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, **without** revealing $x$ at time $t$

Evidence should be permanent
Hash commitments

Recall: Publishing $H(\text{key}, x)$ is a commitment to $x$

- Can’t find an $x' \neq x$ later s.t. $H(\text{key}, x') = H(\text{key}, x)$
- $H(\text{key}, x)$ 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

Matt
@Cheesegod69

apparently someone embedded child porn in the bitcoin block chain, storing it on every bitcoin user’s computer

bitcointalk.org/index.php?topi...

Travis Goodspeed
@travisgoodspeed

Some jerk injected pedo links into the Bitcoin block chain. So it goes.
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”
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”
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

ISSUE 4
4 9

ISSUE 5
5 7

ISSUE 9
3 1

13
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 transaction
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*

Wanna bet $5

Alice

Bob
Real-world lotteries without trust*

Wanna bet $5

Sure, I’ll take heads!
Real-world lotteries without trust*

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

Wanna bet $5

Sure, I’ll take heads!

Alice

Bob
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.

I’ll bet $5 on heads!
Online lotteries without trust?

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

I’ll bet $5 on heads!

But I can’t see your coin!

Network
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

- Alice
- Bob
- Carol
A lottery with commitments

Choose random $x$

Choose random $y$

Choose random $z$
A lottery with commitments

Round 1

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

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
Alice: Choose random $x$  
Bob: Choose random $y$  
Carol: Choose random $z$

Round 2
Alice: Publish $H(x)$  
Bob: Publish $H(y)$  
Carol: Publish $H(z)$

Round 2
Alice: Publish $x$  
Bob: Publish $y$  
Carol: Publish $z$
A lottery with commitments

**Round 1**
- Alice: Choose random $x$ → Publish $H(x)$
- Bob: Choose random $y$ → Publish $H(y)$
- Carol: Choose random $z$ → Publish $H(z)$

**Round 2**
- $w = H(x \oplus y \oplus z) \mod 3$
- switch ($w$){
  - case 0:
    - winner = Alice;
  - case 1:
    - winner = Bob;
  - case 2:
    - winner = Carol;
}
A lottery with commitments

Round 1

Choose random $x$

Publish $H(x)$

Publish $x$

Round 2

Choose random $y$

Publish $H(y)$

Publish $y$

Choose random $z$

Publish $H(z)$

Publish $z$

H(x⊕y⊕z) % 3

switch (w){
  case 0:
    winner = Alice;
  case 1:
    winner = Bob;
  case 2:
    winner = Carol;
}

Hash function guarantees nobody can win with probability more than 1/3
Failure to reveal commitment

Round 1

Alice
Choose random $x$
Publish $H(x)$

Bob
Choose random $y$
Publish $H(y)$

Carol
Choose random $z$
Publish $H(z)$

Round 2

Time

$w = H(x \oplus y \oplus z) \mod 3$

switch ($w$) {
  case 0:
    winner = Alice;
  case 1:
    winner = Bob;
  case 2:
    winner = Carol;
}
Failure to reveal commitment

Round 1

<table>
<thead>
<tr>
<th>Alice</th>
<th>Bob</th>
<th>Carol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choose random $x$</td>
<td>Choose random $y$</td>
<td>Choose random $z$</td>
</tr>
<tr>
<td>Publish $H(x)$</td>
<td>Publish $H(y)$</td>
<td>Publish $H(z)$</td>
</tr>
</tbody>
</table>

Round 2

<table>
<thead>
<tr>
<th>Alice</th>
<th>Bob</th>
<th>Carol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Publish $x$</td>
<td>Publish $y$</td>
<td></td>
</tr>
</tbody>
</table>

$$w = H(x \oplus y \oplus z) \mod 3$$

switch (w){
  case 0:
    winner = Alice;
  case 1:
    winner = Bob;
  case 2:
    winner = Carol;
}"
Failure to reveal commitment

Round 1
- Publish $H(x)$
- Publish $H(y)$
- Publish $H(z)$

Round 2
- Publish $x$
- Publish $y$
- Publish $z$

$w = H(x \oplus y \oplus z) \mod 3$

switch ($w$){
  case 0:
    winner = Alice;
  case 1:
    winner = Bob;
  case 2:
    winner = Carol;
}

I'm about to lose...
Failure to reveal commitment

Round 1

Alice: Choose random $x$ → Publish $H(x)$
Bob: Choose random $y$ → Publish $H(y)$
Carol: Choose random $z$ → Publish $H(z)$

Round 2

Publish $x$
Publish $y$

w = $H(x \oplus y \oplus z) \mod 3$

switch (w){
case 0:
    winner = Alice;
case 1:
    winner = Bob;
case 2:
    winner = Carol;
}


Failure to reveal commitment

Round 1

Choose random $x$
Publish $H(x)$

Round 2

Publish $x$

$w = H(x \oplus y \oplus z) \mod 3$
switch ($w$){
case 0:
  winner = Alice;
case 1:
  winner = Bob;
case 2:
  winner = Carol;
}

Bob

Carol

Choose random $z$
Publish $H(z)$

Ø

Sorry! I, uhh, forgot $z$
Failure to reveal commitment

Round 1

Choose random $y$

Choose random $z$

Round 2

$w = H(x \oplus y \oplus z) \mod 3$

switch ($w$){
  case 0:
    winner = Alice;
  case 1:
    winner = Bob;
  case 2:
    winner = Carol;
}

Not cool 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 $x$ s.t. $H(x) = c$

   SIGNED(Alice)
Timed hash commitments

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

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

MULTISIG

New script!

Bond

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 $x$ s.t. $H(x) = c$

2. Input: 1; Pay $B$ to Bob:
   - $n_{lock\_time}: t$

MULTISIG  

New script!
Timed hash commitments

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

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

2. Input: 1; Pay $B$ to Bob:
   - $n_{lock\_time}: t$
   - $SIGNED(Alice) \ SIGNED(Bob)$

MULTISIG

New script!

Bob can claim the bond at time $t$
Timed hash commitments

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

1. Input: $\ldots$; Pay $B$ to EITHER OF:
   - Alice & Bob, or
   - Alice & anybody who knows $x$ s.t. $H(x) = c$
   - SIGNED(Alice)

2. Input: 1; Pay $B$ to Bob:
   - $n_{\text{lock\_time}}: t$
   - 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 $t$

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

2. Input: 1; Pay B to Bob:
   - $n_{lock\_time}: t$

3. Input: 1; Pay B to Alice:
   - $x$ revealed if Alice reclaims her bond
Lottery with timed commitments

Round 1
- Choose random $x$
- Timed commitment to $H(x)$
- Choose random $y$
- Timed commitment to $H(y)$
- Choose random $z$
- Timed commitment to $H(z)$

Round 2
$w = H(x \oplus y \oplus z) \mod 3$
```
switch (w){
case 0:
  winner = Alice;
case 1:
  winner = Bob;
case 2:
  winner = Carol;
}
```
Lottery with timed commitments

Round 1

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 $H(z)$

Round 2

Publish $x$

Publish $y$

$w = H(x ⊕ y ⊕ z) \mod 3$

switch ($w$){
case 0:
   winner = Alice;
case 1:
   winner = Bob;
case 2:
   winner = Carol;
}
Lottery with timed commitments

Round 1

Choose random $x$

Timed commitment to $H(x)$

Publish $x$

Round 2

Choose random $y$

Timed commitment to $H(y)$

Publish $y$

Choose random $z$

Timed commitment to $H(z)$

I’m about to lose...

Time

$w = H(x \oplus y \oplus z) \mod 3$

switch ($w$){
  case 0:
    winner = Alice;
  case 1:
    winner = Bob;
  case 2:
    winner = Carol;
}
Lottery with timed commitments

Round 1

Choose random $x$

Timed commitment to $H(x)$

Round 2

Publish $x$

Publish $y$

$w = H(x \oplus y \oplus z) \mod 3$

switch ($w$){
  case 0:
    winner = Alice;
  case 1:
    winner = Bob;
  case 2:
    winner = Carol;
}

I’m about to lose...

But I’ll lose my bond if I don’t publish 😞
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
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
Cryptographic beacons

Idea: service to regularly publish random data
● Uniform randomness
● No party can predict in advance
● All parties see the same values

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

Pros: quantum-mechanical randomness
Cons: must trust NIST
Natural phenomena

Sun spots

Weather

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 $\frac{1}{d}$ probability, you’d have a mining shortcut
Turning the block chain into a beacon

prev: \( H( ) \)
mrkl_root: \( H( ) \)
nonce: \( 0x7a83 \)
hash: \( 0x0000 \)

prev: \( H( ) \)
mrkl_root: \( H( ) \)
nonce: \( 0xf77e... \)
hash: \( 0x0000... \)

prev: \( H( ) \)
mrkl_root: \( H( ) \)
nonce: \( 0xf77e... \)
hash: \( 0x0000... \)
Turning the block chain into a beacon

prev: H()
mrkl_root: H()
nonce: 0x7a83
hash: 0x0000

prev: H()
mrkl_root: H()
nonce: 0xf77e...
hash: 0x0000...

prev: H()
mrkl_root: H()
nonce: 0xf77e...
hash: 0x0000...
Turning the block chain into a beacon

<table>
<thead>
<tr>
<th>prev:</th>
<th>H( )</th>
</tr>
</thead>
<tbody>
<tr>
<td>mrkl_root:</td>
<td>H( )</td>
</tr>
<tr>
<td>nonce:</td>
<td>0x7a83</td>
</tr>
<tr>
<td>hash:</td>
<td>0x0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>prev:</th>
<th>H( )</th>
</tr>
</thead>
<tbody>
<tr>
<td>mrkl_root:</td>
<td>H( )</td>
</tr>
<tr>
<td>nonce:</td>
<td>0xf77e...</td>
</tr>
<tr>
<td>hash:</td>
<td>0x0000...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>prev:</th>
<th>H( )</th>
</tr>
</thead>
<tbody>
<tr>
<td>mrkl_root:</td>
<td>H( )</td>
</tr>
<tr>
<td>nonce:</td>
<td>0xf77e...</td>
</tr>
<tr>
<td>hash:</td>
<td>0x0000...</td>
</tr>
</tbody>
</table>
Turning the block chain into a beacon

prev: H( )
mrkl_root: H( )
nonce: 0x7a83
hash: 0x0000

prev: H( )
mrkl_root: H( )
nonce: 0xf77e...
hash: 0x0000...

prev: H( )
mrkl_root: H( )
nonce: 0xf77e...
hash: 0x0000...
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