Symmetric Cryptography

Cryptography: terminology (1/2)

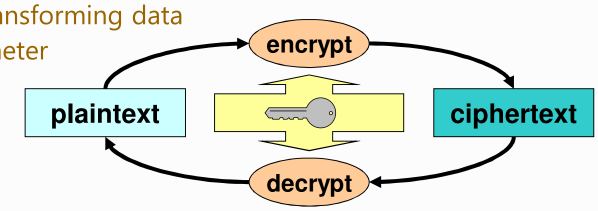
- Cryptography
  - Art or science of hidden writing
    - from Gr. *kryptós*, hidden + *graph*, r. of *graphein*, to write
  - It was initially used to maintain the confidentiality of information
  - Steganography
    - from Gr. *steganós*, hidden + *graph*, r. of *graphein*, to write
- Cryptanalysis
  - Art or science of breaking cryptographic systems or encrypted information
- Cryptology
  - Cryptography + cryptanalysis

Cryptography: terminology (2/2)

- Cipher
  - Specific cryptographic technique
- Cipher operation

  Encryption: plaintext (or cleartext) → ciphertext (or cryptogram)
  Decryption: ciphertext → plaintext

  Algorithm: way of transforming data
  Key: algorithm parameter



The players

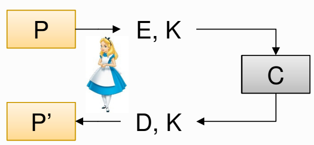
I'VE DISCOVERED A WAY TO GET COMPUTER SCIENTISTS TO LISTEN TO ANY BORING STORY.

- Alice & Bob
  - The fundamental honest people
  - They represent two abstract interacting entities
- Carol, Dave, …
  - More honest entities for complex protocols
- Eve
  - Passive eavesdropper
- Mallory
  - Malicious attacker
- Trent
  - Trusted by all

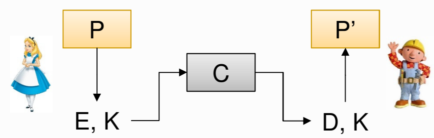
[Who are Alice and Bob? (By Bruce Schneier)]

Use cases

- Self-protection with key K
  - Alice encrypts plaintext P with key K
    - A: $C = \{P\}_K$
  - Alice decrypts cryptogram C with key K
    - A: $P' = \{C\}_K$
  - P' should be equal to P (requires checking)



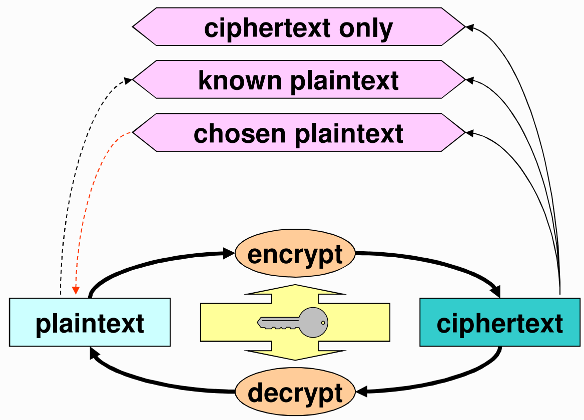
- Secure communication with key K
  - Alice encrypts plaintext P with key K
    - A: $C = \{P\}_K$
  - Bob decrypts C with key K
    - B: $P' = \{C\}_K$
  - P' should be equal to P (requires checking)



Cryptanalysis: goals

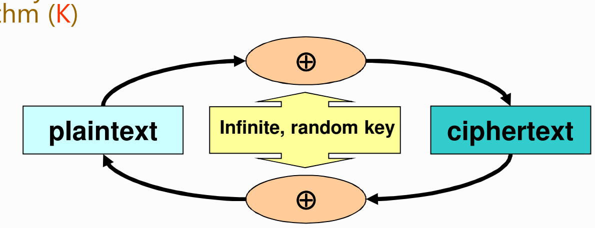
- Discover original plaintext
  - Which originated a given ciphertext
- Discover a cipher key
  - Allows the decryption of ciphertexts created with the same key
- Discover the cipher algorithm
  - Or an equivalent algorithm…
  - Usually algorithms are not secret, but there are exceptions
    - Lorenz, A5 (GSM), RC4 (WEP), Crypto-1 (Mifare)
    - Algorithms for DRM (Digital Rights Management)
  - Reverse engineering

Cryptanalysis attacks: approaches



Cryptography:
Information-theoretic security

- Plaintext space
  - Set of all possible plaintext messages (M)
- Ciphertext space
  - Set of all possible ciphertext values (C)
- Key space
  - Set of all possible key values for a given algorithm (K)
- Perfect security
  - Given $c_j \in C$, $p(m_i, k_j) = p(m_i)$
  - $\#K \geq \#M$
  - Vernam cipher (one-time pad)
- The cipher cannot be broken
  - Even by adversaries with unlimited computing power



Cryptography: computational security

- ▷ The number of possible keys is finite
  - And much less than the number os possible messages
  - #K << #M
- ▷ Thus, security ultimately depends on the computing power of cryptanalysts go through all keys
  - Computations per time period
  - Storage capacity
  - Resistance time is mainly given by key length
- ▷ Provable security
  - The computational security can be demonstrated by comparing it with known hard problems

Key dimensions in perspective

- ▷ 2^{32} (4 Giga)
  - IPv4 address space
  - World population
  - Years for the Sun to become a white dwarf
- ▷ 2^{64}
  - Virtual address space of current CPU architectures
- ▷ 2^{128}
  - IPv6 address space
- ▷ 2^{166}
  - Earth atoms
- ▷ 2^{265}
  - Hydrogen atoms in the known universe
- ▷ 2^{1024} and beyond
  - Only cryptography uses them

Cryptanalysis attacks: approaches

- ▷ Brute force
  - Exhaustive search along the key space until finding a suitable key
  - Usually infeasible for a large key space
    - e.g. 2^{128} random keys (or keys with 128 bits)
    - Randomness is fundamental!
- ▷ Cleaver attacks
  - Reduce the search space to a smaller set of potential candidates

Cryptography: practical approaches (1/4)

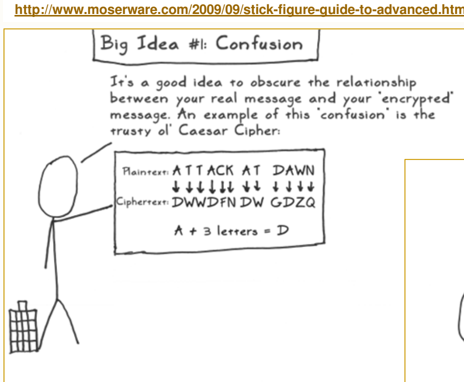
- ▷ Theoretical security vs. practical security
  - Expected use ≠ practical exploitation
  - Defective practices can introduce vulnerabilities
    - Example: reuse of keys
- ▷ Computational security
  - Computational complexity of break-in attacks
    - Using brute force
  - Security bounds:
    - Cost of cryptanalysis
    - Availability of cryptanalysis infra-structure
    - Lifetime of ciphertext

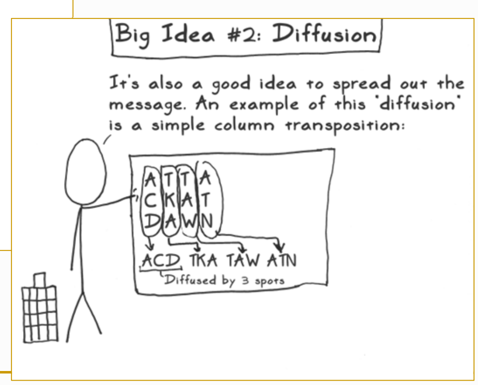
Cryptography: practical approaches (2/4)

- 5 Shannon criteria
  - The amount of offered secrecy
    - e.g. key length
  - Complexity of key selection
    - e.g. key generation, detection of weak keys
  - Implementation simplicity
  - Error propagation
    - Relevant in error-prone environments
    - e.g. noisy communication channels
  - Dimension of ciphertexts
    - Regarding the related plaintexts

Confusion & diffusion

http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html

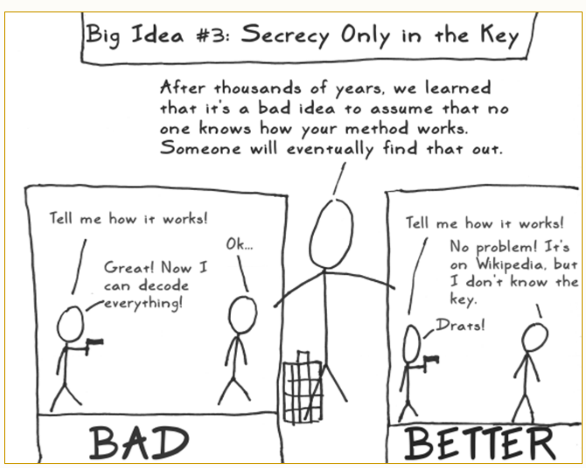




Cryptography: practical approaches (3/4)

- Confusion
  - Complex relationship between the key, plaintext and the ciphertext
    - Output bits (ciphertext) should depend on the input bits (plaintext + key) in a very complex way
- Diffusion
  - Plaintext statistics are dissipated in the ciphertext
    - If one plaintext bit toggles, then the ciphertext changes substantially, in an unpredictable or pseudorandom manner
  - Avalanche effect

What should be secret?



http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html

Cryptography: practical approaches (4/4)

- Always assume the worst case
  - Cryptanalysts know the algorithm
    - Security lies in the key
  - Cryptanalysts know/have many ciphertext samples produced with the same algorithm & key
    - Ciphertext is not secret!
  - Cryptanalysts partially know original plaintexts
    - As they have some idea of what they are looking for
    - Know-plaintext attacks
    - Chosen-plaintext attacks

Cryptographic robustness

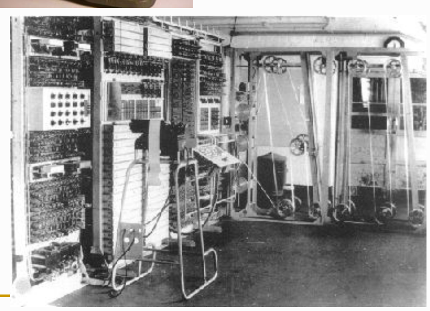
- The robustness of algorithms is their resistance to attacks
  - No one can evaluate it precisely
    - Only speculate or demonstrate using some other robustness assumptions
  - They are robust until someone breaks them
  - There are public guidelines with what should/must not be used
    - Sometimes antecipating future problems
- Algorithms with longer keys are probably stronger
  - And usually slower …
- Public algorithms w/o known attacks are probably stronger
  - More people looking for weaknesses

Cryptographic guidelines

- Guideline for Using Cryptographic Standards in the Federal Government: Cryptographic Mechanisms, NIST Special Publication 800-175B Rev. 1, July 2019
- Cryptographic Storage Cheat Sheet, OWASP Cheat Sheets (last revision: 6/Jun/2020)
- Guidelines on cryptographic algorithms usage and key management, European Payments Council, EPC342-08 v9.0, 9/Mar/2020
- Algorithms, Key Size and Protocols Report, ECRYPT – Coordination & Support Action, Deliverable D5.4, H2020-ICT-2014 Project 645421, 28/Feb/2018

Ciphers: evolution of technology

- Manual
  - Simple transposition or substitution algorithms
- Mechanic
  - From XIX cent.
    - Enigma machine
    - M-209 Converter
  - More complex substitution algorithms
- Informatics
  - Appear with computers
  - Highly complex substitution algorithms
  - Mathematical algorithms



Ciphers: basic types (1/3)

- Transposition
  - Original cleartext is scrambled
    Onexcl raatre ilriad gctsm ilesb
  - Block permutations
    (13524) → **boklc pruem ttoai ns**

O	N	E	X	C	L
R	A	A	T	R	E
I	L	R	I	A	D
G	C	T	S	M	
I	L	E	S	B	

- Substitution
  - Each original symbol is replaced by another
    - Original symbols were letters, digits and punctuation
    - Actually they are blocks of bits
  - Substitution strategies
    - Mono-alphabetic (one→one)
    - Polyalphabetic (many one→one)
    - Homophonic (one→many)

Ciphers: basic types (2/3): Mono-alphabetic

- Use a single substitution alphabet
  - With #α elements
- Examples
  - Additive (translation)
    - crypto-symbol = (symbol + key) mod # α
    - symbol = (crypto-symbol – key) mod # α
    - Possible keys = #α
    - Caesar Cipher (ROT-x)
  - With sentence key
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    QRUVWXZSENTCKYABDFGHIJLMOP
    - Possible keys = # α ! → 26! ≈ 2^{88}
- Problems
  - Reproduce plaintext pattern
    - Individual characters, digrams, trigrams, etc.
  - Statistical analysis facilitates cryptanalysis
    - "The Gold Bug", Edgar Alan Poe

```
53‡‡†305))6*;4826)4‡.)
4‡);806*;48†860))85;1‡
(;:‡*8†83(88)5*†;46(;8
8*96*?;8)*‡(;485);5*†2
:*‡(;4956*2(5*—4)88*;4
069285);)6†8)4‡‡;1(‡9;
48081;8:8‡1;48†85;4)48
5†528806*81(‡9;48;(88;
4(‡?34;48)4‡;161;:188;
‡?;
```

```
A good glass in the
bishop's hostel in the
devil's seat fifty-one
degrees and thirteen
minutes northeast and
by north main branch
seventh limb east side
shoot from the left eye
of the death's-head a
bee line from the tree
through the shot forty
feet out
```

Ciphers: basic types (3/3): Polyalphabetic

- ▷ Use N substitution alphabets
  - Periodical ciphers, with period N
- ▷ Example
  - Vigenère cipher
- ▷ Problems
  - Once known the period, are as easy to cryptanalyze as N mono-alphabetic ones
    - The period can be discovered using statistics
    - Kasiski method
      - Factoring of distances between equal ciphertext blocks
    - Coincidence index
      - Factoring of self-correlation offsets that yield higher coincidences

Vigenère cipher (or the Vigenère square)

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
b	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
c	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
d	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
e	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
f	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
g	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
h	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
i	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
j	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
k	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
l	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
m	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
n	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
o	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
p	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
r	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
s	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
t	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
u	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
v	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
w	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
x	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

- ▷ Example of encryption of character M with key S, yielding cryptogram E
  - Decryption is the opposite, E and S yield M

Cryptanalysis of a Vigenère cryptogram: Example (1/2)

- Plaintext:
  Eles não sabem que o sonho é uma constante da vida
  tão concreta e definida como outra coisa qualquer,
  como esta pedra cinzenta em que me sento e descanso,
  como este ribeiro manso, em serenos sobressaltos
  como estes pinheiros altos
- Cipher with the Vigenère square and key "poema"

```
plaintext  elesnaosabemqueosonhoeumaconstantedavidataoconcretaedefinida
key        poemapoemapoemapoemapoemapoemapoemapoemapoemapoemapoemapoema
cryptogram tzienpcwmbtaugedgszhdsyyarcretpbxqdpjmpaiosoocqvqtpshqfxbmpa
```

- Kasiski test
  - With text above:

mpa	$20 = 2 \times 2 \times 5$
tp	$20 = 2 \times 2 \times 5$

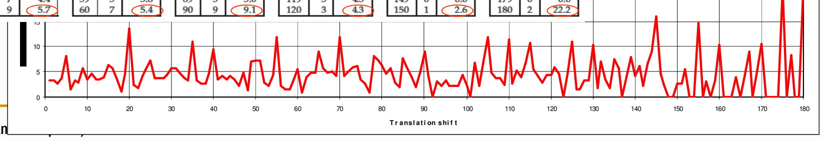
  - With the complete poem:

$175 = 5 \times 5 \times 7$	1
$105 = 3 \times 5 \times 7$	3
$35 = 5 \times 7$	1
$20 = 2 \times 2 \times 5$	4

Cryptanalysis of a Vigenère cryptogram: Example (2/2)

- Coincidence index (with full poem)

D	I	P (%)	D	I	P (%)	D	I	P (%)	D	I	P (%)	D	I	P (%)	D	I	P (%)
1	6	3.2	31	9	5.7	61	1	0.8	91	4	4.1	121	4	5.9	151	1	2.6
2	6	3.2	32	7	4.5	62	5	3.9	92	0	0.0	122	3	4.5	152	2	5.4
3	5	2.7	33	6	3.8	63	6	4.8	93	3	3.1	123	0	0.0	153	0	0.0
4	7	3.8	34	5	3.2	64	6	4.8	94	2	2.1	124	3	4.6	154	0	0.0
5	15	8.2	35	17	11.0	65	11	8.9	95	3	3.2	125	7	10.9	155	5	14.7
6	3	1.6	36	5	3.3	66	7	5.7	96	2	2.2	126	1	1.6	156	0	0.0
7	6	3.3	37	4	2.6	67	6	4.9	97	2	2.2	127	1	1.6	157	1	3.1
8	5	2.8	38	4	2.6	68	6	5.0	98	2	2.2	128	2	3.3	158	0	0.0
9	10	5.6	39	7	4.7	69	5	4.2	99	4	4.4	129	2	3.3	159	1	3.3
10	6	3.4	40	14	9.4	70	14	11.8	100	2	2.2	130	6	10.2	160	3	10.3
11	8	4.5	41	5	3.4	71	5	4.2	101	0	0.0	131	1	1.7	161	0	0.0
12	6	3.4	42	6	4.1	72	6	5.1	102	6	6.9	132	4	7.0	162	0	0.0
13	6	3.4	43	5	3.4	73	7	6.0	103	2	2.3	133	2	3.6	163	0	0.0
14	7	4.0	44	6	4.1	74	7	6.1	104	6	7.1	134	1	1.8	164	1	4.0
15	11	6.3	45	5	3.5	75	4	3.5	105	10	11.9	135	4	7.4	165	0	0.0
16	10	5.8	46	3	2.1	76	3	2.7	106	4	4.8	136	3	5.7	166	1	4.3
17	6	3.5	47	7	4.9	77	1	0.9	107	3	3.7	137	0	0.0	167	2	9.1
18	2	1.2	48	2	1.4	78	9	8.1	108	3	3.7	138	2	3.9	168	0	0.0
19	8	4.7	49	10	7.1	79	8	7.3	109	2	2.5	139	4	8.0	169	1	5.0
20	23	13.6	50	10	7.2	80	7	6.4	110	9	11.4	140	2	4.1	170	2	10.5
21	4	2.4	51	10	7.2	81	5	4.6	111	2	2.6	141	3	6.2	171	0	0.0
22	3	1.8	52	4	2.9	82	6	5.6	112	4	5.2	142	1	2.1	172	0	0.0
23	7	4.2	53	3	2.2	83	3	2.8	113	3	3.9	143	3	6.5	173	0	0.0
24	9	5.5	54	6	4.4	84	2	1.9	114	5	6.7	144	4	8.9	174	0	0.0
25	12	7.3	55	16	11.9	85	8	7.7	115	8	10.8	145	7	15.9	175	3	21.4
26	6	3.7	56	3	2.3	86	6	5.8	116	4	5.5	146	2	4.7	176	0	0.0
27	6	3.7	57	2	1.5	87	4	3.9	117	3	4.2	147	1	2.4	177	1	8.3
28	6	3.7	58	2	1.5	88	2	2.0	118	2	2.8	148	0	0.0	178	0	0.0
29	7	4.4	59	5	3.8	89	5	5.0	119	3	4.3	149	0	0.0	179	0	0.0
30	9	5.7	60	7	5.4	90	9	9.1	120	3	4.3	150	1	2.6	180	2	22.2

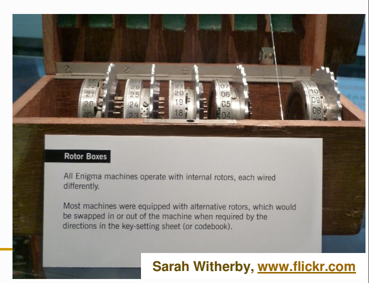


Rotor Machines

David J Morgan, www.flickr.com

Rotor machines

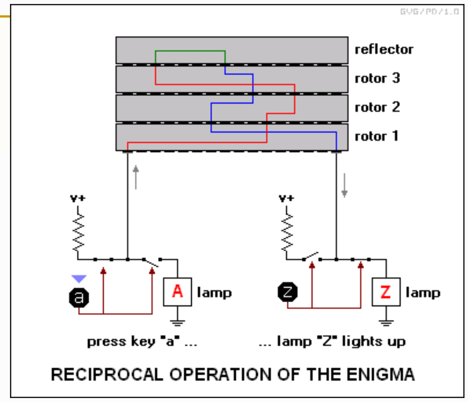
- Rotor machines implement complex polyalphabetic ciphers
  - Each rotor contains a permutation
    - Same as a set of substitutions
  - The position of a rotor implements a substitution alphabet
  - Spinning of a rotor implements a polyalphabetic cipher
  - Stacking several rotors and spinning them at different times adds complexity to the cipher
- The cipher key is:
  - The set of rotors used
  - The relative order of the rotors
  - The position of the spinning ring
  - The original position of all the rotors
- Symmetrical (two-way) rotors allow decryption by “double encryption”
  - Using a reflection disk (half-rotor)



Sarah Witherby, www.flickr.com

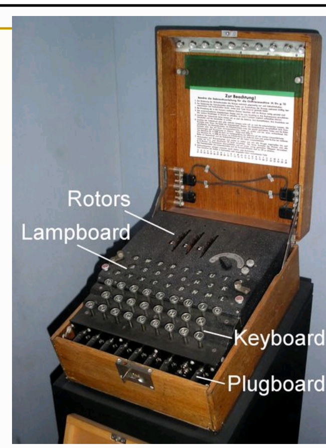
Rotor machines

Andrew Magill, www.flickr.com

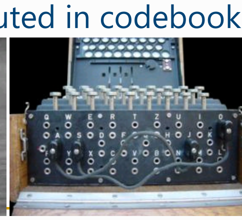


- ▷ Reciprocal operation with reflector
  - Sending operator types "A" as plaintext and gets "Z" as ciphertext, which is transmitted
  - Receiving operator types the received "Z" and gets the plaintext "A"
  - No letter could encrypt to itself !

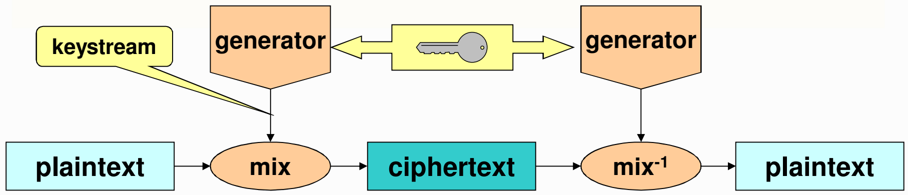
Enigma



- ▷ WWII German rotor machine
  - Many models used
- ▷ Initially presented in 1919
  - Enigma I, with 3 rotors
- ▷ Several variants where used
  - With different number of rotors
  - With patch cord to permute alphabets
- ▷ Key settings distributed in codebooks



Stream ciphers

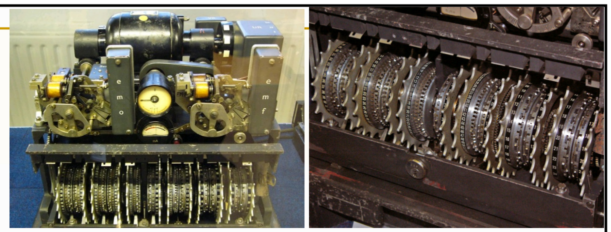


- ▷ Mixture of a keystream with the plaintext or ciphertext
  - Random keystream (Vernam's one-time pad)
  - Pseudo-random keystream (produced by generator using a finite key)
- ▷ Reversible mixture function
  - e.g. bitwise XOR
  - $C = P \oplus ks$ $\qquad$ $P = C \oplus ks$
- ▷ Polyalphabetic cipher
  - Each keystream symbol defines an alphabet

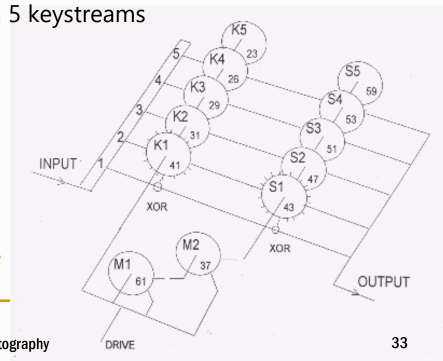
Stream ciphers

- ▷ Keystream may be infinite but with a finite period
  - The period depends on the generator
- ▷ Practical security issues
  - Each **keystream** should be used only **once**!
    - Otherwise, the sum of cryptograms yields the sum of plaintexts
      $C1 = P1 \oplus Ks$, $C2 = P2 \oplus Ks$ $\rightarrow$ $C1 \oplus C2 = P1 \oplus P2$
  - **Plaintext length** should be **smaller** than the **keystream period**
    - Total keystream exposure under know/chosen plaintext attacks
    - Keystream cycles help the cryptanalysts knowing plaintext samples
  - Integrity control is mandatory
    - No diffusion! (only confusion)
    - Ciphertexts can easily be changed deterministically

Lorenz (Tunny)

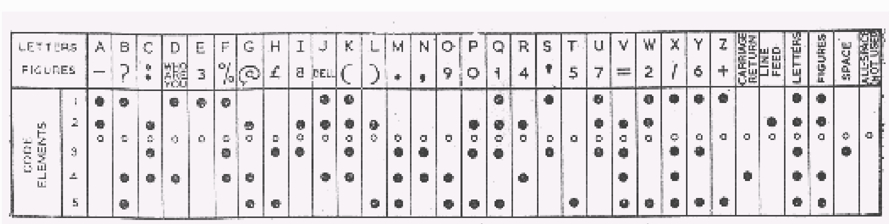


- ▷ 12-Rotor stream cipher
  - Used by the German high-command during the 2nd WW
  - Implements a stream cipher
    - Each 5-bit character is mixed with 5 keystreams
- ▷ Operation
  - 5 regularly stepped (χ) wheels
  - 5 irregularly stepped (ψ) wheels
    - All or none stepping
  - 2 motor wheels
    - For stepping the ψ wheels
  - Number of steps in all wheels is relatively prime



Cryptanalysis of Tunny in Bletchley Park

- ▷ They didn't know Lorenz internal structure
  - They observed one only at the end of the war
  - They knew about them because they could get 5-bit encrypted transmissions
    - Using the 32-symbol Baudot code instead of Morse code



Cryptanalysis of Tunny in Bletchley Park: The mistake (30 August 1941)

- A German operator had a long message (~4,000) to send
  - He set up his Lorenz and sent a 12 letter indicator (wheel setup) to the receiver
  - After ~4,000 characters had been keyed, by hand, the receiver said "send it again“
- The operator resets the machine to the same initial setup
  - Same keystream! Absolutely forbidden!
- The sender began to key in the message again (by hand)
  - But he typed a slightly different message!

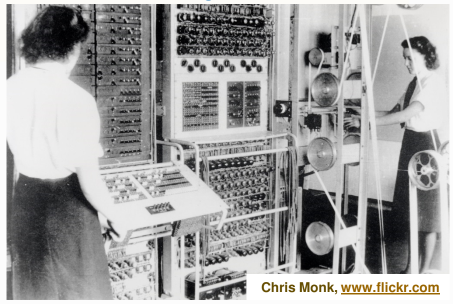
  $C = M \oplus Ks$
  $C' = M' \oplus Ks \rightarrow M' = C \oplus C' \oplus M \rightarrow$ text variations

  - Know parts of the initial text M reveal the variations, M’

Cryptanalysis of Tunny in Bletchley Park: Breakthrough

- Messages began with SPRUCHNUMMER — “msg number"
  - The first time the operator typed S P R U C H N U M M E R
  - The second time he typed S P R U C H N R
  - Thus, immediately following the N the two texts were different!
- John Tiltman at Bletchley Park was able to fully decrypt both messages (called *Depths*) using an additive combination of them
  - The 2nd message was ~500 characters shorter than the first one
  - Tiltman managed to discover the correct message for the 1st ciphertext
- They got for the 1st time a long stretch of the Lorenz keystream
  - They did not know how the machine did it, …
  - … but they knew that this was what it was generating!

Cryptanalysis of Tunny in Bletchley Park: Colossus



Chris Monk, www.flickr.com

- The cipher structure was determined from the keystream
  - But deciphering it required knowing the initial position of rotors
- Germans started using numbers for the initial wheels' state
  - Bill Tutte invented the double-delta method for finding that state
  - The Colossus was built to apply the double-delta method
- Colossus
  - Design started in March 1943
  - The 1,500 valve Colossus Mark 1 was operational in January 1944
  - Colossus reduced the time to break Lorenz from weeks to hours

Modern ciphers: types

- Concerning operation
  - Block ciphers (mono-alphabetic)
  - Stream ciphers (polyalphabetic)
- Concerning their key
  - Symmetric ciphers (secret key or shared key ciphers)
  - Asymmetric ciphers (or public key ciphers)
- Arrangements

	Block ciphers	Stream ciphers
Symmetric ciphers		
Asymmetric ciphers		

Symmetric ciphers

- Secret key
  - Shared by 2 or more peers
- Allow
  - Confidentiality among the key holders
  - Limited authentication of messages
    - When block ciphers are used
- Advantages
  - Performance (usually very efficient)
- Disadvantages
  - N interacting peers, pairwise secrecy ⇒ N x (N-1)/2 keys
- Problems
  - Key distribution

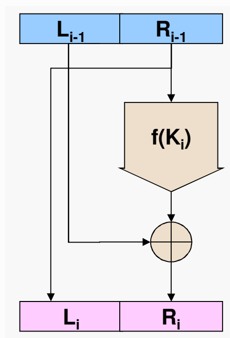
Symmetric block ciphers

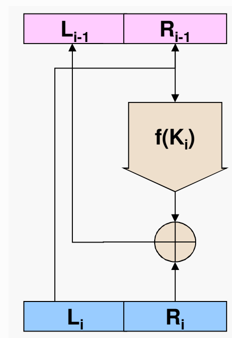
- Usual approaches
  - Large bit blocks for input, output and key
    - 64, 128, 256, etc.
  - Diffusion & confusion
    - Permutation, substitution, expansion, compression
    - Feistel networks, substitution-permutation networks
    - Iterations
    - Sub-keys (key schedules, round keys, etc.)
- Most common algorithms
  - DES (Data Enc. Stand.), D=64 K=56
  - IDEA (Int. Data Enc. Alg.), D=64 K=128
  - AES (Adv. Enc. Stand., aka Rijndael) D=128 K=128, 192, 256
  - Other (Blowfish, CAST, RC5, etc.)

Feistel networks

$L_i = R_{i-1}$
$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$

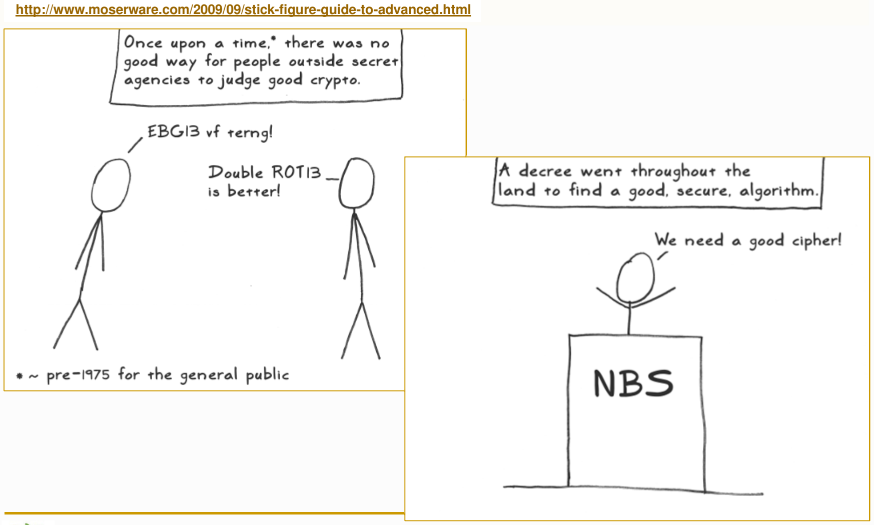
$R_{i-1} = L_i$
$L_{i-1} = R_i \oplus f(L_i, K_i)$





We need a good cipher!

http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html

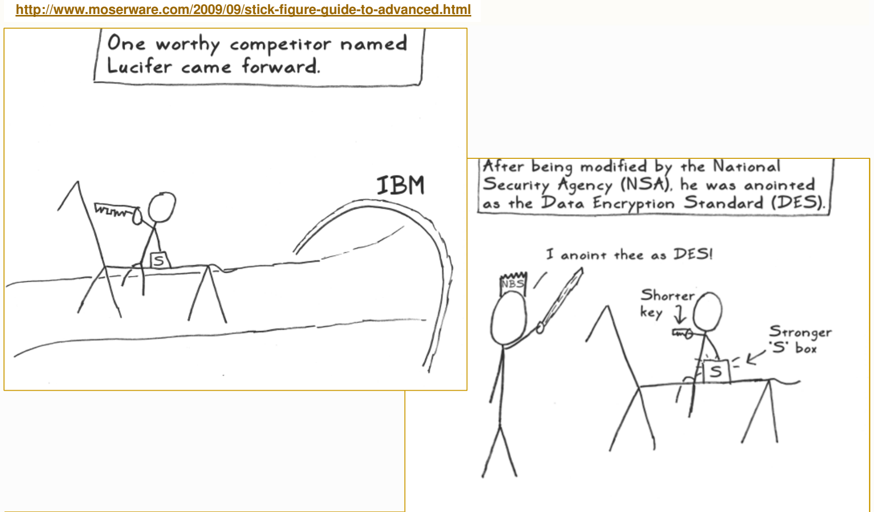


DES (Data Encryption Standard)

- 1970: the need of a standard cipher for civilians was identified
- 1972: NBS opens a contest for a new cipher, requiring:
  - The cryptographic algorithm must be secure to a high degree
  - Algorithm details described in an easy-to-understand language
  - The details of the algorithm must be publicly available
    - So that anyone could implement it in software or hardware
  - The security of the algorithm must depend on the key
    - Not on keeping the method itself (or part of it) secret
  - The method must be adaptable for use in many applications
  - Hardware implementations of the algorithm must be practical
    - i.e. not prohibitively expensive or extremely slow
  - The method must be efficient
  - Test and validation under real-life conditions
  - The algorithm should be exportable

Lucipher and DES

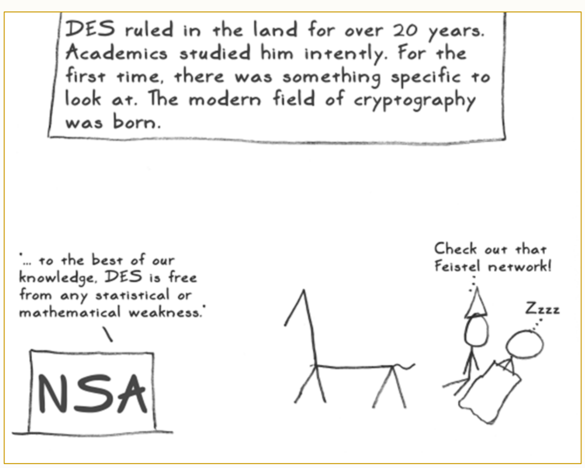
http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html



DES: proposal and adoption

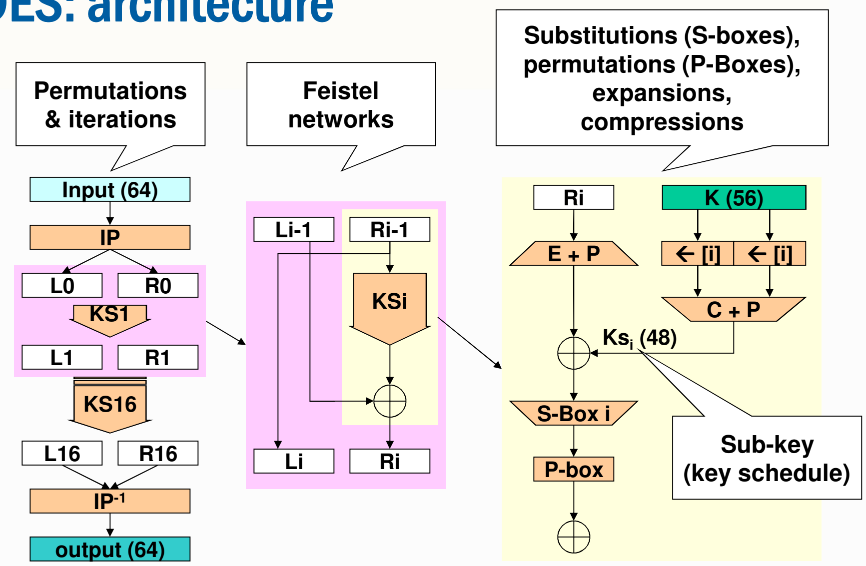
- 1974: new contest
  - Proposal based on Lucifer from IBM
  - 64-bit blocks
  - 56-bit keys
    - 48-bit subkeys (key schedules)
  - Diffusion & confusion
    - Feistel networks
    - Permutations, substitutions, expansions, compressions
    - 16 iterations
  - Several modes of operation
    - **ECB** (Electronic Code Book), **CBC** (Cypher Block Chaining)
    - **OFB** (Output Feedback), **CFB** (Cypher Feedback)
- 1976: adopted at US as a federal standard

DES as a milestone



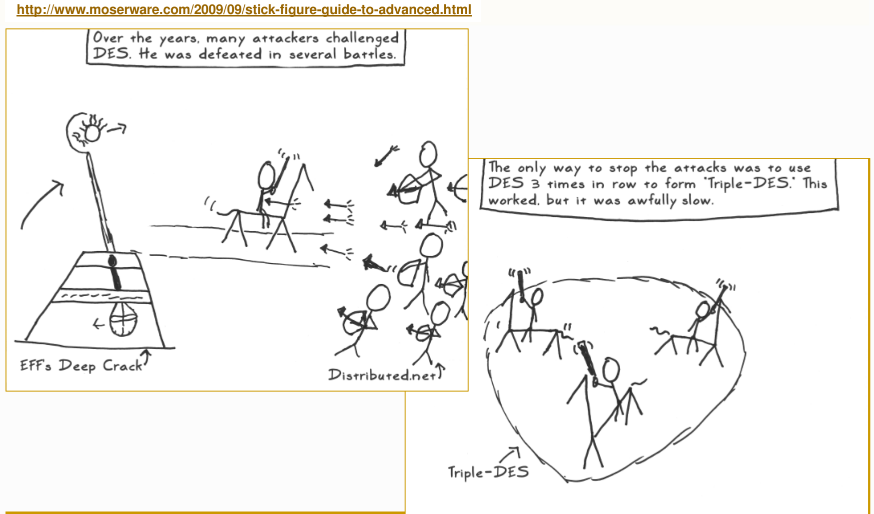
http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html

DES: architecture



DES security

http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html

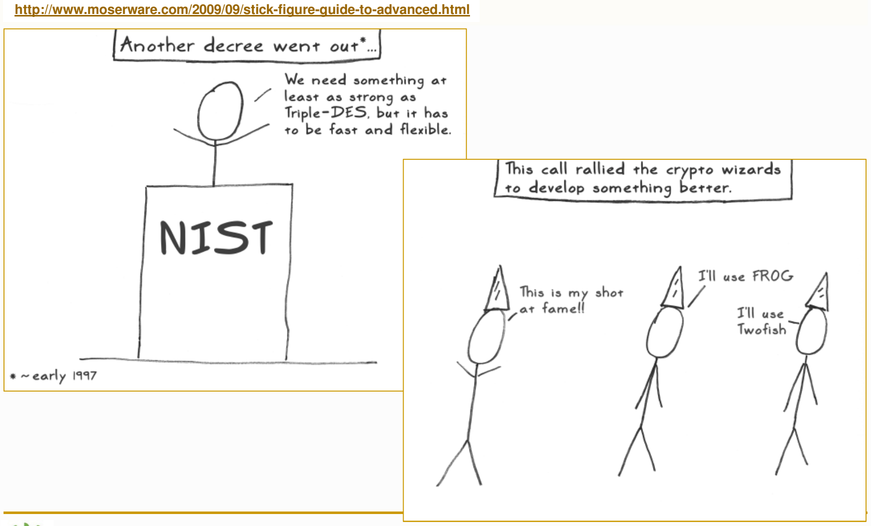


DES: offered security

- Key selection
  - Most 56-bit values are suitable
  - 4 weak, 12 semi-weak keys, 48 possibly weak keys
    - Equal key schedules (1, 2 or 4)
    - Easy to spot and avoid
- Known attacks
  - Exhaustive key space search
- Key length
  - 56 bits are actually too few
  - Exhaustive search is technically possible and economically interesting
- Multiple encryption
  - Double encryption
    - Theoretically not more secure
  - Triple DES (3DES)
    - With 2 or 3 keys
    - Equivalent key length of 112 or 168 bits
    - Slow ...but secure!

Replacement of DES (and DES variants)

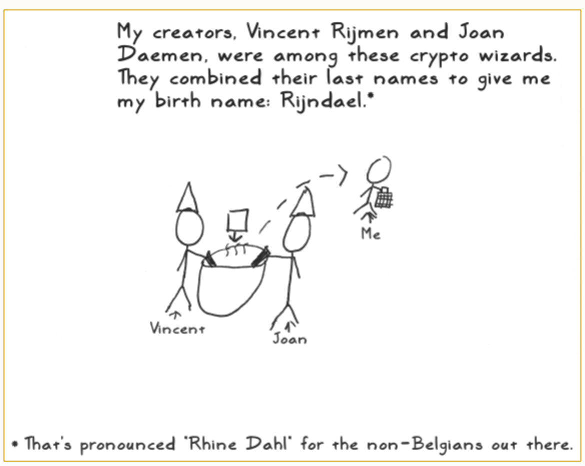
http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html



AES (Advanced Encryption Standard)

- 2/Jan/1997: Call for evaluation criteria
  - NIST publicly asked interested parties to propose a criteria to choose a DES successor
  - Many submissions received during 3 months
- 12/Sep/1997: Call for new algorithms
  - Block ciphers
  - 128-bit blocks
  - 128, 192, and 256-bit keys
  - Such ciphers were rare at the time of the call

Rijndael



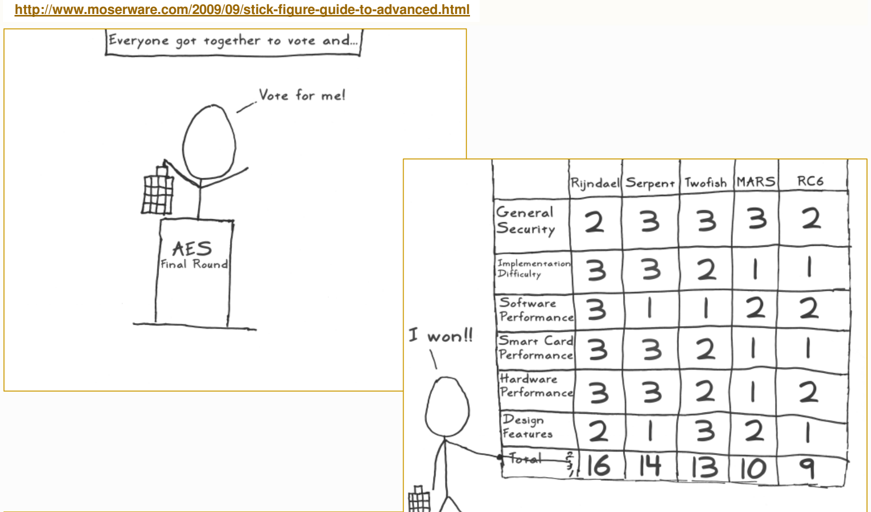
http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html

AES: evaluation rounds

- 1st round
  - 15 candidate algorithms were evaluated by the community
  - Conferences were organized for the evaluation
  - Cryptographic weakness were found
  - Performance issues were identified
    - In a variety of hardware
    - PCs, smart cards, hardware implementations
  - Constrained environment were evaluated
    - Limited memory smart cards, low gate count circuits, FPGAs
- Aug/1999: AES finalists announced
  - MARS, RC6, Rijndael, Serpent, and Twofish

Rijndael selection as AES

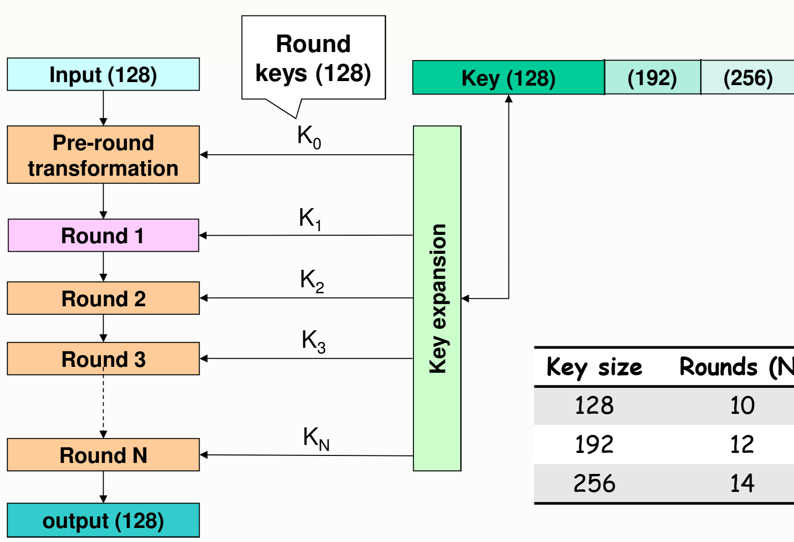
http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html



AES: evaluation rounds

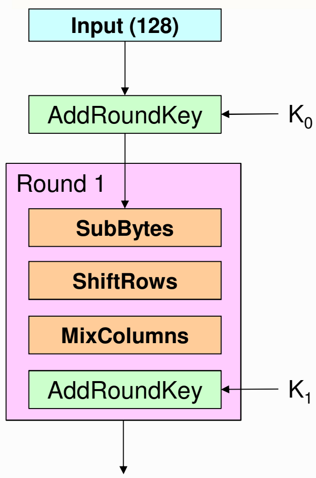
- 2^{nd} round
  - The 5 finalists continued to be evaluated
  - In a final conference the proposal of each algorithm presented their advantage against the other
- 2/Oct/2000: AES algorithm was announced
  - Rijndael was selected
  - Proposed by Vincent Rijmen and Joan Daemen
  - Family of ciphers with different key and block sizes
- 26/Nov/2001: AES was approved by NIST
  - FIPS PUB 197
  - Subset of Rijndael (3 family members)
- Now part of the ISO/IEC 18033-3 standard

AES: architecture

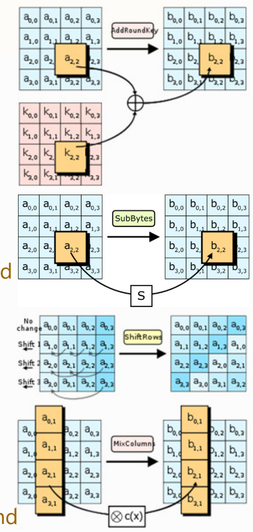


Key size	Rounds (N)
128	10
192	12
256	14

AES: architecture



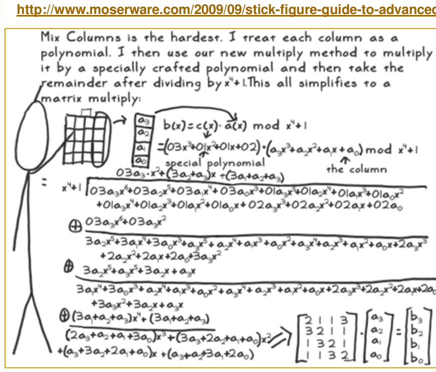
- AddRoundKey:
  - 128-bit XOR
  - Output is a 4x4 byte matrix
- SubBytes:
  - 256-element S-box
  - Each matrix bytes is substituted
- ShiftRows
  - Rows are rotated left
  - Byte shifts vary (0, 1, 2 & 3)
- MixColumns
  - Each column is transformed
  - Not performed in the last round

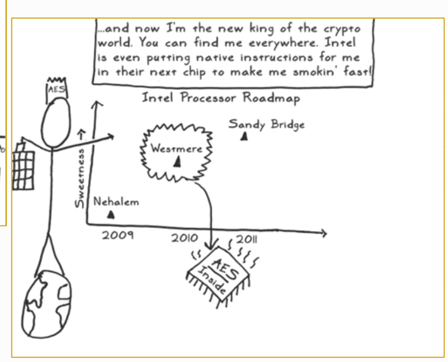


https://aescryptography.blogspot.com

AES complexity and speed-up

http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html





AES in CPU instruction's set

- Intel AES New Instructions (AES-NI)

AESENC	**Perform one round of an AES encryption flow**
AESENCLAST	Perform the last round of an AES encryption flow
AESDEC	Perform one round of an AES decryption flow
AESDECLAST	Perform the last round of an AES decryption flow
AESKEYGENASSIST	Assist in AES round key generation
AESIMC	Assist in AES Inverse Mix Columns

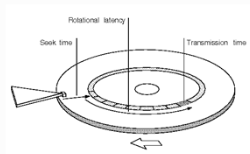
- ARMv8 Cryptographic Extension
- … and other

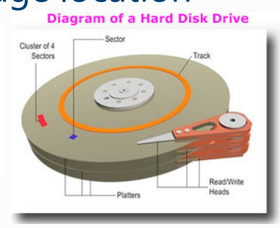
Stream ciphers

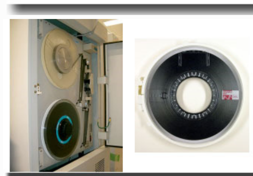
- Approaches
  - Cryptographically secure pseudo-random generators (PRNG)
    - Using linear feedback shift registers (LFSR)
    - Using block ciphers
    - Other (families of functions, etc.)
  - Usually not self-synchronized
  - Usually without uniform random access
    - No immediate setup of generator's state for a given plaintext/cryptogram offset
- Most common algorithms
  - A5/1 (US, Europe), A5/2 (GSM)
  - RC4 (802.11 WEP/TKIP, etc.)
  - E0 (Bluetooth BR/EDR)
  - SEAL (w/ uniform random access)

Uniform random access

- Same time to reach and process any piece of information regardless of its storage location
- Uniform
  - Memory
  - Disks (magnetic, optical)
    - Average $T_{access} = T_{seek} + ½ \, T_{revolution}$
- Non-uniform
  - Tapes (audio, video, computer)

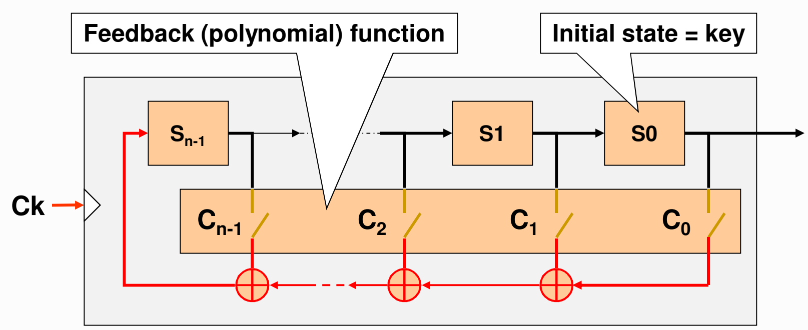






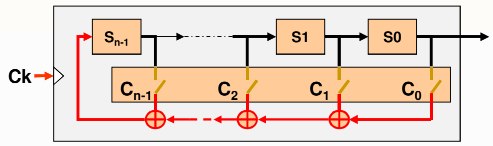
http://osr507doc.sco.com/en/PERFORM/disk_IO_mech.html
https://www.ict4u.net/components/backing-storage.php

Linear Feedback Shift Register (LFSR)



- 2^n-1 non-null sequences
  - If one of them has a 2^n-1 period length, then all have it
- Primitive feedback functions (primitive polynomials)
  - All non-null sequences have a 2^n–1 period length

Linear Feedback Shift Register (LFSR)



▷ Issue

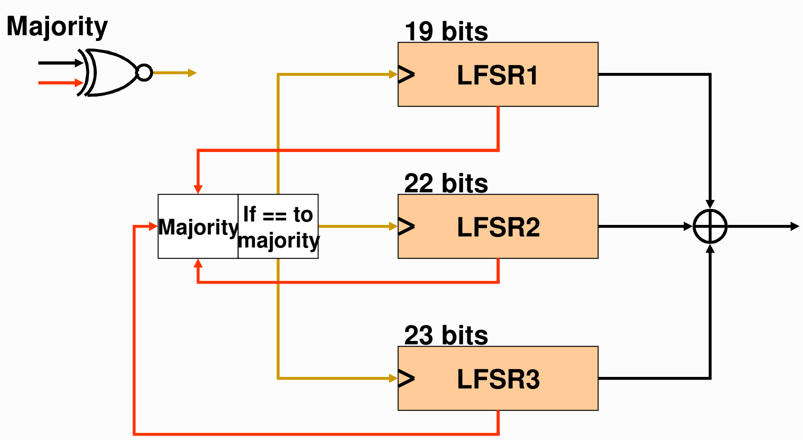
- If you know N consecutive bits of the output, you know the entire sequence ahead

O_0	O_1	O_2	O_3		O_n

$O_n = C_0O_0 + C_1O_1 + \ldots + C_{n-1}O_{n-1}$

- The output must be mixed with something else …

Generators using many LFSR: A5/1 (GSM)



Deployment of (symmetric) block ciphers: Cipher modes

- ▷ Initially proposed for DES
  - ECB (Electronic Code Book)
  - CBC (Cipher Block Chaining)
  - OFB (Output Feeback)
  - CFB (Cipher Feedback)
- ▷ Can be used with other block ciphers
  - In principle …
- ▷ Some other modes do exist
  - CTR (Counter Mode)
  - GCM (Galois/Counter Mode)

Block cipher modes: ECB and CBC

Electronic Code Book

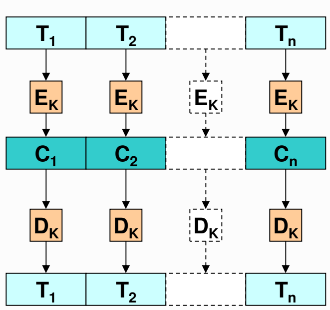
$C_i = E_K(T_i)$

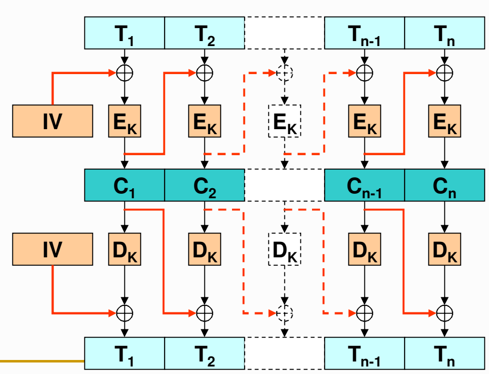
$T_i = D_K(C_i)$

Cipher Block Chaining

$C_i = E_K(T_i \oplus C_{i-1})$

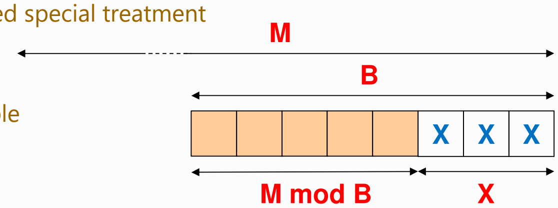
$T_i = D_K(C_i) \oplus C_{i-1}$





ECB/CBC cipher modes: Block alignment with padding

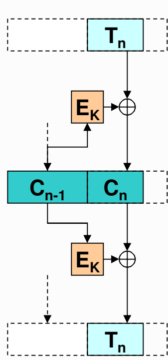
- ▷ Block cipher modes ECB and CBC require block-aligned inputs
  - Trailing sub-blocks need special treatment
- ▷ Alternative 1: padding
  - Of last block, identifiable
  - Adds data
  - PKCS #7
    - X = B – (M mod B)
    - X extra bytes, with the value X
    - PKCS #5 (same as PKCS #7 with B = 8)



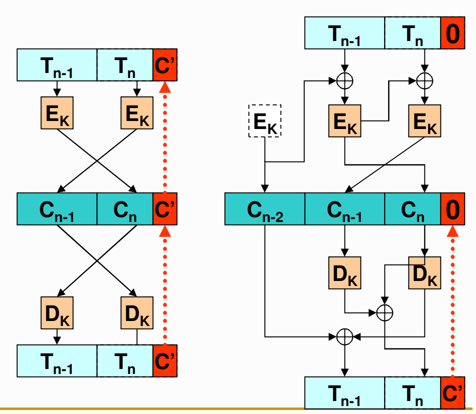
- ▷ Alternative 2: different processing for the last block
  - Adds implementation complexity

ECB/CBC cipher modes: Handling trailing sub-blocks

▷ Sort of stream cipher



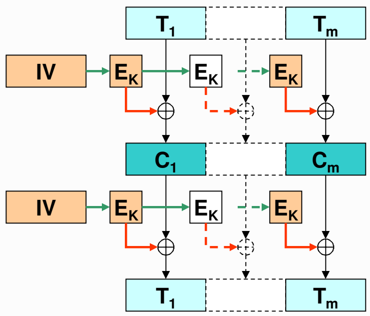
▷ Ciphertext stealing

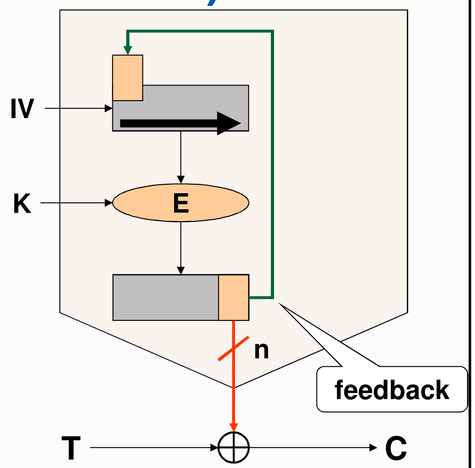


Stream cipher modes: n-bit OFB (Output Feedback)

$C_i = T_i \oplus E_K(S_i)$
$T_i = C_i \oplus E_K(S_i)$

$S_i = f(S_{i-1}, E_K(S_{i-1}))$
$S_0 = IV$

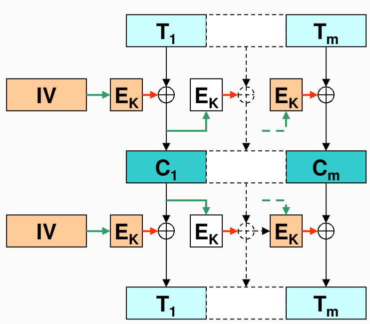


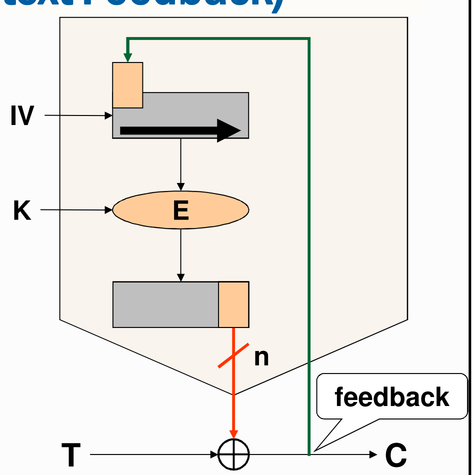


Stream cipher modes: n-bit CFB (Ciphertext Feedback)

$C_i = T_i \oplus E_K(S_i)$
$T_i = C_i \oplus E_K(S_i)$

$S_i = f(S_{i-1}, C_i)$
$S_0 = IV$

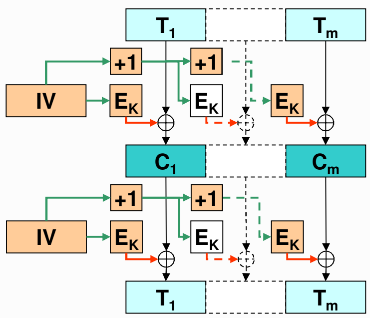


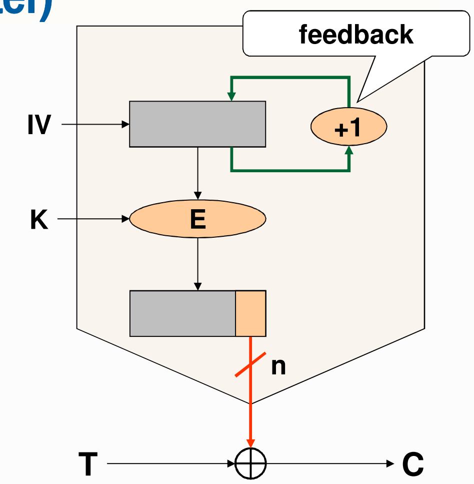


Stream cipher modes:
n-bit CTR (Counter)

$C_i = T_i \oplus E_K(S_i)$
$T_i = C_i \oplus E_K(S_i)$

$S_i = S_{i-1}+1$
$S_0 = IV$



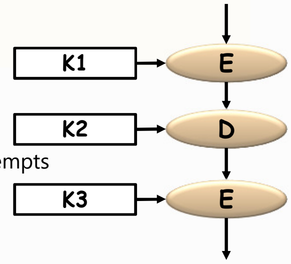


Cipher modes:
Pros and cons

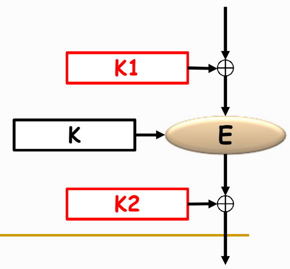
	Block		Stream		
	ECB	CBC	OFB	CFB	CTR
Input pattern hiding		✓	✓	✓	✓
Confusion on the cipher input		✓		✓	Secret counter
Same key for different messages	✓	✓	other IV	other IV	other IV
Tampering difficulty	✓	✓ (...)		✓	
Pre-processing			✓	...	✓
Parallel processing Uniform random access	✓	Decryption Only	w/ pre-processin g	Decryption only	✓
Error propagation	Same block	Same block Next block		Some bits afterward s	
Capacity to recover from losses	Block Losses	Block Losses		✓	

Cipher modes:

Security reinforcement



- Multiple encryption
  - Double encryption
    - Breakable with a meet-in-the-meddle attack in 2^{n+1} attempts
      - With 2 or more known plaintext blocks
      - Using 2^n blocks stored in memory …
    - Not secure enough (theoretically)
  - Triple encryption (EDE)
    - $\mathbf{C_i = E_{K3}(D_{K2}(E_{K1}(T_i)))}$ $\qquad$ $\mathbf{P_i = D_{K1}(E_{K2}(D_{K3}(C_i)))}$
    - Usually $\mathbf{K_1=K_3}$
    - If $\mathbf{K_1=K_2=K_3}$, then we get simple encryption



- Key whitening (DESX or DES-X)
  - Simple and efficient technique to add confusion
  - $\mathbf{C_i = E_K(K_1 \oplus T_i) \oplus K_2}$
  - $\mathbf{T_i = K_1 \oplus D_K(K_2 \oplus C_i)}$