LaurentianUniversity
UniversitéLaurentienne

CPSC5207EL-62: Applied Cryptography

Spring 2025

Project Proposal

Title: Implementation of ChaCha20 Algorithm

Group - 9

Student information

Name	ID
Md Hasan Shahriar	0445469
Fatin Noor Anik	0452573
Hossain Md Shafayet	0446094
Mohammad Omar Faruk	0442538
Md Hasan Khondoker	0439579

Abstract

This report proposes a detailed implementation of the ChaCha20 stream cipher using Python to support the exploration of modern symmetric cryptographic techniques. ChaCha20, recognized for its security and software efficiency, serves as a viable alternative to traditional block ciphers such as AES and is employed in widely used protocols including TLS 1.3 and OpenSSH. The project involves developing the cipher from first principles, allowing user-defined inputs for the key, nonce, and counter, and demonstrating the encryption and decryption workflows. Validation is conducted using official test vectors to ensure algorithmic correctness, with performance assessed in general-purpose computing environments. The solution incorporates a Streamlit-based interface, Docker containerization for platform independence, and comprehensive documentation. This work contributes to both cryptographic education and practical application, offering an open-source, reusable tool that facilitates secure data handling and promotes foundational understanding of stream cipher design.

Introduction

In today's digital world, data security is a fundamental requirement for communication and information systems. Symmetric encryption algorithms are a crucial component of secure systems. ChaCha20 is a modern, safe, and efficient stream cipher widely used in real-world applications such as TLS (Transport Layer Security), SSH, and VPNs. This project focuses on implementing ChaCha20 in Python to understand its internal workings and practical applications.

Motivation for Choosing ChaCha20

ChaCha20 is an improved version of the Salsa20 cipher, created by Daniel J. Bernstein. It is known for its performance and security, even on systems without dedicated hardware acceleration. Unlike AES, which relies on complex operations and benefits greatly from hardware support, ChaCha20 is designed to be fast and secure in purely software-based environments.

Key reasons for choosing ChaCha20:

- **Security**: Uses a 256-bit key and a 96-bit nonce, with 20 rounds of operations to ensure robust encryption.
- **Speed**: Optimized for high performance on general-purpose CPUs.
- **Simplicity**: The algorithm is simpler to understand and implement compared to block ciphers like AES.

- **Widespread Use**: Adopted in protocols like TLS 1.3, WireGuard VPN, and OpenSSH.

Objectives

- Implement the ChaCha20 cipher from scratch in Python.
- Provide functionality for user input of key parameters such as key, nonce, and counter.
- Demonstrate the encryption and decryption process of plaintext.
- Analyze the correctness and efficiency of the algorithm.

Benefits of the Implementation

- **Educational Insight**: Helps understand how stream ciphers function internally.
- **Hands-On Learning**: Develops Python programming and cryptographic algorithm design skills.
- **Security Awareness**: Demonstrates practical applications of symmetric encryption.
- **Reusable Code:** Can be extended into real-world use cases like secure messaging or file encryption.

Methodology

- **Research Phase:** Study the algorithm specification and understand its structure (quarter-rounds, state matrix, key setup, nonce handling).
- **Implementation Phase:**
  - Build core functions such as the quarter-round and block functions.
  - Generate keystream blocks and apply XOR with plaintext.
  - Build functions for both encryption and decryption.

- **Interface Design:** Provide a basic command-line interface or simple GUI to input plaintext, keys, and nonces.
- **Testing Phase:** Validate the implementation using official ChaCha20 test vectors.

Tools and Technologies

- **Language**: Python 3.12 (Latest and secure version)
- **Libraries**: Streamlit for web dashboarding, python-cryptography for implementing the functions.
- **Containerization**: Docker
- **Platform**: OS-independent, can run or host anywhere (via Docker)

Expected Deliverables

- Fully functional ChaCha20 encryption and decryption tool written in Python.
- Public GitHub repository
- Docker image for platform-independent deployment.
- Project report detailing:
    - The theoretical background of ChaCha20
    - Implementation steps
    - Test results with examples

- Challenges and limitations
- A presentation explaining the algorithm and project outcomes
- Sample inputs and outputs demonstrating encryption and decryption processes

Conclusion

ChaCha20 is a powerful example of modern cryptographic design that balances performance and security. By implementing it from scratch in Python, this project will offer valuable insights into cryptographic development and provide a practical tool that reflects real-world encryption applications.