Projet Cryptographie : Génération de clés RSA

Laveus Mick-Wanderly

20 avril 2025

Introduction

Ce projet vise à générer une clé RSA en utilisant des sources d'entropie issues du matériel, telles que la mémoire SRAM au démarrage ou l'ADC. L'objectif est de réaliser un système embarqué sécurisé basé sur des données aléatoires naturelles.

Organisation du code

- `entropie.c` : simule la collecte d'entropie (SRAM + ADC).
- `gener_alea.c` : applique un hachage (Blake2s) sur les données d'entropie.
- `primalite.c` : implémente le test de primalité de Miller-Rabin.
- `main.c` : pilote la séquence avec une interface APDU.

Méthode de génération

L'entropie est collectée, combinée par XOR, puis hachée avec Blake2s :

```
collect_entropy(entropy, ENTROPY_LEN);
blake2s(out, outlen, entropy, ENTROPY_LEN, NULL, 0);
```

Le candidat généré est transformé en entier 64 bits pour être testé :

```
uint64_t number = candidate_to_uint64(candidate);
```

Test de primalité

Le test de Miller-Rabin est utilisé avec 5 tours :

```
if (miller_rabin(number, 5)) {
    printf("Premier probable\n");
}
```

Commande APDU

Un switch-case traite les instructions comme sur carte à puce :

```
switch (ins) {
    case 0xF5: // G n ration RSA
    case 0xB7: // NIST requis
}
```

Conclusion

Cette implémentation fournit une base solide pour la génération sécurisée de clés RSA avec des contraintes matérielles. Le code peut être intégré dans un firmware compilé via `avr-gcc` ou testé localement sur PC.