CS408
Cryptography & Internet Security

Lecture 9:

AES

Admin stuff

- Assignment #1 has been posted and is due on Feb 24, 2015, in the beginning of class (4pm)
- Course webpage (general information):
  http://web.njit.edu/~crix/CS.408
- Course material (lecture slides, assignments etc):
  http://web.njit.edu/~crix/CS.408/content

Advanced Encryption Standard (AES) - History

- 1997: NIST call for candidates to replace DES
  - Requirements:
    - Key sizes of 128, 192, and 256 bits
    - Blocks of 128 bits
    - Should work on a variety of different hardware
    - Fast
    - Cryptographically strong
- 2 rounds:
  - 1st round: 5 finalists were chosen from 15 candidates
  - 2nd round: Rijndael was chosen from the 5 finalists (MARS, RC6, Rijndael, Serpent, Twofish)
- Rijndael was developed by two Belgian cryptographers (Joan Daemen, Vincent Rijmen)
- In 2001, NIST announced AES as a standard (FIPS 197), and in 2002 AES became a US Federal Government standard

AES: Evaluation Criteria

- Security
- Costs
- Intellectual property
- Implementation and execution
- Versatility
- Key agility
- Simplicity

- As a side note, on my laptop:
  - AES-128 encryption: 142 MB/s
  - DES encryption: 48 MB/s
  - DES3 (EDE): 18 MB/s

Rijndael: Overview

- Block cipher with block length of 128 bits
- Three key sizes: 128, 192, or 256 bits
- Number of rounds: 10, 12, or 14 (for keys of size 128, 192, and 256 bits, respectively)
- Decryption does not use the same algorithm as encryption
- Can be used in several modes of operation (ECB, CBC, CFB, OFC, CTR, etc.)
- Is based on a substitution-permutation network (similar to a Feistel network, but has more "inherent parallelism")
- Resistant to all known attacks (including linear and differential cryptanalysis)

Rijndael: Round Structure

- Each round uses several basic steps, one of which depends on the round key
  - Like in DES, for each round there is a round key derived from the original key
  - We'll study the version with 10 rounds (128 bit key)
  - There are 10 round keys (each of 128 bits), for rounds 1-10
  - The original key is considered as 0^{th} round key
- The basic steps:
  - ByteSub transformation (BS): non-linear step which provides resistance against differential and linear cryptanalysis
  - ShiftRow transformation (SR): linear mixing step causes diffusion of the bits over multiple rounds
  - MixColumn transformation (MC): similar purpose to SR
  - AddRoundKey (ARK): the round key is XOR-ed with the result of the previous step
- A round consists of:
  BS $\Rightarrow$ SR $\Rightarrow$ MC $\Rightarrow$ ARK

Rijndael Encryption: The Basic Algorithm

1. AddRoundKey, using round key 0
2. Nine rounds, each consists of:
   - ByteSub
   - ShiftRow
   - MixColumn
   - AddRoundKey

   using round keys 1 to 9
3. A final round (round 10) consisting of:
   - ByteSub
   - ShiftRow
   - AddRoundKey

   using round key 10

The Input for Encryption

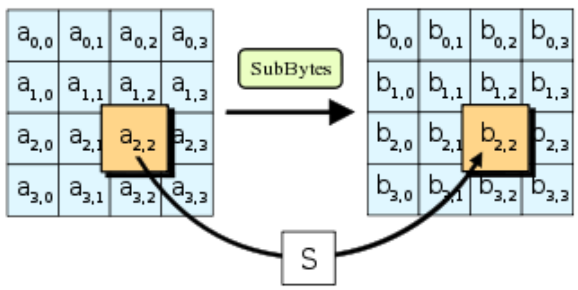
- The plaintext input for encryption is a block of 128 bits, which are grouped into 16 bytes (each of 8 bits):

  $a_{0,0}, a_{1,0}, a_{2,0}, a_{3,0}, a_{0,1}, a_{1,1}, \ldots, a_{3,3}$
- These 16 bytes are arranged into a 4x4 matrix:

$$\begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}$$

The ByteSub Transformation

- Each byte is substituted to another byte, according to the S-Box (a 16 x 16 substitution matrix)
  - If byte a = $a_1a_2a_3a_4a_5a_6a_7a_8$, then byte a is substituted with the byte in S-Box at row $a_1a_2a_3a_4$ and column $a_5a_6a_7a_8$
  - S-Box implements a non-linear substitution



Rijndael S-Box

- How is Rijndael S-Box different than DES S-Box?
  - Only one S-Box
  - S-Box is based on modular arithmetic with polynomials, which can be defined algebraically and are not random
  - Easy to analyze, prove attacks fail

The ShiftRow Transformation

- The four rows of the matrix are shifted cyclically to the left by the offsets of 0, 1, 2, and 3, respectively

$$\begin{pmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,0} & b_{1,1} & b_{1,2} \leftarrow & b_{1,3} \\ b_{2,0} & b_{2,1} \leftarrow & b_{2,2} & b_{2,3} \\ b_{3,0} \leftarrow & b_{3,1} & b_{3,2} & b_{3,3} \end{pmatrix}$$

$$\begin{pmatrix} c_{0,0} & c_{0,1} & c_{0,2} & c_{0,3} \\ c_{1,0} & c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,0} & c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,0} & c_{3,1} & c_{3,2} & c_{3,3} \end{pmatrix} = \begin{pmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,1} & b_{1,2} & b_{1,3} & b_{1,0} \\ b_{2,2} & b_{2,3} & b_{2,0} & b_{2,1} \\ b_{3,3} & b_{3,0} & b_{3,1} & b_{3,2} \end{pmatrix}$$

The MixColumn Transformation

- Regard ($c_{i,j}$) as a 4x4 matrix with entries in GF(2^8) and multiply it by another fixed matrix, again with entries in GF(2^8), to obtain ($d_{i,j}$)
  - GF(2^8) is a finite field, in which addition and multiplication follow special rules
    - Elements in GF(2^8) are polynomials of degree at most 7 whose coefficients are 0 or 1
  - Each byte is seen as element of GF(2^8) as follows:

  Byte $B=B_7B_6B_5B_4B_3B_2B_1B_0$ is $B_7x^7+B_6x^6+\ldots B_1x^1+B_0$

$$\begin{pmatrix} d_{0,0} & d_{0,1} & d_{0,2} & d_{0,3} \\ d_{1,0} & d_{1,1} & d_{1,2} & d_{1,3} \\ d_{2,0} & d_{2,1} & d_{2,2} & d_{2,3} \\ d_{3,0} & d_{3,1} & d_{3,2} & d_{3,3} \end{pmatrix} = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} \begin{pmatrix} c_{0,0} & c_{0,1} & c_{0,2} & c_{0,3} \\ c_{1,0} & c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,0} & c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,0} & c_{3,1} & c_{3,2} & c_{3,3} \end{pmatrix}$$

The AddRoundKey Transformation

- The round key is derived from the original main key
- The round key has 128 bits, arranged in a 4x4 matrix ($k_{i,j}$) consisting of bytes
- The matrix ($k_{i,j}$) is XOR-ed with the output of the previous step

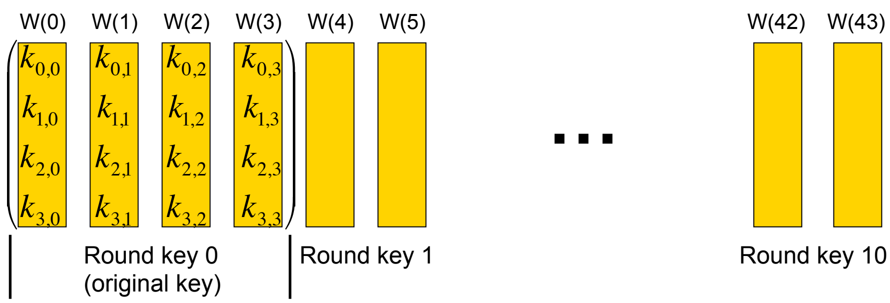
$$\begin{pmatrix} e_{0,0} & e_{0,1} & e_{0,2} & e_{0,3} \\ e_{1,0} & e_{1,1} & e_{1,2} & e_{1,3} \\ e_{2,0} & e_{2,1} & e_{2,2} & e_{2,3} \\ e_{3,0} & e_{3,1} & e_{3,2} & e_{3,3} \end{pmatrix} = \begin{pmatrix} d_{0,0} & d_{0,1} & d_{0,2} & d_{0,3} \\ d_{1,0} & d_{1,1} & d_{1,2} & d_{1,3} \\ d_{2,0} & d_{2,1} & d_{2,2} & d_{2,3} \\ d_{3,0} & d_{3,1} & d_{3,2} & d_{3,3} \end{pmatrix} \oplus \begin{pmatrix} k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} \end{pmatrix}$$

Rijndael Encryption: The Basic Algorithm

1. AddRoundKey, using round key 0
2. Nine rounds, each consists of:
   ByteSub
   ShiftRow
   MixColumn
   AddRoundKey
   using round keys 1 to 9
3. A final round (round 10) consisting of:
   ByteSub
   ShiftRow
   AddRoundKey
   using round key 10

The Key Schedule

- The original key has 128 bits, viewed as a 4x4 matrix of bytes (or 4 columns W(0), W(1), W(2), W(3))
  - This is known as round key 0
- We compute 40 more columns recursively,
  W(4), …, W(43), which are the round keys



The Key Schedule (continued)

- For i=4..43:

  if i mod 4 = 0 then

  $W(i) = W(i-4) \oplus T(W(i-1))$

  else

  $W(i) = W(i-4) \oplus W(i-1)$

- The round key for round i consists of the columns: W(4i), W(4i+1), W(4i+2), W(4i+3)

The Design of the S-Box

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

- The S-Box is implemented as a lookup table, but it has a mathematical description

The Design of the S-Box (continued)

- Given a byte X = $x_7x_6x_5x_4x_3x_2x_1x_0$, how is the corresponding S-Box value computed? (used for substitution in the ByteSub step)
  - Compute its multiplicative inverse Y = $y_7y_6y_5y_4y_3y_2y_1y_0$ in GF(2^8) (i.e., XY=1)
  - Then apply the following affine transformation:

$$\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \\ z_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

  - The byte Z = $z_7z_6z_5z_4z_3z_2z_1z_0$ is the entry in the S-Box

The Design of the S-Box (continued)

- Example: byte X = 11001011
- The inverse of X in $GF(2^8)$ is Y=00000100
- After the affine transformation, we get Z=00011111 (this is 1F in hexadecimal)
- Indeed, in the S-Box, at row 12 and column 11, we find the value 1F

The Design of the S-Box (continued)

- The use of the inverse is to achieve non-linearity and provide resistance against differential and linear cryptanalysis
- The multiplication by the matrix and the addition of the vector (affine transformation) was used to provide resistance against algebraic attacks
  - The matrix was chosen because of its simple form
  - The vector was chosen so that no input ever equals its S-Box output or the bitwise complement of its S-Box output
    - S-Box[x] $\oplus$ x $\neq$ {00}
    - S-Box[x] $\oplus$ x $\neq$ {FF}

Rijndael Decryption

- Each of the steps ByteSub, ShiftRow, MixColumn, and AddRoundKey is invertible:
  - The inverse of ByteSub is another lookup table called InvByteSub
  - The inverse of ShiftRow is obtained by shifting the rows to the right instead of to the left, called InvShiftRow
  - The inverse of MixColumn exists because the matrix used in MixColumn is invertible. The step is called InvMixColumn
  - AddRoundKey is its own inverse (why?)
- Decryption is not as fast as encryption

Rijndael Encryption: The Basic Algorithm

1. AddRoundKey, using round key 0
2. Nine rounds, each consists of:
   ByteSub
   ShiftRow
   MixColumn
   AddRoundKey
   using round keys 1 to 9
3. A final round (round 10) consisting of:
   ByteSub
   ShiftRow
   AddRoundKey
   using round key 10

Rijndael Decryption

- To decrypt, we run through the 10 rounds in reverse order
  - The decryption algorithm is not the same as the encryption algorithm, but the key schedule is the same (keys are used in reverse order);
    what does this imply about existence of weak keys?

1. A first round consisting of:
   AddRoundKey
   InvShiftRow
   InvByteSub
   using round key 10
2. Nine rounds, each consists of:
   AddRoundKey
   InvMixColumn
   InvShiftRow
   InvByteSub
   using round keys 9 to 1
3. AddRoundKey, using round key 0

Encryption and Decryption

Encryption

1. AddRoundKey, using round key 0

2. Nine rounds, each consists of:
   ByteSub
   ShiftRow
   MixColumn
   AddRoundKey
   using round keys 1 to 9

3. A final round (round 10) consisting of:
   ByteSub
   ShiftRow
   AddRoundKey
   using round key 10

Decryption

1. A first round consisting of:
   AddRoundKey
   InvShiftRow
   InvByteSub
   using round key 10

2. Nine rounds, each consists of:
   AddRoundKey
   InvMixColumn
   InvShiftRow
   InvByteSub
   using round keys 9 to 1

3. AddRoundKey, using round key 0

Rijndael Cryptanalysis

- Resistant to differential and linear cryptanalysis
- Theoretical break on weaker version of the cipher, which only has 9 rounds
  - Requires 2^{224} computation and 2^{85} chosen related-key plaintexts
  - Attack is not practical
- You can read more about attacks against AES:

http://en.wikipedia.org/wiki/Advanced_Encryption_Standard#Security

Recommended Reading

- Chapter 5
  - You can read more about design considerations in Chapter 5.4