Problem Set 1

Collin Berman
cmb5nh

September 15, 2015

Problem 1.

a. fcd81f242bdd75bfb773281a3b695eca9d5630baca34171e78c14fc516fe9deb

b. 0.004501 BTC = 1.03 USD

c. For Block #371914:
Total output: 7,531.39109632 BTC = 1,722,655.09 USD
Estimated transactions: 1,755.15212027 BTC = 401,455.94 USD

d. The transaction (received 2015-08-28 15:09:01) was included in Block #371914, so Block #371916 (2015-08-28 15:33:05) was its third confirmation. Then it took 24 minutes, 4 seconds to get 3 confirmations.

Problem 2.

a. Using https://blockchain.info/tree/99847340

   i. 14yyZsLZZn7qkvMfvLBD7cJvuYXytn2bdQ (got 10x as much as everyone else???)

   ii. 112Kh8RHajxsj2yqdvrdcq5L4bLm8xymY (me)

   iii. 1FM45Varjz955Sh9SrEp41XH3gRTcFM4i6 (got 0.000001 BTC less than everyone else ͡ ̈ ͡)

   iv. 1G8Dbnesf35V7gKRuc3Cv4EeGCwWqMdXkE

   v. 1KEz1rFhEoy2vQV4zQcUCFWPYy7EkJyRcM

   vi. 15X2qKbsXxPHbgfFTc6UgXmETVe7fpBUzX

   vii. 1DkHaFKyrXhE28KsybBntCxoYeSoBumPfU

viii. 1GMrGqvF8FwbgCSShCdzijcSUHzmNR1VCz
ix. 1kKvKdXcLvyHJTFuNxkdxm2tEnkG8hB1S
x. 1QLT7GNBnKNrGnKi7HNnZTYSDbPvUaVoZs
xi. 12YwFZNmUoexnQAMKFvRJSSydxTZUZPWx8
xii. 1MDjCqjwnKmMWPxawpgN1UuDfbMUUgqnWw
xiii. 1LeXjaxMbugBveWusTRpbFtx13N5HamcE4
xiv. 15j1jdJsMa4vR71gcZ6FCfYeAWJdPwpn7W
xv. 1MzdKiBn5qr8CS1cbts6TQquxsAo52yFKe
xvi. 163vXnDXSc2hEKMrWHkERzBJWuuKJve5Nc
xvii. 19Y4oNeGcdmBAfDXpJjPbiZ35PnZz57Ar2
xviii. 1MtYZBtRw8XcnTgEY8qQphbddnwwFcoEXH
xix. 1JHoF2bak7KCST3SzeR7e1AwqDm4tiLJjt
xx. 1D6qsGhZqrRSKegqWT3e8TPR8gGczTxMLu
xxi. 1AcKeSkKponv5qeZuEHvkocELf1Uo4ggCE
xxii. 17GBpDP8yHTZcBJ9WmmRMjCmgiqacy5v3n
xxiii. 1fiWBi5u7oDzQrMNjeNLbmAvvHv545oy9
xxiv. 1NyBMbtqZLAmB3CSbjHzDHvedRJJJ7CupY
xxv. 1N96tMSyeeANTRFApr3zA6ML3Qow1gZR9A

b. It looks like 0.2 BTC was purchased (66809ee9b41310b53e3fc94cbff8fe7c870d6df4ca7561614137311f651ed9a8), but I'm not sure how to figure out which exchange it comes from

c. All blockchain.info can tell me is where the transaction was relayed from, which gives me no information as to where you're sending from. For example, note that the transactions sending bitcoin to members of the class come from different countries, but I doubt you're traveling that much in between sends.

Problem 3. a. If the wallet is network connected, the wallet could leak private keys (or the randomness used to generate them) to someone. If this is done well using covert communications, it would be undetectable. Alternatively, the wallet could use a weak source of entropy, so that the wallet developer could generate the same private keys users are generating. This shows why open-source code is important. Even if the code is open, the website could provide

a malicious package. This could be done with plausible deniability, say using compiler bugs or a weak ssl certificate and claiming MitM.

b. If I'm using someone else's wallet program, I'd want to build from source and compare the result to the package provided by an installer, as well as to other deterministic builds from other places using something like Gitian. Alternatively, I could make my own simple wallet program, which wouldn't be malicious.

Problem 4. Let's convert to hexadecimal:

$$2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1 = (16^{64} - 1) - 16^8 - 3 * 16^2 - 13 * 16$$

which is obviously the same modulus

Problem 5.

- The golang package we use needs to be secure / not have bugs that could be used to introduce vulnerabilities
- btcec needs to be secure
- The way we acquire golang and btcec over the network needs to be impervious to tampering
- We need to trust keypair.go, but it's pretty simple
- Our system needs to be secure, that is, no keyloggers or other spyware that could capture the private key (or BadBIOS)

Problem 6.

```
func generateVanityAddress(pattern string)
        (*btcec.PublicKey, *btcec.PrivateKey) {
    var pub *btcec.PublicKey
    var priv *btcec.PrivateKey
    for {
        pub, priv = generateKeyPair()
        addr := generateAddr(pub).String()
        if matched, _ := regexp.MatchString(pattern, addr); matched {
            break
        }
    }
    return pub, priv
}
```

Problem 7. I created the address 1Morexj2GoDiwfuVkzCGUs1dc9FCu1BTaX

Problem 8. If we assume a user who wants to pay us is checking the vanity part of the address, plus the next few characters, then we have extra security. If a malicious adversary wants to create an address that looks like ours, they will have to spend more computing power than we did to generate the address.

Problem 9. 943d61864f03e41cc621beaae331d89b5b29c7252239b822c0033ee802e2ed49

Problem 10. 79e4461fa048361a34de67be7f9c1e0d02018e0aa67cb1883856529f2b923cc0

Problem 11. cc121d12be5756947d8c65735255002f33bd7837462a3c17182b9505c4298d40
cfc2b7f7ced1376392a1ef4730625b0be6f2d49e234f3a741f147117c07209be

Problem 12. To get two transactions with at least one confirmation each, I'd have to transmit the two transactions to different mining pools, which would then have to each mine the block they're working on before they are notified of the latest block. There are normally between 0 and 3 orphaned blocks mined a day, so I would have to resend my transactions each time there is a new block, and even then, I probably wouldn't be able to achieve this.