kiuru™ by methics

Kiuru MSSP 5.0

Metadata Specification

Document version 1.0

Change History

Version	Date	Changes
1.0	2014-01-03	Initial public version

Methics Oy
Vuorikatu 16
FIN-00100 Helsinki
Finland

Business ID: 1778566-8

Table of Contents

1 Introduction

A mobile signature service needs agreements between mobile signature system entities. These entities must share system identifiers, binding support, services, certificates and keys, and so on. A MSSP metadata specification is useful for describing this information in a commonly interpreted way. This document defines a Methics' proprietary metadata format for mobile signature service system entities and their roles in mobile signature service system. Such roles include Mobile Signature Service Providers (MSSP) like Acquiring Entities, Home MSSPs, Identity Issuers, Verifying Entities and Application Provider role.

This is a draft specification produced by Methics Oy. The schema has been tested with a Kiuru MSSP system deployment. You can submit any comments and errata to the <mssp.metadata@methics.fi> address.

1.1 Typographic Conventions

Example	Description
`conf/example.conf`	File names are shown in mono-spaced font.
`$ export`	Typed commands are shown in mono-spaced font and preceded by a dollar-sign or other prompt. If the prompt is "#", it refers to a super-user executable command.
`kiuru.service.connections = 800`	Configuration examples are shown in a mono-spaced font, just like typed commands. Due to typographic reasons, texts have sometimes folded on multiple lines and there might either be no indication of this, or a backslash "\" at the end of the row highlighting that the text contines on the next row.

`$ export …`
`$ cp …`
`$ restart …`
	Configuration and command examples which span several rows are highlighted with gray background color.

Note! *Remember to back up your data before attempting this!*	Notes requiring special attention are shown in italic and are preceded by the word 'Note!' in bold.
...see chapter *15 Configuration Options* for more info...	References to other sections, documents or other resources are shown in italics.

2 Overview

2.1 Introduction to MSSP Mesh

ETSI MCOMM standards based mobile signature service consists of mobile signature service providers (MSSP) and application providers. MSSP entities have different roles such as Acquiring Entity, Routing Entity Identity Issuer and Home MSSP. Connected MSSPs constitute a mobile signature service system known as a Mesh. MSSPs connect to Service Providers (SP) that offer additional services or registration services. The Mesh has been defined in the ETSI TS 102 207 standard (figure 1). This is also known as a mobile signature roaming.

Security of the mobile signature roaming is based on shared communication addresses, mutual TLS authentication of the entities and a message integrity based on XML signatures. This trust model is based on X509 certificates. These certificates must be shared between entities in a secure way and a life cycle of these certificates needs to be taken care of. We need a way of describing and managing this information in a commonly interpreted way.

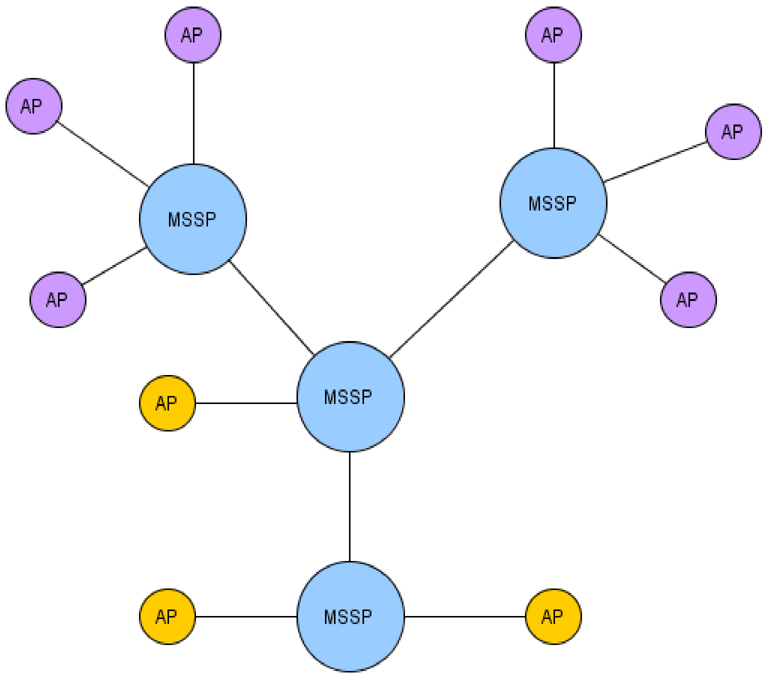

Figure 1: Connected MSSPs Constitute a Mesh

2.2 Introduction to MSSP metadata

The MSSP metadata defines an XML file that contains mobile signature system entities and the security and communication parameters between these entities. By publishing MSSP metadata MSSPs not only extend the Mesh security but also make it easier to manage. The MSSP metadata replaces ETSI TS 102 204 standard MSS HandshakeService method in a secure way.

The MSSP metadata defines XML signature and TLS/SSL metadata among system entities. The MSSP metadata also introduces XML encryption metadata support for mobile signature services. XML encryption can be used especially by a mobile signature registration method when distributing confidential data over the Internet.

Public MSSP metadata is generally needed when establishing large interconnected Mesh on national level. MSSP metadata blocks can be digitally signed and verified. The mechanisms help in establishing trust in the accuracy and authenticity of MSSP metadata.

The base or reference model for this metadata is the Metadata for the OASIS Security Assertion Markup Language V2.0 (SAML2). If you are familiar with the SAML2 metadata, you should find that fundaments of MSSP metadata have been organized in the same way.

3 Metadata for MSSP

3.1 Metadata Schema Structure

MSSP Metadata schema is organized under a top level EntitiesDescriptor element which contains a collection of sub level EntitiesDescriptors and EntityDescriptors. EntityDescriptor defines an entity derived from the mobile signature service MSSP_ID and AP_ID. Each Entity may be collected into a new EntitiesDescriptor container element. We can say that the EntitiesDescriptor is the primary unit of MSSP metadata (figure 2).

Sub level EntitiesDescriptor can be used in large MSSP networks to manage separate mobile signature subnets. For example each mobile operator may have their own EntitiesDescriptor that contains only one operator's VE, AEs and APs. EntityDescriptors contain an MSSPDescriptor, an APDescriptor or a SPDescriptor.

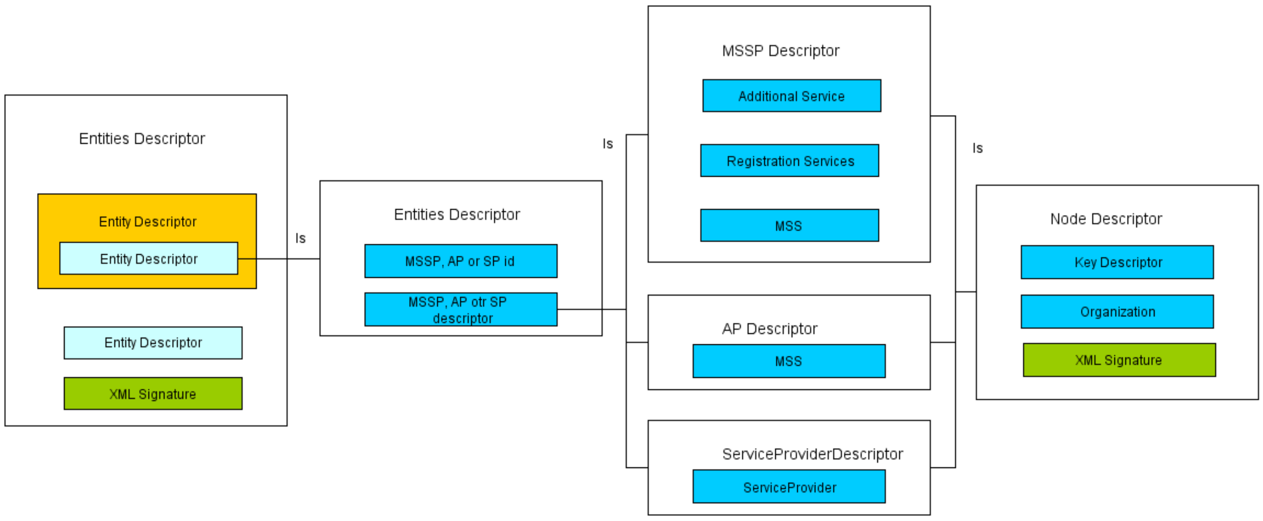

Figure 2: MSSP Metadata Schema Illustration

The MSSP metadata supports certificate lifecycle management. Both EntitiesDescriptor and EntityDescriptor may have attributes "ValidFrom" and "ValidUntil" to inform about when the data is valid. For example an Entity may have multiple certificates to support simultaneously currently used certificate and a coming new certificate to enable smooth transition to new certificate without need for communicating parties to synchronic update certificates. See discussion at chapter 4.3.

3.2 Namespaces

MSSP Metadata uses the following namespaces (defined in the schema mssp-metadata-1.0.xsd). This paper uses the namespace prefix "mmd:" to refer to the mssp metadata namespace.

```
mmd=http://www.methics.fi/MSSPMetadata/v1.0.0#
```

The namespace literal "mss:" refers to the following [ETSI204] namespace:

```
mss=http://uri.etsi.org/TS102204/v1.1.2#
```

The rest of the literals and namespaces are as follows:

```
xs=http://www.w3.org/2001/XMLSchema
ds=http://www.w3.org/2000/09/xmldsig#
xenc=http://www.w3.org/2001/04/xmlenc#
```

In narrative texts we have omitted namespace prefixes in this document when they are "mmd:". Schema fragments are taken from the mssp-metadata-1.0.xsd as is, and full schema file is attached on as Appendix C.

3.3 Mobile Signature Service Elements and Types

3.3.1 MSS Elements

MSS service elements define service URIs that provide Mobile Signature methods. These URIs are used as a SOAP service address when requesting a specific MSS method.

```
<xs:element ref="MSS_SignatureService"         minOccurs="0" maxOccurs="1"/>
<xs:element ref="MSS_NotificationService"      minOccurs="0" maxOccurs="1"/>
<xs:element ref="MSS_ProfileQueryService"       minOccurs="0" maxOccurs="1"/>
<xs:element ref="MSS_ReceiptService"           minOccurs="0" maxOccurs="1"/>
<xs:element ref="MSS_StatusService"            minOccurs="0" maxOccurs="1"/>
<xs:element ref="MSS_HandshakeService"          minOccurs="0" maxOccurs="1"/>
<xs:element ref="MSS_RegistrationService"      minOccurs="0" maxOccurs="1"/>

<xs:element name="MSS_SignatureService"         type="xs:anyURI"/>
<xs:element name="MSS_RegistrationService"      type="xs:anyURI"/>
<xs:element name="MSS_NotificationService"      type="xs:anyURI"/>
<xs:element name="MSS_ProfileQueryService"      type="xs:anyURI"/>
<xs:element name="MSS_ReceiptService"           type="xs:anyURI"/>
<xs:element name="MSS_StatusQueryService"       type="xs:anyURI"/>
<xs:element name="MSS_HandshakeService"         type="xs:anyURI"/>
```

3.3.2 Service Elements

Service elements define additional services or registration services that MSSP is able to provide.

```
<xs:element ref="mmd:AdditionalServices" minOccurs="0" maxOccurs="1"/>
<xs:element ref="mmd:RegistrationServices" minOccurs="0" maxOccurs="1"/>
```

3.3.3 Element <EntityID>

Unique EntityID element identifies an entity which is a MSSP, an AP or a SP. The complexType EntityIDType contains either MSSP_ID, AP_ID or SP_ID. <EntityID> is used as a unique identifier for entities. MSSP_ID and AP_ID has been defined in ETSI204 as follows:

```
MSSP_ID MessageAbstractType:MSSP_Info:MSSP_ID
AP_ID  MessageAbstractType:AP_Info:AP_ID
```

SP_ID is an unique URI identifier for a service provider that provides additional services or registration services.

```
SP_ID Unique URI identifier
```

Note: *SAML2 metadata restricts the EntityIDType (anyURI) to a maximum length of 1024* *characters. This version of the MSSP metadata does not do that but in reality we should do that.*

An <EntityID> must be unique across all entities that has been defined in a metadata of a given deployment. This means that all <AP_ID>s and <SP_ID>s must be unique and all <MSSP_ID>s must be unique.

The following schema fragment defines the EntityIDType complex type:

```
<xs:element name="EntityID" type="mmd:EntityIDType"/>
<xs:complexType name="EntityIDType">
   <xs:sequence>
       <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element name="MSSP_ID" type="mss:MeshMemberType" minOccurs="0"/>
        <xs:element name="AP_ID"   type="xs:anyURI"          minOccurs="0"/>
       <xs:element name="SP_ID"   type="xs:anyURI"          minOccurs="0"/>
     </xs:choice>
   </xs:sequence>
</xs:complexType>
```

Auxiliary complex type MeshMemberType from ETSI204

MeshMemberType is used to denote an MSSP entity, e.g. an Identity Issuer or a Home MSSP. This can be done by means of a DNS-Name, an IP Address, a string that identifies the entity or a URI. The following schema block is snipped from ETSI204 for information only.

```
<xs:complexType name="MeshMemberType">
      <xs:sequence>
            <xs:element name="DNSName"        type="xs:string" minOccurs="0"/>
            <xs:element name="IPAddress"            type="xs:string" minOccurs="0"/>
            <xs:element name="URI"            type="xs:anyURI" minOccurs="0"/>
            <xs:element name="IdentifierString" type="xs:string" minOccurs="0"/>
      </xs:sequence>
</xs:complexType>
```

3.3.4 Simple type AnyURIListType

A simple type AnyURIListType is used for creating an URI list.

```
<xs:simpleType name="AnyURIListType">
      <xs:list itemType="xs:anyURI"/>
</xs:simpleType>
```

3.3.5 ComplexType ExtensionsType

A complex type ExtensionsType is used for any schema extensions. These extensions must use qualified namespace and the namespace must be other than MSSP metadata namespace.

```
<xs:complexType final="#all" name="ExtensionsType">
      <xs:sequence>
            <xs:any namespace="##other" processContents="lax"
maxOccurs="unbounded"/>
      </xs:sequence>
</xs:complexType>
```

3.3.6 Element <AdditionalServices>

This defines a list of AdditionalServices that MSSP implements. The AdditionalServices are used to tell that the AP requests not only a signature of the mobile user but also a validation of user's mobile signature, a timestamp or an archiving of this signature or anything else. Therefore the corresponding additional service is specified by using an URI.

`AdditionalService` [Optional] This element indicates AdditionalService that this MSSP implements.

`Description` [Optional] This sub-element indicates an AdditionalService URI.

`EntityId` [Optional] This sub-element indicates the additional service provider.

```
<xs:element name="AdditionalServices">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="AdditionalService"
                        type="mmd:AdditionalServiceType"
                        minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
      </xs:complexType>
</xs:element>

<xs:complexType name="AdditionalServiceType">
    <xs:sequence>
        <xs:element name="Description" type="xs:anyURI"/>
        <!-- Additional service provider is identified by SP_ID -->
        <xs:element ref="mmd:EntityID"/>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:any namespace="##other" processContents="lax" />
            </xs:choice>
        </xs:sequence>
</xs:complexType>
```

This is used as a part of MSSPDescriptor, and an example fragment of metadata is following:

```
<mmd:AdditionalServices>
    <mmd:AdditionalService>
        <mmd:Description>
          http://www.example.com/MSSP/v3.4.0#XML-Signature
        </mmd:Description>
    </mmd:AdditionalService>
    <mmd:AdditionalService>
        <mmd:Description>
          http://uri.etsi.org/TS102204/v1.1.2#validate
        </mmd:Description>
    </mmd:AdditionalService>
</mmd:AdditionalServices>
```

3.3.7 Element <RegistrationServices>

This defines a list of RegistrationServices that MSSP implements. The RegistrationServices are used for registration authorities where to send registration requests. Therefore the corresponding registration service is specified by using an URI.

`RegistrationService` [Optional] This element indicates RegistrationService that this MSSP implements.

`EntityId` [Optional] This sub-element indicates the registration service provider.

```
<xs:element name="RegistrationServices">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="RegistrationService"
                        type="mmd:RegistrationServiceType"
                        minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
    </xs:complexType>
</xs:element>

<xs:complexType name="RegistrationServiceType">
    <xs:sequence>
        <!-- Registration service provider is identified by SP_ID -->
        <xs:element ref="mmd:EntityID"/>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:any namespace="##other" processContents="lax" />
        </xs:choice>
    </xs:sequence>
</xs:complexType>
```

3.3.8 Element <MSSPDescriptor>

The MSSPDescriptor element provides information about the Mobile Signature Service Provider that is connected in the Mesh. The complex MSSPDescriptorType type extends NodeDescriptorType with several attributes special for a MSSP.

Note: There are not attribute defining MessagingModes while MSSP must support the three messaging modes defined in ETSI204.

`MSS_SignatureService` [Optional] defines MSSP's mobile signature service call address point as defined in ETSI204.

`MSS_RegistrationService` [Optional] defines MSSP's mobile signature service registration call address point as defined in ETSI204.

`MSS_NotificationService` [Optional] defines MSSP's mobile signature service notificationcall address point as defined in ETSI204.

`MSS_ProfileQueryService` [Optional] defines MSSP's mobile signature service profile query call address point as defined in ETSI204.

`MSS_ReceiptService` [Optional] defines MSSP's mobile signature service receipt call address point as defined in ETSI204.

`MSS_StatusService` [Optional] defines MSSP's mobile signature service status call address point as defined in ETSI204.

`MSS_HandshakeService` [Optional] defines MSSP's mobile signature service handshake call address point as defined in ETSI204.

`Roles` [One or more] defines a non empty list of MSSP's role URIs. This paper assumes the following concrete roles:

The following schema fragment defines the MSSPDescriptor and MSSPDescriptorType:

```
<xs:element name="MSSPDescriptor" type="mmd:MSSPDescriptorType"/>

<xs:complexType name="MSSPDescriptorType">
      <xs:complexContent>
            <xs:extension base="mmd:NodeDescriptorType">
            <xs:sequence>
                  <xs:element ref="mmd:MSS_SignatureService”      minOccurs="0"
                        maxOccurs="1"/>
                  <xs:element ref="mmd:MSS_RegistrationService"   minOccurs="0"
                        maxOccurs="1"/>
                  <xs:element ref="mmd:MSS_NotificationService"   minOccurs="0"
                        maxOccurs="1"/>
                  <xs:element ref="mmd:MSS_ProfileQueryService"   minOccurs="0"
                        maxOccurs="1"/>
                  <xs:element ref="mmd:MSS_ReceiptService"  minOccurs="0"
                        maxOccurs="1"/>
                  <xs:element ref="mmd:MSS_StatusService"   minOccurs="0"
                        maxOccurs="1"/>
                  <xs:element ref="mmd:MSS_HandshakeService"      minOccurs="0"
                        maxOccurs="1"/>
                  <xs:element ref=”mmd:AdditionalServices”  minOccurs=”0”
                        maxOccurs=”1”/>
                  <xs:element ref="mmd:RegistrationServices"    minOccurs="0"
                        maxOccurs="1"/>
            </xs:sequence>
            <xs:element name="Roles" type="mmd:AnyURIListType"
                         use="required"/>
            <xs:attribute name="SignatureProfiles" type="mmd:AnyURIListType"
                        use="optional"/>
         <xs:attribute name="SignatureProfileComparison" type="xs:boolean"
          use="optional"/>
            <xs:attribute name="MSSFormats" type="mmd:AnyURIListType"
                        use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
      </xs:complexContent>
</xs:complexType>
```

3.3.9 Element <APDescriptor>

The APDescriptor element provides information about the Application Provider that is connected in the Mesh. The complex APDescriptorType type extends NodeDescriptorType with a pair of attributes special for an AP.

`MSS_NotificationService`	[Optional] defines AP's mobile signature service Server-server asynchronous notification service call address point as defined in ETSI204.

`DisplayName` [Optional] A language-qualified name that is suitable for human consumption. AP's display name should be equal with the ETSI204 mobile signature request's AP name that mobile user receives.

The following schema fragment defines the APDescriptor and APDescriptorType:

```
<xs:element name="APDescriptor" type="mmd:APDescriptorType"/>

<xs:complexType name="APDescriptorType">
  <xs:complexContent>
    <xs:extension base="mmd:NodeDescriptorType">
      <xs:sequence>
        <xs:element ref="mmd:MSS_NotificationService" minOccurs="0"
                maxOccurs="1"/>
        <xs:element name="DisplayName" type="xs:string" minOccurs="0"
                maxOccurs="1"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

3.3.10 Element <ServiceProviderDescriptor>

The ServiceProviderDescriptor element provides information about a service provider (application provider or CA) that is connected to the Mesh. The complex ServiceDescriptorType type extends NodeDescriptorType with one or zero ServiceProvider.

`ServiceProvider`	[zero or one] defines AP's mobile signature service Server-server asynchronous notification service call address point as defined in ETSI204.

The following schema fragment defines the ServiceProviderDescriptor and ServiceProviderDescriptorType:

```
<xs:element name="ServiceProviderDescriptor"
            type="mmd:ServiceProviderDescriptorType"/>
<xs:complexType name="ServiceProviderDescriptorType">
      <xs:complexContent>
            <xs:extension base="mmd:NodeDescriptorType">
                  <xs:sequence>
                        <xs:element name="ServiceProvider"
                                    type="mmd:ServiceProviderType"
                                    minOccurs="0" maxOccurs="1"/>
                  </xs:sequence>
            </xs:extension>
      </xs:complexContent>
</xs:complexType>
```

ServiceProviderDescriptor uses the following types:

`ServiceProviderType`	contains the services that the service provider provides. `ServiceURL in ServiceProviderType denotes the URL the service is provided at.`
`ServiceType`	defines service specific parameters.

```
<xs:complexType name="ServiceProviderType">
      <xs:sequence>
            <xs:element name="Service" type="ServiceType"  minOccurs="0"
                                  maxOccurs="unbounded"/>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
                        maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<xs:complexType name="ServiceType">
      <xs:sequence>
            <xs:element name="Description" type="xs:anyURI" minOccurs="0"
                        maxOccurs="1"/>
            <xs:any namespace="##other" processContents="lax" />
      </xs:sequence>
      <xs:attribute name="Type" type="mmd:ServiceSimpleType"/>
      <xs:attribute name="ServiceURL" type="xs:anyURI" minOccurs="0"
maxOccurs="1"/>
</xs:complexType>
```

`ServiceSimpleType`	defines a simple plain name for the service type. This is used to categorize services. CA, AdditionalService, DSS etc.

```

<xs:simpleType name="ServiceSimpleType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="ca"/>
            <xs:enumeration value="additionalservice"/>
            <xs:enumeration value="dss"/>
            <xs:enumeration value="spml2"/>
            <xs:enumeration value="saml2"/>
            <xs:enumeration value="cmphttp"/>
            <xs:enumeration value="xkms"/>
            <xs:enumeration value="other"/>
        </xs:restriction>
</xs:simpleType>
```

3.3.11 Element <ClientDescriptior>

The ClientDescriptor element provides information about a client of the MSSP (for example an application provider). The complex ClientDescriptorType type extends NodeDescriptorType with one or zero Client.

`Client_ID` [one] defines Client identifier.

`DisplayName`	defines a verbose name fot the Client.
`ServiceProvider`	`defines` Service Provider of this Client.
`Certs`	defines Certificates of this Client.
`Wants`	defines secure protocols this Client wants.

The following schema fragment defines the ClientDescriptor and ClientDescriptorType:

```
    <xs:element name="ClientDescriptor" type="mmd:ClientDescriptorType"/>
    <xs:complexType name="ClientDescriptorType">
        <xs:complexContent>
            <xs:extension base="mmd:NodeDescriptorType">
                <xs:sequence>
```

```
                    <xs:element name="Client_ID"  type="xs:anyURI"
                                                  minOccurs="1"/>
                    <xs:element name="DisplayName"
                                type="mmd:DisplayNameType"
                                minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="ServiceProvider"
                                type="mmd:ServiceProviderType"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:Certs" minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:Wants" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

3.4 Root Elements

3.4.1 Element <EntitiesDescriptor>

MSSP Metadata may contain only one top level <EntitiesDescriptor> element.

```
<xs:element name="EntitiesDescriptor" type="mmd:EntitiesDescriptorType"/>
```

The top level EntitiesDescriptor element contains a sequence of <EntityDescriptor> elements, <EntitiesDescriptor> elements, or both.

`EntitiesDescriptor` or `EntityDescriptor`	[One or More] This element contains the metadata for one or more MSSP or AP entities, or a nested group of metadata. EntitiesDescriptorType consists of set of EntityDescriptor and EntitiesDescriptor and attributes. EntitiesDescriptorType element may contain XML signature.
`ID` [Optional]	A document-unique identifier for the element. ID is used as a XML signature reference point and value therein must be unique within the document.
`ValidFrom` [Optional]	This attribute indicates the begin time of the metadata contained in the element and it's sub elements. It is recommended that only the bottom EntitiesDescriptor element of a metadata instance contain the attribute, if any at all.
`ValidUntil` [Optional]	This attribute indicates the expiration time of the metadata contained in the element and it's sub elements. It is recommended that only the top EntitiesDescriptor element of a metadata instance contain the attribute.
`Name` [Optional]	A string name that identifies a group of MSSP entities and APs in the metadata document.
`DistributionURL`	[Optional] An URL defining a distribution location point for this MSSP EntitiesDescriptor metadata. HTTPS based URL is recommended.
`Signature` [Optional]	An XML signature that authenticates the containing element and its contents, as described in Section 3.

The following schema fragment defines the EntitiesDescriptorType complex type:

```
<xs:complexType name="EntitiesDescriptorType">
      <xs:sequence>
            <xs:element ref="ds:Signature" minOccurs="0"/>
            <xs:element ref="mmd:Extensions" minOccurs="0"
maxOccurs=”unbounded”/>
            <xs:choice minOccurs="1" maxOccurs="unbounded">
                  <xs:element ref="mmd:EntityDescriptor"/>
                  <xs:element ref="mmd:EntitiesDescriptor"/>
            </xs:choice>
      </xs:sequence>
      <xs:attribute name="ValidFrom"  type="xs:dateTime" use="optional"/>
      <xs:attribute name="ValidUntil" type="xs:dateTime" use="optional"/>
      <xs:attribute name="ID" type="xs:ID" use="optional"/>
      <xs:attribute name="DistributionURL" type="xs:URI" use=”optional” />
      <xs:attribute name="Name" type="xs:string" use="optional"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

3.4.2 Element <EntityDescriptor>

The <EntityDescriptor> element specifies metadata for a single MSSP or AP entity. A single MSSP entity may act in many different roles in the support of multiple profiles.

EntityDescriptorType complex type consists of the following elements and attributes:

`EntityID` [Required] Specifies the unique identifier of the MSSP or AP entity whose metadata is described by the element's contents.

`MSSPDescriptor`, `APDescriptor` or `ServiceProviderDescriptor` [One required] The descriptor for the MSSP, AP or SP.

`ID` [Optional] A document-unique identifier for the element, typically used as a reference point when signing and value therein must be unique within the document.

`ValidFrom` [Optional] Optional attribute indicates the beginning time of the metadata contained in the element and any contained elements.

`ValidUntil` [Optional] Optional attribute indicates the expiration time of the metadata contained in the element and any contained elements.

`Signature` [Optional] An XML signature that authenticates the containing element and its contents, as described in Section 3.

The primary content of the element is either a sequence of one or more role descriptor elements, or a specialized descriptor that defines an affiliation.

`Organization` [Optional] Optional element identifying the organization responsible for the MSSP or AP entity described by the element.

`ContactPerson` [Zero or More] Optional sequence of elements identifying various kinds of contact personnel.

`AnyAttribute` [Optional] This attribute gives the opportunity for MSSP metadata to provide other relevant entity information.

The following schema fragment defines the <EntityDescriptor> element and its EntityDescriptorType complex type:

```
<xs:element name="EntityDescriptor" type="mmd:EntityDescriptorType"/>
   <xs:complexType name="EntityDescriptorType">
      <xs:sequence>
         <xs:element ref="ds:Signature" minOccurs="0"/>
                   <xs:element ref="mmd:Extensions" minOccurs="0"/>
         <xs:element ref="mmd:EntityID"/>
         <xs:choice minOccurs="1" maxOccurs="1">
            <xs:element ref="mmd:MSSPDescriptor"/>
            <xs:element ref="mmd:APDescriptor"/>
            <xs:element ref="mmd:ServiceProviderDescriptor"/>
         </xs:choice>
         <xs:element ref="mmd:Organization" minOccurs="0"/>
         <xs:element ref="mmd:ContactPerson" minOccurs="0"
                                   maxOccurs="unbounded"/>
     </xs:sequence>
     <xs:attribute name="ValidFrom"   type="xs:dateTime" use="optional"/>
     <xs:attribute name="ValidUntil"  type="xs:dateTime" use="optional"/>
     <xs:attribute name="ID"          type="xs:ID" use="optional"/>
     <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

3.4.3 Element <NodeDescriptor>

NodeDescriptor contains the common elements and attributes for MSSP_ID and AP_ID entities.

`ValidFrom` [Optional] indicates the beginning time of the metadata contained in the element and any contained elements.

`ValidUntil` [Optional] indicates the expiration time of the metadata contained in the element and any contained elements.

`WantSSLAuthentication` [Optional] “true” indicates that the node wants all incoming HTTPS connections authenticated against contacting node's SSL certificate that must be sent on contact formation as contactee's client certificate.

`WantRequestsSigned` [Optional] “true” indicates that the node entity wants that mobile signature messages (requests and responses) are signed with XML signatures.

`WantRequestsEncrypted` [Optional] indicates if an entity wants that a mobile signature registration messages (RegistrationInput/ RegistrationOutput) are encrypted.

`AnyAttribute` [Optional] This attribute gives the opportunity for MSSP metadata to provide other relevant node information.

The following schema fragment defines the NodeDescriptor and NodeDescriptorType:

```
<xs:element name="NodeDescriptor" type="mmd:NodeDescriptorType"/>

<xs:complexType name="NodeDescriptorType" abstract="true">
```

```
    <xs:sequence>
        <xs:element ref="mmd:KeyDescriptor" minOccurs="0"
                                            maxOccurs="unbounded"/>
        <xs:element ref="mmd:Organization"  minOccurs="0"
                                            maxOccurs=”1”        />
        <xs:element ref="mmd:ContactPerson" minOccurs="0"
                                            maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="ValidFrom"              type="xs:dateTime"
                                                use="optional"/>
    <xs:attribute name="ValidUntil"             type="xs:dateTime"
                                                use="optional"/>
    <xs:attribute name="WantSSLAuthentication” type="xs:boolean"
                                                use="optional"/>
    <xs:attribute name="WantRequestsSigned"     type="xs:boolean"
                                                use="optional"/>
    <xs:attribute name="WantRequestsEncrypted" type="xs:boolean"
                                                use="optional"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

3.4.4 Element <Organization>

The <Organization> element specifies basic information about an organization responsible for a MSSP or AP entity. The use of this element is always optional but highly recommended. Its content is informative in nature and does not directly map to any core MSS elements or attributes. Its OrganizationType complex type consists of the following elements:

`OrganizationName` [One or More] Natural language organization name that is responsible for this MSSP.

`OrganizationURL` [One or More] An URL that specifies a location to which to direct a user for additional information. Note that the URI refers to the human readible material at the specified location.

`Extentions` [Optional] Placeholder for possible extentions.

`AnyAttribute` [Optional] This attribute gives the opportunity for MSSP metadata to provide other relevant organization information.

Arbitrary namespace-qualified attributes from any namespaces may also be included. The following schema fragment defines the <Organization> element and its OrganizationType complex type:

```
<xs:element name="Organization" type="mmd:OrganizationType"/>

<xs:complexType name="OrganizationType">
    <xs:sequence>
        <xs:element ref="mmd:OrganizationName"     maxOccurs="unbounded"/>
            <xs:element ref="mmd:OrganizationURL"  maxOccurs="unbounded"/>
        <xs:element ref="mmd:Extensions"        minOccurs="0" maxOccurs=”1”/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<xs:element name="OrganizationName" type="xs:string"/>
```

```
<xs:element name="OrganizationURL"  type="xs:string"/>
```

3.4.5 Element <ContactPerson>

The <ContactPerson> element specifies basic contact information about a person responsible in some capacity for a MSSP entity. The use of this element is always optional. Its content is informative in nature and does not directly map to any MSSP elements or attributes. Its ContactPersonType complex type consists of the following elements and attributes:

`Company` [Optional] String that specifies the name of the company for the contact person.

`GivenName` [Optional] String that specifies the given name of the contact person.

`SurName` [Optional] String that specifies the surname of the contact person.

`EmailAddress` [Zero or More] Zero or more elements containing mailto: URIs representing e-mail addresses belonging to the contact person.

`TelephoneNumber` [Zero or More] Specifies a telephone number (for example MSISDN) of the contact person.

`ContactType` [Required] Specifies the type of contact person using the ContactTypeType enumeration. The possible values are `technical`, `support`, `administrative`, `billing`, and `other`.

`AnyAttribute` [Optional] This attribute gives the opportunity for MSSP metadata to provide other relevant contact person information. Therefore you can include arbitrary namespace-qualified attributes from any namespaces.

The following schema fragment defines the <ContactPerson> element and its ContactPersonType complex type:

```
<xs:element name="ContactPerson" type="mmd:ContactPersonType"/>

<xs:complexType name="ContactPersonType">
      <xs:sequence>
            <xs:element ref="mmd:Company"          minOccurs="0"
                                                   maxOccurs=”1” />
            <xs:element ref="mmd:GivenName"        minOccurs="0"
                                                   maxOccurs=”1” />
            <xs:element ref="mmd:SurName"          minOccurs="0"
                                                   maxOccurs=”1” />
            <xs:element ref="mmd:EmailAddress"     minOccurs="0"
                                                   maxOccurs="unbounded"/>
            <xs:element ref="mmd:TelephoneNumber" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element ref="mmd:Extensions"       minOccurs="0"
                                                   maxOccurs=”1” />
      </xs:sequence>
   <xs:attribute name="ContactType" type="mmd:ContactTypeType"/>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<xs:element name="Company"         type="xs:string"/>
<xs:element name="GivenName"       type="xs:string"/>
<xs:element name="SurName"         type="xs:string"/>
```

```
<xs:element name="EmailAddress"    type="xs:anyURI"/>
<xs:element name="TelephoneNumber" type="xs:string"/>

<xs:simpleType name="ContactTypeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="technical"/>
    <xs:enumeration value="support"/>
    <xs:enumeration value="administrative"/>
    <xs:enumeration value="billing"/>
    <xs:enumeration value="other"/>
  </xs:restriction>
</xs:simpleType>
```

Note: We think that SAML metadata uses names like ContactType and contactType inconsistently. We do not know the reason for it and therefore have changed the type to ContactTypeType and the attribute to ContantType .

3.5 MSSP Key Management Elements

3.5.1 Element <EncryptionMethod>

The <EncryptionMethod> is an element defined in XML encryption that describes the encryption algorithm applied to the cipher data. If the element is absent, the supported encryption algorithms are not reported publically, but encryption may still be possible.

```
<xs:element name="EncryptionMethod" type="xenc:EncryptionMethodType"/>
```

3.5.2 Element <KeyDescriptor>

The <KeyDescriptor> element provides information about the cryptographic key(s) that an entity uses to sign data or receive encrypted keys, along with additional cryptographic details. Its KeyDescriptorType complex type consists of the following elements and attributes:

`Use` [Optional] Optional attribute specifying the purpose of the key being described. Values are drawn from the KeyTypes enumeration, and consist of the values encryption and signing.

`KeyInfo` [Required] This element directly or indirectly identifies a key. See [XMLSig] for additional details on the use of this element.

`ValidFrom` [Optional] This attribute indicates the first validity date for the KeyInfo data. This is for human informational use and should reflect validity time start data (the NotBefore attribute) in X.509 Certificate. It can also be a time *after* the NotBefore attribute time, in which case it overrides the certificate specified time.

`ValidUntil` [Optional] This attribute indicates the last validity date for the KeyInfo data. This is for human informational use and should reflect validity time end data (the NotAfter attribute) in X.509 Certificate. It can also be a time *before* the NotAfter attribute time, in which case it overrides the certificate specified time.

`EncryptionMethod` [Zero or More] Optional element specifying an algorithm and algorithm-specific settings supported by the entity. The exact content varies based

	on the algorithm supported. See [XMLEnc] for the definition of this element's xenc:EncryptionMethodType complex type.

The following schema fragment defines the <KeyDescriptor> element and its KeyDescriptorType complex type:

Because a system can have multiple active keys for any given purpose, (especially for the key version migration,) following is an example of valid keying data for encryption use:

```
<mmd:KeyDescriptor Use="encryption">
      <ds:KeyInfo>
            <ds:X509Data>
                  <ds:X509Certificate>XXXXXXXXXXXXXXXXXX</ds:X509Certificate>
            </ds:X509Data>
      </ds:KeyInfo>
</mmd:KeyDescriptor>
```

***Note**: There are many ways to define <ds:KeyInfo>, but for interoperability reasons it is recommended that X.509 Certificates are used, and others are optional.*

See also chapter 4 for discussion about data publication, and key version migration.

3.5.3 Simple type KeyTypes

The KeyTypes element provides information about the usage of the cryptographic key(s). KeyTypes simple type consists of the following enumerations:

`encryption`	Key usage is for XML encryption.
`signing`	Key usage is for XML signing.
`ssl`	Key usage is for HTTPS communication, both client and server.

MSSP Servers and APs may publish any subset of these KeyTypes. MSSPs Service providers may publish their

The following schema fragment defines the KeyTypes simple type:

```
<xs:simpleType name="KeyTypes">
      <xs:restriction base="xs:string">
            <xs:enumeration value="encryption"/>
            <xs:enumeration value="signing"/>
            <xs:enumeration value="ssl"/>
      </xs:restriction>
</xs:simpleType>
```

***Note**: While it is possible to have separate keys for HTTPS client and server, resulting logistical burden will be excessive compared to benefits of having separate keys.*

3.6 Examples

The following MSSP metadata is an example of a MSSP system. An MSSP entity (mssp.methics.fi) has published it's MSSP system metadata. XML signatures or certificates are not shown in the example.

```
<?xml version="1.0" encoding="UTF-8"?>
<mmd:EntitiesDescriptor
```

```
     xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
     xmlns:mmd="http://www.methics.fi/MSSPMetadata/v1.0.0#"
     xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#" >
  <mmd:EntitiesDescriptor ID="mssp.methics.fi">
    <mmd:EntityDescriptor ID="ae1">
      <ds:Signature>
        <ds:SignedInfo>
          <ds:CanonicalizationMethod
               Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
                    </ds:CanonicalizationMethod>
          <ds:SignatureMethod

      Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1">
                        </ds:SignatureMethod>
          <ds:Reference URI="#ae1">
            <ds:Transforms>
              <ds:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature">
              </ds:Transform>
            </ds:Transforms>
            <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1">
                          </ds:DigestMethod>
            <ds:DigestValue>tvvo4HAe/jPCfiTzlFbiRdbEjQw=</ds:DigestValue>
          </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>XXXXXXXXXXXXXXXXXX</ds:SignatureValue>
        <ds:KeyInfo>
          <ds:X509Data>
            <ds:X509Certificate>XXXXXXXXXXXXXXXXXX</ds:X509Certificate>
          </ds:X509Data>
        </ds:KeyInfo>
      </ds:Signature>
      <mmd:EntityID>
        <mss:MSSP_ID>
          <mss:IPAddress>124.34.1.53</mss:IPAddress>
          <mss:URI>http://ae1.methics.fi</mss:URI>
        </mss:MSSP_ID>
      </mmd:EntityID>
      <mmd:MSSPDescriptor WantRequestsEncrypted="true"
Roles="http://uri.etsi.org/TS102207/v1.1.2role_AcquiringEntity">
        <mmd:KeyDescriptor Use="encryption">
          <ds:KeyInfo>
            <ds:X509Data>
              <ds:X509Certificate>XXXXXXXXXXXXXXXXXX</ds:X509Certificate>
            </ds:X509Data>
          </ds:KeyInfo>
        </mmd:KeyDescriptor>

<mmd:MSS_SignatureService>http://pavo.methics.fi:8080/soap/services/MSS_Signat
urePort</mmd:MSS_SignatureService>

<mmd:MSS_RegistrationService>http://pavo.methics.fi:8080/soap/services/MSS_Reg
istrationPort</mmd:MSS_RegistrationService>

<mmd:MSS_ReceiptService>http://pavo.methics.fi:8080/soap/services/MSS_ReceiptP
ort</mmd:MSS_ReceiptService>
```

```
      </mmd:MSSPDescriptor>
    </mmd:EntityDescriptor>
    <mmd:EntityDescriptor>
      <mmd:EntityID>
        <mss:MSSP_ID>
          <mss:IPAddress>124.34.1.54</mss:IPAddress>
          <mss:URI>http://ae2.methics.fi</mss:URI>
        </mss:MSSP_ID>
      </mmd:EntityID>
      <mmd:MSSPDescriptor
Roles="http://uri.etsi.org/TS102207/v1.1.2role_AcquiringEntity"
WantRequestsEncrypted="true">
        <mmd:KeyDescriptor Use="encryption">
          <ds:KeyInfo>
            <ds:X509Data>
              <ds:X509Certificate>XXXXXXXXXXXXXXXXXX</ds:X509Certificate>
            </ds:X509Data>
          </ds:KeyInfo>
        </mmd:KeyDescriptor>

<mmd:MSS_SignatureService>https://ae2.methics.fi:8443/soap/services/MSS_Signat
urePort</mmd:MSS_SignatureService>

<mmd:MSS_ReceiptService>https://ae2.methics.fi:8443/soap/services/MSS_ReceiptP
ort</mmd:MSS_ReceiptService>
      </mmd:MSSPDescriptor>
    </mmd:EntityDescriptor>
  </mmd:EntitiesDescriptor>
  <mmd:EntitiesDescriptor ID="Body">
    <ds:Signature>
      <ds:SignedInfo>
        <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-
exc-c14n#"></ds:CanonicalizationMethod>
        <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"></ds:SignatureMethod>
        <ds:Reference URI="#Body">
          <ds:Transforms>
            <ds:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"></ds:Transform>
          </ds:Transforms>
          <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></ds:DigestMethod>
          <ds:DigestValue>y58oZF1O8sh+LwC+FX3nOwr9Q74=</ds:DigestValue>
        </ds:Reference>
      </ds:SignedInfo>
      <ds:SignatureValue>XXXXXXXXXXXXXXXXXX</ds:SignatureValue>
      <ds:KeyInfo>
        <ds:X509Data>
          <ds:X509Certificate>XXXXXXXXXXXXXXXXXX</ds:X509Certificate>
        </ds:X509Data>
      </ds:KeyInfo>
    </ds:Signature>
  </mmd:EntitiesDescriptor>
```

4 MSSP Metadata Signature Processing

<EntitiesDescriptor> and <EntityDescriptor> elements can be digitally signed. Signed MSSP metadata has the following benefits:

- Entity or Entities metadata integrity is guaranteed
- Entity or Entities metadata is authentic
- MSSP may use metadata for TLS/SSL based authentication

Signing of metadata elements is not mandatory, for example if a mobile signature system party obtains the metadata information directly from the publishing entity through a secure channel. Anyhow we recommended that at least the element containing HomeMSSP's <EntityDescriptor> should be signed.

4.1 XML Signature

XML Signature [XMLSig] is a W3C recommendation that defines an XML syntax for digital signatures. MSSP metadata processor needs only a small fixed part of XML Signature processing. This document specifies use of the xs:ID attribute, Reference element, Canonicalization and Transformation on the elements that have been signed.

A metadata element's signature must apply to the content of the signed element plus all descendants including namespaces and attributes but not comments.

4.2 Signing Formats and Algorithms

MSSP metadata must use enveloped signatures when signing the elements.

MSSP processors should support the use of RSA signing and verification for public key operations with the algorithm: http://www.w3.org/2000/09/xmldsig#rsa-sha1.

4.3 ID attribute and Reference Element

Signed metadata element must contain a value for xs:ID attribute. Reference element's XPointer formated URI attribute refers to that ID attribute element. For example Reference URI "#HMSSPMethics1" identifies the element (a fragment of a document) with ID attribute value "HMSSPMethics1". All ID attributes must be unique in the MSSP metadata document.

4.4 Canonicalization Method

MSSP metadata must use Exclusive Canonicalization without comments, both in the <ds:CanonicalizationMethod> element of <ds:SignedInfo>, and as a <ds:Transform> embedded in an XML context can be verified independent of that context.

4.5 Transforms

XML Signatures in MSSP metadata must not contain transforms other than the enveloped signature transform (http://www.w3.org/2000/09/xmldsig#enveloped-signature) or the exclusive canonicalization transforms (with the identifier http://www.w3.org/2001/10/xml-exc-c14n# or http://www.w3.org/2001/10/xml-exc-c14n#WithComments).

4.6 KeyInfo

XML Signature defines usage of the <ds:KeyInfo> element. MSSP metadata does not require the use of <ds:KeyInfo> and does not have any restrictions on its use. Practical recommendation is to prefer <ds:X509Cert> data.

5 MSSP Metadata Management

5.1 Metadata Life Cycle

A new metadata file can be published at any time. The ValidUntil attributes in the metadata elements defines the maximum metadata element validity period. All parties should update their metadata before any ValidUntil attribute has been expired.

A new certificate information (next certificate) can be published in advance and in parallel to current certificate (figure 3). There is no reason to publish a new metadata file only if any certificate in the file has been expired and the next certificate is still valid.

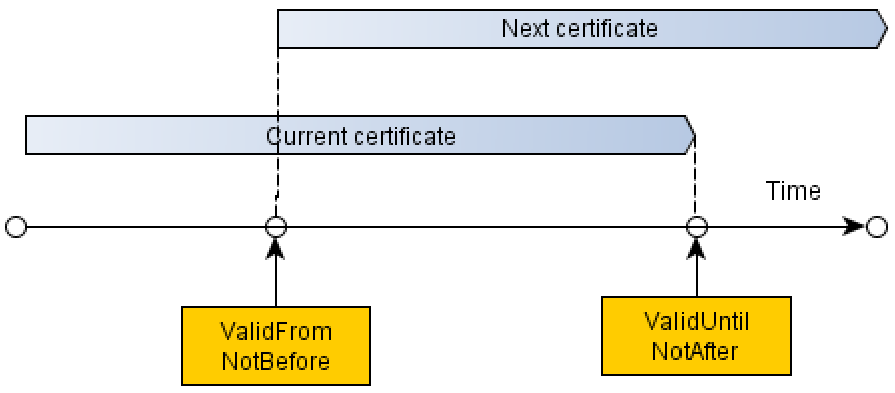

Figure 3: Certificate lifecycle

Note: *In practise a next certificate should be available in the metadata at least 3 months before the current one expires.*

Note: *In practice certificate validity should be up to three years. Longer key validity periods will also mandate longer keys. See [NIST] for more information.*

The X.509 certificate in the metadata may contain both the NotBefore and NotAfter attributes and the metadata KeyDescriptor may contain ValidFrom or ValidUntil attributes. A metadata user must use the most strict validity period. Therefore a certificate in the metadata may have shorter validity than the validity period of the certificate.

TLS/SSL (HTTPS), XML Signing and XML Encryption may have multiple overlapping keys. An entity acting as a HTTPS server presents a SSL certificate to it's clients. A system operator chooses this certificate, and it requires the system operator's action to take a next certificate into use. Client applications that use the metadata should accept all of the metadata certificates when communicating with the entity.

5.2 Metadata Publication

Public MSSP metadata is generally needed when establishing large mobile signature service systems or large interconnected Mesh. Only MSSPs should publish their metadata. APs should provide their metadata to their AE service provider which takes care of the AP metadata. Home MSSPs, Additional service provider MSSPs (like a Verifying Entity MSSP) and the AE may need to know AP metadata.

There are three well known mechanisms for an entity to publish (and for a consumer to resolve the location of) MSSP metadata documents:

1. via a "well-known-location" by directly dereferencing the entity's unique identifier (a URI variously referred to as a MSSP's EntityID element), or
2. indirectly by publishing the location of metadata in the URI or
3. out-of-band mechanisms.

A metadata consumer that supports the first two mechanisms must do resolution via using the "well-known-location" mechanism before DNS dissimilar to SAML2 resolution mechanism. The reason for this is that a MSSP business model may vary and the metadata consumer must be aware of all business related issues before MSS services can be connected.

When retrieval requires network transport of the MSSP metadata document, the transport must be protected with mechanisms providing server authentication and integrity protection.

In any case the metadata provider should sign the MSSP metadata and the consumer should validate MSSP metadata signature and certificates.

The mechanisms described in this section help in establishing trust in the accuracy and authenticity of MSSP metadata, including use of XML signatures and SSL/TLS authentication. Regardless of the mechanism used, the parties should have some means by which to establish trust in metadata information before relying on it.

5.3 Publication via Well-Known Location

Entities may publish their metadata documents at a well known location by placing the document at the location denoted by the DistributionURL. Metadata distribution point URL attribute is recommended only in the top level EntitiesDescriptor. The XML document provided at the well-known location must describe the metadata for the entities managed by the MSSP identifier.

An indirection mechanism supported by the URL scheme (such as an HTTP 1.1 302 redirect) may be used.

Note: *The MSSP metadata distribution model is not the same as the SAML2 metadata distribution model.*

5.4 Publication via DNS

Please see the "Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0" specification for details.

5.5 Processing Metadata File

When you receive the metadata file, you should get the file directly from a source that you trust. The metadata publisher may also calculate a hash code of the metadata file. You should calculate the hash code of the metadata file you have received and verify that the hash code is the same as the published one. Verify that top-level XML Signature in the metadata file is valid.

You can use ETSI204 MSS_Handshake to identify end-to-end knowledge of certificates.

If the metadata file you have received does not contain all entities you need, you may add your own elements into the metadata file and sign the file with your own XML signature.

6 References

ETSI204 ETSI TS 102 204 V1.1.4 (2003-08); Technical Specification; Mobile Commerce (M-COMM); Mobile Signature Service;Web Service Interface.

ETSI207 ETSI TS 102 207 V1.1.3 (2003-08); Technical Specification;Mobile Commerce (M-COMM); Mobile Signature Service; Specifications for Roaming in Mobile Signature Services

SAML2 OASIS Security Services Technical Committee, “Security Assertion Markup Language (SAML) Version 2.0 Specification Set”. OASIS Standard, 15-Mar-2005.

XMLSig XML Signature Syntax and Processing (Second Edition), W3C Recommendation, 10 June 2008

XMLEnc XML Encryption Syntax and Processing, W3C Recommendation, 10 December 2002

DSS Digital Signature Service Core Protocols, Elements, and Bindings Version 1.0, OASIS Standard, 11 April 2007

NIST NIST Special Publication 800-57, Recommendation for key management, draft

Appendix A: Methics Notices

Appendix B: Normative XML Schema

This is the normative version of the XML Schema.

Note that <xs:import> references have been changed from external URLs to internal file names in order to avoid external web loading of them every time the code generators are run. For the ETSI204 there is no URL referrable schema file, material has been copied from the ETSI standard document to local file MSS.xsd.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE schema  PUBLIC "-//W3C//DTD XMLSchema 200102//EN"
                         "XMLSchema.dtd"
 [
   <!ATTLIST schema
     xmlns:ds CDATA #FIXED 'http://www.w3.org/2000/09/xmldsig#'
     xmlns:mmd CDATA #FIXED 'http://www.methics.fi/MSSPMetadata/v1.0.0#'
     xmlns:mss CDATA #FIXED 'http://uri.etsi.org/TS102204/v1.1.2#'
     xmlns:xenc CDATA #FIXED 'http://www.w3.org/2001/04/xmlenc#'
     xmlns:xs CDATA #FIXED 'http://www.w3.org/2001/XMLSchema'>
   <!ENTITY ds   'http://www.w3.org/2000/09/xmldsig#'>
   <!ENTITY mmd  'http://www.methics.fi/MSSPMetadata/v1.0.0#'>
   <!ENTITY mss  'http://uri.etsi.org/TS102204/v1.1.2#'>
   <!ENTITY xenc 'http://www.w3.org/2001/04/xmlenc#'>
   <!ENTITY xs   'http://www.w3.org/2001/XMLSchema'>
   <!ENTITY % p ''>
   <!ENTITY % s ''>
  ]>

<xs:schema
    targetNamespace="http://www.methics.fi/MSSPMetadata/v1.0.0#"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:mmd="http://www.methics.fi/MSSPMetadata/v1.0.0#"
    xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
    xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified">

    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
        schemaLocation="xmldsig-core-schema.xsd"/>
    <xs:import namespace="http://www.w3.org/2001/04/xmlenc#"
      schemaLocation="xenc-schema.xsd"/>
    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
        schemaLocation="xml.xsd"/>

    <xs:import namespace="http://uri.etsi.org/TS102204/v1.1.2#"
               schemaLocation="MSS-plus.xsd"/>

    <xs:annotation>
        <xs:documentation>
            Document identifier: MSSP-metadata-1.0.xsd
            Location: etc/xmlschemas/mssp-metadata-1.0.xsd
            Revision history:
              V1.0.1 (July 2009):
                Schema for MSSP metadata, for use with XML signatures
                and encrypted data.
            TODO: Add network structure for routing.
        </xs:documentation>
    </xs:annotation>

    <xs:element name="EntitiesDescriptor" type="mmd:EntitiesDescriptorType"/>
    <xs:complexType name="EntitiesDescriptorType">
        <xs:sequence>
            <xs:element ref="ds:Signature" minOccurs="0"/>
            <!-- This Organization element is the publisher. -->
            <xs:element ref="mmd:Organization"  minOccurs="0"
                                                maxOccurs="1"/>
            <xs:element ref="mmd:ContactPerson" minOccurs="0"
                                                maxOccurs="unbounded"/>
            <xs:choice minOccurs="1" maxOccurs="unbounded">
                <xs:element ref="mmd:EntityDescriptor"/>
                <xs:element ref="mmd:EntitiesDescriptor"/>
            </xs:choice>
            <xs:element name="Extensions" type="mmd:ExtensionsType"
                                          minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
        <xs:attribute name="ValidFrom"  type="xs:dateTime"    use="optional"/>
        <xs:attribute name="ValidUntil" type="xs:dateTime"    use="optional"/>
        <xs:attribute name="ID"         type="xs:ID"          use="optional"/>
        <xs:attribute name="Name"       type="xs:string"      use="optional"/>
        <xs:attribute name="DistributionURL" type="xs:anyURI" use="optional"/>
        <!-- TODO: type -->
    </xs:complexType>

    <xs:element name="EntityDescriptor" type="mmd:EntityDescriptorType"/>
    <xs:complexType name="EntityDescriptorType">
        <xs:sequence>
            <xs:element ref="ds:Signature"      minOccurs="0" maxOccurs="1"/>
            <xs:element ref="mmd:Organization"  minOccurs="0" maxOccurs="1"/>
            <xs:element ref="mmd:ContactPerson" minOccurs="0"
                                                maxOccurs="unbounded"/>
            <xs:choice minOccurs="1" maxOccurs="1">
              <xs:element ref="mmd:MSSPDescriptor"/>
              <xs:element ref="mmd:APDescriptor"/>
              <xs:element ref="mmd:ServiceProviderDescriptor"/>
              <xs:element ref="mmd:ClientDescriptor"/>
            </xs:choice>
            <xs:element name="Extensions" type="mmd:ExtensionsType"
                                          minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
        <xs:attribute name="LastChanged" type="xs:dateTime" use="optional"/>
        <xs:attribute name="ID"          type="xs:ID"       use="optional"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <xs:element name="Organization" type="mmd:OrganizationType"/>
    <xs:complexType name="OrganizationType">
        <xs:sequence>
            <xs:element ref="mmd:OrganizationName" maxOccurs="unbounded"/>
            <xs:element ref="mmd:OrganizationURL"  maxOccurs="unbounded"/>
            <xs:element name="Extensions" type="mmd:ExtensionsType"
                                          minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
    <xs:element name="OrganizationName" type="xs:string"/>
    <xs:element name="OrganizationURL"  type="xs:string"/>

    <xs:element name="ContactPerson"    type="mmd:ContactPersonType"/>
    <xs:complexType name="ContactPersonType">
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="ContactType" type="mmd:ContactTypeType"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>

    <xs:simpleType name="ContactTypeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="technical"/>
            <xs:enumeration value="support"/>
            <xs:enumeration value="administrative"/>
            <xs:enumeration value="billing"/>
            <xs:enumeration value="other"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="NodeDescriptorType" abstract="true">
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <xs:complexType name="DisplayNameType">
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="Language" type="xs:string" use="optional" />
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>

    <xs:element name="KeyDescriptor" type="mmd:KeyDescriptorType"/>
    <xs:complexType name="KeyDescriptorType">
        <xs:sequence>
            <xs:element ref="ds:KeyInfo"/>
            <xs:element ref="mmd:EncryptionMethod" minOccurs="0"
                                                   maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Use"         type="mmd:KeyTypes" use="optional"/>
    </xs:complexType>
    <xs:simpleType name="KeyTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="encryption"/>
            <xs:enumeration value="signing"/>
            <xs:enumeration value="ssl"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="EncryptionMethod" type="xenc:EncryptionMethodType"/>

    <xs:element name="Certs" type="mmd:CertsType"/>
    <xs:complexType name="CertsType">
        <xs:sequence>
```

```
            <xs:element ref="mmd:KeyDescriptor" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:element name="Wants" type="mmd:WantsType"/>
    <xs:complexType name="WantsType">
        <xs:attribute name="WantSSLAuthentication" type="xs:boolean"
                                                   use="optional"/>
        <xs:attribute name="WantRequestsSigned"    type="xs:boolean"
                                                   use="optional"/>
        <xs:attribute name="WantRequestsEncrypted" type="xs:boolean"
                                                   use="optional"/>
    </xs:complexType>

    <xs:element name="MSSPDescriptor" type="mmd:MSSPDescriptorType"/>
    <xs:complexType name="MSSPDescriptorType">
        <xs:complexContent>
            <xs:extension base="mmd:NodeDescriptorType">
                <xs:sequence>
                    <xs:element name="MSSP_ID" type="mss:MeshMemberType"
                                minOccurs="1" maxOccurs="1"/>
                    <!-- as per mss:MessageAbstractType:MSSP_Info:MSSP_ID -->
                    <xs:element name="DisplayName" type="mmd:DisplayNameType"
                                minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element ref="mmd:MSS_SignatureService"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:MSS_RegistrationService"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:MSS_NotificationService"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:MSS_ProfileQueryService"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:MSS_ReceiptService"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:MSS_StatusQueryService"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:MSS_HandshakeService"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:AdditionalServices"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:RegistrationServices"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:Certs"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:Wants"
                                minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
                <xs:attribute name="Roles"
                              type="mmd:AnyURIListType" use="required"/>
                <xs:attribute name="SignatureProfiles"
                              type="mmd:AnyURIListType" use="optional"/>
                <xs:attribute name="SignatureProfileComparison"
                              type="xs:boolean" use="optional"/>
                <xs:attribute name="MSS_Formats"
                              type="mmd:AnyURIListType" use="optional"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="MSS_SignatureService"    type="xs:anyURI"/>
    <xs:element name="MSS_RegistrationService" type="xs:anyURI"/>
    <xs:element name="MSS_NotificationService" type="xs:anyURI"/>
    <xs:element name="MSS_ProfileQueryService" type="xs:anyURI"/>
    <xs:element name="MSS_ReceiptService"      type="xs:anyURI"/>
    <xs:element name="MSS_StatusQueryService"  type="xs:anyURI"/>
    <xs:element name="MSS_HandshakeService"    type="xs:anyURI"/>
    <xs:simpleType name="AnyURIListType">
        <xs:list itemType="xs:anyURI"/>
    </xs:simpleType>

    <xs:element name="APDescriptor" type="mmd:APDescriptorType"/>
    <xs:complexType name="APDescriptorType">
        <xs:complexContent>
            <xs:extension base="mmd:NodeDescriptorType">
                <xs:sequence>
                    <xs:element name="AP_ID"   type="xs:anyURI"
                                               minOccurs="1"/>
                    <!-- as per mss:MessageAbstractType:AP_Info:AP_ID -->
                    <xs:element name="DisplayName" type="mmd:DisplayNameType"
                                minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element ref="mmd:MSS_NotificationService"
                                minOccurs="0" maxOccurs="1"/>
                    <xs:element name="EncryptedAP_PWD"
                                type="mmd:EncryptedElementType" minOccurs="0"
                                maxOccurs="unbounded"/>
                    <xs:element ref="mmd:Certs" minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:Wants" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <xs:complexType name="EncryptedElementType">
      <xs:sequence>
        <xs:element ref="xenc:EncryptedData"/>
        <xs:element ref="xenc:EncryptedKey" minOccurs="0"
                                            maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>

    <xs:element name="ServiceProviderDescriptor"
               type="mmd:ServiceProviderDescriptorType"/>
    <xs:complexType name="ServiceProviderDescriptorType">
        <xs:complexContent>
            <xs:extension base="mmd:NodeDescriptorType">
                <xs:sequence>
                    <xs:element name="SP_ID" type="xs:anyURI"
                                             minOccurs="1"/>
                    <!-- ServiceProvider ID -->
                    <xs:element name="DisplayName" type="mmd:DisplayNameType"
                                minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="ServiceProvider"
                        type="mmd:ServiceProviderType"
                        minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:Certs" minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:Wants" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <xs:element name="ClientDescriptor" type="mmd:ClientDescriptorType"/>
    <xs:complexType name="ClientDescriptorType">
        <xs:complexContent>
            <xs:extension base="mmd:NodeDescriptorType">
                <xs:sequence>
                    <xs:element name="Client_ID"  type="xs:anyURI"
                                                  minOccurs="1"/>
                    <xs:element name="DisplayName" type="mmd:DisplayNameType"
                                minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="ServiceProvider"
                                type="mmd:ServiceProviderType" minOccurs="0"
                                maxOccurs="1"/>
                    <xs:element ref="mmd:Certs" minOccurs="0" maxOccurs="1"/>
                    <xs:element ref="mmd:Wants" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <xs:complexType name="ServiceProviderType">
      <!--<xs:complexContent>-->
        <xs:sequence>
            <xs:element name="Service" type="ServiceType"  minOccurs="0"
                        maxOccurs="unbounded"/>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
                        maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
        <!--</xs:complexContent>-->
    </xs:complexType>

    <xs:complexType name="ServiceType">
       <xs:sequence>
          <xs:element name="Description" type="xs:anyURI" minOccurs="0"
                                                          maxOccurs="1"/>
        <!--   <xs:any namespace="##other" processContents="lax" />-->
       </xs:sequence>
       <xs:attribute name="Type" type="mmd:ServiceSimpleType"/>
       <xs:attribute name="ServiceURL" type="xs:anyURI" minOccurs="0"
                                                        maxOccurs="1"/>
    </xs:complexType>

    <xs:simpleType name="ServiceSimpleType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="ca"/>
            <xs:enumeration value="additionalservice"/>
            <xs:enumeration value="dss"/>
            <xs:enumeration value="spml2"/>
            <xs:enumeration value="saml2"/>
            <xs:enumeration value="cmphttp"/>
            <xs:enumeration value="xkms"/>
            <xs:enumeration value="other"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:element name="AdditionalServices">
       <xs:complexType>
          <xs:sequence>
             <xs:element name="AdditionalService"
type="mmd:AdditionalServiceType" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
       </xs:complexType>
    </xs:element>

    <xs:complexType name="AdditionalServiceType">
       <xs:sequence>
          <xs:element name="Description" type="xs:anyURI"/>
          <xs:element name="SP_ID"       type="xs:anyURI" minOccurs="0"/>
          <xs:any namespace="##other" processContents="lax" minOccurs="0"
                                      maxOccurs="unbounded" />
       </xs:sequence>
    </xs:complexType>

    <xs:element name="RegistrationServices">
       <xs:complexType>
          <xs:sequence>
             <xs:element name="RegistrationService"
                         type="mmd:RegistrationServiceType"
                         minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
       </xs:complexType>
    </xs:element>

    <xs:complexType name="RegistrationServiceType">
       <xs:sequence>
          <xs:element name="SP_ID"       type="xs:anyURI" minOccurs="0"/>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
             <xs:any namespace="##other" processContents="lax" minOccurs="0"
                                         maxOccurs="unbounded"/>
          </xs:choice>
       </xs:sequence>
    </xs:complexType>

    <xs:complexType final="#all" name="ExtensionsType">
      <xs:sequence>
         <xs:any namespace="##other" processContents="lax"
                 maxOccurs="unbounded"/>
       </xs:sequence>
    </xs:complexType>

</xs:schema>
```