Blockchain Technology and Applications(CS731), Winter 2019
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Aneet Kumar Dutta
Roll Number: 18111401
Date: February 15, 2019

QUESTION 1

Let $H : P * S- > 0, 1, ..., 2^n - 1$ be a collision resistant Hash function.
Let $H' : P * S- > 0, 1, ..., 2^{n'} - 1$ be Hash function where $n' > n$.
$H'(x) = H(x)||H_2(x)$ where $H_2(x)$ is not a collision resistant function.
Since, $H_2(x)$ is not collision resistant so there exists an $x \neq x'$ but $H_2(x) = H_2(x')$.
$H'(x) = h_1 1||h_2$
where $H(x) = h_1$ and $H_2(x) = h_2$
$H'(x') = h_1'||h_2$
where $H(x') = h_1$ and $H_2(x') = h_2$
If $H'(x) = H'(x')$ which means there is a collision on $H'(x)$ then $h1'$ must be equal to $h1$. So, we can say a collision on $H'(x)$ yields a collision on $H(x)$. Since, $H(x)$ is collision resistant then we can say $H'(x)$ is also collision resistant.

For every puzzle $p \epsilon P$ if we can find a solution S which satisfies the condition $H(p|S) < 2^n/d$, where d is the fixed difficulty then that same S will be the solution for the puzzle $H'(p|S) < 2^{n'}/d$
d is said to be fixed and $n < n'$ so the solution S which satisfies the condition $H(p|S) < 2^n/d$ will obviously satisfy the condition $H'(p|S) < 2^{n'}/d$ because $2^n < 2^{n'}$ and d is fixed. Therefore, finding the solution of $H(p|S)$ will yield us the solution of $H'(p|S)$. So, we can say that $H'(x)$ is a collision resistant function but not puzzle friendly.

Blockchain Technology and Applications(CS731), Winter 2019
Indian Institute of Technology Kanpur
Homework Assignment Number 2

QUESTION

2

Student Name: Aneet Kumar Dutta
Roll Number: 18111401
Date: February 15, 2019

Part A:

Here $S = T_1, T_2, ..., T_9$ and $k = 3$ which is a ternary tree. There will be 9 leaf nodes containing the transactions $T_1, T_2, ..., T_9$. Merkle tree root will have 3 children and each children containing $H(T_i + T_{i+1} + T_{i+2})$ and the root hash is equal to $H(H(children_1) + H(children_2) + H(children_3))$. For commitment and for proving the membership of any transaction T_i Alice needs to store the value of the root hash, the hash of the sibling transactions along with the root's children's hash in which transaction T_i is not present.

Membership of T_4 in S.
1. Calculate $Hash(T_4)$.
2. Calculate $Hash(Hash(T_4), Hash(T_5), Hash(T_6))$ where $Hash(T_5), Hash(T_6)$ are previously stored values.
3. Calculate $Hash(Hash(Hash(T_1), Hash(T_2), Hash(T_3)) + Hash(Hash(T_4), Hash(T_5), Hash(T_6)) + Hash(Hash(T_7), Hash(T_8), Hash(T_9)))$ where $Hash(Hash(T_1), Hash(T_2), Hash(T_3))$ and $Hash(Hash(T_7), Hash(T_8), Hash(T_9))$
are previously stored values and $Hash(Hash(T_4), Hash(T_5), Hash(T_6))$ is calculated in the previous step.
4. The calculated hash in the previous step is matched with the earlier stored root hash. If it matches T_4 is a member in S otherwise not.

Part B:

Assuming proof size consists of number of hashes needed to store for verification.

So, in k-ary tree the depth=$\lceil log_k n \rceil$
proof size=$\lceil log_k n \rceil * (k-1) + 1$(for root hash)

Part C:

Proof size overhead for k-ary tree= $\lceil log_k n \rceil * (k-1)$ which is because of storing $(k-1)$ hashes needs to be stored at each level except for the root level.

Proof size overhead for binary tree=$\lceil log_2 n \rceil$

Advantage: Computation cost depends on the depth of the merkle tree.
The computation decreases by the factor of $\lceil log_k n \rceil \,/\, \lceil log_2 n \rceil$ which will be less than 1 when $k > 2$.

Blockchain Technology and Applications(CS731), Winter 2019
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Aneet Kumar Dutta
Roll Number: 18111401
Date: February 15, 2019

QUESTION
3

Part A:

The mentioned scheme does not achieve the intended security goal and Bob can determine the phone numbers because the hash of the phone number is calculated without including any nonce. BobCrypt is taking the $H(phonenumber)$ to create an account and also calculates the $H(phonenumber)$ of all the phone numbers in contact list and check if the hash already exists in the system as an account. BobCrypt Server can pre-compute the hashes of 10^{10} combinations possible for all phone numbers in a conceivable amount of time and store the corresponding hashes with the number. So, when any hashes matches with the pre-computed hash Bob can actually know the phone number.

Part B:

Now, when creating account a 128 bit random nonce is taken,$H(phonenumber, r)$. But this nonce is no where stored. During calculation of hash from the contact list and checking if the user has already an account, the hash will be calculated with different nonce as the earlier one was not committed. So every time the hash of the existing account's phone number is taken with different nonce and will not match.Therefore, the intended functionality can not be achieved.

Blockchain Technology and Applications(CS731), Winter 2019
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Aneet Kumar Dutta
Roll Number: 18111401
Date: February 15, 2019

QUESTION

4

Part A:

ScriptSig: <password>,here password is "okapi" which matches with the hash given in the question.

Part B:

Using password to secure bitcoins is not a secure way because if anyone who can steal the password can redeem the bitcoins. The bitcoins can be redeemed by a password with the given hash, so this mechanism is vulnerable to rainbow attacks and even to some extent to brute forcing. Since, no salting is used rainbow attack is possible in this scenario.

Part C:

Implementing this with Pay-to-script-hash fix the security issues because in P2SH the bitcoins is send to the hash of the redeem script and from the hash we can not guess what is written in the script. P2SH transactions are valid even if the redeem scripts are invalid. So, guessing or brute forcing can not be done here because transaction with a wrong redeem script will be a valid transaction and the bitcoins will be locked forever.

Blockchain Technology and Applications(CS731), Winter 2019
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Aneet Kumar Dutta
Roll Number: 18111401
Date: February 15, 2019

QUESTION

5

Lightweight client have the header of the current blockchain header. Therefore, the lightweight client can traverse the blockchain by using the current blockchain header.

Part A:

Alice will send Bob the hash of the transaction in which Alice pays Bob. To check the membership of the transaction in the merkle tree Bob goes to the first block by using the header and checks for the membership in the current block's merkle tree. If it founds the membership Bob will stop otherwise bob will traverse to the previous block using the previous block's hash in the header and searches for membership.

Part B:

Bob needs to traverse the blocks linearly and in each block search the merkle tree in $log_2 n$ time. If there are k blocks before the head then Bob needs to traverse $k+1$ blocks.
$proof of size = (k+1) * log_2 n$
For $n = 256$ and $k = 8$
proof of size= $(8+1) * log_2(256) = 9 * 8 = 72$
Using SHA-256 hashing algorithm, hash size= $256 bits = 32$ bytes
proof of size= $72 * 32 = 2304$bytes.

Part C:

Bob will now traverse the blockchain to prove membership of the transaction by using the extra field each block header pointing to the last block which has a smaller hash value than the current block. So the number of block access will be k' which is less than k.

Expected proof of size = $klog_2 n32$ bytes

Best case size=$O(log_2 n)$ and worst case size=$O(klog_2 n)$

Blockchain Technology and Applications(CS731), Winter 2019
Indian Institute of Technology Kanpur
Homework Assignment Number 2

Student Name: Aneet Kumar Dutta
Roll Number: 18111401
Date: February 15, 2019

QUESTION
6

Part A:

To prevent immediate claiming of the jackpot an escrow transaction is created using MULTISIG transaction. So, the BitcoinLotto company creates a 3-3 MULTISIG transaction which requires three of three people to sign in order to redeem the transaction where the three parties are the public key in which the jackpot is present, the winner who have the corresponding private key and an intermediary. The transaction is signed with the public key so that the transaction can be validated and verified by others, the winner signs the transaction as soon as he gets to know the address but the intermediary signs only after a certain time period before which the jackpot can not be claimed.

Part B:

To roll forward the unclaimed jackpot from week n to week n+1, at the start of every week such that there is a transaction that sends the jackpot bitcoin from the week n address to the week n+1 address if unclaimed(check balance in the account, if not zero then send the bitcoin). This will also prevent the week n winner to double their winnings because the week n+1 transaction it will show that in the input section it received the bitcoins from the week n address. The week n's winner if now signs the transaction will not get any bitcoin as those bitcoins are transferred to the week n+1 address.