JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

UNMASKING HASHED PASSWORDS FOR CYBERSECURITY

R Suhasini[1], Reddyvari Venkateswara Reddy[2],

[3]CH Bharath Kumar, [4]G Rajesh Goud, [5]G Kavya Sri

[1]Assistant Professor, Department of CSE (Cyber Security)
[2]Associate Professor, Department of CSE (Cyber Security),
[3]Student, [4]Student, [5]Student
[1,2,3]Department of CSE (Cyber Security),
[1,2,3,4,5]CMR College of Engineering & Technology, Hyderabad, India

Abstract: The following advanced hash password cracking techniques is essential in an ever-evolving threat landscape in cybersecurity This paper presents a sophisticated tool designed for users to find and decrypt hash passwords with state-of-the-art methods implementation, and the delicate balance between due diligence such as password recovery comes with ethical considerations in cybersecurity practices Carry a tool that includes available hashing algorithms such as MD5, SHA, bcrypt, NTLM and various password recovery scenarios combine Provides a primed versatile platform to address Ethical considerations loom large in this study, which recognizes the duality of password cracking with the potential of both creation and misuse Through careful use and strict adherence to ethical principles, the tool contains a responsible approach to password and cybersecurity themes etc. It uses libraries to create robust and efficient solutions. Ethics are emphasized, where the tool dynamically identifies hashes, uses special techniques to fit each algorithm, and uses parallel processing to increase performance Each hashing algorithm is thoroughly tested, changing the tool in encryption techniques different faces It also shows versatility.

IndexTerms – Cybersecurity, Ethical Hacking, Hashing, MD5, SHA, bcrypt, NTLM, hashlib, passlib, Digital Security, Encryption, Parallel Processing, Digital Forensics, Password Security

I. INTRODUCTION

Protecting touchy statistics is a key priority inside the ever-evolving cybersecurity landscape, and the usage of hash passwords acts as an essential safety mechanism. As technology advances, the significance of robust password safety becomes clearer. This paper gives pioneering paintings on the micro-region of hash and password cracking—a procedure that is intriguingly linked to the improvement of cybersecurity and is fraught with moral concerns. At the heart of this initiative is a well-designed device built to decipher hash passwords, which navigates the space among valid packages like password restoration and ethical issues inside the cybersecurity landscape does the device used extensively everyday hashing algorithms including MD5, SHA, bcrypt. And NTLM receipt situations offer an adaptive answer Beyond the technical challenges, this study delves into the ethical considerations associated with password cracking, and the tool goes beyond being a mere technological innovation demonstrating its dual power of creativity and purpose in all sins; It includes a commitment to responsible cybersecurity practices. Methodologically, our project leverages Python, using important libraries such as hashlib, passlib, ttkthemes etc. to create flexible and efficient solutions Ethics concepts are seamlessly integrated throughout the project, which requires dynamic hashes, special methods optimized for algorithms, parallel to be efficient. manifested in applications and strategies. Attempting to provide cybersecurity professionals with a deeper understanding of hash password cracking, recommending responsible-legal practices in an ever-changing digital landscape as the digital frontier presents new challenges, this review seeks to enlighten professionals and all the necessary ethical considerations into the complex realm of password security.

II. LITERATURE REVIEW

[1] In the study conducted by Ignatius and Yusuf, the authors introduced a novel CUDA-based brute-force algorithms are designed for password cracking. This innovative approach utilized the powerful computation resources of GPUs, specifically evaluating factors influencing the performance for parallel programs. Through custom algorithms optimized for Tesla C2075, the authors achieved substantial speedups, with shared memory identified as a critical factor impacting the algorithm's efficiency.

[2] Sarah and Robert delved into the realm of password cracking using low-power and energy-efficient hardware. Their research provided a proof-of-concept implementation using Adapteva Inc.'s Epiphany chips, showcasing potential applications in security scenarios with less demanding processing requirements. The study addressed limitations such as available RAM and memory bottleneck, opening avenues for further exploration in energy-efficient brute force algorithms.

[3] Can et al. contributed to the field by suggesting an enhanced GPU-based brute force password recovery method for SHA-512. Leveraging OpenCL and C programming, they optimized GPU utilization and incorporated various techniques, achieving a notable speedup compared to existing solutions like Hashcat. Their work not only presented a technical advancement but also emphasized importance in optimizing password recovery schemes for enhanced cybersecurity.

[4] The research by Feng et al. tackled the security challenges posed by MD5 Crypt in UNIX systems. Through CUDA and constant memory utilization, they achieved a significant 44.6% improvement, introducing a fresh challenge to MD5 Crypt's authentication security. Their work emphasized the need for continuous improvement in cryptographic techniques to counteract evolving threats in the cybersecurity landscape.

[5] The study by David et al. experimented with accelerating hash function operations using MPI and CUDA. They introduced three password recovery algorithms with different memory utilization strategies, showcasing scalability for larger databases. The findings highlighted the potential of their approach to improve system security by identifying weak passwords and processing vast amounts of data efficiently. [6] Anh-Duy et al. proposed a homogeneous parallel brute-force algorithm for SHA1, leveraging the CUDA framework. By categorizing passwords into distinct groups and measuring cracking times, the authors provided insights into the memory-consuming nature of their algorithm. Their work offered a nuanced understanding of password cracking times for unconventional character sets, contributing to the broader discussion on cryptographic vulnerabilities.

[6] Maruthi et al. explored FPGA implementation to accelerate brute force attacks on the MD5 hash algorithm. With a focus on pipelining and parallelization, their work achieved notable performance improvements, demonstrating the potential for FPGA-based solutions in password recovery scenarios. This study provides valuable information into hardware-accelerated approaches for cryptographic challenges.

[7] Laatansa et al. investigated the efficacy of GPGPU-based brute-force and dictionary attacks on SHA-1 hashed passwords. Their study compared the success rates of these attacks on passwords of varying lengths, emphasizing the impact of password length and character patterns on attack success. The introduction of a combined attack strategy offered a more balanced approach to password cracking. In their study, Qingbing and Hao proposed an efficient password recovery strategy for WinRAR3 encrypted files. Through CPU + GPU pipeline collaboration, they achieved a substantial improvement in password cracking speed. The study emphasized the importance of heterogeneous multi-core architecture in password cracking scenarios, providing a valuable contribution to the optimization of password recovery processes.

III. INTERACTION LEVELS

The proposed methodology of this research project is to create sophisticated and ethical tool for hashed password cracking, responding to the escalating significance of robust password security in the dynamic landscape of cybersecurity. Striving to strike a delicate equilibrium between legitimate application, password recovery, and the ethical considerations entailed in password cracking, the project aims to contribute to cybersecurity. Central to the objectives is implementing and integrating prevalent hashing algorithms, encompassing MD5, SHA, bcrypt, and NTLM, thereby creating a versatile platform capable of managing a myriad of password encryption methods. Ethical considerations are embedded into the tool's design, underscoring the potential dual nature of password cracking for both constructive and potentially malicious purposes. Furthermore, the tool is designed to dynamically detect the type of hash used in each password, ensuring adaptability to diverse encryption methods. Leveraging the importance of parallel processing through Python libraries and frameworks, the project seeks to enhance efficiency in the password-cracking process. With a user-friendly graphical interface crafted using the Tkinter library, the tool is intended to facilitate seamless interaction and usage. A comprehensive analysis of each hashing algorithm is undertaken, showcasing the tool's adaptability and effectiveness in decrypting passwords across various encryption methods.

Additionally, the implementation of specialized techniques for different algorithms aims to ensure ethical password recovery, aligning with responsible cybersecurity practices. This research project aspires to contribute meaningfully to the cybersecurity discourse by providing professionals with a reliable and ethical tool for hashed password cracking while advocating for responsible practices in password security.

IV. LIMITATION OF EXISTING SOLUTIONS

1. Feature Engineering Complexity:

Many machines learning-based malware attempt to extract relevant features from malware samples. Building an effective system requires domain expertise and cannot capture all the complex features of advanced malware, resulting in lower accuracy.

2. User Interface:

Some tools might lack a user-friendly interface making it less accessible to uninitiated users in cybersecurity.

3. False Positives:

Hash cracking tools may produce false positives, incorrectly identifying passwords that do not match the actual password. This can lead to confusion or misinterpretation of security issues.

4. Computational Power:

These tools often demand significant computational resources, and existing tools may be resource-intensive, making them unsuitable for less powerful systems.

5. Algorithmic Specific:

Tools are often designed to work with specific hashing algorithms. If a tool supports limited algorithms, it may be ineffective against passwords hashed with different or newer algorithms.

6. Complex Passwords:

Cracking tools may face challenges with complex passwords that involve a combination of uppercase, lowercase, symbols, and digits.

V. METHODOLOGY

1. Hash Type Detection:

The tool initiates hash type detection by employing sophisticated algorithms and external tools like hashid. It thoroughly analyzes the structure and characteristics of the provided target hash, utilizing patterns and length considerations to identify the specific hashing algorithm.

2. Parallel Processing:

To optimize performance, the tool implements parallel processing using ThreadPoolExecutor. This technique allows the simultaneous execution of multiple passwords cracking attempts, significantly improving the overall speed and efficiency of the tool.

3. Hashing Algorithms:

The tool boasts support for a diverse range of hashing algorithms, ensuring versatility in handling several types of hashed passwords. This includes widely used algorithms such as MD5, SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, NTLM, and bcrypt. The tool dynamically adapts its approach based on the detected hash type, demonstrating flexibility and comprehensiveness.

4. Specific algorithm Crack:

Tailoring its strategy to the specifics of each algorithm, the tool incorporates specialized crack functions for NTLM and bcrypt hashes. This ensures an optimized and algorithm-specific approach to cracking, taking advantage of the unique characteristics of each hashing method.

5. User Interface:

The tool features a user-friendly Graphical User Interface (GUI) designed for intuitive interaction. Users input crucial details such as target hash, wordlist path, and optional parameters through the GUI. This user interface enhances accessibility and facilitates a seamless.

6. Result Display:

Upon completion of the password cracking attempts, the tool displays the results GUI. Users receive clear indications of whether the tool successfully cracked the password or if the attempt proved unsuccessful. This transparent and informative result display ensures users are promptly informed about the outcomes of their actions.

VI. IMPLEMENTATION

1. Password Cracking Functionality:

The core of the implementation revolves around a robust crack password function, designed to handle various hash algorithms efficiently. This function employs the hashlib library to compute hash values and dynamically detects the hash type through subprocess calls and regex matching. Depending on the hash type identified, the function employs specialized cracking techniques, such as NTLM and bcrypt cracking, ensuring compatibility with different hash formats. Leveraging the power of parallel processing with ThreadPoolExecutor, the function efficiently iterates through a wordlist to attempt password recovery. Additionally, it utilizes cryptographic operations and password hashing libraries like passlib. hash. bcrypt to validate cracked passwords. Through this comprehensive approach, the implementation provides a versatile and effective solution for password recovery tasks.

2. Graphical User Interface (GUI) Integration:

The implementation seamlessly integrates a user-friendly GUI using the tkinter library, enhancing accessibility and usability. The GUI features labeled input fields for target hash, wordlist path, and maximum password length, allowing users to input parameters easily. A browse button enables convenient selection of the wordlist file, further streamlining the setup process. Upon initiating the cracking process, a progress indicator informs users about the ongoing operation. The GUI also displays the cracked password or notifies users if the password is not found in the wordlist. Overall, the intuitive GUI design empowers users to interact with the password cracking tool effortlessly, making it accessible to both novice and experienced users.

3. Optimization and Scalability:

Efforts have been made to optimize the implementation for performance and scalability. By leveraging parallel processing techniques with ThreadPoolExecutor, the script maximizes CPU utilization and accelerates the cracking process, especially on

multicore systems. The dynamic hash type detection mechanism ensures compatibility with a wide range of hash formats, allowing users to tackle diverse password cracking scenarios effectively. Furthermore, the modular design of the implementation facilitates easy extensibility, enabling future enhancements and additions of new cracking techniques or hash algorithms. Overall, the implementation prioritizes efficiency, scalability, and flexibility, making it a valuable tool for cybersecurity professionals and forensics.

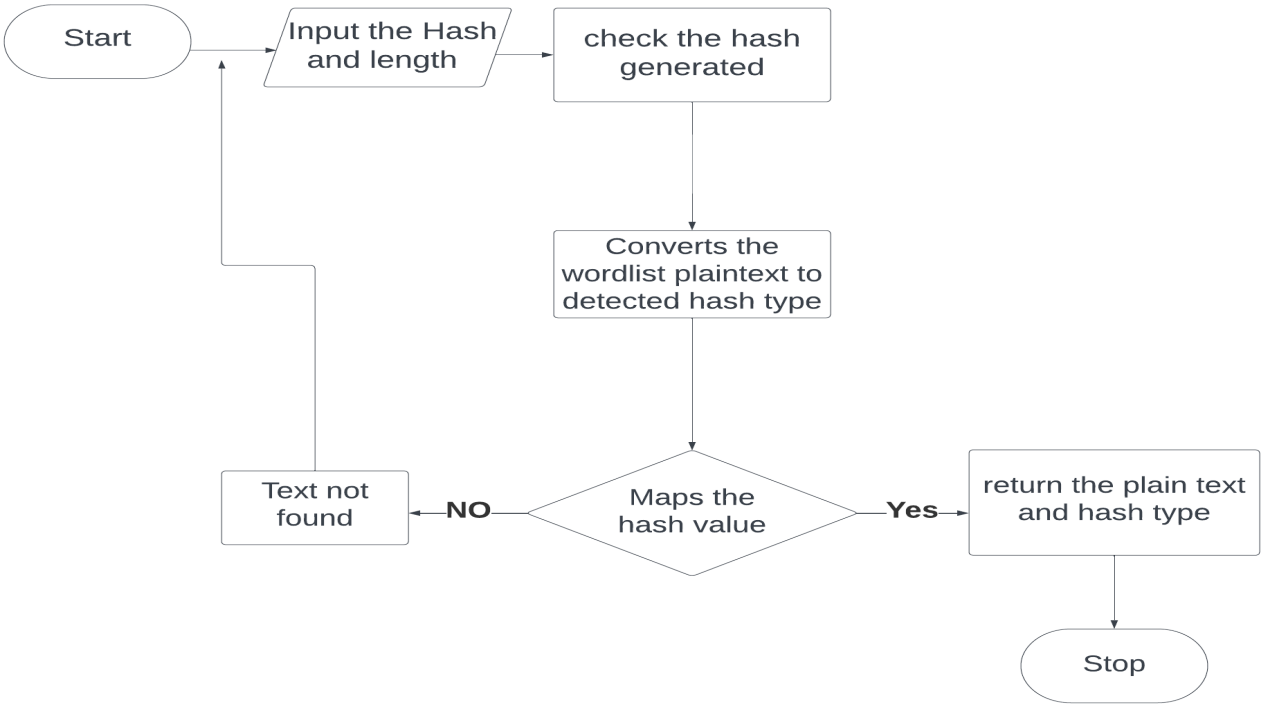

Fig. 1 Working model

VII. RESULT AND DISCUSSION

This Python script offers a versatile and efficient solution for password cracking tasks across various hashing algorithms. Through a well-structured crack_password function, the script harnesses the power of parallel processing and dynamic hash type detection to optimize the cracking process. By utilizing modules such as hashlib, passlib. hash.bcrypt, and tkinter, it seamlessly integrates cryptographic operations and graphical user interface elements. The function's nested structure enables tailored cracking methods for NTLM and bcrypt hashes, ensuring compatibility with a diverse range of hash formats. Moreover, the script's intuitive graphical interface, featuring labeled input fields and browse functionalities, enhances user experience and accessibility. Overall, this script serves as a valuable tool for cybersecurity professionals and forensic analysts seeking to recover passwords efficiently and effectively.

At the heart of the script lies a robust approach to password cracking, with a focus on adaptability and usability. The combination of parallel processing and dynamic hash type detection enables swift and accurate password recovery, even across complex hash formats. Through thoughtful design and integration of GUI elements, the script ensures a seamless user experience, allowing users to initiate and monitor the cracking process with ease. Whether for cybersecurity assessments or forensic investigations, this script stands as a versatile and reliable tool, empowering users to tackle password-related challenges with confidence and efficiency.

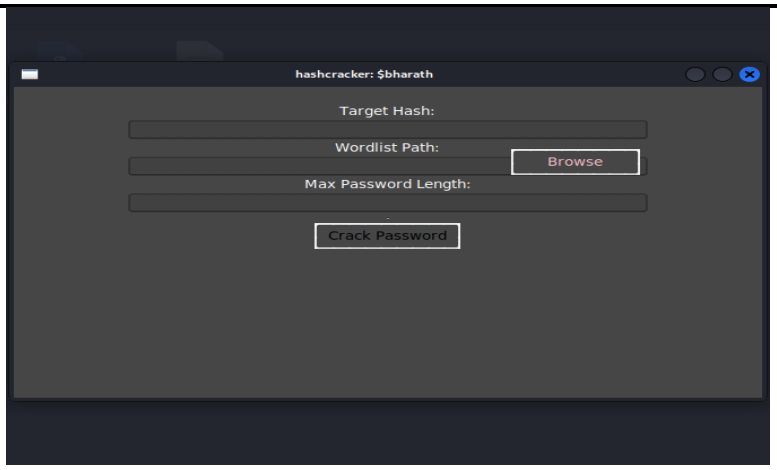

Fig. 2 GUI of Hash Cracker

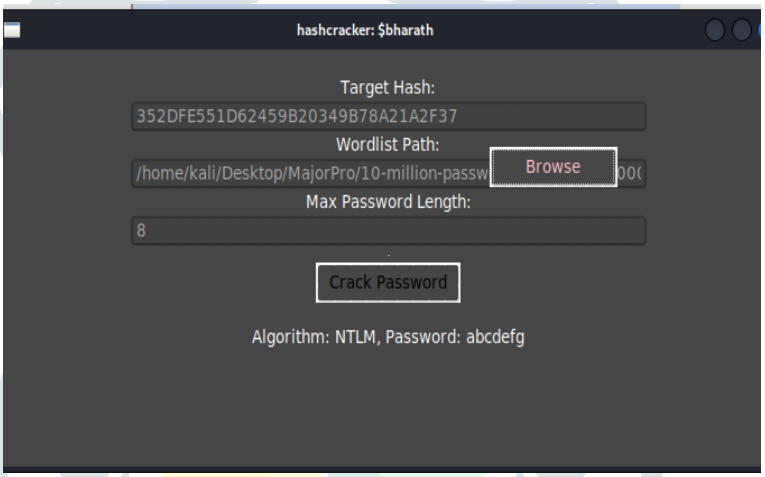

Fig.3 Hash Cracked

VIII. ACKNOWLEDGMENT

The Author is grateful to the CMR College of Engineering & Technology for providing better facilities and practical requirements.

REFERENCES

[1] Chester, J.A., 2015. Analysis of Password Cracking Methods & Applications.

[2] Cryptanalysis of MD5 and SHA: Time for a New Standard, Bruce Schneir Computer World August 19,2004.

[3] Peter Honeyman, Brent Baccala - Performance of Various Unix Password Cracking Programs (2012).

[4] Retrieving NTLM Hashes and what changed in Windows 10-January 21, 2018.

[5] Cyberwar zone, http://cyberwarzone.com/massive- collection-password-wordlists-recover lost-password/, last accessed on 22 March 2015.

[6] Abdulbasit, K. Aoki, J. Guo, K. Matusiewicz, Y. Sasaki, and L. Wang, "Preimages for Step-Reduced SHA-2," Proc. of 15 Int. Conf. on the Theory and Application of Cryptology and Information Security: Advances in Cryptology, Springer, Berlin, Nov. 2008, pp. 578-597.Kumar, S. and Agarwal, D., 2018. Hacking attacks, methods, techniques, and their protection measures.

[7] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your botnet is my botnet: Analysis of a botnet takeover," in Proceeding of the 16th Conference on the Computer and the Communications Security, ser. CCS '09, New York, USA, 2009, pp. 635–647.

[8] R. Veras, J. Thorpe, and C. Collins, "Visualizing semantics in passwords: The role of dates," in proceeding of the Ninth International Symposium on Visualization for Cyber Security, ser. VizSec'12. New York, NY, USA: ACM, 2012, pp. 88–95.

[9] "Hash dumps and Passwords," "http://www.adeptus-echanicus.com/codex/hashpass/hashpass.php".

[10] Vu, A.-D.; Han, J.-I.; Nguyen, H.-A.; Kim, Y.-M.; Im, E.-J. A Homogeneous Parallel Brute Force Cracking Algorithm on the GPU. In proceedings of ICTC 2011, Seoul, Republic of Korea, 28–30 September 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 561–564.