History

- clear a replacement for DES was needed
- can use Triple-DES – but slow with small blocks
- US NIST issued call for ciphers in 1997
- 15 candidates accepted in Jun 98
- 5 were short-listed in Aug-99
- Rijndael was selected as the AES in Oct-2000
- issued as FIPS PUB 197 standard in Nov-2001
### AES Requirements/Criteria

- private key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- active life of 20-30 years
- both C & Java implementations
- criteria
  - general security
  - software & hardware implementation ease
  - implementation attacks
  - flexibility (in en/decrypt, keying, other factors)

### The AES Cipher - Rijndael

- designed by Rijmen-Daemen in Belgium
- has 128 (AES-128), 192 (AES-192), 256(AES-256) bit keys, 128 bit data
- an **iterative** rather than **feistel** cipher
  - treats data in 4 groups of 4 bytes
  - operates an entire block in every round
AES

- processes data as 4 groups of 4 bytes (state)
- has 10/12/14 rounds (depending on the key length), in each the following operations are performed:
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multiply of groups)
  - add round key (XOR state with key material)
- all operations can be combined into XOR and table lookups - hence very fast & efficient
Byte Substitution

- a simple substitution of each byte
- uses one table of 16x16 bytes containing a permutation of all 256 8-bit values
- each byte of state is replaced by byte in row (left 4-bits) & column (right 4-bits)
  - eg. byte (95) is replaced by row 9 col 5 byte
  - which is the value (2A)
- S-box is constructed using a defined transformation of the values in GF(2^8)
- designed to be resistant to all known attacks

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 does shifts to right
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 $\text{GF}(2^8)$ using field polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$

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

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 is identical since XOR is own inverse, just with correct round key
- designed to be as simple as possible
AES Key Expansion

- takes 128-bit (16-byte) key and expands into array of 44 (AES-128), 52 (AES-192), 60 (AES-256) 32-bit columns
- start by copying key into first 4 words
- then loop creating words that depend on values in previous & 4 places back
  - in 3 of 4 cases just XOR these together
  - every 4th has S-box + rotate + XOR constant of previous before XOR together
- designed to resist known attacks
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.
- Works since result is unchanged when:
  - Swap byte substitution & shift rows.
  - Swap mix columns & add round key.

Implementation Aspects

- Can be efficiently implemented on 8-bit CPU.
  - Byte substitution works on bytes using a table of 256 entries.
  - Shift rows is simple byte shifting.
  - Add round key works on byte XORs.
  - Mix columns requires matrix multiply in GF(2^8) which works on byte values, can be simplified to use a table lookup.
Implementation Aspects

- can be efficiently implemented on 32-bit CPU
  - redefine steps to use 32-bit words
  - can pre-compute 4 tables of 256-words
  - then each column in each round can be computed using 4 table lookups + 4 XORs
  - at a cost of 16Kb to store tables
- designers believe this very efficient implementation was a key factor in its selection as the AES cipher

Summary

- have considered:
  - the AES selection process
  - the details of Rijndael – the AES cipher
  - looked at the steps in each round
  - the key expansion
  - implementation aspects