FrosTON: Using FROST for Off-Chain Signing and On-Chain Verification on TON Blockchain

Alireza Tabatabaeian

Introduction

In this project, we utilized the **FROST (Flexible Round-Optimized Schnorr Threshold)** protocol for threshold signatures to enhance the security and efficiency of signature generation and verification on the TON (The Open Network) Blockchain. We leveraged Python bindings to frost-dalek, a library implementing FROST, and extended the wallet-v3 standard to support the new signature scheme. The implementation allowed for the off-chain signing of data and efficient on-chain verification using new TVM (TON Virtual Machine) opcodes **(with less than 7k gas units).**

Key Generation and Off-Chain Signing

The key generation and off-chain signing process involved the following steps:

1. **Threshold Setup**: We performed a secure threshold setup phase where the participants generated their private key shares. This ensured that the private key was divided among the authorized parties while maintaining the desired threshold level.
2. **Python Bindings to frost-dalek**: We utilized the Python bindings to frost-dalek, a Rust library implementing the FROST protocol. These bindings provided us with the necessary functionality to perform key generation, threshold signing, and signature verification using the FROST protocol.
3. **Key Generation Rounds**: We executed the key generation rounds using frost-dalek off-chain. This step involved the collaborative computation of the public key from the private key shares held by the participants. The resulting public key was used for subsequent signing and verification.

4. **Off-Chain Signature Generation**: Leveraging the FROST protocol, we performed off-chain signature generation using frost-dalek. The participants combined their private key shares to produce a valid signature for the given data. This process ensured that the private keys remained secure and were not exposed during the signing operation.

On-Chain Verification and Efficient TVM Opcodes

The on-chain verification process and the usage of custom TVM opcodes optimized for efficiency involved the following steps:

1. **Integration with TON Blockchain**: We integrated the FROST-based signature scheme into the TON Blockchain by extending the wallet-v3 standard. This allowed us to leverage the existing infrastructure and functionality provided by the TON Blockchain. We also wrote a test contract to ensure correct behavior. **The enhanced wallet-v3 empowers users with the capability to send funds securely and efficiently. Additionally, users can also modify the group key, subject to the agreement terms.**
2. **Custom TVM Opcodes**: We leveraged new custom TVM opcodes to enhance the efficiency of signature verification. These opcodes included **HASHEXT_SHA256, HASHEXT_SHA512, RIST255_FROMHASH, RIST255_SUB, RIST255_MUL, RIST255_MULBASE, RIST255_ADD, RIST255_PUSHL, and GASCONSUMED**. These opcodes were used to optimize the signature verification process and minimize gas consumption.
3. **On-Chain Signature Verification**: We implemented the on-chain signature verification logic using the FROST-based signature scheme and the new TVM opcodes. This process involved verifying the signature against the data using the public key derived from the key generation rounds. The verification was performed efficiently, requiring only 6667 gas, which contributed to significant cost savings and is below **the 10k gas limit** for external messages with a good margin.

Conclusion

In conclusion, we successfully utilized the FROST protocol with Python bindings to frost-dalek, performed key generation and off-chain signing, and implemented efficient on-chain verification using new TVM opcodes on the TON Blockchain. The integration

of the FROST-based signature scheme and the custom opcodes provided enhanced security, privacy, and efficiency in signature generation and verification. The refactored wallet-v3 standard and the introduced script can be considered as a toolkit that can be easily utilized in other projects, facilitating the development of efficient and secure multi-sig wallets and cryptographic operations on the TON Blockchain.

Codes

Off-chain DKG and Signing scripts (bindings to frost-dalek)

Contracts (test and wallet-v3)

References

Flexible Round-Optimized Schnorr Threshold Signatures

https://github.com/isislovecruft/frost-dalek

https://github.com/devos50/frost-dalek

https://github.com/devos50/frost-python