Eth Hacker Hash Detector

Student Name: Yara Samed Ali Darawsheh

University: Irbid National University

Faculty/Department: Cybersecurity

Course: Field Training / Cybersecurity

Student ID: 202220859

Date: November 2025

Abstract

This project is a Python command-line tool designed to detect common hash types (MD5, SHA1, SHA256, SHA384, SHA512) and to generate MD5 hashes from plaintext.
The tool validates inputs, handles errors gracefully, and offers both interactive and direct argument modes.
It is intended for educational and ethical purposes in the context of cybersecurity and field training.

Introduction

Hash functions are widely used in cybersecurity for password storage, integrity checking, and digital signatures.
This project aims to create a simple yet effective tool that can detect the type of a given hash string and generate MD5 hashes.
The tool is implemented in Python and runs on Kali Linux, offering both interactive and direct detection modes for ease of use.

Methodology

- The tool accepts a string input (interactive or CLI argument).
- It validates if the input is a valid hexadecimal string.
- Based on the length of the string, it infers possible hash types:
  - MD5: 32 hex characters
  - SHA1: 40 hex characters
  - SHA256: 64 hex characters
  - SHA384: 96 hex characters
  - SHA512: 128 hex characters
- If input is invalid or length does not match any type, a clear message is returned.
- The MD5 generation uses UTF-8 encoding to ensure deterministic output.
- CLI design allows users to either run interactively or pass hashes directly.

Results:

- Example 1: Detecting MD5

  Input: d41d8cd98f00b204e9800998ecf8427e

  Output: MD5

- Example 2: Detecting SHA256

  Input: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855

  Output: SHA256

- Example 3: Generating MD5

  Input: "Hello"

  Output: 8b1a9953c4611296a827abf8c47804d7

- Example 4: Invalid input

  Input: xyz123

  Output: Unknown hash type. Please enter a valid hex string.

Limitations:

- Detection relies only on string length and hex validation.

- Cannot detect hash type if the hash length is incorrect or contains non-hex characters.

- Multiple hash types may theoretically match in rare edge cases.

- Future improvements could include pattern-based or checksum-based hash detection.

Conclusion:

This project successfully demonstrates a Python tool capable of detecting common hash types and generating MD5 hashes.

It is robust, user-friendly, and adheres to ethical standards. The tool provides a practical example of hash analysis in cybersecurity.

References:

1. Python Documentation - hashlib module: https://docs.python.org/3/library/hashlib.html

2. Stallings, W. Data and Computer Communications, 10th Edition