**Documentation of the Hash Cracking Tool  
  
Prepared by:**

**CH: Dinesh (22bce8252)**

**V. Srujan (22bce8087)**

**G. Nikitha (22bce7493)**

**Abstract**

The Hash Cracking Tool is a web-based application designed to decrypt various types of cryptographic hashes. It supports multiple hash formats including MD5, SHA1, SHA256, SHA384, and SHA512. The tool utilizes several online services and databases to attempt hash decryption, providing a user-friendly interface for both single hash cracking and batch processing. Additionally, it offers password strengthening suggestions to help users create more secure passwords.

**Introduction**

Cryptographic hashing is widely used for securing passwords and verifying data integrity. However, weak hashing implementations can be vulnerable to various attacks. This tool demonstrates how hashes can be cracked using online services and databases, highlighting the importance of using strong passwords and proper security measures.

The Hash Cracking Tool serves both educational and practical purposes:

* It helps security professionals test the strength of password hashing implementations
* It assists in recovering forgotten passwords when only the hash is available
* It demonstrates the vulnerability of commonly used hash algorithms
* It educates users about password security by suggesting stronger alternatives

**Basic Information About Hash Cracking**

Cryptographic hash functions convert data of arbitrary size into fixed-size values. These functions are designed to be one-way, meaning it should be computationally infeasible to reverse the process. However, several techniques can be used to "crack" hashes:

1. **Dictionary Attacks**: Testing hashes of common passwords against the target hash
2. **Rainbow Tables**: Pre-computed tables of hash-to-password mappings
3. **Online Services**: Databases containing billions of pre-computed hashes
4. **Brute Force**: Systematically checking all possible combinations

The effectiveness of these methods depends on the hash algorithm used, the complexity of the original password, and whether techniques like salting were employed.

**Existing or Related Tools**

Several hash cracking tools exist in the market:

* Hashcat: A powerful CPU/GPU-based password recovery tool
* John the Ripper: A free and open-source password security auditing tool
* CrackStation: An online service with a large pre-computed hash database
* MD5Decrypt: A web service specifically for MD5 hash cracking

**Drawbacks of Existing Tools**

While existing tools are powerful, they have several limitations:

* Command-line interfaces that are intimidating for non-technical users
* Complex setup requirements including specific hardware dependencies
* Limited integration with multiple online services
* Lack of educational components about password security
* Often require local installation of large software packages

**Proposed Tool**

Our Hash Cracking Tool addresses these limitations by providing:

* A simple, web-based interface accessible from any browser
* Integration with multiple online hash cracking services
* Support for various hash types (MD5, SHA1, SHA256, SHA384, SHA512)
* Batch processing capabilities for multiple hashes
* Educational password strengthening suggestions
* No installation required - runs entirely in the browser

**What the Tool is Used For**

The Hash Cracking Tool can be used for:

1. Recovering forgotten passwords when only the hash is available
2. Testing the security of password storage implementations
3. Demonstrating the vulnerability of weak hashing algorithms
4. Educational purposes to understand hash cracking techniques
5. Generating stronger password alternatives from existing passwords

**Algorithm/Pseudocode**

The core algorithm of our tool is the hash cracking function that attempts to decrypt a hash using multiple services:

python

**def** crack(hashvalue):

"""Determines hash type and attempts to crack it."""

length = len(hashvalue)

**if** length **in** hash\_methods:

hashtype, apis = hash\_methods[length]

**for** api **in** apis:

result = api(hashvalue, hashtype)

**if** result:

**return** result

**return** False

**return** False

This algorithm works by:

1. Determining the hash type based on its length
2. Selecting appropriate API services for that hash type
3. Trying each service sequentially until a match is found
4. Returning the cracked value or False if unsuccessful

The tool uses multiple services (alpha, beta, gamma, theta) to maximize the chance of finding a match:

python

hash\_methods = {

32: ['md5', [alpha, beta, gamma, theta]],

40: ['sha1', [alpha, beta, theta]],

64: ['sha256', [alpha, beta, theta]],

96: ['sha384', [alpha, beta, theta]],

128: ['sha512', [alpha, beta, theta]]

}

**Clear Explanation of the Algorithm**

The hash cracking algorithm follows these steps:

1. **Hash Type Identification**: The tool identifies the hash type based on its length (e.g., 32 characters for MD5, 40 for SHA1).
2. **Service Selection**: For each hash type, the tool has a predefined list of services that can potentially crack that hash.
3. **Sequential Service Querying**: The tool queries each service in sequence:
   * alpha: Uses cmd5.org for hash cracking
   * beta: Uses a WebSocket connection to md5hashing.net
   * gamma: Uses nitrxgen.net specifically for MD5 hashes
   * theta: Uses md5decrypt.net's API with authentication
4. **Result Processing**: If any service returns a valid result, the algorithm immediately returns that value. If all services fail, it returns False.
5. **Batch Processing**: For multiple hashes, the tool uses concurrent processing with thread pools to improve efficiency.
6. **Password Strengthening**: For the password suggestion feature, the tool applies various transformations to create stronger alternatives.

**System Requirement Specification (Minimum)**

**Server Requirements:**

* Python 3.6 or higher
* Flask web framework
* Requests library for HTTP requests
* Concurrent.futures for threading support

**Client Requirements:**

* Modern web browser (Chrome, Firefox, Safari, Edge)
* JavaScript enabled
* Internet connection for accessing online hash cracking services

**Recommended Hardware:**

* Processor: Dual-core 2GHz or higher
* RAM: 4GB or higher
* Storage: 100MB free space
* Network: Broadband internet connection

**Results and Discussions**

The Hash Cracking Tool has been tested with various hash types and demonstrates effective performance in cracking common passwords. The web interface provides an intuitive way to submit hashes and view results.

**Single Hash Cracking:**  
When a single hash is submitted, the tool attempts to crack it using multiple services and displays the result immediately. Testing shows that common passwords hashed with MD5 are typically cracked within seconds, while more complex passwords or stronger hash algorithms may take longer or fail to be cracked.

**Batch Processing:**  
The tool efficiently handles files containing multiple hashes, extracting and attempting to crack each one. Testing with a file of 100 common password hashes showed a success rate of approximately 70% for MD5 hashes, demonstrating the tool's effectiveness.

**Password Strengthening:**  
The password suggestion feature successfully transforms weak passwords into stronger alternatives by applying various techniques such as character substitution, adding special characters, and mixing case.

**Conclusion**

The Hash Cracking Tool provides a user-friendly, web-based interface for cracking various types of cryptographic hashes. By integrating multiple online services, it maximizes the chance of successful decryption while educating users about password security.

The tool demonstrates that many commonly used passwords can be easily cracked, highlighting the importance of using strong, unique passwords and proper security measures such as salting and using modern hashing algorithms.

Future improvements could include support for additional hash types, integration with more services, and enhanced password strength analysis. The tool serves as both a practical utility for recovering forgotten passwords and an educational resource for understanding hash security.

**References**

* Flask Documentation: <https://flask.palletsprojects.com/>
* MD5decrypt.net API: <https://md5decrypt.net/>
* Nitrxgen MD5 Database: <https://www.nitrxgen.net/>
* CMD5.org: <https://www.cmd5.org/>
* OWASP Password Security Guidelines: <https://owasp.org/www-community/controls/Secure_Password_Storage>
* GitHub Python Requests Library: <https://github.com/psf/requests>