INTAREA
Internet-Draft
Intended status: Informational
Expires: 25 October 2026

R. Moskowitz
HTT Consulting
S. Card
AX Enterprize, LLC

23 April 2026

The DRIP DET public Key Infrastructure
draft-ietf-drip-dki-10

Abstract

The DRIP Entity Tag (DET) public Key Infrastructure (DKI) is a specific variant of classic Public Key Infrastructures (PKI) where the organization is around the DET, in place of X.520 Distinguished Names. Further, the DKI uses DRIP Endorsements in place of X.509 certificates for establishing trust within the DKI.

There are two X.509 profiles for shadow PKI behind the DKI, with many of their X.509 fields mirroring content in the DRIP Endorsements. These PKIs can at times be used where X.509 is expected and non-constrained communication links are available that can handle their larger size. It is recommended that a DRIP deployment implement both of these along side the Endorsement trees.

C509 (CBOR) encoding of all X.509 certificates are also provided as an alternative for where there are gains in reduced object size.

Author’s note: This draft is a partial update of all the additions needed for the DRIP-Full PKI to be ICAO ACCP compliant.

Status of This Memo

Table of Contents

1. Introduction

A DRIP Entity Tag (DET, [RFC9374]) public Key Infrastructure (DKI) is designed as a strict hierarchy, governed by the administrator of the DET prefix [IPv6-SPECIAL] and having the authority to authorize RAAs. RAAs in turn authorize HDAs within their domain. This authorization is managed via a set of DETs whose sole use is to define the DKI. The RAA Authorization DETs MUST reside in HID = RAA#|0 (Apex Authorization DET in HID = 0|0).

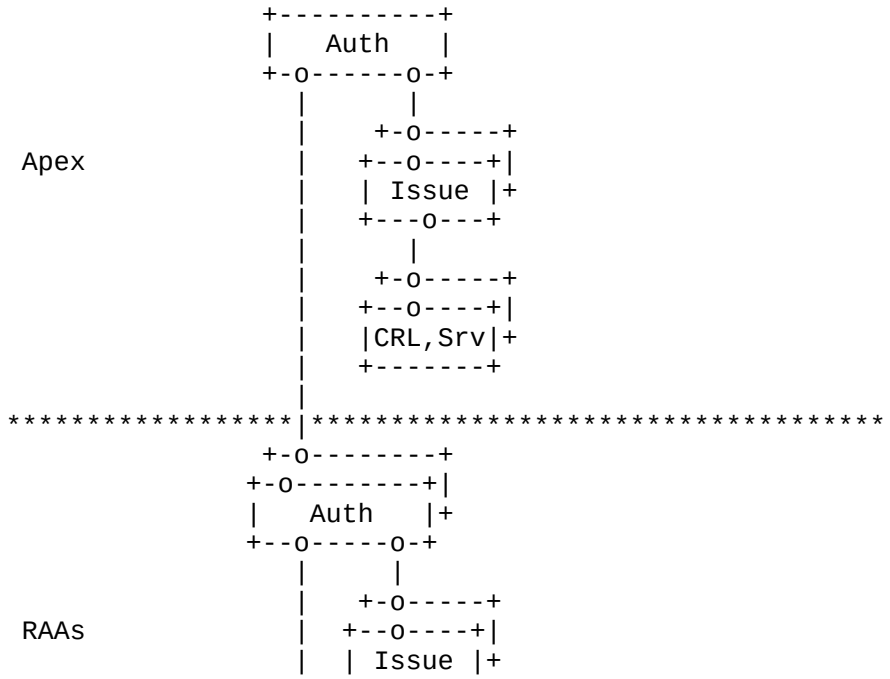
There are three main classifications/types of DETs:

Authorization DETs
: Used to assert the authorization of a DKI level.

Issuing DETs
: Used to assert operations within DKI level.

Operational DETs
: Used by operational entities within DKI level

All DETs exist in DET-Endorsements (Section 5.2 of [RFC9886]). These DET-Endorsements provide the proof of registration and thus trust. These DETs, through chained Endorsements define the DKI as follows:



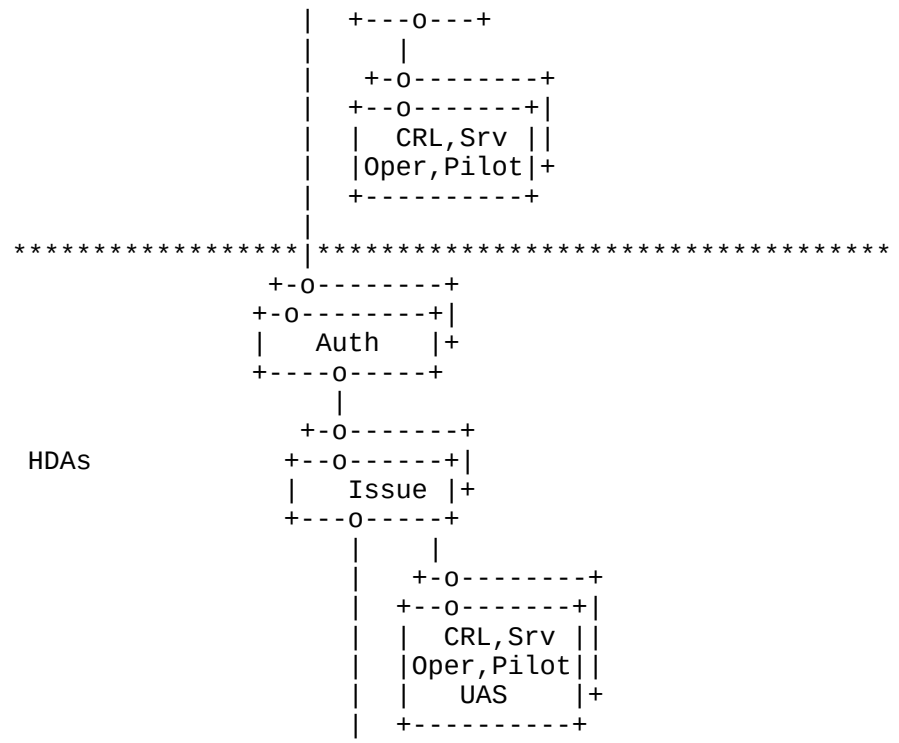


Figure 1: The DKI Endorsements

The Authorization DETs exist in a set of DET-Authorization-Endorsements. The lifetime of these endorsements SHOULD be no less than 1 year, recommended 5 years, and should not exceed 10 years. Endorsements SHOULD be reissued prior to expiry (may be for a new DET). DETs used to define this authorization are replaced per undetermined policy (note these DETs do very little signing, see Section 8.1).

This separation of DET type roles reduce the risk of private key loss for the critical Authentication DETs by making them infrequently used and only used in offline operations. It does make the chain of trust for a HDA customers' Operational DETs to be 4 Endorsements.

1.1. The DKI without an Apex Entity

The hierarchical design of the DKI is the most efficient possible with the least data transmission overhead. But it requires the participation of an Entity, in the role of the Apex, trusted by all the RAAs. The logical Entity for this role is the International Civil Aviation Authority (ICAO), but the processes for ICAO to take on this role are complex. Work is ongoing with the ICAO, but timing is indeterminate and immediately implementable alternatives are needed.

The DKI can work by the RAAs establishing mutual trust within a geographic region. It is envisioned that the initial RAA assignments will follow Section 6.2.1 of [RFC9886], Table 1. Without an Apex, each RAA self-endorses its Authentication DET, acting as its own apex. However, RAAs issued DETs (via their HDAs) will not exist in the air by themselves (except perhaps for some small island nations), thus a geographic regional consortium of RAAs will need to deploy some mechanism for mutual trust for their End Entities to fly together.

There are three reasonable approaches for RAAs to manage their mutual trust and it is likely that all will occur:

1. RAA Trust lists
2. RAA Cross-endorsements
3. Bridge RAA with cross-endorsements to RAAs

It is recommended that the RAA Trust List be used during initial DKI testing. The cross-endorsing options will need their own testing to work out how best to deploy them.

1.1.1. RAA Trust lists

A consortium of RAAs MAY choose to maintain a list of RAAs they trust. It is recommended that this list consist of the RAA’s Authentication DET and HI. Each RAA in the consortium SHOULD maintain its own list, signed with its Authentication DET.

This Trust List MAY contain each RAA’s Authentication DET self-endorsement validity dates. If a trusted RAA has more than one self-endorsement (most likely to support key rollover), including these dates makes it easier to have an RAA duplicated in the list.

How the RAAs communicate between themselves to maintain these lists is out of scope here. Each RAA SHOULD include validity dates in its Trust List. Frequency of Trust List updates is also out of scope here.

Trust Lists is the simplest method to implement, but may not be the simplest to maintain over time.

There is a natural Trust List of ALL RAAs, based on what is allocated in the DRIP DNS tree.

1.1.2. RAA Cross-endorsements

A consortium of RAAs MAY choose to cross-endorse each's Authentication DET. This is done by one RAA endorsing for its community, an other's Authentication DET. This establishes one-way trust; thus, in practice, each RAA needs to cross-endorse each RAA's Authentication DET within the consortium.

RAA Cross-endorsements definitely has a scaling (n^2) problem. It works for a starting point or for a very small group of RAAs.

How these RAA Cross-endorsements are discovered has not been defined at this point. One potential is via a to-be-defined DNS HHIT RR within the endorsing RAA's zone. This information would need to be cached by any potential offline entity.

1.1.3. Bridge RAA with cross-endorsements to RAAs

A consortium of RAAs MAY select one RAA to function as a "Bridge" between all members of the consortium. In this approach, the "Bridge RAA" does not authorize any sub-HDAs. Its sole purpose is the cross-endorse to member RAAs. The Bridge and each RAA cross endorse as in Section 1.1.2.

Bridge RAA Cross-endorsementing reduces the scaling challenge to only the number of RAAs in the consortium. Plus there is little need to communicate any changes in the cross-endorsementing to the various parties within the consortium. Thus this option scales the best out of the three alternatives to DKI Apex hierarchy.

How these RAA Cross-endorsements are discovered has not been defined at this point. The Bridge RAA will have to be known to all parties within the consortium. One potential, as above, is via the DNS HHIT RR (Section 5.1 of [RFC9886]) within the endorsing RAA's zone. This information would need to be cached by any potential offline entity.

1.2. Value add to DKI in X.509 Certificates

The Drip Architecture [RFC9434] does not use or need X.509 certificates or the associated Certificate Authories. However, there is considerable value for the Apex, RAAs, and HDAs to deploy the CAs described herein. Of considerable note is the inclusion of the ICAO Level of Assurance (LOA) policy OIDs, Section 6.1, that inform trust behind the DRIP Endorsement tree and the associated CAs.

A signing entity MUST follow the same LOA for all 3 objects: Endorsements, DRIP-Lite, and DRIP-Full certificates.

There may be other UAS applications that will just work with X.509 certificates, but would have to be customized to use DIRP Endorsements.

1.3. The C509 encoding of X.509 Certificates

A price in object size is paid in the ASN.1 encoding of X.509 certificates. This is often a barrier for use over constrained links and even storage demands on constrained processing platforms. The [C509-Certificates] provides an alternative encoding in two different manners:

1. An invertible CBOR re-encoding of DER encoded X.509 certificates [RFC5280], which can be reversed to obtain the original DER encoded X.509 certificate.

2. Natively signed C509 certificates, where the signature is calculated over the CBOR encoding instead of over the DER encoding as in 1. This removes the need for ASN.1 and DER parsing and the associated complexity but they are not backwards compatible with implementations requiring DER encoded X.509.

The invertible CBOR encoding is recommended for use here. This can be readily implemented through libraries that do the translation, as needed, between X.509 and c509.

2. Terms and Definitions

2.1. Requirements Terminology

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Definitions

This document uses the terms defined in Section 2.2 of Drip Requirements and Terminology [RFC9153] and in Section 2 of Drip Architecture [RFC9434]. The following new terms are used in the document:

Authorization DETs
: DETs whose use is to define a hierarchy level and endorse lower hierarchy level Authorization DETs and finally Issuing DETs at this hierarchy level. They the DETs in the Authentication Endorsements and X.509 certificates.

DKI
: A DRIP Entity Tag (DET) public Key Infrastructure. Similar to an X.509 PKI, but built on the DRIP Endorsements.

IAL (Identity Assurance Level)
: ICAO: "The confidence in the identity verification processes used to establish the identity of an entity associated with a certificate. The robustness of identity proofing and the certainty with which the identity is bound to the certificate."

International Aviation Trust Framework (IATF)
: The ICAO IATF is comprised of a set of policies, requirements, and best practices that will enable resilient and secured ground-ground, air-ground, and air-air exchange of digital information, and among both traditional and newly-emerging system stakeholders.

Issuing DETs
: DETs whose use is to sign Endorsements and X.509 certificates for Operational DETs that are at the same hierarchy level as the Issuing DET.

LOA (Level of Assurance)
: ICAO: "The confidence in the security measures that protect the private key and manage the certificate lifecycle."

Operational DETs
: DETs used by various entities in DRIP protocols and as non-routable IPv6 addresses. A partial list of such entities includes: GCS, Infrastructure (e.g. wireless tower systems), UAS Operators, Pilots-in-command, Servers, UA.

System Wide Information Management (SWIM)
The ICAO SWIM consists of Standards, Infrastructure and Governance enabling the management of Air Navigation Systems (ANS) related information and its exchange between qualified parties via interoperable services.

3. The DET public Key Infrastructure (DKI)

3.1. The DKI Levels

3.1.1. The Apex

The Apex Authorization DET is used to endorse RAA Authorization DETs and its own Apex Issuing DETs; it has no other use. This is the case for all Authorization DETs. Apex Issuing DETs are used to endorse DETs, with HID= 0|0, used by Apex services.

The DET Apex may be only theoretical if no Entity steps forward to provide this role.

3.1.2. The RAAs

Each RAA use its Authorization DET (HID = RAA#|0) to endorse its RAA Issuing DET(s) (also HID = RAA#|0) and for signing its HDA Authorization DETs (HID = RAA#|HDA#).

An RAA may have multiple Issuing DETs (HID = RAA#|0), each for a different use (e.g. CRL signing, RAA server signing). It is expected that, over time, an RAA will rollover its Issuing DETs, thus at times there will be more than ONE Issuing DET per role in use.

These Issuing DETs, like those at the Apex level, constitute an implicit HDA. There is no Authorization DET for this implicit HDA, but other than only signing for entities like servers needed by the RAA, it should be considered as an HDA in terms of policies.

An RAA may directly issue DETs for Operators/Pilots, but it is recommended that if the RAA has this responsiblity, it runs an HDA specifically for this function. This allows separation of the RAA role from the HDA. It is recommended that the RAA only offer Endorsing/Signing services for the functions of running the RAA.

The initial RAA range assignments are defined in Section 6.2.1 of [RFC9886], Table 1. It is anticipated that DRIP usage will expand to use into General/Civil Aviation. Thus at some point a block of RAAs will be set aside much like for the CTA-2063A [CTA2063A] range.

3.1.3. The HDAs

Each HDA use its Authorization DET to endorse its HDA Issuing DETs (e.g. RAA=267, HDA=567).

An HDA Issuing DET is used to endorse Operational DETs; those used by the HDA for its services (e.g. USS) and for Devices (e.g. UA, GCS, ground infrastructure) partaking in the HDA's services.

If the Operational DET is a Manufacturer DET, the "valid not after" date (vna) MUST be 99991231235959Z.

An HDA may directly issue DETs for Operators/Pilots. It is recommended that different Issuing HDAs be used for devices and people. They may be under the same Authentication HDA. Of importance is that there are different LOAs for CAs for people and devices per Section 6.1.

3.2. The Offline Requirement for Authentication DETs

The Authentication DETs private keys MUST NEVER be on a system with any network connectivity. Also efforts MUST be taken to limit any external digital media connections to these offline systems. Compromise of an Authentication DET compromises its and all lower hierarchy levels. Such a compromise could result in a major re-signing effort with a new Authentication DET. Also, during the time of compromise, fraudulent additions to the DKI could have occurred.

This means that the process whereby the Authentication DET is used to sign the Endorsement/X.509 certificate of its level's Issuing DET(s) and lower level Authentication DETs MUST be conducted in an offline manner (e.g. Section 6).

This offline process need not be onerous. For example, QR codes could be used to pass CSR objects to the offline Authentication DET system, and this system could produce QR codes containing the Endorsements and X.509 certificates it signed (Section 6.4).

A video conference between the parties could have one side show its QR code and the other copy and print it to move between the video conferencing system and the offline system. This is a simplification of a larger signing operation, but shows how such a signing need not require travel and expensive hand-off methodologies.

It should be noted that the endorsement of Issuing DETs follow the same restriction, as it is done with the Authentication DET. It MUST be conducted in an offline manner.

3.3. DNS view of DKI

The primary view of the DKI is within DNS. This is in the 3.0.0.1.0.0.2.ip6.arpa zone (Apex level of the DRIP IPv6 DET format).

In the DET DNS structure, only the Apex and RAA levels MUST be DNSSEC signed. The HDA level may be too dynamic for DNSSEC signing (e.g. hundreds of new EE Operational DETs per hour); trust in the EE Operational DETs within the HDA level comes through inclusion of the HDA Endorsement of EE object. A slow-churn HDA MAY use DNSSEC. The RAA and HDA levels MUST contain their Endorsement by higher object; this provides the needed trust in the Endorsement of EE objects. The Apex level Endorsement is self-signed, thus trust in it is only possible via DNSSEC.

Endorsements are stored in the DNS BRID RR (Section 5.2 of [RFC9886]). X.509 certificates in the DNS HHIT RR (Section 5.1 of [RFC9886]).

Other RR within these levels will vary. There also may be HIP, TLSA, and/or URI RR.

Each level continues on down the 3.0.0.1.0.0.2.ip6.arpa zone for its Authorization DET and Issuing DET(s). RR with FQDNs for services offered may also be present in various forms (e.g. a URI for the commercial FQDN for the DKI Entity). TLSA RR of DET SPKI may be directly included here. Same with HIP RR. The Authorization Endorsement SHOULD be present, as SHOULD be Issuing Endorsements.

3.4. Managing DET Revocation

For Operational DETs, there is no direct concept of DET revocation. Operational DETs are either discoverable via DNS or not valid despite being in a non-expired Endorsement signed an Issuing DET. Thus if an Issuing Entity needs to "revoke" an Operational DET it removes all entries for it from DNS, so a short TTL on those records is recommended.

Authorization and Issuing DETs are not so easily "revoked"; something akin to an X.509 CRL mechanism is needed. This could best be dealt with by Endorsements managed in the new DET RR that includes revocation status. Thus Section 5.2 of [RFC9886] defines the specific RR for Endorsements that will be used here. Minimally, at least the revocation status and revocation date(s) need to be in this RR. Until this RR is available, there is no mechanism, other than removal for Authorization and Issuing DET revocations.

3.5. The Offline cache of HDA Issuing Endorsements

The Offline cache of HDA Issuing Endorsements, used to verify various EE signed objects without needing DNS access, SHOULD consist of the HDA Authentication DET Endorsements of the HDA Issuing DETs. Thus the receiver has a trusted source of the HDA Issuing DET Public Key (HI) in a DRIP standard object (136 bytes). If the DKI DNS tree includes GEO location data and coverage, a receiver could query some service for a trusted cache within some radius of its location. Such as, please tell me of all HDAs within 100KM of...

This cache MAY contain the full chain up to the Apex. This could be helpful in limited connectivity environments when encountering an HDA Issuing DET under a unknown Authenticated HDA or RAA. The needed trust chain could be shorter.

3.5.1. HDA Offline Trust cache

There are situations where a list of specific HDAs for an entity to trust for some application is needed. This can best be met by maintaining a cache as above but only of the trusted HDA Issuing Endorsements. How a list of this limited trust is maintain and distributed is out of scope of this document and is left to those needing this specific feature.

4. The DKI’s Shadow PKI

The following defines the components of a DKI’s shadow PKI built from X.509 certificates with content that mirrors that in the DKI Endorsements. There are two profiles provided, DRIP-Lite and DRIP-Full. Both may be used, or the community may select one for deployment. In both cases, the PKI tree mirrors that of the DKI levels (Section 3.1).

It is recommended that at least the DRIP-Full Section 4.2 be deployed, as its CA certificates contain ICAO policy OIDs the reflect on the whole DKI deployment.

A server MUST implement the generation of the DRIP Endorsement object. It SHOULD also generate both the PKI profiles. In this document, Endorsing Server and CA interchangeably.

At this point in defining the shadow PKIs, alternatives to a strict hierarchy is still an open work item. This work will follow the pattern set in Section 1.1.

4.1. Shadow DRIP-Lite with minimal content Certificates

The DRIP-Lite is designed to fully mirror the DKI in the smallest reasonable X.509 certificates (e.g. 240 bytes for DER), but still adhere to [RFC5280] MUST field usage.

4.1.1. DRIP Lite X.509 certificate profile

The following is the profile for the DRIP X.509 Lite certificates

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
        Signature Algorithm: ED25519
        Issuer: CN =
        Validity
            Not Before:
            Not After :
        Subject: {CN = or Empty}
        Subject Public Key Info:
            Public Key Algorithm: ED25519
                ED25519 Public-Key:
                pub:
        X509v3 extensions: {Operation Certs ONLY}
            X509v3 Subject Alternative Name: critical
                IP Address:
    Signature Algorithm: ED25519
    Signature Value:
```

Figure 2: The X.509 Lite Profile

Field	Authorization	Issuing	Operational	Comments
Serial Number	MUST	MUST	MUST	Section 4.1.2.1
Subject	MUST	MUST	MUST NOT	Section 4.1.2.2
Issuer	MUST	MUST	MUST	Section 4.1.2.4
Subject Alternative Name (SAN) IP6	MUST	MUST	MUST	Section 4.1.2.3
Subject Alternative Name (SAN) URI	MAY	MAY	MAY	Section 4.1.4.1
Basic Constraints	MUST	MUST	MUST NOT	Section 4.1.3.1
Subject Key Identifier (SKI)	MUST NOT	MUST NOT	MUST NOT	-
Authority Key Identifier (AKI)	MUST NOT	MUST NOT	MUST NOT	-
Key Usage	MAY	MAY	MAY	Section 4.1.4.2
CA Policy OIDs	MAY	MAY	MAY	Section 4.1.4.3

Table 1: DKIX-Lite Field Matrix

4.1.2. DRIP Lite Mandatory Certificate Content

The following detail the Mandatory to include content in all DRIP Lite certificates.

4.1.2.1. Serial Number

The Serial Number is a MUST field, but it has no usage in this DRIP-Lite. It is 1-byte in size and thus duplicates are guaranteed. To drop this field could make many X.509 parsing libraries fail.

However, CA certificate's Serial Number MAY be the common 20 bytes. This is to avoid possible issues with general software expecting this size Serial Numbers for CAs.

If Serial Numbers are incrementally assigned, 31 bits are sufficient for an Issuing CA with 2B DETs active. A CA should determine its best-used Serial Number field size to limit the impact of this field on the certificate size.

4.1.2.2. Subject

The Subject field is only used in Authentication and Issuing Certificates. For Entity Certificates, the Subject is Empty and the DET will be in Subject Alternative Name (SAN). In the SAN, the DET can be properly encoded as an IPv6 address.

The Subject field in Authentication and Issuing Certificates uses the following format:

DRIP-{APEX|RAA|HDA}-{A|I}[-RAA#][-HDA#]

Examples:

DRIP-RAA-A-16376
DRIP-HDA-I-16376-16376

Figure 3: Lite CA Certificate Subject Name Format

The CA Subject Name serves a duo purpose: foremostly, to place the CA within the DKI tree, but secondly for outside of DRIP usage to tag that this CA's function is to serve DRIP Entities.

4.1.2.3. Subject Alternative Name

Subject Alternative Name is only used in Operational (End Entity) certificates. It is used to provide the DET as an IP address with an Empty Subject (SAN MUST be flagged as Critical).

The Subject Alternative Name is also used in Manufacturer DET certificates. These may contain the hardwareModuleName as described in [IEEE 802.1AR] that references [RFC4108].

Per [RFC5280] and [IEEE 802.1AR], Manufacturer DET certificates with hardwareModuleName MUST have the notAfter date as 99991231235959Z.

4.1.2.4. Issuer

The Issuer MUST be the higher level's DET.

The Issuer for the Apex Authentication certificate MUST be its DET (indicating self-signed). If the RAA Authentication certificate is self-signed (i.e., no Apex), its Issuer is its DET.

The Issuer content of its DET assists in finding this specific Issuer in the DRIP ip6.arpa. DNS tree and any additional information.

4.1.3. DRIP Lite Mandatory CA Certificate Content

The following detail the Mandatory content for DRIP Lite CA certificates.

4.1.3.1. Basic Constraints

CA certificates MUST contain the CA=True object, flagged Critical as a Basic Constraint.

4.1.4. DRIP Lite Optional CA Certificate Content

The following detail the Optional content for DRIP Lite CA certificates. Inclusion of these objects are based on the policies of the CA using them and CAs higher in the trust chain.

4.1.4.1. CA Subject Alternative Name URI

This is the one exception to Section 4.1.2.3. A CA certificate MAY have a SAN URI, containing a pointer where additional information on the CA and certificates under its control. For example, an authorized authority may access EE PII like an Operator phone number through this URI link.

4.1.4.2. Key Usage

The CA certificate MAY contain the keyUsage extension. For example, it may assert Certificate Signing and flag this as Critical.

4.1.4.3. CA Policy OIDs

The CA MAY have policy OIDs defining rules for EEs within its domain. The OIDs used here will tend to be within ICAO's arc of 1.3.27.16.

4.1.5. The test DKI and Lite PKI

The test DKI and Lite PKI, (Appendix A/Appendix B), were created using the python scripts at [drip_scripts]. First csr-gen.py is used to create an X.509 CSR (and optionally the EdDSA keypair). This CSR is minimal in content. For example, a UA might only have knowledge of its Manufacturer Serial Number and can generate its keypair. Per [drip-registration], this CSR may be sent to the CA with additional information provided by the Operator, for example desired validityDates. The raa-endorse.py and hda-endorse.py scripts are provided to produce the DRIP Endorsements and X.509 certificates.

At this time, with no Apex level, each RAA Authorization CA is self-signed. These are created using the RAA's CSR and its own keypair as input to the raa-endorse.py script. Normally, the raa-endorse.py script is used to create the HDA's Authorization and Issuing CAs and the hda-endorse.py script for the End Entity certificates.

4.2. Shadow PKI with DRIP-Full Certificates

The X.509 certificates are minimalistic (less than 400 bytes for DER). Any DRIP specific OIDs should come from the ICAO arc (e.g. 1.3.27.16.1).

4.2.1. DRIP Full X.509 certificate profile

The following is the profile for the DRIP X.509 certificates

```
   Certificate:
       Data:
           Version: 3 (0x2)
           Serial Number:
           Signature Algorithm: ED25519
           Issuer: CN =
           Validity
               Not Before:
               Not After :
           Subject: {CN = or Empty}
           Subject Public Key Info:
               Public Key Algorithm: ED25519
                   ED25519 Public-Key:
                   pub:
           X509v3 extensions:
                X509v3 Subject Alternative Name: critical {in EE}
                   IP Address:
              X509v3 Subject Key Identifier: {not in EE}
               X509v3 Authority Key Identifier:
               X509v3 Basic Constraints: critical
               X509v3 Key Usage: critical
       Signature Algorithm: ED25519
       Signature Value:
```

Figure 4: DRIP X.509 certificate profile

Field	Authorization	Issuing	Operational	Comments
Serial Number	MUST	MUST	MUST	Section 4.2.2.1
Subject	MUST	MUST	MUST NOT	Section 4.2.2.2
Issuer	MUST	MUST	MUST	Section 4.2.2.4
Subject Alternative Name (SAN) IP6	MUST	MUST	MUST	Section 4.2.2.3
Subject Alternative Name (SAN) URI	MAY	MAY	MAY	Section 4.2.4.1
Basic Constraints	MUST	MUST	MUST NOT	Section 4.2.3.1
Subject Key Identifier (SKI)	MUST	MUST	MUST NOT	Section 4.2.2.5
Authority Key Identifier (AKI)	MUST	MUST	MUST	Section 4.2.2.6
Key Usage	SHOULD	SHOULD	SHOULD	Section 4.2.4.2
CA Policy OIDs	RECOMMENDED	RECOMMENDED	RECOMMENDED	Section 4.2.4.3

Table 2: DKIX-Full Field Matrix

4.2.2. DRIP Full Mandatory Certificate Content

The following detail the Mandatory to include content in all DRIP Lite certificates.

4.2.2.1. Serial Number

The certificates will contain a 20-byte randomly generated Serial Number, compliant with CABForum recommendations. Serial Numbers are included for CRL functionality.

4.2.2.2. Subject

The Subject field is only used in Authentication and Issuing Certificates. For Entity Certificates, the Subject is Empty and the DET will be in Subject Alternative Name (SAN). In the SAN, the DET can be properly encoded as an IPv6 address.

The Subject field in Authentication and Issuing Certificates uses the following format:

DRIP-{APEX|RAA|HDA}-{A|I}[-RAA#][-HDA#]

Examples:

DRIP-RAA-A-16376
DRIP-HDA-I-16376-16376

Figure 5: CA Certificate Subject Name Format

The CA Subject Name serves a duo purpose: foremostly, to place the CA within the DKI tree, but secondly for outside of DRIP usage to tag that this CA’s function is to serve DRIP Entities.

4.2.2.3. Subject Alternative Name

Subject Alternative Name is only used in Operational (End Entity) certificates. It is used to provide the DET as an IP address with an Empty Subject (SAN MUST be flagged as Critical).

The Subject Alternative Name is also used in Manufacturer DET certificates. These may contain the hardwareModuleName as described in [IEEE 802.1AR] that references [RFC4108].

Per [RFC5280] and [IEEE 802.1AR], Manufacturer DET certificates with hardwareModuleName MUST have the notAfter date as 99991231235959Z.

4.2.2.4. Issuer

The Issuer MUST be the higher level’s DET.

The Issuer for the Apex Authentication certificate MUST be its DET (indicating self-signed). If the RAA Authentication certificate is self-signed (i.e., no Apex), its Issuer is its DET.

The Issuer content of its DET assists in finding this specific Issuer in the DRIP ip6.arpa. DNS tree and any additional information.

4.2.2.5. Subject Key Identifier

The Subject Key Identifier MUST be the DET. This is a major deviation from "standard" X.509 certificates that hash (normally with SHA2) the Public Key to fill the Subject Key Identifier.

The Subject Key Identifier is NOT included in EE certificates. If needed by some application, it is identical with SAN.

4.2.2.6. Authority Key Identifier

The Authority Key Identifier MUST be the higher level’s Subject Key Identifier (i.e. DET). This partially follows standard practice to chain up the Authority Key Identifier’ from the Subject Key Identifier, except for how the Subject Key Identifiers are populated.

The Authority Key Identifier for the Apex Authentication (or self-signed RAA Authentication) certificate MUST be the Subject Key Identifier (indicating self-signed).

4.2.3. DRIP Full Mandatory CA Certificate Content

The following detail the Mandatory content for DRIP Full CA certificates.

4.2.3.1. Basic Constraints

CA certificates MUST contain the CA=True object, flagged Critical as a Basic Constraint.

4.2.4. DRIP Full Optional CA Certificate Content

The following detail the Optional content for DRIP Full CA certificates. Inclusion of these objects are based on the policies of the CA using them and CAs higher in the trust chain.

4.2.4.1. CA Subject Alternative Name URI

This is the one exception to Section 4.2.2.3. A CA certificate MAY have a SAN URI, containing a pointer where additional information on the CA and certificates under its control. For example, an authorized authority may access EE PII like an Operator phone number through this URI link.

4.2.4.2. Key Usage

The CA certificate SHOULD contain the keyUsage extension. For example, it may assert Certificate Signing and flag this as Critical.

4.2.4.3. CA Policy OIDs

It is recommended that a CA have policy OIDs defining rules for EEs within its domain. The OIDs used here will tend to be within ICAO's arc of 1.3.27.16.

If the CA certificate has policy OIDs, it MUST include an ICAO LOA OID defining "the confidence in the security measures that protect the private key and manage the certificate lifecycle". Currently defined OIDs [ICAO-Doc-10169] in ICAO's LOA arc of 1.3.27.16.1.1.0:

OID	Description	Applicability
1	Low	This level is relevant to environments where Risks and consequences of data Compromise are low. Subscriber Private Keys shall be stored in software at this Identity Assurance Level (IAL).
2	LowDevice	This policy is identical to that defined for the Low Assurance policy (above) with the exception of identity proofing, re-key, and Activation Data.
3	Low-TSP Mediated Signature	This policy is identical to that defined for the Low Assurance policy (above) with the exception that the Private key is not in the possession of the user, but rather by a Trust Service Provider.

4	Medium	This level is relevant to environments where Risks and consequences of data Compromise are moderate. This may include transactions having substantial monetary value or Risk of fraud or involving access to private information where the likelihood of malicious access is substantial. Subscriber Private Keys shall be stored in software at this IAL.
5	MediumDevice	This policy is identical to that defined for the Medium Assurance policy (above) with the exception of identity proofing, re-key, and Activation Data.
6	Medium-TSP Mediated Signature	This policy is identical to that defined for the Medium Assurance policy (above) with the exception that the Private key is not in the possession of the user, but rather by a Trust Service Provider.
7	MediumHardware	This policy is identical to that defined for the Medium Assurance policy (above) with the exception of Subscriber Cryptographic Module requirements described in [ICAO-Doc-10169].
8	MediumDeviceHardware	This policy is identical to that defined for the Medium Hardware Assurance policy (above) with the exception of identity proofing, re-key, and Activation Data.
9	High	This level is relevant to environments where Risks and consequences of data Compromise are high. Certificates issued at the High-cardAuth IAL shall only be issued for Card Authentication, as defined

			by NIST SP 800-73 or equivalent standard. Further, this policy is identical to that defined for the identical to the MediumHardware IAL except where specifically noted in [ICAO-Doc-10169].
10	High-CardAuth		This level is relevant to environments where Risks and consequences of data Compromise are high. Certificates issued at the High-cardAuth IAL shall only be issued for Card Authentication, as defined by NIST SP 800-73 or equivalent standard.
11	High-ContentSigning		This level is relevant to environments where Risks and consequences of data Compromise are High. This may include transactions having substantial monetary value or Risk of fraud or involving access to private information where the likelihood of malicious access is substantial. Certificates issued at the High IAL shall only be issued to human Subscribers. Certificates issued at the High-contentSigning IAL shall only be issued to the CMS for signing the HIGH card security objects (e.g. Certificates, CRLs, OCSP responses). Further, this policy is identical to that defined for the identical to the MediumHardware IAL except where specifically noted in [ICAO-Doc-10169].

Table 3: ICAO LOA OID Assignments under 1.3.27.16.1.1.0

In this document, the term “Device” is defined as a Non-Person Entity, i.e., an entity with a digital identity that acts in cyberspace but is not a human actor. This can include Organizations, hardware devices (e.g. a UA), software applications, and information artifacts.

End Entity Certificates issued to Devices shall assert policies mapped to LowDevice, MediumDevice, MediumDeviceHardware, or High Content Signing policies. All other policies defined here should be reserved for human Subscribers when used in End Entity Certificates. Thus it is recommended that Issuing CAs/HDAs should be segregated by device and human subscribers so policy can be stated at the CA level rather in individual certificates.

4.2.5. The DRIP-Full test PKI

Author’s Note: At this time, the following DRIP-Full test PKI and Appendix B.2 is is a work-in-progress.

The DRIP-Full test PKI, following the test DKI, was built with openSSL using the "req" command to create a CSR and the "ca" command to sign the CSR, making the certificate.

It should be noted that these CSRs have all the content, less the validityDates, for making a DRIP Endorsement, such that a registrar may prefer to receive CSRs and use it to make both structures. The registrar, per CA practices will provide the validityDates per its policy.

The self-signed certificates created by "req -x509" does not allow selection of the validity dates, only the number of days from NOW. The hack used around this limitation is to create a throw-away self-signed certificate as above with the Apex’s (or RAA’s) DET. Then create a CSR with that DET and sign it with the throw-away certificate, setting the validity dates as desired. This now becomes the actual Apex (or RAA’s) self-signed Authentication certificate and the throw-away certificate can now be thrown away.

5. The DKI and the ICAO SWIM PKI

ICAO advocates for a federated implementation model of individual PKIs based on common standards, the total of which can be considered an international aviation trust framework. This is embodied in Aviation Common Certificate Policy (ACCP) [ICAO-Doc-10169]. A test of a compliant PKI is rolling out for testing the Aviation System Wide Information Management (SWIM) environment.

Currently, this PKI is using ECDSA P-256 in its certificates. This is equivalent to DET SuiteID of "3". The subjectNames in use can readily by mapped to RAAs (Section 6.2.1 of [RFC9886], Table 1) and HDAs. Thus it is a potential straight-forward technical work item to add DET support into the PKI.

The DETs can readily be stored in subjectAltName or more interestingly in subjectKeyIdentifier (and thus authorityKeyIdentifier).

There are a number of advantages for SWIM to have DETs and the matching DNS available. For example, the "cost" of adding DETs to these certificates could result in moving much of their content into DNS SRV RR and potentially reduce their size by 1/3rd. DETs as the authorityKeyIdentifier would enable DNS for Trust Chain discovery.

Another approach is direct inclusion in this PKI of the DET "Lite" EE certificates for constrained A2A communications.

Discussions are ongoing with those involved with the ACCP and this could open up DET usage into General/Civil Aviation.

6. DRIP Tamper Evident CA Servers and Protocols

The DRIP architecture [RFC9434] anticipates thousands of CAs for the thousands of administrative entities involved in the DRIP ecosystem. Current business models for deploying public-facing CAs are just not practical or affordable at this volume. Yet these DRIP CAs need to provide an acceptable LOA (Section 4.2.4.3).

In-depth analysis of the CA needs for the DKI have resulted in a Tamper Evident CA design as described here. This Tamper Evident design SHOULD meet the Medium and MediumDevice LOAs in Section 4.2.4.3.

If all data into and out from the DRIP CAs are strictly controlled, the sole hard-to-detect risk is are the keypairs for the CA really generated by the CA and not an artifact of corrupted code.

For the Apex, RAA Auth, and HDA Auth CAs they need to have as input the next level down CSR and associated data and output the resultant DRIP Endorsements and X.509 certificates. These CAs are basically kept locked away except during these occasional signing operations. A CA with all data ports sealed and only the camera to read QR encoded data and the screen to display similarly QR encoded data can provide this Tamper Evident CA design.

The HDA Issuing CA (and any other Issuing CA) will be too heavily used to be locked away. But if it is connected via USB to a USS provider server and ONLY the same data objects as above are processed via that USB connection, almost the same assurance level can be shown.

6.1. CA servers LOA

Apex and RAA CA servers SHOULD meet at least the Medium LOA. They MAY meet the High LOA.

HDA Authentication CA servers SHOULD also meet at least the Medium LOA. They MAY meet the High LOA. If they only support Devices, they MAY assert the appropriate Device LOA.

HDA Issuing CA servers SHOULD also meet at least the Medium LOA. They MAY meet the High LOA. It is recommended that CAs separately service Devices and People and assert the appropriate LOA.

6.2. What Tamper Evident means

Here, Tamper Evident means that any unofficial access to the CA is recorded and steps can be taken to mitigate any damages.

Start with a system with a known software build and all needed applications. This part of the setup MUST be done without any Internet connectivity. Perhaps from known CD/DVD images. During this setup, all data ports, other than that used for the CD/DVD reader are sealed with security tape. After the build, that port is also so sealed.

The only remaining input devices to the system is its camera and keyboard. The only output device is the integral display.

A notebook is best for this purpose, as once closed, security tape can seal it closed thus any attempt to access the keyboard will be evident. Any use of this CA is videoed and the time from the security seal broken to a new one attached is logged.

Thus any tampering with the system can be detected, as security seals will have been removed to gain access.

6.2.1. Issuing CA special case

Issuing CAs present a special case and a serious challenge. These servers could be signing thousands of DETs per day; their use MUST be an automated, always accessible service to the USS.

One approach would be these CAs are hardwired to a USS server via a USB null-modem cable that has security tape on both end connectors. The application controlling this USB port on the CA ONLY accepts, and outputs, a set of expected X.509 and related objects. Any other data sent over this USB to the server will return an error. Any attempt to attach a different device other than the USS server will cause a fault.

By using a serial interface, there are significant limitations on what the OS can be tricked to do. Since this cable has security tape on the connectors, any changing of the cable should be evident. There might be other approaches to using a serial interface.

6.3. The Data Exchange

The full extent of this exchange is a work-in-progress. It will be modeled after the exchanges covered in [drip-registration].

The data used by the CA can be grouped into three catagories:

What the CA knows
- Its keypair
- Its RRA, HDA, and SuiteID (DET generated)
- Its ICAO LOA
- Its serialNumber length in bits
- Does it sign CAs or Entities

Data from USS/Operator
- Signee CSR file
- Validity dates
- Is HDA (if different from its CA’s)
- Is request to create a CA of for an Entity
- If for CA, CA’s name

Data returned to USS/Operator
- DET, Endorsements and Certificates

This breakdown will advise the development of the CA operation. Still missing, and needing work, is signing a trust list.

6.4. QR Codes for the Exchange Protocol

QR codes can encode up to 3KB of data. There are programs that can monitor the server's camera (e.g. zbarcam), producing a data file that can then be reviewed for correctness and processed.

Likewise, there are programs (e.g. qrencode) that can accept a data file to create a QR code. If the data file is larger than 3KB, it MUST be fragmented and then generate multiple QR codes.

These QR codes can be passed between DRIP administrators in a verifiable manner (to mitigate fraudulent activities) so that there may not be a need for in-person key signing. The HDA operator can express a paper with the CSR QR code. So proofing by the RAA operator can validate this paper and the code really came from the HDA operator. After using this CSR, the RAA operator takes pictures of the generated output QR codes and gets those to the HDA operator who inputs them to their server as needed.

6.5. USB for the Exchange Protocol

The USB exchange applications are both simpler and more demanding than the QR code approach. There are standard libraries for accepting data over USB and many ways to build this. The application monitors data coming in over USB and sends back data as appropriate.

The challenge comes in ensuring that any attempts to alter the USB connection, as in unplugging the USS server and attaching a bootable USB drive, are detected and blocked. Only the exchange program has access, at the system level, to the USB port, and only expected data is accepted and returned.

Using a serial null-modem inline on the USB connection may be the simplest way to enforce the USB behavior so that an attack on the USS side could not get the CA side to accept attack code. It would take a physical cable change that would be visibly evident.

7. IANA Considerations

TBD

8. Security Considerations

Risks in the DKI are similar to those in any X.509 PKI. The methodologies to mitigate risk in PKI management should be considered and implemented as appropriate.

The DKI presents a tree-breath problem that is rarely seen in PKIs and needs practical solutions to minimize cost of operations and not introduce risks needlessly. Consider that there can be 16,384 RAAs. Assume only 10,000 RAAs, each of which Authentication DET Endorsement has a 10 year validity period. This means that, on average, 1,000 RAAs per year need to rekey their Authentication DET Endorsement, or on average, 3 per day. Current witnessed key signing processes will not scale to this volume. Some virtual method (like in Section 3.2) is needed.

8.1. Protecting against DKI/PKI compromise

There is always a risk of key compromise that could be a major setback to the operation of a PKI and likewise the DRIP DKI. To mitigate this risk, the Authentication DETs MUST only be used in offline signing operations. They MUST NEVER be used on connected systems. The information needed to create the Endorsements and X.509 certificates are brought to them on media that cannot transfer code, for example in a QR code. The objects that are created are then transferred away from the offline system to be used where needed.

It should be noted that this offline process MUST be followed down the DKI/PKI tree. That is, the Apex has offline operations that include signing the RAA Authentication DET that will be used in the RAA’s set up.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[C509-Certificates] Mattsson, J. P., Selander, G., Raza, S., Hglund, J., Furuhed, M., and L. Liao, "CBOR Encoded X.509 Certificates (C509 Certificates)", Work in Progress, Internet-Draft, draft-ietf-cose-cbor-encoded-cert-18, 22 April 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-cbor-encoded-cert-18>.

[cbor_c509_code]
"CBOR-certificates", July 2024,
<https://github.com/cose-wg/CBOR-certificates/>.

[CTA2063A] ANSI/CTA, "ANSI/CTA 2063-A Small Unmanned Aerial Systems Numbers", September 2019, <https://shop.cta.tech/products/small-unmanned-aerial-systems-serial-numbers>.

[drip-registration]
Wiethuechter, A. and R. Moskowitz, "Registration & Usage of DRIP Entity Tags for Trustworthy Air Domain Awareness", Work in Progress, Internet-Draft, draft-wiethuechter-drip-det-tada-01, 2 October 2025, <https://datatracker.ietf.org/doc/html/draft-wiethuechter-drip-det-tada-01>.

[drip_scripts]
"Python scripts to generate DETs and Endorsements", March 2025, <https://github.com/ietf-wg-drip/drip-scripts>.

[ICAO-Doc-10169]
ICAO, "ICAO Aviation Common Certificate Policy, Doc 10169", To be available by Q2, 2025.

[IEEE 802.1AR]
IEEE, "IEEE Standard for Local and Metropolitan Area Networks - Secure Device Identity", DOI 10.1109/ieeestd.2018.8423794, 31 July 2018, <https://doi.org/10.1109/ieeestd.2018.8423794>.

[IPv6-SPECIAL]
IANA, "IANA IPv6 Special-Purpose Address Registry", <https://www.iana.org/assignments/iana-ipv6-special-registry/>.

[RFC4108] Housley, R., "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages", RFC 4108, DOI 10.17487/RFC4108, August 2005, <https://www.rfc-editor.org/info/rfc4108>.

[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, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC9153] Card, S., Ed., Wiethuechter, A., Moskowitz, R., and A. Gurtov, "Drone Remote Identification Protocol (DRIP) Requirements and Terminology", RFC 9153, DOI 10.17487/RFC9153, February 2022, <https://www.rfc-editor.org/info/rfc9153>.

[RFC9374] Moskowitz, R., Card, S., Wiethuechter, A., and A. Gurtov, "DRIP Entity Tag (DET) for Unmanned Aircraft System Remote ID (UAS RID)", RFC 9374, DOI 10.17487/RFC9374, March 2023, <https://www.rfc-editor.org/info/rfc9374>.

[RFC9434] Card, S., Wiethuechter, A., Moskowitz, R., Zhao, S., Ed., and A. Gurtov, "Drone Remote Identification Protocol (DRIP) Architecture", RFC 9434, DOI 10.17487/RFC9434, July 2023, <https://www.rfc-editor.org/info/rfc9434>.

[RFC9886] Wiethuechter, A., Ed. and J. Reid, "DRIP Entity Tags (DETs) in the Domain Name System", RFC 9886, DOI 10.17487/RFC9886, December 2025, <https://www.rfc-editor.org/info/rfc9886>.

Appendix A. Test DETs and Endorsements

The following are test DETs and Endorsements for the test DKI. This testing environment is open to all. There are 4 RAAs available for others to build out. HDAs under the 4 preset RAAs, or under any of the 4, built out by others, are available. Finally the test HDA is available for setting up a handful of entities. Any tester wanting more than a few DETs for entities should plan on doing that under their own HDA.

The following are the test values and objects. They were generated using the csr-gen.py and endorse.py scripts available at [drip_scripts]. The endorse script currently creates all three objects: Endorsement, Lite-cert, and Full-cert.

Note, that as there is no APEX level at this time, the RAA Endorsement is self-signed.

```
raa16376
    Authorizing DET  (HID=16376|0)

# SN is there just because script needs it.
python csr-gen.py --keyname=raa16376 --serialnumber=0 --raa=16376/
 --hda=0

cat raa16376-server-self.dat
raa = 16376
hda = 0
suiteid = 5
serialnumberbits = 15
cakey = "raa16376"
LOA_str = "1.3.27.16.1.1.0.1"
selfsign = True

cat raa16376-self.dat
hda = 0
vnb = "03/01/2025"
vna = "03/01/2027"
caname = "RAA-A-16376"
clientcsr = "raa16376"
clientpem = "raa16376"
entitycert = False

python endorse.py --serverdat=raa16376-server-self/
 --commandfile=raa16376-self

CA
DET: 2001003ffe000005f885c8ee6ad2a7af
DET: 2001:003f:fe00:0005:f885:c8ee:6ad2:a7af
Client
DET: 2001003ffe000005f885c8ee6ad2a7af
DET: 2001:003f:fe00:0005:f885:c8ee:6ad2:a7af
Client HI: 9229539f2ae6a961d1c24977455da98162e53efc98df9eb30f72537
    6993a7275
Client Endorsement by CA( 136.0  bytes): 67c294506b84fb502001003ff
    e000005f885c8ee6ad2a7af9229539f2ae6a961d1c24977455da98162e53ef
    c98df9eb30f725376993a72752001003ffe000005f885c8ee6ad2a7afc8d8c
    f7eafe60e2ef99fc185521c4f59612cb961fcff24557ca51c66207c293bcc5
    5849238401318c6781acdfebadbb2255fa1bdba376493a7c7f1fa66939c01
client SN: x2344
```

```
   hda16376-16376
       Authorizing DET  (HID=16376|16376)
   # SN is there just because script needs it.
   python csr-gen.py --keyname=hda16376-16376A --serialnumber=0

   cat raa16376-server.dat
   raa = 16376
   hda = 0
   suiteid = 5
   serialnumberbits = 15
   cakey = "raa16376"
   LOA_str = "1.3.27.16.1.1.0.1"
   selfsign = False

   cat hda16376-16376A.dat
   hda = 16376
   vnb = "03/02/2025"
   vna = "03/30/2026"
   caname = "HDA-A-16376-16376"
   clientcsr = "hda16376-16376A"
   clientpem = "hda16376-16376A"
   entitycert = False

   python endorse.py --serverdat=raa16376-server/
    --commandfile=hda16376-16376A

   CA
   DET: 2001003ffe000005f885c8ee6ad2a7af
   DET: 2001:003f:fe00:0005:f885:c8ee:6ad2:a7af
   Client
   DET: 2001003ffe00000505cacfa11e780bd5
   DET: 2001:003f:fe00:0005:05ca:cfa1:1e78:0bd5
   Client HI: b82b27f86b013468fe48d85b54f01bf65385f302ab2e136dc51a3b9
       29c88ce5a
   Client Endorsement by CA( 136.0  bytes): 67c3e5d069c9f5402001003ff
       e00000505cacfa11e780bd5b82b27f86b013468fe48d85b54f01bf65385f30
       2ab2e136dc51a3b929c88ce5a2001003ffe000005f885c8ee6ad2a7af59789
       8999ff2718e701681d4044a853cae8d9bb3b26e7a7cf934ff55f78bcfe6054
       c544f249ec2b751022b7ff2bb53f28f7e012a2780cc5b3b3b01f85737cd09
   client SN: x5589
```

```
      Issuing DET  (HID=16376|16376)

   # SN is there just because script needs it.
   python csr-gen.py --keyname=hda16376-16376I --serialnumber=0

   cat hda16376-16376A-server.dat
   raa = 16376
   hda = 16376
   suiteid = 5
   serialnumberbits = 15
   cakey = "hda16376-16376A"
   LOA_str = "1.3.27.16.1.1.0.2"
   selfsign = False

   cat hda16376-16376I.dat
   vnb="03/02/2025"
   vna="02/27/2026"
   caname = "HDA-I-16376-16376"
   clientcsr = "hda16376-16376I"
   clientpem = "hda16376-16376I"
   entitycert = False

   python endorse.py --serverdat=hda16376-16376A-server/
    --commandfile=hda16376-16376I
   CA
   DET: 2001003ffe3ff805234fa4afcc22b5b4
   DET: 2001:003f:fe3f:f805:234f:a4af:cc22:b5b4
   Client
   DET: 2001003ffe3ff8056dcf2c1a98a46c42
   DET: 2001:003f:fe3f:f805:6dcf:2c1a:98a4:6c42
   Client HI: cc75d75df778734d2e5b682f6ff938abf10a1026f788dca99945cbd
       dacf3d723
   Client Endorsement by CA( 136.0  bytes): 67c3e5d069a124d02001003ff
       e3ff8056dcf2c1a98a46c42cc75d75df778734d2e5b682f6ff938abf10a102
       6f788dca99945cbddacf3d7232001003ffe3ff805234fa4afcc22b5b46fb4b
       40adc6892e306da23322947a04e8119f22b82df3faff4ff995a740131ac47f
       7d76d94f86fc750b82c7a88bd7cf77ab67d4e93ce64b7e2b9200ab93ed40a
   client SN: x5789
```

```
       UA DET in 16376.16376

   python csr-gen.py --keyname=ua1-16376-16376/
    --serialnumber=x1224AABBCCDDEE56789

   cat hda16376-16376I-server.dat
   raa = 16376
   hda = 16376
   suiteid = 5
   serialnumberbits = 23
   cakey = "hda16376-16376I"
   selfsign = False

   cat ua1-16376-16376.dat
   vnb="03/04/2025"
   vna="02/25/2026"
   clientcsr = "ua1-16376-16376"
   clientpem = "ua1-16376-16376"
   entitycert = True

   python endorse.py --serverdat=hda16376-16376I-server/
    --commandfile=ua1-16376-16376
   CA
   DET: 2001003ffe3ff8056dcf2c1a98a46c42
   DET: 2001:003f:fe3f:f805:6dcf:2c1a:98a4:6c42
   Client
   DET: 2001003ffe3ff80560ac736574d2c466
   DET: 2001:003f:fe3f:f805:60ac:7365:74d2:c466
   Client HI: 8a7a47db44c6582f0e1f995d55fe5eddff0b9712445b6368e1a55f6
       0381b4cb7
   Client Endorsement by CA( 136.0  bytes): 67c688d0699e81d02001003ff
       e3ff80560ac736574d2c4668a7a47db44c6582f0e1f995d55fe5eddff0b971
       2445b6368e1a55f60381b4cb72001003ffe3ff8056dcf2c1a98a46c4249c43
       a994531b3562159583e53592032acda4e400ddda58ba303b5a2a2f4d248756
       d314bd9fa7d1dc379714316f47ef5448f5b2495e5a49e160ec104dd5f0408
   client SN: x1224AABBCCDDEE56789
```

Figure 6: Test DKI DETs and Endorsements

A.1. Test DNS

The DNS tree(s) for the above test data is still in limbo and will be added in a later version of this draft with the proposed DET RR from [RFC9886].

Appendix B. Test X.509 certificates

B.1. Test Lite X.509 certificates

The following the test DRIP X.509 certificates that mirror the test Endorsements.

raa16376.pem (der is 300 bytes)

```
-----BEGIN CERTIFICATE-----
MIIBKDCB26ADAgECAgJltTAFBgMrZXAwKzEpMCcGA1UEAwwgMjAwMTAwM2ZmZTAw
MDAwNWY4ODVjOGVlNmFkMmE3YWYwHhcNMjUwMzAxMDAwMTAwWhcNMjcwMzAxMjM1
OTAwWjAbMRkwFwYDVQQDDBBEUklQLVJBQS1BLTE2Mzc2MCowBQYDK2VwAyEAkilT
nyrmqWHRwkl3RV2pgWLlPvyY356zD3JTdpk6cnWjMzAxMA8GA1UdEwEB/wQFMAMB
Af8wHgYDVR0RAQH/BBQwEocQIAEAP/4AAAX4hcjuatKnrzAFBgMrZXADQQDF/Nbb
cqZ0Ij96JWnru7APhPw1g+Ozfg2AvA02EbKX6J807TkfdOaar9jXdCYg5ekGVF1e
lir13hHReY0U1AAO
-----END CERTIFICATE-----
```

```
Certificate: 461 bytes
    Data:
        Version: 3 (0x2)
        Serial Number: 26037 (0x65b5)
        Signature Algorithm: ED25519
        Issuer: CN=2001003ffe000005f885c8ee6ad2a7af
        Validity
            Not Before: Mar  1 00:01:00 2025 GMT
            Not After : Mar  1 23:59:00 2027 GMT
        Subject: CN=DRIP-RAA-A-16376
        Subject Public Key Info:
            Public Key Algorithm: ED25519
                ED25519 Public-Key:
                pub:
                    92:29:53:9f:2a:e6:a9:61:d1:c2:49:77:45:5d:a9:
                    81:62:e5:3e:fc:98:df:9e:b3:0f:72:53:76:99:3a:
                    72:75
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Subject Alternative Name: critical
                IP Address:2001:3F:FE00:5:F885:C8EE:6AD2:A7AF
    Signature Algorithm: ED25519
    Signature Value:
        c5:fc:d6:db:72:a6:74:22:3f:7a:25:69:eb:bb:b0:0f:84:fc:
        35:83:e3:b3:7e:0d:80:bc:0d:36:11:b2:97:e8:9f:34:ed:39:
        1f:74:e6:9a:af:d8:d7:74:26:20:e5:e9:06:54:5d:5e:96:2a:
        f5:de:11:d1:79:8d:14:d4:00:0e
```

Authentication hda16376-16376A.pem (der is 306 bytes)

```
-----BEGIN CERTIFICATE-----
MIIBLjCB4aADAgECAgJ4bDAFBgMrZXAwKzEpMCcGA1UEAwwgMjAwMTAwM2ZmZTAw
MDAwNWY4ODVjOGVlNmFkMmE3YWYwHhcNMjUwMzAyMDAwMTAwWhcNMjYwMzMwMjM1
OTAwWjAhMR8wHQYDVQQDDBZEUklQLUhEQS1BLTE2Mzc2LTE2Mzc2MCowBQYDK2Vw
AyEAuCsn+GsBNGj+SNhbVPAb9lOF8wKrLhNtxRo7kpyIzlqjMzAxMA8GA1UdEwEB
/wQFMAMBAf8wHgYDVR0RAQH/BBQwEocQIAEAP/4AAAUFys+hHngL1TAFBgMrZXAD
QQBb26M21AgqKPYFjZlXhDzqnYMHn5cUSPOHeeEb0vJDb5u7OxxcJ/zIDbdN1lLr
KBhMSTmcvLBCKEimOuBrmqEC
-----END CERTIFICATE-----

Certificate:   469 bytes
    Data:
        Version: 3 (0x2)
        Serial Number: 30828 (0x786c)
        Signature Algorithm: ED25519
        Issuer: CN=2001003ffe000005f885c8ee6ad2a7af
        Validity
            Not Before: Mar  2 00:01:00 2025 GMT
            Not After : Mar 30 23:59:00 2026 GMT
        Subject: CN=DRIP-HDA-A-16376-16376
        Subject Public Key Info:
            Public Key Algorithm: ED25519
                ED25519 Public-Key:
                pub:
                    b8:2b:27:f8:6b:01:34:68:fe:48:d8:5b:54:f0:1b:
                    f6:53:85:f3:02:ab:2e:13:6d:c5:1a:3b:92:9c:88:
                    ce:5a
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Subject Alternative Name: critical
                IP Address:2001:3F:FE00:5:5CA:CFA1:1E78:BD5
    Signature Algorithm: ED25519
    Signature Value:
        5b:db:a3:36:d4:08:2a:28:f6:05:8d:99:57:84:3c:ea:9d:83:
        07:9f:97:14:48:f3:87:79:e1:1b:d2:f2:43:6f:9b:bb:3b:1c:
        5c:27:fc:c8:0d:b7:4d:d6:52:eb:28:18:4c:49:39:9c:bc:b0:
        42:28:48:a6:3a:e0:6b:9a:a1:02
```

```
     Issuing hda16376-16376I.pem (der is 306 bytes)

   -----BEGIN CERTIFICATE-----
   MIIBLjCB4aADAgECAgJEwjAFBgMrZXAwKzEpMCcGA1UEAwwgMjAwMTAwM2ZmZTNm
   ZjgwNTIzNGZhNGFmY2MyMmI1YjQwHhcNMjUwMzAyMDAwMTAwWhcNMjYwMjI3MjM1
   OTAwWjAhMR8wHQYDVQQDDBZEUklQLUhEQS1JLTE2Mzc2LTE2Mzc2MCowBQYDK2Vw
   AyEAzHXXXfd4c00uW2gvb/k4q/EKECb3iNypmUXL3azz1yOjMzAxMA8GA1UdEwEB
   /wQFMAMBAf8wHgYDVR0RAQH/BBQwEocQIAEAP/4/+AVtzywamKRsQjAFBgMrZXAD
   QQCVRoDFK8uZ9NzxcF+p42gfiBvBtA17b98VMYtKMJ7I80BddtxGWFha++x4IW7P
   d7oLtxv75oHYcGqNyuzt23kI
   -----END CERTIFICATE-----

   Certificate:    493 bytes
       Data:
           Version: 3 (0x2)
           Serial Number: 17602 (0x44c2)
           Signature Algorithm: ED25519
           Issuer: CN=2001003ffe3ff805234fa4afcc22b5b4
           Validity
               Not Before: Mar  2 00:01:00 2025 GMT
               Not After : Feb 27 23:59:00 2026 GMT
           Subject: CN=DRIP-HDA-I-16376-16376
           Subject Public Key Info:
               Public Key Algorithm: ED25519
                   ED25519 Public-Key:
                   pub:
                       cc:75:d7:5d:f7:78:73:4d:2e:5b:68:2f:6f:f9:38:
                       ab:f1:0a:10:26:f7:88:dc:a9:99:45:cb:dd:ac:f3:
                       d7:23
           X509v3 extensions:
               X509v3 Basic Constraints: critical
                   CA:TRUE
               X509v3 Subject Alternative Name: critical
                   IP Address:2001:3F:FE3F:F805:6DCF:2C1A:98A4:6C42
       Signature Algorithm: ED25519
       Signature Value:
           95:46:80:c5:2b:cb:99:f4:dc:f1:70:5f:a9:e3:68:1f:88:1b:
           c1:b4:0d:7b:6f:df:15:31:8b:4a:30:9e:c8:f3:40:5d:76:dc:
           46:58:58:5a:fb:ec:78:21:6e:cf:77:ba:0b:b7:1b:fb:e6:81:
           d8:70:6a:8d:ca:ec:ed:db:79:08
     ua1-16376-16376csr.pem  290 bytes

   Certificate Request:
       Data:
           Version: 1 (0x0)
           Subject: serialNumber=x1224AABBCCDDEE56789
           Subject Public Key Info:
               Public Key Algorithm: ED25519
```

```
                    ED25519 Public-Key:
                    pub:
                        8a:7a:47:db:44:c6:58:2f:0e:1f:99:5d:55:fe:5e:
                        dd:ff:0b:97:12:44:5b:63:68:e1:a5:5f:60:38:1b:
                        4c:b7
            Attributes:
                (none)
                Requested Extensions:
        Signature Algorithm: ED25519
        Signature Value:
            2b:73:a9:6a:e7:07:3c:95:b4:71:95:06:43:ee:fc:3d:27:88:
            54:46:68:42:76:c7:7b:e9:1b:4b:6e:e1:4a:37:be:5f:79:e2:
            b8:6d:60:75:ea:49:13:54:75:e6:47:6a:14:8d:90:52:e1:32:
            58:f1:06:29:f6:b1:7d:24:d7:05
```

ua1-16376-16376.pem (der is 256 bytes)

```
-----BEGIN CERTIFICATE-----
MIH9MIGwoAMCAQICAxMuRTAFBgMrZXAwKzEpMCcGA1UEAwwgMjAwMTAwM2ZmZTNm
ZjgwNTZkY2YyYzFhOThhNDZjNDIwHhcNMjUwMzA0MDAwMTAwWhcNMjYwMjI1MjM1
OTAwWjAAMCowBQYDK2VwAyEAinpH20TGWC8OH5ldVf5e3f8LlxJEW2No4aVfYDgb
TLejIjAgMB4GA1UdEQEB/wQUMBKHECABAD/+P/gFYKxzZXTSxGYwBQYDK2VwA0EA
tgNfDC2SyxjynnVchise4K3vSg7bDXg7hxoFn86Dcw3dOESaq7+N9cqO2PHOs9dP
y2vpKm5S54Vb+0XJocgWCA==
-----END CERTIFICATE-----
```

```
Certificate Request:  404 bytes
    Data:
        Version: 1 (0x0)
        Serial Number: 1257029 (0x132e45)
        Signature Algorithm: ED25519
        Issuer: CN=2001003ffe3ff8056dcf2c1a98a46c42
        Validity
            Not Before: Mar  4 00:01:00 2025 GMT
            Not After : Feb 25 23:59:00 2026 GMT
        Subject:
        Subject Public Key Info:
            Public Key Algorithm: ED25519
                ED25519 Public-Key:
                pub:
                    8a:7a:47:db:44:c6:58:2f:0e:1f:99:5d:55:fe:5e:
                    dd:ff:0b:97:12:44:5b:63:68:e1:a5:5f:60:38:1b:
                    4c:b7
        X509v3 extensions:
            X509v3 Subject Alternative Name: critical
                IP Address:2001:3F:FE3F:F805:60AC:7365:74D2:C466
    Signature Algorithm: ED25519
    Signature Value:
```

```
b6:03:5f:0c:2d:92:cb:18:f2:9e:75:5c:86:2b:1e:e0:ad:ef:
4a:0e:db:0d:78:3b:87:1a:05:9f:ce:83:73:0d:dd:38:44:9a:
ab:bf:8d:f5:ca:8e:d8:f1:ce:b3:d7:4f:cb:6b:e9:2a:6e:52:
e7:85:5b:fb:45:c9:a1:c8:16:08
```

Figure 7: Test Lite X.509 certificates

B.2. Test DRIP-Full X.509 certificates

The following the test DRIP DRIP-Full X.509 certificates that mirror the test Endorsements of prior drafts. These certificates are now generated by the endorse.py script. Thus the same CSRs are used in creating all the endorsement objects.

raa16376Full.pem (der is 350 bytes)

```
-----BEGIN CERTIFICATE-----
MIIBWjCCAQygAwIBAgICK1owBQYDK2VwMCsxKTAnBgNVBAMMIDIwMDEwMDNmZmUw
MDAwMDVmODg1YzhlZTZhZDJhN2FmMB4XDTI1MDMwMTAwMDEwMFoXDTI3MDMwMTIz
NTkwMFowGzEZMBcGA1UEAwwQRFJJUC1SQUEtQS0xNjM3NjAqMAUGAytlcAMhAJIp
U58q5qlh0cJJd0VdqYFi5T78mN+esw9yU3aZOnJ1o2QwYjAPBgNVHRMBAf8EBTAD
AQH/MBQGA1UdIAQNMAswCQYHKxsQAQEAATAeBgNVHREBAf8EFDAShxAgAQA//gAA
BfiFyO5q0qevMBkGA1UdDgQSBBAgAQA//gAABfiFyO5q0qevMAUGAytlcANBAMCs
CvrMjSK/UpCX7xN02ueRWSwpGQcPWLw3Wnr9MOrMzayEMXRcdXRh89VEa6LD3lmC
tDf7yBxt3L4Z+hLSkA4=
-----END CERTIFICATE-----

Certificate:
  Data:
      Version: 3 (0x2)
        Serial Number: 11098 (0x2b5a)
        Signature Algorithm: ED25519
        Issuer: CN=2001003ffe000005f885c8ee6ad2a7af
        Validity
            Not Before: Mar  1 00:01:00 2025 GMT
            Not After : Mar  1 23:59:00 2027 GMT
        Subject: CN=DRIP-RAA-A-16376
        Subject Public Key Info:
            Public Key Algorithm: ED25519
                ED25519 Public-Key:
                pub:
                    92:29:53:9f:2a:e6:a9:61:d1:c2:49:77:45:5d:a9:
                    81:62:e5:3e:fc:98:df:9e:b3:0f:72:53:76:99:3a:
                    72:75
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Certificate Policies:
                Policy: 1.3.27.16.1.1.0.1
            X509v3 Subject Alternative Name: critical
                IP Address:2001:3F:FE00:5:F885:C8EE:6AD2:A7AF
            X509v3 Subject Key Identifier:
                20:01:00:3F:FE:00:00:05:F8:85:C8:EE:6A:D2:A7:AF
    Signature Algorithm: ED25519
    Signature Value:
        c0:ac:0a:fa:cc:8d:22:bf:52:90:97:ef:13:74:da:e7:91:59:
        2c:29:19:07:0f:58:bc:37:5a:7a:fd:30:ea:cc:cd:ac:84:31:
        74:5c:75:74:61:f3:d5:44:6b:a2:c3:de:59:82:b4:37:fb:c8:
        1c:6d:dc:be:19:fa:12:d2:90:0e
```

```
     Authentication hda16376-16376AFull.pem (der is 386 bytes)

   -----BEGIN CERTIFICATE-----
   MIIBfjCCATCgAwIBAgICWRswBQYDK2VwMCsxKTAnBgNVBAMMIDIwMDEwMDNmZmUw
   MDAwMDVmODg1YzhlZTZhZDJhN2FmMB4XDTI1MDMwMjAwMDEwMFoXDTI2MDMzMDIz
   NTkwMFowITEfMB0GA1UEAwwWRFJJUC1IREEtQS0xNjM3Ni0xNjM3NjAqMAUGAytl
   cAMhALgrJ/hrATRo/kjYW1TwG/ZThfMCqy4TbcUaO5KciM5ao4GBMH8wDwYDVR0T
   AQH/BAUwAwEB/zAUBgNVHSAEDTALMAkGBysbEAEBAAEwHgYDVR0RAQH/BBQwEocQ
   IAEAP/4AAAUFys+hHngL1TAZBgNVHQ4EEgQQIAEAP/4AAAUFys+hHngL1TAbBgNV
   HSMEFDASgBAgAQA//gAABfiFyO5q0qevMAUGAytlcANBAAGTWI4Iuq7LLjZeMboU
   HGM+Fxn240wny6pkhWjVreWLF86IGO8nOWT7e6Z3mi2ZHHsFZ+JZYDgXhzpJ0KbX
   JgY=
   -----END CERTIFICATE-----

   Certificate:
     Data:
         Version: 3 (0x2)
           Serial Number: 22811 (0x591b)
           Signature Algorithm: ED25519
           Issuer: CN=2001003ffe000005f885c8ee6ad2a7af
           Validity
               Not Before: Mar  2 00:01:00 2025 GMT
               Not After : Mar 30 23:59:00 2026 GMT
           Subject: CN=DRIP-HDA-A-16376-16376
           Subject Public Key Info:
               Public Key Algorithm: ED25519
                   ED25519 Public-Key:
                   pub:
                       b8:2b:27:f8:6b:01:34:68:fe:48:d8:5b:54:f0:1b:
                       f6:53:85:f3:02:ab:2e:13:6d:c5:1a:3b:92:9c:88:
                       ce:5a
           X509v3 extensions:
               X509v3 Basic Constraints: critical
                   CA:TRUE
               X509v3 Certificate Policies:
                   Policy: 1.3.27.16.1.1.0.1
               X509v3 Subject Alternative Name: critical
                   IP Address:2001:3F:FE00:5:5CA:CFA1:1E78:BD5
               X509v3 Subject Key Identifier:
                   20:01:00:3F:FE:00:00:05:05:CA:CF:A1:1E:78:0B:D5
               X509v3 Authority Key Identifier:
                   20:01:00:3F:FE:00:00:05:F8:85:C8:EE:6A:D2:A7:AF
       Signature Algorithm: ED25519
       Signature Value:
           01:93:58:8e:08:ba:ae:cb:2e:36:5e:31:ba:14:1c:63:3e:17:
           19:f6:e3:4c:27:cb:aa:64:85:68:d5:ad:e5:8b:17:ce:88:18:
           ef:27:39:64:fb:7b:a6:77:9a:2d:99:1c:7b:05:67:e2:59:60:
           38:17:87:3a:49:d0:a6:d7:26:06
```

```
  Issuing hda16376-16376IFull.pem (der is 386 bytes)

-----BEGIN CERTIFICATE-----
MIIBfjCCATCgAwIBAgICW+4wBQYDK2VwMCsxKTAnBgNVBAMMIDIwMDEwMDNmZmUz
ZmY4MDUyMzRmYTRhZmNjMjJiNWI0MB4XDTI1MDMwMjAwMDEwMFoXDTI2MDIyNzIz
NTkwMFowITEfMB0GA1UEAwwWRFJJUC1IREEtSS0xNjM3Ni0xNjM3NjAqMAUGAytl
cAMhAMx11133eHNNLltoL2/5OKvxChAm94jcqZlFy92s89cjo4GBMH8wDwYDVR0T
AQH/BAUwAwEB/zAUBgNVHSAEDTALMAkGBysbEAEBAAIwHgYDVR0RAQH/BBQwEocQ
IAEAP/4/+AVtzywamKRsQjAZBgNVHQ4EEgQQIAEAP/4/+AVtzywamKRsQjAbBgNV
HSMEFDASgBAgAQA//j/4BSNPpK/MIrW0MAUGAytlcANBAB0Mcico1RPhPqYLDJeZ
oDnUY/U/eqQH3/IIEyTVvB1ZS30CAduTCFVq1B7G0VD3HGHvdPenj0vGpaak+OUw
0gQ=
-----END CERTIFICATE-----

Certificate:
  Data:
      Version: 3 (0x2)
        Serial Number: 23534 (0x5bee)
        Signature Algorithm: ED25519
        Issuer: CN=2001003ffe3ff805234fa4afcc22b5b4
        Validity
            Not Before: Mar  2 00:01:00 2025 GMT
            Not After : Feb 27 23:59:00 2026 GMT
        Subject: CN=DRIP-HDA-I-16376-16376
        Subject Public Key Info:
            Public Key Algorithm: ED25519
                ED25519 Public-Key:
                pub:
                    cc:75:d7:5d:f7:78:73:4d:2e:5b:68:2f:6f:f9:38:
                    ab:f1:0a:10:26:f7:88:dc:a9:99:45:cb:dd:ac:f3:
                    d7:23
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Certificate Policies:
                Policy: 1.3.27.16.1.1.0.2
            X509v3 Subject Alternative Name: critical
                IP Address:2001:3F:FE3F:F805:6DCF:2C1A:98A4:6C42
            X509v3 Subject Key Identifier:
                20:01:00:3F:FE:3F:F8:05:6D:CF:2C:1A:98:A4:6C:42
            X509v3 Authority Key Identifier:
                20:01:00:3F:FE:3F:F8:05:23:4F:A4:AF:CC:22:B5:B4
    Signature Algorithm: ED25519
    Signature Value:
        1d:0c:72:27:28:d5:13:e1:3e:a6:0b:0c:97:99:a0:39:d4:63:
        f5:3f:7a:a4:07:df:f2:08:13:24:d5:bc:1d:59:4b:7d:02:01:
        db:93:08:55:6a:d4:1e:c6:d1:50:f7:1c:61:ef:74:f7:a7:8f:
        4b:c6:a5:a6:a4:f8:e5:30:d2:04
```

ua1-16376-16376Full.pem (der is 286 bytes)

```
-----BEGIN CERTIFICATE-----
MIIBGjCBzaADAgECAgMpQEAwBQYDK2VwMCsxKTAnBgNVBAMMIDIwMDEwMDNmZmUz
ZmY4MDU2ZGNmMmMxYTk4YTQ2YzQyMB4XDTI1MDMwNDAwMDEwMFoXDTI2MDIyNTIz
NTkwMFowADAqMAUGAytlcAMhAIp6R9tExlgvDh+ZXVX+Xt3/C5cSRFtjaOGlX2A4
G0y3oz8wPTAeBgNVHREBAf8EFDAShxAgAQA//j/4BWCsc2V00sRmMBsGA1UdIwQU
MBKAECABAD/+P/gFbc8sGpikbEIwBQYDK2VwA0EAOpsCcwdgW8A8PysqQxZoZulw
e2OnUpjPUeBx29A/EjRuGOmI1rMzPfFSuTV9VJ+hfUUhc9UgdnTfoF0CPEwSBw==
-----END CERTIFICATE-----

Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 2703424 (0x294040)
        Signature Algorithm: ED25519
        Issuer: CN=2001003ffe3ff8056dcf2c1a98a46c42
        Validity
            Not Before: Mar  4 00:01:00 2025 GMT
            Not After : Feb 25 23:59:00 2026 GMT
        Subject:
        Subject Public Key Info:
            Public Key Algorithm: ED25519
                ED25519 Public-Key:
                pub:
                    8a:7a:47:db:44:c6:58:2f:0e:1f:99:5d:55:fe:5e:
                    dd:ff:0b:97:12:44:5b:63:68:e1:a5:5f:60:38:1b:
                    4c:b7
        X509v3 extensions:
            X509v3 Subject Alternative Name: critical
                IP Address:2001:3F:FE3F:F805:60AC:7365:74D2:C466
            X509v3 Authority Key Identifier:
                20:01:00:3F:FE:3F:F8:05:6D:CF:2C:1A:98:A4:6C:42
    Signature Algorithm: ED25519
    Signature Value:
        3a:9b:02:73:07:60:5b:c0:3c:3f:2b:2a:43:16:68:66:e9:70:
        7b:63:a7:52:98:cf:51:e0:71:db:d0:3f:12:34:6e:18:e9:88:
        d6:b3:33:3d:f1:52:b9:35:7d:54:9f:a1:7d:45:21:73:d5:20:
        76:74:df:a0:5d:02:3c:4c:12:07
```

Figure 8: Test DRIP-Full X.509 certificates

B.3. Test Lite C509 certificates

The CBOR Encoded X.509 Certificates (C509 Certificates) [C509-Certificates] provides a standards-based approach to reduce the size of X.509 certificates both on-the-wire and in storage.

These C509 certificates MAY be stored in the DET RR, but are more likely to by used in over-the-air protocols and exist only for transmission, being converted from/to their source X.509 certificates.

Test Rust code for X.509 to c509 conversion is available at [cbor_c509_code]. The CBOR hex was converted to readable format at http://cbor.me.

Author's Note: This section is still a Work in Progress. Note that we think there is a bug in the c509 code, making the certificate version = 1, not 3.

The following are examples of a C509 cert.

raa16376.cert CBOR:

COSE_X509 (191 bytes)

```
8B 01 42 65 B5 78 20 32 30 30 31 30 30 33 66 66 65 30 30 30 30 30 35
66 38 38 35 63 38 65 65 36 61 64 32 61 37 61 66 1A 67 C2 4E 3C 1A 6B
86 06 44 70 44 52 49 50 2D 52 41 41 2D 41 2D 31 36 33 37 36 0A 58 20
92 29 53 9F 2A E6 A9 61 D1 C2 49 77 45 5D A9 81 62 E5 3E FC 98 DF 9E
B3 0F 72 53 76 99 3A 72 75 84 23 20 22 82 07 50 20 01 00 3F FE 00 00
05 F8 85 C8 EE 6A D2 A7 AF 0C 58 40 C5 FC D6 DB 72 A6 74 22 3F 7A 25
69 EB BB B0 0F 84 FC 35 83 E3 B3 7E 0D 80 BC 0D 36 11 B2 97 E8 9F 34
ED 39 1F 74 E6 9A AF D8 D7 74 26 20 E5 E9 06 54 5D 5E 96 2A F5 DE 11
D1 79 8D 14 D4 00 0E
```

```
[1, h'65B5', "2001003ffe000005f885c8ee6ad2a7af",
 1740787260, 1803945540, "DRIP-RAA-A-16376", 10,
 h'9229539F2AE6A961D1C24977455DA98162E53EFC98DF9EB30F725376993A7275',
 [-4, -1, -3, [7, h'2001003FFE000005F885C8EE6AD2A7AF']], 12,
 h'C5FCD6DB72A674223F7A2569EBBBB00F84FC3583E3B37E0D80BC0D3611B297E
 89F34ED391F74E69AAFD8D7742620E5E906545D5E962AF5DE11D1798D14D4000E']
```

ua1-16376-16376 CBOR:

```
COSE_X509 (174 bytes)
8B 01 43 13 2E 45 78 20 32 30 30 31 30 30 33 66 66 65 33 66 66 38 30
35 36 64 63 66 32 63 31 61 39 38 61 34 36 63 34 32 1A 67 C6 42 BC 1A
69 9F 8C C4 80 0A 58 20 8A 7A 47 DB 44 C6 58 2F 0E 1F 99 5D 55 FE 5E
DD FF 0B 97 12 44 5B 63 68 E1 A5 5F 60 38 1B 4C B7 82 22 82 07 50 20
01 00 3F FE 3F F8 05 60 AC 73 65 74 D2 C4 66 0C 58 40 B6 03 5F 0C 2D
92 CB 18 F2 9E 75 5C 86 2B 1E E0 AD EF 4A 0E DB 0D 78 3B 87 1A 05 9F
CE 83 73 0D DD 38 44 9A AB BF 8D F5 CA 8E D8 F1 CE B3 D7 4F CB 6B E9
2A 6E 52 E7 85 5B FB 45 C9 A1 C8 16 08

[1, h'132E45', "2001003ffe3ff8056dcf2c1a98a46c42", 1741046460,
 1772063940, [], 10,
 h'8A7A47DB44C6582F0E1F995D55FE5EDDFF0B9712445B6368E1A55F60381B4CB7',
 [-3, [7, h'2001003FFE3FF80560AC736574D2C466']], 12,
 h'B6035F0C2D92CB18F29E755C862B1EE0ADEF4A0EDB0D783B871A059FCE8373
 0DDD38449AABBF8DF5CA8ED8F1CEB3D74FCB6BE92A6E52E7855BFB45C9A1C81608']
```

ua1-16376-16376Full CBOR:

```
COSE_X509 (192 bytes)
8B 01 43 29 40 40 78 20 32 30 30 31 30 30 33 66 66 65 33 66 66 38 30
35 36 64 63 66 32 63 31 61 39 38 61 34 36 63 34 32 1A 67 C6 42 BC 1A
69 9F 8C C4 80 0A 58 20 8A 7A 47 DB 44 C6 58 2F 0E 1F 99 5D 55 FE 5E
DD FF 0B 97 12 44 5B 63 68 E1 A5 5F 60 38 1B 4C B7 84 22 82 07 50 20
01 00 3F FE 3F F8 05 60 AC 73 65 74 D2 C4 66 07 50 20 01 00 3F FE 3F
F8 05 6D CF 2C 1A 98 A4 6C 42 0C 58 40 3A 9B 02 73 07 60 5B C0 3C 3F
2B 2A 43 16 68 66 E9 70 7B 63 A7 52 98 CF 51 E0 71 DB D0 3F 12 34 6E
18 E9 88 D6 B3 33 3D F1 52 B9 35 7D 54 9F A1 7D 45 21 73 D5 20 76 74
DF A0 5D 02 3C 4C 12 07

[1, h'294040', "2001003ffe3ff8056dcf2c1a98a46c42", 1741046460,
 1772063940, [], 10,
 h'8A7A47DB44C6582F0E1F995D55FE5EDDFF0B9712445B6368E1A55F60381B4CB7',
 [-3, [7, h'2001003FFE3FF80560AC736574D2C466'], 7,
 h'2001003FFE3FF8056DCF2C1A98A46C42'], 12,
 h'3A9B027307605BC03C3F2B2A43166866E9707B63A75298CF51E071DBD03F123
 46E18E988D6B3333DF152B9357D549FA17D452173D5207674DFA05D023C4C1207']
```

Figure 9: Test Lite C.509 certificates

Acknowledgments

Many people assisted in creating the python scripts for making DETs and DRIP Endorsements. Any roughness in the scripts is all my doing.

The COSE C509 authors are providing active help in creating the C509 equivalent objects.

Authors' Addresses

Robert Moskowitz
HTT Consulting
Oak Park, MI 48237
United States of America
Email: rgm@labs.htt-consult.com

Stuart W. Card
AX Enterprize, LLC
4947 Commercial Drive
Yorkville, NY 13495
United States of America
Email: stu.card@axenterprize.com