The `did:nda` Method Specification

This document defines the `did:nda` DID Method, conforming to W3C DID Core and DID Resolution specifications.

---

1. Introduction

The `did:nda` method identifies subjects (humans, organizations, or items) in the **Verifiable Data Registry (VDR)** powered by the **NDA Chain blockchain**. It is a hybrid architecture designed to provide secure, verifiable, and portable digital identities.

This system follows W3C DID standards, leveraging the NDA Chain for immutable record-keeping of minimal data (like public key references) while storing richer DID Documents off-chain. This hybrid approach balances decentralization with practical governance and scalability.

Resolution is handled by a dedicated resolver service, ensuring that DID Documents are retrievable without querying the blockchain directly for all data, which enhances privacy and efficiency.

---

2. DID Method Name

The registered method is `nda`.

A valid DID starts with: `did:nda:`

---

3. DID Syntax

A `did:nda` DID is formed by the method prefix and a unique identifier derived from an NDA Chain blockchain address.

Regex:

```
^did:nda:0x[0-9a-fA-F]{40}$
```

ABNF:

```
did-nda                  = "did:nda:" method-specific-id
method-specific-id       = "0x" 40*HEXDIG
HEXDIG                   = DIGIT / "a" / "b" / "c" / "d" / "e" / "f" / "A" / "B"
/ "C" / "D" / "E" / "F"
DIGIT                    = %x30-39
```

---

4. Method Operations (CRUD)

This method supports **Create**, **Read**, and **Update** operations.

Create (Register) A `did:nda` is implicitly created when a user generates a new key pair (a private key and its corresponding public address) on the NDA Chain. The DID is the address itself, prefixed by `did:nda:`. No explicit "registration" transaction is required to *create* the identifier, as it is self-sovereign. The DID Document associated with it is generated and published to the VDR via the Resolver API when the DID is first used or resolved.

Read (Resolve) A `did:nda` is resolved by querying the **NDA DID Resolver service**. The resolver fetches the DID Document associated with the DID from its off-chain storage (e.g., application cache, dedicated database) and returns it to the client. See Section 6 for resolution details.

Update (Replace) DID Documents **can be updated**. The `controller` of the DID (who holds the associated private key) can authorize changes to the DID Document, such as rotating keys, adding or removing service endpoints, or modifying authentication methods. Updates are submitted to the DID Resolver service API and must be cryptographically signed by the controller to prove ownership.

Deactivate (Revoke) Deactivation is achieved by performing an **Update** operation. The controller updates the DID Document to remove or revoke all `verificationMethod` and `service` entries. A DID Document can also be updated to point its `controller` to a "null" or un-ownable DID, effectively orphaning it.

---

5. Method-Specific Identifiers

The `did:nda` method-specific identifier is a **blockchain address** from the NDA Chain.

- **Allowed charset**: digits `[0–9]`, lowercase letters `[a–f]`, uppercase letters `[A–F]`, and the prefix `0x`.
- **Length**: 42 characters total (`0x` prefix + 40 hexadecimal characters).
- **Collision management**: Uniqueness is guaranteed by the underlying blockchain's cryptographic address generation process.
- **Example**: `did:nda:0xdfc8044a202f08b9c8df2e42f746355575816828`

---

6. DID Resolution

Resolution is performed via an HTTP GET request to the NDA DID Resolver service.

Reference Endpoint:

```
GET [https://resolver.ndadid.vn/1.0/identifiers/]
(https://resolver.nda.network/1.0/identifiers/){did}
```

Example Request:

```
GET
[https://resolver.ndadid.vn/1.0/identifiers/did:nda:0xdfc8044a202f08b9c8df2e
42f746355575816828]
(https://resolver.nda.network/1.0/identifiers/did:nda:0xdfc8044a202f08b9c8df
2e42f746355575816828)
```

Response: On success, the service returns a `200 OK` with the DID Document (in `application/ld+json` format) as the response body.

Resolution error examples:

- `404 Not Found`: The DID Document for the given DID does not exist in the registry.
- `400 Bad Request`: The DID string is malformed or invalid.

---

7. DID Document

When a `did:nda` is resolved, the resulting DID Document is a JSON-LD object that conforms to the W3C DID Core v1.0 specification.

The document includes multiple contexts:

- `https://www.w3.org/ns/did/v1` (DID Core v1.0)
- `https://w3id.org/security/v1` (Security & Cryptography Standards)

Example DID Document

```
{
  "@context": [
    "[https://www.w3.org/ns/did/v1](https://www.w3.org/ns/did/v1)",
    "[https://w3id.org/security/v1](https://w3id.org/security/v1)"
  ],
  "id": "did:nda:0xdfc8044a202f08b9c8df2e42f746355575816828",
  "controller": "did:nda:0xdfc8044a202f08b9c8df2e42f746355575816828",
  "verificationMethod": [
    {
      "id": "did:nda:0xdfc8044a202f08b9c8df2e42f746355575816828#keys-1",
      "type": "EcdsaSecp256k1VerificationKey2019",
      "controller": "did:nda:0xdfc8044a202f08b9c8df2e42f746355575816828",
      "publicKeyHex":
"04abf141154b386b0203f7e0e7a1c4e7f805f833b9b3e1f0e8f0c3d8e5f0a2a1a0b1c2d3e4f
5a6b7c8d9e0f1a2b3c4d5e6f7a8b9c0d1e2f3a4b5c6d7e8f9a0b1c"
    }
  ],
  "authentication": [
    "did:nda:0xdfc8044a202f08b9c8df2e42f746355575816828#keys-1"
  ],
  "assertionMethod": [
    "did:nda:0xdfc8044a202f08b9c8df2e42f746355575816828#keys-1"
  ]
}
```

---

8. Governance

- **Authority**: The NDA Chain and the associated Verifiable Data Registry (VDR) are maintained by `Vietnam’s National Data Center`.
- **Maintainer**: The resolution service (`resolver.ndadid.vn`) is operated and maintained by `Vietnam’s National Data Center`.
- **Process to obtain a DID**: As described in Section 4 (Create), DIDs are generated by users via cryptographic key-pair creation. The associated DID Document is published to the VDR via the NDA Resolver service API, requiring a signature from the controller.

---

9. Privacy & Security Considerations

Security Considerations

- **Key Management**: The security of the `did:nda` method is entirely dependent on the controller's ability to secure their private key. Private keys should be stored in secure environments, such as a Hardware Security Module (HSM) or a device's Secure Element (SE).

- **Trust Model**: Trust is rooted in the cryptographic security of the NDA Chain and the integrity of the NDA DID Resolver service. Verifiers must trust that the resolver service provides the correct DID Document associated with a given DID.
- **Authentication**: All update operations on a DID Document MUST be cryptographically signed by the key(s) associated with the `controller` DID, proving ownership.

Privacy Considerations

- **Data Minimization (On-Chain)**: The NDA Chain only stores minimal data (DIDs and references). No Personally Identifiable Information (PII) is stored on the blockchain.
- **Off-Chain Documents**: The DID Document itself is stored off-chain. While it does not contain PII, it does contain public keys.
- **Selective Disclosure**: PII is only ever contained within Verifiable Credentials (VCs), which are held by the user (e.g., in a digital wallet) and are never published to the resolver or the blockchain. The user presents VCs and can choose which information to disclose (selective disclosure).
- **Pseudonymity**: A `did:nda` is a pseudonymous identifier. It is not inherently linked to a real-world identity unless the holder chooses to link it by, for example, obtaining a VC from an issuer that verifies their identity.

---

10. Lifecycle

- **Versioning**: This specification follows Semantic Versioning. Breaking changes will require a new version.
- **Deprecation/Deactivation**: As described in Section 4.4, DIDs can be deactivated by updating their DID Document to remove keys and services. A `404 Not Found` or a DID Document with no verification methods may be returned for a deactivated DID.

---

11. W3C Registration

- The method is to be published in the **DID Method Registry**.
- **Initial status**: provisional.

---

12. Test Vectors

- **Example 1:** `did:nda:0xdfc8044a202f08b9c8df2e42f746355575816828` → see `examples/nda-1.json` (This implies a valid DID Document file exists at this path in your repository).

---

13. References

- **givfg[W3C DID Core v1.0]**: https://www.w3.org/TR/did-core/
- **[W3C DID Resolution]**: https://www.w3.org/TR/did-resolution/
- **[W3C Security Vocabulary]**: https://w3id.org/security/v1
- **[EcdsaSecp256k1VerificationKey2019]**: https://w3c-ccg.github.io/data-integrity-spec/#EcdsaSecp256k1VerificationKey2019