OPEN REPUTATION: PORTABLE REPUTATION AS A PRIMAL USE-CASE FOR DECENTRALIZED IDENTIFIERS

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/open-reputation-low-level-whitepaper.pdf

2017

Samuel Smith Ph.D.
sam@xaltry.com

REPUTATION

noun:

The estimation in which a person or thing is held, especially by the community or the public generally.

root:

Latin word reputāre, which is equivalent to re + putāre, that is, to re-think or re-consider.

usage:

A considered evaluation (measure) of past behavior used to predict future behavior.

qualification:

Confidence improves with contextual simularity.

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/open-reputation-low-level-whitepaper.pdf

WHAT IS REPUTATION?
WHAT IS REPUTATION AI?

Contextual predictor of future behavior to enable a transaction

Closed-loop automated reasoning, not just open-loop pattern recognition

Means to filter and modulate transactions

Curator, recommender, decision aid, IA

Contextual predictors are more powerful

Behavior based predictors are more credible

Transitive predictors are more portable

COMPUTATIONAL REPUTATION

Computational generation of a reputation is to aggregate relevant instances of behavior.

Instances of behavior = reputational events or reputes for short

Reputation measures are inferred indirectly from reputes associated with an entity

Contrasting Example:

* Entities provide direct ratings of promptness of another entity

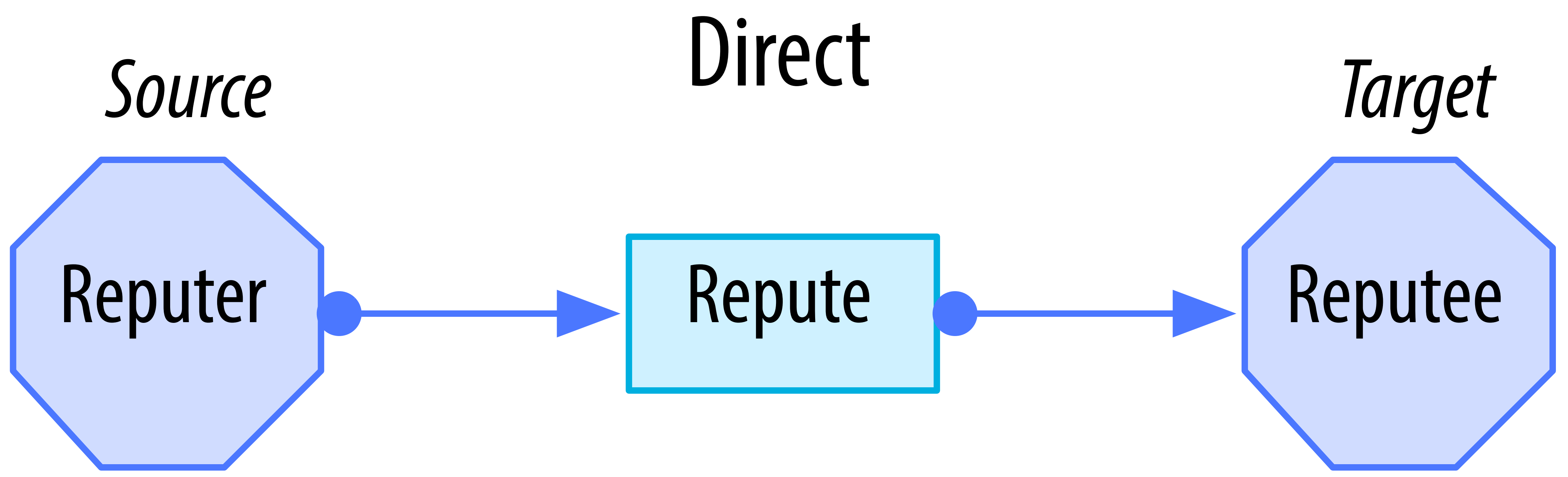
* Collect instances of behavior of entity in context from which promptness can be inferred.

Reputation from reputed behavior allows for re-scoping, re-weighting, re-combination, and re-evaluation of collected reputes

Enables arbitrary levels of nesting, precision, and granularity in the data aggregation process

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/open-reputation-low-level-whitepaper.pdf

REPUTING LEXICON



Repute = Reputational Event (Data)

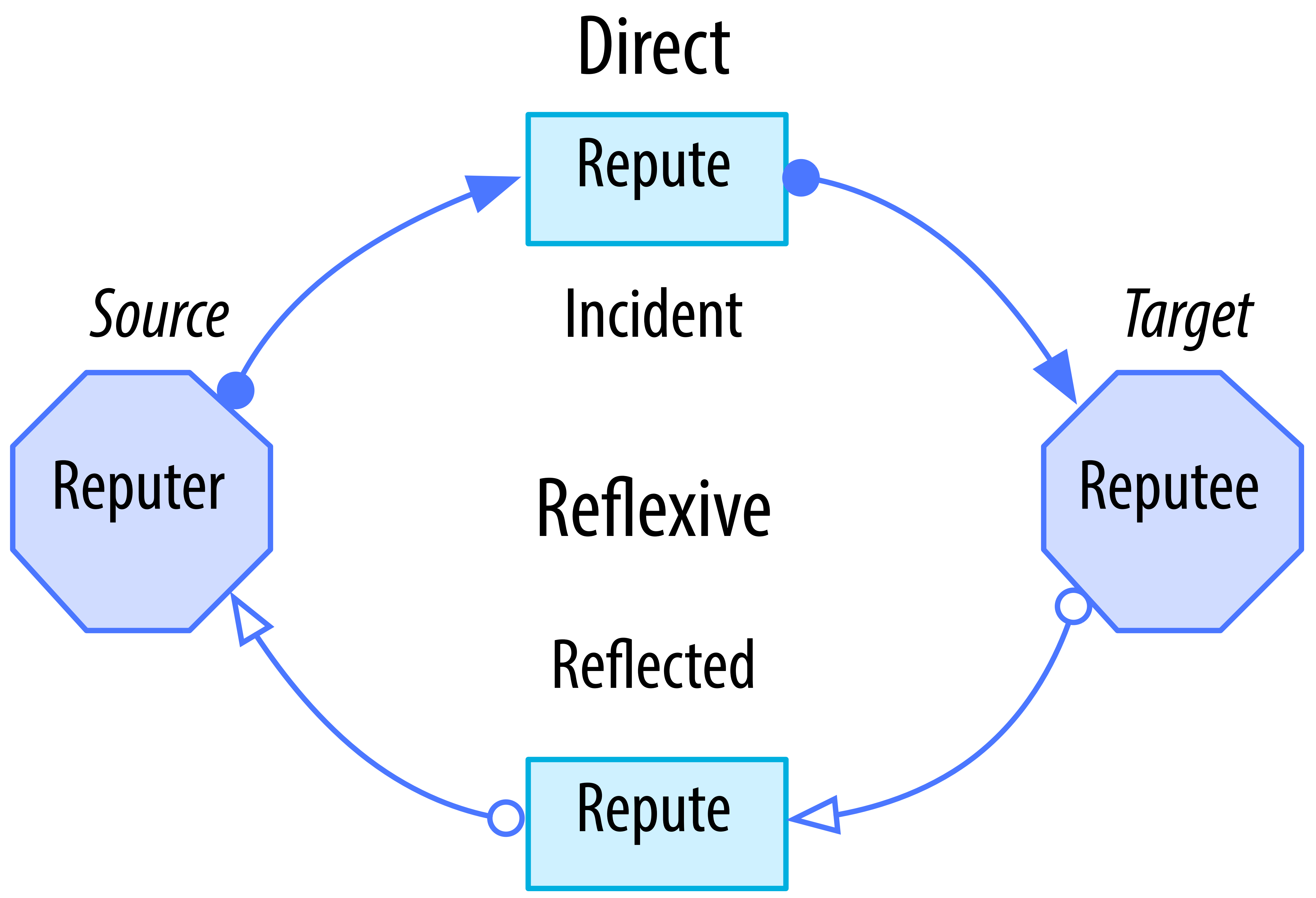
Reputee = Reputational Entity, (Identity)

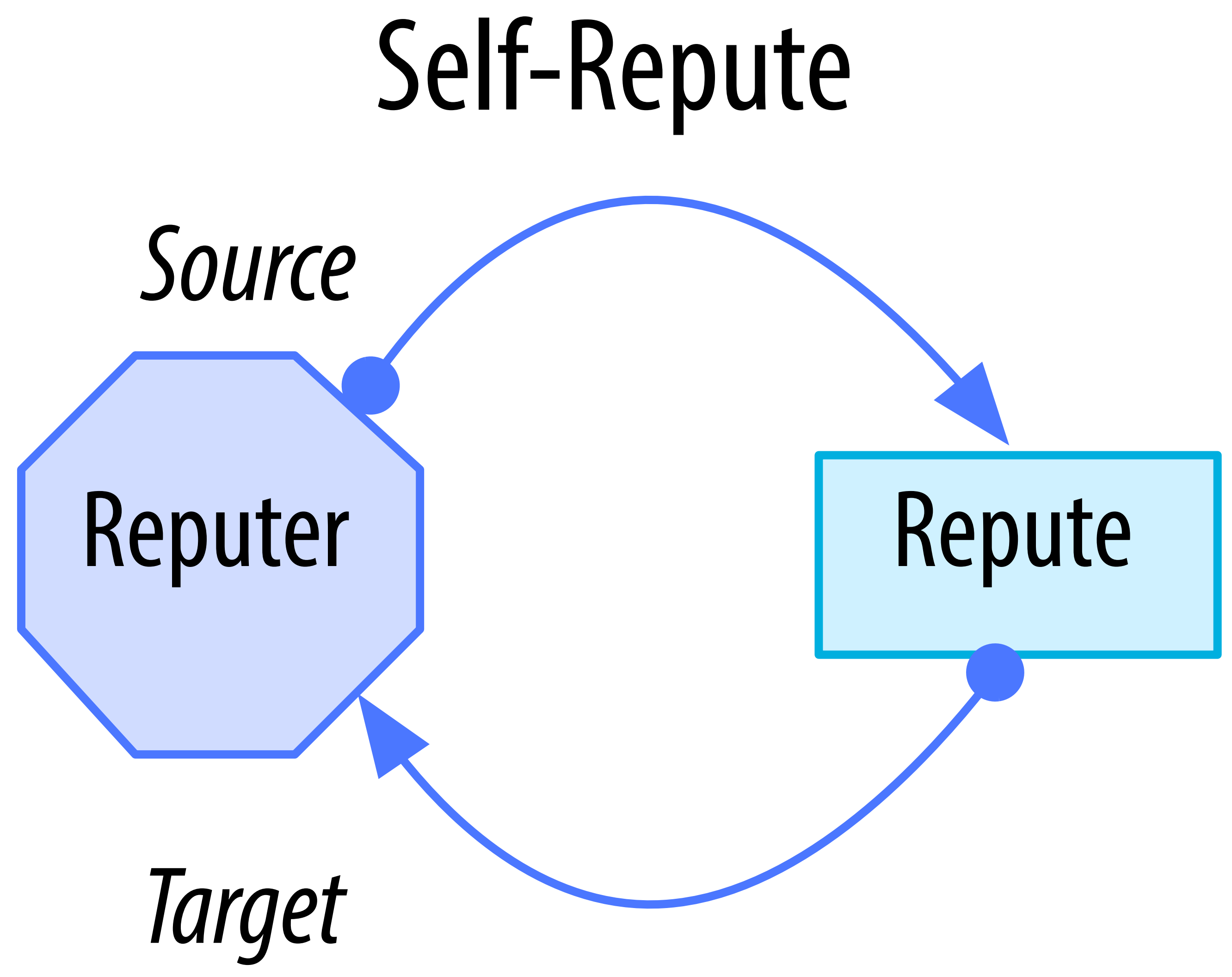
Target of Repute

Reputer = Source of Repute

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/open-reputation-low-level-whitepaper.pdf

REFLEXIVITY





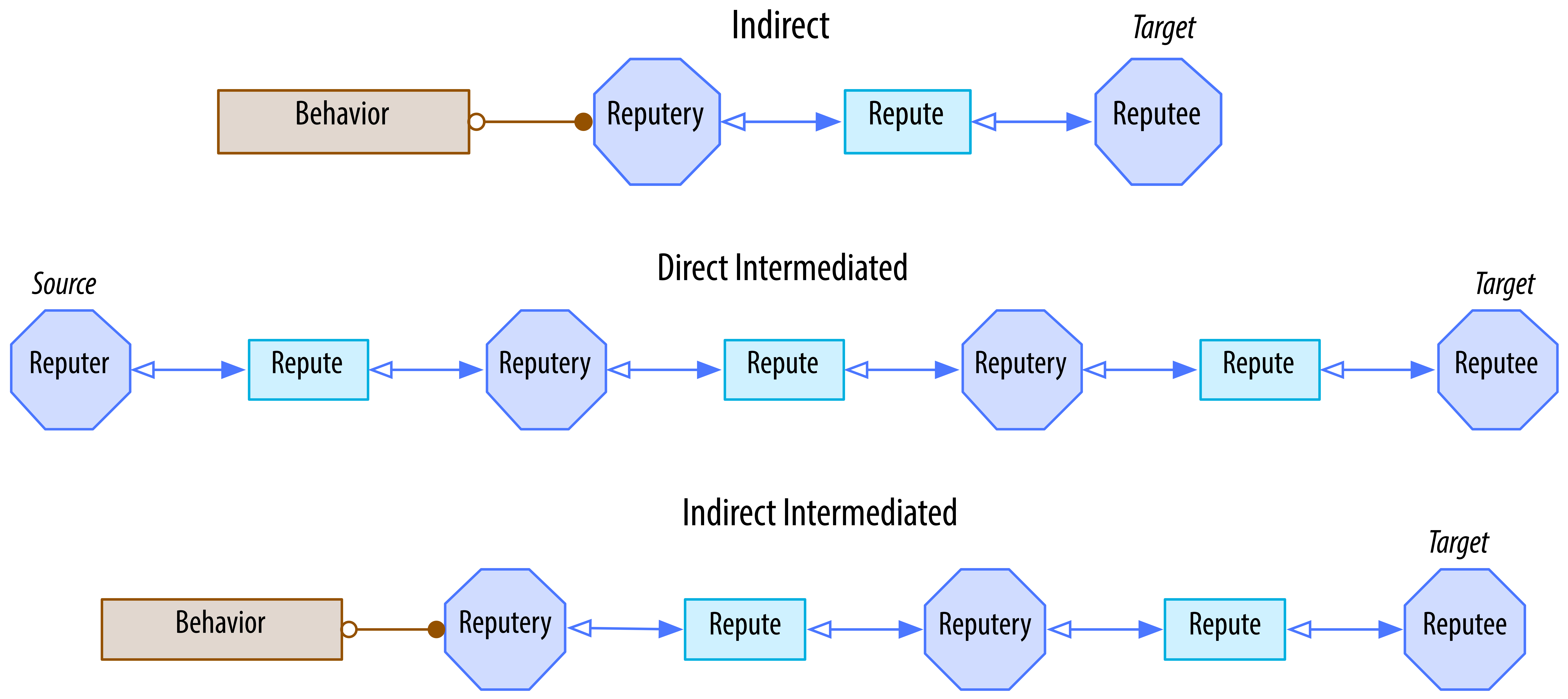
Reputation is Reflexive:

Reputer is simultaneously both a Source and a Target = Reputee

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/open-reputation-low-level-whitepaper.pdf

INDIRECT AND INTERMEDIATED REPUTING

Reputery = Indirect Source (Implied Reputer) or Intermediate Source



https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/open-reputation-low-level-whitepaper.pdf

REPUTAGE

Reputage: *reputational event ancilliary data. Optional or infrequently used information associated with a repute separated from core repute for performance reasons.*

https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/open-reputation-low-level-whitepaper.pdf

REPUTET

Reputet: reputational event transaction. A cryptographically signed and validated transaction of reputes between reputees or reputeries.

Initiator = first party.

Copartent = counter party.

Arbiter = trusted third party (notary).

REPUTE

```
{
  "ruid": "bcd456",
  "stamp": "2015-03-19T10:30:45Z",
  "reputee": "z4def6",
  "reputer": "5efa75",
  "curator": "2bcd4",
  "signer": "2bcd4#0",
  "detail":
  {
    "rating":
    {
      "useful": 90,
      "fair": 80,
    },
    "url": "http://myblog.com/article19/"
  }
  "tags": {},
  "reputage": null,
}
/r/n/r/n
"abcdef987654321"
```

```
{
"puid": "abcefg",
"ruid": "bcd456",
"stamp": "2015-03-19T10:30:45Z",
"comment":
{
  "cuid": "1234abc",
  "url": "http://myblog.com/article19/comment19",
  "contents": "You are so awesome."
}

}
/r/n/r/n
"abcdef9871234567"
}
```

IDENTIFIERS

UUID: Universally Unique Identifier RFC 4122: UUID type 1 -5

16 byte collision resistant decentralized identifier generated with random number generator and optional name spacing data

Enables distributed applications to create unique identifiers without central authority

Prefixed namespacing allows for sorting and searching properties such as time order, lexical order, nesting etc,

URI: Uniform Resource Identifier, URI: Uniform Resource Locator, URN: Uniform Resource Name RFC 3986

```
scheme:[//[user[:password]@]host[:port]][/path][?query][#fragment]
```

Self-Certifying Identifier: Contains fingerprint of public member of cryptographic public/private key pair

Decentralized Self-Certifying Identifier: public/private key pair is generated by user not central registry

Hierarchical Self-Certifiing Identifier: selfcertroot:/path/to/related/data

Tree Hierarchical Deterministic Self-Certifying Identifier: parent/child/child/child

Tupleizable (routable) Identifiers: /channel/host/process/data = (channel, host, process, data)

DID: DECENTRALIZED IDENTIFIER

https://w3c-ccg.github.io/did-spec/

```
did:method:idstring
```

```
did:rep:Qt27fThWoNZsa88VrTkep6H-4HA8tr54sHON1vWl6FE=
```

```
{
  "did": "did:rep:Qt27fThWoNZsa88VrTkep6H-4HA8tr54sHON1vWl6FE=",
  "signer": "did:igo:Xq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148=#0",
  "friend": “did:igo:Xq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148=/next/door”,
  "name": "John Doe",
  "zip": "94088"
}
```

DDO

https://w3c-ccg.github.io/did-spec/

DDO = DID Description Object

```
{
  "@context": "https://example.org/did/v1",
  "id": "did:example:21tDAKCERh95uGgKbJNHYp",
  "owner": [{
    "id": "did:example:21tDAKCERh95uGgKbJNHYp#key-1",
    "type": ["CryptographicKey", "EdDsaPublicKey"],
    "curve": "ed25519",
    "expires": "2017-02-08T16:02:20Z",
    "publicKeyBase64": "lji9qTtkCydxtez/bt1zdLxVMMbz4SzWvlqgOBmURoM="
  }, {
    "id": "did:example:21tDAKCERh95uGgKbJNHYp#key-2",
    "type": ["CryptographicKey", "RsaPublicKey"],
    "expires": "2017-03-22T00:00:00Z",
    "publicKeyPem": "----BEGIN PUBLIC KEY-----\r\nMIIB..
     ... sGbFmgQaRyV\r\n-----END PUBLIC KEY-----"
  }],
  "control": [{
    "type": "OrControl",
    "signer": [
        "did:example:21tDAKCERh95uGgKbJNHYp",
        "did:example:8uQhQMGzWxR8vw5P3UWH1j"
    ]
  }],
  "service": {
    "openid": "https://openid.example.com/456",
    "xdi": "https://xdi.example.com/123"
  },
  "created": "2002-10-10T17:00:00Z",
  "updated": "2016-10-17T02:41:00Z",
  "signature": {
    "type": "RsaSignature2016",
    "created": "2016-02-08T16:02:20Z",
    "creator": "did:example:8uQhQMGzWxR8vw5P3UWH1j#key/1",
    "signatureValue": "IOmA4R7TfhkYTYW8...CBMq2/gi25s="
  }
}
```

Guardian Managed DDO

```
{
"@context": "https://example.org/did/v1",
  "id": "did:example:21tDAKCERh95uGgKbJNHYp",
  "guardian": "did:example:8uQhQMGzWxR8vw5P3UWH1j",
  "control": [ "did:example:8uQhQMGzWxR8vw5P3UWH1j" ],
  "service": {
    "openid": "https://openid.example.com/456",
    "xdi": "https://xdi.example.com/123"
  },
  "type": "http://schema.org/Person",
  "created": "2002-10-10T17:00:00Z",
  "updated": "2016-10-17T02:41:00Z",
  "signature": {
    "type": "RsaSignature2016",
    "created": "2016-02-08T16:02:20Z",
    "creator": "did:example:8uQhQMGzWxR8vw5P3UWH1j#key-1",
    "signatureValue": "IOmA4R7Tf...3CBMq2/gi25s="
  }
}
```

DID FRAGMENT

DID Fragment

`did:method:idstring#keyindex`

A DID fragment *MUST* be used only as a method-independent pointer into the DDO to identify a unique key description or other DDO component.

```
{
  "owner": [{
    "id": "did:example:21tDAKCERh95uGgKbJNHYp#key/1",
    "type": ["CryptographicKey", "EdDsaSAPublicKey"],
    "curve": "ed25519",
    "expires": "2017-02-08T16:02:20Z",
    "publicKeyBase64": "IOmA4R7TfhkY...Mq2/gi25s="
  }, {
    "id": "did:example:21tDAKCERh95uGgKbJNHYp#key/2",
    "type": ["CryptographicKey", "RsaPublicKey"],
    "expires": "2017-03-22T00:00:00Z",
    "publicKeyBase64": "MIIBOg...mgQaRyV"
  }]
}
```

DID PATH

DID Path

```
did:method:idstring/path/to/associated/resource
```

```
did:method:idstring/path/to/associated/resource#05
```

If a DID reference includes a DID path followed by a fragment, that fragment is NOT a DID fragment.

```
{
  "did": "did:rep:Qt27fThWoNZsa88VrTkep6H-4HA8tr54sHON1vWl6FE=",
  "signer": "did:igo:Xq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148=#0",
  "friend": “did:igo:Xq5YqaL6L48pf0fu7IUhL0JRaU2_RxFP0AL43wYn148=/next/door”,
  "name": "John Doe",
  "zip": "94088"
}
```

VERIFIABLE CLAIMS

https://www.w3.org/TR/verifiable-claims-data-model/

Profile

```
{
  "@context": "https://w3id.org/identity/v1",
  "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
  "type": ["Entity", "Person"],
  "name": "Alice Bobman",
  "email": "alice@example.com",
  "birthDate": "1985-12-14",
  "telephone": "12345678910"
}
```

Claim

```
{
  "@context": "https://w3id.org/identity/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
  }
}
```

Verifiable Claim

```
{
  "@context": [
    "https://w3id.org/identity/v1",
    "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
  },
  "signature": {
    "type": "LinkedDataSignature2015",
    "created": "2016-06-18T21:10:38Z",
    "creator": "https://example.com/jdoe/keys/1",
    "domain": "json-ld.org",
    "nonce": "6165d7e8",
    "signatureValue": "g4j9UrpHM4/
uu32NlTw0HDaSaYF2sykskfuByD7UbuqEcJIKa+IoLJLrLjqDnMz0adwpBCHW
aqqpnd47r0NKZbnJarGYrBFcRTwPQSeqGwac8E2SqjylTBbSGwKZkprEXTywy
V7gILlC8a+naA7lBRi4y29FtcUJBTFQq4R5XzI="
  }
}
```

Zero Trust Computing

Security, Privacy, Agency

Diffuse trust perimeter-less security model

Diffuse trust perimeter-less security principles

The network is always hostile, internally & externally; Locality is not trustworthy.

Every network interaction or data flow must be authenticated and authorized using best practice cryptography.

Inter-host communication must be end-to-end signed/encrypted and data must be stored signed/encrypted; Data is signed/encrypted in motion and at rest.

Policies for authentication and authorization must be dynamically modified based on behavior.

Policies must be governed by distributed consensus.

Decentralized Identity Inverts Service Architectures

Conventional (centralized):

- Server creates identifiers (GUID, Database primary keys)
- Server timestamps
  - event ordering relative to server
- Server manages keys, AuthN, AuthZ
- Perimeter Security
  - Signed at rest problematic
  - Encrypted at rest problematic
- Server is source of truth
  - Server controls changes/updates to resources
- Server's role is 2nd party in two party transactions between client and server.

Unconventional (decentralized):

- Client creates identifiers (DIDs)
- Client timestamps
  - event ordering relative to client
- Client manages keys
  - AuthN/AuthZ is Client-to-client
- Perimeter-less Security
  - Client signs at rest
  - Client encrypts at rest
- Client is source of truth
  - Client controls changes/updates to resources
  - Server cannot make changes
- Server's role is either:
  - Trusted 3rd party in 3 (multi) party transactions between 2 (or more) clients and server
  - Agent or proxy for a client in two party transaction with another client.

Heavyweight — Decentralized Identity — Lightweight

Heavyweight	Lightweight
Decentralized Identifiers	Decentralized Identifiers

- Decentralized Identifiers
- Portability
- Verifiable Claims
    - Certification
    - Consent Decrees
- Ledger
    - Revocation
    - Arbitration
    - Provenance
- Key Management
    - Hierarchical Keys
    - Key Recovery
    - Key Hiding
    - Multisignature
    - Group Keys
- Identity Graph
    - Attribute Based Identity
    - Contextual Identity
    - Transitivity
    - Group Identity
    - Least Disclosure

Heavyweight — Reputation — Lightweight

Heavyweight	Lightweight
Portable	Eventually Portable
Transitive	Domain Specific
Cross-Domain	

Graph Based Self Identity/Reputation

Identity = *Identifiers* + *Attributes*

Identifiers = *globally unique decentralized cryptonyms* + *aliases*

Attributes = *user data, proofs*

Facilitate attribute exchange between entities sufficient to enable transaction to proceed

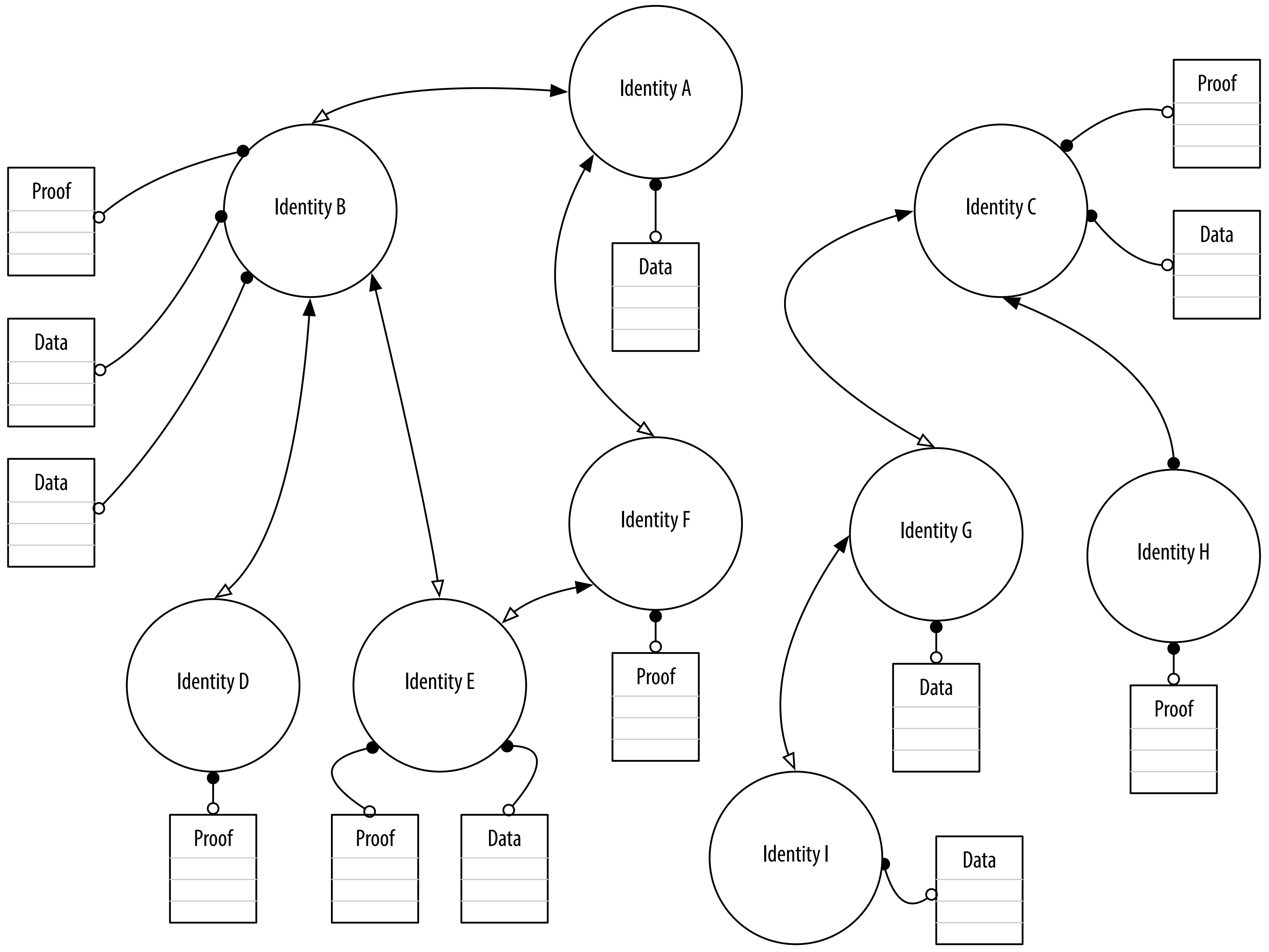
Identity System Features:

- Agency (own your own identity)
- Security (impervious to fraud)
- Privacy (least disclosure)

Agency = portable identifiers + user controlled

Security = distributed consensus + modern crypto

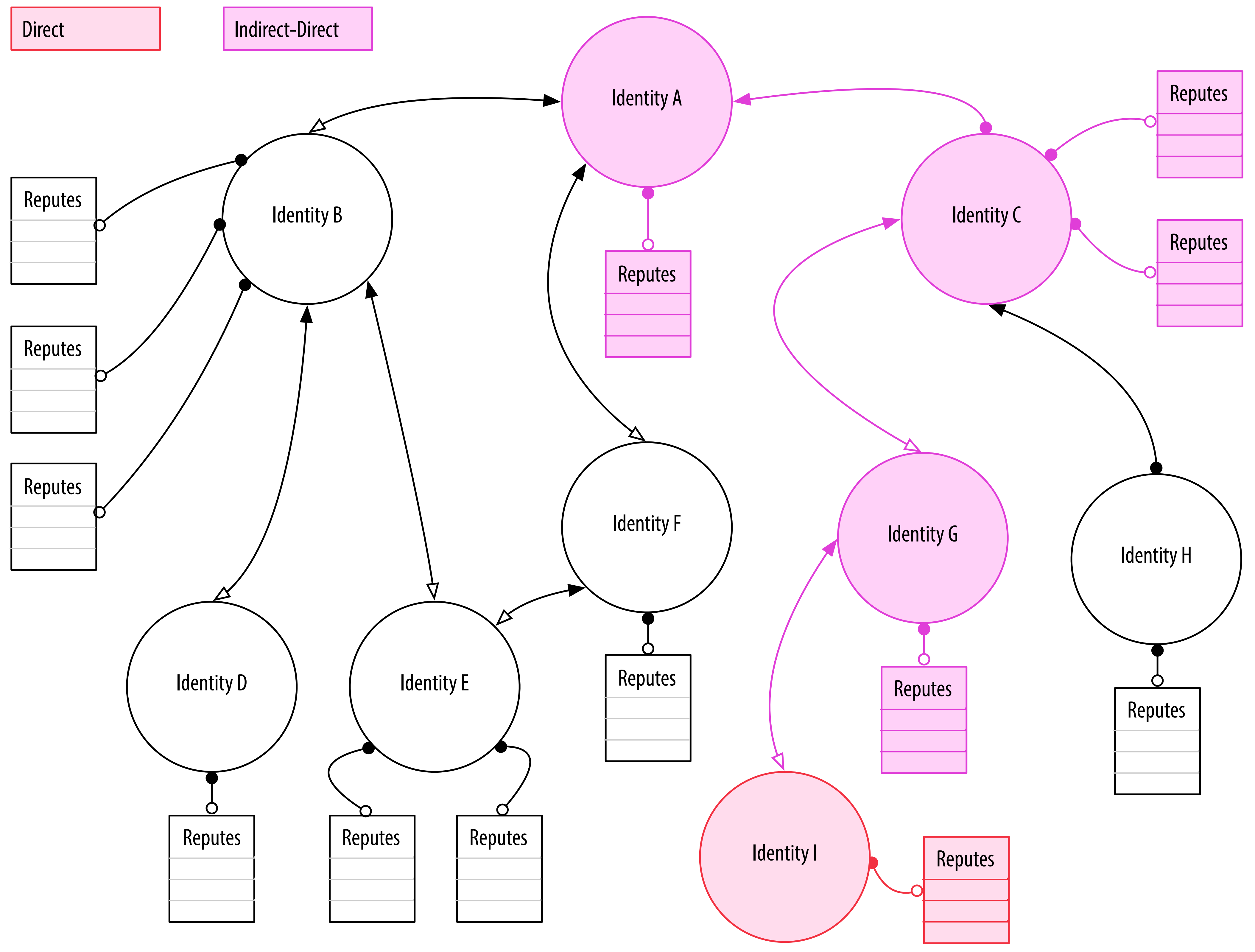
Privacy = granular graph based identities + layered disclosures + zero knowledge disclosures + group identities



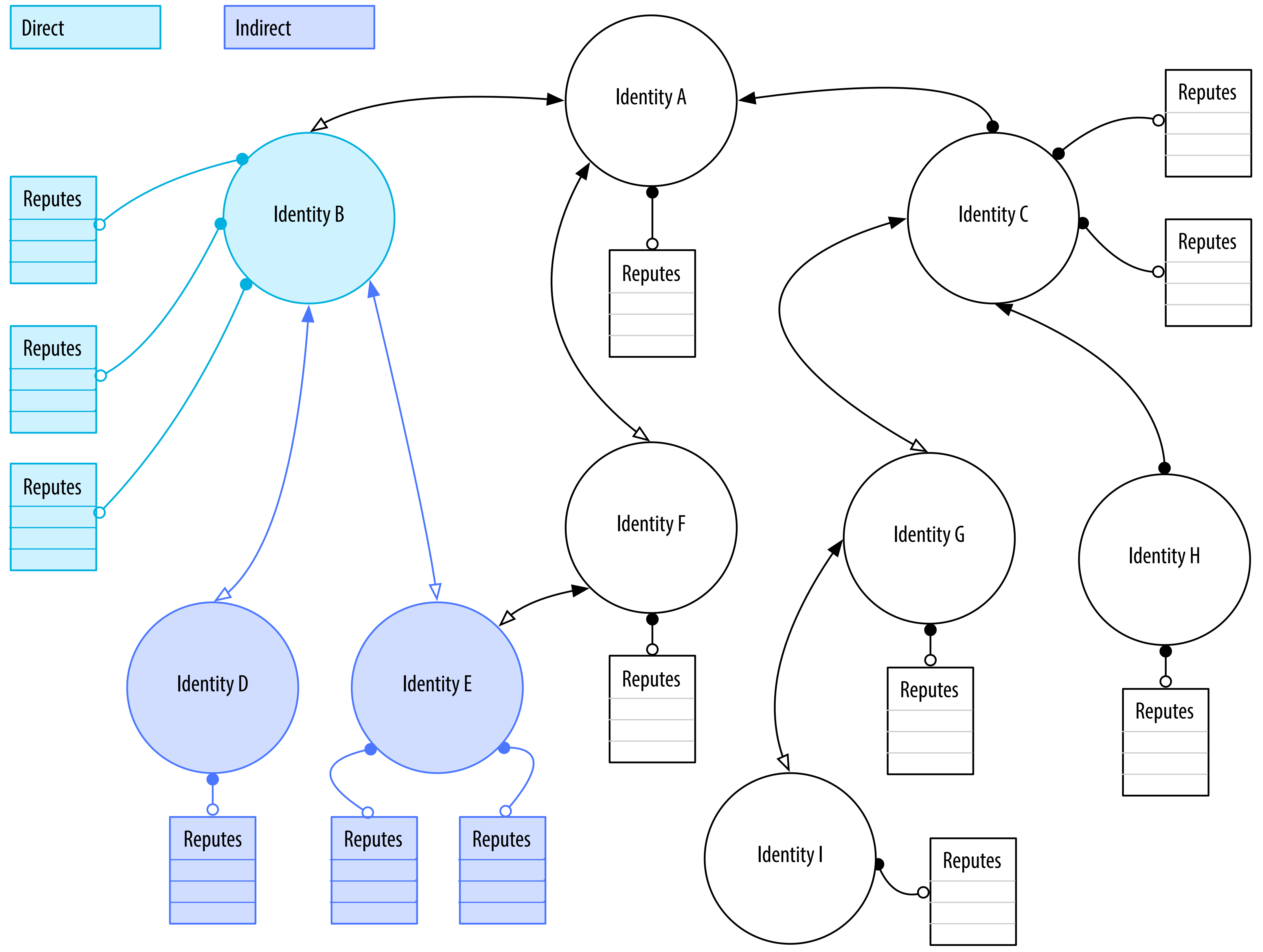
https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/Identity-System-Essentials.pdf

Graph Based Other Identity/Reputation

Repute Down



Repute Up



https://github.com/SmithSamuelM/Papers/blob/master/whitepapers/Identity-System-Essentials.pdf

ISSUES

Data Intensive:

- *High speed concurrent reads*
- *Multi-version Concurrency Control (MVCC) Data Stores*
- *Concurrent Log Structured Data Store (Immutable)*
- *Data Flow Routing for Distributed Stream Processing*

Stored in sorted order for fast block queries

- *did:rep:ABCD123/ruid/did:rep:XYZ987/what*

Repeated Lookups are expensive

Need lightweight cached DDO subset

- *Signed At Rest verification without external lookups*
- *signer: did:rep:ABCD123#key/01*

ISSUES

Associated Repute vs Associated Reputee

- *Associated Repute = DID/path*
  - *Associated Reputee = Hierarchical DID*
  - *Unknown Reputer or Reputee to be claimed later*
  - *Blinded Reputer or Reputee*
- *Meta-Repute = Repute with a Reputation*

Reputation is not a set of verifiable claims but a set of verifiable claims can contribute to a reputation

Representation of Graph Based Identity

Portable Reputation Algorithms

ISSUES

Survivability Paradigm vs IT Security Paradigm

IT Security: Confidentiality, Integrity, Availability

Survivability:

- *Susceptibility: likelihood that the system will be the target of an attack*
- *Vulnerabililty: degree of damage resulting from an attack*
- *Recoverability: time to repair the damage of an attack*

Failure management:

- *Failure avoidance, Failure Resilience, Failure Repair*