx_computeDhSharedSecret` - This function generates a Diffie-Hellman shared secret from a private key and a peer public key.
- Support functions:
  - `fx_random` - This function generates an random number.

- `fx_randomOctetstring` - This function generates a random octetstring.

The following encryption algorithms are supported by the security external functions:

- Rsa: RSA/None/NoPadding format with PKCS#8 private key format.
- DESede: Triple DES Encryption (also known as DES-EDE, 3DES, or Triple-DES). Data is encrypted using the DES algorithm three separate times. It is first encrypted using the first subkey, then decrypted with the second subkey, and encrypted with the third subkey.
- DH: Diffie-Hellman key agreement algorithm suite for Chip Authentication.
- EcDH: Elliptic Curve Diffie-Hellman as defined in ANSI X9.63 [i.16] and as described in RFC 3278 [i.17].

Digital signature algorithms are also supported:

- MD2withRSA: MD2 with RSA encryption signature algorithm which uses the MD2 digest algorithm and RSA to create and verify RSA signatures as defined in PKCS;
- MD5withRSA: MD5 with RSA encryption signature algorithm which uses the MD5 digest algorithm and RSA to create and verify RSA signatures as defined in PKCS;
- SHA1withRSA: The signature algorithm with SH-1 and the RSA encryption algorithm as defined in OSI Interoperability Workshop, using padding convention described in PKCS#1;
- SHA224withRSA: The signature algorithm with SH-224 and the RSA encryption algorithm as defined in OSI Interoperability Workshop, using padding convention described in PKCS#1;
- SHA256withRSA: The signature algorithm with SH-256 and the RSA encryption algorithm as defined in OSI Interoperability Workshop, using padding convention described in PKCS#1;
- SHA384withRSA: The signature algorithm with SH-284 and the RSA encryption algorithm as defined in OSI Interoperability Workshop, using padding convention described in PKCS#1;
- SHA512withRSA: The signature algorithm with SH-512 and the RSA encryption algorithm as defined in OSI Interoperability Workshop, using padding convention described in PKCS#1;
- SHA1withDSA: The DSA with SH-1 signature algorithm which uses the SH-1 digest algorithm and DSA to create and verify DSA signature as defined in FIPS PUB 186;
- SHA1withECDSA: ECDSA with the SHA-1 family of digest algorithms;
- SHA224withECDSA: ECDSA with the both SHA-1 and SH-2 family of digest algorithms;
- SHA256withECDSA: ECDSA with the SHA-256 family of digest algorithms;
- SHA384withECDSA: ECDSA with the SHA-384 family of digest algorithms;
- SHA512withECDSA: ECDSA with the SHA-512 family of digest algorithms.

7.3.2.3 File external functions

The test specification defines configuration sets named profiles in the present document. Four external functions have been defined to manipulate element files (EFs) of the simulated passport:

- `fx_loadPassportConfiguration` - This function has to be called at the beginning of each testcase. It will select a particular EF profile and load all EFs in a cache.
- `fx_readFileData` - This function can be used to retrieve data of a cached EF. Status codes listed in Table 3 are returned.
- `fx_readCertificateData` - This function returns the content of any certificate file selected by its full path.
- `fx_createDg` - This function gives the possibility to create a new EF cache entry.

Table 3: Status codes for fx_readFileData

Status code	Meaning
0x9000	Success
0x6200	Error - No information
0x6282	End of file reached before reading Ne bytes
0x6284	File control information not formatted properly
0x6A82	File not found
0x6A88	Referenced data or reference data not found
0x6D00	Offset beyond EOF

Figure 12 depicts the disk organization proposed to store EFs. Each directory should contains one or more EF as EF.COM, EF.SOD, EF.DG<n>. The physical naming convention for EFs is described below:

1) EF_COM.bin: Common Data Elements

2) EF_SOD.bin: LDS Security Data

3) EF_DG01.bin: MRZ information

4) EF_DG02.bin: Encoded face Image

5) EF_<hh>.bin

Using this disk organization provides the advantages to easily map the configuration identifier provided in test purpose to a disk path. For instance EFs of configuration profile CFG.EAC.LDS.H32 will be found in folder /CFG/EAC/LDS/H32.

In addition files contained in the default profile corresponding to a specific testcase configuration will also be loaded. Thus, for testcase TC_LDS_H32, EF contained in folder /CFG/EAC/LDS/H32 and CFG/DFLT/EAC will be cached and used. If the two folders contain EFs with identical names, the more specific one (in this case /CFG/EAC/LDS/H32) will be loaded.

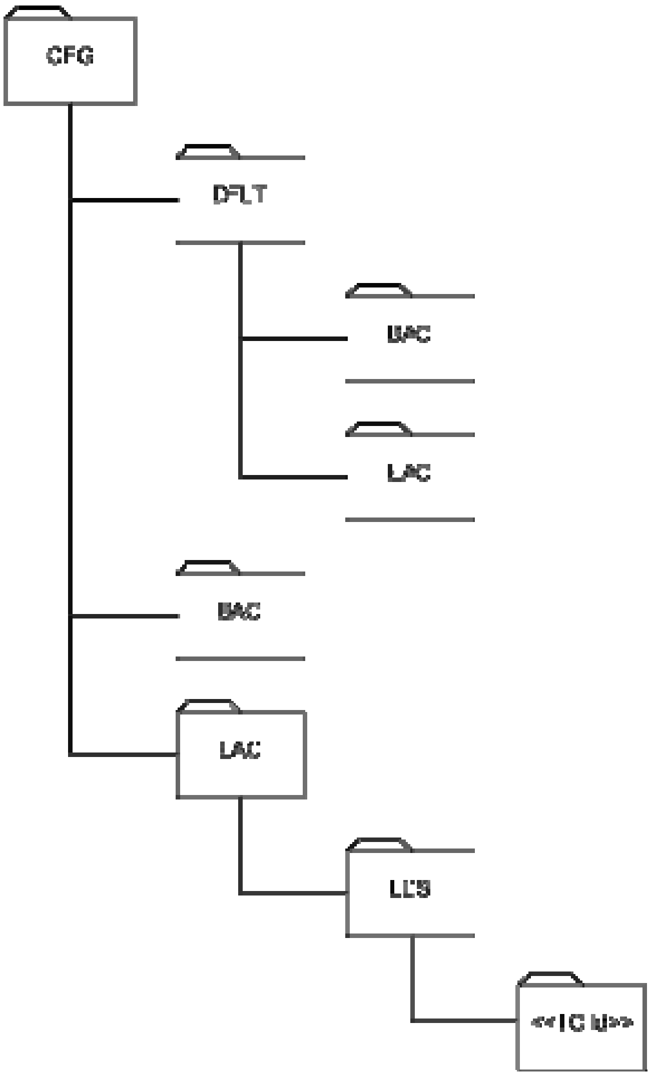


Figure 12: Disk organization of EFs

7.3.3 System adapter

The System Adapter is in charge of transmitting encoded protocol messages exchanged between Test Execution and SUT via MrtdPort, using lower layer. To achieve this task, the Sytem Adapter manages and drives an ISO/IEC 14443 [i.13] compatible contactless smart card simulator.

In addition, the System Adapter is able to handle management messages exchanged via MgmtPort, including IUT verdict retrieval. SA provides at least two implementation of MgmtPort:

- One supporting automatic Automatic Interface Specification, as described in [i.5]. In this case both protocol messages and management messages will be transmitted using ISO/IEC 14443 [i.13] layer. Messages will be distinguished based on their logical channel (bits b1-b0 of the CLASS byte): logical channel 0 for protocol messages; logical channel 1 for management messages.

- One providing a graphical user interface giving the possibility to test operator to manually enter inspection verdicts reported by IUT.

The choice of implementation is driven by module parameter PXT_AUTOMATIC_TEST_INTERFACE.

7.4 Ports and Abstract Services Primitives

Two ports are used by the ePassport ATS:

- The `MrtdPort` which is the protocol port used to receive command APDUs and send response APDUs.
- The `MgmtPort` which is used to configure IUT by sending certificates and/or certificate references. The IUT can also provide inspection verdicts via this port. The ISO/IEC 14443 [i.13] radio layer can be used to implement this port if the IUT supports the automatic test interface described in [i.5].

7.4.1 Primitives of MrtdPort

The primitives supported by the MrtdPort are described in Table 4. They correspond to smartcard messages defined in [i.12].

Table 4: Primitives of MrtdPort

Primitive	Direction
`CommandDeactivateFile`	in
`CommandEraseRecords`	in
`CommandEraseBinary`	in
`CommandPerformScqlOperation`	in
`CommandPerformTransactionOperation`	in
`CommandPerformUserOperation`	in
`CommandVerify`	in
`CommandManageSecurityEnvironment`	in
`CommandChangeReferenceData`	in
`CommandDisableVerificationRequirement`	in
`CommandEnableVerificationRequirement`	in
`CommandPerformSecurityOperation`	in
`CommandResetRetryCounter`	in
`CommandActivateFile`	in
`CommandGenerateAsymmetricKeyPair`	in
`CommandManageChannel`	in
`CommandExternalOrMutualAuthenticate`	in
`CommandGetChallenge`	in
`CommandGeneralAuthenticate`	in
`CommandInternalAuthenticate`	in
`CommandSearchBinary`	in
`CommandSearchRecord`	in
`CommandSelect`	in
`CommandReadBinary`	in
`CommandReadBinaryWithOffsetDataObject`	in
`CommandReadRecords`	in
`CommandGetResponse`	in
`CommandEnvelope`	in
`CommandGetData`	in
`CommandWriteBinary`	in
`CommandWriteRecord`	in
`CommandUpdateBinary`	in
`CommandPutDataWith`	in
`CommandUpdateRecord`	in
`CommandCreateFile`	in
`CommandAppendRecord`	in
`CommandDeleteFile`	in
`CommandTerminateDF`	in
`CommandTerminateCardUsage`	in
`CommandGeneric`	in
`Response`	out

7.4.2 Primitives of MgmtPort

The primitives supported by the MrtdPort are described in Table 5. They correspond to smartcard messages defined in [i.12].

Table 5: Primitives of MrtdPort

Primitive	Direction
`CommandGetData`	in
`CommandPutDataWith`	in
`CommandReadBinary`	in
`CommandSelect`	in
`Response`	out

8 ATS conventions

The ATS conventions are intended to give a better understanding of the ATS but they also describe the conventions made for the development of the ATS. These conventions should be considered during any later maintenance or further development of the ATS.

The ATS conventions contain two clauses, the testing conventions and the naming conventions. The testing conventions describe the functional structure of the ATS. The naming conventions describe the structure of the naming of all ATS elements.

To define the ATS, the guidelines of the document ETS 300 406 [i.6] was considered.

8.1 Testing conventions

8.1.1 Initial state

All test cases start with the function f_cfUp which sets up the test system, by mapping the port and activating defaults. Then the function f_initializeMRTD is called. This function initialize the test system by loading the appropriate profile and initializing internal variables. Finally, the IUT is initialized using the function f_initializeIS. This function optionally sets up the IUT with necessary certificates and activates the antenna so that inspection procedure can start.

8.1.2 Final state

All test cases end with the function f_cleanUp. This function brings the IUT back in an "idle" state by deactivating the antenna. This function is also in charge of unmapping the ports.

8.2 Naming conventions

8.2.1 General guidelines

The naming convention is based on the following underlying principles:

- in most cases, identifiers should be prefixed with a short alphabetic string (specified in Table 6) indicating the type of TTCN-3 element it represents;
- suffixes should not be used except in those specific cases identified in Table 2;
- prefixes and suffixes should be separated from the body of the identifier with an underscore ("_");

EXAMPLE 1: `c_sixteen, t_wait`.

- only module names, data type names and module parameters should begin with an upper-case letter. All other names (i.e. the part of the identifier following the prefix) should begin with a lower-case letter;
- the start of second and subsequent words in an identifier should be indicated by capitalizing the first character. Underscores should not be used for this purpose.

EXAMPLE 2: `f_initialState`.

Table 6 specifies the naming guidelines for each element of the TTCN-3 language indicating the recommended prefix, suffixes (if any) and capitalization.

Table 6: ETSI TTCN-3 generic naming conventions

Language element	Naming convention	Prefix	Example identifier
Module	Use upper-case initial letter	none	IPv6Templates
Group within a module	Use lower-case initial letter	none	messageGroup
Data type	Use upper-case initial letter	none	SetupContents
Message template	Use lower-case initial letter	m_	m_setupInit
Message template with wildcard or matching expression	Use lower-case initial letters	mw_	mw_anyUserReply
Signature template	Use lower-case initial letter	s_	s_callSignature
Port instance	Use lower-case initial letter	none	signallingPort
Test component instance	Use lower-case initial letter	none	userTerminal
Constant	Use lower-case initial letter	c_	c_maxRetransmission
Constant (defined within component type)	Use lower-case initial letter	cc_	cc_minDuration
External constant	Use lower-case initial letter	cx_	cx_macId
Function	Use lower-case initial letter	f_	f_authentication()
External function	Use lower-case initial letter	fx_	fx_calculateLength()
Altstep (incl. Default)	Use lower-case initial letter	a_	a_receiveSetup()
Test case	Use ETSI numbering	TC_	TC_COR_0009_47_ND
Variable (local)	Use lower-case initial letter	v_	v_macId
Variable (defined within a component type)	Use lower-case initial letters	vc_	vc_systemName
Timer (local)	Use lower-case initial letter	t_	t_wait
Timer (defined within a component)	Use lower-case initial letters	tc_	tc_authMin
Module parameters for PICS	Use all upper case letters	PICS_	PICS_DOOROPEN
Module parameters for other parameters	Use all upper case letters	PX_	PX_TESTER_STATION_ID
Formal Parameters	Use lower-case initial letter	p_	p_macId
Enumerated Values	Use lower-case initial letter	e_	e_syncOk

8.2.2 ePassport specific TTCN-3 naming conventions

Next to such general naming conventions, Table 7 shows specific naming conventions that apply to the ePassport TTCN-3 test suite.

Table 7: ePassport specific TTCN-3 naming conventions

Language element	Naming convention	Prefix
ePassport Module	Use upper-case initial letter	ePassport_
Module containing types	Use upper-case initial letter	ePassport_Types
Module containing values	Use upper-case initial letter	ePassport_Values
Module containing templates	Use upper-case initial letter	ePassport_Templates
Module containing test cases	Use upper-case initial letter	ePassport_Testcases
Module containing functions	Use upper-case initial letter	ePassport_Functions
Module containing components, ports and message definitions	Use upper-case initial letter	ePassport_ TestSystem
Module containing the control part	Use upper-case initial letter	ePassport_MainModule

8.2.3 Usage of Log statements

All TTCN-3 log statements use the following format using the same order:

- Four asterisks.
- The TTCN-3 test case or function identifier in which the log statement is defined.
- One of the categories of log: INFO, WARNING, ERROR, PASS, FAIL, INCONC, TIMEOUT.
- Free text.
- Four asterisks.

EXAMPLE 1:

```
log("**** TC_ISO7816_A02: Pass: Inspection System produced expected result
****");
```

Furthermore, the following rules are applied for the Geonetworking ATS:

- Log statements are used in the body of the functions, so that invocation of functions are visible in the test logs.
- All TTCN-3 setverdict statement are combined (as defined in TTCN-3 v3.4.1) with a log statement following the same above rules (see example).

EXAMPLE 2:

```
setverdict(pass, "**** TC_ISO7816_A02: PASS: Inspection System produced
expected result ****");
```

8.2.4 Test Case identifiers

Table 8 shows the test case naming convention, which follows the same naming convention as the test purposes.

Table 8: TC naming convention

Identifier:	**TC_<gr>_<sgr><nn>**		
	<gr> = group	ISO7816	Application protocol
		LDS	Logical data structure
	<sgr> =sub- group	ISO7816_A	Application selection
		ISO7816_B	Basic access control
		ISO7816_C	Reading binary files
		ISO7816_D	Chip authentication
		ISO7816_E	Terminal authentication
		ISO7816_F	Active authentication
		LDS_A	EF.COM
		LDS_B	EF.DG1
		LDS_C	EF.DG2
		LDS_D	EF.DG3
		LDS_E	EF.DG4
		LDS_F	EF.DG14
		LDS_G	EF.CVCA
		LDS_H	EF.SOD
		LDS_I	EF.DG15
	<nn> = sequential number		01 to 99

EXAMPLE: TP identifier: TP/ISO7816/A/02
TC identifier: TC_ISO7816_A02

8.3 PIXIT

Depending of the test campaign, the PIXITS may have to be modified.

Here is the list of PIXITS available:

Test Adapter Pixits:

- PXT_DRIVER_PROVIDER_CLASS_NAME: Name of the Adapter class for ePassport simulator hardware. For Comprion CLT One hardware, use "org.etsi.epassport.comprion.ComprionImplFactory".
- PXT_IS_DRIVER_PROVIDER_CLASS_NAME: Name of the Adapter class for ePassport Automatic interface hardware. For Comprion CLT One hardware, use "org.etsi.epassport.comprion.ComprionImplFactory". If IUT does not support Automatic interface, use: "org.etsi.epassport.grt.GRTImplFactory".
- PXT_IS_DRIVER_IP: Vendor IP address for the IS equipment (Remote IP;Remote port;Local port).
- For the prototype, use: "127.0.0.1;5000;5001".
- PXT_DEBUG_MODE: Activate debug mode (additional logs in the console, from Test Adpater and Codec).

Common Timer Pixits:

- PXT_TIMER_PRECISION: Precision of timers in percentage (default is 5 percent).
- PXT_TAC: Guard timer to control a reaction from the IUT to a stimulus sent by the tester (e.g. a message). On expiry of this timer, the IUT is considered not to be be able to send the expected response.
- PXT_TNOAC: Guard timer to control a non-reaction from the IUT to a stimulus sent by the tester (e.g. a message). On expiry of this timer, it is considered that, as it is expected in the test purpose, the IUT has not responded to the stimulus.
- PXT_TWAIT: Wait for an implicit send. This guard timer is used to limitated the time where the tester is waiting for the response of the IUT that is triggered by an action from the test operator. On expiry of this timer, it is considered that the action has not succeeded, and thus the test case will be terminated with the verdict inconclusive.
- PXT_TGUARD: This guard timer is used to control the timeout of a test case. If, e.g. an external function does not return, then this timer will fire after PXT_TGUARD seconds. On expiry of this timer, it is considered that the test case did not finish within the given test cae life time, and will therefore be terminated.

ePassport Pixits:

- PXT_EPASSPORT_DATA_ROOT: Root directory containing ePassport profiles (EF.*, MRZ, etc.).

IUT Pixits:

- PXT_AUTOMATIC_TEST_INTERFACE: Determine whether IUT supports Automatic Test Interface (see PXT_IS_DRIVER_PROVIDER_CLASS_NAME).
- PXT_VERDICT_MODE: Does Iut report detailed or simple inspection verdicts.

8.4 Online documentation

Using the T3D tool enables providing on-line documentation browser in HTML, by tagging TTCN-3 comments. These tags are defined in Table 9.

Table 9: TTCN-3 comment tags

Tag	Description
@author	Specifies the names of the authors or an authoring organization which either has created or is maintaining a particular piece of TTCN-3 code.
@desc	Describes the purpose of a particular piece of TTCN-3 code. The description should be concise yet informative and describe the function and use of the construct.
@remark	Adds extra information, such as the highlighting of a particular feature or aspect not covered in the description.
@see	Refers to other TTCN-3 definitions in the same or another module.
@return	Provides additional information on the value returned by a given function.
@param	Documents the parameters of parameterized TTCN-3 definitions.
@version	States the version of a particular piece of TTCN-3 code.

The HTML files result from the compilation of the TTCN-3 modules with the T3D tool. These HTML files are ready for browsing, and contain links enabling to navigate through the ATS.

EXAMPLE:

```
/**
 * @shortdesc   SM failure - secured status bytes missing
 * @desc        This test verifies that the inspection system recognizes an
 *              incorrect R-APDU in first secure messaging command. Perform
 *              standard inspection procedure and read BAC protected data
 *              groups from the lower tester.
 * @purpose     TP/ISO7816/B/07
 * @config      Configuration file: default BAC
 *              Profile: SIP
 * @verdict     pass  "ePassport inspection procedure failed"
 *                    Interface: BAC: FAIL, PA: NP, AA: NP, TA: NP, CA: NP,
 *                               COM: FAIL
```

9 Validation Report

9.1 First Validation Campaign

9.1.1 Introduction

The first Validation session of the STF 400 took place on week 46 (15 Nov-19 Nov 2010) in the ETSI premises.

The STF experts who have participated to the validation campaign was Zdenek Riha as evaluator with the technical support of Alexandre Berge, Yann Garcia and Laurent Velez.

The team experts have used 3 Inspection systems for this first validation campaign of the ePassport Reader Test system prototype.

The HW and SW system developed by the STF 400 aimed at conformity testing of inspection system is called just as **THE SYSTEM** in the present document.

The system consists of chip card simulator supplied by Comprion (CLT One, Version 1.0, SN: 21019 together with antennas Comprion Antenna - C- Type01, 200210105, #550114, V1.0, R0.1 and Comprion Antenna - M - 5x36x21 - Type02, 20020202, 550067 V1.0 R01a) and the SW part consisting of TTworkbench Basic (Version 1.1.10, BuildID 2009.12.14.17.16) and relevant configurations, campaigns, scripts, libraries and chip/testcase profiles.

On the side of the inspection systems the RFID reader ACG (ACG id, ACGPass V1.1, RDHS-0404D1-01, firmware: Dual 2.3.1 with JRC inventory number 01RI 2007 02090 17), partly also ACR122 (PN: ACR122U-WB-R, SN: 035-002294 with MU inventory number DHM 281144) with the Golden Reader Tool (GRT) 2.9.4 was used.

Moreover, a Simple Inspection System (evaluator's program written in C/C++) and one real-world inspection system - ARH PRMC233RL092850 together with "Full Page Reader Demo 2.2.2.3" were used.

9.1.2 Evaluation Notes

1) The system was configured and presented to the evaluator. A demonstrative testcase ISO7816_A_02 was working correctly with GRT 2.9.4 running on the same computer and with the upper tester in the form of a clickable graphical user interface program where the result is manually entered.

2) The system is not able to run a sequence of testcases in a single run (campaign). The Comprion HW needs a separate reset before each testcase. The SW is not doing it automatically, so at this moment this needs to be done manually by reloading the test campaign after a single test case.

   ➔ Fixed during the session.

3) After the system has been demonstrated as working with a configured GRT running on the same computer as the system the evaluator tried to setup another computer with the same version of GRT. To be able to setup the GRT a set of certificates was need. It turned out that the (CSCA and TA) certificates are not well organized; there are several sets of certificates with the same filenames and several certificates with the same properties to be used with the same testcases. The certificates need to be well organized, so that it is clear what certificates to use on the side of the IS for each testcase and what configuration/profile files to use on the side of the system to emulate a passport chip.

   ➔ **Pending: Certificates need to be better organized.**

4) There are no CRL files available. Although the test specifications do not explicitly mention the CRLs, real world inspections systems need CRL files for correct operation of their passive authentication procedures.

   ➔ **Pending:** CRLs for CS CAs need to be generated. As they can be valid maximally 90 days a solution should be found (setting a fixed date at the IS or freshly generating CRLs or generating CRLs for the future.

5) Available CSCA and DS certificates were examined. See below for more details. It was found that the validity of the DS certificates is only 1 year (until April 2011). That means the DS certificate will expire before the expiration of the document in 2013. According to section 9.3 of ICAO 9303 [i.4], part 1 volume 2 this is not correct.

   ➔ **Pending:** The DS certificates have to be regenerated with a longer lifetime. That will imply a change of all EF.SOD files of all configurations. We would recommend not using the same Issuer/Subject Name and Serial Number for certificates of different types (PKCS1, PSS, ECDSA). This is concerning both the CSCA and DS certificates. The certificates issuer name should also be ETSI.

6) The TA certificates were examined (see below for details). The provided DV and IS certificates are expired. The test specification does not mention usage of particular days. The specification labels the CVCA effective and expiration date as CVCAeff and CVCAexp. The validity of DV_Cert_01 and IS_Cert_01 is not defined; it should only be consistent with the EAC specs. The validity of IS_Cert_11 is set to "CVCAeff" - "CVCAeff+14 days" and the certificate is made wrong (signature corrupted). The modified IS_Cert_11 is used in one of the selected test cases (E_18).

   ➔ **Pending:** As real inspection systems will examine time validity of the supplied certificates, it can happen that real inspection systems will refuse to work with the provided test certificates.

   This can be solved by fresh generation of TA certificates. Ideally all the CVCA, DV and IS certificates would be generated. But that would require update of the profiles for simulated passports. Easier/simpler implementation would make the CVCA certificates valid for 3 years (2010-2013) and then generate only fresh DV/IS certificates to be provided to the tested inspection system. The tool for generation of DV/IS certificates would be available and be part of the system. For DV_Cert_01 and IS_Cert_01 that's perfectly ok, how to deal with the modified IS_Cert_11 is to be discussed.

   Another solution would be to provide a fixed set of test certificate together with a fixed data (e.g. March 20, 2010). The inspection system then needs to be set to that fixed date that has already passed. The question is whether this is possible with all real world inspection system. The problem of the time validity of TA certificates needs to be discussed with STF 400 experts.

7) During the further tests we again came to the problem of confusion of the certificates. The certificates and profile files are different for the same test case located in 2 different folders: the "default" and special. This is problematic already for the first test case ISO7816_A_02.

8) The evaluator managed to find the right certificates and made the testcases ISO7816_A_02 and ISO7816_F_05 working on the computer of the evaluator running the GRT 2.9.4 and the ACG reader. (Later a problem with the implementation of the ISO7816_F_05 was discovered but that's another issue).

9) In addition to the ACG reader the evaluator tested also ACR122 reader, but communication errors were occurring from time to time. The ACG reader on the other hand worked perfectly with the Comprion HW and there were no communication errors during the whole evaluation session. The perfect communication compatibility of the Comprion and ACG HW is a big success and the choice of the Comprion HW was certainly a very good decision. The card simulator is one of the key elements of the whole system and Comprion HW contributed significantly to the success of the project.

10) The evaluator also used own software working as a simple inspection system to be able to evaluate every single detail of the system and have a full control of the inspection procedure. During the initial tests of the simple inspection system it turned out that the system does not correctly accept APDUs with Le set to 0.

   ➔ This was corrected later in the session.

11) During further testing it turned out that the system does not handle well (long) file identifiers. While the short file identifiers are handled correctly (and the GRT uses short file identifiers by default), the long ones were not working as expected. The problem was easily demonstrated both with GRT (configured not to use short file identifiers) and with the simple inspection system of the evaluator.

   ➔ It turned out that the long file identifiers were working in principle, but the EF.COM file was configured in internal tables with an incorrect file identifier which let to selection of another file (EF.SOD) and the inspection system reported an error as the structure of the file was wrong. This was corrected easily and quickly.

12) During testing of chip authentication it turned out that the secure messaging of the system is implemented with a serious error. Anytime the secured SM response was to send, but no data was included in the response, the securing was not done. So in all the situations where the DO 99 with secure SW and MAC in DO 8E was to be send this all was omitted and only the unsecured status word (SW) was sent. The secured response with data (with DO 87) was not affected with such an implementation error. This was a serious problem, but what is very interesting this issue was not discovered at all by the GRT. Even the evaluator's simple inspection system did not reveal that in all commands, but only during the testing of chip authentication this was revealed.

   ➔ This error was corrected later and the MAC is correctly sent together with the secured DO99 now.

13) During a more detailed investigation of testcase ISO7816_F_05 it turned out that the testcase is not testing what it should. The ISO7816_F_05 is a positive testcase aimed at verification of the a bit unusual AA setting (using sha-265 as the hash algorithm). Unfortunately both the DG15 and the AA implementation in the system is using sha-1, which transforms this testcase into a normal AA (i.e. not testing what should be tested here).

   ➔ **Pending:** This still needs to be fixed. Whether the DG15 file should code sha-256 can be subject to discussion, in any case the implementation of the AA in the system needs to be changed to use sha-256.

14) During certain phases of testing of ISO7816_F_05 the AA was not performed at all. It turned out that the configuration used an incorrect EF.COM file not coding the presence of DG15.

   ➔ This was corrected quickly.

15) After testing basic positive testcases with BAC and EAC (ISO7816_A_02 for BAC and ISO7816_F_05 for EAC with AA - still AA not fully correct for F_05) the focus was moved to using the automated interface.

   The automated interface is an important feature of the project. It is able to automate fully the tests, and so remove the burden of significant human intervention during the whole testing process. The standard manual reporting interface is a clickable UI with a set of buttons consistent with the error codes mentioned in the test specification and in the automated interface specification. So after each testcase the operator clicks on buttons according to the result of the inspection procedure as indicated in the human graphical interface of the inspection system. The automated interface is sending the same results but via the communication channel with the smartcard (simulator). In addition to that the automated interface is able to provide additional information for the inspection system (certs and mrz).

At first reporting the result of the inspection procedure was tested. The test has shown that the automated interface does not work. The result/communication was not expected at the mrtdport, only at the mgmtport, i.e. coming from the clickable user interface and therefore the automated reporting did not work.

➔ This was corrected, now the verdict of the inspection procedure can be reported via the automated interface.

16) Certain modifications of the system led to the situation that after the verification of the expected result and reporting of the final verdict of the system a further running code of the system failed and an exception was reported.

➔ It was corrected later in the session.

17) The next step in automated interface testing was getting the MRZ from the system. At the beginning of the inspection procedure the inspection system asks the system (in a means of a specially crafted APDU) the MRZ to be used during the test case. This can automate the testing and it simulates insertion of the document into the inspection system and starts the inspection procedure. When tested the GET DATA APDU to get the MRZ did not work. Although implemented and having the mrz data in the profiles the get data command did not work as expected.

➔ The debugging showed where the problem was and the issue was corrected. It is now possible to get the mrz from the system via the automated interface.

18) Next the testing focused on the L6 ISO7816 test cases other than initially used ISO7816_A_02 and ISO7816_F_05. It turned out that the test cases ISO7816_B07 and ISO7816_B08 are not yet fully implemented.

➔This was corrected later in the session.

19) Next testing showed that the test case ISO7816_C_02 cannot be performed because reading large files with the odd instruction ReadBinary (B1) command is not yet implemented.

➔This has been fixed later in the session.

20) The following quick review of the current situation showed that the testcase ISO7816_D_11 seems to be ok, but ISO7816_D_04 does not work because the keyID option of the chip authentication is not yet implemented in the system.

➔ This has been fixed later in the session.

21) The comparison of the expected result with the reported result at the end of each test case in the system is done quite strictly, requiring a perfect match for each of the possible result categories (like PA, AA, TA, COM, …). This is not simple as the automated interface can only report where the failures occurred (or that no failure occurred at all). It is not possible to report that a specific activity was not performed or was performed and passed (only a complete "no failure" can be reported). The system has to face the situation not receiving from the inspection system the information that for example passive authentication was performed and passed or was not performed at all and still be able to compare the results to prepare the final PASS/FAIL verdict.

➔ **Pending:** his needs to be discussed; maybe less strict comparison will have to be done.

22) Further testing had to face again the situation that the standard RSASSA-PSS CSCA certificate (RSA 3072, SHA-256) is available in 2 variants and some test cases in the current configuration of profiles need one while other need the other certificate and the situation is confusing. All certificate files and matching EF.SOD elementary files need to be better organized.

23) Testing of the test case of ISO7816_E_06 revealed that the test case is not implemented well. Maybe because of a different file name, maybe because of the changed crypto strength, the positive test case is failing on the side of the system.

➔ This has been fixed later in the session.

24) The test case ISO7816_E_8 seems to work ok, the question is whether the specification requires reasonable reported failures. The TA should fail as indicated, that's ok, but it is a question where an error in EF.CVCA should be reported as well.

   ➔ **Pending:** From the point of view of the IS, the file is syntactically correct, it only in not able to form the valid chain (0002 certificate chain is not provided to the IS). This needs to be discussed with STF 400 experts.

25) The test case ISO7816_E_11 works fine.

   ➔ **Pending:** Further discussion is necessary whether COM failure should also be reported or not. The specification might need to be updated here.

26) The test case ISO7816_E_12 works fine.

   ➔ **Pending:** Again the discussion should clarify if short challenge does not imply also communication (COM) failure.

27) The test case ISO7816_E_18 works fine. Again the discussion should clarify if the inspection system has to identify the problem with the signature and to even start the TA or to perform the TA and learn that it fail from the chip that does reject the signature. As the signature validation is very problematic on the side of the IS (requires CVCA certificates, for ECDSA with the domain parameters etc), the IS will probably start the TA and get the error SW after sending the IS certificate. That would imply the communication error, too.

   ➔ **Pending:** Further discussion with STF experts would be useful.

28) The test case ISO7816_E_28 has passed, but a bit more detailed analyses showed that incorrect DG14 file was used. The file was updates and the testing repeated. The repeated testing has confirmed that the test case is correct.

29) Next the effort was focused on the automated interface. The last part of the automated interface deals with the indication of what certificates (CSCA and TA) should be used for test cases. First of all the documentation/specification was analyzed and overview prepared with all the CSCA, DS certificates for the PA and the CVCA, DV, IS certificates and IS private keys for all selected test cases. The evaluator has prepared a table with the relevant information (see below). During the analysis it has been show that only a few certificates are indeed necessary for the selected testcases. The certificates were labeled as **CSCA:** "CSCA_RSA_PSS","CSCA_RSA_PKCS1_SHA512","CSCA_ECDSA"; **DS:** "DS_RSA_PSS","DS_RSA_PKCS1_SHA512","DS_RSA_PSS_WRONG_KU", "DS_ECDSA".; **DV:** "DV_Cert_01", "DV_Cert_08"; **IS:** "IS_Cert_01", "IS_Cert_08", "IS_Cert_11"; **IS private keys:** "IS_Key_01", "IS_Key_08". CVCA: "CVCA_Cert_01", "CVCA_Cert_08". The CSCA certificates supplied by STF team were examined and found correct, the DS certificates were examined and found ok except for the short time validity (2011) that does not exceed the validity of the document (2013).

   ➔ **Pending:** This needs to be corrected and all configurations resigned (EF.SOD).

   The TA certificates and private keys were also examined. The files were found correct, but the test case ISO7816_E_28 was incorrectly using certificate from the set 14, the specification explicitly mentions the set 01 to be used here.

   ➔ The profile for the ISO7816_E_28 test case was corrected.

   ➔ **Pending:** The situation with expiration of the TA certificates (that need to have a short validity) needs to be solved (see above).

30) When the situation with the organization of the certificates was resolved (at least theoretically, practically the files still need to be sorted out better), we could proceed to the implementation and testing of the automated interface dealing with certificate references and certificates themselves.

   First of all for each testcase a list of relevant certificates was prepared (based on the table prepared by the evaluator) and second the relevant code for the automated interface was implemented.

The automated interface is offering two options to deal with the certificates. The first option puts all the relevant certificates in a single folder which is manually transferred to the inspection system. Before each testcase, the inspection system gets (files)names of the certificates to be used, by mean of the get data APDU. An alternative approach does not transfer any folder with certificates manually, instead a special datagroup with FID 0200h is created and the datagroup is read by the inspection system before the inspection procedure to get the full certificate data to be used during the inspection procedure. Both the approaches were implemented on the side of the system and at the same time at the side of the simple instruction system of the evaluator.

➔ Both options for certificate management in automated interface were tested and found correct.

31) The automated interface specification supports a special file with all certificates together (FID 0200h), the specification mentions that all the certificates are concatenated in a special ASN.1 structure with tag 70h. The certificates should be concatenated in the form of "ASCII encoded certificate value". It is by far not clear what is meant as ASCII encoding. We decided to use the natural format of the certificates - binary DER for X.509 certificates and binary (as sent to the chip) for the TA certificates.

➔ **Pending:** The specification needs to be clarified.

32) As already mentioned above the combination of the Comprion card simulator and ACG reader works reliably. We also tried to measure the speed of the inspection procedure .We selected the testcase ISO7816_F_05 where the configuration includes long DG3 and DG4 files and AA is performed. The time needed to perform the inspection procedure was between 33 and 36 seconds which is acceptable. Surprisingly the total time does not depend much on the maximum air speed allowed on the side of the reader (106, 212, 424, 848 kbps) rising speculations that the air speed is a constant 106 kbps not allowing the reader to switch to higher speeds. Nevertheless the speed is acceptable and the total time to perform a testcase is significantly shorter for testcases less intensive on data transfer of large datagroups. Also configuring the system to suppress many of the debug messages should lead to significantly faster speed. This was, however, not yet tested.

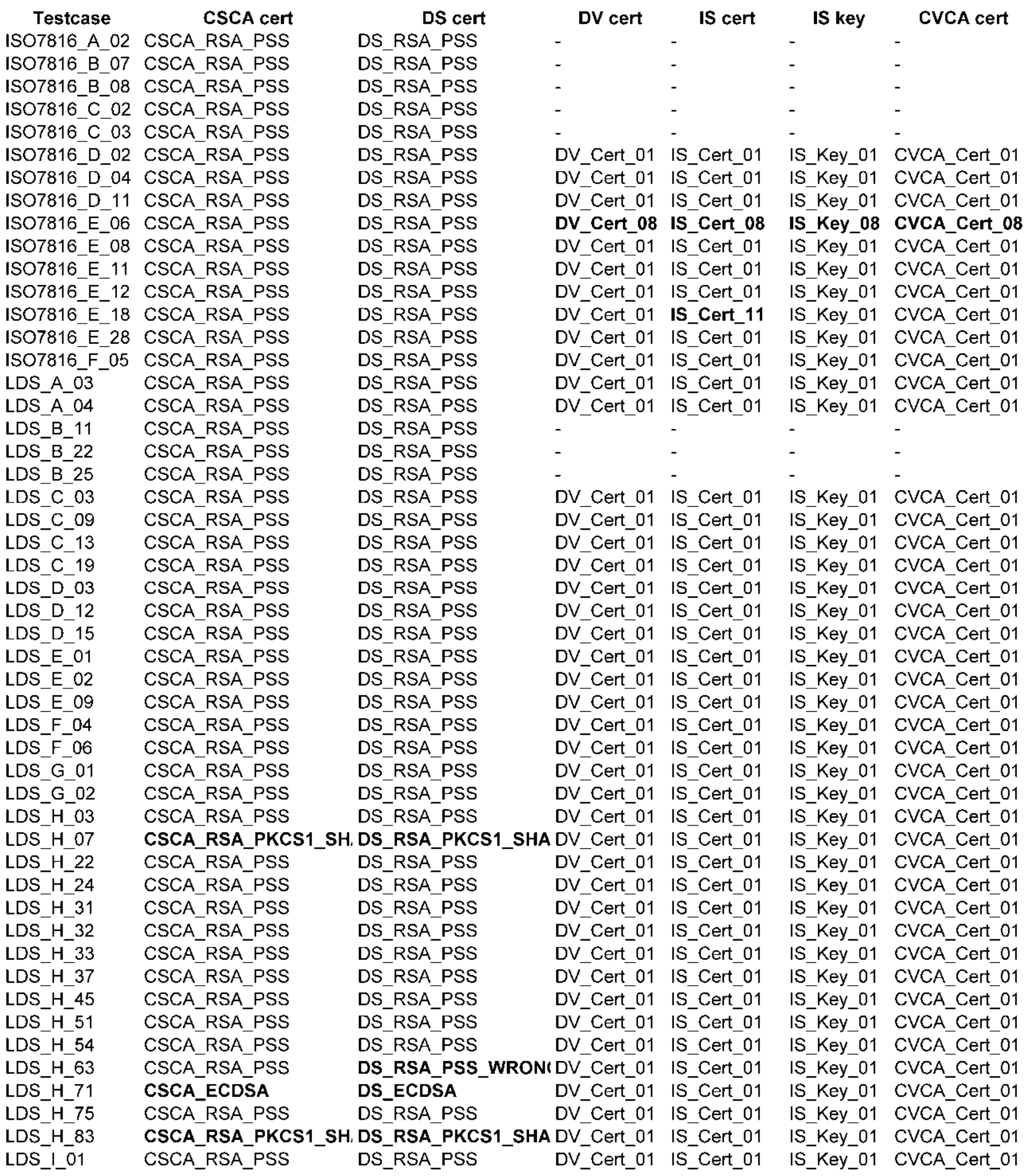
➔ **Pending:** Looking at the speed issues would be interesting once the system is stable enough from the functional point of view.

33) In addition to the ACG reader from JRC and ACR122 reader from MU on the HW level and to the GRT 2.9.4 and the simple inspection system of the evaluator on the SW level we had opportunity to test a sample of a real inspection system (HW) with a demo SW working as a border control inspection system. The full page inspection system PRMC233RL092850 was supplied by the Hungarian company ARH. The demo inspection system was "Full Page Reader Demo 2.2.2.3".

The inspection system was operating well and could be tested. The inspection system does not have an interface for configuration of certificates. The certificates are loaded into a directory and after a change of the certificates the application needs to be rerun. At the beginning the inspection system was performing BAC, AA and PA, but could not perform TA with the test TA certificates provided by STF experts. The certificates were expired (comparing with the real date). That's why the inspection system was ignoring the test certificates, could not build the certificate chain and failed during the terminal authentication. The evaluator prepared manually up-to-date DV and IS certificates and uploaded them into the certificate folder. Then the inspection system was able to perform TA.

Unfortunately the inspection system was not able to cope with the test DG4 elementary file; When reading of DG4 files was disabled, the inspection system was able to finish the inspection procedure. The inspection system is not supporting the automated interface. We tried two other test cases. The inspection system failed in ISO7816_B_07 as it was not recognizing correctly the problem in secure messaging; on the other hand the inspection system passed correctly the ISO7816_B_08 testcase, reporting secure messaging error. We have provided a feedback to the vendor about these issues.

9.1.3 Validation of Certificates



Testcase	CSCA cert	DS cert	DV cert	IS cert	IS key	CVCA cert
ISO7816_A_02	CSCA_RSA_PSS	DS_RSA_PSS	-	-	-	-
ISO7816_B_07	CSCA_RSA_PSS	DS_RSA_PSS	-	-	-	-
ISO7816_B_08	CSCA_RSA_PSS	DS_RSA_PSS	-	-	-	-
ISO7816_C_02	CSCA_RSA_PSS	DS_RSA_PSS	-	-	-	-
ISO7816_C_03	CSCA_RSA_PSS	DS_RSA_PSS	-	-	-	-
ISO7816_D_02	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
ISO7816_D_04	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
ISO7816_D_11	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
ISO7816_E_06	CSCA_RSA_PSS	DS_RSA_PSS	**DV_Cert_08**	**IS_Cert_08**	**IS_Key_08**	**CVCA_Cert_08**
ISO7816_E_08	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
ISO7816_E_11	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
ISO7816_E_12	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
ISO7816_E_18	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	**IS_Cert_11**	IS_Key_01	CVCA_Cert_01
ISO7816_E_28	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
ISO7816_F_05	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_A_03	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_A_04	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_B_11	CSCA_RSA_PSS	DS_RSA_PSS	-	-	-	-
LDS_B_22	CSCA_RSA_PSS	DS_RSA_PSS	-	-	-	-
LDS_B_25	CSCA_RSA_PSS	DS_RSA_PSS	-	-	-	-
LDS_C_03	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_C_09	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_C_13	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_C_19	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_D_03	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_D_12	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_D_15	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_E_01	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_E_02	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_E_09	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_F_04	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_F_06	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_G_01	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_G_02	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_03	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_07	**CSCA_RSA_PKCS1_SH**	**DS_RSA_PKCS1_SHA**	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_22	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_24	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_31	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_32	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_33	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_37	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_45	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_51	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_54	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_63	CSCA_RSA_PSS	**DS_RSA_PSS_WRON**	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_71	**CSCA_ECDSA**	**DS_ECDSA**	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_75	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_H_83	**CSCA_RSA_PKCS1_SH**	**DS_RSA_PKCS1_SHA**	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01
LDS_I_01	CSCA_RSA_PSS	DS_RSA_PSS	DV_Cert_01	IS_Cert_01	IS_Key_01	CVCA_Cert_01

Figure 13: The overview of certificates used in the selected test cases

9.1.4 Validation of PA certificates

1) CRLs are missing. Some real inspection systems might need fresh CRLs to perform correctly the PA.

2) The default CSCA certificate: CSCA_RSA_PSS:

   a) Version 3, SN: 01 88 40, RSA-PSS, sha256

   b) CN=HJP PB CS, OU=Country Signer, O=HJP Consulting, C=DE

   c) Issuer = Subject

   d) Validity: 15 Apr 2010 - 20 Mar 2015

   e) RSA 3072b, Key id = 61 15 07 ff …
   f) Key usage: CertSign, OffileCRL, CRL
   g) Basic constraints: CA=true, length=0

3) Special CSCA certificate: CSCA_RSA_PKCS1_SHA512:
   a) Version 3, SN: 01 88 40, sha512RSA
   b) CN=HJP PB CS, OU=Country Signer, O=HJP Consulting, C=DE
   c) Issuer = Subject
   d) Validity: 15 Apr 2010 - 20 Mar 2015
   e) RSA 3072b, Key id = e4 00 42 6f 34 …
   f) Key usage: CertSign, OffileCRL, CRL
   g) Basic constraints: CA=true, length=0

4) Special CSCA certificate: CSCA_ECDSA:
   a) Version 3, SN: 01 88 40, ECDSA sha256
   b) CN=HJP PB CS, OU=Country Signer, O=HJP Consulting, C=DE
   c) Issuer = Subject
   d) Validity: 15 Apr 2010 - 20 Mar 2015
   e) ECDSA 256b, Key id = f9 af 8f 09 93 …
   f) Key usage: CertSign, OffileCRL, CRL
   g) Basic constraints: CA=true, length=0

5) The default DS cert: DS_RSA_PSS:
   a) Version 3, SN: 01 88 3F, RSA-PSS, sha256
   b) Subject: CN=HJP PB DS, OU=Document Signer, O=HJP Consulting, C=DE
   c) Validity: 15 Apr 2010 - 10 Apr 2011
   d) RSA 2048b
   e) Key id CA: 61 15 07 ff ...
   f) Key id DS: 90 0c ae 26 ...
   g) KeyUsage: DigSig

6) Special DS certificate: DS_RSA_PKCS1_SHA512:
   a) Version 3, SN: 01 88 3F, sha512RSA
   b) Subject: CN=HJP PB DS, OU=Document Signer, O=HJP Consulting, C=DE
   c) Validity: 15 Apr 2010 - 10 Apr 2011
   d) RSA 2048b
   e) Key id CA: e4 00 42 6f ...
   f) Key id DS: 1f 6c d5 ...
   g) KeyUsage: DigSig

7) Special DS certificate: DS_RSA_PSS_WRONG_KU:
    a) Version 3, SN: 01 88 3F, RSA-PSS, sha256
    b) Subject: CN=HJP PB DS, OU=Document Signer, O=HJP Consulting, C=DE
    c) Validity: 15 Apr 2010 - 10 Apr 2011
    d) RSA 2048b
    e) Key id CA: 61 15 07 ff ...
    f) Key id DS: 90 0c ae 26 ...
    g) KeyUsage: nothing
8) Special DS certificate: DS_ECDSA:
    a) Version 3, SN: 01 88 3F, ECDSA sha256
    b) Subject: CN=HJP PB DS, OU=Document Signer, O=HJP Consulting, C=DE
    c) Validity: 15 Apr 2010 - 10 Apr 2011
    d) ECDSA 256b
    e) Key id CA: f9 af 8f 09 ...
    f) Key id DS: 3f 05 b1 66 ...
    g) KeyUsage: DigSig

The DS certificate should be valid at least until the end of validity of the document (31 Oct 2013).

9.1.5 Validation of TA certificates

1) Standard CVCA certificate: CVCA_Cert_01:
    a) DETESTCVCA00001
    b) PK: ECDSA-SHA-256 (256b)
    c) CVCA: reading DG3/DG4
    d) 15 Apr 2010 - 10 Apr 2011
    e) Signature ok
2) Special CVCA certificate: CVCA_Cert_08:
    a) DETESTCVCA0008
    b) PK: ECDSA-SHA-224 (256b)
    c) CVCA: reading DG3/DG4
    d) 15 Apr 2010 - 10 Apr 2011
    e) Signature ok
3) Standard DV certificate: DV_Cert_01:
    a) DETESTCVCA00001/DETESTDV00001
    b) PK: ECDSA-SHA-256 (256b)
    c) Domestic DV: reading DG3/DG4

   d) 15 Apr 2010 - 15 May 2010

   e) Signature ok

4) Special DV certificate: DV_Cert_08:

   a) DETESTCVCA00008/DETESTDV00008

   b) PK: ECDSA-SHA-224 (256b)

   c) Domestic DV: reading DG3/DG4

   d) 15 Apr 2010 - 15 May 2010

   e) Signature ok

5) Standard IS certificate: IS_Cert_01:

   a) DETESTDV00001/DETESTIS00001

   b) PK: ECDSA-SHA-256 (256b)

   c) IS: reading DG3/DG4

   d) 15 Apr 2010 - 29 Apr 2010

   e) Signature ok

6) Special IS certificate: IS_Cert_08:

   a) DETESTDV00008/DETESTIS00008

   b) PK: ECDSA-SHA-224 (256b)

   c) IS: reading DG3/DG4

   d) 15 Apr 2010 - 29 Apr 2010

   e) Signature ok

7) Wrong IS certificate: IS_Cert_11:

   a) DETESTDV00001/DETESTIS00001

   b) PK: ECDSA-SHA-256 (256b)

   c) IS: reading DG3/DG4

   d) 15 Apr 2010 - 29 Apr 2010

   e) Signature wrong

8) Standard IS private key: IS_Key_01:

   a) ECDSA 256 bits

9) Special IS private key: IS_Key_08:

   a) ECDSA 256 bits

TA certificate suggestion: CVCA certificates validity 3 years (maximum according to 2909/2006) from Nov 2010 to Nov 2013, DV and IS certificates generated freshly for the test session with 1 month (DV) /14 days (IS) validity (tool will be available for that purpose).

9.1.6 Graphical documentation

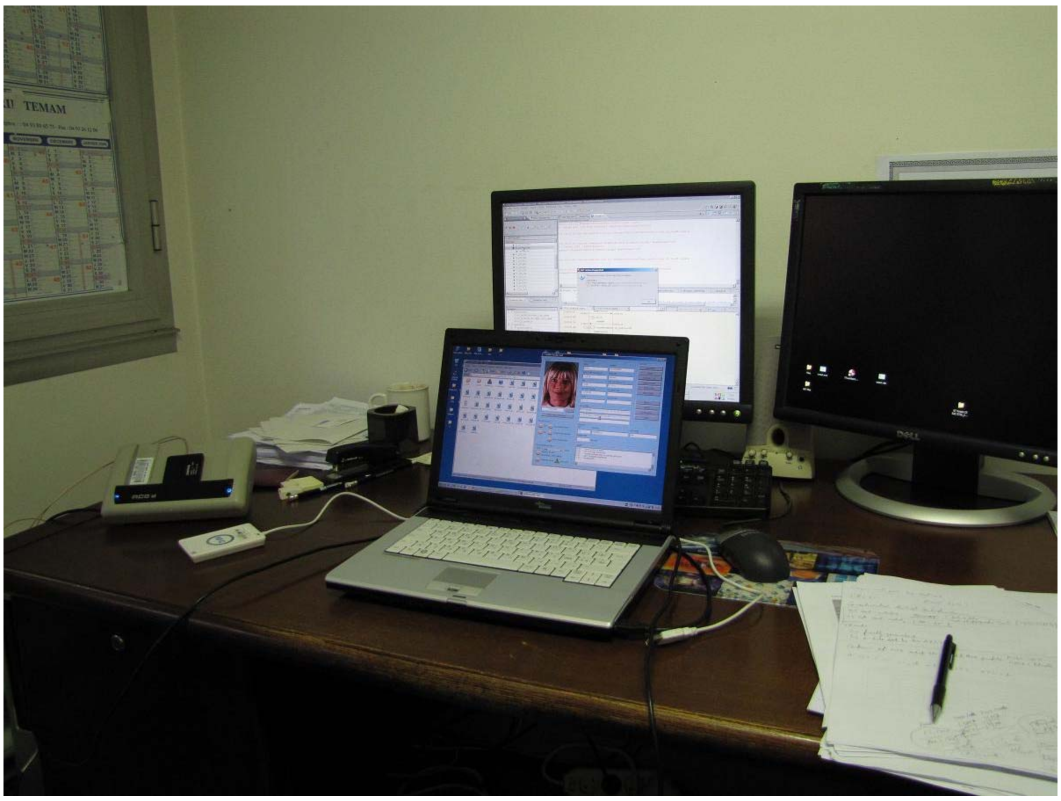


Figure 14: The overview of the test configuration

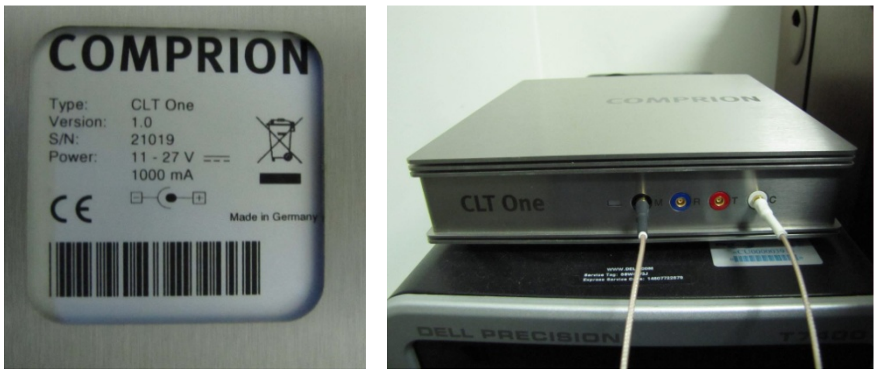


Figure 15: The comprion card simulator

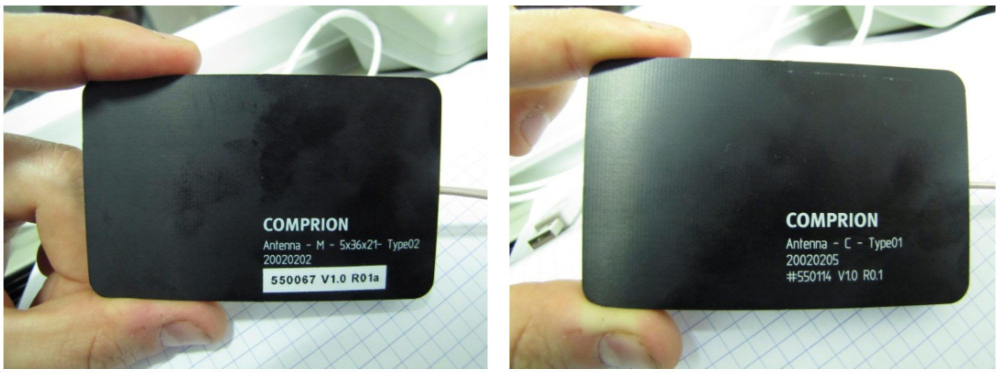


Figure 16: The antenna of the comprion card simulator

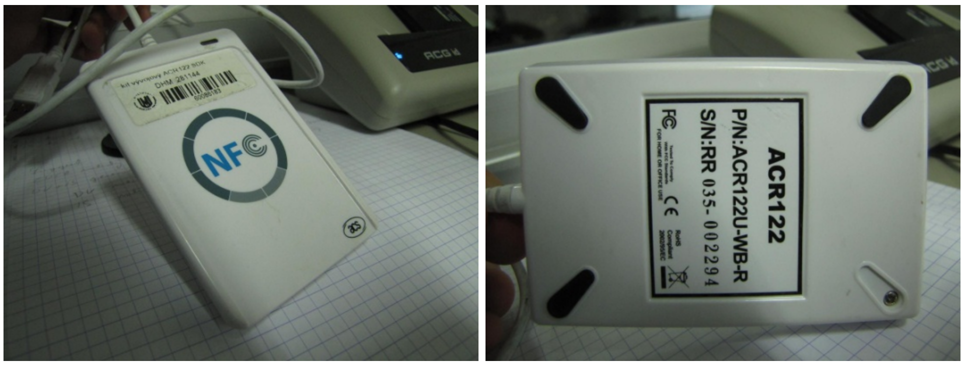


Figure 17: The ACR122 RFID reader

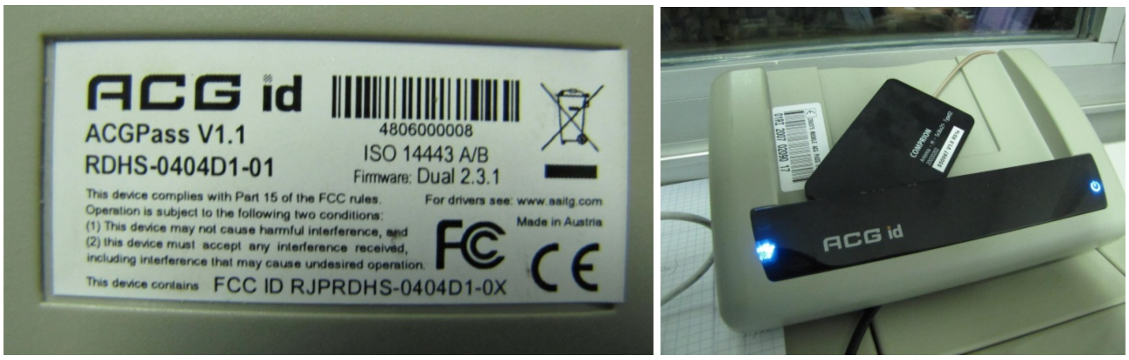

ACG id
ACGPass V1.1
RDHS-0404D1-01
4806000008
ISO 14443 A/B
Firmware: Dual 2.3.1
This device complies with Part 15 of the FCC rules.
Operation is subject to the following two conditions:
(1) This device may not cause harmful interference, and
(2) this device must accept any interference received,
including interference that may cause undesired operation.
For drivers see: www.acg.com
Made in Austria
This device contains FCC ID RJPRDHS-0404D1-0X
ACG id

Figure 18: The ACG id RFID reader

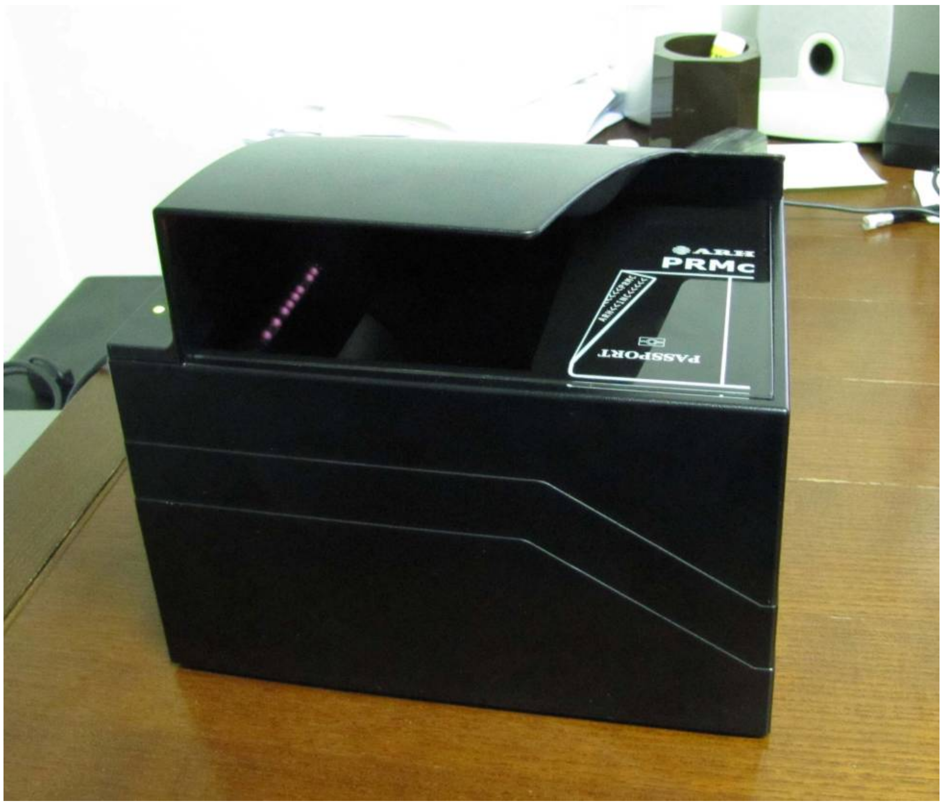
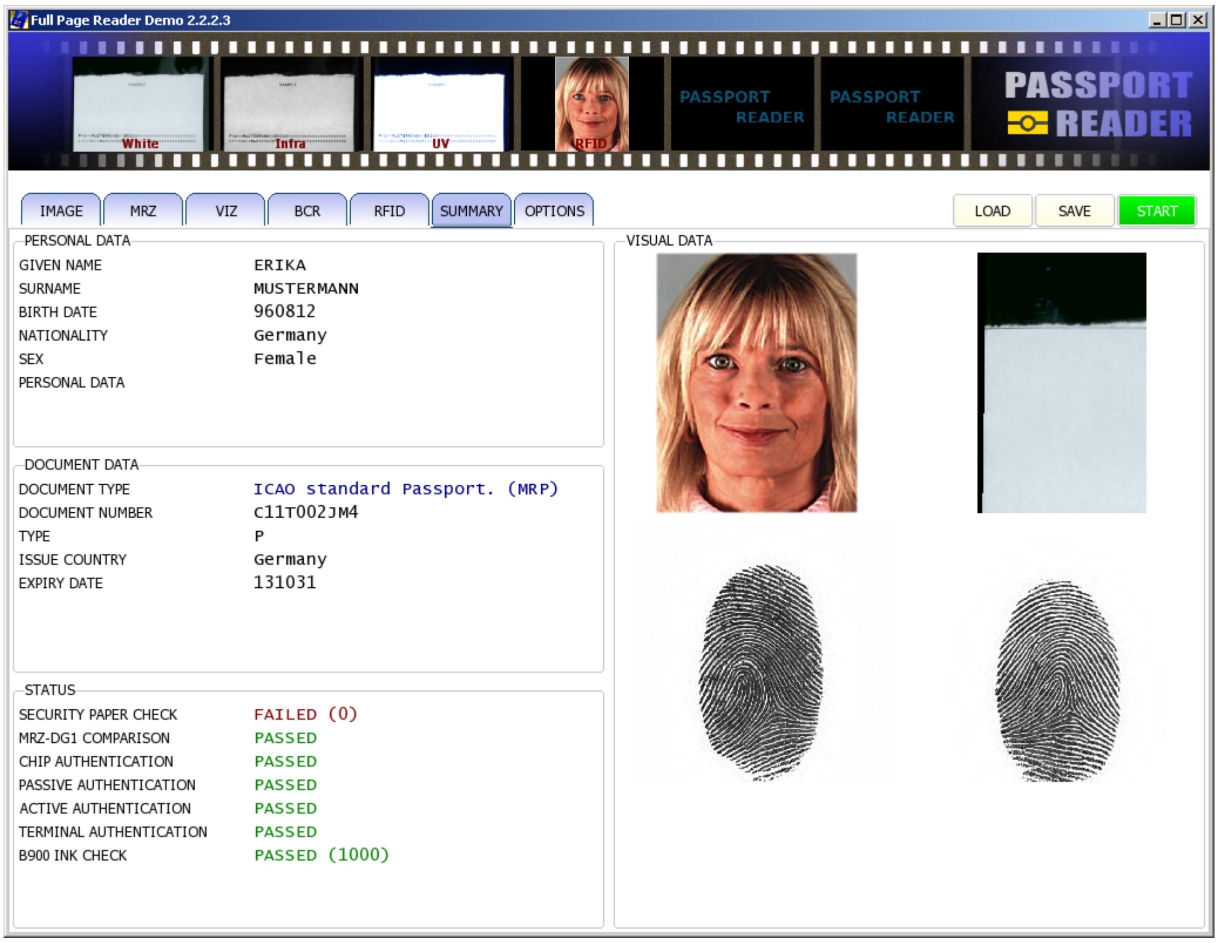


Figure 19: The ARH PRMc inspection system



PERSONAL DATA

GIVEN NAME	ERIKA
SURNAME	MUSTERMANN
BIRTH DATE	960812
NATIONALITY	Germany
SEX	Female
PERSONAL DATA	

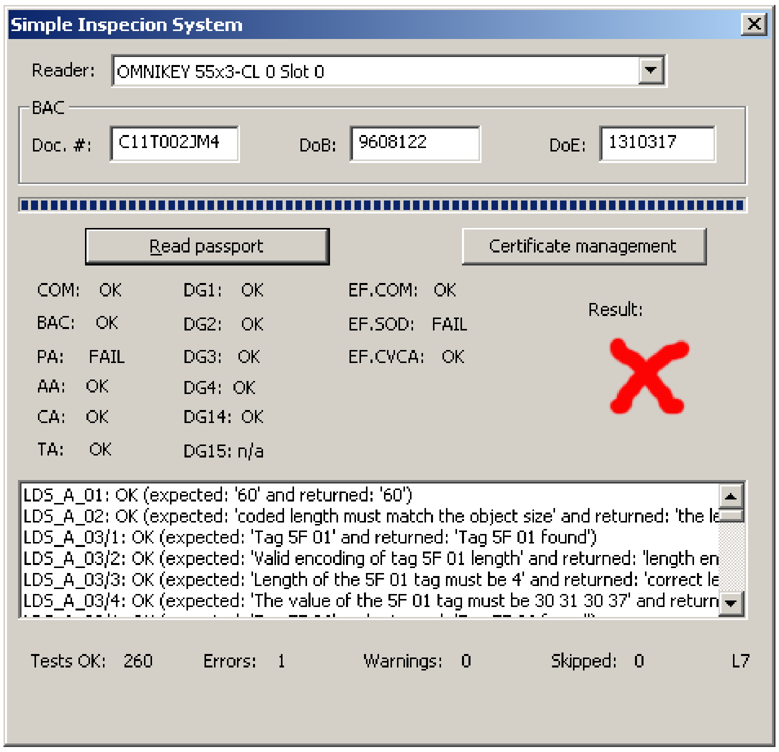
DOCUMENT DATA

DOCUMENT TYPE	ICAO standard Passport. (MRP)
DOCUMENT NUMBER	C11T002JM4
TYPE	P
ISSUE COUNTRY	Germany
EXPIRY DATE	131031

STATUS

SECURITY PAPER CHECK	FAILED (0)
MRZ-DG1 COMPARISON	PASSED
CHIP AUTHENTICATION	PASSED
PASSIVE AUTHENTICATION	PASSED
ACTIVE AUTHENTICATION	PASSED
TERMINAL AUTHENTICATION	PASSED
B900 INK CHECK	PASSED (1000)

Figure 20: The interface of the ARH Full page Reader Demo



Reader: OMNIKEY 55x3-CL 0 Slot 0

BAC

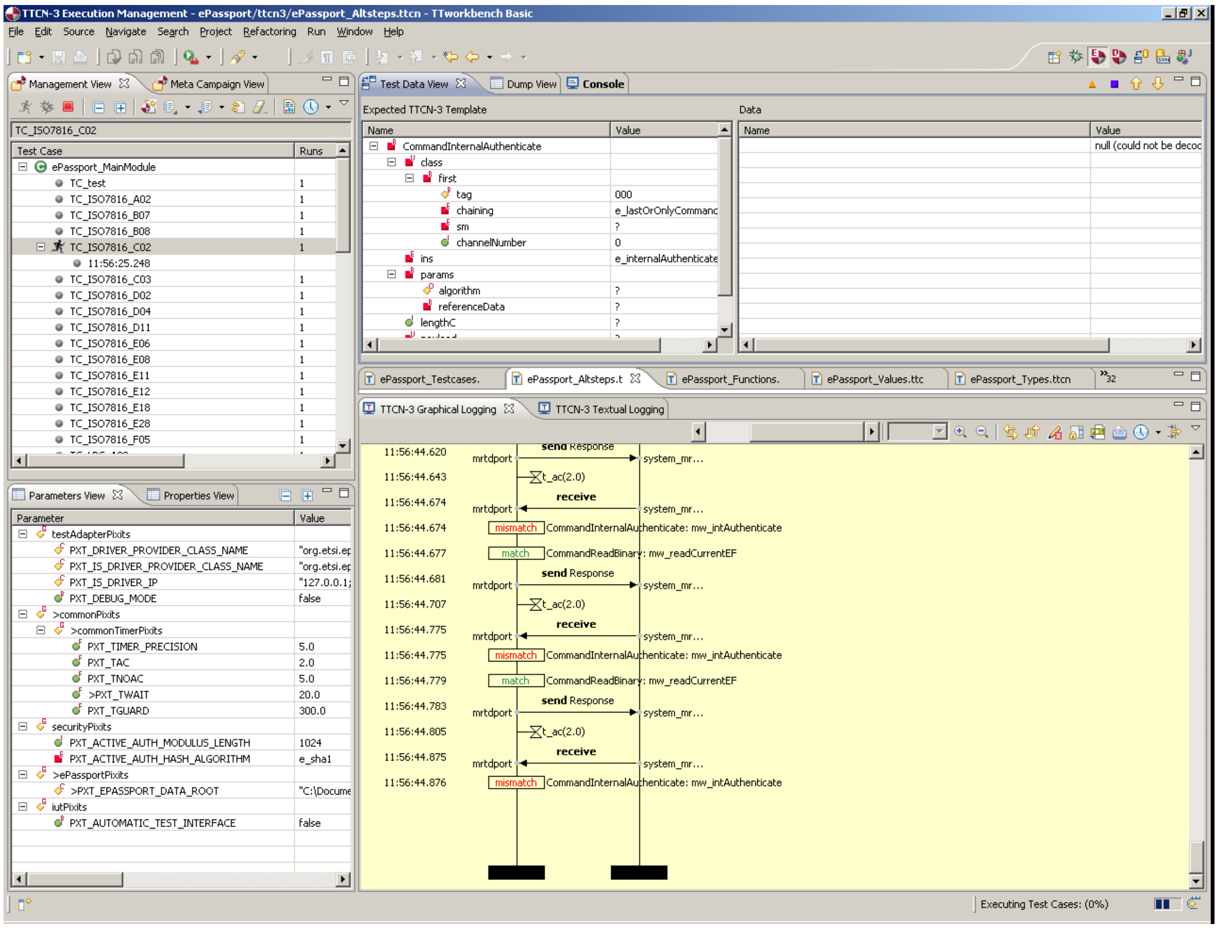
Doc. #: C11T002JM4 DoB: 9608122 DoE: 1310317

COM: OK	DG1: OK	EF.COM: OK
BAC: OK	DG2: OK	EF.SOD: FAIL
PA: FAIL	DG3: OK	EF.CVCA: OK
AA: OK	DG4: OK	
CA: OK	DG14: OK	
TA: OK	DG15: n/a	

```
LDS_A_01: OK (expected: '60' and returned: '60')
LDS_A_02: OK (expected: 'coded length must match the object size' and returned: 'the le
LDS_A_03/1: OK (expected: 'Tag 5F 01' and returned: 'Tag 5F 01 found')
LDS_A_03/2: OK (expected: 'Valid encoding of tag 5F 01 length' and returned: 'length en
LDS_A_03/3: OK (expected: 'Length of the 5F 01 tag must be 4' and returned: 'correct le
LDS_A_03/4: OK (expected: 'The value of the 5F 01 tag must be 30 31 30 37' and return
```

Tests OK: 260 Errors: 1 Warnings: 0 Skipped: 0 L7

Figure 21: The interface of the Simple Inspection System



TC_ISO7816_C02

Test Case	Runs
ePassport_MainModule	
TC_test	1
TC_ISO7816_A02	1
TC_ISO7816_B07	1
TC_ISO7816_B08	1
TC_ISO7816_C02	1
11:56:25.248	
TC_ISO7816_C03	1
TC_ISO7816_D02	1
TC_ISO7816_D04	1
TC_ISO7816_D11	1
TC_ISO7816_E06	1
TC_ISO7816_E08	1
TC_ISO7816_E11	1
TC_ISO7816_E12	1
TC_ISO7816_E18	1
TC_ISO7816_E28	1
TC_ISO7816_F05	1

Parameter	Value
testAdapterPixits	
PXT_DRIVER_PROVIDER_CLASS_NAME	"org.etsi.ep
PXT_IS_DRIVER_PROVIDER_CLASS_NAME	"org.etsi.ep
PXT_IS_DRIVER_IP	"127.0.0.1;
PXT_DEBUG_MODE	false
>commonPixits	
>commonTimerPixits	
PXT_TIMER_PRECISION	5.0
PXT_TAC	2.0
PXT_TNOAC	5.0
>PXT_TWAIT	20.0
PXT_TGUARD	300.0
securityPixits	
PXT_ACTIVE_AUTH_MODULUS_LENGTH	1024
PXT_ACTIVE_AUTH_HASH_ALGORITHM	e_sha1
>ePassportPixits	
>PXT_EPASSPORT_DATA_ROOT	"C:\Docume
iutPixits	
PXT_AUTOMATIC_TEST_INTERFACE	false

Expected TTCN-3 Template

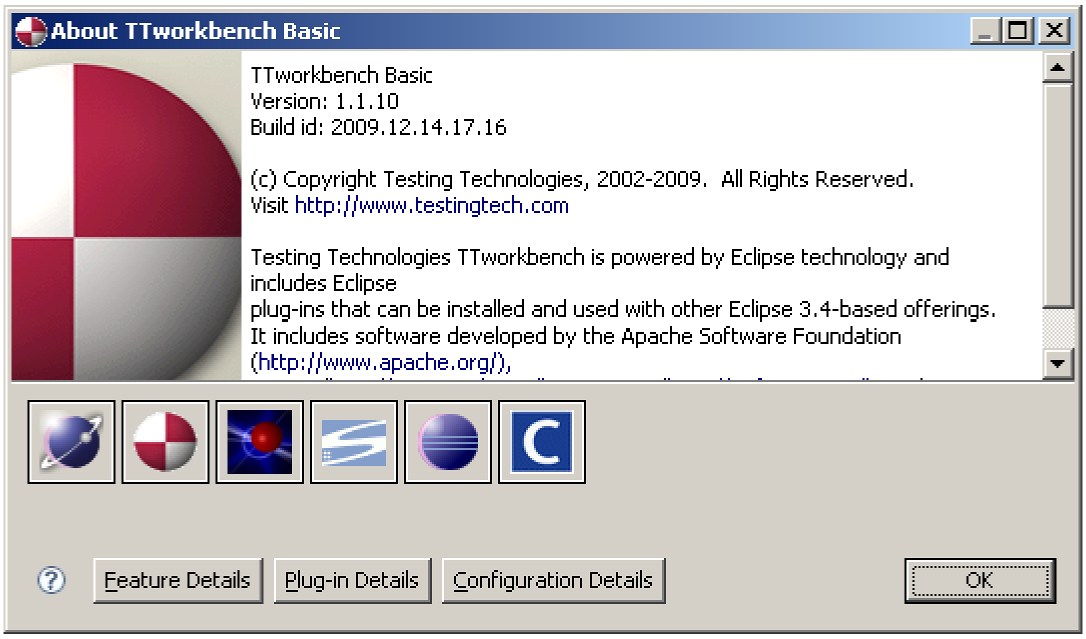
Name	Value
CommandInternalAuthenticate	
class	
first	
tag	000
chaining	e_lastOrOnlyCommanc
sm	?
channelNumber	0
ins	e_internalAuthenticate
params	
algorithm	?
referenceData	?
lengthC	?

Data

Name	Value
	null (could not be decoc

```
11:56:44.620  send Response    mrtdport ---> system_mr...
11:56:44.643  t_ac(2.0)
11:56:44.674  receive          mrtdport <--- system_mr...
11:56:44.674  mismatch  CommandInternalAuthenticate: mw_intAuthenticate
11:56:44.677  match     CommandReadBinary: mw_readCurrentEF
11:56:44.681  send Response    mrtdport ---> system_mr...
11:56:44.707  t_ac(2.0)
11:56:44.775  receive          mrtdport <--- system_mr...
11:56:44.775  mismatch  CommandInternalAuthenticate: mw_intAuthenticate
11:56:44.779  match     CommandReadBinary: mw_readCurrentEF
11:56:44.783  send Response    mrtdport ---> system_mr...
11:56:44.805  t_ac(2.0)
11:56:44.875  receive          mrtdport <--- system_mr...
11:56:44.876  mismatch  CommandInternalAuthenticate: mw_intAuthenticate
```

Figure 22: The TTworkbench Basic during execution of a test case



TTworkbench Basic
Version: 1.1.10
Build id: 2009.12.14.17.16

(c) Copyright Testing Technologies, 2002-2009. All Rights Reserved.
Visit http://www.testingtech.com

Testing Technologies TTworkbench is powered by Eclipse technology and includes Eclipse
plug-ins that can be installed and used with other Eclipse 3.4-based offerings.
It includes software developed by the Apache Software Foundation
(http://www.apache.org/),

Figure 23: The TTworkbench version information

9.1.7 Conclusion

This first validation campaign has been fruitful and gave to the project an excellent opportunity to check the progress perfomed.

A number of issues have been discovered during the testing. Some of issues have been fixed during the sessions and some other will be raised to the STF in order to be fixed before the next validation campaign.

9.2 Second Validation Campaign

9.2.1 Introduction

The second Validation session of the STF 400 took place on 20th and 21st December 2010 in the JRC lab facilities in Ispra (Italy).

The STF experts who have participated to the validation campaign were Z. Riha ((Masaryk University) as the evaluator with the support of L. Velez (ETSI), A. Berge (AMB Consulting), M. Van Den Steen (JRC), J.-M. Chareau (JRC), J. Loeschner (JRC).

The HW and SW system developed by the STF 400 aimed at conformity testing of inspection system is called just as **THE SYSTEM** in the present document.

The following hardware and software components were used during the test session in the JRC lab Identity and Biometric Technology Lab (IBTL):

- The system consists of chip card simulator supplied by Comprion (CLT One, Version 1.0, SN: 21019 together with antennas Comprion Antenna – C- Type01, 200210105, #550114, V1.0, R0.1 and Comprion Antenna - M – 5x36x21 – Type02, 20020202, 550067 V1.0 R01a) and the SW part consisting of TTworkbench Basic (Version 1.1.10, BuildID 2009.12.14.17.16) and relevant configurations, campaigns, scripts, libraries and chip/testcase profiles.
- The following RF readers were used:
    - ACG: ACG id, ACGPass V1.1, RDHS-0404D1-01, firmware: Dual 2.3.1 with JRC inventory number 01RI 2007 02090 17
    - ACG: ACG id, ACGPass V1.1, RDHS-0404D1-01, firmware: Dual 2.3.1 with JRC inventory number 01RI 2007 02088 92
    - SCM SDI010, SN: 21120832200801, PN: 904671
- The following full page passport readers were used:
    - RTE 8000, 5RL11655 together with RTE eMRTD Test Program 1.1.1.46 (see Figures 2 and 4)
- The following software inspection systems were used:
    - Secunet Golden Reader Tool version 2.9.4.
    - Simple inspection system of the evaluator
    - Secunet Platinum Reader Tool version 3.0.2.7. Change 94149, 9.6.2010 (see Figure 1)
    - Secunet Platinum Reader Tool version 3.0.2.7. Change 95355, 2.7.2010
    - ASK ePassport Viewer v1.0.7 (see Figure 3)
- The following computers were used:
    - FSC Lifebook S 7210, inventory number of JRC: 01 RI 2008 01651 72
        - Used for the RTE inspection system and Secunet Platinum Reader Tool version 3.0.2.7. Change 94149, 9.6.2010
    - FSC Lifebook S7220, EPASS173, inventory number of MU: DHM 286616
        - Used for the simple inspection system, the Golden Reader Tool 2.9.4. Secunet Platinum Reader Tool version 3.0.2.7. Change 95355, 2.7.2010 and ASK ePassport Viewer v1.0.7

- The "old" DELL notebook, inventory number of ETSI: CU00000380
    - Running the TTworkbench Basic (Version 1.1.10, BuildID 2009.12.14.17.16)
- The "new" DELL notebook. Inventory number of ETSI: CU00000453
    - Running the TTworkbench Basic (Version 11, BuildID 2010.10.28.13.46.25.965)
- The JRC desktop computer FSC SCENIC W620, EPASS174, inventory number of JRC: 01RI 2005 01723 69
    - Running the TTworkbench Basic (Version 11, BuildID 2010.10.28.13.46.25.965)
- FSC Lifebook S 7210, inventory number of JRC: 01 RI 2008 01649 49
    - Running the RGPA ProxiSpy application Version: 2.5.1045.0

• Additional tools used:
Resonance ProxiSpy XS282, inventory number of JRC: 01 RI 2008 01264 93

9.2.2 Evaluation Notes

1) At first the system was configured on a laptop, many components had to be updated to the latest version and the functionality of the system was verified by running the test cases ISO7816_A_02, ISO7816_D_02, ISO7816_E_06 with the Simple inspection system and GRT 2.9.4. We switched to the new configuration set provided by HPJ on Friday 17.12.2010. The certificates are now better organized. We prepared the sets of certificates for the manual transfer to inspection systems and also for the automated interface.

2) Next the tests focused on the Platinum Reader Tool with the ACG reader. We loaded the certificates to the tool and tested the first test case. The ISO7816_A_02 passed OK. Testing ISO7816_E_06 we have seen some communication errors and certificate chain build errors. We checked the certificates and we made the test case ISO7816_E_06 working.

3) The TTCN workbench Version 11 was installed also on a desktop of the JRC in the IBTL . A 5 years license was assigned. The configuration was working, but it was not faster than the notebook, so we preferred to use the notebook for further tests.

4) The Platinum Reader Tool during the inspection procedure selects twice the ePassport application. This was originally not supported by the system. A small change was made to allow that. This means the tests can be performed in a more comfortable way.

5) Further tests with the Platinum Reader Tool performed the ISO7816_B_07 and ISO7816_B_08 tests. The tests were running ok, but after the test a flag remained set and the error in SM was introduced into all following communication. That means that after such a test the complete TTCN campaign had to be reloaded to reset all parameters. This problem was fixed later during the session.

6) Test case ISO7816_C_02 was performed with the Platinum Reader Tool. The reading of the large file was ok, but passive authentication failed, as the hash of the file read did not match the hash digitally signed. This is a problem of the configuration. The EF.SOD file needs to be regenerated (resigned).

7) During the test case ISO7816_C_03 it turned out that the Platinum Reader Tool continues to read the short file even after it received a warning together with the shorter response. A change of the system was necessary. Not the reading attempts with offset behind the end of the file will end up with error code 6b 00.

8) Testing the test case ISO7816_D_02 we had to cope with many communication errors. We even tried to use the ProxiSpy. It seems that there are some errors on the low level of communication.

9) We tried the ISO7816_D_04 test case and we saw again many communication errors. We also tried with the Simple inspection system and it worked ok.

10) Also the test case ISO7816_D_11, ISO7816_E_08, ISO7816_E_11, ISO7816_E_12, ISO7816_E_18 worked OK (except sometimes for matching perfectly the test results including the items not performed etc). ISO7816_E_28 was crashing on the side of the system. This was fixed later during the test session. The test case ISO7816_F_05 worked ok, but it still does not implement the sha256 AA.

11) At the end of the procedure the system from time to time does not finish giving the final verdict. This is probably an unmap port issue. This still need to be fixed.

12) Also a few test cases of the layer 7 (LDS) were tested: LDS_A_03, LDS_A_04, LDS_H_07, LDS_H_71.

13) A sample test case was performed with the RTE. The ISO7816_A_02 is working ok, including the passive authentication.

14) We tried a new and fast notebook, but it turned out that the notebook ran the tests slower than the "old" notebook. Maybe because the new notebook used the version 11 of the TTCN workbench (and the old one 10). This issue needs to be clarified for further testing.

15) Instead of ACG RF reader we started using the SCM RF reader. The communication was working of with the Simple inspection system for the ISO7816_A_02 test case, next we tried the Platinum Reader Tool. The ISO7816_A_02 test case was working, ISO7816_D_02 as well. With ISO7816_D_04 we have seen again communication errors and the testcase ISO7816_D_11 failed on the system side. ISO7816_E_06 had to cope with communication errors again, but the aim of the test case (TA) worked fine. ISO7816_E_08 was running OK, same for ISO7816_E_11. ISO7816_E_12 had some communication errors, but it is clear that the terminal authentication was working even if the challenge was shorter (7 bytes instead of 8), which means the Platinum Reader tool is not behaving correctly with respect to the ISO7816_E_12 test case. ISO7816_E_18 test case worked fine, the ISO7816_E_28 was failing because the configuration of system used a wrong configuration of CVCA certificate and EF_CVCA file. This was corrected and the ISO7816_E_28 test case passed correctly. During the ISO7816_F_05 test case we have seen again communication errors, but the active authentication went through successfully (based still on SHA1 anyway).

16) A demo "inspection system" written by ASK was tested with the testcase ISO7816_A_02. The configuration was read by the ASK inspection system. The inspection system is very simple and does not support EAC. Therefore we did not perform any other test cases, only verified basic readability of the A_02 configuration.

17) Next the RTE inspection system was tested. The test case ISO7816_A_02 was performed successfully.

18) Next the ISO7816_B_07 was performed. After the Secure Messaging (SM) failed the IS tried to repeat the inspection procedure, reselect the ePassport application, perform the BAC etc. This was not expected by the system and plain reading of the passport followed. The system was modified to handle such situations in a reasonable way (double selection of ePassport application and authentication possible, but always behaving as a BAC protected passport). At the same time a problem with Le in a unprotected APDU appeared and was quickly solved. The RTE works correctly with respect to ISO7816_B_07 and reports SM error (actually "Secure message could not be decrypted").

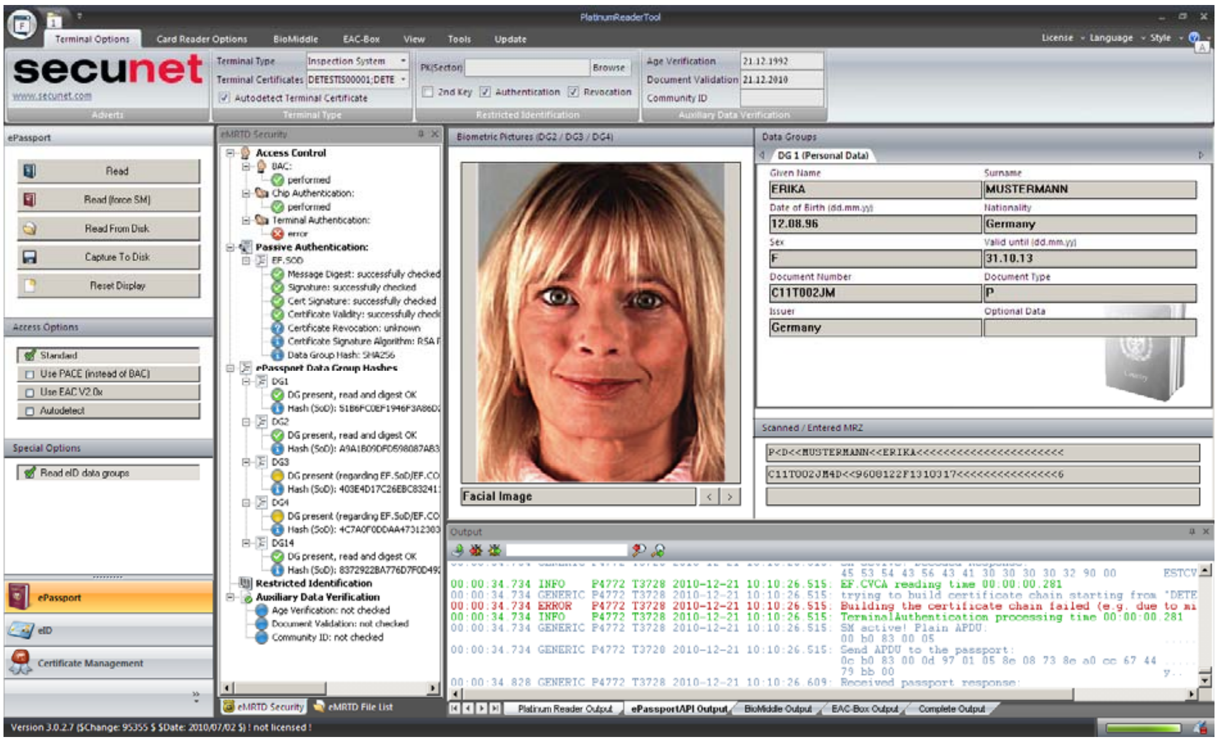
19) The test case ISO7816_B_08 is also working correctly with RTE inspection system.

20) Next the test cases ISO7816_C_02 and ISO7816_C_03 were successfully performed. And then the EAC layer 6 test cases were performed. The positive testcase ISO7816_D_02 works OK.

21) ISO7816_D_04 shows that there's a problem in the RTE implementation of chip authentication with KeyID is wrong and as KeyID the whole TLV is sent (instead of 01 the full 02 01 01 is sent in the 84 KeyID tag).

22) The following test cases were performed with the RTE: ISO7816_D_11 (ok), ISO7816_E_06 (ok), ISO7816_E_08 (ok), ISO7816_E_11 (ok), ISO7816_E_12 (ok), ISO7816_E_18 (ok), ISO7816_E_28 (ok), ISO7816_F_05 (AA based on SHA1, ok), LDS_A_03 (ok), LDS_A_04 (IS behaves incorrectly), LDS_A_04 (IS behaves incorrectly), LDS_B_11 (IS behaves incorrectly), LDS_B_22 (IS behaves incorrectly), LDS_B_25 (ok), LDS_C_03 (ok), LDS_C_09 (ok), LDS_C_13 (IS behaves incorrectly), LDS_C_19 (ok), LDS_D_03 (ok), LDS_D_12 (IS behaves incorrectly), LDS_D_15 (IS behaves incorrectly), LDS_E_02 (IS behaves incorrectly), LDS_E_09 (IS behaves incorrectly), LDS_F_04 (IS behaves incorrectly), LDS_F_06 (IS behaves incorrectly).

9.2.3 Graphical documentation



Terminal Type: Inspection System
Terminal Certificates: DETESTIS00001;DETE
✓ Autodetect Terminal Certificate

PK(Sector):
☐ 2nd Key ✓ Authentication ✓ Revocation

Age Verification: 21.12.1992
Document Validation: 21.12.2010
Community ID:

✓ Standard
☐ Use PACE (instead of BAC)
☐ Use EAC V2.0x
☐ Autodetect

✓ Read eID data groups

Access Control
- BAC:
  - performed
- Chip Authentication:
  - performed
- Terminal Authentication:
  - error

Passive Authentication:
- EF.SOD
  - Message Digest: successfully checked
  - Signature: successfully checked
  - Cert Signature: successfully checked
  - Certificate Validity: successfully check
  - Certificate Revocation: unknown
  - Certificate Signature Algorithm: RSA
  - Data Group Hash: SHA256

ePassport Data Group Hashes
- DG1
  - DG present, read and digest OK
  - Hash (SoD): 51B6FC0EF1946F3A86D
- DG2
  - DG present, read and digest OK
  - Hash (SoD): A9A1B09DFD598087AB3
- DG3
  - DG present (regarding EF.SoD/EF.CO
  - Hash (SoD): 403E4D17C26EBC832411
- DG4
  - DG present (regarding EF.SoD/EF.CO
  - Hash (SoD): 4C7A0F0DDAA47312303
- DG14
  - DG present, read and digest OK
  - Hash (SoD): 8372922BA776D7F0D49

Restricted Identification

Auxiliary Data Verification
- Age Verification: not checked
- Document Validation: not checked
- Community ID: not checked

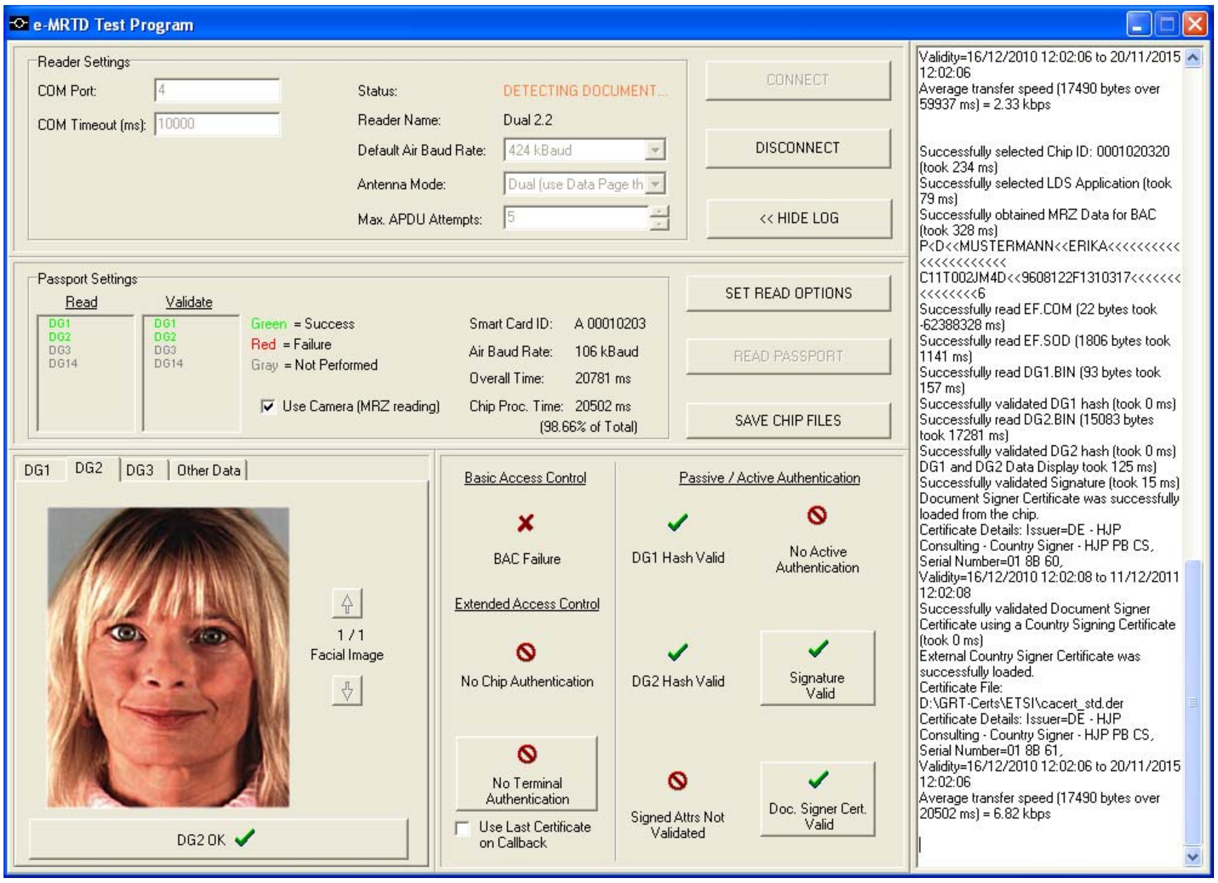
Given Name	Surname
ERIKA	MUSTERMANN
Date of Birth (dd.mm.yy)	Nationality
12.08.96	Germany
Sex	Valid until (dd.mm.yy)
F	31.10.13
Document Number	Document Type
C11T002JM	P
Issuer	Optional Data
Germany	

```
P<D<<MUSTERMANN<<ERIKA<<<<<<<<<<<<<<<<<<<<<<
C11T002JM4D<<9608122F1310317<<<<<<<<<<<<<<<6
```

```
                                                                 45 53 54 43 56 43 41 30 30 30 30 32 90 00        ESTCV
00:00:34.734 INFO    P4772 T3728 2010-12-21 10:10:26.515: EF.CVCA reading time 00:00:00.281
00:00:34.734 GENERIC P4772 T3728 2010-12-21 10:10:26.515: trying to build certificate chain starting from "DETE
00:00:34.734 ERROR   P4772 T3728 2010-12-21 10:10:26.515: Building the certificate chain failed (e.g. due to mi
00:00:34.734 INFO    P4772 T3728 2010-12-21 10:10:26.515: TerminalAuthentication processing time 00:00:00.281
00:00:34.734 GENERIC P4772 T3728 2010-12-21 10:10:26.515: SM active! Plain APDU:
                                                                 00 b0 83 00 05                                   .....
00:00:34.734 GENERIC P4772 T3728 2010-12-21 10:10:26.515: Send APDU to the passport:
                                                                 0c b0 83 00 0d 97 01 05 8e 08 73 8e a0 cc 67 44  .....
                                                                 79 bb 00                                         y..
00:00:34.828 GENERIC P4772 T3728 2010-12-21 10:10:26.609: Received passport response:
```

Version 3.0.2.7 ($Change: 95355 $ $Date: 2010/07/02 $) ! not licensed !

Figure 24: The user interface of Platinum Reader Tool



Reader Settings

COM Port: 4
COM Timeout (ms): 10000
Status: DETECTING DOCUMENT...
Reader Name: Dual 2.2
Default Air Baud Rate: 424 kBaud
Antenna Mode: Dual (use Data Page th
Max. APDU Attempts: 5

Passport Settings

Read	Validate
DG1	DG1
DG2	DG2
DG3	DG3
DG14	DG14

Green = Success
Red = Failure
Gray = Not Performed

✓ Use Camera (MRZ reading)

Smart Card ID: A 00010203
Air Baud Rate: 106 kBaud
Overall Time: 20781 ms
Chip Proc. Time: 20502 ms (98.66% of Total)

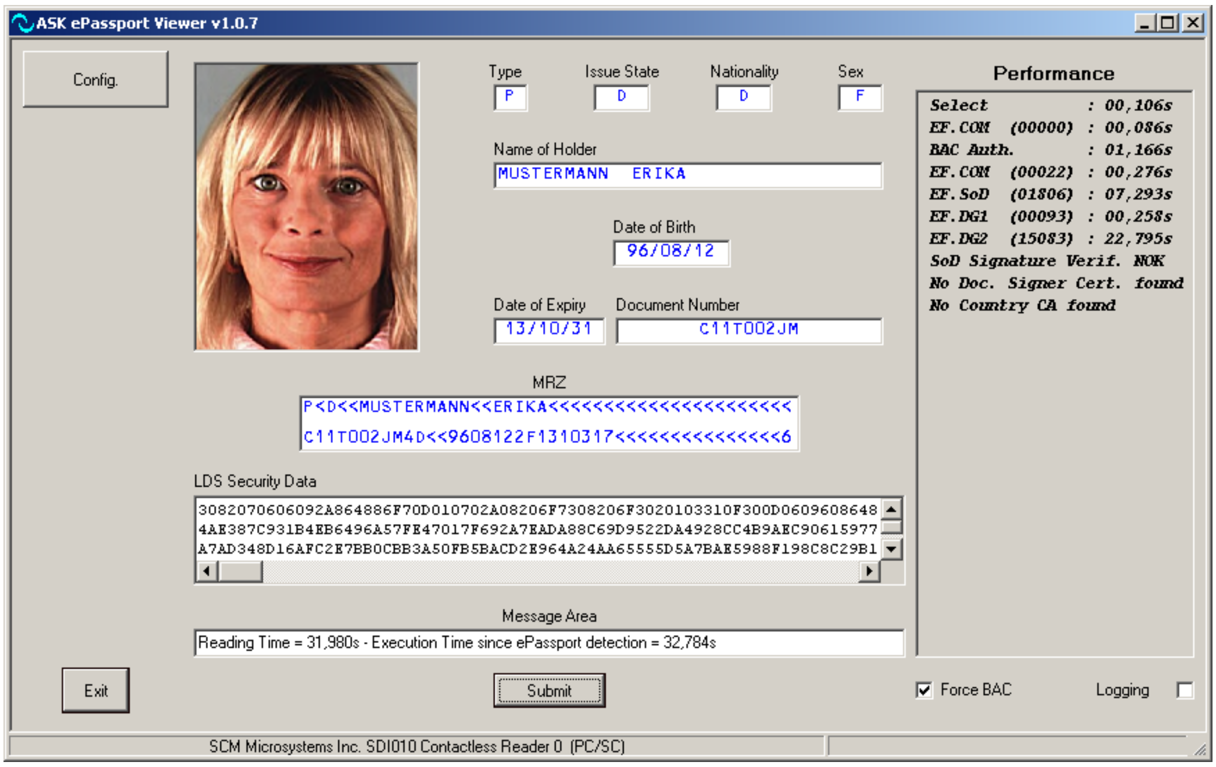
1 / 1
Facial Image
DG2 OK ✓

BAC Failure
No Chip Authentication
No Terminal Authentication
☐ Use Last Certificate on Callback
DG1 Hash Valid
DG2 Hash Valid
Signed Attrs Not Validated
No Active Authentication
Signature Valid
Doc. Signer Cert. Valid

```
Validity=16/12/2010 12:02:06 to 20/11/2015
12:02:06
Average transfer speed (17490 bytes over
59937 ms) = 2.33 kbps


Successfully selected Chip ID: 0001020320
(took 234 ms)
Successfully selected LDS Application (took
79 ms)
Successfully obtained MRZ Data for BAC
(took 328 ms)
P<D<<MUSTERMANN<<ERIKA<<<<<<<<<<
<<<<<<<<<<<<
C11T002JM4D<<9608122F1310317<<<<<<
<<<<<<<<6
Successfully read EF.COM (22 bytes took
-62388328 ms)
Successfully read EF.SOD (1806 bytes took
1141 ms)
Successfully read DG1.BIN (93 bytes took
157 ms)
Successfully validated DG1 hash (took 0 ms)
Successfully read DG2.BIN (15083 bytes
took 17281 ms)
Successfully validated DG2 hash (took 0 ms)
DG1 and DG2 Data Display took 125 ms)
Successfully validated Signature (took 15 ms)
Document Signer Certificate was successfully
loaded from the chip.
Certificate Details: Issuer=DE - HJP
Consulting - Country Signer - HJP PB CS,
Serial Number=01 8B 60,
Validity=16/12/2010 12:02:08 to 11/12/2011
12:02:08
Successfully validated Document Signer
Certificate using a Country Signing Certificate
(took 0 ms)
External Country Signer Certificate was
successfully loaded.
Certificate File:
D:\GRT-Certs\ETSI\cacert_std.der
Certificate Details: Issuer=DE - HJP
Consulting - Country Signer - HJP PB CS,
Serial Number=01 8B 61,
Validity=16/12/2010 12:02:06 to 20/11/2015
12:02:06
Average transfer speed (17490 bytes over
20502 ms) = 6.82 kbps
```

Figure 25: The user interface of RTE e-MRTD Test Program



Type: P
Issue State: D
Nationality: D
Sex: F

Name of Holder: MUSTERMANN ERIKA

Date of Birth: 96/08/12

Date of Expiry: 13/10/31

Document Number: C11T002JM

MRZ

```
P<D<<MUSTERMANN<<ERIKA<<<<<<<<<<<<<<<<<<<<<<
C11T002JM4D<<9608122F1310317<<<<<<<<<<<<<<<6
```

LDS Security Data

```
3082070606092A864886F70D010702A08206F7308206F3020103310F300D0609608648
4AE387C931B4EB6496A57FE47017F692A7EADA88C69D9522DA4928CC4B9AEC90615977
A7AD348D16AFC2E7BB0CBB3A50FB5BACD2E964A24AA65555D5A7BAE5988F198C8C29B1
```

Message Area

Reading Time = 31,980s - Execution Time since ePassport detection = 32,784s

Performance

```
Select              : 00,106s
EF.COM  (00000)     : 00,086s
BAC Auth.           : 01,166s
EF.COM  (00022)     : 00,276s
EF.SoD  (01806)     : 07,293s
EF.DG1  (00093)     : 00,258s
EF.DG2  (15083)     : 22,795s
SoD Signature Verif. NOK
No Doc. Signer Cert. found
No Country CA found
```

[✓] Force BAC   Logging [ ]

SCM Microsystems Inc. SDI010 Contactless Reader 0 (PC/SC)

Figure 26: The user interface of ASK ePassport Viewer

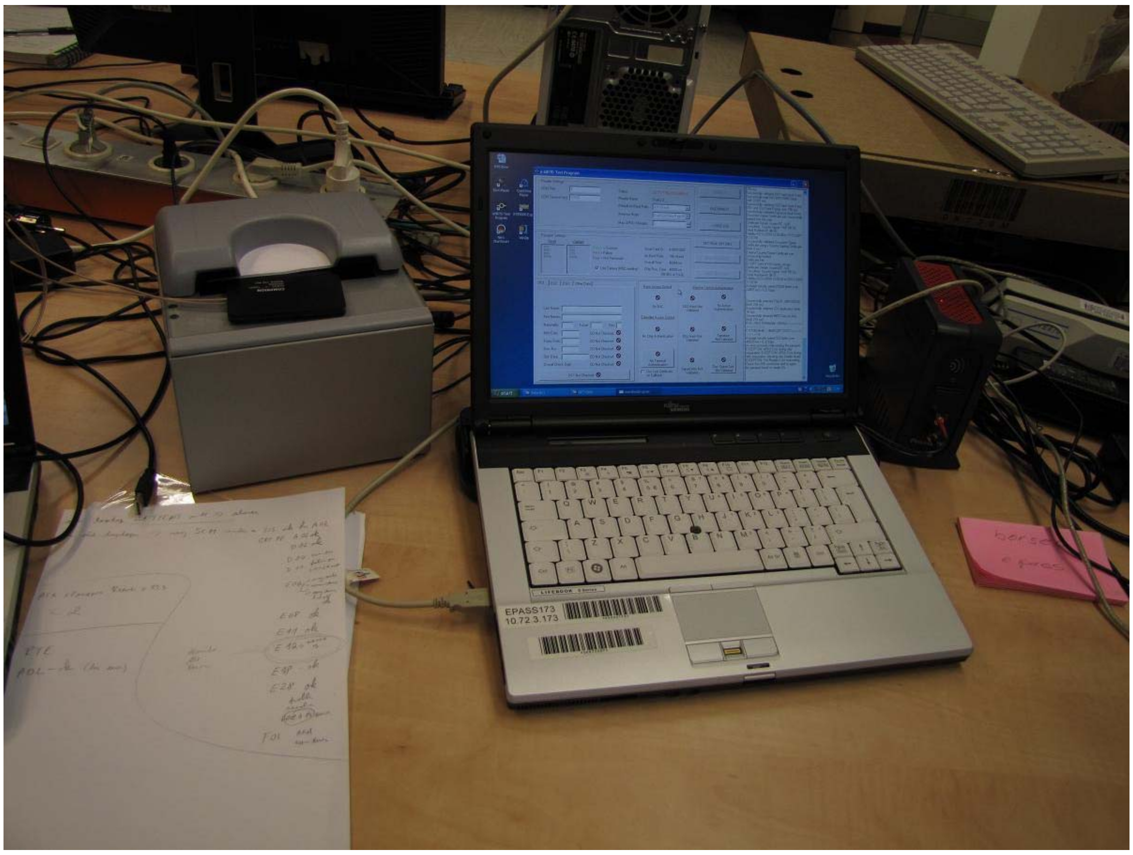


Figure 27: The position of the antenna of the simulator on the RTE 8000

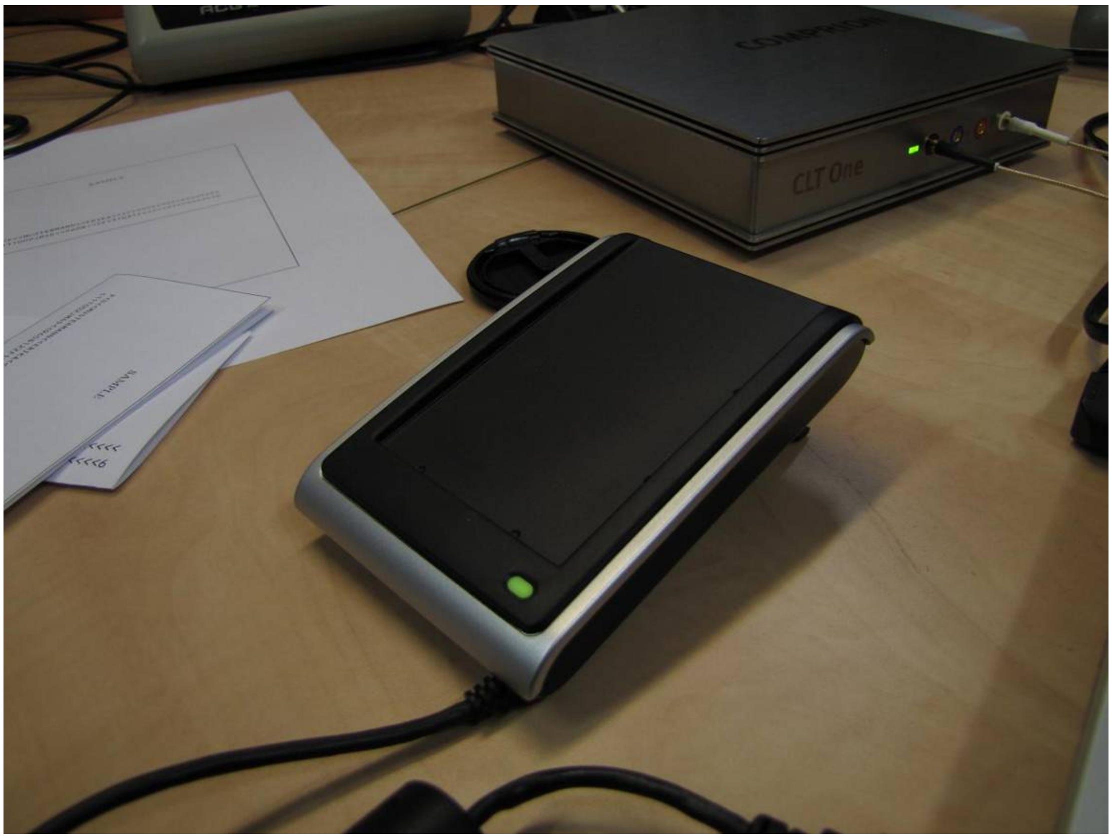


Figure 28: The full page ePassport reader RTE 8000 with one of the used data pages

9.2.4 Conclusion

This Second validation campaign has been an excellent opportunity to work closely with the JRC ePassport experts and use the Biometric Lab facilities.

The team has improved the software and the security profiles by running the test suite against the new Inspection Systems provided by the JRC.

A number of issues have been discovered during the testing. Some of issues have been fixed during the sessions and some other will be raised to the STF in order to be fixed before the 3rd validation campaign.

9.3 Third Validation Campaign

9.3.1 Introduction

The third Validation session of the STF 400 took place on 9th, 10th and 11th February 2011 in the JRC lab facilities in Ispra (Italy).

The STF experts who have participated to the validation campaign were Z. Riha (Masaryk University) as the evaluator with the support of L. Velez (ETSI), A. Berge (AMB Consulting), H. Funke (HJP Consulting) and J.-M. Chareau (JRC).

The HW and SW system developed by the STF 400 aimed at conformity testing of inspection system is called just as **THE SYSTEM** in the present document.

The following hardware and software components were used during the test session in the JRC lab Identity and Biometric Technology Lab (IBTL):

- **The system** consists of chip card simulator supplied by Comprion (CLT One, Version 1.0, SN: 21019 together with antennas Comprion Antenna – C- Type01, 200210105, #550114, V1.0, R0.1 and Comprion Antenna - M – 5x36x21 – Type02, 20020202, 550067 V1.0 R01a) and the SW part consisting of TTworkbench Basic (Version 1.1.10, BuildID 2010.05.04.16.21) and relevant configurations, campaigns, scripts, libraries and chip/testcase profiles.
- The following RF reader was used:
  - **SCM** SCL010, SN: 21160943202111, PN: 905073
- The following full page passport readers were used:
  - **RTE** 8000, 5RL11655 together with RTE eMRTD Test Program 1.1.1.46
  - **ARH** PRMC233RL092850 together with "Full Page Reader Demo 2.2.2.3"
- The following software inspection systems were used
  - Secunet **Golden Reader Tool** version 2.9.4.
- The following computers were used:
  - **DELL desktop**: DELL Optiplex 740, JRC inventory number 01RI 2009 01985 00
    - Running the TTworkbench Basic (Version 1.1.10, BuildID 2010.05.04.16.21)
  - **FSC Lifebook S 7110**, inventory number of JRC: 01 RI 2006 03367 04
    - Used for the RTE and ARH SW part of the inspection system
  - **FSC Lifebook S7220**, inventory number of MU: DHM 286616
    - Used for the simple inspection system, the Golden Reader Tool 2.9.4.

1) The TTworkbench was installed on a DELL desktop computer. First version 12 was installed and used, but after some troubles with stability and functionality the version of the workbench was downgraded to 10.

2) The version 10 of TTworkbench on the DELL desktop running the conformity test system based on the latest configurations obtained from HJP on 31 Jan 2011was quickly verified to be functional as expected against the FSC Lifebook S7220:

   a) One BAC (ISO7816_A02) and one EAC (ISO7816_D02) testcase was performed against GRT 2.9.4 and SCM reader. The behavior was as expected.

   b) One EAC (ISO7816_D02) test case was performed against Simple inspection system with the automated interface and the SCM reader. The behavior was as expected.

3) The testing of inspection system RTE with the eMRTD application started:

   c) The test case ISO7816_A02 ran ok.

   d) The test case ISO7816_B07 ran ok.

   e) The test case ISO7816_B08 ran ok, but the RTE inspection system does not give much detailed description about the problem detected.

   f) The test case ISO7816_C02 ran ok.

   g) The test case ISO7816_C03 ran ok, but the RTE inspection system does not give much detailed description about the problem detected.

   h) The test case ISO7816_D02 ran ok.

i) During test case ISO7816_D04 the inspection system incorrectly sends the chip authentication Key Identifier including the ASN1 coding (i.e. 020101h instead of 01h). The system cannot find the file specified and crashes. The system was corrected later.

j) During test case ISO7816_D11 the chip authentication fails as expected, the inspection system tries to restart the BAC, this is refused by the system and the error is indeed reported as BAC error by the inspection system. The inspection system correctly reports the wrong OID in DG14 during the chip authentication.

k) The test case ISO7816_E06 ran ok. It was noted that the inspection system uses the current date to check the time validity of the certificates.

l) During the ISO7816_E08, the inspection system uses the provided certificates even if it should detect that they cannot be used. As a result the system crashes on signature verification. This problem was corrected later. Moreover the system was modified to reject the cert as the CARs do not match.

m) The test case ISO7816_E11 ran ok.

n) The test case ISO7816_E12 ran ok.

o) The test case ISO7816_E18 ran ok.

p) The test case ISO7816_E28 is using wrong configuration (EF.CVCA file modified instead of DG14). This needs to be fixed.

q) The test case ISO7816_F05 is not yet correctly implemented on the side of the system (the unusual AA), but as implemented currently (the usual AA) it ran ok.

r) The test case LDS_A03 ran ok.

s) The test case LDS_A04 ran ok, but the IS did report any errors.

t) During the test case LDS_B11 it turned out that the system uses DG1 for the BAC key derivation instead of the MRZ configuration information. This was corrected.

u) The test case LDS_B22 ran ok.

v) The test case LDS_B25 ran ok.

w) The test case LDS_C03 ran ok.

x) The test case LDS_C09 ran ok.

y) The test case LDS_C13 ran ok, but the IS did report any errors.

z) The test case LDS_C19 ran ok.

aa) The test case LDS_D03 ran ok.

bb) The test case LDS_D12 ran ok, but the IS did report any errors.

cc) The test case LDS_D15 ran ok, but the IS did report any errors.

dd) The test case LDS_E01 ran ok.

ee) The test cases LDS_E02 and LDS_E09 were skipped as the inspection system does not display the iris image.

ff) The test case LDS_F04 ran ok, but the IS did report any errors.

gg) The test case LDS_F06 ran ok, but the IS did report any errors.

hh) During the test case LDS_G01 the IS did crash.

ii) The test case LDS_G02 ran ok.

jj) The test case LDS_H03 ran ok.

kk) The test case LDS_H07 ran ok.

ll) The test case LDS_H22 ran ok, but the TA was not performed.

mm) The test case LDS_H24 ran ok.

nn) The test case LDS_H31 ran ok, but the TA was not performed.

oo) The test case LDS_H32 ran ok.

pp) The test case LDS_H33 ran ok, but the TA was not performed.

qq) The test case LDS_H37 ran ok.

rr) The test case LDS_H45 ran ok.

ss) The test case LDS_H51 ran ok, but the IS did report any errors.

tt) The test case LDS_H54 ran ok, but the IS did report any errors.

uu) The test case LDS_H63 ran ok, but the IS did report any errors.

vv) During the test case LDS_H71 the IS reported "unsupported algorithm".

ww) During the test case LDS_H75 the IS reported "signature validation failed".

xx) The test case LDS_H83 ran ok.

yy) During the test case LDS_I01 the IS did crash.

4) The testing continues with the ARH inspection system. Due to some known issues with the IS, the reading of DG4 was disabled in all following tests!

zz) The test case ISO7816_A02 ran ok. It was noted that the inspection system loads the certificates during startup and does not update the list of certificates after the folder update.

aaa) The test case ISO7816_D02 ran ok. It was noted that the private key file name needs to be the same as the name of the IS certificate (except for the file extension).

bbb) The test case ISO7816_B07 ran ok, but the IS did report any errors.

ccc) The test case ISO7816_B08 ran ok.

ddd) The test case ISO7816_C02 ran ok.

eee) The test case ISO7816_C03 ran ok.

fff) The test case ISO7816_D02 ran ok.

ggg) The test case ISO7816_D04 ran ok, but IS report PA fail as well.

hhh) The test case ISO7816_D11 ran ok.

iii) The test case ISO7816_E06 ran ok.

jjj) The test case ISO7816_E08 ran ok.

kkk) The test case ISO7816_E11 ran ok.

lll) During the test case ISO7816_E12 the short challenge was accepted, the IS does not report any TA error.

mmm) The test case ISO7816_E18 ran ok, the IS does not even try to use the IS Certificate 11.

nnn) The test case ISO7816_E28 ran ok.

ooo) The test case ISO7816_F05 implementing AA RSA with sha-1 ran ok, implementing AA RSA with sha-256 failed with "algorithm unsupported" error.

ppp) The test case LDS_A03 ran ok.

qqq) The test case LDS_A04 ran ok, but no problems were reported by the IS.

rrr) The test case LDS_B11 ran ok.

sss) The test case LDS_B22 ran ok, but no problems were reported by the IS.

ttt) The test case LDS_B25 ran ok.

uuu) The test case LDS_C03 ran ok.

vvv) The test case LDS_C09 ran ok, no problems were reported by the IS, but the image was not shown.

www) The test case LDS_C13 ran ok, but no problems were reported by the IS.

xxx) The test case LDS_C19 ran ok, but no problems were reported by the IS.

yyy) The test case LDS_D03 ran ok.

zzz) The test case LDS_D12 ran ok, but no problems were reported by the IS.

aaaa) The test case LDS_D15 ran ok, but no problems were reported by the IS and no fingerprints were displayed.

bbbb) Test cases LDS_E01, LDS_E02, LDS_E09 were skipped as reading of DG4 was disabled.

cccc) The test case LDS_F04 ran ok, but no problems were reported by the IS.

dddd) The test case LDS_F06 ran ok, but no problems were reported by the IS.

eeee) The test case LDS_G01 ran ok.

ffff) The test case LDS_G02 ran ok.

gggg) The test case LDS_H03 ran ok.

hhhh) The test case LDS_H07 ran ok.

iiii) The test case LDS_H22 ran ok.

jjjj) The test case LDS_H24 ran ok.

kkkk) The test case LDS_H31 ran ok.

llll) The test case LDS_H32 ran ok.

mmmm) The test case LDS_H33 ran ok.

nnnn) The test case LDS_H37 ran ok.

oooo) The test case LDS_H45 ran ok.

pppp) The test case LDS_H51 ran ok, but no problems were reported by the IS.

qqqq) The test case LDS_H54 ran ok, but no problems were reported by the IS.

rrrr) The test case LDS_H63 ran ok, but no problems were reported by the IS.

ssss) During the test case LDS_H71 the IS rejected the signature with "unsupported algorithm error".

tttt) The test case LDS_H75 ran ok.

uuuu) The test case LDS_H83 ran ok.

vvvv) The test case LDS_I01 ran ok, but no problems were reported by the IS.

All issues from the previous test sessions were principally resolved:

1) The number of certificates and keys needed to load into the IS under test will be minimal. Table 1 from the first test session will be used.

2) A tool to generate fresh 90-days-valid empty CRL will be created.

3) The validity of the CSCA certificates will be 5 years, of DS certificates 3 years, the certificates will use names mentioning ETSI and different certificates will use different Serial Numbers.

4) The validity of CVCA certificates will be 3 years. A tool will be prepared for generation of fresh DV and IS certificates (DV01, DV08, IS01, IS08, IS11) valid 1 months (DV) and 2 weeks (IS).

5) The test case ISO7816_F05 was correctly implemented. Now the AA using RSA with sha256 is used. The testcase was validated against GRT (pass), RTE (pass) and ARH (fail).

6) Comparison of the results will be possible in 2 modes. The Simple mode will only compare final YES/NO i.e. PASS/FAIL of the inspection procedure as a whole. The advanced (or strict) mode will compare all the items (categories) of sub-results. Mode used for testing will be configurable.

9.3.2 Graphical documentation

Figure 29: STF experts in JRC lab

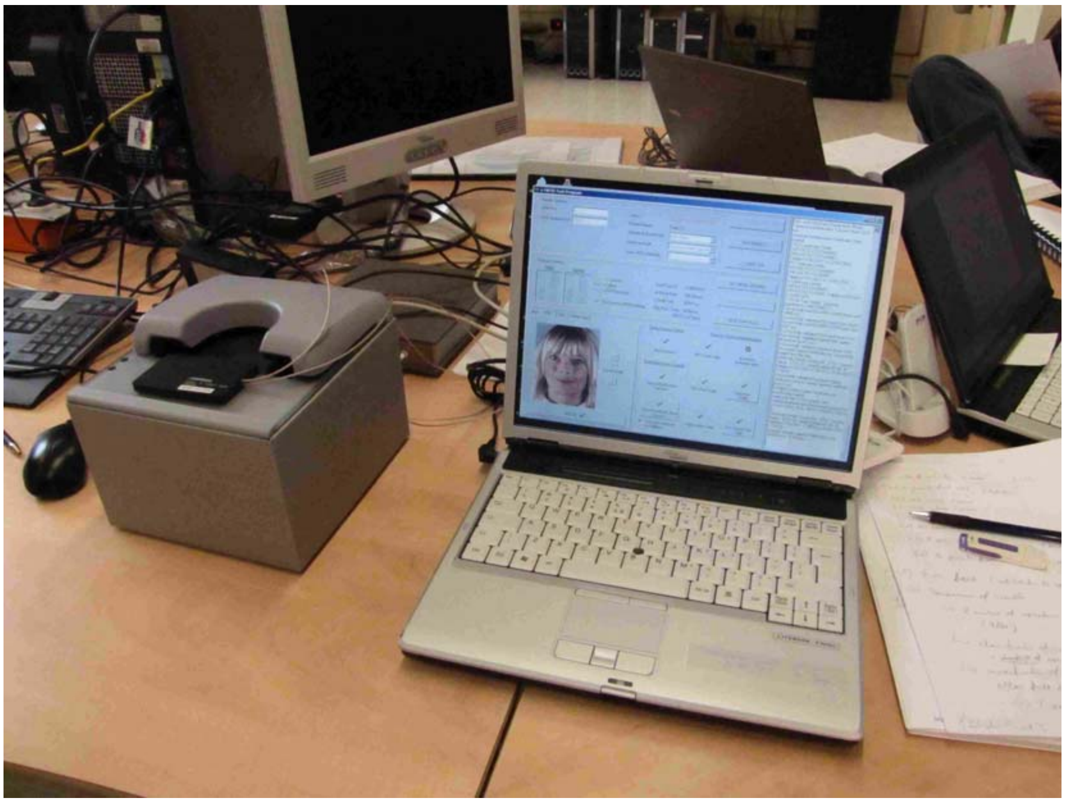


Figure 30: Setup with RTE inspection system

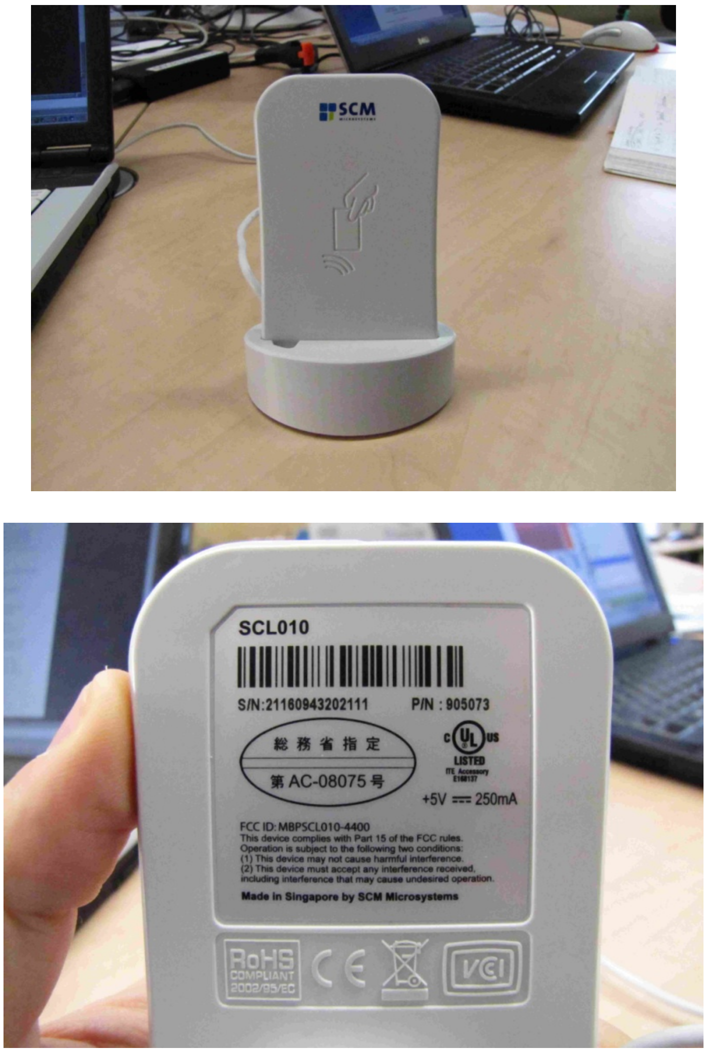

SCM
SCL010
S/N:21160943202111
P/N : 905073
総務省指定
第 AC-08075 号
LISTED
+5V ═══ 250mA
FCC ID: MBPSCL010-4400
This device complies with Part 15 of the FCC rules.
Operation is subject to the following two conditions:
(1) This device may not cause harmful interference.
(2) This device must accept any interference received,
including interference that may cause undesired operation.
Made in Singapore by SCM Microsystems
RoHS COMPLIANT 2002/95/EC
CE
VCI

Figure 31: SCL010 Reader

9.3.3 Conclusion

This third validation campaign has allowed us to test the prototype versus several new inspection systems provided by the JRC Biometric Lab facilities.

The team has validated the fixes of the issues discovered during the previous validation campaigns, mainly in the security profiles.

Some comments concerning Automatic Interface Specification have been raised and transmitted to the corresponding standardisation body. The main problem is that this interface does not give the opportunity to report intermediate success verdicts.

For instance, when the ePassport inspection is globally unsuccessful, it is only possible to report the failed steps, but not the passed steps. As a consequence, the test system is not able to distinguish if a step has been successfull or not performed.

9.4 Final Validation Campaign

9.4.1 Introduction

The final evaluation was performed on June 13, 2011 at the ETSI premises.

The STF experts who have participated to the validation campaign were Z. Riha (Masaryk University) as the evaluator with the support of L. Velez (ETSI) and A. Berge (AMB Consulting).

First of all the updated profile configuration was loaded into the system.

Then L6 testcases and selected L7 test cases were performed. Both the manual result interface and automated certificate distribution and result reported was tested. The Golden Reader Tool 2.9.4 and the simple inspection system was used during the tests. The tests were running as expected. One small bug in the report handling of automated result reporting was fixed during the final evaluation session.

9.4.2 Evaluation of configuration files

Test case:	ISO7816_A_02
Source configuration:	CFG.DFLT.BAC
Modified files:	The BAC configuration consists of the following files: EF.COM, EF.SOD, DG1, DG2.
Modifications:	-
Evaluation:	The configuration is as described in the test specification.

Test case:	ISO7816_B_07
Source configuration:	CFG.DFLT.BAC
Modified files:	-.
Modifications:	-
Evaluation:	No modifications to CFG.DFLT.BAC required.

Test case:	ISO7816_B_08
Source configuration:	CFG.DFLT.BAC
Modified files:	-.
Modifications:	-
Evaluation:	No modifications to CFG.DFLT.BAC required.

Test case:	ISO7816_C_02
Source configuration:	CFG.DFLT.BAC
Modified files:	DG2, EF.SOD
Modifications:	DG2: replaced facial image so that the size of DG2 exceeds 32kB.
Evaluation:	The configuration is as described in the test specification.

Test case:	ISO7816_C_03
Source configuration:	CFG.DFLT.BAC
Modified files:	DG2, EF.SOD
Modifications:	DG2: facial image file truncated after 200 bytes, DG2 truncated as well.
Evaluation:	The configuration is as described in the test specification.

Test case:	ISO7816_D_02
Source configuration:	CFG.DFLT.EAC
Modified files:	The EAC configuration consists of the following files: EF.COM, EF.SOD, EF.CVCA (+ trustpoint), DG1, DG2, DG3, DG4, DG14 (+ CA private key).
Modifications:	-
Evaluation:	The configuration is as described in the test specification.

Test case:	ISO7816_D_04
Source configuration:	CFG.DFLT.EAC
Modified files:	DG14 (+ additional CA private key), EF.SOD
Modifications:	DG14: Two keys with key references present.
Evaluation:	The configuration is as described in the test specification (two keys present with Key references 01 and 02).

Test case:	ISO7816_D_11
Source configuration:	CFG.DFLT.EAC
Modified files:	DG14, EF.SOD
Modifications:	DG14: wrong OID in SubjectPublicKeyInfo.
Evaluation:	The configuration is as described in the test specification.

Test case:	ISO7816_E_06
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.CVCA (+ the trustpoint)
Modifications:	EF.CVCA: DETESTCVCA00008 (trustpoint with ECDSA-SHA224 is used).
Evaluation:	The configuration is as described in the test specification.

Test case:	ISO7816_E_08
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.CVCA
Modifications:	EF.CVCA: DETESTCVCA00002 (not matching the trustpoint name).
Evaluation:	The configuration is as described in the test specification.

Test case:	ISO7816_E_11
Source configuration:	CFG.DFLT.EAC
Modified files:	-
Modifications:	Internally stored document number is wrong.
Evaluation:	No need to update the configuration.

Test case:	ISO7816_E_12
Source configuration:	CFG.DFLT.EAC
Modified files:	-
Modifications:	-
Evaluation:	No modifications to CFG.DFLT.EAC required.

Test case:	ISO7816_E_18
Source configuration:	CFG.DFLT.EAC
Modified files:	IS certificate supplied to the IS (IS_Cert_11)
Modifications:	Incorrect signature in IS_Cert_11
Evaluation:	No modifications to CFG.DFLT.EAC required, IS_Cert_11 is implemented as specified in the documentation.

Test case:	ISO7816_E_28
Source configuration:	CFG.DFLT.EAC
Modified files:	DG14, EF.SOD
Modifications:	DG14: CA based on 224bit ECDH.
Evaluation:	The configuration is as described in the test specification.

Test case:	ISO7816_F_05
Source configuration:	CFG.DFLT.EACAA
Modified files:	The configuration CFG.DFLT.EACAA expands CFG.DFLT.EAC by adding DG15 (+ AA private key) and updates the EF.SOD.
Modifications:	-
Evaluation:	The configuration is as described in the test specification.

Test case:	LDS_A_03
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.COM
Modifications:	EF.COM: wrong length in the first TLV of EF.COM
Evaluation:	The configuration is as described in the test specification.

Test case:	LDS_A_04
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.COM
Modifications:	EF.COM: wrong version of LDS (3.0)
Evaluation:	The configuration is as described in the test specification.

Test case:	LDS_B_11
Source configuration:	CFG.DFLT.BAC
Modified files:	DG1, EF.SOD
Modifications:	DG1: modified name
Evaluation:	The configuration is as described in the test specification.

Test case:	LDS_B_22
Source configuration:	CFG.DFLT.BAC
Modified files:	DG1, EF.SOD
Modifications:	DG1: wrong checksum of optional data
Evaluation:	The configuration is as described in the test specification.

Test case:	LDS_B_25
Source configuration:	CFG.DFLT.BAC
Modified files:	DG1, EF.SOD
Modifications:	DG1: incomplete birth date
Evaluation:	The configuration is as described in the test specification.

Test case:	LDS_C_03
Source configuration:	CFG.DFLT.EAC
Modified files:	DG2, EF.SOD
Modifications:	DG2: additional feature points
Evaluation:	The configuration is as described in the test specification (one feature point is added, photo is full frontal image).

Test case:	LDS_C_09
Source configuration:	CFG.DFLT.EAC
Modified files:	DG2, EF.SOD
Modifications:	DG2: missing format owner in BHT
Evaluation:	The configuration is as described in the test specification.

Test case:	LDS_C_13
Source configuration:	CFG.DFLT.EAC
Modified files:	DG2, EF.SOD
Modifications:	DG2: incorrect biometric type in BHT
Evaluation:	The configuration is as described in the test specification (biometric type is 01).

Test case:	LDS_C_19
Source configuration:	CFG.DFLT.EAC
Modified files:	DG2, EF.SOD
Modifications:	DG2: incorrect hair color in FIB
Evaluation:	The configuration is as described in the test specification (hair color is 08 - green).

Test case:	LDS_D_03
Source configuration:	CFG.DFLT.EAC
Modified files:	DG3, EF.SOD
Modifications:	DG3: 3 images (WSQ coding)
Evaluation:	The configuration is as described in the test specification (3 WSQ fingerprint images stored).

Test case:	LDS_D_12
Source configuration:	CFG.DFLT.EAC
Modified files:	DG3, EF.SOD
Modifications:	DG3: Incorrect biometric subtype in BHT of first instance.
Evaluation:	The configuration is as described in the test specification (biometric subtype is FEh in the first instance and 0Ah in the second instance).

Test case:	LDS_D_15
Source configuration:	CFG.DFLT.EAC
Modified files:	DG3, EF.SOD
Modifications:	DG3: missing fingerprint image in the second instance.
Evaluation:	The configuration is as described in the test specification (the tag 5F2E is missing in the second instance).

Test case:	LDS_E_01
Source configuration:	CFG.DFLT.EAC
Modified files:	DG4, EF.SOD
Modifications:	DG4: Two JPEG2000 iris images.
Evaluation:	The configuration is as described in the test specification

Test case:	LDS_E_02
Source configuration:	CFG.DFLT.EAC
Modified files:	DG4, EF.SOD
Modifications:	DG4: Two RAW iris images.
Evaluation:	The configuration is as described in the test specification.

Test case:	LDS_E_09
Source configuration:	CFG.DFLT.EAC
Modified files:	DG4, EF.SOD
Modifications:	DG4: Not allowed format type in the first instance.
Evaluation:	The configuration is as described in the test specification (value is 000F).

Test case:	LDS_F_04
Source configuration:	CFG.DFLT.EAC
Modified files:	DG14, EF.SOD
Modifications:	DG14: Not allowed chip authentication public key OID.
Evaluation:	The configuration is as described in the test specification (0.4.0.127.0.7.2.2.1.3).

Test case:	LDS_F_06
Source configuration:	CFG.DFLT.EAC
Modified files:	DG14, EF.SOD
Modifications:	DG14: Incorrect version element in ChipAuthenticationInfo.
Evaluation:	The configuration is as described in the test specification (The value is 0Ah).

Test case:	LDS_G_01
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.CVCA
Modifications:	EF.CVCA: First CAR not encoded in 42 tag.
Evaluation:	The configuration is as described in the test specification (The tag is 00h instead of 42h).

Test case:	LDS_G_02
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.CVCA
Modifications:	EF.CVCA: File empty.
Evaluation:	The configuration is as described in the test specification (File consists of a sequence of zeros 00h).

Test case:	LDS_H_03
Source configuration:	CFG.DFLT.EAC
Modified files:	-
Modifications:	-
Evaluation:	The standard configuration CFG.DFLT.EAC does not have to be modified.

Test case:	LDS_H_07
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: RSA-PKCS15-SHA512 with SHA512 for DG hash is used.
Evaluation:	The configuration is as described in the test specification (and signature is correct).

Test case:	LDS_H_22
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: SignedData uses MD5.
Evaluation:	The configuration is as described in the test specification

Test case:	LDS_H_24
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: Incorrect OID for ICAO OID.
Evaluation:	The configuration is as described in the test specification

Test case:	LDS_H_31
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: Missing digest algorithm.
Evaluation:	The configuration is as described in the test specification (the element of digestAlgorithm is missing in SignerInfo).

Test case:	LDS_H_32
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: incorrect message digest in signed attributes of SignerInfo
Evaluation:	The configuration is as described in the test specification

Test case:	LDS_H_33
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: Missing messageDigest signed attribute in SignerInfo.
Evaluation:	The configuration is missing.

Test case:	LDS_H_37
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: Incorrect signature.
Evaluation:	The configuration is as described in the test specification (signature value increased by 1).

Test case:	LDS_H_45
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: Added hash value for DG10.
Evaluation:	The configuration is as described in the test specification (added hash for DG10).

Test case:	LDS_H_51
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: Modified DS certificate.
Evaluation:	The configuration is as described in the test specification (in DS cert the certified algorithm is RSAsha1, but signed by RSA-PSS).

Test case:	LDS_H_54
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: Modified issuer of DS certificate.
Evaluation:	The configuration is as described in the test specification (country changed from DE to UTO).

Test case:	LDS_H_63
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: Modified keyUsage in DS certificate.
Evaluation:	The configuration is as described in the test specification (digitalSignature bit not present).

Test case:	LDS_H_71
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: ECDSA 256 bit with SHA256.
Evaluation:	The configuration is as described in the test specification (OID 1.2.840.10045.4.3.2 is used).

Test case:	LDS_H_75
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: Different valid algorithms in DS.
Evaluation:	The configuration is as described in the test specification (RSA-sha1 vs. RSA-PSS).

Test case:	LDS_H_83
Source configuration:	CFG.DFLT.EAC
Modified files:	EF.SOD
Modifications:	EF.SOD: RSA-PKCS15-SHA512.
Evaluation:	The configuration is as described in the test specification (RSA-PKCS15-SHA512, hash of DGs based on SHA512).

Test case:	LDS_I_01
Source configuration:	CFG.DFLT.EACAA
Modified files:	DG15, EF.SOD
Modifications:	DG15: Wrong tag of the DG.
Evaluation:	The configuration is as described in the test specification (DG tag is replaced with 70h).

9.4.3 Validation of the PA certificates

1) The default CSCA certificate: CSCA_RSA_PSS:
   a) Version 3, SN: 01 2f ba c7 f8 6d, RSA-PSS, sha256
   b) CN = ETSI CS, OU = Country Signer, O = ETSI, C = DE
   c) Issuer = Subject
   d) Validity: 4 May 2011 - 7 Apr 2016
   e) RSA 3072b, Key id = f5 f7 39 0a …
   f) Key usage: CertSign, OffileCRL, CRL
   g) Basic constraints: CA=true, length=0
2) Special CSCA certificate: CSCA_RSA_PKCS1_SHA512:
   a) Version 3, SN: 01 2f ba e1 ba b9, sha512RSA
   b) CN = ETSI CS, OU = Country Signer, O = ETSI, C = DE
   c) Issuer = Subject
   d) Validity: 4 May 2011 - 7 Apr 2016
   e) RSA 3072b, Key id = 57 41 cd ed …
   f) Key usage: CertSign, OffileCRL, CRL
   g) Basic constraints: CA=true, length=0
3) Special CSCA certificate: CSCA_ECDSA:
   a) Version 3, SN: 01 2f bb 10 2d 69, ECDSA sha256 (ANSI OID)
   b) CN = ETSI CS, OU = Country Signer, O = ETSI, C = DE
   c) Issuer = Subject
   d) Validity: 4 May 2011 - 7 Apr 2016
   e) ECDSA 256b, Key id = 55 99 1a 74…
   f) Key usage: CertSign, OffileCRL, CRL
   g) Basic constraints: CA=true, length=0
4) The default DS cert: DS_RSA_PSS:
   a) Version 3, SN: 01 2f ba c8 11 73, RSA-PSS, sha256
   b) Subject: CN = ETSI DS, OU = Document Signer, O = ETSI, C = DE
   c) Validity: 4 May 2011 - 24 May 2014
   d) RSA 2048b.
   e) Key id CA: f5 f7 39 0a...
   f) Key id DS: 9a c4 21 de...
   g) KeyUsage: DigSig

5) Special DS certificate: DS_RSA_PKCS1_SHA512:
    a) Version 3, SN: 01 2f ba e1 c6 62, sha512RSA
    b) Subject: CN = ETSI DS, OU = Document Signer, O = ETSI, C = DE
    c) Validity: 4 May 2011 - 24 May 2014
    d) RSA 2048b.
    e) Key id CA: 57 41 cd ed...
    f) Key id DS: 27 8c 48 a3...
    g) KeyUsage: DigSig
6) Special DS certificate: DS_RSA_PSS_WRONG_KU:
    a) Version 3, SN: 01 2f ba c8 11 73, RSA-PSS, sha256
    b) Subject: CN = ETSI DS, OU = Document Signer, O = ETSI, C = DE
    c) Validity: 4 May 2011 - 24 May 2014
    d) RSA 2048b.
    e) Key id CA: f5 f7 39 0a...
    f) Key id DS: 9a c4 21 de...
    g) KeyUsage: none
7) Special DS certificate: DS_ECDSA:
    a) Version 3, SN: 01 2f bb 10 2d f5, ECDSA sha256 (ANSI)
    b) Subject: CN = ETSI DS, OU = Document Signer, O = ETSI,C = DE
    c) Validity: 4 May 2011 - 24 May2014
    d) ECDSA 256b.
    e) Key id CA: 55 99 1a 74...
    f) Key id DS: 71 57 79 a7...
    g) KeyUsage: DigSig

9.4.4 Validation of the TA certificates

1) Standard CVCA certificate: CVCA_Cert_01:
    a) DETESTCVCA00001
    b) PK: ECDSA-SHA-256 (256b)
    c) CVCA: reading DG3/DG4
    d) 4 May 2011 - 24 May 2014
    e) Signature ok
2) Special CVCA certificate: CVCA_Cert_08:
    a) DETESTCVCA0008
    b) PK: ECDSA-SHA-224 (256b)

   c) CVCA: reading DG3/DG4
   d) 4 May 2011 - 24 May 2014
   e) Signature ok

3) Standard DV certificate: DV_Cert_01:
   a) DETESTCVCA00001/DETESTDV00001
   b) PK: ECDSA-SHA-256 (256b)
   c) Domestic DV: reading DG3/DG4
   d) 4 May 2011 - 24 May 2014
   e) Signature ok

4) Special DV certificate: DV_Cert_08:
   a) DETESTCVCA00008/DETESTDV00008
   b) PK: ECDSA-SHA-224 (256b)
   c) Domestic DV: reading DG3/DG4
   d) 15 Apr 2010 - 15 May 2010
   e) Signature ok

5) Standard IS certificate: IS_Cert_01:
   a) DETESTDV00001/DETESTIS00001
   b) PK: ECDSA-SHA-256 (256b)
   c) IS: reading DG3/DG4
   d) 4 May 2011 - 18 May 2011
   e) Signature ok

6) Special IS certificate: IS_Cert_08:
   a) DETESTDV00008/DETESTIS00008
   b) PK: ECDSA-SHA-224 (256b)
   c) IS: reading DG3/DG4
   d) 4 May 2011 - 18 May 2011
   e) Signature ok

7) Wrong IS certificate: IS_Cert_11:
   a) DETESTDV00001/DETESTIS00001
   b) PK: ECDSA-SHA-256 (256b)
   c) IS: reading DG3/DG4
   d) 4 May 2011 - 18 May 2011
   e) Signature wrong

8) Standard IS private key: IS_Key_01:

   a) ECDSA 256 bits

9) Special IS private key: IS_Key_08:

   a) ECDSA 256 bits

10 Lab Procedure

This clause depicts the test bed architecture and introduces the procedures to follow in order to execute a test campaign properly.

10.1 Test bed description

This clause depicts with details, the technical architecture of the test bed.

10.1.1 Hardware description

The test bed hardware includes:

- A computer with Microsoft Windows Operating System (preferably Windows XP): to execute the TTCN-3 execution environment;
- An ISO/IEC 14443 [i.13] Type A and B card simulator: such as the Comprion CLT One as the e-passport simulator.

The clauses below provide technical description of the Test System prototype equipments:

The computer

It is a Dell Inspiron with an Intel Dual core 3 GHz, 4 GB Dual Channel DDR3 memory, 500 Gb SATA hard disk, and some additional standard equipments as DVD reader running Microsoft Windows XP SP 3.

The Comprion hardware

The ISO/IEC 14443 [i.13] Type A and B card simulator (CLT One) from Comprion is associated with a Type A antenna . The CLT One equipment is connected to the computer running the TTCN-3 environment.

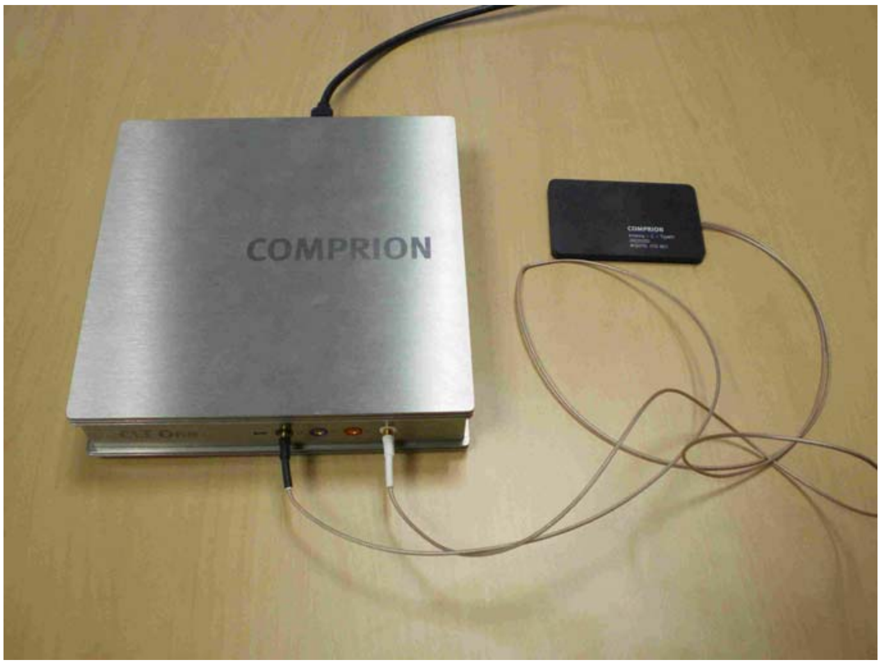


Figure 32: Comprion CLT One and its antenna

10.1.2 Software description

This clause describes, with details (version, build, etc.), softwares used by the test bed.

TTCN-3 Execution environment

The ATS is executed on TTworkbench Version 1.1.10, build id: 2009.12.14.17.16, provided by Testing Technologies. Install TTworkbench using instructions provided by Testing Technologies.

Project workspace is then installed by unzipping the file ePassport_workspace.zip in the folder of your choice. Select this workspace when running TTworkbench (see clause 10.2.2, step 2).

Comprion "CLT One" tool

The version of CLT One tool is 2.1.0.0. The tool installs the Comprion driver version is 10/07/2009,1.51.0.4702.

Security Profiles

The security profiles described in document DMI/MTS-00127 are correctly installed in the Test system and valid (certificates validity). The installation path is referenced in the module parameter PXT_EPASSPORT_DATA_ROOT.

10.2 Test Execution procedure

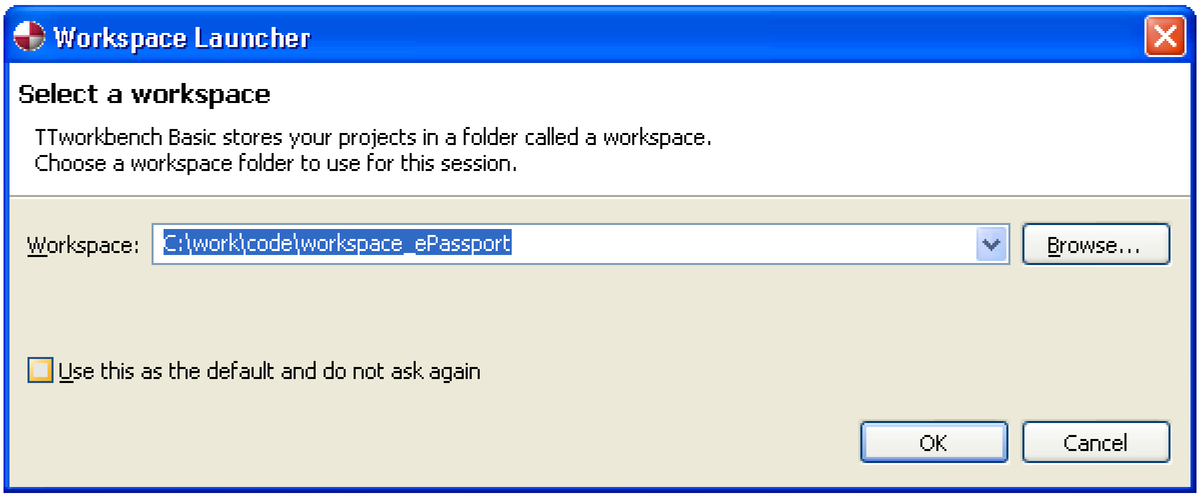
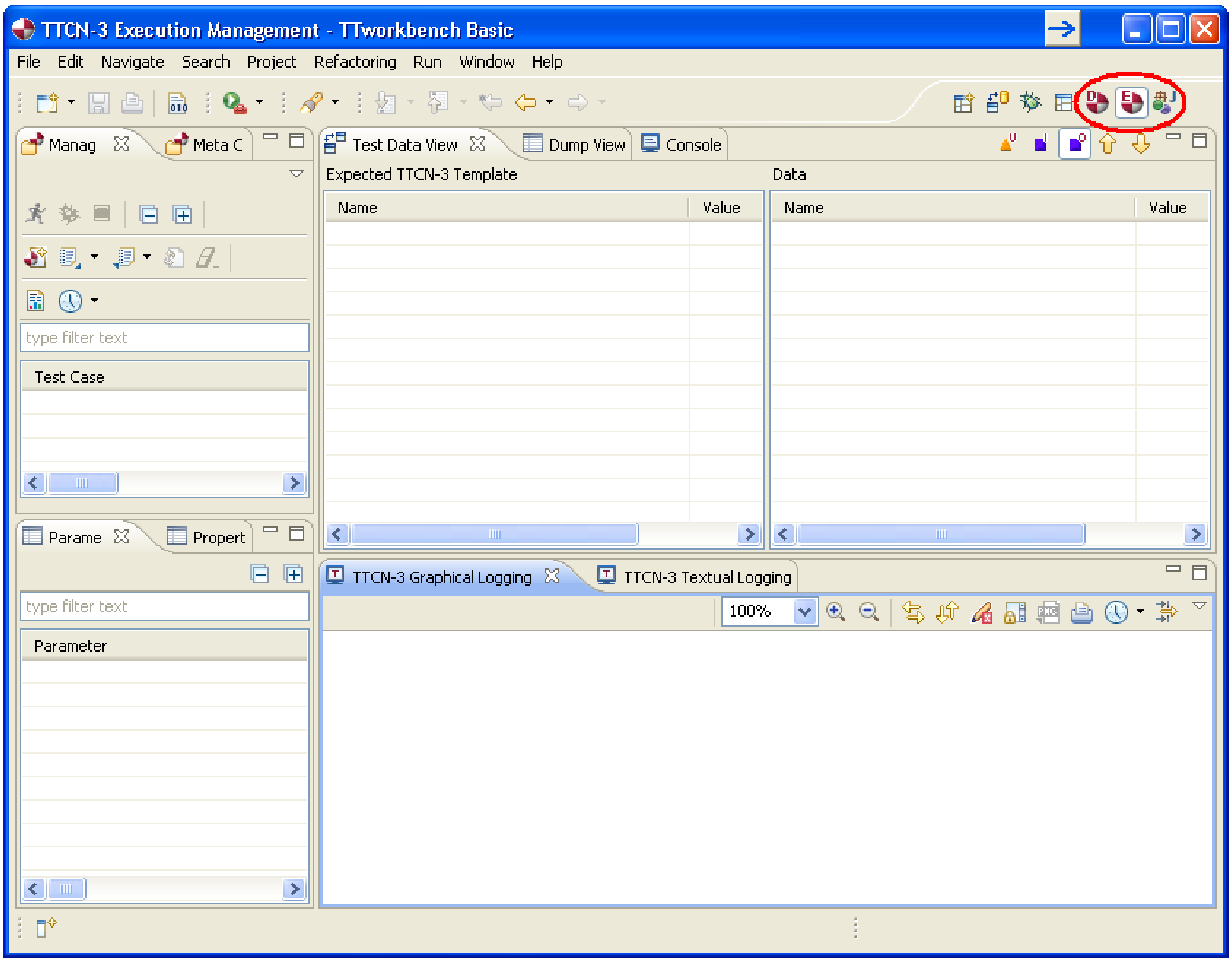
10.2.1 IUT Configuration

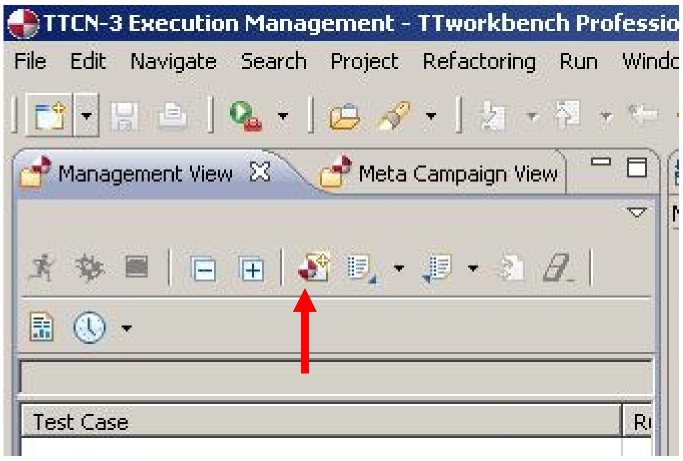
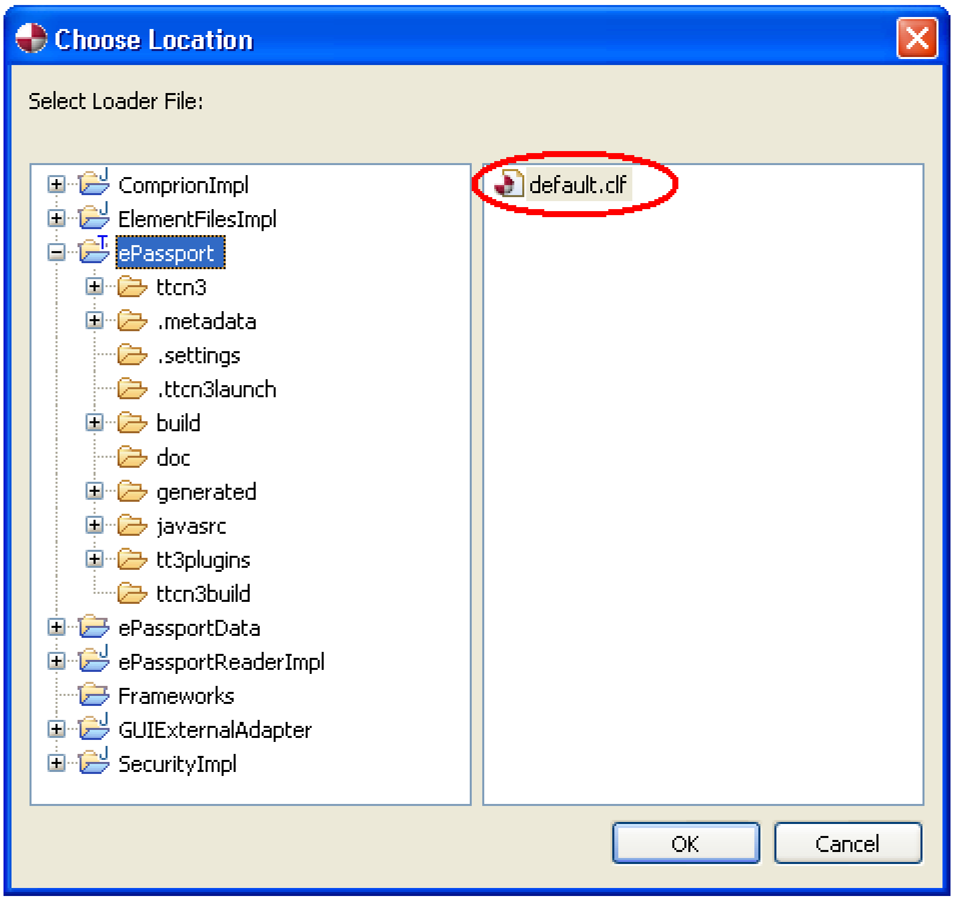
Table 10: IUT configuration

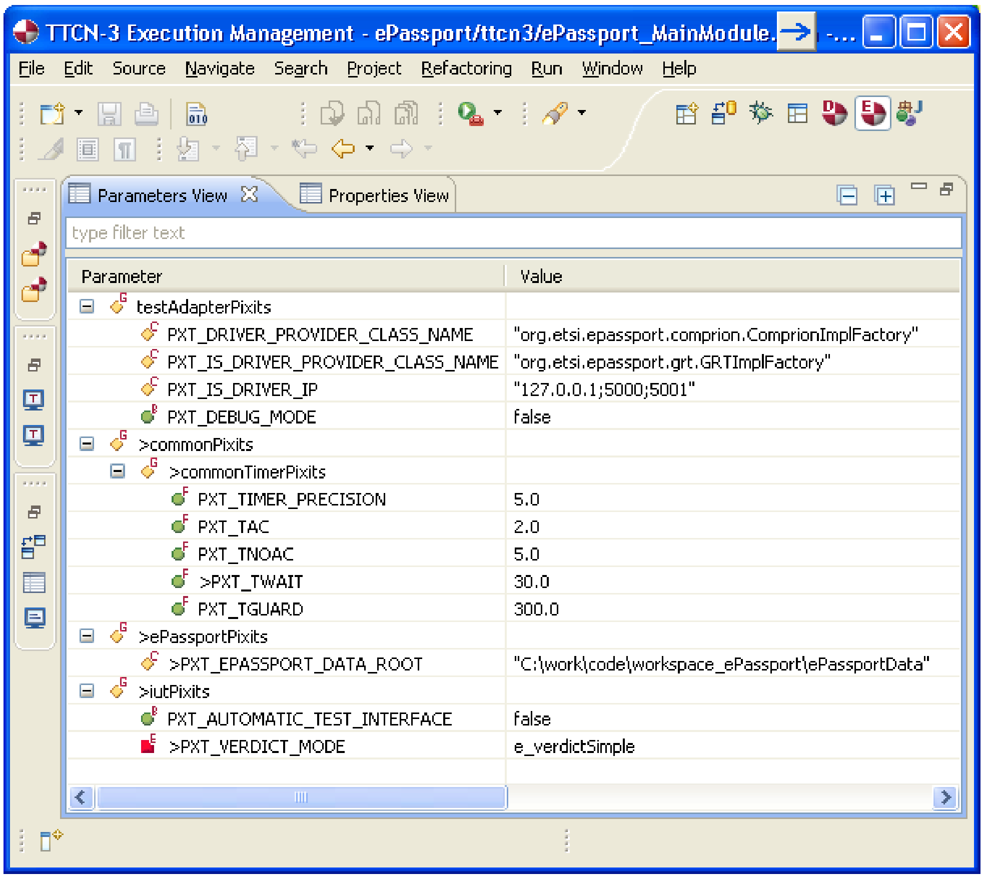
Steps	Actions
1	Print the PIXIT Proforma for ePassport (see Annex B)
2	Fill clauses B.1, B.3, B.4 and B.5
3	Print the PCTR Proforma for ePassport (see Annex C)
4	Fill the clause C.1
5	Put the Test System antenna on the IUT's card reader.
6	If IUT is a full page reader, print MRZs and place it in front of the IUT optical camera. Otherwise, provide MRZ info as required by IUT.
7	If IUT does not support Automatic interface: - Add the right Country Signer certificates, depending of the test cases to be executed during your test campaign. - Add the Country Verifying Certificates chain (CVCA/DV/IS certificates, IS private key).

10.2.2 Test System Configuration

Table 11: Test system configuration

1	Start TTworkbench tool.
2	Select the ePassport workspace as defined in the Testbed Description.  Workspace Launcher Select a workspace TTworkbench Basic stores your projects in a folder called a workspace. Choose a workspace folder to use for this session. Workspace: C:\work\code\workspace_ePassport Browse... Use this as the default and do not ask again OK Cancel
3	Switch to TTCN-3 Execution perspective.  TTCN-3 Execution Management - TTworkbench Basic File Edit Navigate Search Project Refactoring Run Window Help Manag Meta C type filter text Test Case Parame Propert type filter text Parameter Test Data View Dump View Console Expected TTCN-3 Template Name Value Data Name Value TTCN-3 Graphical Logging TTCN-3 Textual Logging 100%

4	Click on the icon pointed by the red arrow, select the item 'Import Test Campaign'.  TTCN-3 Execution Management - TTworkbench Professio File Edit Navigate Search Project Refactoring Run Windo
5	Select the file default.clf, as shown below:  Click on OK button to validate your choice.

6	Identify the PIXITs view shown below. 

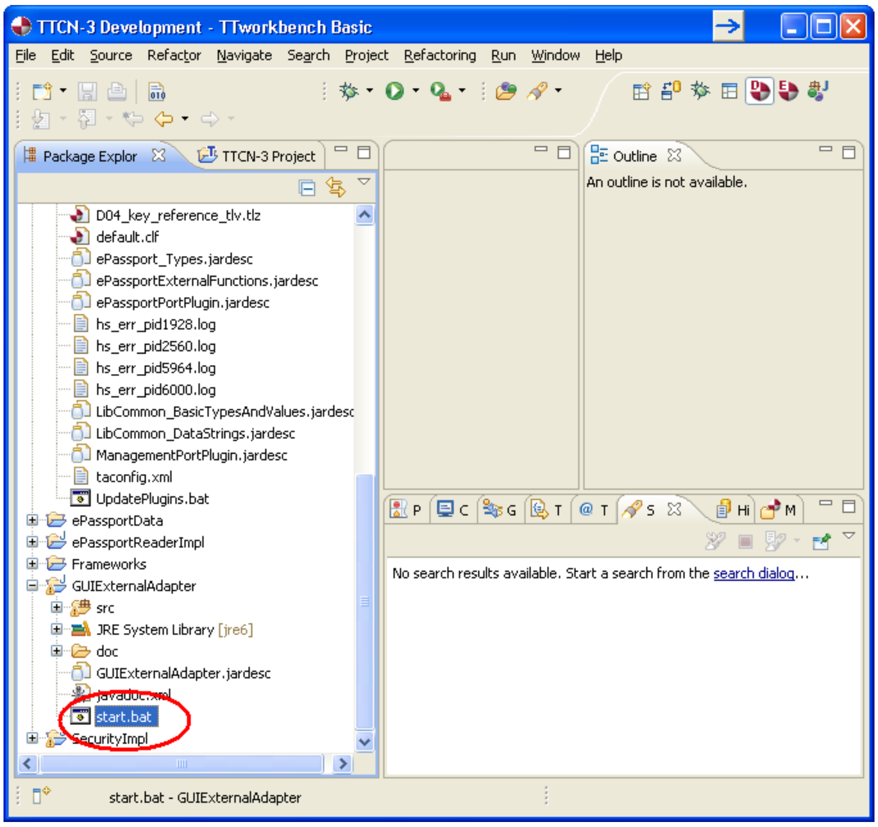
TTCN-3 Execution Management - ePassport/ttcn3/ePassport_MainModule. -...

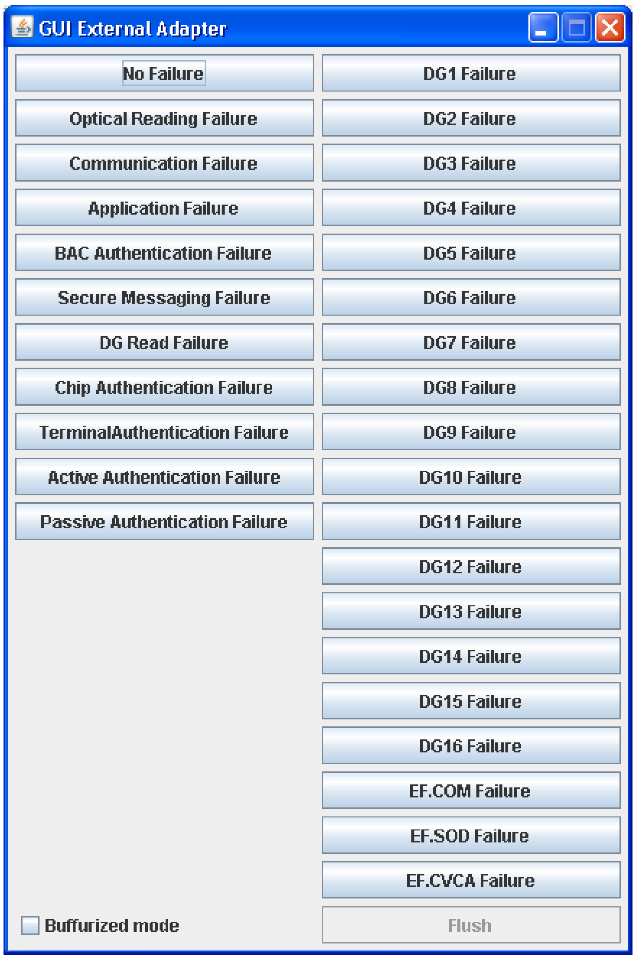
File Edit Source Navigate Search Project Refactoring Run Window Help

Parameters View | Properties View

type filter text

Parameter	Value
testAdapterPixits	
PXT_DRIVER_PROVIDER_CLASS_NAME	"org.etsi.epassport.comprion.ComprionImplFactory"
PXT_IS_DRIVER_PROVIDER_CLASS_NAME	"org.etsi.epassport.grt.GRTImplFactory"
PXT_IS_DRIVER_IP	"127.0.0.1;5000;5001"
PXT_DEBUG_MODE	false
>commonPixits	
>commonTimerPixits	
PXT_TIMER_PRECISION	5.0
PXT_TAC	2.0
PXT_TNOAC	5.0
>PXT_TWAIT	30.0
PXT_TGUARD	300.0
>ePassportPixits	
>PXT_EPASSPORT_DATA_ROOT	"C:\work\code\workspace_ePassport\ePassportData"
>iutPixits	
PXT_AUTOMATIC_TEST_INTERFACE	false
>PXT_VERDICT_MODE	e_verdictSimple

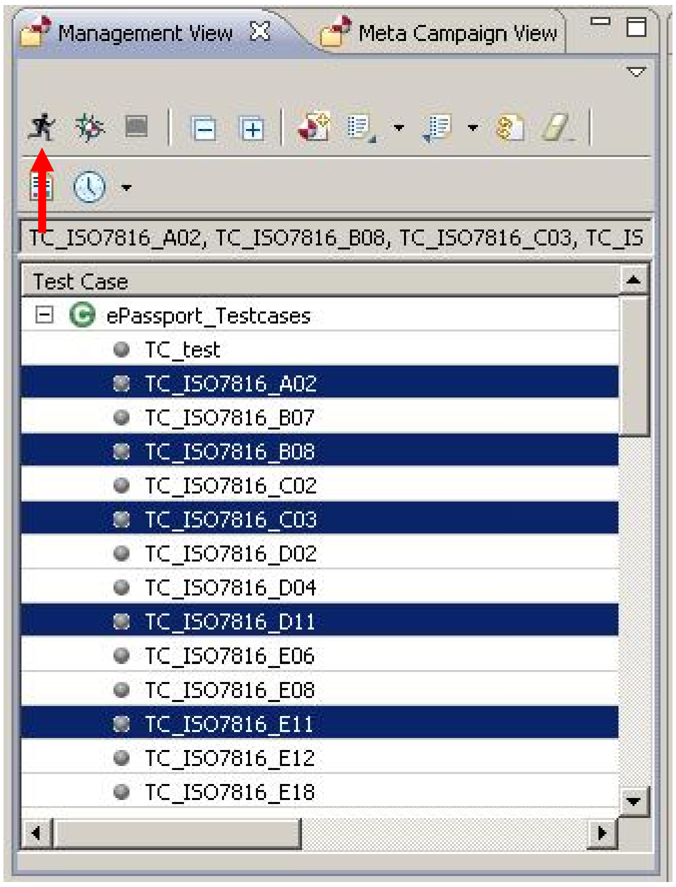
7	Modify the PIXITs according to IUT PCTR filled by IUT vendors. The character ">" will appear in front of each modified PIXIT (see screenshot above) reference to the ATS chapter. Indicate the Security profiles path into the PXT_EPASSPORT_DATA_ROOT . Optionally, you could activate the debug mode modifying the PIXIT 'PXT_DEBUG_MODE' By default, the debug mode is disabled (value set to false), this means no traces will be generated.
8	If IUT does not support the Automatic Interface, launch the operator GUI using the file start.bat as indicated in the following screenshot (accessible from the Development perspective). It is used to reports Inspection System verdicts to the Test System.  TTCN-3 Development - TTworkbench Basic File Edit Source Refactor Navigate Search Project Refactoring Run Window Help Package Explor / TTCN-3 Project D04_key_reference_tlv.tlz default.clf ePassport_Types.jardesc ePassportExternalFunctions.jardesc ePassportPortPlugin.jardesc hs_err_pid1928.log hs_err_pid2560.log hs_err_pid5964.log hs_err_pid6000.log LibCommon_BasicTypesAndValues.jardesc LibCommon_DataStrings.jardesc ManagementPortPlugin.jardesc taconfig.xml UpdatePlugins.bat ePassportData ePassportReaderImpl Frameworks GUIExternalAdapter src JRE System Library [jre6] doc GUIExternalAdapter.jardesc javadoc.xml start.bat SecurityImpl Outline An outline is not available. P C G T T S Hi M No search results available. Start a search from the search dialog... start.bat - GUIExternalAdapter

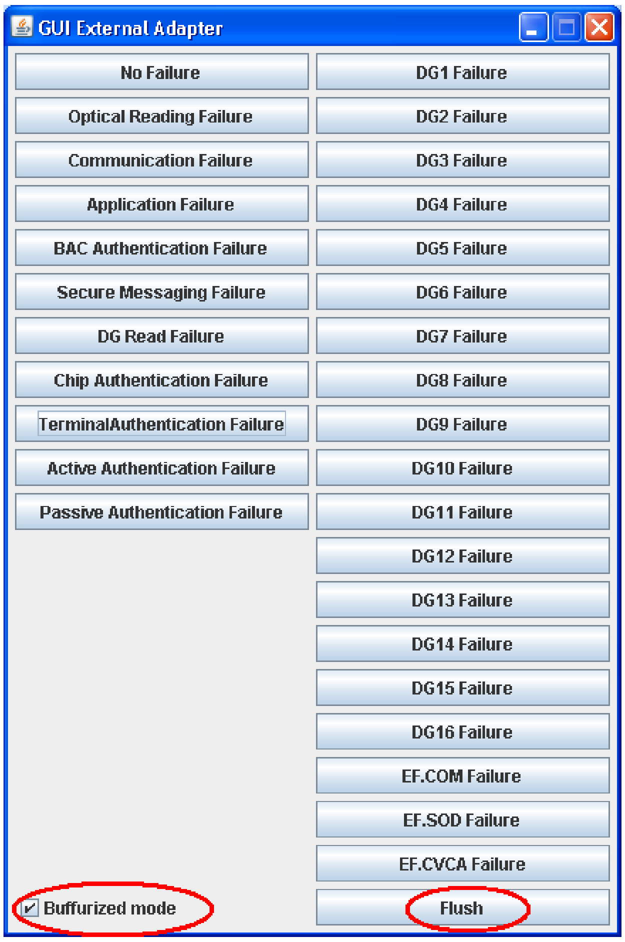

GUI External Adapter
No Failure
DG1 Failure
Optical Reading Failure
DG2 Failure
Communication Failure
DG3 Failure
Application Failure
DG4 Failure
BAC Authentication Failure
DG5 Failure
Secure Messaging Failure
DG6 Failure
DG Read Failure
DG7 Failure
Chip Authentication Failure
DG8 Failure
TerminalAuthentication Failure
DG9 Failure
Active Authentication Failure
DG10 Failure
Passive Authentication Failure
DG11 Failure
DG12 Failure
DG13 Failure
DG14 Failure
DG15 Failure
DG16 Failure
EF.COM Failure
EF.SOD Failure
EF.CVCA Failure
Buffurized mode
Flush

10.2.3 Test Execution

This should be started only if the 'Configuration' procedure is achieved properly. During this stage, you will execute the test cases specified by the test campaign.

Table 12: Test execution

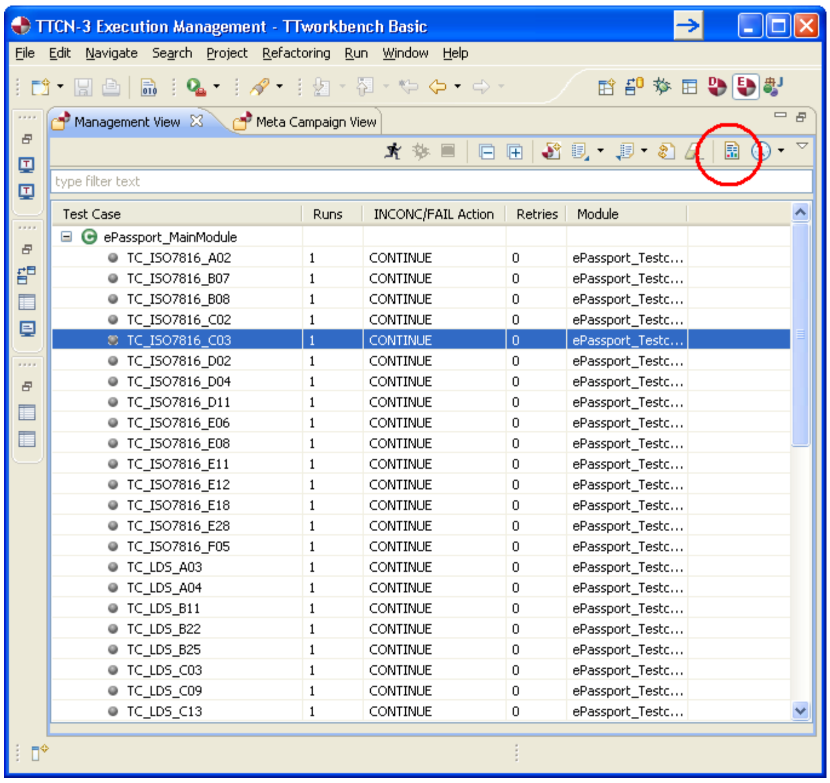
Steps	Actions
1	Select one or several test cases to execute using Ctrl and Shift keys while selecting the test cases with your mouse.
2	Check that you got the this kind of appearance:  Management View / Meta Campaign View TC_ISO7816_A02, TC_ISO7816_B08, TC_ISO7816_C03, TC_IS Test Case ePassport_Testcases TC_test TC_ISO7816_A02 TC_ISO7816_B07 TC_ISO7816_B08 TC_ISO7816_C02 TC_ISO7816_C03 TC_ISO7816_D02 TC_ISO7816_D04 TC_ISO7816_D11 TC_ISO7816_E06 TC_ISO7816_E08 TC_ISO7816_E11 TC_ISO7816_E12 TC_ISO7816_E18 Note that if you do not select any test case, all test cases of the campaign will be executed.
3	Click on the 'Execute Tests' icon (red arrow in screenshot above) to start execution of the selected test cases.
4	Use 'TTCN-3 Graphical Logging' to visualize the test case execution.
5	If IUT does not support Automatic Interface, the Inspection System verdicts have to be reported using the Operator GUI. It is possible to select several verdicts using flush buffer.

	 GUI External Adapter No Failure DG1 Failure Optical Reading Failure DG2 Failure Communication Failure DG3 Failure Application Failure DG4 Failure BAC Authentication Failure DG5 Failure Secure Messaging Failure DG6 Failure DG Read Failure DG7 Failure Chip Authentication Failure DG8 Failure TerminalAuthentication Failure DG9 Failure Active Authentication Failure DG10 Failure Passive Authentication Failure DG11 Failure DG12 Failure DG13 Failure DG14 Failure DG15 Failure DG16 Failure EF.COM Failure EF.SOD Failure EF.CVCA Failure Buffurized mode Flush
6	Update your PCTR documents and test campaign report according to the test case execution results.
7	The test execution sequence can be relaunched without configuring the Test System.

10.2.4 Test Reporting

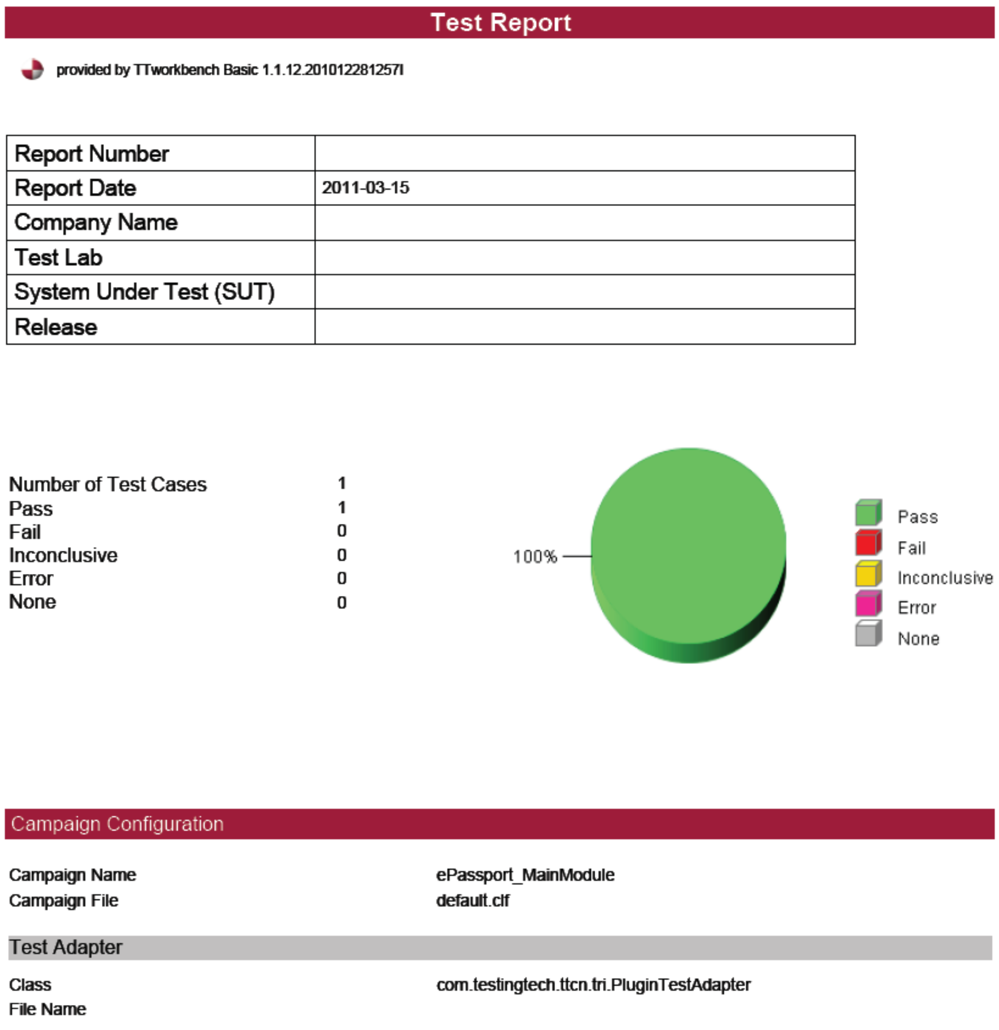
- The TTworkbench tool is able to generate HTML and PDF reports of a test campaign.
- It can also generate a picture jpg of the message flow.

Figure 33 shows the "reporting" icons to identify on the desktop.


TTCN-3 Execution Management - TTworkbench Basic
File Edit Navigate Search Project Refactoring Run Window Help
Management View
Meta Campaign View
type filter text

Test Case	Runs	INCONC/FAIL Action	Retries	Module
ePassport_MainModule				
TC_ISO7816_A02	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_B07	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_B08	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_C02	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_C03	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_D02	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_D04	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_D11	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_E06	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_E08	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_E11	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_E12	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_E18	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_E28	1	CONTINUE	0	ePassport_Testc...
TC_ISO7816_F05	1	CONTINUE	0	ePassport_Testc...
TC_LDS_A03	1	CONTINUE	0	ePassport_Testc...
TC_LDS_A04	1	CONTINUE	0	ePassport_Testc...
TC_LDS_B11	1	CONTINUE	0	ePassport_Testc...
TC_LDS_B22	1	CONTINUE	0	ePassport_Testc...
TC_LDS_B25	1	CONTINUE	0	ePassport_Testc...
TC_LDS_C03	1	CONTINUE	0	ePassport_Testc...
TC_LDS_C09	1	CONTINUE	0	ePassport_Testc...
TC_LDS_C13	1	CONTINUE	0	ePassport_Testc...

Figure 33



Test Report

provided by TTworkbench Basic 1.1.12.201012281257I

Report Number	
Report Date	2011-03-15
Company Name	
Test Lab	
System Under Test (SUT)	
Release	

Number of Test Cases	1
Pass	1
Fail	0
Inconclusive	0
Error	0
None	0

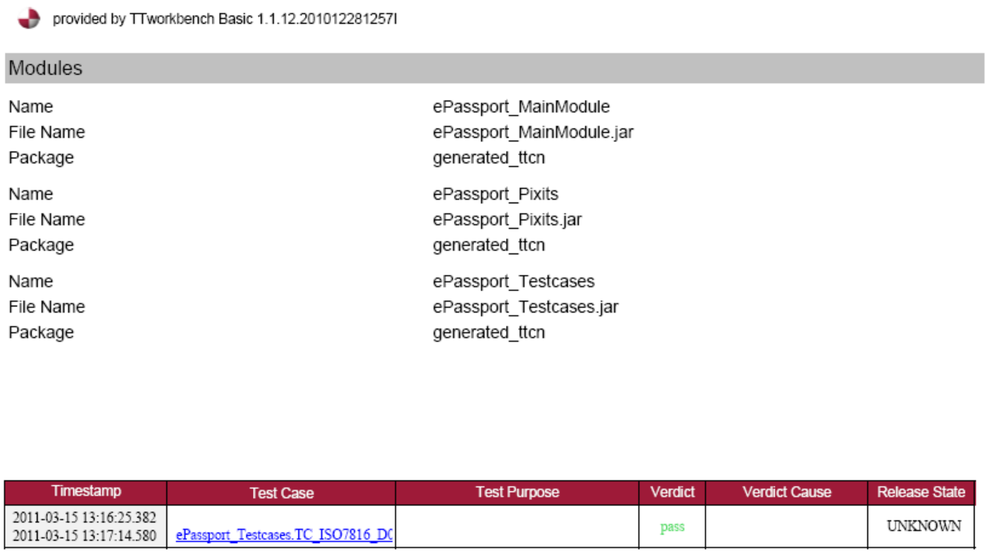
Campaign Configuration

Campaign Name	ePassport_MainModule
Campaign File	default.clf

Test Adapter

Class	com.testingtech.ttcn.tri.PluginTestAdapter
File Name	

Figure 34



provided by TTworkbench Basic 1.1.12.201012281257I

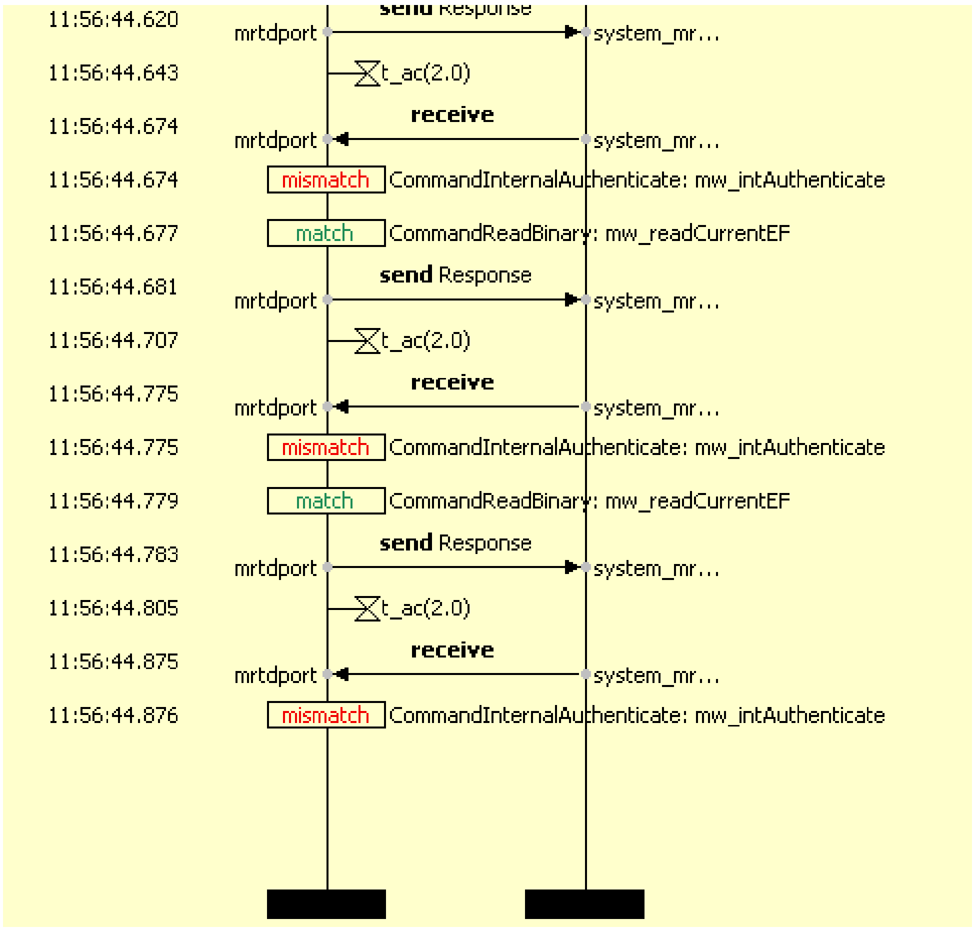
Modules

Name	ePassport_MainModule
File Name	ePassport_MainModule.jar
Package	generated_ttcn
Name	ePassport_Pixits
File Name	ePassport_Pixits.jar
Package	generated_ttcn
Name	ePassport_Testcases
File Name	ePassport_Testcases.jar
Package	generated_ttcn

Timestamp	Test Case	Test Purpose	Verdict	Verdict Cause	Release State
2011-03-15 13:16:25.382 2011-03-15 13:17:14.580	ePassport_Testcases.TC_ISO7816_D		pass		UNKNOWN

Figure 35

Snapshop of the messages flow, extracted from the reporting tool:



```
11:56:44.620   mrtdport  send Response  system_mr...
11:56:44.643   t_ac(2.0)
11:56:44.674   mrtdport  receive  system_mr...
11:56:44.674   mismatch  CommandInternalAuthenticate: mw_intAuthenticate
11:56:44.677   match  CommandReadBinary: mw_readCurrentEF
11:56:44.681   mrtdport  send Response  system_mr...
11:56:44.707   t_ac(2.0)
11:56:44.775   mrtdport  receive  system_mr...
11:56:44.775   mismatch  CommandInternalAuthenticate: mw_intAuthenticate
11:56:44.779   match  CommandReadBinary: mw_readCurrentEF
11:56:44.783   mrtdport  send Response  system_mr...
11:56:44.805   t_ac(2.0)
11:56:44.875   mrtdport  receive  system_mr...
11:56:44.876   mismatch  CommandInternalAuthenticate: mw_intAuthenticate
```

Figure 36

Annex A: TTCN-3 library modules

This ATS has been produced using the Testing and Test Control Notation (TTCN) according to ES 201 873-1 [i.1]. This test suite has been compiled error-free using three different commercial TTCN-3 compilers.

A.1 Electronic annex, zip file with TTCN-3 code

The TTCN-3 library modules, which form parts of the present technical report, are contained in archive tr_103200v010101p0.zip which accompanies the present document.

A.2 Electronic annex, zip file with HTML documentation

The HTML documentation, which forms parts of the present technical report, is contained in archive tr_103200v010101p0.zip which accompanies the present document. Start the index.htm file in any preferred web browser.

Annex B: Partial PIXIT proforma for ePassport

The PIXIT Proforma is based on ISO/IEC 9646-6 [i.11]. Any needed additional information can be found in this international standard document.

B.1 Identification summary

Table B.1

PIXIT Number:	
Test Laboratory Name:	
Date of Issue:	
Issued to:	

B.2 ATS summary

Table B.2

Protocol Specification:	ISO/IEC 7816-4 [i.12] BSI Technical Guideline TR-03110 1.11 [i.2]
Protocol to be tested:	ePassport Inspection System
ATS Specification:	
Abstract Test Method:	Clause 7

B.3 Test laboratory

Table B.3

Test Laboratory Identification:	
Test Laboratory Manager:	
Means of Testing:	
SAP Address:	

B.4 Client identification

Table B.4

Client Identification:	
Client Test manager:	
Test Facilities required:	

B.5 SUT

Table B.5

Name:	
Version:	
SCS Number:	
Machine configuration:	
Operating System Identification:	
IUT Identification:	
PICS Reference for IUT:	
Limitations of the SUT:	
Environmental Conditions:	

B.6 Protocol layer information

B.6.1 Protocol identification

Table B.6

Name:	Advanced Security Mechanisms for Machine Readable Travel Documents - Extended Access Control (EAC)
Version:	1.11

Annex C:
PCTR Proforma for ePassport

The PCTR proforma is based on ISO/IEC 9646-6 [i.11]. Any needed additional information can be found in this International standard document.

C.1 Identification summary

C.1.1 Protocol conformance test report

Table C.1

PCTR Number:	
PCTR Date:	
Corresponding SCTR Number:	
Corresponding SCTR Date:	
Test Laboratory Identification:	
Test Laboratory Manager:	
Signature:	

C.1.2 IUT identification

Table C.2

Name:	
Version:	
Protocol specification:	
Previous PCTR if any:	

C.1.3 Testing environment

Table C.3

PIXIT Number:	
ATS Specification:	
Abstract Test Method:	
Means of Testing identification:	
Date of testing:	
Conformance Log reference(s):	
Retention Date for Log reference(s):	

C.1.4 Limits and reservation

Additional information relevant to the technical contents or further use of the test report, or the rights and obligations of the test laboratory and the client, may be given here. Such information may include restriction on the publication of the report.

....
....
....
....
....

C.1.5 Comments

Additional comments may be given by either the client or the test laboratory on any of the contents of the PCTR, for example, to note disagreement between the two parties.

....
....
....
....
....

C.2 IUT Conformance status

This IUT has or has not been shown by conformance assessment to be non-conforming to the specified protocol specification.

Strike the appropriate words in this sentence. If the PICS for this IUT is consistent with the static conformance requirements (as specified in clause C.3 in this report) and there are no "FAIL" verdicts to be recorded (in clause C.6 in this report) strike the words "has or", otherwise strike the words "or has not".

C.3 Static conformance summary

The PICS for this IUT is or is not consistent with the static conformance requirements in the specified protocol.

Strike the appropriate words in this sentence.

C.4 Dynamic conformance summary

The test campaign did or did not reveal errors in the IUT.

Strike the appropriate words in this sentence. If there are no "FAIL" verdicts to be recorded (in clause C.6 of this report) strike the words "did or" otherwise strike the words "or did not".

Summary of the results of groups of test:

....
....
....
....
....
....
....
....
....

C.5 Static conformance review report

If clause C.3 indicates non-conformance, this clause itemizes the mismatches between the PICS and the static conformance requirements of the specified protocol specification.

....
....
....
....
....
....
....
....
....

C.6 Test campaign report

Table C.4: Test cases

ATS Reference	Selected?	Run?	Verdict	Observations (Reference to any observations made in clause C.7)
TC_ISO7816_A02	Yes/No	Yes/No		
TC_ISO7816_B07	Yes/No	Yes/No		
TC_ISO7816_B08	Yes/No	Yes/No		
TC_ISO7816_C02	Yes/No	Yes/No		
TC_ISO7816_C03	Yes/No	Yes/No		
TC_ISO7816_D02	Yes/No	Yes/No		
TC_ISO7816_D04	Yes/No	Yes/No		
TC_ISO7816_D11	Yes/No	Yes/No		
TC_ISO7816_E06	Yes/No	Yes/No		
TC_ISO7816_E08	Yes/No	Yes/No		
TC_ISO7816_E11	Yes/No	Yes/No		
TC_ISO7816_E12	Yes/No	Yes/No		
TC_ISO7816_E18	Yes/No	Yes/No		
TC_ISO7816_E28	Yes/No	Yes/No		
TC_ISO7816_F05	Yes/No	Yes/No		
TC_LDS_A03	Yes/No	Yes/No		
TC_LDS_A04	Yes/No	Yes/No		
TC_LDS_B11	Yes/No	Yes/No		
TC_LDS_B22	Yes/No	Yes/No		
TC_LDS_B25	Yes/No	Yes/No		
TC_LDS_C03	Yes/No	Yes/No		
TC_LDS_C09	Yes/No	Yes/No		
TC_LDS_C13	Yes/No	Yes/No		
TC_LDS_C19	Yes/No	Yes/No		
TC_LDS_D03	Yes/No	Yes/No		
TC_LDS_D12	Yes/No	Yes/No		
TC_LDS_D15	Yes/No	Yes/No		
TC_LDS_E01	Yes/No	Yes/No		
TC_LDS_E02	Yes/No	Yes/No		
TC_LDS_E09	Yes/No	Yes/No		
TC_LDS_F04	Yes/No	Yes/No		
TC_LDS_F06	Yes/No	Yes/No		
TC_LDS_G01	Yes/No	Yes/No		
TC_LDS_G02	Yes/No	Yes/No		
TC_LDS_H03	Yes/No	Yes/No		
TC_LDS_H07	Yes/No	Yes/No		
TC_LDS_H22	Yes/No	Yes/No		
TC_LDS_H24	Yes/No	Yes/No		
TC_LDS_H31	Yes/No	Yes/No		
TC_LDS_H32	Yes/No	Yes/No		
TC_LDS_H33	Yes/No	Yes/No		
TC_LDS_H37	Yes/No	Yes/No		
TC_LDS_H45	Yes/No	Yes/No		
TC_LDS_H51	Yes/No	Yes/No		
TC_LDS_H54	Yes/No	Yes/No		
TC_LDS_H63	Yes/No	Yes/No		
TC_LDS_H71	Yes/No	Yes/No		
TC_LDS_H75	Yes/No	Yes/No		
TC_LDS_H83	Yes/No	Yes/No		
TC_LDS_I01	Yes/No	Yes/No		

C.7 Observations

Additional information relevant to the technical content of the PCTR is given here.

....
....
....
....
....
....
....
....
....

List of Figures

List of Tables

History

Document history		
V1.1.1	September 2011	Publication