SoK: Rijndael-256

Dessalegn Ayalneh, Anas Hlayhel, Setareh Sharifian, Alexander Tereschenko*
Intel Corporation
{dessalegn.ayalneh,anas.hlayhel,setareh.sharifian,aleksandr.v.tereschenko}@intel.com

***Abstract*—Most symmetric modes of encryption that rely on PRP primitives are limited by the birthday bound over the block size (can't encrypt more than $2^{n/2}$ blocks). This could be a severe limitation if current block width of 128 is used (can't encrypt more than 2^{64} blocks) for cloud systems that transact a large amount of data. This limitation can be overcome by either realizing a mode of encryption based on a PRF (that doesn't suffer from the birthday bound) or by using a wider block cipher like $Rijndael-256$ which allows us to encrypt 2^{128} blocks. In this paper, we focus on the wider block method as manifested in $Rijndael-256$. We also look at implementations of $Rijndael-256$ that take advantage of vectorized $AES-NI$ which optimizes performance.**

***Index Terms*—symmetric encryption, block ciphers, Rijndael, wide block, cryptanalysis**

I. Introduction

Block ciphers are among the most fundamental primitives in symmetric cryptography. They operate on fixed-size input blocks (typically 64 or 128 bits) under a secret key to produce ciphertext blocks of equal length. By themselves, block ciphers only define a permutation over a finite space; to encrypt variable-length messages securely, they are combined with modes of operation such as CBC, CTR, and GCM. The security of such constructions depends critically on both the block cipher's pseudorandomness and the structure of the surrounding mode. Foundational examples include the Data Encryption Standard (DES), introduced in 1977, and its successor, the Advanced Encryption Standard (AES), standardized in 2001 based on the Rijndael design [12]. The standardized version of Rijndael, also known as AES, supports a block size of 128 bits and key sizes of 128, 192, and 256 bits.

Although AES remains secure and widely deployed, the fixed block size of 128 bits introduces several practical considerations when encryption systems operate at very large scales. In modern computing environments, particularly in distributed cloud infrastructures, encryption schemes may process extremely large volumes of data under long-lived keys. In such settings, practical deployments must account for constraints that arise both from the finite block size of the underlying cipher and from requirements imposed by widely used encryption modes. For example, large data volumes may place pressure on the amount of data that can be safely encrypted under a single key, potentially requiring periodic key rotation. In addition, authenticated encryption modes such as GCM require careful management of nonces in order to avoid reuse of the same (key, IV) pair. These considerations have motivated renewed interest in block cipher designs with larger block sizes that provide greater security margins for high-volume encryption workloads [32]. Such constraints are well understood within the framework of provable security for block-cipher–based encryption modes.

Beginning with the work of Bellare and Rogaway [4], provable security—also known as reduction-based security—became the standard framework for analyzing block cipher modes. For encryption modes, a particularly strong notion of privacy, indistinguishability from random strings, was introduced in [23]. This definition implies the more traditional privacy notions from [2] and models an adversary operating under an adaptive chosen-plaintext attack (CPA) scenario. The adversary's goal is to distinguish the ciphertext from a uniformly random string of the same length, where the counter (CTR) value is not considered to be a part of the ciphertext.

The CTR mode was first formally analyzed by Bellare, Desai, Jokipii, and Rogaway [2], and a nonce-based formulation was later proposed by Rogaway [48]. Under the assumption that the underlying block cipher behaves as a secure pseudorandom permutation (PRP), their proof shows that an adversary making σ block queries has a distinguishing advantage bounded by $\frac{\sigma(\sigma-1)}{2^{n+1}}$. This is the well-known birthday bound, which arises from the collision probability among n-bit blocks.

The same birthday-bound limitation appears in foundational work on pseudorandom permutations. In particular, Luby and Rackoff [41] proved that a 4-round Feistel construction is computationally indistinguishable from a truly random permutation if each round function is an n-bit pseudorandom function [30]. Their proof holds even under chosen-ciphertext attack (CCA) for up to $q << 2^{n/2}$ queries, yielding $O(2^{n/2})$-security—again, security limited to the birthday bound.

While this bound represents a natural ceiling for many symmetric constructions, later research showed that beyond-birthday-bound security can be achieved in certain cases, such as stateful encryption and stateful authentication [3]. Nevertheless, AES, with its fixed 128-bit block size, remains inherently limited by the birthday bound, which becomes insufficient for modern cloud-scale data processing volumes [11]. This motivates renewed attention to Rijndael-256, the broader variant of the Rijndael cipher proposed by Daemen and Rijmen (1998) [12], which supports block sizes up to 256 bits. By doubling the block size relative to AES, Rijndael-256 extends the theoretical security margin beyond the birthday limit of AES, making it a compelling candidate for high-throughput or long-lived encryption systems that demand stronger guarantees against large-data-volume attacks.

In the new wider variant of AES, as NIST calls it, only a

*Authors are listed in alphabetical order.

block size of 256 bits and a key size of 256 bits are supported. In the rest of the paper, the term Rijndael-256 refers to the wider variant of AES with a block size of 256 bits and a key size of 256 bits.

A. Rijndael-256 specification

Rijndael-256 maintains a structure largely analogous to AES, with minor variations in certain transformation operations. One obvious difference is the state matrix of Rijndael-256 of 32 bytes (4x8 bytes) compared to the 4x4-byte state matrix in AES. Since $SubBytes$, $MixColumns$, and $AddRoundKey$ operations are not dependent on block size, they are similar to AES except they are applied to a bigger state matrix. On the other hand, $ShiftRows$ and $KeyExpansion$ are dependent on the block size, therefore these operations are slightly different in AES and Rijndael-256.

1) Data Representation: All data in Rijndael-256 is represented as a two-dimensional 4×8 array of bytes [14]. Each byte represents the binary coefficients of a polynomial in $GF(2^8)$. See Figure 1 below.

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$	$a_{0,6}$	$a_{0,7}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$	$a_{1,6}$	$a_{1,7}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$	$a_{2,6}$	$a_{2,7}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$	$a_{3,6}$	$a_{3,7}$

Fig. 1: Data Representation for Rijndael-256

2) SubBytes: This step provides **confusion** so attacker can't learn about the plaintext from the ciphertext. It's the only non-linear step. It works at the byte level. It substitutes each byte with another based on an S-box lookup table. The mathematics of the S-box is a multiplicative inverse followed by an algebraic affine transformation. The triangles in Figure 2 below indicate that each byte is individually and independently substituted.

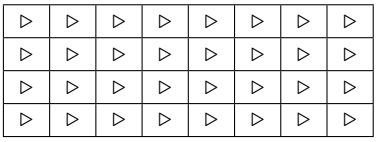


Fig. 2: $SubBytes$ in Rijndael-256

3) ShiftRows and MixColumns: These two steps together provide **diffusion**. The purpose of diffusion is that one bit change in the input has an avalanche effect and spreads to many bits of the output. The specific effect of $ShiftRows$ is to cause a permutation of the columns. $ShiftRows$ is dependent on the block size, therefore the operation is slightly different between AES and Rijndael-256. It is defined to have different shift offsets depending on the block size, i.e., for a 128-bit block it is [0,1,2,3] and for Rijndael-256 it is [0,1,3,4], where the 4 entries in the array indicate shift offset on each row. Figure 3 illustrates the $ShiftRows$ effect by showing the state array before (top) and after (bottom) the shift. Notice that each column is represented by one digit before the shift.

1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8

1	2	3	4	5	6	7	8
2	3	4	5	6	7	8	1
4	5	6	7	8	1	2	3
5	6	7	8	1	2	3	4

Fig. 3: The Effect of $ShiftRows$ on Rijndael-256

The specific effect of $MixColumms$ is to spread each byte of the column to the other bytes of the same column.

4) KeyExpansion and AddRoundKey: Key expansion operation is another operation that is dependent on block size. In AES, a round key has 4 words, and the g(x) ($ShiftRow$, $SubWord$, and $RoundConst$) operation is applied on every 4th word of the previous round key before XORing it with the first word of the next round key. In Rijndael-256, since a round key contains 8 words, the g(x) operation is applied on every 8th word of the previous round key before XORing it with the first word of the next round key. In addition, the $SubWord$ operation is applied to the 3rd word of the current round key before it gets XORed with the 4th word of the previous round key. Figure 4 illustrates the keys expansion operation for Rijndael-256. $AddRoundKey$ is an operation that is essential for confusion by mixing round keys with the state matrix using a bitwise XOR. The only difference in $AddRoundKey$ operation for Rijndael-256 and AES is the size of state matrices/round keys, which is 256-bit and 128-bit, respectively.

B. Unique features of Rijndael-256 as opposed to AES

1) ShiftRows for wider blocks: Diffusion optimality is a common design criterion for all block sizes where $ShiftRows$ offsets are made different for each row. This will provide resistance against differential and linear cryptanalysis. The other diffusion effects are only relevant for blocks wider than 128 bits, where they seek to maximize resistance against truncated differential and saturated attacks. For 128-bit block size, diffusion optimality is accomplished by the simple shift offsets 0, 1, 2, and 3. For wider block sizes, there are more possibilities. More research should be done here to find the optimal choice of row offsets. So far, detailed studies of truncated differential attacks and saturation attacks on reduced versions of Rijndael have shown that not all choices are the same [14].

2) number of rounds for wider blocks: For wider Rijndael, the number of rounds is raised by one for every additional 32 bits in block size for two reasons. First, it takes 3 rounds to realize full diffusion in wider blocks while it takes only 2 rounds in 128-bit blocks. Second, wider block increases the range of possible patterns that can be applied to the input of a round, which in turn gives more flexibility to the attacker.

3) wider blocks are free: In Rijndael, basic operations and structures, bytes and columns, keep their length when the

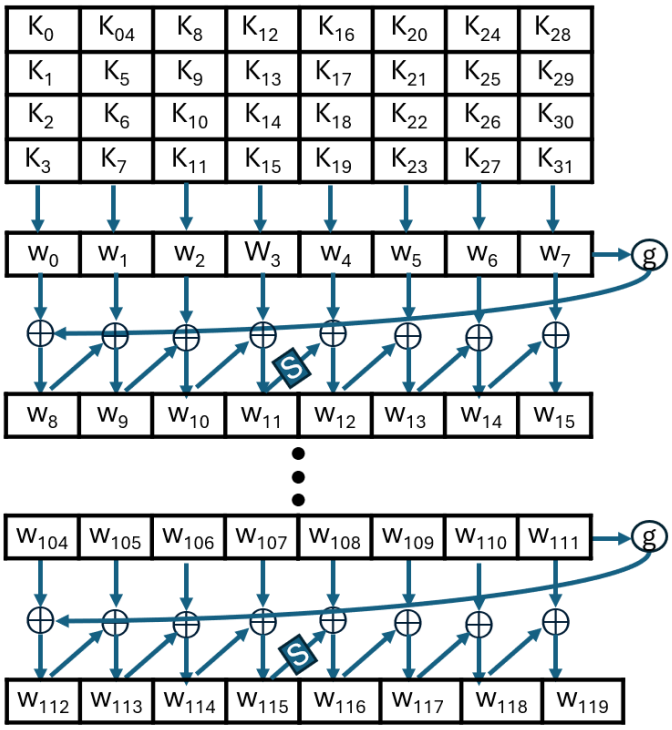


Fig. 4: Rijndael-256 Key Expansion Operation

block size increases. This gives Rijndael a unique advantage that block size can be increased without affecting its computational cost per byte on any platform.

C. Main Considerations for Rijndael-256

When switching from AES to Rijndael-256, there are several issues to consider. One can't simply expect to swap AES with Rijndael without causing functional, security, or performance issues. Below are some examples of issues that could arise when considering the wider Rijndael block cipher.

1) Fourteen rounds and security margins: Recent research, a 2024 paper [51], shows that 7-round Rijndael-256 does not have any Impossible Differentials (ID) under sole assumption that round keys are uniformly random. IDs are useful for cryptanalyst to discard wrong keys. This means that a 14-round Rijndael-256 still has enough security margin, hence it's secure from ID attacks. A 2013 paper [9] shows that maximal algebraic degree can't be attained till the 7th round of Rijndael-256. Though it was originally thought that a much smaller number, only 3 rounds, was enough to attain maximal degree, 7 rounds still provide enough margin of nonlinearity for a 14-round cipher. A 2009 paper [45] computes the bounds on integral attacks. For Rijndael-256, a 4-round distinguisher requires 2^{16} plaintexts. The best distinguisher reaches 6 rounds but requires 2^{192} plaintexts which makes it a non-practical attack. Another paper of the same year [52] looks at 3-round Integral Distinguisher of Rijndael-256. [35] has the original integral attack. A 2005 paper [46] includes an impressive table that displays the complexity bounds of a multiset attack on Rijndael with all sizes larger than 128 bit blocks. It shows the best attack on Rijndael-256 to reach 7 rounds with complexities of $6 * 2^{128}$ plaintext sets, 2^{128} memory size, and 2^{144} computations.

2) The Speed of Rijndael-256: Research is limited here. One of the few papers that addresses this issue properly is a 2022 paper [19] that looks at Rijndael-256 speed with various optimizations including AES-NI. It shows the proper way of optimization and warns against wrong ones that could invite timing attacks and some key leakage. It reports a record speed of 0.27 cycles per byte for Rijndael-256 after exploring several implementations.

3) Potential Misconfiguration: Since the design of Rijndael decouples block size from key size, there is a potential of a mismatch. For example, using a 128-bit key with a 256-bit block.

4) Rijndael-256 incompatibility with common modes: Rijndael-256 might be incompatible with modes of operations optimized for 128-bit blocks. A good example is the Galois/Counter Mode (GCM). GCM is optimized for 128-bit block size. Changing GCM to Rijndael-256 means the standards have to change and new security proofs have to be developed. Consider using new modes tailored for 256-bit blocks such as Accordion. Also watch for implementation issues.

5) Attacks on AES inherited by Rijndael-256: Although we will explore attacks in greater detail later, this is to remind the reader that some AES attacks will carry over to Rijndael-256, despite their differences outlined above. A 2011 paper [8] looks at Biclique attack. Although this attack is on AES (Rijndael-128), it is important to consider since it exploits weaknesses in the key schedule; something that Rijndael-256 could inherently have. A 2020 paper [1] that looks at improved key recovery attacks on AES can also be considered to see if it extends to Rijndael-256.

D. Related Work

Existing approaches to achieving large-domain (wide-block) encryption follow two main directions: some designs natively operate on wide blocks, while others rely on domain-extension techniques to construct larger-domain ciphers from smaller primitives. Besides Rijndael (in its 256-bit block variant), examples of native wide-block ciphers include Threefish [21], SHACAL-2 (SHA-256–based variant of SHACAL [28]), and Kalyna [24]. Domain-extension techniques include Feistel-based constructions (e.g., the Luby–Rackoff construction [42]), arbitrary-domain and format-preserving encryption methods (e.g., Black and Rogaway [7]), and shuffle-based constructions such as The Swap-or-Not Shuffle [29], as well as wide-block encryption modes such as CMC and EME [26], [27].

Liu et al. [39] study large-block variants of Rijndael and present improved impossible differential attacks on several rounds.

Through a new symmetric cryptography design, Bhaumik et al. [5] claim to provide n-bit security, where n is the block size of the underlying block cipher, against a quantum adversary by doubling the size of blocks and keys.

Drucker and Gueron [19] describe SW optimizations for fast computations of Rijndael-256 on modern x86-64 CPU's using vectorized AES-NI. They report a speed record of 0.27 cycles/byte.

II. Provable Security of block ciphers

Unlike public-key schemes, where security is often tied to hard mathematical problems, block ciphers rely on their structural design for strength—making formal guarantees much harder to come by. Despite their widespread use, we still lack comprehensive proofs of security for most concrete block cipher designs. Instead, we must rely on frameworks that allow us to make meaningful claims within specific models or against defined classes of attacks. In the following two subsections, we review two important notions—indifferentiability and t-wise independence—that help frame these guarantees. Both offer a way to reason about security in the absence of traditional reductions, and each captures a different but complementary aspect of what it means for a block cipher to behave like an ideal primitive.

1) Indifferentiability: In the absence of reductions to well-defined mathematical problems, one approach to reasoning about the security of block ciphers is to ask whether they can safely replace an ideal primitive in a larger cryptographic construction. This perspective leads to the notion of indifferentiability, introduced by Maurer, Renner, and Holenstein [43], which allows to reason about the security of composed systems by asking whether a real construction behaves like an ideal component when plugged into an arbitrary cryptographic environment. This is especially relevant in the analysis of domain extension frameworks, such as hash functions built from compression functions or wide/tweakable block ciphers built from standard ones.

Indifferentiability is considerably stronger than indistinguishability. While indistinguishability focuses on whether an adversary can tell two systems apart in a fixed, standalone setting (typically defined by a single oracle or game), indifferentiability asks whether the real construction can be substituted for the ideal one in any context without the adversary noticing, assuming that a simulator has access to the ideal primitive. This makes it a powerful tool for reasoning about how cryptographic components interact, not just how they perform in isolation.

For block ciphers based on Substitution-Permutation Networks (SPNs), proving indifferentiability to a random permutation is a challenging task. The work of Cogliati et al. [10] analyzes SPNs in a model where the permutation layer is a public random permutation and the S-boxes are deterministic and public. They show that when the permutation layer is linear—as is the case in most real-world SPN constructions—three rounds are both necessary and sufficient to achieve provable security in their model. Interestingly, even a single-round SPN can be secure if the permutation is allowed to be non-linear. Their results demonstrate that adding non-linearity can significantly improve security bounds: two-round non-linear SPNs can achieve "beyond-birthday" security (up to $2^{2n/3}$ adversarial queries), and with more rounds, their bounds scale well to nearly the full 2^n query space.

The situation is less favorable for standard SPNs like AES. These ciphers use small S-boxes (e.g., 8-bit in AES or Rijndael-256), and Cogliati et al. identify a fundamental "2^n barrier" in current provable models. As a result, SPNs using small S-boxes cannot, under this model, reach strong indifferentiability bounds regardless of the number of rounds. This is a structural limitation rather than a mere technical shortcoming, and the authors highlight it as one of the key open challenges in symmetric-key cryptography—namely, how to develop a sound theory for building secure block ciphers from small components.

2) t-wise Independence: Security validation of block ciphers has historically relied on cryptanalysis techniques including linear, differential, impossible differential, algebraic, and integral attacks.

In the absence of full security proofs, the focus shifts to demonstrating that certain broad classes of attacks cannot succeed. Among the strongest such notions is t-wise independence, which measures how close the outputs of a block cipher on any t distinct inputs are to being independent and uniformly distributed. Formally, for a block cipher $E : \{0,1\}^s \times \{0,1\}^n \to \{0,1\}^n$, t-wise independence demands that the joint distribution of $E(S, x_1), \ldots, E(S, x_t)$ for distinct inputs x_i and a uniformly random key S is statistically close to t-wise independent uniform n-bit strings.

This property is significant because it implies resilience to many known attacks: pairwise independence (the case $t = 2$) already guarantees robustness against linear and differential cryptanalysis as well as truncated differential attacks, while higher values of t imply protection against order $log_2(t)$ differential attacks and beyond.

Concrete security analyses highlight the challenge: Liu, Tessaro, and Vaikuntanathan [38] showed that over 9000 rounds of AES are needed to achieve closeness to pairwise independence, an impractical figure.

Recent work has made significant strides in quantifying the round complexity needed for achieving statistical closeness to t-wise independence. In particular, the work of Liu et. al. [37] shows that for a stripped-down version of AES called censored-AES—where most of the mixing layers are removed—only 192 rounds suffice to achieve 2^{-128}-closeness to pairwise independence.

In conclusion, current proof frameworks provide a more detailed view of the security of Rijndael-based designs. For standard Rijndael (AES parameters), which uses small S-boxes, existing indifferentiability analyses for SPN ciphers identify a structural limitation: in current models, SPNs built from small S-boxes cannot achieve strong indifferentiability bounds regardless of the number of rounds. Similarly, results on t-wise independence indicate that achieving strong statistical independence guarantees would require an impractically large number of rounds for AES-like structures, with even pairwise independence requiring thousands of rounds in analyzed

settings. These limitations are primarily driven by the size of the nonlinear components and the round structure.

For wide-block Rijndael variants, which operate over larger state sizes, the picture is more complicated. The same limitations due to small S-boxes still apply, but increasing the block size changes how the bounds scale. In particular, a larger state implies more parallel nonlinear operations per round and hence improved diffusion across the state, which can lead to better guarantees in analyses based on independence and mixing. As a result, wide-block variants tend to fit better within frameworks where meaningful statistical and compositional bounds can be obtained, even if the underlying limitations are not removed.

III. Detailed Survey of Attacks

Block cipher cryptanalysis is slow. Over the last thirty years, tens of papers were published on the cryptanalysis of block ciphers. Of special interest was reduced-round AES, or its more general form Rijndael. The common denominator of these attacks is key recovery. However, only few techniques became seminal. It's important to note that in the case of block cipher cryptanalysis, it is common for a technique to take years to mature. It usually starts with the invention of a technique, then several incremental improvements till we arrive to current form. In this section, we survey the main attacks and how they specifically fare with Rijndael-256.

A. Integral Attack

The integral attack is the oldest and most effective attack on Rijndael. It dates back to an attack on the Square cipher, hence it was initially called the "Square attack" in 1997 [13]. Later it came to be known by several names such as saturation attack (2000), structural attack (2001), partial sum, and multiset attack. All these attacks were unified under one name, the integral attack, by Knudsen in 2002 [34]. The essence of integral attack is when the sum or integral of many chosen plaintexts (CP) after going through some rounds, at one or several byte locations of the Rijndael state, equals to zero. The most obvious case is when we rotate all 256 values of one byte. This usually uncovers a structural anomaly that distinguishes Rijndael from an ideal cipher.

The basic attack started on 4-round reduced AES. This attack can be extended from the end or from the beginning to a 5-round attack. It can also be extended from both ends to a 6-round attack. A further extension is called the herds attack (herds = 2^{120} encryptions) where 7-round is attained. An extension of herds lands an 8-round attack on AES. Table I summarizes these attacks on AES.

Ferguson [20] in his seminal paper on partial sums attacks says, "Although similar in structure, Rijndael with larger block sizes is different enough—the byte alignments that are so crucial to some of our attacks are different—that it will have to be analyzed separately." In 2005, Nakahara [46] was first to provide results for reduced-round integral attacks on wider Rijndael variants including Rijndael-256. He attained 7-round attacks with complexities above 2^{128}. Later, Minier in 2009 [45] and 2012 [44], and Li in 2011 [36] attained higher rounds like 8 and 9 with higher complexities (See Table II for full details). Those could be attained by higher order integrals $(256)^2$, $(256)^3$, $(256)^4$, etc. Todo (2015) [49] introduced a variant of integral attack based on a new property called the Division Property. This property seems to be more effective on Rijndael-like ciphers rather than Rijndael itself. In fact, the complexity numbers he showed for even a 4-round Rijndael-256 were very prohibitive (2^{232} CP, less than 2^{30} from exhaustive attack). The Division Property took the integral cryptanalysis in a different direction away from Rijndael. Hence, no significant improvements have been recorded in recent years for integral attacks on Rijndael, let alone its wider variants.

Attack	Rounds	No. of CP	No. of Encryptions	Memory
Basic	4	2^9	2^{14}	small
Ext. Basic (from end)	5	2^{11}	2^{46}	small
Ext. Basic (from start)	5	2^{33}	2^{38}	2^{32}
Double-extended Basic	6	2^{35}	2^{46}	2^{32}
Herds	7	$2^{128} - 2^{119}$	2^{120}	2^{64}
Extended Hers	8	$2^{128} - 2^{119}$	2^{204}	2^{104}

TABLE I: Summary of integral attacks on AES

Paper	Rounds	Data Comp.	Time Comp.
Nakahara (2005) [46]	7	$6 * 2^{128}$	2^{144}
Galice et al. (2008) [22]	9	$2^{128} - 2^{119}$	2^{204}
Minier et al. (2009) [45]	6	2^{192}	NA
Wei et al. (2009) [52]	3	32	NA
Li et al. (2011) [36]	9	2^{132}	NA
Minier et al. (2012) [44]	5	2^{32}	NA
Todo (2015) [49]	4	2^{232}	NA

TABLE II: Summary of integral attacks on Rijndael-256

B. Impossible Differential Attack

Impossible differential cryptanalysis was independently introduced by Knudsen [33] and Biham et al. [6] in 1999. The core idea is to construct a differential path with zero probability—known as an impossible differential—to eliminate incorrect key candidates. As implied from its name, this attack is based on the differential characteristics of a block cipher. Input differences are usually represented by symbol Δ while output differences are represented by symbol δ. Hence, a differential (Δ, δ) is when a Δ at the input produces a δ at the output. We usually look at the probability of a differential $Pr[\Delta \rightarrow \delta]$. An Impossible Differential (ID) is when $Pr[\Delta \rightarrow \delta] = 0$ or when a particular Δ at the input can never produce a particular δ at the output, hence the main focus of the attack is to search and find IDs in a block cipher. This is not as trivial as it may sound. Consider a block cipher with n-bit block size, the space of non-zero Δs is 2^n. Same for the space of non-zero δs. If every possible Δ maps to every

possible δ, then the total differential search space is 2^{2n}. This makes the complexity of exhaustive ID search far greater than the exhaustive key search itself. This also reveals that the main strategy of improving this attack lies in improving the search method for IDs.

Impossible differential attacks reported in the literature target several large-block variants of the Rijndael cipher family. Although this work focuses on Rijndael-256, cryptanalytic studies often consider several variants of the Rijndael family with different block and key sizes. Because these variants share the same round structure and core transformations ($SubBytes$, $ShiftRows$, $MixColumns$, and $AddRoundKey$), techniques developed for one configuration are often expected to apply, with minor adaptations, to other members of the family. In the literature, the notation Rijndael-b typically denotes the variant with a b-bit block size, while the key size may be specified separately when needed. When both parameters are given, the notation Rijndael-b-k is used, where b denotes the block size and k denotes the key size in bits. For example, Rijndael-224-256 refers to the variant with a 224-bit block and a 256-bit key. The main impossible differential attacks reported in the literature on these large-block Rijndael variants are as below and summarized in Table III.

- In 2007, Nakahara and Pavao [47] presented impossible differential attacks targeting seven rounds of both Rijndael-224 and Rijndael-256.
- In 2008, Zhang et al. [53] introduced impossible differential attacks on 9-round versions of Rijndael-224 and Rijndael-256.
- In 2012, Wang et al. [50] improved upon earlier results by mounting impossible differential attacks on 9-round Rijndael-224 and 10-round Rijndael-256.
- In 2019, Wang et al. [40] proposed enhanced impossible differential attacks on 10-round Rijndael-256-256, 10-round Rijndael-224-256, and 9-round Rijndael-224-224. Their approach leveraged precomputation tables, redundancies in the key schedule, and multiple overlapping impossible differentials to extend the reach of the attack.

Attack	Rounds and B-length	Data Comp.	Time Comp.
Nakahara & Pavao (2007)	7 (Rijndael-224)	2^{138}	2^{167}
Nakahara & Pavao (2007)	7 (Rijndael-256)	2^{153}	2^{182}
Zhang et al. (2008)	9 (Rijndael-224)	$2^{212.3}$	2^{209}
Zhang et al. (2008)	9 (Rijndael-256)	$2^{244.3}$	$2^{208.8}$
Wang et al. (2012)	7 (Rijndael-224)	$2^{93.2}$	$2^{113.4}$
Wang et al. (2012)	9 (Rijndael-224)	$2^{198.1}$ or 2^{216}	$2^{195.2}$ or 2^{130}
Wang et al. (2012)	9 (Rijndael-256)	$2^{229.3}$ or $2^{245.3}$	2^{194} or $2^{127.1}$
Wang et al. (2012)	10 (Rijndael-256)	$2^{244.2}$	$2^{253.8}$
Wang et al. (2019)	10 (Rijndael-224-224)	$2^{113.4}$ or $2^{153.6}$	$2^{214.4}$ or $2^{206.6}$
Wang et al. (2019)	10 (Rijndael-224-256)	$2^{241.3}$	$2^{214.4}$
Wang et al. (2019)	10 (Rijndael-256-256)	$2^{244.4}$	$2^{240.1}$

TABLE III: Summary of impossible differential attacks on large-block Rijndael

C. Meet-in-the-middle (MitM) Attacks

Fundamental intuition of the MitM attack technique is that we split the n rounds of the block cipher into three consecutive parts: $n = n_1 + n_2 + n_3$ and find a distinguishing property that certain input messages would exhibit (or "verify", as some of the prior work calls it) in the middle n_2 rounds, then use this information to eliminate wrong key guesses. The attack typically involves two phases: offline precomputation of a lookup table based on the distinguishing property and online trials. The latter phase first identifies the messages verifying the given property, then comes from "the other end" and uses the corresponding partial decryption through n_3, checking whether the result belongs to the precomputed table [17]. The technique is generic and its successful application hinges upon finding a good property to target and formalizing it. A suboptimal property or the way of describing and storing it in the lookup table may lead to attack resource blowup and lower attack efficiency. Over the years, many variations of attacks using the MitM technique were proposed: [15], [8], [17], [16], [1], to name but a few. They typically use various hybrid approaches to construct full attack flows, such as using collisions, bicliques, or differential attacks (or their elements) for optimizing and aiding the MitM one. None of the sources we surveyed covered Rijndael-256, although some do cover the block sizes longer than 128 bits and hint at MitM attacks being less effective than Impossible Differential ones [18]. In the hashing space, MitM attacks are well-used and some of the works include Rijndael-256 analysis in the context of preimages [54].

To sum up, while in principle we expect this class of attacks to be applicable due to its generic nature, finding specific instantiations and particular "helper" properties for optimizing the search, and details of their applicability to the Rijndael encryption in the wide-block setting, remain an open research question.

IV. Implementation and Optimization

Building upon the Rijndael-256 specification detailed in the Introduction, this section demonstrates optimized implementation of Rijndael-256 using AES New Instructions (AES-NI).

A. Optimization using AES-NI

AES has been implemented and optimized on a variety of platforms in software, firmware, and hardware. The AES New Instructions (AES-NI) has been one of the widely used instruction sets to optimize AES on x86 platforms. Table IV shows the list of AES-NI instructions added to the x86 instruction set that help AES implementation in software since 2010.

Instruction	Description
AESENC	Single round of AES encryption.
AESENCLAST	Last round of AES encryption.
AESDEC	Single round of AES decryption.
AESDECLAST	Last round of AES decryption.
AESIMC	Decryption key schedule.
AESKEYGENASSIST	Assists generating round keys for encryption.

TABLE IV: x86 AES New Instructions Set

AES-NI is primarily designed for the implementation of the FIPS-197 standard AES. Gueron [25], in his paper that introduces the performance and security benefits of AES-NI, demonstrated that the agile design of the new AES instruction

sets allows us to implement different variations of Rijndael, including the wide-block AES (Rijndael-256). The pseudo-code in Listing 1 in Appendix A shows how Rijndael-256 can be implemented using AES-NI intrinsics [31].

The 256-bit input is stored in a two 128-bit registers data1 and data2, the main goal is to apply AES-NI instructions to each half of the input data and be able to achieve the desired Rijndael-256 output. The BLEND_MASK and SHUFFLE_MASK at lines 4 & 5 are used to compensate for the differences in $ShiftRows$ transformation in Rijndael-256 compared to AES. When $ShiftRows$ is applied, values circularly shift: byte values in data1 will move to data2, and the same number of bytes will move from data2 to data1. This is easily implemented using the _mm_blendv_epi8 intrinsic with BLEND_MASK values. In addition, Rijndael-256 applies an extra shift on rows 3 and 4. Before processing each 128-bit block separately using AESENC, one additional left circular shift is applied on rows 3 and 4 using the shuffling intrinsic, _mm_shuffle_epi8, with SHUFFLE_MASK values. Finally, the AESENC instruction, _mm_aesenc_si128, with round keys is applied to each 128-bit block separately and gives the desired output. The same idea applies to the last round transformation.

In [19], the authors proposed an optimization technique targeting the Rijndael-256 block cipher on capable modern x86-64 platforms. The main capabilities that are utilized in the paper are vectorized AES instructions: VAESENC, VAESENCLAST, VAESDEC, and VAESDECLAST. The authors demonstrate how the VPERMB instruction can replace the sequential operations of VPBLENDVB and PSHUFB, consolidating them into a single instruction. This approach reduces register dependencies, thereby enhancing execution efficiency and minimizing latency by shortening dependency chains. Such optimizations are crucial for high-performance computing environments, where rapid processing of large data volumes is essential. The authors have achieved record performance of 0.27 cycles per byte for Rijndael-256 in ECB mode on then-latest 10th Intel® Core™ Generation (Micro architecture Codename "Ice Lake"[ICL]), demonstrating significant improvements in encryption throughput on capable processors.

REFERENCES

[1] A. Bar-On, O. Dunkelman, N. Keller, E. Ronen, and A. Shamir, "Improved Key Recovery Attacks on Reduced-Round AES with Practical Data and Memory Complexities," *Journal of Cryptology*, vol. 33, no. 3, pp. 1003–1043, Jul. 2020. [Online]. Available: https://doi.org/10.1007/s00145-019-09336-w

[2] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway, "A concrete security treatment of symmetric encryption," in *Proceedings 38th annual symposium on foundations of computer science*. IEEE, 1997, pp. 394–403.

[3] M. Bellare, K. G. Paterson, and P. Rogaway, "Security of symmetric encryption against mass surveillance," in *Annual cryptology conference*. Springer, 2014, pp. 1–19.

[4] M. Bellare and P. Rogaway, "The security of triple encryption and a framework for code-based game-playing proofs," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2006, pp. 409–426.

[5] R. Bhaumik, A. Chailloux, P. Frixons, B. Mennink, and M. Naya-Plasencia, "Block Cipher Doubling for a Post-Quantum World," 2022, publication info: Preprint. [Online]. Available: https://eprint.iacr.org/2022/1342

[6] E. Biham, A. Biryukov, and A. Shamir, "Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1999, pp. 12–23.

[7] J. Black and P. Rogaway, "Ciphers with arbitrary finite domains," in *Topics in Cryptology – CT-RSA*, 2002.

[8] A. Bogdanov, D. Khovratovich, and C. Rechberger, "Biclique Cryptanalysis of the Full AES," in *Advances in Cryptology – ASIACRYPT 2011*, D. H. Lee and X. Wang, Eds. Berlin, Heidelberg: Springer, 2011, pp. 344–371.

[9] C. Boura and A. Canteaut, "On the Influence of the Algebraic Degree of F^{-1} on the Algebraic Degree of G \circ F," *IEEE Transactions on Information Theory*, vol. 59, no. 1, pp. 691–702, Jan. 2013, conference Name: IEEE Transactions on Information Theory. [Online]. Available: https://ieeexplore.ieee.org/document/6303910/?arnumber=6303910

[10] B. Cogliati, Y. Dodis, J. Katz, J. Lee, J. Steinberger, A. Thiruvengadam, and Z. Zhang, "Provable security of (tweakable) block ciphers based on substitution-permutation networks," in *Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part I 38*. Springer, 2018, pp. 722–753.

[11] I. T. L. Computer Security Division, "NIST Proposes to Standardize a Wider Variant of AES | CSRC," Dec. 2024. [Online]. Available: https://csrc.nist.gov/News/2024/nist-proposes-to-standardize-wider-variant-of-aes

[12] J. Daemen, "Aes proposal : Rijndael," 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:17885291

[13] J. Daemen, L. Knudsen, and V. Rijmen, "The block cipher Square," in *Fast Software Encryption*, E. Biham, Ed. Berlin, Heidelberg: Springer, 1997, pp. 149–165.

[14] J. Daemen and V. Rijmen, *The Design of Rijndael: The Advanced Encryption Standard (AES)*, 2nd ed., ser. Information Security and Cryptography Ser. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2020.

[15] H. Demirci and A. A. Selçuk, "A Meet-in-the-Middle Attack on 8-Round AES," in *Fast Software Encryption*, K. Nyberg, Ed. Berlin, Heidelberg: Springer, 2008, pp. 116–126.

[16] P. Derbez, "Meet-in-the-Middle Attacks on AES," phdthesis, Ecole Normale Supérieure de Paris - ENS Paris, Dec. 2013. [Online]. Available: https://theses.hal.science/tel-00918146

[17] P. Derbez, P.-A. Fouque, and J. Jean, "Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting," in *Advances in Cryptology – EUROCRYPT 2013*, T. Johansson and P. Q. Nguyen, Eds. Berlin, Heidelberg: Springer, 2013, pp. 371–387.

[18] X. Dong and Y. Wei, "New attacks against reduced Rijndael-160," *IET Information Security*, vol. 16, no. 1, pp. 28–40, Jan. 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1049/ise2.12038

[19] N. Drucker and S. Gueron, "Software Optimization of Rijndael for Modern x86-64 Platforms," in *ITNG 2022 19th International Conference on Information Technology-New Generations*, S. Latifi, Ed. Cham: Springer International Publishing, 2022, pp. 147–153.

[20] N. Ferguson, J. Kelsey, S. Lucks, B. Schneier, M. Stay, D. Wagner, and D. Whiting, "Improved Cryptanalysis of Rijndael," in *Fast Software Encryption*, G. Goos, J. Hartmanis, J. Van Leeuwen, and B. Schneier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, vol. 1978, pp. 213–230, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/3-540-44706-7_15

[21] N. Ferguson, S. Lucks, B. Schneier, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker, "The skein hash function family," in *Submission to NIST (SHA-3 Competition)*, 2010.

[22] S. Galice and M. Minier, "Improving Integral Attacks Against Rijndael-256 Up to 9 Rounds," in *Progress in Cryptology – AFRICACRYPT 2008*, S. Vaudenay, Ed. Berlin, Heidelberg: Springer, 2008, pp. 1–15.

[23] S. Goldwasser and S. Micali, "Probabilistic encryption & how to play mental poker keeping secret all partial information," in *Providing sound foundations for cryptography: on the work of Shafi Goldwasser and Silvio Micali*, 2019, pp. 173–201.

[24] O. O. Gorbenko *et al.*, "Kalyna block cipher," Ukrainian National Standard, 2015.

[25] S. Gueron, "Intel advanced encryption standard (aes) new instructions set," 2010.

[26] S. Halevi and P. Rogaway, "A tweakable enciphering mode," in *Advances in Cryptology – CRYPTO 2003*, ser. LNCS, vol. 2729. Springer, 2003, pp. 482–499.

[27] ——, "A parallelizable enciphering mode," in *Topics in Cryptology – CT-RSA 2004*, ser. LNCS, vol. 2964. Springer, 2004, pp. 292–304.

[28] H. Handschuh, D. Naccache, and L. Rogier, "Shacal: A family of block ciphers," in *Fast Software Encryption (FSE)*, 2001.

[29] V. T. Hoang, B. Morris, and P. Rogaway, "An enciphering scheme based on a card shuffle," in *CRYPTO*, 2012.

[30] R. Impagliazzo, L. A. Levin, and M. Luby, "Pseudo-random generation from one-way functions," in *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, 1989, pp. 12–24.

[31] Intel Corporation, "Intel Intrinsics Guide," 2024, online interactive reference guide. [Online]. Available: https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html

[32] P. Kampanakis *et al.*, "Practical challenges with aes-gcm and the need for a 256-bit block cipher," in *NIST Workshop on Block Cipher Modes of Operation*, 2023.

[33] L. Knudsen, "Deal-a 128-bit block cipher," *complexity*, vol. 258, no. 2, p. 216, 1998.

[34] L. Knudsen and D. Wagner, "Integral Cryptanalysis (Extended abstract)."

[35] ——, "Integral Cryptanalysis," in *Fast Software Encryption*, J. Daemen and V. Rijmen, Eds. Berlin, Heidelberg: Springer, 2002, pp. 112–127.

[36] Y.-J. LI and W.-L. WU, "Improved Integral Attacks on Rijndael." *Journal of Information Science & Engineering*, vol. 27, no. 6, pp. 2031–2045, Nov. 2011, publisher: Institute of Information Science, Academia Sinica. [Online]. Available: https://research.ebsco.com/linkprocessor/plink?id=707b7952-1d2f-37f7-8173-15a5a4820a62

[37] T. Liu, A. Pelecanos, S. Tessaro, and V. Vaikuntanathan, "Layout graphs, random walks and the t-wise independence of spn block ciphers," in *Annual International Cryptology Conference*. Springer, 2023, pp. 694–726.

[38] T. Liu, S. Tessaro, and V. Vaikuntanathan, "The t-wise independence of substitution-permutation networks," in *Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part IV 41*. Springer, 2021, pp. 454–483.

[39] Y. Liu, Y. Shi, D. Gu, B. Dai, F. Zhao, W. Li, Z. Liu, and Z. Zeng, "Improved impossible differential cryptanalysis of large-block Rijndael," *Science China Information Sciences*, vol. 62, no. 3, p. 32101, Jul. 2018. [Online]. Available: https://doi.org/10.1007/s11432-017-9365-4

[40] ——, "Improved impossible differential cryptanalysis of large-block rijndael," *Science China Information Sciences*, vol. 62, pp. 1–14, 2019.

[41] M. Luby and C. Rackoff, "How to construct pseudorandom permutations from pseudorandom functions," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 373–386, 1988.

[42] ——, "How to construct pseudorandom permutations from pseudorandom functions," in *SIAM Journal on Computing*, 1988.

[43] U. Maurer, R. Renner, and C. Holenstein, "Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology," in *Theory of Cryptography: First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004. Proceedings 1*. Springer, 2004, pp. 21–39.

[44] M. Minier, R. C.-W. Phan, and B. Pousse, "On Integral Distinguishers of Rijndael Family of Ciphers," *Cryptologia*, vol. 36, no. 2, pp. 104–118, Apr. 2012, publisher: Taylor & Francis _eprint: https://doi.org/10.1080/01611194.2012.635110. [Online]. Available: https://doi.org/10.1080/01611194.2012.635110

[45] M. Minier and B. Pousse, "Improving Integral Cryptanalysis against Rijndael with Large Blocks," Oct. 2009, arXiv:0910.2153 [cs]. [Online]. Available: http://arxiv.org/abs/0910.2153

[46] J. Nakahara, D. S. de Freitas, and R. C. W. Phan, "New Multiset Attacks on Rijndael with Large Blocks," in *Progress in Cryptology – Mycrypt 2005*, E. Dawson and S. Vaudenay, Eds. Berlin, Heidelberg: Springer, 2005, pp. 277–295.

[47] J. Nakahara Jr and I. C. Pavao, "Impossible-differential attacks on large-block rijndael," in *International Conference on Information Security*. Springer, 2007, pp. 104–117.

[48] P. Rogaway, "Evaluation of some blockcipher modes of operation," *Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan*, vol. 630, p. 11, 2011.

[49] Y. Todo, "Structural Evaluation by Generalized Integral Property," in *Advances in Cryptology – EUROCRYPT 2015*, E. Oswald and M. Fischlin, Eds. Berlin, Heidelberg: Springer, 2015, pp. 287–314.

[50] Q. Wang, D. Gu, V. Rijmen, Y. Liu, J. Chen, and A. Bogdanov, "Improved impossible differential attacks on large-block rijndael," in *International Conference on Information Security and Cryptology*. Springer, 2012, pp. 126–140.

[51] S. Wang, D. Feng, T. Shi, B. Hu, J. Guan, K. Zhang, and T. Cui, "New Methods for Bounding the Length of Impossible Differentials of SPN Block Ciphers," *IEEE Transactions on Information Theory*, vol. 70, no. 12, pp. 9165–9178, Dec. 2024, conference Name: IEEE Transactions on Information Theory. [Online]. Available: https://ieeexplore.ieee.org/document/10706887/?arnumber=10706887

[52] Y. Wei, B. Sun, and C. Li, "New Integral Distinguisher for Rijndael-256," 2009, publication info: Published elsewhere. Unknown where it was published. [Online]. Available: https://eprint.iacr.org/2009/559

[53] L. Zhang, W. Wu, J. H. Park, B. W. Koo, and Y. Yeom, "Improved impossible differential attacks on large-block rijndael," in *International Conference on Information Security*. Springer, 2008, pp. 298–315.

[54] T. Zhang, "Comprehensive Preimage Security Evaluations on Rijndael-Based Hashing," in *Applied Cryptography and Network Security Workshops*, J. Zhou, L. Batina, Z. Li, J. Lin, E. Losiouk, S. Majumdar, D. Mashima, W. Meng, S. Picek, M. A. Rahman, J. Shao, M. Shimaoka, E. Soremekun, C. Su, J. S. Teh, A. Udovenko, C. Wang, L. Zhang, and Y. Zhauniarovich, Eds. Springer Nature Switzerland, 2023, pp. 23–42.

APPENDIX A
CODE IMPLEMENTATION

Listing 1: Rijndael-256 implementation using AES-NI instructions

```
Initialize state matrices and round keys:
    data1, data2: 256-bit State matrix input stored in two 128-bit registers (xmm)
    roundKeys: Array containing 256-bit round keys for each encryption round
    SHUFFLE_MASK = _mm_set_epi32(0x03020d0c, 0x0f0e0908, 0x0b0a0504, 0x07060100)
    BLEND_MASK = _mm_set_epi32(0x80000000, 0x80800000, 0x80800000, 0x80808000)
For each round from 1 to number_of_rounds - 1:
    // Blend operation to adjust for shift rows between two 128-bit blocks
    tmp1 = _mm_blendv_epi8(data1, data2, BLEND_MASK)
    tmp2 = _mm_blendv_epi8(data2, data1, BLEND_MASK)
    // Shuffle operation to compensate for additional shifts in rows 3 and 4
    tmp1 = _mm_shuffle_epi8(tmp1, SHUFFLE_MASK)
    tmp2 = _mm_shuffle_epi8(tmp2, SHUFFLE_MASK)
    // Encryption step: sub bytes, shift rows, mix columns, XOR with round key
    data1 = _mm_aesenc_si128(tmp1, roundKeys[j * 2])
    data2 = _mm_aesenc_si128(tmp2, roundKeys[j * 2 + 1])
// Final round (without MixColumns)
tmp1 = _mm_blendv_epi8(data1, data2, BLEND_MASK)
tmp2 = _mm_blendv_epi8(data2, data1, BLEND_MASK)
tmp1 = _mm_shuffle_epi8(tmp1, SHUFFLE_MASK)
tmp2 = _mm_shuffle_epi8(tmp2, SHUFFLE_MASK)
data1 = _mm_aesenclast_si128(tmp1, roundKeys[number_of_rounds * 2])
data2 = _mm_aesenclast_si128(tmp2, roundKeys[number_of_rounds * 2 + 1])
```