Dictao
152, avenue de Malakoff
75116 PARIS

Security Target
AdSigner 5 signature module

Réf. : dictao_adsigner_TSS_Public
Version 1.2,

TABLE OF CONTENTS

1. ST INTRODUCTION

1.1 ST Reference

This document is the security target.

ST's title: Security Target – AdSigner 5 signature module

Version: 1

Reference: dictao_adsigner_TSS_Public

1.2 TOE Reference

The TOE is:

Author: Dictao

TOE identification: **AdSigner 5**

TOE version: 5.0.0.1

Platform: see 1.4.2.9

Keywords: qualified electronic signature, legally binding signature, electronic signature creation application, signature qualifiée, signature présumée fiable, application de création de signature

1.3 TOE overview

AdSigner 5 is a signature creation application designed for business and technical web-based solutions. The TOE allows the creation of XAdES electronic signatures (version 1.3.2, see **[XAdES]**), using an external signature creation device (which is outside the scope of evaluation). Signing the documents as HTML and plain text, by interpreting and displaying itself the HTML document, AdSigner5 can be used for online financial transaction requests and signing statements and forms (*téléTVA, téléIR* ...).

It is part of a comprehensive system of electronic signature creation, including the calling application (see below) and the signature creation device (SCDev). The later is the only one to access the signer's private key and which can use it for cryptographic operations. It can take several forms, including a smart card, a USB token or a software file. In order to meet the requirements for the creation of a qualified signature, the SCDev must be a Secure Signature Creation Device (SSCD, most likely, a smartcard).

The TOE is executed within a "web page" (named hereafter, the calling application). Its implementation is through the Internet browser (outside the scope of evaluation) displaying the web page. The document to be signed is passed to the TOE by the web page as a parameter. Several other data are also transmitted as parameters (see 1.4.5).

1.4 TOE description

1.4.1 Physical representation

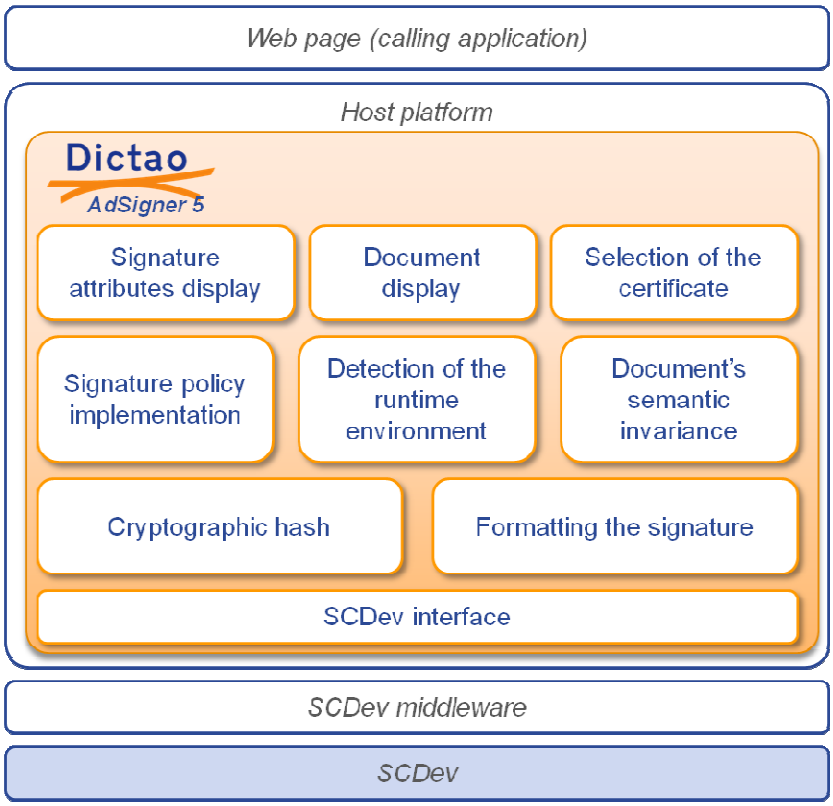
The TOE comes in one ".jar" file (Java).The ".jar" file is to be executed by a Java virtual machine, covering a large panel of web browsers and operating systems (see "1.4.3.1, The host platform", p. 7).

The file is usually hosted on a web server and is downloaded when needed onto the user's computer. For the sake of the evaluation, that file is already installed on the user's computer because the downloading itself is outside the scope of evaluation.

Note that the file is signed (and hence may be authenticated by the signer) and that the hash value of the corresponding binary files is provided in the integration guide for the TOE administrator.

1.4.2 Architecture

The picture below presents the TOE and its internal design. Some external elements are also shown: the detection of the environment by the calling application, the SCDev's middleware, and the SCDev itself.



1.4.2.1 Component enforcing the document's semantic invariance

The TOE only allows signing documents that are either in "raw text" format or in HTML format.

- Text documents are invariant by nature
- HTML documents are considered invariant if they satisfy the criteria given in *Annexe: Supported subset of the HTML*, p. 70.

For details on the semantic invariance of documents, the reader may refer to 1.4.7.

1.4.2.2 Component allowing to display the document

The TOE displays the document to be signed within a specific frame.

- Text document are directly embedded within <pre> tags.
- HTML documents are interpreted according to the HTML standards.

If a document is not displayable, the TOE aborts the signature process.

1.4.2.3 Component allowing to display the signature attributes

This component displays the signature attributes to the signatory.

1.4.2.4 Component for the selection of the certificate

This component retrieves the list of the certificates that are stored and available in the SCDev. That retrieval is done through the cryptographic provider (see 1.4.3.3). Then, that list is filtered according to the signature policy before being presented to the signatory; finally, the signatory selects the certificate to be used for the signature creation.

For each certificate in the filtered list, the following data is displayed:

- The certificate's CN (Common Name)
- The certificate's issuing date
- The certificate's CN of the issuing certificate authority

1.4.2.5 Component to detect the runtime environment

This component is used by the TOE to select which subcomponent is to be used to connect to the SCDev's middleware.

1.4.2.6 Component managing/implementing the signature policy

This component enforces the signature policy specified by the calling applications during the whole signing process. Section 1.4.8 gives details on the data included in a signature policy, together with default values, should they be unspecified by the calling application.

1.4.2.7 Formatting component

This component builds up the XML structure of the XAdES electronic signature.

1.4.2.8 Hashing component

This component computes the hash value of the data to be signed. Supported algorithm is SHA-256, SHA-1 is excluded from this security target (see *FCS_COP.1/Hash function*, p. 44).

Note that the hash of the data to be signed is separately computed by the SCDev and the middleware.

1.4.2.9 Component piloting the interface with the SCDev

This component provides the interface to the SCDev's middleware. It is itself made of several subcomponents, each one providing the support for a particular runtime environment. Globally, this component provides the following functions:

- To obtain from the SCDev the references of the certificates usable by the signatory, or the certificates themselves;
- To indicate to the SCDev the signature key to be activated;
- To transfer the DTBSR to the SCDev;
- For each signed document, to receive from the SCDev the electronic signature as well as the carrying out status indicating the success or the failure of the signature creation process;
- To manage (open or close) sessions with the SCDev.

Note: The term “session” is defined here as *“the period of time during which the private key of the signatory is activated in the SCDev and during which the signatory can generate signatures. A session starts as soon as the signatory correctly authenticates himself to the SCDev (through the TOE) in order to use his pair private key/given certificate. It finishes when the TOE closes it explicitly.”*

1.4.3 Environment of use of the TOE

The electronic signature creation application is integrated on a host platform (a personal computer, a public terminal, a personal organizer…). The elements of the technical environment of the TOE are the following ones:

- the host platform,
- the middleware, that is, the software components installed on the operating system allowing to communicate with the SCDev,
- the SCDev (such as a smartcard, a USB token or a software component installed on the host platform).

1.4.3.1 The host platform

The platform upon which the TOE runs is outside the scope of evaluation. It consists of:

- the computer's hardware
- the computer's operating system
- the web browser
- the Java virtual machine (Java runtime environment)

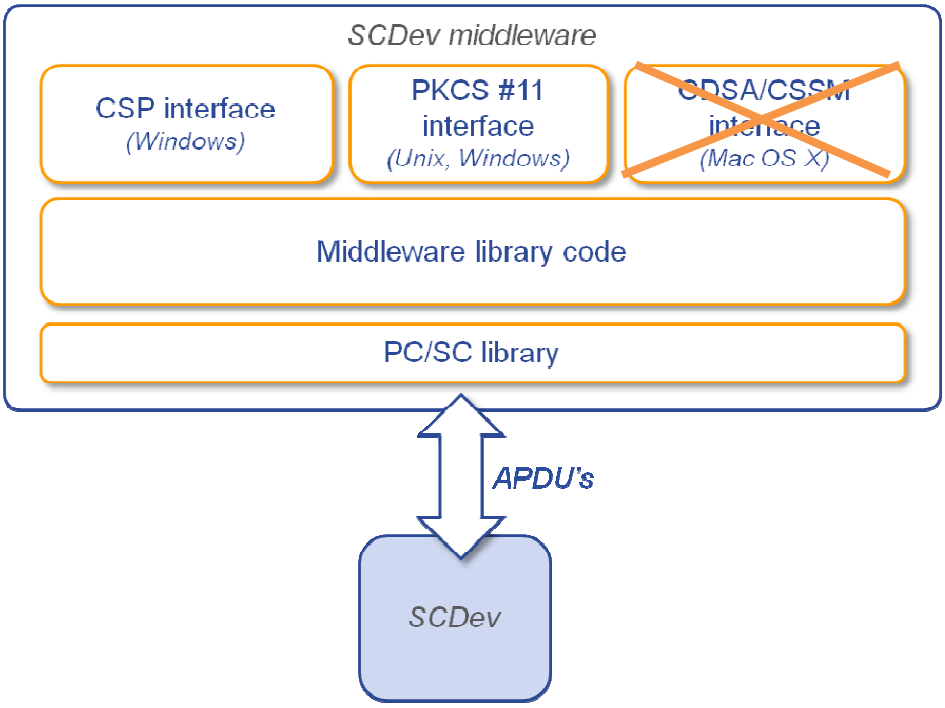
The TOE will be evaluated on the following host platforms:

OS	Windows XP SP 3 Windows Vista SP2+ Windows Seven SP 1 Ubuntu Linux 9.04
JRE	1.5 and 1.6

Browser	Windows Sun JVM	Linux Sun JVM
Internet Explorer 7+	x	
Firefox 3+	x	x
Chrome 12 +	x	

1.4.3.2 The middleware

The following diagram gives an overview of the middleware and the SCDev's relationship:



At the level of the TOE, the middleware presents one of the following interfaces (generally, depending on the OS)

- A PKCS #11 interface
- A *cryptographic service provider* (CSP)
- A *Common Data Security Architecture/Common Security Services Manager* (CDSA/CSSM) API
- A vendor-specific API

All these define a cryptographic interface called by the signature application to create the cryptographic signature and compute the signed data's hash. In some cases, the hash computation may be shared between the middleware and the SCDev: all but the last round of the hash are computed by the middleware's code, and the last round is computed by the SCDev. That process is inconspicuous to the calling application.

1.4.3.3 The SCDev

The signature creation device could be a smartcard, a USB token or a software component. However, in order to meet the requirements for the creation of a qualified signature in the sense of the European Directive 1999/93/EC on a Community framework for electronic signatures, and for that of a “deemed reliable” one (“signature présumée fiable”) in the sense of the French decree on electronic signatures (2001-272), the SCDev must be a SSCD (hence, most likely, a smartcard).

In the same vein, the TOE usually computes the signed data’s hash itself but, for the creation of a qualified signature, that hash is at least partially computed by the SSCD (such as the last hashing round).

1.4.4 Product’s life-cycle

The TOE's life-cycle is as follows:

Step	Description	Actor
Specification	TOE specification	Dictao
Development	TOE development	Dictao
Test	Testing and product validation	Dictao
Delivery	TOE delivery, with its documentation	Implementor of the calling application
Integration	TOE web integration	Implementor of the calling application
Installation	TOE's files are copied onto the web server	Implementor of the calling application
Download	The TOE is downloaded onto the user's computer	Signer
Use	The TOE is runt to produce an digital signature	Signer

1.4.5 TOE's runtime behaviour

The TOE's overall process is as follows:

1. The TOE is invoked by the calling application (web page) with the following parameters:
   a. The document to be signed, base64-encoded
   b. The type of the data, that is, raw text or HTML
   c. Resource file for the application
   d. Language of the graphical user interface (GUI)
   e. URL for the JavaScript module that will receive the TOE’s errors
   f. Name of the JavaScript function that will receive the TOE’s errors
   g. The XAdES version of the signature: 1.3.2
   h. The hashing algorithm to use
   i. The reference to the signature policy. The signature policy specifies many parameters, such as the kind of authorised certificates (cf. 1.4.8 for details)

2. If the type of the data is HTML, the TOE checks the HTML's semantic invariance. If the data are deemed unstable, the signature process is aborted.
3. The TOE displays:
   a. The document to be signed (in text or HTML format)
   b. The available signing certificates, that is, the certificates stored in the SCDev that fulfil the signature policy's criteria.
   c. The type of engagement, if available in the signature policy.
4. At this point, the user may also check the signature attributes by asking the TOE to display them.
5. The user selects his certificate and gives explicitly his approval for the signature by clicking on a checkbox and the "OK button" (that button is inactive unless the checkbox has been checked).
   a. The TOE builds up the XAdES structure of the signature.
   b. The SCDEV and the middleware compute the DTBS' hash.
   c. The TOE sends the data to signed and the signing key's id to the SCDev (signature request)

   Note: Depending on the case, the TOE may directly send the DTBS to the SCDev (through the middleware, of course). In that case, the hash's computation may be split between the SCDev and the middleware or be entirely done by the SCDev.
6. The TOE receives the signature value from the SCDev (or a failure notice, in which case the signature process is aborted). The TOE checks the data received from the SCDev to ensure that, on one hand, it is a genuine digital signature (and not random data), on the other hand, it corresponds to the DTBS it has previously sent.
7. The TOE includes the signature value in the XAdES structure and returns it to the calling application.

1.4.6 What You See Is What You Sign (WYSIWYS)

Following **[PP-ACSE]**, **AdSigner 5** ensures a faithful representation of the document to be signed through the combination of three mechanisms:

- Allowing the signer to see the document to be signed, either in its raw format
- Checking the semantic invariance of the document to be signed, so that its representation will be the same on all W3C-conformant web browsers
- Allowing the signer to see the signature attributes

AdSigner 5 implements these functions itself.

1.4.7 Semantic invariance and presentation of the document

The document to be signed may contain dynamic fields or executable code (script) whose value may depend on external, contextual data. In the following, we will call this notion "semantic invariance" or "semantic stability".

The TOE exclusively signs documents that are in "raw text" format, which are, by definition, semantically stable, or documents in HTML format.

In order to ensure that the HTML is semantically stable and displayable, the TOE only accepts a subset of the HTML language (authorised mark-ups). That subset is defined in the *Annexe: Supported subset of the HTML*, p. 70 of this document.

Moreover, as the TOE displays itself the document to be signed to the signer, and because it only supports that same subset of the HTML, the TOE may only display stable documents. For this reason, the signature policy does not contain an option allowing signing an unstable document.

1.4.8 Signature policy

The signature policy contains the following data:

- The list of authorised CA's (optional parameter): the user may only select a certificate issued by one of these CA's.
- The certificate's public key (optional parameter): the user may only select a certificate with the given public key. Several public keys may be given as a list for that parameter.
- The OID and the hash of the document defining the signature policy.
- The filtering date (optional parameter): the user may only select a certificate whose validity period includes the given date.
- The engagement type (optional parameter): if specified, the engagement type is displayed to the signer for approval, and is included in the DTBS.
- The signer's role (optional parameter): if specified, the signer's role is included in the DTBS.
- The signing location (optional parameter): if specified, the signing location is included in the DTBS.
- A Boolean flag indicating whether the user may or may not select a certificate without the non-repudiation attribute. A true value requires the non-repudiation attribute.
- A Boolean flag indicating whether the user may or may not select a certificate without the digital signature attribute. A true value requires the digital signature attribute; contrary to the other flags, its default value is “true”.
- A Boolean flag indicating whether the user may or may not select a certificate without the S/MIME encryption attribute. A true value requires the S/MIME encryption attribute.
- A Boolean flag indicating whether the user may or may not select a certificate without the encipherment attribute. A true value requires the encipherment signature attribute.
- A Boolean flag indicating whether the policy requires a qualified certificate **[EXT_TS_101_862]**. This information is available through a `QCStatement`. A true value requires the corresponding `QCStatement`.
- A Boolean flag indicating whether the policy requires the private key corresponding to the public key to be protected by a SCDev. This information is available through a `QCStatement` **[EXT_TS_101_862]**. A true value requires the corresponding `QCStatement`.
- Image for the "OK" button
- Image for the "Cancel" button
- Image for the "Update" button (to update the certificates’ list)
- The GUI’s background colour
- The GUI’s foreground colour
- The GUI’s font

The default values for these data are:

- any CA is authorised
- no restriction on the certificate's public key
- the current date is used for the filtering date
- no engagement type
- no signer's role
- no signing location
- all Boolean flags are "false", save the “digital signature” one, which is “true” by default.

The default values for the GUI parameters (buttons, colours and fonts) come from the standard software libraries (such as "OK" labels, black on white, plain font...).

1.4.9 Signature attributes

The following signature attributes are displayed to the signer by the TOE upon request:

- The reference to the signature policy. If that reference is an URL, a hyperlink is displayed.
- The engagement type, if present
- The signer's role, if present
- A summary of the selected certificate (its DN, serial number, the issuing CA's DN, its validity period)

1.4.10 The XAdES format

The TOE only produces enveloping signatures.

The XAdES (*XML Advanced Electronics Signature*) specification extends the XML-DSig specification on the representation of electronic signatures in XML format. It also addresses issues such as long-term conservation of electronic signatures and qualified electronic signatures in the sense of the European directive 1999/93/EC.

2. CONFORMANCE CLAIMS

2.1 CC conformance claim

This security target claims a strict conformance with the *Common Criteria* version 3.1, third revision.

It was written in accordance with **[CC3.1]**.

2.2 PP claim

This security target claims a *demonstrable conformance* with **[PP-ACSE]**, according to the definition in **[CC3.1]**, Part 1.

In normative sections, removals from the protection profile are typeset, while added text is typeset like that.

In brief, AdSigner 5 differs from the TOE in the PP on the following points:

- AdSigner 5 displays itself the document to be signed (the PP relies on the environment)
- AdSigner 5 only signs a single document (the PP allows many documents to be signed at once)
- AdSigner 5 manages a given number of documents' formats (the PP considers the capability to configure this)
- AdSigner 5 checks itself the documents' stability (the PP relies on the environment) and forbids the user to sign an unstable document (the PP allows it)
- AdSigner 5 does not manage countersignatures

2.3 Package claim

This security target claims conformance with the assurance package defined by the standard qualification process **[QUA_STD]**: EAL3+ augmented with ALC_FLR.3 and AVA_VAN.3.

2.4 Conformance Rationale

The demonstrable conformance with respect to **[PP-ACSE]** is required by the protection profile itself, section 3.4.

3. SECURITY PROBLEM DEFINITION

3.1 Threats

"This section describes the threats to be countered by the TOE. Because all the security objectives are justified by assumptions and OSPs, the definition of the threats is not necessary. In this case, the section is not applicable and is therefore considered as fulfilled." **[PP-ACSE]**

3.2 Assets

This section describes the assets to be protected by the TOE. All the assets must be integrity-protected.

3.2.1 User data

D.Signatory's_Document
The signatory's document (SD) during the invocation of the signature process could contain:a single electronic document, The SD is made of the raw data transmitted to the TOE by the calling application as a call parameter. Protection: integrity, confidentiality

The three following assets correspond to successive representations of the data to be signed.

D.Data_To_Be_Signed
The Data To Be Signed (DTBS) are information for which electronic signature is needed. The DTBS are transmitted to the TOE by the calling application as a call parameter. They include: ■ the document to be signed (as raw data, *D.Signatory's_Document*) ■ the signature attributes explicitly selected by the signatory or implicitly selected by the application. The signature attributes contain: ■ the signatory's certificate ■ a non-ambiguous reference of this certificate (its serial number, the issuing CA's DN, and its hash value) ■ the reference to the signature policy, ■ the type of commitment (if specified) ■ the presumed place of the signature (if specified) ■ the signer's role (if specified) ■ the presumed date/hour of the signature, ■ the format of the contents Protection: integrity, confidentiality

D.DTBS_Formatted
These data correspond to a XAdES formatting of the *D.Data_To_Be_Signed* (envelope). Protection: integrity, confidentiality

D.DTBS_Digest
That data is the hash value of the formatted DTBS *(D_DTBS_Formatted)*. Protection: integrity

D.DTBS_Representation
This asset corresponds to the hash value of the data to be signed *(D.DTBS_Digest)* after having undergone a formatting operation, before its sending to the SCDev. The formatting operation consists in adding the OID of the hashing algorithm used in the electronic signature. Protection: integrity

Note: in the case where the hash is computed by the SCDev (and, eventually, partially by the middleware), that asset is entirely managed by these components.

3.2.2 Data returned by the TOE

D.Electronic_Signature
The Electronic Signature is an envelope containing: ■ the DTBS hash, ■ the signature; ■ additional data facilitating the verification of the signature (as required by the XAdES specification): the signed data, the canonicalization method, the signature algorithm and the signing certificate. This asset must be to protected by the TOE during its generation before its transmission to the signatory. Protection: integrity

3.2.3 TOE sensitive assets (TSF data)

This section defines the assets of the TOE involved in the TOE operations.

D.Signature_Policy
The TOE performs the signature operations according to a signature policy containing the following data: ■ the list of authorised CA's (optional parameter) ■ the certificate's public key (optional parameter) ■ the OID and the hash of the document defining the signature policy. ■ the filtering date (optional parameter) ■ the engagement type (optional parameter) ■ the signer's role (optional parameter) ■ the signing location (optional parameter) ■ whether the user may or may not select a certificate without the non-repudiation attribute. ■ whether the user may or may not select a certificate without the digital signature attribute. ■ whether the user may or may not select a certificate without the S/MIME encryption attribute. ■ whether the user may or may not select a certificate without the encipherment attribute. ■ whether the policy requires a qualified certificate. ■ whether the policy requires the private key corresponding to the public key to be protected by an SCDev. ■ Image for the "OK" button ■ Image for the "Cancel" button

- Image for the "Update" button (to update the certificates' list)
- The GUI's background colour
- The GUI's foreground colour
- The GUI's font

Protection: integrity

D.Services

This asset represents the executable code implementing the services provided by the TOE.

Protection: integrity

D.Data_Representations_Association

The data within the TOE often have a representation different from those presented to the signatory or input to the TOE.

Protection: integrity

D.DocFormat_Application_Association

This asset is a parameter managed by the TOE which allows it to decide how to display the document according to the format of the document having to be presented to the signatory.

Protection: integrity

Application note

The TOE displays itself the document to be signed, according to the call parameters specifying the document's format.

3.3 Subjects and roles

S.Signatory

The signatory interacts with the TOE to sign document according to the signature policy specified by the calling application.

The security administrator role is split in the three separate roles hereafter.

S.Calling Application

The calling application manages the TOE through the call parameters it gives it at runtime. The calling application provides:

- the document to be signed
- the signature policy to be used by the TOE

S.Calling Application Administrator

The calling application administrator is the entity responsible for the list of the signature policies available to the TOE.

This role must not be confused with the security administrator, as defined in the protection profile. The present administrator manages the calling application and not the TOE itself.

This role may be inapplicable if the calling application is not configurable. In this case, the calling application administrator is the same as the calling application developer.

Application note

This role is distinct from the administrator of the host computer (the one onto which the TOE runs).

S.Calling Application Developer
The calling application developer has two duties: ■ He develops the calling application ■ He manages it as the *S.Calling Application Administrator* if the calling application is not configurable.

3.4 Organisational security policies

This section defines the rules applicable to the TOE.

3.4.1 Policies related to the validity of the created signature

P.Signatory_Certificate_Conformity
To avoid the creation of invalid signatures, the TOE must control that the certificate selected by the signatory is in compliant with the signature policy to be applied.

P.Signatory_Certificate_Validity
To avoid the creation of invalid signatures, the TOE must control that the certificate selected by the signatory is used during its validity period.

P.Signature_Attributes_Conformity
To avoid the creation of invalid signatures, the TOE must control: ■ that the signature attributes selected by the signatory are in compliant with the signature policy to be applied, and ■ that all the signature attributes required by the signature policy are present.

3.4.2 Control of the invariance of the document's semantics

P.Document_Stability_Control
The TOE must check whether the document to be signed is semantically stable or unstable. The TOE must inform the signatory if the document's semantics can not be considered as being invariant.

By design, the TOE cannot sign an unstable document (see 1.4.7, p. 10).

3.4.3 Presentation to the signatory of the document and of the signature attributes

P.Document_Presentation
The TOE must allow the signatory to view a reliable representation of the document to be signed. The TOE must not allow the signature of a document if it cannot be viewed by the signatory.

P.Signature_Attributes_Presentation
The TOE must allow the signatory to view the signature attributes.

3.4.4 Compliance with standards

P.Hash_Algorithms
The hash algorithm(s) implemented in the TOE must not make it possible to create two documents producing the same hash. The algorithms must conform to the cryptography requirements **[CRYPT]**.

3.4.5 Interaction with the signatory

P.Multiple_Documents_Signature
The TOE must allow to sign a single document.

Note: The TOE may only sign a single document.

P.Signature_Process_Interruption
The signatory must be able to interrupt the process of signature before the activation of the signature key.

P.Explicit_Agreement
The TOE must compel the signatory to perform a succession of non-trivial operations to check the agreement of the signatory before executing the process of signature.

3.4.6 Miscellaneous

P.Certificate/Private_Key_Association
The TOE must transfer the necessary information to the SCDev so that it can activate the private key corresponding to the selected certificate.

P.Electronic_Signature_Export
At the end of the process of signature, the TOE must transmit to the signatory the XAdES Electronic signature of the document comprising at least: ■ the signature of the document; ■ the hash of all the data to be signed; ■ a reference of the certificate (or the actual certificate) of the signatory; ■ a reference of the applied signature policy.

Application note
Other information facilitating the verification of the signature can be added (e.g. the certificate of the signatory, time-stamping tokens, etc).

P.Administration
The TOE must allow the calling application (*S.Calling Application*) to manage (to add/remove) the signature policies [*D.Signature_Policy*] and the table of association between the viewer applications and the document formats input to the TOE [*D.DocFormat_Application_Association*].

Note: although this is the calling application administrator (*S.Calling Application Administrator*) who configures it, and thus, these parameters, this is actually the calling application (*S.Calling Application*) which transmits them to the TOE.

P.SCDev Response Validation
The TOE must ensure that the data returned by the SCDev is indeed a **[PKCS #1]** digital signature.

3.5 Assumptions

This section describes the assumptions on the operational environment of the TOE.

3.5.1.1 Assumptions on the host platform

A.Host_Platform
It is supposed that the host platform on which the TOE is installed is either directly under the responsibility of the signatory or under the control of the organization to which the signatory belongs or of which he is the customer. The operating system of the host platform is supposed to provide separate execution contexts for the various processes executed. In addition, it is presumed that following security measures are implemented: ■ the host platform is protected from the viruses; ■ the data exchange between the host platform and other IT elements via an open network are controlled by a firewall; ■ the access to the administration functions of the host platform is restricted to the administrators of the platform (thereafter the “Host administrator”). The user account is different from the host administrator account; ■ the installation and the update of the software of the host platform is under the control of the host administrator; ■ the operating system of the host platform does not allow the execution of untrusted applications. ■ the host platform is synchronized with a reliable time source

Application note
The role of Host administrator mentioned above is distinct from the role of Security administrator of the calling application which has particular prerogatives such as management of TOE’s and calling application’s sensitive assets their and configuration parameters.

3.5.1.2 Assumptions on the SCDev

The following assumptions are related to the signature creation device itself and to the possible different interactions of the TOE environment with it.

A.SCDev
It is presumed that the SCDev has the capability to generate a digital signature from the data communicated by the TOE. It is presumed that the SCDev performs the authentication of the signatory allowing him or not to use the private key corresponding to the selected certificate. The SCDev is responsible for the protection of the signatory’s data. The following data are presumed to be stored and used in a secure manner by the SCDev: Assets related to the generation of the signature: ■ the private key(s) of the signatory, protected in confidentiality and integrity; ■ the signatory's certificate(s) protected in integrity or, by default, a non ambiguous reference of the signatory's certificate(s); ■ the private key/certificate association, protected in integrity Assets related to the authentication of the signatory:

- the authentication data of the signatory, protected in integrity and confidentiality;
- the association between authentication data and the private key/certificate pair, protected in integrity (1)

(1) the association can concern an authentication data and private key/certificate pair. Thus, several pairs can be stored in the same SCDev. Their access could be protected by different authentication data.

A.TOE/SCDev_Communications

It is presumed that the software and/or hardware components providing the interface between the TOE and the SCDev is able to manage (to open/close) a secure channel guaranteeing the integrity and the exclusiveness of the communication.

It is presumed that this middleware is able to compute all but the last round of the hash, leaving the last round's computation to the SCDev.

Application note

The components implementing the communication between the TOE and the SCDev can contain various software and/or hardware components installed on the operating system (e.g. PKCS#11 drivers or cryptographic service providers (CSP) defining a cryptographic interface called by the signature application to access a module generating the signature).

A.Signatory_Authentication_Data_Protection

It is presumed that the software and hardware components allowing the signatory to authenticate himself to the SCDev in order to activate the private key corresponding to the selected certificate, guarantee the confidentiality and the integrity of the authentication data during the data entry and during the transfer of these data towards the SCDev.

3.5.1.1 Assumptions on document presentation

As the TOE is itself the viewer application, the following assumption is inapplicable.

~~**A.Document_Presentation**~~

~~It is presumed that the system of signature creation on which the TOE is installed has one or several viewer applications which:~~

~~either accurately display the document to be signed,~~

~~either warn the signatory of possible problems of incompatibilities between the viewer application and the characteristics of the document.~~

The following assumption has been deleted because TOE does not allow counter-signing.

~~**A.Previous_Signatures_Presentation**~~

~~In the case of a countersignature, it is supposed that the signatory has a means of knowing at least the identity of previous signatory(s) and at best of verifying these signatures.~~

3.5.1.2 Assumption on the control of invariance of the document's semantics

As the TOE performs itself the semantic invariance check, the following assumption is inapplicable.

~~A.Document_Stability_Control~~
~~It is presumed that the environment of the TOE provides a module able to determine if the document's semantics to be signed is invariant and to communicate the status of this control to the TOE.~~

3.5.1.3 Assumptions on the context of operations

A.Signatory_Presence
To avoid the modification of the list of the documents to be signed without his knowledge, the signatory is supposed to remain present between the moments when he wishes to sign the documents and when he enters his authentication data to activate the key of signature.

According to the new definition of the roles (p.16), the following assumption is reformulated into *A.Trusted Calling Application* and *A.Trusted Calling Application Developer and Administrator*.

~~A.Trusted_Security_Administrator~~
~~The Security administrator of the TOE is presumed to be trusted, to be trained for the use of the TOE and to have the means necessary to the execution of his tasks.~~

A.Trusted Calling Application
The calling application (*S.Calling Application*) is presumed to be trusted.

A.Trusted Calling Application Developer and Administrator
The calling application developer (*S.Calling Application Developer*) and its administrator (*S.Calling Application Administrator*) are presumed to be trusted, trained for the use of the TOE and to have the means necessary to the execution of their tasks.

A.Services_Integrity
The environment of the TOE is presumed to provide to the calling application (*S.Calling Application*) and its administrator (*S.Calling Application Administrator*) means of controlling the integrity of the services and of the parameters of the TOE.

A.Signature_Policy_Origin
The origin of the signature policies usable by the TOE is supposed to be authentic.

A.Web Communications
The communication between the host computer upon which the TOE runs and the web server from which are downloaded the calling application and the TOE ensures the integrity of the TOE's call parameters.

A.Web Server
The web server hosting the TOE and the calling application (web page) is presumed to guarantee the integrity of the TOE and that of the calling application.

The following security measures are presumed to be implemented:

- the host platform must be protected from the viruses;
- the data exchange between the host platform and other IT elements via an open network must be controlled by a firewall;
- the access to the administration functions of the host platform must be restricted to the administrators of the platform (thereafter the “Host administrator”). The user account must be different from the Host administrator account;
- the installation and the update of the software of the host platform must be under the control of the Host administrator;
- the operating system of the host platform must not allow the execution of untrusted applications.

4. SECURITY OBJECTIVES

4.1 Security objectives for the TOE

4.1.1 General objectives

O.Certificate/Private_Key_Association
The TOE shall transfer the necessary information to the SCDev so that it can activate the private key corresponding to the selected certificate.

4.1.2 Interactions with the signatory

O.Signature_Attributes_Presentation
The TOE shall present to the signatory an exact representation of the attributes that will be signed.

O.Explicit_Agreement
The TOE shall provide to the signatory the means of explicitly expressing (i.e., in a voluntary and non-ambiguous way) its agreement to select document(s) and to start the process of signature of the selected documents.

O.Signature_Process_Interruption
The TOE shall provide to the signatory the means to cancel the process of signature before the activation of the signature key.

O.Documents_To_Be_Signed
After the signatory's agreement for signature, the TOE shall guarantee that the actually processed document correspond exactly to the document selected to be signed. The signature attributes used in the signature must correspond to those specified by the signer.

4.1.3 Signature policy applications

O.Signatory_Certificate_Conformity
The TOE shall check that the certificate selected by the signatory is compliant with the signature policy to be applied.

O.Signatory_Certificate_Validity
The TOE shall control that the certificate selected by the signatory is used during its validity period.

Application note
The time reference used for this purpose is the time provided by the operating system of the host platform or, when provided, by the calling application.

O.Signature_Attributes_Conformity
The TOE shall control the presence and the compliance of the signature attributes selected by the signatory with the signature policy to be applied.

O.Electronic_Signature_Exportou
At the end of the process of signature, the TOE shall transmit to the signatory the XAdES Electronic signature of the document containing at least: ■ the signature of the document; ■ the hash of all the data to be signed; ■ a reference of the certificate (or the actual certificate) of the signatory; ■ a reference of the applied signature policy.

Application note
Other information facilitating the verification of the signature can be added (e.g. the certificate of the signatory, time-stamping tokens, etc).

4.1.4 Data protection

O.Administration
The TOE shall allow the calling application (*S.Calling Application*) to manage (to add/remove) the signature policies [*D.Signature_Policy*] and the table of association between the viewer applications and the document formats input to the TOE [*D.DocFormat_Application_Association*].

4.1.5 Cryptographic operations

O.Cryptographic_Operations
The TOE shall implement cryptographic algorithms having the following properties: ■ Hash algorithms must not allow to create two documents producing the same hash ■ Hash algorithms must conform to the ANSSI cryptography requirements **[CRYPT]**.

Supported algorithm is SHA-256, SHA-1 is excluded from this security target (see *FCS_COP.1/Hash function*, p. 44).

4.1.6 Control of the invariance of the document semantics

O.Document_Stability_Control
The TOE must check whether the document to be signed is semantically stable or unstable. The TOE shall inform the signatory if this module determines that the document's semantics are unstable.

By design, the TOE cannot sign an unstable document (see 1.4.7, p. 10).

4.1.7 Presentation of the documents to be signed

O.Document Presentation
The TOE must allow the signatory to view a reliable representation of the document to be signed. The TOE must not allow the signature of a document if it cannot be viewed by the signatory. In order to faithfully display the document to be signed, the TOE shall manage the correspondence between the document formats it supports and the means to display them to the signer.

The TOE must allow the signatory to view the signature attributes. The TOE does not support countersignatures.

O.SCDev Response Validation
The TOE must ensure that the data returned by the SCDev is indeed a [PKCS #1] digital signature.

4.2 Security objectives for the operational environment

4.2.1 Security objectives for the host platform

OE.Host_Platform
The host platform on which the TOE is installed shall be either directly under the responsibility of the signatory or under the control of the organization to which the signatory belongs or of which he is the customer. The operating system of the host platform shall provide contexts of execution separated for the various tasks which it carries out. The following security measures shall be implemented: ■ the host platform must be protected from the viruses; ■ the data exchange between the host platform and other IT elements via an open network must be controlled by a firewall; ■ the access to the administration functions of the host platform must be restricted to the administrators of the platform (thereafter the “Host administrator”). The user account must be different from the Host administrator account; ■ the installation and the update of the software of the host platform must be under the control of the Host administrator; ■ the operating system of the host platform must not allow the execution of untrusted applications; ■ the host platform is synchronized with a reliable time source.

Application note
The role of Host administrator mentioned above is distinct from the role of Security administrator of the TOE which has particular prerogatives such as management of TOE sensitive assets and configuration parameters.

4.2.2 Security objectives for the SCDev and its environment

The following security objectives are related to the signature creation device itself or the components of its environment allowing the interactions of the signatory or the TOE with it.

OE.SCDev
The SCDev shall have at least the capability to generate a signature of the data transmitted by the TOE. Moreover, the SCDev shall perform the authentication of the signatory allowing him to use the private key corresponding to the selected certificate. The SCDev is responsible for the protection of the signatory data. The following data shall be stored and used in a secure manner by the SCDev: Assets related to the generation of the signature: ■ the private key(s) of the signatory, protected in confidentiality and integrity; ■ the actual certificate(s) protected in integrity or, by default, a reference to the certificate(s) of the signatory; ■ the private key/certificate association, protected in integrity Assets related to the authentication of the signatory:

- the authentication data of the signatory, protected in integrity and confidentiality;
- the association between authentication data and the private key/certificate pair, protected in integrity

OE.TOE/SCDev_Communications

The software and/or hardware components providing the interface between the TOE and the SCDev shall be able to manage (to open/close) a secure channel guaranteeing the integrity and the exclusiveness of the communication.

This middleware must be able to compute all but the last round of the hash, leaving the last round's computation to the SCDev.

OE.Signatory_Authentication_Data_Protection

The software/hardware components allowing the signatory to authenticate himself to the SCDev in order to activate the private key corresponding to the selected certificate, shall guarantee the confidentiality and the integrity of the authentication data of during the data input and during the transfer of these data towards the SCDev.

4.2.3 Presence of the signatory

OE.Signatory_Presence

The signatory shall remain present between the moments when he agrees to sign the documents and when he enters his authentication data to activate the key of signature.

Application note

If for any reason the signatory cannot remain present, he must start again the process from the beginning: selection of the documents to be signed, selection of the attributes, etc

4.2.4 Document presentation

In this ST, this objective has been replaced by the new objective *O.Document Presentation*.

~~**OE.Document_Presentation**~~

~~The host platform on which the TOE is installed shall have viewer applications which:~~

~~either accurately display the document to be signed,~~

~~either warn the signatory of possible problems of incompatibilities between the viewer application and the characteristics of the document.~~

~~In case the document to be signed already contains signatures, the environment of the TOE allows the signatory at least to know the identity of previous signatories, at best to verify the validity of these signatures.~~

4.2.5 Miscellaneous

In this ST, the following objective has been replaced by the updated objective for the TOE *O.Document_Stability_Control*.

~~**OE.Document_Stability_Control**~~

~~The environment of the TOE shall provide a module able to determine if the semantics of the document to be signed is invariant and to communicate the status of this analysis to the TOE.~~

OE.Signature_Policy_Origin
The administrator of the calling application (*S.Calling Application Administrator*) shall verify the authenticity of the origin of the signature policies before the TOE uses them.

OE.Trusted Calling Application
The calling application (*S.Calling Application*) is to be trusted.

OE.Trusted Calling Application Developer and Administrator
The calling application developer (*S.Calling Application Developer*) and its administrator (*S.Calling Application Administrator*) are to be trusted, trained for the use of the TOE and have the means necessary to the execution of their tasks.

OE.Services_Integrity
The environment of the TOE shall provide to the calling application (*S.Calling Application*) and its administrator (*S.Calling Application Administrator*) the means of controlling the integrity of the services and of the parameters of the TOE.

OE.Web Communications
The communication between the host computer upon which the TOE runs and the web server from which are downloaded the calling application and the TOE must ensure the integrity of the TOE's call parameters.

OE.Web Server
The web server hosting the TOE and the calling application (web page) must guarantee the integrity of the TOE and that of the calling application. The following security measures are implemented: ■ the host platform must be protected from the viruses; ■ the data exchange between the host platform and other IT elements via an open network must be controlled by a firewall; ■ the access to the administration functions of the host platform must be restricted to the administrators of the platform (thereafter the “Host administrator”). The user account must be different from the Host administrator account; ■ the installation and the update of the software of the host platform must be under the control of the Host administrator; ■ the operating system of the host platform must not allow the execution of untrusted applications.

4.3 Security objectives rationale

4.3.1 Security problems coverage by the objectives

This section demonstrates that each threat, OSP and assumption has at least one security objective tracing to it. It also provides a rationale for the coverage.

4.3.1.1 Threats

The present ST defines no threat.

4.3.1.2 Assumptions

A.Host_Platform

This assumption is covered completely by the *OE.Host_Platform* objective which reuses all its elements.

A.SCDev

This assumption is covered completely by the *OE.SCDev* objective which reuses all its elements.

A.Services_Integrity

This assumption is covered entirely by the security objective *OE.Services_Integrity* which uses the same terms.

A.Signatory_Authentication_Data_Protection

This assumption is covered entirely by the *OE.Signatory_Authentication_Data_Protection* objective which reuses all its elements.

A.Signatory_Presence

This assumption is completely covered by the security objective *OE.Signatory_Presence* which reuses its elements.

A.Signature_Policy_Origin

This (updated) assumption is covered by the (updated) security objective *OE.Signature_Policy_Origin* requiring the administrator of the calling application (*S.Calling Application Administrator*) to make sure of the authenticity of the origin of the signature policies usable by the TOE.

A.TOE/SCDev_Communications

This assumption is covered entirely by the *OE.TOE/SCDev_Communications* objective which reuses all its elements.

A.Trusted Calling Application

This assumption is covered entirely by the *OE.Trusted Calling Application* objective which reuses all its elements.

A.Trusted Calling Application Developer and Administrator

This assumption is covered entirely by the *OE.Trusted Calling Application Developer and Administrator* objective which reuses all its elements.

A.Web Communications

This assumption is covered entirely by the *OE.Web Communications* objective which reuses all its elements.

A.Web Server

This assumption is covered entirely by the *OE.Web Server* objective which reuses all its elements.

4.3.1.3 Organisational security policies

P.Administration

The organisational security policy is covered on the one hand by the *O.Administration* objective which uses the same terms and on the other hand by the security objective on the environment . OE.Trusted Calling Application Developer and Administrator which ensures that the calling application administrator (*S.Calling Application Administrator*) of the TOE is not a threatening agent.

P.Certificate/Private_Key_Association

The organisational security policy *P.Certificate/Private_Key_Association* is completely covered by the security objective *O.Certificate/Private_Key_Association*, which uses the same terms.

P.Document_Presentation

The organisational security policy *P.Document_Presentation* is covered by the (new) security objective O.Document Presentation which uses the same terms.

Note: In **[PP-ACSE]**, this organisational security policy is covered by the defunct O.Viewer_Application_Execution objective.

P.Document_Stability_Control

The organisational security policy *P.Document_Stability_Control* is covered by the security objective for the TOE *O.Document_Stability_Control* which requires the TOE to check the semantic invariance of the document.

P.Electronic_Signature_Export

The organisational security policy is covered entirely by the *O.Electronic_Signature_Export* objective, which uses the same terms.

P.Explicit_Agreement

This organisational security policy is covered by the *O.Explicit_Agreement* objective. This objective requires the signatory to express without ambiguity his agreement to sign.

P.Hash_Algorithms

The organisational security policy is covered entirely by the *O.Cryptographic_Operations* objective, which uses the same terms.

P.Multiple_Documents_Signature

This policy is covered by the *O.Documents_To_Be_Signed* objective which requires that the TOE guarantees that the signed documents are those selected by the signatory (no addition, no suppression, no substitution of documents in the list).

P.SCDev Response Validation

This organisational security policy is completely covered by the security objective *O.SCDev Response Validation*, which requires the TOE to check that the data returned by the SCDev is indeed a PKCS #1 electronic signature.

P.Signatory_Certificate_Conformity

This policy is covered by the *O.Signatory_Certificate_Conformity* objective which requires that the TOE controls the compliance of the certificate selected by the signatory with respect to the requirements of the signature policy.

P.Signatory_Certificate_Validity

This policy is covered by the *O.Signatory_Certificate_Validity* objective which requires that the TOE controls that the certificate selected by the signatory is valid.

P.Signature_Attributes_Conformity

This policy is covered by the *O.Signature_Attributes_Conformity* objective by requiring that the TOE controls the presence and the compliance of all the signature attributes required by the signature policy.

P.Signature_Attributes_Presentation

This policy is first covered by the *O.Signature_Attributes_Presentation* objective which requires that the TOE offers to the signatory a representation of the signature attributes compliant with those which will be signed.

This coverage is completed by the (new) security objective *O.Document Presentation*, which also requires the TOE to allow the signatory to view the signature attributes.

P.Signature_Process_Interruption

This policy is covered by the *O.Signature_Process_Interruption* objective by requiring that the TOE provides the means of canceling the process of signature at any moment before the activation of the signature private key.

4.3.2 Reverse coverage

The following tables map each objective to the security problem elements it has a relationship with. This mapping directly comes from the previous section and shows that there is no spurious objective.

OE.Host_Platform	*A.Host_Platform*
OE.SCDev	*A.SCDev*
OE.Services_Integrity	*A.Services_Integrity*
OE.Signatory_Authentication_Data_Protection	*A.Signatory_Authentication_Data_Protection*
OE.Signatory_Presence	*A.Signatory_Presence*
OE.Signature_Policy_Origin	*A.Signature_Policy_Origin*
OE.TOE/SCDev_Communications	*A.TOE/SCDev_Communications*
OE.Trusted Calling Application	*A.Trusted Calling Application*, *P.Document_Stability_Control*
OE.Trusted Calling Application Developer and Administrator	*P.Administration*, *P.Document_Stability_Control*
OE.Trusted Calling Application Developer and Administrator	*A.Trusted Calling Application Developer and Administrator*
OE.Web Communications	*A.Web Communications*
OE.Web Server	*A.Web Server*

O.Administration	*P.Administration*
O.Certificate/Private_Key_Association	*P.Certificate/Private_Key_Association*
O.Cryptographic_Operations	*P.Hash_Algorithms*
O.Document Presentation	*P.Document_Presentation*,

	P.Signature_Attributes_Presentation
O.Document_Stability_Control	*P.Document_Stability_Control*
O.Documents_To_Be_Signed	*P.Multiple_Documents_Signature*
O.Electronic_Signature_Export	*P.Electronic_Signature_Export*
O.Explicit_Agreement	*P.Explicit_Agreement*
O.SCDev Response Validation	*P.SCDev Response Validation*
O.Signatory_Certificate_Conformity	*P.Signatory_Certificate_Conformity*
O.Signatory_Certificate_Validity	*P.Signatory_Certificate_Validity*
O.Signature_Attributes_Conformity	*P.Signature_Attributes_Conformity*
O.Signature_Attributes_Presentation	*P.Signature_Attributes_Presentation*
O.Signature_Process_Interruption	*P.Signature_Process_Interruption*

5. EXTENDED COMPONENTS DEFINITION

There is no extended component in this security target.

6. SECURITY REQUIREMENTS

The following table lists the subjects, the objects, the operations and their security attributes used in the functional security requirements statement.

Subject	Object/Information	Operation	Security attributes
The Signatory	a document to be signed	To import the document	**The Signatory:** ■ signature policy **A document to be signed:** ■ document's identifier ■ document's stability status
The Signatory	The signatory's certificate	To import the signatory's certificate	**The Signatory:** ■ applied signature policy **the signatory's certificate:** ■ key usage status ■ QCStatement ■ certificate identifier
The signatory **The SCDev**	The formatted DTBS The electronic signature	To transfer the formatted DTBS to the SCDev	**The Signatory:** ■ applied signature policy ■ signatory's certificate **the formatted DTBS:** ■ the formatted DTBS **the Electronic signature:** ■ signature policy identifier ■ commitment type ■ claimed role ■ presumed signature date and time ■ presumed signature location
The signatory **The SCDev**	The formatted DTBS The electronic signature	To export the Electronic signature to the Signatory	**The SCDev:** ■ the status of signature generation process **the Electronic signature:** ■ the generated electronic signature ■ the signed document's hash ■ the reference to the signatory's certificate ■ the reference of the applied signature policy

6.1 Security functional requirements

The operations (assignment, selection, etc.) on the SFR contained in this section are identified in ***bold italics***. The nature of the operation (assignment, selection, etc.) is stated between brackets before the instantiated text.

6.1.1 Document stability control

The following requirements are related to the control of the invariance of the signed document's semantics.

6.1.1.1 Control during importation of the document

FDP_IFC.1/Document acceptance

FDP_IFC.1.1/Document acceptance
The TSF shall enforce the [assignment] ***document acceptance information flow control policy*** on [assignment] ***subjects: the signatory,*** ***information: a document to be signed*** ***operation: to import the document***

FDP_IFF.1/Document acceptance

FDP_IFF.1.1/Document acceptance
The TSF shall enforce the [assignment] ***document acceptance information flow control policy*** based on the following types of subject and information security attributes: [assignment] ***subjects: the signatory (signature policy,*** and signer's certificate***)*** ***information: a document to be signed (document's identifier*** ***operation: to import the document***

FDP_IFF.1.2/Document acceptance
The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [assignment] ***Importation of the document: the document's stability status equal "stable"***

FDP_IFF.1.3/Document acceptance
The TSF shall enforce the [assignment] ***set of* rules: none.**

FDP_IFF.1.4/Document acceptance
The TSF shall explicitly authorise an information flow based on the following rules: [assignment] ***controls succeed.***

FDP_IFF.1.5/Document acceptance
The TSF shall explicitly deny an information flow based on the following rules: [assignment] ***controls fail and controls cannot be bypassed***

Application note (FDP_IFF.1/Document acceptance)
The TOE shall provide the means: ■ to execute an module controlling if the semantics of the document to be signed are invariant, ■ to warn the signatory of the document if the semantics is not invariant,

FDP_ITC.1/Document acceptance

FDP_ITC.1.1/Document acceptance
The TSF shall enforce the [assignment] ***document acceptance information flow control policy*** when importing user data, controlled under the SFP, from outside of the TOE.

FDP_ITC.1.2/Document acceptance
The TSF shall ignore any security attributes associated with the user data when imported from outside the TOE.

FDP_ITC.1.3/Document acceptance
The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [assignment] ■ ***determine whether the document's semantics is invariant or not by invoking a dedicated module,*** ■ ***the TOE shall invoke an module in charge of controlling that the semantics of the document to be signed are invariant,*** ■ ***the TOE shall inform the signatory when the document's semantics are not stable.*** *Refinement:* The TOE shall inform the signatory when the document's semantics is unstable or cannot be checked.

Application note (FDP_ITC.1/Document acceptance)
The document semantics could vary for example if the document includes fields or active code that uses information external to the document.

FMT_MSA.3/Document's acceptance

FMT_MSA.3.1/Document's acceptance
The TSF shall enforce the [assignment] ***document acceptance information flow control policy*** to provide [selection] ***restrictive*** default values for security attributes that are used to enforce the SFP. *Refinement:* If the signature policy does not explicitly include a parameter specifying what to do in case the document is not detected as stable, then the default behavior will be to stop the signature process when the document is not detected as stable.

Note: the TOE will always abort the signature process when the document is not detected as being stable.

FMT_MSA.3.2/Document's acceptance
[Editorial refinement] The TSF shall allow [assignment] ***nobody*** to specify alternative initial values to override the default values when an object or information is created.

FMT_MSA.1/Selected documents

FMT_MSA.1.1/Selected documents
The TSF shall enforce the [assignment] document acceptance information flow control policy to restrict the ability to [selection] ***select*** the security attributes [assignment] ***documents' to be signed identifiers*** to the [assignment]calling application. Note: The TOE displays the contents of the document to be signed to the user. However, the document has been sent to the TOE by the calling application, through a parameter.

FMT_SMF.1/Selection of a list of documents

FMT_SMF.1.1/Selection of a list of documents
The TSF shall be capable of performing the following management functions:[assignment]

selecting a document to be signed.

Refinement:
The TSF shall allow the selection of documents to be signed until the signatory has given his agreement to sign.

Refinement:
The TOE can sign only one document

Application note (FMT_SMF.1/Selection of a list of documents)

The list of the documents to be signed can not change after the signatory's signature agreement. Nevertheless he can cancel the signature process at any moment (see *FDP_ROL.2/Abort of the signature process)*.

FMT_MSA.1/Document's semantics invariance status

FMT_MSA.1.1/Document's semantics invariance status

[Editorial refinement] The TSF shall enforce the [assignment] ***document acceptance information flow control policy*** to restrict the ability to [selection] ***modify*** the security attribute [assignment] ***document's stability status*** to [assignment] ***nobody***.

FMT_SMF.1/Getting document's semantics invariance status

FMT_SMF.1.1/Getting document's semantics invariance status

The TSF shall be capable of performing the following management functions: [assignment]

invoking an module to get the status indicating whether the document's semantics are invariant or not.

The two following security functional requirements are inapplicable because the TOE does not allow signing or displaying an unstable document.

~~**FMT_MSA.1.1/Signatory agreement to sign an unstable document**~~

~~The TSF shall enforce the document acceptance information flow control policy to restrict the ability to modify the security attributes signatory agreement to sign an unstable document to the signatory.~~

~~**FMT_SMF.1.1/Getting signatory agreement to sign an unstable document**~~

~~The TSF shall be capable of performing the following management functions:~~

~~get the explicit agreement of the signatory to sign a document whose semantics is unstable.~~

6.1.2 Interaction with the signatory

FDP_ROL.2/Abort of the signature process

FDP_ROL.2.1/Abort of the signature process

The TSF shall enforce the [assignment] ***signature generation information flow control policy*** to permit the rollback of [assignment] ***all the operations*** on the [assignment] ***electronic signature and its related attributes***.

FDP_ROL.2.2/Abort of the signature process
The TSF shall permit operations to be rolled back [assignment] ***before the formatted DTBS are transferred to the SCDev***.

6.1.3 Validation rules

6.1.3.1 Validation rules related to the signature attributes

The following requirements deal with the signature attributes.

FMT_MSA.1/Signature attributes

FMT_MSA.1.1/Signature attributes
The TSF sh**all enforce** the [assignment] ***signature generation information flow control policy*** to restrict the ability to [selection] ***select*** the [assignment] ***security attributes signature attributes*** to the [assignment] calling application.

FMT_SMF.1/Modification of signature attributes

FMT_SMF.1.1/Modification of signature attributes
The TSF shall be capable of performing the following management functions: [assignment] ***permit the signatory to change the value of the signature attributes required by the applied signature policy.*** *Refinement:* The TSF shall allow the modification of signature attributes until the signatory has given his agreement to sign.

6.1.3.2 Rules related to the signatory's certificate

The following requirements deal with the verification rules on the signatory's certificate.

FDP_IFC.1/Signatory's certificate import

FDP_IFC.1.1/Signatory's certificate import
The TSF shall enforce the [assignment] ***signatory's certificate information flow control policy*** on [assignment] ***subjects: the signatory*** ***information: the signatory's certificate*** ***operations: to import the signatory's certificate***

FDP_IFF.1/Signatory's certificate import

FDP_IFF.1.1/Signatory's certificate import
The TSF shall enforce the [assignment] signatory's certificate information flow control policy based on the following types of subject and information security attributes: [assignment] ***subjects: the signer (applied signature policy)*** ***information: the signatory's certificate (key usage,*** the Certificate Authority of the signer's certificate, the validity period time of the signer's certificate.***).***

FDP_IFF.1.2/Signatory's certificate import
The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [assignment] ***To import the signatory's certificate*** ■ ***the "key usage" of the selected signatory's certificate indicates that this certificate is usable for non repudiation purposes (Application note: bit 1 of keyUsage set)*** ■ ***the certificate is a Qualified Certificate if required by the signature policy (Application note: information available using a QCStatement, see RFC 3739 and [EXT_TS_101_862]),*** ■ ***the private key corresponding to public key is protected by an SCDev (Application note: information available using a QCStatement, see RFC 3739*** and [EXT_TS_101_862]***).*** ■ the certificate is issued by one of the Certificate Authorities defined by the calling application (through the signature policy) ■ the certificate is valid at the time indicated by the calling application (through the signature policy) or by the system's time if no time is specified.

FDP_IFF.1.3/Signatory's certificate import
The TSF shall enforce***. the other rules explicitly defined in the Signature SFP (eventually including the QCStatement).***

FDP_IFF.1.4/Signatory's certificate import
The TSF shall explicitly authorise an information flow based on the following rules: [assignment] ***controls succeed.***

FDP_IFF.1.5/Signatory's certificate import
The TSF shall explicitly deny an information flow based on the following rules: [assignment] ***controls fail.***

FMT_MSA.3/Signatory's certificate import

FMT_MSA.3.1/Signatory's certificate import
The TSF shall enforce the [assignment] ***signatory's certificate information flow control policy*** to provide [selection] ***restrictive*** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/Signatory's certificate import [Editorial refinement]
The TSF shall allow [assignment] ***nobody*** to specify alternative initial values to override the default values when an object or information is created.

FMT_MSA.1/Signatory's certificate

FMT_MSA.1.1/Signatory's certificate
The TSF shall enforce the [assignment] ***signatory's certificate information flow control policy*** to restrict the ability to [selection] ***select*** the security attributes [assignment] ***certificate identifier*** to the [assignment] ***signatory***.

FDP_ITC.2/Signatory's certificate

FDP_ITC.2.1/Signatory's certificate
The TSF shall enforce the [assignment] ***signatory's certificate information flow control policy*** when importing user data, controlled under the SFP, from outside of the TOE.

FDP_ITC.2.2/Signatory's certificate
The TSF shall use the security attributes associated with the imported user data.

FDP_ITC.2.3/Signatory's certificate
The TSF shall ensure that the protocol used provides for the unambiguous association between the security attributes and the user data received.

FDP_ITC.2.4/Signatory's certificate
The TSF shall ensure that interpretation of the security attributes of the imported user data is as intended by the source of the user data.

FDP_ITC.2.5/Signatory's certificate
The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [assignment] ***none***.

FPT_TDC.1/Signatory's certificate

FPT_TDC.1.1/Signatory's certificate
The TSF shall provide the capability to consistently interpret [assignment] ***certificates*** when shared between the TSF and another trusted IT product.

FPT_TDC.1.2/Signatory's certificate
The TSF shall use [assignment] the following list of interpretation rules : ■ interpretation of the DN of the signer's certificate ■ interpretation of the Certificate Authority of the signer's certificate ■ interpretation of the validity period time of the signer's certificate ■ interpretation of Key usage ■ interpretation of the QCStatements (qualified certificate and SSCD private key protection) when interpreting the TSF data from another trusted IT product.

Application note (FPT_TDC.1/Signatory's certificate)
The ST authors must here define standards supported by the TOE.

FMT_SMF.1/Signatory's certificate selection

FMT_SMF.1.1/Signatory's certificate selection
The TSF shall be capable of performing the following management functions: [assignment] ***allow the signatory to select a certificate among the list of certificates suitable for the applied signature policy.***

6.1.4 Application of the signature policy and generation of the signature

FDP_IFC.1/Signature generation

FDP_IFC.1.1/Signature generation
The TSF shall enforce the [assignment] ***signature generation information flow control policy*** on [assignment] ***subjects: the signatory, the SCDev*** ***information: the formatted DTBS, the electronic signature (once generated)*** ***operations: to transfer the formatted DTBS to the SCDev.***

FDP_IFF.1/Signature generation

FDP_IFF.1.1/Signature generation
The TSF shall enforce the [assignment] ***signature generation information flow control policy*** based on the following types of subject and information security attributes: [assignment] ***subjects: the signatory (applied signature policy, signatory's certificate, [assignment: any other signatory's attribute] (see FDP_IFF.1.2/Signature generation, the SCDev ([assignment: SCDev's attribute])*** ***information: the formatted DTBS (the data to be signed format), the electronic signature (signature policy identifier, commitment type, claimed role, presumed signature date and time, presumed signature location, [assignment: list of supported signature attributes]).***

FDP_IFF.1.2/Signature generation
The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [assignment] ***To transfer of the formatted DTBS:*** ■ ***to communicate the signature attributes to the signatory before the signature generation*** ■ ***todisplay the document in the format corresponding to the document's format according to the document format/viewer association table*** ■ ***to activate the signing key corresponding to the selected signatory's certificate.*** ***Electronic signature:*** ■ ***if the signature policy requires the inclusion of the signature attribute "signature policy identifier", then its value shall be included;*** ■ ***if the signature policy requires the inclusion of the signature attribute "commitment type", then its value shall be included;*** ■ ***if the signature policy restricts the values to be taken by the "commitment type" attribute, then its value shall be conformant to the signature policy;*** ■ ***if the signature policy requires the inclusion of the signature attribute "claimed role", then its value shall be included;*** ■ ***if the signature policy restricts the values to be taken by the "claimed role" attribute then its value shall be conformant to the signature policy;*** ■ ***if the signature policy prevents the inclusion of the signature attribute "presumed signature date and time", then its value shall not be included;*** ■ ***if the signature policy requires the inclusion of the signature attribute "presumed signature location", then its value shall be included;***

Refinement:
The signature policy never restricts the “commitment type” and the “claimed role”. Indeed, these parameters are specified by the calling application; the TOE only provide default values for these signature attributes.

FDP_IFF.1.3/Signature generation

The TSF shall enforce the [assignment] ***others rules explicitly defined in the applied signature policy.***

FDP_IFF.1.4/Signature generation

The TSF shall explicitly authorise an information flow based on the following rules: [assignment]

- ***Security attributes are compliant with Signature SFP***
- ***and the formatted DTBS semantic control succeed.***

FDP_IFF.1.5/Signature generation

The TSF shall explicitly deny an information flow based on the following rules: [assignment]

- ***Security attributes are not compliant with the Signature SFP***
- ***or the formatted DTBS semantics control fails.***

Application note (FDP_IFF.1/Signature generation)

The TOE must provide the means for:

- the communication of the signature attributes to the signatory before the generation of the signature,
- the execution of a viewer application for the format of the document to be signed according to the association table “format/viewer”
- the activation of the signature private key associated with the selected signatory’s certificate.

Note that the conformance of the signatory's certificate with respect to the applied signature policy is not check in the present policy but in the signatory's certificate information flow control policy that is the subject of component *FDP_IFC.1/Signatory's certificate import*. In the present component the conformance of the signatory's certificate is assumed established.

The “viewer application” is the calling application and does not depend on the document’s format, as the TOE provides it a reliable HTML representation for the document.

FMT_MSA.3/Signature generation

FMT_MSA.3.1/Signature generation

The TSF shall enforce the [assignment] ***signature generation information flow control policy*** to provide [selection] ***restrictive*** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/Signature generation [Editorial refinement]

The TSF shall allow [assignment] ***nobody*** to specify alternative initial values to override the default values when an object or information is created.

FDP_ITC.1/Explicit signatory agreement

FDP_ITC.1.1/Explicit signatory agreement

The TSF shall enforce the [assignment] ***signature generation information flow control policy*** when importing user data, controlled under the SFP, from outside of the TOE.

FDP_ITC.1.2/Explicit signatory agreement
The TSF shall ignore any security attributes associated with the user data when imported from outside the TOE.

FDP_ITC.1.3/Explicit signatory agreement
The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [assignment] the user has to check a checkbox to explicitly give his agreement.

Application note (FDP_ITC.1/Explicit signatory agreement)
FDP_ITC.1.3: the ST author must identify the list of actions that the TOE will consider as a proof of agreement for signature.

6.1.5 Electronic signature export

FDP_IFC.1/Electronic signature export

FDP_IFC.1.1/Electronic signature export
The TSF shall enforce the [assignment] ***electronic signature export information flow control policy*** on [assignment] ***subjects: the signatory, the SCDev*** ***information: the Electronic signature*** ***operations: to export the Electronic signature to the signatory.***

FDP_IFF.1/Electronic signature export

FDP_IFF.1.1/Electronic signature export
The TSF shall enforce the [assignment] ***electronic signature export information flow control policy*** based on the following types of subject and information security attributes: [assignment] ***subjects: the signatory, the SCDev (the status of signature generation process***) ***information: the Electronic signature (the generated electronic signature, the signed*** data's ***hash, the signatory's certificate, the reference of the applied signature policy).***

FDP_IFF.1.2/Electronic signature export
The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [assignment] ***Export of the electronic signature to the signatory is allowed if the signature generation (performed by the SCDev) succeeded.***

FDP_IFF.1.3/Electronic signature export
The TSF shall enforce the [assignment] following additional set of rules: Export of the electronic signature to the signer is allowed if the digital signature is in a PKCS #1 format.

FDP_IFF.1.4/Electronic signature export
The TSF shall explicitly authorise an information flow based on the following rules: [assignment] ***Signature generation succeeds.***

FDP_IFF.1.5/Electronic signature export
The TSF shall explicitly deny an information flow based on the following rules: [assignment] ***Signature generation fails.***

FDP_ETC.2/Electronic signature export

FDP_ETC.2.1/Electronic signature export
The TSF shall enforce the [assignment] ***electronic signature export information flow control policy*** when exporting user data, controlled under the SFP(s), outside of the TOE.

FDP_ETC.2.2/Electronic signature export
The TSF shall export the user data with the user data's associated security attributes.

FDP_ETC.2.3/Electronic signature export
The TSF shall ensure that the security attributes, when exported outside the TOE, are unambiguously associated with the exported user data.

FDP_ETC.2.4/Electronic signature export
The TSF shall enforce the following rules when user data is exported from the TOE: [assignment] the TOE shall export the signature using the XAdES (version1.3.2) format.

FMT_MSA.3/Electronic signature export

FMT_MSA.3.1/Electronic signature export
The TSF shall enforce the [assignment] ***electronic signature export information flow control policy*** to provide [selection] ***restrictive*** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/Electronic signature export [Editorial refinement]
The TSF shall allow [assignment] ***nobody*** to specify alternative initial values to override the default values when an object or information is created.

FMT_MSA.1/SCDev signature generation status

FMT_MSA.1.1/SCDev signature generation status
The TSF shall enforce the [assignment] ***electronic signature export information flow control policy*** to restrict the ability to [selection] ***modify*** the security attributes [assignment] ***SCDev's signature generation status*** to [assignment] ***nobody***.

FMT_SMF.1/Getting SCDev's signature generation status

FMT_SMF.1.1/Getting SCDev's signature generation status
The TSF shall be capable of performing the following management functions: [assignment] ***getting the SCDev's signature generation status (discriminate whether the signature generation process completed or failed).***

6.1.6 Cryptographic operations

FCS_COP.1/Hash function

FCS_COP.1.1/Hash function
The TSF shall perform [assignment] ***hash*** generation in accordance with a specified cryptographic algorithm [assignment] SHA-256 that meet the following: [assignment] [CRYPT]**,**

Application note
The ST author must select a hash generating algorithm which does not produce identical message-hashes out of two distinct documents.

6.1.7 User identification and authentication

FMT_SMR.1 Security roles

FMT_SMR.1.1
The TSF shall maintain the roles [assignment] ***Signatory*** ***Calling application***

FMT_SMR.1.2
The TSF shall be able to associate users with roles.

FIA_UID.2 User identification before any action

FIA_UID.2.1
The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

Application note (FIA_UID.2 User identification before any action)
The authentication mechanism must be compliant with the authentication reference document of the ANSSI **[AUTH]**.

6.1.8 TOE administration

6.1.8.1 Capability to view the document to be signed

FMT_MTD.1/Document format/viewer association table

FMT_MTD.1.1/Document format/viewer association table
The TSF shall restrict the ability to [selection] ***modify*** the [assignment] ***document format/viewer association table*** to [assignment] *nobody* Note: this table is defined once for all in the TOE and contains two entries: one for the text format and one for the HTML format.

FMT_SMF.1/Management of the document format/viewer association table

FMT_SMF.1.1/Management of the document format/viewer association table
The TSF shall be capable of performing the following management functions: [assignment] ***none***

Application note (FMT_SMF.1/Management of the document format/viewer association table)
In the "assignment", the ST author must define the operations on the document format/viewer association table allowed by the TOE to the security administrator. The possible operations could be addition and deletion of entries, the modification of the viewer application, etc.

6.1.8.2 Signature policy management

FMT_MTD.1/Management of the signature policies

FMT_MTD.1.1/Management of the signature policies
The TSF shall restrict the ability to[selection] define the [assignment] ***signature policies*** to the [assignment] ***calling application*** of the TOE. Note: The signature policy is transmitted to the TOE by the calling application. However, it is specified by the calling application administrator.

Application note (FMT_MTD.1/Management of the signature policies)
The assignment must be consistent with the assignment of the component *FMT_SMF.1/Management of the signature policies*.

FMT_SMF.1/Management of the signature policies

FMT_SMF.1.1/Management of the signature policies
The TSF shall be capable of performing the following management functions: [assignment] *define the security policy*.

Application note (FMT_SMF.1/Management of the signature policies)
The assignment must be consistent with the assignment of the component *FMT_MTD.1/Management of the signature policies*.

6.2 Security assurance requirements

The required evaluation level is EAL3 augmented with AVA_VAN.3 and ALC_FLR.3.

The evaluation level comes from **[QUA_STD]**.

Security assurance requirements are:

Requirements
ADV_ARC.1
ADV_FSP.3
ADV_TDS.2
AGD_OPE.1
AGD_PRE.1

Requirements
ALC_CMC.3
ALC_CMS.3
ALC_DEL.1
ALC_DVS.1
ALC_FLR.3
ALC_LCD.1
ASE_CCL.1
ASE_ECD.1
ASE_INT.1
ASE_OBJ.2
ASE_REQ.2
ASE_SPD.1
ASE_TSS.1
ATE_COV.2
ATE_FUN.1
ATE_IND.2
ATE_DPT.1
AVA_VAN.3

6.3 Security requirements rationale

6.3.1 Security objectives for the TOE

6.3.1.1 General objectives

O.Certificate/Private_Key_Association

The objective is covered by the requirement **FDP_IFF.1/Signature generation**. This requirement requires that the TOE is able to activate the private key of signature corresponding to the certificate selected by the signatory.

6.3.1.2 Interaction with the signatory

O.Signature_Attributes_Presentation

The objective is covered by the **FDP_IFF.1/Signature generation** requirement which requires in particular that the TOE can present the signature attributes to the signatory before the beginning of the signature process.

O.Explicit_Agreement

The objective is covered by the **FDP_ITC.1/Explicit signatory agreement** requirement by which the TOE requires that a succession of non-trivial operations is carried out before considering the effective agreement to sign.

O.Signature_Process_Interruption

The objective is covered by the **FDP_ROL.2/Abort of the** signature process requirement which ensures that the signatory has the possibility of cancelling the signature before sending the data to the SCDev.

O.Documents_To_Be_Signed

The objective is covered by the functional requirements:

FMT_MSA.1/Selected documents which restricts the capacity to select documents to be signed to the calling application only.

FMT_SMF.1/Selection of a list of documents which requires that the TOE allows to select a single document to be signed as long as the signatory did not give his agreement to sign.

FMT_MSA.1/Signature attributes which restricts to the signatory only the capacity to select the signature attributes.

FMT_SMF.1/Modification of signature attributes which requires that the TOE makes it possible to modify the value of the signature attributes as long as the signatory did not give his agreement to sign.

6.3.1.3 Signature policy application

O.Signatory_Certificate_Conformity

The objective is covered in the following way:

The TOE must apply a flow control policy during the importation of a certificate (**FDP_IFC.1/Signatory's certificate import**). The functional component **FDP_IFF.1/Signatory's certificate import** defines that this policy allows the importation of the certificate in the TOE if the rules defined in the signature policy are fulfilled. These rules are related to the signatory's certificate. The compliance of the selected certificate is guaranteed if its attributes fulfill the rules defined in the signature policy.

The functional components **FDP_ITC.2/Signatory's certificate** and **FPT_TDC.1/Signatory's certificate** ensure on the one hand that the TOE applies the rules of the flow control policy during the importation of the selected certificate and on the other hand that the TOE is able to exploit the data contained in the imported certificate.

The following functional components, related to the management of the security attributes of the subjects and information concerned in the flow control policy also contribute to cover this objective:

- The functional component **FMT_MSA.3/Signatory's certificate import** guarantees that the default values assigned to the attributes of security concerned in the flow control policy take restrictive values.
- The functional components **FMT_MSA.1/Signatory's certificate** and **FMT_SMF.1/Signatory's certificate selection** guarantee to the signatory the exclusive right to select the suitable certificate for electronic signatures he wishes to perform.
- Component **FMT_SMR.1 Security roles** requires of the TOE to distinguish the role of signatory from the role of the calling application.

- Component **FIA_UID.2 User identification before** any action requires that the TOE does not allow the realization of any operation before having identified successfully the user.

O.Signatory_Certificate_Validity

The objective is covered in the following way:

The TOE must apply a flow control policy during the importation of a certificate (**FDP_IFC.1/Signatory's certificate import**). The functional component **FDP_IFF.1/Signatory's certificate import** defines that this policy allows the importation of the certificate in the TOE if the rules defined in the signature policy are fulfilled. These rules are related to the signatory's certificate. The compliance of the selected certificate is guaranteed if its attributes fulfil the rules defined in the signature policy.

The functional components **FDP_ITC.2/Signatory's certificate** and **FPT_TDC.1/Signatory's certificate** ensure on the one hand that the TOE observes the rules of the flow control policy during the importation of the selected certificate and on the other hand that the TOE is able to exploit the data contained in the imported certificate.

The following functional components, related to the management of the security attributes of the subjects and information concerned in the flow control policy also contribute to cover this objective:

- The functional component **FMT_MSA.3/Signatory's certificate import** guarantees that the default values assigned to the security attributes concerned in the flow control policy take restrictive values.
- The functional components **FMT_MSA.1/Signatory's certificate** and FMT_SMF.1/Signatory's certificate selection guarantee to the signatory the exclusive right to select the suitable certificate for electronic signatures he wishes to perform.
- Component **FMT_SMR.1 Security roles** requires of the TOE to distinguish the role of signatory from the role of the calling application.
- Component **FIA_UID.2 User identification before** any action requires that the TOE does not allow the realization of any operation before having identified successfully the user.

O.Signature_Attributes_Conformity

The objective is covered in the following way:

The TOE must apply a flow control policy during the generation of a signature (**FDP_IFC.1/Signature generation**). The functional component **FDP_IFF.1/Signature generation** defines that this policy allows the generation of the signature (i.e. the sending of the formatted DTBS to the SCDev) if the rules defined in the signature policy are fulfilled. This component also defines rules related to the signature attributes. The compliance of the signature attributes is guaranteed if these attributes fill the rules defined in the signature policy.

The following functional components, related to the management of the security attributes of the subjects and information concerned in the flow control policy also contribute to cover this objective:

- The functional component **FMT_MSA.3/Signature generation** guarantees that the default values of the attributes concerned in the flow control policy have restrictive values.
- The functional component **FMT_MSA.1/Signature attributes** and **FMT_SMF.1/Modification of signature** attributes guarantee to the signatory the exclusive right to select the suitable certificate for electronic signatures he wishes to perform.

- Component **FMT_SMR.1 Security roles** requires of the TOE to distinguish the role of signatory from the role of the calling application.
- Component **FIA_UID.2 User identification before** any action requires that the TOE does not allow the realization of any operation before having identified successfully the user.

O.Electronic_Signature_Export

The objective is covered in the following way:

The TOE must apply an information flow control policy during the importation of a document into the TOE (**FDP_IFC.1/Electronic signature export**). The functional component **FDP_IFF.1/Electronic signature export** defines the rules to be applied by the TOE to export the created electronic signatures.

The component FDP_ETC.2/**Electronic** signature export requires the TOE to export the electronic signature together with its security attributes, as defined in the electronic signature export information flow control policy. These attributes cover all the data to be exported to the user, as required by the objectiveMoreover, as per the objective, that information flow policy requires the TOE to produce a XAdES formatted signature.

The following components related to the management of the security attributes of the subjects and information concerned in the flow control policy also contribute to cover this objective:

- The functional component **FMT_MSA.3/Electronic signature export** guarantees that the default values of the security attributes concerned in the flow control policy have restrictive values.
- The functional component **FMT_SMF.1/Getting SCDev's signature** generation status requires that the TOE is able to receive from the SCDev the status of the operation of generation of the electronic signature.
- The functional component **FMT_MSA.1/SCDev signature generation** status which does not allow anybody to modify the status of the operation of generation of the signature returned by the SCDev.
- Component **FMT_SMR.1 Security roles** requires of the TOE to distinguish the role of signatory from the role of the calling application.
- Component **FIA_UID.2 User identification before** any action requires that the TOE does not allow the execution of any operation before having identified successfully the user.

6.3.1.4 Data protection

O.Administration

The objective is covered by the following functional components:

- **FMT_SMR.1 Security roles** which requires the to TOE distinguish the role of the calling application from the role of signatory;
- **FIA_UID.2 User identification before** any action requires the TOE to identify any user before any action;
- **FMT_MTD.1/Document format/viewer** association table and **FMT_SMF.1/Management of the** document format/viewer association table which allows nobody to modify the table of association between the document formats and the viewer applications;
- **FMT_SMF.1/Management of the** signature policies which defines the operations of management of the signature policies and **FMT_MTD.1/Management of the** signature policies which restricts their use to the the calling application.

6.3.1.5 Cryptographic operations

O.Cryptographic_Operations

The objective is covered by the requirement **FCS_COP.1/Hash function** which allows the security targets authors to define the hash algorithms implemented in the TOE.

6.3.1.6 Control of the invariance of the document's semantics

O.Document_Stability_Control

The objective is covered in the following way:

The TOE must apply a flow control policy during the importation of a document into the TOE (**FDP_IFC.1/Document acceptance**). The functional component **FDP_IFF.1/Document acceptance** defines the rules to be applied by the TOE to accept the document.

The component **FDP_ITC.1/Document acceptance** requires that the TOE check using a module whether the document's semantics is invariant or not when it imports the document.

The following functional components related to the management of the security attributes of the subjects and information concerned in the flow control policy also contribute to cover this objective:

- The functional component **FMT_MSA.3/Document's acceptance** guarantees that the default values of the security attributes concerned in the flow control policy have restrictive values.
- The functional components **FMT_MSA.1/Document's semantics invariance** status and **FMT_SMF.1/Getting document's semantics** invariance status which require on the one hand that the TOE has a means of executing an external module to determine whether the document's semantics is invariant, on the other hand that nobody can modify the result of the control.
- **FMT_SMR.1 Security roles** which requires the to TOE distinguish the role of the calling application from the role of signatory;
- Component **FIA_UID.2 User identification before** any action requires that the TOE does not allow the realization of any operation before having identified successfully the user.

6.3.1.7 Presentation of the documents to be signed

O.Document Presentation

The objective *O.Document Presentation* is covered by the following components:

- FDP_IFF.1/Signature generation ensures that the user will be able to visualize the document as displayed by the TOE itself. The TOE uses the appropriate visualization module according to the document format/viewer table and the document's format, as specified by the calling application.
- FMT_MTD.1/Document format/viewer association table and FMT_SMF.1/Management of the document format/viewer association table guarantee that the “document format/viewer table” cannot be modified.

O.SCDev Response Validation

The objective is covered by FDP_IFF.1/Electronic signature export, which requires the TOE to check that the data returned by the SCDev is indeed a [PKCS #1] electronic signature.

6.3.2 Completness of the coverage

The following table ensures that there is no spurious SFR.

SFR	Objectives
FCS_COP.1/Hash function	*O.Cryptographic_Operations*
FDP_ETC.2/Electronic signature export	*O.Electronic_Signature_Export*
FDP_IFC.1/Document acceptance	*O.Document_Stability_Control*
FDP_IFC.1/Electronic signature export	*O.Electronic_Signature_Export*
FDP_IFC.1/Signatory's certificate import	*O.Signatory_Certificate_Validity,* *O.Signatory_Certificate_Conformity*
FDP_IFC.1/Signature generation	*O.Signature_Attributes_Conformity*
FDP_IFF.1/Document acceptance	*O.Document_Stability_Control*
FDP_IFF.1/Electronic signature export	*O.SCDev Response Validation,* *O.Electronic_Signature_Export*
FDP_IFF.1/Signatory's certificate import	*O.Signatory_Certificate_Validity,* *O.Signatory_Certificate_Conformity*
FDP_IFF.1/Signature generation	*O.Signature_Attributes_Conformity,* *O.Certificate/Private_Key_Association,* *O.Document Presentation,* *O.Signature_Attributes_Presentation*
FDP_ITC.1/Document acceptance	*O.Document_Stability_Control*
FDP_ITC.1/Explicit signatory agreement	*O.Explicit_Agreement*
FDP_ITC.2/Signatory's certificate	*O.Signatory_Certificate_Validity,* *O.Signatory_Certificate_Conformity*
FDP_ROL.2/Abort of the signature process	*O.Signature_Process_Interruption*
FIA_UID.2 User identification before any action	*O.Signature_Attributes_Conformity,* *O.Signatory_Certificate_Validity,* *O.Administration,* *O.Electronic_Signature_Export,* *O.Document_Stability_Control,* *O.Signatory_Certificate_Conformity*
FMT_MSA.1/Document's semantics invariance status	*O.Document_Stability_Control*
FMT_MSA.1/SCDev signature generation status	*O.Electronic_Signature_Export*
FMT_MSA.1/Selected documents	*O.Documents_To_Be_Signed*
FMT_MSA.1/Signatory's certificate	*O.Signatory_Certificate_Validity,* *O.Signatory_Certificate_Conformity*
FMT_MSA.1/Signature attributes	*O.Signature_Attributes_Conformity,* *O.Documents_To_Be_Signed*
FMT_MSA.3/Document's acceptance	*O.Document_Stability_Control*
FMT_MSA.3/Electronic signature export	*O.Electronic_Signature_Export*
FMT_MSA.3/Signatory's certificate import	*O.Signatory_Certificate_Validity,* *O.Signatory_Certificate_Conformity*
FMT_MSA.3/Signature generation	*O.Signature_Attributes_Conformity*
FMT_MTD.1/Document format/viewer association table	*O.Administration, O.Document Presentation*
FMT_MTD.1/Management of the signature policies	*O.Administration*
FMT_SMF.1/Getting SCDev's signature generation status	*O.Electronic_Signature_Export*
FMT_SMF.1/Getting document's semantics invariance status	*O.Document_Stability_Control*

SFR	Objectives
FMT_SMF.1/Management of the document format/viewer association table	*O.Administration, O.Document Presentation*
FMT_SMF.1/Management of the signature policies	*O.Administration*
FMT_SMF.1/Modification of signature attributes	*O.Signature_Attributes_Conformity, O.Documents_To_Be_Signed*
FMT_SMF.1/Selection of a list of documents	*O.Documents_To_Be_Signed*
FMT_SMF.1/Signatory's certificate selection	*O.Signatory_Certificate_Validity, O.Signatory_Certificate_Conformity*
FMT_SMR.1 Security roles	*O.Signature_Attributes_Conformity, O.Signatory_Certificate_Validity, O.Administration, O.Electronic_Signature_Export, O.Document_Stability_Control, O.Signatory_Certificate_Conformity*
FPT_TDC.1/Signatory's certificate	*O.Signatory_Certificate_Validity, O.Signatory_Certificate_Conformity*

6.3.3 Dependencies of the functional security requirements

6.3.3.1 Satisfied dependencies

FCS_COP.1/Hash function	(FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1) and FCS_CKM.4	*See below*
FDP_ETC.2/Electronic signature export	(FDP_ACC.1 or FDP_IFC.1)	**FDP_IFC.1/Electronic signature export**,
FDP_IFC.1/Document acceptance	FDP_IFF.1	**FDP_IFF.1/Document acceptance**,
FDP_IFC.1/Electronic signature export	FDP_IFF.1	**FDP_IFF.1/Electronic signature export**
FDP_IFC.1/Signatory's certificate import	FDP_IFF.1	**FDP_IFF.1/Signatory's certificate import**
FDP_IFC.1/Signature generation	FDP_IFF.1	**FDP_IFF.1/Signature generation**
FDP_IFF.1/Document acceptance	FDP_IFC.1 and FMT_MSA.3	**FDP_IFC.1/Document acceptance**, , **FMT_MSA.3/Document's acceptance**,
FDP_IFF.1/Electronic signature export	FDP_IFC.1 and FMT_MSA.3	**FDP_IFC.1/Electronic signature export**, **FMT_MSA.3/Electronic signature export**,
FDP_IFF.1/Signatory's certificate import	FDP_IFC.1 and FMT_MSA.3	**FDP_IFC.1/Signatory's certificate import**, **FMT_MSA.3/Signatory's certificate import**,
FDP_IFF.1/Signature generation	FDP_IFC.1 and FMT_MSA.3	**FDP_IFC.1/Signature generation**, **FMT_MSA.3/Signature generation**
FDP_ITC.1/Document acceptance	(FDP_ACC.1 or FDP_IFC.1) and FMT_MSA.3	**FDP_IFC.1/Document acceptance**, **FMT_MSA.3/Document's acceptance**
FDP_ITC.1/Explicit signatory agreement	(FDP_ACC.1 or FDP_IFC.1) and FMT_MSA.3	**FDP_IFC.1/Signature generation**, **FMT_MSA.3/Signature generation**

FDP_ITC.2/Signatory's certificate	(FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1) and FPT_TDC.1	**FDP_IFC.1/Signatory's certificate import**, **FPT_TDC.1/Signatory's certificate** *See below for FPT_TDC.1*
FDP_ROL.2/Abort of the signature process	(FDP_ACC.1 or FDP_IFC.1)	**FDP_IFC.1/Signature generation**
FIA_UID.2 User identification before any action	(none)	
FMT_MSA.1/Document's semantics invariance status	(FDP_ACC.1 or FDP_IFC.1) and FMT_SMR.1 and FMT_SMF.1	**FDP_IFC.1/Document acceptance**, **FMT_SMF.1/Getting document's semantics** invariance status, **FMT_SMR.1 Security roles**
FMT_MSA.1/SCDev signature generation status	(FDP_ACC.1 or FDP_IFC.1) and FMT_SMR.1 and FMT_SMF.1	**FDP_IFC.1/Electronic signature export**, **FMT_SMF.1/Getting SCDev's signature** generation status, **FMT_SMR.1 Security roles**
FMT_MSA.1/Selected documents	(FDP_ACC.1 or FDP_IFC.1) and FMT_SMR.1 and FMT_SMF.1	**FDP_IFC.1/Document acceptance**, **FMT_SMF.1/Selection of a** list of documents, **FMT_SMR.1 Security roles**
FMT_MSA.1/Signatory's certificate	(FDP_ACC.1 or FDP_IFC.1) and FMT_SMR.1 and FMT_SMF.1	**FDP_IFC.1/Signatory's certificate import**, **FMT_SMF.1/Signatory's certificate selection**, **FMT_SMR.1 Security roles**
FMT_MSA.1/Signature attributes	(FDP_ACC.1 or FDP_IFC.1) and FMT_SMR.1 and FMT_SMF.1	**FDP_IFC.1/Signature generation**, **FMT_SMF.1/Modification of signature** attributes, **FMT_SMR.1 Security roles**
FMT_MSA.3/Document's acceptance	FMT_MSA.1 and FMT_SMR.1	**FMT_MSA.1/Document's semantics invariance** status, **FMT_MSA.1/SCDev signature generation** status, **FMT_MSA.1/Selected documents**, **FMT_MSA.1/Signatory's certificate**, **FMT_MSA.1/Signature attributes**, **FMT_SMR.1 Security roles**
FMT_MSA.3/Electronic signature export	FMT_MSA.1 and FMT_SMR.1	**FMT_MSA.1/Document's semantics invariance** status, **FMT_MSA.1/Selected documents**, **FMT_SMR.1 Security roles**
FMT_MSA.3/Signatory's certificate import	FMT_MSA.1 and FMT_SMR.1	**FMT_MSA.1/Signatory's certificate**, **FMT_SMR.1 Security roles**
FMT_MSA.3/Signature generation	FMT_MSA.1 and FMT_SMR.1	**FMT_MSA.1/Signatory's certificate**, **FMT_MSA.1/Signature attributes**, **FMT_SMR.1 Security roles**
FMT_MTD.1/Document format/viewer association table	FMT_SMR.1 and FMT_SMF.1	**FMT_SMF.1/Management of the** document format/viewer association table, **FMT_SMR.1 Security roles**
FMT_MTD.1/Management of the signature policies	FMT_SMR.1 and FMT_SMF.1	**FMT_SMF.1/Management of the** signature policies, **FMT_SMR.1 Security roles**
FMT_SMF.1/Getting document's semantics invariance status	(none)	
FMT_SMF.1/Getting SCDev's signature generation status	(none)	

FMT_SMF.1/Management of the document format/viewer association table	(none)	
FMT_SMF.1/Management of the signature policies	(none)	
FMT_SMF.1/Modification of signature attributes	(none)	
FMT_SMF.1/Selection of a list of documents	(none)	
FMT_SMF.1/Signatory's certificate selection	(none)	
FMT_SMR.1 Security roles	FIA_UID.1	**FIA_UID.2 User identification before** any action
FPT_TDC.1/Signatory's certificate	(none)	

6.3.3.2 Rationale for the unsatisfied dependencies

Dependence FCS_CKM.4 of **FCS_COP.1/Hash function** is not supported. The dependence with FCS_CKM.4 is not satisfied because the hash function does not require any cryptographic key.

Dependence FCS_CKM.1 OR FDP_ITC.1 OR FDP_ITC.2 of **FCS_COP.1/Hash function** is not supported. The dependence with FCS_CKM.1, FDP_ITC.1 or FDP_ITC.2 is not satisfied because the hash function requires neither the generation nor the importation of keys in the TOE.

Dependence FTP_ITC.1 OR FTP_TRP.1 of **FDP_ITC.2/Signatory's certificate** is not supported. The dependence between the component **FDP_ITC.2/Signatory's certificate** and one of components FTP_ITC.1 or TFP_TRP.1 is not satisfied because the protocols used in the public key infrastructures are self-protected and guaranteed, not immediately, but during the verification of the signature:

- the integrity of the certificates of the certification chain is guaranteed thanks to the self-signed certificate (or trusted point) defined in the signature policy whose integrity is maintained by the environment of the TOE
- during the verification of the signature, the fact of building a valid certification chain between the signatory's certificate and the trusted point defined in the signature policy allows to guarantee the authenticity of the origin of the various certificates composing this chain.
- finally, the signatory's certificate does not require any confidentiality protection.

6.3.4 Evaluation assurance level rationale

The assurance level of this ST is EAL3+, because it is required by the ANSSI standard qualification process **[QUA_STD]**.

6.3.5 EAL augmentation rationale

6.3.5.1 AVA_VAN.3 Focused vulnerability analysis

Augmentation required by the standard qualification process.

6.3.5.2 ALC_FLR.3 Systematic flaw remediation

Augmentation required by the standard qualification process.

6.3.6 Dependencies of the security assurance requirements

Requirements	CC Dependencies	Satisfied Dependencies
ADV_ARC.1	(ADV_FSP.1) and (ADV_TDS.1)	ADV_FSP.3, ADV_TDS.2
ADV_FSP.3	(ADV_TDS.1)	ADV_TDS.2
ADV_TDS.2	(ADV_FSP.3)	ADV_FSP.3
AGD_OPE.1	(ADV_FSP.1)	ADV_FSP.3
AGD_PRE.1	No dependence	
ALC_CMC.3	(ALC_CMS.1) and (ALC_DVS.1) and (ALC_LCD.1)	ALC_CMS.3, ALC_DVS.1, ALC_LCD.1
ALC_CMS.3	No dependence	
ALC_DEL.1	No dependence	
ALC_DVS.1	No dependence	
ALC_FLR.3	No dependence	
ALC_LCD.1	No dependence	
ASE_CCL.1	(ASE_ECD.1) and (ASE_INT.1) and (ASE_REQ.1)	ASE_ECD.1, ASE_INT.1, ASE_REQ.2
ASE_ECD.1	No dependence	
ASE_INT.1	No dependence	
ASE_OBJ.2	(ASE_SPD.1)	ASE_SPD.1
ASE_REQ.2	(ASE_ECD.1) and (ASE_OBJ.2)	ASE_ECD.1, ASE_OBJ.2
ASE_SPD.1	No dependence	
ASE_TSS.1	(ADV_FSP.1) and (ASE_INT.1) and (ASE_REQ.1)	ADV_FSP.3, ASE_INT.1, ASE_REQ.2
ATE_COV.2	(ADV_FSP.2) and (ATE_FUN.1)	ADV_FSP.3, ATE_FUN.1
ATE_FUN.1	(ATE_COV.1)	ATE_COV.2
ATE_IND.2	(ADV_FSP.2) and (AGD_OPE.1) and (AGD_PRE.1) and (ATE_COV.1) and (ATE_FUN.1	ADV_FSP.3, AGD_OPE.1, AGD_PRE.1, ATE_COV.2, ATE_FUN.1
ATE_DPT.1	(ADV_ARC.1) and (ADV_TDS.2) and (ATE_FUN.1)	ADV_ARC.1, ADV_TDS.2, ATE_FUN.1

Requirements	CC Dependencies	Satisfied Dependencies
AVA_VAN.3	(ADV_ARC.1) and (ADV_FSP.4) and (ADV_IMP.1) and (ADV_TDS.3) and (AGD_OPE.1) and (AGD_PRE.1) and (ATE_DPT.1)	ADV_ARC.1, AGD_OPE.1, AGD_PRE.1, ATE_DPT.1

6.3.6.1 Rationale for the unsatisfied dependencies

Dependence ADV_FSP.4 of AVA_VAN.3 is not supported.

Dependence ADV_IMP.1 of AVA_VAN.3 is not supported.

Dependence ADV_TDS.3 of AVA_VAN.3 is not supported.

The dependence of the components with AVA_VAN.3 ADV_FSP.4, and ADV_IMP.1 ADV_TDS.3 is not satisfied. This security target claims conformance with the assurance package defined by the standard qualification process. So all unsatisfied dependencies of QS_STD are unchanged.

7. TOE SUMMARY SPECIFICATION

7.1 TOE Summary specification

F.SIGNATURE

This function signs a document. Its parameters are:

- From the calling application, as a parameter call:
  - the document to be signed, base64-encoded
  - the XAdES version of the signature to produce: 1.3.2 or ~~1.4.1~~
  - the hashing algorithm to be used
- From *F.Semantic invariance check*:
  - The document's stability status
- From *F.Signature policy application*:
  - The reference to the signature policy
  - The engagement type
  - The signer's role
  - The signature's location (optional parameter)
  - The explicit signer's approval to sign the document
- From *F.Signing certificate selection*:
  - The signing certificate to be used

The function *F.Signature* sends a signature request to the SCDev once it has received the signer's approval to sign. To this end:

1. It formats the data to be signed into the XAdES format and initializes the XAdES signature using *F.Signature policy application*
2. It computes the hash value of these data using the specified algorithm.

   Depending on the case, the TOE sends either that hash or the DTBS to the SCDev. In the former, this is the SCDev (and, eventually, the middleware) which computes the hash before generating the digital signature. In all cases, the hash value is returned together with the digital signature in the next step.
3. It asks the SCDev to electronically sign the hash value, using the signing certificate

   At any time before this step, the user may abort the signature process. Once the SCDev has produced the electronic signature to this function, the signature process is considered completed and may no longer be aborted by the user.
4. It add the hash value and the hash algorithm's OID to the XAdES signature
5. It adds the digital signature returned by the SCDev to the XAdES signature
6. It returns the XAdES signature

The XAdES signature contains, among others:

- The digital signature returned by the SCDev
- The hash value and the hash algorithm's OID

- The signing certificate
- A reference to the signature policy

Should the process fail or be interrupted, the applet's runtime ends and the user is sent to the appropriate web page.

F.SEMANTIC INVARIANCE CHECK

The TOE checks whether the document's semantics is stable or not, depending on its format. If the data are in "raw text" format, the document is considered to be stable.

If the data are in HTML format, the TOE checks that the HTML conforms to the criteria given in *Annexe: Supported subset of the HTML*.

- If this is the case, the document's semantics are considered to be stable.
- If this is not the case, the document's semantics are considered to be unstable.

If the semantics are deemed unstable, this function returns an error and the signature process is immediately aborted.

F.SIGNING CERTIFICATE SELECTION

This function asks the signer to select a certificate among a list of available certificates. That list is build up according to the signature policy, that is, the TOE only presents the SCDev-stored certificates that conform to the signature policy's criteria.

F.DOCUMENT VIEWER

This function displays the document to the signer. If the document's format is "raw text", the document is displayed within a textbox.

If the document's format is HTML, then it is formatted according to the HTML's semantics (note that the TOE only supports a subset of the HTML).

This function has the following input parameters:

- The document itself
- The document's format (raw text or HTML)

F.ADMINISTRATION

This function allows the calling application to specify the following parameters:

- The signature policy (whose contents are detailed in *F.Signature policy application*)
- The document's format (raw text or HTML)
- GUI resources (all these parameters are optional and have conservative default values):
  - Image for the "OK" button
  - Image for the "Cancel" button
  - Image for the "Update" button (to update the certificates' list)
  - The GUI's background colour
  - The GUI's foreground colour

- The GUI's font

F.SIGNATURE POLICY APPLICATION

A signature policy is defines the following data:

- The list of authorised CAs
- The signing certificate's public key
- A reference to the signature policy (usually, its OID), to be included in the data to be signed
- A reference date for the signature
- An engagement type (part of the data to be signed)
- The signer's role (part of the data to be signed)
- The signature's location (part of the data to be signed)
- Must the signing certificate have the non-repudiation *keyUsage* set?
- Must the signing certificate have the digital signature *keyUsage* set?
- Must the signing certificate have the S/MIME encryption *keyUsage* set?
- Must the signing certificate have the encipherment *keyUsage* set?
- Must the signing certificate be qualified?
- Must the signing certificate's private key be stored on a SSCD?

If some of these data are not explicitly specified by the calling application, the following default values are used:

- any CA is authorised
- no restriction on the certificate's public key
- the current date is used for the filtering date
- no engagement type
- no signer's role
- no signing location
- No restriction on the signing certificate's *keyUsage*, qualification or private key storage, except for the digital signature *keyUsage*, which is required by default.

The reference to the signing policy is mandatory. If the calling application does not provide this parameter, the signature process is aborted.

This function applies the signature policy:

1. It filters the SCDev-stored certificates according to the authorised CA's, the certificate's public key (if present), the given date (that is, only the certificates that are valid at this time are eligible), the *keyUsage* constraint and the other constraints on the certificate's attributes (qualification, SSCD).
2. It inserts, in the XAdES signature's attributes, the signature policy's reference, the engagement type, the signer's role, the signature's location, the signature's date.

F.TRANSFERT TO THE SCDEV

This function communicates with the SCDev:

- It retrieves from the SCDev the list of the certificates it contains
- It asks the SCDev to generate a *PKCS #1* signature of a given hash value, using a given private key.

  Depending on the case, that function may directly send the DTBS to the SCDev. In that case, the SCDev (together with, eventually, the middleware) computes the hash before generating the PKCS #1 digital signature.
- It checks that the generated signature is indeed a PKCS #1 signature and returns an error if this is not the case (this aborts the signature generation process).

7. It checks the data the generated signature corresponds to the data (hash value or DTBS) it has previously sent and returns an error if this is not the case (this also aborts the signature generation process).

F.SIGNATURE ATTRIBUTE VIEWER

This function displays the signature attributes to the signer. These are:

- The reference of the signature policy
- The engagement type
- The signer's role
- A summary of the signing certificate: its CN, its serial number, the issuer's CA, its validity period
- The signing date
- The signature's location

7.2 TSS Rationale

F.Signature

This function covers the following SFR's:

- **FMT_MSA.1/Selected documents**, **FMT_SMF.1/Selection of a** list of documents, as the calling application is the only one to "select" the document to be signed.
- **FDP_ROL.2/Abort of the** signature process, as this function supports the user-initiated abortion of the signature process before the call to the SCDev.
- **FDP_ITC.1/Explicit signatory agreement**, as it requires the signer's approval before generating the signature
- **FDP_IFC.1/Electronic signature export**, **FDP_IFF.1/Electronic signature export**, **FDP_ETC.2/Electronic signature export** and **FMT_MSA.3/Electronic signature export**, as it generates and builds up the XAdES signature.
- **FCS_COP.1/Hash function**, because it computes the signed data's hash.

This function also contributes to the coverage of **FMT_SMR.1 Security roles** because it provides separate interfaces to the signer and to the calling application.

F.Semantic invariance check

This function covers the following SFR's:

- **FDP_IFC.1/Document acceptance**, **FDP_IFF.1/Document acceptance** and **FDP_ITC.1/Document acceptance**, as it implements the checking of the document's semantic invariance.
- **FMT_MSA.3/Document's acceptance**, as the default value for the document's invariance status is "false".
- **FMT_MSA.1/Document's semantics invariance** status, **FMT_SMF.1/Getting document's semantics** invariance status, as it implements the checking of the document's semantic invariance.

F.Signing certificate selection

This function covers the following SFR's:

- **FMT_MSA.1/Signature attributes** and **FMT_SMF.1/Modification of signature** attributes, as it allows the signer to select the signing certificate.
- **FMT_MSA.3/Signatory's certificate import**, **FMT_MSA.1/Signatory's certificate**, **FDP_ITC.2/Signatory's certificate**, **FPT_TDC.1/Signatory's certificate** and **FMT_SMF.1/Signatory's certificate selection**, as it allows the signer, and only him, to select the signing certificate.

F.Signing certificate selection also contributes to the coverage of **FMT_SMR.1 Security roles** and **FIA_UID.2 User identification before** any action, as it provides separate interfaces to the signer and to the calling application.

F.Document viewer

This function covers the following SFR's:

- **FDP_IFF.1/Signature generation**, as it displays the document to be signed, according to its format.
- **FMT_MTD.1/Document format/viewer** association table and **FMT_SMF.1/Management of the** document format/viewer association table, as the association table is defined once for all in the TOE for text documents and because the **F.Document viewer** allows nobody to modify it.

F.Administration

This function implicitly covers **FMT_SMR.1 Security roles** and **FIA_UID.2 User identification before** any action, as it provides separate interfaces to the signer and to the calling application.

It also covers **FMT_MTD.1/Management of the** signature policies and **FMT_SMF.1/Management of the** signature policies, as it allows the calling application (exclusively) to specify the signature policy.

F.Signature policy application

This function covers the following SFR's:

- **FDP_IFF.1/Signatory's certificate import** and **FPT_TDC.1/Signatory's certificate**, as it filters the list of available certificates according to the signature policy's parameters.
- **FDP_IFF.1/Signature generation** and **FMT_MSA.3/Signature generation**, as it includes, in the data to be signed and in the XAdES signature, the signature's attributes.

F.Transfert to the SCDev

This function covers the following SFR's:

- **FDP_IFC.1/Signatory's certificate import** and **FDP_IFF.1/Signatory's certificate import**, as it retrieves from the SCDev the list of available certificates.
- **FDP_IFF.1/Electronic signature export**, as it checks that the SCDev-generated signature is indeed a PKCS #1 signature.
- **FDP_IFC.1/Signature generation** and **FDP_IFF.1/Signature generation**, as it requires the SCDev to compute the digital signature of the hash value with the private key corresponding to the signing certificate.
- **FMT_MSA.3/Electronic signature export**, **FMT_MSA.1/SCDev signature generation** status and **FMT_SMF.1/Getting SCDev's signature** generation status, as the success or the failure of the generation of the digital signature by the SCDev is managed by this function.

F.Signature attribute viewer

This function covers the following SFR's:

- **FDP_IFF.1/Signature generation**, as it displays to the user the signature's attributes.

The following table summarizes the SFR's coverage and demonstrates that the TSS completely covers the SFR's.

SFR	TSS
FCS_COP.1/Hash function	**F.Signature**
FDP_ETC.2/Electronic signature export	**F.Signature**
FDP_IFC.1/Document acceptance	**F.Semantic invariance check**
FDP_IFC.1/Electronic signature export	**F.Signature**
FDP_IFC.1/Signatory's certificate import	**F.Transfert to the SCDev**
FDP_IFC.1/Signature generation	**F.Transfert to the SCDev**
FDP_IFF.1/Document acceptance	**F.Semantic invariance check**
FDP_IFF.1/Electronic signature export	**F.Transfert to the SCDev**, **F.Signature**
FDP_IFF.1/Signatory's certificate import	**F.Transfert to the SCDev**, **F.Signature policy application**
FDP_IFF.1/Signature generation	**F.Transfert to the SCDev**, **F.Document viewer**, **F.Signature policy application**, **F.Signature attribute viewer**
FDP_ITC.1/Document acceptance	**F.Semantic invariance check**
FDP_ITC.1/Explicit signatory agreement	**F.Signature**
FDP_ITC.2/Signatory's certificate	**F.Signing certificate selection**
FDP_ROL.2/Abort of the signature process	**F.Signature**
FIA_UID.2 User identification before any action	**F.Signing certificate selection**, **F.Administration**
FMT_MSA.1/Document's semantics invariance status	**F.Semantic invariance check**
FMT_MSA.1/SCDev signature generation status	**F.Transfert to the SCDev**
FMT_MSA.1/Selected documents	**F.Signature**

SFR	TSS
FMT_MSA.1/Signatory's certificate	**F.Signing certificate selection**
FMT_MSA.1/Signature attributes	**F.Signing certificate selection**
FMT_MSA.3/Document's acceptance	**F.Semantic invariance check**
FMT_MSA.3/Electronic signature export	**F.Transfert to the SCDev**, **F.Signature**
FMT_MSA.3/Signatory's certificate import	**F.Signing certificate selection**
FMT_MSA.3/Signature generation	**F.Signature policy application**
FMT_MTD.1/Document format/viewer association table	**F.Document viewer**
FMT_MTD.1/Management of the signature policies	**F.Administration**
FMT_SMF.1/Getting SCDev's signature generation status	**F.Transfert to the SCDev**
FMT_SMF.1/Getting document's semantics invariance status	**F.Semantic invariance check**
FMT_SMF.1/Management of the document format/viewer association table	**F.Document viewer**
FMT_SMF.1/Management of the signature policies	**F.Administration**
FMT_SMF.1/Modification of signature attributes	**F.Signing certificate selection**
FMT_SMF.1/Selection of a list of documents	**F.Signature**
FMT_SMF.1/Signatory's certificate selection	**F.Signing certificate selection**
FMT_SMR.1 Security roles	**F.Signature**, **F.Signing certificate selection**, **F.Administration**
FPT_TDC.1/Signatory's certificate	**F.Signature policy application**, **F.Signing certificate selection**

8. ANNEXE: GLOSSARY, ABBREVIATIONS AND BIBLIOGRAPHY

8.1 Abbreviations

CA	Certification authority
CRL	*Certificate revocation list*
CSP	*Cryptographic service provider*
DN	*Distinguished name*
D2S	*Dictao signature server*
DTBS	*Data to be signed*
DTBSR	*Data to be signed representation*
HSM	*Hardware security module*
JVM	*Java virtual machine*
OCSP	*Online certificate status protocol*
OID	*Object identifier*
SAV	*Signature attribute viewer*
SCA	*Signature creation application*
SCDev	*Signature creation device*
SD	*Signer's document*
SDC	*Signer's document composer*
SDO	*Signed data object*
SDP	*SD presentation component*
SSCD	*Secure signature creation device*
ST	*Security target*
TOE	*Target of evaluation*
VPN	*Virtual private network*
XAdES	*XML advanced electronic signature*
XSL	*XML stylesheet*
XSLT	*XSL transformation*

8.2 Specific terms

Calling application	The web browser and web page which invoke the TOE.

8.3 Bibliography

Référence	Document
[AUTH]	*Authentification – Règles et recommendations concernant les mécanismes d'authentification.* Version 1.0 du 13 janvier 2010 Réf.: ANSSI/ACE
[CC3.1]	*Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model.* Ref. CCMB-2009-07-001 Common Criteria for Information Technology Security Evaluation, Part 2: Security functional requirements. Ref. CCMB-2009-07-002 *Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements.* Ref. CCMB-2009-07-003 Version 3.1, Revision 3 Final July 2009
[CRYPT]	*Mécanismes cryptographiques – Règles et recommandation concernant le choix et dimensionnement des mécanismes cryptographiques* *Ref.: ANSSI/ACE* *Version 1.20 du 26 janvier 2010*
[PP-ACSE]	*Profil de protection Application de création de signature électronique* *Ref.: PP-ACSE-CCv3.1 (~~DCSSI~~ ANSSI-PP-2008/05)* version ~~1.6~~ 1.7 ~~July 17th, 2008~~ March 2nd, 2011
[PP-ACSE-UK]	*Courtesy translation of the protection profile registered and certified* by the French Certification Body under the reference ~~DCSSI~~ANSSI-PP-2008*/05.*
[EXT_TS_101_862]	*Qualified Certificate Profile* *version 1.3.1* *March 2004* *ETSI* Standard
[PKCS #1]	*PKCS #1 – RSA Cryptography Standard* *Version 2.1* *June* 2002 RSA Laboratories
[QUA_STD]	*Référentiel general de sécurité, Processus de qualification d'un produit de sécurité – niveau standard* *Version 1.2* *n°* /SGDN/DCSSI/SDR
[XAdES]	*XML* Advanced Electronic Signatures *ref. ETSI* TS 101 903 versions 1.3.2 ~~et 14.1~~

9. ANNEXE: SUPPORTED SUBSET OF THE HTML

This chapter specifies the subset of the HTML that is supported by **AdSigner 5**. That subset implicitly defines:

- The set of semantically stable HTML documents: *AdSigner 5* will only allow to sign HTML document whose contents (mark-ups) that belong to this subset.
- The set of viewable documents: *AdSigner 5* will only display HTML document whose contents (mark-ups) that belong to this subset.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="table">
    <xs:complexType>
      <xs:sequence maxOccurs="unbounded">
        <xs:element ref="tr"/>
      </xs:sequence>
      <xs:attribute name="width" type="xs:int" use="required"/>
      <xs:attribute name="bgcolor" type="color" use="optional"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="tr">
    <xs:complexType>
      <xs:sequence minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="td"/>
      </xs:sequence>
      <xs:attribute name="height" type="xs:int" use="required"/>
      <xs:attribute name="bgcolor" type="color" use="optional"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="td">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="Inline">
          <xs:attribute name="width" type="xs:int" use="required"/>
          <xs:attribute name="colspan" type="xs:int" use="optional"/>
          <xs:attribute name="bgcolor" type="color" use="optional"/>
          <xs:attribute name="align" type="aligntype" use="optional"/>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="b" type="Inline"/>
  <xs:element name="i" type="Inline"/>
  <xs:element name="u" type="Inline"/>
  <xs:element name="font">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="Inline">
          <xs:attribute name="color" use="optional"/>
          <xs:attribute name="face">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:pattern value="Arial"/>
                <xs:pattern value="Verdana"/>
                <xs:pattern value="Courier New"/>
```

```
                <xs:pattern value="Times New Roman"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:group name="fontstyle">
    <xs:choice>
      <xs:element ref="b"/>
      <xs:element ref="u"/>
      <xs:element ref="i"/>
      <xs:element ref="font"/>
    </xs:choice>
  </xs:group>
  <xs:complexType name="Inline" mixed="true">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:group ref="fontstyle"/>
    </xs:choice>
  </xs:complexType>
  <xs:simpleType name="color">
    <xs:restriction base="xs:string">
      <xs:pattern value="#[0-9A-Fa-f]{6}"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="aligntype">
    <xs:restriction base="xs:string">
      <xs:pattern value="center"/>
      <xs:pattern value="right"/>
      <xs:pattern value="left"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

10. ANNEXE : REFERENCE TABLES

ASSETS AND SUBJECTS

OBJECTIVES

ORGANISATIONAL SECURITY POLICIES

ASSUMPTIONS

SECURITY FUNCTIONAL REQUIREMENTS

TOE SECURITY FUNCTIONS