6.857 Homework	**Problem Set 2**	# 2-1 Secret Sharing
Changping Chen		
Peinan Chenn		
Skanda Koppula		March 9, 2015

(a) We can give only A and B shares of the secret and then require that there must be 2 out of 2 parties to construct the secret in a Shamir secret sharing scheme.

(b) Assuming we have some combination of gates that take inputs and outputs the secret if the inputs are correct, we will start our definition of our scheme by working top down starting from the output. Starting from the output, whenever there is an OR gate, we make it so each input of the OR gate will construct the same secret, which is also the output secret. Whenever we see an AND gate, we give each of the inputs of the gate a different share that is also itself a secret and make it so that we need shares equal to the number of inputs of the AND gate to produce the output secret with Shamir secret-sharing. Each input to an AND gate has its own secret (its share), different from that of the other inputs to that AND gate as well as the output. The input to the AND gate needs to determine its own secret before it can provide an input to the AND gate. In other words, we make each AND gate a N out of N gate and make it so that each input to the AND gate needs to construct its share since the share is also a secret. For each T out of N gate, we use Shamir secret-sharing. Part c will illustrate how this scheme works with an example.

(c) We make it so both inputs of the OR gate will construct the same secret. So, Professor Rivest gets the secret and ((2 out of 3 TA's) AND (10 out of 20 students)) will be able to construct the same secret. We then make the AND gate a 2 out of 2 Shamir secret-sharing gate and give both inputs a share which is also a secret. In this case, (2 out of 3 TA's) will be enough to reconstruct S1 and (10 out of 20 students) will reconstruct S2. Where S1 and S2 are secrets, S1 is not S2, and S1 and S2 are the shares that together will reconstruct out original secret. Then, we use Shamir secret-sharing for both (2 out of 3 TA's) and (10 out of 20 students) such that (2 out of 3 TA's) will reconstruct the secret S1 and (10 out of 20 students) will reconstruct the secret S2.

We spent about 1 hour on this problem.

6.857 Homework	**Problem Set 2**	# 2-2 Hash Functions
Changping Chen		
Peinan Chenn		
Skanda Koppula		March 9, 2015

(a) $E[collisions] = \sum P(h(x_i) = P(x_j)) = \binom{n}{2} \cdot 2^{-d}$. When n, the number of hashes we compute, is $c \cdot 2^{d/2}$:

$$E[collisions] = \binom{c \cdot 2^{d/2}}{2} \cdot 2^{-d}$$

This approximates to $\frac{(c \cdot 2^{d/2})^2}{2} \cdot 2^{-d} = \frac{c^2}{2}$.

(b) XOR'ing two inputs does not change one-way of the function. $x \oplus y$ distributes evenly across the input space $\{0,1\}^n$ of $h(x)$; if finding n given $h(n)$ is is worst case $O(2^d)$ operations, finding $x \oplus y$ (and thus x, y) given $h'(x, y) = h(x \oplus y)$, is still $O(2^d)$.

However, in the case of AND'ing the two inputs, $x \wedge y$ distributes unevenly across $\{0,1\}^n$; for example, given a random x, y, our input to $h(n)$ is much more likely to have be entirely 0's than entirely 1's. Because our input space collapsed in a certain direction, iterating through x, y to find our $h'(x, y)$ no longer is expected $\Theta(2^d)$.

(c) Not collision resistant. For any x_1, x_2, pick y_1, y_2 such that $y_1 = h(x_2)$ and $y_2 = h(x_1)$.

$$h'(x_1, y_1) = h(x_1) \oplus y_1 = h(x_1) \oplus h(x_2) = h(x_2) \oplus h(x_1) = h(x_2) + y_2 = h'(x_2, y_2)$$

We have a collision.

(d) Not weak collision resistant. Because $h(x)$ is only TCR, we can assume that finding x_1, x_2 s.t. $h(x_1) = h(x_2)$ is easy. That means for target $h'(x, y) = 0$, finding colliding x, y is easy with the aforementioned pairs because $h(x_1) \oplus h(x_2) = h(x_1) \oplus h(x_1) = 0$. More generally, for any target $h'(x, y)$ and input (x, y), the input (y, x) always collides with the target, because of the commutivity of XOR.

We spent about 2 hours on this problem.

6.857 Homework	**Problem Set 2**	# 2-3 Cryptocurrency
Changping Chen		
Peinan Chenn		
Skanda Koppula		March 9, 2015

(a) 000000c4d41c812229f06b454c5be8ed7578a839ce14a564e29439700b0000d1

(b) We (team 17) were on the longest chain until yesterday. We had several hashes on that chain, one of which was 000000000013e89b64470493ae52b798b8c0c3392be7664bf913851134eae35f at that time. There were some teams that published their entire chain today. We coded a GPU based Java program using the aparapi library and continuously tried to add to the current longest chain. The program works by randomly creating hashes and checking if they have the correct number of zeros. Every some seconds, we query to see if the current node has a next node. If so, we stop trying to build a block and work on the next block instead. Generally, we waited at most 4 confirmations to determine that our block was on the longest chain. We have about 300 Megahashes/second and have been running this continuously since 2/26. There was probably some downtime during that time, but we were essentially mining the entire time. We cooperated with teams 11, 15, and 32 and had about 1.2 Gigahashes/second together. Unfortunately, we are no longer on the longest chain. For next year, I think that people should become unable to add to blocks after some time. That is, blocks should expire over time. This would prevent the issues with /explore and make the current /head a lot more transparent. Another possible way is to generate a random value when a block is submitted that users must query for to work on the next block. We generally didn't like the complete lack of transparency throughout the process. We constantly were paranoid about whether we were actually on the longest chain or not because of this.

(c) To reverse a transaction six levels deep into the chain, you'll need a hashing rate higher than the Bitcoin network hash rate to catch up and create blocks faster than the network. The current hashrate is 339617 Terahash/s (`https://bitcoinwisdom.com/bitcoin/difficulty`). A high end ASICs designed for mining (e.g. AntMiner S4) is advertised at 2 million Mhash/sec, costs $1400, draws 1400 Watts (`https://en.bitcoin.it/wiki/Mining_hardware_comparison`). To achieve a faster than network speed, we'd need about $\frac{339617 \cdot 10^{12}}{2 \cdot 10^6 \cdot 10^6} = 169,809$ of these miners. Wed also need a few more machines to not just match, but also outpace the network to catch up our chain, so well chuck in 1106 more: 172,500 mining ASICs, for a total of $241500000 in initial hardware costs. Bitcoin requires about 51 zeroes at the end of the hash, so itll take about 2^{51} hash guesses to find a valid block hash. That means that to recover 6 blocks with our 1000 extra machines would take $\frac{2^{51}}{2 \cdot 10^6} \cdot \frac{1}{60 \cdot 60 \cdot 60} \cdot 6 \cdot 1000 = 78$ days. Assuming a $0.12 kWH electricity rate, power consumpetion would add another $0.12 \cdot 1400 \cdot (24 \cdot 78) = \$314496$.

(d) Peinan and Skanda spent it on food.

We spent over a week on this problem. If you don't include the active mining and politics, we spent about 6 hours on this problem. These 6 hours include the time to code the miner and the time to do 3c.