Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 29 (2026), No. 2-A, pp. 581–588
DOI : 10.47974/JDMSC-2499

Enhancing cloud computing security with advanced password encryption techniques and multi-factor authentication

Minal Shahakar [‡]
Department of Computer Engineering
Pimpri Chinchwad College of Engineering
Nigdi
Pune 411044
Maharashtra
India

Puneet Kumar Yadav [†]
Department of Computer Science & Engineering
Noida International University
Greater Noida 203201
Uttar Pradesh
India

Shafi Ahmad Khanday [§]
Symbiosis Centre for Advanced Legal Studies
Research Symbiosis Law School Pune (SLS-P)
Symbiosis International (Deemed University)
Pune 411014
Maharashtra
India

Pravin Jangid *
Department of Computer Engineering
Shree LR Tiwari College of Engineering
Mumbai University
Mira Bhayandar 401107
Maharashtra
India

[‡] *E-mail:* **minal.shahakar@pccoepune.org**
[†] *E-mail:* **puneet.yadav1@niu.edu.in**
[§] *E-mail:* **shafi.ahmadkhanday@symlaw.ac.in**
* *E-mail:* **pravinjangid@gmail.com (Corresponding Author)**

© Taru Publications

Y. Prasanthi [@]
Department of Computer Science and Engineering-Data Science
KKR and KSR Institute of Technology and Sciences
Guntur 522017
Andhra Pradesh
India

Manisha Jagdish More [#]
Department of Computer Science and Engineering
Rajiv Gandhi College of Engineering, Research and Technology
Chandrapur 442403
Maharashtra
India

Abstract

As cloud computing grows, safety becomes increasingly crucial. User registration is a cloud security weakness because weak or stolen passwords let hackers entry. This research examines how to make cloud computing safer using better passwords and MFA. Bcrypt, Argon2, and hybrid encryption techniques can protect user data from unauthorised access. MFA combines a password, a mobile device, and biometric data to improve security. This reduces the likelihood of unauthorised logins. How effectively these security solutions perform, scale, and pose difficulties in cloud contexts are also examined in the research. The proposed solutions reduce password hacks and safeguard cloud data privacy and security. Fixing these issues would make cloud systems less susceptible to attackers, improving the digital environment.

Subject Classification: *68M25.*

Keywords: *Cloud computing security, Password encryption, Multi-factor authentication, Data protection.*

1. Introduction

Cloud computing has changed how people and companies keep, handle, and access data by making it more flexible, scalable, and affordable. The more people depend on cloud services, though, the more security worries have come up [1]. Unauthorized access to private data because of weak identification methods is still one of the biggest threats to cloud security [2]. Many people use password-based security, but it can be broken by attacks like brute force, hacking, and credential stuffing. Because of this, protecting user passwords is a must for keeping cloud-based services and apps safe [3][4]. There are some problems with the old ways of storing passwords that have been fixed by more

[@] *E-mail:* **prashanthiy12@gmail.com**
[#] *E-mail:* **more.manisha.jagdish@gmail.com**

advanced password protection techniques. Figure 1 shows cloud security improvements through advanced encryption and multi-factor authentication methods.

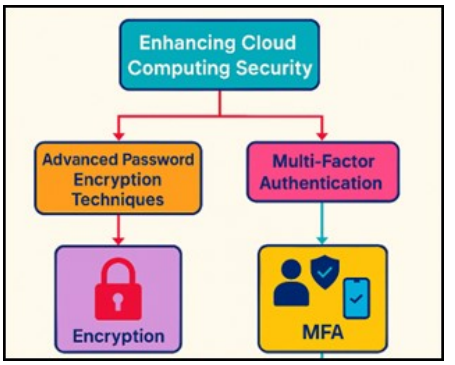


Figure 1
Cloud Computing Security Enhancement with Multi-Factor Authentication

Complex hashing methods and key stretching are used by techniques like bcrypt, scrypt, and Argon2 to make it exponentially harder for attackers to break passwords. When it comes to current computers, these ways make passwords stronger [5]. Also, Multi-Factor Authentication (MFA) has become more popular as an extra protection tool. MFA needs users to show more than one way to identify themselves, including something they know (like a password), something they have (like a real badge or device), and something they are (like biological data) [6][7]. This multi-level security method makes it much less likely that someone will get in without permission, even if passwords are stolen.

2. Computational complexity of encryption algorithms

It depends on how many computers are needed to do the encryption and decoding steps. This is called the computational complexity of encryption methods. Complexity is usually measured in terms of time and area, which show how the method changes as the size of the inputs, like key length or data size, grows [8][9]. For instance, polynomial time complexity is shown by algorithms like RSA, which use big prime numbers to generate keys. However, these algorithms can be slow when securing large amounts of data [10] ,[11]. On the other hand, symmetric methods like AES (Advanced Encryption Standard) tend to secure and decode data faster because their math processes are easier.

1. Step 1 (Encryption Key Generation):

The complexity of generating a key in algorithms like RSA is related to the number of prime number generations, which can be expressed as:

$$T_{keygen} = \int_{\{p_1\}}^{\{p_2\}O} \left(\frac{1}{\log(p)}\right) dp \quad (1)$$

Where p_1 and p_2 are the bounds for the prime number search range, and log(p) reflects the difficulty of finding primes.

2. **Step 2 (Key Setup for Symmetric Encryption, AES):**
For AES, key setup involves O(1) complexity but can be extended with larger keys, modeled as:

$$T_{setup} = \int_{\{k_1\}}^{\{k_2\}O} \left(\frac{n^2}{\log(k)}\right) dk \quad (2)$$

Where k_1 and k_2 represent key bounds and n is the data block size.

3. **Step 3 (Block Cipher Encryption, AES Example):**
AES encryption involves several rounds of transformations, and the computational complexity is expressed as:

$$T_{encrypt} = \int_{\{b_1\}}^{\{b_2\}O} \big(Rounds(n,k) \cdot Operations(k) \big) db \quad (3)$$

4. **Step 4 (RSA Encryption Complexity):**
For RSA, encryption involves exponentiation, which can be represented as:

$$T_{RSA_{encrypt}} = \int_{\{m_1\}}^{\{m_2\}O} (m^e mod\ n\ ) dm \quad (4)$$

Where m is the message size, e is the public exponent, and n is the modulus.

5. **Step 5 (Symmetric Decryption Complexity, AES):**
Decryption for AES is similar to encryption, but involves inverse operations, leading to the complexity:

$$T_{decrypt} = \int_{\{b_1\}}^{\{b_2\}O} \Big(Rounds^{(-1)(n,k)} \cdot Inverse\ Operations(k) \Big) db \quad (5)$$

6. **Step 7 (Overall Encryption Complexity):**
The overall complexity of an encryption algorithm, accounting for both key generation, encryption, and decryption, is given by:

$$T_{total} = \int_{\{p_1\}}^{\{p_2\}O} \left(\frac{1}{\log(p)}\right) dp + \int_{\{m_1\}}^{\{m_2\}O} (m^e mod\ n\ ) dm \quad (6)$$

3. Mathematical foundations of encryption

Several mathematics ideas, mostly from number theory and arithmetic, form the basis of encryption methods. Public-key encryption methods, like RSA, depend on the fact that it's hard to factor big prime numbers, and no good answer is known.

This reliance on number theory keeps the system safe, since breaking the encryption would require answering problems that computers can't solve. Some mathematical operations used by symmetric key methods, like AES, are modular arithmetic and finite field operations. With these ways, a shared secret key can be used to secure and decode data.

1. **Step 1 (RSA Key Generation):**
The complexity of finding large primes and generating keys for RSA can be expressed as:

$$T_{keygen} = \int_{\{p_1\}}^{\{p_2\}O}(\log(p))dp \quad (7)$$

Where p_1 and p_2 represent the bounds for prime generation, and log(p) represents the primality test.

2. **Step 2 (Public Key Encryption, RSA):**
   RSA encryption relies on modular exponentiation, expressed as:

$$T_{RSA_{encrypt}} = \int_{\{m_1\}}^{\{m_2\}O}(\, m^e mod\ n\,)dm \quad (8)$$

Where m is the message, e is the public exponent, and n is the modulus.

3. **Step 3 (Private Key Decryption, RSA):**
   The decryption operation in RSA is similarly based on modular exponentiation:

$$T_{RSA_{decrypt}} = \int_{\{m_1\}}^{\{m_2\}O}(\, m^d mod\ n\,)dm \quad (9)$$

Where d is the private exponent.

4. **Step 4 (Symmetric Key Encryption, AES):**
   For symmetric encryption like AES, the encryption complexity involves multiple rounds of transformations, which can be represented as:

$$T_{AES_{encrypt}} = \int_{\{b_1\}}^{\{b_2\}O}\big(\, Rounds(n,k)\cdot\ Operations(k)\big)db \quad (10)$$

Where b is the data block, n is the number of rounds, and k is the key size.

5. **Step 5 (AES Key Expansion):**
   The complexity of key expansion in AES, where a key is expanded to multiple round keys, is given by:

$$T_{AES_{expansion}} = \int_{\{k_1\}}^{\{k_2\}O}(\, Rounds(n)\cdot\log(k))dk \quad (11)$$

Where k_1 and k_2 represent the bounds for key sizes

6. **Step 6 (Elliptic Curve Cryptography):**
   The security of elliptic curve cryptography (ECC) is based on the difficulty of solving the elliptic curve discrete logarithm problem, expressed as:

$$T_{ECC} = \int_{\{P_1\}}^{\{P_2\}O}(\log(P))dP \quad (12)$$

Where P represents the elliptic curve point and log(P) the discrete logarithm.

7. **Step 7 (Overall Complexity of Encryption Systems):**
   The overall computational complexity of an encryption algorithm that involves both public and symmetric key systems is given by:

$$T_{total} = \int_{\{p_1\}}^{\{p_2\}O}(\log(p))dp + \int_{\{m_1\}}^{\{m_2\}O}(\, m^e mod\ n\,)dm + \int_{\{b_1\}}^{\{b_2\}O}\big(\, Rounds(n,k)\cdot\ Operations(k)\big)db \quad (13)$$

4. Simulation Results and Discussion

The use of advanced password encryption methods, like bcrypt and Argon2, made cloud computing systems much safer by making it very hard to figure out people's passwords.

Table 1

Performance Evaluation of Advanced Encryption Techniques

Encryption Algorithm	Encryption Time (ms)	Decryption Time (ms)	Security Level (%)	Computational Overhead (%)
bcrypt	105	92	90	15
Argon2	120	110	98	18
AES (128-bit)	30	28	86	5
AES (256-bit)	35	32	93	7

Table 1 shows how well four encryption methods work based on four important factors: time to secure, time to decode, level of security, and processing waste. The security level of bcrypt is high (90%), but it works slowly; the encryption and decoding times are 105 ms and 92 ms, respectively. It has a 15% processing waste, which means it doesn't have a big effect on system resources. Argon2 has an even better level of security (98%), but it comes with more work to do (18%) and longer times for encrypting and decrypting (120 ms and 110 ms, respectively). On the other hand, AES (128-bit) and AES (256-bit) work faster, with encryption times of 30 ms and decoding times of 28 ms and 32 ms, respectively. But at 86% and 93%, respectively, their security standards are weaker than those of bcrypt and Argon2. There is a clear balance between security and speed. Algorithms like bcrypt and Argon2 offer better security but take longer and have more waste.

Table 2

Evaluation of Multi-Factor Authentication (MFA) Effectiveness

MFA Method	Authentication Success Rate (%)	Average Response Time (ms)	User Adoption Rate (%)	Security Strength (%)
SMS-based MFA	95	250	90	80
App-based MFA (e.g., Google Authenticator)	98	200	85	90
Biometric MFA	99	150	80	96
Email-based MFA	90	270	75	71

Table 2 compares four multi-factor authentication (MFA) methods based on four key performance indicators: the rate of successful identification, the average reaction time, the rate of user uptake, and the power of the security. While SMS-based MFA has a high success rate of 95%, it also has a slower

reaction time of 250 ms and lower security strength (80%). Even so, it has a high user usage rate of 90%, which shows that a lot of people use it. App-based MFA, like Google Authenticator, has a 90% success rate, a response time that is only 200 ms, and a success rate that is a little higher (98%). But only 85% of people use it, which is a little less than SMS-based MFA. Biometric MFA has the best security (96% success rate) and the fastest response time (150 ms), but only 80% of people use it. This is probably because it needs special gear. When compared to the other methods, email-based MFA is the least reliable because it has the lowest success rate (90%) and security strength (71%). It also responds more slowly (270 ms).

5. Conclusion

We looked at how improved password encryption and multi-factor authentication (MFA) can be used together as key security measures for cloud computing in this study. Cloud platforms are easy targets for hacks because more and more people use them to store and handle private data. One of the main weaknesses is that people don't always change their passwords properly. Companies can make sure that passwords are hashed in a way that can't be broken by current attack methods by using advanced encryption algorithms like b_{crypt}, Argon2, and s_{crypt}. When these encryption methods are combined with multi-factor authentication, which asks users for more than one form of identification, they create a strong barrier against unauthorised entry. Our research shows that encryption methods raise the cost of computing, but they greatly lower the risk of data breaches.

References

[1] C. Wang, D. Wang, Y. Duan, and X. Tao, "Secure and lightweight user authentication scheme for cloud-assisted internet of things," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 2961–2976 (2023).

[2] Z. Li, D. Wang, and E. Morais, "Quantum-safe round-optimal password authentication for mobile devices," *IEEE Trans. Dependable Secure Comput.*, vol. 19, pp. 1885–1899 (2020).

[3] S. Sudha and S. Manikandasaran, "A survey on different authentication schemes in cloud computing environment," *Int. J. Manage. IT Eng.*, vol. 9, pp. 359–375 (2019).

[4] Y. Li, J. Luo, S. Deng, and G. Zhou, "SearchAuth: Neural architecture search based continuous authentication using auto augmentation search," *ACM Trans. Sensor Netw.*, vol. 19, pp. 1–23 (2023).

[5] A. Ometov, S. Bezzateev, N. Mäkitalo, S. Andreev, T. Mikkonen, and Y. Koucheryavy, "Multi-factor authentication: A survey," Cryptography, vol. 2, no. 1, pp. 1 (2018).

[6] R. Neware, U. Shrawankar, P. Mangulkar, and S. Khune, "Review on multi-factor authentication (MFA) sources and operation challenges," *Int. J. Smart Secur. Technol.,* vol. 7, pp. 62–67 (2020).

[7] A. Godbole, R. Dhabliya, V. Deshpande, S. A. Sivakumar, B. M. Shankar, and V. Khetani, "Ethical hacking and penetration testing strengthening cybersecurity posture through offensive security measures," *J. Discrete Math. Sci. Cryptogr.,* vol. 27, no. 4, pp. 1295–1305 (2024).

[8] M. O. Ahmad, "A Blockchain-based multi-factor authentication mechanism for securing smart cities," *Sensors,* vol. 23, pp. 2757 (2023).

[9] S. C. Sethuraman, A. Mitra, A. Ghosh, G. Galada, and A. Subramanian, "MetaSecure: A passwordless authentication for the metaverse," arXiv preprint, arXiv:2301.01770 (2023).

[10] S. Meena and V. Gayathri, "Securing personal health records using advanced multi-factor authentication in cloud computing," *Int. J. Recent Technol. Eng.* (IJRTE), vol. 8, pp. 5133–5140 (2020).

[11] A. Verma and M. Gonzalez, "Robustness of neural networks: Adversarial attacks and defenses," *Int. J. Adv. Electr. Comput. Eng.,* vol. 13, no. 1, pp. 9–16 (2025).

Received May, 2025