Identity Net: Building an identity net through self-authenticated data graphs

Author: Christopher Chung

The utility for an identifier allows any third-party entity to be able to identify and assert attributes, claims or otherwise data about some other entity. These identifiers are used almost always in relation to other data points whether it is other identifiers or descriptors. This is observed in the discussions around functional and contextual identity, and in the developments with VCs and DIDs. It becomes clear that identifiers themselves are meaningless without such associated data that describes it, providing the much needed contexts that which a third-party entity uses to evaluate a counterpart.

By modelling an identity as a set of data points and establishing connections between each data point through witnessing an authentication of those data points, we can construct a complex graph of relations and derive a net of identities that can rebuild a web of trust.

Identity as data points

As its most basic, identity is simply a construction of several data points. Some of these data points may be used as an identifying component to allow discerning entities a reference point for identifying other entities. Using DIDs as an example, they have an `id` and reference a `subject`, a `controller`, and an `authentication` method to name a few properties. On its own, the identifier (`id` in DIDs) does not provide any real reference without much needed additional data to give context on how the identifier should be used and what it describes (DID documents). For example, a full name does not necessarily equate to an identity, but is often commonly used as a default reference for someone's identity, such that referring to Christopher Chung automatically assumes one version of the entity behind the name. However until you collate multiple other data points such as date of birth, only then can you begin to construct a stronger notion of identity, as a data net.

An identity in this case is made up of a net of data points. Observing a collection of these data points together gives you a view on such an identity and can be identifiable by any of the data points that it is composed of (public key, social security number, passport number, name, etc.) depending on what a discerning entity will need. By making a single data point the lowest common denominator we remove any reliance on existing trust architectures or assumptions on how data gets verified and sourced and how that data is structured with relation to other pieces of data.

Instead, we make data verifiability a first class citizen and define the notion of *Data Element*.

Data Elements

A Data Element is a data point that also defines its own authentication method for that data point. Essentially it allows a third-party observer of an identifier to be able to verify the authenticity of that identifier. Similar to public and private key pairs, the private key proves ownership or control over the public key. In this way the public key component is a public identifier, whilst the private key is a secret material that can 'authenticate' against the public key. This symmetry is adopted in data elements where there is a public identifier component and a private authentication component where the protocol or process to authenticate is also publicly defined so it can be verified/authenticated by any other entity. The authentication method itself can be defined freely and verifiers can decide whether or not to trust outputs from such a method i.e. publicly verifiable protocols such as DSA are stronger than centralised assertions about data.

When a data element is authenticated, the identifier's authenticity or ownership is verified. By itself, it simply acts as a self-referential data point that can prove its own authenticity, similar to a self-signed certificate. It's only when you intersect multiple data elements does identity begin to emerge.

Data elements can then be used to construct *Data Compounds* where multiple data elements are authenticated within the same session, similar to 2FA, making a connection between these data elements. The result is a *Data Compound* where one data element is related to another through the dual-authentication of both elements. This allows you to then be able to build identity graphs by connecting your public key to your phone number for example, by signing an SMS OTP with your private key. This can be verified and publicly shown that your phone number and public key are connected and belong to the same identity (or pseudonymous entity). By creating a sequence of dual-authentication of elements, you can make indirect connections between many data points, or heighten the authentication level by creating plural-authentication of n-elements, you can make direct connections.

A discerning party can then identify you using either your phone number OR your public key instead of the current proprietary systems that rely on custom top-level identifiers that fragment data. This choice of identifier also depends on what connected data might be of interest in an entity's identity net and which sources of data are trusted by them or not. Some entities may choose not to trust the authentication method of certain data elements as they can be centralised processes.

It's important to note that some data points exist as strict subsets of others, for example, the account state of the Ethereum blockchain relies on the top-level identifier of the Ethereum address, so account balances and other state exists within the Ethereum domain under each address. This can still be modelled but would exist as a data compound relating to both the proof of such data (merkle proofs with miner/validator signatures) and the identifier itself (ethereum address -> public/private key pair). For more centrally stored data such as

your Google account data, a similar structure would be adopted except using a relation between Google signature to your Google account public identifier (email address) which will have means for authentication (proving access to your inbox via password).

In this way, we are able to elegantly express all data as a network of data points and construct expressive relationships between each data in order to build identity.

Webs of Trust

Armed with the ability to authenticate arbitrarily complex data compounds, identities can begin to emerge as we witness connections being made between various different types of data (full name, address, ethereum address, etc.), each capable of being the primary identifying component of the identity. With this, identities can assert or witness other entities and make guarantees about each other and publicly display such relationships. For example, I can sign an assertion that I have met the owner of another public key, and through that assertion provide a guarantee made by myself that public key A belongs to John Smith. Now trusting this assertion is a different question, and depends on who else believes assertions about me, but these webs of trust can be built through identities making assertions about each other similar to how keys were exchanged in the past.

We can start to rely less on de facto standards for trust and roll back to a more subjective view of trust through only relying on a clear set of connections through which we based our decision making. After all, trust is a choice, and identity nets can help inform our choice of counterpart in increasingly adversarial interactions.