

Blockchain-Based Secure Cloud Data Transmission: A Review of DNA Cryptography, Salt-Based Security, and RSA

Megha Sharma^{1 a)} and Neha Dutta^{2 b)}

¹ Computer Science & Engineering, University of Institute & Engineering, Chandigarh University, Mohali-140413, Punjab, India

² Computer Science & Engineering, University of Institute & Engineering, Chandigarh University, Mohali-140413, Punjab, India

^{a)} Corresponding author: megha3210@gmail.com

^{b)} Neha.e12830@cumail.in

Abstract: Cloud computing has become an essential platform for storing, processing, and transmitting digital information. However, cloud environments continue to face challenges involving data confidentiality, integrity, authentication, key management, and trust. Blockchain can support tamper-evident records through cryptographic hash chaining, while conventional cryptographic mechanisms such as RSA and symmetric encryption provide complementary security functions. DNA-inspired cryptography has also attracted research interest through nucleotide-based encoding, sequence transformations, and hybrid encryption architectures. This review critically examines the potential integration of DNA-based encoding and encryption, salt-assisted key derivation, RSA-based key protection, and blockchain block chaining for secure cloud data transmission. It distinguishes DNA encoding from cryptographic protection and examines the roles of salts, encryption keys, cryptographic hashes, and digital signatures. A comparative evaluation framework is developed around key size, execution time, storage overhead, reliability, throughput, and resistance to relevant attacks. The review identifies challenges including inconsistent evaluation practices, unclear separation of encoding and encryption, inadequate threat modelling, key-management complexity, and limited reproducibility. It proposes a structured framework for evaluating Modified DNA and DNA Salt approaches against conventional cryptographic baselines. The review provides a foundation for future experimental studies while emphasizing that security improvements must be established through rigorous cryptanalysis and reproducible testing rather than encoding complexity or key-space estimates alone.

Keywords: Blockchain security; DNA cryptography; cloud computing; salt-assisted key derivation; secure data transmission.

1. Introduction

Cloud computing enables organizations and individuals to access computing, storage, and network resources on demand. Its scalability and flexibility have encouraged the storage and processing of increasingly sensitive information in remote infrastructure. Nevertheless, transmitting and storing data through cloud platforms creates security challenges involving unauthorized disclosure, data modification, identity verification, key compromise, and trust in service providers. Encryption can protect data confidentiality, but it must be supported by sound key management, integrity verification, and secure communication protocols [1–4].

Blockchain provides a distributed data structure in which records can be linked using cryptographic hashes. In a typical hash-chained structure, each block contains a reference to the preceding block, so alteration of an earlier record changes subsequent hash relationships. This structure can make unauthorized modifications detectable when the chain is verified. However, blockchain does not

automatically encrypt data stored in blocks, guarantee that original information is correct, or prevent every attack on the underlying storage and network [5–9].

Conventional cryptographic algorithms remain essential components of secure cloud architecture. RSA can support public-key operations, including key establishment or protection of symmetric keys, while efficient symmetric encryption can protect large data objects. Salt values are used in appropriate key-derivation and password-hashing constructions to reduce the effectiveness of precomputed attacks. Their benefits depend on how they are generated and used; a salt is not, by itself, an encryption algorithm or a substitute for a secret key [10–14].

DNA-inspired cryptography represents another direction in information security research. Computational DNA cryptography may map binary data into nucleotide symbols—adenine (A), cytosine (C), guanine (G), and thymine (T)—and may use sequence permutations, complement operations, or key-dependent transformations. These transformations can form part of a cryptographic construction, but nucleotide representation alone does not establish confidentiality or resistance to cryptanalysis [15–20].

2. Literature Review

A simplified block identifier can be represented as $H_i = H(D_i || H_{(i-1)} || M_i)$, where H is a cryptographic hash function, D_i is the block data, $H_{(i-1)}$ is the previous block hash, M_i is relevant metadata, and $||$ denotes concatenation. The exact serialization and hash input must be defined. Hash chaining can make modifications detectable when the chain is verified, but it does not independently provide confidentiality, authenticity, availability, or consensus. Depending on the application, stronger protection may require authenticated encryption, digital signatures, access controls, and a suitable consensus protocol [5–9].

DNA-inspired computational schemes often map pairs of binary bits to nucleotide symbols. A scheme may also apply secret-key-dependent permutations, substitution, diffusion, or other transformations. Three concepts should be distinguished: DNA encoding (a representation change), DNA-inspired transformation (operations based on nucleotide rules), and DNA-based encryption (a defined cryptographic construction intended to resist a specified threat model). Encoding alone is not encryption. Security claims require a precise algorithm, key generation and management, correct decryption, and analysis against relevant attacks. The size of a nucleotide sequence does not by itself establish effective secret-key entropy or cryptographic strength [15–20].

A salt is generally a non-secret value used as an input to password hashing or a key-derivation function. A conceptual expression is $K = \text{KDF}(P, S, \theta)$, where P is a secret input, S is the salt, and θ represents algorithm parameters. Salt helps prevent identical passwords from producing identical derived values and reduces the usefulness of precomputed tables. It does not create entropy in a weak password or make an insecure cipher secure. The KDF, parameters, salt generation, storage, and transmission must be specified [10–12].

RSA is a public-key cryptographic algorithm. Modern hybrid systems generally use public-key cryptography to protect or establish a symmetric data-encryption key rather than encrypting large files directly. The term “RSA+Salt” is not a complete cryptographic specification. The design must state whether salt is used in password-based derivation or another justified step, identify the RSA padding scheme and key size, and specify the data-encryption and authentication mechanisms [10, 13, 14].

Confidentiality, integrity, authenticity, and availability are distinct properties. Encryption without authentication may fail to detect ciphertext modification. A secure block-transmission design should

explain sender authentication, block-order verification, replay handling, transmission failures, and data recovery. Reliability metrics should match the application and should not be treated as a substitute for cryptographic analysis.

Research has investigated DNA-inspired approaches as alternative or complementary data-protection mechanisms. Sohal and Sharma's BDNA approach describes client-side encryption for cloud data and compares its performance with conventional symmetric algorithms [15]. Other work has combined DNA cryptography and steganography for cloud-based IoT infrastructure [16]. 2023 review surveys genetics-inspired cryptographic algorithms for cloud security [17], while a 2025 review discusses pseudo-DNA and natural DNA cryptography and highlights concerns involving measurement, standardization, robustness, and operability [18]. A 2025 cloud-based DNA cryptography article and a 2026 systematic review further demonstrate the continuing activity in this field [19, 20]. These works mean that a new review should make a narrower contribution than a general survey of DNA cryptography.

Hybrid cryptography combines mechanisms with complementary functions. Symmetric encryption is suited to bulk data; public-key schemes support key establishment or protection; hashes and digital signatures support integrity and authenticity when correctly applied. DNA-inspired transformations can be assessed as additional processing steps, but their added security benefit must be demonstrated independently. Likewise, salt should be evaluated according to its specified use in a standardized derivation process rather than assumed to strengthen arbitrary ciphertext [10–14].

Surveys of blockchain-cloud integration identify potential applications in trust management, data integrity, access control, and auditability, alongside challenges involving scalability, privacy, and performance [6–9]. For large cloud objects, storing encrypted data off-chain while placing hashes, object references, or integrity commitments on-chain can reduce ledger burden. However, the architecture must avoid exposing sensitive plaintext or metadata on a public ledger and must specify how keys, permissions, and revocation are managed.

The literature suggests that DNA-inspired methods, conventional cryptography, and blockchain may serve complementary roles, but combining them does not guarantee stronger security or better performance. The unresolved issue is a consistently defined comparison that isolates the contribution of DNA transformations, salt-assisted key derivation, public-key operations, and blockchain integrity processing.

The reviewed literature suggests that DNA-inspired methods, conventional cryptography, and blockchain can serve complementary roles, but their combination does not automatically guarantee stronger security or better performance. A fair comparison must isolate the contributions of DNA transformations, salt-assisted key derivation, public-key operations, and blockchain integrity processing.

3. Review Methodology and Proposed Evaluation Framework

A transparent review methodology is necessary to reduce selection bias and support reproducibility. The search should cover IEEE Xplore, Scopus, Web of Science, ScienceDirect, SpringerLink, and the ACM Digital Library, subject to access. The search period can cover January 2010 through the final search date, while retaining earlier foundational publications. Search strings should combine terms such as "DNA cryptography" OR "DNA encryption"; "blockchain" OR "hash chaining"; "cloud security" OR "cloud data transmission"; "salt" OR "key derivation" OR "RSA"; and "execution time" OR "throughput" OR "storage overhead".

Include peer-reviewed journal and conference papers that describe a relevant cryptographic mechanism, blockchain-integrity architecture, or measurable evaluation of DNA-inspired encryption. Include standards and foundational cryptographic publications when needed to establish correct terminology. Exclude biological DNA studies without an information-security application, duplicate publications, and studies with insufficient algorithmic or methodological detail.

For each study, record author/year, construction, application domain, key-management method, integrity mechanism, threat model, metrics, limitations, and reproducibility. Report actual numbers identified, screened, excluded, and included only after the searches have been conducted and shown in table 1. PRISMA 2020 can inform transparent reporting, adapted to the scope of a computer-science review [21].

Table 1. Functional comparison of cryptographic and blockchain security approaches

Approach	Primary function	Potential benefit	Main evaluation requirement
Symmetric authenticated encryption	Confidentiality and integrity	Efficient bulk-data protection	Measure processing time, key management, and authentication-tag verification
RSA-based hybrid encryption	Key protection or establishment	Public-key-based key distribution	Specify padding, key handling, and encapsulation overhead
DNA encoding	Data representation	Alternative symbolic representation	Measure encoding overhead; do not treat encoding alone as encryption
Modified DNA encryption	Proposed confidentiality mechanism	Potential key-dependent transformations	Specify threat model and perform independent cryptanalysis
DNA + Salt	DNA processing with a defined salt/KDF integration	Potentially sound password-derived key handling if correctly designed	Specify KDF, key entropy, salt handling, and overhead
RSA + Salt with hash chaining	Key protection, derivation, and integrity linkage	Separation of key and integrity functions	Validate construction, authentication, and end-to-end cost
DNA + Salt with block chaining	Proposed block-protection architecture	Allows measurement of transformation and integrity costs	Use matched baselines, attack testing, and reproducible methods

The following workflow is a proposed framework for future implementation and evaluation; it is not a report of completed experimental validation and comparative evaluation shown in table 2.

1. Input and block preparation: divide the object into blocks, assign identifiers, and define unambiguous serialization.
2. Key management: generate cryptographic keys using a secure random source; use a standardized KDF when password-derived keys are required; use RSA-based key protection only where appropriate.
3. Data protection: apply a fully specified encryption construction; document the exact DNA mapping and key-dependent operations in Modified DNA and DNA + Salt.
4. Integrity and chaining: hash defined block contents and metadata, link blocks to preceding hashes, and use authenticated encryption or signatures as required.
5. Cloud transmission and verification: transmit protected blocks, verify order and integrity, decrypt, and measure successful recovery under defined failures and attacks.

Table 2. Metrics and operational definitions for comparative evaluation

Parameter	Operational definition	Measurement
Key size	Length of each cryptographic key; report each key type separately	Bits
Execution time	Key derivation, encryption, decryption, verification, and total processing	Milliseconds or seconds
Space used	Ciphertext, salt, nonce, tag, metadata, hashes, and relevant ledger overhead	Bytes and percentage overhead
Reliability	Proportion of valid transmissions recovered and verified correctly under defined tests	Percentage
Throughput	Successfully processed data per unit time	MB/s or transactions/s
Integrity	Ability to detect unauthorized modification of data or block references	Detection rate under defined tests
Security	Resistance to specified attacks under documented assumptions	Formal and empirical analysis
Scalability	Performance as file size, block count, and concurrency increase	Time, throughput, and storage curves

4. Experimental Setup and Results

The source manuscript does not report a completed implementation, hardware configuration, dataset, measured runtime, or numerical experimental results. Accordingly, this section specifies a reproducible evaluation protocol and identifies the results that should be reported after experiments are conducted. No performance values or rankings are inferred or fabricated.

Compare a standardized authenticated symmetric-encryption baseline (for example, AES-GCM), an RSA-based hybrid key-protection architecture, Modified DNA, DNA + Salt, RSA + Salt with hash chaining, and DNA + Salt with hash chaining, where each implementation is fully specified and available.

Use identical payload sizes, datasets, hardware, software environment, and repeated trials across approaches. Record processor, memory, operating system, programming language, library versions, block size, key sizes, KDF parameters, salt generation, and blockchain configuration.

Separate key generation, key derivation, encryption, decryption, network transfer, integrity verification, and blockchain processing so their costs can be attributed correctly.

Test ciphertext modification, invalid keys, corrupted or reordered blocks, replay attempts, and documented key-compromise scenarios. Define the success criterion for each test before collecting results.

Calculate storage overhead from all transmitted and stored fields, including salt, nonce, authentication tag, metadata, hashes, and any ledger references. Report throughput with an explicit unit and payload size. Present encryption, decryption, verification, and end-to-end latency separately. Compare integrity-detection results against a defined set of tampering and reordering tests. Do not interpret a successful integrity check as proof of confidentiality or overall cryptographic security.

5. Discussion and Conclusion

The integration of DNA-inspired transformations with established cryptographic techniques is a potentially useful research direction, but its value depends on demonstrable benefits. An additional transformation can increase execution time and storage without improving security if it contributes no independently justified cryptographic property. Future work should specify the Modified DNA mapping and encryption procedures, including key generation, initialization values, sequence operations, and decryption. The DNA+Salt design should clarify whether salt is used in a standardized KDF. Security analysis should address relevant known-plaintext, chosen-plaintext, statistical, key-recovery, ciphertext-manipulation, and replay threats.

Comparison with standardized baselines is essential. AES-GCM or another suitable authenticated-encryption construction can provide a bulk-data reference, while RSA-based hybrid encryption can be assessed for key-management overhead. The blockchain component should be measured separately so the cost of hashing, metadata, storage, and consensus is not incorrectly attributed to the encryption algorithm. If the target journal expects social-science relevance, implications such as personal-information protection, accountability, trust in service providers, and deployment cost should be discussed using appropriate evidence. Encoding versus encryption: DNA representation and sequence manipulation do not necessarily create a secure cipher. Secret-key-dependent operations and attack resistance must be demonstrated. Unclear salt function: Studies should document salt generation, storage, transmission, and its exact use in a KDF or other construction. Block integrity versus confidentiality: Hash chaining may detect modifications but does not replace authenticated encryption, signatures, or access control.

Inconsistent performance evaluation: Results are comparable only when implementation details, workloads, hardware, and measurement boundaries are disclosed. Limited reproducibility and threat modelling: Algorithm details and attacker capabilities must be stated clearly enough to permit replication and meaningful security assessment. Input and block preparation: divide the object into blocks, assign identifiers, and define unambiguous serialization. Key management: generate cryptographic keys using a secure random source; use a standardized KDF when password-derived keys are required; use RSA-based key protection only where appropriate. Data protection: apply a fully specified encryption construction; document the exact DNA mapping and key-dependent operations in Modified DNA and DNA+Salt. Integrity and chaining: hash defined block contents and metadata, link blocks to preceding hashes, and use authenticated encryption or signatures as required. Cloud transmission and verification: transmit protected blocks, verify order and integrity, decrypt, and measure successful recovery under defined failures and attacks.

This framework is a basis for later experimental work; it does not establish that either proposed DNA approach is more secure than standardized cryptography. The main open question is whether the specified DNA-based processing contributes a measurable security or practical benefit beyond

established cryptographic mechanisms. Added transformations may increase execution time and storage without improving security unless their properties are independently justified. Conclusions about superiority must therefore be based on explicit threat models, cryptanalysis, and reproducible experiments.

This review examines the potential integration of DNA-inspired encoding and encryption, salt-assisted key derivation, RSA-based key protection, and blockchain hash chaining for secure cloud data transmission. These mechanisms have different purposes and their combination does not automatically establish stronger security. DNA encoding must be distinguished from a cryptographic cipher, salt must be used according to a defined construction, and blockchain integrity must be separated from confidentiality and authenticity. A comparative framework has been developed around key size, execution time, storage overhead, reliability, throughput, integrity, and attack resistance. Future research should provide complete algorithm specifications, explicit threat models, matched test conditions, and independent security analysis to establish whether Modified DNA and DNA+Salt architectures offer measurable practical advantages.

Because no completed experimental results are included in the source manuscript, the present work should be described as a critical review and proposed evaluation framework. Any future claim that Modified DNA or DNA + Salt is more secure, faster, or more reliable than a conventional baseline must be supported by measured results and independent security analysis.

References

- [1] Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. NIST Special Publication 800-145. National Institute of Standards and Technology. <https://doi.org/10.6028/NIST.SP.800-145>
- [2] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., & Zaharia, M. (2010). A view of cloud computing. *Communications of the ACM*, 53(4), 50–58. <https://doi.org/10.1145/1721654.1721672>
- [3] Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6), 599–616. <https://doi.org/10.1016/j.future.2008.12.001>
- [4] Kaaniche, N., & Laurent, M. (2017). Data security and privacy preservation in cloud storage environments based on cryptographic mechanisms. *Computer Communications*, 111, 120–141. <https://doi.org/10.1016/j.comcom.2017.07.006>
- [5] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. <https://bitcoin.org/bitcoin.pdf>
- [6] Zou, J., et al. (2021). Integrated blockchain and cloud computing systems: A systematic survey, solutions, and challenges. *ACM Computing Surveys*, 54(8), Article 160. <https://doi.org/10.1145/3456628>
- [7] Casino, F., Dasaklis, T. K., & Patsakis, C. (2019). A systematic literature review of blockchain-based applications: Current status, classification and open issues. *Telematics and Informatics*, 36, 55–81. <https://doi.org/10.1016/j.tele.2018.11.006>
- [8] Christidis, K., & Devetsikiotis, M. (2016). Blockchains and smart contracts for the Internet of Things. *IEEE Access*, 4, 2292–2303. <https://doi.org/10.1109/ACCESS.2016.2566339>
- [9] Uddin, M., Khalique, A., Jumani, A. K., Ullah, S. S., & Hussain, S. (2021). Next-generation blockchain-enabled virtualized cloud security solutions: Review and open challenges. *Electronics*, 10(20), 2493. <https://doi.org/10.3390/electronics10202493>
- [10] National Institute of Standards and Technology. (2020). Recommendation for key derivation using pseudorandom functions (NIST SP 800-108 Rev. 1). <https://doi.org/10.6028/NIST.SP.800-108r1>
- [11] Turan, M. S., Barker, E., Burr, W., & Chen, L. (2010). Recommendation for password-based key derivation: Part 1: Storage applications (NIST SP 800-132). <https://doi.org/10.6028/NIST.SP.800-132>

- [12] Biryukov, A., Dinu, D., & Khovratovich, D. (2016). Argon2: New generation of memory-hard functions for password hashing and other applications. 2016 IEEE European Symposium on Security and Privacy, 292–302. <https://doi.org/10.1109/EuroSP.2016.31>
- [13] Moriarty, K., Kaliski, B., Jonsson, J., & Rusch, A. (2016). PKCS #1: RSA cryptography specifications version 2.2 (RFC 8017). Internet Engineering Task Force. <https://doi.org/10.17487/RFC8017>
- [14] National Institute of Standards and Technology. (2007). Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC (NIST SP 800-38D). <https://doi.org/10.6028/NIST.SP.800-38D>
- [15] Sohal, M., & Sharma, S. (2022). BDNA—A DNA inspired symmetric key cryptographic technique to secure cloud computing. *Journal of King Saud University—Computer and Information Sciences*, 34(1), 1417–1425. <https://doi.org/10.1016/j.jksuci.2018.09.024>
- [16] A secure cryptosystem using DNA cryptography and DNA steganography for the cloud-based IoT infrastructure. (2022). *Computers & Electrical Engineering*, 104, 108426. <https://doi.org/10.1016/j.compeleceng.2022.108426>
- [17] Can, O., Thabit, F., Aljahdali, A. O., Al-Homdy, S., & Alkhzaimi, H. A. (2023). A comprehensive literature of genetics cryptographic algorithms for data security in cloud computing. *Cybernetics and Systems*, 56(5), 413–447. <https://doi.org/10.1080/01969722.2023.2175117>
- [18] A Review of DNA Cryptography. (2025). *Intelligent Computing*, 4. <https://doi.org/10.34133/icomputing.0106>
- [19] Kairi, A., & Bhadra, T. (2025). A cloud-based DNA cryptography (CBDC) for enhanced data protection. *Security and Privacy*. <https://doi.org/10.1002/spy2.70126>
- [20] Ahamed, N. N., & Murugan, T. (2026). Research advances in DNA cryptography: Integration with cloud computing, internet of things, and DNA computing. *Discover Applied Sciences*, 8, Article 624. <https://doi.org/10.1007/s42452-026-08554-9>
- [21] Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., et al. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ*, 372, n71. <https://doi.org/10.1136/bmj.n71>
- [22] Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2), 120–126. <https://doi.org/10.1145/359340.359342>
- [23] Daemen, J., & Rijmen, V. (2002). The design of Rijndael: AES—The advanced encryption standard. Springer. <https://doi.org/10.1007/978-3-662-04722-4>
- [24] National Institute of Standards and Technology. (2023). Advanced Encryption Standard (AES) (FIPS PUB 197, updated). <https://doi.org/10.6028/NIST.FIPS.197-upd1>
- [25] National Institute of Standards and Technology. (2015). Secure Hash Standard (SHS) (FIPS PUB 180-4). <https://doi.org/10.6028/NIST.FIPS.180-4>
- [26] National Institute of Standards and Technology. (2015). SHA-3 standard: Permutation-based hash and extendable-output functions (FIPS PUB 202). <https://doi.org/10.6028/NIST.FIPS.202>
- [27] Mathur, G., Pandey, A., & Goyal, S. (2023). A review on blockchain for DNA sequence: Security issues, application in DNA classification, challenges and future trends. *Multimedia Tools and Applications*. <https://doi.org/10.1007/s11042-023-15857-1>
- [28] Zheng, Z., Xie, S., Dai, H.-N., Chen, X., & Wang, H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. 2017 IEEE International Congress on Big Data, 557–564. <https://doi.org/10.1109/BigDataCongress.2017.85>
- [29] Li, X., Jiang, P., Chen, T., Luo, X., & Wen, Q. (2020). A survey on the security of blockchain systems. *Future Generation Computer Systems*, 107, 841–853. <https://doi.org/10.1016/j.future.2017.08.020>
- [30] Zou, W., Lo, D., Kochhar, P. S., Le, X.-B. D., Xia, X., Feng, Y., Chen, Z., & Xu, B. (2019). Smart contract development: Challenges and opportunities. *IEEE Transactions on Software Engineering*, 47(10), 2084–2106. <https://doi.org/10.1109/TSE.2019.2942301>
- [31] Shamir, A. (1979). How to share a secret. *Communications of the ACM*, 22(11), 612–613. <https://doi.org/10.1145/359168.359176>
- [32] Diffie, W., & Hellman, M. E. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6), 644–654. <https://doi.org/10.1109/TIT.1976.1055638>

