The Direct Project

Implementation Guide for Direct Project Trust Bundle Distribution

Version 1.0

14 March 2013

Contents

Change Control

Date	Version	Description of changes
14-Mar-2013	1.0	Published.
7-Mar-2013	0.92	Updates based on comments from community members.
3-Mar-2013	0.9	Updates based on Consensus Votes, specifically related to the bundle packaging.
20-Feb-2013	0.8	Formatting and updates for consistency with other Direct Project specifications and IGs.
20-Feb-2013	0.7	Incorporation of changes from Feb 13 Trust Bundle SWG meeting.
10-Dec-2012	0.1	Initial draft.

Status of this Guide

This document is PUBLISHED.

Introduction

Overview

One the most important aspects of Directed exchange is the establishment of "trust" between sending and receiving parties. The definition of trust for Direct in a pure technical sense is the mutual exchange and inclusion of one STA's set of trust anchor(s) into another STA's trust store and vice versa. From a policy perspective, the definition of trust is broader and may imply different semantics based on organizations policies, processes and perspective. Regardless, a universal definition of trust includes the mutual agreement of two STAs to exchange Direct messages with one another based on a set of policies. These policies are asserted by the inclusion of each other's trust anchors into their respective trust stores.

Establishing trust between STAs has shown to be a potential barrier in universal adoption of Direct, particularly the ability to establish trust on a large scale. One solution has been to create "Trust Communities" where each community adopts a set of policies by which its member STA must attest to for compliance. Upon attesting to compliance, the member STA is considered to be in good standing with the community and can participate in Direct exchange with all other member of the community. To facilitate the exchange of trust anchors between members of the community, each member may submit its trust anchors to be included in a "Trust Bundle" that is managed by the community.

Trust Bundles are simply a collection of trust anchors that meet a common set of minimum policy requirements within a Trust Community. Relying parties may include the bundles into their STA implementations with confidence that each trust anchor adheres to the policies set by the Trust Community managing the bundle.

This document provides guidance on the packaging and distribution of Trust Bundles to facilitate scalable trust between STAs. The Trust Bundle distribution and packaging is a complementary capability to the existing Direct Certificate Discovery capability using DNS and LDAP. In other words, publication and discovery of Direct public certificates via DNS+LDAP as required by Direct Project's *Applicability Statement for Secure Health Transport* since version 1.1 must still be performed as normal. The Trust Bundle implementation guide (this document) deals with the exchange of Trust Anchors only and not Direct signing and encryption certificates.

Scope

This guide details standard packaging options for sets of Direct Project trust anchors and specifies requirements for parties publishing, requesting, and consuming those packages. Requirements and policies for the management of and inclusion of anchors into a trust bundle are outside the scope of this guide and are a matter for implementing trust communities. Similarly, requirements for importing bundles into an STA and STA configuration are left to STA implementers.

Definitions and Context

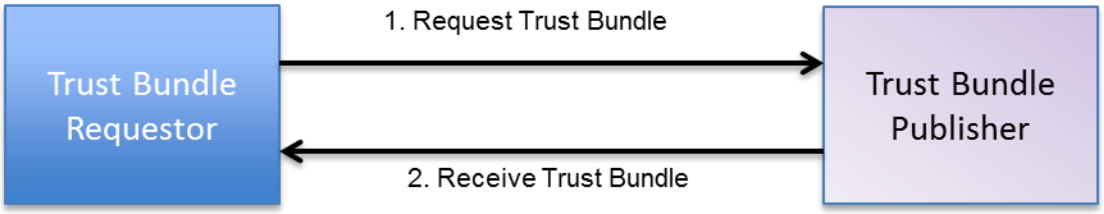
This section describes the top-level actors and definitions required to outline the specific requirements.

Trust Community: Trust Communities are formed by organizations electing to follow a common set of policies and processes related to health information exchange. Examples of these policies include identity proofing policies, certificate management policies, and HIPAA compliance processes.

Trust Community Profile: A Trust Community can create multiple sets of policies and processes and enforce these sets of policies on selected organizations that want to conform. For example, a Trust Community can create a set of policies and processes that organizations have to conform to for regular treatment related use cases, a different set of policies and processes that organizations have to conform to for Behavioral Health related use cases, and so on. These sets of policies and processes are called Trust Community Profiles. The word "Profile" indicates a distinct set of policies and processes.

Trust Bundle Distribution Context

The following figure and descriptions describe the context and actors involved in Trust Bundle distribution.



Trust Bundle: Trust Bundle is a collection of Direct Trust Anchors within a Trust Community that conform to a Trust Community Profile. Trust Anchors of member organizations that have elected to conform to a Trust Community Profile are included in the Trust Bundle for that particular Trust Community Profile. Some examples of Trust Bundles conforming to different Trust Community Profiles are:

- A Trust Bundle could have Trust Anchors that conform to NIST Level of Assurance 3
- A Trust Bundle could have Trust Anchors that are FBCA Cross-certified at Medium Level of Assurance.

Trust Bundle Requestor: A Trust Bundle Requestor is an entity (person, software system, Direct STA, etc) that requests a Trust Bundle from a Trust Bundle Publisher.

Trust Bundle Publisher: A Trust Bundle Publisher is an entity that publishes one or more Trust Bundles for a Trust Community.

Assumptions

The decision to include a Trust Bundle into an STA is non-systemic, meaning there is a clear conscientious decision by a policy administrator to include the bundle. The actual process and workflow of identifying and including the bundle is left to the STA implementation, but should consist of steps that require user intervention. Once a bundle distribution point has been identified and configured, it is desirable that including bundle updates into the STA be systemic.

Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

An implementation is not compliant if it fails to satisfy one or more of the MUST, SHALL, or REQUIRED level requirements for the protocols it implements. An implementation that satisfies all the MUST, SHALL, or REQUIRED level and all the SHOULD level requirements for its protocols is said to be "unconditionally compliant"; one that satisfies all the MUST, SHALL, or REQUIRED level requirements but not all the SHOULD level requirements for its protocols is said to be "conditionally compliant."

In addition, annotations called "*Implementation Note*:" are used to provide additional clarification to implementers. These are non-normative and provided for clarification and informational purposes only.

1.0 Trust Bundle Packaging

Trust Bundles are packaged using Cryptographic Message Syntax (CMS) formats specified in RFC 5652. CMS was identified as the preferred standard by the community because Direct STA's already process, validate and consume S/MIME messages that are based on the CMS standard.

Trust Bundle packages MUST conform to one of the following formats:

- Unsigned Trust Bundles described further in Section 1.1

- Signed Trust Bundles described further in Section 1.2

1.1 Unsigned Trust Bundles

Unsigned Trust Bundles are CMS SignedData objects described in section 5 of RFC 5652. Unsigned Trust Bundles are not signed, meaning that the authenticity of the container and its contents cannot be directly verified.

- Trust Bundle Publishers using Unsigned Trust Bundles MUST publish the bundle with file name extension of ".p7b" or ".p7c"
    - *Implementation Note*: p7b and p7c files are structurally the same files. The p7b file is based on the PKCS7 (RFC 2315) specifications and the p7c file is based on the RFC 5751 specifications.
- Trust Bundle Publishers MUST create the Unsigned Trust Bundle following RFC 5751 section 3.7, steps 1 and 2 and MUST NOT execute step 3.
    - *Implementation Note*: Although RFC 5751 requires ".p7c" file to conform to all three steps outlined in section 3.7, the ContentInfo object generated after step 2 of RFC 5751 Section 3.7 suffices for an Unsigned Trust Bundle ".p7c" file. Step 3 adds additional MIME wrapping which is not necessary for the Unsigned Trust Bundle.
- Trust Bundle Publishers MAY include optional metadata in the encapContentInfo eContent data field of the CMS SignedData object as UTF-8 encoded XML.
    - *Implementation Note*: This overrides RFC 5751 Section 3.7 and RFC 5652 Section 5.2, which require unsigned CMS SignedData objects to omit the encapContentInfo eContent field.

- Unsigned Trust Bundles MUST contain the following structure
    - MUST contain one or more Trust Anchors in the CMS SignedData certificates field
    - MUST contain zero CMS SignedData SignerInfos
    - MAY contain metadata in the CMS SignedData encapContentInfo eContent field as UTF-8 encoded XML. If metadata is not included, then the eContent field MUST be absent. In both cases, eContentType MUST be id-data.

1.2 Signed Trust Bundles

A Signed Trust Bundle is similar to an Unsigned Trust Bundle except that it is signed by a trusted entity. The advantage of a Signed Trust Bundle is that the authenticity and integrity of the bundle can be validated by a relying party.

- Trust Bundle Publishers using Signed Trust Bundles MUST publish the bundle with file name extension of ".p7m" only.
    - *Implementation Note*: Some implementers might want to use p7s as a file extension which is not valid because p7s file extensions are reserved for

multipart/signed messages with detached signatures and cannot be used for Signed Trust Bundles.

- Trust Bundle Publishers MUST create the Signed Trust Bundle following RFC 5652 section 5 requirements for signed-data content type.
- Signed Trust Bundles MUST contain the following structure:
    - MUST contain one or more certificates in the CMS SignedData certificates field.
        - *Implementation Note*: Unlike Unsigned Trust Bundles, the Signed Trust Bundles certificates field will contain one or more certificates intended to assist a Relying party in building a certification path from a trusted "root" or "top-level certification authority" to one or more of the Signers in the Signer Infos field. At a minimum, this MUST include each signer's certificate.
    - MUST contain one or more Signer Info in the CMS SignedData SignerInfos field.
    - MUST contain the Unsigned Trust Bundle in the CMS SignedData encapContentInfo eContent field wrapped in a CMS Data object, with eContentType of id-data.

1.2.1 Digest Algorithms

Trust Bundle Publishers MUST support the following digest algorithms to create Signed Trust Bundles.

- SHA1
- SHA256

Trust Bundle Publishers MUST NOT support less secure digest algorithms such as MD5.

Trust Bundle Publishers MAY support more secure digest algorithms listed as SHOULD+ in RFC 5752 section 2.1, but publishers should be aware that bundle consumers may not support more secure algorithms.

1.3 Metadata for a Trust Bundle

This section describes the metadata that can be included within a Trust Bundle by a publisher. The use of metadata by a Trust Bundle Requestor is optional at this point of time and Trust Bundle Publishers should consider the metadata purely informational.

- Trust Bundle Publishers MUST format the metadata as an XML file validated by the TrustBundleMetadata.xsd schema.
- Trust Bundle Publishers choosing to include metadata MUST include the following data elements within the metadata file
    - TrustCommunityProfile description element which describes the profile name and purpose of the Trust Bundle.

    - DistributionPoint element which identifies the URL from where the Trust Bundle is distributed.
- Trust Bundle Publishers choosing to include metadata MAY include the following data elements within the metadata file
    - ValidFrom time element to indicate the timestamp from which the Trust Bundle is valid.
    - ValidTo time element to indicate the timestamp until which the Trust Bundle is valid.
    - An Anchor element for each Trust Anchor that is part of the Trust Bundle.
        - Each Anchor element MUST contain the X509IssuerName, and X509SerialNumber elements.

An example of a metadata file is present in Appendix A: Trust Bundle Examples.

1.3.1 TrustBundle metadata schema

The TrustBundle metadata schema file is described below for usage by the Trust Bundle Publishers to create metadata in compliance with the schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
 <xs:element name="TrustBundle">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="Profile" minOccurs="1" maxOccurs="1"/>
    <xs:element ref="DistributionPoint" minOccurs="1" maxOccurs="unbounded"/>
    <xs:element ref="ValidFrom" minOccurs="0" maxOccurs="1"/>
    <xs:element ref="ValidTo" minOccurs="0" maxOccurs="1"/>
    <xs:element ref="Anchors" minOccurs="0" maxOccurs="1"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="Profile" type="xs:string"/>
 <xs:element name="Anchors">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="Anchor" minOccurs="1" maxOccurs="unbounded"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="Anchor">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="X509IssuerSerial" minOccurs="1" maxOccurs="1"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="X509IssuerSerial">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="X509IssuerName" minOccurs="1" maxOccurs="1"/>
```

```
      <xs:element ref="X509SerialNumber" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="X509IssuerName" type="xs:string"/>
  <xs:element name="X509SerialNumber" type="xs:integer"/>
  <xs:element name="ValidFrom" type="xs:dateTime"/>
  <xs:element name="ValidTo" type="xs:dateTime"/>
  <xs:element name="DistributionPoint" type="xs:anyURI"/>
</xs:schema>
```

2.0 Trust Bundle Distribution

- Trust Bundle Publishers MUST publish the packaged Trust Bundle using a unique, publicly available URL (Trust Bundle URL) per RFC 1738.
    - Trust Bundle Publishers MUST support the HTTP GET request on the Trust Bundle URL per RFC 2616
    - Trust Bundle publishers SHALL NOT require authentication of Trust Bundle Requestors to obtain access to the Trust Bundle.
- Trust Bundle Publishers using Unsigned Trust Bundles MUST support transport integrity of the bundle during distribution using, at a minimum, TLS 1.0 or SSL v3.0 protocols; Trust Bundle Publishers MAY support higher versions of TLS and SSL.
- Trust Bundle Publishers using a Signed Trust Bundles MAY distribute the bundle using a secure (https) or non-secure transport (http) protocol.
- Trust Bundle Publishers MAY publish more than one Trust Bundle for a Trust Community. In such cases each of the Trust Bundles MUST have its own unique Trust Bundle URL.

3.0 Trust Bundle Requestors

- Trust Bundle Requestors MUST support the retrieval and consumption of Trust Bundles published via the following mechanisms
    - Unsigned Trust Bundles described in section 1.1
    - Signed Trust Bundles described in Section 1.2
- Trust Bundle Requestors MUST verify the signature of a Signed Trust Bundle. Signature verification includes:
    - Verifying that the signing certificate is not expired, revoked, or invalid.
    - Validating the message digest.
    - Verifying that the signing certificate chains back to a trusted “root” or “top level certification” authority.
- Use of any metadata present within a Trust Bundle is optional for Trust Bundle Requestors; however it is RECOMMENDED as a best practice to validate any metadata present within a Trust Bundle for conformance to TrustBundleMetadata.xsd schema prior to its usage with the system.

4.0 Security Best Practices and Recommendations

The section outlines some of the security best practices and recommendations for Trust Bundle Publishers.

- Trust Bundle Publishers should develop operational policies and processes to obtain Trust Anchors and package them into a Trust Bundle to ensure the integrity of the Trust Anchor and the Trust Bundle.
- The Trust Bundle packaging and distribution mechanisms detailed in this guide are intended to be used for Trust Anchors only. Direct certificates for signing and encryption should not be included in the Trust Anchor bundle unless they are also serving as Trust Anchors.

5.0 References

1. RFC 1738 - Uniform Resource Locators
2. RFC 2119 - Keywords to use in RFC's for Requirement Levels
3. RFC 2246 - TLS Protocol
4. RFC 2315 - PKCS#7 Specification (Cryptographic Message Syntax Version 1.5)
5. RFC 2616 - Hyper Text Transfer Protocol
6. RFC 5652 - Cryptographic Message Syntax
7. RFC 5751 - S/MIME v 3.2
8. RFC 5752 - Multiple Signatures in Cryptographic Message Syntax

Appendix A: Trust Bundle Examples

This section provides implementers some non-normative examples of Trust Bundle structures.

A.1 Trust Bundle without Metadata

The following is a partial view of a Trust Bundle file without metadata. The encapContentInfo eContent field is absent. The output was generated with: openssl cms -print -cmsout -in [filename].p7b -inform der

```
CMS_ContentInfo:
  contentType: pkcs7-signedData (1.2.840.113549.1.7.2)
  d.signedData:
    version: 1
    digestAlgorithms:
      <EMPTY>
    encapContentInfo:
      eContentType: pkcs7-data (1.2.840.113549.1.7.1)
      eContent: <ABSENT>
    certificates:
      d.certificate:
        cert_info:
          version: 2
          serialNumber: -132802250398816262802584483327709843078
          signature:
            algorithm: sha1WithRSA (1.3.14.3.2.29)
            parameter: NULL
          issuer: emailAddress=example.com, CN=example.com
          validity:
            notBefore: Apr 19 19:22:19 2011 GMT
            notAfter: Dec 31 23:59:59 2039 GMT
          subject: emailAddress=example.com, CN=example.com
          key:
            algor:
              algorithm: rsaEncryption (1.2.840.113549.1.1.1)
              parameter: NULL
            public_key:  (0 unused bits)
              0000 - 30 81 89 02 81 81 00 ca-bc d8 93 da a6 0a   0.............
              000e - 86 0c fe 10 01 41 6a b4-8e 1f f7 8f 33 cb   .....Aj.....3.
              001c - 28 79 cf 01 0c 15 72 01-d9 19 79 43 56 94   (y....r...yCV.
              002a - 0f ec 2e ae 07 48 a3 f2-6e b5 36 75 76 ea   .....H..n.6uv.
              0038 - df 0d 84 c3 f7 74 3c 37-5e e6 e9 94 45 a7   .....t<7^...E.
```

A.2 Trust Bundle with Metadata

The following is a partial view of a Trust Bundle file with metadata. The encapContentInfo eContent field contains UTF-8 encoded XML. The output was generated with: openssl cms -print -cmsout -in [filename].p7b -inform der

```
CMS_ContentInfo:
  contentType: pkcs7-signedData (1.2.840.113549.1.7.2)
  d.signedData:
    version: 1
    digestAlgorithms:
      <EMPTY>
    encapContentInfo:
      eContentType: pkcs7-data (1.2.840.113549.1.7.1)
      eContent:
        0000 - 3c 3f 78 6d 6c 20 76 65-72 73 69 6f 6e 3d 22   <?xml version="
        000f - 31 2e 30 22 3f 3e 0d 0a-3c 74 65 73 74 53 74   1.0"?>..<testSt
        001e - 72 75 63 74 75 72 65 3e-53 61 6d 70 6c 65 20   ructure>Sample
        002d - 4d 65 74 61 64 61 74 61-3c 2f 74 65 73 74 53   Metadata</testS
        003c - 74 72 75 63 74 75 72 65-3e                     tructure>
    certificates:
      d.certificate:
        cert_info:
          version: 2
          serialNumber: -132802250398816262802584483327709843078
          signature:
            algorithm: sha1WithRSA (1.3.14.3.2.29)
            parameter: NULL
          issuer: emailAddress=example.com, CN=example.com
          validity:
            notBefore: Apr 19 19:22:19 2011 GMT
            notAfter: Dec 31 23:59:59 2039 GMT
          subject: emailAddress=example.com, CN=example.com
          key:
            algor:
              algorithm: rsaEncryption (1.2.840.113549.1.1.1)
              parameter: NULL
            public_key:  (0 unused bits)
              0000 - 30 81 89 02 81 81 00 ca-bc d8 93 da a6 0a   0.............
              000e - 86 0c fe 10 01 41 6a b4-8e 1f f7 8f 33 cb   .....Aj.....3.
              001c - 28 79 cf 01 0c 15 72 01-d9 19 79 43 56 94   (y....r...yCV.
              002a - 0f ec 2e ae 07 48 a3 f2-6e b5 36 75 76 ea   .....H..n.6uv.
```

A.3 Sample Metadata

A sample metadata file for a Trust Bundle conforming to the TrustBundleMetadata.xsd schema is shown below:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- MetaData File For LoA 3 TrustBundle -->
<TrustBundle xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="TrustBundleMetaData.xsd">
   <Profile>DirectTrust LoA-3</Profile>
   <DistributionPoint>http://www.trust.org/trustbundle1.p7b</DistributionPoint>
   <ValidFrom>2013-02-12T00:00:00</ValidFrom>
   <ValidTo>2014-02-12T00:00:00</ValidTo>
```

```
  <Anchors>
    <Anchor>
      <X509IssuerSerial>
        <X509IssuerName>CN=Direct L3 CA, OU=Issued By DigiCert, O=HISP Name, L=Town, ST=UT,
C=US</X509IssuerName>
        <X509SerialNumber>12345678</X509SerialNumber>
      </X509IssuerSerial>
    </Anchor>
  </Anchors>
</TrustBundle>
```