The Need for 256/512-bit Block Ciphers

Aman Pathak
Independent Researcher
vajradevam@gmail.com

Abstract

The Advanced Encryption Standard (AES) with a 128-bit block size has been the cornerstone of symmetric encryption for over two decades. However, the exponential growth of data and the increasing need for long-term secure storage are beginning to expose the theoretical limitations of a 128-bit block size. This review discusses the cryptographic principles, namely the birthday paradox and the pigeonhole principle, that dictate a non-negligible probability of block collisions after encrypting approximately 2^{64} blocks of data. In an era of petabyte and exabyte datasets intended for long-term archival, this threshold is becoming increasingly relevant. We argue for the proactive standardization and hardware-accelerated deployment of block ciphers with larger block sizes, such as 256-bit or 512-bit variants of the Rijndael algorithm, to ensure robust security for massive datasets at rest over extended periods.

1 Introduction

The dawn of the 21st century has been characterized by an unparalleled explosion in digital information. Global data creation is not merely increasing; it is expanding at a geometric rate, with forecasts predicting volumes exceeding 180 zettabytes by 2025 [1]. This deluge of data is fueled by a confluence of transformative technologies: the pervasive instrumentation of our environment by the Internet of Things (IoT) generating continuous sensor streams [2], the sophisticated data processing and generation capabilities of Artificial Intelligence (AI) and Machine Learning (ML) algorithms [3], and the massive datasets produced by cutting-edge scientific endeavors, from genomic sequencing to astrophysical observatories [4]. Beyond these, enterprise systems, social media, and everyday digital interactions contribute to this ever-growing digital footprint.

A significant portion of this data, often termed "data at rest," must be stored securely for extended periods, sometimes for decades, due to regulatory compliance, business intelligence, historical record-keeping, or its intrinsic long-term value [5]. The imperative to protect this data from unauthorized access, modification, or disclosure is paramount. Symmetric encryption algorithms have long been the workhorse for ensuring the confidentiality of such data, and for over two decades, the Advanced Encryption Standard (AES) has been the globally recognized and ubiquitously deployed benchmark [6]. AES, often implemented as AES-128 (using a 128-bit key and a 128-bit block), has proven remarkably resilient. Its various key sizes (128, 192, and 256 bits) offer robust defense against brute-force key recovery attacks, with a 128-bit key, for instance, being computationally infeasible to break with current and foreseeable technology.

However, the security of a block cipher is not solely dependent on its key length; the block size plays an equally crucial, albeit different, role in its overall security profile, particularly when encrypting vast quantities of data. AES, in all its key-size variants, employs a fixed block size of 128 bits. While this block size has been adequate for a wide array of applications, the sheer scale of modern datasets and the trend towards long-term archival are beginning to press against its theoretical security boundaries. Specifically, the cryptographic principle known as the "birthday paradox" indicates that the probability of encountering a collision between ciphertext blocks (where two different plaintext blocks encrypt to the same ciphertext block under the same key, or vice-versa depending on the mode) becomes non-negligible after encrypting approximately 2^{64} blocks of data [7]. With a 128-bit block (2^{128} possible blocks), this 2^{64} threshold, equating to 256 Exabytes of data, is no longer a purely academic concern but an approaching horizon for large-scale systems.

This review paper aims to scrutinize the implications of this cryptographic reality. We will delve into the theoretical underpinnings of block cipher limitations, focusing on the birthday paradox. Subsequently, we will assess the practical risks these limitations pose in the context of encrypting and storing massive, long-lived datasets. Building on this analysis, the paper will advocate for the proactive consideration, standardization, and widespread adoption of symmetric block ciphers featuring larger block sizes, such as 256-bit or potentially 512-bit variants. We will explore the suitability of extending the Rijndael algorithm, from which AES was derived, for this purpose. Finally, we will underscore the indispensable role of dedicated hardware acceleration in ensuring that such next-generation ciphers can be deployed efficiently, thereby maintaining both security and performance in our increasingly data-centric world. The objective is to highlight the emerging need for an evolution in our cryptographic toolkit to safeguard the integrity and confidentiality of information for the decades to come.

2 Theoretical Limitations of 128-bit Block Ciphers

When encrypting data with a block cipher in common modes of operation (like CBC, CTR, GCM), each block of plaintext is processed into a block of ciphertext. If the same plaintext block is encrypted multiple times with the same key and initialization vector (IV) in certain modes, it will produce the same ciphertext block. More generally, even with unique plaintexts, the finite size of the block space (2^{128} for AES-128) leads to a probabilistic chance of output block collisions.

2.1 The Birthday Paradox and its Cryptographic Implications

The "birthday paradox," a well-known phenomenon in probability theory, provides a surprisingly counterintuitive result: in a group of randomly chosen people, the probability of at least two individuals sharing the same birthday is much higher than one might instinctively expect. For instance, in a group of just 23 people, this probability exceeds 50%. This paradox arises because one is comparing every pair of individuals, and the number of possible pairs grows quadratically with the number of people.

This principle has direct and significant analogues in cryptography, particularly concerning the security of block ciphers [8]. When a block cipher with a b-bit block size is used to encrypt data, it processes plaintext in b-bit chunks, producing b-bit ciphertext blocks. The total number of possible distinct blocks is 2^b. If the encryption process, for a

given key, behaves like a random mapping from plaintext blocks to ciphertext blocks (an idealization of a pseudo-random permutation or PRP), then after encrypting a certain number of blocks, we can expect to see a "collision." A collision, in this context, refers to an event where two distinct input blocks (e.g., plaintext blocks P_i and P_j where $i \neq j$) encrypt to the same output block ($C_i = C_j$) under the same key, or, depending on the mode of operation, the same input block occurs multiple times leading to predictable output patterns if not handled carefully (e.g., with unique IVs). More generally, it refers to any two b-bit values in a sequence of inputs or outputs of the cipher (or related values derived from them) becoming identical.

The approximate number of blocks that need to be encrypted before a collision is likely (i.e., probability ≈ 0.5) is given by the birthday bound, which is roughly $\sqrt{2^b} = 2^{b/2}$ [7]. This is a direct consequence of the generalized birthday problem.

For AES, which employs a 128-bit block size ($b = 128$), the total space of possible output blocks is $d = 2^{128}$. Applying the birthday paradox, one would expect a collision after encrypting approximately $n \approx \sqrt{d} = \sqrt{2^{128}} = 2^{64}$ blocks. More formally, the probability of at least one collision, $P(n; d)$, after n items are randomly chosen from a space of d distinct possibilities (with replacement, which is a good model for random ciphertext blocks) can be approximated by:

$$P(n; d) \approx 1 - e^{-n(n-1)/(2d)}$$

For $n \ll d$, a simpler approximation is often used:

$$P(n; d) \approx \frac{n(n-1)}{2d} \approx \frac{n^2}{2d}$$

Let's apply this to AES-128. When $n = 2^{64}$ (which is approximately 1.84×10^{19} blocks):

$$P(2^{64}; 2^{128}) \approx \frac{(2^{64})^2}{2 \cdot 2^{128}} = \frac{2^{128}}{2 \cdot 2^{128}} = \frac{1}{2}$$

This calculation confirms that after encrypting 2^{64} blocks of data using AES-128 with a single key, the probability of at least one ciphertext block collision reaches 50% [9]. Each block is 16 bytes (128 bits / 8 bits per byte), so 2^{64} blocks translate to $2^{64} \times 16$ bytes, which is $2^{64} \times 2^4$ bytes $= 2^{68}$ bytes. Since 1 Exabyte (EB) is 2^{60} bytes, this amount is 2^8 EB = 256 Exabytes.

The occurrence of such a collision is not merely a theoretical curiosity; it has tangible security implications. While a collision does not directly reveal the encryption key, it can leak critical information about the plaintexts, especially when certain modes of operation are employed [10]. For example:

- In Cipher Block Chaining (CBC) mode, if $C_i = C_j$ for $i \neq j$, then it implies $E_K(P_i \oplus C_{i-1}) = E_K(P_j \oplus C_{j-1})$. Assuming the cipher behaves as a permutation, this means $P_i \oplus C_{i-1} = P_j \oplus C_{j-1}$. An adversary who observes this collision can deduce the XOR sum of two plaintext blocks: $P_i \oplus P_j = C_{i-1} \oplus C_{j-1}$. This can be highly detrimental if the adversary has some knowledge about one of the plaintext blocks.

- In Counter (CTR) mode, encryption is performed by $C_i = P_i \oplus E_K(\text{nonce}||\text{counter}_i)$. A collision $C_i = C_j$ would imply $P_i \oplus E_K(\text{nonce}||\text{counter}_i) = P_j \oplus E_K(\text{nonce}||\text{counter}_j)$. While less direct, the primary concern in CTR mode is the reuse of a (key, nonce,

counter) tuple, which is a catastrophic failure. However, the birthday bound also applies to the output of the cipher $E_K(\cdot)$, and repeated output blocks (keystream blocks) could also provide statistical advantages to an attacker over large amounts of data.

- For Authenticated Encryption with Associated Data (AEAD) modes like GCM (Galois/Counter Mode), the security proofs often rely on the uniqueness of counter blocks and the underlying block cipher's PRP security, which is affected by the birthday bound. For GCM, NIST SP 800-38D recommends limiting the amount of plaintext encrypted with a single key to less than 2^{32} full 128-bit blocks when IVs are generated randomly to avoid collisions that could undermine authentication [11]. While this is a stricter limit for IV collision in GCM, the general 2^{64} data limit for the underlying block cipher remains a more fundamental concern for confidentiality over vast datasets.

The SWEET32 attack demonstrated the practical feasibility of birthday attacks against 64-bit block ciphers (like Triple-DES and Blowfish), where the 2^{32} block birthday bound was reached, allowing plaintext recovery in TLS and OpenVPN sessions [12]. While 2^{64} is vastly larger than 2^{32}, the relentless growth in data processing capabilities and storage capacities suggests that this higher threshold for 128-bit ciphers cannot be considered permanently out of reach for well-funded adversaries or for systems encrypting truly enormous volumes of data over long periods. Thus, a collision is a foundational security concern that signals the erosion of the cipher's intended security properties.

3 Implications for Huge Data and Long-Term Storage

The theoretical birthday bound of 2^{64} blocks for a 128-bit block cipher, equating to 256 Exabytes of data encrypted under a single key, might have been dismissed as a purely academic concern in the early days of AES's adoption. However, the trajectory of data generation and the evolving requirements for data retention in the 21st century have brought this cryptographic threshold into sharper, more practical focus. The era of "Big Data" is not merely a buzzword but a reality characterized by volumes, velocities, and varieties of data that were previously unimaginable [13]. This new reality, coupled with the imperative for long-term, secure data archival, means that the 2^{64} block limit is no longer a comfortably distant milestone for an increasing number of applications and systems.

- **The Scale of Modern Massive Datasets:** The sheer volume of data generated and processed in various domains is staggering.
  - *Scientific Research:* Fields like genomics now routinely deal with sequencing data that can run into many petabytes for large-scale population studies [14]. High-energy physics experiments, such as those at CERN's Large Hadron Collider, already generate hundreds of petabytes, with future projects like the Square Kilometre Array (SKA) in radio astronomy projected to produce exabytes of raw data per year [15].

- *Cloud Computing and Services:* Large cloud service providers (CSPs) manage aggregated data stores that collectively measure in the hundreds of exabytes, if not zettabytes, globally [16]. While this data is partitioned across many users and keys, the infrastructure itself handles encryption at an unprecedented scale, and individual large tenants or services might approach significant fractions of the birthday bound over time.
- *Internet of Things (IoT) and Autonomous Systems:* The proliferation of IoT devices, from smart cities to industrial sensors and autonomous vehicles, generates continuous streams of data. An autonomous vehicle, for instance, can generate terabytes of sensor data per day, which may need to be stored and analyzed [17].
- *Enterprise Data Lakes and Warehouses:* Modern enterprises are increasingly centralizing vast quantities of structured and unstructured data into data lakes and warehouses for analytics, business intelligence, and machine learning, often reaching petabyte scale [18].

While a single dataset rarely reaches 256 EB today, the cumulative data encrypted by a large organization or a widely deployed system over its lifetime under a limited set of master keys (or keys derived in a way that doesn't fully reset the birthday bound count for all data) can become a concern.

- **The Imperative of Long-Term Archival:** Beyond immediate processing, a vast amount of digital information must be preserved securely for extended durations, often spanning decades or even longer.
  - *Regulatory and Compliance Mandates:* Numerous sectors, including finance (e.g., SEC regulations), healthcare (e.g., HIPAA in the US, GDPR in Europe for patient data), and government, have stringent data retention policies requiring secure archival for many years, sometimes indefinitely for records of historical significance [19].
  - *Intellectual Property and Business Value:* Companies archive research data, design documents, and other forms of intellectual property that hold long-term value. Historical business data is also crucial for trend analysis and long-range planning.
  - *Cultural and Scientific Heritage:* Libraries, archives, and research institutions are tasked with preserving digital cultural artifacts and scientific datasets for future generations, a task that requires ensuring not only bit preservation but also enduring confidentiality and integrity [20].

When data is encrypted and stored for such long periods, the original encryption choices must withstand not only the test of time against evolving cryptanalytic capabilities but also the cumulative effect of encrypting more data under keys that may not be frequently rotated for deeply archived, "cold" storage due to logistical complexities. The longer data is stored, the more opportunities arise for the conditions leading to birthday bound concerns to manifest, especially if key management practices are not impeccably maintained across data silos and generations of storage technology.

- **Aggravated Security Implications of Collisions in Large-Scale, Long-Term Contexts:** As detailed in Section 2.1, a ciphertext collision can leak plaintext information, with the specific type of leakage dependent on the block cipher's mode of operation [10]. For massive datasets stored long-term, these implications are amplified:
  - *Increased Probability of Impact:* While the 2^{64} threshold pertains to data encrypted under a single key, the sheer volume processed by large systems or archived over decades increases the likelihood that some key, at some point, will have been used to encrypt enough data to approach this bound. Even if individual datasets are smaller, if they are, for example, all encrypted with keys derived from a common master key using a deterministic process, the effective number of blocks "seen" by the underlying cryptographic primitive could accumulate in unexpected ways across the system.
  - *Severity of Information Leakage:* For large archives, even a small leakage from a collision (e.g., $P_i \oplus P_j$) could be significant if P_i and P_j are sensitive records. If an attacker can collect many such collided blocks from a vast ciphertext pool, the potential for reconstructing meaningful information increases.
  - *Difficulty of Remediation:* Discovering and remediating the effects of such collisions in exabyte-scale archives would be an extraordinarily complex and costly undertaking, potentially requiring re-encryption of massive swathes of data.

  The challenge is that the security guarantees of the block cipher itself begin to degrade as one approaches the $2^{b/2}$ data limit, making the data more susceptible to structural attacks that exploit these mathematical properties rather than attacking the key itself.

Addressing these concerns solely through rekeying strategies—encrypting data with a new key after a certain amount of data has been processed by an old key—can mitigate the risk of hitting the birthday bound for any single key. However, implementing and managing rekeying across exabytes of data, especially for data at rest that might be distributed, infrequently accessed, and stored on diverse media, presents its own formidable operational and security challenges. These include secure key generation and distribution, robust key lifecycle management, the performance impact of re-encrypting vast data volumes, and ensuring the atomicity and correctness of the re-encryption process without data loss or introducing new vulnerabilities [21]. The complexity and potential for error in such large-scale rekeying operations mean that relying on it as the sole defense against birthday bound issues for 128-bit ciphers may not be a tenable long-term solution for the most demanding scenarios.

4 The Case for 256/512-bit Block Ciphers

To address the impending limitations of 128-bit blocks for future-proofing data security, a move towards larger block sizes is a logical step.

4.1 Revisiting Rijndael: A Foundation for Larger Block Sizes

The Rijndael algorithm, selected by NIST as the Advanced Encryption Standard (AES), was intrinsically designed with greater flexibility in block and key sizes than what was ultimately standardized in FIPS PUB 197 [6]. The original Rijndael specification detailed support for block sizes of 128, 160, 192, 224, and 256 bits, independently of key sizes which could also be 128, 160, 192, 224, or 256 bits [22, 23]. While AES exclusively fixed the block size at 128 bits, the foundational cryptographic framework of Rijndael provides a well-analyzed and understood basis for considering ciphers with larger block capacities.

- **Rijndael with a 256-bit Block (Rijndael-256):** Adopting a 256-bit block size using the original Rijndael framework is a natural extension. In this configuration, the internal state of the cipher would be represented as a 4x8 array of bytes (4 rows, 8 columns, as $N_b = 256/32 = 8$), compared to the 4x4 array for AES's 128-bit blocks ($N_b = 4$). The core operations would adapt as follows:
  - *SubBytes:* This non-linear substitution step would apply to each byte of the 4x8 state individually, using the same S-box as in AES.
  - *ShiftRows:* The cyclic shift of rows would use different offset values for the 8-column state. For example, in the original Rijndael specification for $N_b = 8$, rows 0, 1, 2, and 3 are shifted by 0, 1, 3, and 4 bytes respectively (offsets can vary based on specific Rijndael versions/proposals, the key is they are defined for larger states) [23]. This contrasts with the 0, 1, 2, 3 byte shifts for AES's 4-column state.
  - *MixColumns:* This operation, crucial for diffusion, would operate on each of the 8 columns of the state independently, using the same mathematical principles as in AES but applied to more columns per round.
  - *AddRoundKey:* The round key, derived from the main key via the key schedule (which also supports these larger block/key sizes), would be XORed with the 256-bit state.

  The number of rounds (N_r) in Rijndael is defined by the formula $N_r = \max(N_k, N_b) + 6$, where N_k is the key length in 32-bit words and N_b is the block length in 32-bit words [23, Chapter 4]. For a 256-bit block ($N_b = 8$) and any standard key length (e.g., 128-bit $N_k = 4$, 192-bit $N_k = 6$, or 256-bit $N_k = 8$), the number of rounds would be $N_r = \max(N_k, 8) + 6$, resulting in 14 rounds.

  The primary security benefit of using a 256-bit block size is the elevation of the birthday bound for collisions to $2^{256/2} = 2^{128}$ blocks. This is a colossal number: 2^{64} (approximately 1.8×10^{19}) times larger than the 2^{64} bound for AES-128. Encrypting 2^{128} blocks, where each block is 32 bytes, would amount to $2^{128} \times 32 = 2^{128} \times 2^5 = 2^{133}$ bytes of data (or 2^{73} Exabytes). This quantity is so vast that it effectively renders block collision concerns due to the birthday paradox purely theoretical for any conceivable data volume, providing an immense and enduring margin of safety.

- **Hypothetical Considerations for a 512-bit Block Cipher:** Extending block ciphers to 512 bits is a more speculative endeavor, as the original Rijndael specification did not explicitly detail a 512-bit block variant ($N_b = 16$). While the core principles of a substitution-permutation network could, in theory, be extended, several considerations arise:

– *New Design and Analysis:* Such an extension would constitute a new cipher design rather than a straightforward application of an existing one. It would require a thorough security analysis, including resistance to known attacks like differential and linear cryptanalysis, integral cryptanalysis, and structural attacks, tailored to the significantly larger state size [24].

– *Performance Implications:* A 512-bit state (e.g., a 4x16 byte array) would significantly increase the computational cost per block in software. Hardware implementations would require wider data paths and more resources. The ShiftRows offsets and MixColumns operations would need careful definition to ensure adequate diffusion across such a large state within a reasonable number of rounds.

– *Number of Rounds:* The round function might need re-evaluation, and the number of rounds would likely increase (e.g., following $N_r = \max(N_k, 16)+6 = 22$ rounds for $N_b = 16$), further impacting performance.

A 512-bit block cipher would shift the birthday collision boundary to an astronomical $2^{512/2} = 2^{256}$ blocks. This offers an almost absolute safeguard against collisions ($2^{256} \times 64$ bytes of data before 50% collision probability), effectively eliminating this specific concern for eternity. However, the practical necessity and performance trade-offs for such a large block size would need to be carefully weighed against the already substantial security gains offered by a 256-bit block. For the scope of this paper, the existing, well-analyzed 256-bit block capabilities of Rijndael present a more immediate and pragmatic path forward.

Standardizing a Rijndael variant with a 256-bit block size, perhaps under a new designation like "AES-256-256" (to denote a 256-bit key and 256-bit block, though other naming conventions are possible), would offer a direct and well-understood evolution. This approach leverages the extensive body of existing security analysis and implementation experience associated with Rijndael's design philosophy [23]. Such a standardization effort, likely spearheaded by bodies like NIST, would involve a public process of evaluation and comment, ensuring broad cryptographic community consensus before any formal adoption.

4.2 The Linchpin of Practicality: Hardware Acceleration

The widespread and successful deployment of AES as the global encryption standard was significantly bolstered by its efficient implementation characteristics, not only in software but critically, through dedicated hardware support. The introduction of instruction set extensions, most notably Intel's AES-NI (Advanced Encryption Standard New Instructions) [25] and similar technologies from other CPU vendors like AMD (e.g., via an equivalent part of their "Cryptography Extensions") and ARM (e.g., ARMv8 Crypto Extensions [26]), has revolutionized the performance of AES operations. These instructions provide direct hardware execution of AES rounds, leading to order-of-magnitude speedups over pure software implementations and making strong encryption ubiquitous and computationally inexpensive on platforms ranging from servers to personal computers and even mobile devices [27].

For any new cryptographic standard based on larger block sizes, such as a 256-bit or 512-bit block variant of Rijndael, achieving a similar level of hardware acceleration is

not merely desirable but essential for its viability and broad adoption. Without it, the inherent computational overhead of processing larger blocks and potentially more rounds would likely render such ciphers too slow for many practical applications, negating their security advantages.

- **Achieving Performance Parity and Beyond:** Software implementations of ciphers with larger blocks (e.g., 256 bits) would naturally be slower than AES-128 if both were running on general-purpose CPU instructions. This is due to the increased state size requiring more data movement, more complex (or more instances of) operations like ShiftRows and MixColumns per round, and potentially a higher number of rounds for equivalent security strength. Without hardware support, this performance deficit would create a significant disincentive for adoption, particularly in high-throughput network devices, storage systems, and latency-sensitive applications, even where the enhanced security against block collisions is theoretically compelling [28]. Dedicated hardware acceleration can overcome this deficit, aiming not just for parity with existing AES-128 hardware speeds but potentially exceeding them on a per-byte basis if parallelism is effectively exploited.

- **Feasibility of Hardware Implementation for Larger Blocks:** The architectural principles that make AES amenable to hardware acceleration are largely applicable to Rijndael variants with larger block sizes. The byte-oriented nature of Rijndael and its regular, iterative structure are well-suited for hardware pipelines [23].

  - *Adaptable Core Operations:* The S-box (SubBytes) operation can be implemented using lookup tables or combinational logic, replicated as needed for the wider state. ShiftRows is fundamentally a permutation of byte positions, achievable through wiring. MixColumns involves matrix multiplications over $GF(2^8)$, which can be parallelized across the increased number of columns in a wider state (e.g., 8 columns for a 256-bit block).
  - *Wider Datapaths and Parallelism:* Modern CPUs already incorporate wide datapaths (e.g., 256-bit, 512-bit, or even wider for SIMD/vector processing units like AVX). Leveraging such datapath widths for cryptographic operations is a known technique. A 256-bit block cipher could be processed in a single cycle in the datapath if designed appropriately, or multiple rounds could be unrolled and pipelined [29].
  - *Resource Considerations:* While a larger state size and potentially more rounds would necessitate a larger die area for the cryptographic unit and slightly increased power consumption per operation compared to AES-128, these increases are generally considered to be well within the capabilities and power budgets of modern silicon manufacturing processes, especially given the significant security benefits. The design of efficient key schedulers for larger key/block sizes in hardware is also a critical, but solvable, engineering challenge [30].

- **Critical Role of Energy Efficiency:** In an increasingly energy-conscious world, the power consumption of cryptographic operations is a vital consideration. Dedicated hardware accelerators are significantly more energy-efficient than software running on general-purpose CPUs for cryptographic tasks. By offloading encryption/decryption to specialized circuits that perform the operations in fewer clock

cycles and with less overhead, overall system power consumption is reduced [31]. This is paramount for:

- *Mobile and IoT Devices:* Prolonging battery life is a key design constraint. Efficient hardware encryption enables robust security without unduly draining power resources on these often resource-constrained platforms.
- *Data Centers and Cloud Computing:* At scale, even marginal improvements in energy efficiency per operation can translate into substantial savings in electricity costs and a reduced carbon footprint for the massive volumes of data encrypted and decrypted daily.

Therefore, the path towards standardized large-block ciphers must involve proactive and concurrent engagement between cryptographic researchers, standards bodies (like NIST or ISO), and hardware manufacturers (CPU vendors, SoC designers, FPGA producers). A co-design approach, where algorithm selection and refinement are influenced by hardware implementation feasibility and efficiency from an early stage, is crucial [32]. This ensures that new standards are not only theoretically sound but also practically implementable in performant and energy-efficient hardware, paving the way for their rapid adoption and long-term success in securing the vast datasets of the future. Open discussions and potentially open-source hardware reference designs could further accelerate this ecosystem development.

5 Challenges and Considerations

Transitioning to a new cryptographic standard, or an extension of an existing one, is a significant undertaking.

- **Standardization Process:** A rigorous process of selection, analysis, and public scrutiny is required.
- **Performance Overhead:** Even with hardware acceleration, larger block sizes might inherently carry some performance penalty due to increased internal state and more complex round functions (if the number of rounds is scaled). This needs to be balanced against the security gains.
- **Backward Compatibility and Transition:** Systems would need to support both old and new standards during a transition period, adding complexity.
- **Cryptanalysis:** Any proposed cipher, even if based on Rijndael, would need fresh cryptanalysis specifically for the new block/key size configurations. The original Rijndael analysis covered these larger block sizes, but renewed scrutiny would be prudent [23].

6 Conclusion

The 128-bit block size of AES has served the digital world well. However, as data volumes continue their exponential ascent and long-term secure storage becomes paramount, the 2^{64} block birthday bound presents a looming, if not yet immediate, threat for high-volume, long-retention applications. The cryptographic community and standards bodies should

proactively explore, standardize, and promote the adoption of block ciphers with larger block sizes, such as a 256-bit block variant of Rijndael. Crucially, this effort must be paralleled by commitments from hardware vendors to provide the necessary acceleration, ensuring that enhanced security does not come at an unacceptable performance cost. Waiting until the limitations of 128-bit blocks become a widespread practical problem would be a reactive stance; a proactive approach is essential for the continued trust and security of our increasingly data-driven world.

References

[1] Statista Research Department, "Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2025." Statista, June 2021.

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[3] G. Marcus, "The next decade in ai: Four steps towards robust artificial intelligence," *arXiv preprint arXiv:2002.06177*, 2020.

[4] V. Marx, "The big challenges of big data," *Nature*, vol. 498, no. 7453, pp. 255–260, 2013.

[5] J. Rothenberg, "Ensuring the longevity of digital documents," *Scientific American*, vol. 272, no. 1, pp. 42–47, 1995.

[6] National Institute of Standards and Technology, "FIPS PUB 197: Advanced Encryption Standard (AES)," November 2001.

[7] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 2nd ed., 2014.

[8] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996.

[9] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. Wiley, 2nd ed., 1996.

[10] M. Bellare and C. Namprempre, "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm," in *Advances in Cryptology — ASIACRYPT 2000*, pp. 531–545, Springer Berlin Heidelberg, 2000.

[11] National Institute of Standards and Technology, "NIST Special Publication 800-38D: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC," tech. rep., NIST, November 2007.

[12] K. Bhargavan and G. Leurent, "On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*, pp. 456–467, ACM, 2016.

[13] D. Laney, "3D Data Management: Controlling Data Volume, Velocity, and Variety," February 2001.

[14] Z. D. Stephens, S. Y. Lee, F. Fiaz, Z. Cui, I. Torkashvand, B. Pachev, Y. Demyanova, G. J. Torkashvand, and M. B. Gerstein, "Big data in genomics: Challenges and opportunities," *Trends in Biotechnology*, vol. 33, no. 10, pp. 577–580, 2015.

[15] SKA Telescope, "The scale of the data challenge." SKA Telescope Website, 2023.

[16] Cisco Systems, "Cisco Global Cloud Index: Forecast and Methodology, 2016–2021 (White Paper – Note: This specific report is older, look for latest cloud traffic/storage reports from major analysts like Gartner, IDC, or Cisco for up-to-date figures)," November 2018.

[17] M. Gerla, E.-K. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, pp. 241–246, 2014.

[18] W. H. Inmon, D. Linstedt, and M. Levins, *Data Lake Architecture: Designing the Data Lake and Avoiding the Garbage Dump*. Technics Publications, 2019.

[19] A. Grimsrud, O. Kvelland, and E. Snekkenes, "Data retention: balancing privacy, security, and proving legal compliance," *International Journal of Information Security*, vol. 19, pp. 347–363, 2020.

[20] UNESCO, "Charter on the Preservation of Digital Heritage," 2003. Accessed on May 20, 2025.

[21] H. Shacham, "The pitfalls of rekeying," in *Theory of Cryptography Conference (TCC 2007)*, vol. 4392 of *Lecture Notes in Computer Science*, pp. 458–474, Springer, 2007.

[22] J. Daemen and V. Rijmen, "AES Proposal: Rijndael," tech. rep., National Institute of Standards and Technology (NIST), September 1999. Version 2.

[23] J. Daemen and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer Science & Business Media, 2002.

[24] L. R. Knudsen and V. Rijmen, *The Block Cipher Companion*. Information Security and Cryptography, Springer, 2011.

[25] Intel Corporation, "Intel Advanced Encryption Standard Instructions (AES-NI)." Intel White Paper, 2010.

[26] ARM Limited, "ARMv8-A Architecture Reference Manual – Cryptography Extension." ARM Developer Documentation, 2023. Refer to the latest version of ARM Architecture Reference Manual for ARMv8-A, specific section on Cryptography Extensions.

[27] S. Gueron, "AES-NI: A New Instruction Set for AES Cryptography," in *Intel Technology Journal*, vol. 14, pp. 65–81, 2010. Often cited indirectly; direct access to Intel Tech Journal can be limited. Focus on its impact through various performance studies.

[28] D. J. Bernstein, "Why not larger blocks? (Or: How to protect 2^{64} blocks of data with a 128-bit block cipher)." Blog post, cr.yp.to, January 2016.

[29] V. E. E. T. Güneysu and J.-M. S. for CHES 2023, eds., *Cryptographic Hardware and Embedded Systems – CHES YEAR*, Lecture Notes in Computer Science, Springer, Various, annually. CHES proceedings regularly feature papers on efficient hardware implementations of cryptographic algorithms, including AES and explorations of other ciphers on FPGAs and ASICs. Specific papers would detail parallel/pipelined architectures.

[30] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A Compact Rijndael Hardware Architecture with S-Box Optimization," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E85-A, pp. 50–58, January 2002. This is an early paper on efficient AES hardware; many more advanced designs exist now, but it shows foundational work. Look for more recent CHES papers for state-of-the-art techniques.

[31] N. Pramstaller, J. Wolkerstorfer, and V. Rijmen, "Energy and Power Analysis of an AES S-Box Implementation," in *Proceedings of the 2005 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2005)*, vol. 3659 of *Lecture Notes in Computer Science*, pp. 355–365, Springer, 2005.

[32] O. Reparaz, S. S. Roy, F. Vercauteren, and I. Verbauwhede, "A Masked Implementation of NewHope," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2017, no. 2, pp. 144–167, 2017. While this paper is on post-quantum crypto, TCHES as a venue and papers like these often discuss co-design principles and hardware considerations for new cryptographic primitives.