Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 6, 2011

S. Turner
IECA

March 7, 2011

Secure Object Delivery Protocol (SODP)
draft-turner-sodp-00.txt

Abstract

This document describes the Secure Object Delivery Protocol (SODP). SODP enables clients to access secure packages produced by a Key Management Systems (KMS). Client access is ideally direct and web-based, but access via agents acting on behalf of clients is supported.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 6, 2011.

Copyright Notice

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

The Secure Object Delivery Protocol (SODP) enables clients to obtain secured packages from a supporting Key Management System (KMS). Client access is via the HyperText Transfer Protocol (HTTP) over Transport Security Layer (TLS). Clients can directly access the KMS or an agent can act on the client's behalf. Clients access the KMS to retrieve a Product Availability List (PAL), which provides the location of their packages with a User Resource Identifier (URI), or can directly retrieve the package if the client obtains the URI via another method. Packages are secured using the Cryptographic Message Syntax (CMS).

The remainder of this document will explain the SODP model, provide requirements for the KMS, client, and agent, as well specify the PAL format.

1.1 Definitions

Agent: An entity that performs functions on behalf of a client.
Asymmetric Key Package: A package that includes an asymmetric key content type [RFC5959].

Certificate Management Packages: A package that contains a PKI Data or PKI Response content types [RFC5272][RFC5912].

Clients: An entity that contains one or more End Cryptographic Unit (ECU). Clients consume products generated by the Key Management

System (KMS).

Encrypted Key Package: A package that includes an encrypted key content type [RFC6032].

Firmware Package: A package that contains a firmware content type [RFC4108][RFC5911].

NOTE: [RFC4108] defines the semantics for the firmware content type's fields. [RFC5911] provides the 2002 ASN.1 definitions.

Identity and Authentication (IA) Key/Certificate: Key/Certificate

used to support IA of the client, when the client communicates with the KMS as well as with other end-entities. It provides the KMS or other end-entities with an appropriate degree of confidence in the client's identity before delivering products, services or sensitive information to the client.

Key Exchange (KE) Key/Certificate: Key/Certificate used when the client and the KMS or other end-entity must cooperatively create a wrapping key to protect the delivery of products or sensitive information for use by the client. It is also used to establish secure sessions (e.g., TLS) from a client to the KMS. Other examples include traffic encryption keys and transmission security keys.

Key Management System (KMS): A set of one or more components that is designed to protect, manage, and distribute cryptographic products. In this document, cryptographic products are referred to as packages.

Operator: A person who "runs" the device (e.g., network administrator).

Package: An object that contains one or more CMS content types. At a minimum, all packages are protected using the CMS [RFC5652] SignedData structure. There are numerous types of packages: Asymmetric, Certificate Management, Encrypted Key, Firmware, Publication, and Symmetric Packages.

NOTE: This document does not define any packages they are all defined elsewhere. Product Availability List (PAL): A PAL is an XML file that furnishes information for KMS service messages that

are currently available and authorized for retrieval by a client or agent.

Publication Package: A package that contains certificates and Certificate Revocation Lists (CRLs). These are typically additional CA certificates or CRLs not provided as part of other packages. The package is a degenerate CMS SignedData, which is sometimes referred to as a "certs-only" message.

Service Messages: KMS-produced packages are the instantiation of the KMS services. This document defines three services that manifest in three types of service messages: publication, distribution, and certificate management. One, registration, does not manifest itself in a service message.

Source Authority: A source authority is trusted by clients to generate particular package types. Clients determine this by validating the digital signature on the package back to a Trust Anchor (TA).

Sponsor: A person that is accountable for use of the client's identity. This may or may not be the entity that operates the client (i.e., the operator).

Symmetric Key Package: A package that contains a symmetric key content type [RFC6031].

Trust Anchor (TA): From [RFC5934], a TA contains a public key that is used to validate digital signatures. In this document, a TA represents an authoritative entity via a public key and associated data. The public key is used to verify digital signatures and the associated data is used to constrain the types of information for which the TA is authoritative. A relying party uses TAs to determine if a digitally signed object is valid by verifying a digital signature using the TA's public key, and by enforcing the constraints expressed in the associated data for the TA.

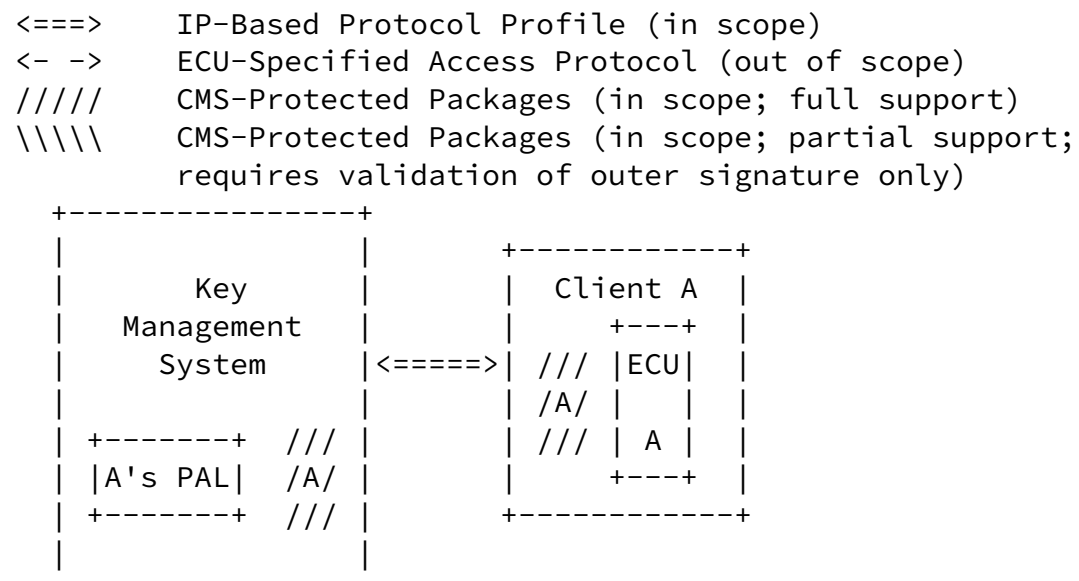
1.2 Key Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and

"OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. SODP Model

Figure 1 depicts the SODP model. It is comprised of three entities: the key management system, one or more clients, and agents acting on behalf of clients. KMS-to-client and KMS-to-agent protocol interactions are in-scope; agent-to-client protocol interactions are out-of-scope. KMS-to-client and KMS-to-agent interactions support mutual authentication, provide integrity, and optionally provide confidentiality through the use of HTTPS. Confidentiality for KMS-to-client and KMS-to-agent interactions is OPTIONAL because when confidentiality is needed the packages are encrypted for the client. See Section 10 for requirements on cryptographic suites.



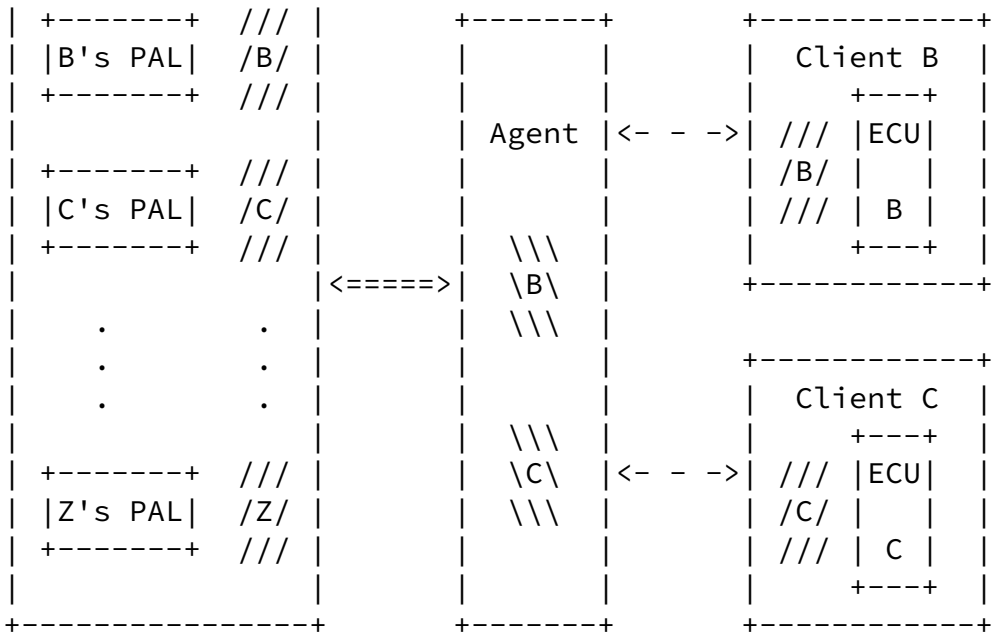


Figure 1 - SODP Model

Clients or agents access the KMS via HTTPS to retrieve and inspect their PAL. The PAL is an XML file that contains Uniform Resource Identifiers (URIs) for client packages. Retrieval of packages referenced in the PAL delivers KMS services to the client. Alternatively, clients can retrieve packages directly from the KMS if they obtain URIs from another source. KMS services are discussed in Section 2.1; the PAL and URI format are discussed in Section 5.

While the KMS is viewed as being a single entity, operationally the issuance of different packages can be assigned to different authorities within the KMS. These authorities are referred to as source authorities. A source authority is trusted by clients to

generate particular package types. Entities validate their source authorities when validating the digital signature(s) in/on packages. That is, when a client retrieves a package referred to in their PAL the client ensures that the signatures in/on the package validate to an installed trust anchor (TA). See Section 4.1 for more information on clients' TAs.

Packages may be encrypted for the client. Packages that contain

cleartext (i.e., unencrypted) symmetric keys or asymmetric private keys MUST be encrypted for the client to ensure that the keys are not disclosed to another party. Relying on encrypted packages instead of relying on HTTPS-encrypted links allows agents to further distribute the packages to clients without disclosing the cleartext to the agent. Encrypted packages also enable alternate distribution paths such as store-and-forward, which is beyond the scope of this document. Package requirements are discussed in Section 4.

Prior to clients accessing the KMS, clients need be registered with the KMS. The process for this will vary. One possible process involves sponsorship by an individual. This individual collects information about the client and enters the information into the KMS's database. Also during this time, the client is assigned an initial identity. Once registered, the client is issued a certificate, which is later used to access the KMS. Clients and agents use what is referred to as IA certificates when communicating with the KMS. An IA certificate provides the client's/agent's identity and allows the KMS to authenticate that the entity accessing the KMS is in fact the client/agent. The registration and client IA certificate issuance process is described in more detail in Section 3.1. The format and protocol for communicating the registration data and sending the initial IA certificate directly to client is out-of-scope. The client authenticates itself to the KMS with this certificate using HTTPS. After the IA certificate is installed, the client requests a KE certificate. KE certificates allow clients to perform key establishment with the KMS to decrypt/encrypt packages.

Some implementations may require further separation for some clients who are issued another set of certificates that support client-to-client interactions, which is the client's joie de vivre or the client's mission. The initial certificate set is only used to communicate with the KMS and the second set is only ever used to communicate with other clients. In this case the first set is referred to as IA(I)/KE(I) certificates for (I)nfrastructure certificates and the second set is referred to as IA(M)/KE(M) certificates for (M)ission certificates. Not all clients need the second set of certificate, if clients only need symmetric key, then only one set of certificates is issued. *(I) certificates are issued to it and instead of IA(M)/KE(M) certificates issued later only

symmetric key packages are provided.

3. Key Management System

The SODP is the interface to the KMS that clients use to access KMS-services and associated KMS-generated packages. The internal components of the KMS and their interactions are out-of-scope. However, if a KMS provides all of the KMS packages (see Section 3.2), it will need the capability to package trust anchors (TAs), generate and package symmetric keys, package firmware, generate and package asymmetric keys, issue and package public key certificates, and issue and package Certificate Revocation Lists (CRLs). It will also need to generate and receive packages, which includes generating and verifying digital signatures on packages as well as encrypting and decrypting of packages. Additionally, it will need a repository to store information about clients and their packages.

The remainder of this section is split in to two parts. The first part, Section 3.1, describes the KMS services and the second part, Section 3.2, describes the KMS package requirements.

3.1. KMS Services

This section addresses the four services provided by the KMS: Registration, Distribution, Publication, and Certificate Management. The latter three services are instantiated in packages.

3.1.1. Registration Service

The KMS only provides services to clients that are KMS-registered. Registration information collected is KMS-specific. However, the information collected MUST include a permanent identifier that is used to identify the client throughout its lifecycle. This permanent identifier is referred to as an Electronic Serial Number (ESN). See Section 6 for more information on ESNs.

Other OPTIONAL information to collect includes:

- Client Manufacturer
- Client Name
- Client Type

The KMS could also assign a KMS user number for an internal index, label, or abbreviated name for associating data elements pertaining to that user. This number is not sent to the client and is only used by the KMS.

During this step the client is also assigned an identity, which the

KMS stores in its database. At a minimum the identity is an identifier but it can also include additional information such as a client's sponsor (e.g., Alexa Morris), the client's operator (e.g., Alexa Morris), and the sponsor's organizational affiliation (e.g., AMS). That is, the KMS MUST assign and record an identifier to the client, but recording other client-related identity data is OPTIONAL. Additionally:

- For cases where the sponsor isn't the entity that operates the client, the identity can also include an indication of the entity operating the client. This allows the network group to sponsor the client, but the security group to operate the client (i.e., network groups say it's okay to add client to the network but doesn't want to manage the clients keys).

- For cases where the client can be transferred from one operator to another, the identity MUST include identity of the previous operator. This provides a "chain-of-control" over the device for its lifetime. A KMS can support a wide variety of environments:

- For a KMS that support non-X.509 certificate and non-X.509 CRL types, the identity SHOULD include an indication of certificate type.

  NOTE: This supports cases where the client uses alternate certificate formats such as Pretty Good Privacy (PGP) [RFC4880]. Alternative certificate formats are supported by many security protocols including Internet Key Exchange v2 (IKEv2) [RFC5996], TLS [RFC5246], and CMS [RFC5652].

- For a KMS that supports humans as well as clients, the identity SHOULD include an indication of the type of user (e.g., client/device, human, administrator).

The KMS MUST ensure that the client identity is KMS-unique. That is, the collection of data that comprises the client identity MUST NOT match another client served by the KMS. After this check passes, the final step in the registration process occurs: client IA certificate issuance. The KMS MUST issue a certificate [RFC5280] to the client that contains the client's permanent identifier (see Section 6). NOTE: 1) The process for delivering the IA certificate directly to the client is out-of-scope; 2) the format and protocol for communicating the registration data is out-of-scope; and 3) the client need not contribute to or respond to the supplied identity information.

3.1.2. Distribution Service

The KMS employs the distribution service to provide clients' access to their packages. The KMS provides access to packages through the use of URIs, which uniquely refers to specifically CMS-wrapped packages for delivery to the target client. The KMS generates a PAL that clients can use to retrieve packages. Alternatively, the client can directly access the package, but this assumes the client obtained the URI(s) via another mechanism, which is out-of-scope. Packages include symmetric key packages as well as centrally-generated asymmetric key packages.

> NOTE: Certificates associated with client generated asymmetric keys (i.e., locally-generated public-private keys) are delivered via the Certificate Management Service (See Section 3.1.3).

Figure 2 depicts an example ladder diagram for a protocol flow. The first step is to establish a mutually authenticated HTTPS connection between the client/agent and KMS. The client then requests their PAL from the KMS (via HTTP GET). The KMS replies with the client's PAL (via HTTP GET Response). Once a client has successfully downloaded their PAL, it will process it to obtain the included packages(s). The processing provided will depend on the PAL entry. Section 3.2 details the KMS-package requirements, Section 4 details clients-package requirements, and Section 5 details agent-package requirements.

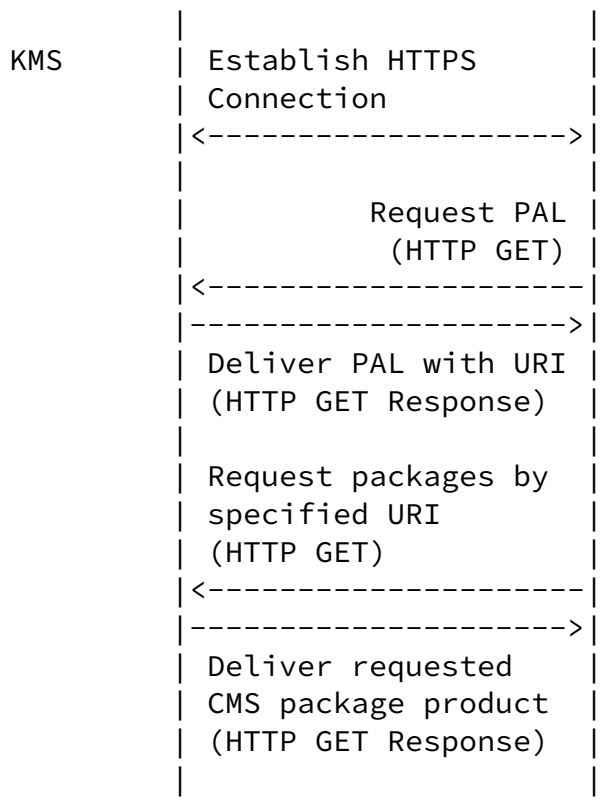


Figure 2 - SODP Distribution Service Message Sequence

A device can request (via HTTP GET) and download (via HTTP GET Response) any, or all, packages and new PALs by repeating the necessary sequence of steps. When the client is finished, it SHOULD terminate the connection. See Section 8 for more information on SODP's HTTP requirements.

The KMS MUST support generation of a PAL. The KMS MUST support access to client packages directly and through a PAL.

3.1.3. Publication Service

The KMS Publication Service provides clients that are PKI subscribers and relying parties with a means to obtain publicly-available, ancillary services related to PKIs namely: Certificates, CRLs, Certificate Policies (CPs), and Certificate Practice Statements (CPSs) packages. The KMS MUST support distribution of CRLs but MAY support distribution of CPs and CPSs.

> NOTE: CPs and CPSs are the one exception to the Package definition found in section 1.1. CPs and CPSs are not encapsulated in CMS, they are URIs to the location on the KMS for the CP and CPS.

Certificates delivered can include additional CA certificates or peer

client certificate(s).

Clients may elect to obtain the CRLs that they rely on from sources other than the (e.g., a local directory).

CRLs are offered in the form, or forms, produced by the responsible Certification Authority (CA). The form of the CRL is transparent to the KMS Publication Service. CAs may choose to publish compact versions of CRLs (e.g., partitioned CRLs) that are compatible with a disadvantaged client within the overall subscriber population. The PAL provided to a client will always contain a URI for the most current version of each CRL needed to verify the packages in the form used by the particular client. The KMS Publication Service will not list CRLs that a client does not need or cannot use. Based on its capabilities, the freshness of currently held CRLs, and the circumstances, the client will determine whether it needs to download each offered CRL. KMS Publication Services packages will be signed, but need not be encrypted. The information in the package is already signed; CAs sign the certificates and CRLs so there is no need to sign a package containing them.

> NOTE: The KMS Publication Service is not meant to be a general repository for all relying parties. Access is only provided to registered clients.

3.1.4. Certificate Management Service

The KMS Certificate Management Service allows a client to develop an asymmetric key pair and obtain the public key certificate associated with the key pair. It additionally provides certificates and CRLs necessary to validate the asymmetric key pair to an installed TA.

The KMS Certificate Management Service supports two kinds of certificate management processes:

- Issuance: Where a new public/private key pair is established for a KE certificate.
- Rekey: Where an existing IA certificate is provided with new keying material.

CA MUST generate public key certificates in accordance with [RFC5280]. A Registration Authority (RA) may be used to register subscribers as well as assist the CA when issuing and rekeying certificates for clients.

3.2. KMS Packages

The KMS Distribution, Publication, and Certificate Management services translate into KMS packages. The primary packages are key packages, but they also include firmware packages necessary to use the key packages, TAMP packages to validate the package's source of authority, publication packages that contain additional certificates and CRLs, and collections of key packages. This section lists the package requirements for the KMS.

There are many different key packages, but at their core there are three types:

- Symmetric key packages are defined in [RFC6031]. A symmetric key package can contain one or more symmetric keys. It also can contain attributes that apply to one or more keys. The KMS MUST support the ct-symmetric-key-package content type encapsulated in a ct-signed-data content type [RFC5652][RFC5911].

- Asymmetric key packages are defined in [RFC5958]. An asymmetric key package can contains one or more private asymmetric keys and associated algorithm parameters. It can also contain the public key and other attributes. This key package is used in conjunction with the certificate management packages when the KMS generates the client's key pair. The KMS MUST support the ct-asymmetric-key-package content type encapsulated in a ct-signed-data content type.

- Certificate management packages are defined in [RFC5272][RFC5912]. PKI Data and PKI Response content types are used to manage public key certificates [RFC5280]. The KMS MUST support the ct-PKIData and ct-PKIResponse content types. The KMS MUST also support encapsulating ct-PKIData in the ct-signed-data content type.

Distribution of the symmetric and asymmetric key packages require that these keys be disclosed only to the client and to not to anyone else. The key packages needs to be enveloped. The encrypted key package [RFC6032] supports encrypting key packages in one of three ways: with key exchange algorithms (i.e., using EnvelopedData), with previously distributed symmetric algorithms (i.e., using EncryptedData), and with authenticated-encryption algorithms (i.e., using AuthEnvelopedData). The KMS MUST support the ct-encrypted-key-package content type and the EnvelopedData choice (i.e., support ct-enveloped-data). The KMS MUST support encapsulating ct-encrypted-key-package in a ct-signed-data content type.

The KMS distributes object code for implementing one or more

cryptographic algorithms in a cryptographic module and software to implement a communications protocol with the Firmware package [RFC4108][RFC5911]. The KMS MUST support the ct-firmwarePacakge content type. It MUST support receipt of the ct-firmwareLoadReceipt and ct-firmwareLoadError content types. The KMS MUST support encapsulating the ct-firmwarePackage content type in a ct-signed-data content type.

To support sending multiple package types to a client, the KMS can use the Content Collection [RFC4073] CMS content type. To allow the KMS to apply additional attributes to the package the can use the

Content With Attributes [RFC4073] CMS content type. The KMS SHOULD support the ct-contentCollection any MAY support the ct-contentWithAttributes content type. The KMS MUST support encapsulating these in a ct-signed-data content type.

The publication package is supported by the KMS with the "certs-only" package [RFC5751], which is a CMS SignedData with no content just CRLs and certificates. The KMS MUST support the "certs-only" package with ct-data content type with no eContent. The KMS manages TAs to support validating packages with the Trust Anchor Management Protocol (TAMP) [RFC5934]. TAMP supports multiple formats for the TA. The KMS MUST support the Certificate choice. The KMS MUST support the tamp-update content type [RFC5934]. As specified in [RFC5934], tamp-update MUST be encapsulated in a ct-signed-data content type.

TO DO: Add TAMP to Service Identifiers.

The KMS MUST support validating package signatures back to a TA [RFC5652][RFC5280].

4. Client

Clients use SODP to access the KMS-services and associated KMS-generated packages. This section addresses client registration, use, and package requirements.

4.1. Registration

Section 3.1.1 addresses client registration. As noted there, the client need not contribute to or respond to the supplied identity information. After registration is completed, the client is supplied with an IA certificate. Prior to using this certificate, the client MUST verify that the certificate back to an installed trust anchor. The number of TAs is implementation KMS-specific, but in general:

- If the client supports locally-generated asymmetric keys, then it MUST support at least one TA.

- If the client support centrally-generated asymmetric keys, then it MUST also support at least one TA.

- If the client supports symmetric keys, then it MUST support two

TAs: one for symmetric keys and one for the asymmetric keys (i.e., the PKI Root).

- If the client support firmware, the it MUST support two TAs: one for the firmware and one for the asymmetric keys (i.e., the PKI Root).

More complicated scenarios are possible. For example in Figure 3, a KMS and client support centrally-generated asymmetric keys. The KMS supports two TAs: one for the certificate and one for the asymmetric keys (a Key TA (KTA)). The KTA delegates source authority to a Key Source Authority (KSA) and distribution authority to a Key Distribution Authority (KDA). The KSA creates the asymmetric key places it in the symmetric key content type, signs it (signed data content type), includes the corresponding certificate, and encrypts it (encrypted key content type). The KDA applies an additional signature layer around the encrypted data. Upon receipt the client validates KDA's certificate and signature to the KTA, decrypt the message, the KSA's signature and certificates to the KTA, the client validates their certificate to the PKI TA, and the client checks that the private key corresponds to the public key.

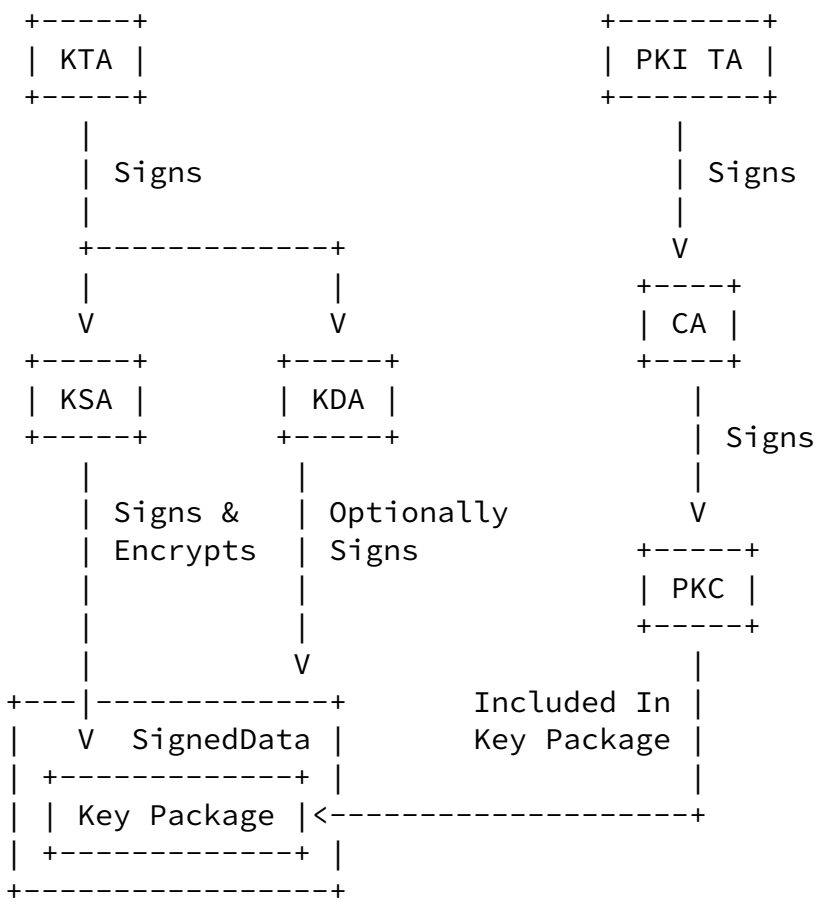


Figure 3 - Example Authority Architecture

4.2. Activation and Operation

The activation/operation phase of the client lifecycle is where the client performs its prime mission (e.g., secure Voice Over IP (VoIP), cable box).

Activation can occur immediately following registration, when the client receives an IA certificate. Activation can also occur when the client resides at and is associated with its intended operator (i.e., the client is registered and sponsored in the Canada but not activated by the operator until it arrives where they are located in Greenland). In other words, the client can be immediately actived or it can occur at a later time.

> NOTE: A client only needs to be loaded with an IA key to perform KMS Services.

If the client needs additional certificates (e.g., for confidentiality or separate mission certificates), the client or agent can retrieve them via the PAL. Client retrieval of packages via the PAL is OPTIONAL. Clients may elect to obtain product package

URI information using a different mechanism (e.g., inputs from a

human or agent).

4.3. Packages

Client support for packages varies depending on the type of service they desire. All clients MUST support the ct-signed-data content type to ensure the packages source of authority can be determined. They MUST also support validating package signatures back to a TA [RFC5652][RFC5280].

For clients that support symmetric key packages [RFC6031], they MUST support the ct-symmetric-key-package content type. Additionally, clients MUST support the ct-encrypted-key-package content type and the EnvelopedData choice (i.e., support ct-enveloped-data) to support encrypting the cleartext symmetric key.

For clients that support certificate management packages with locally-generated keys, they MUST support certs-only [RFC5751][RFC5911], ct-PKIData [RFC5272][RFC5912], and ct-PKIResponse [RFC5272][RFC5912].

Retrieval of CRLs and additional certificates via the certs-only package, is OPTIONAL. Clients can retrieve CRLs and additional certificate via other mechanisms. Client support for the ct-contentCollection and the ct-contentWithAttributes content types is OPTIONAL.

For clients that support firmware packages [RFC4108][RFC5911], they MUST support the ct-firmwarePacakge content type. Client support for the ct-firmwareLoadReceipt and ct-firmwareLoadError content types is OPTIONAL, as per [RFC4108].

For clients that support the Trust Anchor Management Protocol (TAMP) [RFC5934], they MUST support the Certificate choice of the TA format and MUST support the tamp-update content type [RFC5934].

TO DO: Complete the following:

For clients that support certificate management packages with centrally-generated keys, they MUST support ct-asymmetric-key-package

[RFC5958], ct-PKIData [RFC5272][RFC5912], and ct-PKIResponse [RFC5272][RFC5912].

5. Agents

Agents act on behalf of the client. Agents MUST support PAL processing.

TO DO: Fill this out.

6. Electronic Serial Number

The Electronic Serial Number (ESN) is a permanent identifier that is used to identify the client throughout its lifecycle. Certificates include the ESN with the Hardware Module Name from [RFC4108] in the Subject Alternative Name extension [RFC5280]. The hardware module name form is an hwType (an object identifier) and hwSerialNumber (octet string). The combination of the object identifier and octet string guarantees global uniqueness. For example, a company uses their private enterprise number they received from IANA and includes their serial number the octet string. The KMS, clients, and agents SHOULD support ESNs at least 8 octets in length.

7. Product Availability List

The PAL provides clients with:

- Advertisements for available packages and transactions that can be retrieved from the KMS;
- Advertisement for another PAL.

TO DO: Add definition of Notification in Section 1.1. Need to explain it's an exception the PAL including packages.

An example PAL is provided in Figure 4. The explanation of the fields is explained in the subsequent text and sections.

```
<?xml version="1.0"encoding="us-ascii" ?>
<pal>
  <message>
```

```
      <type>TBD</type>
      <date>00000000000000</date>
      <size>1996</size>
      <info>https://www.example.com/pki/12</info>
    </message>
    <message>
      <type>100</type>
      <date>00000000000000</date>
      <size>0</size>
      <info>DN of subject</info>
    </message>
    <message>
      <type>TBD</type>
      <date>00000000000000</date>
      <size>2390</size>
```



```
      <info>https://www.example.com/distribution/100</info>
    </message>
    <message>
      <type>1</type>
      <date>00000000000000</date>
      <size>0</size>
      <info>https://www.example.com/distribution/12345</info>
    </message>
  </pal>
```

Figure 4 - Example PAL

TO DO: Include legal encoding for DN in Figure 4.

PAL processing by clients is OPTIONAL, yet RECOMMENDED. PAL retrieval can be performed by a client or by an agent that is assisting the device. Agents that service clients which do not process PALs, MUST process the PAL on behalf of the client. The agent MUST retrieve and process the PAL from the KMS as well as the packages advertised within the PAL. Once delivered to the agent, the agent MUST provide the package to the target client in an implementation specific manner. The method of delivery of the package to the target client may or may not implement a PAL type distribution mechanism.

When a client or agent requests a PAL, the KMS dynamically assembles

a PAL based on the current information and packages it has for the requesting client or agent. The KMS servicing the request relies on the knowledge of the requesting client's ESN, in order to amass the proper list of items.

The following identifies the items for each KMS service the KMS could include in a PAL for an identified Device:

- Publication: Anywhere from zero (0) to a maximum of i CA certificates, client certificate, and CRLs or other issuers offering public publications.

- Certificate: Anywhere from 0 to a maximum of j candidate entries (i.e., pending certificate management transactions or certificate notifications) where j <= the maximum number of certificates the device can have.

- Distribution: Anywhere from zero (0) to a maximum of q packages where q is less than or equal to the total number of independently-deliverable keys, and bundled packages the client is designed to accept.

An order of precedence for PAL offerings is based on the following rationale:

- Publication packages are the most important because they support validation decisions on certificates used to sign and encrypt other listed PAL items.

- Certificate Management packages items are next in importance, since they can impact an IA certificate used by the device to sign CMS content or a KE certificate to establish keys for encrypting content exchanged with the client.

  * A client engaged in a certificate management should accept and process CA-provided transactions as soon as possible to avoid undue delays that might lead to protocol failure.

- Distribution packages containing keys and other types of products are last. Precedence SHOULD be given to KMS packages that the client has not previously downloaded. The items listed in a PAL

may not identify all of the packages available for a device. This can be for any of the following reasons:

- The KMS may temporarily withhold some outstanding PAL items to simplify client processing.
  - Certificate Management PAL entries linked to a near-real-time CA device protocol (i.e., not staged through intermediary media devices or store and forward communication systems that may significantly delay interactions) will be limited to one-at-a-time.
  - If a CA has more than one certificate ready to begin a certificate management protocol with a client, the KMS will provide a notice for one at a time. Pending notices will be serviced in order of the earliest date when the certificate will be used.
  - The KMS will complete a certificate management activity for one certificate, before beginning the process for another. At most one pending certificate management transaction will be advertised in the PAL at a time.
- A PAL is limited to a maximum of thirty-two entries. If more than thirty-two entries are available for the client, additional PALs will be identified in the last entry of the PAL. The first PAL in the chain is identified as the Initial PAL.
- Packages will be removed when their contents are superseded or at

the direction of a KMS Manager.

The remainder of this section describes the PAL format and its use of URIs.

7.1. PAL Format

The PAL furnishes information for KMS messages that are currently available and authorized for retrieval by a client or an agent. The PAL is used to identify the following information:

- The KMS Package type and unique package identifier of each

package available.

- The size of each package.

- The last time and date the device downloaded the data, if any.

- The presence of KMS notifications and the ancillary data the client may need to respond to that notification.

- The availability of another PAL listing packages that were not included on the current PAL.

- For those package delivered out of the KMS Distribution and KMS Certificate Management Services, the KMS Service message type.

The initially offered PAL, will contain anywhere from zero to thirty-two XML-encoded PAL entries following the XML Header. The PAL's XML schema can be found in Section 12. Each PAL entry is composed of the following four REQUIRED subcomponents:

- The Type subcomponent is provided for each PAL entry. The Type uniquely identifies each KMS package defined within this specification that a client may retrieve from KMS with a 4-digit field. The Types are defined in Section 9 and registered in Section 11.

- The Last Download Date subcomponent is provided for each PAL entry. It is a 14-character field that contains either:

- The date and time (expressed as Generalized Time) that the client last successfully downloaded the identified package from the KMS, or

- All zeroes characters, if:

  * There is no indication the device has successfully loaded the

identified KMS package,

- The PAL entry is a notification, or

- The PAL entry corresponds to a notification or pointer to a

next PAL.

- The Package Size subcomponent is provided for each PAL entry. If the PAL entry is for a notification, this subcomponent will be populated with a zero character. Otherwise, it indicates the size of the identified package in bytes. The maximum size of packages is 2.1 Gbytes.

- The Additional Information subcomponent will be provided for each PAL entry and will either provide a Distinguished Name (DN) or a URI of where the identified KMS package can be retrieved. When the entry is a notification, the subcomponent is a DN that identifies a certificate that is the subject of the notification.

When more than thirty-two PAL entries are available, an additional PAL is advertised in the thirty second PAL entry. The additional PAL will have between one and thirty-two PAL entries.

The Last Download Date MUST be represented in a form that matches the dateTime production in "canonical representation" [XMLSCHEMA]. Implementations SHOULD NOT rely on time resolution finer than milliseconds and MUST NOT generate time instants that specify leap seconds.

7.2. URIs

A client that supports the PAL will use URIs to obtain both the KMS packages they need from the KMS, and to post device information KMS requires. Clients that support PALs and agents MUST be capable of using URIs [RFC3986].

In order to GET or POST, the client or agent needs to have a currently valid URI associated with that information. The URI can correspond to:

- A PAL that provides a unique URI for each KMS package that the KMS holds for the client and URIs identifying client actions that need to be taken, or

- A KMS package that the client believes is being held by the KMS. The data may contain product, a protocol-related transaction, or a collection of packages with various contents.

When a client performs an HTTP POST operation, the URI indicates the specific KMS Service that is targeted to process the information. A client SHALL be capable of requesting information by providing a URI in an HTTP GET request to a connected KMS.

A client may know, or believe they know, a specific KMS package URI, because:

- They discovered the URI on a PAL,
- They are anticipating the next step in a protocol initiated by a prior URI submission, or
- They were provided with the URI out-of-band by a human or an agent. Clients and agents MUST be capable of accepting a URI that uniquely identifies the location of a KMS Service package that is available for delivery.

Clients and agents MUST be capable of accepting a URI that identifies an action that is to be taken by the client.

In order to POST information, the client or agent supplies a URI that identifies associated information to the KMS. For example, the URI could correspond to a request to initiate, furnish intermediate results for, or conclude a certificate management protocol.

Regardless of whether an HTTP GET or HTTP POST request is being made, URI components have consistent definitions and usage requirements. These are specified in the following subsections. Figure 5 provides a view of the URI components:

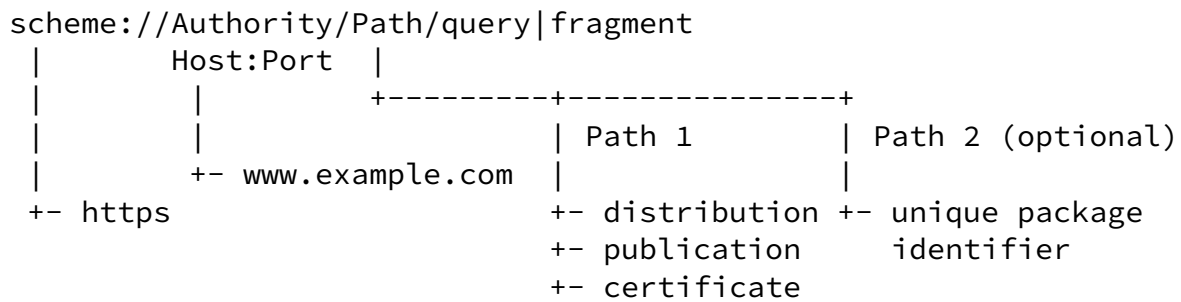


Figure 5 - PAL URI Components

7.2.1. URI Scheme

All HTTP GET and POST requests and responses MUST use "https" as the scheme [RFC2818]. All processing of scheme data will be case-insensitive as required in [RFC3986].

PALs that do not specify "https" as the URI scheme for every PAL

entry MUST be rejected.

7.2.2. URI Authority

The authority component of a URI identifies the KMS that the client is requesting the specific KMS Service from. The authority component is in the form of a host name and an optional "https" port number. The host name identifies the HTTP server by name, and the port number identifies the HTTP server port that will service the request. Inclusion of the port number is OPTIONAL, as port 443 MUST be used.

Clients and agents that access KMS Services are configured with the applicable registered name(s) or corresponding IP address(es) of the KMS with which they may establish a connection to.

When generating a URI, the KMS SHALL populate the Authority Component of the URI with the registered name of the target KMS.

When generating a URI, clients and agents SHALL populate the Authority Component of the URI with the registered name of the target KMS.

Clients and agents SHALL reject the delivery of a received PAL, if any URI Authority Component contains a registered name that does not correspond to the connected KMS.

7.2.3. URI Path

The Path component of a URI identifies a resource that can be retrieved from, or a location that information can be posted to, at the KMS. Path components are presented in the hierarchical form of KMS Service Identifier followed by a Product Identifier. They adhere to the rules for path-absolute parsing as defined in [RFC3986].

Service Identifiers that constitute the first path (aka Path 1) segment in a URI received or generated by a device are listed below together will a brief description of their purpose:

- distribution - This identifier is used for PALs, product packages, and bundled packages with one or more collections of content types as offered by the KMS Distribution Service.

- publication - This identifier is used to obtain publicly-available CA, CRLs, and CPs as offered by the KMS Publication Service.

- certificate - This identifier is used in PKI issuance and rekey protocols as offered by the KMS Certificate Management.

The Product Identifier (aka Path 2), when present, is always the second path segment. It is formatted as an integer and represents the unique identifier the KMS has associated with the package to be retrieved. Message types are included in the Message Type registry found in Section 11. The Product Identifier is only present in the URIs that will be included in HTTP GET requests to obtain a package. The Product Identifier is not included in:

- The URI a client uses to obtain the initial PAL,

- The URI portion of a KMS Distribution Service PAL entry a KMS uses to point to other PALs beyond the initial PAL,

- The KMS Certificate Service URIs that a KMS uses to provide the device notification for a suggested action, and

- URIs that a device provides as a part of an HTTP POST request.

A client SHALL reject the delivery of any PAL received that contains a URI with the first path component not equal to one of the following service names:

- distribution,
- pki, and
- certificate.

When generating a URI for the inclusion in a POST operation, a client SHALL only populate the first Path component of the URI. When generating a URI for the inclusion in a GET operation for the initial PAL, a client SHALL only populate the first Path component of the URI. When generating a URI, clients SHALL populate the first Path component of the URI with one of the service names defined by this specification. A client SHALL reject the delivery of any PAL received that contains a URI with the second path component not equal to an integer.

7.2.4. URI Query and Fragments

The KMS does not use Query and Fragment elements in support of KMS Services. They are not supported by clients in the processing of received URIs, or in the generation of URIs.

The KMS MUST omit query and fragment components from PALs.

The KMS SHOULD reject the delivery of any PAL that contains a URI with a query or fragment components.

Clients and agents SHOULD reject the delivery of any PAL that

contains a URI with a query or fragment component.

When generating a URI, clients and agents MUST NOT populate the URI with any query or fragment components.

8. SODP Transport Requirements

This section provides the requirements for SODP interactions.

8.1. KMS Requirements

The KMS MUST support HTTP 1.1 [RFC2616]; the KMS MUST support generating HTTP GET and POST responses and receiving HTTP GET and POST requests; the KMS MUST support HTTPS [RFC2818] over TCP [RFC793] on port 443, and; the KMS MUST support both IPv4 [RFC791] and IPv6 [RFC2460]. TLS 1.2 [RFC5246][I-D.tls-ssl2-must-not] MUST be implemented in conjunction with HTTPS. To ensure only authorized clients and agents access the KMS, the KMS MUST support authentication with both client-side certificates and username/password. See Section 10 for cipher suite requirements.

When the KMS receives and processes an HTTP request from a client, it will provide a response. HTTP responses include status information and may include a message body, when a request is successfully processed. The status information provided in responses to client requests will be restricted to the three-digit HTTP status code.

HTTP response status codes fall into five general classes (where the

class is indicated by the first digit of the code).

- Informational - The KMS will not make use of the Informational class of status codes. Protocol switches and continued client processing are not expected.

- Success - The KMS will return this class when the GET results in the requested information being returned or the POST action is successfully completed.

- Redirection - The KMS will not make use of the Redirection class of status codes. The KMS will not ask a client to take further action to fulfill a request.

- Client Error - The KMS will return this class when they cannot fulfill the requested GET or POST because of a client error.

- Server Error - The KMS may return this class, when a valid POST or GET request was received, but the KMS cannot fulfill the request for other reasons.

8.2. Client Requirements

Clients MUST support HTTP 1.1 [RFC2616]; clients MUST support HTTP generating GET and POST requests and HTTP GET and POST responses; clients MUST support HTTPS [RFC2818] over TCP [RFC793] on port 443, and; clients MUST support either IPv4 [RFC791] or IPv6 [RFC2460] (IPv6 is preferred). TLS 1.2 [RFC5246][I-D.tls-ssl2-must-not] MUST be implemented in conjunction with HTTPS. Clients MUST support client-side certificate authentication when connecting to the KMS. See Section 10 for cipher suite requirements.

If a client receives an HTTP response with an Informational or Redirection class status code, it SHALL interpret the response as a request failure and terminate its session with the KMS.

When an Informational or Redirection class status code is received, a client MAY, if configured for an alternate KMS, terminate the current session and attempt to connect with an alternate KMS to obtain the originally requested KMS Service.

If a client receives an HTTP response with a Success class status

code, it SHALL continue to process the response to determine the outcome of an HTTP POST request or to use the information contained in the included package.

If a client receives an HTTP response with a Client Error class status code, it SHALL abandon the desired action and not repeat the same request to the same KMS during the connection session.

The client can provide additional processing of Client Error class status codes for a given request; however, this is out-of-scope of this document.

A client can attempt other (different) HTTP requests after a request that failed with a Client Error class status code. However, the client incorporate a means to limit the number of consecutive requests that fail for any reason in a given connection session with the KMS.

If a client receives an HTTP response with a Server Error class status code, it SHOULD either:

- Reattempt the request after a non-deterministic delay, or
- Attempt the request with a different KMS.

8.3. Agent Requirements

Agent requirements are identical to those for clients with one exception and that is that agents MUST support either agent-side certificate authentication when connecting to the KMS or username/password.

9. Message Sequences

This section depicts message sequences when using a PAL.

9.1. Distribution

The KMS Distribution service instantiates itself with the distribution of symmetric key packages and firmware packages. The message types are defined as follows:

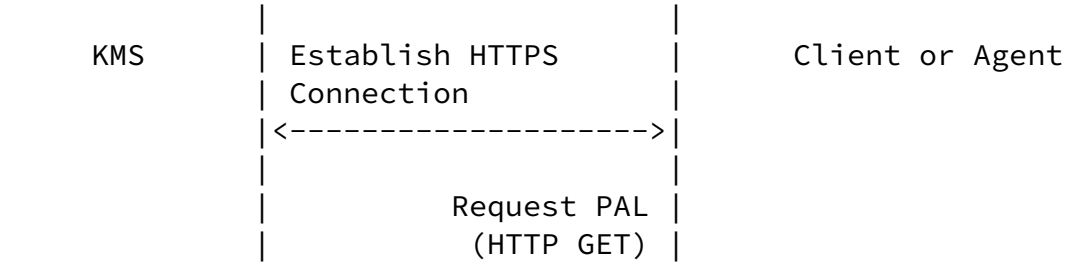
Message Type	Package
TBD	Symmetric Key Package
TBD	Firmware Package

An example PAL entry for a distribution package is as follows:

```
<message>
  <type>TBD</type>
  <date>00000000000000</date>
  <size>1996</size>
  <info>https://www.example.com/distribution/symmtrickey1</info>
</message>
```

The message type TBD indicates the message is a symmetric key. The date and time indicates that the package has not been downloaded. The message size indicates the size of the package and the additional info element provides a link to the symmetric key.

The sequence for both symmetric key and firmware packages is identical, as shown in Figure 6. The client or agent connects to the KMS, retrieves their PAL, and the requests the package from the URI provided in the additional info component.



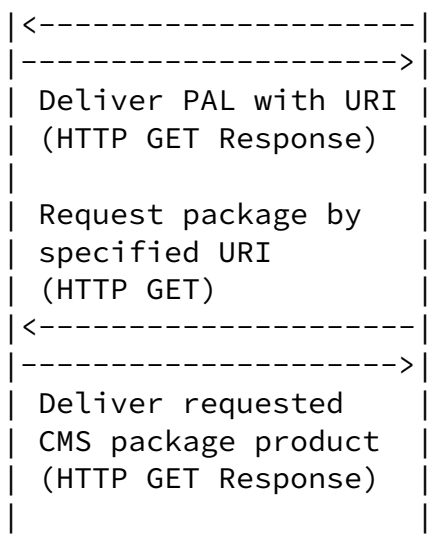


Figure 6 - SODP Distribution Service Message Sequence

9.2. Publication

The KMS Publication service instantiates itself with the distribution of additional certificates, CRLs, CPs, and CPSs. The message types are defined as follows:

Message Type	Package
TBD	Root CRL
TBD	non-Root CRL

TO DO: Add additional certificates, CPs, and CPSs.

An example PAL entry for a publication package is as follows:

```
<message>
  <type>TBD</type>
  <date>00000000000000</date>
  <size>1996</size>
  <info>https://www.example.com/publication/Root.crl</info>
</message>
```

The message type TBD indicates the message is a Root CRL. The date

and time indicates that the package has not been downloaded. The message size indicates the size of the package and the additional info element provides a link to the Root CRL. The message sequence is identical to Figure 6.

9.3. Certificate Management

The KMS Certificate Management service instantiates itself with the distribution of notifications (i.e., start rekey), and CMC transactions. The message types are defined as follows:

Message Type	Package
100	IA Certificate Rekey Notification
N/A	IA Certificate Rekey Transaction One
TBD	IA Certificate Rekey Transaction Two (Success)
TBD	IA Certificate Rekey Transaction Two (Failure)
TBD	KE Certificate Issuance Notification
N/A	KE Certificate Issuance Transaction One
TBD	KE Certificate Issuance Transaction Two (Success)
TBD	KE Certificate Issuance Transaction Two (Failure)

An example PAL entry for a publication package notification is as follows:

```
<message>
  <type>100</type>
  <date>00000000000000</date>
  <size>1996</size>
  <info>DN of IA certificate</info>
</message>
```

TO DO: Get legal encoding of DN for IA certificate.

The message type TBD indicates the message is a IA Certificate Rekey Notification. The date and time indicates that the package has not been downloaded. The message size indicates the size of the package and the additional info element provides a link to the rekey notification.

The message sequence for certificate rekey and issuance is a three-step process. The initial step is client/agent retrieval of the PAL and then retrieval of a notification for either IA rekey or KE issuance. Step two is the client/agent posting of the CMC package. Step three is certificate request response (success or failure) from the KMS. Prior to each interaction with the KMS, the client/agent authenticates itself with the KMS. The three steps are depicted in

Figures 7-9.

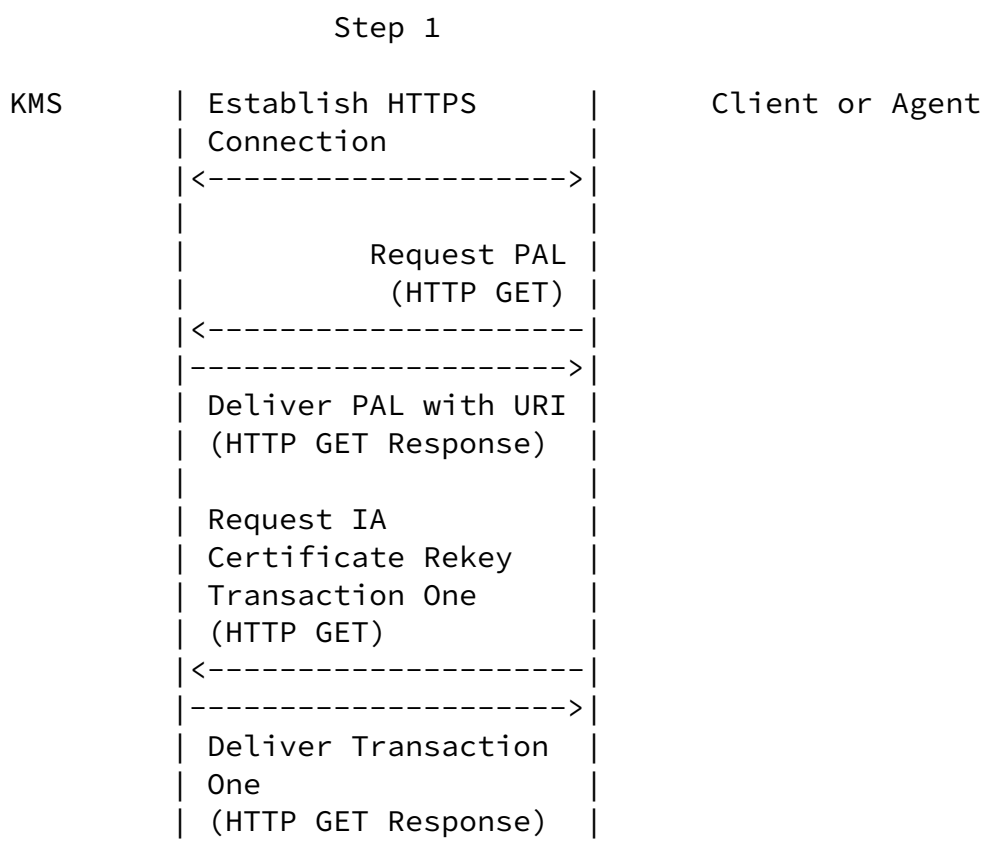


Figure 7 - SODP Certificate Management Service Message Sequence - Step 1

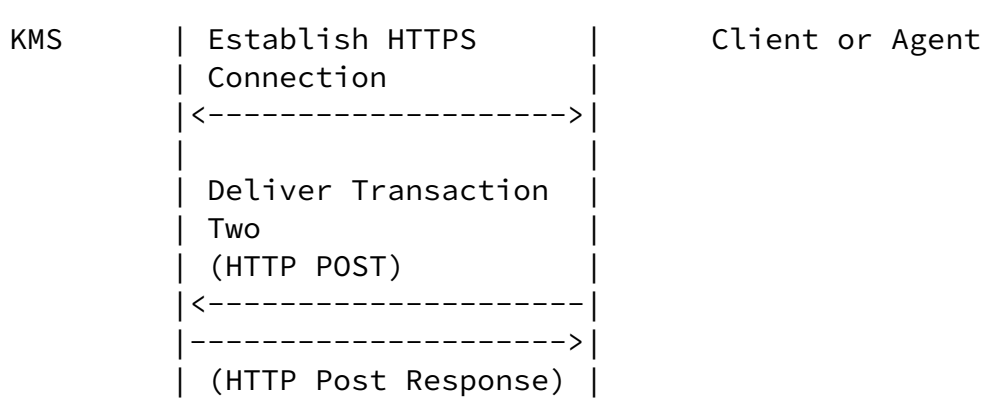


Figure 8 - SODP Certificate Management Service Message Sequence Step 2

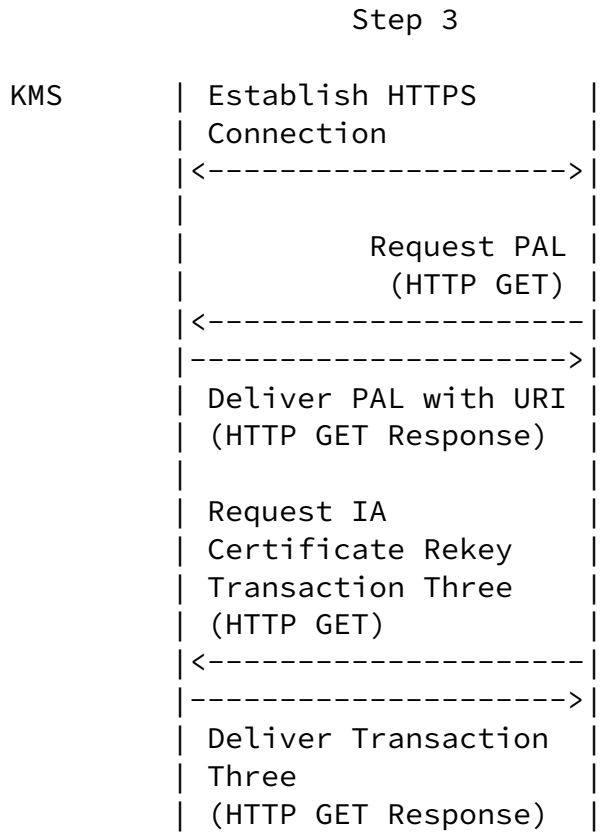


Figure 9 - SODP Certificate Management Service Message Sequence Step 3

10. Cryptographic Algorithm Requirements

This section defines the cryptographic algorithm requirements for SODP. There are three types: package protection requirements, TLS cipher suites, and certificate requirements.

10.1. Package Protection

For [RFC5958] algorithm requirements see [RFC5959].

For [RFC6031] algorithm requirements see [I-D.turner-cms-symmetrickeypackage-algs].

For [RFC6032] algorithm requirements see [RFC6033].

NOTE: The "cert-only" package does not have algorithm requirements because no cryptographic operations are performed while generating this package.

TO DO: Include text or reference(s) for the following:

For [RFC4108][RFC5911] algorithm requirements see [TO DO].

For [RFC5934] algorithm requirements see [TO DO].

For [RFC5280] algorithm requirements see [TO DO].

10.2. TLS Cipher Suites

The following requirements apply to the KMS, client, and agent:

- Cipher suites supported MUST include: "TLS_RSA_WITH_", "TLS_DH_", "TLS_DHE_", and "TLS_ECDH_".
- Cipher suites that include "anon" MUST NOT be used. These suites do not support mutual authentication.
- Cipher suite that include "EXPORT" and "DES" MUST NOT be used. These ciphers do not offer a sufficient level of protection; 40-bit crypto in '11 doesn't cut the mustard and the use of DES is deprecated.
- When confidentially is supported (recall that is optional), the "AES_128" ciphers MUST be supported and "AES_256" cipher SHOULD be supported.
- Cipher suites that include "SHA256" MUST be supported and "SHA384" SHOULD be supported.

10.3. Certificates

Client, agents, and the KMS MUST support certificate path validation on key packages and TLS connections [RFC5280].

TO DO: Need to add text that lines up algorithm requirements for packages with certificates. Also add CCC [RFC6010] as an OPTIONAL extension for source authorities.

11. Security Considerations

TO DO: Expand this section!

This document relies on many other specifications. For IP and TCP security considerations see [RFC791], [RFC793], and [RFC2460]; for HTTP, HTTPS, and TLS security considerations see [RFC2616], [RFC2818], and [RFC5246]; for URI security considerations see [RFC3986]; for content type security considerations see [RFC4073], [RFC4108], [RFC5272], [RFC5652], [RFC5751], [RFC5958], [RFC5934], [RFC6031], and [RFC6032]; for certificate security considerations see

[RFC5280], [RFC5480], and [RFC6010], and; for algorithm security considerations see [RFC5959], [RFC6033], [I-D.turner-cms-symmetrickeypackage-algs].

TO DO: Probably more references are needed above for algorithms based on what gets added in Section 9.1.

It is critical that the KMS encrypt symmetric keys and centrally-generated asymmetric private keys for the end client. Failure encrypt these keys will allow any intermediaries to intercept the key and eavesdrop and/or impersonate the client.

When packages are encrypted, the source of the package must randomly generate package-encryption keys. Also, the generation of public/private signature key pairs relies on a random numbers. The use of inadequate pseudo-random number generators (PRNGs) to generate cryptographic keys can result in little or no security. An attacker may find it much easier to reproduce the PRNG environment that produced the keys, searching the resulting small set of possibilities, rather than brute-force searching the whole key space. The generation of quality random numbers is difficult. [RFC4086] offers important guidance in this area.

12. IANA Considerations

IANA is requested to perform four registrations: SODP Name Space, SODP XML Schema, SODP Message Types, and SODP URI String Types.

12.1. SODP Name Space

This section registers a new XML namespace, "urn:ietf:params:xml:ns:TBD" per the guidelines in [RFC3688]:

TO DO: Fill in TBDs.

```
URI:  urn:ietf:params:xml:ns:TBD
Registrant Contact: Sean Turner (turners@ieca.com)
XML:
   BEGIN
      <?xml version="1.0"?>
      <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
         "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
      <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
      <head>
         <title>SODP Messages</title>
      </head>
      <body>
         <h1>Namespace for SODP Messages</h1>
```



```
         <h2>urn:ietf:params:xml:ns:TBD</h2>
         <p>See TBD</p>
      </body>
      </html>
   END
```

12.2. SODP Schema

This section registers an XML schema as per the guidelines in [RFC3688].

TO DO: Fill in TBDs.

```
URI: urn:ietf:params:xml:ns:TBD
Registrant Contact:  Sean Turner turners@ieca.com
```

```
   XML:
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd=http://www.w3.org/2001/XMLSchema
 xmlns:sodp=TBD
 targetNamespace=TBD
 elementFormDefault="qualified" attributeFormDefault="unqualified"
 version="0.1">

<!-- ===== Element Declarations ===== -->

<xsd:element name="pal" type="sodp:PalType" />

<!-- ===== Complex Data Element Type Definitions ===== -->

<xsd:complexType name="PalType">
  <xsd:sequence>
    <xsd:element name="message" type="sodp:SODPMessageType"
                 minOccurs="0"  maxOccurs="32">
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="SODPMessageType">
  <xsd:sequence>
    <xsd:element name="type" type="sodp:MessageType" />
    <xsd:element name="date" type="sodp:GeneralizedTimeType" />
    <xsd:element name="size" type="sodp:PackageSizeType" />
    <xsd:element name="info" type="sodp:MessageInfoType" />
  </xsd:sequence>
</xsd:complexType>

<!-- =====Simple Data Element Type Definitions ===== -->
```



```
<xsd:simpleType name="MessageType">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="[0-9]+" />
    <xsd:maxLength value="4" />
  </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="GeneralizedTimeType">
```

```
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[0-9]{14}" />
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="PackageSizeType">
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[0-9]+" />
      <xsd:maxLength value="19" />
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="MessageInfoType">
    <xsd:restriction base="xsd:string" />
  </xsd:simpleType>

  </xsd:schema>
```

12.3. SODP Message Types

This section registers the SODP Message Types. SODP Message Types registrations are to be subject to Specification Required, as per RFC 5226 [RFC5226]. The registry has the following values:

Value	Message Type	Specification
0	Reserved	This document
1	Additional PAL value present	This document
100	IA Rekey Notification	This document
TBD	Symmetric Key Package	This document
TBD	Firmware Package	This document
TBD	Root CRL	This document
TBD	non-Root CRL	This document
TBD	IA Certificate Rekey Transaction Two - Success	This document
TBD	IA Certificate Rekey Transaction Two - Fail	This document
TBD	KE Certificate Issuance Transaction One	This document
TBD	KE Certificate Issuance Transaction Three - Success	This document
TBD	KE Certificate Issuance Transaction Three - Fail	This document

TO DO: Add values from Section 9 to the above table.

12.4. SODP Path 1 String Values

This section registers SODP Path String Types as per [RFC3688]. SODP Path 1 String Value registrations are to be subject to Specification Required, as per RFC 5226 [RFC5226]. The registry has the following structure:

SODP Message Types	Specification
distribution	This document
publication	This document
certificate	This document

TO DO: Verify that specification required is appropriate.

13. IANA Considerations

None. Please remove this section prior to publication as an RFC.

14. References

14.1. Normative References

[RFC791] Postel, J. (ed.), "Internet Protocol - DARPA Internet Program Protocol Specification", RFC 791, September 1981.

[RFC793] Postel, J. (ed.), "Transmission Control Protocol," RFC 793, September 1981.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2460] Deering, S., and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460, December 1998.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4073] Housley, R., "Protecting Multiple Contents with the Cryptographic Message Syntax (CMS)", RFC 4073, May 2005.

[RFC4108] Housley, R., "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages", RFC 4108, August 2005.

[RFC5226] Naten, T., and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, May 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R. and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, March 2009.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, September 2009.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, June 2008.

[RFC5751] Ramsdell, B. and S. Turner, "Secure/Multipurpose Internet Mail Extensions(S/MIME) Version 3.2 Message Specification", RFC 5751, January 2010.

[RFC5911] Hoffman, P. and J. Schaad, "New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME", RFC 5911, June 2010.

[RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, June 2010.

[RFC5934] Housley, R., Ashmore, S., and C. Wallace, "Trust Anchor Management Protocol (TAMP)", RFC 5934, August 2010.

[RFC5958] Turner, S., "Asymmetric Key Packages", RFC 5958, August 2010.

[RFC5959] Turner, S., "Algorithms for Asymmetric Key Packages", RFC 5959, August 2010.

[RFC6010] Housley, R., Ashmore, S., and C. Wallace, "Cryptographic Message Syntax (CMS) Content Constraints Extension", RFC 6010, September 2010.

[RFC6031] Turner, S., and R. Housley, "Symmetric Key Package Content Type", RFC 6031, December 2010.

[RFC6032] Turner, S. and R. Housley, "Cryptographic Message Syntax (CMS) Encrypted Key Package Content Type", RFC 6032, December 2010.

[RFC6033] Turner, S., "Algorithms for Cryptographic Message Syntax (CMS) Encrypted Key Package Content Type", RFC 6033, December 2010.

[I-D.tls-ssl2-must-not]
Turner, S., and T. Polk, "Prohibiting SSL Version 2.0", draft-ietf-tls-ssl2-must-not, work-in-progress.

[I-D.turner-cms-symmetrickeypackage-algs]
Turner, S., "Algorithms for Cryptographic Message Syntax (CMS) Protection of Symmetric Key Package Content Types", draft-turner-cms-symmetrickeypackage-algs, work-in-progress.

[XMLSCHEMA]
Malhotra, A. and P. Biron, "XML Schema Part 2: Datatypes Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-2-20041082, October 2004,

<http://www.w3.org/TR/2004/REC-xmlschema-2-20041028>.

[TO DO] Insert references for Section 9.1.

14.2. Informative References

[RFC4086] Eastlake, D., 3rd, Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, November 2007.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

[XMLNS] Hollander, D., Bray, T., and A. Layman, "Namespaces in XML", World Wide Web Consortium FirstEdition REC-xml-names-19990114, January 1999, <http://www.w3.org/TR/1999/REC-xml-names-19990114>.

Appendix A. Example Encodings

TO DO: Include BASE64 encodings of ASN.1 encodings of selected packages. They're a lot smaller than the ASN.1 pretty prints and there are tons of available to tools to convert.

Authors' Addresses

Sean Turner
IECA, Inc.
3057 Nutley Street, Suite 106
Fairfax, VA 22031
USA

EMail: turners@ieca.com