ISSN No:-2456-2165

Enhancement of Time-Based One-Time Password for 2-Factor Authentication

Danrich U. Balasta[1], Stacy Marie C. Pelito[1], Mark Christopher R. Blanco[1], Antolin J. Alipio[1], Khatalyn E. Mata[1], Dan Michael A. Cortez[1]
Computer Science Department, College of Engineering and Technology
Pamantasan ng Lungsod ng Maynila (University of the City of Manila)
Intramuros, Manila 1002, Philippines

Abstract:- In the interest of digital security, 2-Factor Authentication (2FA), has been widely used throughout different sites and applications to secure and authenticate a user's identity, Time-Based One-Time Password (TOTP) algorithm is one of the most utilized algorithms when it comes to 2FA due to its reliability when it comes to securing user access through generating a code that has a limited validity, usually for 30 seconds or less. TOTP generates a code with the use of current time and a secret key. Despite the security TOTP provides, the delivery of the code through SMS is still vulnerable to interception by a third party since the connection between the client and the server can be insecure. This paper proposes an enhancement to the TOTP algorithm by applying AES encryption to the generated code before delivering it to the client. This paper shows that the implementation of AES to the TOTP algorithm has helped generate a stronger OTP and has made it harder for hackers to crack.

Keywords—AES; OTP; secret key; Time-Based One-Time Password Algorithm; 2-Factor Authentication.

I. INTRODUCTION

As time passes, an increasing number of people go online to connect and trade information, among other things. Different transactions are now going online and the protection of one's online identity is more valuable now than ever. Nowadays, a static password, which are passwords that can be reused and may or may not expire are usually created by users [1], that is supposed to safekeep your personal information and activities in an online platform, can easily be cracked in a matter of seconds. Static passwords are no longer enough to secure one's personal information and online assets which is why 2-Factor Authentication (2FA) is being implemented as an extra layer of security in authenticating a user's identity and actions online.

One-Time Password (OTP) is an algorithm that allows users to be authenticated for a single transaction or login session using passwords made up of random alphanumeric characters. In the light of modern threats and attacks to online accounts, the use of OTP to authenticate a user's identity has proven to be vital in hindering easy access to accounts and online profiles. It is unquestionably one of the most straightforward and widely used methods in securing online accounts. OTP has two types; HOTP or HMAC-based One-Time Password and TOTP or Time-based One-Time Password. HOTP is based on a counter value that increases monotonically and a static symmetric key that is only known by the client and the server while in TOTP, an authentication token or password is generated for a limited time using the current time and a secret key.

In this study, the researchers opted to enhance the TOTP algorithm by implementing AES to the algorithm to be used for 2-Factor Authentication.

II. RELATED STUDIES

One-Time passwords (OTPs) are a string of random characters, both numeric and alphanumeric, that are produced automatically and allow users to be authenticated by agreeing on the shared possession of a shared secret key [2]. OTPs are valid only for one login session. OTPs are used to bypass the limitations of static password-based verification which is why it is used in Two-Factor Authentication (2FA). Leslie Lamport initially discussed the idea of one-time password in her article "Password Authentication with Insecure Communication" and the concept was later commercialized as the S/Key system in the late 1980s by Neil Haller, Phil Karn, and John Walden at Bell Communications Research Inc. This results in the first standardization of OTPs which was introduced under RFC 2289 by Neil Haller, Craig Metz, Phillip J. Nesser II, and Mike Straw [3].

OTPs are classified as symmetric key cryptography since they employ a single secret key for both encryption and decryption [4]. The said single secret key is mutually agreed upon by two parties for encryption, decryption, and digital signatures, and it is utilized to generate OTPs. OTPs can be created by several triggers or processes. These triggers can occur whenever the user authenticates or simply when a static password fails. There are two different triggers for OTPs - event-based and time-based. Time-based triggers (Time-based One-time Password) generate OTPs that are valid for only a certain length of time, often as little as 30 or 60 seconds while Event-based OTP (HMAC-based One-time Password) triggers generate OTPs after an event, typically the press of a button, and remain valid until a user uses them.

HOTP or HMAC-based One-time Password algorithm is an algorithm introduced by OATH (Open Authentication) members in 2005 that uses a growing counter value and a static symmetric key that is only known by the token and the validation service [5]. HOTP is also known as Event-based OTP since it generates an event-based token using HMAC-SHA1 algorithm. A counter determines whether the resulting key value is static or dynamic [6].

TOTP or Time-based One-time Password is a variation of HOTP that was proposed by OATH members in 2008 and was formally accepted in 2011. TOTP is a time-based algorithm that differs from HOTP, which employs the counter as a moving factor. M'Raihi et.al. [7] defined TOTP as

$$TOTP = HOTP(K, T) \qquad (1)$$

where T is an integer that specifies the number of time steps between $T0$ and the current Unix time.

OTP system has proven to be the most effective method of user authentication [8]. Even though the OTP authenticates the correct user, the OTP value is stolen by attackers due to many types of cyber-attacks such as active attacks, passive attacks, phishing, etc. Attackers can gain access to the OTP through such means which makes it critical to protect the OTP delivered to the authorized user. Thus, OTP encryption is required to prevent unauthorized access. In the study conducted by Khishipah and Muhammad in 2014 [9], the authors implemented and tested three OTP techniques, TOTP, HOTP, and CROTP (Challenge Response One-time Password) to prevent replay attacks in RADIUS (Remote Authentication Dial-In User Service) protocol. Based on their study, TOTP is considered as the fastest and most efficient in CPU overhead compared to CROTP and HOTP. The authors also stated that TOTP is the most secure algorithm because of its time-limited generation of OTP.

SMS-based OTP is the most extensively used multi-factor authentication and authorization system since most of them demand the use of mobile phones as a secondary device. Users of an SMS-based OTP system must supply information such as their login and password, as well as the generated verification code before they can access secured systems, thus, it limits the frequency or chance of unauthorized access [10]. SMS is transmitted from one MS to another, the information included in it is conveyed as plaintext [11]. When sensitive data is sent in plaintext over an unprotected network, attackers can target it with various attacks (interception, message manipulation, etc.) before it reaches the SMS Center. If SMS is encrypted, it is encrypted with A5/1, which is a very poor encrypting technique that is readily broken, or it is sent as plain text, which can be easily intercepted by an attacker. As a result, SMS security is critical in OTP delivery [12].

Nugroho E., Rachman J., & Iman M. implemented AES-256 and OTP in their new applicant account system in 2016 [13], which updated AES-256 after three researchers from various institutions and firms, Andrey Bogdanov from K.U. Leuven, Dimitri Khovratovich from Microsoft and Christian Reachberger from ENS Paris discovered in 2011 that the algorithm allows probability to decode secret keys quicker than ever [14]. The authors updated the S-box and ShiftRow to increase the complexity of AES by allowing their dynamics to follow the keys provided. The modification in S-Box is designed to enhance the confusion of AES and modification on ShiftRows is designed to boost the diffusion of AES. The authors then assess their modifications using the avalanche effect and randomness test, with the AES-256 scoring roughly 50% on the Avalanche effect and passing five basic random tests in the Randomness test. The stated algorithm, however, takes longer to perform than the standard AES-256 algorithm due to its modifications.

Sudar C., Arjun S.K., & Deepthi L.R. used TOTP to develop their proposed Wi-Fi authentication system [15]. According to the authors, SSID and password combinations are vulnerable to security breaches such as phishing and brute-force attacks, hence they planned to "black box" (view the inputs and outputs with looking into its internal processing) the process of connecting to a Wi-Fi network for the user and the process of creating periodic secure passwords for the network without human intervention. Password-based attacks are useless since the time-restricted capabilities of TOTP, and even if an attacker is successful, the password will have changed by the time the attack is successful. Phishing attacks are also ineffectual since the user does not connect directly to any fraudulent access point because the application handles all access point connections and can distinguish between authentic and fake access points.

Kaur J.& Kaler N. developed an OTP-based data security model that uses AES and SHA2 (Security Hash Algorithm 2) [16] to keep data secure in the cloud, believing that data breaches occur due to a lack of or ineffective authentication. Kaur and Kaler utilized the MD5 algorithm to produce an OTP, which was then encrypted using the AES algorithm and used to log into a cloud server. Because of the sensitive data hosted in the cloud, they additionally used SHA2 to ensure data integrity. Multiple factors were calculated and evaluated, including processing time, processing cost, AES encryption time, OTP generation, and encryption time. After executing the proposed methods, it has been concluded that the proposed model can improve cloud security. The suggested method minimizes the system's complexity and processing costs, increasing its overall efficiency.

In a study done by Seta H., Wati T., & Kusuma I.C. in 2019 in their paper entitled "Implement Time-Based One-Time Password and Secure Hash Algorithm 1 for Security of Website Login Authentication" [17], the authors employed the use of the Google Authenticator application to authenticate registered users of the developed website. Google Authenticator uses HMAC-SHA-1 algorithm to produce a time-based one-time password. Google Authenticator generates a 6-digit one-time usable code that consists purely of numerals. According to a study done by Pittman and Robinson in 2020 regarding password strength, they defined a weak password as a string of less than 7 characters and a string that consists of only alphabetical or numerical characters [18].

Kurniawan D.E, Iqbal M., Friadi J., Hidayat F., & Permatasari, R.D. created a login system in 2021 using the AES algorithm and the blowfish algorithm in OTP verification code that could be sent via SMS gateway [19]. Their research looked at how AES and blowfish algorithms may improve the security of providing OTP codes by routing them through an encryption-enabled gateway server. They assessed the security performance and compared the performance of the encryption and decryption methods of AES and Blowfish. Both algorithms' performance is assessed by time speed, which they term as a "comparative parameter" that aims to calculate the speed of the OTP encryption, decryption, and Avalanche effect. In several tests, AES exceeds blowfish in terms of encryption and decryption speed, according to their research. The avalanche effect generated by the AES algorithm is superior to that generated by the Blowfish algorithm.

III. EXISTING ALGORITHM

A. *Overview*

TOTP or Time-based One-time Password is a variation of HOTP that was proposed by OATH members in 2008 and was formally accepted in 2011. TOTP is a time-based algorithm that differs from HOTP, which employs the counter as a moving factor. M'Raihi et.al. [7] defined TOTP as

$$TOTP = HOTP(K, T) \qquad (1)$$

, where T is an integer that specifies the number of time steps between $T0$ and the current Unix time.

TOTP is usually partnered with OTP authenticator apps such as Google Authenticator, Authy, etc. and same as the HOTP, it uses a shared secret key. As it is a variation of HOTP, with a time-based moving factor, is also applies cryptographic hash, namely HMAC-SHA1.

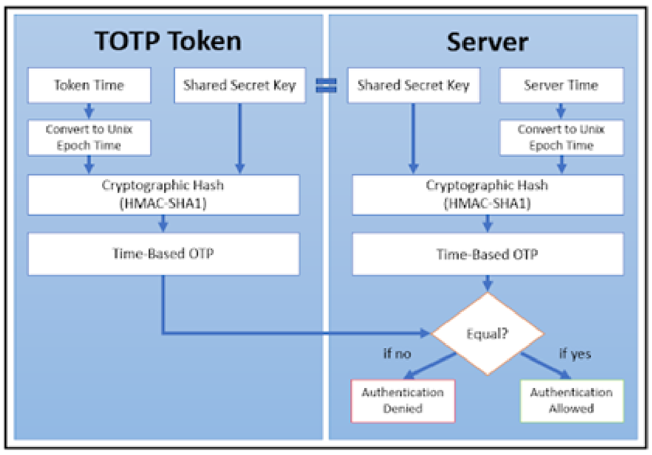

Fig. 1. Time-Based One-time Password Algorithm Overview

B. *The Problem of TOTP Algorithm*

The delivery of the OTP code generated by TOTP can easily be intercepted by an attacker due to the connection between server and client being insecure. According to a paper done by Saxena, et. al. [11], SMS when transferred from one MS to another, it is conveyed as plaintext. This plaintext message can be intercepted by hackers before it reaches the SMS center. Furthermore, due to the poor encryption of SMS using A/51, it is easily broken and intercepted by hackers [12].

C. *Pseudocode of TOTP Algorithm*

```
Generate secret_key
gen_code = secret_key and set interval = 30
convert gen_code to string
If gen_code < 6:
        append "0"
End if
code_sent = "OTP Code: " + gen_code
Send code_sent via SMS
Enter code_sent in the box and substitute to client_input
Try:
        If time interval > 30:
                Show ERROR: OTP code Invalid
        Else:
                If code_sent == client_input:
                        Show SUCCESS: OTP code Valid
                Else:
                        Show ERROR: OTP code Invalid
                End if
        End if
Except:
        Show ERROR: OTP code Invalid
End try
.
```

IV. ENHANCED TOTP ALGORITHM

A. *Enhancement of the Algorithm*

To resolve the problem of security, the generated OTP is encrypted and delivered through SMS using the AES (Advanced Encryption Standard) algorithm.

AES was developed by the National Institute of Standards and Technology (NIST) in 1997 as a replacement for the Data Encryption Standard (DES), which was growing vulnerable to brute-force assaults. AES is a symmetric encryption technique that uses block ciphers to convert plaintext to ciphertext and vice versa using a cipher key. Rounds are used in encryption and decryption, which involves a series of processes such as substitution, transposition, infusion, and addition of round keys to generate the final output of ciphertext and vice versa [20]. Each block cipher encrypts and decrypts data in 128-bit blocks, depending on the length of the cipher key. AES provides three block ciphers based on key length: AES-128, which uses a 128-bit key length and goes through 10 rounds, AES-192, which uses a 192-bit key length and goes through 12 rounds, and AES-256, which goes through 14 rounds.

AES utilizes modes of operation in addition to cipher key length when dealing with plaintext larger than a block size. [Based on the study conducted by Almuhammadi S. & Al-Hejri I. in 2017 [21], there are five primary block cipher modes of operation based on NIST recommendations: Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher FeedBack (CFB), Output FeedBack (OFB), and Counter (CTR). ECB refers to the usage of a symmetric cipher in its most basic form, in which each block is encrypted individually. The most significant disadvantage of ECB mode is that identical plaintext blocks are converted to similar ciphertext blocks. CBC overcomes the ECB disadvantage. CBC reduces the chance of recurring patterns appearing in the ciphertext. But because of its chaining method, CBC mode takes longer to process than ECB mode. The CBC mode, unlike ECB, does not permit parallelism, hence it is not recommended for disk encryption. CFB mode processes data in segments rather than blocks and employs an initial chaining vector (ICV). OFB, like CFB, employs ICV, but it must be unique for each mode execution under the provided key. In the CTR mode, instead of using a shift-register, a counter R with an initial value and a nonce is utilized to produce a key stream. To create the ciphertext, the key stream is XORed with the plaintext. The CTR mode, like the OFB and CFB modes, does not require any padding to match the cipher's block size.

The AES algorithm is not only safe but also quick in terms of encryption and decryption [22]. Salkanovic A., Ljubic S., Stankovic L., & Lerga J. [23] published a comprehensive evaluation of symmetric key encryption algorithms available on the Android operating system in 2021, including DES, 3DES, AES, Blowfish, RC4, and CHACHA20. When evaluating CPU use by measuring the time required to encrypt and decode text files of varied sizes, RC4 and ChaCha20 stream ciphers outperform other encryption algorithms. On both devices, the AES algorithm performed the best in terms of block ciphers. The 3DES cipher performed the poorest, especially on devices with less capable hardware specs.

The method that will be implemented in this research study is AES-128 in the mode CBC since it is faster, more efficient, and less likely to have a full attack designed against it. In the figure 2, the generated OTP will be converted to plaintext by combining the string "unlock_msg#" with the OTP. The plaintext will be transformed into cipher text using AES-128 in CBC mode and a secret cipher key. The cipher text will be kept in the authentication code database with the plaintext, and the first 12 characters of the cipher text will be delivered to the client via SMS gateway.

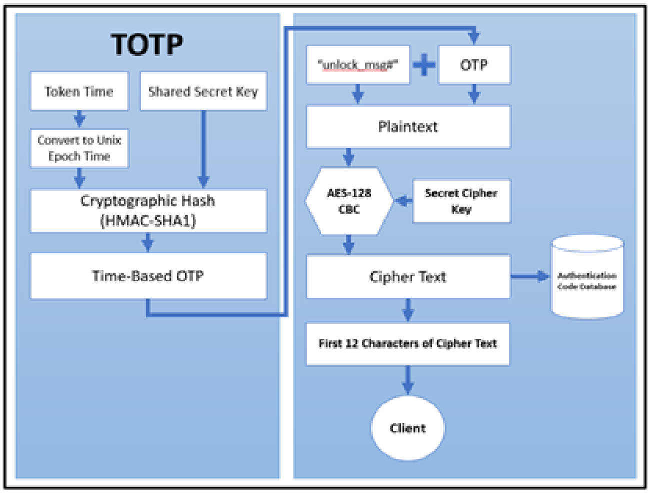
TOTP
Token Time
Shared Secret Key
Convert to Unix Epoch Time
Cryptographic Hash (HMAC-SHA1)
Time-Based OTP
"unlock_msg#"
OTP
Plaintext
AES-128 CBC
Secret Cipher Key
Cipher Text
Authentication Code Database
First 12 Characters of Cipher Text
Client

Fig. 2. Enhanced TOTP Algorithm Overview

B. Pseudocode of the Enhanced Algorithm

```
Generate secret_key
gen_code = secret_key and set interval = 30
convert gen_code to string
If gen_code < 6:
    append "0"
End if
OTP_code = "unlock_msg#" + gen_code
Generate cipher_key
Enter cipher_key to AES.CBC_Mode and substitute to cipher
Convert OTP_code into bytes and substitute to plaintext
Encrypt plaintext with cipher and substitute to ciphertext
Save ciphertext and OTP_code to database
code_sent = "OTP Code: " + first 12 characters of ciphertext
Send code_sent via SMS
Enter code_sent in the box and substitute to client_input
Try:
    If time interval > 30:
        Show ERROR: OTP code Invalid
        Delete ciphertext and OTP_code to database
    Else:
        If code_sent == client_input:
            Search database using client_input and retrieve ciphertext and OTP_code
            Decrypt ciphertext with cipher and substitute to OTP_retrieved
            If OTP_retrieved == OTP_code:
                Show SUCCESS: OTP code Valid
            End if
            Delete ciphertext and OTP to database
        Else:
            Show ERROR: OTP code Invalid
        End if
    End if
Except:
    Show ERROR: OTP code Invalid
End try
```

V. METHODOLOGY

The researchers employed an experimental design to see whether the proposed approach will enhance the original TOTP Algorithm. The baseline will be the original TOTP algorithm without the AES, which will be compared against the proposed enhancement in section IV-B. The baseline and proposed enhancement will be evaluated by looking at the performance time after building an experimental design. Time speed will be a metric for calculating the performance speed of OTP generation.

The researchers will also examine the strength of the OTP generated for secure authentication by performing a password strength and validation. Password Strength will be employed as the testing method. Password Strength is a Python module that can be loaded into source code and used to test the password generated. The library includes numerous testing techniques, including entropy bits, which measure how diverse a password is. In layman's terms, the greater the entropy of a password, the more characters it contains. The

Password Strength library may also measure a password's complexity. The complexity of a password is determined by how difficult it is for a hacker to crack it. The difficulty numbers in the library vary from 0.00 to 0.99, with 0.66 serving as the baseline for a decent and strong password.

VI. RESULTS

The researchers designed a simple two-factor authentication login system based on the proposed enhancement in section IV-B. Login credentials, such as username and password, will be required by the system. The OTP insert box will display before you can successfully log in to the system. Check the mobile phone for an SMS, then input the OTP code into the OTP key insert box. The entered OTP code will be validated by the system, and it will only be used once. The OTP will be invalid after 30 seconds, and the user can request another OTP code.

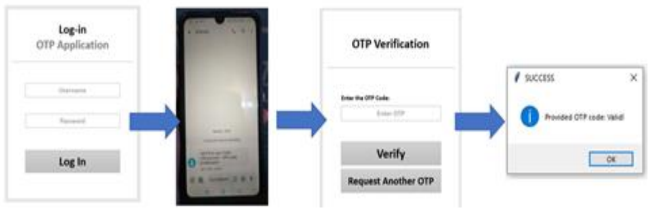

Fig. 3. 2-Factor Authentication Log-in Application

The application is functioning well throughout testing, fulfilling its primary functions of log-in credential validation and OTP verification. If the login credentials or the OTP are wrong, the system will display an error message. The user will receive an SMS after clicking the log-in button, which will be recorded in the database for the verification process.

In addition, the performance of the original TOTP algorithm and the proposed enhancement were compared. Fig. 4 depicts the time speed of algorithms.

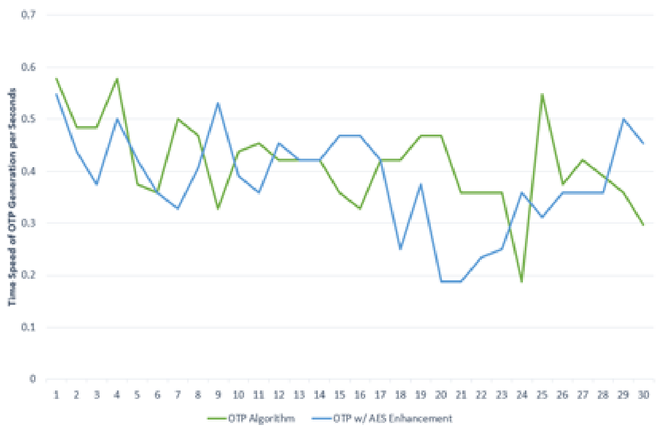

Fig. 4. Time speed of OTP generation

The speed aspect of OTP generation is examined 30 times during performance testing. The original OTP algorithm's average generation speed is 0.414583 seconds, whereas the suggested enhancement is 0.383333 seconds. In terms of OTP generation speed, these values indicate that the suggested enhancement outperforms the original algorithm. The password strength test is shown in Fig. 5.

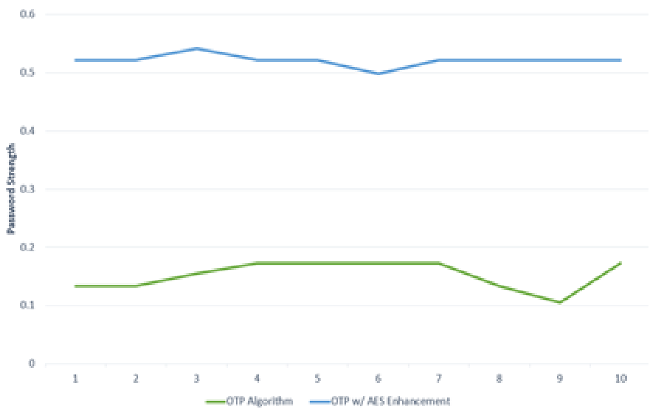

Fig. 5. Password Strength Generation of OTP

During the password strength testing, the strength of generated OTPs is tested 10 times. The average password strength of the original OTP algorithm is 0.152211, whereas the proposed enhancement is 0.521071. These results indicate that the proposed enhancement outperforms the original algorithm in terms of password strength. The suggested enhancement's average is also greater since the generated OTP password is 12 characters long, whereas the original algorithm is just 6 characters long.

VII. CONCLUSION

The proposed enhancement to the 2-factor authentication login system has been implemented successfully. The generated OTP was transmitted through SMS gateway, encrypted using AES-128 in CBC mode. Testing is carried out by comparing the original algorithm and the proposed enhancement in terms of OTP generating speed and password strength. The findings of this study demonstrate that the proposed enhancement outperforms the original algorithms by 0.03125 seconds in terms of performance speed. The proposed enhancement generates a stronger password than the original algorithm by a difference of 0.36886. Although the suggested enhancement's produced OTP falls short of the password strength baseline of 0.66.

ACKNOWLEDGMENT

We would like to express our deepest gratitude to our family, Balasta and Pelito family, for their unwavering support and encouragement when we were doing this study especially during a pandemic.

We would also like to recognize and thank the efforts of our thesis adviser, Sir Mark Christopher R. Blanco, for giving us the necessary tools and advice needed to guide our paper to the right direction and help us to be able to get our desired results. We would like to thank him for accepting us and mentoring us during the creation of this paper.

Finally, we would like to thank God, Our Almighty Father, for giving us the strength and faith to push forth and complete this paper. We would like to express our thanks to Him for guiding us throughout our difficult times and giving us the will and determination needed to finish this study.

REFERENCES

[1]. E. Conrad, S. Misenar, and J. Feldman. **Domain 5**, *Eleventh Hour CISSP®*, pp. 117-134, 2017.

[2]. M. Anderson. **TOTP two-factor authentication (2FA) – pros and cons**, *JumpCloud*, 2020. [Online]. https://jumpcloud.com/blog/totp-2fa-pros-cons

[3]. E. Erdem and M.T. Sandikkaya. **OTPaaS—one time password as a service**, *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 743-756, 2019.

[4]. T. Virtue and j. Rainey. **Privacy and security in healthcare**, *HCISPP Study Guide*, pp. 61-89, 2015.

[5]. D. M'Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen. **RFC 4226: HOTP: An HMAC-Based One-Time Password Algorithm**, *RFC Editor*, December 2005. [Online]. https://www.rfc-editor.org/rfc/rfc4226.html

[6]. K. Brindhashree and S.J. Prakash. **Data security based on steganography and cryptography combined with OTP algorithm and huffman coding in the cloud environment**, *International Research Journal of Modernization in Engineering Technology and Science*, vol. 2, no. 10, 2020.

[7]. D. M'Raihi, S. Machani, M. Pei, and J. Rydell. **TOTP: time-based one-time password algorithm**, *IETF Data Tracker*, May 2011. [Online]. https://datatracker.ietf.org/doc/html/rfc6238

[8]. C.S. Thirumalai and S. Budugutta. **Public key encryption for SAFE transfer of one time password**, *International Journal of Pure and Applied Mathematics*, vol. 118, no. 8, pp. 283-287, 2018.

[9]. A. Khishipah and A.S. Muhammad. **Implementation and comparison of OTP techniques (TOTP,HOTP,CROTP) to prevent replay attack in RADIUS protocol**, *Islamic University Palestine (Gaza Strip)*, vol. 1, no. 1, pp. 1-L, 2014.

[10]. A.R. Reyes and E. Festijo. **Securing one time password (OTP) for multi-factor out-of-band authentication through a 128-bit blowfish algorithm**, *International Journal of Communication Networks and Information Security (IJCNIS)*, vol. 10, no. 1, pp.242-247, 2018.

[11]. N. Saxena, H. Shen, N. Komninos, K.K.R. Choo, and N.S. Chaudhari. **BVPSMS: a batch verification protocol for end-to-end secure SMS for mobile users**, *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 3, pp. 550-565, 2018.

[12]. M.H. AbouSteit, A.F. Tammam, and A. Wahdan. **A novel approach for generating one-time password with secure distribution**, in *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, 2020.

[13]. E.P. Nugroho, Rizky Rachman Judhie Putra, and Iman Muhamad Ramadhan. **SMS authentication code generated by advance encryption standard (AES) 256-bits modification algorithm and one time password (OTP) to activate new applicant account**, in *2016 2nd International Conference on Science in Information Technology (ICSITech)*, pp. 175-180, 2016.

[14]. D. Neal. **AES encryption is cracked**, *The Inquirer*, July 2011. [Online]. http://www.theinquirer.net/inquirer/news/2102435/aes-encryption-cracked

[15]. C. Sudar, S.K. Arjun, and L.R. Deepthi. **Time-based one-time password for wi-fi authentication and security**, in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2017.

[16]. J. Kaur and N. Kaler. **Design and implementation of an OTP-based data security model incorporating AES and SHA2 in cloud environment**, *International Journal of Computers & Technology*, vol. 17, no. 1, pp. 7081-7091, 2018.

[17]. H. Seta, T. Wati, and I.C. Kusuma. **Implement time-based one-time password and secure hash algorithm 1 for security of website login authentication**, in *2019 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS)*, 2019.

[18]. J. Pittman and N. Robinson. **Shades of perception- user factors in identifying password strength**, *ArXiv Computer Science*, 2020.

[19]. D.E. Kurniawan, M. Iqbal, J. Friadi, F. Hidayat, and R.D. Permatasari. **Login security using one time password (OTP) application with encryption algorithm performance**, *Journal of Physics: Conference Series*, vol. 1783, no. 1, pp. 012041, February 2021.

[20]. C. Bernstein and M. Cobb. **Advanced encryption standard (AES)**, *SearchSecurity*, September 2021. [Online]. https://www.techtarget.com/searchsecurity/definition/Advanced-Encryption-Standard

[21]. S. Almuhammadi and I. Al-Hejri. **A comparative analysis of AES common modes of operation**, in *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2017.

[22]. P. Mahajan and A. Sachdeva. **A study of encryption algorithms AES, DES and RSA for security**, *Global Journal of Computer Science and Technology*, vol. 13, no. 15-E, 2013.

[23]. A. Salkanovic, S. Ljubic, L. Stankovic, and J. Lerga. **Analysis of cryptography algorithms implemented in android mobile application**, *Information Technology and Control*, vol. 50, no. 4, pp. 786-807, 2021.