HITACHI
Inspire the Next

ETHTerakoya

ブロックチェーン向けゼロ知識証明の研究

株式会社日立製作所
長沼健

目次

HITACHI
Inspire the Next

- ゼロ知識証明とは？
- ブロックチェーンでどうやってゼロ知識証明を使うのか？
- 応用：セキュアスマートコントラクト
- Hyperledger Fabricでの実装アーキテクチャ
- 評価実験

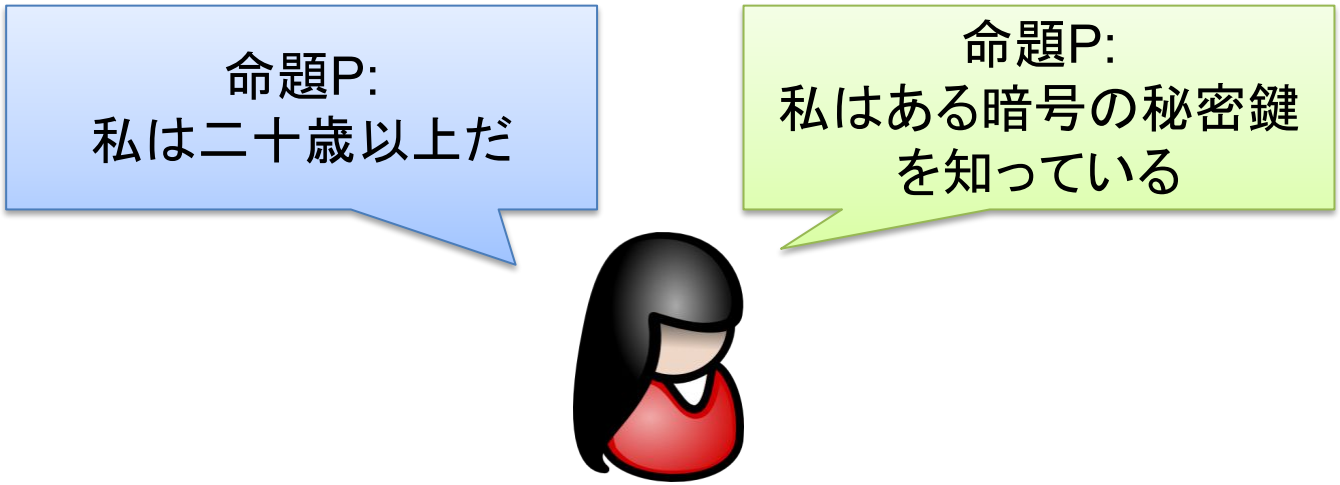
目次

HITACHI
Inspire the Next

- ゼロ知識証明とは？
- ブロックチェーンでどうやってゼロ知識証明を使うのか？
- 応用：セキュアスマートコントラクト
- Hyperledger Fabricでの実装アーキテクチャ
- 評価実験

ゼロ知識証明とは？

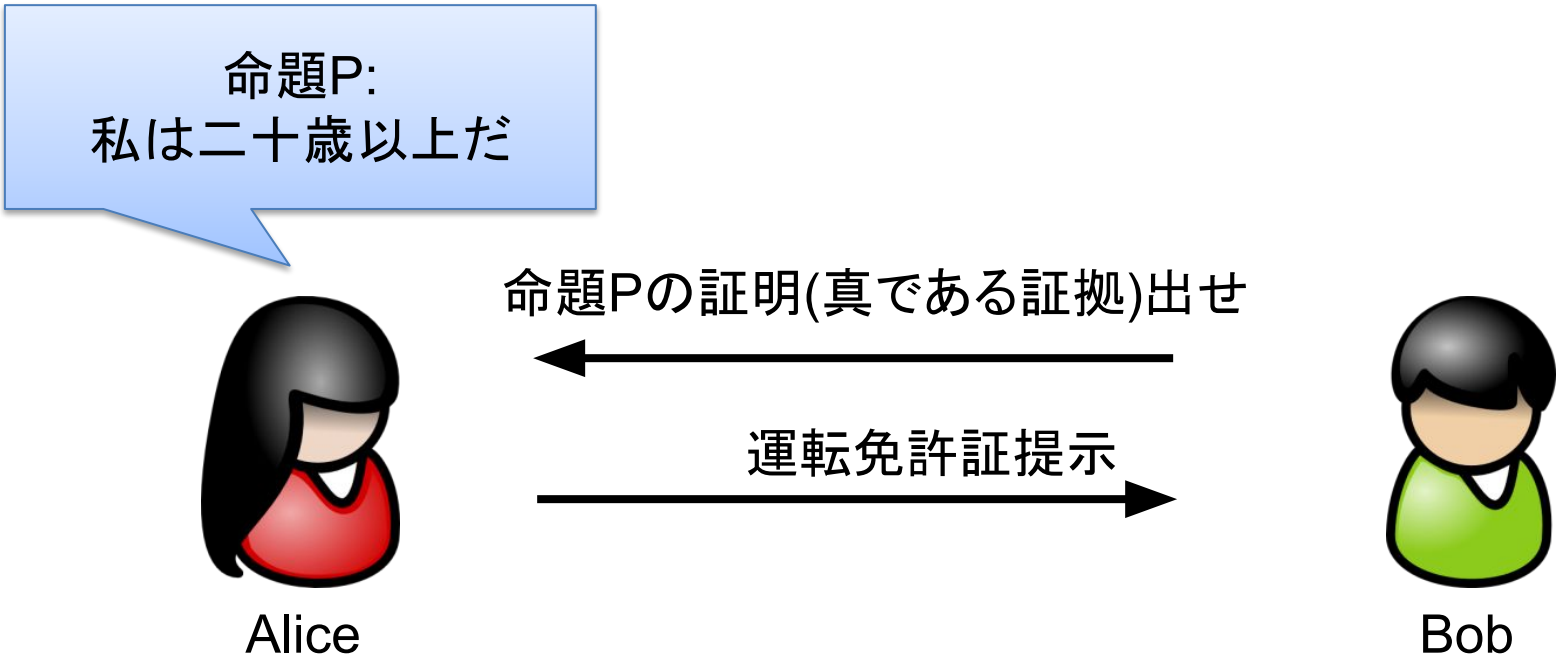
ゼロ知識証明（zero-knowledge proof）とは、証明者(Prover)が検証者(Verifier)に、自分の持っている命題が真であることを伝えるのに、真であること以外、何の知識も伝えることなく証明できるような対話(非対話)知識証明プロトコルである。 by Wikipedia



Alice (Prover)は命題Pが真(True)である事を誰かに証明したい

ゼロ知識証明とは？

Alice (Prover)は命題Pが真(True)である事をBob(Verifier)にゼロ知識証明したい



運転免許証提示でAlice (Prover)は命題Pが真(True)である事をBobに証明できるがゼロ知識ではない。生年月日から年齢などが漏れている。
どんなものがゼロ知識証明と言えるのか？

ゼロ知識証明とは？

HITACHI
Inspire the Next

Alice (Prover)は命題Pが真(True)である事をBob(Verifier)にゼロ知識証明したい

命題P:
私は公開鍵[pk]に対する
秘密鍵を知っている

Enc(pk, r)

r

乱数[r]を生成

秘密鍵を知っているのでEnc(pk, r)
を復号して平文[r]を計算可能

Bob(Verifier)は自分で生成した乱数[r]を貰うだけなので、
対話終了後に秘密鍵に関する知識をゲットしていない⇒ゼロ知識っぽい

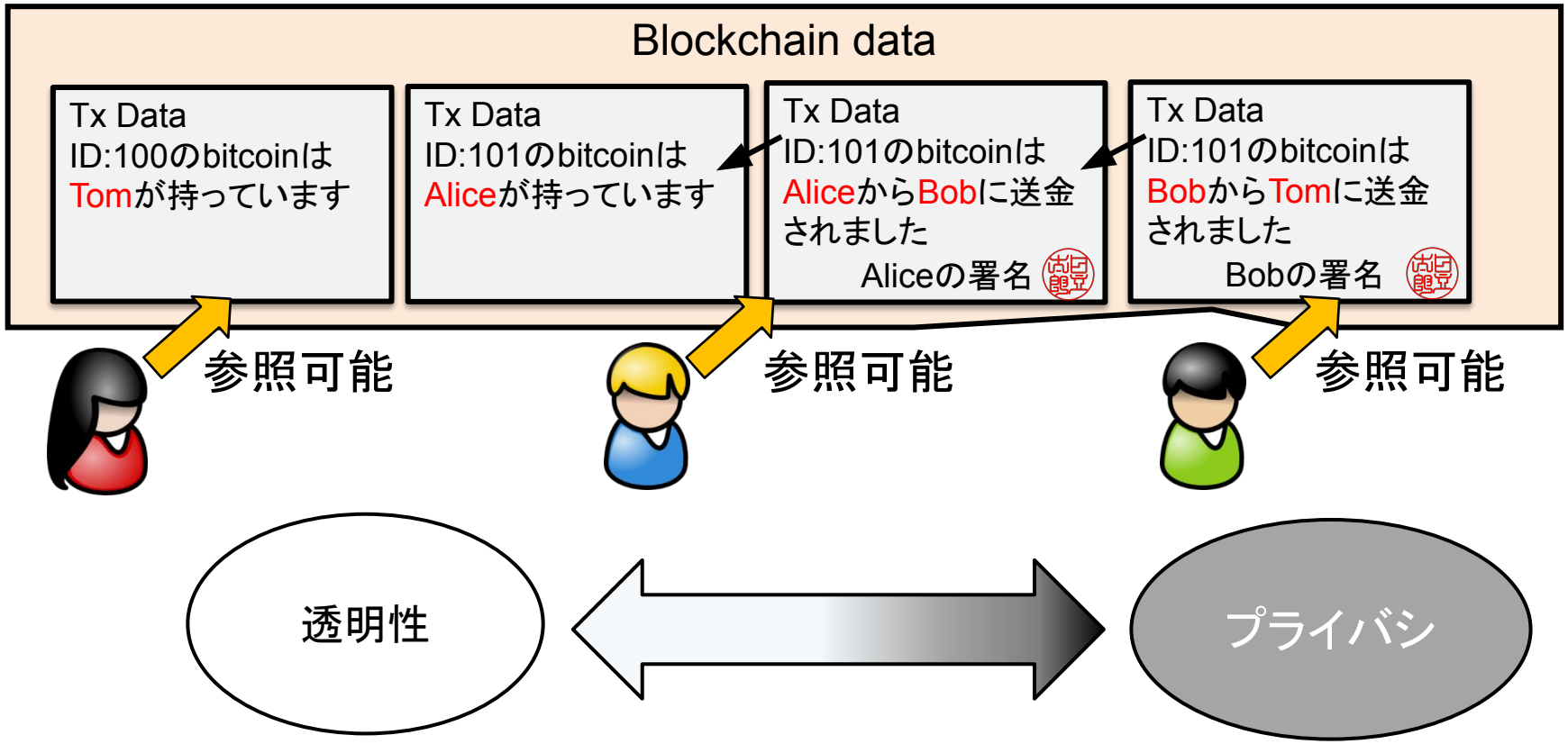
目次

HITACHI
Inspire the Next

- ゼロ知識証明とは？
- ブロックチェーンでどうやってゼロ知識証明を使うのか？
- 応用：セキュアスマートコントラクト
- Hyperledger Fabricでの実装アーキテクチャ
- 評価実験

プラバシー vs 透明性

Public Blockchainの台帳データはネットワーク上の誰でも参照可能
⇒透明性maxでプライバシーなし

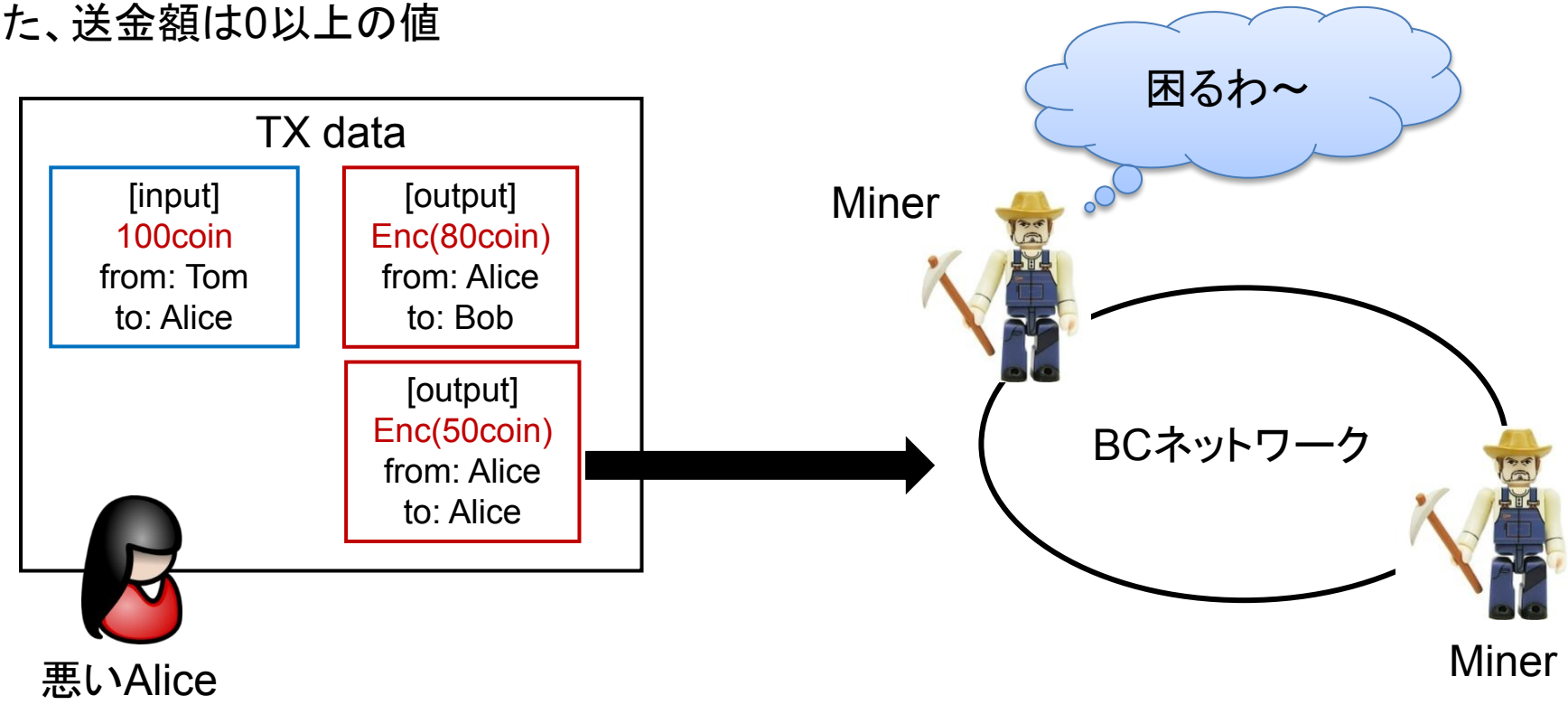


トランザクションデータを暗号化(or hash化)したら、確かにプライバシーは保たれるが、
マイナーがトランザクションの正当性を確認不能になる

もしトランザクションが暗号化されていると・・・

HITACHI
Inspire the Next

プライバシ保護のためUTXOで送金額が暗号化されたとすると・・・
[inputの金額の合計]=[outputの金額の合計] (手数料は0とする)
また、送金額は0以上の値



Aliceは100coin分のunspent transactionのうち80をBobに送金、20をAliceに送金(おつり)するが、送金額が暗号化されている事をいいことに、おつりを50にちょろまかした。
マイナーは復号化鍵をもっていないので金額不正に気付かない！

ブロックチェーンとゼロ知識証明の関係

HITACHI
Inspire the Next

一般的なゼロ知識証明の使い方
トランザクション内のプライバシ情報を秘匿化した際に正当性をゼロ知識で与える

ZCashのイメージ

秘匿化されたTx
+ゼロ知識証明

プライバシ情報は秘匿化されているが、
正当性を示すゼロ知識証明が付いている

Miner

ここまでをまとめると

HITACHI
Inspire the Next

ブロックチェーンの台帳情報は誰でも閲覧可能
なのでプライバシの観点で問題あり

それならプライバシ情報は暗号化しましょう

でも暗号化したらマイナーが正当性を確認できません

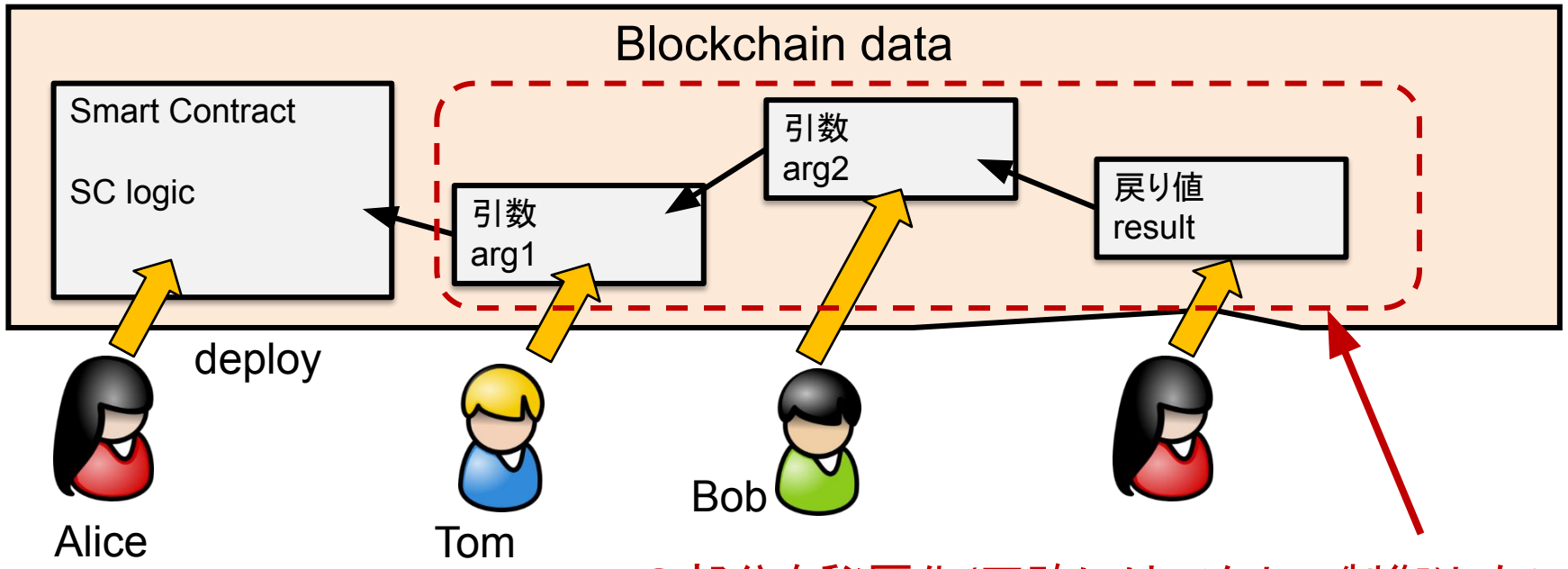
正当性を証明するゼロ知識証明を付けましょう！

目次

HITACHI
Inspire the Next

- ゼロ知識証明とは？
- ブロックチェーンでどうやってゼロ知識証明を使うのか？
- 応用：セキュアスマートコントラクト
- Hyperledger Fabricでの実装アーキテクチャ
- 評価実験

Secure Smart Contractのイメージ

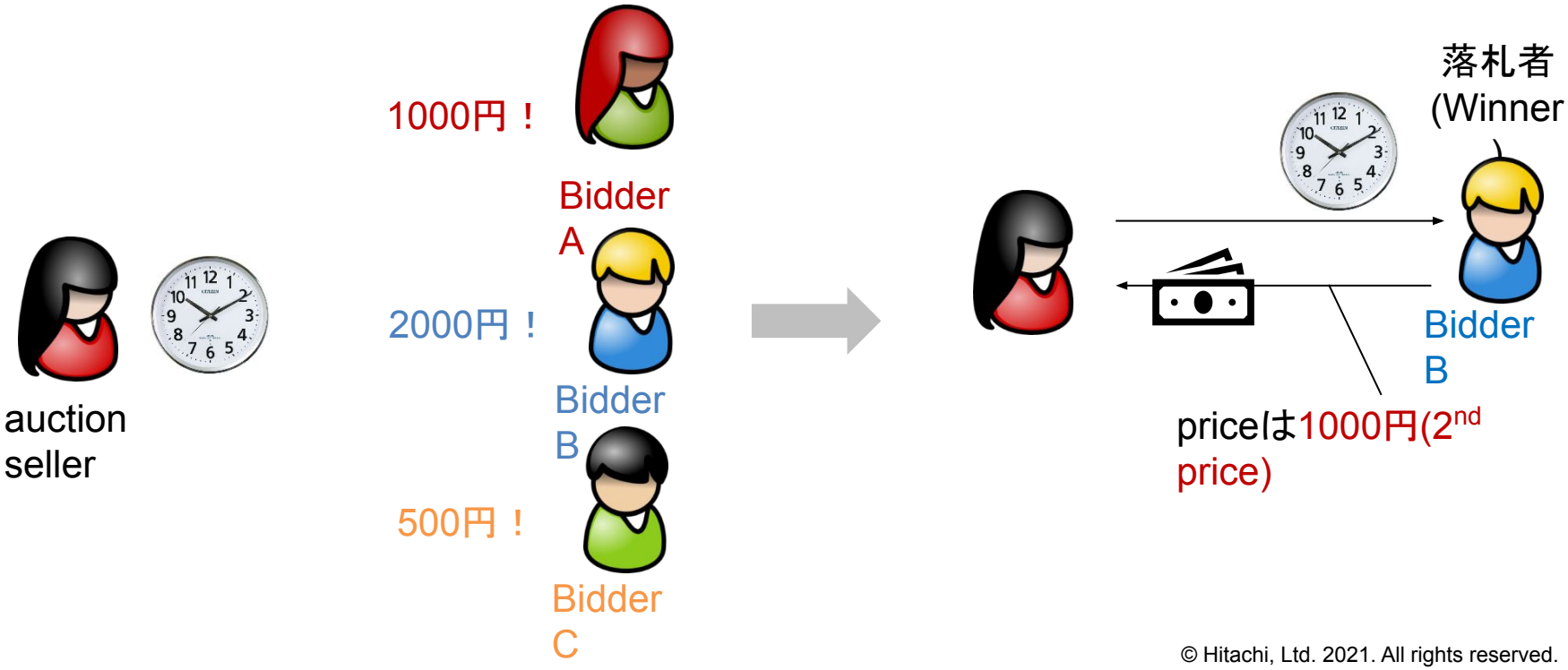


・Alice(contract owner)はSCをBC上にdeploy
・Tom, BobはそのSCに対してargument txを送信
・Alice(or 誰かが)SCをargに対して実行し、結果をBCに記録
この時、arg, resultを適切にアクセス制御したい
(i.e., argは本人とAlice(contract owner)のみ閲覧可能、resultも関係者のみ閲覧可)
SCのロジック自体は誰でも閲覧可能(open)

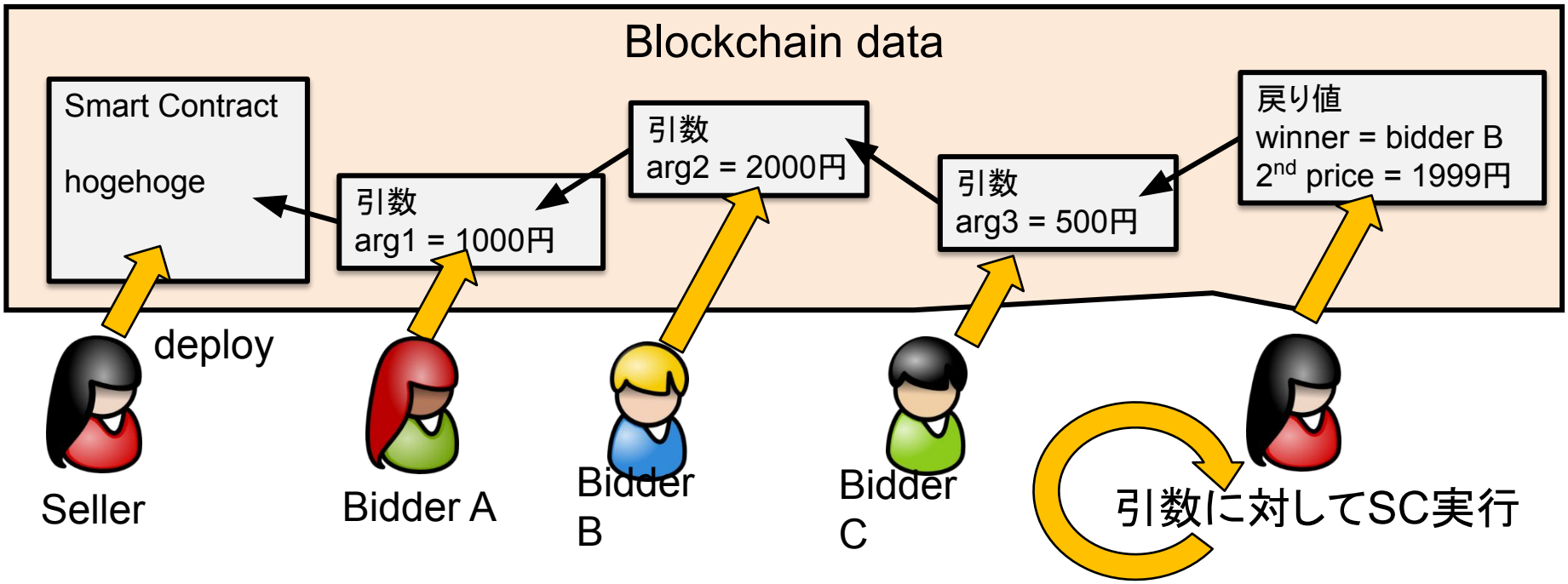
実例：2nd price auction

・各入札者は1度だけ払っても良い最高額を入札する(bidding)
・最高額を入札した人が落札者winnerになる(下の図ではbidder B)
・落札者が支払う金額は入札額2位の人の金額(下の図ではbidder Aの1000円)

※1st priceオークションと違い、払いすぎる心配がない



BC上で2nd priceオークションは可能か？



【懸念点】引数、戻り値がopenだと・・・
・2nd priceオークションにならない、公開入札(English auction)になる
・Bidder B入札後、Sellerの利得最大化のために、1,999円で入札される(1^{st} price)
・みんなの入札金額、落札者がopenになる(プライバシ問題)
⇒引数、戻り値の暗号化が必要

【懸念点】引数、戻り値を暗号化し、Sellerにのみ開示すると・・・
・Sellerが実行結果を改ざんできる(e.g. 2^{nd} price=1,999円)

2nd priceオークションのロジック

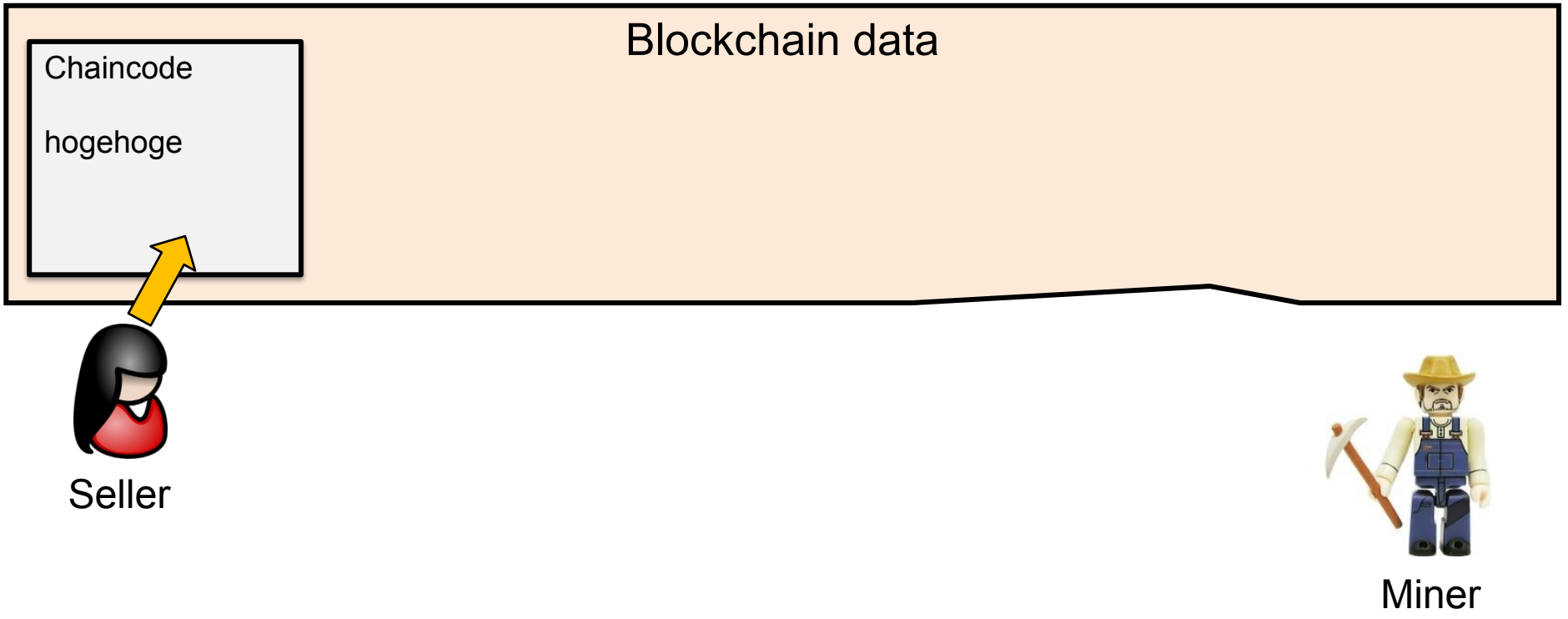
後でSellerが実行する。

※疑似コード

```
1: winner = 0
2: max = 0
3: 2nd_price = 0
4:
5: // Nは入札者数
6: for i=0;i<N;i++
7:
8: // 勝者判定
9:  if bidder[i].price > max
10:    winner = i
11:    2nd_price = max
12:    max = bidder[i].price
13:
14: return {winner, 2nd_price}
```

全員の入札金額はSellerしか知らないので、不正改ざんしてもばれない。

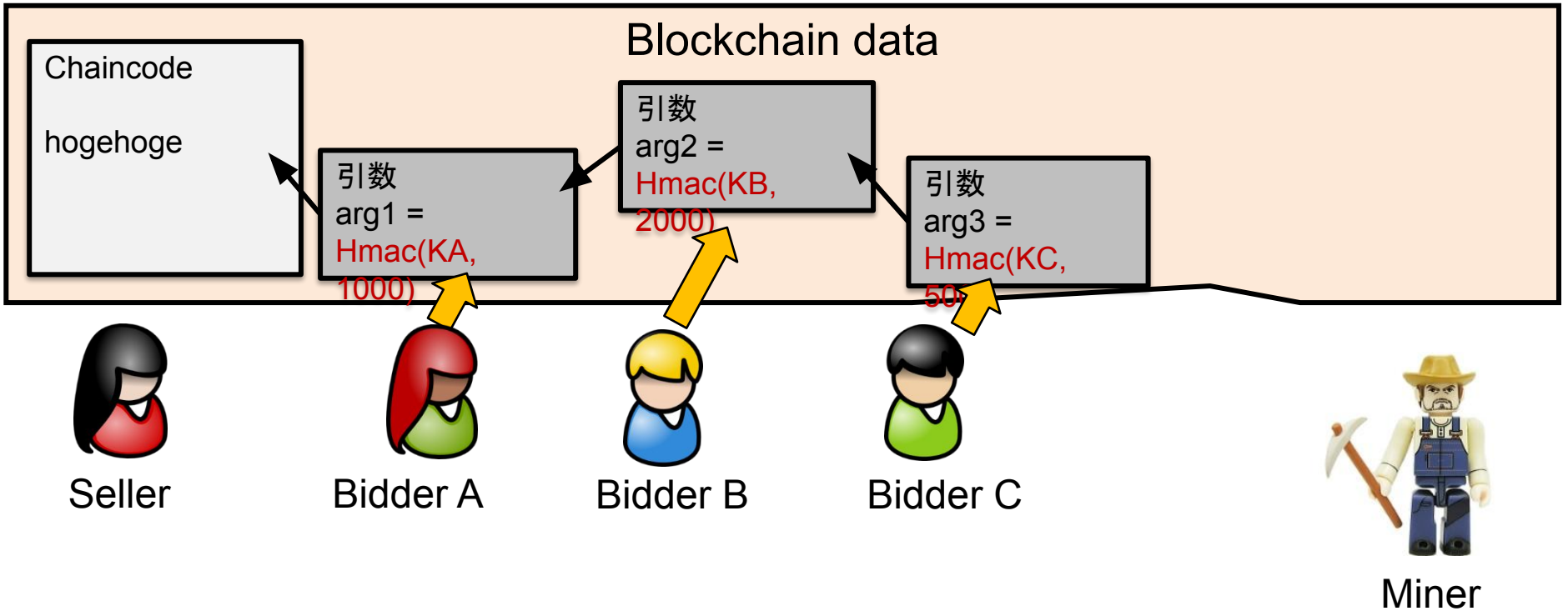
ゼロ知識証明を使って不正改ざんを防止



Phase 1, 2(initialization & deploy):
Sellerは2nd priceオークションのチェインコードをデプロイ

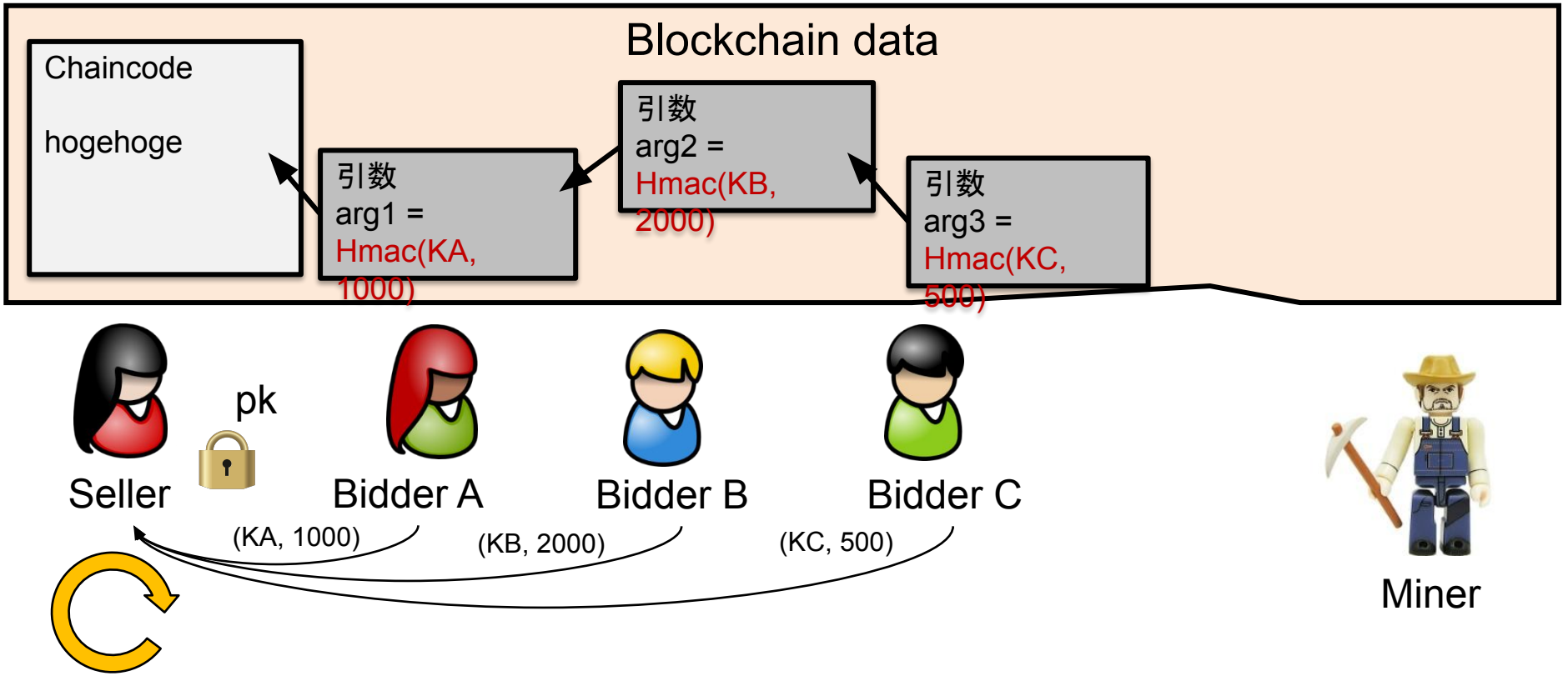
(Fabric)CAはこのチェインコードに対して以下のzk-SNARKのパラメータを生成する
・ゼロ知識証明生成用の鍵pk⇒Sellerに渡す
・ゼロ知識証明検証用の鍵vk⇒Peerに渡す

ゼロ知識証明を使って不正改ざんを防止



Phase 3(betting):
各Bidderはtemp key(KA,KB,KC)を生成し、入札金額のHmac値を台帳に書く

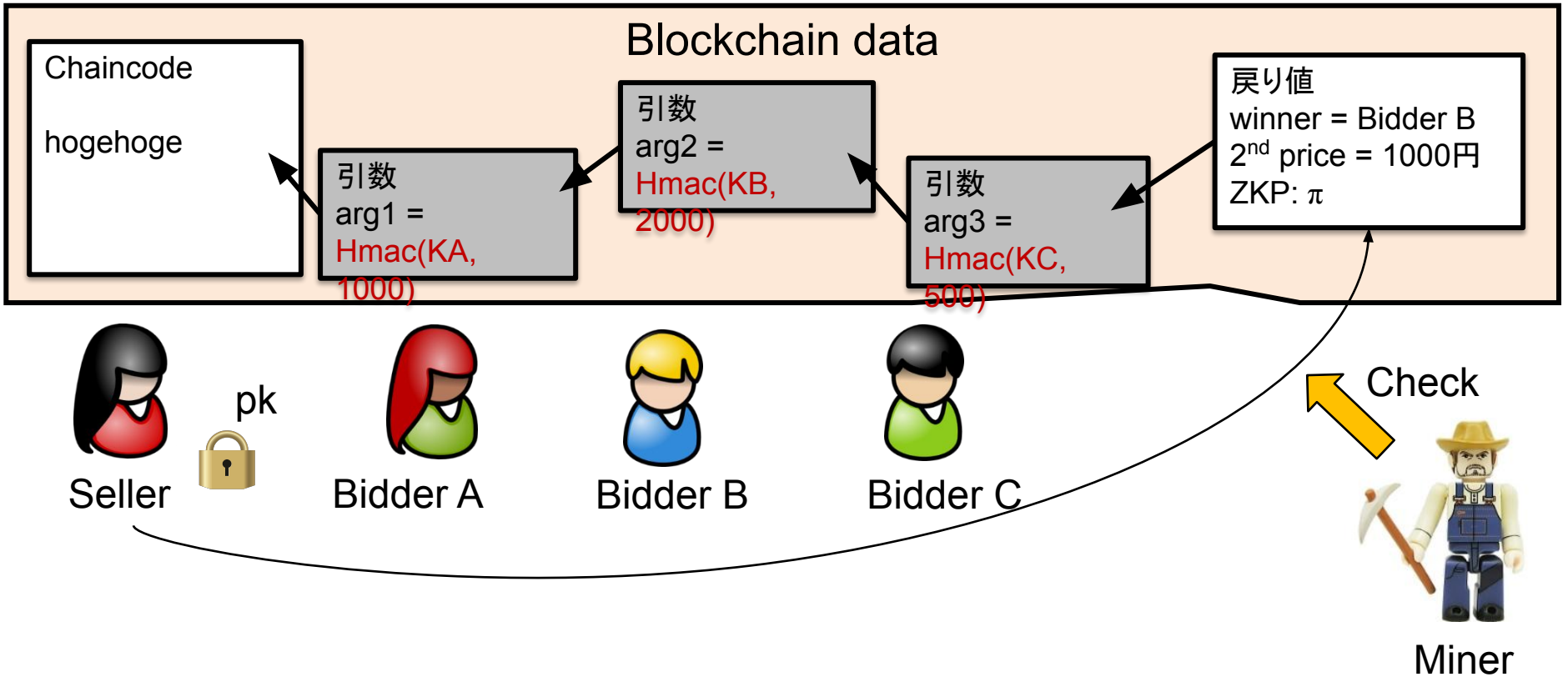
ゼロ知識証明を使って不正改ざんを防止



Phase 4(execution): オフライン処理に注意！！
各Bidderはtemp keyと入札金額をSellerに渡す(Logicの引数たち)

Sellerは2nd priceオークションのロジックを実行し、winnerと2nd priceを算出、
確かにロジックを実行した、という証拠にpkを使ってゼロ知識証明[π]を生成

ゼロ知識証明を使って不正改ざんを防止



Phase 5(result):
Sellerは(winner, 2nd price, π)を台帳に書き込む(現状版では戻り値はopen)

が、この際にMinerが[正しい引数(1000, 2000, 500)]で[2nd priceのロジック]が実行されたかを検証する(invalidならtxは書き込まれない)
この検証は検証鍵[vk]と[π]と台帳上の[HMAC値]のみで可能

目次

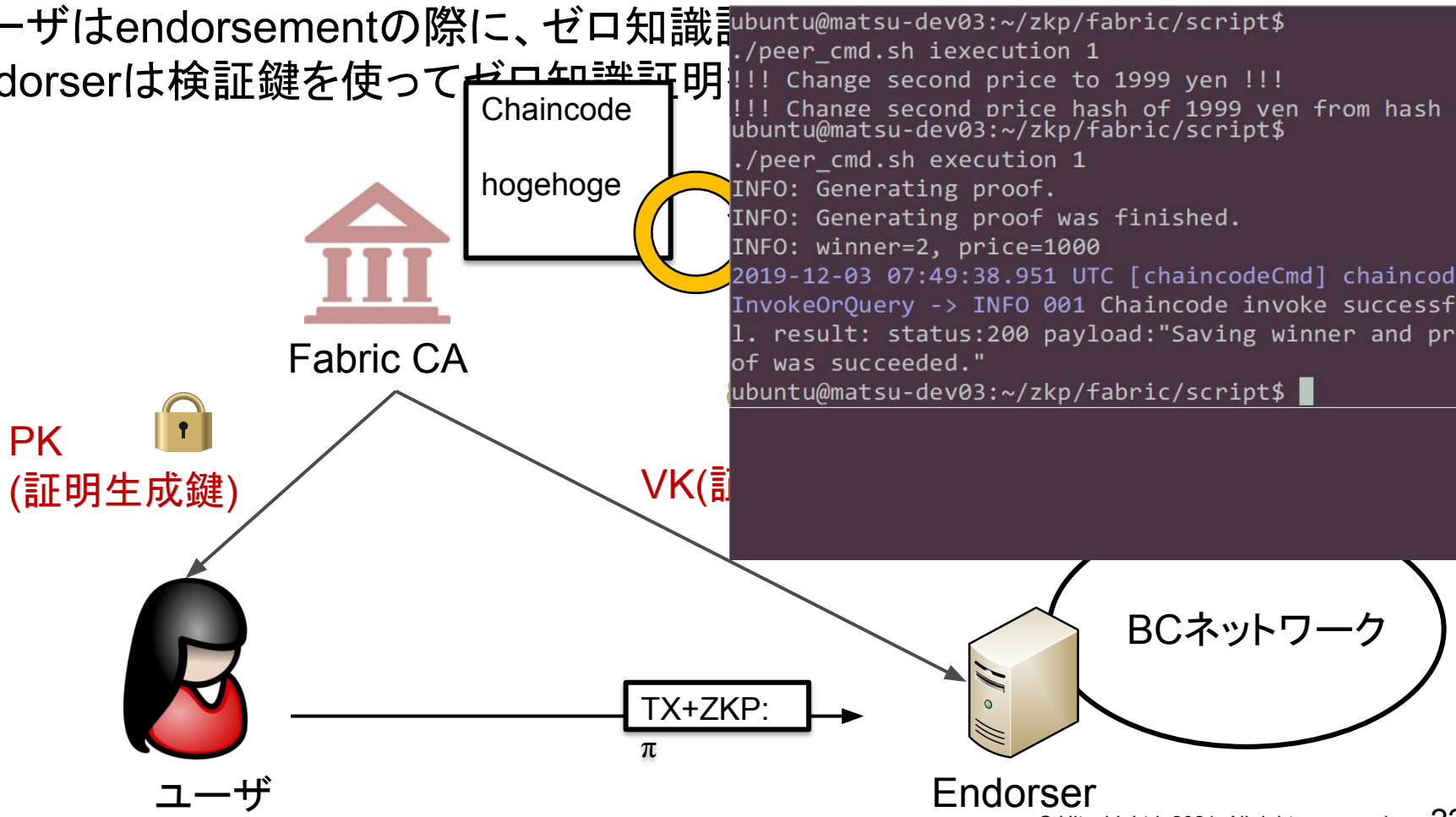
HITACHI
Inspire the Next

- ゼロ知識証明とは？
- ブロックチェーンでどうやってゼロ知識証明を使うのか？
- 応用：セキュアスマートコントラクト
- Hyperledger Fabricでの実装アーキテクチャ
- 評価実験

Hyperledger Fabricでの実装アーキテクチャ

HITACHI
Inspire the Next

・ゼロ知識証明したいコントラクトをユーザが指定
・Fabric CAがコントラクトのロジックに対して、ゼロ知識証明生成鍵/検証鍵を生成
・ゼロ知識証明生成鍵をユーザ、検証鍵をEndorserに配布
・ユーザはendorsementの際に、ゼロ知識証
・Endorserは検証鍵を使ってゼロ知識証明




```
ubuntu@matsu-dev03:~/zkp/fabric/script$
./peer_cmd.sh iexecution 1
!!! Change second price to 1999 yen !!!
!!! Change second price hash of 1999 yen from hash k
ubuntu@matsu-dev03:~/zkp/fabric/script$
./peer_cmd.sh execution 1
INFO: Generating proof.
INFO: Generating proof was finished.
INFO: winner=2, price=1000
2019-12-03 07:49:38.951 UTC [chaincodeCmd] chaincode
InvokeOrQuery -> INFO 001 Chaincode invoke successfu
l. result: status:200 payload:"Saving winner and pro
of was succeeded."
ubuntu@matsu-dev03:~/zkp/fabric/script$
```

修正版：2nd priceオークションのロジック

HITACHI
Inspire the Next

ロジックに入力金額の不正が無いかチェックする機能を追加(8～10行目)

※疑似コード

```
1: winner = 0
2: max = 0
3: 2nd_price = 0
4:
5: // Nは入札者数
6: for i=0;i<N;i++
7:
8: // 入力値の不正チェック
9: if bidder[i].hashed_price !=  hmac(bidder[i].hash_key, bidder[i].price)
10:   abort()
11:
12: // 勝者判定
13:  if bidder[i].price > max
14:    winner = i
15:    2nd_price = max
16:    max = bidder[i].price
17:
18: return {winner, 2nd_price}
```

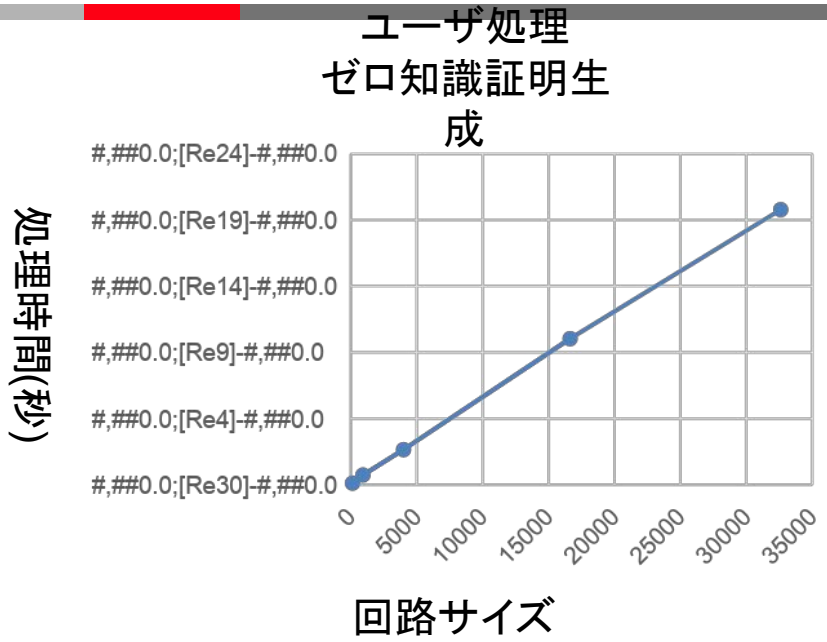
ZKP + verification keyと公開情報のみを用いて、上記のロジックが
確かに実行された事を確認可能(verifiable computation)

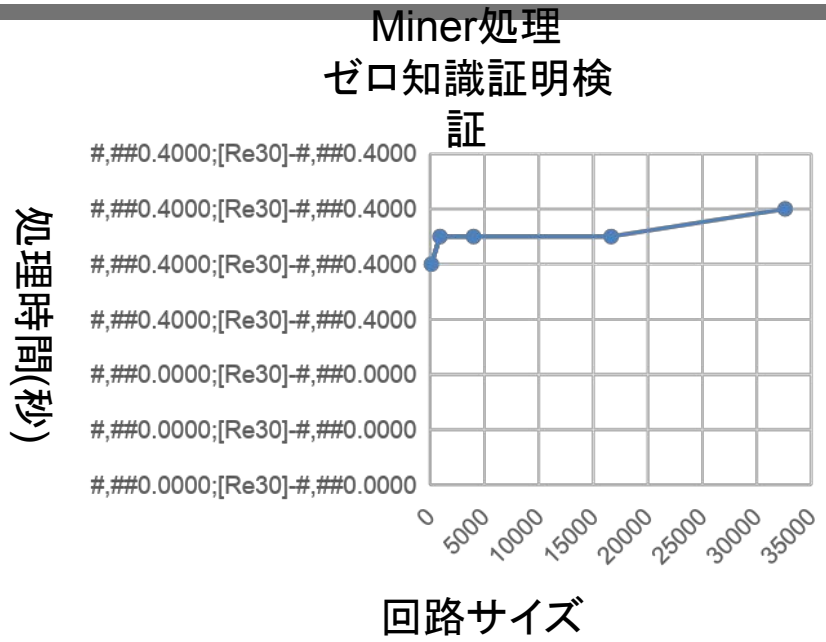
目次

HITACHI
Inspire the Next

- ゼロ知識証明とは？
- ブロックチェーンでどうやってゼロ知識証明を使うのか？
- 応用：セキュアスマートコントラクト
- Hyperledger Fabricでの実装アーキテクチャ
- 評価実験

性能評価実験





評価方法
量子計算機への耐性を考慮し、LWE暗号方式をベースに方式設計
zk-SNARKの標準ライブラリlibsnarkの一部機能を利用し実装
様々な回路サイズ(スマートコントラクトのサイズ)に対して評価実験を実施

実験結果(評価PCはIntel(R) Core(TM) i7-9700K CPU @ 3.60GHz ×8)
・ユーザ側の処理時間(ゼロ知識証明生成)は回路サイズに比例して増加
⇒Hash関数の評価には210秒, 300GBのメモリが必要になる計算
・Minerの処理時間(ゼロ知識証明検証)は回路サイズによらず1msec程度

HITACHI

Inspire the Next