INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL ARTICLE

DTAAE: Data-Type-Aware Adaptive Encryption for Cloud Databases Using Sensitivity-Guided Attribute Packing

L Leelavathy[1]*

1 Assistant Professor, Department of Computer Science, Bishop Heber College (Autonomous), (Affiliated to Bharathidasan University), Tiruchirapalli -17, Tamil Nadu, India

OPEN ACCESS

Received: 20/10/2026

Accepted: 03/11/2026

Published: 21/11/2026

Citation: Leelavathy L (2026) DTAAE: Data-Type-Aware Adaptive Encryption for Cloud Databases Using Sensitivity-Guided Attribute Packing. Indian Journal of Science and Technology 19(41): 2817-2823. https://doi.org/10.17485/IJST/v19i41.1375

* **Corresponding author**.

leelakarthikeyen@gmail.com

Funding: None

Competing Interests: None

Copyright: © 2026 Leelavathy. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: To assess the storage, client-side preparation and indexed-retrieval trade-offs of packing heterogeneous sensitive database fields under a single authenticated encryption envelope. **Method**: A type-preserving JSON and AES-256-GCM prototype was evaluated on synthetic SQLite datasets of 2,000, 8,000 and 16,000 rows against plaintext, full-row and four-field protection. The submitted four-trial data were retained; a separate 12-trial replication repeated all configurations. Paired Wilcoxon tests and exploratory bootstrap intervals assessed within-host variation. Additional negative tests examined modification, substitution and replay. **Findings**: In the submitted 16,000-row experiment, packed and field-wise storage allocated 2,152 and 3,488 KiB, respectively (38.3% less); protection preparation medians were 190.55 versus 439.32 ms, and filtered 40-row retrieval was 0.46 versus 0.94 ms. A separate 12-trial replication preserved the same storage allocations but had different absolute timings: 63.37 versus 171.85 ms for preparation and 0.160 versus 0.462 ms for lookup (packed versus field-wise). Median paired reductions were 63.5% and 65.5%; exploratory one-sided Wilcoxon p=0.000244 for each comparison. All 2,003 supplementary records round-tripped; five tampering categories each failed in 1,000 tests, but an authentic earlier record replay remained acceptable. None of these findings establishes cloud-scale performance or a new cryptographic security theorem. **Novelty**: Sensitivity-driven group serialization reduces repeated AES-GCM envelope overhead while authenticating public query metadata; the contribution is an evaluated database layout, not a new encryption primitive.

Keywords: Cloud database; Adaptive encryption; Authenticated encryption; Sensitivity policy; Attribute packing

1 Introduction

Cloud database outsourcing creates a trade-off between protecting confidential fields and supporting efficient queries. A useful design distinction is the granularity of client-side authenticated encryption: encrypting each sensitive field separately enables selective field updates, while encrypting a group can amortize nonce and tag overhead. Neither strategy permits server-side searches on arbitrarily encrypted sensitive values. This study examines only index-assisted filtering on fields deliberately classified as public; thus, the privacy exposure of those fields and their access patterns must be included in the threat model.

Encrypted SQL surveys categorize systems by functionality and leakage[1]. Trusted execution environment (TEE) proposals address richer trusted computation[2], whereas maintenance-aware databases[3] and GaussDB column protection[4] expose operational integration challenges. Searchable encryption addresses protected search at the cost of distinct leakage and index assumptions[5]. Recent data-center adaptive encryption[6] and attribute-based encryption (ABE) systems[7],[8],[9],[10] address either alternative cryptographic processing, authorization or encrypted search. They are relevant but not functionally equivalent benchmarks for a client-side SQLite layout. Section 1.2 and Table 1 provide a prioritized comparison of ten studies, with different threat models and capabilities explicitly separated.

Leelavathy and Sugumar previously investigated DNA-based data protection for cloud storage, including their EUDSFDNA framework and SNDDNA numerical transformation[11],[12],[13],[14]. The present study does not repeat DNA transformations. It instead asks whether an explicitly specified group-encryption layout can reduce overhead in heterogeneous numerical, textual and mixed records while retaining an indexed query over designated public labels.

1.1 Research gap and objectives

A narrowly defined gap remains in the reproducible, like-for-like comparison of four independently authenticated sensitive fields versus the same fields packed into one authenticated row envelope, with identical AES-GCM construction, client-side trust boundary and public index. RQ1 compares physical database allocation; RQ2 compares protection preparation and filtered retrieval; RQ3 examines typed restoration and authentication of associated metadata. The result is a measured implementation trade-off, not proof of first-in-class cryptography or superiority to TEE, ABE or searchable encryption systems.

1.2 Critical comparison with ten relevant studies

Table 1 identifies the primary capability and an important limitation or incompatibility for ten related studies[1],[2],[3],[4],[5],[6],[7],[8],[9],[10]. The closest application comparison is field-wise authenticated encryption with the same public index, because it holds the cipher, query predicate and key trust assumptions constant. Results published for ABE, trusted DBMSs or compression-enhanced systems cannot be converted into direct speed ratios without reproducing their workload and implementation. DTAE-Pack deliberately does not implement fuzzy encrypted search, policy-based key revocation, enclave processing or encrypted aggregation.

2 Methodology

2.1 Trust boundary and supported data types

A trusted client holds a 256-bit AES-GCM key and encrypts prior to storage. An unauthorized database observer can view record IDs, the labels region and segment, record sizes, query counts and access patterns, and can modify, substitute, delete or replay database rows; the observer cannot initially read the client key. The record schema consists of rid (integer), region and segment (public categorical strings), balance_cents (signed integer), customer_name, account_ref and description (Unicode strings). The last four fields are sensitive. Serialization uses UTF-8 JSON in a prescribed field order and retains the tested integer/string types; it does not support general SQL NULL, binary, decimal or timestamp semantics. AES-GCM supplies confidentiality and authentication under its required unique-nonce and uncompromised-key assumptions[15],[16].

2.2 DTAE-Pack construction and overhead

For record i, the client serializes [balance_cents, customer_name, account_ref, description], creates an independent 96-bit nonce and uses AES-256-GCM to encrypt and authenticate the serialized bytes. Additional authenticated data (AAD) binds protocol version, row identifier, public region and segment, and the ordered schema identity. The database stores rid, the two indexed public labels, and nonce || ciphertext || 128-bit GCM tag. Retrieval uses the (region, segment) index, reconstructs AAD, verifies the tag, and only then deserializes typed fields. The algorithm is summarized in Figure 1.

Table 1. Critical comparison of ten prioritized studies

Study	Primary focus	Relevant advance	Reason not a direct baseline
Sun *et al.*, 2024 [1]	Encrypted SQL survey	Organizes function/leakage trade-offs	No executable like-for-like group-AEAD comparator
Lutsch *et al.*, 2025 [2]	Trusted cloud DBMS/TEE	Addresses richer trusted processing	Requires specialized trust/hardware assumptions
Li *et al.*, 2023 [3]	Maintainable secure DBMS	Operational maintenance of protected databases	Different deployment and update semantics
Zhu *et al.*, 2021 [4]	GaussDB end-to-end encryption	Practical database-integrated protection	Not a client SQLite test of envelope count
Sharma, 2023 [5]	Searchable encryption survey	Search/leakage catalog	No numeric benchmark for this record format
Zhang *et al.*, 2025 [6]	Data-center adaptive cryptography	Cross-mapping fusion and AES variants	Different workload and cryptographic design
Routray/Bera, 2026 [7]	Dynamic policy ABE	Context-aware access control	Not field/row AES-GCM grouping
Naikwade *et al.*, 2026 [8]	Revocable ABE + image compression	Access revocation and storage efficiency	Image/ABE costs not comparable to SQL rows
Shu *et al.*, 2026 [9]	Fuzzy searchable ABE	Ranked encrypted keyword search	Extra query capability absent in DTAE-Pack
Meng/Cheng, 2025 [10]	Traceable cloud ABE	Accountable key and audit design	Different adversary and key infrastructure

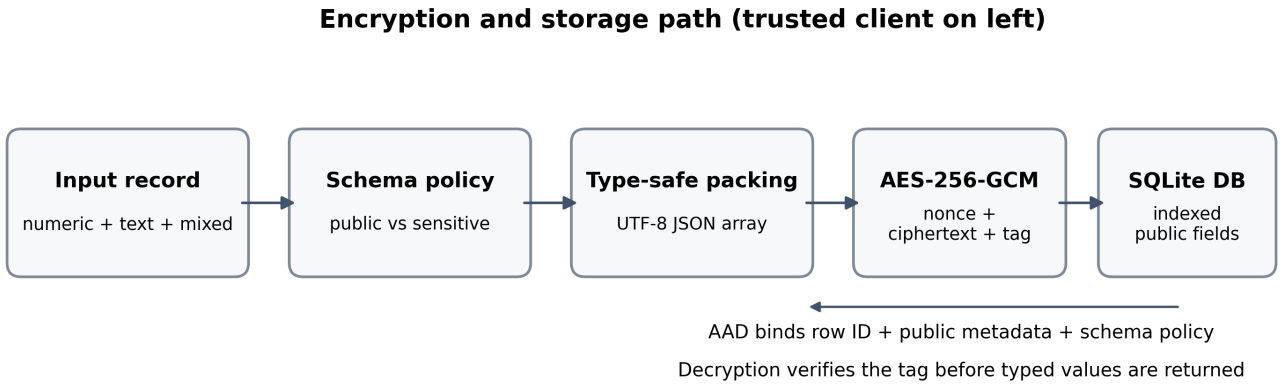

Fig 1. Trusted client classifies fields; only designated public labels remain indexed. Client-managed AES-256-GCM authenticates the packed sensitive tuple and associated metadata

Ignoring serialization and SQL-page overhead, a group of sensitive fields serialized to L bytes requires $L + 12 + 16 = L + 28$ bytes. Four separately encrypted fields incur $4 \times (12 + 16) = 112$ bytes of fixed nonce/tag overhead. Thus, the design removes 84 bytes of fixed envelope overhead per four-field row before accounting for variations in JSON and table representation. Grouping does not strengthen AES-GCM or conceal public labels. A change to a single sensitive field also requires re-encryption of the entire group, unlike separately encrypted fields.

2.3 Algorithms and reproducibility

Algorithm 1 (protect and store):
(1) Classify region/segment as public and the four typed attributes as sensitive;
(2) Create a standard ordered JSON byte array out of the confidential tuple;
(3) Generate a new random nonce;
(4) Use the protocol version, row ID, public region/segment, and sorted sensitive field names to compute AAD;
(5) Encrypt with AES-256-GCM;
(6) Store row ID, public fields and nonce–ciphertext–tag in SQLite; and
(7) Maintain a B-tree index on (region, segment).

Algorithm 2 (retrieve and verify):
(1) Issue a database lookup against public region/segment;
(2) For every row retrieved, reconstruct the AAD using the publicly available information and the schema information.;
(3) Verify and decrypt the AEAD envelope using the client key;
(4) Parse the JSON array;
(5) Restore the *integer and string properties and discard the row if authentication and schema validation fails. Modification of the protected bytes, replacing the protected attribute from another row or altering the authenticated public attributes must cause the tag validation to fail.*

The accompanying reproducibility package includes the unchanged original benchmark, a separately labeled 12-trial replication script, both raw CSV datasets, the dataset generator, environment metadata, supplementary validation code and a README. Synthetic data use pseudorandom seed 20261013, regions R0–R7, segments S0–S3, integer cents, and Unicode/mixed strings. Configurations use 2,000, 8,000 and 16,000 rows; 4,096-byte SQLite pages; one randomly created key per mode and trial; distinct random 96-bit nonces per protected row; and six timed lookups after a warm-up. The prepared plaintext plus ciphertext formation time excludes key generation; insertion measures executemany and commit; allocation is page_count × page_size; retrieval measures the 40-row public-label predicate plus client decryption. Comparators are: plaintext (insecure reference), full-row AES-GCM (public labels encrypted, client scan), four-field AES-GCM (same index as proposed), and packed AES-GCM (proposed). The full-row comparator provides different functionality and is not interpreted as a controlled performance baseline for the indexed modes.

The supplementary run used Python 3.13.5, cryptography 46.0.4 and SQLite 3.46.1 on a single Linux x86-64 host (see machine and package details in JSON). It retains all size/mode conditions 12 times using the same algorithm and data-generation seed, with rotated method order. Runtime conditions differ from the originally submitted four-trial experiment, so the two batches are reported separately rather than pooled or substituted. The key and all random nonces were generated independently for each trial, whereas data content was reproducibly regenerated from the fixed non-secret seed.

2.4 Statistical and security evaluation

For each configuration, trial-level medians of six lookups form the retrieval observations. I report medians, sample standard deviations and trial-level paired percentage reductions [(field-wise − packed)/field-wise] × 100. Nonparametric bootstrap 95% percentile intervals (20,000 draws; fixed analysis seeds) summarize median paired reductions; a one-sided paired Wilcoxon signed-rank test compares the 12 replicated observations at each dataset size. The p values are exploratory, unadjusted for multiple comparisons, and reflect repeat measurements on one host and one generated dataset, not independent production installations. In particular, statistical significance here cannot establish external validity or broader cloud performance.

The submitted tests covered 1,000 exact round-trips per mode and 100 rejection attempts in each category. I added a DTAE-Pack-only supplementary suite with 2,003 round-trip rows (including unusual Unicode and large positive/negative integers), and 1,000 cases each of ciphertext mutation, wrong key, cross-row substitution, altered public label and truncated envelope. I separately demonstrated that replay of an older authentic envelope is accepted when row identity and AAD match; row deletion likewise cannot be detected by AEAD alone. These are functional checks consistent with the standard primitive, not proof of protection against client compromise, record rollback, metadata inference or chosen-ciphertext attacks outside the modeled library interface.

3 Results and Discussion

3.1 Original four-trial results: retained without modification

The submitted 16,000-row experiment reported 1,500 KiB for plaintext, 1,980 KiB for full-row AEAD, 3,488 KiB for four-field AEAD and 2,152 KiB for DTAE-Pack. The proposed allocation was 38.3% lower than field-wise storage, but 43.5% above plaintext. Original preparation medians were 190.55 versus 439.32 ms (56.6% lower), and indexed lookup/decryption medians were 0.46 versus 0.94 ms (51.1% lower), proposed versus field-wise. Table 2 retains key reported values; the originally submitted four-run raw observations remain in the reproduction package. Those original results did not provide a reliable uncertainty estimate; substantial outliers also appeared in the 8,000-row batch and therefore must not be described as generalizable speedups.

Table 2. Original four-trial 16,000-row measurements (retained)

Mode	Preparation median (ms)	40-row lookup median (ms)	SQLite allocation (KiB)
Plaintext	6.28	0.11	1,500
Full-row AES-GCM	244.60	9.51	1,980
Four-field AES-GCM	439.32	0.94	3,488
DTAE-Pack	190.55	0.46	2,152

Original trials and all three dataset sizes appear in original_raw_four_trials.csv; no historical observations were changed.

3.2 Independent 12-trial within-host replication

The new 12-trial replication reproduced all three size-dependent storage allocations exactly. On the current machine, the 16,000-row preparation medians were 63.37 ms (DTAE-Pack) and 171.85 ms (field-wise), with sample SDs 2.16 and 5.48 ms. The 40-row filtered lookup medians were 0.160 and 0.462 ms, with SDs 0.013 and 0.030 ms. The median paired trial-level reduction was 63.5% (95% exploratory bootstrap interval 62.5–63.8%) for preparation and 65.5% (62.8–67.4%) for lookup. In both comparisons all twelve trial pairs favored packing; one-sided Wilcoxon p=0.000244. These times are not directly interchangeable with the original four-trial medians because the execution environment changed. Storage differences arise deterministically from the envelope and schema formats; timing is workload- and implementation-dependent.

Table 3 presents medians and sample SDs at each record count, while accompanying raw CSV data permit more complete sensitivity analysis. The numerical consistency of storage allocation and directional consistency of timing support the narrow claim that grouping reduces overhead versus per-field protection for this benchmark. The measurements do not support a claim of statistical superiority to other published cloud encryption systems with unmatched query semantics or environments.

Table 3. Separate 12-trial replication: medians, sample SD, and paired analysis

Rows	Metric	Field-wise median ± SD (ms)	Packed median ± SD (ms)	Paired saving median (95% bootstrap CI)	Wilcoxon p (one-sided)
2,000	Preparation	21.274 ± 0.780	7.373 ± 0.199	65.1% (63.8–66.2%)	0.000244
2,000	Indexed lookup	0.456 ± 0.073	0.154 ± 0.001	66.2% (66.0–66.8%)	0.000244
8,000	Preparation	83.787 ± 3.474	30.536 ± 1.402	63.8% (63.0–64.0%)	0.000244
8,000	Indexed lookup	0.460 ± 0.010	0.158 ± 0.009	65.5% (64.5–66.0%)	0.000244
16,000	Preparation	171.847 ± 5.482	63.372 ± 2.155	63.5% (62.5–63.8%)	0.000244
16,000	Indexed lookup	0.461 ± 0.030	0.160 ± 0.013	65.5% (62.8–67.4%)	0.000244

3.3 Exact recovery, tampering and security boundaries

The submitted results recorded 1,000/1,000 exact reconstructions per mode and 100/100 expected rejection cases for each applicable tamper category. In the supplementary packed-mode suite, all 2,003 records recovered exactly and each of the five enumerated malformed or mismatched conditions failed authentication in 1,000/1,000 cases. Conversely, replay of a previously

valid record after its update remained decryptable (1/1), demonstrating the absence of freshness protection. The observed failures are predicted by AES-GCM tag verification and correct AAD binding, not evidence of a novel mathematical security property. A trustworthy deployment would need persistent nonce-uniqueness controls, protected key storage, version counters or an externally authenticated integrity log for rollback detection, database authorization, and operational recovery procedures.

Table 4. Supplemental correctness and negative checks (DTAE-Pack)

Test condition	Observed outcome	Interpretation
Exact typed round-trip	2,003/2,003	Includes unusual Unicode and integer boundaries
Ciphertext bit alteration	1,000/1,000 rejected	Authentication failure under GCM
Wrong AES key	1,000/1,000 rejected	Authentication failure under GCM
Cross-row ciphertext swap	1,000/1,000 rejected	AAD row binding tested
Public-label change	1,000/1,000 rejected	AAD index-label binding tested
Truncated envelope	1,000/1,000 rejected	Tag/length validation tested
Earlier authentic blob replay	1/1 accepted	Freshness protection is absent
Complete row deletion	Not detected by AEAD	Completeness protection is absent

3.4 Scope of novelty, limitations and future validation

Table 1 shows why direct percent-gain comparisons with searchable encryption, trusted DBMS, ABE revocation and data-compression systems would be misleading. The proposal occupies a much narrower design point: client-managed AES-GCM, public metadata indexes and bulk row retrieval. Compared with the four-field AES-GCM baseline, the grouping minimizes repeated nonce/tag metadata and library invocation overhead but sacrifices isolated-field update efficiency. Compared with full-row AES-GCM, public categorical labels become observable, enabling SQL indexing at the price of leakage (Table 4). Compared with plaintext it retains security benefits under the threat model but increases space and work. These trade-offs, rather than enhanced cryptographic strength, constitute the engineering contribution.

The experiment has only synthetic data, one CPU, in-memory SQLite, short trials, a fixed data distribution, no database server, no remote latency, no durability, no concurrent access, no sensitive-field queries and no persistent key vault. Public fields can reveal categories, access patterns, and result counts. Tampering tests do not cover adversaries with the client key or synchronized rollback of full trusted state. Future validation should prioritize (i) $\geq$30 paired repetitions on at least three different machines, (ii) PostgreSQL/cloud SQL with network and disk I/O, (iii) update-heavy versus full-record retrieval workloads, (iv) concurrency and larger volumes, (v) key rotation and nonce persistence, (vi) policy changes and leakage analysis, and (vii) a functionally matched implementation of an independently developed column-encryption comparator. At present, no cloud benchmark or external system speedup is claimed.

4 Conclusion

DTAE-Pack is a versioned, sensitivity-guided record format that encrypts numerical and Unicode fields together while authenticating public indexed labels. In the submitted 16,000-row test it reduced database allocation by 38.3% relative to independent four-field AES-GCM protection. A newly executed 12-trial single-host replication preserved this allocation difference and measured lower preparation and indexed lookup times under the same workload, with exploratory paired statistics reported rather than an unqualified performance generalization. Correctness and selected tamper checks support the implementation of AES-GCM with row-bound metadata; they do not establish new cryptographic security, key-theft resilience, replay prevention or cloud scalability. The practical use case is bulk retrieval of several sensitive fields when revealing a limited public index is acceptable. The immediate next step is a cloud-DBMS deployment with concurrency, isolated-field updates and persistent key/nonce management.

References

1) Sun B, Zhao S, Tian G. SQL queries over encrypted databases: a survey. *Connection Science*. 2024;36(1):2323059. https://doi.org/10.1080/09540091.2024.2323059.
2) Lutsch A, El-Hindi M, István Z, Binnig C. Towards high-performance and trusted cloud DBMSs. *Datenbank-Spektrum*. 2025;25:39–50. https://doi.org/10.1007/s13222-025-00495-8.

3) Li M, Zhao X, Chen L, Tan C, Li H, Wang S, et al. Encrypted databases made secure yet maintainable. In: Proceedings of the 17th USENIX Symposium on Operating Systems Design and Implementation. 2023;p. 117–133. Available from: https://www.usenix.org/conference/osdi23/presentation/li-mingyu.
4) Zhu J, Cheng K, Liu J, Guo L. Full encryption: an end-to-end encryption mechanism in GaussDB. *Proceedings of the VLDB Endowment*. 2021;14(12):2811–2814. https://doi.org/10.14778/3476311.3476351.
5) Sharma D. Searchable encryption: a survey. *Information Security Journal: A Global Perspective*. 2023;32(2):76–119. https://doi.org/10.1080/19393555.2022.2033367.
6) Zhang D, Yang S, Chen M, Zheng L, Fan J, Dong A. Adaptive encryption method of sensitive data in data center database based on big data cross-mapping fusion algorithm. *Discover Applied Sciences*. 2025;7:924. https://doi.org/10.1007/s42452-025-07581-2.
7) Routray K, Bera P. Dynamic attribute-based encryption for secure data sharing in cloud applications. *Future Generation Computer Systems*. 2026;185:108671. https://doi.org/10.1016/j.future.2026.108671.
8) Naikwade N, Pathan S, Dhotre P. Cost efficient storage and swift time-bound access revocation using attribute-based encryption in untrusted cloud environments. *Journal of Cloud Computing*. 2026;15:71. https://doi.org/10.1186/s13677-026-00861-1.
9) Shu G, Hong H, Xie M, Ren Z, Wu Y, Liu Y. A novel searchable attribute-based encryption scheme supporting fuzzy keywords ranking for the cloud environment. *Concurrency and Computation: Practice and Experience*. 2026;38(1):e70479. https://doi.org/10.1002/cpe.70479.
10) Meng F, Cheng L. Cloud-aided attribute-based encryption with efficient tracing and accountability auditing. *Journal of Systems Architecture*. 2025;163:103407. https://doi.org/10.1016/j.sysarc.2025.103407.
11) Leelavathy L, Sugumar R. EUDSFDNA: Enhanced User Data Security Framework using DNA Cryptography in Cloud Computing Environment. *Journal of Shanghai Jiaotong University*. 2020;16(9):568–574. Available from: https://www.academia.edu/120300458/EUDSFDNA_Enhanced_User_Data_Security_Framework_using_DNA_Cryptography_in_Cloud_Computing_Environment.
12) Leelavathy L, Sugumar R. SNDDNA: Secure Numerical Data using DNA Cryptography Method in Cloud Computing Environment. *Mukt Shabd Journal*. 2020;9(9):100–107.
13) Leelavathy L, Sugumar R. DNA Cryptography Data Storage Security Algorithm Analysis in Cloud Computing Environment. *Journal of the Maharaja Sayajirao University of Baroda*. 2020;54(2(X)):74–79.
14) Leelavathy L, Sugumar R. DNA Cloud Data Security for Numerical Data using Cryptography Methods. *Solid State Technology*. 2020;63(3):4160–4165. Available from: https://solidstatetechnology.us/index.php/JSST/article/view/6154.
15) Dworkin M. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. In: NIST Special Publication 800-38D. 2007. https://doi.org/10.6028/NIST.SP.800-38D.
16) National Institute of Standards and Technology. Advanced Encryption Standard (AES). In: FIPS 197. 2023. Updated edition. https://doi.org/10.6028/NIST.FIPS.197-upd1.