# [MS-KPS]:

# Key Protection Service Protocol

---

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs**. To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.
- **Trademarks**. The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

**Reservation of Rights**. All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

**Tools**. The Open Specifications documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

**Support.** For questions and support, please contact dochelp@microsoft.com.

## Revision Summary

| Date | Revision History | Revision Class | Comments |
|---|---|---|---|
| 3/16/2017 | 1.0 | New | Released new document. |
| 6/1/2017 | 2.0 | Major | Significantly changed the technical content. |
| 9/15/2017 | 3.0 | Major | Significantly changed the technical content. |
| 12/1/2017 | 3.0 | None | No changes to the meaning, language, or formatting of the technical content. |


*Copyright © 2017 Microsoft Corporation*

# Table of Contents


**1 Introduction ..... 5**
1.1 Glossary ..... 5
1.2 References ..... 5
1.2.1 Normative References ..... 5
1.2.2 Informative References ..... 6
1.3 Overview ..... 6
1.4 Relationship to Other Protocols ..... 6
1.5 Prerequisites/Preconditions ..... 6
1.6 Applicability Statement ..... 6
1.7 Versioning and Capability Negotiation ..... 6
1.8 Vendor-Extensible Fields ..... 6
1.9 Standards Assignments ..... 6

**2 Messages ..... 7**
2.1 Transport ..... 7
2.2 Common Data Types ..... 7
2.2.1 HTTP Methods ..... 7
2.2.1.1 RollTransportKey ..... 7
2.2.1.2 GetMetaData ..... 7
2.2.2 Complex Types ..... 8
2.2.2.1 RollTransportKeyRequest ..... 9
2.2.2.2 RollTransportKeyResponse ..... 10
2.2.2.3 Protector ..... 11
2.2.2.4 Wrapping ..... 12
2.2.2.5 Error ..... 12
2.2.2.6 WrappingCollection ..... 13
2.2.2.7 TransportKeySignature ..... 13
2.2.2.8 GuardianSignature ..... 13
2.2.2.9 KeyDerivationMethod ..... 14
2.2.2.10 Signature ..... 14
2.2.2.11 EncryptedData ..... 15
2.2.2.12 SigningCertificateSignature ..... 15
2.2.2.13 EncryptionCertificateSignature ..... 15
2.2.2.14 TransportKey ..... 16
2.2.2.15 Parameters ..... 16
2.2.3 Simple Types ..... 16
2.2.3.1 IngressProtector ..... 17
2.2.3.2 HealthCertificate ..... 17
2.2.3.3 TransferKeyEncryptionAlgorithm ..... 18
2.2.3.4 WrappingKeyEncryptionAlgorithm ..... 18
2.2.3.5 TransportKeyEncryptionAlgorithm ..... 18
2.2.3.6 EgressProtector ..... 19
2.2.3.7 EncryptedTransferKey ..... 19
2.2.3.8 EncryptedWrappingKey ..... 19
2.2.3.9 EncryptedTransportKeys ..... 20
2.2.3.10 Version ..... 20
2.2.3.11 Certificate ..... 20
2.2.3.12 Algorithm ..... 20

**3 Protocol Details ..... 22**
3.1 Server Details ..... 22
3.1.1 Abstract Data Model ..... 22
3.1.2 Timers ..... 22
3.1.3 Initialization ..... 22
3.1.4 Higher-Layer Triggered Events ..... 23




*Copyright © 2017 Microsoft Corporation*

3.1.5 Message Processing Events and Sequencing Rules ..... 23
3.1.5.1 Service APIs ..... 23
3.1.5.1.1 RollTransportKey ..... 23
3.1.5.1.1.1 Request Body ..... 23
3.1.5.1.1.2 Response Body ..... 24
3.1.5.1.1.3 Processing Details ..... 24
3.1.5.1.2 GetMetaData ..... 25
3.1.5.1.2.1 Request Body ..... 25
3.1.5.1.2.2 Response Body ..... 25
3.1.5.1.2.3 Processing Details ..... 26
3.1.6 Timer Events ..... 26
3.1.7 Other Local Events ..... 26
3.2 Client Details ..... 26
3.2.1 Abstract Data Model ..... 26
3.2.2 Timers ..... 27
3.2.3 Initialization ..... 27
3.2.4 Higher-Layer Triggered Events ..... 27
3.2.4.1 Application Requests RollTransportKey ..... 27
3.2.4.2 Application Requests GetMetaData ..... 27
3.2.5 Message Processing Events and Sequencing Rules ..... 27
3.2.5.1 RollTransportKey ..... 27
3.2.5.2 GetMetaData ..... 28
3.2.6 Timer Events ..... 28
3.2.7 Other Local Events ..... 28

**4 Protocol Examples ..... 29**

**5 Security ..... 30**
5.1 Security Considerations for Implementers ..... 30
5.2 Index of Security Parameters ..... 30

**6 Appendix A: Full XML Schema ..... 31**
6.1 Protector Schema ..... 31
6.2 RollTransportKey Request Schema ..... 32
6.3 RollTransportKey Response Schema ..... 32
6.4 MetaData Resposne Schema ..... 33
6.5 Crypto Schema ..... 33

**7 Appendix B: Product Behavior ..... 35**

**8 Change Tracking ..... 36**

**9 Index ..... 37**


*Copyright © 2017 Microsoft Corporation*

# 1 Introduction

This document specifies the Key Protection Service (KPS) Protocol,a component of the Host Guradian service, which provides security assurance for shielded virtual machines.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

## 1.1 Glossary

This document uses the following terms:

**base64 encoding**: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [RFC4648].

**binary large object (BLOB)**: A discrete packet of data that is stored in a database and is treated as a sequence of uninterpreted bytes.

**HTTP 1.1**: Hypertext Transfer Protocol -- HTTP/1.1 [RFC2616]

**HTTP method**: In an HTTP message, a token that specifies the method to be performed on the resource that is identified by the Request-URI, as described in [RFC2616].

**Hypertext Transfer Protocol (HTTP)**: An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

**UTF-8**: A byte-oriented standard for encoding Unicode characters, defined in the Unicode standard. Unless specified otherwise, this term refers to the UTF-8 encoding form specified in [UNICODE5.0.0/2007] section 3.9.

**X.509**: An ITU-T standard for public key infrastructure subsequently adapted by the IETF, as specified in [RFC3280].

**MAY, SHOULD, MUST, SHOULD NOT, MUST NOT:** These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

## 1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

### 1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[MS-HGSA] Microsoft Corporation, "Host Guardian Service: Attestation Protocol".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.rfc-editor.org/rfc/rfc2616.txt


*Copyright © 2017 Microsoft Corporation*

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, http://www.rfc-editor.org/rfc/rfc2818.txt

### 1.2.2 Informative References

None.

## 1.3 Overview

Host Guardian Service is a server role that provides the security services Attestation Service and Key Protection Service. Together these two services help provide security assurance for Shielded VMs by ensuring that Shielded VMs can be run only on known and trusted fabric hosts that have a legitimate configuration. This specification defines Key Protection Service. The Attestation Service is defined in the [MS-HGSA] specification.

## 1.4 Relationship to Other Protocols

For its attestation service, Key Protection Service uses the Host Guardian Service: Attestation Protocol as specified in [MS-HGSA].

## 1.5 Prerequisites/Preconditions

None.

## 1.6 Applicability Statement

The Host Guardian Service includes Attestation Service and Key Protection Service as critical components that secure virtual machines in a cloud-based environment.

## 1.7 Versioning and Capability Negotiation

None.

## 1.8 Vendor-Extensible Fields

There are no vendor-extensible fields for the Key Protection Service Protocol.

## 1.9 Standards Assignments

None.


*Copyright © 2017 Microsoft Corporation*

# 2 Messages

## 2.1 Transport

The Key Protection Service Protocol uses **HTTP** or secure **HTTP 1.1** as transport, as specified in [RFC2616] and [RFC2818].

## 2.2 Common Data Types

### 2.2.1 HTTP Methods

This protocol defines the following common **HTTP methods** in addition to the existing set of standard HTTP methods.

| Method | Section | Description |
|---|---|---|
| RollTransportKey | 2.2.1.1 | Extracts the **TransportKey** from the **IngressProtector**, generates a new transport key, creates the **EgressProtector** and returns both transport keys to the caller. |
| GetMetaData | 2.2.1.2 | Returns the metadata content containing the guardian information to the client. |

#### 2.2.1.1 RollTransportKey

The **RollTransportKey** method validates that the **IngressProtector** defined in section 2.2.3.1 is well-formed, performs Key Protection Service (KPS) checks by using an encryption algorithm in an implementation-specific manner, and generates the **EgressProtector**.

This method is invoked from the following URI:

```
http://<server>/keyprotection/service/{version}/rolltransportkey
```

#### 2.2.1.2 GetMetaData

The **GetMetaData** method provides the list of KPS-supported certificates, which are used in validating that the **KeyProtector** was properly signed by KPS or to create a new protector and encrypt the transport keys.

This method is invoked from the following URI with HTTP GET request:


Copyright © 2017 Microsoft Corporation

```
http://<server>/keyprotection/service/metadata/2014-07/metadata.xml
```

### 2.2.2 Complex Types

The following table summarizes the set of common complex type definitions that are included in this specification and use the XML format.

| Complex type | Section | Description |
|---|---|---|
| RollTransportKeyRequest | 2.2.2.1 | Requests the **BLOB** from the client with the protector descriptor and Health Certificate received after Attestation Services. |
| RollTransportKeyResponse | 2.2.2.2 | Response to the **RollTransportKeyRequest**. |
| Protector | 2.2.2.3 | Represents a protector. |
| Wrapping | 2.2.2.4 | Consists of certificates of type **base64-encoded** strings and a transport key. |
| Error | 2.2.2.5 | Possible error codes received from methods processed by the KPS **RollTransportKeyResponse** and **GetMetaData**. |
| WrappingCollection | 2.2.2.6 | Defines the list of **Wrapping elements** of the transport keys received from the client. |
| TransportKeySignature | 2.2.2.7 | Denotes the digital signature of the **TransportKey**. |
| GuardianSignature | 2.2.2.8 | Denotes the KPS signing certificate specified by **WrappingId** over the entire **Wrappings** element. |
| KeyDerivationMethod | 2.2.2.9 | Denotes the set of cryptographic parameters and algorithms needed to perform the KPS. |
| Signature | 2.2.2.10 | Denotes a digital signature that provides details about the entity that is used for providing Key Protection Services. |
| EncryptedData | 2.2.2.11 | Denotes the payload of data used by **KeyProtector** to perform the Key Protection Services. |
| SigningCertificateSignature | 2.2.2.12 | Denotes the signing certificate signature of the specified parent wrapping ID. |
| EncryptionCertificateSignature | 2.2.2.13 | Denotes the signature of the encryption certificate. |
| TransportKey | 2.2.2.14 | A base64-encoded string of type **UTF-8** format, which contains the wrapping key's transfer key |


*Copyright © 2017 Microsoft Corporation*

| Complex type | Section | Description |
|---|---|---|
| | | encrypted by the health certificate. |
| Parameters | 2.2.2.15 | Possible namespaces and process contents used to perform Key Protection Services. |

### 2.2.2.1 RollTransportKeyRequest

The **RollTransportKeyRequest** structure is sent by the client to request the encrypted transport keys and to perform Key Protection.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://schemas.microsoft.com/kps/2014/07/service"
           elementFormDefault="qualified"
           xmlns="http://schemas.microsoft.com/kps/2014/07/service"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:element name="RollTransportKeyRequest" type="RollTransportKeyRequest_T"/>

   <xs:complexType name="RollTransportKeyRequest_T">
      <xs:annotation>
         <xs:documentation>RollTransportKey request.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="IngressProtector">
            <xs:annotation>
               <xs:documentation>The ingress protector.</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
               <xs:restriction base="xs:base64Binary">
                  <xs:minLength value="1"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element name="HealthCertificate">
            <xs:annotation>
               <xs:documentation>The health certificate.</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
               <xs:restriction base="xs:base64Binary">
                  <xs:minLength value="1"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element name="TransferKeyEncryptionAlgorithm">
            <xs:annotation>
               <xs:documentation>The algorithm to be used to encrypt the wrapping key's
transfer key.</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
               <xs:restriction base="xs:anyURI">
                  <xs:minLength value="1"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element name="WrappingKeyEncryptionAlgorithm">
            <xs:annotation>
               <xs:documentation>The algorithm to be used to encrypt the transport keys'
wrapping key.</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
               <xs:restriction base="xs:anyURI">
```


Copyright © 2017 Microsoft Corporation

```
                  <xs:minLength value="1"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element name="TransportKeysEncryptionAlgorithm">
            <xs:annotation>
               <xs:documentation>The algorithm to be used to encrypt the transport
keys.</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
               <xs:restriction base="xs:anyURI">
                  <xs:minLength value="1"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>
</xs:schema>
```

**IngressProtector**: A **base64-encoded** string of type **UTF-8** format that contains the entire ingress protector as serialized to a file.

**HealthCertificate**: A base64-encoded binary string of type **X.509** format.

**TransferKeyEncryptionAlgorithm**: The algorithm to be used to encrypt the wrapping key's transfer key.

**WrappingKeyEncryptionAlgorithm:** The algorithm to be used to encrypt the transport keys' wrapping key.

**TransportKeyEncryptionAlgorithm:** The algorithm to be used to encrypt the transport keys.

#### 2.2.2.2 RollTransportKeyResponse

The **RollTransportKeyResponse** structure is sent by the KPS with encrypted keys, which is useful in allowing the guarded host to run on a VM.

```
<?xml version="1.0" encoding="utf-8"?>

<xs:schema targetNamespace="http://schemas.microsoft.com/kps/2014/07/service"
           elementFormDefault="qualified"
           xmlns="http://schemas.microsoft.com/kps/2014/07/service"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:element name="RollTransportKeyResponse" type="RollTransportKeyResponse_T"/>
   <xs:complexType name="RollTransportKeyResponse_T">
      <xs:annotation>
         <xs:documentation>RollTransportKey response.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="EgressProtector">
            <xs:annotation>
               <xs:documentation>The egress protector containing the new transport
key.</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
               <xs:restriction base="xs:base64Binary">
                  <xs:minLength value="1"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element name="EncryptedTransferKey">
            <xs:annotation>
```


Copyright © 2017 Microsoft Corporation

```
                <xs:documentation>The wrapping key's transfer key encrypted by the health
certificate.</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
                <xs:restriction base="xs:base64Binary">
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element name="EncryptedWrappingKey">
            <xs:annotation>
                <xs:documentation>The transport keys' wrapping key encrypted by the
transfer key.</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
                <xs:restriction base="xs:base64Binary">
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element name="EncryptedTransportKeys">
            <xs:annotation>
                <xs:documentation>The ingress and egress transport keys encrypted by the
transport keys' wrapping key.</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
                <xs:restriction base="xs:base64Binary">
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

**EgressProtector:** A **base64-encoded** string of type **UTF-8** format that contains the entire egress protector as serialized to a file.

**EncryptedTransferKey:** A base64-encoded string of type UTF-8 format that contains the wrapping key's transfer key, which is encrypted by the health certificate as defined in section 2.2.3.7.

**EncryptedWrappingKey:** A base64-encoded string of type UTF-8 format that contains the transport keys' wrapping key, which is encrypted by the transfer key as defined section 2.2.3.8.

**EncryptedTransportKeys:** A base64-encoded string of type UTF-8 format contains the ingress and egress transport keys, which are encrypted by the transport keys' wrapping key as defined in section 2.2.3.9.

### 2.2.2.3 Protector

The **Protector** structure is the cryptographically authenticated collection of different wrappings of the transport key, signed by the **Guardian**.

```
<xs:element name="Protector" type="Protector_T" />
  <xs:complexType name="Protector_T">
     <xs:annotation>
        <xs:documentation>A protector contains a list of wrappings of the transport
key.</xs:documentation>
     </xs:annotation>
     <xs:sequence>
        <xs:element name="Wrappings" type="WrappingCollection_T" />
        <xs:element name="TransportKeySignature" type="TransportKeySignature_T" />
        <xs:element name="GuardianSignature" type="GuardianSignature_T" />
     </xs:sequence>
```


Copyright © 2017 Microsoft Corporation

```
</xs:complexType>
```

**Wrappings:** A list of wrappings of the transport key to be included in the new protector of the type defined in section 2.2.2.4.

**TransportKeySignature**: A **UTF-8** converted signature computed by using a key derived from the actual transport key over the entire **Wrappings** element of the type defined in section 2.2.2.7.

**GuardianSignature:** A UTF-8 converted signature computed by using the signing certificate specified by **WrappingId** over the entire **Wrappings** element as defined in section 2.2.2.8.

### 2.2.2.4 Wrapping

The **Wrapping** structure consists of certificates of type **base64-encoded** strings and the **TransportKey**. This wrapping involves the authenticated encryption of concatenation of the ingress and egress keys.

```
<xs:element name="Wrapping" type="Wrapping T" />
  <xs:complexType name="Wrapping T">
     <xs:sequence>
        <xs:element name="Id" type="xs:unsignedInt" />
        <xs:element name="SigningCertificate" type="Certificate_T" />
        <xs:element name="SigningCertificateSignature"
type="SigningCertificateSignature T" />
        <xs:element name="EncryptionCertificate" type="Certificate_T" />
        <xs:element name="EncryptionCertificateSignature"
type="EncryptionCertificateSignature_T" />
        <xs:element name="TransportKey" type="TransportKey_T" />
     </xs:sequence>
  </xs:complexType>
```

**Id:** A 32-bit unsigned integer that contains the wrapping ID.

**SigningCertificate:** Signing certificate of type Certificate_T as defined in section 2.2.3.11.

**SigningCertificateSignature:** Signing certificate signature as defined in section 2.2.2.12.

**EncryptionCertificate:** Encryption certificate of type Certificate_T as defined in section 2.2.2.13.

**EncryptionCertificateSignature:** Encryption certificate signature as defined in section 2.2.2.13.

**TransportKey:** Encrypted transport key as defined in section 2.2.2.14.

### 2.2.2.5 Error

The **Error** structure denotes the possible error codes that are received from methods processed by the Key Protection Service's **RollTransportKey** and **GetMetaData** methods.

```
<xs:element name="Error" type="Error_T" />
  <xs:complexType name="Error T">
     <xs:annotation>
        <xs:documentation>Error response.</xs:documentation>
     </xs:annotation>
     <xs:sequence>
        <xs:element name="Code" type="xs:string">
           <xs:annotation>
              <xs:documentation>Error code.</xs:documentation>
           </xs:annotation>
        </xs:element>
```


Copyright © 2017 Microsoft Corporation

```
        <xs:element name="Message" type="xs:string">
           <xs:annotation>
              <xs:documentation>Error message.</xs:documentation>
           </xs:annotation>
        </xs:element>
     </xs:sequence>
  </xs:complexType>
```

**Code:** A string that represents the error response received from **RollTransportKey** or **GetMetaData**.

**Message:** A string that represents the error message of the error code received.

#### 2.2.2.6 WrappingCollection

The **WrappingCollection** structure defines the list of wrappings of the transport keys received from the client.

```
<xs:element name="Wrappings" type="WrappingCollection_T" />
  <xs:complexType name="WrappingCollection_T">
     <xs:sequence>
        <xs:element name="Wrapping" type="Wrapping T" minOccurs="1" maxOccurs="unbounded"
/>
     </xs:sequence>
  </xs:complexType>
```

**Wrapping:** Wrapping structure as defined in section 2.2.2.4

#### 2.2.2.7 TransportKeySignature

The **TransportKeySignature** structure denotes the digital signature of the transport key.

```
<xs:element name="TransportKeySignature" type="TransportKeySignature_T" />
  <xs:complexType name="TransportKeySignature_T">
     <xs:annotation>
       <xs:documentation>The transport key signature is computed using a key derived from
the actual transport key over the entire Wrappings element after exclusive xml
canonicalization (http://www.w3.org/2001/10/xml-exc-c14n#) and conversion to UTF-
8.</xs:documentation>
      </xs:annotation>
        <xs:sequence>
           <xs:element name="KeyDerivationMethod" type="KeyDerivationMethod_T" />
           <xs:element name="Signature" type="Signature_T" />
         </xs:sequence>
   </xs:complexType>
```

**KeyDerivationMethod**: Set of cryptographic parameters and algorithms needed to perform Key Protection Services as defined in section 2.2.2.9.

**Signature:** Provides details about the entity that is used for providing Key Protection Services as defined in section 2.2.2.10.

#### 2.2.2.8 GuardianSignature

The **GuardianSignature** structure denotes the digital signature using the KPS signing certificate specified by **WrappingId** over the entire **Wrappings** element.

```
<xs:element name="GuardianSignature" type="GuardianSignature_T" />
```


Copyright © 2017 Microsoft Corporation

```
  <xs:complexType name="GuardianSignature_T">
     <xs:annotation>
        <xs:documentation>The guardian signature is computed using the signing
certificate specified by WrappingId over the entire Wrappings element after exclusive xml
canonicalization (http://www.w3.org/2001/10/xml-exc-c14n#) and conversion to UTF-
8.</xs:documentation>
     </xs:annotation>
     <xs:sequence>
        <xs:element name="Signature" type="Signature_T" />
     </xs:sequence>
     <xs:attribute name="WrappingId" type="xs:unsignedInt" use="required" />
  </xs:complexType>
```

**Signature:** The guardian signature is computed by using the KPS signing certificate's private key of the type defined in section 2.2.2.10.

**WrappingId:** A 32-bit unsigned integer that contains a unique wrapping ID.

### 2.2.2.9 KeyDerivationMethod

The **KeyDerivationMethod** structure denotes the set of cryptographic parameters and algorithms needed to perform the KPS.

```
<xs:element name="KeyDerivationMethod" type="KeyDerivationMethod T" />
  <xs:complexType name="KeyDerivationMethod_T">
     <xs:sequence>
        <xs:element name="Parameters" type="CryptoParameters T" minOccurs="0" />
     </xs:sequence>
     <xs:attribute name="Algorithm" type="CryptoAlgorithm T" use="required" />
  </xs:complexType>
```

**Parameters:** Set of cryptographic parameters used to perform Key Protection Services of the type defined in section 2.2.2.15

**Algorithm:** Cryptographic algorithm used to perform Key Protection Services of the type defined in section 2.2.3.12

### 2.2.2.10 Signature

The **Signature** structure denotes a digital signature that provides the details about the entity that is used for providing Key Protection Services.

```
<xs:element name="Signature" type="Signature_T" />
  <xs:complexType name="Signature T">
     <xs:sequence>
        <xs:element name="Parameters" type="CryptoParameters T" minOccurs="0" />
        <xs:element name="SignatureValue">
           <xs:simpleType>
              <xs:restriction base="xs:base64Binary" />
           </xs:simpleType>
        </xs:element>
     </xs:sequence>
     <xs:attribute name="Algorithm" type="CryptoAlgorithm_T" use="required" />
  </xs:complexType>
```

**Parameters:** Set of cryptographic parameters used to perform Key Protection Services of the type defined in section 2.2.2.15.

**SignatureValue:** A **base64-encoded** binary string that represents the value of the **Signature**.


Copyright © 2017 Microsoft Corporation

**Algorithm:** Cryptographic algorithm used to perform Key Protection Services of the type defined in section 2.2.3.12.

#### 2.2.2.11 EncryptedData

The **EncryptedData** structure denotes the payload with the encrypted data to perform Key Protection Services.

```
<xs:element name="EncryptedData" type="EncryptedData_T" />
  <xs:complexType name="EncryptedData_T">
     <xs:sequence>
        <xs:element name="Parameters" type="CryptoParameters_T" minOccurs="0" />
        <xs:element name="CipherValue">
           <xs:simpleType>
              <xs:restriction base="xs:base64Binary" />
           </xs:simpleType>
        </xs:element>
     </xs:sequence>
     <xs:attribute name="Algorithm" type="CryptoAlgorithm_T" use="required" />
  </xs:complexType>
```

**Parameters:** Set of cryptographic parameters used to perform Key Protection Services of the type defined in section 2.2.2.15.

**CipherValue:** A **base64-encoded** binary string that holds the cipher value.

**Algorithm:** Cryptographic algorithm used to perform Key Protection Services of the type defined in section 2.2.3.12.

#### 2.2.2.12 SigningCertificateSignature

The **SigningCertificateSignature** structure denotes the digital signature of the wrapping's signing certificate computed by using the signing certificate of the specified parent wrapping ID.

```
<xs:element name="SigningCertificateSignature" type="SigningCertificateSignature_T" />
  <xs:complexType name="SigningCertificateSignature_T">
     <xs:annotation>
        <xs:documentation>The signing certificate signature is computed using the signing
certificate of the parent wrapping over this wrapping's signing
certificate.</xs:documentation>
     </xs:annotation>
     <xs:sequence>
        <xs:element name="Signature" type="Signature_T" />
     </xs:sequence>
     <xs:attribute name="ParentWrappingId" type="xs:unsignedInt" use="required" />
  </xs:complexType>
```

**Signature:** A digital signature as defined in section 2.2.2.10

**ParentWrappingId:** A 32-bit unsigned integer that contains the wrapping ID of the parent.

#### 2.2.2.13 EncryptionCertificateSignature

The **EncryptionCertificateSignature** structure denotes the signature of the encryption certificate.

```
<xs:element name="EncryptionCertificateSignature" type="EncryptionCertificateSignature_T"
/>
  <xs:complexType name="EncryptionCertificateSignature_T">
     <xs:annotation>
```


Copyright © 2017 Microsoft Corporation

```
        <xs:documentation>The encryption certificate signature is computed using this
wrapping's signing certificate over this wrapping's encryption
certificate.</xs:documentation>
     </xs:annotation>
     <xs:sequence>
        <xs:element name="Signature" type="Signature T" />
     </xs:sequence>
  </xs:complexType>
```

**Signature:** A digital signature as defined in section 2.2.2.10

#### 2.2.2.14 TransportKey

The **TransportKey** element is used to help protect data secured by the key protectors.

```
<xs:element name="TransportKey" type="TransportKey_T" />
  <xs:complexType name="TransportKey T">
     <xs:sequence>
        <xs:element name="EncryptedData" type="EncryptedData_T" />
     </xs:sequence>
  </xs:complexType>
```

#### 2.2.2.15 Parameters

The **Parameters** element denotes the cryptographic parameters used to perform Key Protection Services.

```
<xs:element name="Parameters" type="CryptoParameters_T />
  <xs:complexType name="CryptoParameters_T">
     <xs:sequence>
            <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
     </xs:sequence>
  </xs:complexType>
```

### 2.2.3 Simple Types

The following table summarizes the set of common simple type definitions that are included in this specification.

| Simple type | Section | Description |
|---|---|---|
| IngressProtector | 2.2.3.1 | The **IngressProtector** contains the entire ingress protector as serialized to a file and converted to a **base64-encoded** string. |
| HealthCertificate | 2.2.3.2 | A base64-encoded binary string of type **X.509** format received as input from the client for which Key Protection Services needs to be provided. |
| TransferKeyEncryptionAlgorithm | 2.2.3.3 | The algorithm to encrypt the wrapping key's transfer key. |
| WrappingKeyEncryptionAlgorithm | 2.2.3.4 | The algorithm to encrypt the transport keys' wrapping key. |


Copyright © 2017 Microsoft Corporation

| Simple type | Section | Description |
|---|---|---|
| TransportKeyEncryptionAlgorithm | 2.2.3.5 | The algorithm to encrypt the transport keys. |
| EgressProtector | 2.2.3.6 | A base64-encoded string of type **UTF-8** format, which contains the entire egress protector as serialized to a file. |
| EncryptedTransferKey | 2.2.3.7 | A base64-encoded string of type UTF-8 format, which contains the wrapping key's transfer key encrypted by the health certificate. |
| EncryptedWrappingKey | 2.2.3.8 | A base64-encoded string of type UTF-8 format, which contains the transport keys' wrapping key that is encrypted by the health certificate received from the client. |
| EncryptedTransportKeys | 2.2.3.9 | A base64-encoded string of type UTF-8 format, which contains the wrapping key's transfer key encrypted by the health certificate. |
| Version | 2.2.3.10 | A base64-encoded string of type UTF-8 format, which contains the wrapping key's transfer key encrypted by the health certificate. |
| Certificate | 2.2.3.11 | Certificate used to generate the key protectors. |
| Algorithm | 2.2.3.12 | Cryptographic algorithm used to perform Key Protection Services. |

#### 2.2.3.1 IngressProtector

The **IngressProtector** element denotes the entire ingress protector as serialized to a file and converted to a **base64-encoded** binary string.

```
<xs:element name="IngressProtector">
   <xs:annotation>
      <xs:documentation>The ingress protector.</xs:documentation>
   </xs:annotation>
   <xs:simpleType>
      <xs:restriction base="xs:base64Binary">
         <xs:minLength value="1"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

#### 2.2.3.2 HealthCertificate

The **HealthCertificate** element is a **base64-encoded** binary string of type **X.509** format received as input from the client for which Key Protection Services needs to be provided.

```
<xs:element name="HealthCertificate">
```


Copyright © 2017 Microsoft Corporation

```
      <xs:annotation>
         <xs:documentation>The health certificate.</xs:documentation>
      </xs:annotation>
      <xs:simpleType>
         <xs:restriction base="xs:base64Binary">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
```

#### 2.2.3.3 TransferKeyEncryptionAlgorithm

The **TransferKeyEncryptionAlgorithm** element denotes the algorithm to encrypt the wrapping key's transfer key.

```
   <xs:element name="TransferKeyEncryptionAlgorithm">
      <xs:annotation>
         <xs:documentation>The algorithm to be used to encrypt the wrapping key's transfer
   key.</xs:documentation>
      </xs:annotation>
      <xs:simpleType>
         <xs:restriction base="xs:anyURI">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
```

#### 2.2.3.4 WrappingKeyEncryptionAlgorithm

The **WrappingKeyEncryptionAlgorithm** element denotes the algorithm to encrypt the transport keys' wrapping key.

```
   <xs:element name="WrappingKeyEncryptionAlgorithm">
      <xs:annotation>
         <xs:documentation>The algorithm to be used to encrypt the transport keys' wrapping
   key.</xs:documentation>
      </xs:annotation>
      <xs:simpleType>
         <xs:restriction base="xs:anyURI">
            <xs:minLength value="1"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:element>
```

#### 2.2.3.5 TransportKeyEncryptionAlgorithm

The **TransportKeyEncryptionAlgorithm** element denotes the algorithm to encrypt the transport keys.

```
   <xs:element name="TransportKeysEncryptionAlgorithm">
      <xs:annotation>
         <xs:documentation>The algorithm to be used to encrypt the transport
   keys.</xs:documentation>
      </xs:annotation>
      <xs:simpleType>
         <xs:restriction base="xs:anyURI">
            <xs:minLength value="1"/>
```


Copyright © 2017 Microsoft Corporation

```
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

### 2.2.3.6 EgressProtector

The **EgressProtector** element denotes a **base64-encoded** string of type **UTF-8** format, which contains the entire egress protector as serialized to a file.

```
<xs:element name="EgressProtector">
   <xs:annotation>
      <xs:documentation>The egress protector containing the new transport
key.</xs:documentation>
   </xs:annotation>
   <xs:simpleType>
      <xs:restriction base="xs:base64Binary">
         <xs:minLength value="1"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

### 2.2.3.7 EncryptedTransferKey

The **EncryptedTransferKey** element denotes a **base64-encoded** string of type **UTF-8** format, which contains the wrapping key's transfer key encrypted by the health certificate.

```
<xs:element name="EncryptedTransferKey">
   <xs:annotation>
      <xs:documentation>The wrapping key's transfer key encrypted by the health
certificate.</xs:documentation>
   </xs:annotation>
   <xs:simpleType>
      <xs:restriction base="xs:base64Binary">
         <xs:minLength value="1"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

### 2.2.3.8 EncryptedWrappingKey

The **EncryptedWrappingKey** element denotes a **base64-encoded** string of type **UTF-8** format, which contains the transport keys’ wrapping key that is encrypted by the health certificate received from the client.

```
<xs:element name="EncryptedWrappingKey">
   <xs:annotation>
      <xs:documentation>The transport keys' wrapping key encrypted by the health
certificate.</xs:documentation>
   </xs:annotation>
   <xs:simpleType>
      <xs:restriction base="xs:base64Binary">
         <xs:minLength value="1"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```


*Copyright © 2017 Microsoft Corporation*

#### 2.2.3.9 EncryptedTransportKeys

The **EncryptedTransportKeys** element denotes a **base64-encoded** string of type **UTF-8** format, which contains the ingress and egress transport keys encrypted by the transport keys' wrapping key.

```
<xs:element name="EncryptedTransportKeys">
   <xs:annotation>
      <xs:documentation>The ingress and egress transport keys encrypted by the transport
keys' wrapping key.</xs:documentation>
   </xs:annotation>
   <xs:simpleType>
      <xs:restriction base="xs:base64Binary">
         <xs:minLength value="1"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

#### 2.2.3.10 Version

The following table summarizes the list of supported API versions.

| Version | Description |
|---|---|
| V1.0 | Original API version. |

**V1.0:**

**DataSize(4 bytes):** Total size of the **TransportKey** **BLOB**.

**Version(4 bytes):** Version of the **TransportKey** BLOB set to value 1.

**NumberOfKeys(4 bytes):** Total number of keys contained in the TransportKey BLOB.

**KeyLength(4 bytes):** The size of the key in bytes.

**KeyValue(Variable):** The actual key data of variable length.

#### 2.2.3.11 Certificate

The **Certificate** element is used to generate the key protectors.

```
<xs:element name="Certificate" type="Certificate_T" />
  <xs:simpleType name="Certificate T">
     <xs:annotation>
        <xs:documentation>A certificate in the DER-encoded binary X.509
format.</xs:documentation>
     </xs:annotation>
     <xs:restriction base="xs:base64Binary" />
  </xs:simpleType>
```

#### 2.2.3.12 Algorithm

The **Algorithm** element denotes the cryptographic algorithm identifier used to perform Key Protection Services.

```
<xs:element name="Algorithm" type="CryptoAlgorithm_T" use="required" />
  <xs:simpleType name="CryptoAlgorithm T">
     <xs:restriction base="xs:anyURI" />
```


*Copyright © 2017 Microsoft Corporation*

```
</xs:simpleType>
```


Copyright © 2017 Microsoft Corporation

# 3 Protocol Details

## 3.1 Server Details

### 3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

- The server implements the following:
  - **IngressProtector:** Contains the entire ingress protector as serialized to a file and converted to **base64-encoded** string as defined in section 2.2.3.1.
  - **Protector:** A collection of different cryptographic wrappings of the transport key as defined in section 2.2.2.3.
  - **Wrapping:** Consists of certificates of type base64-encoded strings and the transport key.
  - **EgressProtector:** A base64-encoded string of type **UTF-8** format, which contains the entire egress protector as serialized to a file as defined in section 2.2.3.6.
  - **PrimaryEncryptionCertificate:** A base64-encoded binary string of type **X.509** format as defined in section 2.2.3.11.
  - **PrimarySigningCertificate:** A base64-encoded binary string of type X.509 format as defined in section 2.2.3.11.
  - **Error:** A string representing the error response as defined in section 2.2.2.5.
  - **IngressTransportKey:** Key extracted from the ingress protector.
  - **EgressTransportKey:** Key generated from the **RollTransportKey BLOB** after Protector Validation.

### 3.1.2 Timers

None.

### 3.1.3 Initialization

**IngressProtector:** MUST be set to empty.

**Protector:** MUST be set to empty.

**Wrapping:** MUST be set to empty.

**EgressProtector:** MUST be set to empty.

**PrimaryEncryptionCertificate:** MUST be set to empty.

**PrimarySigningCertificate:** MUST be set to NULL.

**Error:** MUST be set to empty.


*Copyright © 2017 Microsoft Corporation*

**IngressTransportKey:** MUST be set to empty.

**EgressTransportKey:** MUST be set to empty.

### 3.1.4 Higher-Layer Triggered Events

None.

### 3.1.5 Message Processing Events and Sequencing Rules

#### 3.1.5.1 Service APIs

The following HTTP methods are allowed to be performed on this resource.

| HTTP method | Section | Description |
|---|---|---|
| RollTransportKey | 3.1.5.1.1 | Used to protect the keys by KPS. |
| GetMetaData | 3.1.5.1.2 | Retrieves or adds the list of valid certificates present in the KPS. |

##### 3.1.5.1.1 RollTransportKey

The following operations are allowed to be performed on this resource.

| HTTP method | Description |
|---|---|
| POST | Requests that a web server accept and store the data enclosed in the body of the request message. |

This operation is transported by an HTTP POST request.

The operation can be invoked through the following URI.

```
http://<server>/keyprotection/service/{version}/rolltransportkey
```

**Version**: Version of the **RollTransportKey** request as defined in section 2.2.3.10.

###### 3.1.5.1.1.1 Request Body

**RollTransportKey** handles the unwrapping of a **TransportKey** from a **Protector** by this KPS, as well as the generation of a new **TransportKey** and corresponding **Protector** for use in subsequent serialization of the object. The resulting new key protector will be owned by the original **Owner**, and Key Protection Service will be the **Guardian**.

The request body for this method contains the following as defined in section 2.2.2.1.

| Entry | Type |
|---|---|
| HealthCertificate | A certificate in **X.509** format that is converted to a **base64-encoded** string. |
| IngressProtector | Entire ingress protector as serialized to a file (**UTF-8** format, for example) and converted into a base64- |


Copyright © 2017 Microsoft Corporation

| Entry | Type |
|---|---|
| | encoded string. |
| TransferKeyEncryptionAlgorithm | The algorithm used to encrypt the wrapping key's transfer key defined in section 2.2.3.3. |
| WrappingKeyEncryptAlgorithm | The algorithm used to encrypt the wrapping key defined in section 2.2.3.4. |
| TransportKeyEncryptAlgorithm | The algorithm used to encrypt the transport key defined in section 2.2.3.5. |

#### 3.1.5.1.1.2 Response Body

The response body of this method contains the following as defined in section 2.2.2.2.

| Entry | Type |
|---|---|
| EgressProtector | A **base64-encoded** string of type **UTF-8** format, which contains the entire egress protector as serialized to a file as defined in section 2.2.3.6. |
| EncryptedTXBlob | The **BLOB** containing the **EncryptedTransferKey** as defined in section 2.2.3.7. |
| EncryptedTWBlob | The BLOB containing the **EncryptedWrappingKey** as defined in section 2.2.3.8. |
| EncryptedTKBlob | The BLOB containing the **EncryptedTransportKeys** as defined in section 2.2.3.9. |

A successful operation returns status code 200 (OK). Otherwise, an error is returned.

The response message for this method can result in the following status codes.

| Status code | Description |
|---|---|
| 200 | OK |
| 204 | No content. |

#### 3.1.5.1.1.3 Processing Details

The server MUST perform the following steps after receiving **RollTransportKey**.

- Validate the **HealthCertificate** in an implementation-specific manner and return an error "HealthCertificateException" if validation fails.
- Validate that the **IngressProtector** is in a valid XMLformat and returns the error "InvalidProtectorException" with error code 204 if the Guardian Wrapping is NULL.
- Verify that **Protector** has a wrapping, process the **IngressProtector** and extract the **IngressTransportKey**, generate **EgressTransportKey**, generate an **EgressProtector**, and


Copyright © 2017 Microsoft Corporation

encrypt and sign both **IngressProtector** and **EgressProtector** in an implementation-specific manner.

Returns **IngressProtector, EgressProtector,** and **EncryptedTransportKeys** to the calling application.

#### 3.1.5.1.2 GetMetaData

The following operations are allowed to be performed on this resource.

| HTTP method | Description |
|---|---|
| GET | Retrieves information from the server. |

This operation is transported by an HTTP GET request.

The operation can be invoked through the following URI:

```
http://<server>/keyprotection/service/metadata/2014-07/metadata.xml
```

The list of possible certificates includes **PrimaryEncryptionCertificate**, **PrimarySigningCertificate** as defined in section 2.2.3.11, and non-primary certificates, including **OtherSigningCertificates**, as defined in section 2.2.3.11.

The KPS metadata consists of:

- Optional metadata about the **Custodian** or **Owner**.
- The KPS **PrimarySigningCertificate**.
- The KPS **PrimaryEncryptionCertificate**.
- The signature over the KPS encryption certificate by the KPS signing private key.

##### 3.1.5.1.2.1 Request Body

The following operations are allowed to be performed on this resource.

| HTTP method | Description |
|---|---|
| GET | Retrieves information from the server. |

This operation is transported by an HTTP GET request.

The operation can be invoked through the following URI:

```
http://<server>/keyprotection/service/metadata/2014-07/metadata.xml
```

The list of possible certificates includes **PrimaryEncryptionCertificate**, **PrimarySigningCertificate** as defined in section 2.2.3.11, and non-primary certificates, including **OtherSigningCertificates**, as defined in section 2.2.3.11.

The KPS metadata consists of:

- Optional metadata about the **Custodian** or **Owner**.
- The KPS **PrimarySigningCertificate**.
- The KPS **PrimaryEncryptionCertificate**.

The signature over the KPS encryption certificate by the KPS signing private key.


*Copyright © 2017 Microsoft Corporation*

#### 3.1.5.1.2.2 Response Body

The response body of this method contains the following.

**GetMetadata** computes a new metadata document. A successful operation returns status code 200 (OK). Otherwise, an error is returned.

The response message for this method can result in the following status codes.

| Status code | Description |
|---|---|
| 200 | OK |
| Error | A string representing the error response as defined in section 2.2.2.5. |

#### 3.1.5.1.2.3 Processing Details

The server MUST perform the following steps after receiving **GetMetaData**:

- Validate whether **PrimaryEncryptionCertificate** is found in the registry. If **PrimaryEncryptionCertificate** is not found, return the error string "Primary Encryption Certificate not found".
- Validate whether **PrimarySigningCertificate** is found in the registry. If **PrimarySigningCertificate** is not found, return error string "Primary Signing Certificate not found".
- If include **OtherSigningCertificates** is TRUE, include non-primary certificates, and perform the wrapping of the primary certificates in an implementation-specific manner and return the new metadata of the Key Protection Service.

Otherwise, perform wrapping of the primary certificates in an implementation-specific manner and return the new metadata of the Key Protection Service.

### 3.1.6 Timer Events

None.

### 3.1.7 Other Local Events

None.

## 3.2 Client Details

### 3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

**Retries:** An integer that indicates the number of retries to send the **RollTransportKey** request.


Copyright © 2017 Microsoft Corporation

### 3.2.2 Timers

None.

### 3.2.3 Initialization

**Retries:** A default value that is equivalent to the number of distinct host addresses available for the server URI.

### 3.2.4 Higher-Layer Triggered Events

The following sections describe the operations performed by the client in response to events triggered by higher-layer applications.

#### 3.2.4.1 Application Requests RollTransportKey

The application provides the following:

- Health Certificate issued by an Attestation Service as defined in [MS-HGSA] and accredited by the KPS.

The client MUST perform the following:

- Create a valid XML **BLOB** containing **RollTransportKeyRequest** as defined in section 2.2.2.1.
- Perform the steps as specified in section 3.2.5.1.

#### 3.2.4.2 Application Requests GetMetaData

The application provides the following:

- A **GetMetaData** request to retrieve the list of KPS-supported certificates as defined in section 2.2.1.2 to verify **Protector** was properly signed by the KPS.

The client MUST perform the following:

- Create a valid XML **BLOB** containing **GetMetaData**.
- Perform the steps as specified in section 3.2.5.2.

### 3.2.5 Message Processing Events and Sequencing Rules

The following sections describe the sequence of operations performed by the client in **RollTransportKey** and **GetMetaData** scenarios.

#### 3.2.5.1 RollTransportKey

The client MUST send a POST request on the **RollTransportKey** resource as specified in section 3.1.5.1.1 by using the URI specified.

If the client receives the **RollTransportKeyResponse** specified in section 2.2.2.2 with the status code 200(OK), the client's health certificate is protected and the guarded host is enabled to run securely on a VM.

If the client receives an error, the client MAY retry sending the **RollTransportKey** request based on **Retries**.


*Copyright © 2017 Microsoft Corporation*

#### 3.2.5.2 GetMetaData

The client MUST send a POST request on the **GetMetaData** resource as specified in section 3.1.5.1.2 by using the URI specified.

If the operation is successful, the client receives the metadata content with status code 200(OK).

### 3.2.6 Timer Events

None.

### 3.2.7 Other Local Events

None.


Copyright © 2017 Microsoft Corporation

# 4 Protocol Examples

None.


Copyright © 2017 Microsoft Corporation

# 5 Security

## 5.1 Security Considerations for Implementers

None.

## 5.2 Index of Security Parameters

None.


Copyright © 2017 Microsoft Corporation

# 6 Appendix A: Full XML Schema

For ease of implementation, the following is the full XML schema for this protocol.

| Schema name | Prefix | Section |
|---|---|---|
| Protector Schema | P | 6.1 |
| RollTransportKey Request | Req | 6.2 |
| RollTransportKey Response | Res | 6.3 |
| MetaData Response | M | 6.4 |
| Crypto Schema | Not applicable | 6.5 |

## 6.1 Protector Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://schemas.microsoft.com/kps/2014/07"
           elementFormDefault="qualified"
           xmlns="http://schemas.microsoft.com/kps/2014/07"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:include schemaLocation="Crypto.xsd" />

   <xs:complexType name="SigningCertificateSignature_T">
      <xs:annotation>
         <xs:documentation>The signing certificate signature is computed using the
signing certificate of the parent wrapping (specified by ParentWrappingId) over this
wrapping's signing certificate.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="Signature" type="Signature_T" />
      </xs:sequence>
      <xs:attribute name="ParentWrappingId" type="xs:unsignedInt" use="required" />
   </xs:complexType>

   <xs:complexType name="EncryptionCertificateSignature_T">
      <xs:annotation>
         <xs:documentation>The encryption certificate signature is computed using this
wrapping's signing certificate over this wrapping's encryption
certificate.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="Signature" type="Signature_T" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="TransportKey_T">
      <xs:sequence>
         <xs:element name="EncryptedData" type="EncryptedData_T" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="Wrapping_T">
      <xs:sequence>
         <xs:element name="Id" type="xs:unsignedInt" />
         <xs:element name="SigningCertificate" type="Certificate_T" />
```


Copyright © 2017 Microsoft Corporation

```
         <xs:element name="SigningCertificateSignature"
type="SigningCertificateSignature_T" />

         <xs:element name="EncryptionCertificate" type="Certificate_T" />
         <xs:element name="EncryptionCertificateSignature"
type="EncryptionCertificateSignature_T" />
         <xs:element name="TransportKey" type="TransportKey_T" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="WrappingCollection T">
      <xs:sequence>
         <xs:element name="Wrapping" type="Wrapping_T" minOccurs="1"
maxOccurs="unbounded" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="TransportKeySignature_T">
      <xs:annotation>
         <xs:documentation>The transport key signature is computed using a key derived
from the actual transport key over the entire Wrappings element after exclusive xml
canonicalization (http://www.w3.org/2001/10/xml-exc-c14n#) and conversion to UTF-
8.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="KeyDerivationMethod" type="KeyDerivationMethod T" />
         <xs:element name="Signature" type="Signature_T" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="GuardianSignature_T">
      <xs:annotation>
         <xs:documentation>The guardian signature is computed using the signing
certificate specified by WrappingId over the entire Wrappings element after exclusive xml
canonicalization (http://www.w3.org/2001/10/xml-exc-c14n#) and conversion to UTF-
8.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="Signature" type="Signature_T" />
      </xs:sequence>
      <xs:attribute name="WrappingId" type="xs:unsignedInt" use="required" />
   </xs:complexType>

   <xs:element name="Protector" type="Protector_T" />

   <xs:complexType name="Protector T">
      <xs:annotation>
         <xs:documentation>A protector contains a list of wrappings of the transport
key.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="Wrappings" type="WrappingCollection_T" />
         <xs:element name="TransportKeySignature" type="TransportKeySignature T" />
         <xs:element name="GuardianSignature" type="GuardianSignature_T" />
      </xs:sequence>
   </xs:complexType>
</xs:schema>
```

## 6.2 RollTransportKey Request Schema

RollTransportKey request schema is defined in section 2.2.2.1

## 6.3 RollTransportKey Response Schema

RollTransportKey response schema is defined in section 2.2.2.2.


Copyright © 2017 Microsoft Corporation

## 6.4 MetaData Resposne Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://schemas.microsoft.com/kps/2014/07"
           elementFormDefault="qualified"
           xmlns="http://schemas.microsoft.com/kps/2014/07"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   <xs:import namespace="http://www.w3.org/2000/09/xmldsig#" />

   <xs:include schemaLocation="Crypto.xsd" />

   <xs:complexType name="GuardianInformation_T">
      <xs:annotation>
         <xs:documentation>The guardian information for an entity.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="Version" type="xs:unsignedInt" />
         <xs:element name="EncryptionCertificate" type="Certificate_T" />
         <xs:element name="SigningCertificate" type="Certificate_T" />
         <xs:element name="EncryptionCertificateSignature" type="Signature_T" />
         <xs:element name="SigningCertificateSelfSignature" type="Signature_T" />
         <xs:element name="OtherSigningCertificates" type="CertificateCollection_T"
minOccurs="0" />
      </xs:sequence>
   </xs:complexType>

   <xs:element name="Metadata" type="Metadata_T" />

   <xs:complexType name="Metadata_T">
      <xs:annotation>
         <xs:documentation>The metadata document contains information about the
entity.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
         <xs:element name="GuardianInformation" type="GuardianInformation_T" />
         <xs:element ref="ds:Signature" />
         <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="ID" use="optional" type="xs:ID" />
      <xs:attribute name="Version" use="required" type="xs:unsignedInt" />
      <xs:anyAttribute namespace="##any" processContents="lax" />
   </xs:complexType>
</xs:schema>
```

## 6.5 Crypto Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://schemas.microsoft.com/kps/2014/07"
           elementFormDefault="qualified"
           xmlns="http://schemas.microsoft.com/kps/2014/07"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:complexType name="CryptoParameters_T">
      <xs:sequence>
         <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
   </xs:complexType>

   <xs:simpleType name="CryptoAlgorithm_T">
      <xs:restriction base="xs:anyURI" />
   </xs:simpleType>
```


Copyright © 2017 Microsoft Corporation

```
   <xs:complexType name="Signature_T">
      <xs:sequence>
         <xs:element name="Parameters" type="CryptoParameters_T" minOccurs="0" />
         <xs:element name="SignatureValue">
            <xs:simpleType>
               <xs:restriction base="xs:base64Binary" />
            </xs:simpleType>
         </xs:element>
      </xs:sequence>
      <xs:attribute name="Algorithm" type="CryptoAlgorithm_T" use="required" />
   </xs:complexType>

   <xs:element name="EncryptedData" type="EncryptedData_T" />

   <xs:complexType name="EncryptedData_T">
      <xs:sequence>
         <xs:element name="Parameters" type="CryptoParameters T" minOccurs="0" />
         <xs:element name="CipherValue">
            <xs:simpleType>
               <xs:restriction base="xs:base64Binary" />
            </xs:simpleType>
         </xs:element>
      </xs:sequence>
      <xs:attribute name="Algorithm" type="CryptoAlgorithm_T" use="required" />
   </xs:complexType>

   <xs:complexType name="KeyDerivationMethod_T">
      <xs:sequence>
         <xs:element name="Parameters" type="CryptoParameters T" minOccurs="0" />
      </xs:sequence>
      <xs:attribute name="Algorithm" type="CryptoAlgorithm T" use="required" />
   </xs:complexType>

   <xs:simpleType name="Certificate T">
      <xs:annotation>
         <xs:documentation>A certificate in the DER-encoded binary X.509
format.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:base64Binary" />
   </xs:simpleType>

   <xs:complexType name="CertificateCollection T">
      <xs:sequence>
         <xs:element name="Certificate" type="Certificate_T" minOccurs="0"
maxOccurs="unbounded" />
      </xs:sequence>
   </xs:complexType>
</xs:schema>
```


Copyright © 2017 Microsoft Corporation

# 7 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

- Windows 10 v1703 operating system
- Windows Server 2016 operating system
- Windows Server operating system

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.


*Copyright © 2017 Microsoft Corporation*

# 8 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.


Copyright © 2017 Microsoft Corporation

# 9 Index


**A**

Abstract data model
  client 26
  server 22
Applicability 6

**C**

Capability negotiation 6
Change tracking 36
Client
  Abstract data model 26
  Higher-layer triggered events 27
  Initialization 27
  message procesing 27
  Message processing events and sequencing rules 27
  Other local events 28
  sequencing rules 27
  Timer events 28
  Timers 27
Complex types 8

**D**

Data model – abstract
  client 26
  server 22

**E**

Examples 29

**F**

Fields - vendor-extensible 6
Full XML schema 31

**G**

Glossary 5

**H**

Higher-layer triggered events
  client 27
  server 23
HTTP methods 7

**I**

Implementer - security considerations 30
Index of security parameters 30
Informative references 6
Initialization
  client 27
  server 22
Introduction 5

**L**

Local events
  client 28
  server 26

**M**

Message processing
  client 27
    GetMetaData 28
    RollTransportKey 27
  server - service APIs 23
Messages
  complex types 8
  HTTP methods 7
  simple types 16
  transport 7

**N**

Normative references 5

**O**

Overview (synopsis) 6

**P**

Parameters - security index 30
Preconditions 6
Prerequisites 6
Product behavior 35

**R**

References
  informative 6
  normative 5
Relationship to other protocols 6

**S**

Security
  implementer considerations 30
  parameter index 30
Sequencing rules
  client 27
  server 23
Server
  Abstract data model 22
  Higher-layer triggered events 23
  Initialization 22
  message processing 23
  Other local events 26
  sequencing rules 23
  Timer events 26
  Timers 22
Simple types 16
Standards assignments 6

**T**




*Copyright © 2017 Microsoft Corporation*

Timer events
  client 28
  server 26
Timers
  client 27
  server 22
Tracking changes 36
Transport 7

## V

Vendor-extensible fields 6
Versioning 6

## X

XML schema 31


Copyright © 2017 Microsoft Corporation