

Open XML Open Packaging Conventions Low Assurance Security Target

Prepared for

Centro Criptológico Nacional

Versión 2.0 Final

Prepared by

Héctor Sánchez

Contributors

Alfonso Zorrilla

Ignacio Lago

José Luis de Prat

Microsoft[®]

Change Control

Date	Author	Version	Comments
14/02/2008	Alfonso Zorrilla Hector Sánchez Ignacio Lago José Luis de Prat	1.0	Initial release
26/11/2008	Alfonso Zorrilla Hector Sánchez Ignacio Lago José Luis de Prat	1.1	Addressing evaluation observation OPC-OR-001, improve the description of the TOE. Addressing evaluation observation OPC-OR-002, include crypto operational environment objectives.
02/03/2009	Alfonso Zorrilla Hector Sánchez Ignacio Lago José Luis de Prat	2.0	TSS Update aligned with white paper

Versions

Name	Approved Version	Role	Date
Alfonso Zorrilla Hector Sánchez Ignacio Lago José Luis de Prat	1.0	Microsoft Team	14/02/2008
Alfonso Zorrilla Hector Sánchez Ignacio Lago José Luis de Prat	1.1	Microsoft Team	26/11/2008
Alfonso Zorrilla Hector Sánchez Ignacio Lago José Luis de Prat	2.0	Microsoft Team	02/03/2009

Contents

Introduction.....	4
ST reference.....	4
TOE reference.....	4
References.....	4
TOE overview.....	4
TOE usage.....	4
TOE type.....	5
Non TOE Hardware and Software.....	6
TOE description.....	14
Physical scope.....	14
Logical scope.....	14
PP conformance claims.....	16
CC Conformance Claim.....	16
PP Claim, Package Claim.....	16
Security Objectives.....	17
Security Objectives for the Operational Environment.....	17
Security Requirements for the TOE.....	18
Functional Security Requirements.....	18
Assurance Security Requirements.....	19
TOE Summary Specification.....	25

INTRODUCTION

ST reference

- 1 **Title:** OpenXML Open Packaging Low Assurance Security Target
2 **Version:** 2.0
3 **Author:** Microsoft Ibérica
4 **Publication date:** March, the 2nd 2009

TOE reference

- 5 **Developer:** Microsoft Corporation
6 **TOE name:** Microsoft SDK for Open XML Formats, OpenXMLSDK.msi
7 **TOE version:** 1.0

References

- 8 [CLR] OpenXMLSDK.chm v 1.0.0531 - Class Library Reference
9 [SIO] System.IO.Packaging Namespace
 .NET Framework 3.5 Class Library
 msdn2.microsoft.com/en-uss/library/system.io.packaging.aspx
10 [CRY] <http://msdn.microsoft.com/en-us/library/system.security.cryptography.aspx>

TOE overview

TOE usage

- 11 The Office Open XML specifies the structure and functionality of a package in terms of a package model and a physical model.
- 12 The package model defines a package abstraction that holds a collection of parts. The parts are composed, processed, and persisted according to a set of rules. Parts can have relationships to other parts or external resources, and the package as a whole can have relationships to parts it contains or external resources. The package model specifies how the parts of a package are named and related. Parts have content types and are uniquely identified using the well-defined naming guidelines provided in this Standard.
- 13 The physical mapping defines the mapping of the components of the package model to the features of a specific physical format, namely a ZIP archive.

- 14 The Office Open XML also describes certain features that might be supported in a package, including core properties for package metadata, a thumbnail for graphical representation of a package, and digital signatures of package contents.
- 15 Packages are designed to accommodate extensions and support compatibility goals in a limited way. The versioning and extensibility mechanisms support compatibility between software systems while allowing package creators to make use of new or proprietary features.
- 16 The Office Open XML specifies requirements for package implementers, producers, and consumers.
- 17 The TOE is a set of libraries that implement the Office Open XML Open Packaging Conventions. These libraries are available as a “SDK for Open XML Formats”, provided by Microsoft.
- 18 The TOE, based on .NET technology, allows to create a valid Open Office XML document, to electronically sign it, and to compose a package according to the Open Packaging Conventions.
- 19 A number of security properties may be desired to be supported by an electronic document format specification. This TOE demonstrates that the specification of the Open XML Packaging Conventions allows to implement them in such a secure way that the TOE can ensure the integrity of the packaged document, albeit being distributed by an untrusted environment.
- 20 In particular, the TOE initially allows to
- Create an Open Office XML compliant document;
 - Electronically sign this document;
 - Package and ZIP it according to the Open XML Packaging Conventions.
- 21 Then, the TOE allows to reverse the process, by
- Unzipping the document and validating its compliance with respect to the Office Open XML specification;
 - Verifying the electronic signature of the signed Office Open XML document components, thus detecting any modification of the document.

TOE type

- 22 The TOE is a set of .NET libraries that implements the creation of Office Open XML compliant electronic documents, their electronic signature and packaging in accordance with the Office Open XML Open Packaging Conventions, and the reverse operation of unpackaging, electronic signature verification and Office Open XML validation.

- 23 The TOE may be used by any third party developer that may wish to add the TOE capabilities to their third party products.

Non TOE Hardware and Software

- 24 The TOE relies in .NET technology, and requires the Microsoft .NET Framework 3.0.
- 25 The .NET Framework in turn requires any of the following supported operating systems: Longhorn (Windows Code Name); Windows Server 2003 Service Pack 1; Windows Vista; Windows XP Service Pack 2. Each of these operating systems has a number of supported hardware configurations, none of which having any impact in the claimed security features of the TOE.
- 26 The TOE, the Microsoft SDK for Open XML Formats, OpenXMLSDK.msi, version 1, is a set of .NET libraries and APIs that allows to create packages and manipulate the files that make up the packages. In turn, these libraries rely on the cryptographic service providers of the underlying operating system.
- 27 The following figures represent a partial class diagram of the Open XML object model:

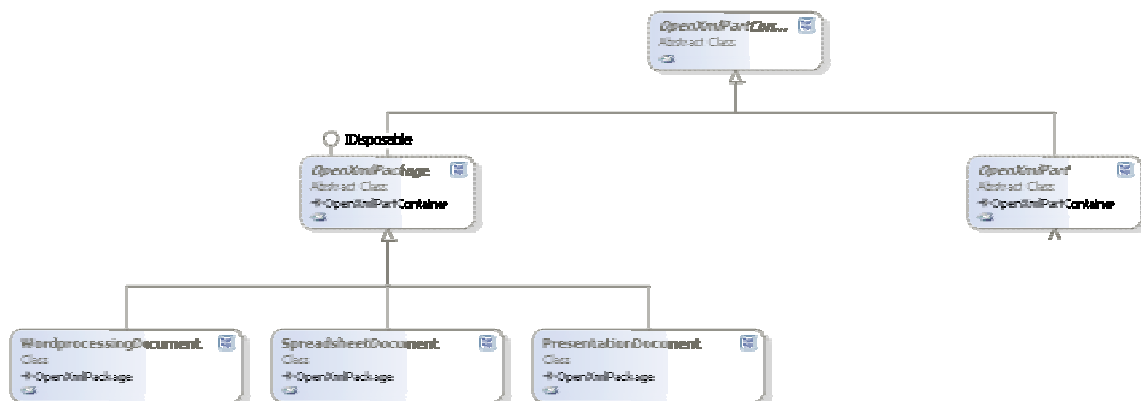


Figure 1 - Open XML object model (I)

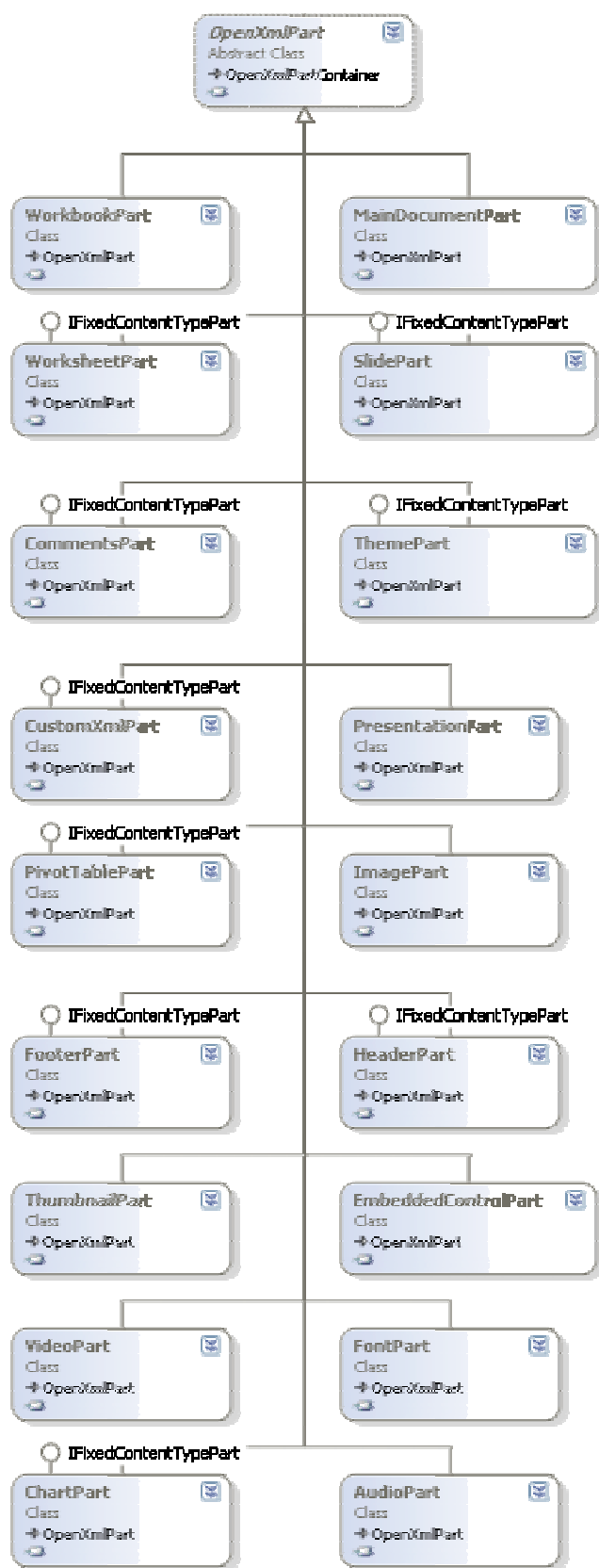


Figure 2 - Open XML object model (II)

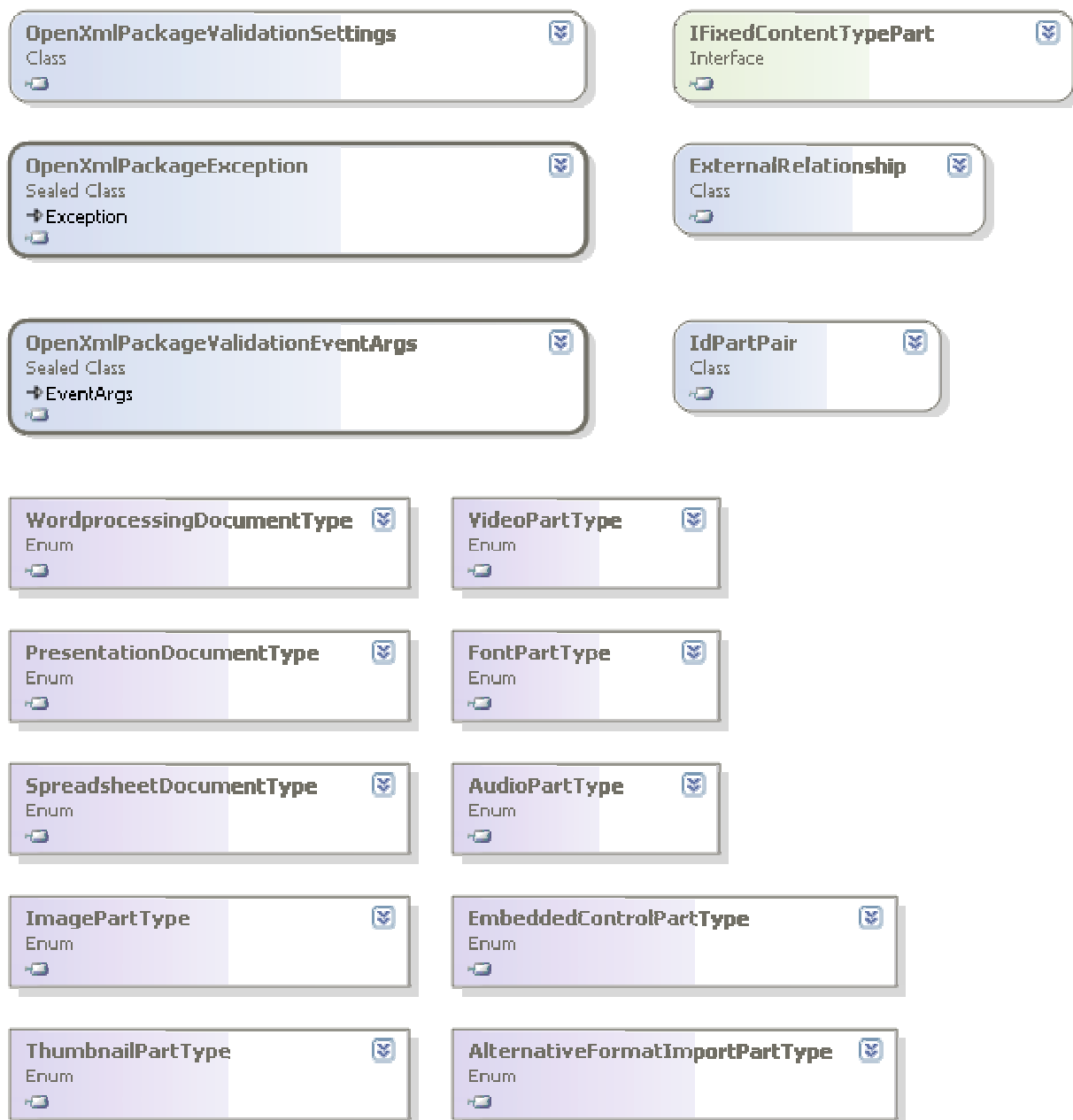


Figure 3 - Open XML object model (III)

28

The Open XML Formats Class Library Reference provides documentation for the complete set of classes, interfaces, and enumerations included in the Open XML object model.

29 The following objects are exercised by the TOE:

- **OpenXmlPackage Class**

(Microsoft.Office.DocumentFormat.OpenXml.Packaging)
Base class for strong typed package (document) class.

Namespace:

Microsoft.Office.DocumentFormat.OpenXml.Packaging

Assembly: Microsoft.Office.DocumentFormat.OpenXml (in
microsoft.office.documentformat.openxml.dll)

Syntax Visual Basic (Declaration)

```
Public MustInherit Class OpenXmlPackage
Inherits OpenXmlPartContainer
Implements IDisposable
```

C#

```
public abstract class OpenXmlPackage : OpenXmlPartContainer, IDisposable
```

Inheritance Hierarchy

System.Object

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPartContainer

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPackage

Microsoft.Office.DocumentFormat.OpenXml.Packaging.PresentationDocument

Microsoft.Office.DocumentFormat.OpenXml.Packaging.SpreadsheetDocument

Microsoft.Office.DocumentFormat.OpenXml.Packaging.WordprocessingDocument

Thread Safety

Any public static (Shared in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

- **OpenXmlPackageException**

Namespace: Microsoft.Office.DocumentFormat.OpenXml.Packaging

Assembly: Microsoft.Office.DocumentFormat.OpenXml (in
microsoft.office.documentformat.openxml.dll)

Syntax

Visual Basic (Declaration)

SerializableAttribute

```
Public NotInheritable Class OpenXmlPackageException
```

```
Inherits Exception
```

C#

```
[SerializableAttribute]
```

```
public sealed class OpenXmlPackageException : Exception
```

Inheritance Hierarchy

System.Object
System.Exception
Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPackageException

Thread Safety

Any public static (Shared in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

- **OpenXmlPackageValidationEventArgs Class**

(Microsoft.Office.DocumentFormat.OpenXml.Packaging)
Class for OpenXmlPackage validation event

Namespace: Microsoft.Office.DocumentFormat.OpenXml.Packaging
Assembly: Microsoft.Office.DocumentFormat.OpenXml (in microsoft.office.documentformat.openxml.dll)

Syntax

Visual Basic (Declaration)

SerializableAttribute

Public NotInheritable Class OpenXmlPackageValidationEventArgs
Inherits EventArgs

C#

[SerializableAttribute]

public sealed class OpenXmlPackageValidationEventArgs : EventArgs

Inheritance Hierarchy

System.Object
System.EventArgs

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPackageValidationEventArgs

Thread Safety

Any public static (Shared in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

- **OpenXmlPackageValidationSettings Class**

(Microsoft.Office.DocumentFormat.OpenXml.Packaging)

Specifies the event handlers that will handle OpenXmlPackage validation events and the OpenXmlPackageValidationEventArgs.

Namespace: Microsoft.Office.DocumentFormat.OpenXml.Packaging
Assembly: Microsoft.Office.DocumentFormat.OpenXml (in microsoft.office.documentformat.openxml.dll)

Syntax

Visual Basic (Declaration)

Public Class OpenXmlPackageValidationSettings

C#

```
public class OpenXmlPackageValidationSettings
```

Inheritance Hierarchy

System.Object

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPackageValidationSettings

Thread Safety

Any public static (Shared in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

- **OpenXmlPart Class**

(Microsoft.Office.DocumentFormat.OpenXml.Packaging)

Abstract base class for all OpenXml parts.

Namespace: Microsoft.Office.DocumentFormat.OpenXml.Packaging

Assembly: Microsoft.Office.DocumentFormat.OpenXml (in microsoft.office.documentformat.openxml.dll)

Syntax

Visual Basic (Declaration)

```
Public MustInherit Class OpenXmlPart
Inherits OpenXmlPartContainer
```

C#

```
public abstract class OpenXmlPart : OpenXmlPartContainer
```

Inheritance Hierarchy

System.Object

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPartContainer

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPart

Thread Safety

Any public static (Shared in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

- **OpenXmlPartContainer Class**

(Microsoft.Office.DocumentFormat.OpenXml.Packaging)

Part container, base class for OpenXmlPackage and OpenXmlPart.

Namespace: Microsoft.Office.DocumentFormat.OpenXml.Packaging

Assembly: Microsoft.Office.DocumentFormat.OpenXml (in

microsoft.office.documentformat.openxml.dll)

Syntax

Visual Basic (Declaration)

Public MustInherit Class OpenXmlPartContainer

C#

public abstract class OpenXmlPartContainer

Inheritance Hierarchy

System.Object

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPartContainer

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPackage

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPart

Thread Safety

Any public static (Shared in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

- **XmlSignaturePart Class**

(Microsoft.Office.DocumentFormat.OpenXml.Packaging)

XmlSignaturePart

Namespace: Microsoft.Office.DocumentFormat.OpenXml.Packaging

Assembly: Microsoft.Office.DocumentFormat.OpenXml (in microsoft.office.documentformat.openxml.dll)

Syntax

Visual Basic (Declaration)

Public Class XmlSignaturePart

Inherits OpenXmlPart

Implements IFixedContentTypePart

C#

public class XmlSignaturePart : OpenXmlPart, IFixedContentTypePart

Inheritance Hierarchy

System.Object

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPartContainer

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPart

Microsoft.Office.DocumentFormat.OpenXml.Packaging.XmlSignaturePart

Thread Safety

Any public static (Shared in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

- **DigitalSignatureOriginPart Class**

(Microsoft.Office.DocumentFormat.OpenXml.Packaging)

DigitalSignatureOriginPart

Namespace: Microsoft.Office.DocumentFormat.OpenXml.Packaging

Assembly: Microsoft.Office.DocumentFormat.OpenXml (in
microsoft.office.documentformat.openxml.dll)

Syntax

Visual Basic (Declaration)

```
Public Class DigitalSignatureOriginPart
    Inherits OpenXmlPart
    Implements IFixedContentTypePart
```

C#

```
public class DigitalSignatureOriginPart : OpenXmlPart,
    IFixedContentTypePart
```

Inheritance Hierarchy

System.Object

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPartContainer

Microsoft.Office.DocumentFormat.OpenXml.Packaging.OpenXmlPart

Microsoft.Office.DocumentFormat.OpenXml.Packaging.DigitalSignatureOriginPart

Thread Safety

Any public static (Shared in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

TOE description

- 30 The TOE is a set of libraries that implement the Office Open XML Open Packaging Conventions (OPC). These libraries are available as a “SDK for Open XML Formats”, provided by Microsoft.
- 31 The TOE, based on .NET technology, allows creating a valid Office Open XML document, to electronically sign it, and to compose a package according to the Open Packaging Conventions.
- 32 Office Open XML is an open standard for word-processing documents, presentations, and spreadsheets that can be freely implemented by multiple applications on different platforms. OpenXML is designed to faithfully represent existing word-processing documents, presentations, and spreadsheets that are encoded in binary formats defined by Microsoft Office applications.
- 33 An OpenXML file is stored in a ZIP archive for packaging and compression. The structure of any OpenXML file can be viewed using a ZIP viewer. Structurally, an OpenXML document is an Open Packaging Conventions (OPC) package composed of a collection of parts. The relationships between the parts are themselves stored in parts. The ZIP format supports random access to each part.
- 34 The parts in an OpenXML package are created as XML markup. Because XML is structured plain text, the contents of a part can be viewed using text readers or can be parsed using processes such as XPath.

Physical scope

- 35 Due to TOE nature (software product), there is no physical boundary.

Logical scope

- 36 The product performs two independent tasks – the package creation and the signature creation and validation – each of one can be addressed by a separate framework as shown below:

a. OPEN XML package creation

To create an OPEN XML package, two approaches may be applied:

- **Microsoft.Office.DocumentFormat.OpenXml.Packaging** Namespace (is in an external assembly that may be download and install - the classes are in the Microsoft.Office.DocumentFormat.OpenXml assembly).
- **System.IO.Packaging** Namespace (System.IO.Packaging is part of the .NET Framework 3.0 and 3.5 - The classes are in the WindowsBase assembly).

b. Sign and validate signatures

To sign and validate signatures, applications can use the .NET 3.0 classes **PackageDigitalSignatureManager**. The package-specific classes, defined in the **System.IO.Packaging** namespace, build on the digital signature classes of the Microsoft .NET 3.0 Framework defined in the **System.Security.Cryptography** namespace.

The **PackageDigitalSignatureManager** class is used for creating and validating signatures, and placing the signature infrastructure in a package. The signature is represented by an object based on the **PackageDigitalSignature** class.

37 The scope of the evaluation does not include the creation and validation signatures operations but only the construction the Open package. The **PackageDigitalSignatureManager** class is build on the **System.Security.Cryptography** namespace (external entity “Cryptographic Operations” in the figure), which in charge of performing the signature operations. This latter is out of the TOE logical scope.

38 The TOE logical boundary is depicted in the following figure (fig 4):

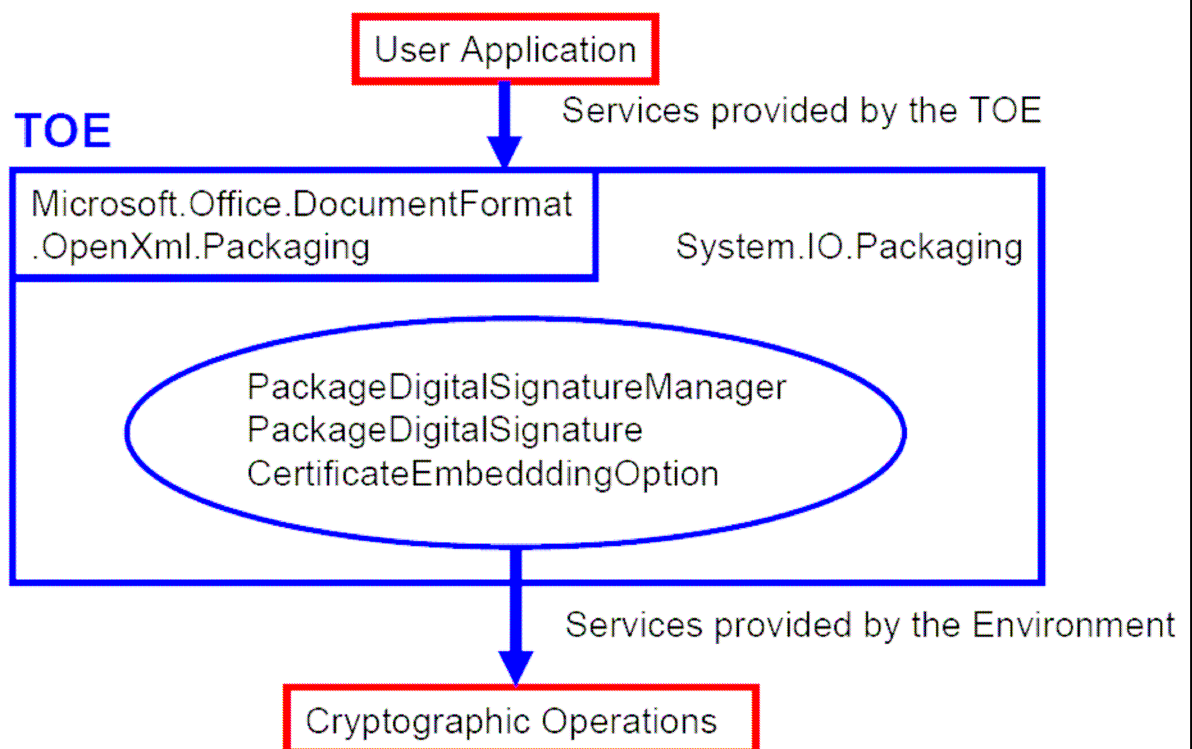


Figure 4: TOE Logical Scope

PP CONFORMANCE CLAIMS

CC Conformance Claim

CC Common Criteria for Information Technology Security Evaluation, v. 3.1, release 2, September 2007.

CEM Common Methodology for Information Technology Security Evaluation, v. 3.1, release 2, September 2007.

39 This Security Target is conformant to the **CC** v3.1, release 2, parts 2 and 3.

40 This Security Target is expected to be evaluated using the **CEM** v3.1, release 2.

PP Claim, Package Claim

41 This ST does not claim conformance to any PP. It is conformant to the assurance level EAL1, as defined by **CC** Part 3.

SECURITY OBJECTIVES

Security Objectives for the Operational Environment

- **SIGNATURE KEYS AND CERTIFICATES;**

The operational environment of the TOE is in charge of maintaining the signature keys under the sole control of the signatory of the Open Office XML document, and also in charge of maintaining the integrity of the public certificates required to validate a signed Open Office XML document.

- **CRYPTO SERVICES;**

The operational environment of the TOE is the provider of the cryptographic services that the TOE invokes to perform the digital signature creation and verification of the Open Office XML documents. The TOE does not prefer any cryptographic algorithm or key length, but uses one of the many provided by the operational environment cryptographic services, as selected by the user.

Note that the strength of the cryptographic algorithms is out of the scope of the Common Criteria certification.

SECURITY REQUIREMENTS FOR THE TOE

Functional Security Requirements

FDP_DAU.1 Basic Data Authentication

Family Behaviour

42 Data authentication permits an entity to accept responsibility for the authenticity of information (e.g., by digitally signing it). This family provides a method of providing a guarantee of the validity of a specific unit of data that can be subsequently used to verify that the information content has not been forged or fraudulently modified.

User application notes

43 This component may be satisfied by one-way hash functions (cryptographic checksum, fingerprint, message digest), to generate a hash value for a definitive document that may be used as verification of the validity or authenticity of its information content.

Hierarchical to: No other components.

Dependencies: No dependencies.

FDP_DAU.1.1 The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of [assignment: [the components of an Open Office XML document which complies with the Open Packaging Conventions \(a_1\)](#)].

44 (a_1) the PP/ST author should specify the list of objects or information types for which the TSF shall be capable of generating data authentication evidence.

FDP_DAU.1.2 The TSF shall provide [assignment: [any user of the TOE \(a_1\)](#)] with the ability to verify evidence of the validity of the indicated information.

45 (a_1) the PP/ST author should specify the list of subjects that will have the ability to verify data authentication evidence for the objects identified in the previous element. The list of subjects could be very specific, if the subjects are known, or it could be more generic and refer to a “type” of subject such as an identified role.

Assurance Security Requirements

46 The development and the evaluation of the TOE shall be done in accordance to the following security assurance requirements:

- EAL1

ADV_FSP.1 Basic functional specification

Dependencies: No dependencies.

Developer action elements:

ADV_FSP.1.1D The developer shall provide a functional specification.

ADV_FSP.1.2D The developer shall provide a tracing from the functional specification to the SFRs.

Content and presentation of evidence elements:

ADV_FSP.1.1C The functional specification shall describe the purpose and method of use for each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.2C The functional specification shall identify all parameters associated with each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.3C The functional specification shall provide rationale for the implicit categorisation of interfaces as SFR-non-interfering.

ADV_FSP.1.4C The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

AGD_OPE.1 Operational user guidance

Dependencies: ADV_FSP.1 Basic functional specification

Developer action elements:

AGD_OPE.1.1D The developer shall provide operational user guidance.

Content and presentation of evidence elements:

AGD_OPE.1.1C The operational user guidance shall describe, for each user role, the user-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.

AGD_OPE.1.2C The operational user guidance shall describe, for each user role, how to use the available interfaces provided by the TOE in a secure manner.

AGD_OPE.1.3C The operational user guidance shall describe, for each user role, the available functions and interfaces, in particular all security parameters under the control of the user, indicating secure values as appropriate.

AGD_OPE.1.4C The operational user guidance shall, for each user role, clearly present each type of security-relevant event relative to the user-accessible functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_OPE.1.5C The operational user guidance shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.

AGD_OPE.1.6C The operational user guidance shall, for each user role, describe the security measures to be followed in order to fulfil the security objectives for the operational environment as described in the ST.

AGD_OPE.1.7C The operational user guidance shall be clear and reasonable.

AGD_PRE.1 Preparative procedures

Dependencies: No dependencies.

Developer action elements:

AGD_PRE.1.1D The developer shall provide the TOE including its preparative procedures.

Content and presentation of evidence elements:

AGD_PRE.1.1C The preparative procedures shall describe all the steps necessary for secure acceptance of the delivered TOE in accordance with the developer's delivery procedures.

AGD_PRE.1.2C The preparative procedures shall describe all the steps necessary for secure installation of the TOE and for the secure preparation of the operational environment in accordance with the security objectives for the operational environment as described in the ST.

ALC_CMC.1 Labelling of the TOE

Dependencies: ALC_CMS.1 TOE CM coverage

Developer action elements:

ALC_CMC.1.1D The developer shall provide the TOE and a reference for the TOE.

Content and presentation of evidence elements:

ALC_CMC.1.1C The TOE shall be labelled with its unique reference.

ALC_CMS.1 TOE CM coverage

Dependencies: No dependencies.

Developer action elements:

ALC_CMS.1.1D The developer shall provide a configuration list for the TOE.

Content and presentation of evidence elements:

ALC_CMS.1.1C The configuration list shall include the following: the TOE itself; and the evaluation evidence required by the SARs.

ALC_CMS.1.2C The configuration list shall uniquely identify the configuration items.

ASE_INT.1 ST introduction

Dependencies: No dependencies.

Developer action elements:

ASE_INT.1.1D The developer shall provide an ST introduction.

Content and presentation of evidence elements:

ASE_INT.1.1C The ST introduction shall contain an ST reference, a TOE reference, a TOE overview and a TOE description.

ASE_INT.1.2C The ST reference shall uniquely identify the ST.

ASE_INT.1.3C The TOE reference shall identify the TOE.

ASE_INT.1.4C The TOE overview shall summarise the usage and major security features of the TOE.

ASE_INT.1.5C The TOE overview shall identify the TOE type.

ASE_INT.1.6C The TOE overview shall identify any non-TOE hardware/software/firmware required by the TOE.

ASE_INT.1.7C The TOE description shall describe the physical scope of the TOE.

ASE_INT.1.8C The TOE description shall describe the logical scope of the TOE.

ASE_CCL.1 Conformance claims

Dependencies: ASE_INT.1 ST introduction

ASE_ECD.1 Extended components definition

ASE_REQ.1 Stated security requirements

Developer action elements:

ASE_CCL.1.1D The developer shall provide a conformance claim.

ASE_CCL.1.2D The developer shall provide a conformance claim rationale.

Content and presentation of evidence elements:

ASE_CCL.1.1C The conformance claim shall contain a CC conformance claim that identifies the version of the CC to which the ST and the TOE claim conformance.

ASE_CCL.1.2C The CC conformance claim shall describe the conformance of the ST to CC Part 2 as either CC Part 2 conformant or CC Part 2 extended.

ASE_CCL.1.3C The CC conformance claim shall describe the conformance of the ST to CC Part 3 as either CC Part 3 conformant or CC Part 3 extended.

ASE_CCL.1.4C The CC conformance claim shall be consistent with the extended components definition.

ASE_CCL.1.5C The conformance claim shall identify all PPs and security requirement packages to which the ST claims conformance.

- ASE_CCL.1.6C** The conformance claim shall describe any conformance of the ST to a package as either package-conformant or package-augmented.
- ASE_CCL.1.7C** The conformance claim rationale shall demonstrate that the TOE type is consistent with the TOE type in the PPs for which conformance is being claimed.
- ASE_CCL.1.8C** The conformance claim rationale shall demonstrate that the statement of the security problem definition is consistent with the statement of the security problem definition in the PPs for which conformance is being claimed.
- ASE_CCL.1.9C** The conformance claim rationale shall demonstrate that the statement of security objectives is consistent with the statement of security objectives in the PPs for which conformance is being claimed.
- ASE_CCL.1.10C** The conformance claim rationale shall demonstrate that the statement of security requirements is consistent with the statement of security requirements in the PPs for which conformance is being claimed.
- ASE_OBJ.1** Security objectives for the operational environment
- Dependencies: No dependencies.
- Developer action elements:
- ASE_OBJ.1.1D** The developer shall provide a statement of security objectives.
- Content and presentation of evidence elements:
- ASE_OBJ.1.1C** The statement of security objectives shall describe the security objectives for the operational environment.
- ASE_ECD.1** Extended components definition
- Dependencies: No dependencies.
- Developer action elements:
- ASE_ECD.1.1D** The developer shall provide a statement of security requirements.
- ASE_ECD.1.2D** The developer shall provide an extended components definition.
- Content and presentation of evidence elements:
- ASE_ECD.1.1C** The statement of security requirements shall identify all extended security requirements.
- ASE_ECD.1.2C** The extended components definition shall define an extended component for each extended security requirement.
- ASE_ECD.1.3C** The extended components definition shall describe how each extended component is related to the existing CC components, families, and classes.
- ASE_ECD.1.4C** The extended components definition shall use the existing CC components, families, classes, and methodology as a model for presentation.
- ASE_ECD.1.5C** The extended components shall consist of measurable and objective elements such that conformance or nonconformance to these elements can be demonstrated.

ASE_REQ.1 Stated security requirements

Dependencies: ASE_ECD.1 Extended components definition

Developer action elements:

ASE_REQ.1.1D The developer shall provide a statement of security requirements.

ASE_REQ.1.2D The developer shall provide a security requirements rationale.

Content and presentation of evidence elements:

ASE_REQ.1.1C The statement of security requirements shall describe the SFRs and the SARs.

ASE_REQ.1.2C All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.

ASE_REQ.1.3C The statement of security requirements shall identify all operations on the security requirements.

ASE_REQ.1.4C All operations shall be performed correctly.

ASE_REQ.1.5C Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.

ASE_REQ.1.6C The statement of security requirements shall be internally consistent.

ASE_TSS.1 TOE summary specification

Dependencies: ASE_INT.1 ST introduction

ASE_REQ.1 Stated security requirements

ADV_FSP.1 Basic functional specification

Developer action elements:

ASE_TSS.1.1D The developer shall provide a TOE summary specification.

Content and presentation of evidence elements:

ASE_TSS.1.1C The TOE summary specification shall describe how the TOE meets each SFR.

ATE_IND.1 Independent testing - conformance

Dependencies: ADV_FSP.1 Basic functional specification

AGD_OPE.1 Operational user guidance

AGD_PRE.1 Preparative procedures

Developer action elements:

ATE_IND.1.1D The developer shall provide the TOE for testing.

Content and presentation of evidence elements:

ATE_IND.1.1C The TOE shall be suitable for testing.

AVA_VAN.1 Vulnerability survey

Dependencies: ADV_FSP.1 Basic functional specification

AGD_OPE.1 Operational user guidance

AGD_PRE.1 Preparative procedures

Developer action elements:

AVA_VAN.1.1D The developer shall provide the TOE for testing.

Content and presentation of evidence elements:

AVA_VAN.1.1C The TOE shall be suitable for testing.

TOE SUMMARY SPECIFICATION

- 47 FDP_DAU.1 Basic Data Authentication requests the TOE to be able to include a means to ensure the authenticity of an Open Office XML document, as packaged and signed in accordance with the Open Packaging Conventions.
- 48 The TOE is a set of libraries provided as an SDK. A number of examples can be used to show how to fulfill the FDP_DAU.1 Basic Data Authentication requirement, a some of these examples are included in the proper TOE.
- 49 When a document needs to be sent electronically, there should be techniques to ensure that the document sent is authentic and the content is not tampered which ensures integrity. This can be achieved by applying Digital Signature to the document.
- 50 A Digital Signature is a type of asymmetric cryptography used to simulate the security properties of a signature in a digital way, rather than in written form. When a Digital Signature is applied it normally generates two algorithms, one for signing which involves the user's secret or private key and the other for verifying signatures which involves the user's public key. The output of the signature process is called the "Digital Signature".
- 51 Document author appends Digital Signature to the content of the document along with digital certificate, if required. Document consumer validates the integrity of the content and authenticity of the document.
- 52 Steps to apply Digital Signature to an Open XML document are as follows:
- Create an Open XML document.
 - Add some content into it.
 - Save the document
 - Apply Digital signature along with Digital Certificate
 - Add the digital signature to the Open Package structure.
 - Give necessary details along with digital certificate
 - Close the document. This ensures the package is created, including all the relationships between the document components, including the digital signature as a payload of the ZIP file.

- 53 When the document is unzipped, the following parts are created to hold the Digital Signature details
- Digital Signature Origin Part. This is the starting point for the digital signature. The package can include only one Digital Signature part. This file is of ZERO byte size and does not contain any data. The presence of this part indicates that the document is signed.
 - Digital Signature XML signature Part. This part is linked from the Digital Signature origin part. This contains Digital Signature markup details. A package/document can contain more than one Digital Signature XML part.
 - The reference from the Digital Signature Origin to the Digital Signature XML Signature part is by a relationship ID.
- 54 The XML signature contains the following details.
- The algorithm used for hashing the content
 - The reference element used for signature
 - The hash algorithm and hashed value
 - Signature value (digital signature)
 - Key information for encryption and decryption
 - Digital certificate details
 - The hashed value of the content
 - For a selected relationships and parts, the hash algorithm and hashed value
 - Signature property details for package specific object.
- 55 When the document is validated, a double validation occurs. First, the compliance of the package with respect to the Open Package Conventions and with respect to the proper Office Open XML schemas, and then the verification of the digital signature(s), that ensures the validity of the document. The TOE provides the results of these validations to the user.
- 56 The following interfaces exercise the application of a digital signature to an Open XML document fulfilling the FDP_DAU.1 requirement and becoming therefore, the TSFIs of the Functional Specification labelled as SFR-enforcing or SFR-supporting.

Task	Class	Package	Method
WordProcessing Open XML package creation	WordprocessingDocument	Microsoft.Office.DocumentFormat.OpenXml.Packaging	Create
			AddMainDocumentPart
Presentation Open XML package creation	PresentationDocument	Microsoft.Office.DocumentFormat.OpenXml.Packaging	Create
			AddPresentationPart
SpreadSheet Open XML package creation	SpreadsheetDocument	Microsoft.Office.DocumentFormat.OpenXml.Packaging	Create
			AddWorkbookPart
Signature creation	Package	System.IO.Packaging	Open
			GetRelationshipsByType
			GetPart
	PackageRelationshipSelector	System.IO.Packaging	PackageRelationshipSelector
	PackageDigitalSignatureManager	System.IO.Packaging	PackageDigitalSignatureManager
	PackagePart	System.IO.Packaging	Sign
Signature validation	Package	System.IO.Packaging	GetRelationships
			ResolvePartUri
			Open
	PackageDigitalSignatureManager	System.IO.Packaging	Close
			GetParts
			GetPart
PackageDigitalSignatureManager	System.IO.Packaging	PackageDigitalSignatureManager	
			VerifyCertificate
			VerifySignatures

Table 1 TOE security functionality interfaces

- 57 The Open XML Formats Class Library Reference ([CLR]) and the System.IO.Packaging Namespace .NET Framework 3.5 Class Library ([SIO]) provide documentation for the complete set of classes, interfaces, and enumerations included in the Open XML object model.
- 58 The signature operations provided by the environment (by means of the methods and classes included in the **System.Security.Cryptography** package) represent interfaces to functionality in the IT environment and are not TSFIs. This interface is detailed in [CRY].