CS251: Cryptocurrencies and Blockchain Technologies **Fall 2019**

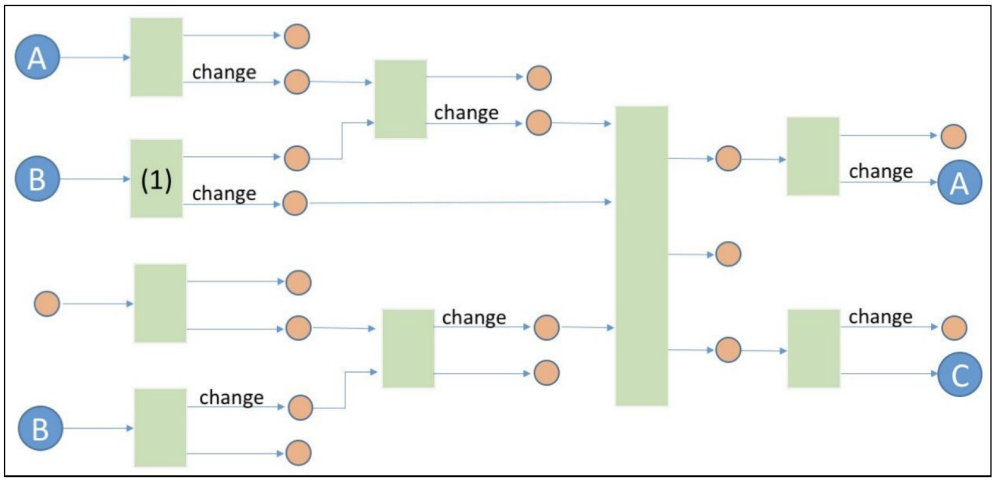
Assignment #3

Due: 11:59pm on Wed., **Dec. 4, 2019**
Submit via Gradescope (each answer on a separate page) code: **MG7EP3**

Problem 1. Idioms of use. Consider the transaction graph in the figure below – rectangles represent transactions, empty circles represent fresh addresses, and filled in circles represent addresses controlled by the named entity (i.e., A stands for Alice, B stands for Bob, and C stands for Carol). An edge labeled "change" means that the end node is the change address for that transaction, as identified by the heuristics discussed in class. Note that not every transaction has an identified change address.

a. Can an observer predict the identity of whoever was paid by Bob in the transaction marked (1)? Explain how or explain why not.

b. Can an observer predict the identity of whoever paid Carol? Explain how or explain why not.



Problem 2. Vulnerable 3-party payment channel. Three parties, A, B, and C, are constantly making pairwise payments and thus design a 3-party payment channel based on the revocable hashed timelock contracts we saw in class. At each step, A gets a revocable commitment that it can sign and submit with three outputs, one for B, one for C, and one that A can spend 48-hours after the transaction is mined, but either B or C can spend immediately given a hash preimage initially known only to A (and released by A to invalidate the transaction). Similarly, B and C each gets a corresponding commitment transaction with an output that either of the other two parties can claim given a hash preimage. Explain how two colluding parties may be able to steal funds from the third.

Problem 3. Briefly explain what is an Ethereum re-entrancy attack and why it can lead to loss of funds.

Problem 4. SNARKs. Let M be an $n \times n$ matrix over a field $\mathbb{F}$, and let $\lambda \in \mathbb{F}$. Both the prover and verifier know M and λ. The prover wants to convince the verifier that λ is an eigenvalue of M, that is, there exists a vector $\boldsymbol{v} \in \mathbb{F}^n$ such that $M\boldsymbol{v} = \lambda\boldsymbol{v}$. The verifier should be able to check the proof in constant time, independent of n.

a. Let $C_M(\lambda, \boldsymbol{v})$ be an arithmetic circuit that outputs $0 \in \mathbb{F}$ if and only if $M\boldsymbol{v} = \lambda\boldsymbol{v}$ (the inner-workings of C_M are not important). Design a linear PCP (P, V_1, V_2) for C_M, where V_1 issues only two linear queries. Recall that a linear PCP works as follows:

i. the prover P outputs the proof $\boldsymbol{\pi} := \boldsymbol{v}$,

ii. then V_1 issues two linear queries $\boldsymbol{u}, \boldsymbol{r}$ where $\boldsymbol{u}, \boldsymbol{r} \in \mathbb{F}^n$,

iii. finally, V_2 gets back the query responses $a_u := \langle \boldsymbol{u}, \boldsymbol{\pi} \rangle \in \mathbb{F}$ and $a_r := \langle \boldsymbol{r}, \boldsymbol{\pi} \rangle \in \mathbb{F}$, and outputs *yes* or *no*.

The verifier $V_2(\lambda, a_u, a_r)$ should work in constant time (independent of n).

- First, explain how V_1 chooses $\boldsymbol{u}, \boldsymbol{r}$ and how V_2 decides when to output *yes*.
- Then prove that a malicious prover cannot fool the verifier. That is, if $M\boldsymbol{v} = \lambda\boldsymbol{v} + \Delta$, where $\Delta \neq 0 \in \mathbb{F}^n$, then the verifier will accept the proof with probability at most $1/|\mathbb{F}|$ over the choice of $\boldsymbol{r} \in \mathbb{F}^n$.

Hint: V_1 will choose a random vector $\boldsymbol{r} \in \mathbb{F}^n$, and compute $\boldsymbol{u} := \boldsymbol{r}^\mathsf{T} M \in \mathbb{F}^n$. The first linear query from V_1 is $\boldsymbol{u} \in \mathbb{F}^n$, and the second linear query is $\boldsymbol{r} \in \mathbb{F}^n$. Explain how V_2 works.

b. In class we showed that a linear PCP implies a pre-processing SNARK (S, P, V) using linear-only encodings. Describe the resulting pre-processing SNARK for $C_M(\lambda, \boldsymbol{v})$ obtained from the linear PCP in part (a). In particular, describe how the algorithms $S(M)$, $P\big(S_P, (M, \lambda), \boldsymbol{v}\big)$, and $V(S_V, \lambda, \pi)$ work. These algorithm use the algorithms (*Gen*, *Enc*, *Verify*, *Add*, *QuadTest*) defined by the underlying linear-only encoding scheme. The proof π output by P contains only three elements, and is verified in constant time, no matter how big the matrix M is (!)

c. The linear PCP from part (a) is not zero-knowledge. Show how to enhance it so that is becomes honest-verifier zero-knowledge. To do so, expand the proof $\boldsymbol{\pi}$ to $\tilde{\boldsymbol{\pi}} := (s, \boldsymbol{v}) \in \mathbb{F}^{n+1}$, where the prover chooses s at random in $\mathbb{F}$. Then expand both queries from V_1 so that they become vectors in $\mathbb{F}^{n+1}$. Specifically, set $\tilde{\boldsymbol{u}} := (\lambda, \boldsymbol{u}) \in \mathbb{F}^{n+1}$ and $\tilde{\boldsymbol{r}} := (1, \boldsymbol{r}) \in \mathbb{F}^{n+1}$. Now, explain how V_2 works, and explain why the resulting protocol is honest-verifier zero-knowledge. It is best to do so by constructing a fast simulator $Sim(M, \lambda)$ that outputs a tuple $(\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{r}}, a_u, a_r)$ that is distributed as this tuple in the real protocol (by "fast" we mean faster than the time to compute an eigenvector $\boldsymbol{v}$ — *Sim*'s running time should be dominated by computing a single matrix-vector product).

Problem 5. In class we discussed the MakerDAO system, where DAI is intended to be a stable currency governed by MKR token holders. A brief description of the MakerDAO system is available here, and a more in-depth description is available here. It is recommended that you read one of these articles before answering the question.

Suppose that the MakerDAO pricing oracle (elected by MKR token holders) temporarily malfunctions and advertises that the price of ETH is \$1,000, when in reality it is only \$100.

a. How might an attacker exploit this situation to make money?

b. Assuming the error is corrected quickly enough not to destroy MakerDAO, who would bear the losses from such an attack and through what mechanism?