Cryptography

AES, Decryption, Key Expansion

M S Vilkhu

26 Oct 2024 (C1/C3/C5)

28 Oct 2024 (C1/C3)

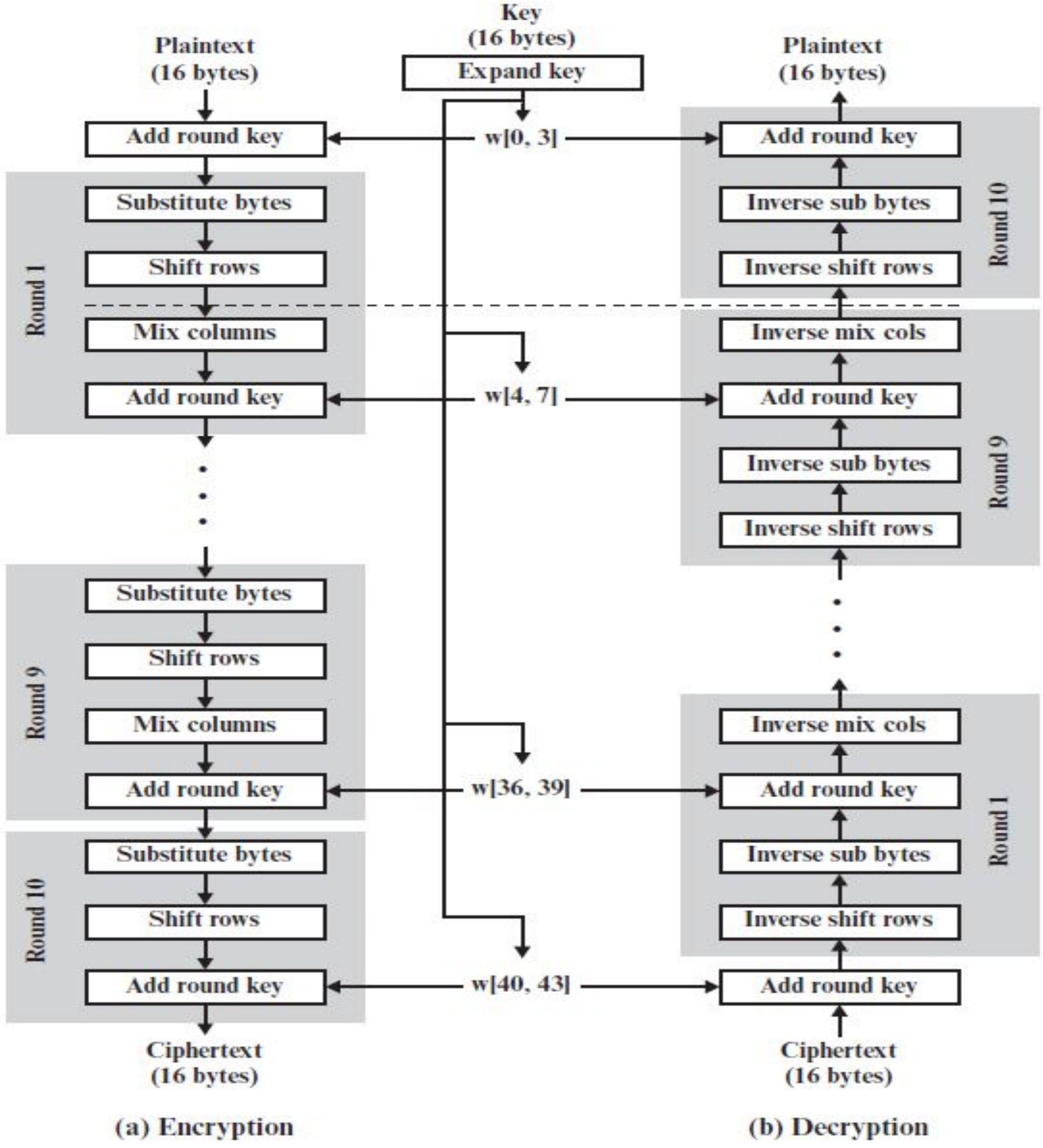
01 Nov 2024 (C5)

--07 Sep 24 (C1)

Topics

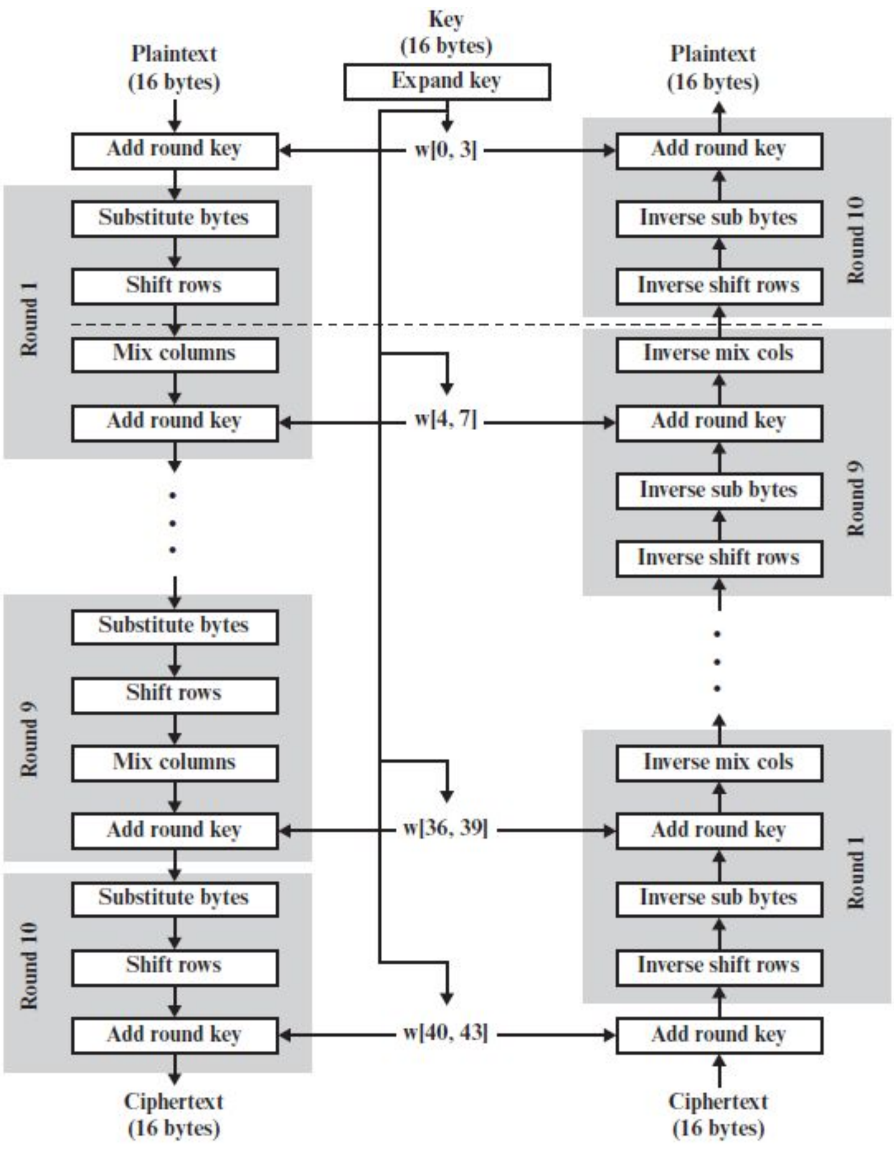
- Finite Field Arithmetic
- AES Structure
  - AES transformation structure
  - Fix row transformation
  - Mixed row transformation
  - Addround transformation
- AES Key Expansion
- Avalanche effect
- Relationship between Rijndael and AES

AES Encryption and Decryption



AES Encryption and Decryption

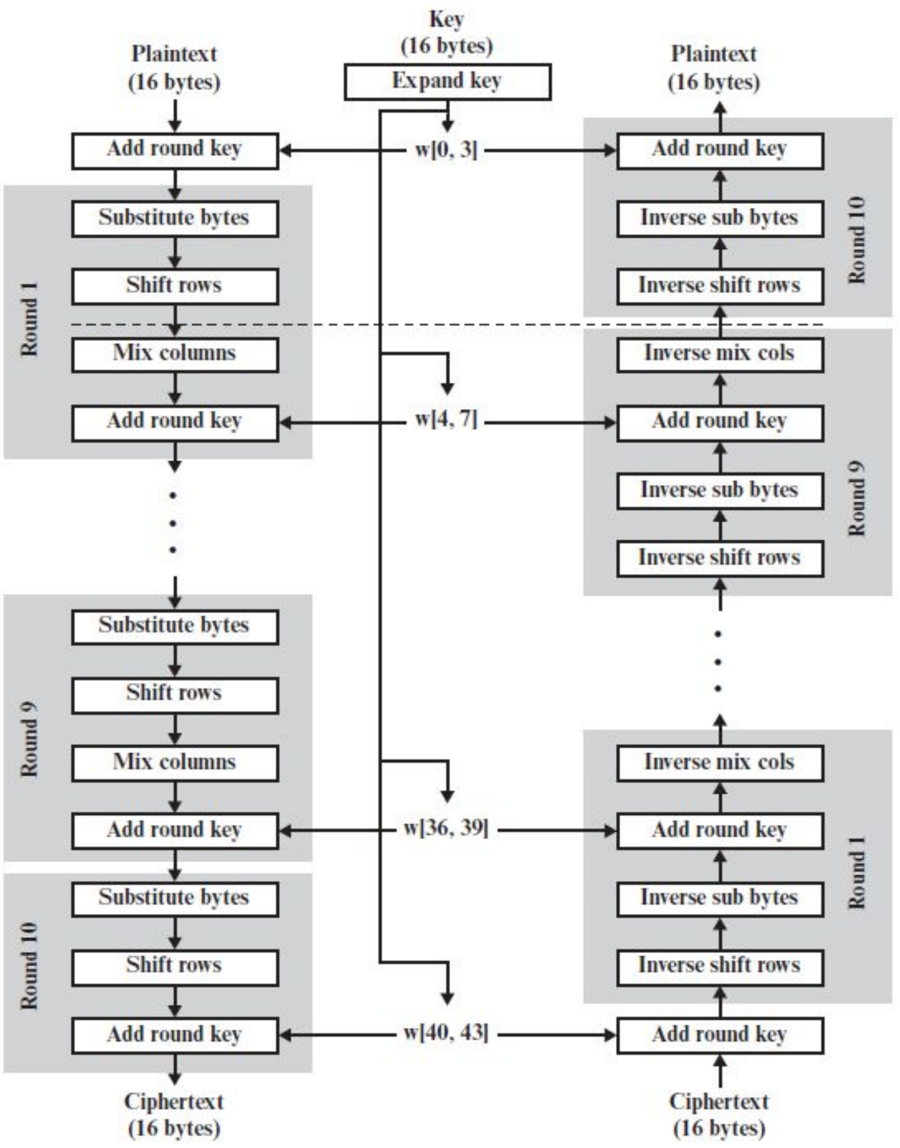
- Each stage is easily reversible. For the Substitute Byte, ShiftRows, and MixColumns stages, an **inverse function is used** in the decryption algorithm.
- For the AddRoundKey stage, the **inverse** is achieved by XORing the **same round key to the block, using the result that A ⊕B ⊕B = A**
- As with most block ciphers, the decryption algorithm makes use of the expanded key in **reverse order.**
- However, the **decryption algorithm is not identical to the encryption algorithm**. This is a consequence of the particular structure of AES.



(a) Encryption (b) Decryption

AES Encryption and Decryption

- Once it is established that all **four stages are reversible,** it is easy to **verify that decryption does recover the plaintext**
- The **final round** of both encryption and decryption consists of **only three stages**.
- this is a consequence of the particular structure of AES and is required to make the cipher reversible.

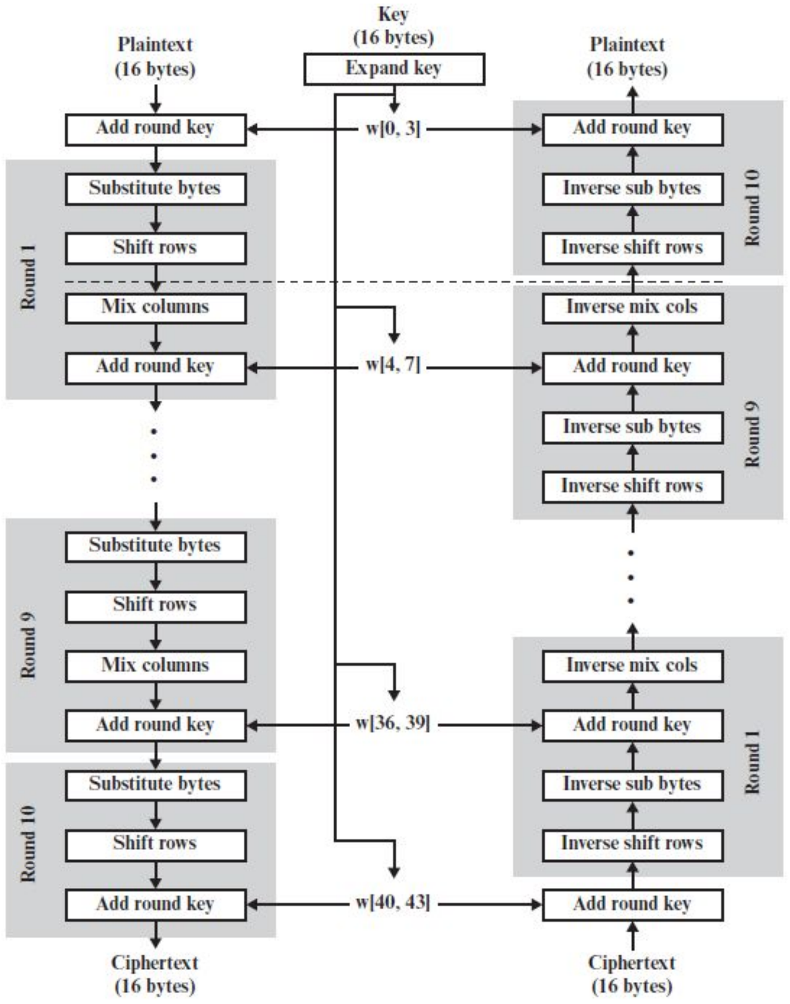


(a) Encryption

(b) Decryption

AES Encryption and Decryption

- AES decryption cipher is **not identical to the encryption** cipher. That is, the sequence of transformations for decryption differs from that for encryption.
- the form of the key schedules for encryption and decryption is the same.
- This has the disadvantage that **two separate software or firmware modules are needed for applications that require both encryption and decryption**.



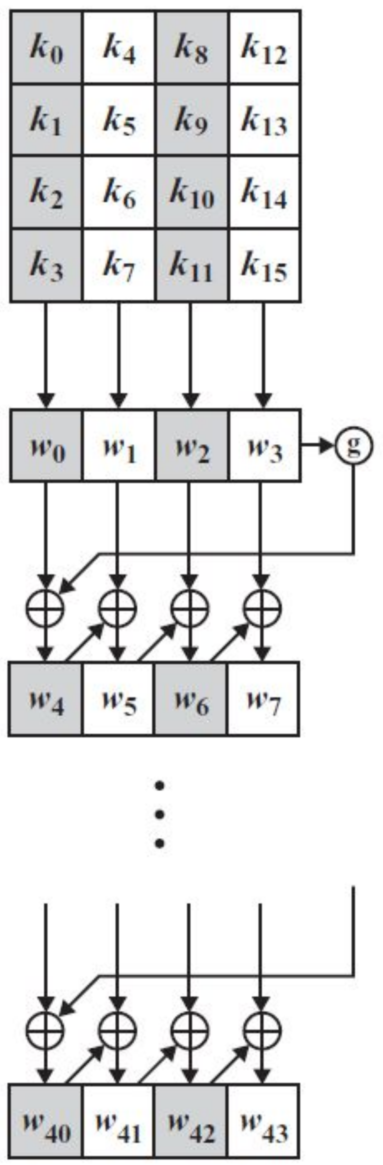
(a) Encryption
(b) Decryption

AES Analysis

- **AES Analysis**
- In present day cryptography, AES is widely adopted and supported in both hardware and software.
- Till date, **no practical cryptanalytic attacks** against AES has been discovered.
- Additionally, **AES has built-in flexibility of key length**, which allows a degree of ‘**future-proofing’** against progress in the ability to perform exhaustive key searches.
- However, just as for DES, the AES security is assured only if it is **correctly implemented**, and good **key management** is employed.

Key Expansion Algorithm

- The **AES key expansion algorithm** takes as input a **four-word (16-byte) key** and produces a **linear array of 44 words** (176 bytes).
- This is **sufficient to provide a four-word round key** for the **initial AddRoundKey stage** and **each of the 10 rounds** of the cipher.
- The **key is copied** into the **first four words of the expanded key**.
- The **remainder of the expanded key** is filled in **four words at a time**.
- Each added word w[i] depends on the **immediately preceding word**, w[i - 1], and the word four positions back, w[i - 4]. In three out of four cases, a simple XOR is used.
- For a word whose position in the w array is a multiple of 4, a more complex function is used.



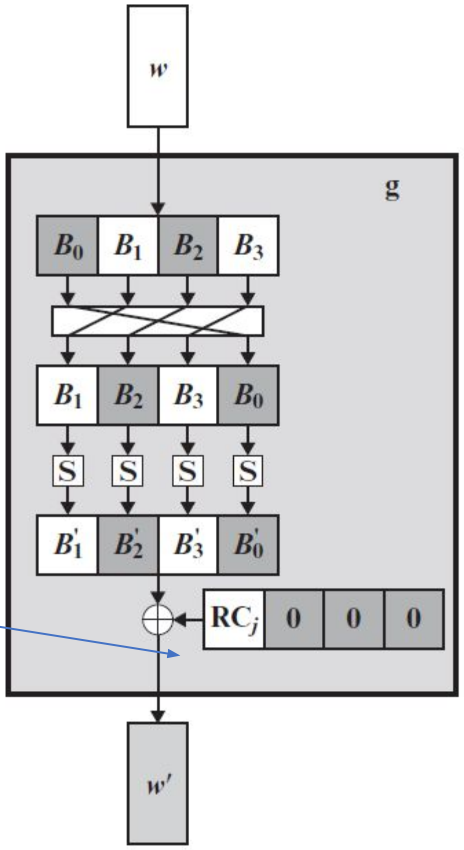
Key Expansion : How algorithm works?

- **Initial Key** – The algorithm makes use of an **initial key**. Depending on the level of security needed, this key's length can range from 128 to 256 bits.
- **Round Constants** – The approach makes use of a **set of round constants**, which are **predefined values** used in the **key expansion process**.
- **Word Size** – The key divides **words into individual blocks**. A **word typically has 32 bits** in it. For example, four **32-bit words are created from a 128-bit key**.
- **Key Schedule** – The term "key schedule" refers to a **set of round keys generated using the key expansion process**. The initial round key and the extra round keys that were derived from it are both included in this schedule.

expansion of the 16-byte key into 10 round keys. As previously explained, this process is performed word by word, with each four-byte word occupying one column of the word round-key matrix.

Key Expansion : How algorithm wor

- **Expansion Rounds** – The algorithm performs several tasks in each expansion round, such as –
- **Complex function**. The **function g** consists of the following subfunctions.
  - **RotWord** – This function **rotates the bytes** in a word.
  - **SubWord** – Applies a **substitution operation** using a **predetermined S-box**.
  - **Rcon** – XORs the word using a **round constant**.
- **Round Keys** – The order of round keys that are still in place following all expansion rounds defines the key schedule. Each round key is used in the corresponding round of AES encryption or decryption.



(b) Function g

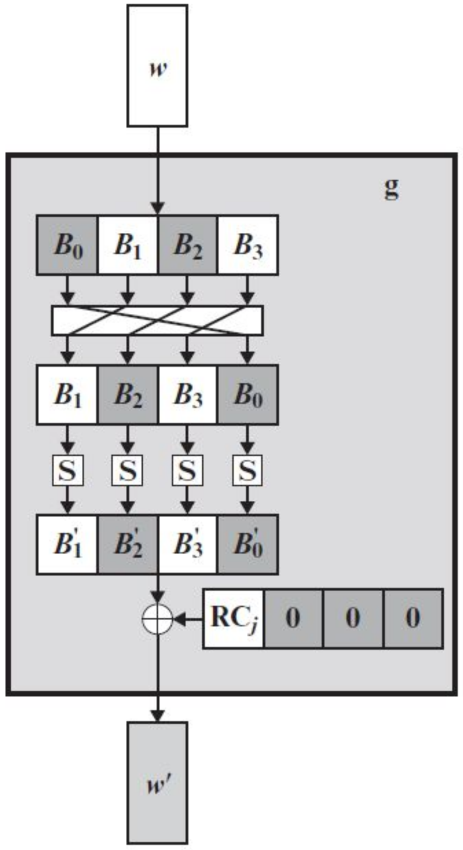
Overall, the key expansion method **increases security and prevents cryptographic** attacks by ensuring that each AES encryption and decryption round has a **unique round key**.

Key Expansion :How algorithm works

Complex function. The **function g** consists of the following subfunctions.

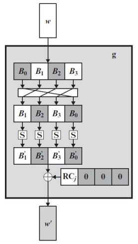
The result of steps 1 and 2 is XORed with a round constant, **Rcon[j].**

- The **round constant** is a word in which the three rightmost bytes are always 0.
- Thus, the effect of an XOR of a word with Rcon is to only perform an **XOR on the leftmost byte** of the word.
- The round constant is different for each round and is defined as Rcon[j] = (RC[j], 0, 0, 0), with
- RC[1] = 1, RC[j] = 2 . RC[j - 1]
- and with **multiplication defined over the field GF(28).**



(b) Function g

How algorithm works?

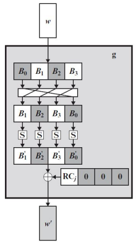


(b) Function g

- **Rationale**
- The Rijndael developers **designed the expansion** key algorithm to be **resistant to known cryptanalytic attacks**.
- The inclusion of a **round-dependent round constant** eliminates **the symmetry, or similarity, between the ways in which round keys are generated in different rounds**.

- specific criteria …

How algorithm works?



(b) Function g

The **specific criteria** that were used are [DAEM99]

- Knowledge of a part of the **cipher key or round key** does **not enable calculation of many other round-key bits.**
- An **invertible transformation** [i.e., knowledge of any Nk consecutive words of the expanded key enables regeneration of the entire expanded key (Nk = key size in words)].
- **Speed** on a wide range of processors.
- Usage of **round constants** to **eliminate symmetries**.
- Diffusion of cipher key differences into the round keys; that is, each key bit affects many round key bits.
- **Enough nonlinearity** to prohibit the full determination of round key differences from cipher key differences only.
- Simplicity of description.

AES Example

- Table shows the progression of State through the AES encryption process.
- The **first column shows** the value of State at the start of a round. For the first row, State is just the matrix arrangement of the plaintext.
- The **second, third, and fourth columns** show the value of State for that round after the SubBytes, ShiftRows, and MixColumns transformations, respectively.
- The **fifth column** shows the round key.
- ??
- The **first column** shows the value of State resulting from the bitwise XOR of State after
- the preceding MixColumns with the round key for the preceding round.

AES Example

Start of Round	After SubBytes	After ShiftRows	After MixColumns	Round Key
01 89 fe 76 23 ab dc 54 45 cd ba 32 67 ef 98 10				0f 47 0c af 15 d9 b7 7f 71 e8 ad 67 c9 59 d6 98
0e ce f2 d9 36 72 6b 2b 34 25 17 55 ae b6 4e 88	ab 8b 89 35 05 40 7f f1 18 3f f0 fc e4 4e 2f c4	ab 8b 89 35 40 7f f1 05 f0 fc 18 3f c4 e4 4e 2f	b9 94 57 75 e4 8e 16 51 47 20 9a 3f c5 d6 f5 3b	dc 9b 97 38 90 49 fe 81 37 df 72 15 b0 e9 3f a7
65 0f c0 4d 74 c7 e8 d0 70 ff e8 2a 75 3f ca 9c	4d 76 ba e3 92 c6 9b 70 51 16 9b e5 9d 75 74 de	4d 76 ba e3 c6 9b 70 92 9b e5 51 16 de 9d 75 74	8e 22 db 12 b2 f2 dc 92 df 80 f7 c1 2d c5 1e 52	d2 49 de e6 c9 80 7e ff 6b b4 c6 d3 b7 5e 61 c6
5c 6b 05 f4 7b 72 a2 6d b4 34 31 12 9a 9b 7f 94	4a 7f 6b bf 21 40 3a 3c 8d 18 c7 c9 b8 14 d2 22	4a 7f 6b bf 40 3a 3c 21 c7 c9 8d 18 22 b8 14 d2	b1 c1 0b cc ba f3 8b 07 f9 1f 6a c3 1d 19 24 5c	c0 89 57 b1 af 2f 51 ae df 6b ad 7e 39 67 06 c0
71 48 5c 7d 15 dc da a9 26 74 c7 bd 24 7e 22 9c	a3 52 4a ff 59 86 57 d3 f7 92 c6 7a 36 f3 93 de	a3 52 4a ff 86 57 d3 59 c6 7a f7 92 de 36 f3 93	d4 11 fe 0f 3b 44 06 73 cb ab 62 37 19 b7 07 ec	2c a5 f2 43 5c 73 22 8c 65 0e a3 dd f1 96 90 50
f8 b4 0c 4c 67 37 24 ff ae a5 c1 ea e8 21 97 bc	41 8d fe 29 85 9a 36 16 e4 06 78 87 9b fd 88 65	41 8d fe 29 9a 36 16 85 78 87 e4 06 65 9b fd 88	2a 47 c4 48 83 e8 18 ba 84 18 27 23 eb 10 0a f3	58 fd 0f 4c 9d ee cc 40 36 38 9b 46 eb 7d ed bd
72 ba cb 04 1e 06 d4 fa b2 20 bc 65 00 6d e7 4e	40 f4 1f f2 72 6f 48 2d 37 b7 65 4d 63 3c 94 2f	40 f4 1f f2 6f 48 2d 72 65 4d 37 b7 2f 63 3c 94	7b 05 42 4a 1e d0 20 40 94 83 18 52 94 c4 43 fb	71 8c 83 cf c7 29 e5 a5 4c 74 ef a9 c2 bf 52 ef
0a 89 c1 85 d9 f9 c5 e5 d8 f7 f7 fb 56 7b 11 14	67 a7 78 97 35 99 a6 d9 61 68 68 0f b1 21 82 fa	67 a7 78 97 99 a6 d9 35 68 0f 61 68 fa b1 21 82	ec 1a c0 80 0c 50 53 c7 3b d7 00 ef b7 22 72 e0	37 bb 38 f7 14 3d d8 7d 93 e7 08 a1 48 f7 a5 4a
db a1 f8 77 18 6d 8b ba a8 30 08 4e ff d5 d7 aa	b9 32 41 f5 ad 3c 3d f4 c2 04 30 2f 16 03 0e ac	b9 32 41 f5 3c 3d f4 ad 30 2f c2 04 ac 16 03 0e	b1 1a 44 17 3d 2f ec b6 0a 6b 2f 42 9f 68 f3 b1	48 f3 cb 3c 26 1b c3 be 45 a2 aa 0b 20 d7 72 38
f9 e9 8f 2b 1b 34 2f 08 4f c9 85 49 bf bf 81 89	99 1e 73 f1 af 18 15 30 84 dd 97 3b 08 08 0c a7	99 1e 73 f1 18 15 30 af 97 3b 84 dd a7 08 08 0c	31 30 3a c2 ac 71 8c c4 46 65 48 eb 6a 1c 31 62	fd 0e c5 f9 0d 16 d5 6b 42 e0 4a 41 cb 1c 6e 56
cc 3e ff 3b a1 67 59 af 04 85 02 aa a1 00 5f 34	4b b2 16 e2 32 85 cb 79 f2 97 77 ac 32 63 cf 18	4b b2 16 e2 85 cb 79 32 77 ac f2 97 18 32 63 cf		b4 ba 7f 86 8e 98 4d 26 f3 13 59 18 52 4e 20 76
ff 08 69 64 0b 53 34 14 84 bf ab 8f 4a 7c 43 b9				

Mathematical Representation

- Let us denote –

- K is the first key, having N bits in its length.
- The word count (Nk) of the key is its total word count (e.g., 4 for a 128-bit key, 6 for a 192-bit key, and 8 for a 256-bit key).
- Nr, the AES round identifier, is 10 for AES-128, 12 for AES-192, and 14 for AES-256.

Key Expansion Process

- **Step 1**: Make the **round keys first**. Establish the **first word K** in a **word array W**.
- Iterate to generate (Nr + 1) round keys –
- **Step 2**: Finalize Round Keys: After the loop, the **array W has all round keys**.

Notations

- **RotWord(w) –** Rotate the word w's bytes in a manner that is cyclic.
- **SubWord(w**) – Use the AES S-box to swap out each byte in the word w.
- **Rcon(i) –** Produce the current round's i 's round constant, or Rcon.

Example of AES Key Expansion

- Now we are going to discuss the AES (Advanced Encryption Standard) key expansion algorithm with an example. For this example, we will be using AES-128, so our initial key will be 128 bits (16 bytes).
- The first key can be considered as a set of bytes −
- 2b 7e 15 16 28 ae d2 a6 ab f7 97 66 76 15 13 1

- This key is 128 bits long.
- Let's look at every step of the key expansion process −
- Initial Key

  2b 7e 15 16

  28 ae d2 a6

  ab f7 97 66

  76 15 13 1

Example of AES Key Expansion

- Expansion Rounds
- We start by adding the first key to our list of round keys. Following that, we iteratively generate more round keys until we reach a total of 11.
- Round 1
- In the RotWord, SubWord, and XOR operations, we use the round constant –

  RotWord: 7e 15 16 2b

  SubWord: 63 cb e7 8c

  Rcon: 01 00 00 00

  Round 1 Key: 63 cb e7 8c 09 cf 4f 3c 3b a9 82 fb 11 13 d8 2c

Round 2

The word from the previous round key is XORed four positions back –

Round 2 Key: a0 fa fe 17 88 54 2c b1 23 a3 39 39 2a 6c 76 05

Example of AES Key Expansion

- Subsequent Rounds
- Round keys are generated in this way until a total of 11 keys have been generated.
- Final Round Keys

  Round 0 (Initial Key): 2b 7e 15 16 28 ae d2 a6 ab f7 97 66 76 15 13 1

  Round 1: 63 cb e7 8c 09 cf 4f 3c 3b a9 82 fb 11 13 d8 2c

  Round 2: a0 fa fe 17 88 54 2c b1 23 a3 39 39 2a 6c 76 05

  ...

  Round 10: 3d 47 0e 52 77 37 2e 10 1f 7e 0e 20 6a 51 7f a7

In this example, an original key was expanded into a number of round keys using the AES key expansion technique. Every round key is created from the one before it using round constants and operations like XOR, RotWord, and SubWord. These round keys are then used in each round of AES encryption to increase security and prevent cryptographic attacks.

Key Expansion in AES

- **Key expansion**, also known as the **AES key schedule**, is a critical process in AES (Advanced Encryption Standard) that **generates multiple round keys** from the original encryption key.
- These round keys are used in each round of AES encryption and decryption.
- Overview of Key Expansion:
- AES uses keys of different lengths depending on the variant:
  - AES-128 uses a 128-bit key (16 bytes).
  - AES-192 uses a 192-bit key (24 bytes).
  - AES-256 uses a 256-bit key (32 bytes).
- The key expansion algorithm takes the initial key and derives a series of round keys from it. Each round of AES (which is either 10, 12, or 14 rounds depending on the key size) uses a unique round key.

Steps in Key Expansion:

1. Initial Key:

- The **original key** is the starting point of the key schedule.
- It's divided into **4-byte words** (W[0], W[1], ..., W[N-1]), where N depends on the key size.
- For AES-128, the original key is split into 4 words (W[0] to W[3]).

2. Word Generation:

- The key expansion generates a total of **44 words** (for AES-128), **52 words** (for AES-192), or **60 words** (for AES-256), depending on the key length.
- **Each round** uses **4 words (16 bytes)** as the **round key**, which is why the n**umber of words is directly linked to the number of rounds** in AES.

3. **RotWord**:

- **Every four words**, a **transformation** called **RotWord is applied** to the **last word**. This operation takes a word and **performs a cyclic left shift** on its bytes. For example:
  - Input: W[3] = {0x1a, 0x2b, 0x3c, 0x4d}
  - RotWord result: {0x2b, 0x3c, 0x4d, 0x1a}

Steps in Key Expansion:

4. SubWord:

- After RotWord, **each byte of the word undergoes** a **substitution** using the **AES S-Box**, a **non-linear substitution box** used to provide **confusion**.
- Each byte is replaced with its corresponding value from the S-Box table.

5. Rcon (Round Constant):

- A **round constant (Rcon**) is **XORed with the first byte** of the word after **SubWord.**
- The **round constant is a pre-defined** set of values that **depend on the round number**.
- For example, for the **first round, Rcon might be 0x01**, and it **increases exponentially** for each subsequent round.

6. XOR Operations:

- The result of the **SubWord** and **Rcon** operations **is XORed** with the word **from four positions back**. This is the **key expansion's recursive part**, ensuring that **each round key is dependent on the previous one.**

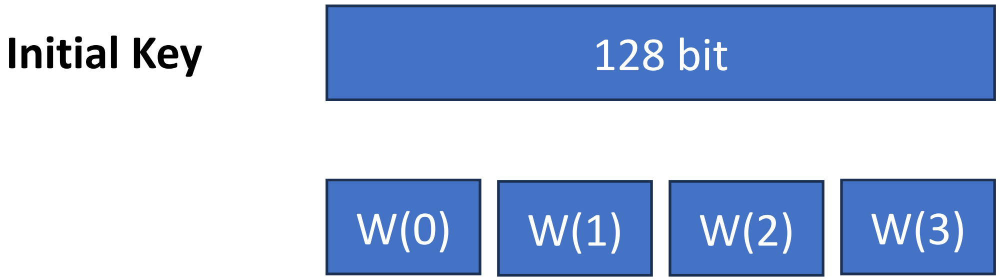
Steps in Key Expansion:

7. Final Round Keys:

- The **process continues until enough** round keys are generated for the encryption process.
- For AES-128, this means **generating 44 words**, which correspond to **10 rounds + 1 initial round key (because the round key is added at the start).**

Different explanation

Steps in Key Expansion:



Key Expansion for AES-128 (Example)

- Assume the original key for AES-128 is:

Key = [2b7e151628aed2a6abf7158809cf4f3c]

- This **16-byte (128-bit)** key will be **expanded into 44 words (176 bytes**).
- Here's an outline of the process:
- Initial words **(W[0] to W[3]):** The first four words are simply **taken from the key itself**.

- W[0] = 2b7e1516
- W[1] = 28aed2a6
- W[2] = abf71588
- W[3] = 09cf4f3c

Different explanation

Steps in Key Expansion:

Key Expansion for AES-128 (Example)

- Next words (W[4] to W[43]):
- These words are generated **using the process** of **RotWord, SubWord, Rcon, and XORing** with the **previous words**.
- Here's a simple breakdown:

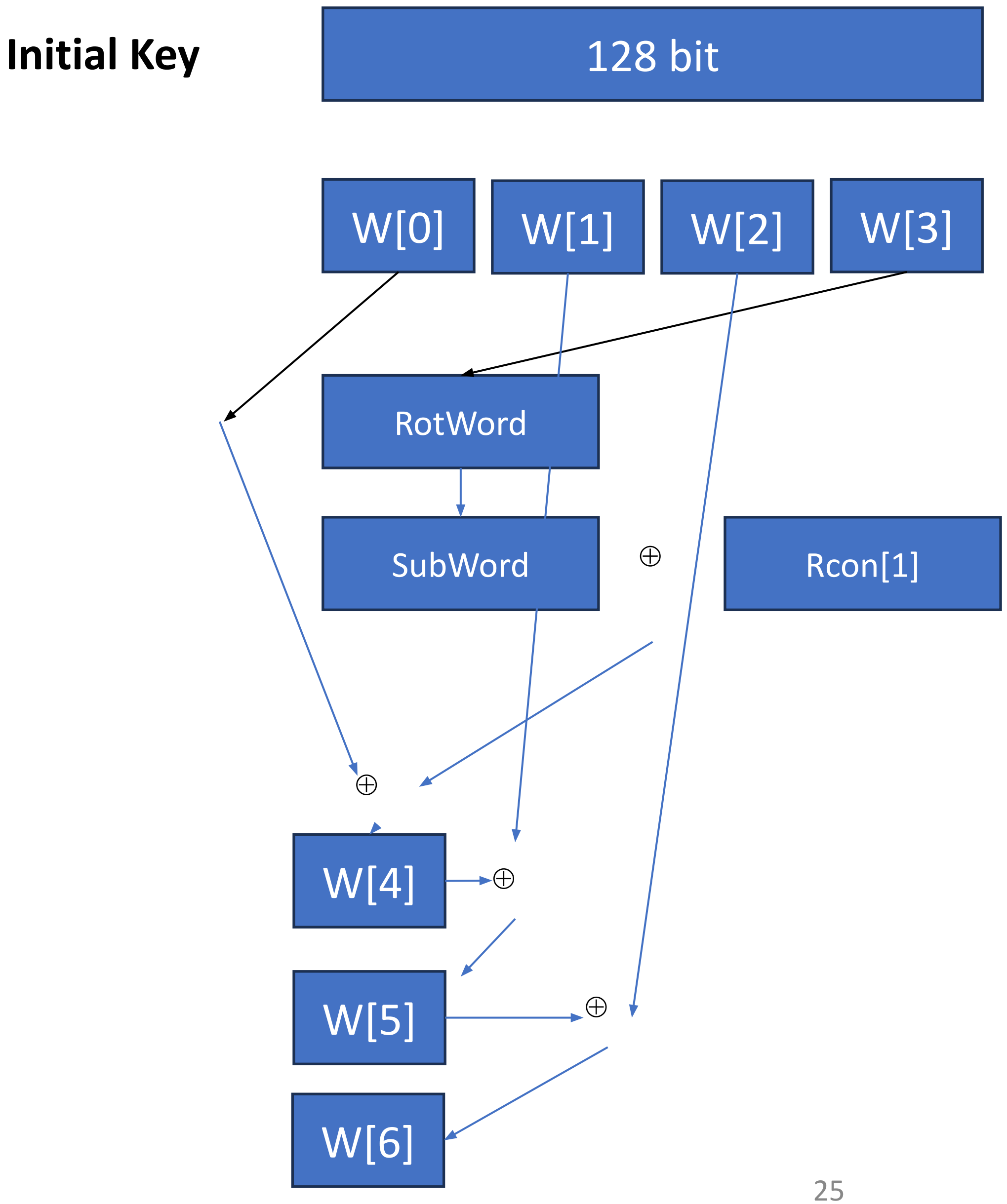
  W[4] = W[0] ⊕ SubWord(RotWord(W[3])) ⊕ Rcon[1]

  W[5] = W[1] ⊕ W[4]

  W[6] = W[2] ⊕ W[5]

  W[7] = W[3] ⊕ W[6]

- This process continues until 44 words are generated for AES-128, providing one round key for each of the 10 rounds and the initial round key.



Avalanche Effect

- If a small change in the key or plaintext were to produce a corresponding small change in the ciphertext, this might be used to **effectively reduce the size of the plaintext (or key) space to be searched.**
- What is desired is the avalanche effect, in which a **small change in plaintext or key produces a large change in the ciphertext.**

Table 6.5 Avalanche Effect in AES: Change in Plaintext

Round		Number of Bits that Differ
	0123456789abcdeffedcba9876543210 0023456789abcdeffedcba9876543210	1
0	0e3634aece7225b6f26b174ed92b5588 0f3634aece7225b6f26b174ed92b5588	1
1	657470750fc7ff3fc0e8e8ca4dd02a9c c4a9ad090fc7ff3fc0e8e8ca4dd02a9c	20
2	5c7bb49a6b72349b05a2317ff46d1294 fe2ae569f7ee8bb8c1f5a2bb37ef53d5	58
3	7115262448dc747e5cdac7227da9bd9c ec093dfb7c45343d689017507d485e62	59
4	f867aee8b437a5210c24c1974cffeabc 43efdb697244df808e8d9364ee0ae6f5	61
5	721eb200ba06206dcbd4bce704fa654e 7b28a5d5ed643287e006c099bb375302	68
6	0ad9d85689f9f77bc1c5f71185e5fb14 3bc2d8b6798d8ac4fe36a1d891ac181a	64
7	db18a8ffa16d30d5f88b08d777ba4eaa 9fb8b5452023c70280e5c4bb9e555a4b	67
8	f91b4fbfe934c9bf8f2f85812b084989 20264e1126b219aef7feb3f9b2d6de40	65
9	cca104a13e678500ff59025f3bafaa34 b56a0341b2290ba7dfdfbddcd8578205	61
10	ff0b844a0853bf7c6934ab4364148fb9 612b89398d0600cde116227ce72433f0	58

Avalanche Effect

- **Change in Plaintext.**
- Table shows the result when the **eighth bit of the plaintext is changed.**
- The **second column** of the table shows the value of the State matrix at the **end of each round** for the two plaintexts.
- **just one round, 20 bits** of the State vector differ.
- After two rounds, **close to half** the bits differ.
- This magnitude of difference **propagates through the remaining rounds**.
- A bit difference in **approximately half the positions** in the most **desirable outcome**.

Table 6.5 Avalanche Effect in AES: Change in Plaintext

Round		Number of Bits that Differ
	0123456789abcdeffedcba9876543210 0023456789abcdeffedcba9876543210	1
0	0e3634aece7225b6f26b174ed92b5588 0f3634aece7225b6f26b174ed92b5588	1
1	657470750fc7ff3fc0e8e8ca4dd02a9c c4a9ad090fc7ff3fc0e8e8ca4dd02a9c	20
2	5c7bb49a6b72349b05a2317ff46d1294 fe2ae569f7ee8bb8c1f5a2bb37ef53d5	58
3	7115262448dc747e5cdac7227da9bd9c ec093dfb7c45343d689017507d485e62	59
4	f867aee8b437a5210c24c1974cffeabc 43efdb697244df808e8d9364ee0ae6f5	61
5	721eb200ba06206dcbd4bce704fa654e 7b28a5d5ed643287e006c099bb375302	68
6	0ad9d85689f9f77bc1c5f71185e5fb14 3bc2d8b6798d8ac4fe36a1d891ac181a	64
7	db18a8ffa16d30d5f88b08d777ba4eaa 9fb8b5452023c70280e5c4bb9e555a4b	67
8	f91b4fbfe934c9bf8f2f85812b084989 20264e1126b219aef7feb3f9b2d6de40	65
9	cca104a13e678500ff59025f3bafaa34 b56a0341b2290ba7dfdfbddcd8578205	61
10	ff0b844a0853bf7c6934ab4364148fb9 612b89398d0600cde116227ce72433f0	58

Avalanche Effect

- **Change in Key**
- Table shows the change in State matrix values when the same plaintext is used and the two keys **differ in the eighth bit**.
- . Again, one round produces a **significant change, and the magnitude of change** after all subsequent rounds is roughly half the bits.
- Thus, based on this example, AES exhibits a very **strong avalanche effect.**

Table 6.6 Avalanche Effect in AES: Change in Key

Round		Number of Bits that Differ
	0123456789abcdeffedcba9876543210 0123456789abcdeffedcba9876543210	0
0	0e3634aece7225b6f26b174ed92b5588 0f3634aece7225b6f26b174ed92b5588	1
1	657470750fc7ff3fc0e8e8ca4dd02a9c c5a9ad090ec7ff3fc1e8e8ca4cd02a9c	22
2	5c7bb49a6b72349b05a2317ff46d1294 90905fa9563356d15f3760f3b8259985	58
3	7115262448dc747e5cdac7227da9bd9c 18aeb7aa794b3b66629448d575c7cebf	67
4	f867aee8b437a5210c24c1974cffeabc f81015f993c978a876ae017cb49e7eec	63
5	721eb200ba06206dcbd4bce704fa654e 5955c91b4e769f3cb4a94768e98d5267	81
6	0ad9d85689f9f77bc1c5f71185e5fb14 dc60a24d137662181e45b8d3726b2920	70
7	db18a8ffa16d30d5f88b08d777ba4eaa fe8343b8f88bef66cab7e977d005a03c	74
8	f91b4fbfe934c9bf8f2f85812b084989 da7dad581d1725c5b72fa0f9d9d1366a	67
9	cca104a13e678500ff59025f3bafaa34 0ccb4c66bbfd912f4b511d72996345e0	59
10	ff0b844a0853bf7c6934ab4364148fb9 fc8923ee501a7d207ab670686839996b	53

Relationship between Rijndael and AES

- The **relationship between Rijndael and AES** is that AES (Advanced Encryption Standard) is a **standardized version** of the Rijndael cipher. In 2001, the **National Institute of Standards and Technology (NIST)** selected the Rijndael algorithm, designed by cryptographers **Vincent Rijmen** and **Joan Daemen**, as the algorithm to become the new Advanced Encryption Standard (AES) for secure encryption. Here's how they are connected and what differentiates them:
- **AES as a Subset of Rijndael**:
- While **Rijndael supports** a variety of block sizes (128, 192, and 256 bits) and key lengths (in multiples of 32 bits), **AES is a standardized subset** with a fixed block size of **128 bits**.
- AES supports three key sizes: **128 bits, 192 bits, and 256 bits**, corresponding to 10, 12, and 14 rounds, respectively.
- This **fixed block size (128 bits) and limited key sizes (128, 192, 256 bits)** were chosen by NIST to make the encryption standard simpler, more uniform, and compatible across various platforms.

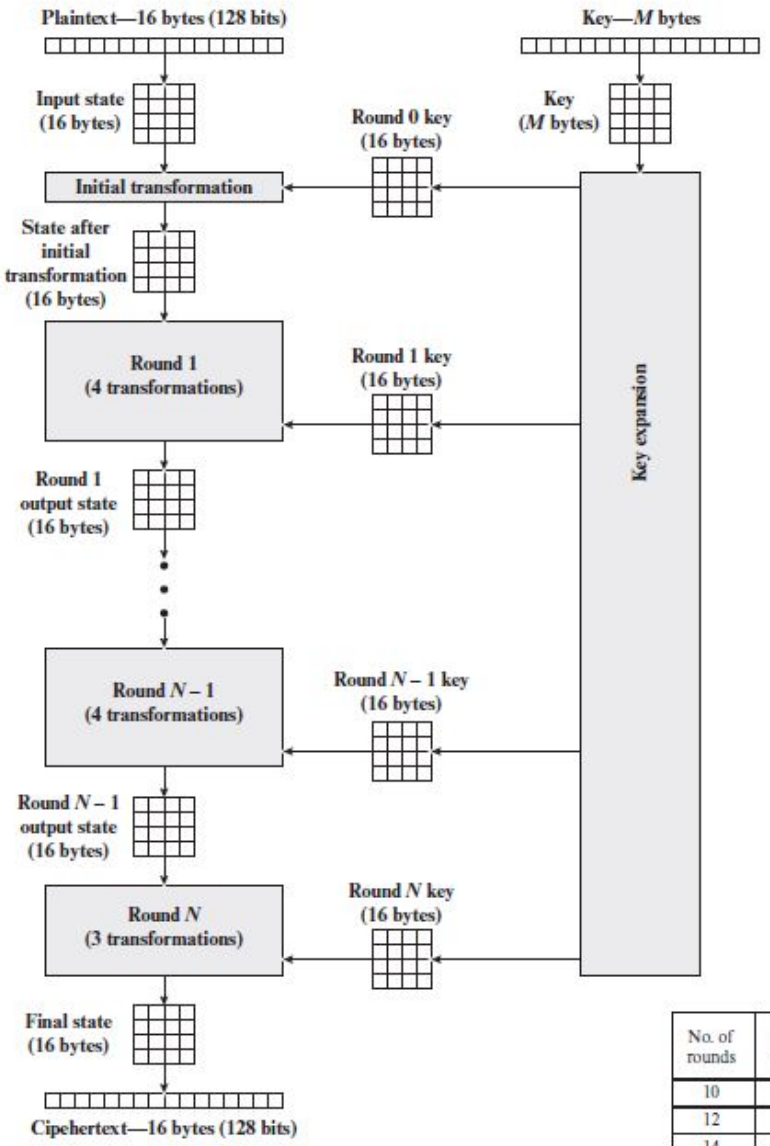
Relationship between Rijndael and AES

- **Structural Similarity**:
- The core structure and steps of AES and Rijndael are the same: both use a combination of **SubBytes, ShiftRows, MixColumns,** and **AddRoundKey** operations in a **Substitution-Permutation Network (SPN)**.
- The only difference is that Rijndael's design allows for more flexibility in block and key sizes, while AES restricts these to a specific standard.

- **Purpose and Implementation**:
- AES, as a standardized form of Rijndael, was widely adopted in government, industry, and technology for secure data encryption.
- Rijndael can still theoretically be implemented with different block sizes (e.g., 192 or 256 bits), but **AES is the globally recognized and widely used implementation** of Rijndael.

Thank You

Encryption Process



No. of rounds	Key Length (bytes)
10	16
12	24
14	32

Encryption Process

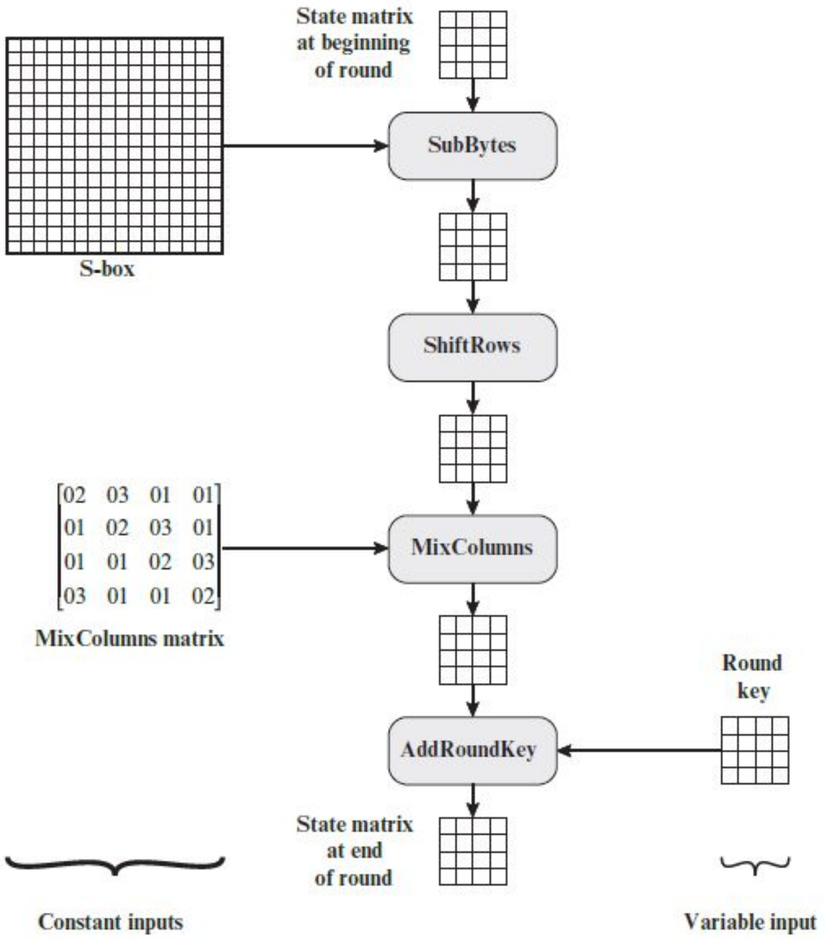


Figure 6.8 Inputs for Single AES Round