Sovereign Identity Namespaces

A paper for the ID2020 Design Shop (the second Rebooting the Web of Trust event)
Drummond Reed, Respect Network, 2016-05-16

Introduction

Sovereign identity is a new type of digital identity that enables a **principal** (person, group, or thing) to have a digital identity that is completely owned and controlled by the principal. A sovereign identity can never be taken away by another authority. This and other principles of sovereign identity are explained in another paper for the ID2020 Design Shop, A Self-Sovereign Identity Architecture by Christopher Allen, and also in the blog post Why Companies Need Self-Sovereign Identity from Phil Windley at Brigham Young University.

The main enabler of sovereign identity is distributed ledger technology ("blockchain"). However, to implement the principles of sovereign identity, these principles must be followed right down to the most foundational architectural component: how identities are registered on a distributed ledger. This paper proposes the use of three globally unique namespaces:

1. **DIDs** (decentralized identifiers)—machine-generated globally unique identifiers that do not have cryptographic properties.
2. **CIDs** (cryptographic identifiers)—machine-generated globally unique identifiers with cryptographic properties.
3. **Aliases**—registered identifiers from other global namespaces that simplify discovery of DIDs and CIDs.

The paper also explains why new **HFNs** (human-friendly namespaces) are not recommended for sovereign identity because this capability is better served by two alternatives:

1. **Sovereign identity directory services** that index DIDs and CIDs by human-friendly names.
2. **Linked local names** that provide peer-to-peer naming without reliance on centralized registries.

DIDs (Decentralized IDentifiers)

The term "DID" originated with the WebDHT proposal from the W3C Credentials Community Group. The need for DIDs is explained in the Introduction section of that proposal:

> *The Web currently does not have a mechanism where people and organizations can claim identifiers that they have sole ownership over. Identifiers, such as those rooted in*

domain names like emails addresses and website addresses, are effectively rented by people and organizations rather than owned. Therefore, their use as long-term identifiers is dependent upon parameters outside of their control. One danger is that if the rent is not paid, all data associated with the identifier can be made temporarily or permanently inaccessible.

The main reason this problem exists is two fold: it's a particularly hard one to solve and there was no time to build out the solution when the Web was starting its rapid rise in the late 1990s. The solutions for creating identifiers that we have in place today, particularly DNS, HTTP, and URLs, are powerful tools that have scaled the Web to the billions of people that use it today.

However, as the Web expands and becomes more personal, there is a growing need for a type of dereference-able identifier that can be created and owned by people and organizations without the need for a central organization or federation of organizations to control the namespace. Having such a scheme would enable more personal and organizational freedom on the Web.

The specification goes on to give the following definition of a DID:

decentralized identifier

A portable URI-based identifier, also known as a DID, that is associated with an entity. These identifiers are most often used in a credential and are associated with recipients such that the credential itself can be easily ported from one identity provider to another without the need to reissue the credential. An example of a DID is: `did:b6922d8e-20df-4939-95cd-f79375979178`

The WebDHT proposal specifies that the value of a DID is a type 4 UUID. This means the **did**: URI space is essentially identical to the **urn:uuid:** namespace specified in RFC 4122 except it is restricted to type 4 UUIDs (considered the strongest form with the lowest chance of collision).

As an identifier, the primary properties of a DID are:

1. It is globally unique with an extremely high probability.
2. It can be machine-generated without any reference to a centralized registry.
3. It is immutable, i.e., it never needs to expire.
4. It does not have any cryptographic properties (and thus does not need to be revoked if those cryptographic properties are compromised).

In addition, since a DID is a UUID, it has a fixed length of 128 bits. RFC 4122 standardizes serialization of a UUID in a 36 character string (32 hex digits plus four hyphens). However DID architecture may use base 64 or 58 for a more compact representation (22 characters).

CIDs (Cryptographic IDentifiers)

A CID is identical to a DID in all respects except:

1. It has a specific set of cryptographic properties as defined in an associated specification.
2. Due to these properties, different types of CIDs may have different lengths.

The cryptographic properties of a CID derive from its association with some form of a secret. Knowledge of that secret may be used by the principal to provide cryptographic proof of control of the CID.

This does not necessarily mean that a CID must be a public key as defined in public/private key cryptography. While it may be a public key, it may also be a truncated public key, a hash of a public key, a truncated hash of a public key, or some other algorithmically-defined variant. A CID may also employ some other form of cryptography not based on public/private keys.

Because the cryptographic properties of a CID vary with the CID type, it is strongly recommended that:

1. **A CID include a type indicator that unambiguously identifies the associated CID specification.** Note that the XDI Core 1.0 specification accomplishes this with a reserved CID identifier scheme prefix in the form **cid-x:** where X is a sequence of digits that uniquely identifies the associated CID specification.
2. **The associated CID specification must unambiguously define the cryptographic properties of the CID** such that any implementation generating or consuming CIDs of this type may use these properties with confidence.

If CIDs have all the properties of DIDs along plus the advantages of cryptographic proof, why does sovereign identity architecture still need DIDs? The answer is that *some resources do not need cryptographic properties or need to define them independent of the identifier for that resource*. If a CID is used in these use cases, it sends a false signal of cryptographic properties that do not exist. A DID is by definition independent of any cryptographic properties and therefore does not need to be updated, invalidated, retired, or otherwise modified as may be required with a CID whose cryptographic properties are compromised.

Aliases

The third proposed class of identifiers for sovereign identity is also proposed in the WebDHT proposal: **aliases**. An alias is a globally unique identifier in another pre-established namespace that can be used as an alternative identifier for a principal. This paper proposes four primary alias namespaces:

1. Mobile phone numbers.

2. Email addresses.
3. Twitter handles.
4. Web addresses.

Aliases from these namespaces fulfill two primary purposes:

1. **Ease of use.** DIDs and CIDs, despite all their benefits, are essentially unusable by humans. Alias are just the opposite: familiar identifiers in well-known formats that people use every day.
2. **Ease of discovery.** Aliases are identifiers that already exist in user's address books (mobile phone, email client, Twitter client, browser, etc.) Thus they make it easy to discover and connect (either manually or programmatically) with other principals.

However, precisely because aliases are not generated algorithmically, their ownership cannot be verified solely through registration on a sovereign identity ledger. This introduces the requirement to verify a principal's ownership of an alias prior to registration. Various methods can be implemented to accomplish this, e.g., closed-loop authentication or OAuth verification, (the details are beyond the scope of this paper). Because these methods are relatively weak, however, they should be used only to verify control of an alias and not as any form of authentication for the principal. Authentication of a principal discovered through an alias should always be based on the associated DID or CID.

For privacy purposes, an alias should be hashed prior to registration on the ledger (see the example in the WebDHT proposal). Note that hashing does not render the plaintext alias entirely private—it is still discoverable using a simple dictionary or rainbow-table style attack, which is particularly feasible within known contexts (e.g., different email addresses within a specified domain). However this is mitigated by two factors:

1. In most cases the actual plain-text alias is already public or easily discoverable.
2. The associated DID or CID should not reveal any other information about the principal.

Unlike DIDs or CIDs, aliases are not permanent. They come from external namespaces where ownership may change from one principal to another. Therefore the rule must be "most recent wins". In other words, if Alice verifies control of spacecowboy@gmail.com, and then Bop verifies control of spacecowboy@gmail.com 3 months later, Bob is the new owner.

Why New HFN (Human-Friendly Namespaces) For Sovereign Identity Are Not Recommended

It is tempting to introduce a new human-friendly namespace dedicated to sovereign identity. For example, namespaces based on the XDI global context symbols = (for people) and + (for groups/organizations) would appear to be ideal for this purpose. However this is not recommended for the following reasons:

1. **HFNs raise many difficult and controversial governance issues.** Look no further than the history of issues surrounding ICANN's governance of DNS namespaces.
2. **HFNs are not actually sovereign, i.e., they are mutable identifiers that may be taken away from a principal.** As an identifier with semantic meaning, an HFN is inherently subject to legal policies beyond of the scope or power of a sovereign identity ledger. So the introduction of a new HFN namespace for sovereign identity may actually undermine the core concept.
3. **HFNs typically have first-come-first-served policies that can alienate those who are not first in line.** The resulting backlash can often offset the advantage of introducing a new HFN.
4. **A new HFN requires that users learn a new namespace syntax.** The online world is already confusing enough. (On the other hand, a new syntax that catches on can develop its own branding, as illustrated by Twitter @names.)
5. **Most of the benefits of a new HFN can be delivered by aliases without the resulting downsides.** Alias namespaces (mobile phone numbers, email addresses, Twitter handles, URLs) already deal with these difficult HFN issues. A sovereign identity ledger can simply inherit these solutions.

In addition to aliases, there are two other alternatives for integrating human-friendly names into sovereign identity infrastructure.

1. **Sovereign identity directories**. Essentially a "white pages for sovereign ID", these are search engines that specialize in indexing sovereign identities by human-friendly names and attributes. For example, you could search for "Drummond Reed, Seattle" or "Drummond Reed, XDI" and quickly narrow in on likely matches.
2. **Linked local names** are explained in this paper by Christopher Allen written for the first Rebooting the Web of Trust Design Shop in November 2015. They are human-friendly names that are not globally unique, but locally unique for each principal. For example, if your sovereign identity agent was connected to Christopher Allen and had permission to query his linked local name list, you could ask him for "Drummond" and receive back the DID or CID he calls by that name.

Conclusion

The following table summarizes the properties recommended for sovereign identity namespaces. It also illustrates why the difference between using aliases vs. new HFN namespaces are minimal enough that the benefits of the former outweigh the disadvantages of the latter.

Identifier Property	DID	CID	Alias	HFN

Globally unique	Yes	Yes	Yes	Yes
Decentralized	Yes	Yes	No	No
Immutable (need not expire)	Yes	Yes	No	No
Cryptographic	No	Yes	No	No
Can bootstrap discovery	No	No	Yes	Partial
Human-friendly	No	No	Partial	Yes