Advanced Encryption Standard (AES)

Origins

- A clear replacement for DES was needed
  - Since the Key size was too small
  - The variants are just patches
- It can use Triple-DES – but slow, has small blocks
- US NIST issued call for ciphers in 1997
- 15 candidates accepted in Jun 98
- 5 were shortlisted in Aug-99

AES Competition Requirements

- private key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger & faster than Triple-DES
- provide full specification & design details
- both C & Java implementations
- NIST have released all submissions & unclassified analyses

AES Evaluation Criteria

- Initial criteria:
  - security – effort for practical cryptanalysis
  - cost – in terms of computational efficiency
  - algorithm & implementation characteristics
- Final criteria
  - general security
  - ease of software & hardware implementation
  - implementation attacks
  - flexibility (in en/decrypt, keying, other factors)

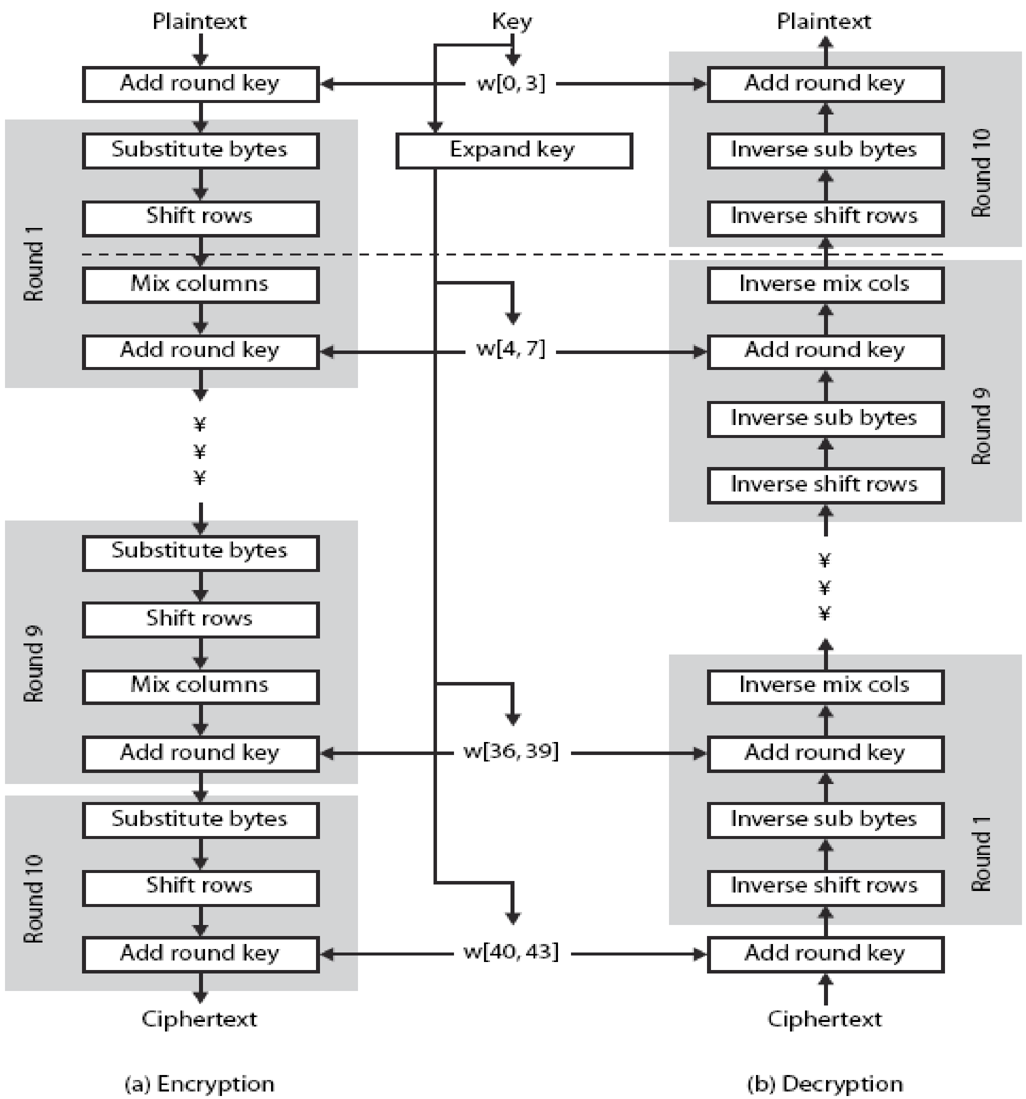
The AES Cipher - Rijndael

- Rijndael was selected as the AES in Oct-2000. Designed by Joan Rijmen and VincentDaemen in Belgium
- It has 128/192/256 bit keys, 128 bit data
- It is an **iterative** rather than **Feistel** cipher
  - processes data as block of 4 columns of 4 bytes
  - operates on entire data block in every round
- Designed to be:
  - resistant against known attacks
  - speed and code compactness on many CPUs
  - design simplicity

Rijndael

- Data block viewed as 4-by-4 table of bytes
- Such a table is called the **current state**
- key is expanded to array of words
- It has 10 rounds in which state the following transformations (called `layers'):
  - BS- byte substitution (1 S-box used on every byte)
  - SR- shift rows (permute bytes between groups/columns)
  - MC- mix columns (uses matrix multiplication in GF(256))
  - ARK- add round key (XOR state with round key)
- First and last round are a little different

Rijndael

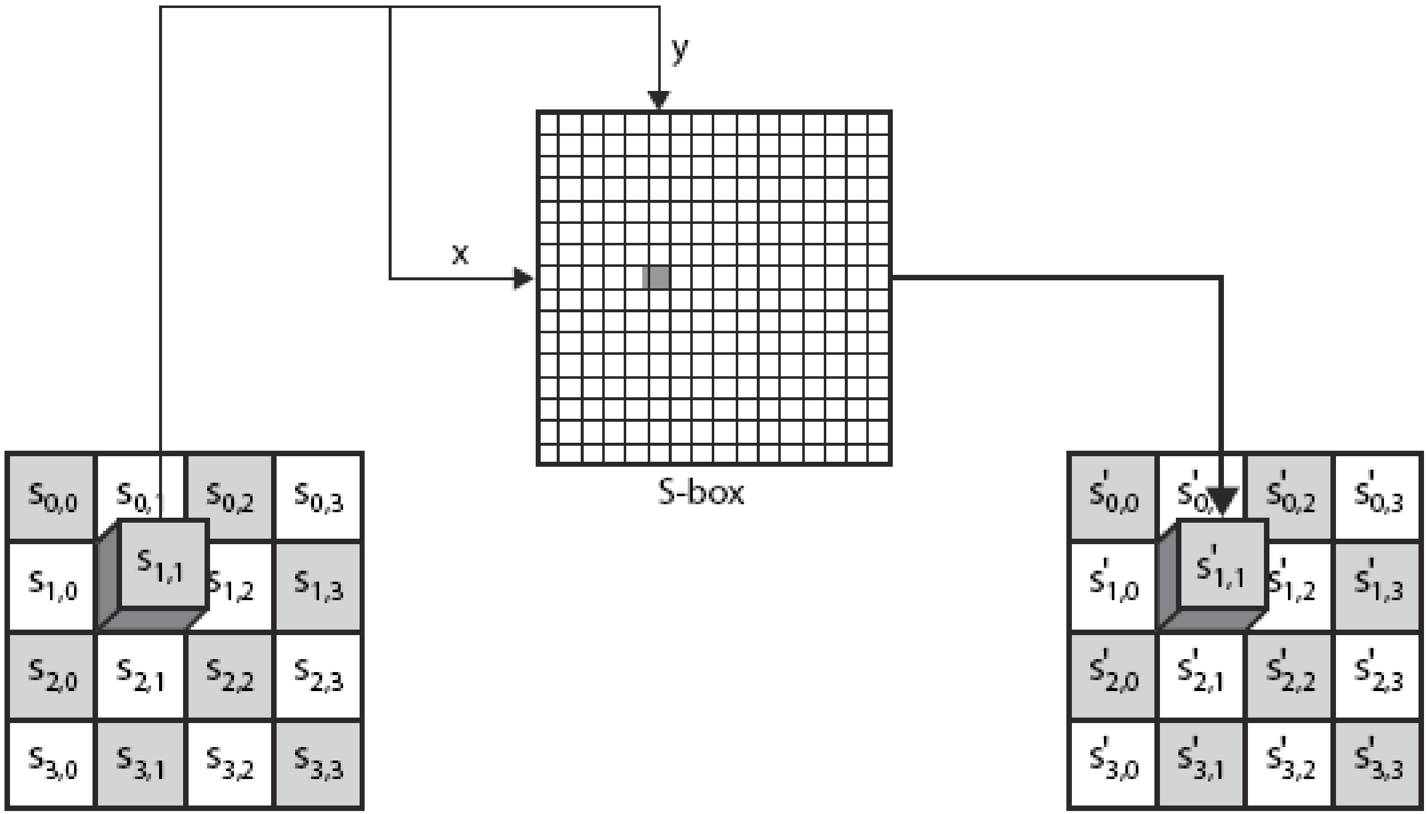


Plaintext
Key
Plaintext
Add round key
w[0, 3]
Add round key
Substitute bytes
Expand key
Inverse sub bytes
Shift rows
Inverse shift rows
Round 1
Round 10
Mix columns
Inverse mix cols
Add round key
w[4, 7]
Add round key
Inverse sub bytes
Round 9
Inverse shift rows
¥
¥
¥
¥
¥
¥
Substitute bytes
Shift rows
Round 9
Mix columns
Inverse mix cols
Add round key
w[36, 39]
Add round key
Inverse sub bytes
Substitute bytes
Round 1
Shift rows
Inverse shift rows
Round 10
Add round key
w[40, 43]
Add round key
Ciphertext
Ciphertext
(a) Encryption
(b) Decryption

Byte Substitution

- A simple substitution of each byte
- Uses one s-box of 16x16 bytes containing a permutation of all 256 8-bit values
- Each byte of state is replaced by byte indexed by row (left 4-bits) & column (right 4-bits)
  - Eg. Byte {95} is replaced by byte in row 9 column 5 which has value {2A}
- S-box constructed using defined transformation of values in GF(256)
- S-box constructed using a simple math formula using a non-linear function : 1/x.
- Construction of S-Box (on board)

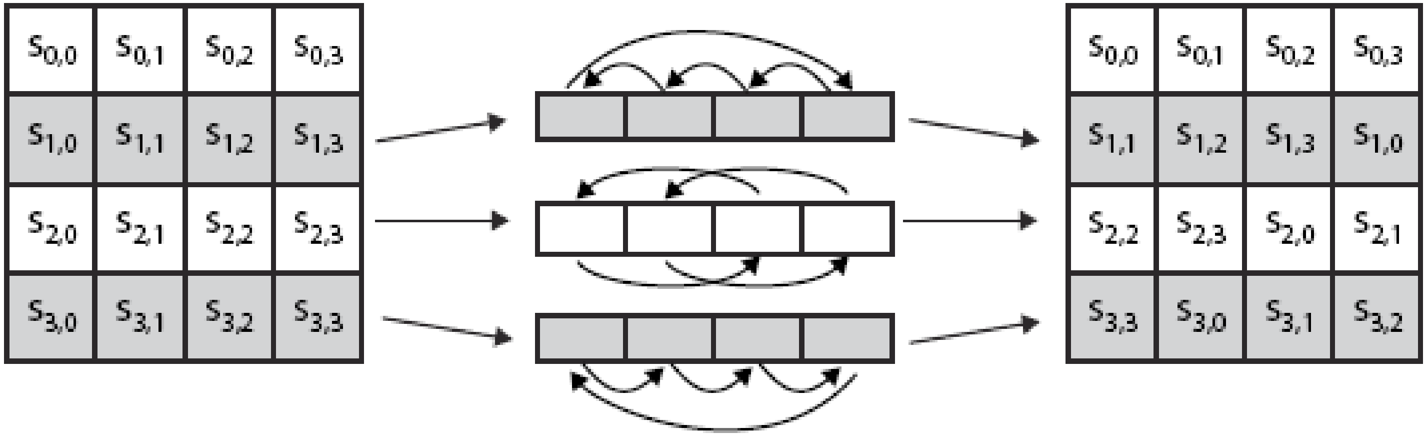
Byte Substitution



Shift Rows

- A circular byte shift in each row
  - 1st row is unchanged
  - 2nd row does 1 byte circular shift to left
  - 3rd row does 2 byte circular shift to left
  - 4th row does 3 byte circular shift to left
- Decrypt inverts using shifts to right.
- Since state is processed by columns, this step permutes bytes between the columns.

Shift Rows



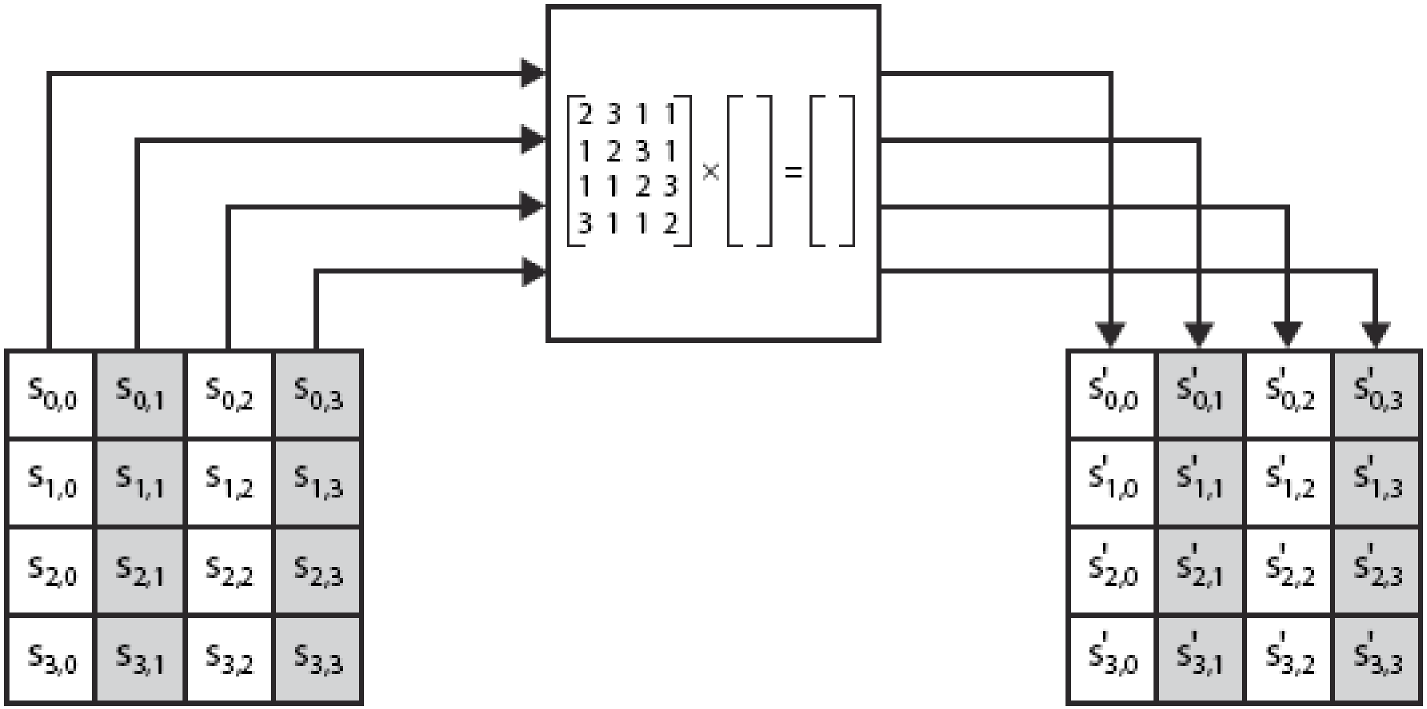
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{1,1}$ $s_{1,2}$ $s_{1,3}$
$s_{2,0}$ $s_{2,1}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,1}$ $s_{3,2}$ $s_{3,3}$
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,1}$ $s_{1,2}$ $s_{1,3}$ $s_{1,0}$
$s_{2,2}$ $s_{2,3}$ $s_{2,0}$ $s_{2,1}$
$s_{3,3}$ $s_{3,0}$ $s_{3,1}$ $s_{3,2}$

Mix Columns

- Each column is processed separately.
- Each byte is replaced by a value dependent on all 4 bytes in the column.
- Effectively a matrix multiplication in GF(2^8) using prime poly $m(x) = x^8+x^4+x^3+x+1$

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

Mix Columns



$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{1,1}$ $s_{1,2}$ $s_{1,3}$
$s_{2,0}$ $s_{2,1}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,1}$ $s_{3,2}$ $s_{3,3}$
$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \times \begin{bmatrix} \\ \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \end{bmatrix}$
$s'_{0,0}$ $s'_{0,1}$ $s'_{0,2}$ $s'_{0,3}$
$s'_{1,0}$ $s'_{1,1}$ $s'_{1,2}$ $s'_{1,3}$
$s'_{2,0}$ $s'_{2,1}$ $s'_{2,2}$ $s'_{2,3}$
$s'_{3,0}$ $s'_{3,1}$ $s'_{3,2}$ $s'_{3,3}$

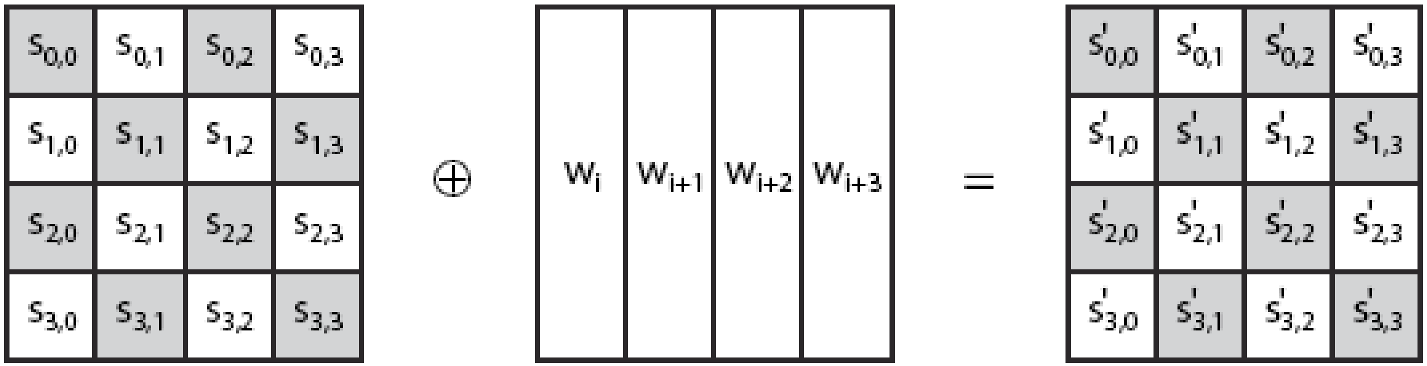
Mix Columns

- Expresses each col of the new state as 4 equations.
  - One equation to derive each new byte in col
- Decryption requires use of inverse matrix with larger coefficients, hence a little harder
- Have an alternate characterization
  - each column a 4-term polynomial
  - with coefficients in $GF(2^8)$
  - and polynomials multiplied modulo (x^4+1)

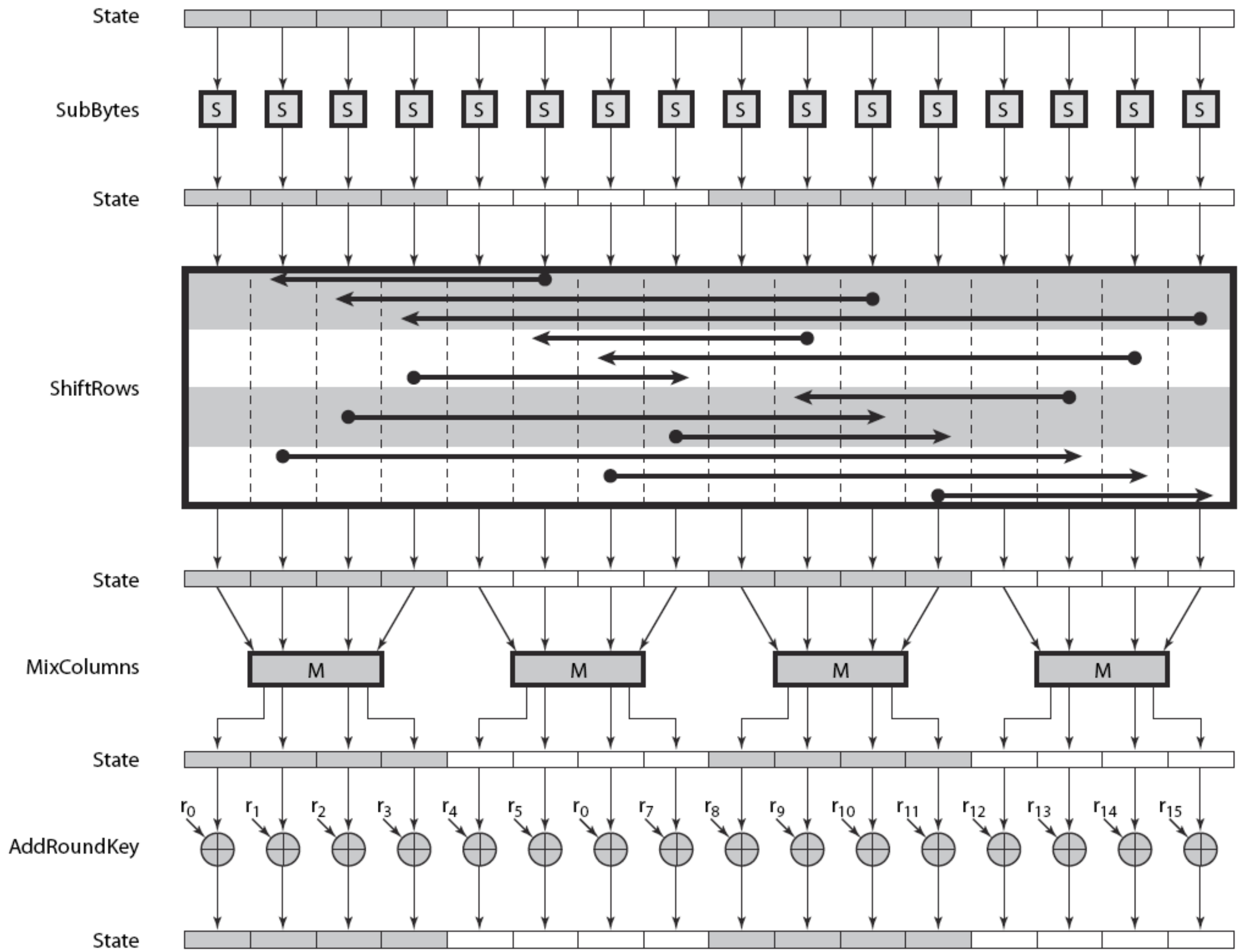
Add Round Key

- Xor state with 128-bits of the round key.
- Again processed by column (though effectively a series of byte operations).
- Inverse for decryption identical
  - since XOR own inverse, with reversed keys
- Designed to be as simple as possible.

Add Round Key



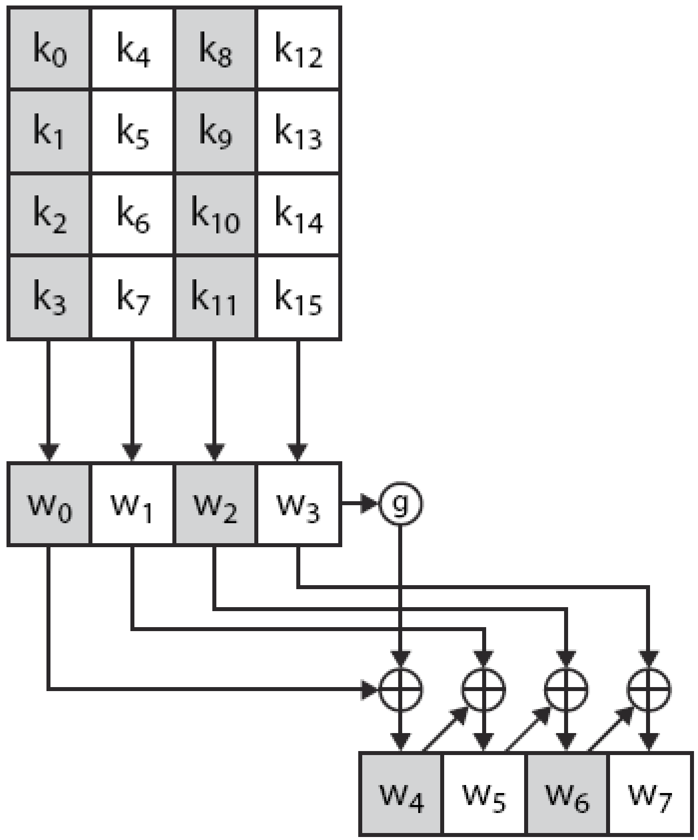
AES Round


State
SubBytes
S
State
ShiftRows
State
MixColumns
M
State
AddRoundKey
r_0 r_1 r_2 r_3 r_4 r_5 r_0 r_7 r_8 r_9 r_{10} r_{11} r_{12} r_{13} r_{14} r_{15}
State

AES Key Scheduling

- Takes 128-bit (16-byte) key and expands into array of 44 32-bit words

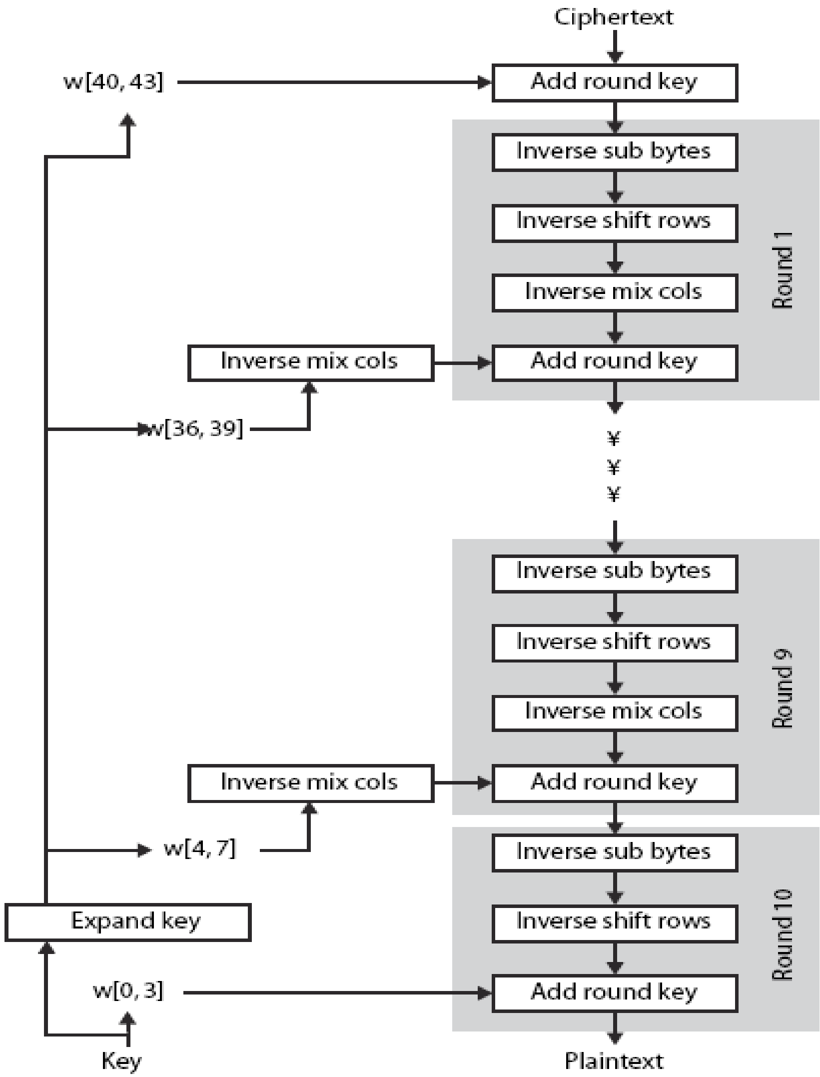
AES Key Expansion



AES Decryption

- AES decryption is not identical to encryption since steps done in reverse.
- But can define an equivalent inverse cipher with steps as for encryption.
  - but using inverses of each step
  - with a different key schedule
- It works since result is unchanged when
  - swap byte substitution & shift rows
  - swap mix columns & add (tweaked) round key

AES Decryption



AES- Design considerations

- Not a Feistel scheme: so diffusion is faster, but it’s a new scheme, so less analyzed.
- S-box: mathematically constructed: based on the x → x^(-1) transformation.
- Shift row- to resist two recent attack: truncated differential and the square attack.
- Key scheduling – nonlinear (uses the S-box) mixing of the key bits.
- 10 rounds: there are attacks better than brute-search for Rijndael-with-7-rounds, so extra 3 rounds for safety.