暗号基礎
秘密のゲーム

Thaddeus Dryja
BC-2 2017-02-05

ビットコイン使ってる暗号学

- ハッシュ　ファンクション　(proof of work)
- 楕円曲線を利用した署名
- それだけ！
  - No encryption (ブロックチェーンには）
  - No key agreement (BIP151以外）

説明とゲーム

- 手作業 proof of work

- 隣の人と Diffie-Hellman

Hash function proof of work

- みなのパソコンには sha256sum あります
  - $echo "hi" | sha256sum

    98ea6e4f216f2fb4b69fff9b3a44842c38686ca685f3f55dc48c5d3fb1107be4 -
- 98 で始まる;うまくいかなかった
- Mac には | shasum -a 256

Hash function proof of work

```
$echo "hi" | sha256sum
```

```
98ea6e4f216f2fb4b69fff9b3a44842c38686ca685f3f55dc48c5d3fb1107be4w
```

```
$echo "hi 38" | sha256sum
```

```
2b8957bff0d9c5ba9f86efe3337b5ace3fca82fcd89a2cd69283c1e766d8121a
```

まあまあ、近くなってきた

Hash function proof of work

```
$echo "hi" | sha256sum
```

98ea6e4f216f2fb4b69fff9b3a44842c38686ca685f3f55dc48c5d3fb1107be4w

echo "hi 48" | sha256sum

08ff361e13d1c4cd90035b1c174d127d5b015fa1ca602e2589e5e58311fa74c8

ゼロで始まった、ご苦労さん！　うまくいきましたね！

Hash function proof of work

```
$echo "Tadge 99" | sha256sum
```

```
407dfd1e8ff3a2cdf923381e6807c9dd210faa5e1d27c9e830416c64b0709f08
```

echo "Tadge 45" | sha256sum

```
05d394d1abc280cc3076f206e2564d9e7728a4fbf1b35317e04a8afd3f56662f
```

私だけのproof of work

勝負しようぜ

```
$echo "MyName 00" | sha256sum
```

```
407dfd1e8ff3a2cdf923381e6807c9dd210faa5e1d27c9e830416c64b0709f08
```

見つけた数字の中で一番低いものを、slackにアップしてください

みんな簡単に確かめられる

3つのゼロ　000　で始まるはブロック見つけられれば、ブロックを作る権利を得て、お金を貰います！

Diffie-Hellman Key Exchange

- 電子署名の仕組みではないですが、その基礎となる数学を使っている、秘密の数字を共有する方法
- 電子署名よりシンプル
- 秘密の鍵と公開できる共通の情報を使います
- ２人で同じような計算をして、shared key見つけます
- 通信路上で情報を盗聴している人はその情報からshared keyを知ることはできない

Diffie-Hellman

- 冪乗剰余（べきじょうじょうよ: Modulo exponentiation）
- $g^a \bmod p = A$, $g^b \bmod p = B$
- $(g^a)^b \bmod p = (g^b)^a \bmod p$
- $A^b \bmod p = B^a \bmod p$ = shared key
- aとbが秘密、AとBは見せる
- aとbが秘密である限り、盗聴したAとBからshared keyを計算するのは難しい

Pythonで隣の人とやってみましょう

- python

  >>> pow(2, 10)

  1024

  2^{10} ですね

  >>>pow(2, 10, 100)

  24

  1024 / 100 の　剰余　は 24

Parameters

Base = 10

Modulus = 97

(97が素数、10は97のprimitive root)

秘密鍵を作る

２桁の数字をランダムで決める

自分で決めなければ、前のpowから取る

誰にも言わない、秘密です!

秘密鍵作業はこれで終了

Public 鍵 を作る

```
>>> pow(10, ___, 97)
```

public _key

例えば

```
>>> pow(10, 21, 97)
51
```

10^{21} mod 97 = 51

10^{21} =

100000000000000000000

100000000000000000000 / 97 =

10309278350515463917,

剰余は　51

Public keyをみんなに発表して

私のpublic keyは51です！

隣の人と、shared key を作ります

例えば 相手の public key は 25 (秘密鍵は 22)

$25^{21} \bmod 97 = 12$

>>> pow(25, 21, 97)

12

$51^{22} \bmod 97 = 12$

>>> pow(51, 22, 97)

12

凄い数学です

隣の人とする

```
>>> pow(their_pub, my_priv, 97)
shared_secret


>>> pow(their_pub, my_priv, 97)
shared_secret
```