CS 578 – Cryptography

Prof. Michael Backes

Basics of Block Ciphers, DES, AES

April 28, 2006

Administrative Announcements

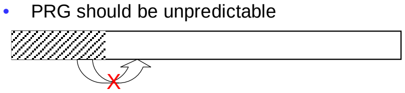
- Practical classes:
  - Start next week, assignment to groups on the web page this weekend
- Lecture Notes:
  - Usually put out on Tuesday for the current and following lecture (sometimes more)
- Quizzes:
  - Quizzes always treat topics of the Friday-Tuesday lecture block that the exercise sheet tackled, quizzes last 15 min.
  - Quizzes are written in English
  - First quiz next Wednesday (on lectures 1 + 2)
- Discussion board
  - Please register as announcements on the course/exercises/quizzes, etc. will be given there
  - http://infsec.cs.uni-sb.de/wbb2/

Recall: Ciphers, OTP and Perfect Secrecy

- Ciphers as pair (E,D) of algorithms defined over ($\mathcal{K},\mathcal{M},\mathcal{C}$) such that for all K,m: D(K,E(K,m)) = m.
- First "proven secure" cipher: OTP
  Encryption: $c = E(K,m) = K \oplus m$
  Decryption: $m = D(K,c) = K \oplus c$
- A cipher (E,D) defined over ($\mathcal{M},\mathcal{K},\mathcal{C}$) has perfect secrecy if for all $m_0, m_1 \in \mathcal{M}$, and for all $c \in \mathcal{C}$:
  $\Pr[c = c';\ K \leftarrow_R \mathcal{K},\ c' \leftarrow E(K,m_0)]$
  $= \Pr[c = c';\ K \leftarrow_R \mathcal{K},\ c' \leftarrow E(K,m_1)]$

Recall: Stream Ciphers and PRGs

- OTP has perfect secrecy, but unpractical
- Bad news: any cipher with perfect secrecy has len(K) ≥ len(m)
- Circumventing the problem: stream ciphers, E(K,m) = m ⊕ PRG(K)
- PRG: $\{0,1\}^k \rightarrow \{0,1\}^{p(k)}$
  - Public deterministic function (only unknown seed)
  - PRG should be unpredictable



Recall: Stream Cipher Examples

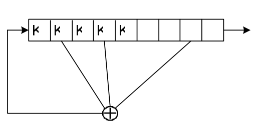
- RC4, LFSR (again today), …
- Another example: T-Function (Shamir 2002)
  1. $x \leftarrow x + (x^2 \vee C) \bmod 2^n$.
  2. Output msb(x)
  3. Go to 1.

  Very long repetition period (cycle of length 2^n), seems unpredictable

  Only requirement: lsb(C) = lsb_3(C) = 1
- Important for stream ciphers: Never use key more than once

Hardware PRG

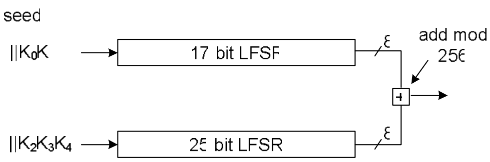
- LFSR (Linear Feedback Shift Register)
  - used in CSS, GSM
  - Standard solution for doing cheap hardware encryption (as a stream cipher)



- Seed: Initial value of the register
- On their own not usable: first bits output are key bits

CSS

- Content Scrambling System (CSS)
- Key = 40 bits = 5 bytes ($K_0K_1K_2K_3K_4$)



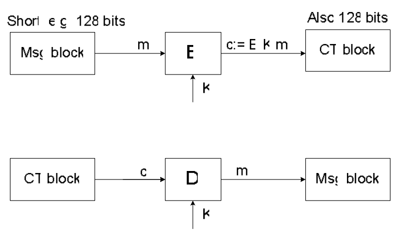
- Easy to break in time ca. 2^{20} (2^{40} with brute force)

Performance of Stream Ciphers

Crypto++ 5.2.1 Benchmarks [by Wei Dei]

	Algorithm	Megabytes(2^20 bytes) Processed	Time Taken	MB/Second
Stream ciphers	RC4	512	4.517	113.350
	SEAL	1.024	3.485	293.831
	BBS 512	0.25	4.096	0.070
Block ciphers	DES	128	5.998	21.340
	DES-X	128	6.159	20.783
	3-DES	64	6.499	9.848
	IDEA	64	3.375	18.963
	Rijndael (128-bit key)	256	4.196	61.010
	Rijndael (192-bit key)	256	4.817	53.145
	Rijndael (256-bit key)	256	5.308	48.229

Block Ciphers



Data Encryption standard (DES)

History:

1. 1967: Feistel at IBM: Lucifer
   key-len = 128 bits, msg-len = CT-len = 128 bits
2. 1972: National Bureau of Standards (NBS, now NIST) asked for federal encryption standard
   → IBM developed DES
3. 1975: DES became the standard:
   key-len = 56 bits, msg-len = CT-len = 64 bits
4. DES (somewhat) vulnerably to brute-force today
   - 3-DES; 56 bits → 3*56 = 168 bits
   - Successor of DES: Advanced Encryption Standard (AES):
     - 1998: NIST: Competition for DES replacement
     - 2000: Adopted Rijndael as AES

Performance overview of DES

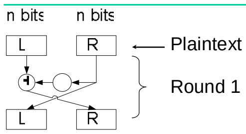
Crypto++ 5.2.1 Benchmarks [by Wei Dei]

	Algorithm	Megabytes(2^20 bytes) Processed	Time Taken	MB/Second
Stream ciphers	RC4	512	4.517	113.350
	SEAL	1.024	3.485	293.831
	BBS 512	0.25	4.096	0.070
Block ciphers	DES	128	5.998	21.340
	DES-X	128	6.159	20.783
	3-DES	64	6.499	9.848
	IDEA	64	3.375	18.963
	Rijndael (128-bit key)	256	4.196	61.010
	Rijndael (192-bit key)	256	4.817	53.145
	Rijndael (256-bit key)	256	5.308	48.229

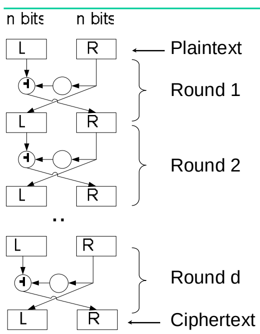
Intuition of DES

- Basic Idea: Feistel Networks:
  - Also used in IDEA, RC5, Skipjack, …
  - AES does not use a Feistel Network!
- Feistel Networks:
  $f_1,\ldots,f_d: \{0,1\}^n \rightarrow \{0,1\}^n$
  (for DES: n = 32, d=16)

Feistel Networks



Feistel Networks



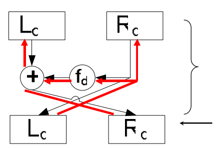
Intuition of DES

- Basic Idea: Feistel Networks:
  - Also used in IDEA, RC5, Skipjack, …
  - AES does not use a Feistel Network!
- Feistel Networks:
  - $f_1,\ldots,f_d$: $\{0,1\}^n \rightarrow \{0,1\}^n$
    (for DES: n = 32, d=16)
  - $(L_0,R_0) \leftarrow$ Partition (PT)
  - For i = 1 to d
    - $L_i \leftarrow R_{i-1}$
    - $R_i \leftarrow L_{i-1} \oplus f_i(R_{i-1})$

Feistel is one-to-one

- Claim: For any functions $f_1,\ldots,f_d$, a Feistel network is a one-to-one map $F: \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$
- Proof: Construct F^{-1} given CT in $\{0,1\}^{2n}$, CT = (L_d, R_d)

Feistel Networks



Feistel is one-to-one

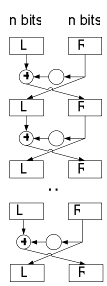
- Claim: For any functions $f_1,\ldots,f_d$, a Feistel network is a one-to-one map $F: \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$
- Proof: Construct F^{-1} given CT in $\{0,1\}^{2n}$, CT = (L_d, R_d)
- In symbols:
  - $R_{d-1} = L_d$ $\quad L_{d-1} = R_d \oplus f_d(L_d)$
  - $R_{d-2} = L_{d-1}$ $\quad L_{d-2} = R_{d-1} \oplus f_{d-1}(L_{d-1})$
  - …
  - $R_0 = L_1$ $\quad L_0 = R_1 \oplus f_1(L_1)$
- Feistel Networks inverts itself!
  - Apply functions f_i in reverse order, f_d, f_{d-1}, f_{d-2} …

Feistel “encryption” and “decryption”

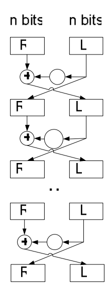
- Encryption by Feistel Network:
  - $f_1,\ldots,f_d$: $\{0,1\}^{32} \rightarrow \{0,1\}^{32}$
  - $(L_0,R_0) \leftarrow$ Partition (PT)
  - $L_1 = R_0$ $\quad R_1 = L_0 \oplus f_1(R_0)$
  - $L_2 = R_1$ $\quad R_2 = L_1 \oplus f_2(R_1)$
  - …
  - $L_d = R_{d-1}$ $\quad R_d = L_{d-1} \oplus f_d(R_d)$
- Decryption by backwards Feistel traversal:
  - $R_{d-1} = L_d$ $\quad L_{d-1} = R_d \oplus f_d(L_d)$
  - $R_{d-2} = L_{d-1}$ $\quad L_{d-2} = R_{d-1} \oplus f_{d-1}(L_{d-1})$
  - …
  - $R_0 = L_1$ $\quad L_0 = R_1 \oplus f_1(L_1)$

Decrypting Feistel Networks

Encryption

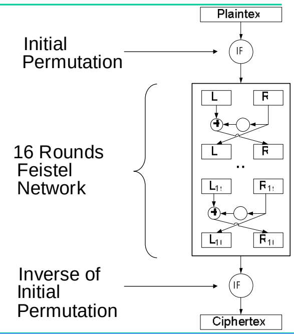


Decryption



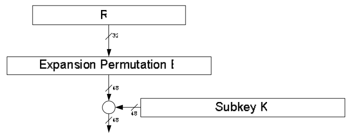
DES

- DES: 16-round Feistel Network:
  - $f_1,\ldots,f_{16}$: $\{0,1\}^{32} \rightarrow \{0,1\}^{32}$

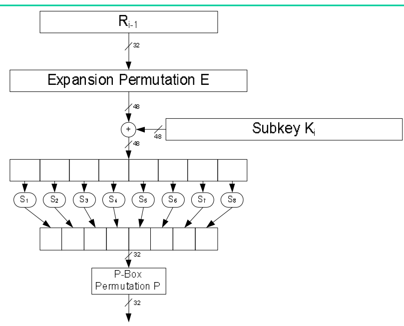


Feistel Functions in DES

- $f_i(R_{i-1}) := f(R_{i-1}, K_i)$, R_{i-1} in $\{0,1\}^{32}$
- The keys K_i (48 bits) are derived from K (56 bits) by specified key schedule (bit-subset relation)
- $f(R_{i-1}, K_i)$: R_{i-1} (32 bits), K_i (48 bits)
  1. R_{i-1} (32 bits) → R_{i-1}' (48 bits) (blowup by bit replication)
  2. $f(R_{i-1}, K_i) := R_{i-1}'$ (48 bits) $\oplus$ K_i (48 bits)



Feistel Functions in DES (cont'd)



S-Boxes

S_1:

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	O	6	13

S_2:

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3:

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4:

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5:

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	O	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S-Boxes and P-Box

S_6:

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7:

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

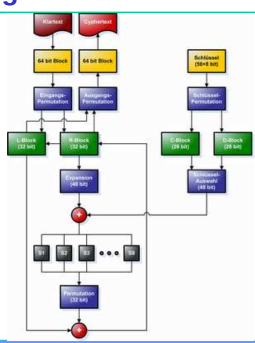
S_8:

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

P:

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

High-level Review of DES



History of AES

- AES: Advanced Encryption Standard
- 1997: NIST publishes RFP
- 1998: 165 submissions, 5 susceptible to attacks
- 1999: NIST chooses 5 finalists
- 2000: Rijndael selected the winner
- Key sizes: 128, 192, or 256 bits
- Block sizes: 128 bits

Parameters of Rijndael and AES

- AES: September 2000
- Rijndael (secure for the next 20-30 years?):
  - Flexible key sizes: 128, 192, or 256 bits
  - Flexible block sizes: 128, 192 or 256 bits
- AES: required Rijndael to use blocks of size 128 bits, keys of 128 bits typical

The Rijndael Cipher (for 128-bit Key)

- Rijndael Key K:

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

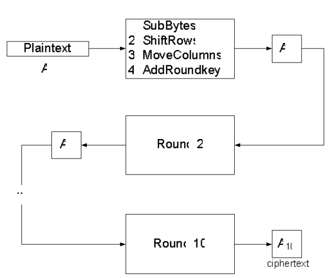
4x4 matrix of bytes

- Rijndael state A:

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

4x4 matrix of bytes

The Rijndael Cipher (cont'd)

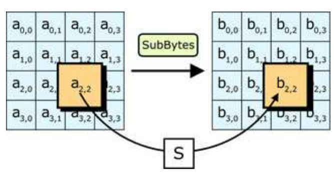


Details on AES (cont'd)

- AES round:
  1. SubBytes: A[i,j] ← s-box(A[i,j])

Details on AES

- SubBytes: non-linear substitution step, each byte replaced according to a lookup table
- S = 8-bit lookup table (Matrix from GF(2^8))

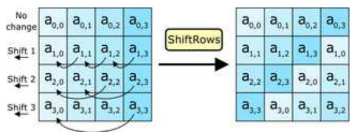


Details on AES (cont'd)

- AES round:
  1. SubBytes: A[i,j] ← s-box(A[i,j])
     (s-box based on inversion in GF(2^8))
  2. ShiftRows: For i=0,1,2,3: Rotate left row i by i pos.

Details on AES (cont'd)

- ShiftRows: transposition step, each row shifted cyclically a certain number of steps.

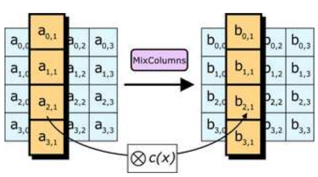


Details on AES (cont'd)

- AES round:
  1. SubBytes: A[i,j] ← s-box(A[i,j]) (s-box based on inversion in GF(2^8))
  2. ShiftRows: For i=0,1,2,3: Rotate left row i by i pos.
  3. MixColumns: Multiply each column to fixed matrix (over GF(2^8))

Details on AES (cont'd)

- MixColumns: multiplication of columns by 4x4 matrix; mixing operating on columns combining four bytes in each column using a linear transformation.

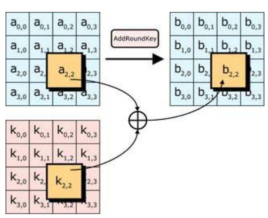


Details on AES (cont'd)

- AES round:
  1. SubBytes: A[i,j] ← s-box(A[i,j])
     (s-box based on inversion in GF(2^8))
  2. ShiftRows: For i=0,1,2,3: Rotate left row i by i pos.
  3. MixColumns: Multiply each column to fixed matrix (over GF(2^8)) → $A_i^{(3)}$
  4. AddRoundKey:
     - $A_{i+1} \leftarrow A_i^{(3)} \oplus K_i$
     - K_i = i-th round key derived from 128-bit key K

Details on AES (cont'd)

- AddRoundKey: XOR state with round key; round key derived from the key by explicit key schedule



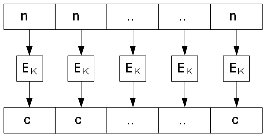
Performance of AES

Crypto++ 5.2.1 Benchmarks [by Wei Dei]

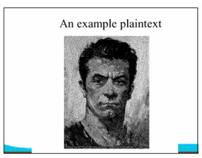
	Algorithm	Megabytes(2^20 bytes) Processed	Time Taken	MB/Second
Stream ciphers	RC4	512	4.517	113.350
	SEAL	1.024	3.485	293.831
	BBS 512	0.25	4.096	0.070
Block ciphers	DES	128	5.998	21.340
	DES-X	128	6.159	20.783
	3-DES	64	6.499	9.848
	IDEA	64	3.375	18.963
	Rijndael (128-bit key)	256	4.196	61.010
	Rijndael (192-bit key)	256	4.817	53.145
	Rijndael (256-bit key)	256	5.308	48.229

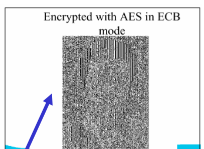
Outlook: How (not) to use Block Ciphers

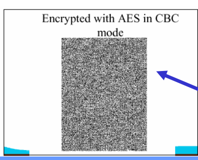
- Electronic Codebook (ECB)
  - Intuitive but naïve way (how not to do it)



Comparison (for AES, by Bart Preneel)







Similar plaintext blocks produce similar ciphertext (see outline of head)

No apparent pattern