Carnegie Mellon University

95-733 Internet of Things
IOT and Self Sovereign Identity

Schneier on Identity Theft

"I could give you advice like don't stay at a hotel (the Marriott breach), don't get a government clearance (the Office of Personnel Management hack), don't store your photos online (Apple breach and others), don't use email (many, many different breaches), and don't have anything other than an anonymous cash-only relationship with anyone, ever (the Equifax breach). But that's all ridiculous advice for anyone trying to live a normal life in the 21st century.

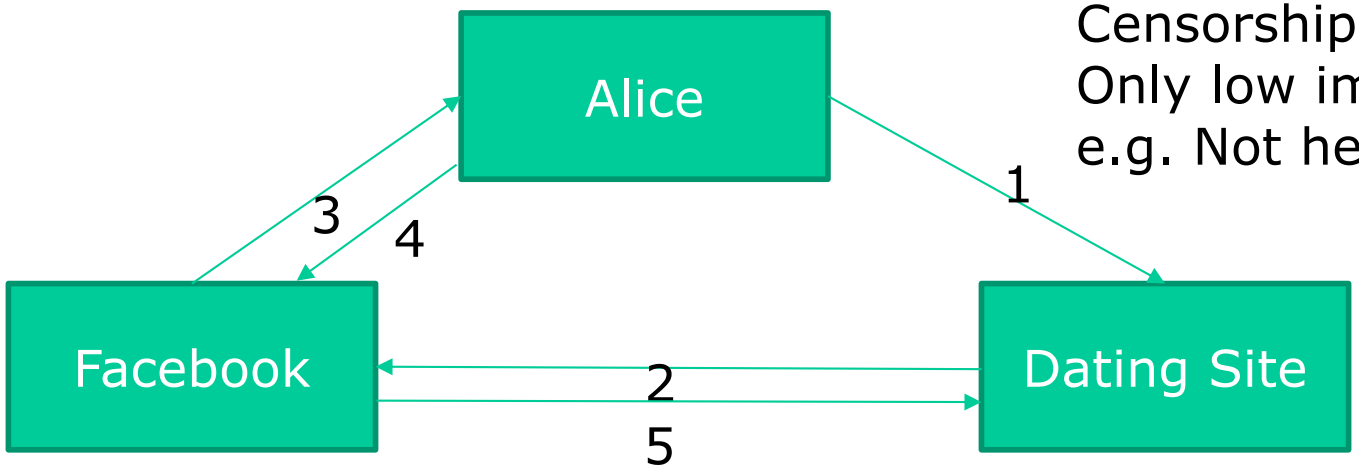
The reality is that your sensitive data has likely already been stolen, multiple times."

See https://www.schneier.com/tag/identity-theft/

Cybercrime & cloud computing

- People have many online personas at many organizations.
- Each organization we interact with must store our personal information in massive databases. These 'silos' become gold mines to hackers and toxic liabilities for anyone obligated to store the data.
- Many of these organizations maintain data about people. These data maybe correlated and stolen or sold.
- Trust at a distance is tricky. One must provide provide personal details to a service provider before the service provider trusts that it is really you.
- Some of this data ends up correlated and on the dark web.
- Cybercriminals love this!

Currently



Concerns:
Controlled by others
Surveillance
Censorship
Only low impact use case
e.g. Not healthcare

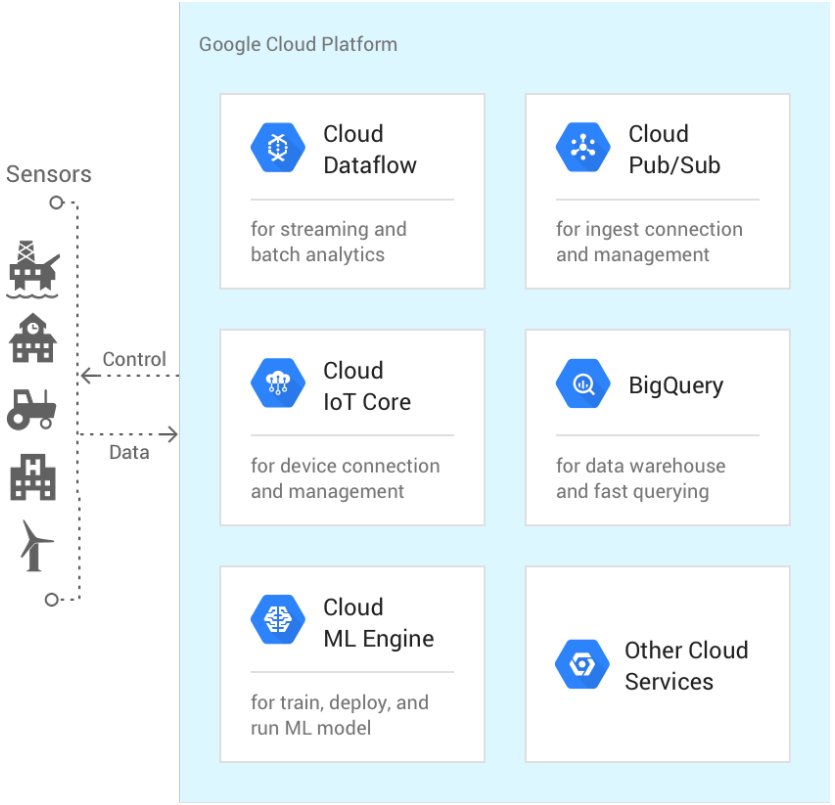
Self-sovereign Identity (SSI) offline

The credential may be used for other activities.
The credential may be used to obtain additional credentials.
The tavern need not contact the DMV to verify the credential.
The tavern trusts the DMV to provide a valid birth date.
A CMU ID (based on MIFARE from NXP uses NFC within 10 cm) may be presented to an instructor to show you are eligible to take an exam or placed on a reader in a bus to show your organization has paid for the ride.
See: https://www.nxp.com/video/tap-into-mifare:TAP-INTO-MIFARE
It is up to the verifier to decide whether the credentials are acceptable.

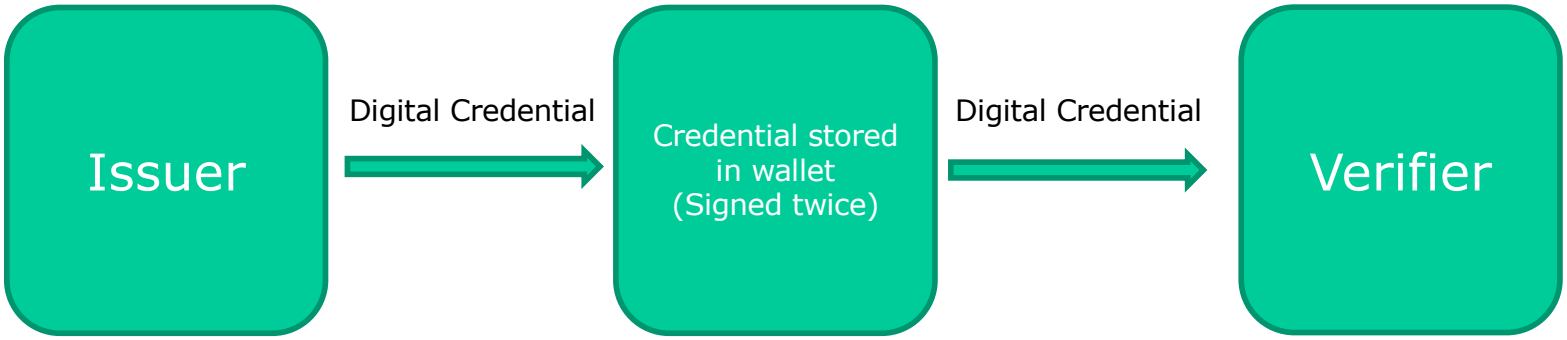
Google's IOT cloud reference architecture

Not shown here but Google requires that every device store a private key and every JSON message be signed with that key.

Google keeps a copy of the public key to verify the signature.

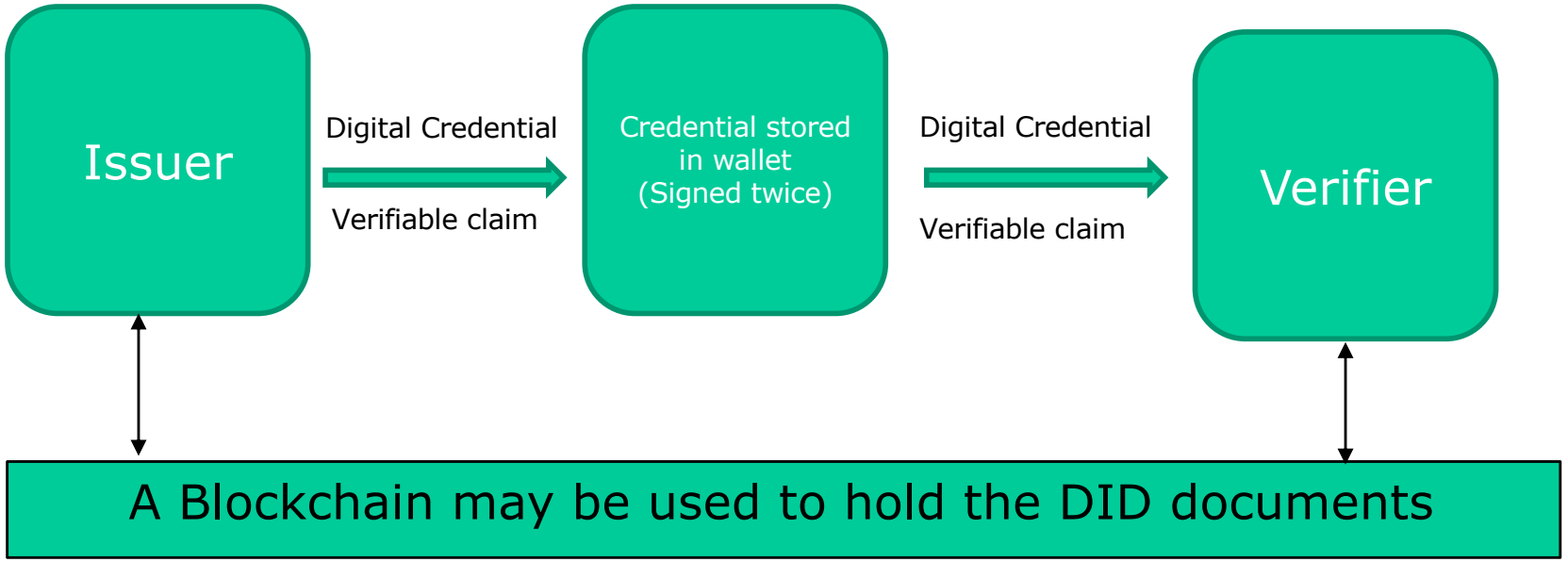


Self-sovereign identity online



Players are identified with decentralized identifiers (DID's).
did:v1:DwkYwcoyUXHNkpj3whn4DgXB4fcg9gj95vKxYN2apkZD
You can prove that a particular DID is yours. (Unlike SS#'s)
DID's resolve to DID documents holding public keys.
No central authority controls the DID or DID document.
The public keys are needed by the verifier to verify the signatures.
Where does the verifier get the public keys from?

Self-sovereign identity online

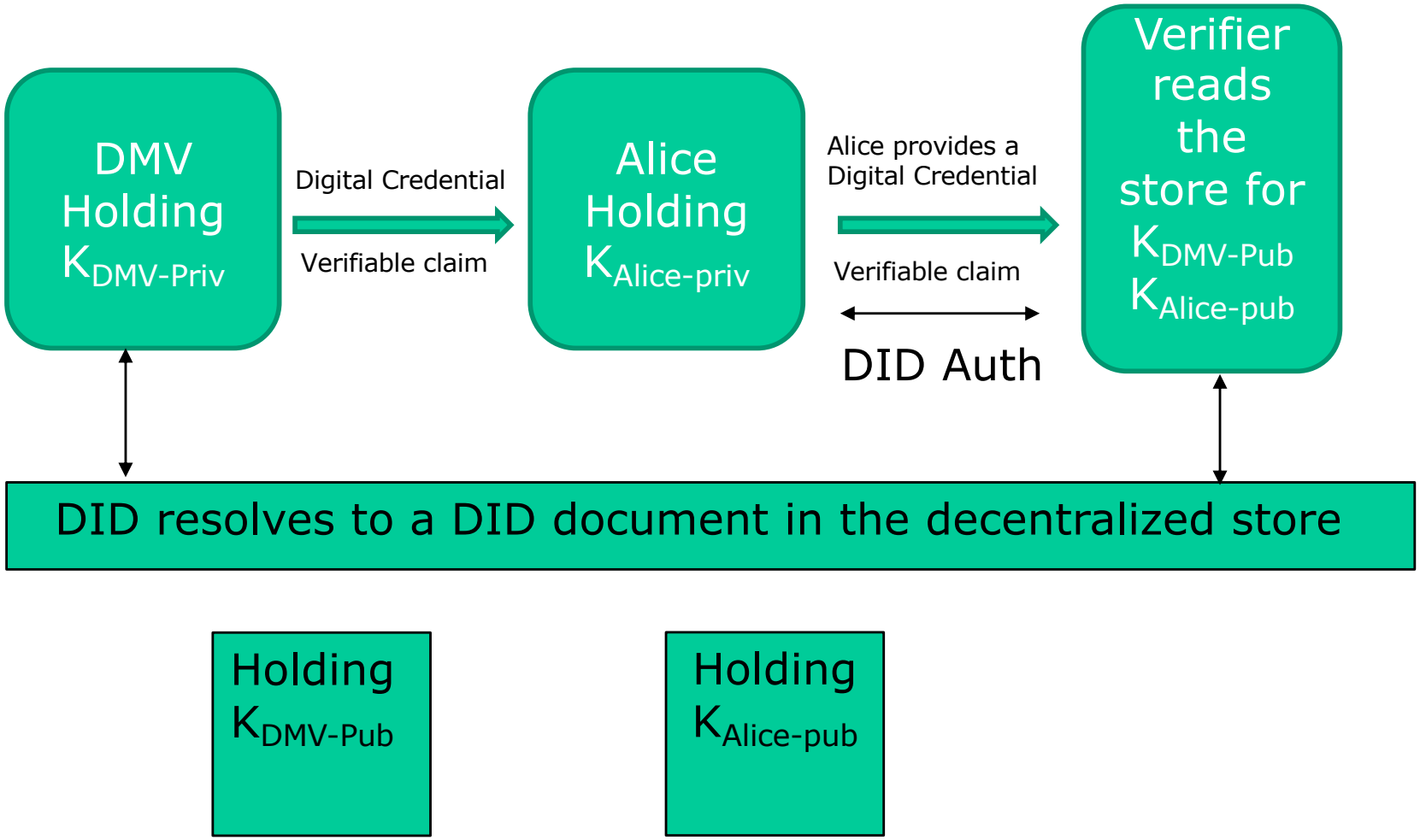


On the Sovrin blockchain, anyone may read but only authorized players may write. Reading the chain is fast and cheap. Writing is slow and typically requires a payment.

No credentials are stored on the blockchain. No PII either.

All the credential sharing occurs off ledger – on the edge. This scales well. W3C has standard credentials, did's, and did documents.

Signatures checked with public keys



This is decentralized PKI.

A Verifiable Claim

EXAMPLE 3: A simple verifiable claim

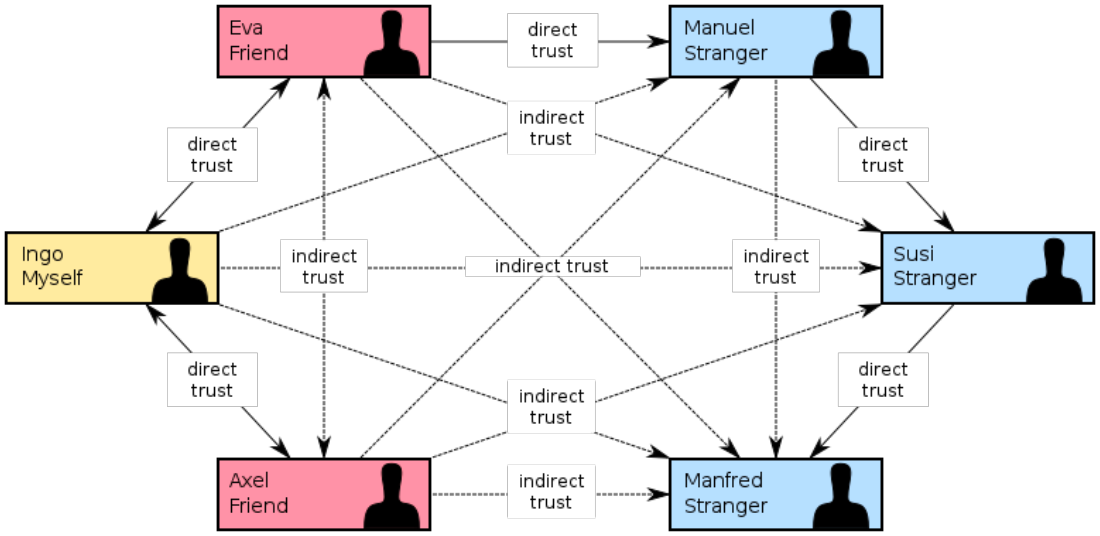
```
{
  "@context": "https://w3id.org/security/v1",
  "id": "http://example.gov/credentials/3732",
  "type": ["Credential", "ProofOfAgeCredential"],
  "issuer": "https://dmv.example.gov",
  "issued": "2010-01-01",
  "claim": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "ageOver": 21
  },
  "revocation": {
    "id": "http://example.gov/revocations/738",
    "type": "SimpleRevocationList2017"
  },
  "signature": {
    "type": "LinkedDataSignature2015",
    "created": "2016-06-18T21:19:10Z",
    "creator": "https://example.com/jdoe/keys/1",
    "domain": "json-ld.org",
    "nonce": "598c63d6",
    "signatureValue": "BavEll0/I1zpYw8XNi1bgVg/sCneO4Jugez8RwDg/+
    MCRVpjOboDoe4SxxKjkCOvKiCHGDvc4krqi6Z1n0UfqzxGfmatCuFibcC1wps
    PRdW+gGsutPTLzvueMWmFhwYmfIFpbBu95t501+rSLHIEuujM/+PXr9Cky6Ed
    +W3JT24="
  }
}
```

A Minimal DID Document

EXAMPLE 2: Minimal self-managed DID Document

```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:example:123456789abcdefghi",
  "publicKey": [{
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "RsaVerificationKey2018",
    "owner": "did:example:123456789abcdefghi",
    "publicKeyPem": "-----BEGIN PUBLIC KEY...END PUBLIC KEY-----\r\n"
  }],
  "authentication": [{
    // this key can be used to authenticate as DID ...9938
    "type": "RsaSignatureAuthentication2018",
    "publicKey": "did:example:123456789abcdefghi#keys-1"
  }],
  "service": [{
    "type": "ExampleService",
    "serviceEndpoint": "https://example.com/endpoint/8377464"
  }]
}
```

How are public keys verified?



The Web of Trust is back

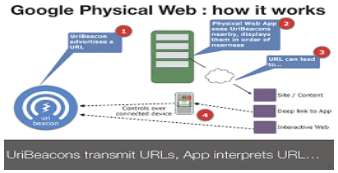
Suppose Axel (bar owner) trusts Ingo (DMV).
Ingo signs a credential for Eva. Axel trusts Eva indirectly.
The verifier decides what credentials to honor.

Things hold credentials

Inspection sticker

License plate

Water heater inspection



URL beacons (signed)

Hypothetical: SSI and IOT

- Google wants to know if a particular device has been recently inspected.
- Google requests the inspection credential from the device. The device provides its credential and keeps secret its private key.
- Google retrieves the DID documents to access the public keys. (one for the device and another for the inspector).
- Google challenges the device and the device proves to Google it is in possession of the corresponding private key.
- Google trusts this particular inspector.
- The device is approved for data acquisition.

Safe credentials: five requirements

1. Prevent correlation by decoupling issuers and verifiers. The verifier should not need to contact the issuer. Otherwise, the issuer will know how you are using the credential.
2. If you show the same signature to every verifier, the signature provides correlation. The signature is unique to you. Only share a proof that the issuer signed your credential. Do not share the signature.
3. Credentials should be portable and interoperable. Avoid identity silos.
4. Enable flexibility and data minimization. Be able to prove a value without sharing the value. Be able to combine attributes from several credentials into a single proof. Use zero-knowledge proofs.
5. Ensure that trust goes both ways. An individual can verify the authenticity of an organization and vice versa. You should be able to verify that you are communicating with your bank.

From Evernym

Digital ID: systems and standards

- Each of these supports decentralized, self-sovereign identity but differ in how claims are issued and presented.
- Sovrin (Hyperledger Indy)
- uPort (Ethereum based, ERC 780)
- Veres One (Built for purpose blockchain)
- Microsoft Identity Overlay Network (on Bitcoin)
- Standards for decentralized identifiers and verifiable claims are being developed to provide interoperability. See W3C Verifiable Credentials WG.
- Trust over IP Foundation (MasterCard and Evernym)
- ID2020 Alliance
- Watch this space.