TCG Infrastructure Working Group
Reference Manifest (RM) Schema
Specification

Specification Version 1.0
Revision 1.0
17 November 2006
FINAL

Contacts:
thardjono@signacert.com (editor, co-chair)
Ned.Smith@intel.com (co-chair)
Copyright © 2006 Trusted Computing Group, Incorporated.

Disclaimer

THIS SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Without limitation, TCG disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification and to the implementation of this specification, and TCG disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this specification or any information herein.

No license, express or implied, by estoppel or otherwise, to any TCG or TCG member intellectual property rights is granted herein.

Except that a license is hereby granted by TCG to copy and reproduce this specification for internal use only.

Contact the Trusted Computing Group at www.trustedcomputinggroup.org for information on specification licensing through membership agreements.

Any marks and brands contained herein are the property of their respective owners.
Acknowledgement

The TCG wishes to thank all those who contributed to this specification. This document builds on numerous work done in the various working groups in the TCG.

<table>
<thead>
<tr>
<th>Name</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diana Arroyo</td>
<td>IBM</td>
</tr>
<tr>
<td>Lee Terrell</td>
<td>IBM</td>
</tr>
<tr>
<td>Markus Gueller</td>
<td>Infineon</td>
</tr>
<tr>
<td>Ned Smith (co-chair)</td>
<td>Intel Corporation</td>
</tr>
<tr>
<td>Thomas Hardjono (editor, co-chair)</td>
<td>SignaCert</td>
</tr>
<tr>
<td>David Bleckman</td>
<td>SignaCert</td>
</tr>
<tr>
<td>Ron Forrester</td>
<td>SignaCert</td>
</tr>
<tr>
<td>Jeff Nisewanger</td>
<td>Sun</td>
</tr>
<tr>
<td>Wyllys Ingersoll</td>
<td>Sun</td>
</tr>
<tr>
<td>Paul Sangster</td>
<td>Symantec</td>
</tr>
<tr>
<td>Greg Kazmierczak</td>
<td>Wave Systems</td>
</tr>
<tr>
<td>Len Veil</td>
<td>Wave Systems</td>
</tr>
</tbody>
</table>
# Table of Contents

1 **Scope and Audience** .................................................................................................................. 6

2 **Introduction** ................................................................................................................................. 7
   2.1 Schema Version .............................................................................................................................. 7
   2.2 Schema Namespace ....................................................................................................................... 7
   2.3 Dependent Schema Definitions ................................................................................................... 7
       2.3.1 W3C XML Schema Syntax ....................................................................................................... 7
       2.3.2 W3C XML-Signature Syntax ..................................................................................................... 7
   2.4 Notice Regarding Notation ........................................................................................................... 7

3 **Reference Manifest (RM) Schema** .............................................................................................. 8
   3.1 Complex Types ............................................................................................................................. 8
       3.1.1 complexType RimmType ......................................................................................................... 8
       3.1.2 complexType core:AssertionType ............................................................................................ 10
       3.1.3 complexType core:ComponentIDType .................................................................................... 11
       3.1.4 complexType core:ComponentRefType .................................................................................. 13
       3.1.5 complexType core:ConfidenceValueType ................................................................................ 14
       3.1.6 complexType core:DigestMethodType ..................................................................................... 15
       3.1.7 complexType core:DigestValueType ......................................................................................... 16
       3.1.8 complexType core:HashType .................................................................................................. 17
       3.1.9 complexType core:IntegrityManifestType ............................................................................... 18
       3.1.10 complexType core:PlatformClassType ................................................................................ 20
       3.1.11 complexType core:SignerInfoType ....................................................................................... 21
       3.1.12 complexType core:TransformMethodType ........................................................................... 22
       3.1.13 complexType core:Value>Type ............................................................................................... 23
       3.1.14 complexType core:VendorIDType ......................................................................................... 24

   3.2 Elements ......................................................................................................................................... 26
       3.2.1 element Rimm .......................................................................................................................... 26
       3.2.2 element RimmType/CompositeHash ....................................................................................... 27
       3.2.3 element core:ComponentIDType/VendorID ........................................................................... 28
       3.2.4 element core:ComponentRefType/ComponentID .................................................................. 29
       3.2.5 element core:ComponentRefType/ComponentIDREF ............................................................ 29
       3.2.6 element core:IntegrityManifestType/ComponentID ................................................................. 30
       3.2.7 element core:IntegrityManifestType/SignerInfo .................................................................... 32
       3.2.8 element core:IntegrityManifestType/ConfidenceValue .......................................................... 33
       3.2.9 element core:IntegrityManifestType/Collector ...................................................................... 34
       3.2.10 element core:IntegrityManifestType/TransformMethod ......................................................... 35
       3.2.11 element core:IntegrityManifestType/DigestMethod ............................................................... 36
       3.2.12 element core:IntegrityManifestType/Values ......................................................................... 36
       3.2.13 element core:IntegrityManifestType/AssertionInfo ............................................................... 37
       3.2.14 element core:IntegrityManifestType/PlatformClass .............................................................. 38
       3.2.15 element core:IntegrityManifestType/SubComponents ........................................................... 38
       3.2.16 element core:SignerInfoType/SigningComponent .................................................................. 39
       3.2.17 element core:VendorIDType/TcgVendorID ............................................................................ 40
       3.2.18 element core:VendorIDType/SmiVendorID ........................................................................... 41
       3.2.19 element core:VendorIDType/VendorGUID ................................................................ .......... 41

4 **References** ..................................................................................................................................... 43
1 Scope and Audience

This specification is integral to the TCG Infrastructure Working Group’s (IWG) reference architecture, and is directly related to the TCG’s Integrity Management Model. Specifically, the Reference Manifest (RM) XML schema defines the structure with which integrity information is communicated between entities.

Architects, designers, developers, and technologists interested in the development, deployment, and interoperation of trusted systems will find this document necessary in providing a specific mechanism for communicating integrity information.
2 Introduction

The purpose of this document is to provide a detailed description of the TCG Infrastructure Working Group’s Reference Manifest (RM) schema, hereafter referred to as the \textit{RM schema}. The RM schema is derived from the core integrity metadata XML schema \cite{TCG1}.

2.1 Schema Version

The RM schema’s version number is defined using the \texttt{version} attribute of the schema’s root-level \texttt{schema} element:

\begin{verbatim}
  version="version_number"
\end{verbatim}

This document refers to version 1.00 of the RM schema.

2.2 Schema Namespace

The RM schema’s namespace is defined using the \texttt{targetNamespace} attribute of the schema’s root-level \texttt{schema} element:

\begin{verbatim}
  targetNamespace="namespace"
\end{verbatim}

The schema’s namespace reflects the schema version, and is currently defined as follows:

http://www.trustedcomputinggroup.org/XML/SCHEMA/Reference_Manifest_v1_0#

2.3 Dependent Schema Definitions

2.3.1 W3C XML Schema Syntax

The RM schema relies upon data structures defined by the World Wide Web Consortium’s (W3C) XML-Schema syntax. Consequently, the RM schema imports the W3C’s XML schema with the following namespace:

http://www.w3.org/2001/XMLSchema

The RM schema associates the abovementioned schema with the “xs” namespace prefix.

2.3.2 W3C XML-Signature Syntax

The RM schema relies upon data structures defined by the World Wide Web Consortium’s (W3C) XML-Signature digital signature syntax. Consequently, the RM schema imports the W3C’s digital signature XML schema with the following namespace:

http://www.w3.org/2000/09/xmldsig#

The RM schema associates the abovementioned schema with the “ds” namespace prefix.

2.4 Notice Regarding Notation

The current specification document is heavily dependent on the Core Integrity Schema specification \cite{TCG3} since the RM Schema inherits from the Core Integrity Schema. As such, the Core Integrity Schema is the authoritative source regarding the description and definition of elements and attributes inherited from the Core Integrity Schema.

In order to provide ease of readability of the current document and for completeness, in this document any text with gray shading or background (such as here) is denoted as having been obtained from the \textit{Core Integrity Schema} specification \cite{TCG3}. As such, further information regarding such elements or attributes should be found in \cite{TCG3}.
3 Reference Manifest (RM) Schema

Elements

<table>
<thead>
<tr>
<th>Complex types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rimm</td>
</tr>
</tbody>
</table>

Complex types

- AssertionType
- ComponentIDType
- ComponentRefType
- ConfidenceValueType
- DigestMethodType
- DigestValueType
- HashType
- IntegrityManifestType
- PlatformClassType
- SignerInfoType
- TransformMethodType
- ValueType
- VendorIdType

3.1 Complex Types

3.1.1 complexType RimmType

3.1.1.1 Description

The Reference Manifest (RM) is the representation of a component’s reference measurements [1] created by an authoritative party (e.g. vendor, manufacturer, or IT department). The purpose is to support a comparison of an operational environment against known good (Reference) measurements. Each RM describes a single component; however a component may have many subcomponents, each of which may have a corresponding RM.

The RM structure is derived from the Core Integrity Schema [3]. The RM extends the core schema through restriction as defined by xml schema.

The RM relies on domain-specific schemas for describing the metadata of the reference values. Examples domain-specific schemas include the Simple Values Schema and Binary Values Schema.

It is useful to note that RM has attributes that it inherits from the Core Schema, as well attributes that it extends from the Core Schema.
3.1.1.2 Attribute Details

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>This is the RM ID. This is a globally unique identifier (e.g. UUID as defined by …) for the RM structure which can be used to distinguish multiple RM instances describing the same component.</td>
</tr>
<tr>
<td>RevLevel</td>
<td>The revision level of the RM covers the cases where the component ID is the same, but the integrity measurement values in the RM instance change. 2 specific cases are as follows: 1) Values change, component ID the same 2) RM xml does not change, but the domain specific xml changes (e.g. instead of simple values, use binary image)</td>
</tr>
<tr>
<td>UUID</td>
<td>This is the identifier for the instance of the structure. It needs to be globally unique in order to be able to be used to correlate a snapshot instance and a RM instance.</td>
</tr>
</tbody>
</table>

3.1.1.3 XML

```xml
<xs:complexType name="RimmType">
  <xs:complexContent>
    <xs:extension base="core:IntegrityManifestType">
      <xs:sequence>
        <xs:element name="CompositeHash" type="core:HashType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="UUID" type="xs:NMTOKEN" use="required"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

3.1.2 complexType core:AssertionType

3.1.2.1 Description [3]

AssertionType consists of a record identifier and any other element containing assertions expressed in XML. Assertions are specific to a domain of interpretation, hence should be described using an applicable schema definition. AssertionType provides an extensibility feature for incorporating domain-specific assertions into integrity manifest and reporting structures. The TCG Security Qualities [5] schema is an example of an XML schema containing assertions.

3.1.2.2 Diagram
3.1.2.3 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Globally unique record instance identifier. ID may be used to distinguish multiple instances of elements of type AssertionType. If the domain-specific schema defines an xs:ID identifier, it should have the same value as ID.</td>
</tr>
</tbody>
</table>

3.1.2.4 XML

```xml
<xs:complexType name="AssertionType" abstract="false">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax"/>
  </xs:sequence>
  <xs:attribute name="Id" type="xs:ID"/>
</xs:complexType>
```

3.1.3 complexType core:ComponentIDType

3.1.3.1 Description [3]

The ComponentIDType complex type represents an atomic integrity element identifying a particular program code or logic (hereafter referred to as a component). The identifier does not try to distinguish multiple instances of the same code or logic. For example, the ComponentIDType complex type is used within a TCG Reference Manifest (RM) to represent application integrity values derived from a baseline build image. ComponentIDType is also used by the Snapshot complex type in the TCG Integrity Report (IR) schema to capture actual measurements of components that may be extended into PCRs.

ComponentIDType is a set of attributes accommodating a wide range of change management schemes that when combined uniquely identifies a change-controlled item. The package, program code or logic under change management will have processes for ensuring integrity of its image. VendorID must uniquely identify an entity that maintains the change management process. If the VendorID is a GUID, then it is assumed the change management process owner can be obtained some other way (e.g. via database lookup using the GUID as a database key).

Most attributes are optional to ensure applicability across a variety of change management systems. However the vendorID element must be unique with respect to all possible vendors.
### 3.1.3.2 Attribute Details [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Record instance identifier – recommended globally unique</td>
</tr>
<tr>
<td>SimpleName</td>
<td>String-ified version information for simple compare operations</td>
</tr>
<tr>
<td>ModelName</td>
<td>Model name with which the component is marketed</td>
</tr>
<tr>
<td>ModelNumber</td>
<td>Alphanumeric model number with which the component is identified</td>
</tr>
<tr>
<td>ModelSerialNumber</td>
<td>Alphanumeric model serial number with which the component is identified</td>
</tr>
<tr>
<td>ModelSystemClass</td>
<td>Vendor-specific system type or environment with which the component is associated</td>
</tr>
<tr>
<td>VersionMajor</td>
<td>Major version number of the component</td>
</tr>
<tr>
<td>VersionMinor</td>
<td>Minor version number of the component</td>
</tr>
<tr>
<td>VersionBuild</td>
<td>Build number of the component</td>
</tr>
<tr>
<td>VersionString</td>
<td>String with which the component’s version may be identified</td>
</tr>
<tr>
<td>BuildDate</td>
<td>Date on which the component was manufactured</td>
</tr>
<tr>
<td>PatchLevel</td>
<td>Patch level of the component</td>
</tr>
<tr>
<td>DiscretePatches</td>
<td>Token strings enumerating each discrete patch that has been applied to the component; that is not also represented by PatchLevel or other attributes in ComponentType</td>
</tr>
</tbody>
</table>

### 3.1.3.3 XML

```
<xs:complexType name="ComponentIDType">
  <xs:sequence>
    <xs:element name="VendorID" type="VendorIdType"/>
  </xs:sequence>
  <xs:attribute name="Id" type="xs:ID" use="required"/>
  <xs:attribute name="SimpleName" type="xs:normalizedString" use="optional"/>
  <xs:attribute name="ModelName" type="xs:normalizedString" use="optional"/>
  <xs:attribute name="ModelNumber" type="xs:normalizedString" use="optional"/>
  <xs:attribute name="ModelSerialNumber" type="xs:normalizedString" use="optional"/>
  <xs:attribute name="ModelSystemClass" type="xs:normalizedString" use="optional"/>
  <xs:attribute name="VersionMajor" type="xs:integer" use="optional"/>
  <xs:attribute name="VersionMinor" type="xs:integer" use="optional"/>
  <xs:attribute name="VersionBuild" type="xs:integer" use="optional"/>
  <xs:attribute name="VersionString" type="xs:normalizedString" use="optional"/>
  <xs:attribute name="MfgDate" type="xs:dateTime" use="optional"/>
  <xs:attribute name="PatchLevel" type="xs:normalizedString" use="optional"/>
  <xs:attribute name="DiscretePatches" type="xs:NMTOKENS" use="optional"/>
</xs:complexType>
```
3.1.4 complexType core:ComponentRefType

3.1.4.1 Description [3]
The ComponentRefType complexType is used to refer to components in other locations documents or repositories. There are three references that are useful in identifying a component.

- ComponentIDREF – a reference within an XML document.
- ComponentLoc – a reference to a web resource.
- ComponentID element – a ComponentIDType structure whose attributes may be used to perform a database query.

3.1.4.2 Diagram

```xml
<xs:complexType name="ComponentRefType">
  <xs:choice>
    <xs:element name="ComponentID">
      <xs:complexType>
        <xs:attribute name="ComponentID" type="ComponentIDType" use="required" />
        <xs:attribute name="DerivedBy" type="core:ExtensionType" use="optional"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="ComponentLoc">
      <xs:complexType>
        <xs:attribute name="ComponentLoc" type="xs:anyURI" use="optional" />
      </xs:complexType>
    </xs:element>
    <xs:element name="ComponentID" type="ComponentIDType" use="optional"/>
  </xs:choice>
</xs:complexType>
```

3.1.4.3 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ComponentLoc</td>
<td>A URI referencing a document containing an element of type ComponentIDType</td>
</tr>
</tbody>
</table>

3.1.4.4 XML

```xml
<source
  <xs:complexType name="ComponentRefType">
    <xs:choice>
      <xs:element name="ComponentID">
        <xs:complexType>
          <xs:attribute name="ComponentID" type="ComponentIDType" use="required" />
          <xs:attribute name="DerivedBy" type="core:ExtensionType" use="optional"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="ComponentLoc">
        <xs:complexType>
          <xs:attribute name="ComponentLoc" type="xs:anyURI" use="optional" />
        </xs:complexType>
      </xs:element>
      <xs:element name="ComponentID" type="ComponentIDType" use="optional"/>
    </xs:choice>
  </xs:complexType>
</source>
```
3.1.5 complexType core:ConfidenceValueType

3.1.5.1 Description [3]

The ConfidenceValueType complex type represents the level of confidence (hereafter referred to as a confidence value) with which a numerical representation of trust may be given to the assertion with which it is associated. For example, the ConfidenceValueType complex type is applied within the IntegrityMetadataType complex type to identify the level of confidence with which to trust a single collection of integrity metadata.

Further examples of assertions that may be assigned confidence values include integrity assertions and integrity values (represented using IntegrityAssertionType and IntegrityValueType complex types, respectively).

A confidence value is a rational number. Two values are integral to the calculation of a confidence value:

- **Score** – The confidence points given to an assertion. The score must be greater than or equal to 0, and less than or equal to the specified basis.
- **Basis** – The maximum number of confidence points that may be given to an assertion. The basis must be an integer greater than 0.
- **Authority** – The entity that defines criteria for establishing the Basis is optionally provided in the form of a URI.

An assertion’s confidence value is calculated by dividing its score into its basis. For example, given a basis of 100, an assertion whose score is 95 will receive a confidence value of 0.95.

Cooperation between producers and consumers of documents containing ConfidenceValue may establish scoring conventions such that all have a common frame of understanding. This specification does not define such a convention. However, a URI reference to an entity that defines such criteria can be provided.

```xml
<xs:complexType>
  <xs:complexContent>
    <xs:extension base="ComponentIDType"/>
  </xs:complexContent>
</xs:complexType>
<xs:element name="ComponentIDREF" type="xs:IDREF"/>
</xs:choice>
<xs:attribute name="ComponentLoc" type="xs:anyURI" use="optional"/>
</xs:complexType>
```
3.1.5.2 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Score</td>
<td>Confidence points given to an assertion. Greater than or equal to 0, and less than or equal to the specified basis.</td>
</tr>
<tr>
<td>Basis</td>
<td>Maximum number of confidence points that may be given to an assertion. Greater than 0.</td>
</tr>
<tr>
<td>Authority</td>
<td>Reference to an authoritative source that defines criteria for establishing the Basis value.</td>
</tr>
</tbody>
</table>

3.1.5.3 XML

```xml
<xs:complexType name="ConfidenceValueType">
  <xs:attribute name="Score" type="xs:integer" use="required"/>
  <xs:attribute name="Basis" type="xs:integer" use="required"/>
  <xs:attribute name="Authority" type="xs:anyURI" use="optional"/>
</xs:complexType>
```

3.1.6 complexType core:DigestMethodType

3.1.6.1 Description [3]

DigestMethodType identifies cryptographic hash algorithms. There may be several different digest algorithms used when generating a Reference Manifest (RM). Instances of elements of type DigestMethodType are referenced using the ID attribute.

3.1.6.2 Diagram

![Diagram of DigestMethodType]
3.1.6.3 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Document unique record instance identifier. Id is used in other parts of the document to reference instances of hash algorithm identifiers.</td>
</tr>
<tr>
<td>Algorithm</td>
<td>xs:anyURI defining a well-known digest algorithm. SHA-1 must be implemented as a minimum for interoperability. (e.g. <a href="http://www.w3.org/2000/09/xmldsig#sha1">http://www.w3.org/2000/09/xmldsig#sha1</a>)</td>
</tr>
<tr>
<td>any##other</td>
<td>Defines other digest algorithms not available through the Algorithm attribute.</td>
</tr>
</tbody>
</table>

3.1.6.4 XML

```xml
<xs:complexType name="DigestMethodType">
  <xs:complexContent>
    <xs:extension base="ds:DigestMethodType">
      <xs:attribute name="Id" type="xs:ID" use="required"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

3.1.7 complexType core:DigestValueType

3.1.7.1 Description [3]

DigestValueType is derived by extension from XML Signature schema. It is used as a convenience for deriving other types (such as HashType) that may be extended or restricted with other attributes.

DigestValueType is a xs:base64binary containing the result of a cryptographic digest operation.

3.1.7.2 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Document unique record instance identifier. Id is used to reference instances of hash algorithms that may be in use by a bounding document.</td>
</tr>
<tr>
<td>AlgRef</td>
<td>AlgRef refers to a hash algorithm as defined by DigestMethodType.</td>
</tr>
<tr>
<td>TransformRefs</td>
<td>Refers to transformation functions defined by TransformMethod elements of type TransformMethodType.</td>
</tr>
</tbody>
</table>
3.1.8 complexType core:HashType

3.1.8.1 Description [3]

HashType extends DigestValueType appending the ExtendOrder attribute. ExtendOrder is used to identify the sequence in which documents are extended (hashed). AlgRef identifies the hash algorithm used. TransformRefs identifies transformation algorithms that are applied to the document prior to applying the hash algorithm.

3.1.8.2 Diagram

```
namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#

type extension of core:DigestValueType

properties
  base DigestValueType

used by RimmType/CompositeHash

attributes
  Name  Type  Use  Default  Fixed
  Id    xs:ID  required
  AlgRef xs:IDREF  required
  TransformRefs xs:IDREFS
  ExtendOrder xs:IDREFS
```

3.1.8.3 Attribute Detail

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExtendOrder</td>
<td>ExtendOrder contains an ordered list of xs:IDREF values. Values at the beginning of the list occur before values at the end. Therefore, the first entry in the list would be the first value extended, the last entry would be the last value extended.</td>
</tr>
</tbody>
</table>

3.1.8.4 XML

```
<xs:complexType name="HashType">
  <xs:simpleContent>
    <xs:extension base="core:DigestValueType">
      <xs:attribute name="Id" type="xs:ID" use="required"/>
      <xs:attribute name="AlgRef" type="xs:IDREF" use="required"/>
      <xs:attribute name="TransformRefs" type="xs:IDREFS"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```
3.1.9 complexType core:IntegrityManifestType

3.1.9.1 Description [3]

The IntegrityManifestType complex type can be used to describe integrity attributes of program code, discrete logic and packages of components. Any element that can be placed under change control is a candidate for being described using IntegrityManifestType complex type.

Elements of IntegrityManifestType include:

- ComponentID – is a unique complex identifier linking the component to a change management process.
- SignerInfo – is a signature over the Integrity Manifest. It includes information about the entity that produced the signature. A single signature may be applied.
- ConfidenceValue – contains a score as a single value aggregating several criteria for establishing a degree of assurance (or trust) that the values and assertions made by the manifest are correct.
- Collector – is a reference to the utility (component) used to construct an integrity manifest. A manifest for the Collector may be separately obtained for information relating to the environment that produced this integrity manifest. A single collector may be referenced.
- TransformMethod – contains algorithm identifiers for transforms that may have been applied prior to applying a digest method. Multiple transformation methods may be defined.
- DigestMethod – contains algorithm identifiers for hash algorithms that are used to compute message digests. Multiple digest methods may be defined.
- Values – contains integrity measurements (message digests) that pertain to this component. It is reasonable (even desirable) that schemas capturing domain specific structure should incorporate a composite hash structure that is incorporated into an instantiation of Integrity Manifest with an element of type HashType. Multiple instances of Values elements may be supplied.
- AssertionInfo – contains domain specific description of attributes affecting quality, assurance or reliability assessments, but where it isn’t possible for measurement engines to collect actual values. Multiple instances of AssertionInfo elements may be supplied.
- PlatformClass – identifies the type of platform that integrity values pertain to. In particular, the methodology for PCR allocation is specified by platform specific specifications.
- SubComponents – are references to finer grain components that make up this component.
3.1.9.2 Diagram

diagram

```
namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#

ComponentID
  type ComponentIDType

SignerInfo
  type SignerInfoType

ConfidenceValue
  type ConfidenceValue

Collector
  type ComponentRefType

TransformMethod
  type TransformMethodType

DigestMethod
  type DigestMethodType

Values
  type ValueType

AssertionInfo
  type AssertionType

PlatformClass
  type PlatformClassType

SubComponents
  type ComponentRefType
```

namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#

properties
  abstract true

children
  ComponentID SignerInfo ConfidenceValue Collector TransformMethod DigestMethod Values AssertionInfo PlatformClass SubComponents

used by
  complexType RimmType

attributes
  Name Type Use Default Fixed
  Id xs:ID required
  RevLevel xs:integer required

3.1.9.3 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Revision 1.0    FINAL    Public    Page 19 of 43
Id
Globally unique record instance identifier. Id may be used by external systems, documents and this document to reference an instance of a component structure.

RevLevel
RevLevel is a revision number (increment for more recent revision) to distinguish revisions of an integrity manifest structure. RevLevel applies to instances of integrity manifest structures having the same Id value.

3.1.9.4 XML

```xml
<xs:complexType name="IntegrityManifestType" abstract="true">
  <xs:sequence>
    <xs:element name="ComponentID" type="ComponentIDType"/>
    <xs:element name="SignerInfo" type="SignerInfoType" minOccurs="0"/>
    <xs:element name="ConfidenceValue" type="ConfidenceValueType" minOccurs="0"/>
    <xs:element name="Collector" type="ComponentRefType" minOccurs="0"/>
    <xs:element name="TransformMethod" type="TransformMethodType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="DigestMethod" type="DigestMethodType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Values" type="ValueType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="AssertionInfo" type="AssertionType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="PlatformClass" type="PlatformClassType" minOccurs="0"/>
    <xs:element name="SubComponents" type="ComponentRefType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="Id" type="xs:ID" use="required"/>
  <xs:attribute name="RevLevel" type="xs:integer" use="required"/>
</xs:complexType>
```

3.1.10 complexType core:PlatformClassType

3.1.10.1 Description [3]

PlatformClassType enumerates platform classifications as determined by the Trusted Computing Group (TCG). Platform classifications can be used to apply platform specific interpretations of integrity values and quality assertions.

PlatformClassType associates a component to a platform family or classification. The association can be used to qualify usage conventions associated with digest creation, the number of allowable digests and semantics for digest association with other components in a system.

A vendor specific classification may be provided by defining a platform identifier based on a vendor specific namespace.

3.1.10.2 Diagram

```xml
<core:PlatformClassType>
  <!-- attributes -->
</core:PlatformClassType>
```

3.1.10.3 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>A vendor specific platform classification. If the URI does not unambiguously determine the vendor, the VendorID of the ComponentID for the integrity manifest is taken to be the vendor.</td>
</tr>
</tbody>
</table>
TCG defines platform class URIs. They can be used to identify the TCG platform-specific specification that applies to the platform. In particular, it can be used to distinguish how Trusted Platform Module (TPM) resources, such as PCRs, can be interpreted.

Class Identifiers:

- http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#PC_CLIENT_X86_BIOS
  Signifies an x86-based system with BIOS-based firmware.
- http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#PC_CLIENT_X86_EFI
  Signifies an x86-based system with EFI-based firmware.

### 3.10.4 XML

```xml
<xs:complexType name="PlatformClassType">
  <xs:attribute name="Class" type="xs:anyURI" use="optional"/>
</xs:complexType>
```

### 3.11 complexType core:SignerInfoType

#### 3.11.1 Description [3]

Each SignerInfoType structure has the following structure:

- **Digital signature** – Contains the digital signature resulting from signing Integrity Metadata elements. Authority to sign is determined in large part by verifier policies. The structure is represented by the `ds:Signature` element.
- **Confidence value** – Identifies the level of confidence with which trust may be given to the integrity information assumed within the structure. Represented by the `ConfidenceValue` element.
- **SigningComponent** – Identifies the program code or logic responsible for compiling, measuring and formatting the integrity information contained within the structure. Represented by a `ComponentID` element.

#### 3.11.2 Diagram

```
http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#
ds:Signature SigningComponent

attributes
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Use</th>
<th>Default</th>
<th>Fixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DateTime</td>
<td>xs:dateTime</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nonce</td>
<td>xs:base64Binary</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```xml
<source>
<xs:complexType name="SignerInfoType">
  <xs:sequence>
    <xs:element ref="ds:Signature"/>
    <xs:element ref="core:ComponentIDType"/>
  </xs:sequence>
</xs:complexType>
```

Revision 1.0     FINAL     Public
3.1.11.3 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DateTime</td>
<td>The date and time that the signature was generated. This attribute, if specified, must be included in the signature calculation.</td>
</tr>
<tr>
<td>Nonce</td>
<td>A value obtained from a remote party that is included with a signature to guarantee freshness and to avoid replay attack. This attribute, if specified, must be included in the signature calculation.</td>
</tr>
</tbody>
</table>

3.1.11.4 XML

```xml
<xs:complexType name="SignerInfoType">
  <xs:sequence>
    <xs:element ref="ds:Signature"/>
    <xs:element name="SigningComponent" type="ComponentRefType" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="DateTime" type="xs:dateTime"/>
  <xs:attribute name="Nonce" type="xs:base64Binary"/>
</xs:complexType>
```

3.1.12 complexType core:TransformMethodType

3.1.12.1 Description [3]

The TransformMethodType is used to define an element that identifies a transformation algorithm to be applied prior to a hash computation operation.

The Id attribute is used by other elements that reference one or more transformation algorithms.

3.1.12.2 Diagram

```
namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#

type extension of ds:TransformType

properties base ds:TransformType
```
3.1.12.3 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>URI pointing to a transformation algorithm identifier</td>
</tr>
<tr>
<td>Id</td>
<td>An identifier unique to this document</td>
</tr>
</tbody>
</table>

3.1.12.4 XML

```
<xs:complexType name="TransformMethodType">
    <xs:complexContent>
        <xs:extension base="ds:TransformType">
            <xs:attribute name="Id" type="xs:ID" use="required"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

3.1.13 complexType core:ValueType

3.1.13.1 Description [3]

ValueType consist of a record identifier and an element `any##other`. It is anticipated that another schema defines integrity measurements to be included in the parent element. The Id attribute is used to uniquely identify instances of child elements included in this document.

3.1.13.2 Diagram

```
diagram

namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#
used by element core:IntegrityManifestType/Values
attributes Name Id Type xs:ID
```

3.1.13.3 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Record instance identifier of a child element whose schema definition is not in the current namespace. The Id is unique to the parent XML document.</td>
</tr>
</tbody>
</table>
3.1.14 complexType core:VendorIdType

3.1.14.1 Description [3]
The VendorIdType is used to uniquely identify a vendor, manufacturer or other entity. There are two elements (SmiVendorId and TcgVendorId) that have managed number spaces ensuring uniqueness. VendorGUID uniqueness is derived algorithmically.

Only one form of VendorID element is required by the choice. More than one VendorID elements may be specified.

A familiar name can be specified, but should not be used to establish uniqueness properties.

3.1.14.2 Diagram

3.1.14.3 Attribute Detail [3]
3.1.14.4 XML

```xml
<xs:complexType name="VendorIdType">
  <xs:choice maxOccurs="unbounded">
    <xs:element name="TcgVendorId">
      <xs:simpleType>
        <xs:restriction base="xs:normalizedString">
          <xs:maxLength value="4"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
    <xs:element name="SmiVendorId" type="xs:integer"/>
    <xs:element name="VendorGUID" type="xs:NMTOKEN"/>
  </xs:choice>
  <xs:attribute name="Name" type="xs:normalizedString"/>
</xs:complexType>
```
3.2 Elements

3.2.1 element Rimm

3.2.1.1 Description
See Section 3.1.1.

3.2.1.2 Diagram

```
namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Reference_Manifest_v1_0#

type RimmType

properties content complex
```
3.2.1.3 Attribute Details
See Section 3.1.1.

3.2.1.4 XML

```xml
<xs:element name="Rimm" type="RimmType"/>
```

3.2.2 element RimmType/CompositeHash

3.2.2.1 Description
CompositeHash is an element of RimmType that is used to contain the RM integrity measurement value. The CompositeHash is the final hash value covering the Values (digests) and assertions obtained from the AssertionInfo structure contained in the RM.

AlgRef refers to the digest algorithm that was used to generate the composite hash. TransformRefs refers to the transformation algorithms and filters used to preprocess the covered values.

3.2.2.2 Diagram

![Diagram of CompositeHash element](image_url)

```xml
namespace http://www.trustedcomputinggroup.org/XML/SHEMA/Reference_Manifest_v1_0#

type core:HashType

properties
  isRef 0 complex
  content

attributes
  Name Type Use Default Fixed
  Id xs:ID required
  AlgRef xs:IDREF required
  TransformRefs xs:IDREFS required
  ExtendOrder xs:IDREFS required
```

Revision 1.0

FINAL Public
### 3.2.2.3 Attribute Detail

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Document unique record instance identifier. Id is used to reference instances of hash algorithms that may be in use by a bounding document.</td>
</tr>
<tr>
<td>AlgRef</td>
<td>AlgRef refers to a hash algorithm as defined by DigestMethodType.</td>
</tr>
<tr>
<td>TransformRefs</td>
<td>Refers to transformation functions defined by TransformMethod elements of type TransformMethodType.</td>
</tr>
<tr>
<td>ExtendOrder</td>
<td>ExtendOrder contains an ordered list of xs:IDREF values. Values at the beginning of the list occur before values at the end. Therefore, the first entry in the list would be the first value extended, the last entry would be the last value extended.</td>
</tr>
</tbody>
</table>

### 3.2.2.4 XML

```xml
<xs:element name="CompositeHash" type="core:HashType" minOccurs="0" maxOccurs="unbounded"/>
```

### 3.2.3 element core:ComponentIDType/VendorID

#### 3.2.3.1 Description [3]

The VendorIdType is used to uniquely identify a vendor, manufacturer or other entity. There are two elements (SmiVendorId and TcgVendorId) that have managed number spaces ensuring uniqueness. VendorGUID uniqueness is derived algorithmically.

Only one form of VendorID element is required by the choice. More than one VendorID elements may be specified.

A familiar name can be specified, but should not be used to establish uniqueness properties.

#### 3.2.3.2 Diagram

```
VendorIdType

attributes
Name

type xs:normalizedString

TcgVendorId
type xs:normalizedString

derivedBy restriction
minLen/maxLen

SmiVendorId
type xs:integer

VendorGUID
type xs:NMTOKEN

VendorId

type VendorIdType

```

namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#
3.2.3.3 Attribute Detail [3]

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Familiar name associated with the component manufacturer or vendor</td>
</tr>
</tbody>
</table>

3.2.3.4 XML

```xml
<xs:element name="VendorID" type="VendorIdType"/>
```

3.2.4 element core:ComponentRefType/ComponentID

3.2.4.1 Description

See Section 3.1.3 for description.

```
<component>ComponentID</component>
```

```
<attributes name="VendorID">VendorIdType
```

```
<core:ComponentIDType>
```
3.2.4.2 Attribute Details
See Section 3.1.3 for more attribute details.

3.2.4.3 XML

```
<xsl:element name="ComponentID">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="ComponentIDType"/>
    </xs:complexContent>
  </xs:complexType>
</xsl:element>
```

3.2.5 element core:ComponentRefType/ComponentIDREF

3.2.5.1 Description
ComponentIDREF element is a reference to a ComponentID within the current document.

3.2.5.2 Diagram

```
<Diagram>
  <ComponentIDREF type="xs:IDREF"/>
</Diagram>
```

3.2.5.3 XML

```
<xsl:element name="ComponentIDREF" type="xs:IDREF"/>
```

3.2.6 element core:IntegrityManifestType/ComponentID

3.2.6.1 Description
ComponentID is an element of type RimmType that is used to create RM structures, which in turn hold metadata information pertaining to the component in question.

The ID is actually made-up of the concatenation of the manufacturer, model, number and patch level (in a string format). This allows a verification entity to perform a string-match between an ID found in an Integrity Report against one found in a RM record. Note that both a reporting and verifying entity can still look-up the actual entries in the (optional) fields (eg. ModelName, VersionMajor, etc).

There is a single ComponentID element in an Integrity Manifest [3].
3.2.6.2 Diagram

```
diagram

coreComponentIDType

attributes

  id
    type xsID
    use required

  SimpleName
    type xsnormalizedString
    use optional

  ModelName
    type xsnormalizedString
    use optional

  ModelNumber
    type xsnormalizedString
    use optional

  ModelSerialNumber
    type xsnormalizedString
    use optional

  ModelSystemClass
    type xsnormalizedString
    use optional

  VersionMajor
    type xsInteger
    use optional

  VersionMinor
    type xsInteger
    use optional

  VersionBuild
    type xsInteger
    use optional

  VersionString
    type xsnormalizedString
    use optional

  EffectiveDate
    type xsdateTime
    use optional

  PatchLevel
    type xsnormalizedString
    use optional

  DiscretePatches
    type xsint64
    use optional

VendorID

namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#

type core:ComponentIDType

properties

  isRef 0

  content complex

children VendorID
```
### 3.2.6.3 Attribute Details

See Section 3.1.3.

### 3.2.6.4 XML

```
source <xs:element name="ComponentID" type="ComponentIDType"/>
```

### 3.2.7 element core:IntegrityManifestType/SignerInfo

#### 3.2.7.1 Description [3]

The SignerInfo element contains a single signature over the Integrity Manifest. The signer may provide a confidence value and reference the component used to apply the signature. The signature may also include a timestamp supplied by the signer or a nonce supplied by a verifier.

#### 3.2.7.2 Diagram

![Diagram of core:SignerInfoType](attachment:image)
3.2.7.3  XML

source  
<xs:element name="SignerInfo" type="SignerInfoType" minOccurs="0"/>

3.2.8  element core:IntegrityManifestType/ConfidenceValue

3.2.8.1  Description core:IntegrityManifestType/ConfidenceValue

The ConfidenceValue element is a score given to the signed manifest describing the level of trust the signer has attributed to integrity values included in the signature.

If the signer determines that confidence can be described in terms of levels and there are four possible levels then the first level could have a score of (1) with a basis of (4). Alternatively, a score of (25) would have a basis of (100).

If specified, this value must be included in the signature computation.

Basis values MUST be greater than 0.

3.2.8.2  Diagram

diagram

namespace  http://www.trustedcomputinggroup.org/XML/SHEMA/Core_Integrity_v1_0_1#

type  core:ConfidenceValueType

properties  isRef 0 complex

attributes  Name Type Use Default Fixed
Score type xs:integer use required
Basis type xs:integer use required
Authority type xs:anyURI use optional
3.2.8.3 Attribute Details
See Section 3.1.5.

3.2.8.4 XML

source
<xs:element name="ConfidenceValue" type="ConfidenceValueType" minOccurs="0"/>

3.2.9 element core:IntegrityManifestType/Camcorder

3.2.9.1 Description
The Collector element contains information about the component used to construct the integrity manifest. If the signerInfo/SigningComponent element is the same as the Collector element, the Collector element may be omitted [3].

The Collector is the tool (e.g., software and hardware) used to collect measurements. The structure is general enough to be used to describe a collecting software at a client (to generate an Integrity Report) – within runtime measurements context -- as well as the software-tool used by a manufacturer or third party to create RM records – within static out-of-band measurements collecting.

It is described as componentRef type since it is described as another ComponentID structure. That is, the tool itself is described as a component type in order to allow is manufacturer information and other metadata to be captured and be made accessible to a verifier entity.

ComponentLoc is an optional URI pointing to an external location of the component structure.

3.2.9.2 Diagram
3.2.9.3 XML

```xml
source <xs:element name="Collector" type="ComponentRefType" minOccurs="0"/>
```

3.2.10 element core:IntegrityManifestType/TransformMethod

3.2.10.1 Description [3]
The TransformMethod element identifies a filtering algorithm applied prior to generating a digest value.

3.2.10.2 Diagram

```
namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#
type core:TransformMethodType
properties isRef 0 complex
children XPath
attributes Name Type xs:anyURI Use required Default required Id xs:ID use required

3.2.10.3 Attribute Details
See Section 3.1.12.

3.2.10.4 XML

```xml
source <xs:element name="TransformMethod" type="TransformMethodType" minOccurs="0" maxOccurs="unbounded"/>
```
3.2.11 element core:IntegrityManifestType/DigestMethod

3.2.11.1 Description
The DigestMethod element is defined by the DigestMethodType complex type [3].

DigestMethod is an element of RimmType that is used to describe the cryptographic hash algorithm that is used to create the RM digests.

RM issuers may capture reference measurements using multiple hash algorithms. Each algorithm used will have a DigestMethod to identify the algorithm. The DigestMethod:Id can be used to reference the algorithm from the CompositeHash structure.

3.2.11.2 Diagram

![Diagram of DigestMethodType]

3.2.11.3 Attribute Details
See Section 3.1.6.

3.2.11.4 XML

```
<xs:element name="DigestMethod" type="DigestMethodType" minOccurs="0" maxOccurs="unbounded"/>
```

3.2.12 element core:IntegrityManifestType/Values

3.2.12.1 Description
The Values element in IntegrityManifestType is defined by ValueType complex type [3].

Values is an element of RimmType that contains the digest value of a measured component’s objects or images.
3.2.12.2 Diagram

3.2.12.3 Attribute Details
See Section 3.1.13.

3.2.12.4 XML

    source <xs:element name="Values" type="ValueType" minOccurs="0" maxOccurs="unbounded"/>

3.2.13 element core:IntegrityManifestType/AssertionInfo

3.2.13.1 Description
The AssertionInfo element in IntegrityManifestType is defined by AssertionInfoType complex type [3].

AssertionInfo is an element of RimmType that contains assertions relating to the components and the values captured in the RM.

3.2.13.2 Diagram
3.2.13.3 Attribute Details
See Section 3.1.2.

3.2.13.4 XML

```
<xs:element name="AssertionInfo" type="AssertionType" minOccurs="0" maxOccurs="unbounded"/>
```

3.2.14 element core:IntegrityManifestType/PlatformClass

3.2.14.1 Description
The PlatformClass element in IntegrityManifestType is of type PlatformClassType [3].

3.2.14.2 Diagram

```
namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#
type core:PlatformClassType

<table>
<thead>
<tr>
<th>properties isRef</th>
<th>content</th>
<th>0 complex</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes Name</td>
<td>Type</td>
<td>Use</td>
</tr>
<tr>
<td></td>
<td>xs:ID</td>
<td></td>
</tr>
<tr>
<td></td>
<td>xs:anyURI</td>
<td>optional</td>
</tr>
</tbody>
</table>
```

3.2.14.3 XML

```
<xs:element name="PlatformClass" type="PlatformClassType" minOccurs="0" maxOccurs="unbounded"/>
```

3.2.15 element core:IntegrityManifestType/SubComponents

3.2.15.1 Description
The SubComponents element identifies components of a system that are a decomposition of this component. An arbitrary nesting of subcomponents can be described if the referenced subcomponent is itself an element of type IntegrityManifestType [3].

A given component may in fact be composed of more than one identifiable subcomponent. That is, a component (e.g. NIC card) may consists of subcomponents (e.g. TPM within the NIC card).
Thus, the RM schema must support the case where this is case. Note that the element points to further ComponentIDs.

### 3.2.15.2 Diagram

![Diagram](http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#)

- **namespace**: http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#
- **type**: core:ComponentRefType
- **properties**:
  - `isRef`: 0
  - `content`: complex
- **children**: ComponentID, ComponentIDREF
- **attributes**:
  - **Name**
  - **Type**: xs:anyURI
  - **Use**: optional

### 3.2.15.3 Attribute Details

See Section 3.1.3.

### 3.2.15.4 XML

```xml
<xs:element name="SubComponents" type="ComponentRefType" minOccurs="0" maxOccurs="unbounded"/>
```

### 3.2.16 element core:SignerInfoType/SigningComponent

#### 3.2.16.1 Description

The SigningComponent element identifies the tool that was used to generate the signed manifest. The signature over the manifest should include the SigningComponent element. Signing component is a reference to a document that may exist external to this document. The integrity values for signing component are not contained in the SigningComponent element. If specified, this value must be included in the signature computation [3].
3.2.16.2 Diagram

```
<xs:element name="SigningComponent" type="ComponentRefType" minOccurs="0"/>
```

3.2.16.3 Attribute Details
See Section 3.1.11.

3.2.16.4 XML

```
<TcgVendorId
     type="xs:normalizedString"
     derivedBy="restriction"
     minOccurs="4"
/>
```

3.2.17 element core:VendorIdType/TcgVendorId

3.2.17.1 Description
This is the vendor Id issued by the TCG. It is used to uniquely identify the party responsible for applying change management to the component. Typically this is the component manufacturer or IT.

3.2.17.2 Diagram
3.2.17.3 XML

source
<xs:element name="TcgVendorId">
<xs:simpleType>
<xs:restriction base="xs:normalizedString">
<xs:maxLength value="4"/>
</xs:restriction>
</xs:simpleType>
</xs:element>

3.2.18 element core:VendorIdType/SmiVendorId

3.2.18.1 Description
This is a vendor Id corresponding to an SMI Network Management Private Enterprise Code
issued by the Internet Assigned Number Authority (IANA). It is used to uniquely identify the party
responsible for applying change management to the component. Typically this is the component
manufacturer or IT.

3.2.18.2 Diagram

diagram

namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#

type xs:integer

properties
isRef 0
content simple

3.2.18.3 XML

source
<xs:element name="SmiVendorId" type="xs:integer"/>

3.2.19 element core:VendorIdType/VendorGUID

3.2.19.1 Description
VendorGUID is used to uniquely identify the party responsible for applying change management
to the component. Typically this is the component manufacturer or IT.

3.2.19.2 Diagram

diagram

namespace http://www.trustedcomputinggroup.org/XML/SCHEMA/Core_Integrity_v1_0_1#

type xs:NMTOKEN

properties
isRef 0
content simple
3.2.19.3 XML

source  <xs:element name="VendorGUID" type="xs:NMTOKEN"/>
4 References


