High Assurance Verifiable Identifiers (HAVID) Specification

Draft v0.1, 1st October 2024

Abstract

This specification details a standardized way of bridging different Verifiable Identifier (VID) types, including but not limited to, X.509 Certificates, Decentralized Identifiers (DIDs) and Domain Name Service (DNS) records. This bridge will be comprised of two main components:

1. **A Cryptographic bridge:** The assertion of control of a shared key pair between two or more identifiers, enabling cryptographic verifiability.
2. **A Non-Cryptographic bridge**: A mutual reference where each identifier explicitly points to the other, ensuring a bi-directional linkage.

By combining these bridges, verifiers and users can correlate identifiers across ecosystems with a high degree of assurance by validating both their referential and cryptographic integrity.

While cryptographic and non-cryptographic bridges can be established in various ways depending on the VID type, standardized methods must be provided for each VID type to ensure the pattern remains practical and usable for implementers. As such, this specification currently focuses on the bridging of DIDs, DNS domains, and X509 certificates.

Status of this document

In progress

Table of Contents

In an era where digital identity systems are rapidly evolving, the need for interoperability between disparate identifier ecosystems has never been greater. This specification addresses that need by standardizing the method for bridging various Verifiable Identifier (VID) types—ranging from traditional X.509 certificates and DNS records to more recent Decentralized Identifiers (DIDs). At the heart of this approach are two complementary mechanisms: a cryptographic bridge that establishes mutual control over a shared key pair, and a non-cryptographic bridge that ensures each identifier explicitly references the other. Together, these bridges offer verifiers and users a straightforward procedure for confirming

that two identifiers are related—achieved by validating the integrity of their mutual references alongside their shared ownership of a keypair.

The specification therefore is intended to provide patterns and implementation guidance for how the controller of a verifiable identifier (VID) can enable greater assurance in the integrity of that VID through mechanisms that enable the controller to bridge across different VID types.

This “bridged approach” to VIDs will lay the foundation for High Assurance (HA) VID Profiles that can work with many different types of VIDs. This approach allows decision-makers to choose the most suitable VID for specific contexts and purposes without foregoing interoperability between VID systems. Below is one example of how such an HA VID identifying an example company could be deployed based on a DID, a DNSSEC record (following the High Assurance DIDs with DNS RFC), and an X.509 digital certificate containing an LEI (following ISO 17442 Part 2).

1.1. Conformance

In addition to sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative. The key words MAY, MUST, MUST NOT, OPTIONAL, and SHOULD 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. Terminology

- Decentralized Identifier (DID): As defined in [DID-CORE].
- DID controller: As defined in [DID-CORE].
- DID document: As defined in [DID-CORE].
- DID URL: As defined in [DID-CORE].
- DID URL Dereferencing: As defined in [DID-CORE].
- X.509 Certificate: As defined in [Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile]

3. High Assurance VIDs

This section explains the context for building High Assurance VIDs, as well as their fundamental components for construction.

3.1. High Assurance VID Context

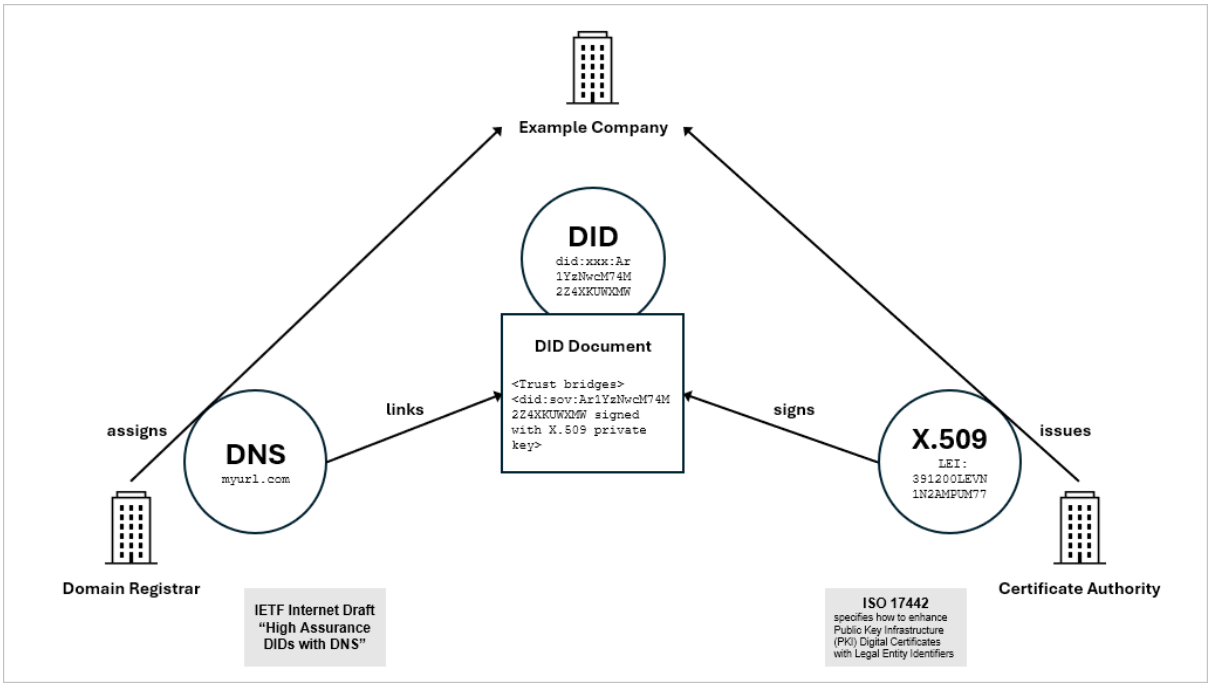
Since the development of the Internet, there have been two separate and distinct methods of encapsulating claims backed by the usage of a tightly-controlled private key matched with a publicly discoverable public key to verify the private key signer and prove immutability of the message. Let's call the camps x.509 and VCs.

In the X.509 camp, the ability to make claims come from trusted anchor organizations, public Certification Authorities (CAs) which endorse signing capabilities and perpetrate their trust by being registered on Trust Registries managed by Internet browser companies. The identifier used within these IETF-based claims is asserted to be unique to the issuer (https://csrc.nist.gov/glossary/term/x_509_public_key_certificate) but not to a global scheme.

In the VC camp, claims are encapsulated in containers promulgated by the W3C Verifiable Credentials Standard (https://www.w3.org/TR/vc-data-model-2.0/) using a few globally recognizable identifiers (https://www.w3.org/TR/vc-data-model-2.0/#identifiers). The trust infrastructure depends not on CAs and Browsers, but on a "trust diamond" (issuers, Holders, Verifiers and Governance Authorities) that participate and rely upon a network backed by the Internet Domain Name Service for market acceptance.

The purpose of this spec is to marry the two approaches and allow the best aspects of the existing infrastructure and roll-out of Verifiable Identifiers (VIDs) for maximum market acceptance and high confidence of the binding quality of identifiers to its subjects.

4. Architecture Overview



The bridging pattern described in this specification generalizes across different VID types by leveraging two complementary mechanisms: **cryptographic bridges** and

non-cryptographic bridges. These bridges function together to establish both **a verifiable assertion of shared control** and **a semantic reference between identifiers**, allowing entities to correlate identifiers across ecosystems with a high degree of assurance.

An example of a non-cryptographic bridge… talk about LEIs and X.509

4.1. Generalizing the Bridging Pattern

At a high level, this pattern can be applied to any ecosystem where identifiers exist in different trust frameworks but need to be linked in a verifiable way. The core principle is that an entity (or controller) should be able to:

- **Prove control over multiple identifiers cryptographically** by demonstrating possession of the same cryptographic key material across different VID types.
- **Establish an explicit reference between identifiers** through metadata or records stored within each identifier's respective system.
- **Enable verifiers to confirm both types of links** with minimal additional trust assumptions.

By applying this model to different VID types—such as Decentralized Identifiers (DIDs), X.509 certificates, and DNS records—the specification ensures interoperability while maintaining security and trustworthiness.

4.2. Context of Each Bridge Type

Each type of bridge serves a distinct role in verifying the relationship between identifiers:

4.2.1. Cryptographic Bridge

- **Purpose:** Establishes verifiable control over a shared key pair across different VID types.
- **Mechanism:** The same cryptographic key (or a cryptographically provable linkage between keys) is used to sign proofs, certificates, or metadata records across multiple systems.
- **Implementation Considerations:**
    - Ensuring that key material remains secure and is not exposed across weaker security domains.
    - Handling key rotation and expiration across different VID types.
    - Cryptographic algorithm compatibility between ecosystems.

4.2.2. Non-Cryptographic Bridge

- **Purpose**: Provides an explicit reference between identifiers, creating a bidirectional association that can be independently verified.
- **Mechanism**: Each identifier system must support metadata fields or records where references to the other identifier can be stored (e.g., DID documents referencing DNS names, X.509 extensions embedding DIDs, etc.).
- **Implementation Considerations**:

- Ensuring that metadata storage is persistent and accessible.
- Preventing tampering or unauthorized modification of references.
- Addressing governance issues, such as who is authorized to create and update references.

By standardizing the way cryptographic and non-cryptographic bridges are established, this specification provides a robust foundation for linking identifiers across different ecosystems. These bridges enable greater interoperability while maintaining high assurance in both the cryptographic and referential integrity of linked VIDs. Future extensions may further refine these bridging mechanisms to accommodate additional VID types and evolving security considerations.

5. High Assurance Bridges

The following 3 sections describe the implementation guidelines for the establishment of High Assurance Bridges between 3 main VID ecosystems:

1. DIDs [https://www.w3.org/TR/did-core/],
2. DNS [https://datatracker.ietf.org/doc/html/rfc1035]
3. x509 certificates [https://datatracker.ietf.org/doc/html/rfc5280].

Each entry comprises a detailed description of the technical implementation of the bridge originating from each identifier type (i.e DID to DNS and DNS to DID) as well as any context and additional considerations required to establish this construct.

6. Bridging DIDs and X.509

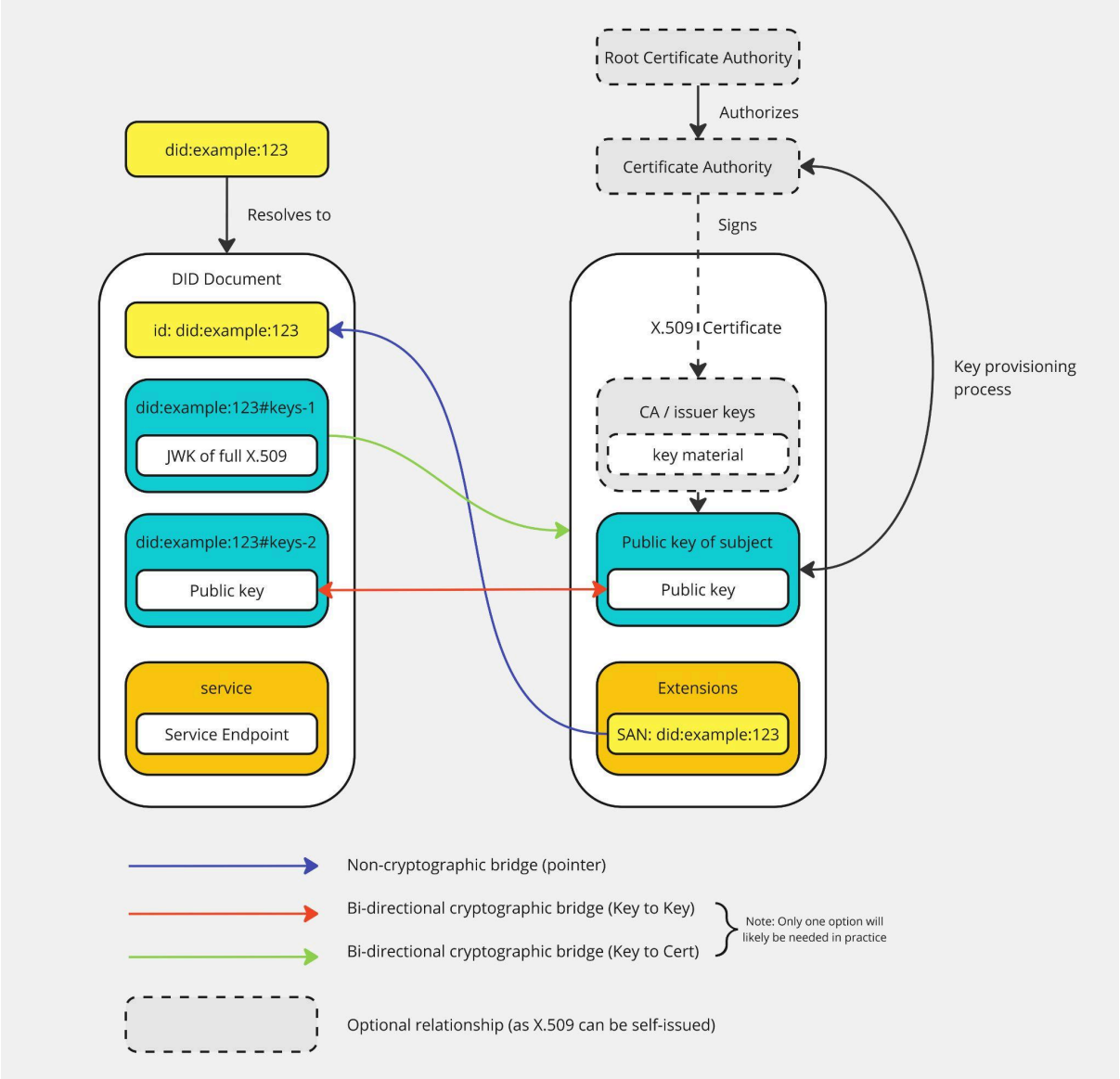
Public Key Infrastructure composed of x.509 certificates and Decentralized Identifiers (DIDs) both dot the landscape and have various comparisons. This is because DIDs and X.509 certificates are both inherently representations of public/private keypairs, described by a common structure and set of associated metadata.

However, there are nuanced differences in the general structure of both identifier types; where, crucially, it becomes difficult to align both identifier types at both a technical and governance level.

The primary difference is that X.509 certificates are primarily **hierarchical**, meaning that the recipient of the public/private key pair has their certificate issued from a Certificate Authority, where often the Certificate Authority acts an attestor to the contents and changes to the X.509 certificate itself. While it is possible to self sign and issue certificates, this usage is much less common in practice. On the other hand, DIDs **MAY** be controlled via a hierarchical structure, but are often solely controlled by one party with a public/private key pair, who must self-attest to these issuances or changes themself.

This creates a level of dissonance between the general usage of both identifier types, since an ecosystem of X.509 certificates operates in a vertical chain-like tree, whereas a DID is more flexible to operate in a singular, heterarchical structure or a hierarchical structure.

As such, this specification will not only point to the **technical** ways that an X.509 and a DID can be bridged, but will provide an acknowledgement of the **different governance relationships** that can be created via this technical bridge.



6.1. Identity Relationship Anchor (Non-Cryptographic Bridge)

6.1.1. DID to X.509

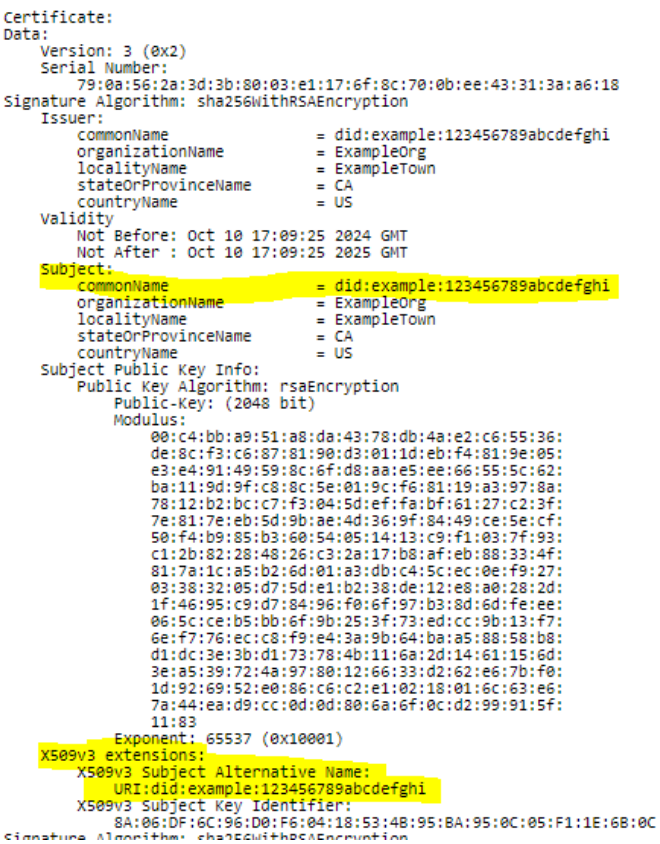
The establishment of a pointer from a DID to an X.509 is accomplished by the direct inclusion of the X.509 in one of the DID's verificationMethods, such that the public key represented by the verificationMethod is also the public key represented by the X.509. The verificationMethod MUST be of type "publicKeyJwk", and the X.509 must be present in either the "x5u" parameter as a URI pointing to a resource where the X.509 can be retrieved, or via the "x5c" parameters as an individual certificate or complete certificate chain.

Example DIDs which have x509 Certificates in their DID documents:

DID (click to resolve)	Comment
did:web:danubetech.com:did:test4	Uses 'x509CertificateChain'
did:web:danubetech.com:did:test4-jwk	Uses 'x5c'
did:web:danubetech.com:did:test5	Uses 'x509CertificateChain'
did:web:danubetech.com:did:test5-jwk	Uses 'x5c'
did:web:danubetech.com:did:test6	Uses 'x509CertificateChain'
did:web:danubetech.com:did:test6-jwk	Uses 'x5c'

6.1.2. X.509 to DID

The establishment of a pointer from a X.509 to a DID is accomplished via the direct inclusion of the DID's ID in the SubjectAltName (SAN) extension of the X.509 certificate.



```
Certificate:
Data:
    Version: 3 (0x2)
    Serial Number:
        79:0a:56:2a:3d:3b:80:03:e1:17:6f:8c:70:0b:ee:43:31:3a:a6:18
Signature Algorithm: sha256WithRSAEncryption
    Issuer:
        commonName               = did:example:123456789abcdefghi
        organizationName         = ExampleOrg
        localityName             = ExampleTown
        stateOrProvinceName      = CA
        countryName              = US
    Validity
        Not Before: Oct 10 17:09:25 2024 GMT
        Not After : Oct 10 17:09:25 2025 GMT
    Subject:
        commonName               = did:example:123456789abcdefghi
        organizationName         = ExampleOrg
        localityName             = ExampleTown
        stateOrProvinceName      = CA
        countryName              = US
    Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
            Public-Key: (2048 bit)
            Modulus:
                00:c4:bb:a9:51:a8:da:43:78:db:4a:e2:c6:55:36:
                de:8c:f3:c6:87:81:90:d3:01:1d:eb:f4:81:9e:05:
                e3:e4:91:49:59:8c:6f:d8:aa:e5:ee:66:55:5c:62:
                ba:11:9d:9f:c8:8c:5e:01:9c:f6:81:19:a3:97:8a:
                78:12:b2:bc:c7:f3:04:5d:ef:fa:bf:61:27:c2:3f:
                7e:81:7e:eb:5d:9b:ae:4d:36:9f:84:49:ce:5e:cf:
                50:f4:b9:85:b3:60:54:05:14:13:c9:f1:03:7f:93:
                c1:2b:82:28:48:26:c3:2a:17:b8:af:eb:88:33:4f:
                81:7a:1c:a5:b2:6d:01:a3:db:c4:5c:ec:0e:f9:27:
                03:38:32:05:d7:5d:e1:b2:38:de:12:e8:a0:28:2d:
                1f:46:95:c9:d7:84:96:f0:6f:97:b3:8d:6d:fe:ee:
                06:5c:ce:b5:bb:6f:9b:25:3f:73:ed:cc:9b:13:f7:
                6e:f7:76:ec:c8:f9:e4:3a:9b:64:ba:a5:88:58:b8:
                d1:dc:3e:3b:d1:73:78:4b:11:6a:2d:14:61:15:6d:
                3e:a5:39:72:4a:97:80:12:66:33:d2:62:e6:7b:f0:
                1d:92:69:52:e0:86:c6:c2:e1:02:18:01:6c:63:e6:
                7a:44:ea:d9:cc:0d:0d:80:6a:6f:0c:d2:99:91:5f:
                11:83
            Exponent: 65537 (0x10001)
    X509v3 extensions:
        X509v3 Subject Alternative Name:
            URI:did:example:123456789abcdefghi
        X509v3 Subject Key Identifier:
            8A:06:DF:6C:96:D0:F6:04:18:53:4B:95:BA:95:0C:05:F1:1E:6B:0C
Signature Algorithm: sha256WithRSAEncryption
```

6.2. Cross-Domain Trust Anchor (Cryptographic Bridge)

The establishment of a cryptographic bridge between an X.509 and DID is not accomplished via a referential value like the Identity Relationship Anchors described above, but via the presence of the X.509 in the DID document and subsequent interaction and verification of its related verificationMethod, much like the traditional usage of X.509s in TLS. The cryptographic bridge between an X.509 and DID is verifiable by demonstrating proof of possession of the corresponding private key. This can be achieved by signing a challenge or a predefined message using the private key associated with the X.509 certificate and then verifying the signature using the public key referenced in verificationMethod, among a variety of alternative methods and protocols.

6.3. **Additional Considerations**

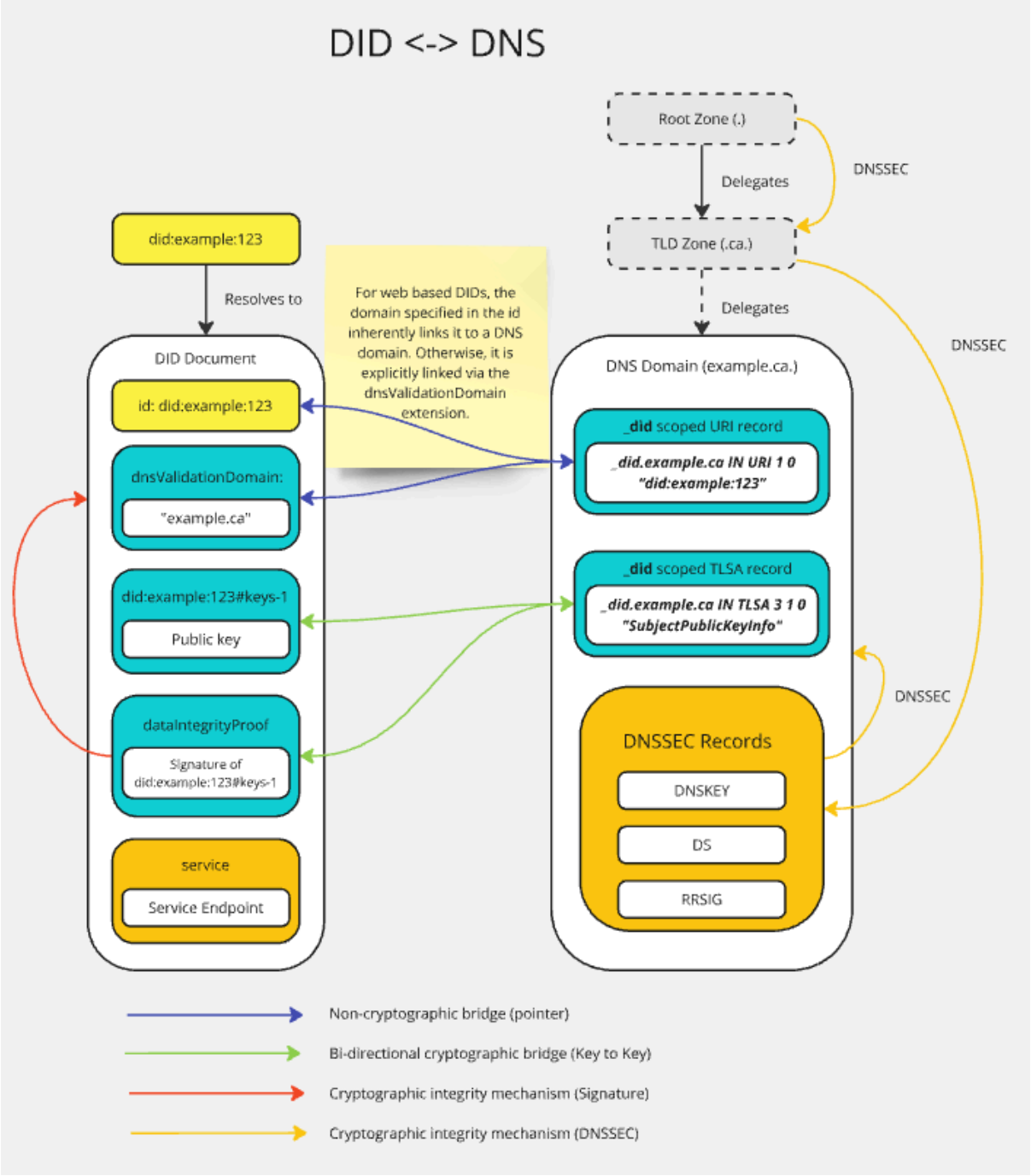
6.3.1. Hierarchical-governed bridge (Issuing CA is also a controller of the DID Doc)

6.3.2. Issuing CA also creates the DID at same time as X.509

6.3.3. CA Issuing Guidance (Verification of the DID, rekeying the cert etc.)

7. Bridging DIDs and DNS

DIDs and the DNS are two fundamentally different yet complementary systems for managing digital identifiers. While DIDs are self-sovereign identifiers that can be resolved by a number of different mechanisms, the DNS serves as a ubiquitous globally distributed network for translating human-readable domain names, or in other words identifiers, into actionable resources. Building a high assurance bridge between these systems as defined in [High Assurance DIDs with DNS] enables secure interactions that combine the strengths of both ecosystems. By linking the cryptographic assurances of DIDs with the DNS's global reach and resolvability, this bridge enables robust identifier verification, simplifies trust establishment by connecting the semantic and organizational context of human-readable domains to cryptographically verifiable identifiers, and facilitates practical use cases across a wide range of digital environments.



7.1. Identity Relationship Anchor

7.1.1. DID to DNS

The establishment of a pointer between a DID and a DNS domain varies based on whether the DID method in question is web based or otherwise.

For web based DIDs: The binding between a DID and a domain in the case of Web based DIDs is inherent to the identifier itself. The domain pointed to by the DID used for its resolution creates a unidirectional relationship between the DID and that domain.

Ex: did:web:example.ca -> example.ca

Non Web based DIDs: The binding between a non web based DID and a DNS domain is established via the “dnsValidationDomain” extension as defined in [https://www.w3.org/TR/did-extensions-properties/#dnsvalidationdomain] and [https://www.ietf.org/archive/id/draft-carter-high-assurance-dids-with-dns-06.html#section-3.2.1].

Ex: {"dnsValidationDomain": "mydomain.example"}

	Search in DID Document	Resolves to DNS Domain
Non Web Based DID methods	{“dnsValidationDomain”: “example.ca”}	“example.ca”
Web Based DID methods	{“id”:”did:web:example.ca”}	“example.ca”

7.1.2. DNS to DID

The DNS to DID pointer is established via a URI record as currently defined in [https://www.ietf.org/archive/id/draft-carter-high-assurance-dids-with-dns-06.html#section-3.3] and originally defined in [https://datatracker.ietf.org/doc/html/draft-mayrhofer-did-dns-05].

Ex: _did.example.ca URI 10 1 "did:web:example.ca"

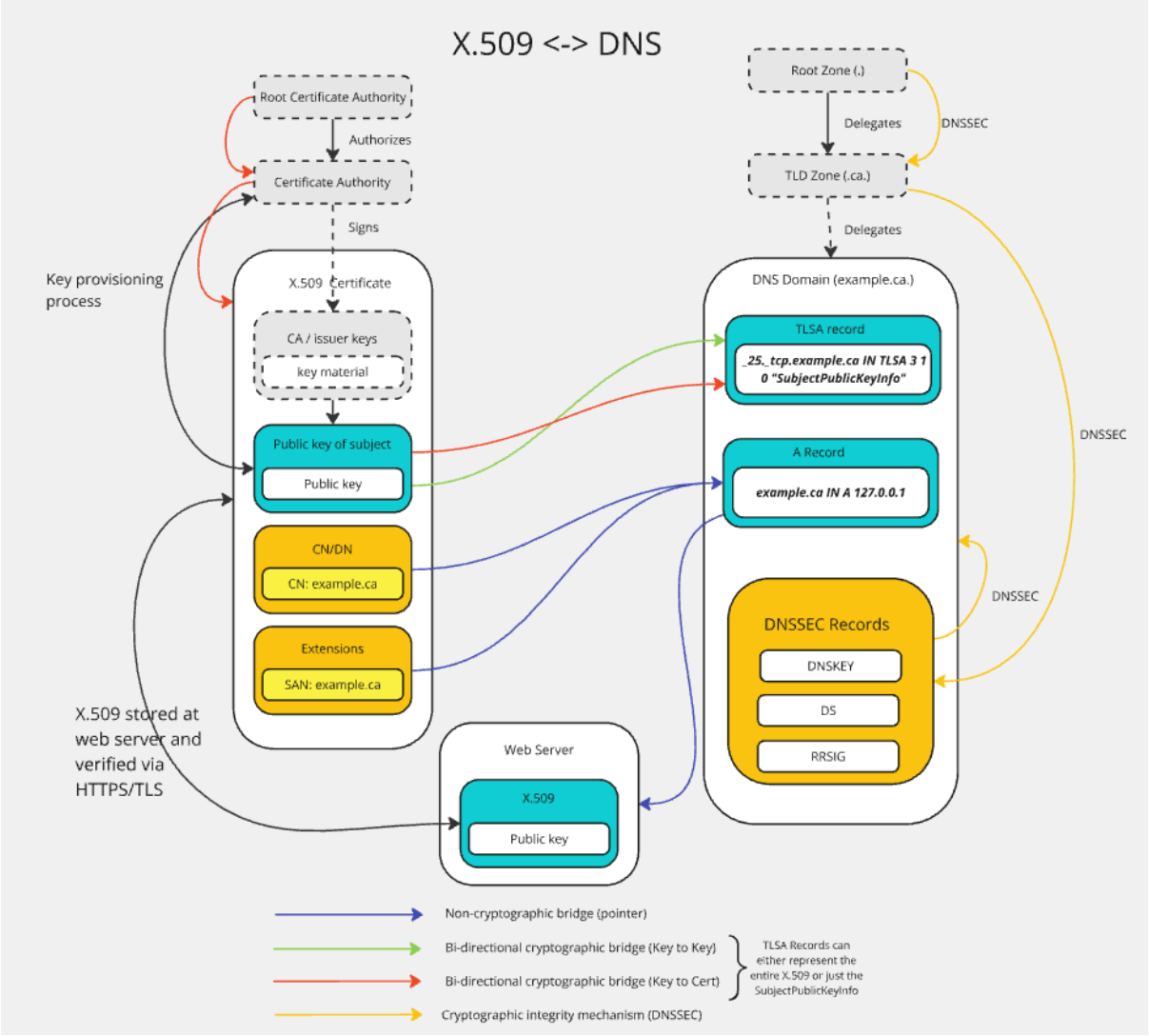
	Search for URI record DNS	Resolves to DID ID
	“_did.<domain name>”	“did:web:example”

7.2. Cross-Domain Trust Anchor

The establishment of a cryptographic bridge between a DID and a DNS domain is similar to the cryptographic bridge and verification process for a DID and X.509. The bridge itself is established through the mutual possession and verification of a key pair which is embedded in the DID document via a verificationMethod, and in the DNS domain as a TLSA record as defined in [https://www.ietf.org/archive/id/draft-carter-high-assurance-dids-with-dns-06.html#section-3.4]. However, unlike in the example above bridging a DID and X.509, where a non-static entity such as a DID is being bound to a static document like an X.509, a DNS domain is also non-static. To accommodate this, the security and integrity of both non-static entities needs to be guaranteed through some alternative means. In the case of the DNS, this is accomplished via DNSSEC, and in the case of the DID this is accomplished via the inclusion of a dataIntegrityProof on top of whatever integrity mechanisms the DID method itself provides.

8. X.509 and DNS

The vast majority of x.509 certificates in use today are domain certificates, which are primarily employed to secure HTTPS communications on the web. In this context, x.509 certificates are issued to domain names and are used to authenticate servers to clients, ensuring that user traffic is encrypted to the legitimate site they intend to access. This widespread application of x.509 certificates for securing HTTPS is a cornerstone of modern internet security.



8.1. Identity Relationship Anchor

8.1.1. X.509 to DNS

In early X.509 implementations, the CN was used to express the primary domain such as "www.example.com", as defined in [https://datatracker.ietf.org/doc/html/rfc5280#section-4.1.2.6]. However, the now recommended—and in many cases mandatory—method for binding an X.509 certificate to one or more DNS names is via the Subject Alternative Name extension as defined in [https://www.rfc-editor.org/rfc/rfc6125#section-6.4.4]

	Search in x.509	DNS Domain
Bridge Point	"CN"	"example.ca"
Bridge Point (2)	"SAN"	"example.ca"

8.1.2. DNS to X.509

In the common HTTPS usage of X.509 domain certificates, the DNS to X.509 bridge is not used. When a user contacts a web server (whose IP is resolved via DNS), the client relies solely on the static, CA-issued certificate as a fixed trust anchor. The DNS in this scenario serves merely to locate the server, while the certificate's integrity is ensured through a traditional chain of trust, certificate transparency, and revocation mechanisms.

Although less common, DANE (DNS-Based Authentication of Named Entities) is a DNS based protocol used to explicitly bind an X.509 certificate (or certificate-related data) to a domain. This mechanism, defined in [https://datatracker.ietf.org/doc/html/rfc6698], operates very similarly to the cryptographic layer of the DID to DNS binding, which expands on its usage of TLSA records. Under DANE, a domain's authoritative DNS zone (secured by DNSSEC) includes one or more TLSA records. These records can specify a hash of the expected certificate (or public key), or even the certificate itself, for a given service (typically defined by the transport protocol and port, such as _443._tcp.example.com). When a client wishes to leverage DANE for verification, it first retrieves the TLSA record from the target domain and verifies that it is properly secured by DNSSEC. The client then compares the information in the TLSA record (e.g., a hash of or complete certificate or public key) with the certificate presented during the TLS handshake for the target resource. A match confirms that the certificate not only bears the internal binding to a DNS name (via the SAN extension) but also that the domain's DNS zone asserts the same certificate, creating a bi-directional association between the two.

	DNS Domain	x.509
Bridge Point	example.ca. IN TLSA 3 0 1 3d0a67fc7e9f8a6a4b7c5d1e8f9c0a1b2d3e4f5a6b7c8d9e0f1a2b3c4d5e6f70	Full certificate or hash
Bridge Point (2)	example.ca. IN TLSA 3 1 1 3d0a67fc7e9f8a6a4b7c5d1e	Full SubjectPublicKeyInfo or hash

	8f9c0a1b2d3e4f5a6b7c8d9e0f1a2b3c4d5e6f70	

8.2. **Cross-Domain Trust Anchor**

9. Resolution and Dereferencing process

This section outlines the process for resolving to retrieve a High Assurance VID.

…

10. Resolution and Dereferencing examples

…

11. Applications of HAVIDs

12. Security considerations

This section discusses the security considerations associated with linking and dereferencing High Assurance VIDs.

13. Governance considerations

This section discusses the security considerations associated with linking and dereferencing High Assurance VIDs.

13.1 Certificate Policy Requirements for Certificate Authorities Resolving and Verifying DIDs

- MUST requirements
    - E.g. Go to DID Resolver
    - Match keys from Controller of DID to those in Certificate Signing Request (CSR)

14. Privacy considerations

This section explores the privacy considerations associated with High Assurance VIDs.

15. References

Normative References

[RFC2119]
S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. March 1997. Best Current Practice. URL: https://datatracker.ietf.org/doc/html/rfc2119

Informative References

[DID-CORE]
Manu Sporny; et al. Decentralized Identifiers (DIDs) v1.0. URL: https://w3c.github.io/did-core/
[RFC8174]
B. Leiba. Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words. May 2017. Best Current Practice. URL: https://www.rfc-editor.org/rfc/rfc8174
https://www.ietf.org/archive/id/draft-carter-high-assurance-dids-with-dns-05.html#section-3.3
Originally defined in: https://datatracker.ietf.org/doc/html/draft-mayrhofer-did-dns-05
Ex: _did.example.ca IN URI 1 0 “did:web:XXXXXX”

Notes:

Resolution Procedures:

- **DID to X.509:**
  - Look for a verificationMethod in the DID document which is of type JWK and contains the x5c parameter.
  - Decode the X.509 certificate from the x5c parameter.
  - Does the X.509 need to be in the DID document, or can the X.509 just correspond to a verificationMethod (public key) in the DID document, and the expectation is the verifier retrieves or happens upon the X.509 via other means?
- **X.509 to DID:**
  - Look for the DID contained in the SAN field of the X.509 certificate.
  - Resolve the DID document according to its DID method.
  - Same as question above, can either verify the key pair in the X.509 is represented in the DID doc as a verificationMethod, or this is no bridge necessary as the X.509 is already contained in a verificationMethod in the DID doc.
- **DID to DNS:**
  - For web based DIDs: The domain name specified in the identifier denotes the domain the DID is associated with (I.e did:web:example.ca -> example.ca)

    - For non web based DIDs: The dnsValidationDomain extension denotes the domain the DID is associated with.
- **DNS to DID:**
    - Query the DNS domain for URI records with the **_did** node prefix (I.e **_did.example.ca URI 10 1 "did:web:example.ca"**).
    - Resolve the DID according to its DID method.
- **X.509 to DNS:**
    - Look for the domain name in either the Subject/Common Name or SAN fields in the X.509.
- **DNS to X.509:**
    - Perform HTTPS/TLS with the webserver hosted at the given domain using the X.509.
    - Query for a TLSA record in the domain containing an X.509 certificate.
        - Verify the returned records contain either the X.509 certificate in full or its SubjectPublicKeyInfo

Non-Cryptographic Bridge: Pointing to X.509 certificates from DIDs

- The JWK spec supports the "x5c" and "x5t" parameters: https://datatracker.ietf.org/doc/html/rfc7517#section-4.7. Consistent with: https://www.w3.org/TR/did-core/#verification-material, a JWK is a valid verificationMethod format, so you just include the x5c or x5t params.

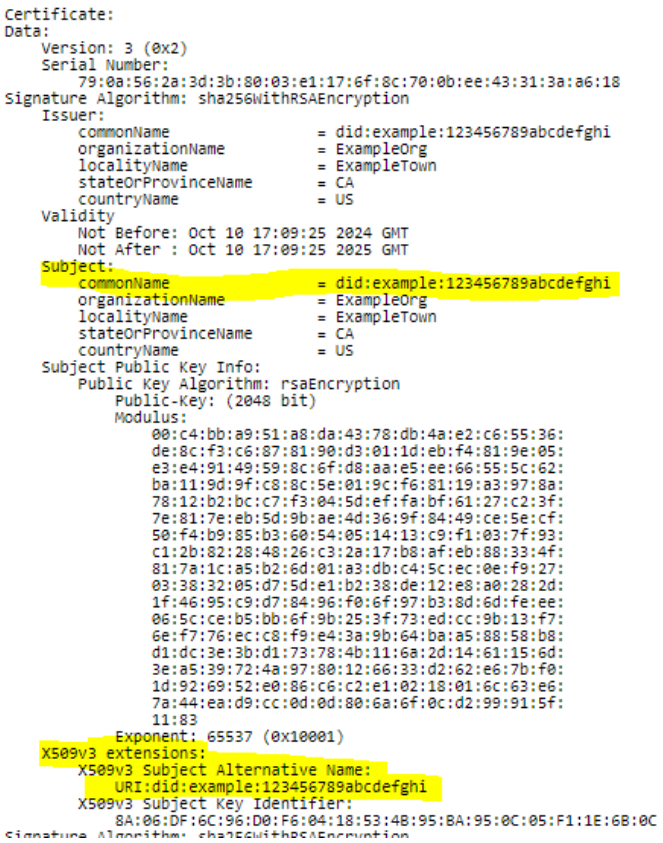
- Alternatively, just include the key from the certificate as a verificationMethod format of your choosing, then do an explicit binding on the x509 pointing directly to the verificationMethod id.
- A verificationMethod can also include an x509 certificate by using the x509CertificateChain property.
- The DIF ID WG has defined an extension DID resolution option and DID resolution error code related to verifying x509 certificates in DID documents: https://github.com/decentralized-identity/did-spec-extensions/blob/main/error-codes/not-allowed-certificate.md
- Other prior art related to this topic:
    - https://github.com/WebOfTrustInfo/rwot11-the-hague/blob/master/advance-readings/hybrid_wallet_solutions_x509_DIDs_VCs.md
    - https://lf-toip.atlassian.net/wiki/spaces/HOME/pages/22993854/X.509+PKD+Interop
    - https://github.com/transmute-industries/openssl-did-web-tutorial

15.1. **Non-Cryptographic Bridge:** Pointing to a DID from an x509 Certificate

- Potential candidates including but not limited to:
- Subject CN (Common Name) field

- SAN field

Ex:



- Specify guidance such that for an x.509 to be associated with a DID, that association (at minimum) is only valid as long as the DID contains a verificationMethod corresponding to the same key pair as the x509.
    - If this is the case, shouldn't it point to a verificationMethod rather than the DID?
    - Is there an intuitive way we can point to the specific verificationMethod fragment AND the DID? Is it adequate to have the guidance specify the general case applies to whatever DID is part of the fragment?
- Considerations for Hashlinks:
    - Make it optional and specify that if a hashlink is used, it denotes that the x509 only corresponds to the DID if it is exactly as hashed.

Cryptographic Bridge: Issuing X.509 cert and simultaneously creating DID with same keys, referenced in SAN

- As X.509 cert keys are generated and issued by the Certificate Authority, the CA could when issuing the X.509 cert, simultaneously use the keys to create a DID Document
- This would then create an equivalent X.509 certificate and DID Document

- Within the created X.509 certificate, it could reference the DID in the SAN. And within the DID Document, it could embed the X.509 certificate in a verificationMethod section such as keyAgreement.

- Needs to be some type of challenge/response
    - The signed mapping /relationship is unequivocal. Leading up to this signed mapping - there may be additional safeguards/gov mechanisms put in place. However, those are management/gov issues not technical issues. It is up to the issuer on how to decide that. We're just scoping out the unequivocal mechanism of how the bridges connect

- The syntax needs to be unequivocal, the semantics is not for us to decide

- **REQUIREMENT For an X.509 to relate to a DID, the DID must assert control over the same keypair. The entity is the controller over the key pairs on both sides.**

- **Potential for a section of the DIDDoc verification relationship called "bridgeMethod"**

- **did:example:12343#bridgeIdx509-key-1**

Proposed Requirements:

- **Put the DID in the SAN, may use a hashlink (https://datatracker.ietf.org/doc/html/draft-sporny-hashlink-05), maybe use the Common Name (see what vLEI)**
- **The keypair used to generate the CSR, and if necessary rekey the x509, needs to be present in the DID document as a verificationMethod.**
- **Before thou issue the x509 need to do a challenge and response with the DID using the keypair pertaining to the x509. (Establish resource ownership).**

Questions for us:

1. X.509 points to the domain + is signed hierarchically (if desired, can also be self signed)
2. Domain contains TLSA record corresponding to the cert (with above makes a bi directional pointer)
3. Domain is signed with DNSSEC (which is hierarchical, only the person with the zone signing key provided to the parent domain can sign records for that zone)
    a. In typical DANE usages, the record points to a resource which has access to the cert/key pair for some protocol, hence proving control of the private key.
    b. There is nothing signed from the key pair in the x509 in the domain
        i. Does there need to be? Is the above 2 stipulations sufficient?