A Manifold Paper

Distributed Identity: The future of identity.

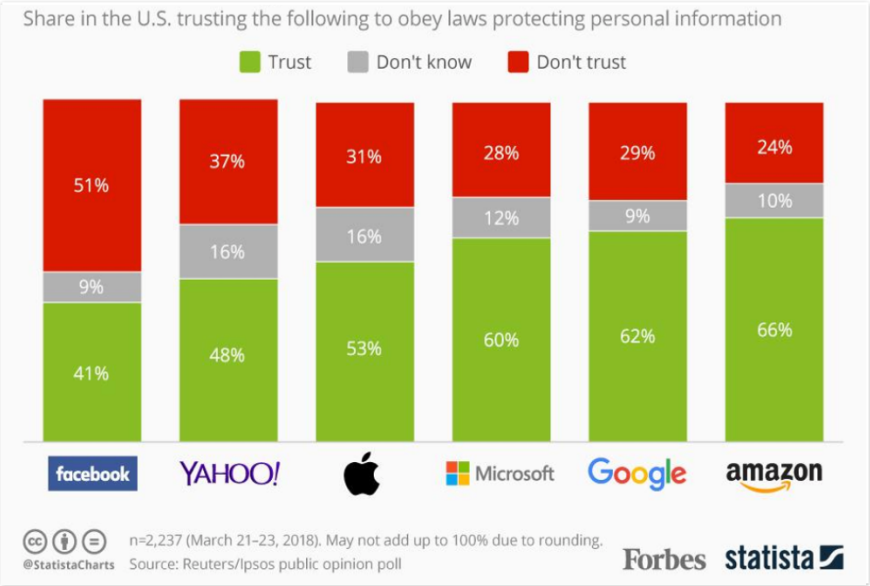
Introduction

Identity has always been a challenging problem. How does a person prove they are who they say they are? The specifics of the solutions vary across time and cultures, but the heart of the problem will never change: one cannot simply trust another's word. And so, the heart of the solution is always the same: one must trust an oracle.

Technology cannot remove the need for an oracle. To date it has been used solely to ease access to an oracle and that ease has created something of an arms race over who will be The Oracle. Federated identity technologies have actually centralized identity and, in so doing, removed individuals from the system. People log into everything with Google, Facebook, LinkedIn or Amazon; they are the sum of what any one of those companies says they are.

Distributed Identity is a technology designed to eliminate that centralization and give ownership of an identity entirely to the person. And not just for secure logins. But to make all components of an individual's identity: citizenship, age, eye color, credit score - secure, fully under their control, as easy to access as it is to "Login with Facebook," and with no certificate authorities needed.

Regardless of how well-intentioned a centralized identity provider may be though, people desire and deserve more control over their identifying data. Distributed Identity eliminates the need to trust a corporation. Individuals are once again an essential component of the system.



Corporate trust deficit

Concepts

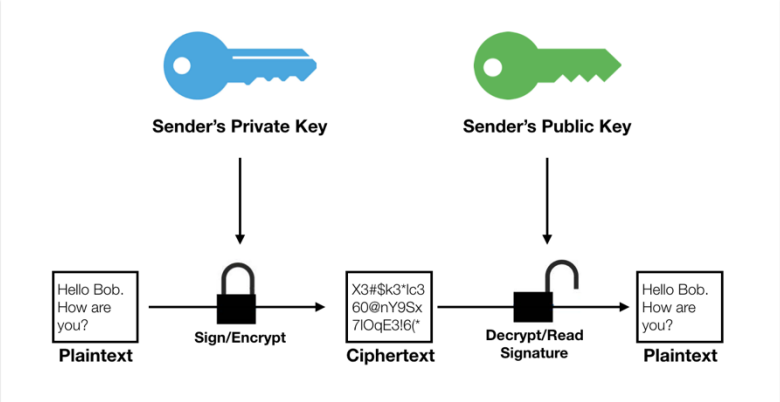
While some familiarity with identity technologies and challenges will be assumed, the following are fundamental to properly understanding Distributed Identity.

Public Key Infrastructure

Two keys can be created such that one can be used to prove ownership or control of the other. In fact, one can be used to encrypt data such that only people with the second can read it. And

this technique can be used to “sign” something. To prove that the owner of the other key created this specific data.

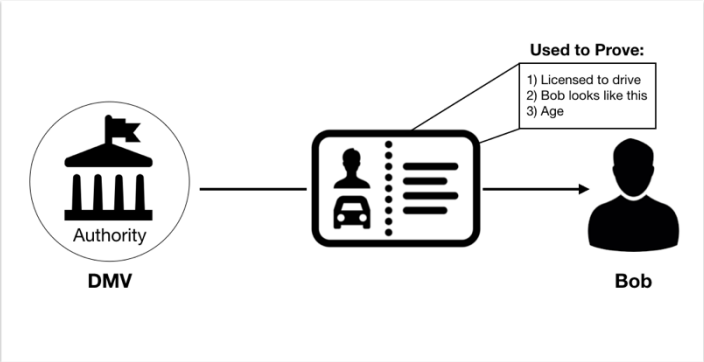
The keys are often referred to as ‘public’ and ‘private’ in a shorthand to refer to the key shared publicly, vs the one kept private. The public key, that everyone has access to, can be used to prove that signed data could only have originated from the owner of the corresponding private key.



Digitally signing a message

Authority

This is the oracle for some piece of information. The Department of Motor Vehicles, for example, is the authority on whether someone is legally allowed to drive or not. One’s license tends also to be used as the authority for other information such as birth date, citizenship, etc. Though it is not the only, or even primary, authority on such information.



Authorities serve as the oracle for discrete data

Wallet

This is simply where identifying information is stored. In the real world, it’s where one keeps their driver’s license and other IDs. In the digital world, similarly, it is simply a file with all of one’s identifying information collected securely and encrypted. When speaking about ‘providing’ or ‘collecting’ information and data, this is the source or destination, respectively.

Wallets securely store identifying information

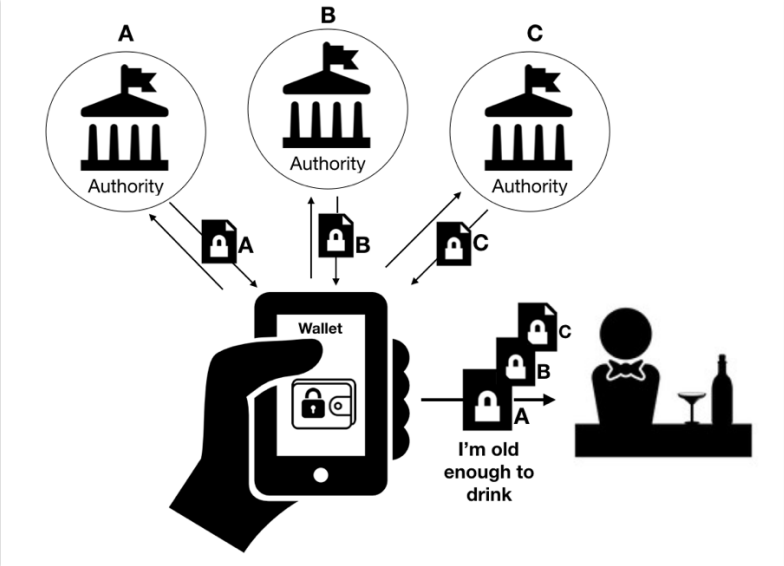
APIs, Protocols, Routing, Transfers

These are all important for actually *doing* everything discussed here, but they are unnecessary for understanding what is being done. Of course, more details happily available upon request.

High Level Flow

At its most basic, Distributed Identity works like this:

A person creates their wallet. They then contact different authorities, say the DMV, and request their personally identifying information. Their age, for example. The authorities provide these in discrete, provable, packages. When one wants to prove something about themselves, exactly as they would with a physical driver's license, they simply pull the appropriate package out and share it with whomever needs to see it.



Distributed identity flow

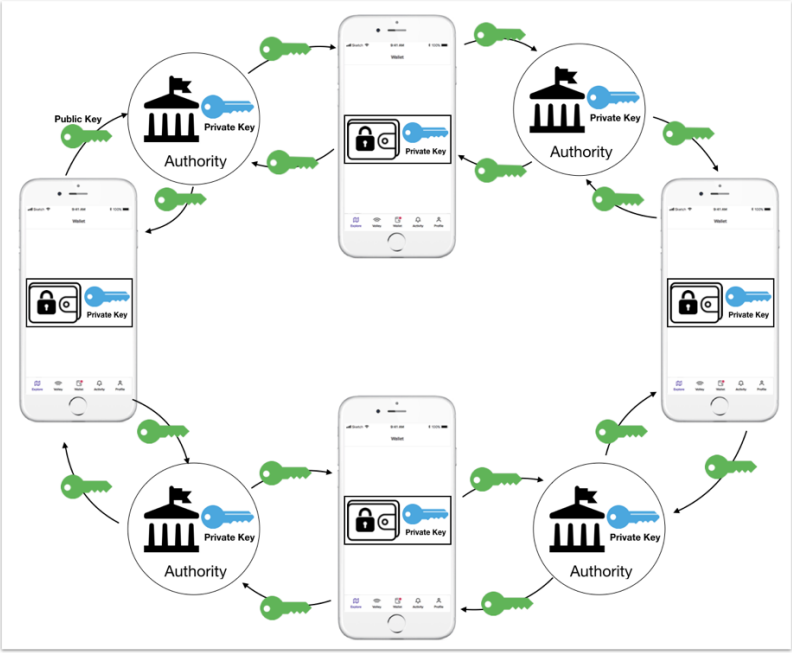
Keys

One of the primary challenges, and failures, of public key infrastructure, is that it requires a certificate authority. Someone to issue and hold all of the public keys, vouch for their authenticity, and manage their revocation and replacement. This is untenable both because it is actually a very challenging technical problem, but also because it effectively centralizes the system. The certificate authority is, effectively, The Authority.

Distributed Identity turns the whole problem inside out. Each party becomes responsible for managing their own keys. Managing one, or a very small set of keys, is a relatively simple technical problem though there are two distinct facets: people and authorities.

For people, one might imagine an, or many, interface(s) which would allow them to create, review, and secure their wallet as well as use it appropriately for actual identification purposes. Authorities need to manage a slightly more complex, but easily abstracted, system wherein they use their keys to sign all information transmitted to people. Both can then trivially share their own public key(s) directly to all who query for them.

Certificate authorities don't simply solve the problem of distribution though, they also solve problems of "how do I know this is really you?" and revocation. In true distributed fashion, these problems need to be separated in order to be solved efficiently.



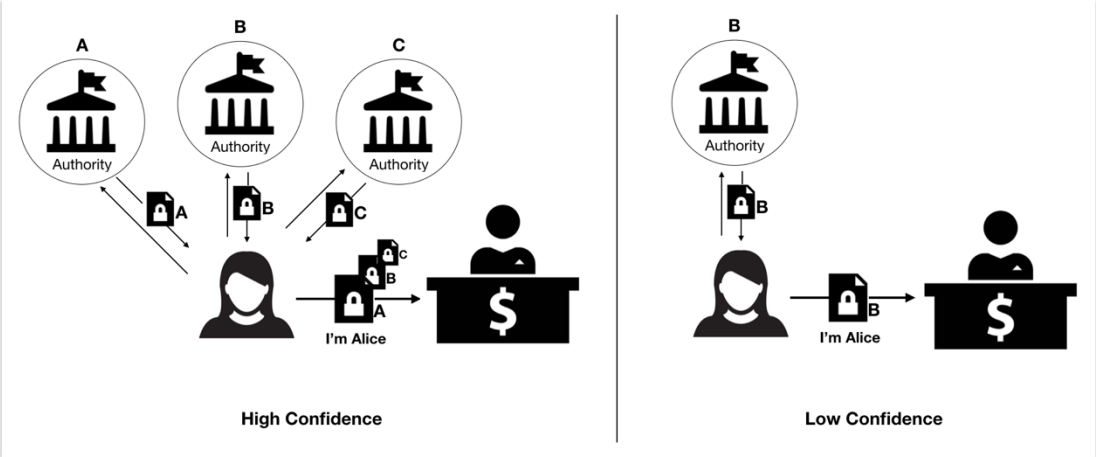
Decentralized key management

How do I know this is really you?

Well, you don't. Whether it's a centralized certificate authority or a Distributed Identity situation, ultimately you have to trust an oracle. Or many. As part of the request for identity information, a person submits their public key for signing. Each little package that comes back contains a piece of identity information, a signed timestamp (more on that in a second), and a signed copy of the person's own public key.

When a person shares one of these pieces of information, their license to drive for example, what they are actually sharing is effectively, "on this date, this DMV attested that I am both who I say I am and legally allowed to drive." In the case where two forms of ID may be required, remember that every information package, from all authorities, includes the person's public key signed by that authority. It could be Google, the bank, and the DMV all attesting that "I am who I say I am" should that level of certainty be required.

In many ways this system is stronger than a centralized authority because multiple independent authorities are independently attesting to this being a single person. But, as outlined so far, it dramatically complicates key revocation.



Identity is established through confidence in attestations

Compromise, Loss, and Revocation

Loss

Key loss is fairly straightforward. Either the person put in place a backup/recovery option, perhaps a curated service, maybe just a copy of it on a thumb drive in a drawer, or they did not. In order for the system to remain secure, catastrophic key loss must result in the person starting from scratch. Much like they would after a house fire claimed all of their identifying documents.

An authority would follow their protocol to revoke and refresh their key rather than simply being able to start from scratch.

Compromise

Compromise is more complicated. In the case of a person, compromise of the key would actually need to be a compromise of the entire wallet. Similar to obtaining access to someone's primary email, an attacker could then systematically gain access to, and use all of, the person's identifying information. This is unfortunate, but again unavoidable if the user is to have control of their identifying information.

In the case of an authority, current best practices turn this into an all hands security disaster which must be fully addressed and taken care of **immediately**. Once the key has been compromised, no one can be certain whether data signed by that key actually came from the owner or not.

Unless you know *when* the signing occurred.

Time

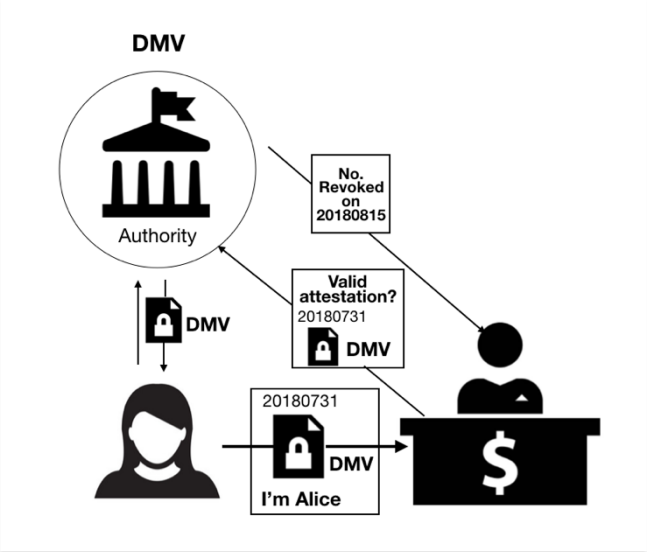
If a person's key is compromised today, it stands to reason that data signed by them last week was actually signed by *them*. If it could be proven that the signing occurred during a time when the key was fully secured, the attestation would stand. This is the reason for each data package

to incorporate a signed timestamp. With such a timestamp in place, the only immediately necessary step is creating a new key pair and updating the key history appropriately. Everything else can then be handled in due time.[1]

Revocation or Refresh

In the event of revocation or refresh, each individual piece of data doesn't need to be immediately resigned and re-issued. The authority simply needs to keep a record of when various keys were valid. It is then part of the responsibility of the challenger (the one who is asking for identity information) to do as much diligence as is needed to satisfactorily accept, or reject, the validity of the attestation (current libraries are very diligent about this, assuming they have the connectivity necessary to be so).

When a person, or more accurately the wallet software the person is using, is informed that one of the keys they are using for attestation has been revoked, the software can then (in most cases[2]) automatically retrieve a refreshed signature. This generally needs no user intervention as the authority will always accept its own signatures, so long as they are within a valid time period, as proof that the user should have a freshly signed copy of the data.



Challenger queries authority for validity of attestation

Conclusion

Centralization, to include certificate authorities, is not required to enable a robust identity infrastructure. Proper distribution of the necessary components of an identity solution fully addresses current and future needs, the security requirements of such, and does so in a way that also minimizes the technical complexity for each individual participant.

Distributed Identity is the future of identity.

[1] An important security consideration, though not otherwise critical, is that the timestamp authority cannot be the same authority attesting to the data. The data authority must request an independently signed timestamp and include that in the package they return to the originator.

[2] The only reason a user would need to intervene is if the authority was unwilling to accept its own, or any other authorities', attestation that the user is who they say they are. And, in that case, the user will need to either log in or otherwise prove themselves via "old" channels. This should be quite rare.