

CRYPTOCURRENCY FOR THE GIFT ECONOMY

DOCUMENT COIN

Chris Anderson

@jchris

jchris@gmail.com

QCon London 2016

TAKEHOME LESSONS

- Blockchain without global consensus is interesting
- WebCrypto APIs will bring innovation to blockchain applications
- Application developers not cryptographers?!?
- IETF JOSE means standardized implementations



DOYLE OWL TRADING CARD

Number 4 of 7



The Owlphant says,
"I shall return."

This card is worth 1 time point.

Brought to you by Tom Petersen and Gloria, too!



BACKGROUND ASSUMPTIONS

IDEOLOGY

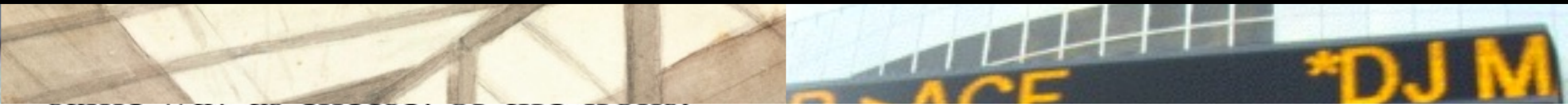
WHERE WOULD YOU RATHER LIVE?

LEGIBILITY



HOW WE KNOW SOMETHING IS IMPORTANT

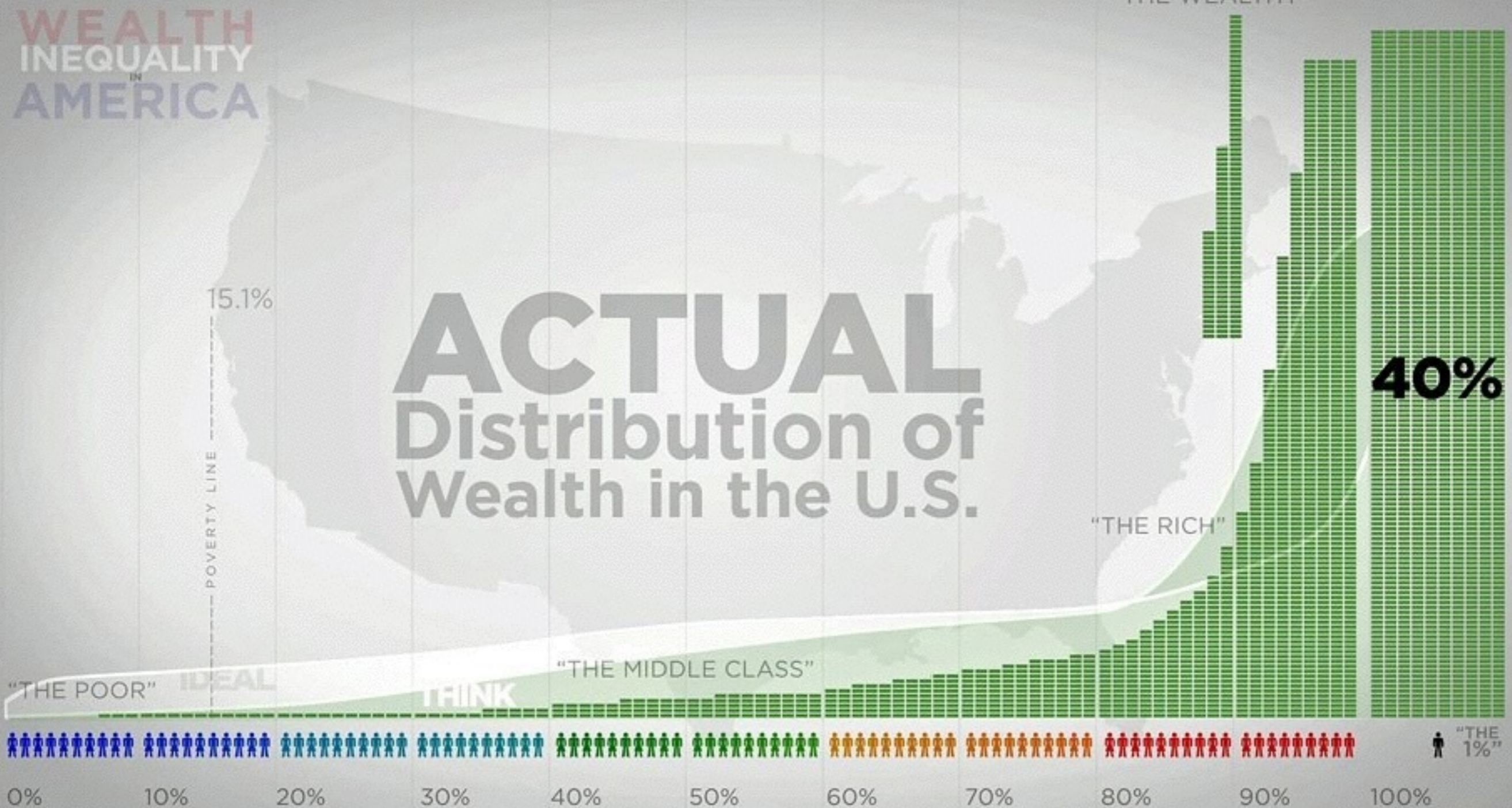
VALUE



Gregory lays out a tidy set of oppositions. Gifts are transactions that are meant to create or effect “qualitative” relations between persons; they take place within a preexisting web of personal relations; therefore, even the objects involved have a tendency to take on the qualities of people. Commodity exchange, on the other hand, is meant to establish a “quantitative” equivalence of value between objects; it should ideally be done quite impersonally; therefore, there is a tendency to treat even the human beings involved like things. Giving someone a gift usually puts that person in your debt; hence, success in gift exchange becomes a matter of giving away as much wealth as possible, so as to gain a social advantage. In a commodity system, it’s the things that are important; therefore, people try to accumulate as much wealth as they can.

99% OF PEOPLE HAVE 60% OF THE WEALTH

INEQUALITY

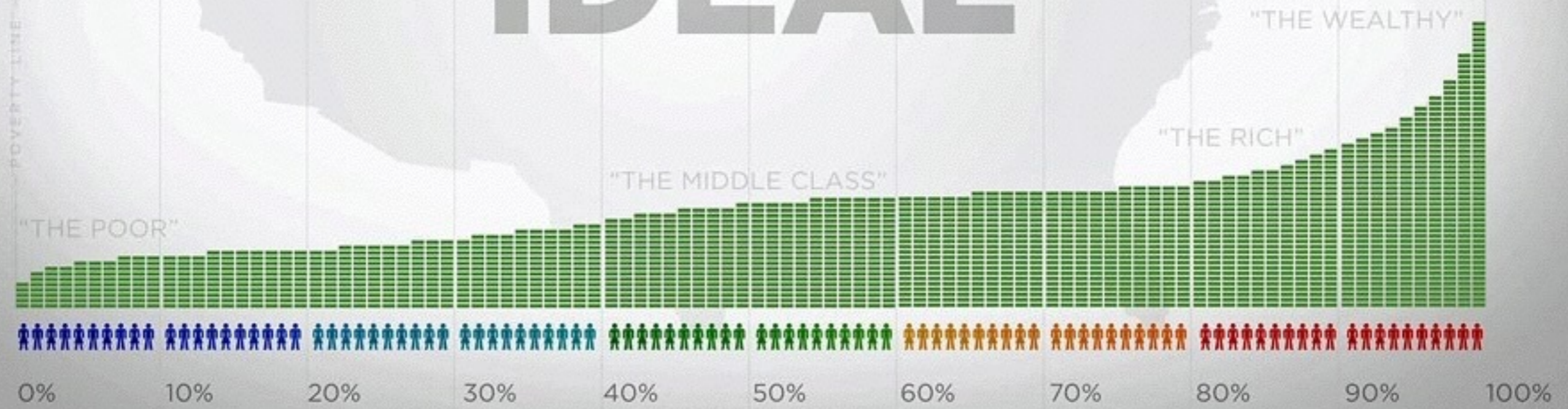


ADJUSTING THE CURVE

PROGRESSIVISM

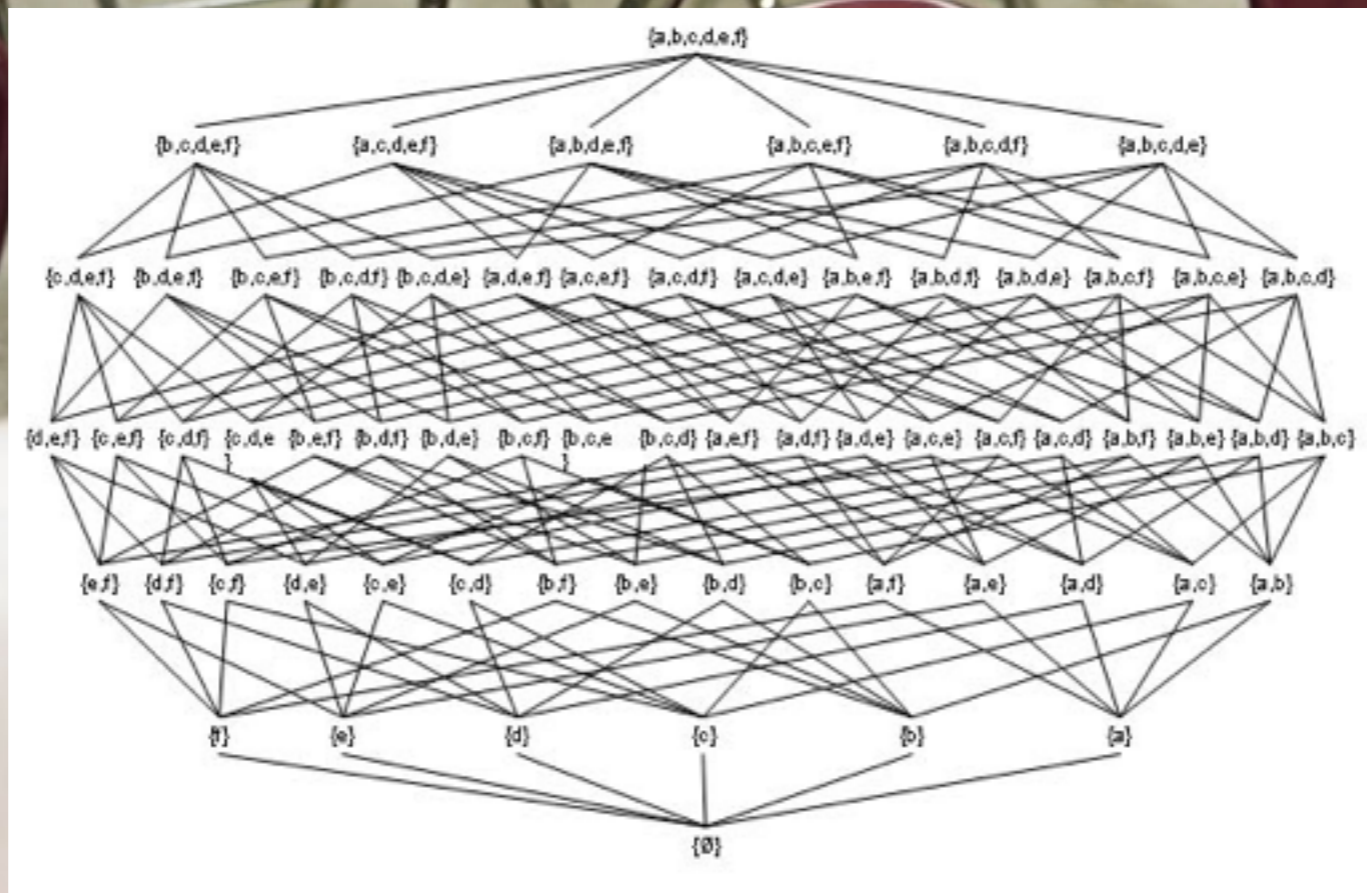
WEALTH
INEQUALITY
IN
AMERICA

Distribution that **92%**
of Americans Choose as
IDEAL



DO WE NEED IT?

GLOBAL PARTIAL ORDERING



WE CAN LIVE WITHOUT NUMBERS

PIRAHÃ



CAN WE SCALE IT?

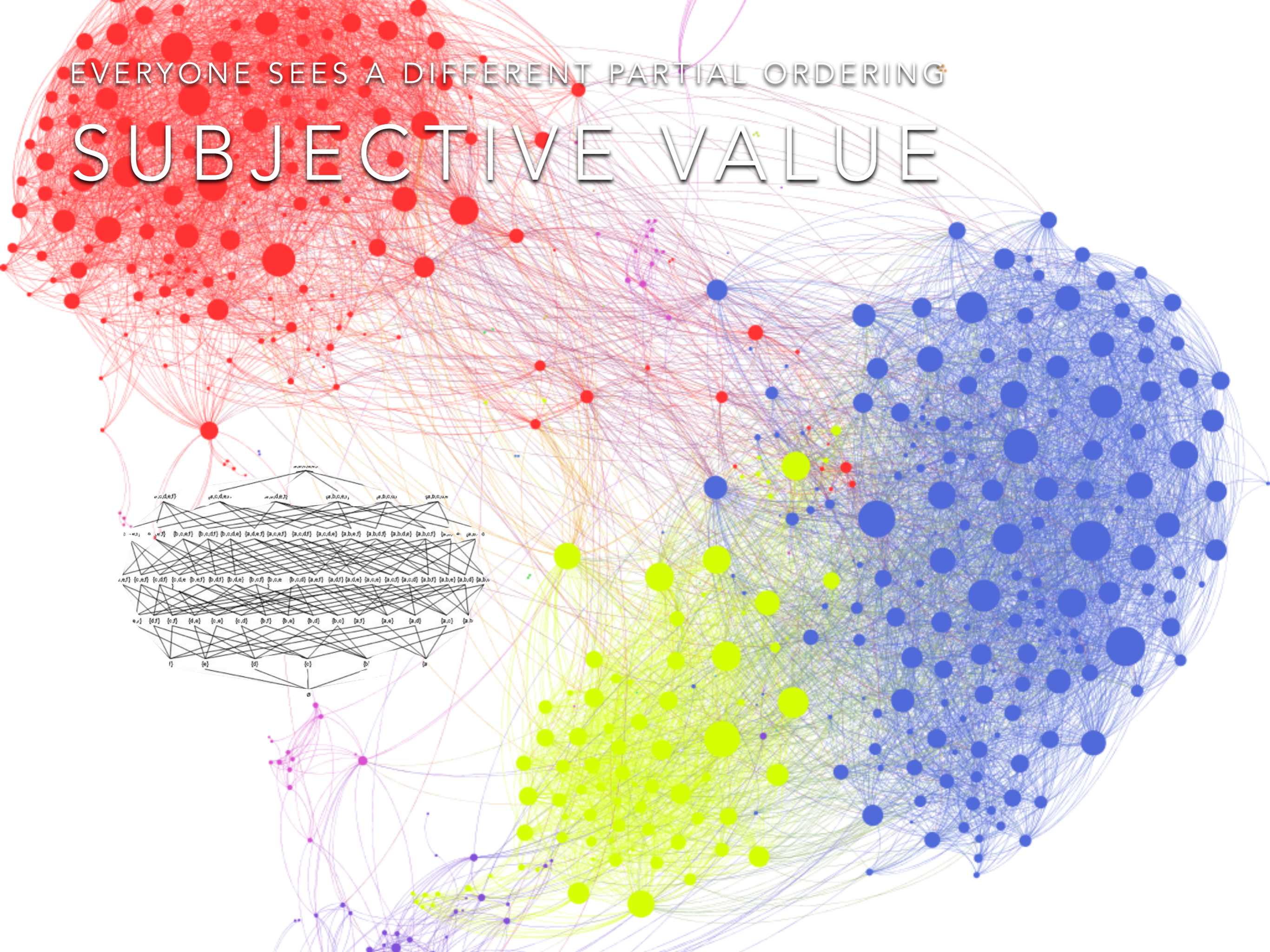
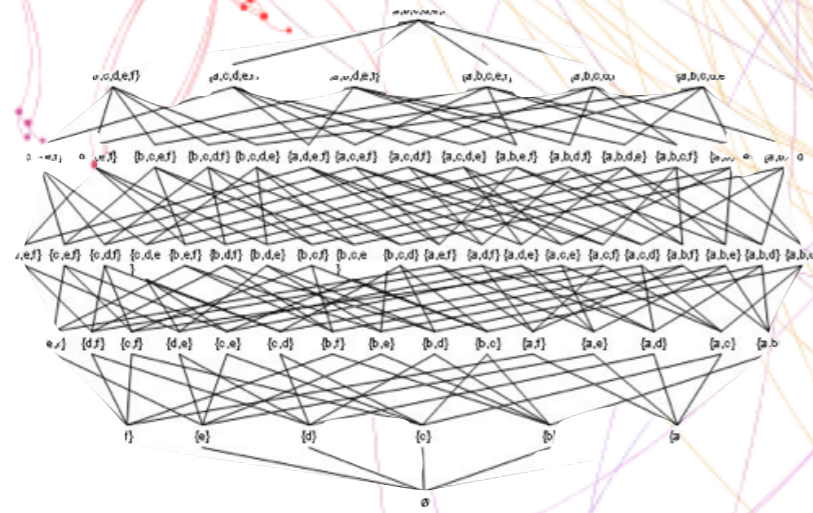
GIFT ECONOMY



The Gift Economy

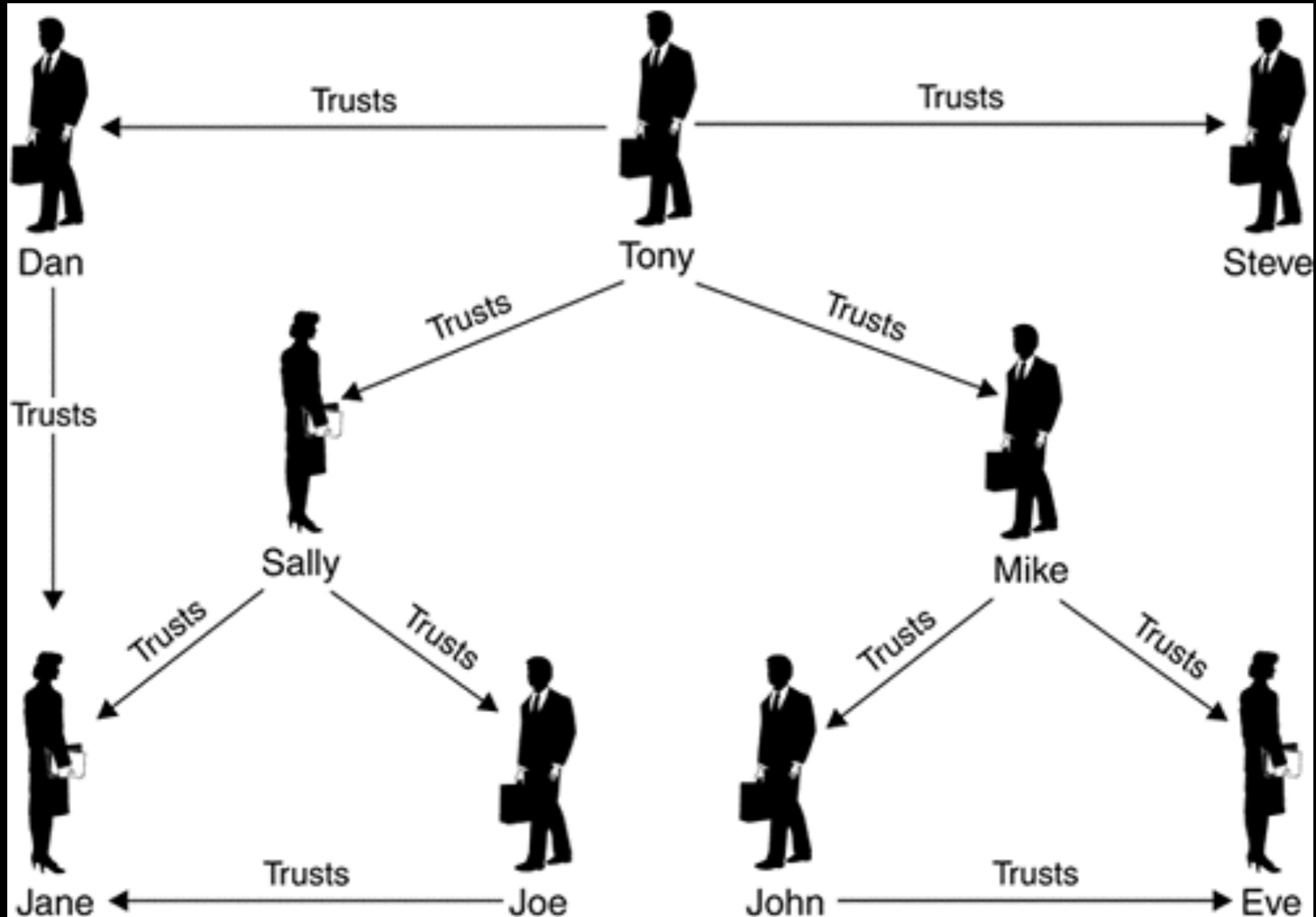
EVERYONE SEES A DIFFERENT PARTIAL ORDERING

SUBJECTIVE VALUE



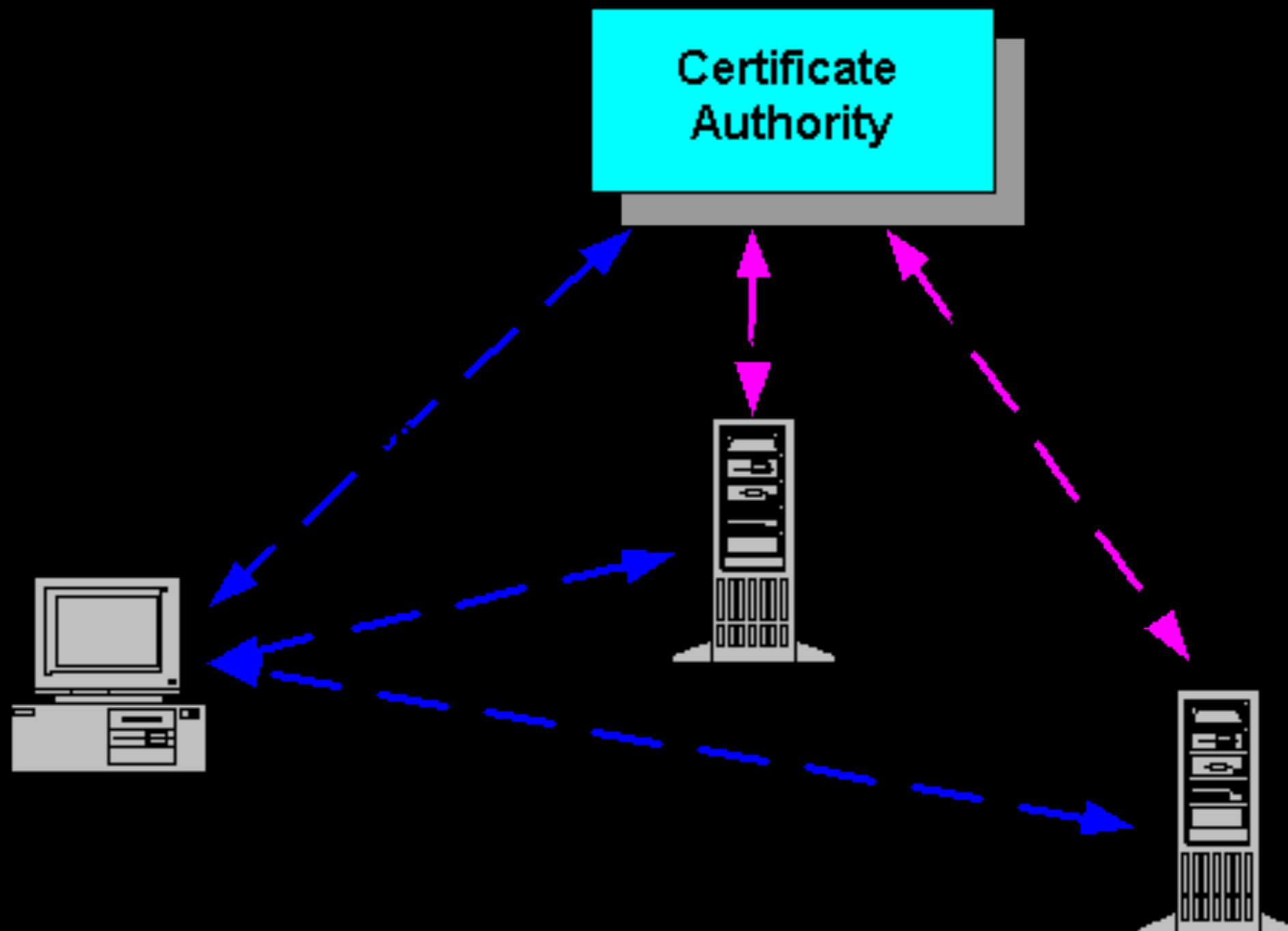
PRIOR ART

WEB OF TRUST



WHO WATCHES THE WATCHERS?

CERTIFICATE AUTHORITY



melty.

KEY SIGNING PARTY

NOW WE HAVEN PROBLEMS



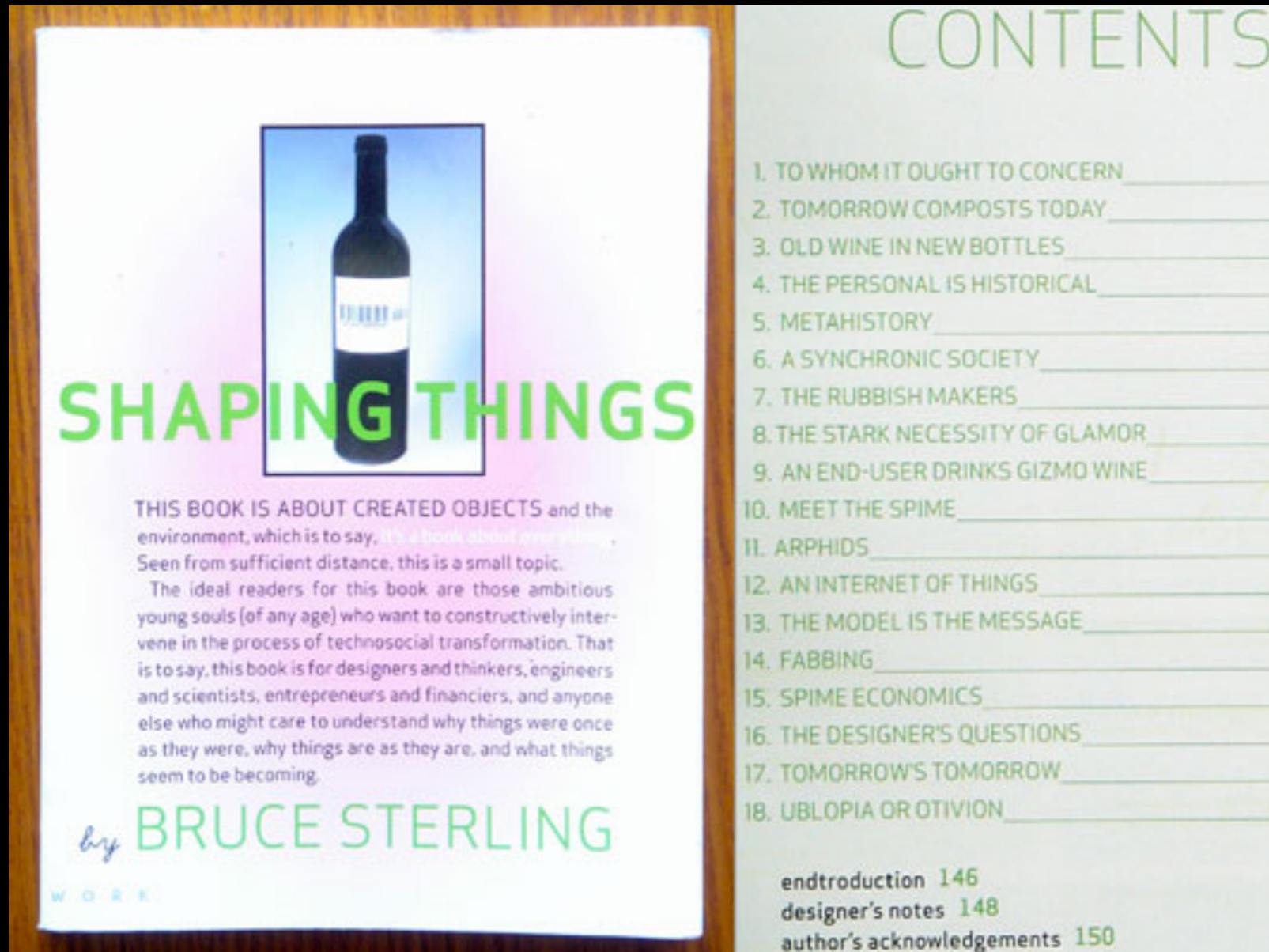
OBJECT BLOCKCHAIN

- Blockchain allows us to create digital objects
- Heirlooms
- Gamify the web of trust



FROM CLASSES TO INSTANCES

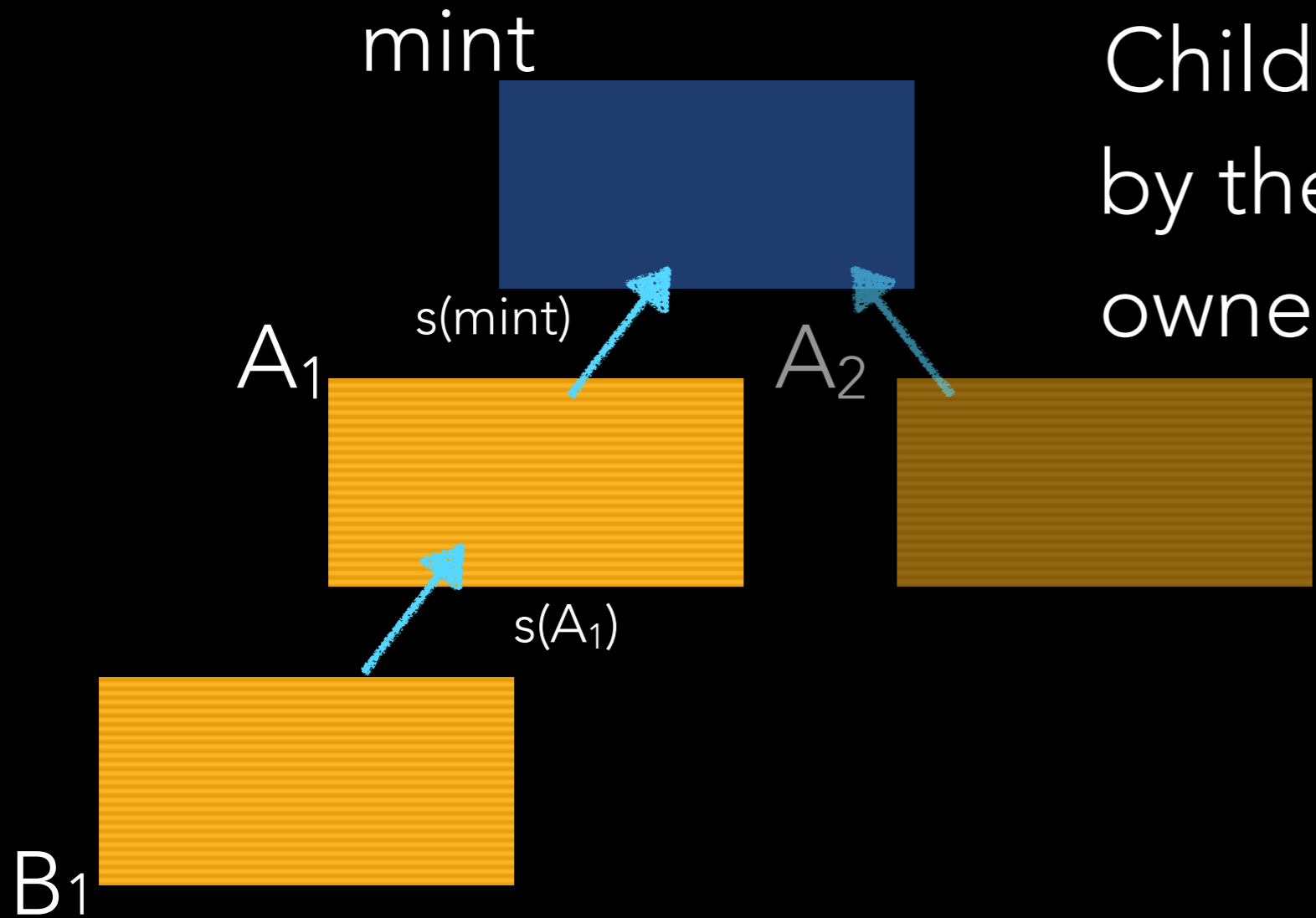
INTERNET THINGS



GIVE TREE

- Per document blockchain
- Don't wait for global consensus
- Data provenance structures allows the tracking of how ownership of a piece of data got to you
- TODO: Encrypt the content of the coins.

GIVE TREE



Child nodes are signed by the keys encoded as owner of parent blocks.

VALIDATE THE GIVE TREE

- Verify the root content hash and mint signature match the unique coin id.
- For each child in the tree, verify that the signature is correct and comes from the public key listed in the parent block.
- Application level policy can flag branches of the tree that exceed a **give limit**. Physical objects are simulated with a give limit of one.

GIVE TREE

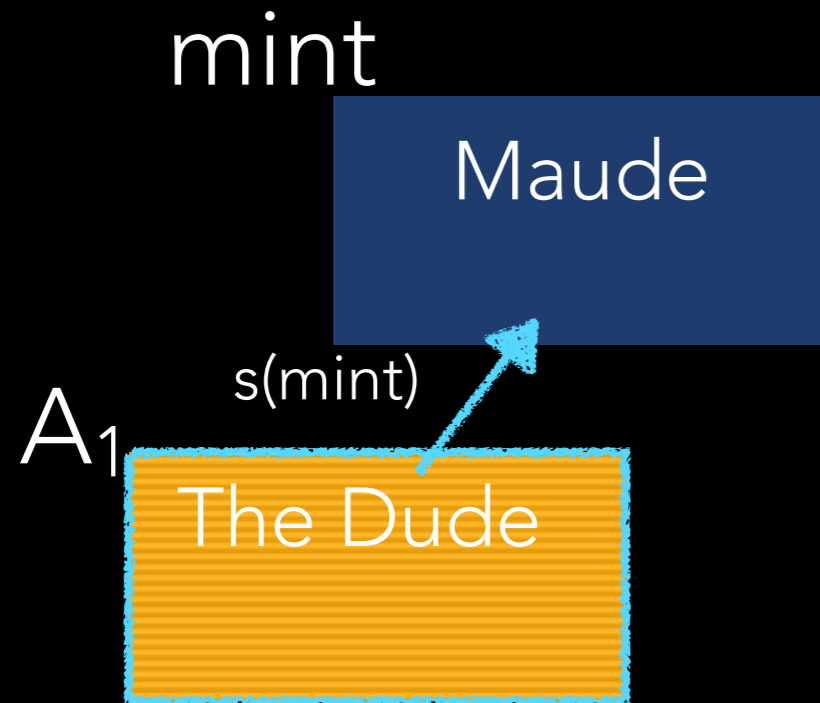
mint

Maude

mint

to (Maude keys)
on (content hash)
signed (Maude)

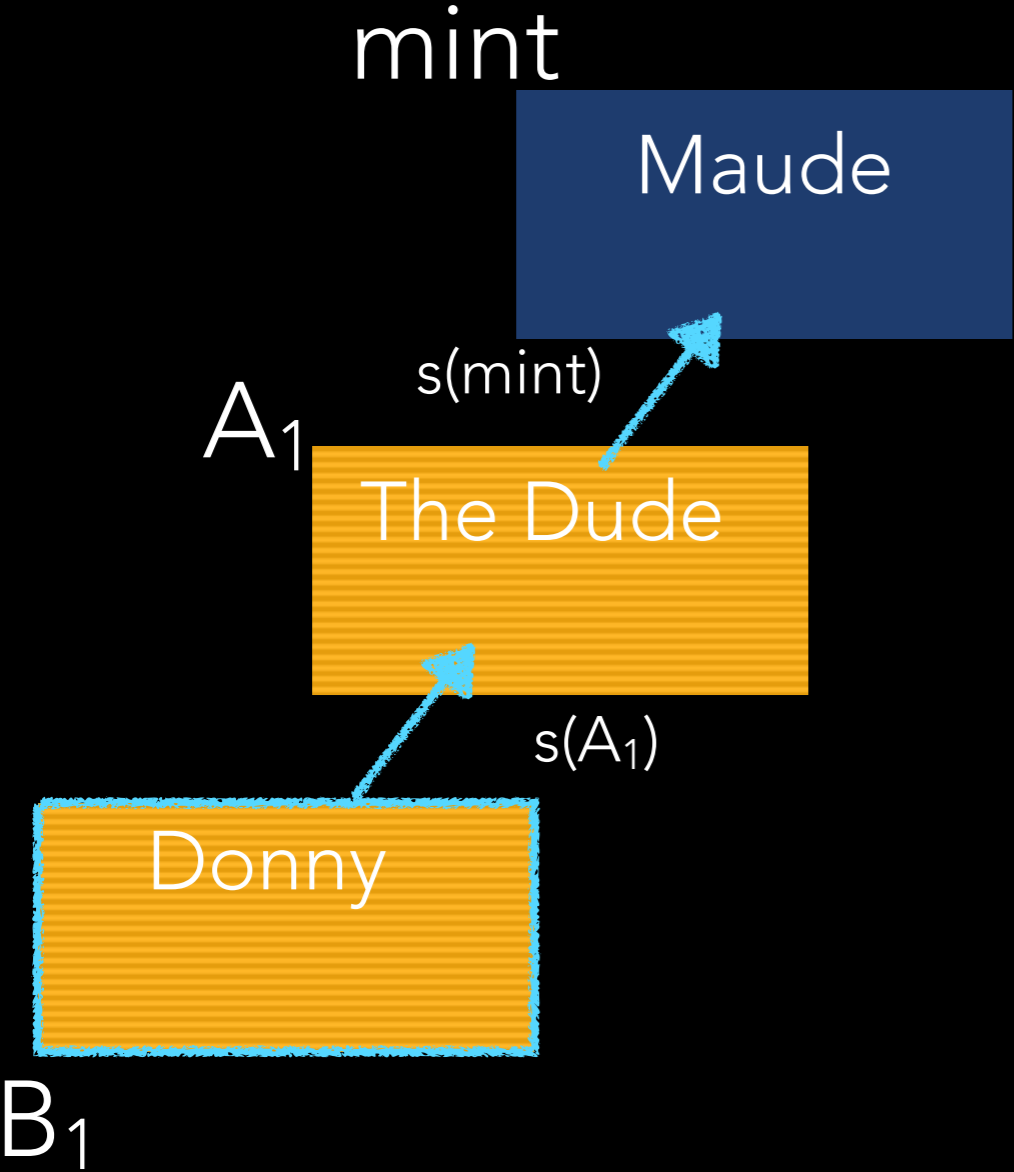
GIVE TREE



A₁

to (The Dude)
on (parent hash)
signed (Maude)

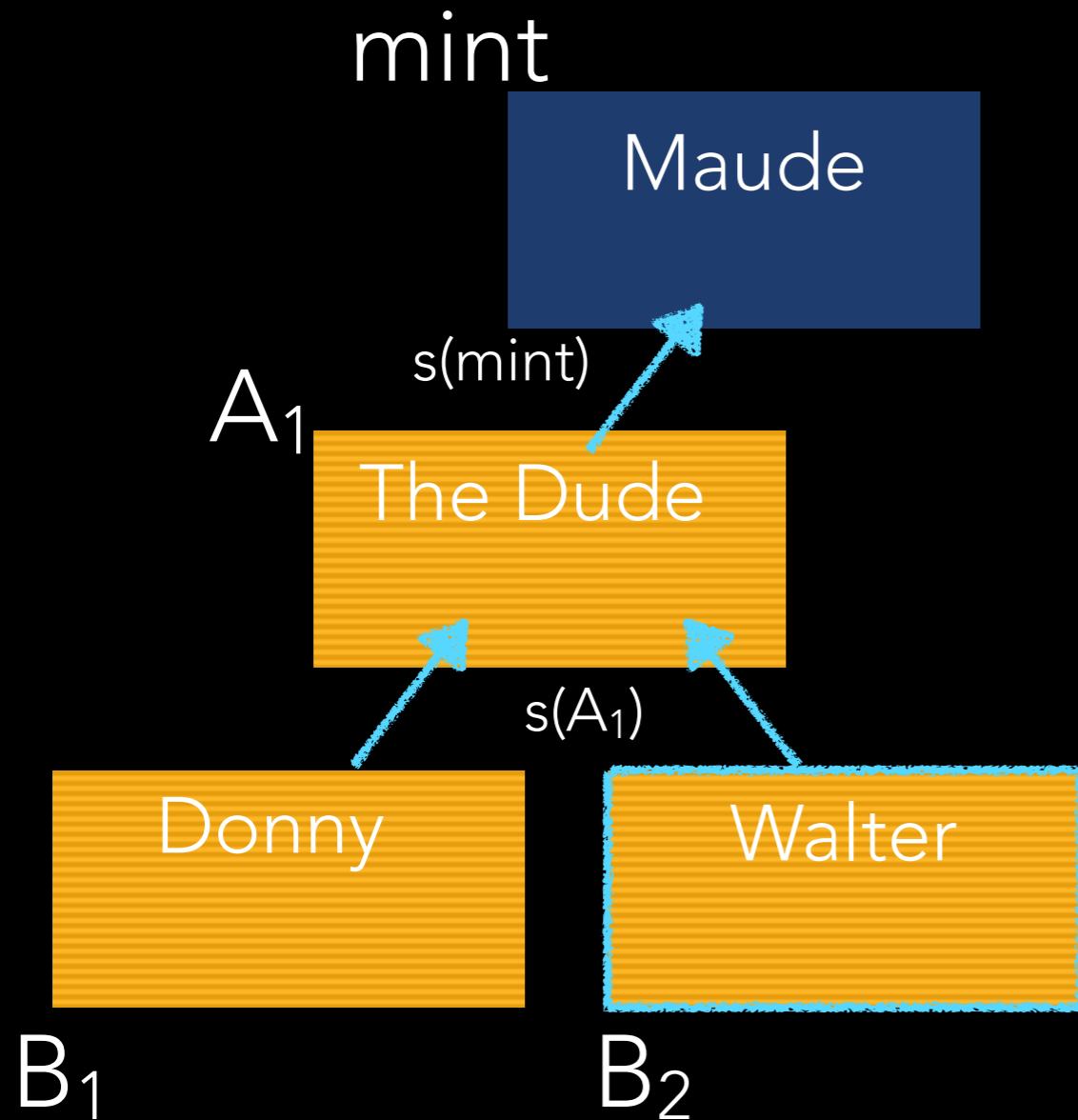
GIVE TREE



B_1

to (Donny keys)
on (parent hash)
signed (The Dude)

GIVE TREE

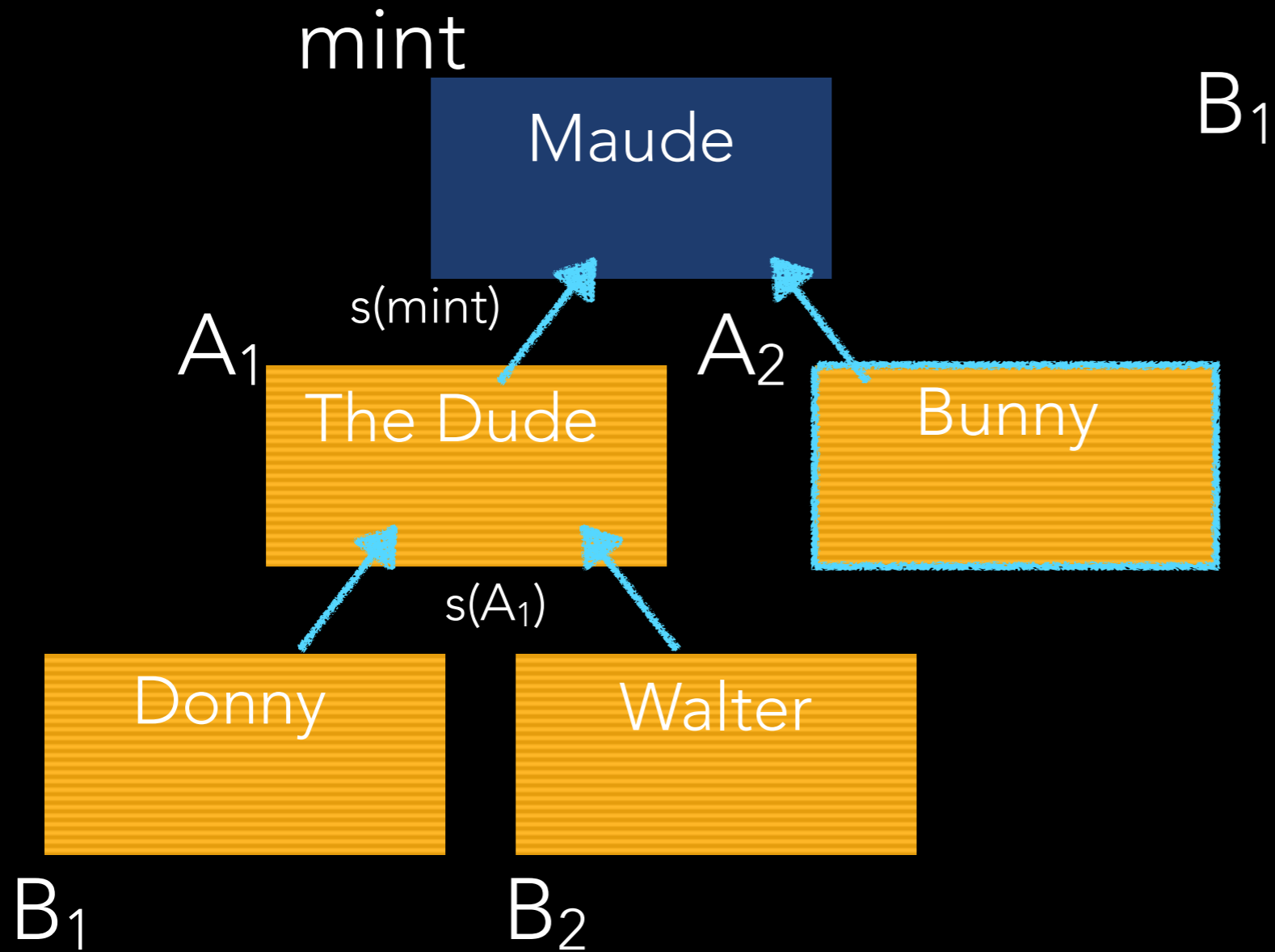


B_2

to (Walter keys)
on (parent hash)
signed (The Dude)

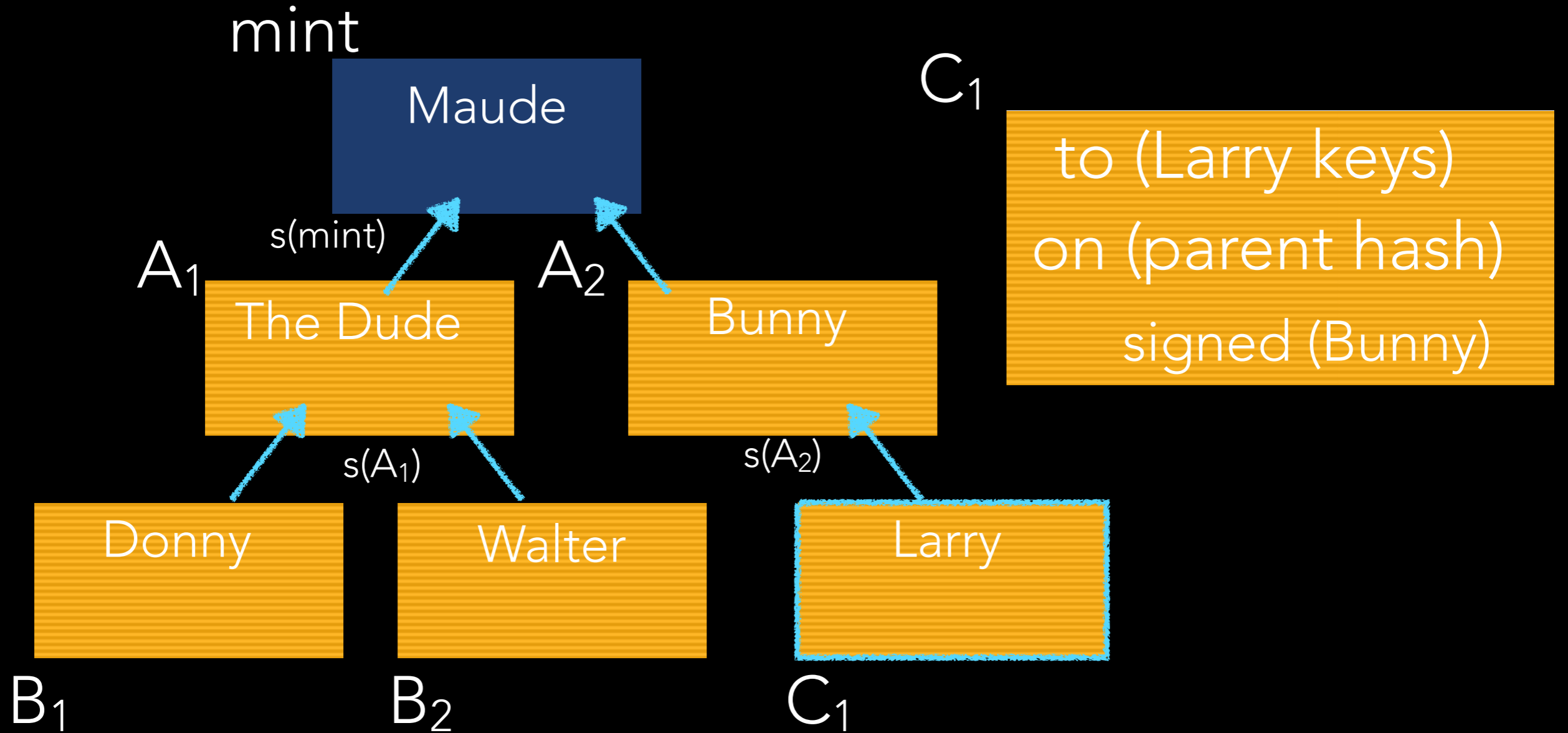
"It went OK, the old man told me to take any rug in the house."

GIVE TREE



to (Walter keys)
on (parent hash)
signed (The Dude)

GIVE TREE



LIMITATIONS

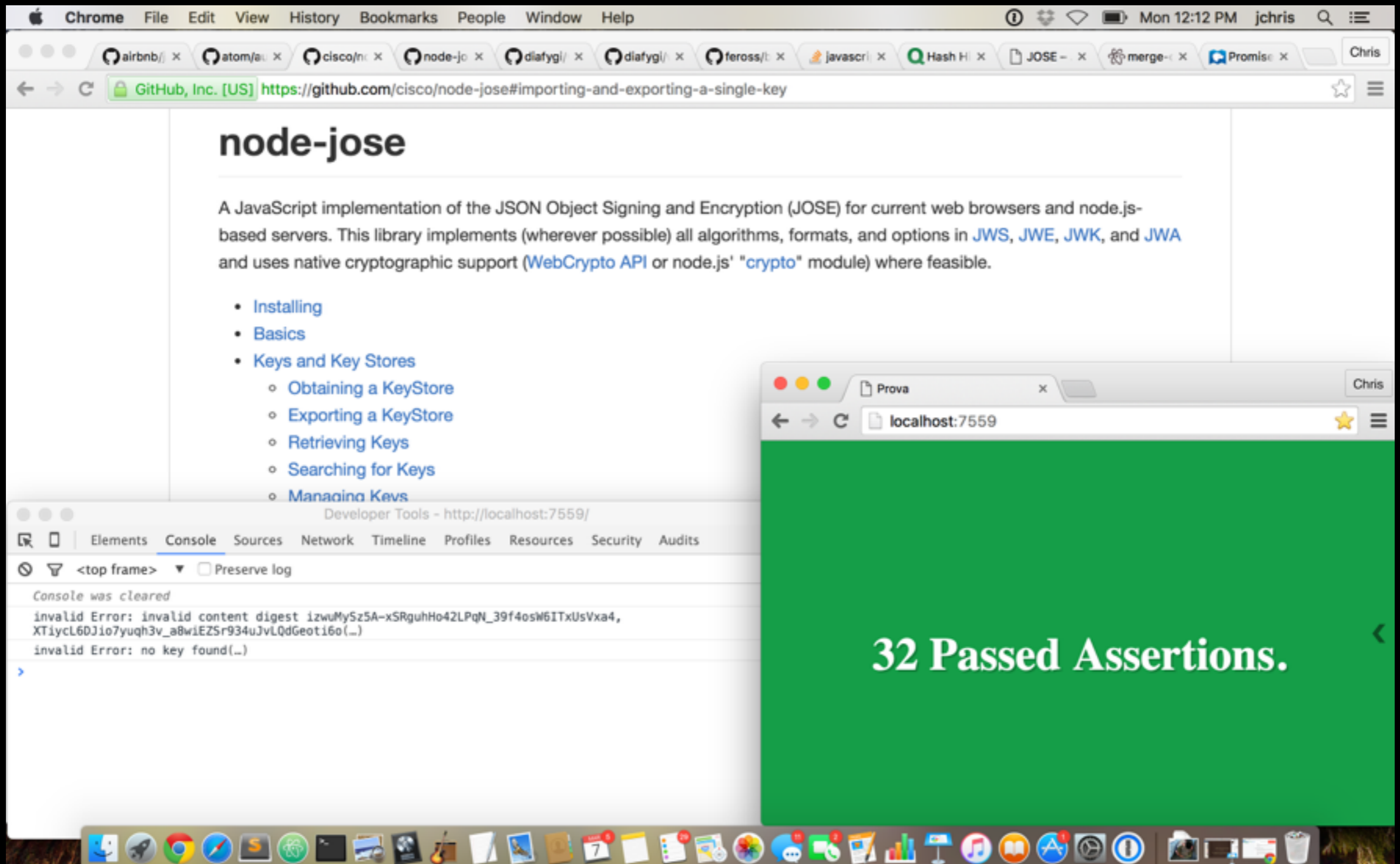
- The model, even if perfectly implemented, isn't designed to prevent double spending, just to detect it.
- Research project, currently only dealing with public content and signatures.
- Haven't implemented the Bill Murray rule yet.
- Want multi-sig mint blocks

IMPLEMENTATION

- Cisco's JOSE <https://github.com/cisco/node-jose>
- Prova test runner <http://github.com/azer/prova>
- Recursion!

TEST IN THE BROWSER WITHOUT ANY BUILD SCRIPTS

PROVA



The image shows a Chrome browser window displaying the GitHub page for the `node-jose` library. The page title is `node-jose` and the description states: "A JavaScript implementation of the JSON Object Signing and Encryption (JOSE) for current web browsers and node.js-based servers. This library implements (wherever possible) all algorithms, formats, and options in [JWS](#), [JWE](#), [JWK](#), and [JWA](#) and uses native cryptographic support ([WebCrypto API](#) or node.js' `"crypto"` module) where feasible."

- [Installing](#)
- [Basics](#)
- [Keys and Key Stores](#)
 - [Obtaining a KeyStore](#)
 - [Exporting a KeyStore](#)
 - [Retrieving Keys](#)
 - [Searching for Keys](#)
 - [Managing Keys](#)

The browser's developer tools are open, showing the console with the following messages:

```
Console was cleared
invalid Error: invalid content digest izwuMySz5A-xSRguhHo42LPqN_39f4osW6ITxUsVxa4,
XTiycL6DJio7yuqh3v_a8wiEZSr934uJvLQdGeoti6o(...)
invalid Error: no key found(...)
```

Overlaid on the bottom right of the browser window is a green box with the text "32 Passed Assertions." in white, indicating a successful test run.

AUTOMAGICAL RELOAD AND RUN PROVA

The image shows a Chrome browser window displaying the GitHub page for `node-jose`. The page title is `node-jose` and the description reads: "A JavaScript implementation of the JSON Object Signing and Encryption (JOSE) for current web browsers and node.js-based servers. This library implements (wherever possible) all algorithms, formats, and options in [JWS](#), [JWE](#), [JWK](#), and [JWA](#) and uses native cryptographic support ([WebCrypto API](#) or node.js' `"crypto"` module) where feasible." Below the description is a list of links: `Installing`, `Basics`, `Keys and Key Stores` (with sub-links: `Obtaining a KeyStore`, `Exporting a KeyStore`, `Retrieving Keys`, `Searching for Keys`, `Managing Keys`).

Overlaid on the bottom right is a Prova application window. The address bar shows `localhost:7559`. The application has a green header with buttons: `REPEAT`, `WATCHING FILE CHANGES`, and `MAXIMIZE FRAME`. The main content area displays a list of tasks, each with a checkmark:

- ✓ create a wallet
- ✓ mint a coin
- ✓ give a coin
- ✓ give a coin multiple times
- ✓ invalid coin content
- ✓ invalid coin

The Chrome Developer Tools console is open at the bottom, showing the following messages:

```
Console was cleared
invalid Error: invalid content digest izwuMySz5A-xSRguhHo42LPqN_39f4osW6ITxUsVxa4,
XTiycL6DJio7yuqh3v_a8wiEZSr934uJvLQdGeoti6o(...)
invalid Error: no key found(...)
```

EASY AND CONSISTENT WRAPPER AROUND WEB CRYPTO

CISCO NODE-JOSE

Chrome File Edit View History Bookmarks People Window Help Mon 12:31 PM jchris

airbnb/ x atom/a/ x cisco/n/ x node-jc x diafyg/ x diafyg/ x feross/t/ x javascr/ x Hash H/ x JOSE - x merge- x Promise x Chris

GitHub, Inc. [US] <https://github.com/cisco/node-jose#importing-and-exporting-a-single-key>

Encrypting Content

At its simplest, to create a JWE:

```
// {input} is a Buffer
jose.JWE.createEncrypt(key).
  update(input).
  final().
  then(function(result) {
    // {result} is a JSON Object -- JWE using the JSON General Serialization
  });
```

How the JWE content is encrypted depends on the provided Key.

- If the Key only supports content encryption algorithms, then the preferred algorithm is used to encrypt the content and the key encryption algorithm (i.e., the "alg" member) is set to "dir". The preferred algorithm is the first item returned by `key.algorithms("encrypt")`.
- If the Key supports key management algorithms, then the JWE content is encrypted using "A128CBC-HS256" by default, and the Content Encryption Key is encrypted using the preferred algorithms for the given Key. The preferred algorithm is the first item returned by `key.algorithms("wrap")`.

To create a JWE using a different serialization format:

```
jose.JWE.createEncrypt({ format: 'compact' }, key).
  update(input).
  final().
  then(function(result) {
    // {result} is a String -- JWE using the Compact Serialization
  });
```

Taskbar icons: Mail, Safari, Chrome, Firefox, VS Code, Docker, Slack, Calendar, Photos, Messages, App Store, System Preferences, Spotlight, Trash.

VALIDATE A COIN

- Validate the root of the Give Tree
 - Coin ID is correct
 - Content signature matches
- Recursively validate all child nodes
 - Give limit policy violations are business errors, not validation errors

VALIDATE THE ROOT

```
validateRoot() {
  return this.rootDigest().then((signatureDigest) => {
    const basedDigest = URLSafeBase64.encode(signatureDigest);
    if (this.coinID !== basedDigest) {
      throw new Error("invalid coinID " + this.coinID + ", " + basedDigest)
    }
    // validate contentDigest
    return jose.JWA.digest("SHA-256", this.content).then((contentDigest)=>{
      const basedContentDigest = URLSafeBase64.encode(contentDigest);
      return this.decodeRootNode().then((results) => {
        if (results.rootPayload[0] !== basedContentDigest) {
          throw new Error("invalid content digest " +
            results.rootPayload[0] + ", " + basedContentDigest)
        }
      })
    })
  })
}

rootDigest() {
  const rootBlock = this.givetree[0];
  const blockBuffer = new Buffer(rootBlock, "utf8");
  return jose.JWA.digest("SHA-256", blockBuffer)
}
```


VALIDATE THE ROOT

```
rootDigest() {  
  const rootBlock = this.givetree[0];  
  const blockBuffer = new Buffer(rootBlock, "utf8");  
  return jose.JWA.digest("SHA-256", blockBuffer)  
}
```

```
decodeRootNode() {  
  var holderKey = this.mintKey;  
  var node = this.givetree;  
  var block = node[0];  
  var children = node[1];  
  return this.decodeBlock(block, holderKey).then((decoded) => {  
    var rootPayload = JSON.parse(decoded.payload.toString())  
    return {rootPayload:rootPayload, block:block, children:children};  
  })  
}
```

VALIDATE CHILDREN

```
validate() {  
  return this.validateRoot().then(()=>{  
    return this.foldBlockNodes((decodedBlock, rawBlock, children) => {  
    })  
  }).then(()=>{  
    return true;  
  }).catch((e)=>{  
    console.log("invalid", e);  
    return false;  
  })  
}
```



Empty Callback!

VALIDATE CHILDREN

```
foldBlockNode(callback, parentPayload, rawParentBlock, children) {
  callback(parentPayload, rawParentBlock, children)
  // for each child, decode the child and call foldBlockNode on it
  if (children.length > 0) {
    return jose.JWK.asKey(parentPayload[1]).then((holderKey) => {
      var decoders = children.map((childNode)=>{
        const rawChildBlock = childNode[0];
        const childChildren = childNode[1];
        return this.decodeBlock(rawChildBlock, holderKey).then((decoded) => {
          var childPayload = JSON.parse(decoded.payload.toString())
          return this.foldBlockNode(callback, childPayload, rawChildBlock, childChildren)
        })
      })
      return Promise.all(decoders);
    });
  } else {
    return true;
  }
}
```



Recursion!

NEXT STEPS

- Encrypted content
- PouchDB storage & sync
- Photo sharing application
- Public key graph browser



... PROFIT

- Distributed PKI
- Illustrate to users the value of partial disclosure
- Ecosystem interop
- Native HTML5 blockchain
- Platform for p2p social apps



CONCLUSION

- We are the world computer, build expressive software
- Blockchain assumptions can be inverted
- There is more room for innovation than we think



CRYPTOCURRENCY FOR THE GIFT ECONOMY

DOCUMENT COIN

Chris Anderson

@jchris

jchris@gmail.com

QCon London 2016