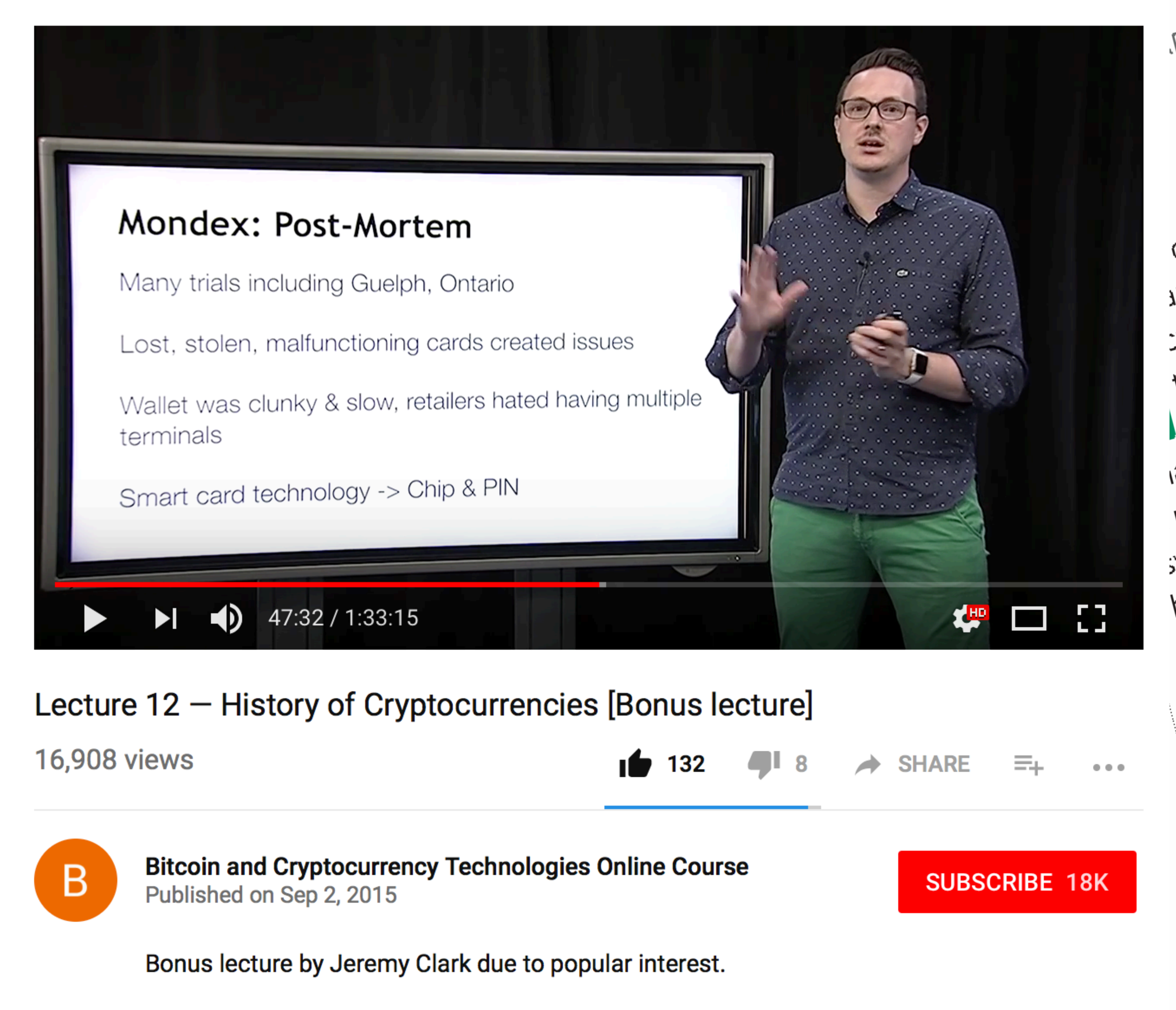
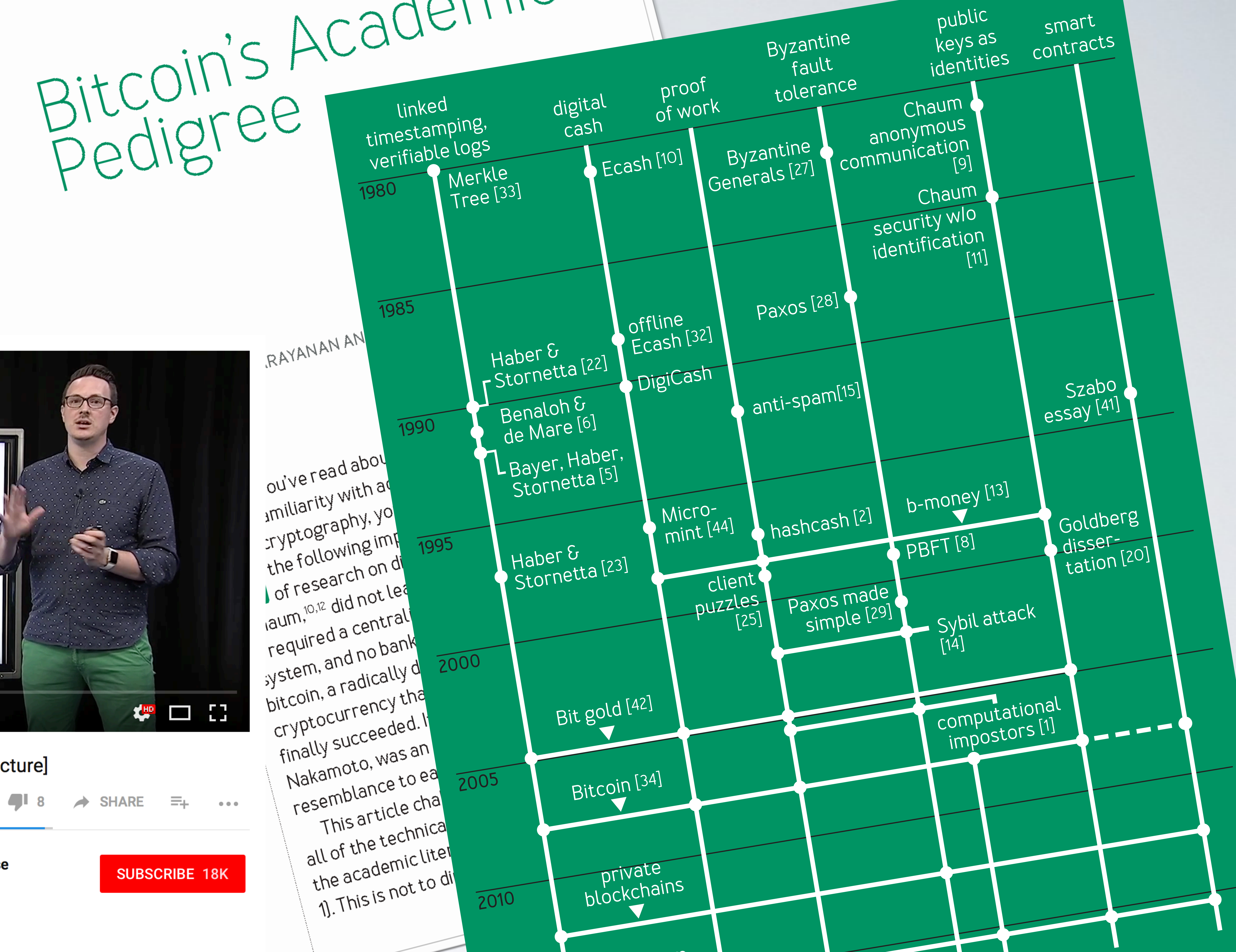
EthWord

Deploying PayWord on Ethereum

M. Elsheikh
Amr Youssef
Jeremy Clark


Mondex: Post-Mortem
Many trials including Guelph, Ontario
Lost, stolen, malfunctioning cards created issues
Wallet was clunky & slow, retailers hated having multiple terminals
Smart card technology -> Chip & PIN
47:32 / 1:33:15
Lecture 12 — History of Cryptocurrencies [Bonus lecture]
16,908 views
132
8
SHARE
Bitcoin and Cryptocurrency Technologies Online Course
Published on Sep 2, 2015
SUBSCRIBE 18K
Bonus lecture by Jeremy Clark due to popular interest.


cryptocurrency
Bitcoin's Academic Pedigree
linked timestamping, verifiable logs
digital cash
proof of work
Byzantine fault tolerance
public keys as identities
smart contracts
1980
1985
1990
1995
2000
2005
2010
Merkle Tree [33]
Ecash [10]
Byzantine Generals [27]
Chaum anonymous communication [9]
Chaum security w/o identification [11]
Paxos [28]
offline Ecash [32]
Haber & Stornetta [22]
DigiCash
Benaloh & de Mare [6]
anti-spam[15]
Szabo essay [41]
Bayer, Haber, Stornetta [5]
Micro-mint [44]
hashcash [2]
b-money [13]
Goldberg disser-tation [20]
Haber & Stornetta [23]
PBFT [8]
client puzzles [25]
Paxos made simple [29]
Sybil attack [14]
Bit gold [42]
computational impostors [1]
Bitcoin [34]
private blockchains

Pre-Bitcoin	Post-Bitcoin
Auditable, anonymous electronic cash [Sander & Ta-Shma]	Zerocoin, etc [Miers et al]
Lottery Payments [Rivest] [Wheeler] [Jarecki & Odlyzko]	Micropayments for decentralized currencies [Pass, shelat]
An efficient distributed currency [Laurie]	RSCoin [Danezis & Meiklejohn]

Pre-Bitcoin	Post-Bitcoin
Auditable, anonymous electronic cash [Sander & Ta-Shma]	Zerocoin, etc [Miers et al]
Lottery Payments [Rivest] [Wheeler] [Jarecki & Odlyzko]	Micropayments for decentralized currencies [Pass, shelat]
An efficient distributed currency [Laurie]	RSCoin [Danezis & Meiklejohn]
PayWord [Rivest & Shamir]	

Pre-Bitcoin	Post-Bitcoin
Auditable, anonymous electronic cash [Sander & Ta-Shma]	Zerocoin, etc [Miers et al]
Lottery Payments [Rivest] [Wheeler] [Jarecki & Odlyzko]	Micropayments for decentralized currencies [Pass, shelat]
An efficient distributed currency [Laurie]	RSCoin [Danezis & Meiklejohn]
PayWord [Rivest & Shamir]	EthWord [You are here]

PayWord and MicroMint: Two Simple Micropayment Schemes

Ronald L. Rivest[1] and Adi Shamir[2]

[1] MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, Mass. 02139, rivest@theory.lcs.mit.edu

[2] Weizmann Institute of Science, Applied Mathematics Department, Rehovot, Israel, shamir@theory.lcs.mit.edu

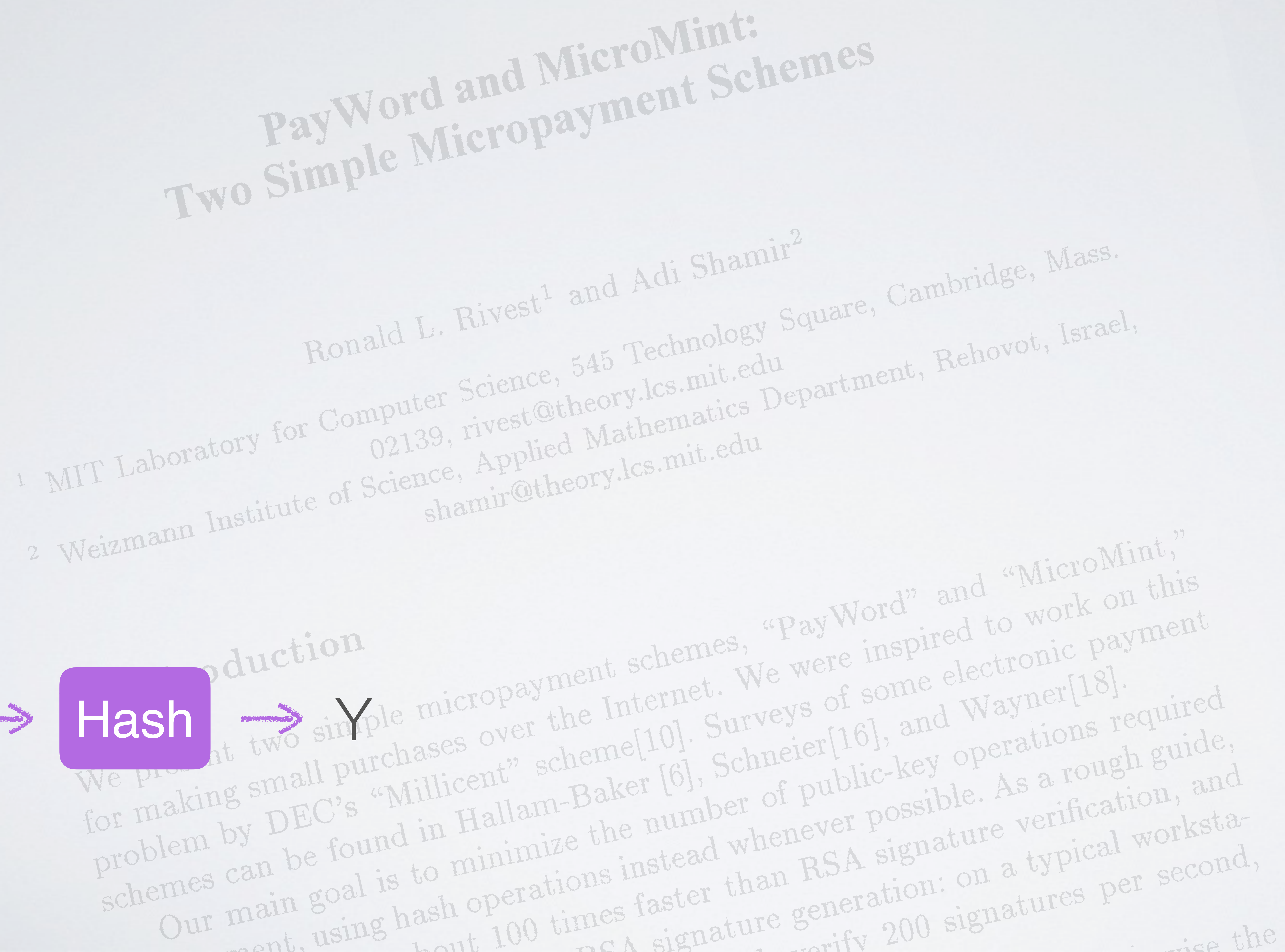
1 Introduction

We present two simple micropayment schemes, "PayWord" and "MicroMint," for making small purchases over the Internet. We were inspired to work on this problem by DEC's "Millicent" scheme[10]. Surveys of some electronic payment schemes can be found in Hallam-Baker [6], Schneier[16], and Wayner[18].

Our main goal is to minimize the number of public-key operations required ...ment, using hash operations instead whenever possible. As a rough guide, ...bout 100 times faster than RSA signature verification, and ... RSA signature generation: on a typical worksta- ... verify 200 signatures per second,

Hash

X → Hash → Y


PayWord and MicroMint:
Two Simple Micropayment Schemes

Ronald L. Rivest[1] and Adi Shamir[2]

[1] MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, Mass. 02139, rivest@theory.lcs.mit.edu

[2] Weizmann Institute of Science, Applied Mathematics Department, Rehovot, Israel, shamir@theory.lcs.mit.edu

oduction

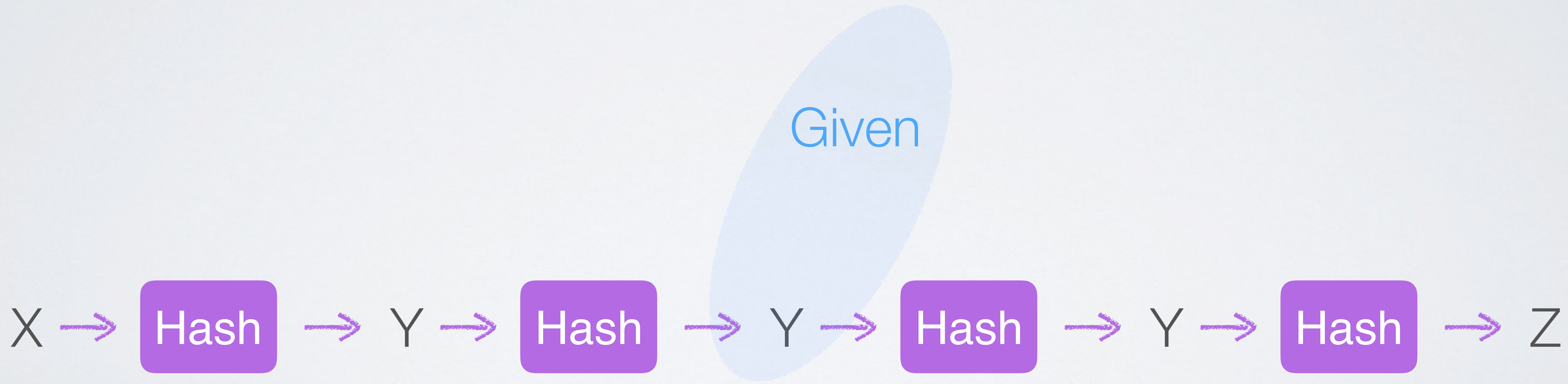
We present two simple micropayment schemes, "PayWord" and "MicroMint," for making small purchases over the Internet. We were inspired to work on this problem by DEC's "Millicent" scheme[10]. Surveys of some electronic payment schemes can be found in Hallam-Baker [6], Schneier[16], and Wayner[18].

Our main goal is to minimize the number of public-key operations required
ment, using hash operations instead whenever possible. As a rough guide,
out 100 times faster than RSA signature verification, and
RSA signature generation: on a typical worksta-
verify 200 signatures per second

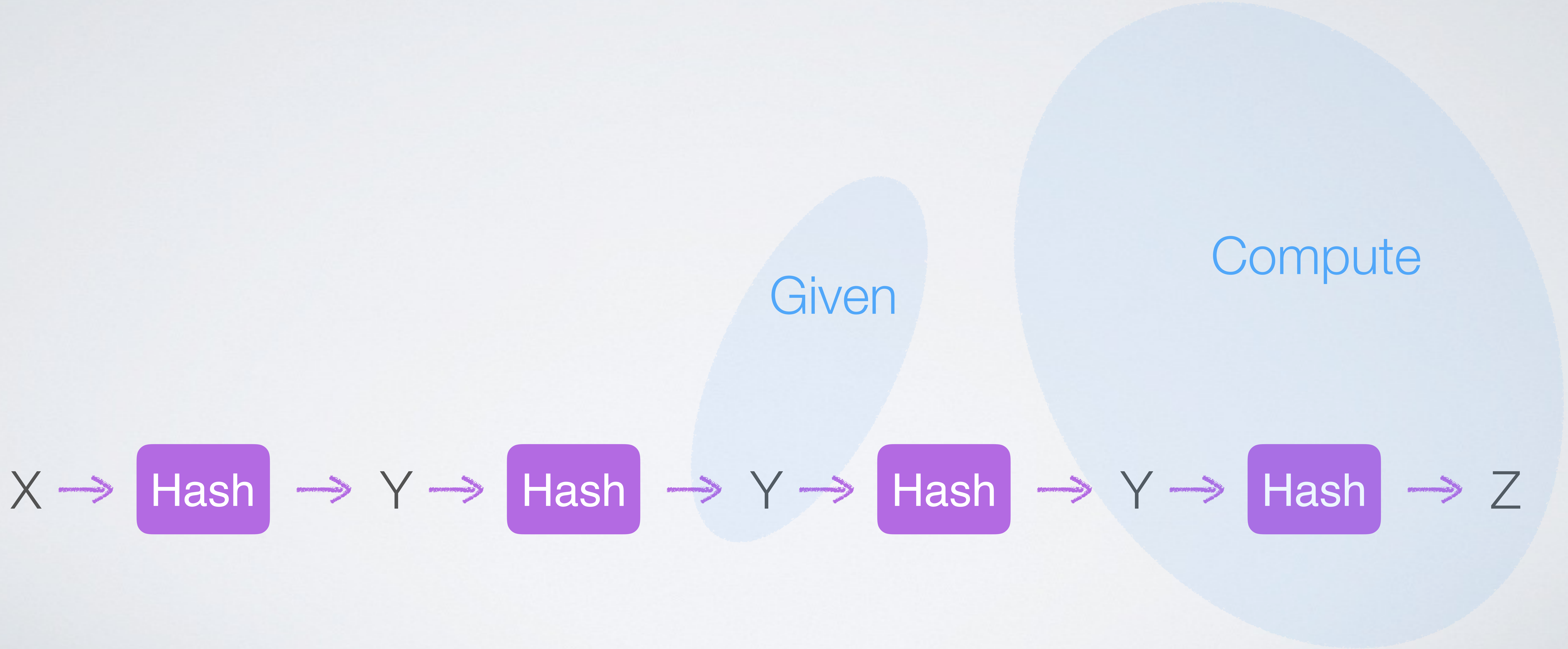
Iterative Hash

Hash Chain

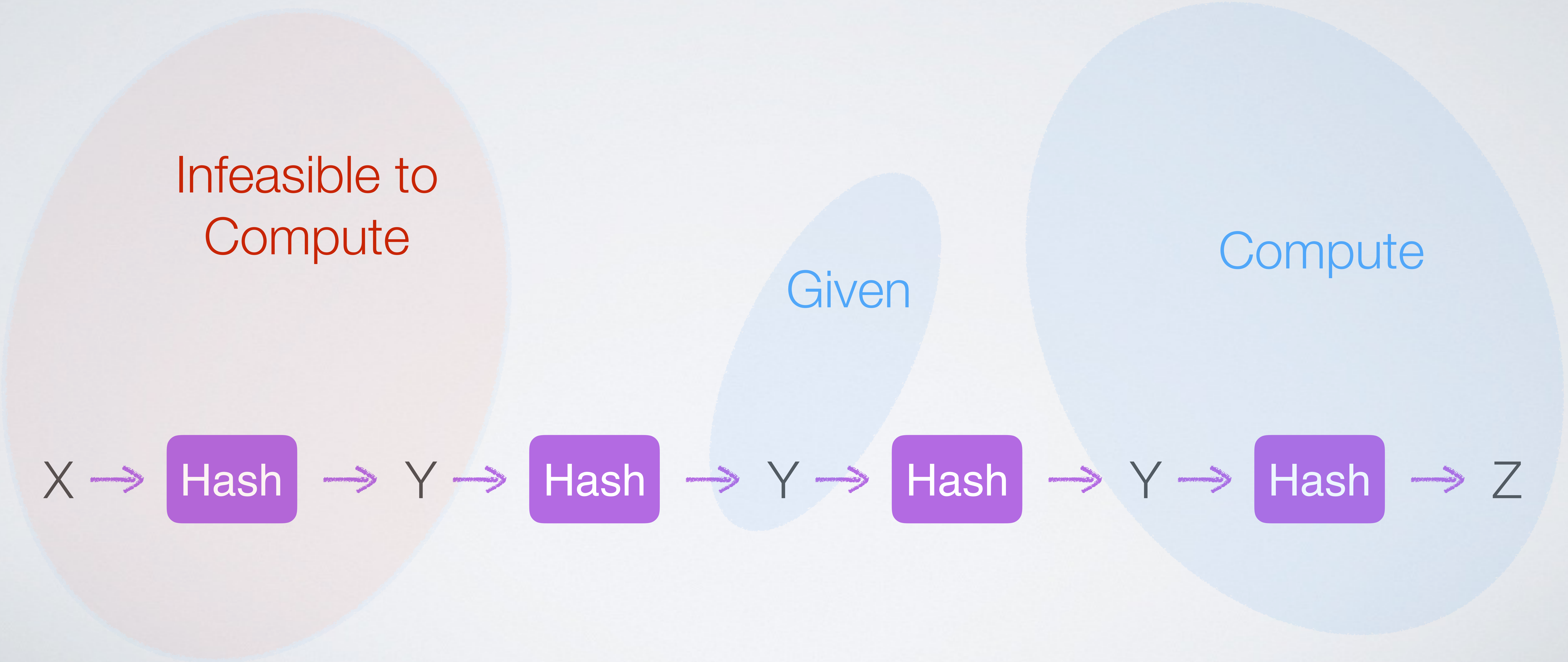
Hash Chain



Hash Chain

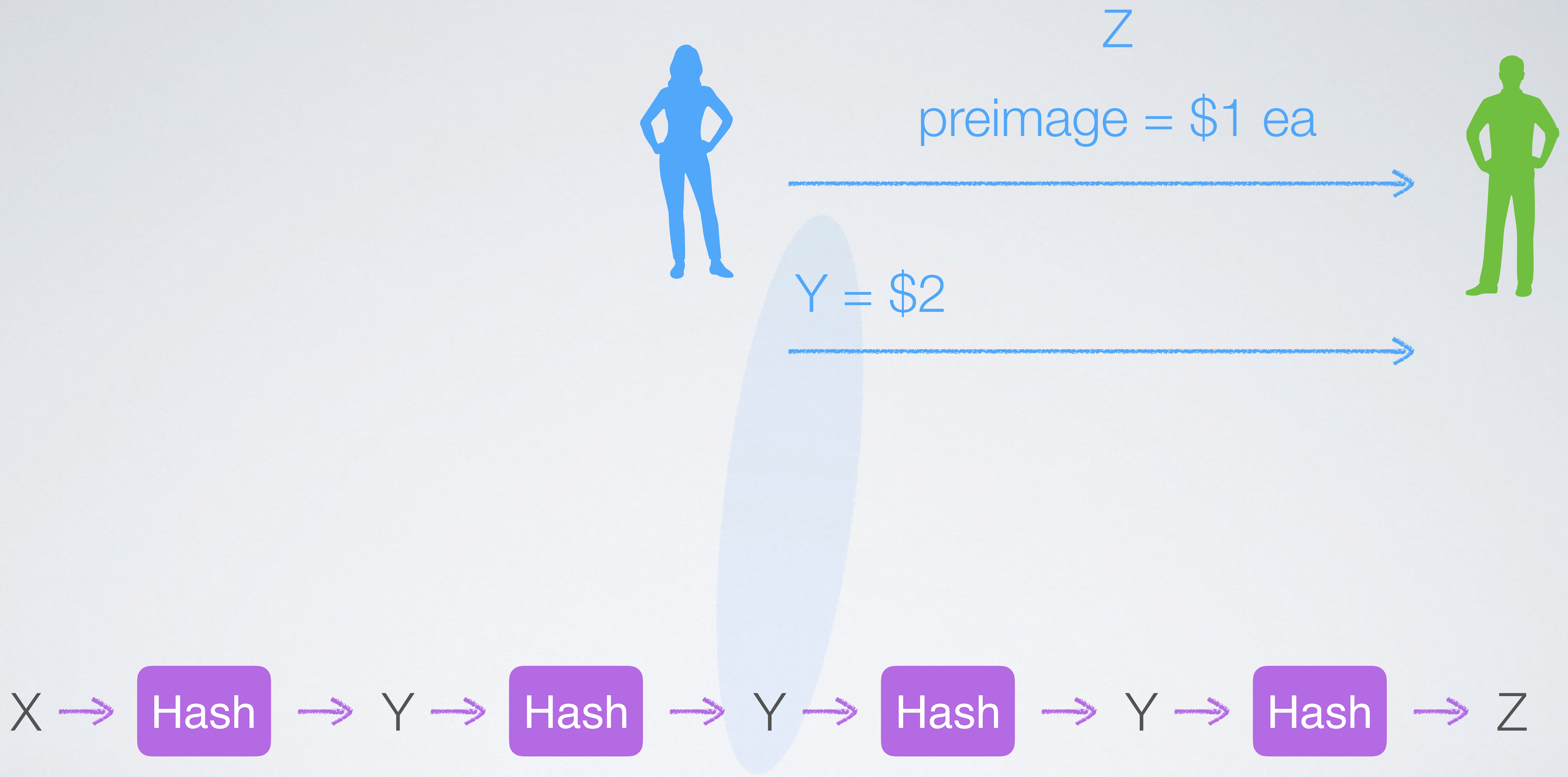


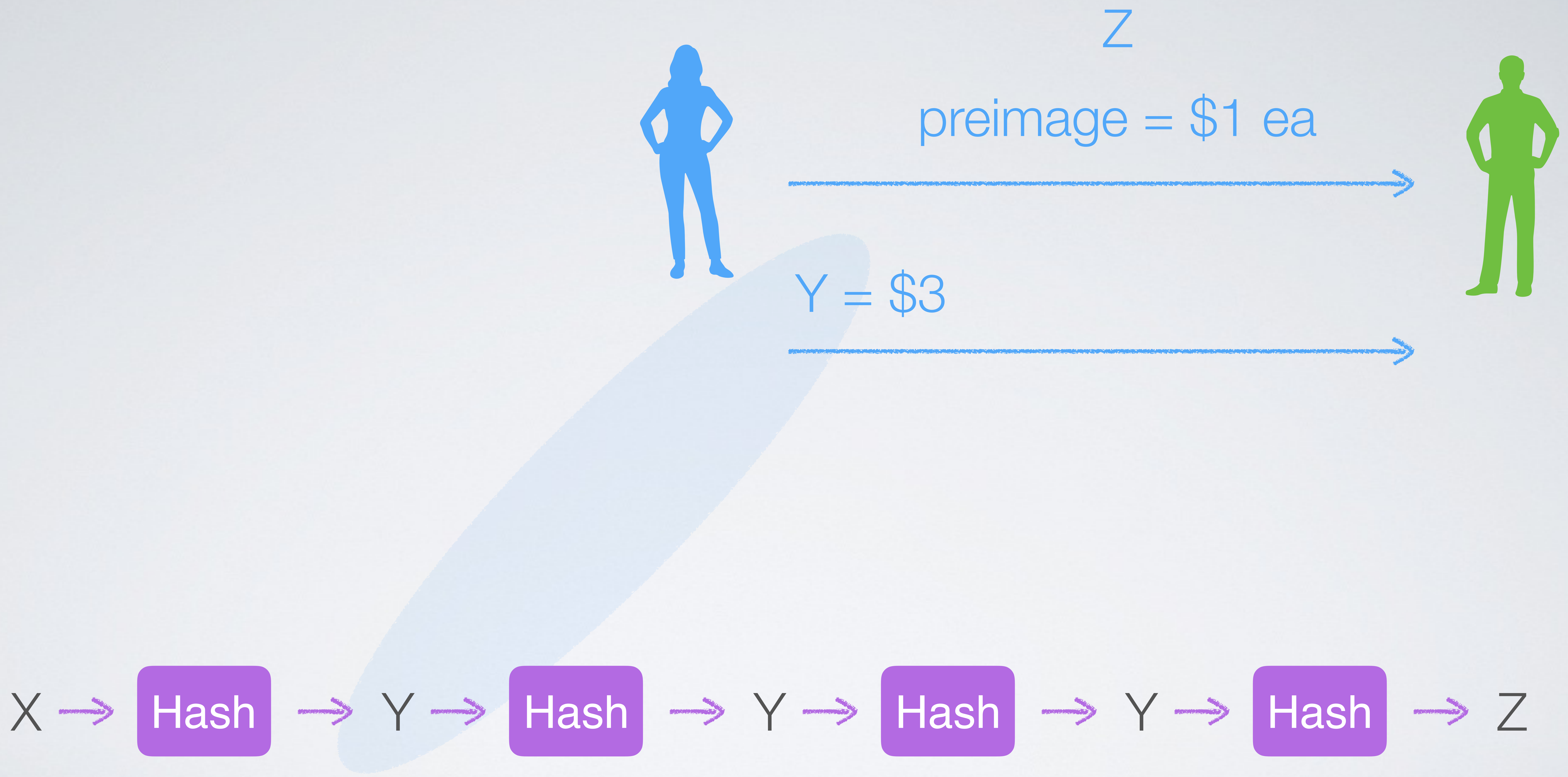
Hash Chain

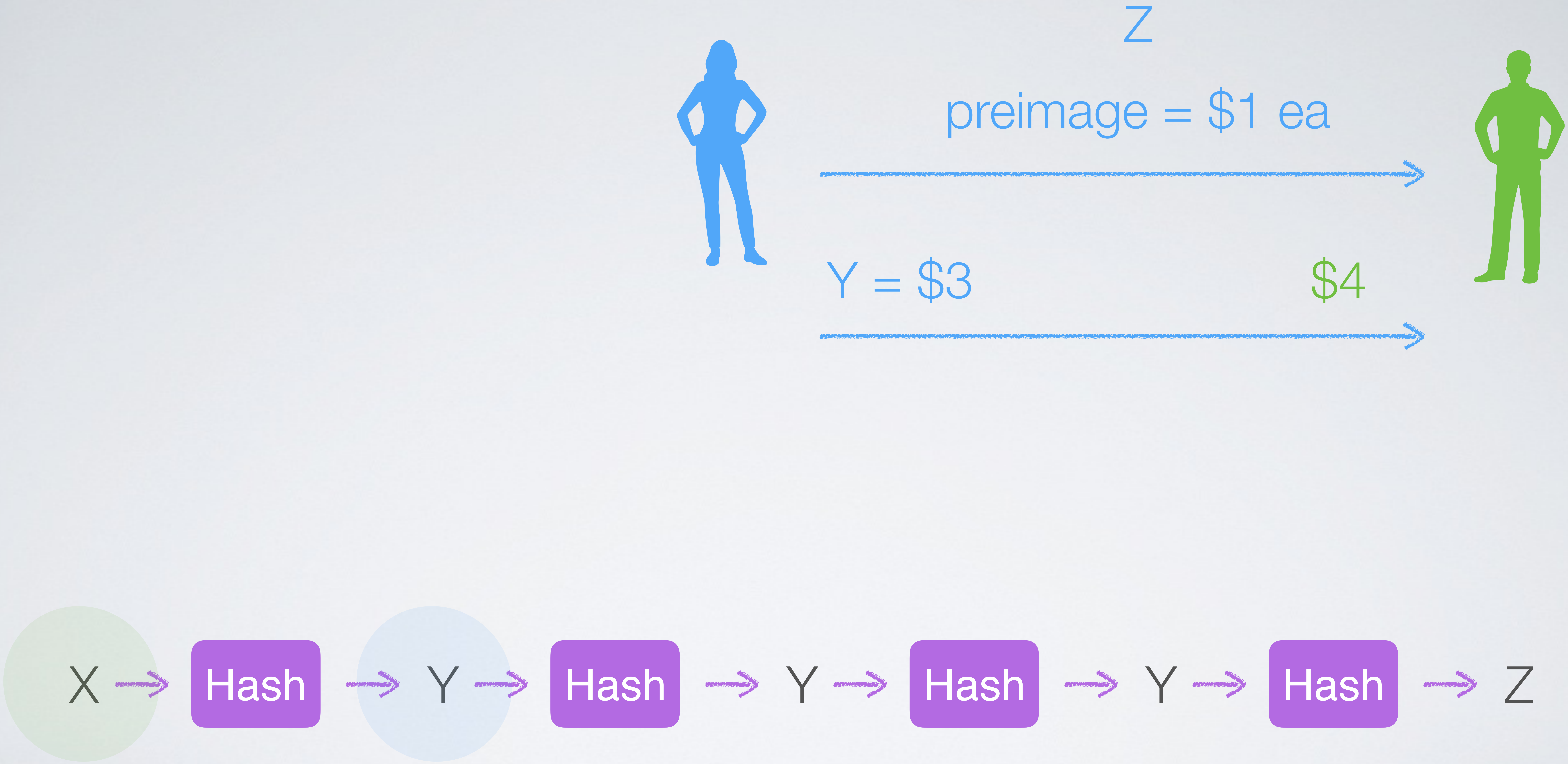


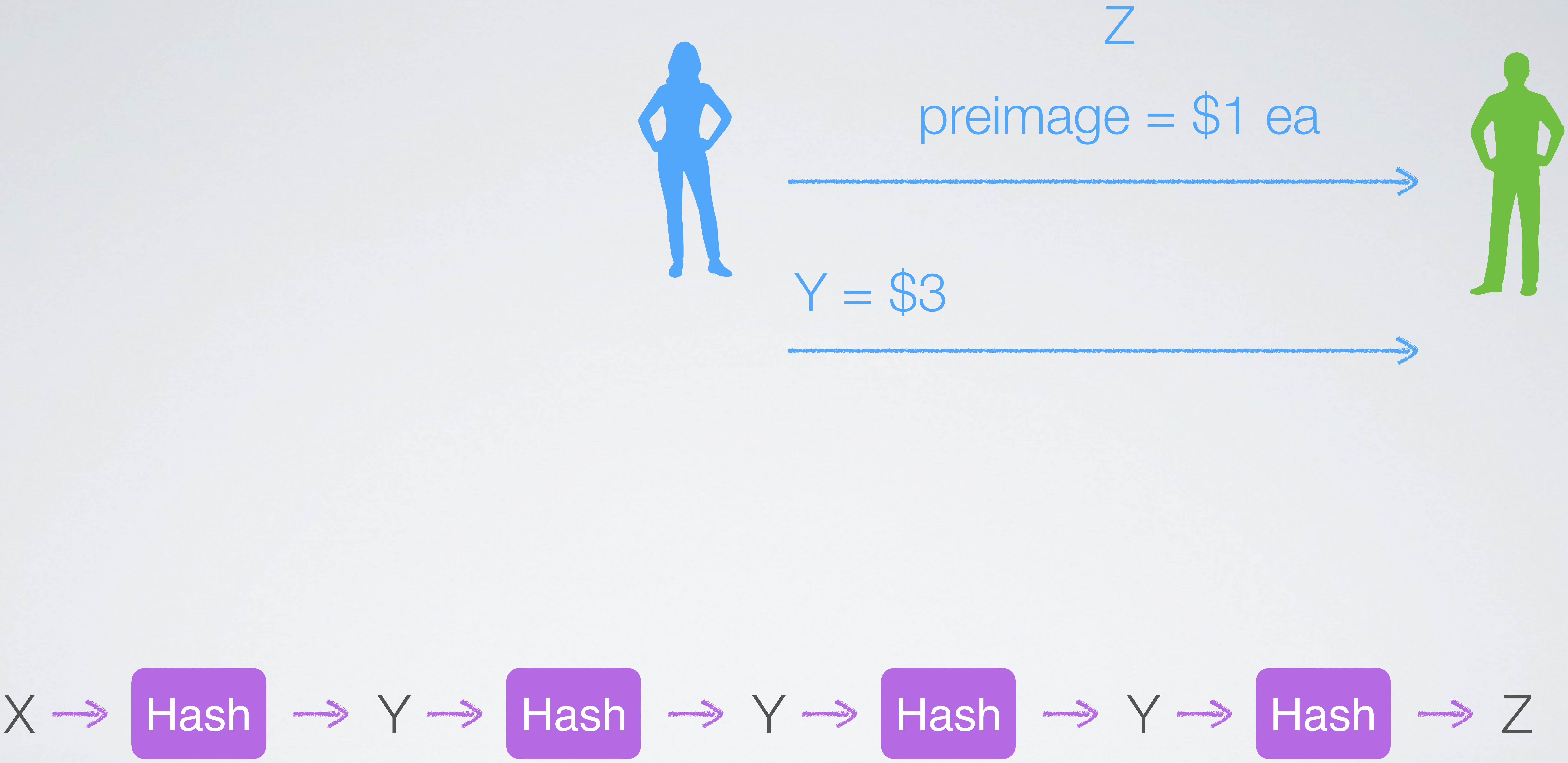
Infeasible to Compute
Given
Compute
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z

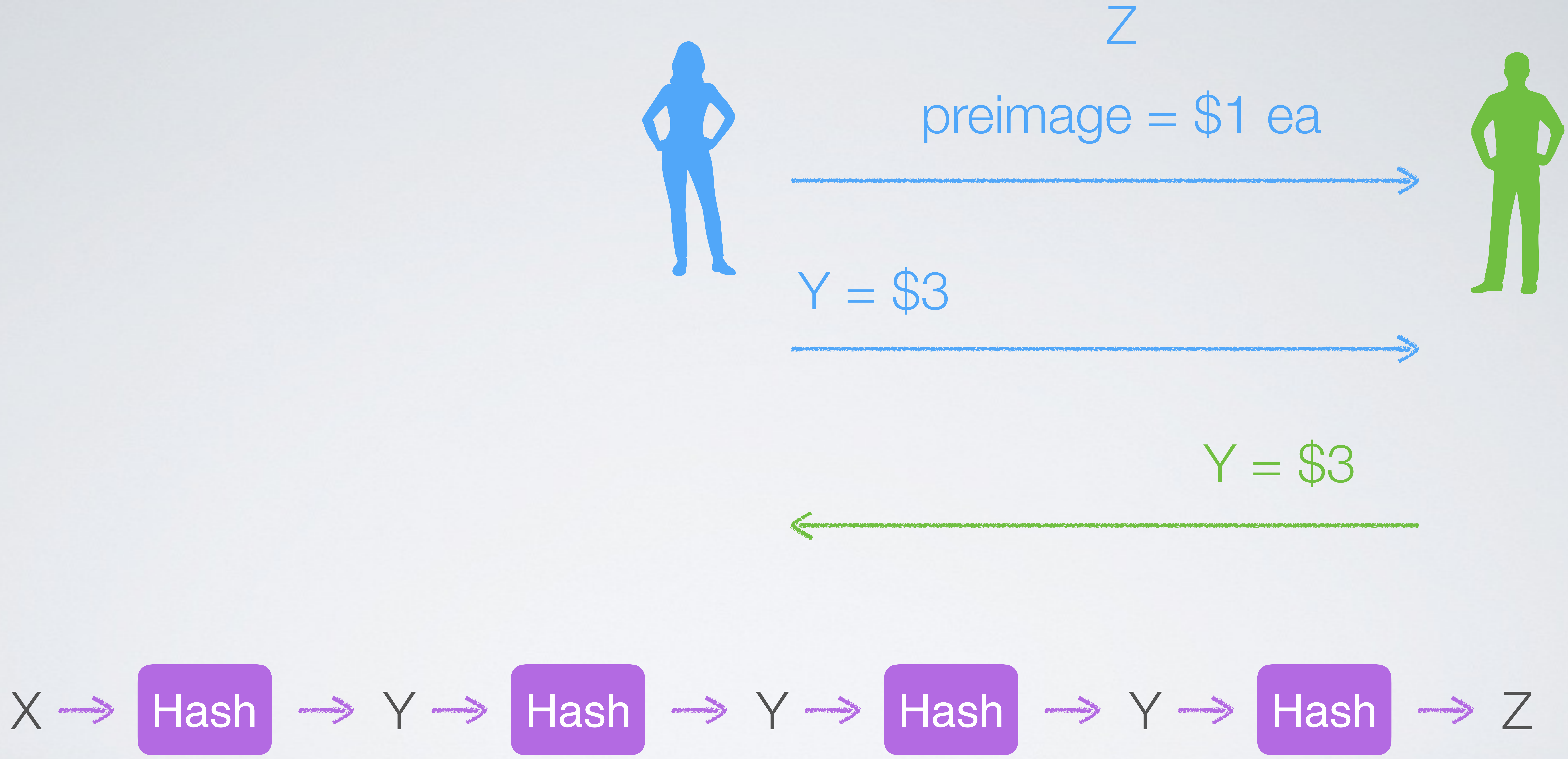
PayWord

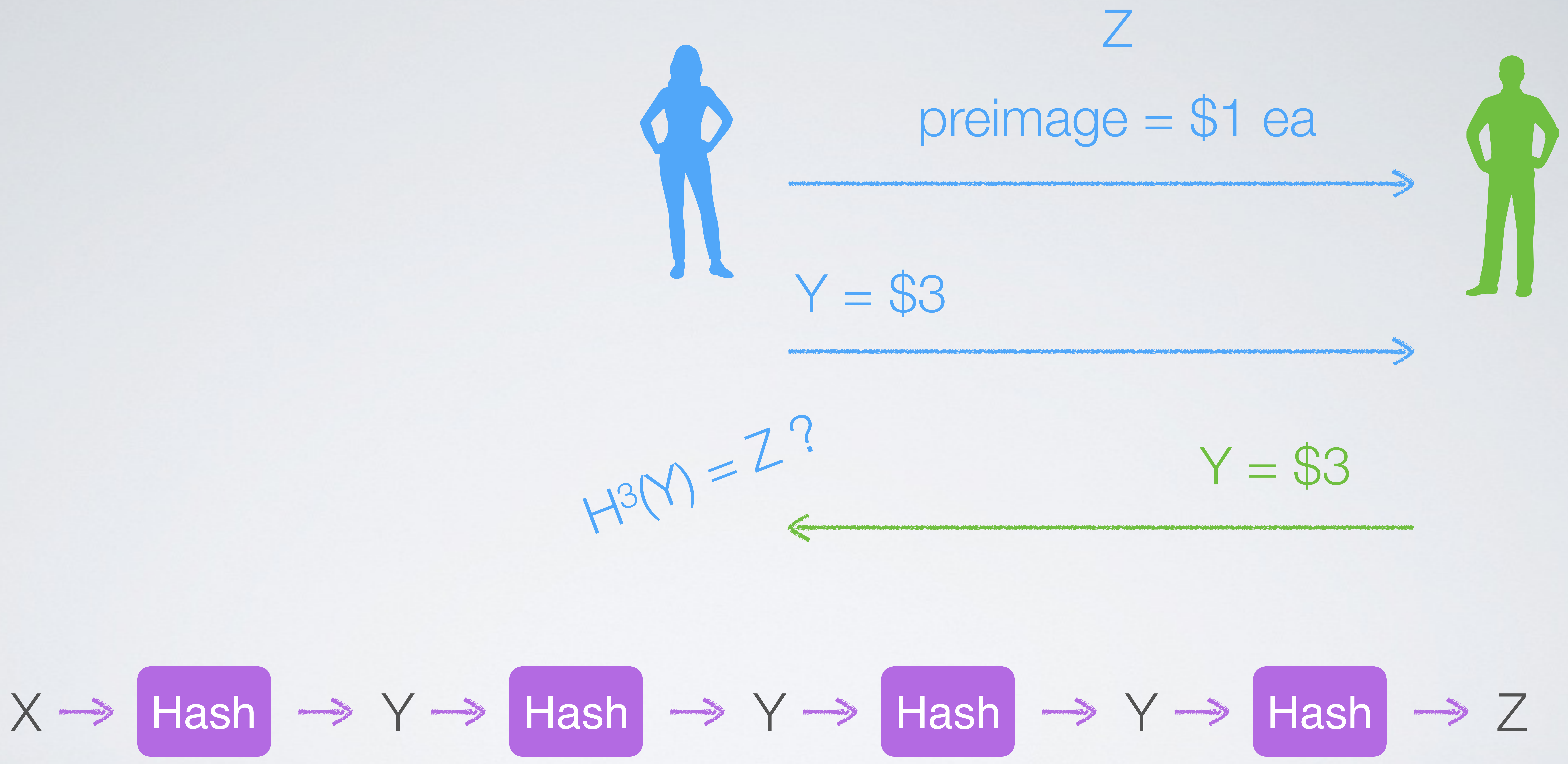

Z
preimage = $1 ea
Y = $2
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z


Z
preimage = $1 ea
Y = $3
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z

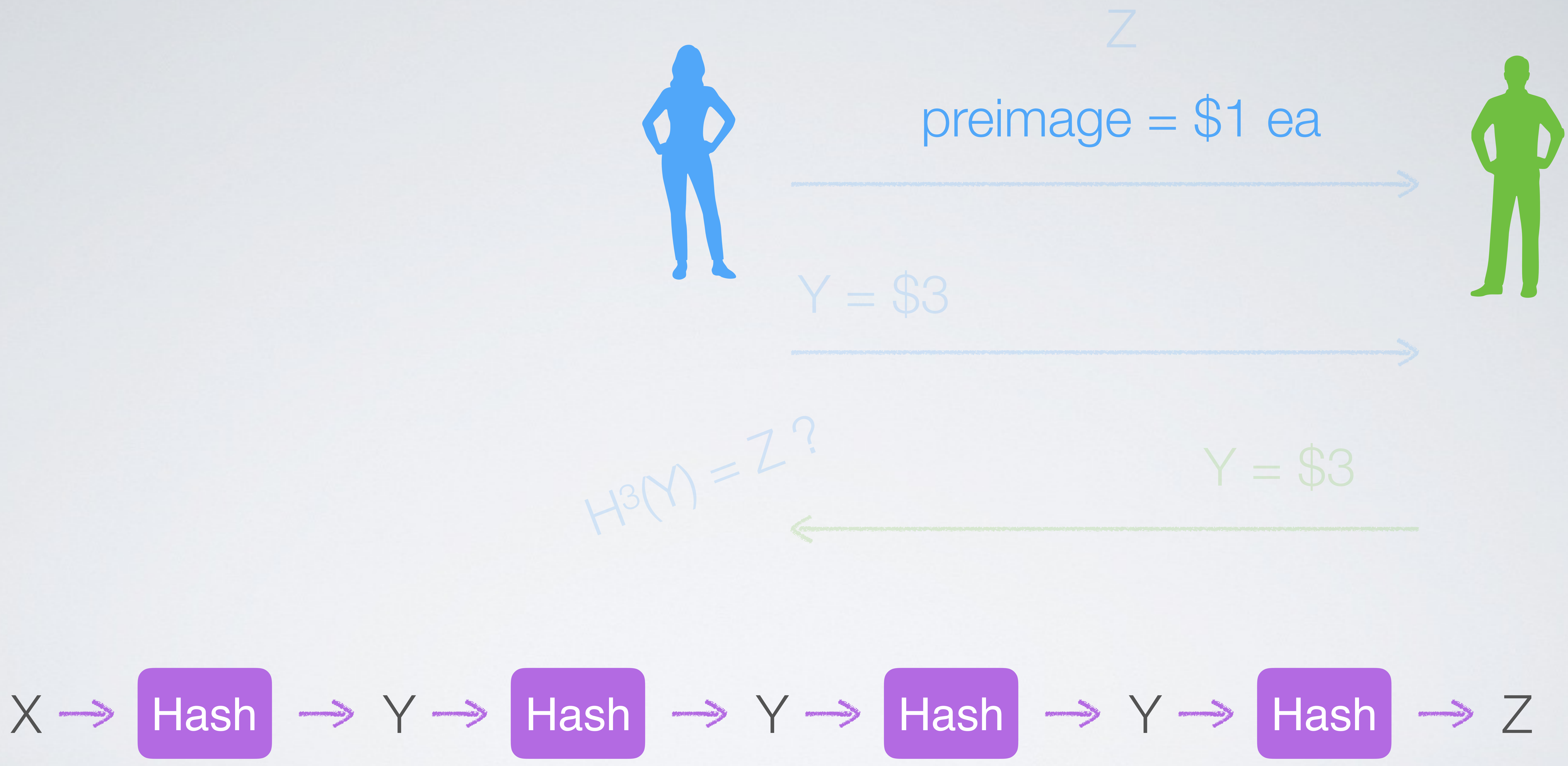

Z
preimage = $1 ea
Y = $3
$4
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z


Z
preimage = $1 ea
Y = $3
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z

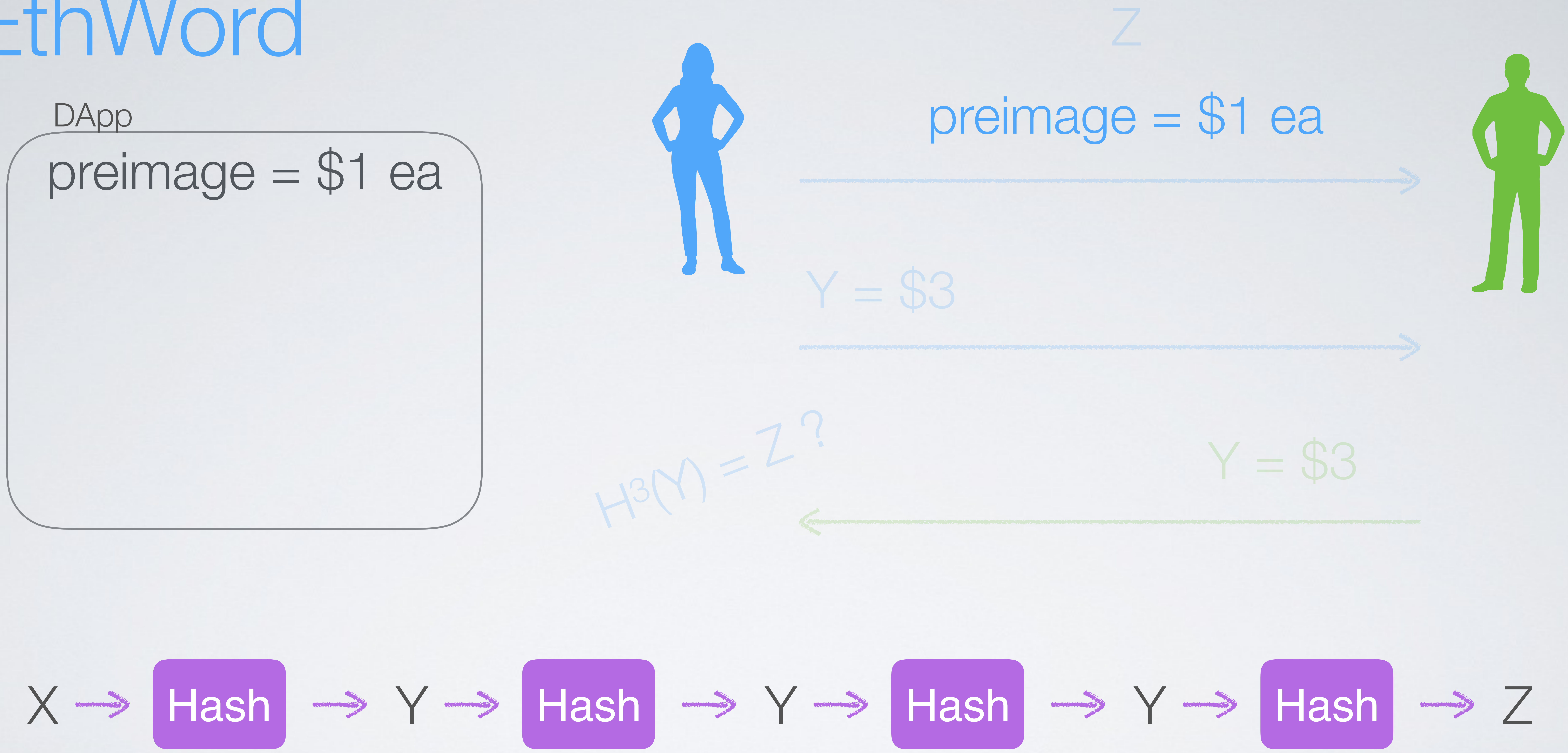

Z
preimage = $1 ea
Y = $3
Y = $3
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z

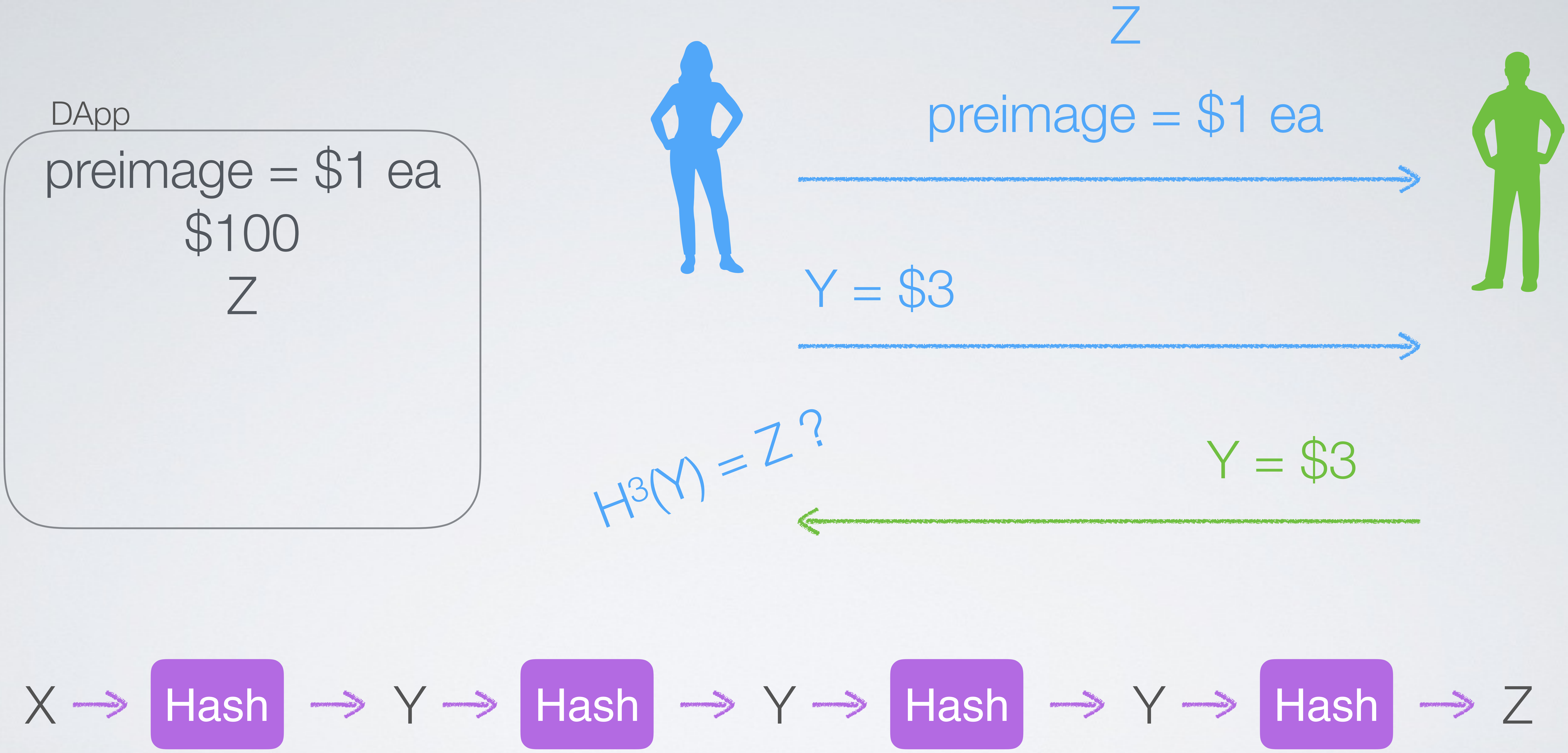


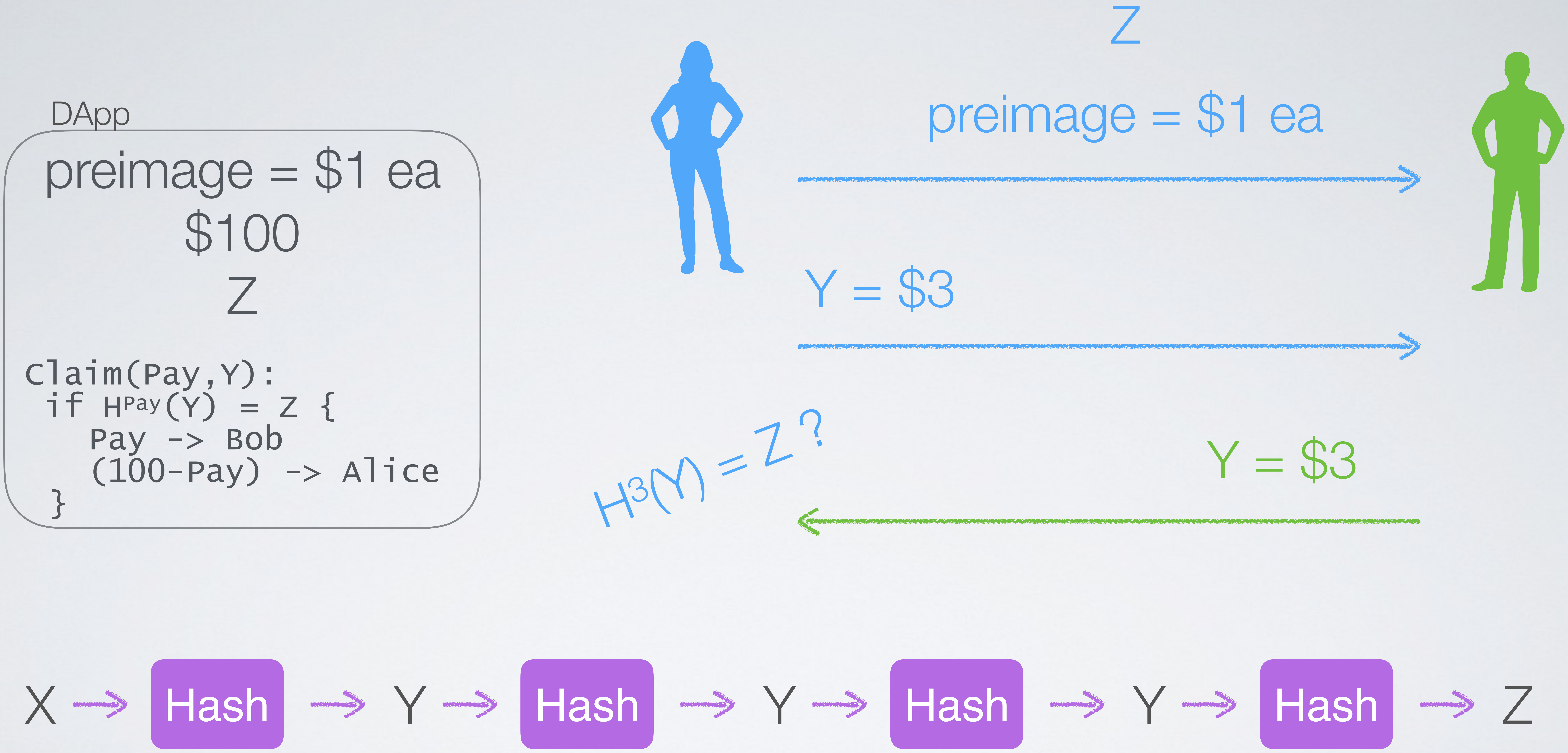
Z
preimage = $1 ea
Y = $3
Y = $3
H3(Y) = Z ?
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z


Z
preimage = $1 ea
Y = $3
H3(Y) = Z ?
Y = $3
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z

EthWord


DApp
preimage = $1 ea
Z
preimage = $1 ea
Y = $3
H3(Y) = Z ?
Y = $3
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z


DApp
preimage = $1 ea
$100
Z
Z
preimage = $1 ea
Y = $3
H3(Y) = Z ?
Y = $3
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z


DApp
preimage = $1 ea
$100
Z
Claim(Pay,Y):
if H^Pay(Y) = Z {
Pay -> Bob
(100-Pay) -> Alice
}
Z
preimage = $1 ea
Y = $3
Y = $3
H3(Y) = Z ?
X
Hash
Y
Hash
Y
Hash
Y
Hash
Z

DApp

preimage = $1 ea

$100

Z

```
Claim(Pay,Y):
 if H^Pay(Y) = Z {
    Pay -> Bob
    (100-Pay) -> Alice
 }
```

Payment Channel?

DApp

preimage = $1 ea

$100

Z

```
Claim(Pay,Y):
 if H^Pay(Y) = Z {
    Pay -> Bob
    (100-Pay) -> Alice
 }
```

Payment Channel?

Offline / Monotonic / Unidirectional

DApp

preimage = $1 ea

$100

Z

```
Claim(Pay,Y):
 if H^Pay(Y) = Z {
    Pay -> Bob
    (100-Pay) -> Alice
 }
```

Payment Channel?

~~Bitcoin~~

Ethereum

DApp

preimage = $1 ea

$100

Z

```
Claim(Pay,Y):
 if H^Pay(Y) = Z {
    Pay -> Bob
    (100-Pay) -> Alice
 }
```

Payment Channel?

~~Digital Signatures~~

Hash

DApp

preimage = $1 ea

$100

Z

```
Claim(Pay,Y):
 if H^Pay(Y) = Z {
    Pay -> Bob
    (100-Pay) -> Alice
 }
```

Payment Channel?

~~Digital Signatures~~

Hash & msg.sender

DApp

preimage = $1 ea

$100

Z

```
Claim(Pay,Y):
 if H^Pay(Y) = Z {
    Pay -> Bob
    (100-Pay) -> Alice
 }
```

Payment Channel?

Very Compact

112 -> 256 bits

Ethereum Payment Channel in 50 Lines of Code

Matthew Di Ferrante Follow

Jun 5, 2017 · 4 min read

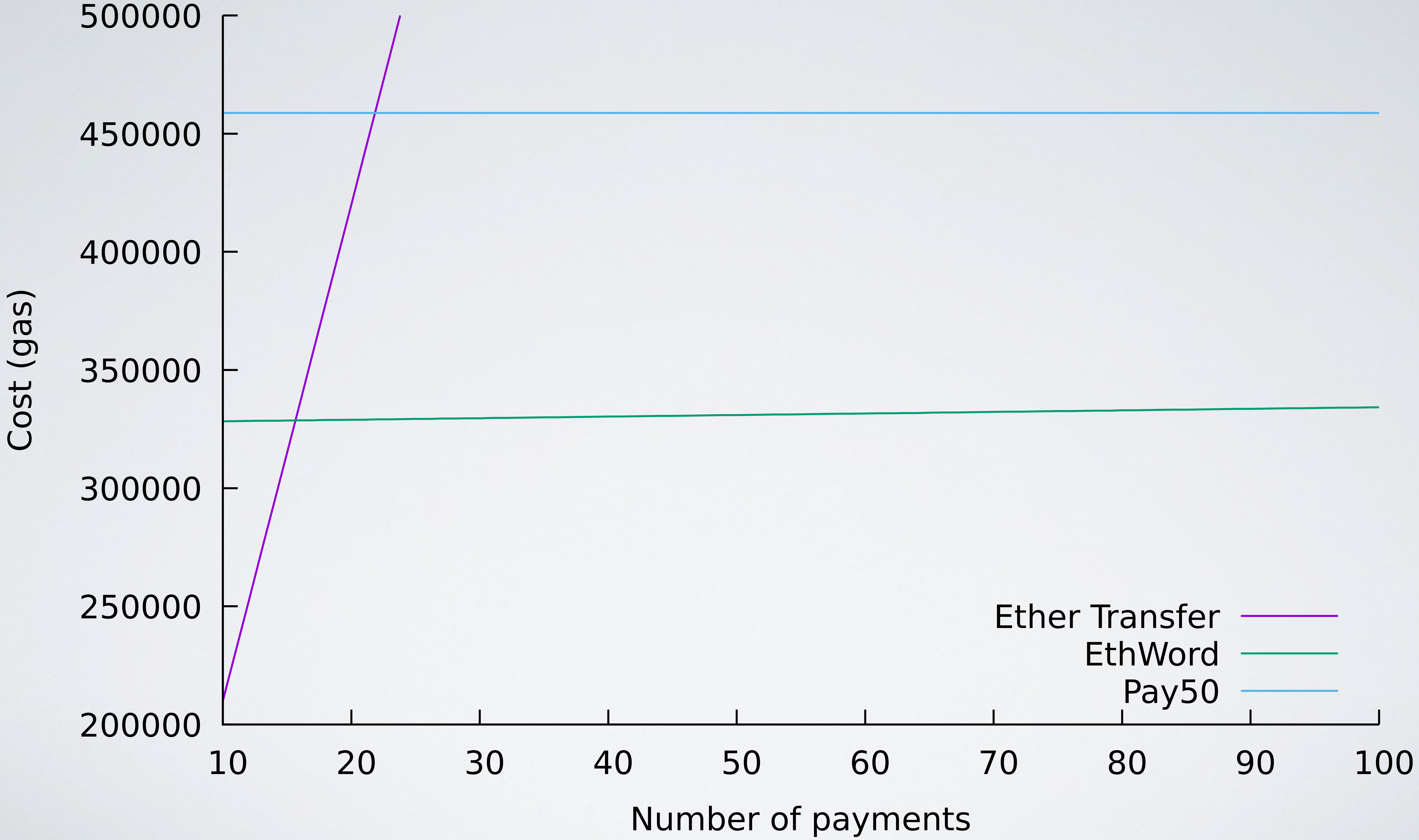
With the talk of state/payment channels being a "future" scalability option in Ethereum, I wanted to write a contract to show that they're more than doable now. You don't need to wait for Raiden, you can set up your own trustless channels right now.

I'll walk through the solidity code in channel.sol here: https://github.com/mattdf/payment-channel

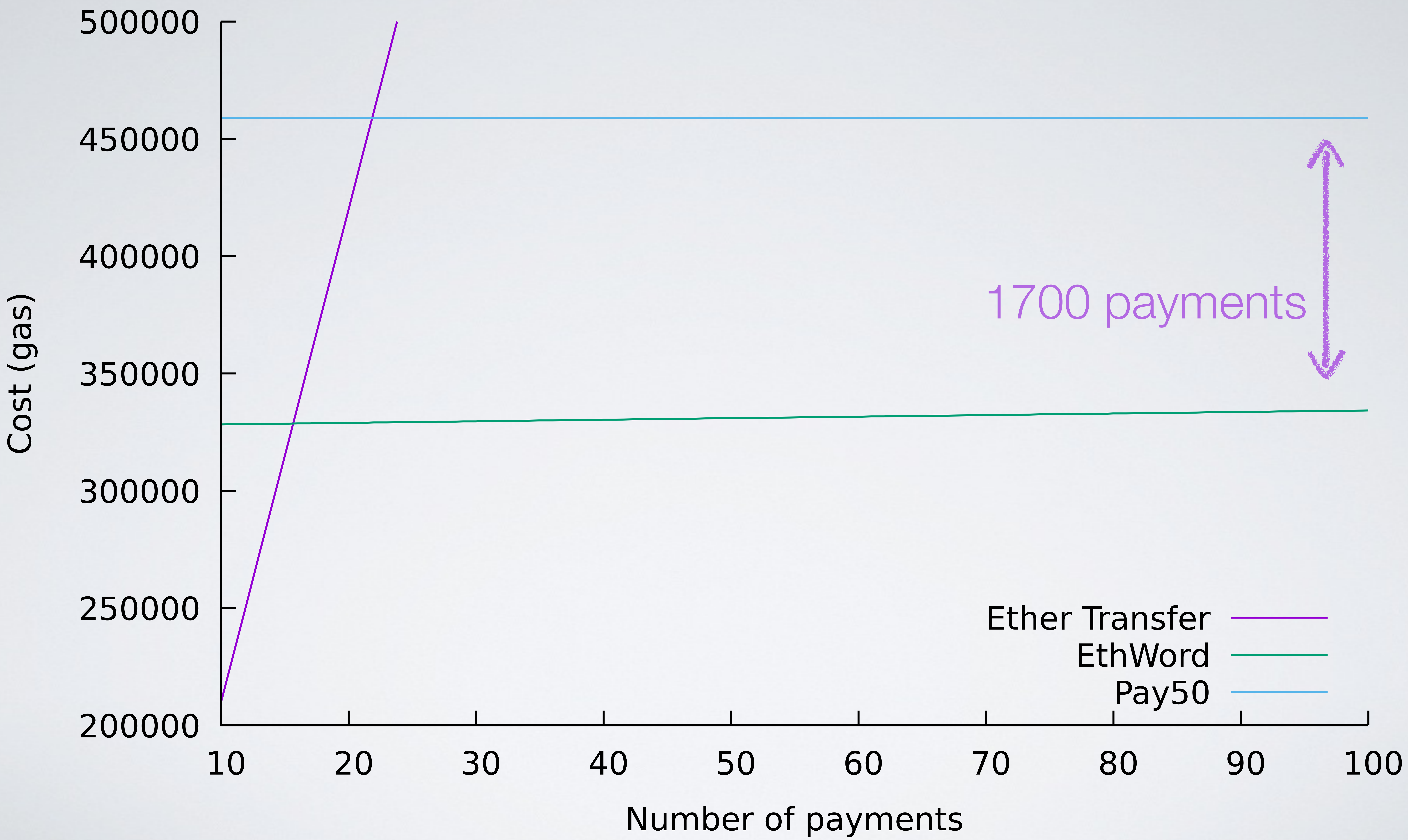
Let's say Alice and Bob want to set up a payment channel for something that requires micropayments that they don't want to commit on chain to save on transaction fees. In this case, Bob may be paying Alice to manage a social media presence, and he pays her 0.001 ETH per tweet(24 cents) —if Bob were to make an on-chain transaction for each tweet, 20% of Alice's income would be eaten up by fees.

On one hand, Alice does not want to do 100 tweets of work and trust Bob will pay her at the end for all 100 tweets, and on the other hand, Bob doesn't want to pay Alice for 100 tweets all at once for her to just disappear and not do any work.

…t channel where Bob commits



Cost (gas)
500000
450000
400000
350000
300000
250000
200000
10
20
30
40
50
60
70
80
90
100
Number of payments
Ether Transfer
EthWord
Pay50

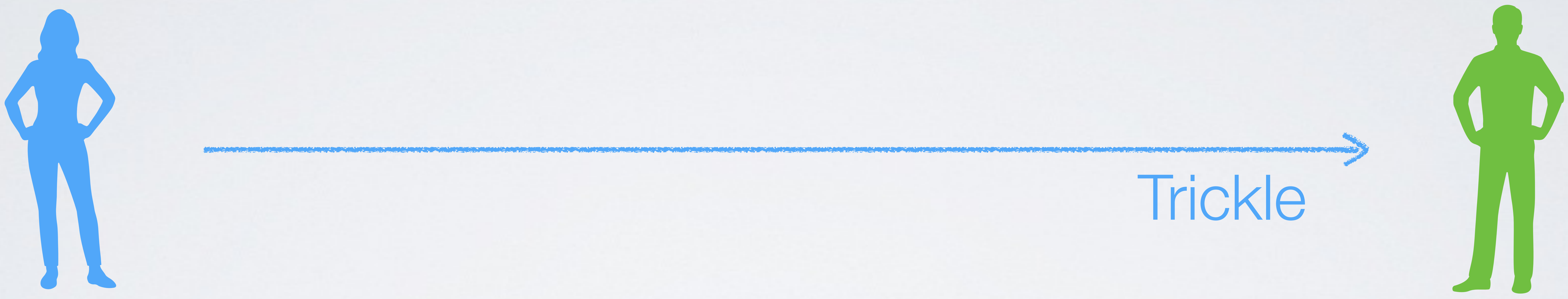

500000
450000
400000
350000
300000
250000
200000
Cost (gas)
10
20
30
40
50
60
70
80
90
100
Number of payments
1700 payments
Ether Transfer
EthWord
Pay50

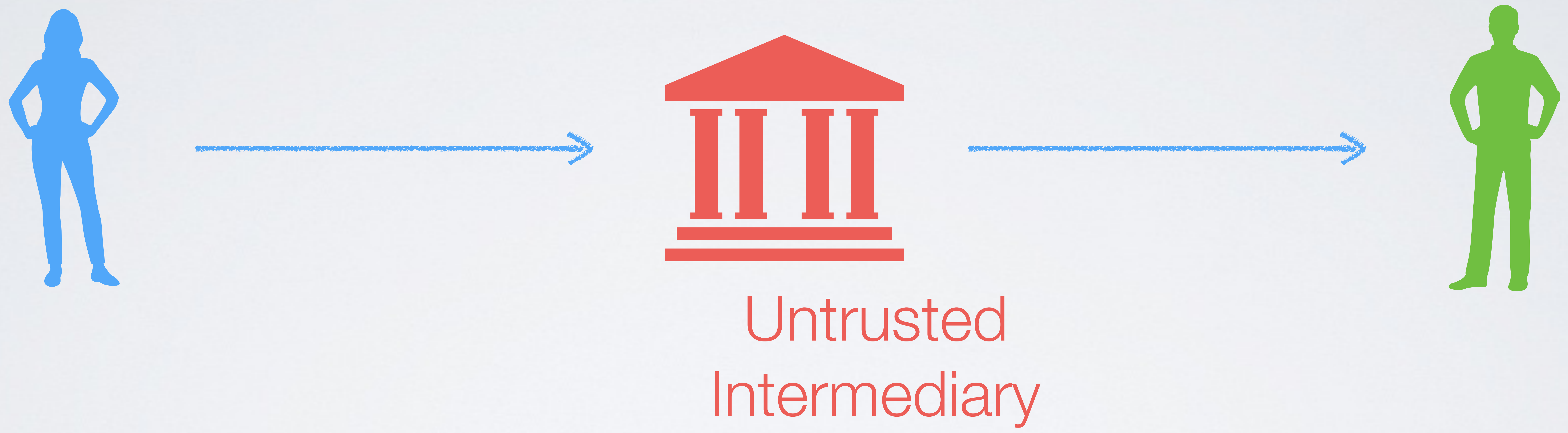
EthWord Function	Gas	ETH	USD
Channel	312 031	0.00539	$0.689
closeChannel (50)	18 905	0.00033	$0.042
closeChannel (100)	22 205	0.00038	$0.049

EthWord Function	Gas	ETH	USD
Channel	312 031	0.00539	$0.689
closeChannel (50)	18 905	0.00033	$0.042
closeChannel (100)	22 205	0.00038	$0.049

5% fee -> settle for amounts ~$15

1% fee -> settle for amounts ~$75


Trickle


Untrusted
Intermediary

Other payment channel results?
Open research


Q
@PulpSpy