Network Working Group
Internet-Draft
Intended status: Informational
Expires: 6 November 2025

J. Carter
J. Latour
CIRA
M. Glaude
Northern Block
T. Bouma
Digital Governance Council

5 May 2025

High Assurance DIDs with DNS
draft-carter-high-assurance-dids-with-dns-07

Abstract

This document outlines a method for improving the authenticity, discoverability, and portability of Decentralized Identifiers (DIDs) by utilizing the current DNS infrastructure and its technologies. This method offers a straightforward procedure for a verifier to cryptographically cross-validate a DID using data stored in the DNS, separate from the DID document.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://ciralabs.github.io/high-assurance-dids-with-dns/draft-carter-high-assurance-dids-with-dns.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-carter-high-assurance-dids-with-dns/.

Source for this draft and an issue tracker can be found at https://github.com/CIRALabs/high-assurance-dids-with-dns.

Status of This Memo

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

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

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

This Internet-Draft will expire on 6 November 2025.

Table of Contents

1. Introduction

In the ever-evolving digital world, the need for secure and verifiable identities is paramount. DIDs have emerged as a promising solution, providing a globally unique, persistent identifier that does not require a centralized registration authority. However, like any technology, DIDs face challenges in terms of authenticity, discoverability, and portability.

This is where the Domain Name System (DNS), a well-established and globally distributed internet directory service, comes into play. By leveraging the existing DNS infrastructure, we can enhance the verification process of DIDs. Specifically, we can use Transport Layer Security Authentication (TLSA) and Uniform Resource Identifier (URI) DNS records to add an additional layer of verification and authenticity to DIDs.

TLSA records in DNS allow us to associate a certificate or public key with the domain name where the record is found, thus providing a form of certificate pinning. URI records, on the other hand, provide a way to publish mappings from hostnames to URIs, such as DIDs.

By storing crucial information about a DID, such as the DID itself and its Public Key Infrastructure (PKI) in these DNS records, we can provide a verifier with a simple yet effective method to cross-validate and authenticate a DID. This not only ensures the authenticity of the DID document but also allows for interaction with material signed by the DID without access to the DID document itself.

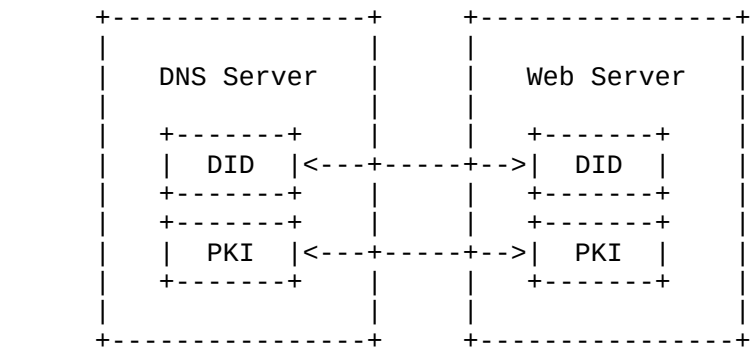
In essence, the integration of DIDs with DNS, specifically through the use of TLSA and URI records, provides a robust solution to some of the challenges faced by DIDs, paving the way for a more secure and trustworthy digital identity landscape.

2. Conventions and Definitions

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.

3. Securing a DID using the DNS

Much like presenting two pieces of ID to provide a higher level of assurance when proving your identity or age, replicating important information about a DID into a different domain (like the DNS) enables a similar form of cross validation. This enhances the initial trust establishment between the user and the DID document, as the key information can be compared and verified across two segregated sets of infrastructure. This also acts as a form of ownership verification in a similar way to 2FA, as the implementer must have control over both the DNS zone and the DID document to properly duplicate the relevant information.



The diagram above illustrates how a web server storing the DID document, and the DNS server storing the URI and TLSA records shares and links the key information about the DID across two independent sets of infrastructure.

3.1. Specifically for did:web

With did:web, there's an inherent link between the DNS needed to resolve the associated DID document and the domain where the relevant supporting DNS records are located. This means that the domain specified by the did:web identifier (for example, did:web:*example.ca*) is also the location where you can find the supporting DNS records.

3.2. Consideration for other DID methods

In the case of other DID methods, the association between a DID and a DNS domain is still possible although less inherent than with the aforementioned did:web. As such, it provides much of the same benefits as the [wellKnownDidConfiguration], but the method in which it accomplishes this is slightly different. Specifically, the integrity of the DID document is secured by including a

dataIntegrityProof inside the DID document itself rather than in a seperate resource, and the key material used to verify this proof is explicitly duplicated in the DNS, rather than only being referenced back to the DID document which is being verified.

3.2.1. dnsValidationDomain

To faciliate the linking of a non did:web to the DNS, we propose the inclusion of an optional property "dnsValidationDomain" to the DID document.

{"dnsValidationDomain": "example.ca"}

In the case of non did:webs that wish to use DNS for increased assurance, the verification process is identical to the one used for did:web but instead of referencing the domain in the identifier, the verifier MUST use the domain referenced by the "dnsValidationDomain" property instead.

3.3. Mapping DIDs to Domains with URI records

The association to a domain stemming only from the did is unidirectional. By leveraging URI records as outlined in [DID-in-the-DNS], we can create a bidirectional relationship, allowing a domain to publish their associated DID in the DNS.

_Ex: _did.example-issuer.ca IN URI 1 0 “did:web:example-issuer.ca” _

This relationship enhances security, as an entity would require control over both the DID and the domain’s DNS server to create this bidirectional association, reducing the likelihood of malicious impersonation.

3.3.1. URI record scoping

* The records MUST be scoped by setting the global underscore name of the URI RRset to __did_ (0x5F 0x64 0x69 0x64).

3.3.2. Entity Handles

An implementer may have multiple sub entities operating and issuing credentials on their behalf, like the different deparments in a university issuing diplomas or publishing research. For this reason, the introduction of an entity handle, represented as a subdomain in the resource record name, provides a simple way to facilitate the distinction of DIDs, their public keys, and credentials they issue in their relationship to another entity or root authority.

*_Ex: _did.diplomas.example-issuer.ca IN URI 1 0
“did:web:diplomas.example-issuer.ca” _*

*_Ex: _did.certificates.example-issuer.ca IN URI 1 0
“did:web:certificates.example-issuer.ca” _*

3.4. Mapping verificationMethods to the DNS with TLSA records

The DID to DNS mapping illustrated in section 3.2 provides a way of expressing the association between a DID and a domain, but no way of verifying that relationship. By hosting the public keys of that DID in its associated domain’s zone, we can provide a cryptographic linkage to bolster this relationship while also providing access to the DID’s public keys outside of the infrastructure where the DID document itself resides, facilitating interoperability and increasing availability.

TLSA records [RFC6698] provide a simple way of hosting cryptographic information in the DNS. Key material or full x509 certificates can be represented in TLSA records either hashed or unhashed depending on the requirements and use case of the implementer.

It is important to note that as key sizes increase in respect to the needs of post-quantum cryptography, TLSA records can support these keys via the hashed representation, making this implementation post-quantum compatible.

3.4.1. TLSA Record Scoping, Selector Field

When public keys related to DIDs are published in the DNS as TLSA records:

* The records MUST be scoped by setting the global underscore name of the TLSA RRset to __did_ (0x5F 0x64 0x69 0x64).

* The Selector Field of the TLSA record must be set to 1, SubjectPublicKeyInfo: DER-encoded binary structure as defined in [RFC5280].

When x509 certificates related to DIDs are published in the DNS as TLSA records:

* The records MUST be scoped by setting the global underscore name of the TLSA RRset to __did_ (0x5F 0x64 0x69 0x64).

* The Selector Field of the TLSA record must be set to 0, full certificate: the Certificate binary structure as defined in [RFC5280].

3.4.2. Instances of Multiple Key Pairs

Depending on the needs of the implementer, it is possible they may use multiple keypairs associated with a single DID to sign and issue credentials or enable other PKI related interactions. In this case, a TLSA record will be created per [verificationMethod] and then be bundled into the corresponding TLSA RRset. A resolver can then parse the returned records and match the key content to the verificationMethod they wish to interact with or verify.

_Ex: _did.example-issuer.ca IN TLSA 3 1 0 "4e18ac22c00fb9...b96270a7b4"_

_Ex: _did.example-issuer.ca IN TLSA 3 1 0 "5f29bd33d11gc1...b96270a7b5"_

3.4.2.1. Security Consideration

It is RECOMMENDED implementers limit the total number of TLSA records for a given domain to 255 to mitigate DoS style attacks, such as creating a problematic number of TLSA records to then be resolved and parsed by the verifier.

If total number of TLSA records returned to a verifier exceeds this threshold, it is RECOMMENDED the verifier abort the verification process and deem the target DID insecure.

3.4.3. Benefits of Public Keys in the DNS

Hosting the public keys in TLSA records provides a stronger mechanism for the verifier to verify a did and its associated entity with, as they are able to perform a cryptographic challenge against the DID using the corresponding TLSA records, or against the domain using the corresponding [verificationMethod] in the DID document. The accessibility of the public keys is also beneficial, as the verifier does not need to resolve the DID document to accesss its associated key material, enhancing interoperability.

4. Role of DNSSEC for Assurance and Revocation

It is RECOMMENDED that all the participants in this digital identity ecosystem enable DNSSEC signing for the DNS instances they operate. See [RFC9364].

DNSSEC provides cryptographic assurance that the DNS records returned in response to a query are authentic and have not been tampered with. This assurance within the context of the __did_ URI and __did_ TLSA records greatly strengthens the mechanism to ensure the integrity of

the DID and its public keys. DNSSEC vastly reduces the possible attack vectors in which the repudiated DID information in the DNS can be tampered with.

Within this use-case, DNSSEC also provides revocation checks for both DIDs and their public keys. In particular, a DNS query for a specific __did_ URI record or __did_ TLSA record can return an NXDOMAIN [RFC8020] response if the DID or public key has been revoked. This approach can simplify the process of verifying the current validity of DIDs and public keys by reducing the need for complex revocation mechanisms or implementation specific technologies.

5. Digital Signature and Proof Value of the DID Document

Digital signatures ensure the integrity of the DID Document, and by extent the public keys, authentication protocols, and service endpoints necessary for initiating trustworthy interactions with the identified entity. The use of digital signatures in this context provides a robust mechanism for verifying that the DID Document has not been tampered with and indeed originates from the correct entity.

In accordance with W3C specifications, we propose including a data integrity proof such as those outlined in [dataIntegrityProofECDSA] and [dataIntegrityProofEdDSA], with the mandatory inclusions of the "created" and "expires" fields. The inclusion of which acts as a lifespan for the document, similar to the TTL for a DNS record. Depending on the use case and security requirements, a longer or shorter expiry period would be used as necessary.


```
"proof": {
   "type": "DataIntegrityProof",
   "cryptosuite": "ecdsa-jfc-2019",
   "created": "2023-10-11T15:27:27Z",
   "expires": "2099-10-11T15:27:27Z",
   "proofPurpose": "assertionMethod",
   "verificationMethod": "did:web:trustregistry.ca#key-1",
   "proofValue": "zQeVbY4oey5q2M3XKaxup3tmzN4DRFTLVqpLMweBrSxMY2xHX5XTYV8nQApmEcqaqA3Q1gV
HMrXFkXJeV6doDwLWx"
}
```


The data integrity proof SHOULD be signed using a verificationMethod that has an associated TLSA record to allow for the verification of the data integrity proof using pki material contained outside of the DID document. This provides an added layer of authenticity, as the PKI information contained in the DID document would need to be repudiated across 2 different domains, the resource hosting the DID document and its associated DNS domain.

5.1. Use of Alternative Cryptosuites

While [dataIntegrityProofECDSA] and [dataIntegrityProofEdDSA] are the cryptosuites we have chosen to highlight in this specification, it is important to note that this implementation for a high assurance did using dns is cryptosuite agnostic. It is interoperable with any new and existing cryptosuites and associated key types as required by the implementers and verifiers.

6. Verification Process

Using the new DNS records and proof object in the DID document, we enable a more secure and higher assurance verification process for the DID. It is important to note that while not strictly necessary, DNSSEC verification SHOULD be performed each time a DNS record is resolved to ensure their authenticity.

The process below outlines the general steps required to complete the higher assurance did verification process;

1. *Verification of the DID:* The user verifies the DID is represented as a URI record in the associated domain.
    1. In the case of did:web, the domain and record name to be queried is indicated by the last segment of the did. In example, *did:web:example.ca* would translate to a URI record with the name *_did.example.ca*.
    2. In the case of other did methods, the domain and record name to be queried is indicated by the "dnsValidationDomain" property. In example, *{"dnsValidationDomain": "example.ca"}* would translate to a URI record with the name *_did.example.ca*.
2. *Verification of the PKI:* The user verifies the verificationMethod/s in the DID document are represented as TLSA record/s in the associated domain.
    1. The domain and record name for the TLSA record to be queried is determined identically to steps 1.a or 1.b.
        1. Note: The matching of the TLSA record content and verificationMethod may require some conversion, as TLSA records store key material as hex encoded DER format, and this representation is not supported by [verificationMethod]. However, there are many well supported cryptography libraries in a variety of languages that facilitate the conversion process.

3. *Verification of the DID document’s integrity:* The user verifies the "proof" object to ensure the integrity of the DID document.

    1. This can be accomplished by using either the [verificationMethod] directly from the DID document, or using the key material stored in the TLSA record. Using the TLSA record would provide a higher level of assurance as this confirms the key material is being accurately represented across 2 different domains, both at the DID document level and the DNS level.

        1. Note: Unlike with matching the verificationMethod and TLSA record in step 2, DER is a widely supported encoding format for key material enabling a verifier to directly use the TLSA record content to verify the signature without having to convert the key back to its representation in the verificationMethod.

Note: The order of the steps presented does not specify a required order for verification. As a general rule (and depending on the use case) the 3 verification steps outlined above may be performed in any order as best fits the verifier’s needs. In example, a verifier may arrive at the DID document during a credential verification process, in which case it makes sense to peform step 3 before steps 1 and 2. Alternatively, a verifier may arrive at the DID document after exploring an organization’s domain, in which case it may make more sense to perform steps 1 and 2 prior to step 3. So long as the 3 steps are performed together, the same level of assurance is achieved irrespective of their order.

6.1. Verification Failure

If at any given step verification fails, the DID document should be deemed INSECURE. Whether it is due to the DID and DNS being out of sync with recent updates, or the resource hosting the DID document or DNS zone themselves being compromised, a failed verification MAY indicate malicious activity. It is then up to the verifier to determine, according to their requirements and use case, the appropriate course of action regarding interactions with the target DID until successful verification is restored.

7. Control Requirements

This section defines a simple framework to define a set of technical controls that can be implemented and mapped into levels of assurance for did:web identifiers. To assist in decision-making and implementation, The controls are ordered in increasing level of security assurance and are grouped into levels of assurance from *LOW-* to *HIGH+* - *Issuing Authority* is the entity accountable for the did:web identifier. - *Issuing Service* is the entity responsible for operating the did:web identifier infrastructure. In many cases the *Issuing Authority* may delegate elements of providing a high assurance did:web identifier to an *Issuing Service* that may be a commercial provider. In the simplest case, the *Issuing Authority* can be regarded as the same as the *Issuing Service*. Note that Controls 9, 10, and 11 CANNOT BE DELEGATED to an *Issuing Service*

11 technical controls are defined. These controls would be implemented in order of precedence for an increasing level of security assurance. (e.g., Control No. N would need to be implemented before implementing Control No. N+1)

Control No.	Control Name	Description
1	DID Resource Control	The Issuing Service MUST control the resource that generates the DID document. (i.e., website)
2	DID Document Management	The Issuing Service MUST have the ability to do CRUD operations on the DID document.
3	DID Document Data Integrity	The Issuing Service MUST ensure the data integrity of the DID document by cryptographic means, typically a digital signature or other means. The use of approved or established cryptographic algorithms is HIGHLY RECOMMENDED
4	DID Document Key Control	The Issuing Service MUST control the keys required to sign the DID document.
5	DID Document Key	With proper delegation from the Issuing Authority, the DID Document signing key MAY be generated by the Issuing Service.

	Generation	Otherwise, the signing key must be generated by the Issuing Authority.
6	Domain Zone Control	The Issuing Service MUST have control of the domain zone (or subdomain zone).If direct control of the domain is not feasible, the use of an accredited DNS provider is HIGHLY RECOMMENDED
7	Domain Zone Mapping	There MUST be domain zone records that map the necessary URI, TLSA, CERT and/or TXT records to the specified did:web identifier.
8	Domain Zone Signing	The domain zone records MUST be signed according to DNSSEC. (RRSIG)
9	Domain Zone Signing Key Control	The Issuing Authority MUST have control over the domain zone keys used for signing and delegation. (KSK and ZSK)
10	Domain Zone Signing Key Generation	The domain zone signing key MUST be generated under the control of the Issuing Authority.
11	Hardware Security Module	A FIPS 140-2 compliant hardware security module must be under the control of the Issuing Authority.

Table 1

In addition to the technical controls specified in the table it is advisable to add in DANE (DNS-based Authentication of Named Entities) [RFC6698] to secure TLS communications. TLS uses certificates to bind keys to names, which are published by public "Certification Authorities" (CAs). It is important to realize that the public CA model is fundamentally vulnerable because it allows any CA to issue a certificate for any domain name. Thus, a compromised CA can issue a fake replacement certificate which could be used to subvert TLS-protected websites. DANE offers the option to use the DNSSEC infrastructure to store and sign keys and certificates that are used by a TLS-protected website. The keys are bound to names in the Domain Name System (DNS), instead of relying on arbitrary keys and names issued in a potentially compromised certificate.

8. Levels of Assurance

Many trust frameworks specify levels of assurance to assist in determining which controls must be implemented.

The following table is not a definitive mapping to trust framework levels of assurance. It is intended to assist in determining mappings by grouping the controls within a range from *LOW-* to *HIGH+* relating to the appropriate risk level. Note that controls are additive in nature. (i.e.,, controls of the preceding level must be fulfilled).

Level of Assurance	Controls	Description
LOW-	Control 1	SHOULD only be used for low risk transactions where attribution to originator is desirable.
LOW	Control 2	SHOULD only be used for lower risk transactions where establishing the accountability of the originator is desirable.
MEDIUM	Controls 3, 4 and 5	MAY be used for medium risk commercial transactions, such as correspondence, proposals, etc.
MEDIUM+	Controls 6 and 7	MAY be used for higher risk transactions, such as signing and verifying invoices, contracts, or official/legal documentation
HIGH	Controls 8, 9 and 10	MUST be high risk transactions, such as government transactions for signing and verifying licenses, certifications or identification
HIGH+	Control 11	MUST be used for extremely high risk transactions where there may be systemic or national security implications

Table 2

9. Security Considerations

TODO Security

10. IANA Considerations

Per [RFC8552], IANA is requested to add the following entries to the "Underscored and Globally Scoped DNS Node Names" registry:

RR Type	_NODE NAME	Reference
TLSA	_did	[draft-ietf-high-assurance-dids-with-dns]
URI	_did	[draft-mayrhofer-did-dns-01]

11. References

11.1. Normative References

[dataIntegrityProofECDSA]
"Data Integrity ECDSA Cryptosuites v1.0", n.d., <https://www.w3.org/TR/vc-di-ecdsa/#proof-representations>.

[dataIntegrityProofEdDSA]
"Data Integrity ECDSA Cryptosuites v1.0", n.d., <https://www.w3.org/TR/vc-di-eddsa/#proof-representations>.

[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/rfc/rfc2119>.

[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/rfc/rfc5280>.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <https://www.rfc-editor.org/rfc/rfc6698>.

[RFC8020] Bortzmeyer, S. and S. Huque, "NXDOMAIN: There Really Is Nothing Underneath", RFC 8020, DOI 10.17487/RFC8020, November 2016, <https://www.rfc-editor.org/rfc/rfc8020>.

[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/rfc/rfc8174>.

[RFC8552] Crocker, D., "Scoped Interpretation of DNS Resource Records through "Underscored" Naming of Attribute Leaves", BCP 222, RFC 8552, DOI 10.17487/RFC8552, March 2019, <https://www.rfc-editor.org/rfc/rfc8552>.

[RFC9364] Hoffman, P., "DNS Security Extensions (DNSSEC)", BCP 237, RFC 9364, DOI 10.17487/RFC9364, February 2023, <https://www.rfc-editor.org/rfc/rfc9364>.

[verificationMethod] "Decentralized Identifiers (DIDs) v1.0", n.d., <https://www.w3.org/TR/did-core/#verification-methods>.

11.2. Informative References

[DID-in-the-DNS] "The Decentralized Identifier (DID) in the DNS", n.d., <https://datatracker.ietf.org/doc/html/draft-mayrhofer-did-dns-05#section-2>.

[DIDCore] "Decentralized Identifiers (DIDs) v1.0", n.d., <https://www.w3.org/TR/did-core>.

[didSpecRegistries] "Did Specification Registries", n.d., <https://w3c.github.io/did-spec-registries>.

[Self-Sovereign-Identity] Reed, D. and A. Preukschat, "Self-Sovereign Identity", ISBN 9781617296598, 2021.

[W3C-VC-Data-Model] "Verifiable Credentials Data Model v2.0", n.d., <https://www.w3.org/TR/vc-data-model/>.

[wellKnownDidConfiguration] "Well Known DID Configuration", n.d., <https://identity.foundation/.well-known/resources/did-configuration/>.

Appendix A. W3C Considerations

1. We propose the inclusion of an optional data integrity proof for the DID document, as outlined in [dataIntegrityProofECDSA] and [dataIntegrityProofEdDSA].

2. We propose the inclusion of the optional "dnsValidationDomain" property to the [didSpecRegistries] as outlined in section 3.2.

Acknowledgments

TODO acknowledge.

Authors' Addresses

Jesse Carter
CIRA
Email: jesse.carter@cira.ca

Jacques Latour
CIRA
Email: jacques.latour@cira.ca

Mathieu Glaude
Northern Block
Email: mathieu@northernblock.ca

Tim Bouma
Digital Governance Council
Email: tim.bouma@dgc-cgn.org