FibroHash: A Cryptographically Secure Password Generation Framework for System Administration

Spyros Lefkaditis
Independent Researcher
ORCID: 0009-0000-8432-4667

November 6, 2025

Abstract

FibroHash is a cryptographically secure password generation framework designed for system administrators and security professionals. The framework implements standard PBKDF2-HMAC-SHA256 key derivation combined with Python's cryptographically secure random number generator (CSPRNG) to produce high-entropy passwords with comprehensive security analysis capabilities.

The framework provides secure password generation using established cryptographic primitives, proper salt handling, and quality assessment tools. FibroHash operates entirely offline using only Python's standard library, ensuring no external dependencies or network communications that could compromise security.

Keywords: Python, cryptography, password generation, security, system administration, PBKDF2, entropy analysis

1 Introduction

System administrators and security professionals require password generation tools that provide both high entropy and reproducible security analysis. Existing solutions often suffer from predictable patterns, insufficient entropy, or lack proper cryptographic foundations. Recent research on password behavior through persuasion techniques [1] demonstrates the importance of user-centered approaches to secure password creation. Many tools also require external dependencies or network connectivity, introducing potential security vulnerabilities, while contemporary studies on password manager adoption [2] reveal ongoing challenges in organizational credential management practices. Recent analysis of password hashing methods using CSPRNG and PBKDF2 [3] demonstrates the critical importance of implementing proper cryptographic foundations in password generation tools.

FibroHash addresses these limitations by providing:

1. **Cryptographic Security**: Implementation of PBKDF2-HMAC-SHA256 with configurable iterations (1,000-10,000) using Python's secrets module for cryptographically secure randomness

2. **Entropy Analysis**: Built-in entropy analysis tools providing Shannon entropy calculations and character distribution analysis

3. **Quality Assessment**: Password quality validation with detailed security scoring and pattern detection

4. **Research Reproducibility**: Comprehensive test suite enabling security researchers to validate and extend the methodology

The framework has been designed with system administrators in mind, providing both command-line interfaces for operational use and programmatic APIs for integration into larger security frameworks.

2 Research Contribution and Methodology

FibroHash implements a secure password generation approach combining PBKDF2-HMAC-SHA256 key derivation with HMAC-based entropy generation. The framework uses multiple entropy sources including multi-round HMAC sequence generation and Python's cryptographically secure random number generator (secrets module) to provide reproducible security analysis while maintaining cryptographic security.

2.1 Cryptographic Architecture

The password generation process follows a multi-stage cryptographic pipeline:

1. **Input Processing**: User phrases undergo validation and sanitization to prevent injection attacks
2. **Key Derivation**: PBKDF2-HMAC-SHA256 transforms user input and cryptographic salt into derived keys
3. **Entropy Generation**: Multiple entropy sources including HMAC-based mathematical sequence generation and Python's secrets module for cryptographically secure randomness
4. **Character Encoding**: Secure base conversion using extended character sets with 90+ characters
5. **Quality Assurance**: Automated validation of character diversity and entropy levels

2.2 Security Analysis

The framework provides comprehensive security analysis including entropy calculations based on character set size and password length. Security analysis includes:

- **Timing Attack Resistance**: Consistent operation times regardless of input characteristics
- **Salt Uniqueness**: Cryptographically secure salt generation for each password instance
- **Pattern Avoidance**: Detection and mitigation of sequential, keyboard, and dictionary patterns

2.3 Validation Framework

FibroHash includes a comprehensive validation framework enabling reproducible security research:

Listing 1: Security Analysis Example

```
from main import generate_password
from security_utils import generate_security_report

# Generate cryptographically secure password
password = generate_password("research phrase", 32, "maximum")

# Perform comprehensive security analysis
```

```
report = generate_security_report(password)
print(f"Entropy: {report['audit_results']['entropy_analysis']['
    theoretical_entropy']} bits")
print(f"Security Score: {report['audit_results']['security_score']}/100")
```

3 Examples

3.1 Basic Usage

Listing 2: Basic Password Generation

```
from main import generate_password

# Generate password with default settings (32 chars, high security)
password = generate_password("secure research phrase")

# Generate with custom parameters
password = generate_password("phrase", password_length=24, security_level="
    maximum")
```

3.2 Security Analysis

Listing 3: Advanced Security Analysis

```
from security_utils import SecurityAuditor, SecurePasswordValidator

auditor = SecurityAuditor()
validator = SecurePasswordValidator()

# Comprehensive security audit
audit_results = auditor.audit_password_quality(password)

# Policy validation
is_valid, violations = validator.validate(password)
```

3.3 Configuration and Testing

Listing 4: Setup and Testing Commands

```
# Setup and configuration
./setup.sh

# Run comprehensive security test suite
python3 test.py

# Interactive password generation
./init.sh
```

4 Impact and Applications

FibroHash has applications in:

- **System Administration**: Secure password generation for server and service accounts
- **Security Research**: Reproducible password security analysis and entropy validation

- **Quality Assessment**: Automated password quality validation and security scoring
- **Educational Use**: Teaching cryptographic principles and password security

The framework's emphasis on reproducible security analysis makes it particularly valuable for security researchers studying password generation algorithms and entropy analysis techniques.

5 Acknowledgements

The author acknowledges the Python cryptography community for establishing secure cryptographic practices and the broader cybersecurity research community for advancing password security methodologies.

References

[1] Rizu Paudel and Mahdi Nasrullah Al-Ameen. Priming through persuasion: Towards secure password behavior. *Proceedings of the ACM on Human-Computer Interaction*, 8(CSCW1):1–27, 2024. doi:10.1145/3637387.

[2] Xiaoguang Tian. Unraveling the dynamics of password manager adoption: a deeper dive into critical factors. *Information and Computer Security*, 33(1):117–139, 2025. doi:10.1108/ICS-09-2023-0156.

[3] Nada Abdul Aziz Mustafa. Analysis attackers' methods with hashing secure password using csprng and pbkdf2. *Wasit Journal of Engineering Sciences*, 12(2):60–70, 2024. doi:10.31185/ejuow.Vol12.Iss2.502.