Projet MiRitH

Maxime Coute Jules Magois

30 janvier 2025

1 Explication générale de la signature

1.1 Problème MinRank

1.2 Authentification zero-knowledge utilisant MPC-in-the-head

1.3 Fiat-Shamir pour obtenir une signature

1.4 Sécurité

Soient :

1. N le nombre total de parties,
2. τ le nombre de tours effectués.

La probabilité qu'un faux signataire arrive à une signature correcte est proportionnelle à $N^{-\tau}$.

La taille d'une signature (en nombre de bits) est proportionnelle à τ.

On peut donc faire différents choix de N et τ en fonction de nos besoins :

— N petit et τ élevé pour générer rapidement une signature de grande taille,

— N grand et τ faible pour générer une signature courte.

2 Proposition de structure du code

1. un fichier `field_arithmetics.c` pour l'addition et la multiplication sur $GF(16)$,
2. un fichier `constants.c` contenant les constantes utiles pour tous les autres fichiers :
   — la définition du corps `GF_16`
   — quelques paramètres de signature standards
   — les tables d'addition et de multiplication dans `GF_16`
3. un fichier `matrix.c` pour gérer les opérations sur les matrices :
   — allocation de mémoire
   — libération de mémoire
   — addition de deux (ou une liste de) matrices

   — multiplication de deux matrices
4. un fichier `key_generation.c` pour générer la clé,
5. un fichier `party.c` qui implémente les calculs de chaque partie,
6. un fichier `main.c` qui implémente la signature.

3 Bibliothèques utilisées

1. `gmp` pour la génération de nombres aléatoires,
2. `openssl` pour l'utilisation du hash *Keccak*.