Modification of SHA-512 using Bcrypt and salt for secure email hashing

Sean Eljim S. Castelo, Ruben Jolo L. Apostol IV, Dan Michael A. Cortez, Raymund M. Dioses, Mark Christopher R. Blanco, Vivien A. Agustin
Department of Computer Science, College of Engineering, Pamantasan ng Lungsod ng Maynila, Manila, Philippines

Article Info

Article history:

Received Jun 23, 2023
Revised Oct 4, 2023
Accepted Oct 30, 2023

Keywords:

Bcrypt
Cryptographic hash function
Email hashing
Salt
SHA-512

ABSTRACT

Email security, particularly against phishing, spoofing, and distributed denial-of-service (DoS) attacks, is a pressing concern given the essential role email plays in accessing various online accounts. The study introduced a modified SHA-512 algorithm, implementing additional security layers including randomly generated salt and the Bcrypt algorithm. The modified SHA-512 was comprehensively evaluated on parameters like hash construction, computational efficiency, data integrity, collision resistance, and attack resistance. The results showed its avalanche percentage exceeded the 50% target, reaching 50.08%. Experimental hash-cracking failed to decode the hashes created by the modified algorithm, verifying its protective efficiency. The algorithm also successfully demonstrated data integrity and collision resistance. This indicates that the enhanced SHA-512 algorithm is an effective, more secure hashing method, particularly applicable to email addresses.

Corresponding Author:

Sean Eljim S. Castelo
Department of Computer Science, College of Engineering, Pamantasan ng Lungsod ng Maynila
Manila, Philippines
Email: seancastelo23@gmail.com

1. INTRODUCTION

As the digitalization of businesses continues to increase globally that relies mainly on online platforms. Emails represent a primary communication channel. Despite the availability of competing technologies, emails remain a crucial source of enterprise information and serve ‘as a virtual extension of the users’ workplace’ [1]. However, alongside these benefits, email communication has led to an escalation in security issues. The possibility of email addresses being compromised raises concerns over potential privacy breaches, such as cross-device tracking and the linking of online and offline activities [2]. Due to the significant security challenges posed by traditional email usage, several solutions have been proposed to improve the security of email addresses. One promising approach is the use of hashed emails (HEMs), which offer an unprecedented opportunity for marketers by enabling the connection of targeting data across devices, platforms, and channels [3]. Despite their advantages, HEMs are not immune to attacks from hackers who, once they gain access to email addresses, can cause serious harm, ranging from sending phishing emails to engaging in fraudulent activities [4].

One promising approach is the use of HEMs, which offer an unprecedented opportunity for marketers by enabling the connection of targeting data across different devices, platforms, and channels [3]. Notably, previous research in this field introduced a modified SHA-512 algorithm to address email security concerns. This study modified the algorithm by altering the message scheduling, hash construction, and compression function, and by reducing the iterations. The modified algorithm’s effectiveness was then evaluated through tests simulating various attacks, including brute-force, rainbow table, dictionary, and online cracking attacks. The findings revealed insights into the algorithm's computational efficiency, avalanche effect, and its capacity to resist different forms of attacks. They found that there's a need for a salt to make it more secure efficient [4].

The increasing frequency of data breaches and the sophistication of email scams, such as email phishing, underline the urgency of enhancing email address security [5]. Cryptographic hashing algorithms, such as the secure hashing algorithms (SHA) mainly the SHA-512, are an integral part of the solution to the information security problem. Implementation of the SHA 512 algorithm method produces the longest number of bits of 512 bits so as to ensure system security and data confidentiality [6], [7]. However, the resilience of these algorithms against evolving cyber threats is still under debate.

To address these challenges, we have introduced innovative enhancements to the SHA-512 algorithm, establishing a novel approach distinct from prior methodologies. Our approach centers on a dual-layered protection strategy, combining the salting technique which involves adding random data to the input of a hash function, has been shown to improve security against brute-force attacks [8]–[10] and the Bcrypt algorithm a cryptographic technique specifically designed for secure password hashing and has been widely recognized for its effectiveness in protecting sensitive data [11]–[14]. A pivotal feature is the meticulous layering of safeguards. Introducing a 26-byte random salt as an additional layer of randomness amplifies the algorithm's cryptographic strength. This newfound salt seamlessly integrates into the Bcrypt algorithm, which employs another salt layer with 12 rounds by incorporating these techniques into the hashing process, the modified algorithm is expected to significantly increase its resistance against various attacks and ensure better data integrity and security [15], [16]. This tandem of salt layers significantly heightens the algorithm's defense against potential attacks, creating an intricate and robust security framework. Importantly, our methodology achieves this heightened security without compromising efficiency. Although there is a slight increase in construction time, the substantial fortification it brings far outweighs this marginal trade-off. Hashing email addresses will play huge a role in safeguarding user's data and which specific algorithm will be used that will suffice to entirely secure the user's data [17], [18]. By applying modifications to the SHA-512 algorithm, including the integration of salt and Bcrypt, this research aims to significantly enhance the security of HEMs [19].

This research presents unique security enhancements that address contemporary threats, ensuring the integrity of email data and safeguarding against potential tampering. Additionally, it adapts to concerns regarding email privacy and user identification raised by hashed email approaches. Our comprehensive evaluation, spanning hash construction time comparison, avalanche testing, and attack resistance assessment, provides a holistic understanding of security concerns and validates the proposed enhancements [20]. The real-world applicability of our findings is significant, given the fundamental role of email communication in modern life. By introducing novel modifications to a widely-used algorithm and rigorously assessing its effectiveness against potential attacks, this research contributes to the broader cybersecurity domain, offering insights into innovative techniques for enhancing data protection. By addressing unsolved problems, introducing innovative enhancements, and demonstrating their relevance, this study not only contributes to the academic realm but also offers practical solutions to pressing security challenges in today's digital landscape.

2. METHOD

The study will utilize a modified SHA-512 algorithm with additional security layers, including randomly generated salt and the Bcrypt algorithm [14], to enhance email security against phishing, spoofing, and A distributed denial-of-service (DDoS) attacks [21]–[25]. Comprehensive evaluation will encompass hash construction, computational efficiency, data integrity, collision resistance, and attack resistance. The avalanche effect will be quantified, and hash-cracking experiments will verify protective efficiency [21]. Additionally, data integrity and collision resistance will be rigorously assessed. The methodology will affirm the efficacy of the enhanced SHA-512 algorithm in providing heightened security for email addresses.

2.1. Implementation of enhanced security layers

In (1) gives a random generated 26-byte salt [8], [9]. In (2) generates a hashed using Bcrypt algorithm with 12 rounds of hashing, resulting in the final hashed value [14], [16]:

$$\text{os.urandom}(26) \tag{1}$$

$$\text{bcrypt.hashpw}(\text{salted\_username.encode}(), \text{bcrypt.gensalt }(\text{rounds}=12)) \tag{2}$$

2.2. Comprehensive assessment of enhanced algorithm's security through multi-dimensional evaluation and hash-cracking validation

In (3) is used to calculate the construction time of both modified SHA-512 and SHA-512 to compare their efficiency in terms of hash construction time. In (4) is to ensure that the hash value remains consistent for different inputs. In (5) is to determine whether the algorithm can resist producing the same hash output for different inputs. In (6) is to determine if even minor alterations in input lead to significantly different hash outputs [19]. In (7) various different cracking tools is used to assess the protective efficiency [22]–[26].

Hash construction time comparison (3)

Data integrity test (4)

Collison test (5)

Avalanche test (6)

Attack resistance (7)

3. RESULTS AND DISCUSSION

The researchers enhanced an algorithm for hashed email addresses, with a focus on evaluating and analyzing the modified SHA-512. The study explores the runtime execution and security trade-off between the Bcrypt+hashlib SHA-512 algorithm and the Python SHA-512 algorithm. It delves into construction time, security level considerations, data integrity test results, collision resistance, avalanche effect, and attack resistance [19], [21]–[25]. The findings underscore the enhanced security and resistance of the modified algorithm, positioning it as a valuable solution for data protection, while also carrying implications for the broader field of data security and cryptography.

The results in Table 1 highlight a distinctive aspect: the different construction times and security implications of the Bcrypt+hashlib SHA-512 algorithm versus the Python SHA-512 algorithm. The Bcrypt-infused approach entails a longer hash construction time, attributed to the additional salting and Bcrypt steps. Yet, this temporal trade-off aligns with an essential revelation-the enhanced security given by these measures. Such discussion regarding construction time and security efficiency points out the approaches that system designers must make. The paper addresses a crucial issue in the field of algorithmic design by giving insight into this trade-off between security improvement and construction speed.

Table 1. Runtime execution and security trade-off between modified SHA-512 and Python SHA-512 algorithms

	Modified SHA-512 code	SHA-512 from Python (HASHLIB)
Plain text	Ab3naleZ@yahoo.com	Ab3naleZ@yahoo.com
Message digest generated	6da617c6bc733767792575bad9ad69c031b895c9a3a1725d5c699a4f86ba087ba0ac1d2c92c5190f8312dd7f7bb3073be785a8db2d11208980a8aa498a14dade	104598872d3dae8d3e4257817d9009c5d1dfaae6a0e8fbf922e3f1d601f7ca27e28d8c876157f83081f4ce37819daee2c1f1f7924770b404e90f9e6ee21f7caa
Construction time	0.6077208518981934	0.0001912117004395
Comparison	Slower	Faster

The purpose of the test illustrated in Table 2 is to demonstrate that hash functions serve as effective tools for confirming data integrity. Conducting the integrity test multiple times using distinct random usernames ensures the consistent generation of expected hash values by the modified SHA-512 and normal SHA-512 algorithms, validating their reliable functionality.

In the experiment, two hashing methods were tested: the “Modified SHA-512,” which employs a combination of randomly generated salt, Bcrypt, and SHA-512, and the “Normal SHA-512,” which solely utilizes SHA-512. Both methods achieved a perfect 100% data integrity preservation rate across 100 trials, as indicated by a “Success rate” of 100/100 for both. This outcome underscores the proficiency of both methods within the specific test parameters, prompting a comprehensive examination of additional factors like computational expense and security considerations to comprehensively assess their respective benefits and drawbacks.

Table 2. Data integrity test results for modified SHA-512 and Python SHA-512 algorithms

	Modified SHA-512 Code	SHA-512 from Python (HASHLIB)
Plain text (1)	KMgxa@LdLgq.com	KMgxa@LdLgq.com
First-MD (Message digest) generated	30e8a57662e6576c1cdc1852fa690581dfe1110cc90e780eda5cbda0cfaf93f90bd01316839472579ad4a39e237a7294045756a261e68b110b51cadd9e0e020e	1afcad26e88f4591e1a8dba4ff1e426ccc1257d5ec35c710b7fa44f902ce61d8640ce43c74579cb79c327abefedd90cf20dba03bc9b6a5843238cba4f74193a1
Second-MD (Message digest) generated	30e8a57662e6576c1cdc1852fa690581dfe1110cc90e780eda5cbda0cfaf93f90bd01316839472579ad4a39e237a7294045756a261e68b110b51cadd9e0e020e	1afcad26e88f4591e1a8dba4ff1e426ccc1257d5ec35c710b7fa44f902ce61d8640ce43c74579cb79c327abefedd90cf20dba03bc9b6a5843238cba4f74193a1
Data integrity check	True (The first MD = the second MD)	True (The first MD = the second MD)
…	…	…
Plain text (100)	Hsubl@sJ7FK.com	Hsubl@sJ7FK.com
First-MD (Message digest) generated	0c195e59a9f2ddabcfcf5c962fdd316d709608e7d395fb65b26509480db1d1e23cea3ff2d09d03974524e335bba4d19a0e1e0928a3a19d93c5646651bb0c496b	08ee88ed6008566406594fbcfaa37acd2d2cf32fe75de2c0d3d574fa4167ae4b29259fd9982e7a119dc2804ddd55502a75b5de868d3ff24cb4355871b4e53c45
Second-MD (Message digest) generated	0c195e59a9f2ddabcfcf5c962fdd316d709608e7d395fb65b26509480db1d1e23cea3ff2d09d03974524e335bba4d19a0e1e0928a3a19d93c5646651bb0c496b	08ee88ed6008566406594fbcfaa37acd2d2cf32fe75de2c0d3d574fa4167ae4b29259fd9982e7a119dc2804ddd55502a75b5de868d3ff24cb4355871b4e53c45
Data integrity check	True (The first MD = the second MD)	True (The first MD = the second MD)

In Table 3, the collision resistance test outcomes for the modified SHA-512 and SHA-512 algorithms are presented, assessing their capacity to withstand collisions that could potentially undermine hashing algorithm integrity and security. The test results shows that both algorithms effectively preserved data integrity, with no instances of collisions detected throughout the 100 trials conducted. The modified SHA-512 algorithm ensured the distinctiveness of hash outputs, affirming its ability to generate unique hash values for various input data.

Table 3. Collision resistance test results for modified SHA-512 and Python SHA-512 algorithms

	Modified SHA-512 code	SHA-512 from Python (HASHLIB)
Collision resistance	True	True
Result	Both modified SHA-512 and SHA-512 from Python (HASHLIB) was able to generate unique hash value for various input data	

Table 4 is an avalanche test that tried 3 different scenarios such as 1-byte difference, difference lengths, random strings with increased length, and lastly overall average.

- 1-byte difference: as observed in Table 4 when comparing two usernames with a 1-byte difference, the modified SHA-512 algorithm demonstrates an average avalanche effect of 0.520, while the normal SHA-512 algorithm has an average avalanche effect of 0.505. This suggests that both algorithms exhibit a considerable ability to propagate changes, but the modified SHA-512 algorithm shows a slightly stronger avalanche effect.
- Different lengths: as observed in Table 4, when evaluating the avalanche effect for usernames with different lengths, the modified SHA-512 algorithm has an average avalanche effect of 0.493, while the normal SHA-512 algorithm has an average avalanche effect of 0.480. This indicates that both algorithms exhibit a noticeable avalanche effect, with the modified SHA-512 algorithm showing a slightly higher impact.
- Random strings with increased length: as observed in Table 4, when considering random strings with increased length, the modified SHA-512 algorithm has an average avalanche effect of 0.500, and the normal SHA-512 algorithm has an average avalanche effect of 0.495. These results suggest that both algorithms effectively propagate changes, with the modified SHA-512 algorithm demonstrating a slightly stronger avalanche effect.
- Overall average avalanche effect: as observed in Table 4, the overall average avalanche effect for the modified SHA-512 algorithm is calculated as 0.505, while for the normal SHA-512 algorithm, it is 0.495. These values indicate that both algorithms exhibit a significant avalanche effect, but the modified SHA-512 algorithm shows a slightly higher average impact.

Table 5 shows that attacks were unsuccessful on the modified SHA-512, considering no hash value was cracked during experimentation with the use of John the Ripper, hashcat cracking tool with the use of Crackstation.net an online cracking tool. Attacks on the normal SHA-512 were successful, as all hashes with the use of John the Ripper, Hashcat cracking tool with the use of Crackstation.net an online cracking tool. It was cracked and plaintext passwords were exposed. Testing results showed that the modified SHA-512 has a 100% attack resistance capacity against known password-cracking attacks and can protect hashed passwords from attackers.

Table 4. Avalanche effect test for modified SHA-512 and Python SHA-512 algorithms

Test scenario	Modified SHA-512 code	SHA-512 from Python (HASHLIB)
1-Byte difference	0.508	0.496
Different lengths	0.498	0.487
Random strings with increased length	0.514	0.504
Overall average avalanche effect	0.508	0.497

Table 5. Attack resistance evaluation

	The number of hashes cracked			
Attack	SHA-512 (Hashcat)	SHA-512 (John the Ripper)	Modified SHA-512 (Hashcat)	Modified SHA-512 (John the Ripper)
Brute-force	0	0	0	0
Rainbow table	5	5	0	0
Dictionary	5	5	0	0
Online cracking	0		0	

3.1. Brute force attack

A brute-force attack observed in Table 5 uses all possible combinations of the given parameters to crack the password hash [21], [22]. In a brute force attack, the user uses every possible combination of the alphabet hoping that at least one combination is correct. This attack is faster when it is used to check for short passwords. The only drawback of this method is that, if the password is a long one it takes longer to find the right password, hence consumes lots of system resources [23]. The John the Ripper cracking tool and the Hashcat cracking tool are successful in a Brute force attack, cracking all 5 input samples that are hashed using the proposed modified SHA-512 and the normal SHA-512.

3.2. Dictionary attack

A dictionary attack observed in Table 5 breaks hashes using a massive wordlist of passwords. Each word's hash value is computed and compared to a predefined hash value; if the values match, the plaintext password is exposed [24]. The John the Ripper cracking tool and the Hashcat cracking tool are unsuccessful in dictionary attack, unable to crack all input samples that are hashed using the proposed modified SHA-512 while the normal SHA-512 is cracked using these tools.

3.3. Rainbow table attack

A rainbow table attack observed on Table 5 cracks password hashes using rainbow tables. Philippe Oechslin introduced the rainbow table in 2003, which used the time-memory trade-off technique. When generating the table, the disk space can be specified by determining the number of chain counts [24], [25]. The John the Ripper cracking tool and the Hashcat cracking tool is unsuccessful in dictionary attack, unable to crack all input samples that is hashed using the proposed modified SHA-512 while the normal SHA-512 is cracked using these tools.

3.4. Online cracking tool

CrackStation observed in Table 5 utilizes massive, precomputed lookup tables for cracking hashes. It shows that the attack was successful on the normal SHA-512. Meanwhile, the modified SHA-512's hashes were not found, and the hash type is unknown, as shown in [24], [26].

4. CONCLUSION

The Bcrypt+salt+hashlib SHA-512 algorithm provides enhanced security but takes more time to construct the hash. Both the modified SHA-512 and the normal SHA-512 algorithms demonstrate good collision resistance and therefore was able to maintain the integrity of the data, while the modified SHA-512 algorithm exhibits a slightly stronger avalanche effect, and attack resistance evaluation-implies that the modified SHA-512 algorithm is more capable of propagating changes in the input, leading to more significant changes in the resulting hash. The researcher also found out, while undergoing experimentations that the modified version of SHA-512 can be implemented in password security with a little less complexity compared to the researcher's implementation of the modified algorithm in email addresses While the study explores modifications such as the addition of salt, Bcrypt, or other cryptographic functions, future researchers can delve deeper into advanced modifications or combinations of techniques. This can include exploring key stretching algorithms, adaptive hashing methods, or hybrid approaches that combine multiple cryptographic functions and using "Cain and Abel" a password cracking tool that is safer in implementation with the use of a virtual machine. The study can also be applied to password hashing.

REFERENCES

[1] P. A. Gloor, A. F. Colladon, and F. Grippa, "The digital footprint of innovators: using email to detect the most creative people in your organization," *Journal of Business Research*, vol. 114, pp. 254–264, Jun. 2020, doi: 10.1016/j.jbusres.2020.04.025.

[2] S. Englehardt, J. Han, and A. Narayanan, "I never signed up for this! Privacy implications of email tracking," *Proceedings on Privacy Enhancing Technologies*, vol. 2018, no. 1, pp. 109–126, Jan. 2018, doi: 10.1515/popets-2018-0006.

[3] LiveRamp, "Email hashing: the trouble with hashed emails (HEMs)," *LiveRamp*, 2022. https://liveramp.com/blog/the-trouble-with-hashed-emails-hems/.

[4] B. Nelson, "Here's What hackers can do with just your email address," *Reader's Digest*, 2022. https://www.rd.com/list/what-hackers-can-do-with-email-address/.

[5] R. Wash, "How experts detect phishing scam emails," *Proceedings of the ACM on Human-Computer Interaction*, vol. 4, no. CSCW2, pp. 1–28, Oct. 2020, doi: 10.1145/3415231.

[6] N. Kishore and B. Kapoor, "Attacks on and advances in secure hash algorithms," *IAENG International Journal of Computer Science*, vol. 43, no. 3, pp. 326–335, 2016.

[7] M. Sumagita and I. Riadi, "Analysis of secure hash algorithm (SHA) 512 for encryption process on web based application," *International Journal of Cyber-Security and Digital Forensics (IJCSDF)*, vol. 7, no. 4, pp. 373–381, 2018.

[8] D. Arias, "Adding salt to hashing: a better way to store passwords," *Auth0.com*, pp. 1–13, 2018, [Online]. Available: https://auth0.com/blog/adding-salt-to-hashing-a-better-way-to-store-passwords/.

[9] J. Lake and J. Lake, "Encryption, hashing, salting – what's the difference? Comparitech," 2018. https://www.comparitech.com/blog/information-security/encryption-hashing-salting/.

[10] U. Rathod, M. Sonkar, and B. R. Chandavarkar, "An experimental evaluation on the dependency between one-way hash functions and salt," in *2020 11th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2020*, Jul. 2020, pp. 1–7, doi: 10.1109/ICCCNT49239.2020.9225503.

[11] J. Jeong, D. Woo, and Y. Cha, "Enhancement of website password security by using access log-based salt," in *Proceedings - 2019 4th International Conference on Systems of Collaboration, Big Data, Internet of Things and Security, SysCoBIoTS 2019*, Dec. 2019, pp. 1–3, doi: 10.1109/SysCoBIoTS48768.2019.9028012.

[12] K. C. Pradeep, R. Soman, and P. Honnavalli, "Validity of forensic evidence using hash function," in *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, Jun. 2020, pp. 823–826, doi: 10.1109/icces48766.2020.9138061.

[13] M. Marx, E. Zimmer, T. Mueller, M. Blochberger, and H. Federrath, "Hashing of personally identifiable information is not sufficient," *Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI)*, vol. P-281, pp. 55–68, 2018, doi: 10.18420/sicherheit2018_04.

[14] T. P. Batubara, S. Efendi, and E. B. Nababan, "Analysis performance BCRYPT algorithm to improve password security from brute force," *Journal of Physics: Conference Series*, vol. 1811, no. 1, p. 012129, Mar. 2021, doi: 10.1088/1742-6596/1811/1/012129.

[15] N. Kumar and P. Chaudhary, "Password security using bcrypt with AES encryption algorithm," in *Smart Innovation, Systems and Technologies*, vol. 77, 2018, pp. 385–392.

[16] S. C. A and N. Francis, "Password security using BCrypt," in *Proceedings of the National Conference on Emerging Computer Applications (NCECA)-2021*, 2021, vol. 3, no. 1, p. 281, doi: 10.5281/zenodo.5094166.

[17] J. P. Conley, "Encryption, hashing, PPK, and blockchain: a simple introduction," Vanderbilt University, 2019.

[18] L. Demir, A. Kumar, M. Cunche, and C. Lauradoux, "The pitfalls of hashing for privacy," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 1, pp. 551–565, 2018, doi: 10.1109/COMST.2017.2747598.

[19] P. J. F. Bemida, A. M. Sison, and R. P. Medina, "Modified SHA-512 Algorithm for Secured Password Hashing," in *3rd IEEE International Virtual Conference on Innovations in Power and Advanced Computing Technologies, i-PACT 2021*, Nov. 2021, pp. 1–9, doi: 10.1109/i-PACT52855.2021.9696928.

[20] R. L. Quilala, A. M. Sison, and R. P. Medina, "Modified SHA-1 algorithm," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 11, no. 3, pp. 1027–1034, Sep. 2019, doi: 10.11591/ijeecs.v11.i3.pp1027-1034.

[21] C. Ge *et al.*, "Optimized password recovery for SHA-512 on GPUs," in *Proceedings - 2017 IEEE International Conference on Computational Science and Engineering and IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, CSE and EUC 2017*, Jul. 2017, vol. 2, pp. 226–229, doi: 10.1109/CSE-EUC.2017.226.

[22] L. Bosnjak, J. Sres, and B. Brumen, "Brute-force and dictionary attack on hashed real-world passwords," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018 - Proceedings*, May 2018, pp. 1161–1166, doi: 10.23919/MIPRO.2018.8400211.

[23] T. Kakarla, A. Mairaj, and A. Y. Javaid, "A real-world password cracking demonstration using open source tools for instructional use," in *IEEE International Conference on Electro Information Technology*, May 2018, vol. 2018-May, pp. 387–391.

[24] M. A. D. Brogada, A. M. Sison, and R. P. Medina, "Cryptanalysis on the head and tail technique for hashing passwords," in *Proceeding - 2019 IEEE 7th Conference on Systems, Process and Control, ICSPC 2019*, Dec. 2019, pp. 137–142.

[25] L. Zhang, C. Tan, and F. Yu, "An improved rainbow table attack for long passwords," *Procedia Computer Science*, vol. 107, pp. 47–52, 2017, doi: 10.1016/j.procs.2017.03.054.

[26] Crackstation, "CrackStation - online password hash cracking - MD5, SHA1, Linux, Rainbow Tables, etc.," 2023. https://crackstation.net/.

BIOGRAPHIES OF AUTHORS

Sean Eljim S. Castelo is currently studying Bachelor of Science in Computer Science at Pamantasan ng Lungsod ng Maynila, University of the City of Manila, Intramuros, Manila, Philippines. His research interests include cybersecurtiy, cryptography, and data science. He can be contacted at email: seancastelo23@gmail.com.

Ruben Jolo L. Apostol IV is currently studying Bachelor of Science in Computer Science at Pamantasan ng Lungsod ng Maynila, University of the City of Manila, Intramuros, Manila, Philippines. His research interests include cybersecurtiy, cryptography, and data science. He can be contacted at email: subjoloapostol@gmail.com.

Dr. Dan Michael A. Cortez is currently an Associate Professor at the Pamantasan ng Lungsond ng Maynila. He is the Program Chair of the Computer Science Department Acting Assistant Dean of Office of Student Development and Services and Assistant Vice President for University Priorities. He has ten (10) years of teaching experience. He graduated with the degree of Bachelor of Science in Information Technology from the Pamantasan ng Lungsod ng Maynila. He also obtained his Master of Science in Information and Communications Technology degree from the same university. He finished his Doctor in Information Technology from Technological Institute of the Philippines-Quezon City Campus. He is a member of the Philippine Society of Information Technology Educators (PSITE-NCR) and Computing Society of the Philippines. He is also an author of various books and has already published his research in the field of information technology, both locally and internationally. He can be contacted at email: dmacortez@plm.edu.ph.

Prof. Raymund M. Dioses is currently Assistant Professor I at Pamantasan Ng Lungsod ng Maynila. He is from the Department of Education Senior High School Department before he entered at Pamanatasan ng Lungsod ng Maynila. His teaching abilities were greatly enhanced by working experience at CORE Gateway College Inc. (CGCI) where he served as one of the College Faculty and Chairperson of the Computer Education Department for eight years and five years is Senior High School as Teacher II at Department of Education. He is graduated with the degree of Bachelor of Science in Computer Science at St. Jude College. He finished his master’s degree program in Master of Arts in Education major in Educational Management at CORE Gateway College, San Jose City Nueva Ecija. He studies another master’s degree program in Master of Information Technology major in Computer Education, ongoing thesis at Nueva Ecija University of Science and Technology, Cabanatuan City, Nueva Ecija. He can be contacted at email: rmdioses@plm.edu.ph.

Prof. Mark Christopher R. Blanco is currently a Chief Director for Information and Communications Technology Office (ICTO) at Pamantasan ng Lungsod ng Maynila Intramuros, Philippines at the Pamantasan ng Lungsod ng Maynila. Experienced Parttime Professor with a demonstrated history of working in the information technology and services industry. Skilled in Search Engine Optimization (SEO), PHP, Fiber Optic Cable, WordPress, and Documentation. Strong education professional with a Master’s Degree focused in Science in Information Technology from Bulacan State University. He can be contacted at email: mcrblanco@plm.edu.ph.

Prof. Vivien A. Agustin is currently an Assistant Professor at the Pamantasan ng Lungsod ng Maynila. Prior to joining Pamantasan ng Lungsod ng Maynila, she had been a professor at the Universidad de Manila for 22 years. She also served as the program coordinator in the aforementioned institution. She earned a bachelor’s degree in information technology from St. Paul University in Tuguegarao, Cagayan. She also holds a Master’s Degree in Information Technology from Pamantasan ng Lungsod ng Maynila in 2021 and Master of Public Management Governance from Universidad de Manila in 2015. She is a member of the Philippine Society of Information Technology Educators (PSITE-NCR), Institute of Industry and Academic Research Incorporated and Aloysian Publication. She was able to publish her research on a global scale in the field of information technology, and she is currently working on her new research that she hopes to publish and present. She can be contacted at email: vaagustin@plm.edu.ph.