# MetaCRYPT-API

# Common Criteria Security Target

| | |
|---|---|
| Version | 1.5 |
| Document date: | Mayl 06th 2015 |
| Doc. Ref: | EVALCC-MCRYPT-ST-01 |

# Status

| | |
|---|---|
| **Drafting** | BLAD Christophe |
| **Validation** | KAHOUL Vincent |
| **Classification** | company limited |
| **State of document** | Valid |
| **Current version** | 1.5 |
| **Reference** | EVALCC-MCRYPT-ST-01 |
| **Applicable product version** | 1.2.1 |

# Distribution

| Name | Company |
|---|---|
| Certifiers | ANSSI |
| Evaluators | OPPIDA |
| | |

# Revision history

| Date | Version | Comments |
|---|---|---|
| 2015/03/12 | 1.0 | First official release |
| 2015/04/13 | 1.1 | Modification of TOE description according to version 1.2.0 of TOE |
| 2015/04/16 | 1.2 | Addition of details in the A.HOST assumption |
| 2015/05/06 | 1.3 | Correction of anomalies detected by the evaluator described in RTI ASE V1.0 (dated of 17th April) |
| 2015/06/11 | 1.4 | Change Cryptographic Creation Device in TOE environment to "Gemalto IDPrime MD 840" |
| 2015/09/15 | 1.5 | JAVA update requirement<br>Update common libraries |

# Contents

# 1 ST Introduction

## 1.1 ST identification

| | |
|---|---|
| Title | Common Criteria Security Target – MetaCRYPT-API |
| Version | 1.5 |
| Auteur(s) | Bull SAS |
| Date | Mayl 06th 2015 |

The present document is the security target of MetaCRYPT-API v1.2.1.
It specifies the requirements in terms of security functionality and of security evaluation tasks to be fulfilled by the product.
It also summarized how the product complies with these requirements.

## 1.2 TOE identification

| | |
|---|---|
| TOE Developer | Bull SAS |
| Product name | MetaCRYPT-API |
| Version | 1.2.1 |

## 1.3 TOE overview

### 1.3.1 TOE type

MetaCRYPT-API is a java library offering cryptographic services to applications with an application programming interface (API).

### 1.3.2 Usage and major security features of the TOE

MetaCRYPT-API offers to java applications encryption and decryption capabilities. It supports CMS and XML encryption.
The following services are available:

- Document encryption: document encryption is possible with a secret key explicitly provided to the library[1] (encrypted data mode) or with a secret key generated during the operation and then encrypted with public keys extracted from certificates of declared document recipients (enveloped data mode).
- Document decryption: document decryption is possible with a secret key explicitly provided to the library or by the decryption of the secret key included in the document using private keys of the declared document recipients.
- Utilities: few additional services are also offers by the library: extraction of public data of encrypted documents, definition of an encryption policy, library configuration.

The two main modes are summarized in the following figures:
In enveloped data mode, MetaCRYPT generated a secret key for the document to be encrypted and then encrypt that secret key with the public keys of all the document recipients.

[1] If no key is provided by the calling application, the library generates itself the secret key.

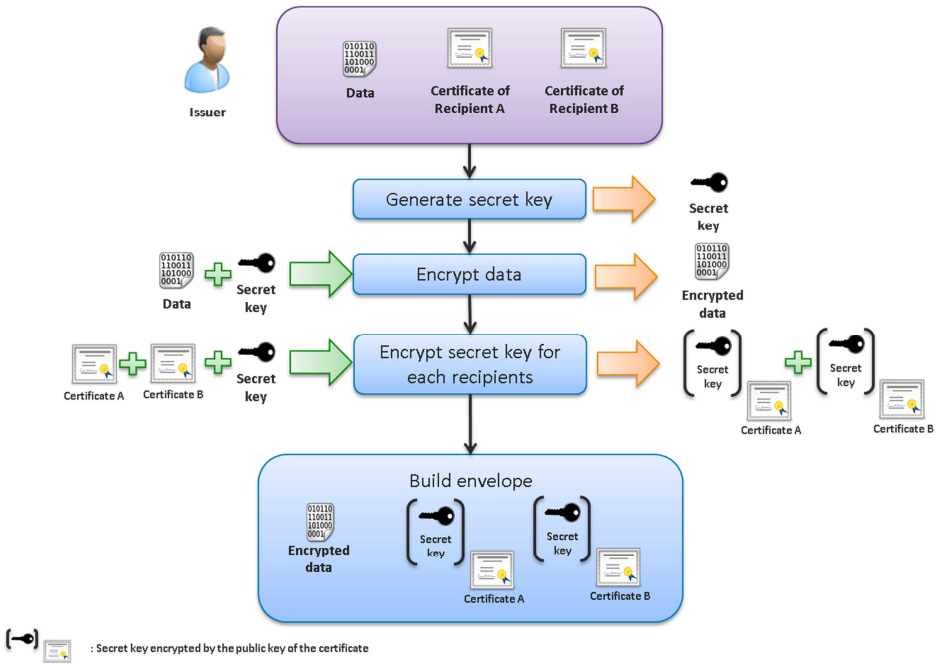


Figure 1: Enveloped data encryption operation

At the reception of the encrypted document, the recipients can decrypt the secret key and the decrypt the document.

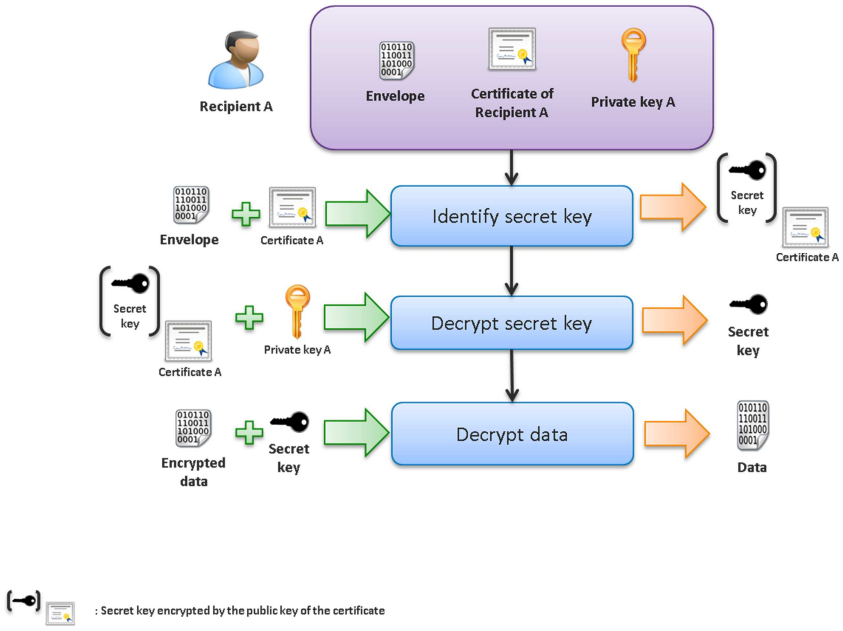


Figure 2: Enveloped data decryption operation

In encrypted data mode, the issuer and the recipients must share the secret key by other means. The secret key is not included within the encrypted document.

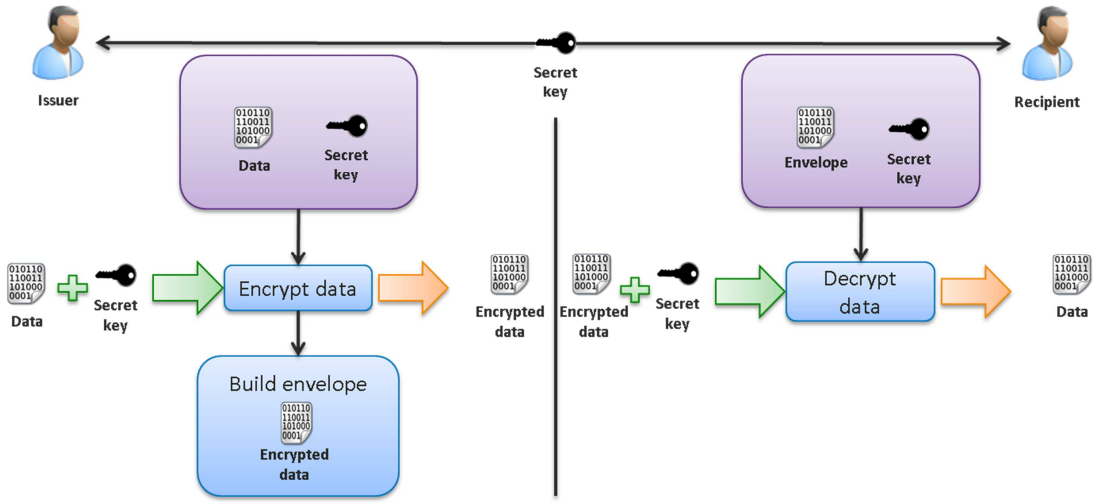


Figure 3: Encrypted data mode

The following formats of encrypted data are supported for both encryption and decryption functions:

- CMS: data encoded in ASN1 as defined in RFC5652 standards [CMS]. Enveloped-data, Encrypted-data and Authenticated-enveloped-data (RFC5083) formats are supported.
- XMLenc: data in XML as defined in W3C standard [XMLenc]. Enveloped-data and Encrypted-data formats are supported.

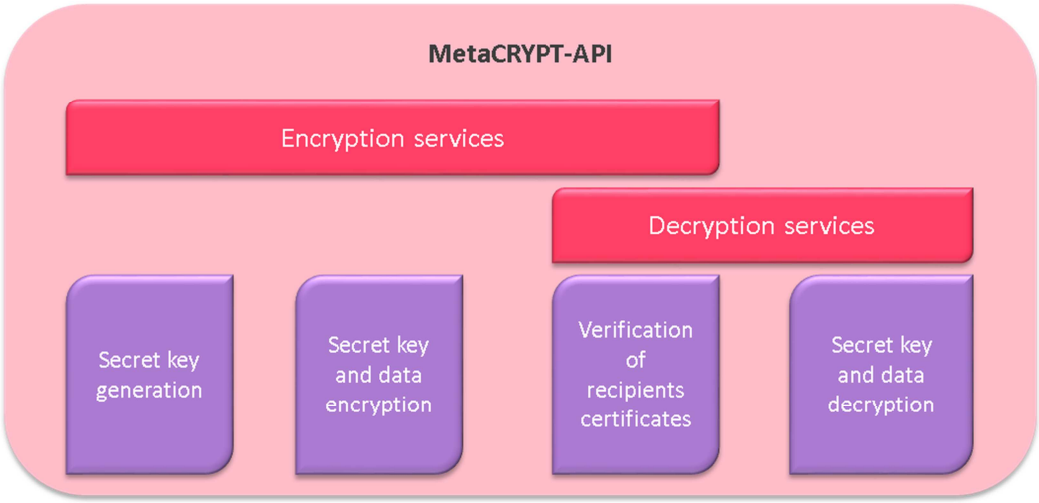


Figure 4 TOE functional architecture

Encryption and decryption operations can rely on an *encryption policy*. Encryption policy is optional for the decryption. This policy defines constraints that must be satisfied to allow the encryption or the decryption operation.
The encryption policy is referenced by a unique OID. It includes the following parameters:

- The policy OID (mandatory)
- The name of the policy (mandatory)
- The description of the policy (optional)
- Allowed symmetric algorithms (mandatory)
- Allowed asymmetric algorithms (mandatory)
    - Set the minimum key size (optional)
- Trusted CA certificates to be used for the recipients certificates validation (mandatory)
- Accepted values for the *certificatePolicies* extension of the recipients certificates (mandatory)

- Accepted values for the *keyUsage* extension of the recipients certificates (mandatory)
- Status checking of the recipients certificates before encryption (mandatory)
- Status checking of the recipients certificates before decryption (mandatory)

## 1.4 TOE description

MetaCRYPT-API TOE is a set of java binaries (.jar files). All jar files are signed, allowing users to check their integrity.
MetaCRYPT TOE is composed of 3 kinds of files:

1. The main library: metacrypt-api.jar
2. external libraries developed by Bull:
    - com.bull.security.common.checker.certificate.jar and com.bull.security.common.checker.jar for the certificates validation
    - com.bull.security.common.jaxb.adapters.jar for XML parsing
    - com.bull.security.common.retriever.key.jar graphical interface for retrieving certificates and private key
    - com.bull.security.common.server.connection.jar for the management of connections with external servers
    - com.bull.security.common.server.connection.ldap.jar extension of the server.connection.jar for LDAP support
    - com.bull.security.common.viewer.x509.jar for view certificate and CRL
    - com.bull.security.common.filechooser.jar for retrieving data (certificates, CRLs, encryption policies)
    - common.jar for common tools
    - hsm_tools.jar for PKCS#11 support
3. external third-party libraries:
    - iaik_cms.jar for the support of CMS documents
    - iaik_hlapi.jar for the API of the IAIK library
    - iaik_jce_full.jar for the cryptographic primitives
    - iaik_xsect.jar for the support of XMLEnc,
    - iaik_eccelerate.jar for the support of ECC
    - iaik_eccelerate_cms.jar for the support of ECC with iaik_cms.jar
    - iaik_eccelerate_ssl.jar for the support of ECC with iaik_ssl.jar
    - iaikPkcs11Wrapper.jar, iaikPkcs11Provider.jar for the communication with PKCS#11 tokens
    - iaik_ssl.jar for SSL connection
    - serializer.jar, xalan.jar, xml-apis.jar for the management of XML files
    - slf4j-api-1.6.1.jar, slf4j-log4j12-1.6.1.jar and log4j.jar for the log generation
    - w3c_http.jar for the HTTP connexions
    - unbounded-ldapskd-se.jar for LDAP support

The evaluated versions of the libraries are:

| **Name** | **Developer** | **version** |
|---|---|---|
| metacrypt-api.jar | Bull SAS | 1.2.1 |
| com.bull.security.common.checker.certificate.jar | Bull SAS | 1.37.1 |
| com.bull.security.common.checker.jar | Bull SAS | 1.37.1 |
| com.bull.security.common.jaxb.adapters.jar | Bull SAS | 1.37.1 |
| com.bull.security.common.retriever.key.jar | Bull SAS | 1.37.1 |
| com.bull.security.common.server.connection.jar | Bull SAS | 1.37.1 |
| com.bull.security.common.server.connection.ldap.jar | Bull SAS | 1.37.1 |
| com.bull.security.common.viewer.x509.jar | Bull SAS | 1.37.1 |
| com.bull.security.common.filechooser.jar | Bull SAS | 1.37.1 |
| common.jar | Bull SAS | 1.37.1 |
| hsm_tools.jar | Bull SAS | 1.37.1 |
| iaik_cms.jar | IAIK | 5.0 |
| iaik_hlapi.jar | IAIK | 1.1 |

| Name | Developer | version |
|---|---|---|
| iaik_jce_full.jar | IAIK | 5.24 |
| iaik_xsect.jar | IAIK | 1.191 |
| iaik_eccelerate.jar | IAIK | 2.5 |
| iaik_eccelerate_cms.jar | IAIK | 2.5 |
| iaik_eccelerate_ssl.jar | IAIK | 2.5 |
| iaikPkcs11Wrapper.jar | IAIK | 1.4 |
| iaikPkcs11Provider.jar | IAIK | 1.3 |
| iaik_ssl.jar | IAIK | 5.0 |
| serializer.jar | Apache | 2.7.1 |
| xalan.jar | Apache | 2.7.1 |
| xml-apis.jar | Apache | 2.0.2 |
| slf4j-api-1.6.1.jar | SLF4J | 1.6.1 |
| slf4j-log4j12-1.6.1.jar | SLF4J | 1.6.1 |
| log4j.jar | Apache | 1.2.14 |
| w3c_http.jar | IAIK | 1.0 |
| unbounded-ldapskd-me | Unbounded | 2.3.8 |

## 1.5 Required non-TOE hardware/software/firmware

As a java library, MetaCRYPT-API does not have any requirements for the underlying operating system. It can be used on Windows 7, 8 and 8.1 32/64 bits or any Linux 32/64 bits platform (i.e RHEL, Suse …).
Minimum version for Oracle Java Runtime Environment is 7 update 75 or 8 update 45.

**The platform for the evaluation is:**

- **Microsoft Windows7 Enterprise 64bits**
- **Oracle Java Runtime Environment 8 update 45, 32bits**

The certificates and the private keys can be stored in several device types:

- CCDev (Cryptographic Creation Device): smartcard or hardware security module (HSM). Access to the device relies on a PKCS#11 interface.
- Software key store: certificates and keys are stored in a PKCS#12 file protected by a password or in a java key store (JKS)
- Windows key store: certificates and keys are stored in the Microsoft Windows key store. Access to the store relies on the MSCAPI (Microsoft Cryptographic API) interface.

**For the evaluation, certificates and keys are stored:**

- **in PKCS#12 files protected by a password, or**
- **in a Gemalto IDPrime MD 840 token**

Certificates can also be downloaded from a LDAP (with SSL or not) directory or from a HTTP/HTTPS server.

**For the evaluation, the following methods are used:**

- **certificates are downloaded from an OpenLDAP 2.4.39 server with LDAP/LDAPS protocol ;**
- **certificates are downloaded from an Tomcat server with HTTP/HTTPS protocol ;**
- **certificates are recovering from the local file system**

The certificate validation relies on OCSP responder or on a local repository of CA certificates and CRLs.

**For the evaluation, the following methods are used:**

- **certificate validation relies on a Bull MetaPKI v9.5.8 OCSP responder;**
- **certificate validation relies on CRLs store**

# 2 Conformance claims

## 2.1 Common Criteria conformance claim

This security target claims a strict conformance with the Common Criteria version 3.1 revision 4 [CC].

## 2.2 Protection profile conformance claim

This security target does not claim any conformance with a protection profile.

## 2.3 Package conformance claim

The evaluation assurance level claimed by this security target is EAL3 augmented with ALC_FLR.3 and AVA_VAN.3.

# 3 Security problem definition

This section identifies the security aspects of the environment in which the TOE is operated.

## 3.1 Assumptions

**A.HOST**
It is assumed that the platform hosting the TOE is managed by a competent and trusted administrator. Security safeguards are implemented:

- The access to privileged accounts is protected,
- The platform is protected against malware,
- All software update have been installed.

**A.KEYSTORE**
It is assumed that the keystore storing the private keys assure their confidentiality and integrity.

## 3.2 Threats

In the operational environment of the TOE, as described by the assumptions, no threat agents have been identified. All the security requirements for the TOE are derived from OSPs.

## 3.3 Organizational security policies

**P.ENCRYPTION_SERVICES**
Encryption and decryption services must be available to applications through API. Supported formats are: CMS [CMS] and XMLEnc [XMLEnc].

**P.ACCESS**
Only a stated list of recipients must be able to decrypt a document.

**P.CERTIFICATE_VERIFICATION**
Before encryption and decryption, recipient's certificates must be verified, if required by the encryption policy. The encryption policy defines if the verification only deals with the validity of the certificate or if it deals with the status of the certificate too.

# 4 Security objectives

The security objectives reflect the stated intent and are suitable to counter all identified threats and to cover all identified organisational security policies and assumptions.

## 4.1 Security objectives for the TOE

**OT.ENCRYPTION_SERVICES**
The TOE shall offer to applications an API for encryption and decryption. Supported formats are: CMS [CMS] and XMLEnc [XMLEnc].

**OT.ACCESS**
The TOE shall allow only the stated recipients of an encrypted document to decrypt it. If the encryption secret key is included within the encrypted document, it has to be extracted and decrypted by the recipients. If not, the calling application must explicitly provide decryption key to the TOE.

**OT.CERTIFICATE_VERIFICATION**
Before to encrypt or the decrypt a document and if required by the encryption policy, the TOE shall verify the certificate of the stated recipients. The list of the verification is defined by the encryption policy.

## 4.2 Security objectives for the TOE environment

**OE.HOST**
The platform hosting the TOE shall be managed by a competent and trusted administrator. Security safeguards shall be implemented:

- The access to privileged accounts shall be protected,
- The platform shall be protected against malware,
- All software update shall be installed.

**OE.KEYSTORE**
Safeguards shall be implemented to protect the keystore in integrity and confidentiality.

## 4.3 Security objectives rationale

### 4.3.1 Assumptions coverage

**A.HOST:** *The assumption is directly covered by the objective [OE.HOST]*

**A.KEYSTORE:** *The assumption is directly covered by the objective [OE.KEYSTORE]*

### 4.3.2 Threats coverage

No threats are defined in this security target.

### 4.3.3 OSP coverage

**P.ENCRYPTION_SERVICES:** *This OSP is directly covered by [OT.ENCRYPTION_SERVICES].*

**P.ACCESS:** *This OSP is directly covered by [OT.ACCESS].*

**P.CERTIFICATE_VERIFICATION:** *This OSP is directly covered by [OT.CERTIFICATE_VERIFICATION].*

# 5 Security requirements

## 5.1 Subjects, operation, objects

### 5.1.1 Subjects

Table 1 : Subjects list

| Subjects | Description | Security attributes |
|---|---|---|
| Calling application | The calling application is the application that uses the services offered by the TOE. | - |
| Document recipient | The document recipient is the person for who the document is encrypted. | - |

### 5.1.2 Objects

Table 2 : Objects list

| Objects | Description | Security attributes |
|---|---|---|
| Clear document | The document to be encrypted. All document formats (incl. XML) can be encrypted. | - |
| Encrypted document | The encrypted document is the document after the encryption process, or the document provided to the API in order to be decrypted. | Recipients list |
| Secret key | Secret key used to encrypt and decrypt documents.<br>The secret key is either explicitly provided by the calling application or either included within the encrypted document. | - |
| Recipient certificate | X509 certificate of a document recipient | Public key, Public key exponent<br>Issuer<br>Certification chain<br>Key usage<br>Revocation status<br>Validity period |
| Recipient private key | The private key of the recipient (associated with the public key included in the recipient certificate) is used to decrypt the encrypted secret key when the secret key is present within the encrypted document. | - |

### 5.1.3 Operations

Table 3 : Operations list

| Subjects | Operations | Objects |
|---|---|---|
| Calling application | The calling application encrypts the clear document and generates an encrypted document using the encryption secret key.<br>The secret key is integrated within the encrypted document (encrypted envelope mode) or not (encrypted data mode). | Clear document<br>Secret key<br>Encrypted document |
| Calling application | The calling application decrypts the | Encrypted document |

| | | |
|---|---|---|
| | encrypted document. The secret key is explicitly provided by the calling application (encrypted data mode) or is extracted from the document (encrypted envelope mode). | Secret key |
| Document recipient | In encrypted envelope mode, the encrypted secret key is extracted from the document and is decrypted by the recipient using its private key. | Encrypted document<br>Recipient private key<br>Secret key |

## 5.2 Explicit security requirements

This security target does not include any explicit security requirements. All the security requirements are strictly conform with the Common Criteria Part 2 and Part 3.

## 5.3 TOE security functional requirements

### 5.3.1 Security functional requirements (SFRs) summary

| Components | |
|---|---|
| **FCS_COP.1/Document encryption** | Document encryption |
| **FCS_COP.1/Secret Key encryption** | Secret Key encryption |
| **FCS_CKM.1** | Secret Key generation |
| **FCS_CKM.4** | Private and secret keys destruction |
| **FDP_ITC.1** | Secret Key extraction |
| **FDP_ACC.1/Secret key protection** | Access control to the secret key |
| **FDP_ACF.1/Secret key protection** | Access control to the secret key |
| **FDP_ITC.2** | Certificates validation |
| **FPT_TDC.1** | Certificates support |
| **FDP_ACC.1/Certificates validation** | Certificates validation rules |
| **FDP_ACF.1/Certificates validation** | Certificates validation rules |

All the security functional components are strictly conform with Common Criteria part 2 [CC].

### 5.3.2 List of SFRs

Assignments and selections are between brackets []. Refinements are in *italics*. Iterations are identified by the "/" sign after the SFR acronym; for example *FCS_COP.1/Document encryption* and *FCS_COP.1/Secret Key encryption*.

**FCS_COP.1/Document encryption: Document encryption**
FCS_COP.1.1
The TSF shall perform [encryption and decryption] in accordance with a specified cryptographic algorithm [see the following list] and cryptographic key sizes [128 bits (DESede), 128 bits (AES128), 192 bits (AES192), 256 bits (AES256)] that meet the following: [see the following list].

Supported algorithms for CMS encryption are:
- DESede/CBC/PKCS5Padding;
- AES128/CBC/PKCS5Padding;

- AES192/CBC/PKCS5Padding;
- AES256/CBC/PKCS5Padding;
- AES128/GCM/NoPadding;
- AES192/GCM/NoPadding;
- AES256/GCM/NoPadding;

Supported algorithms for the XML encryption are:
- DESede/CBC/ISO10126Padding;
- AES128/CBC/ISO10126Padding;
- AES192/CBC/ISO10126Padding;
- AES256/CBC/ISO10126Padding;

**FCS_COP.1/Secret Key encryption: Secret Key encryption**
FCS_COP.1.1
The TSF shall perform [Secret Key encryption] in accordance with a specified cryptographic algorithm [see the following list] and cryptographic key sizes [2048 bits or 4096 bits] that meet the following: [see the following list].

Supported algorithms for the secret key encryption are:
- RSA/ECB/PKCS1Padding
- RSA/ECB/OAEP

**FCS_CKM.1: Secret Key generation**
FCS_CKM.1.1
The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [assignment: cryptographic key generation algorithm] and specified cryptographic key sizes [128 bits (DESede), 128 bits (AES128), 192 bits (AES192), 256 bits (AES256)] that meet the following: [assignment: list of standards].

The key generation types in software mode:

| | |
|---|---|
| AES | Key generation method for the Advanced Encryption Standard (AES, Rijndael) block cipher. |
| AES192 | Key generation method for creating 192bit keys for the Advanced Encryption Standard (AES, Rijndael) block cipher. |
| AES256 | Key generation method for creating 256bit keys for the Advanced Encryption Standard (AES, Rijndael) block cipher. |
| DESede | Key generation method for the DES(ede) cipher which is defined by NIST in FIPS PUB 46-1 and FIPS PUB 46-2 |

The key generation type for PKCS#11 mode are using the following specific template:

| Template attribute identifier | Enveloped-data cipher mode | Encrypted-data cipher mode (key conservation to send to recipient) |
|---|---|---|
| CKA_SENSITIVE | True | True |
| CKA_EXTRACTABLE | True | True |
| CKA_UNWRAP | False | False |
| CKA_PRIVATE | True | True |
| CKA_WRAP | False | False |
| CKA_TOKEN | False | True |
| CKA_KEY_TYPE | CKK_AES or CKK_DES3 | CKK_AES or CKK_DES3 |
| CKA_VALUE_LEN | Only defined for AES | Only defined for AES |
| CKA_CLASS | CKO_SECRET_KEY | CKO_SECRET_KEY |

**FCS_CKM.4: Private and secret keys destruction**
FCS_CKM.4.1
The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [Java mechanisms] that meets the following: [none].
*Both secret keys used for the encryption of the document and private keys used for the decryption of the secret keys must be securely deleted.*

**FDP_ITC.1: Secret Key extraction**
FDP_ITC.1.1
The TSF shall enforce the [secret key protection policy] when importing user data, controlled under the SFP, from outside of the TOE.
FDP_ITC.1.2
The TSF shall ignore any security attributes associated with the user data when imported from outside the TOE.
FDP_ITC.1.3
The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [only the stated recipients of the document can extract the secret keys used to encrypt the document].
*User data is the secret key used for the document encryption.*

**FDP_ACC.1/Secret key protection: Access control to the secret key**
FDP_ACC.1.1
The TSF shall enforce the [secret key protection policy] on [encrypted documents].

**FDP_ACF.1/Secret key protection: Access control to the secret key**
FDP_ACF.1.1
The TSF shall enforce the [secret key protection policy] to objects based on the following: [encrypted documents (recipients list)].
FDP_ACF.1.2
The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [only stated recipients of the document can decrypt the secret key].
FDP_ACF.1.3
The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [the document recipient must possess the private key permitting to decrypt the secret key].
FDP_ACF.1.4
The TSF shall explicitly deny access of subjects to objects based on the [incapacity to decrypt the secret key].

**FDP_ITC.2: Certificates validation**
FDP_ITC.2.1
The TSF shall enforce the [certificates validation policy] when importing user data, controlled under the SFP, from outside of the TOE.
FDP_ITC.2.2
The TSF shall use the security attributes associated with the imported user data.
FDP_ITC.2.3
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
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
The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [
The TOE shall:

- Verify that the recipient certificate has a certification chain including a trusted CA certificate
- Examine if the recipient certificate has a *keyUsage* extension conform to the encryption policy

- Determine if the recipient certificate has a public key with sufficient length (optional, set in the encryption policy)
- Check the validity of the signature of the certificate with its associated issuer public key
- If required by the encryption policy, check the recipient certificate revocation status.
- Determine if the recipient certificate is inside its validity period at a specified date
- Determine if the recipient certificate has a RSA public key exponent strictly higher than 65536
- Determine if the certification chain has a *CertificatePolicies* extension conform to the encryption policy

]

**FPT_TDC.1: Certificates support**
FPT_TDC.1.1
The TSF shall provide the capability to consistently interpret [x509 certificates] when shared between the TSF and another trusted IT product.
FPT_TDC.1.2
The TSF shall use [the certificates validation policy] when interpreting the TSF data from another trusted IT product.

**FDP_ACC.1/Certificates validation: Certificates validation rules**
FDP_ACC.1.1
The TSF shall enforce the [certificates validation policy] on [recipients' certificates].

**FDP_ACF.1/Certificates validation: Certificates validation rules**
FDP_ACF.1.1
The TSF shall enforce the [certificates validation policy] to objects based on the following: [recipients certificates attributes].
FDP_ACF.1.2
The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [

| **Certificate attributes** | **Rules** |
|---|---|
| Public key | The public key must have a sufficient key length (set in the encryption policy). |
| Public key exponent | A RSA public key exponent must be strictly higher than 65536. |
| Issuer | The signature of the certificate by the Issuer must be valid. |
| Certification chain | The certification chain must include a trusted CA certificate. |
| Key usage | The Key usage must conform the one specified in the encryption policy. |
| Validity period | The current date must be before the end of the certificate validity. |
| Revocation status | If required by the encryption policy, the certificate must not be revoked. |
| Certificate Policies | The certification chain must have a *CertificatePolicies* extension conform to the encryption policy |

].
FDP_ACF.1.3
The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [see certificates validation rules].
FDP_ACF.1.4
The TSF shall explicitly deny access of subjects to objects based on the [non satisfaction of the certificates validation rules].

## 5.4 TOE security assurance requirements

The claimed evaluation assurance level is EAL3 augmented with ALC_FLR.3 and AVA_VAN.3. All the security assurance components are extracted from Common Criteria part 3 [CC].

| Components | | Title |
|---|---|---|
| **ADV : Development** | | |
| | ADV_ARC.1 | Security architecture description |
| | ADV_FSP.3 | Functional specification with complete summary |
| | ADV_TDS.2 | Architectural design |
| **AGD : Guidance documents** | | |
| | AGD_OPE.1 | Operational user guidance |
| | AGD_PRE.1 | Preparative procedures |
| **ALC : Life-cycle support** | | |
| | ALC_CMC.3 | Authorisation controls |
| | ALC_CMS.3 | Implementation representation CM coverage |
| | ALC_DEL.1 | Delivery procedures |
| | ALC_DVS.1 | Identification of security measures |
| | ALC_LCD.1 | Developer defined life-cycle model |
| | ALC_FLR.3 | Systematic flaw remediation |
| **ASE : Security Target evaluation** | | |
| | ASE_CCL.1 | Conformance claims |
| | ASE_ECD.1 | Extended components definition |
| | ASE_INT.1 | ST introduction |
| | ASE_OBJ.2 | Security objectives |
| | ASE_REQ.2 | Derived security requirements |
| | ASE_SPD.1 | Security problem definition |
| | ASE_TSS.1 | TOE summary specification |
| **ATE : Tests** | | |
| | ATE_COV.2 | Analysis of coverage |
| | ATE_DPT.1 | Testing: basic design |
| | ATE_FUN.1 | Functional testing |
| | ATE_IND.2 | Independent testing - sample |
| **AVA : Vulnerability assessment** | | |
| | AVA_VAN.3 | Focused vulnerability analysis |

## 5.5 Security requirements rationale

### 5.5.1 Security objectives coverage

**OT.ENCRYPTION_SERVICES:** *The objective is mainly refined into the [FCS_COP.1/Document encryption] component that requires the encryption and decryption features. That component requires the following dependent components: [FCS_CKM.1] for the generation of the encryption key before the encryption operation [FDP_ITC.1] for the extraction of the encryption key during the decryption operation, [FCS_CKM.4] for the deletion of the encryption key after use.*

**OT.ACCESS:** *The objective is refined into [FCS_COP.1/Secret Key encryption] for the decryption operation. If the encryption secret key is integrated within the encrypted document, [FDP_ACC.1/Secret key protection] and [FDP_ACF.1/Secret key protection] components that restrict the possibility to decrypt the encryption key only to a list of stated recipients are also required.*

**OT.CERTIFICATE_VERIFICATION:** *The objective is refined into [FDP_ITC.2] component that requires the verification of the certificate status, into [FPT_TDC.1] for the extraction of the necessary data from the certificate and into [FDP_ACC.1/Certificates validation] and [FDP_ACF.1/Certificates validation] for the specification of the certificates acceptation rules.*

### 5.5.2 Dependencies

Table 4: SFRs dependencies

| Components | Required dependencies | Satisfied dependencies | Comments for non-satisfaction |
|---|---|---|---|
| **FCS_COP.1/Document encryption** | (FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1) and FCS_CKM.4 | [FCS_CKM.1] for the generation of the secret key during the document encryption, [FDP_ITC.1] for the extraction of the secret key during the document decryption, [FCS_CKM.4] for the deletion of the secret keys after use | |
| **FCS_COP.1/Secret Key encryption** | (FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1) and FCS_CKM.4 | [FDP_ITC.2] for the import of the recipients certificates, [FCS_CKM.4] for the deletion of the secret and private keys after use | |
| **FCS_CKM.1** | (FCS_CKM.2 or FCS_COP.1) and FCS_CKM.4 | [FCS_COP.1/Document encryption] for the encryption operations, [FCS_CKM.4] for the deletion of the secret keys after use | |
| **FCS_CKM.4** | (FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1) | [FCS_CKM.1] for the secret key generation before encryption and [FDP_ITC.1] for the extraction of the secret key before decryption | |
| **FDP_ITC.1** | (FDP_ACC.1 or FDP_IFC.1) and FMT_MSA.3 | [FDP_ACC.1/Secret key protection] to limit the capability of decryption of the secret key only for stated recipients of the document | FMT_MSA.3 component is not required because decryption is not possible if none recipient is defined. |
| **FDP_ACC.1/Secret key protection** | FDP_ACF.1 | [FDP_ACF.1/Secret key protection] for the policy specification | |
| **FDP_ACF.1/Secret key protection** | FDP_ACC.1 and FDP_MSA.3 | [FDP_ACC.1/Secret key protection] to limit the capability of decryption of the secret key only for stated recipients of the document | FMT_MSA.3 component is not required because decryption is not possible if none recipient is defined. |

| Components | Required dependencies | Satisfied dependencies | Comments for non-satisfaction |
|---|---|---|---|
| **FDP_ITC.2** | (FDP_ACC.1 or FDP_IFC.1) and (FTP_ITC.1 or FTP_TRP.1) and FPT_TDC.1 | [FDP_ACC.1/Certificates validation] for the verification of the recipient certificate before encryption, [FDP_ITC.1] for the extraction of the secret key before decryption, [FPT_TDC.1] to extract the necessary information from x509 certificates | |
| **FPT_TDC.1** | - | - | |
| **FDP_ACC.1/Certificates validation** | FDP_ACF.1 | [FDP_ACF.1/Certificates validation] for the verifications specifications | |
| **FDP_ACF.1/Certificates validation** | FDP_ACC.1 and FDP_MSA.3 | [FDP_ACC.1/Certificates validation] for the verification of the recipient certificate before encryption | FMT_MSA.3 component is not required because decryption is not possible if none recipient is defined. |

Table 5: SARs dependencies

| Components | Required dependencies | Satisfied dependencies | Comments for non-satisfaction |
|---|---|---|---|
| **ADV_ARC.1** | ADV_FSP.1 and ADV_TDS.1 | ADV_FSP.3 and 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** | - | - | |
| **ALC_CMC.3** | ALC_CMS.1 and ALC_DVS.1 and ALC_LCD.1 | ALC_CMS.3 and ALC_DVS.1 and ALC_LCD.1 | |
| **ALC_CMS.3** | - | - | |
| **ALC_DEL.1** | - | - | |
| **ALC_DVS.1** | - | - | |
| **ALC_FLR.3** | - | - | |
| **ALC_LCD.1** | - | - | |
| **ASE_INT.1** | - | - | |
| **ASE_CCL.1** | ASE_INT.1 and ASE_ECD.1 and ASE_REQ.1 | ASE_INT.1 and ASE_ECD.1 and ASE_REQ.2 | |
| **ASE_SPD.1** | - | - | |
| **ASE_OBJ.2** | ASE_SPD.1 | ASE_SPD.1 | |
| **ASE_ECD.1** | - | - | |
| **ASE_REQ.2** | ASE_ECD.1 and ASE_OBJ.2 | ASE_ECD.1 and ASE_OBJ.2 | |
| **ASE_TSS.1** | ASE_INT.1 and ASE_REQ.1 and ADV_ARC.1 | ASE_INT.1 and ASE_REQ.2 and ADV_ARC.1 | |
| **ATE_COV.2** | ADV_FSP.2 and ATE_FUN.1 | ADV_FSP.3 and ATE_FUN.1 | |

| Components | Required dependencies | Satisfied dependencies | Comments for non-satisfaction |
|---|---|---|---|
| **ATE_DPT.1** | ADV_ARC.1 and ADV_TDS.2 and ATE_FUN.1 | ADV_ARC.1 and ADV_TDS.2 and ATE_FUN.1 | |
| **ATE_FUN.1** | ATE_COV.1 | ATE_COV.2 | |
| **ATE_IND.2** | ADV_FSP.2 and AGD_PRE.1 and AGD_OPE.1 and ATE_COV.1 and ATE_FUN.1 | ADV_FSP.3 and AGD_PRE.1 and AGD_OPE.1 and ATE_COV.2 and ATE_FUN.1 | |
| **AVA_VAN.3** | ADV_ARC.1 and ADV_FSP.4 and ADV_TDS.3 and ADV_IMP.1 and AGD_PRE.1 and AGD_OPE.1 and ATE_DPT.1 | ADV_ARC.1 and ADV_FSP.3 and ADV_TDS.2 and AGD_PRE.1 and AGD_OPE.1 and ATE_DPT.1 | The assurance package has been defined by the French certification scheme (ADV_FSP.4, ADV_TDS.3 and ADV_IMP.1 dependencies are not required) |

# 6 TOE Summary Specification

Table 1 : SFRs coverage

| SFR | Coverage |
|---|---|
| **FCS_COP.1/Document encryption** | MetaCRYPT offers to applications API for encryption and decryption in CMS or XMLENC format. Encryption algorithms can be configured through API parameters.<br>Supported algorithms for CMS encryption are :<br>- DESede/CBC/PKCS5Padding;<br>- AES128/CBC/PKCS5Padding;<br>- AES192/CBC/PKCS5Padding;<br>- AES256/CBC/PKCS5Padding;<br>- AES128/GCM/NoPadding;<br>- AES192/GCM/NoPadding;<br>- AES256/GCM/NoPadding;<br><br>Supported algorithms for the XML encryption are:<br>- DESede/CBC/ISO10126Padding;<br>- AES128/CBC/ISO10126Padding;<br>- AES192/CBC/ISO10126Padding;<br>- AES256/CBC/ISO10126Padding. |
| **FCS_COP.1/Secret Key encryption** | The secret key used for the document encryption is encrypted with each public key of the document recipients. |
| **FCS_CKM.1** | Before encryption, MetaCRYPT can generate the secret key that will be used to encrypt the document. |
| **FCS_CKM.4** | After use, MetaCRYPT deletes secret keys used for the encryption of the document and private keys used for the decryption of the secret keys. |
| **FDP_ITC.1, FDP_ACC.1/Secret key protection, FDP_ACF.1/Secret key protection** | Before decryption, the secret key used for the document encryption is decrypted only if the calling application is able to provide the private key associated to the public used to encrypt the secret key. |
| **FDP_ITC.2, FPT_TDC.1, FDP_ACC.1/Certificates validation, FDP_ACF.1/Certificates validation** | MetaCRYPT:<br>• Verifies that the recipients' certificates have the certification chain includes a trusted CA certificate.<br>• Checks if the recipient's certificate *keyUsage* are conform to the one specified in the encryption policy.<br>• If required by the encryption policy, checks if the recipients' certificates have a public key with sufficient length.<br>• Checks the validity of the recipients' certificates signatures with its issuer public key.<br>• If required by the encryption policy, checks if the recipients' certificates have not been revoked.<br>• Checks if the recipients' certificates are inside their validity period.<br>• Checks if the RSA public keys exponents are strictly higher than 65536.<br>• Checks if the certification chain has a *CertificatePolicies* extension conform to the encryption policy |

# 7 References


| | |
|---|---|
| [CC] | Common Criteria for Information Technology Security Evaluation, version 3.1 revision 4<br>• Part 1: Introduction and general model, ref. CCMB-2012-09-001<br>• Part 2: Security functional requirements, ref. CCMB-2012-09-002<br>• Part 3: Security assurance requirements, ref. CCMB-2012-09-003 |
| [XMLenc] | XML Encryption Syntax and Processing version 1.1 |
| [CMS] | Cryptographic Message Syntax (CMS), RFC 5652 |

# 8 Terms and acronyms

| | |
|---|---|
| CC | Common Criteria [CC] |
| OSP | Organisational Security Policy: organisational security policy statements or rules with which the TOE must comply |
| ST | Security Target |
| TOE | Target Of Evaluation: product or system target of the evaluation specified in the present document. |
| TSF | TOE Security Functions: part of the product or of the system to be evaluated that implements the security functional requirements |