Lecture 13

Applications of Blockchains - I

What can we build on top of Bitcoin/Blockchains?

Why Applications from Bitcoin/Blockchains?

- Decentralization: Many applications easy to realize with a central trusted authority. Bitcoin/Blockchains can often help in removing central trust

Applications

- Timestamping
- Token tracking
- Public randomness
- Prediction markets
- Fair protocols: Multiparty lotteries, MPC
- One-time Programs
- Non-Interactive Zero Knowledge
- …

Bitcoin as an append-only ledger

Secure timestamping

Goal: Prove knowledge of x at time t

If desired, <u>without</u> revealing x at time t

Evidence should be permanent

Hash commitments

Recall: Publishing H(key,*x*) is a *commitment* to *x*

- Can't find an $x' \neq x$ later s.t. $H(key,x') = H(key,x)$
- H(key,*x*) does not reveal *x* in RO model

Can publish a commitment to x, reveal later

Secure timestamping applications

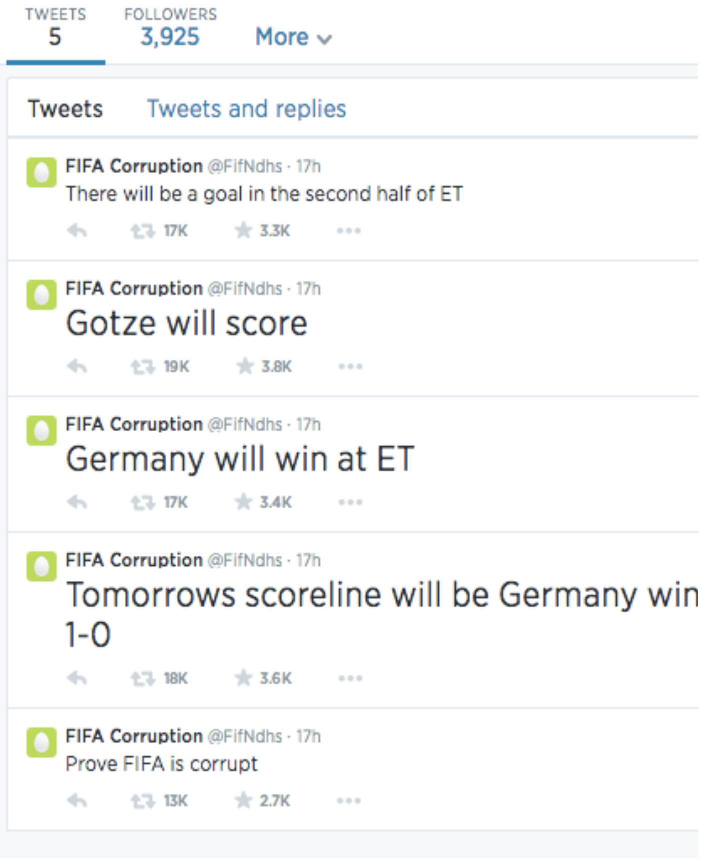
- Proof of knowledge
- Proof of receipt
- Hash-based signature schemes
- many, many more…

Non-application: proof of clairvoyance

Proof that FIFA is corrupt??

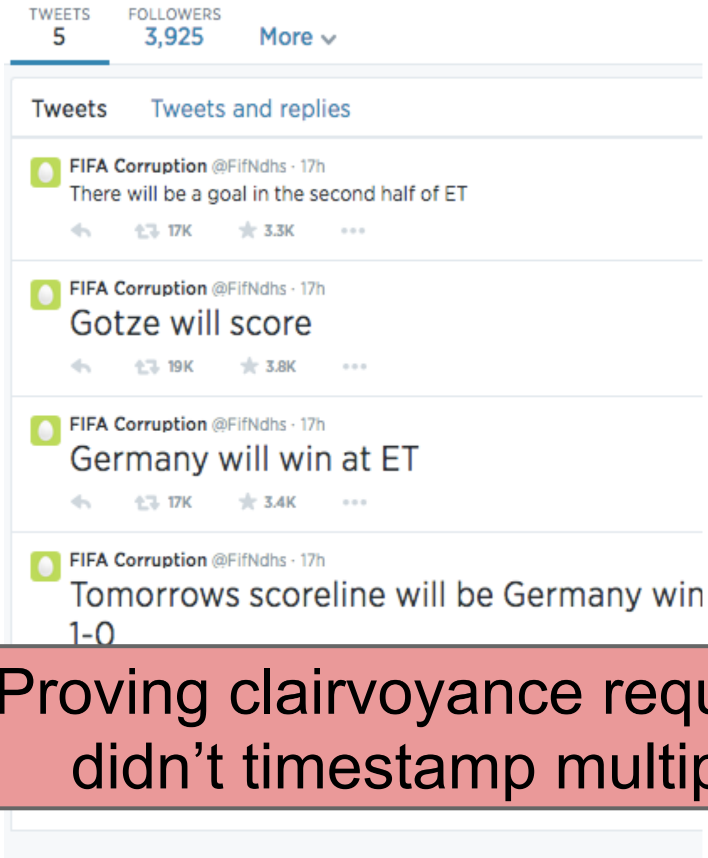
Non-application: proof of clairvoyance

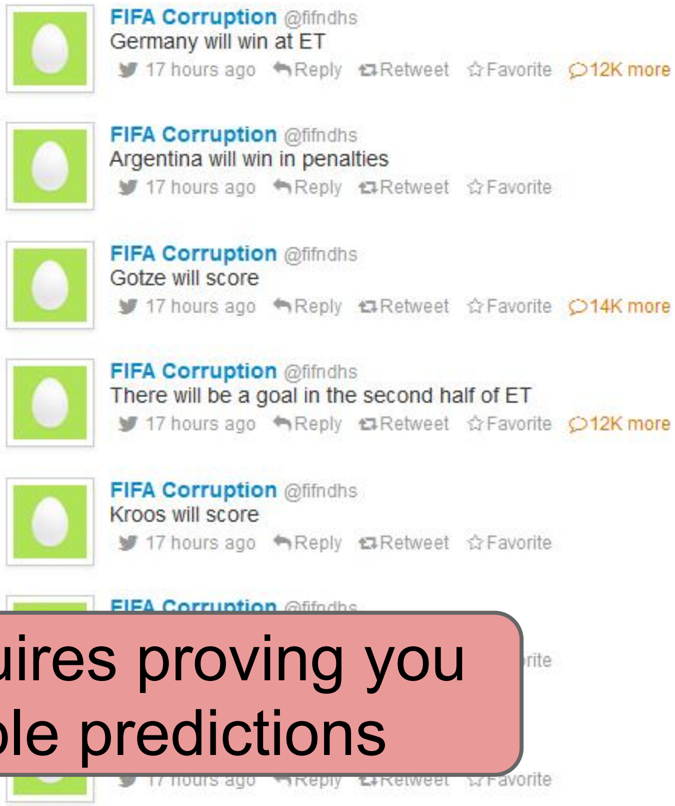
Proof that FIFA is corrupt??



Non-application: proof of clairvoyance

Proof that FIFA is corrupt??





Proving clairvoyance requires proving you didn’t timestamp multiple predictions

Offline solution: newspaper timestamp

Timestamping in Bitcoin

- **Idea**: Specify the hash of your data instead of a valid public key
- Send 1 satoshi to the address

Timestamping in Bitcoin

- **Idea**: Specify the hash of your data instead of a valid public key
- Send 1 satoshi to the address

Pros: compatible, easy
Cons: creates unspendable UTXO forever

Provably unspendable commitments

OP_RETURN
<arbitrary data>

Provably unspendable commitments

```
OP_RETURN
<arbitrary data>
```

Pros: cheap, no UTXO bloat
Cons: not a standard transaction

Block chain poisoning

Overlay currencies

- **Observation**: timestamping is all we need!

- Write all data to the Bitcoin block chain
  - No new mining/consensus required
- Invalid transactions may now be included
  - Need new rules-first valid tx wins

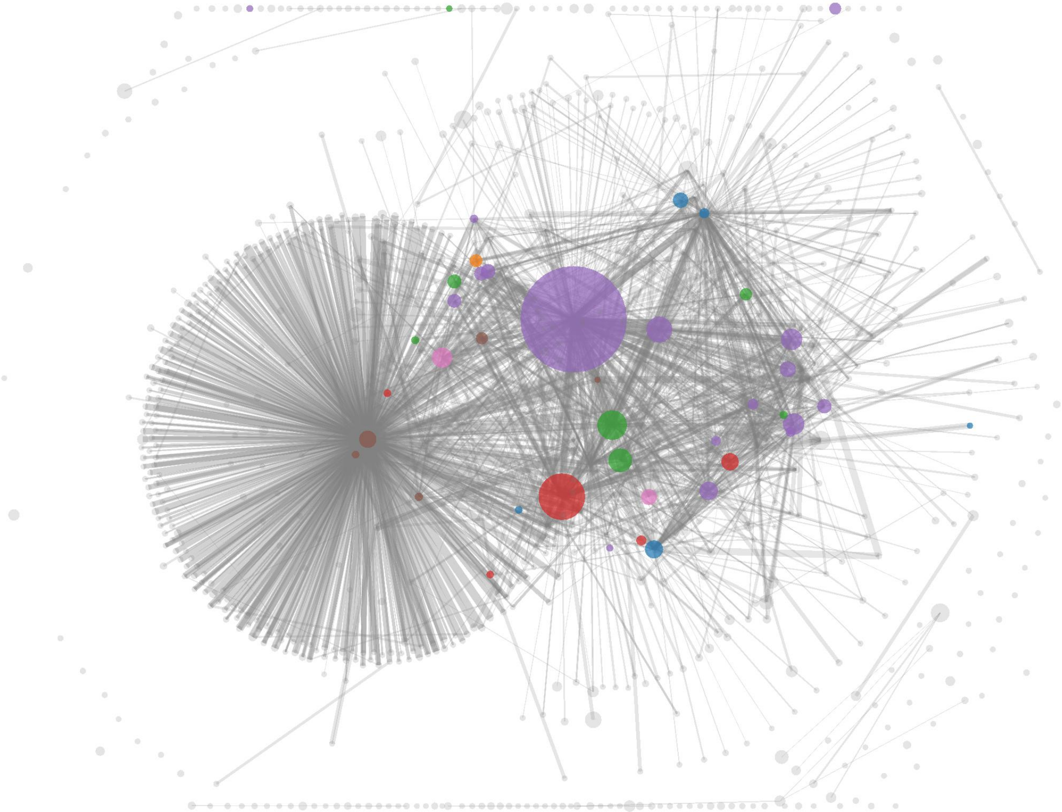
Mastercoin

- **Goals**: overlay currency with richer transaction set
  - Smart property, smart contracts
  - User-defined currency

Pros: more features, faster development
Cons: reliant on Bitcoin, can be inefficient

Bitcoins as “smart property”

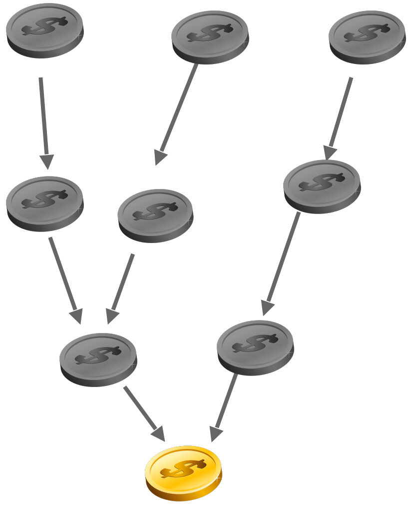
Recall: the transaction graph



Every bitcoin* carries a history

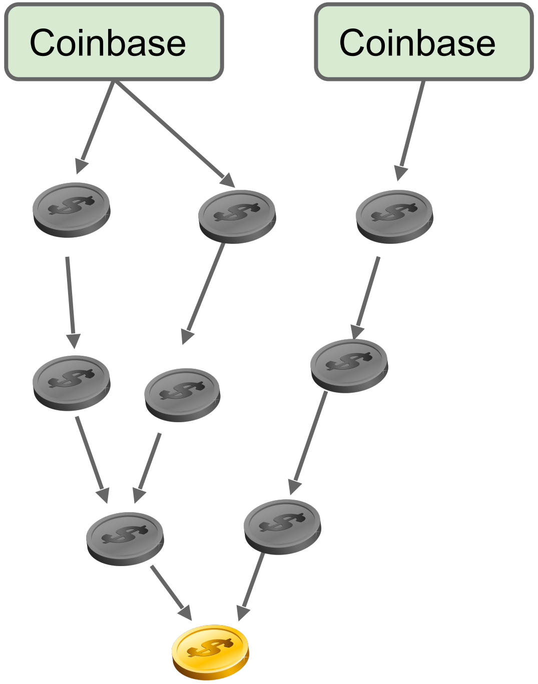
*There are no “bitcoins”, just unspent tx outputs

Every bitcoin* carries a history



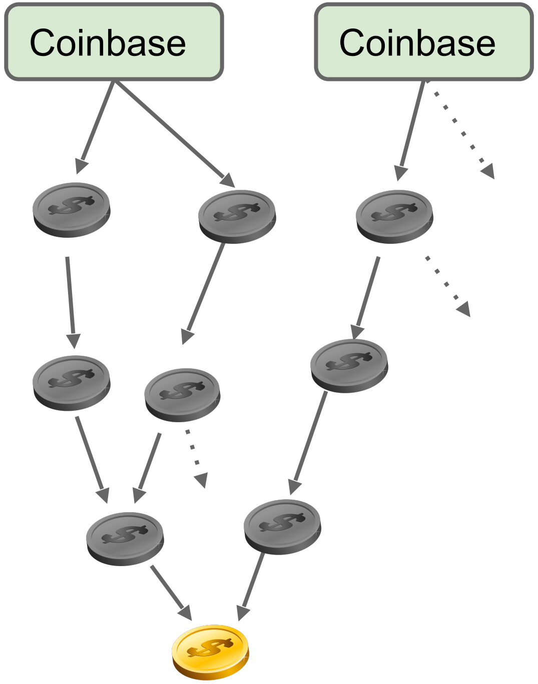
*There are no “bitcoins”, just unspent tx outputs

Every bitcoin* carries a history



*There are no “bitcoins”, just unspent tx outputs

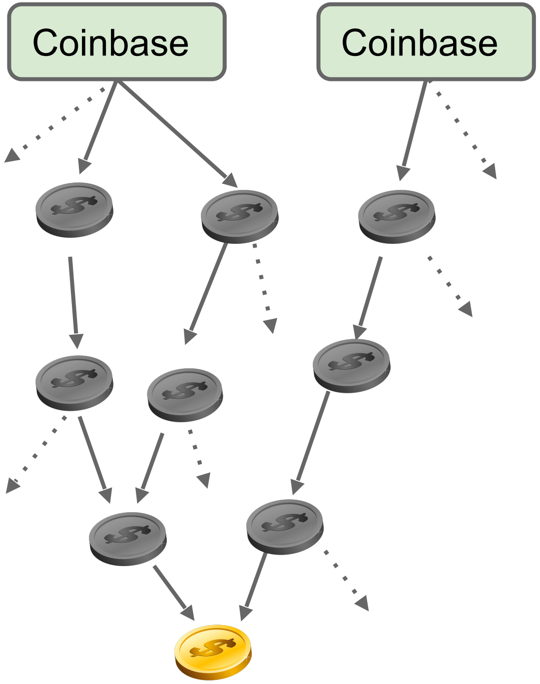
Every bitcoin* carries a history



*There are no “bitcoins”, just unspent tx outputs

Every bitcoin* carries a history

- Bad for anonymity
- Enables blacklisting
- **Observation:** bitcoins aren't fungible! Every one is unique

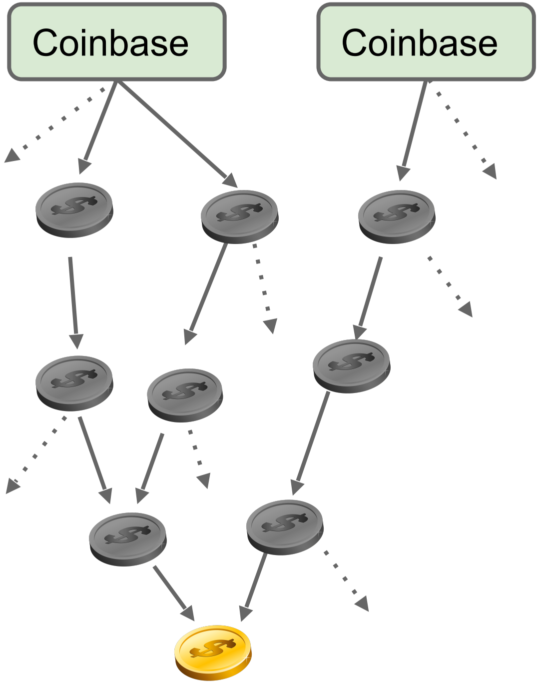


*There are no "bitcoins", just unspent tx outputs

Every bitcoin* carries a history

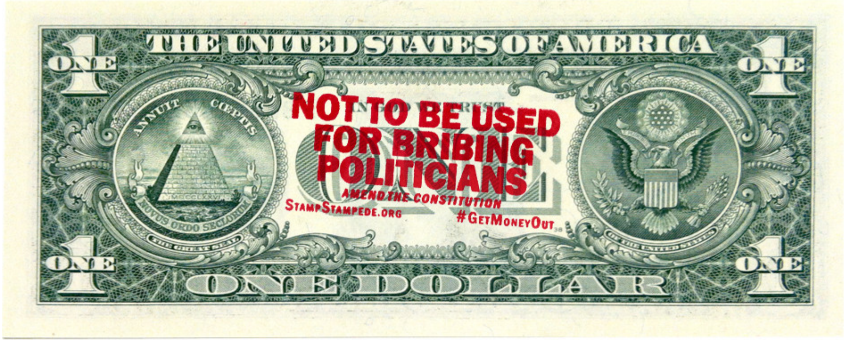
- Bad for anonymity
- Enables blacklisting
- **Observation:** bitcoins aren't fungible! Every one is unique

Can this property be useful?

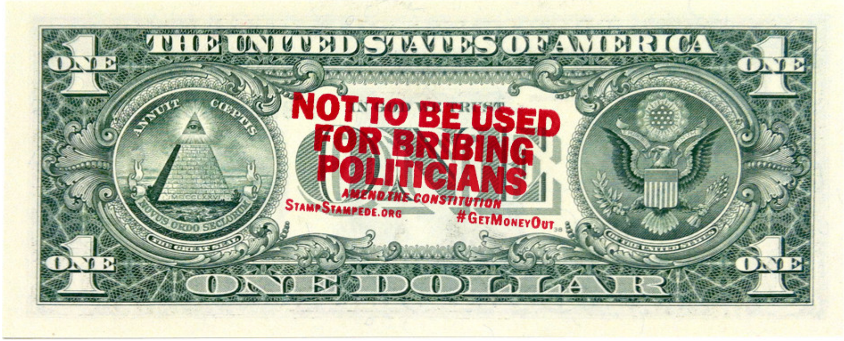


*There are no "bitcoins", just unspent tx outputs

Adding metadata to currency



Adding metadata to currency



Without limitations on issuance, just a novelty

Authenticated metadata for currency

Idea: sign desired metadata + banknote serial #

Authenticated metadata for currency

Idea: sign desired metadata + banknote serial

“Bill #L11180916G hereby grants the holder admission to the Yankees game on Aug 18, 2014”

Stadium

Authenticated metadata for currency

Idea: sign desired metadata + banknote serial

“Bill #L11180916G hereby grants the holder admission to the Yankees game on Aug 18, 2014”

Stadium

$\text{SIGN}_K(M, \#)$

Authenticated metadata for currency

Idea: sign desired metadata + banknote serial

“Bill #L11180916G hereby grants the holder admission to the Yankees game on Aug 18, 2014”

Stadium

$\text{SIGN}_K(M, \#)$

Authenticated metadata for currency

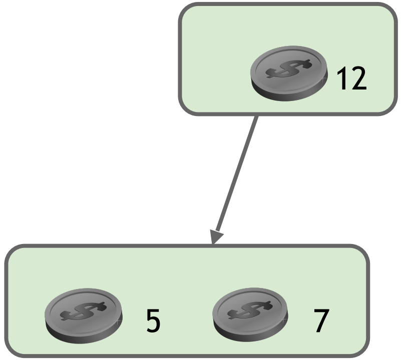
- Currency can now represent anything!
- Anti-counterfeiting properties are inherited
- Underlying value also maintained!
- New meaning relies on trust in the issuer
- Some users may not understand new metadata

Authenticated metadata for currency

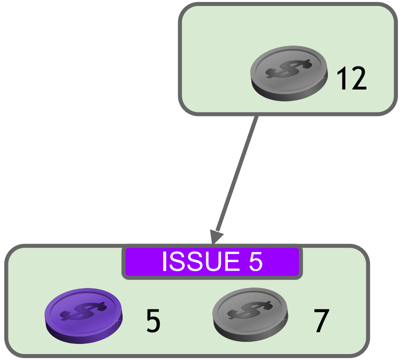
- Currency can now represent anything!
- Anti-counterfeiting properties are inherited
- Underlying value also maintained!
- New meaning relies on trust in the issuer
- Some users may not understand new metadata

Can we build this on top of Bitcoin?

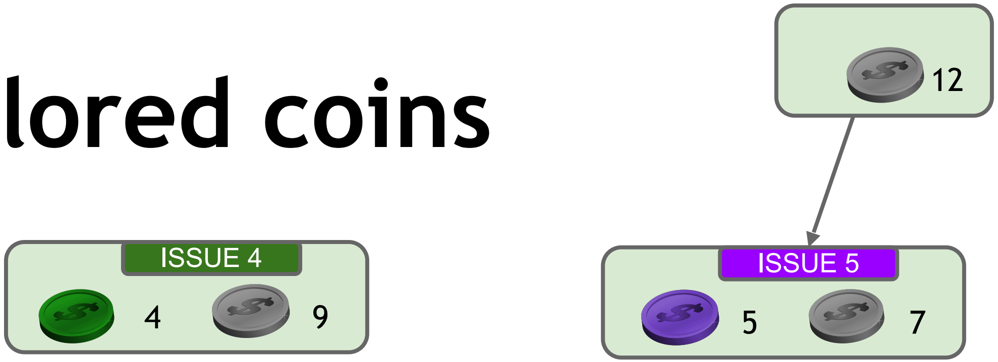
Colored coins



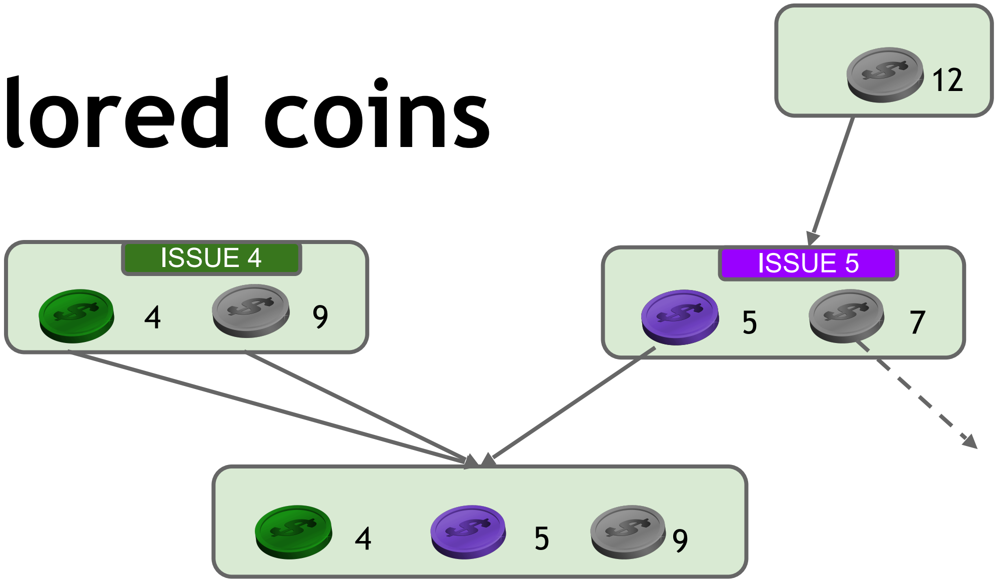
Colored coins



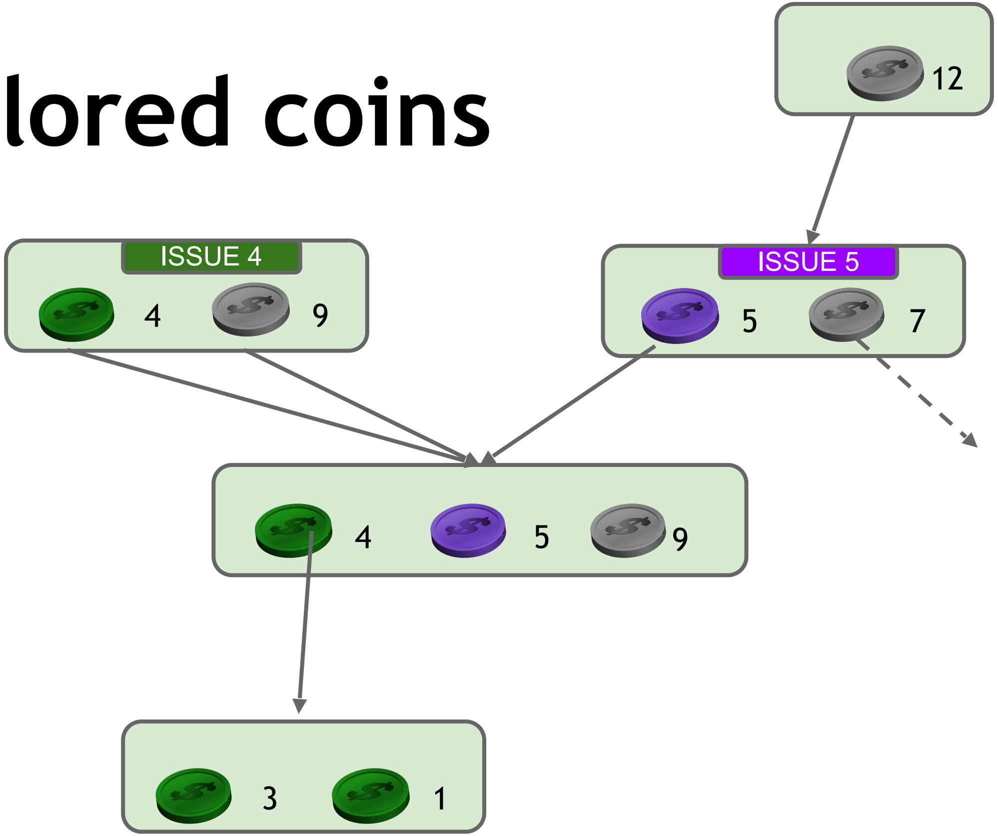
Colored coins



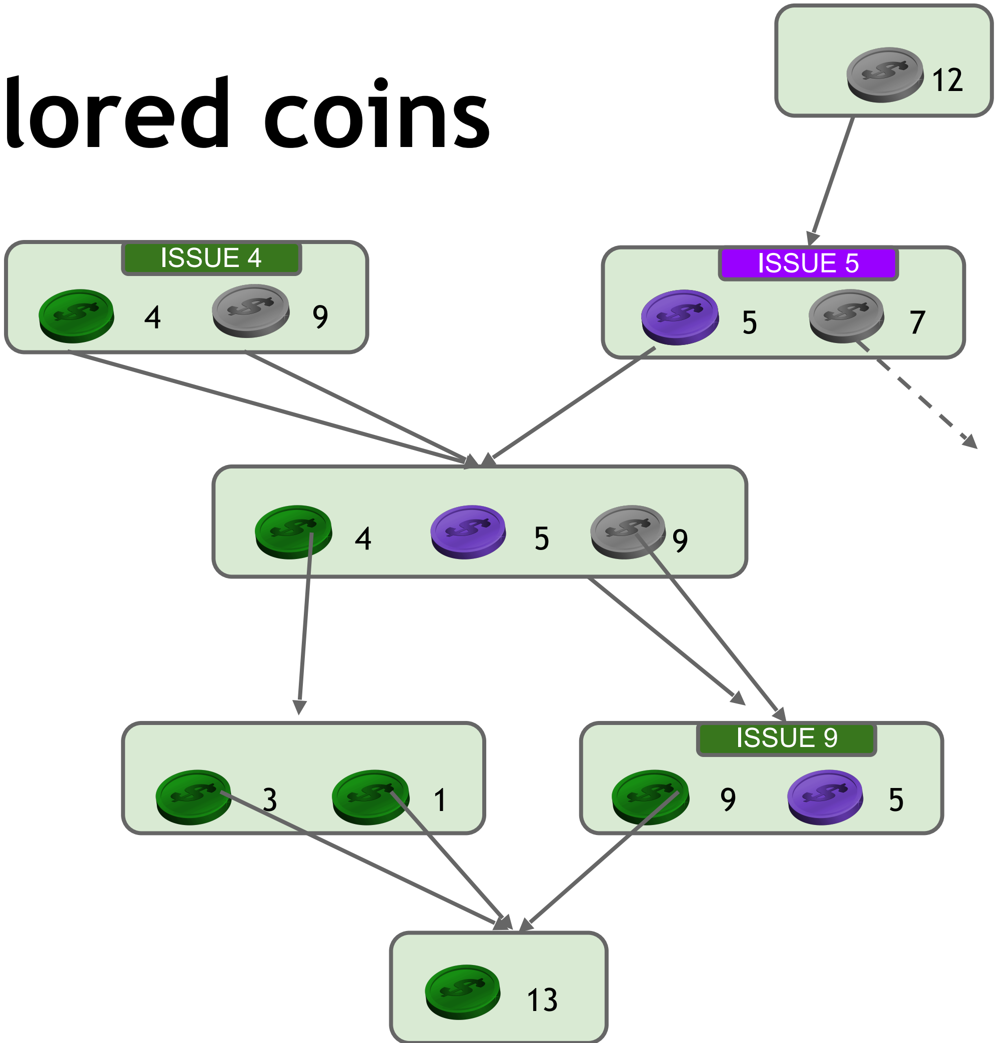
Colored coins



Colored coins



Colored coins



Implementation: OpenAssets protocol

- Coins issued by passing through P2SH address
  - Issuer declares address with an exchange
- Special unspendable “marker” output inserted
  - Match colored inputs to outputs
  - Can add extra metadata

Colored Coins

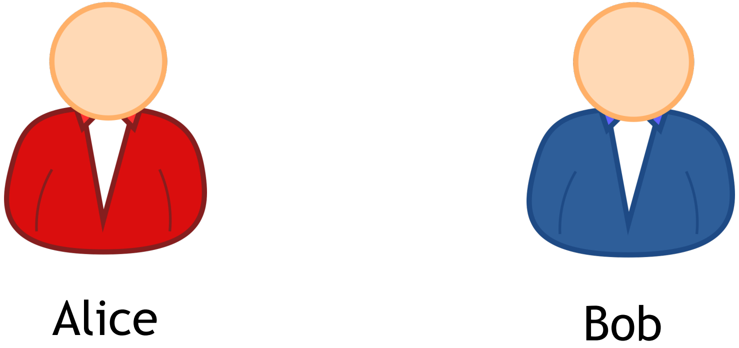
- Pros
  - compatible with Bitcoin
  - flexible to represent any asset
  - ignored by community
- Cons
  - small cost of unspendable markers
  - must check every previous tranasction

Applications

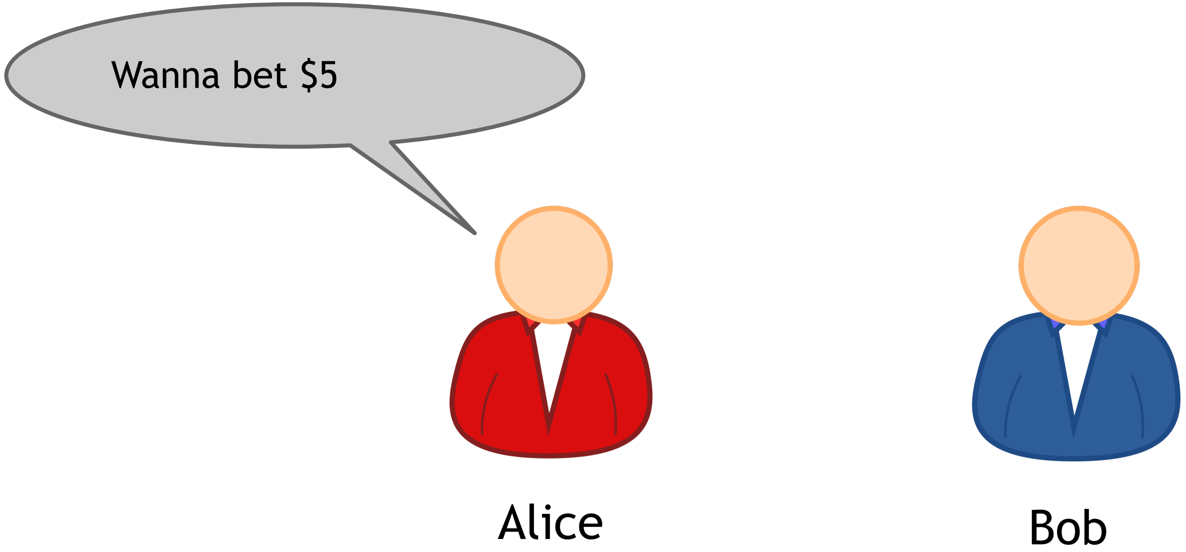
- stock certificates
- tickets
- deeds to real-world property
  - houses?
  - cars?
- ownership of domain names (Namecoin)

Secure multi-party lotteries in Bitcoin

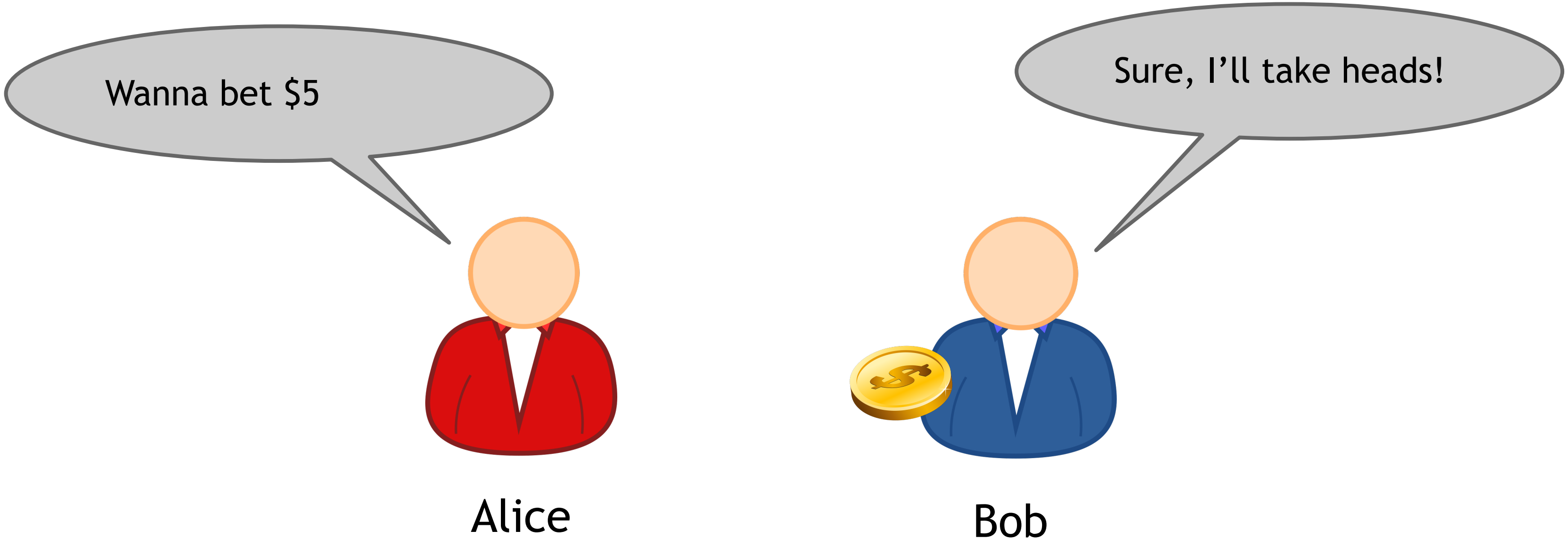
Real-world lotteries without trust*



Real-world lotteries without trust*

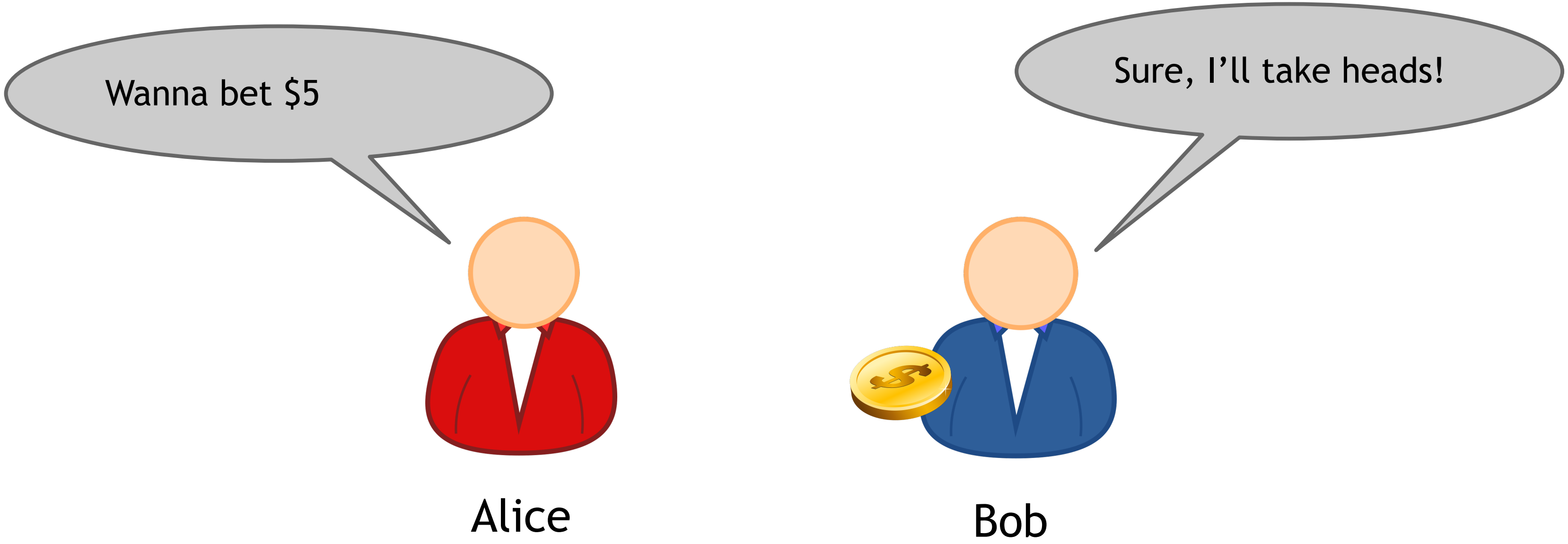


Real-world lotteries without trust*

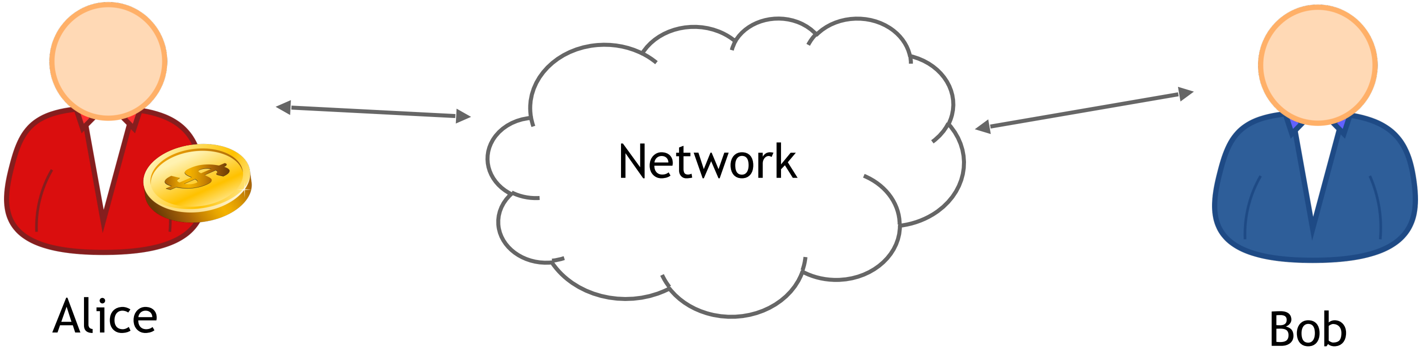


Real-world lotteries without trust*

*The outcome is fair, but both parties have to trust the other will actually pay up

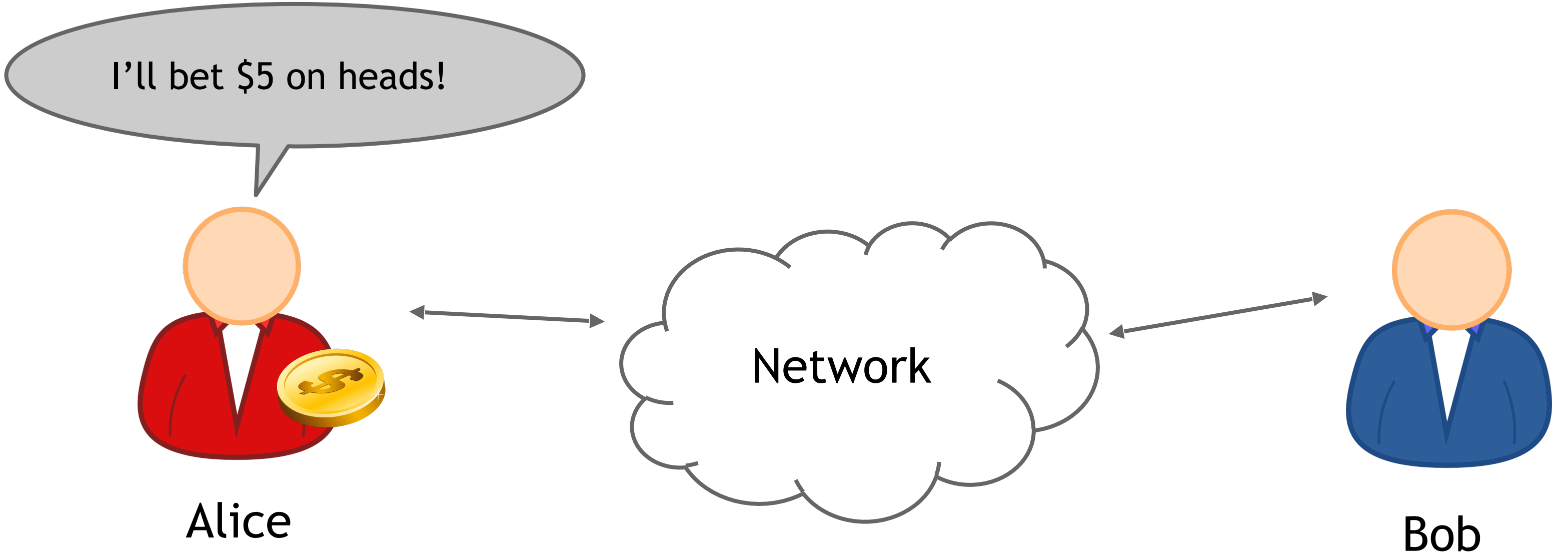


Online lotteries without trust?



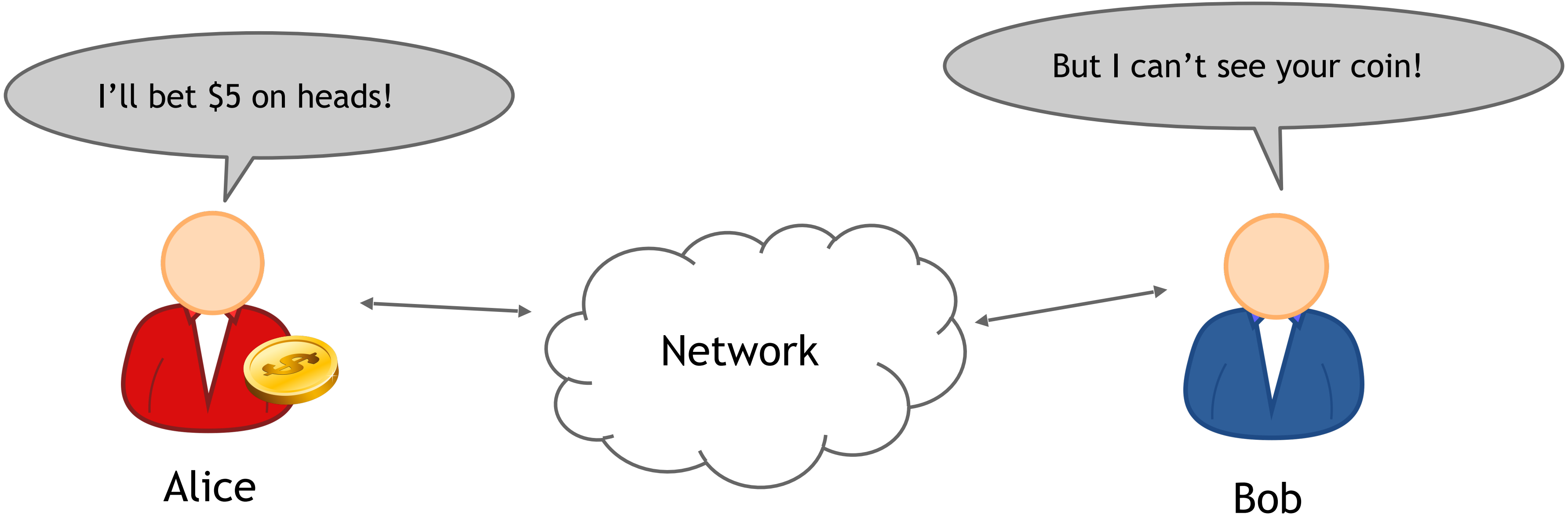
Problem: Alice and Bob want to bet on a coin flip remotely

Online lotteries without trust?



Problem: Alice and Bob want to bet on a coin flip remotely

Online lotteries without trust?



Problem: Alice and Bob want to bet on a coin flip remotely

Hash commitments

Recall: Publishing H(key,*x*) is a *commitment* to *x*

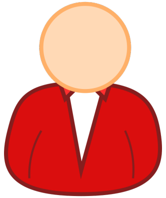
- Can't find an $x' \neq x$ later s.t. $H(key,x') = H(key,x)$
- H(key,*x*) does not reveal *x* in RO model

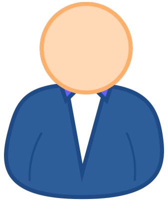
Can publish a commitment to x, reveal later

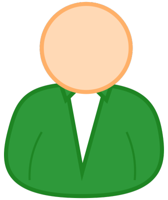
A lottery with commitments

A lottery with commitments

time

Alice  Choose random x

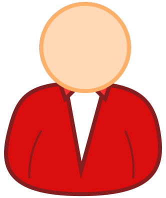
Bob  Choose random y

Carol  Choose random z

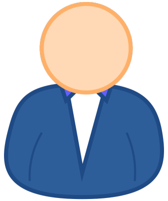
A lottery with commitments

Round 1

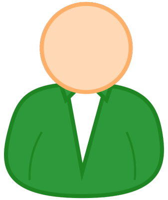
time →

Alice 

Choose random x — Publish H(x)

Bob 

Choose random y — Publish H(y)

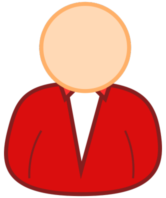
Carol 

Choose random z — Publish H(z)

A lottery with commitments

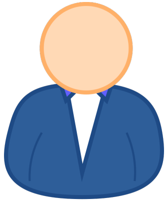
Round 1

time

Alice 

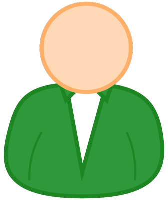
Choose random x

Publish H(x)

Bob 

Choose random y

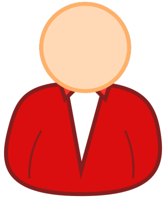
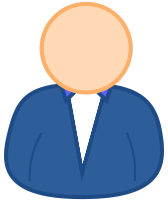
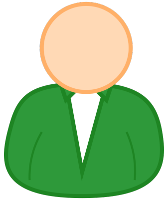
Publish H(y)

Carol 

Choose random z

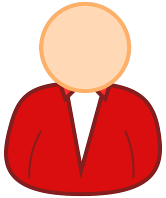
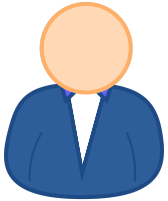
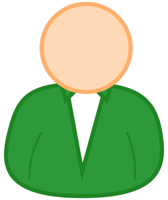
Publish H(z)

A lottery with commitments

			Round 1	Round 2
Alice		Choose random x	Publish H(x)	Publish x
Bob		Choose random y	Publish H(y)	Publish y
Carol		Choose random z	Publish H(z)	Publish z

time

A lottery with commitments

			Round 1	Round 2
Alice		Choose random x	Publish $H(x)$	Publish x
Bob		Choose random y	Publish $H(y)$	Publish y
Carol		Choose random z	Publish $H(z)$	Publish z

time →

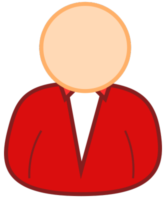
```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

A lottery with commitments

Round 1 **Round 2**

time

Alice

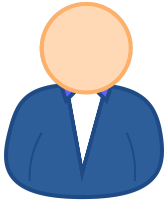


Choose
random

Hash function guarantees nobody can win with probability more than 1/3

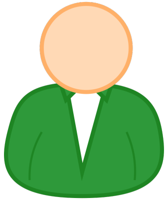
```
H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

Bob



Choose random *y* | Publish H(*y*) | Publish *y*

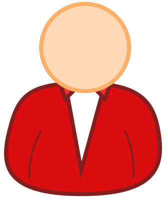
Carol

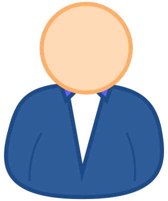


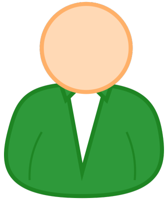
Choose random *z* | Publish H(*z*) | Publish *z*

Failure to reveal commitment

Round 1 | Round 2 | time

Alice

Choose random x — Publish $H(x)$

Bob

Choose random y — Publish $H(y)$

Carol

Choose random z — Publish $H(z)$

```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

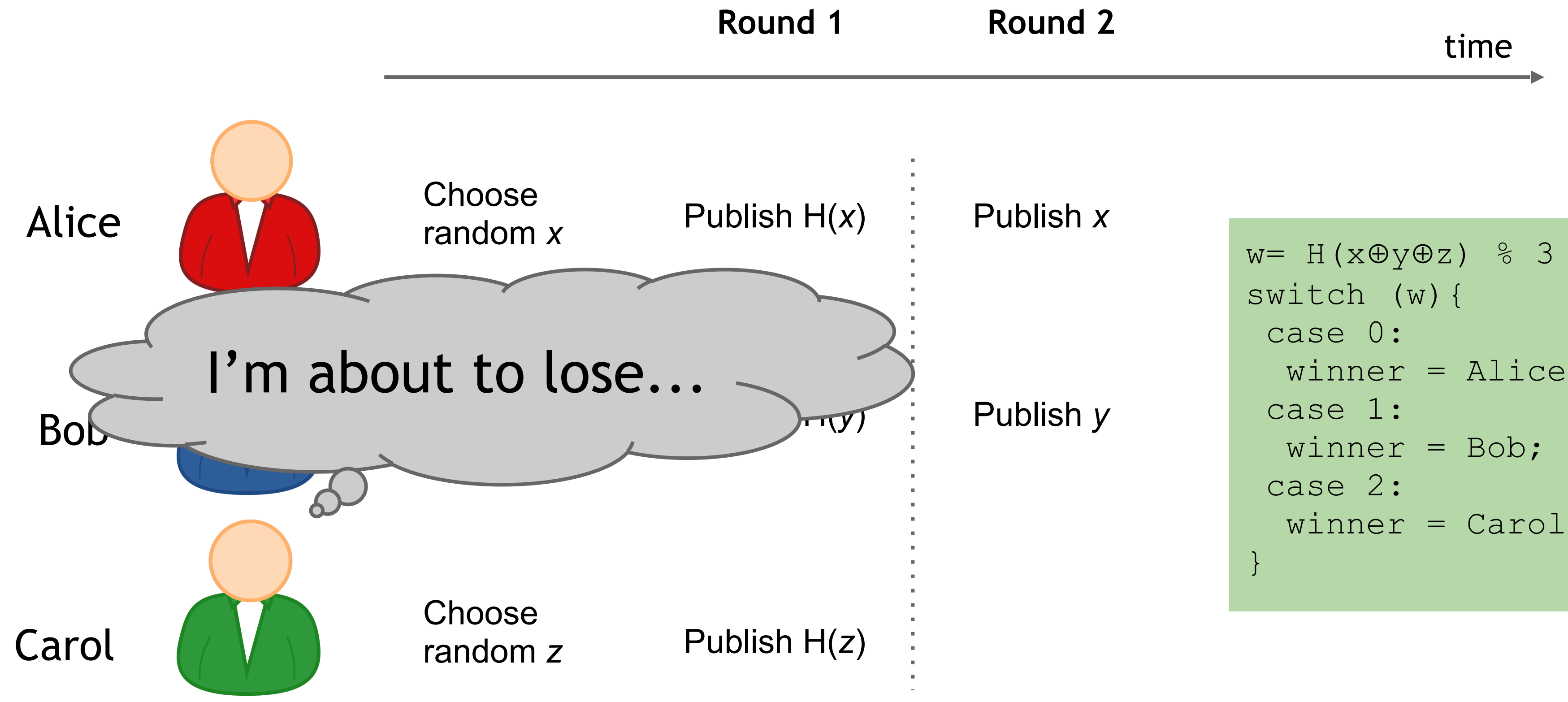
Failure to reveal commitment

	Round 1		Round 2
Alice	Choose random x	Publish $H(x)$	Publish x
Bob	Choose random y	Publish $H(y)$	Publish y
Carol	Choose random z	Publish $H(z)$	

time →

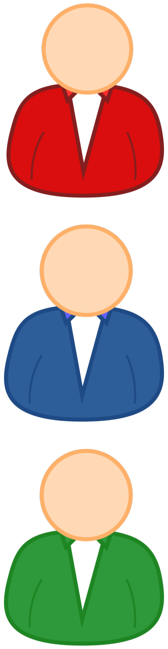
```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

Failure to reveal commitment




```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

Failure to reveal commitment



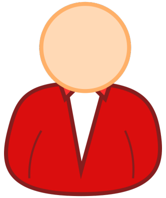
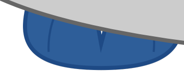
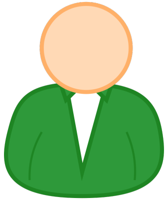
		Round 1	Round 2
Alice	Choose random *x*	Publish H(*x*)	Publish *x*
Bob	Choose random *y*	Publish H(*y*)	Publish *y*
Carol	Choose random *z*	Publish H(*z*)	

time →

```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

Failure to reveal commitment

time →

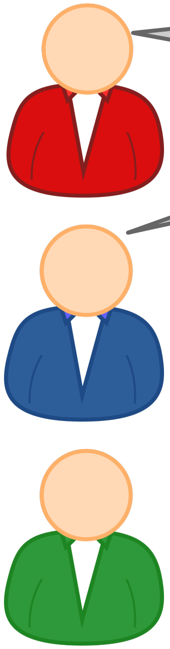
			Round 1	Round 2
Alice		Choose random *x*	Publish H(*x*)	Publish *x*
Bob				Publish *y*
Carol		Choose random *z*	Publish H(*z*)	Ø

Sorry! I, uhh, forgot *z*

```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

Failure to reveal commitment

Not cool Carol ☹️



	Choose	Round 1	Round 2
Alice	rando		
Bob	Choose random y	Publish H(y)	Publish y
Carol	Choose random z	Publish H(z)	Ø

time

```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

Timed hash commitments

Timed hash commitments

Idea: Force *x* to be revealed by time *t*

1 Input: …; Pay *B* to EITHER OF:
Alice & Bob, or
Alice & anybody who knows *x* st. H(*x*) = *c*

SIGNED(Alice)

Timed hash commitments

Idea: Force *x* to be revealed by time *t*

1 Input: ...; Pay *B* to EITHER OF:
Alice & Bob, or
Alice & anybody who knows *x* st. H(*x*) = *c*

SIGNED(Alice)

MULTISIG

New script!

Bond

Timed hash commitments

Idea: Force *x* to be revealed by time *t*

MULTISIG

New script!

1 Input: ...; Pay *B* to EITHER OF:
Alice & Bob, or
Alice & anybody who knows *x* st. H(*x*) = *c*

SIGNED(Alice)

Bond

2 Input: 1; Pay *B* to Bob:
n_lock_time: *t*

SIGNED(Alice) SIGNED(Bob)

Timed hash commitments

Idea: Force *x* to be revealed by time *t*

1 Input: …; Pay *B* to EITHER OF:
Alice & Bob, or
Alice & anybody who knows *x* st. H(*x*) = *c*

SIGNED(Alice)

MULTISIG

New script!

Bond

2 Input: 1; Pay *B* to Bob:
n_lock_time: *t*

SIGNED(Alice) SIGNED(Bob)

Bob can claim the bond at time *t*

Timed hash commitments

Idea: Force *x* to be revealed by time *t*

MULTISIG

New script!

1 Input: ...; Pay *B* to EITHER OF:
Alice & Bob, or
Alice & anybody who knows *x* st. H(*x*) = *c*

SIGNED(Alice)

Bond

Bob can claim the bond at time *t*

2 Input: 1; Pay *B* to Bob:
n_lock_time: *t*

SIGNED(Alice) SIGNED(Bob)

3 Input: 1; Pay *B* to Alice:

SIGNED(Alice), *x*

Timed hash commitments

Idea: Force *x* to be revealed by time *t*

MULTISIG

New script!

1 Input: …; Pay *B* to EITHER OF:
Alice & Bob, or
Alice & anybody who knows *x* st. $H(x) = c$

SIGNED(Alice)

Bond

Bob can claim the bond at time *t*

2 Input: 1; Pay *B* to Bob:
n_lock_time: *t*

SIGNED(Alice) SIGNED(Bob)

x revealed if Alice reclaims her bond

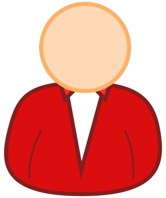
3 Input: 1; Pay *B* to Alice:

SIGNED(Alice), *x*

Lottery with timed commitments

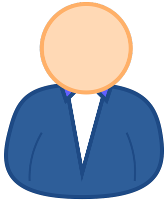
Round 1 **Round 2**

time

Alice 

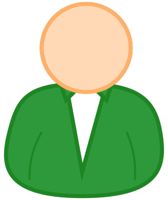
Choose random x

Timed commitment to H(x)

Bob 

Choose random y

Timed commitment to H(y)

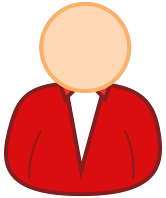
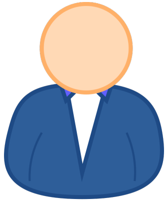
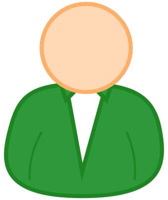
Carol 

Choose random z

Timed commitment to H(z)

```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

Lottery with timed commitments

			Round 1	Round 2 → time
Alice		Choose random x	Timed commitment to H(x)	Publish x
Bob		Choose random y	Timed commitment to H(y)	Publish y
Carol		Choose random z	Timed commitment to H(z)	

```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

Lottery with timed commitments

Round 1 **Round 2**

time

Alice — Choose random x — Timed commitment to H(x) — Publish x

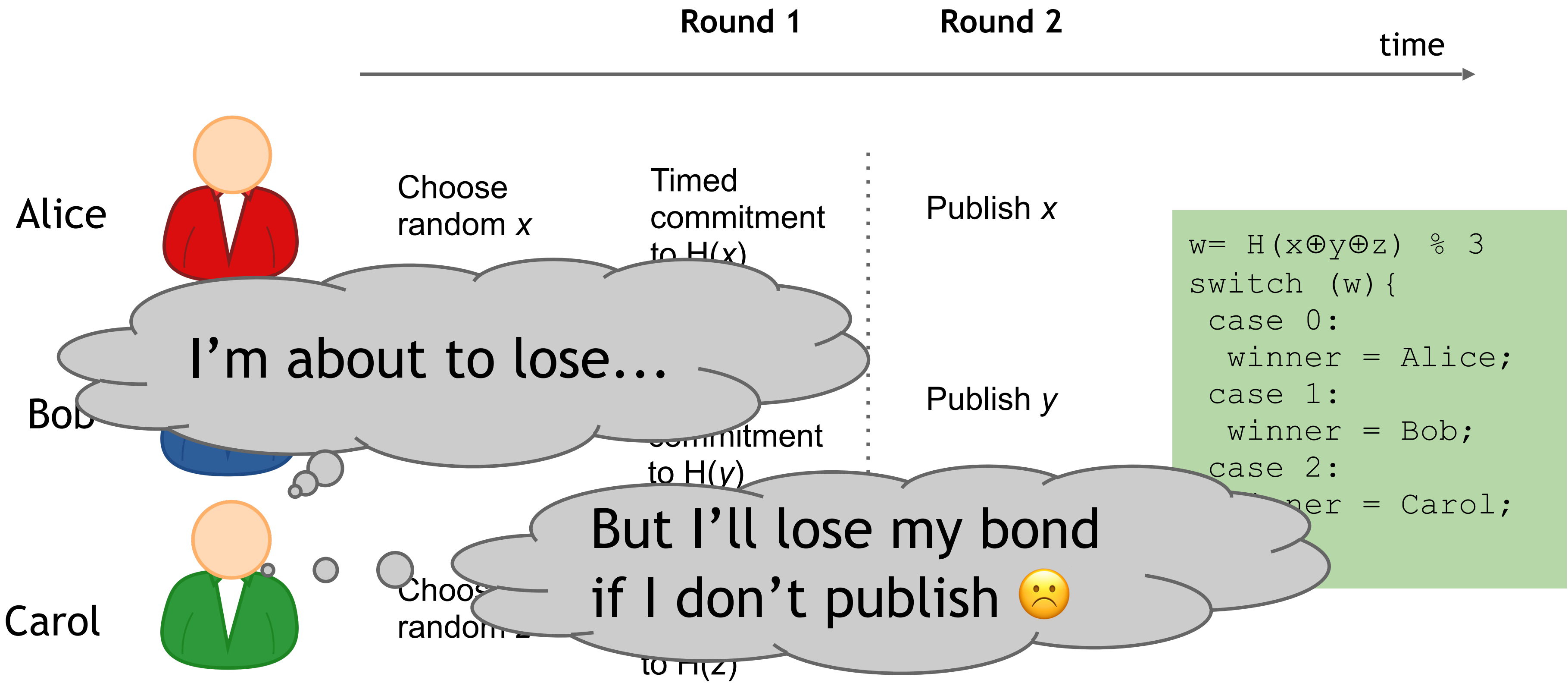
I'm about to lose...

Bob — commitment to H(y) — Publish y

Carol — Choose random z — Timed commitment to H(z)

```
w= H(x⊕y⊕z) % 3
switch (w){
 case 0:
  winner = Alice;
 case 1:
  winner = Bob;
 case 2:
  winner = Carol;
}
```

Lottery with timed commitments



Lottery with timed commitments

Pros:

- can be implemented on Bitcoin today
  - Andrychowicz, Dziembowski, Malinowski, Mazurek 2014

Cons:

- complexity is $O(N^2)$
- bonds must be higher than amount bet

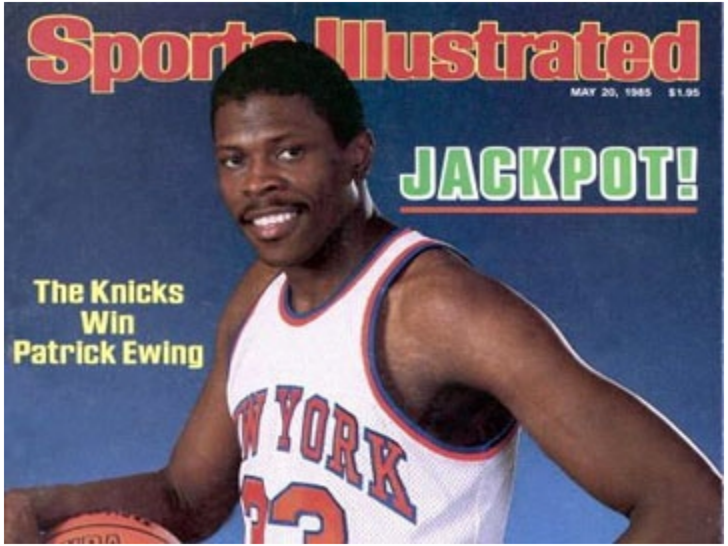
Bitcoin as randomness source

Public randomness protocols

- Interactive coin-tossing protocols known in the literature
- “Non-interactive” source of convincing randomness?

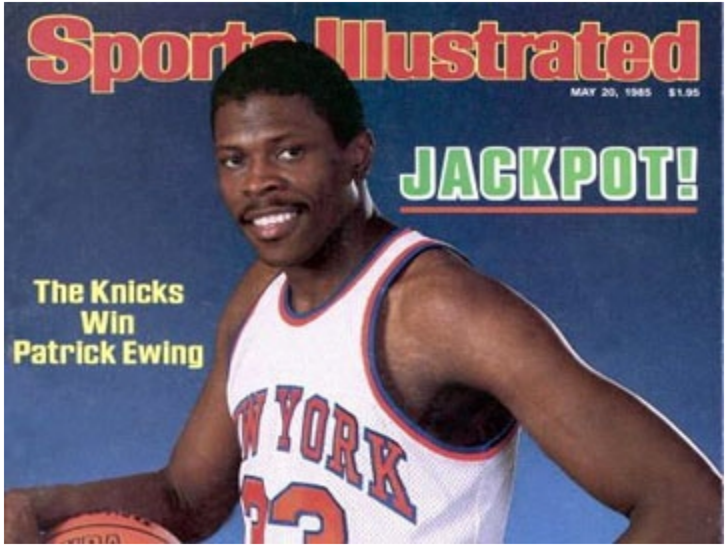
NBA draft lottery

NBA draft lottery



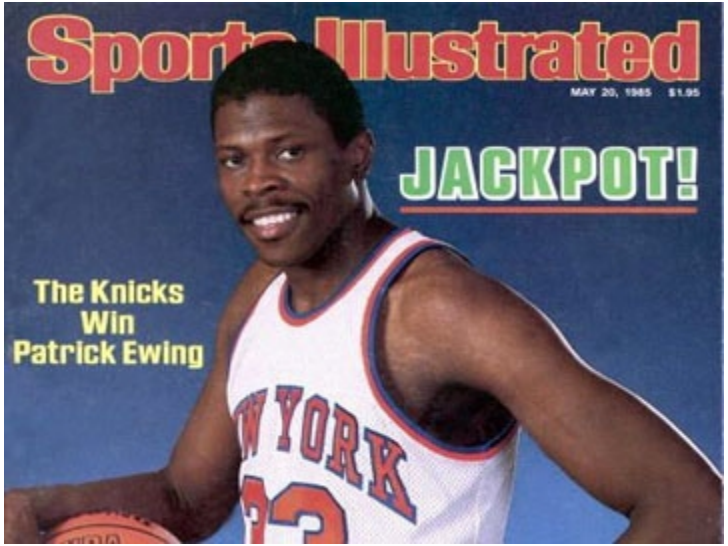
1985: Knicks win rights to Patrick Ewing

NBA draft lottery



1985: Knicks win rights to Patrick Ewing

NBA draft lottery



1985: Knicks win rights to
Patrick Ewing

INTERNATIONAL BUSINESS TIMES

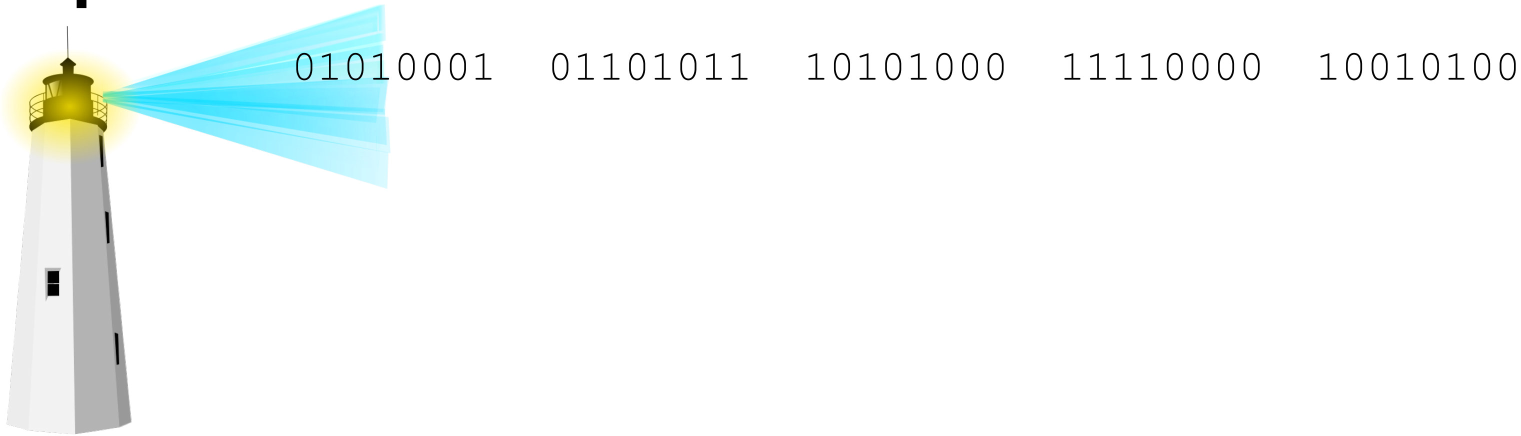
NBA Lottery 2014: Conspiracy Theories Plague Annual Event

By Anthony Riccobono @tony_riccobono a.riccobono@ibtimes.com
on May 20 2014 1:35 PM

Cryptographic beacons

Idea: service to regularly publish random data

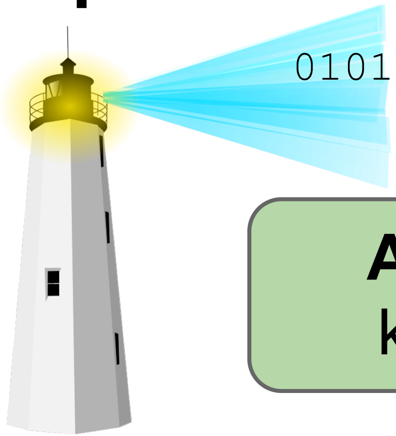
- **Uniform randomness**
- **No party can predict in advance**
- **All parties see the same values**



Cryptographic beacons

Idea: service to regularly publish random data

- **Uniform randomness**
- **No party can predict in advance**
- **All parties see the same values**

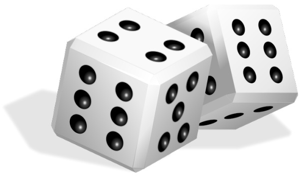
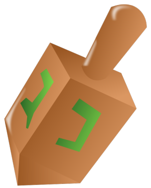


01010001 01101011 10101000 11110000 10010100

Applications: lotteries, auditing, zero-knowledge proofs, cut-and-choose, ...

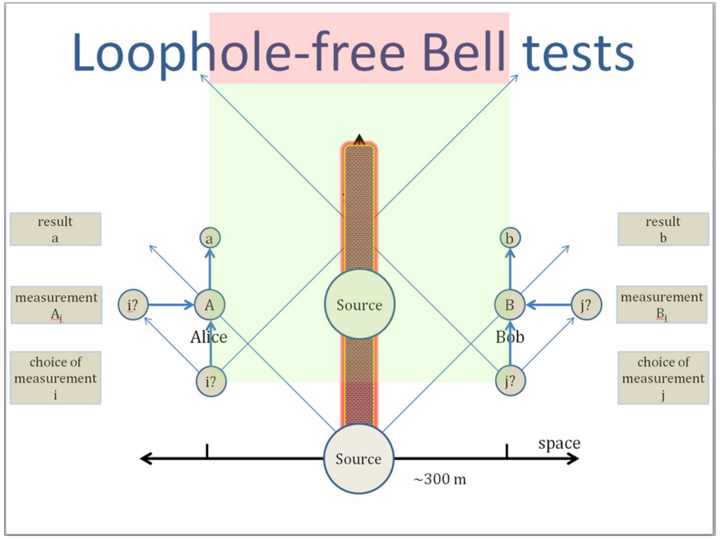
Public display of randomness



Pros: cheap, easy, simple to understand
Cons: must trust/audit operator
hard to trust remotely!

NIST beacon



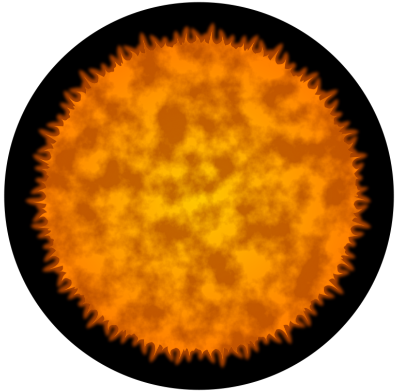
Beacon Record

Version:	Version 1.0
Frequency:	60 seconds
Time:	08/13/2014 12:36 pm (1407947760)
Seed Value:	27D7280A657B5E0A99721D47E21A2276C80B5CDFDCA605E397D8BBAA51C24A06 40CC9C6EEB83BBB3D837011CA5B6CA08FADC78E2B8D36C75CC971757F82068A4
Previous Output:	2F2DE0662028D3C4D6F8DD7936262D9AFBDCFD0BD14BC733E257B14F48881A99 206BBC9429FD9BFE719551EAB840CEE8157ACAEBC80342CE4B66443C0859E216
Signature:	986C73CF88056635C5E0A018358D0D91CF10A2F2B16C8B8D91AA34B0A04D103B CFF347B714DAC343D5838E07FFDFC49BE6E39811350DC0193D17CFE1BC4EDB5B 7E3AC425EF7840EF4E549D66D0F0FB383DD9F29DFDAEF2E520B8606A4F6C55FB 3B766CC9D66494FAC1FE8983D58525224778F5AE3C3727FF0AC71DCE3B30E33B A6CFD767EE3D299A5324E371AFB49AEC46F88D6DCAE6FCBF8B93D461B84C59CB 7577BE9A63FE0DB7C83944B545C501A4C787F87B15A0F8CFD8FB7FC191F677FB C4FB1C07E47C01B0D090BAC564FEAFBD0E24D90F01DE2B2E66A31E7012CACD42 30EA94EF415C8F2B1751F09BD8255A2C142CE2C8C69587EE6CE788273E55AFA7
Output Value:	15E3B39DA53DE7C20A60D3EC2DECC2C6B2DB65FE07B1188D666A8A8476E4910F 592FB3F8D49E4A01E5624FDF161A698EB0AA52515A79A46F3AFA1B8D7CEBB320
Status:	0: Normal

Pros: quantum-mechanical randomness

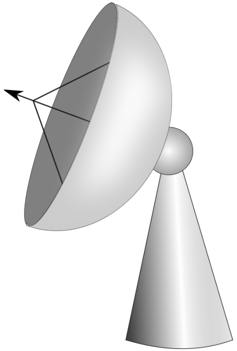
Cons: must trust NIST

Natural phenomena



Sun spots

Weather

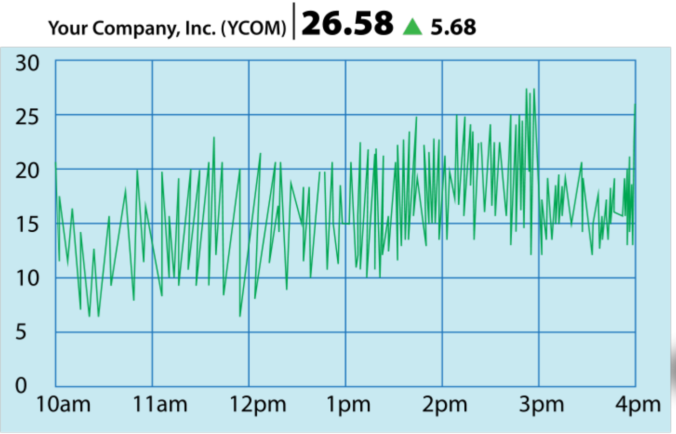


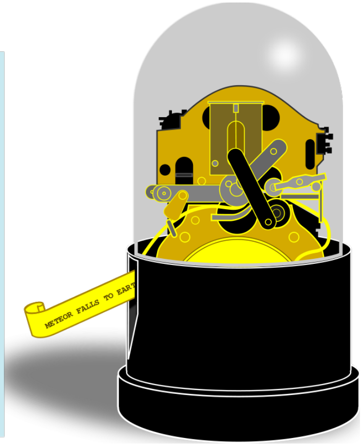
Cosmic background radiation

Pros: publicly observable, random

Cons: slow, need a trusted observer?

Stock-market beacon






Pros: good randomness, costly to manipulate

Cons: slow, insider attacks?

Why not use the block chain?

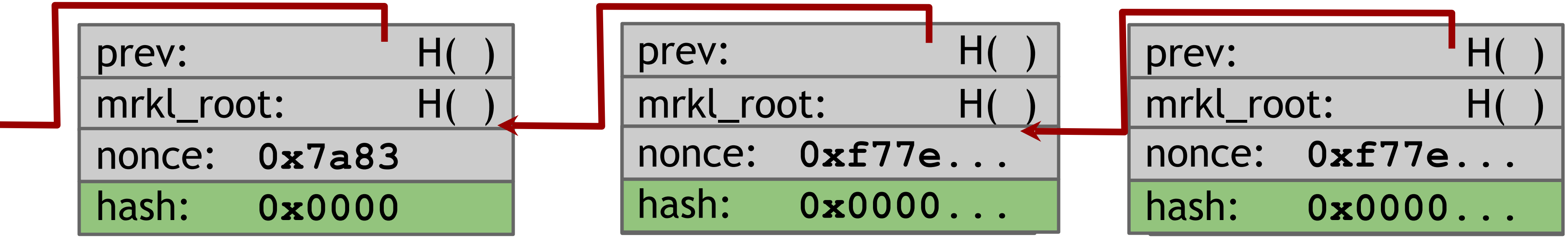
Recall: miners find random nonce for each block

Why not use the block chain?

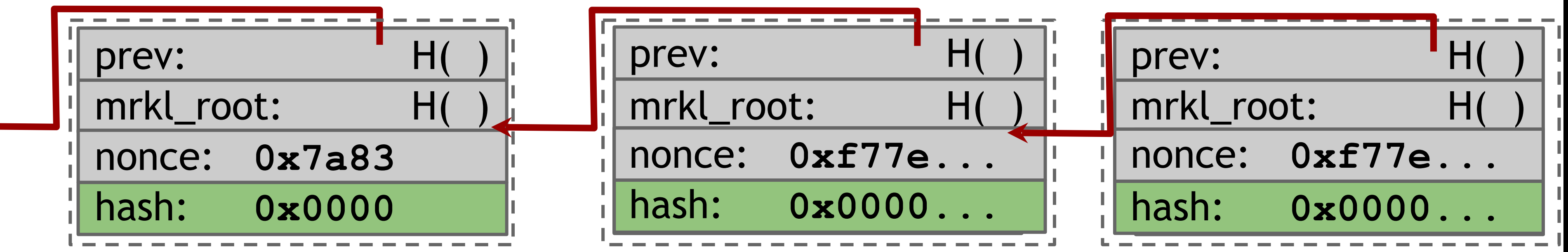
Recall: miners find random nonce for each block

If you could predict the next nonce with a greater than **1/d** probability, you'd have a mining shortcut

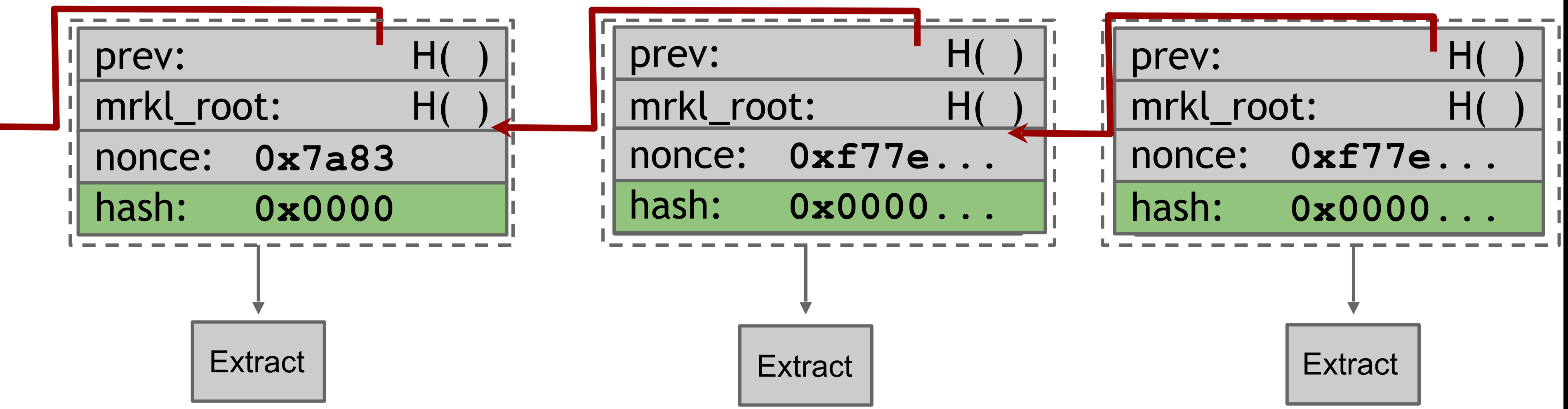
Turning the block chain into a beacon



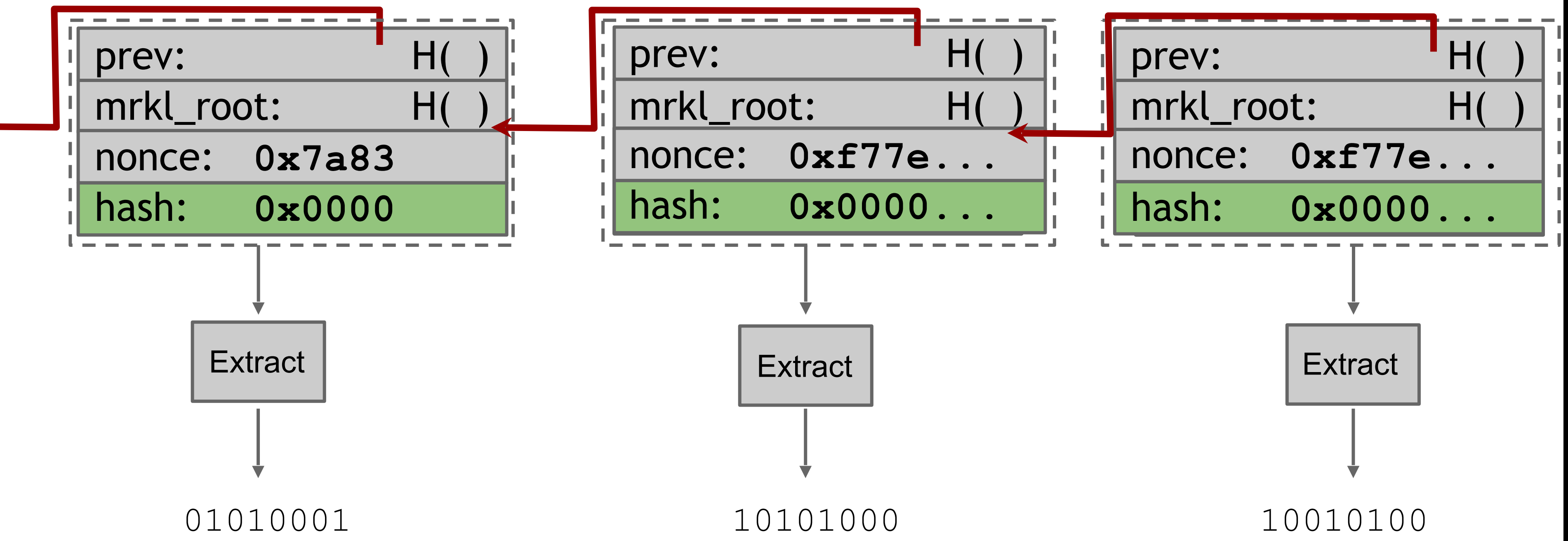
Turning the block chain into a beacon



Turning the block chain into a beacon



Turning the block chain into a beacon



Cost of manipulation

Attacker might mine a block but discard it

- Or bribe other miners to do so

Bernoulli trials: forcing a beacon outcome with probability p requires discarding $1/p - 1$ blocks

Discarding a block “costs” 12.5 BTC

Cost of manipulation

Single coin flip: "secure" if wager is < 12.5 BTC

N-party lottery: "secure" if pool is < 12.5 (*n*-1) BTC

Pros

- Decentralized beacon
- Output every 10 minutes
- Can precisely analyze manipulation costs
- Can extend security with multiple blocks
  - Not very efficient

Cons

- Timing is imprecise
    - Block chain not synchronized w/ real time
- Need to delay to insure against forks
- Manipulation may be too cheap for some applications

Cons

- Timing is imprecise
  - Block chain not synchronized w/ real time
- Need to delay to insure against forks
- Manipulation may be too cheap for some applications