1 GHR Signature

Let’s take another look at the signature scheme due to Gennaro, Halevi and Rabin that we saw in class.

Recall the definition of a chameleon hash function. Let \( \text{KeyGen}(1^k) \) be a poly-time algorithm that outputs a key \( K \) and a trapdoor \( t \); let \( \text{Hash}_K \) be a poly-time algorithm that takes in a pair of values \( (x, r) \in D_K^2 \) (where \( D \) is some domain parameterized by the key \( K \)) and outputs some string \( z \in Z_K \). We say that \( (\text{KeyGen}, \text{Hash}) \) are a family of chameleon hash functions if:

**Collision-resistance** On input \( K \), no probabilistic poly-time adversary can find \( (x, r, x', r') \) such that \( \text{Hash}_K(x, r) = \text{Hash}_K(x', r') \) with non-negligible probability.

**Uniformity** For every \( x \in D_K \) and for a randomly chosen \( r \in D_K \), \( \text{Hash}_K(x, r) \) is a uniformly random element of \( Z_K \). In other words, \( \{ z \leftarrow Z_K : z = \text{Hash}_K(x, r) \} = \{ r \leftarrow D_K : z = \text{Hash}_K(x, r) \} \).

**Patching with a trapdoor** There is an algorithm \( \text{Patch} \) that, on input \( (K, t, x, r, x') \) outputs a random \( r' \) such that \( \text{Hash}_K(x, r) = \text{Hash}_K(x', r') \).

a. Let \( P_k \) be the set of primes of length \( k \). Suppose that we are given a chameleon hash function family where, for \( K \in \text{KeyGen}(1^k) \), \( Z_K \subseteq P_k \); in other words, we always interpret the value \( \text{Hash}_K(x, r) \) as a \( k \)-bit prime.

\( \text{KeyGen}(1^k) \): For the security parameter \( k \), pick an appropriate RSA modulus \( N = p_1 p_2 \) and a random \( u \in Z_n^* \). Let \( (K, t) \leftarrow \text{KeyGen}(1^k) \). Output \( \text{vk} = (N, u, K) \), \( \text{sk} = (p_1, p_2) \).

\( \text{Sign}(\text{vk}, \text{sk}, x) \) To sign \( x \in D_k \), choose random \( r \leftarrow D_K \), and compute \( z = \text{Hash}_K(x, r) \in P_k \). Use the factorization of \( N \) to find \( v \) such that \( v^2 = u \mod N \). Output \( \sigma = (r, v) \).

\( \text{Verify}(\text{vk}, x, (r, v)) \) Accept the signature of \( x \in D_K \) if \( v^{\text{Hash}_K(x, r)} = u \mod N \); else reject.

In class, we gave a reduction that, given an adversary that breaks the security of this signature scheme, breaks the Strong RSA assumption. However, our reduction
assumed that the adversary’s forgery \((x, (r, v))\) had the property that \(\text{Hash}_K(x, r) \neq \text{Hash}_K(x_i, r_i)\) for any previously queried \(x_i, \sigma_i = (r_i, v_i)\). Let \(B\) be the reduction that you have seen in class.

Prove that this signature scheme is secure if the underlying chameleon hash function is secure and the Strong RSA assumption holds. In your proof, you don’t need to redo the reduction \(B\); you may just use it as a subroutine in your reduction.

b. This signature scheme works to sign messages that are elements of \(D_K\). Using a regular collision-resistant hash function with the appropriate domain and range, show how to extend this construction to messages that are arbitrary strings. What are the domain and range of the collision-resistant hash function? Give a brief sketch of the proof of security of the resulting construction.

c. Let us instantiate the chameleon hash function with the discrete-logarithm based construction presented in class. Namely, on input \(1^k\), \(\text{KeyGen}\) outputs a group \(G\) with a generator \(g\) of prime order \(q\), and another generator \(h = g^\alpha\) for a random \(\alpha\), and sets \(K = (G, q, g, h)\), and \(t = \alpha\). Then \(D_K = \mathbb{Z}_q\), \(Z_K = G\), and \(\text{Hash}_K(x, r) = g^x h^r\).

But for the construction to work, we require that elements of \(Z_K\) be interpreted as \(k\)-bit primes. Explain how this can be done (drawing on what we did in class).

d. Prove that the discrete-logarithm based construction of a chameleon hash function is secure under the discrete logarithm assumption. (You may draw on what we did in class.)

2 Digital Signatures

Prove that the existence of secure digital signature schemes implies the existence of one-way functions.

3 Zero Knowledge Proof

In class we saw a physical and a cryptographic protocol for a zero-knowledge proof for the 3-colorability problem.

Now let’s design a proof for a different NP-complete problem:

**Definition 1 (Vertex Cover Problem)** Given a graph \(G = (V, E)\) and an integer \(k\), does there exist a size \(k\) vertex cover, i.e. a size \(k\) subset \(C \subseteq V\) such that for all edges \(e \in E\) at least one endpoint is in \(C\)?
Suppose Alice and Bob share a graph $G$. Alice claims that $G$ has a size $k$ vertex cover. Alice will attempt to prove it as follows (see figure below for example):

- Alice sends Bob out of the room.

- On one side of the table, Alice lists the graph’s vertices in random order. We assume that each vertex of the original graph is labeled with unique number from 1 to $|V|$. Alice now labels each vertex with this number in green and covers the label with a paper cup. She then gives each vertex a new random label (written in blue and not covered by paper cups).

- On the other side of the table, Alice draws a bunch of parallel lines, one for each edge. She labels the endpoints using the new (blue) vertex labelling. Finally, she covers these labels with paper cups.

- On the vertex list, next to each vertex that is in the vertex cover set $C$, Alice writes a “C”. Next to all other vertices, Alice writes a “Not C”. She then covers all the “C”’s and “Not C”’s with paper cups.

- On the edge list, next to each edge, Alice draws an arrow that points to an endpoint which is in the cover. She covers this arrow with a piece of paper.

- Alice calls Bob back into the room.
How can Bob verify this proof? We’re going to divide the solution into several small steps.

a. Suppose Bob only wanted to check that the graph is represented correctly (i.e. that the graph is the same one, $G$, that he and Alice have agreed on.) How can he confirm this without learning any other information?

   Hint: Recall that the graph isomorphism problem (given graphs $G_1$, $G_2$, determine whether there is a relabelling of the vertices of $G_1$ which make it identical to $G_2$) is considered hard.

b. Now suppose Bob only wants to check that the cover has the appropriate size (the agreed upon $k$). How can he confirm this without learning any other information?

c. Finally, suppose Bob only wants to check that each edge is covered. How can Bob do this without learning any other information (he is allowed to examine only one edge in each round)? With what probability is he guaranteed to catch Alice if she does not know a $k$-cover?

d. What should Bob’s overall strategy for verifying Alice’s vertex cover proof be, and what is the minimum probability that Alice will be caught if she cheats? (Hint: the strategy will probably have to be randomized.)

e. Assuming Alice is willing to repeat this process as many times as necessary, how many times should Bob run this algorithm so that if Alice cheats she will be caught with probability at least $O(1)$?