HashCracker Pro - Detailed Documentation Report

1. Project Overview

Project Name: HashCracker Pro

Author: Ashish Negi

Language: Python 3

Purpose:

HashCracker Pro is a versatile Python-based hash cracking tool designed to demonstrate password vulnerabilities. It supports multiple hashing algorithms and attack modes. Intended for educational and security testing purposes.

Key Features:

- Multi-algorithm support (MD5, SHA variants, Blake2)
- Multiple attack modes (dictionary, rule-based, brute-force, mask, combo)
- Verbose progress tracking
- Logging results in structured JSON format
- Automatic hash algorithm detection
- Force mode to attempt cracking even on invalid hashes

2. Supported Hash Algorithms

Algorithm	Hash Length (Hex)	Notes
MD5	32	Common, fast, insecure
SHA-1	40	Legacy, still used
SHA-256	64	Strong, widely used
SHA-512	128	Very strong, slower
SHA3-256	64	SHA3 variant, strong
SHA3-512	128	SHA3 variant, very strong
Blake2b	128	Fast and strong
Blake2s	64	Lightweight variant

3. Code Structure & Functionality

Utilities:

- hash_word(word, algo): Returns the hashed value of a word with the specified algorithm.

- validate_hash(hash_value, algo): Checks if the hash is valid.
- detect_algo(hash_value): Detects algorithm based on hash length.

Attack Modes:
1. Dictionary Mode: Uses a wordlist to find the matching hash.
2. Rule Mode: Modifies words using rules (append123, reverse, capitalize, replace).
3. Brute-Force Mode: Generates all possible combinations of characters from a charset.
4. Mask Mode: Uses a pattern to reduce search space (?l, ?u, ?d, ?s).
5. Combo Mode: Combines two wordlists in all possible ways to find the hash.

Logging & UI:
- Real-time status table using rich
- JSON logs in logs/cracked.json with timestamp

4. Command-Line Usage

General Syntax:
python3 hash_cracker.py --mode <attack_mode> --hash <hash_value> [options]

Examples:
1. Dictionary Attack:
python3 hash_cracker.py --mode dictionary --hash <hash> --algo md5 --wordlist /usr/share/wordlists/rockyou.txt

2. Rule-Based Attack:
python3 hash_cracker.py --mode rule --hash <hash> --algo md5 --wordlist /usr/share/wordlists/rockyou.txt --rules custom_rules.txt

3. Brute-Force Attack:
python3 hash_cracker.py --mode brute --hash <hash> --algo md5 --charset abc123 --maxlen 6

4. Mask Attack:
python3 hash_cracker.py --mode mask --hash <hash> --algo md5 --mask "?u?l?l?l?d?d"

5. Combo Attack:

python3 hash_cracker.py --mode combo --hash <hash> --algo md5 --wordlist wordlist1.txt --wordlist2 wordlist2.txt

5. Development Notes

Development Approach:

- Modularized into utilities, attack modes, logging, UI
- Flexible with custom wordlists, rules, masks, charsets
- Progress bar with tqdm, status table with rich

Challenges:

- Efficient brute-force generation
- Handling large wordlists
- Clean real-time console UI

Future Enhancements:

- Multiprocessing for brute-force
- Automatic integration with popular wordlists
- Hybrid attack modes
- GPU acceleration

6. Example Run

Dictionary Attack Example:

Command:

python3 hash_cracker.py --mode dictionary --hash 6baf0190a5881b48d8ed9fbf3f34e5e7 --algo md5 --wordlist /usr/share/wordlists/rockyou.txt

Output Table:

Mode | Algorithm | Hash | Attempts | Found

Dictionary | MD5 | 6baf0190a5881b48d8ed9fbf3f34e5e7 | 4321 | mypass123

JSON Log Entry:

{

```
  "result": "mypass123",
  "stats": {
    "mode": "dictionary",
    "algo": "md5",
    "hash": "6baf0190a5881b48d8ed9fbf3f34e5e7",
    "attempts": 4321,
    "found": "mypass123",
    "timestamp": "2025-09-22T12:45:00"
  },
  "logged_at": "2025-09-22T12:45:00Z"
}
```


7. Conclusion

HashCracker Pro is a complete Python-based hash cracking framework with multiple attack modes, logging, and algorithm support. The modular design allows easy extensions, custom rules, and wordlists. Ideal for educational purposes and penetration testing exercises.