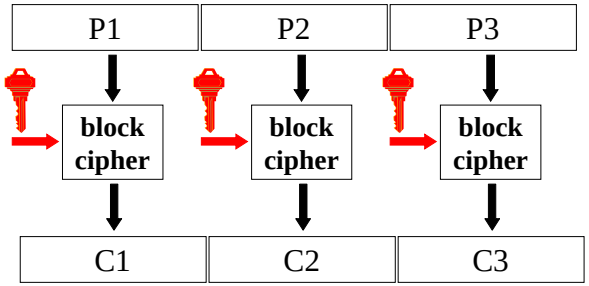
Crypto primitives

Today's talk includes slides from:
Bart Preneel and Dan Wallach

Comp527 status items

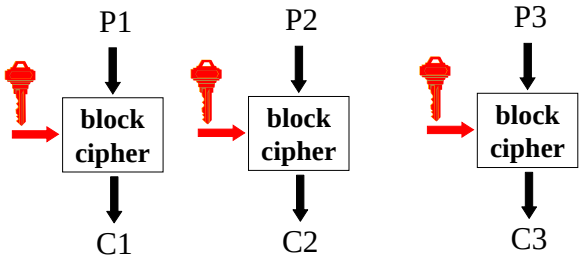
- Install the smart card software!
  - Dan out of town until Thursday
  - Bring software back to Dennis Lu (DH3057)
- Today: more on crypto primitives
- Wednesday: Eric Allen on *cryptyc*

Block cipher

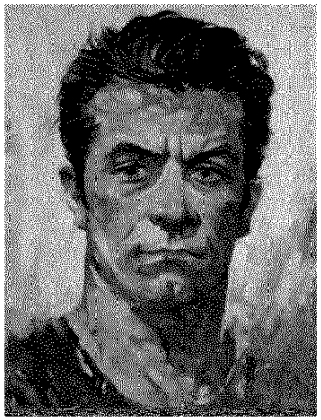


- larger data units: 64…128 bits
- memoryless
- repeat simple operation (round) many times

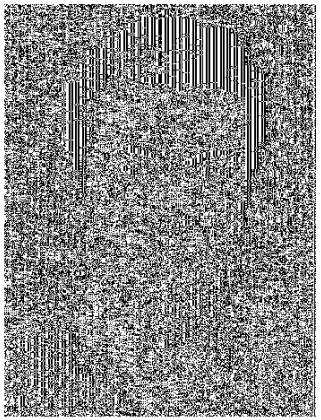
How NOT to use a block cipher: ECB mode



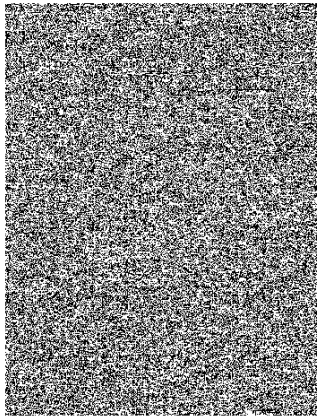
An example plaintext



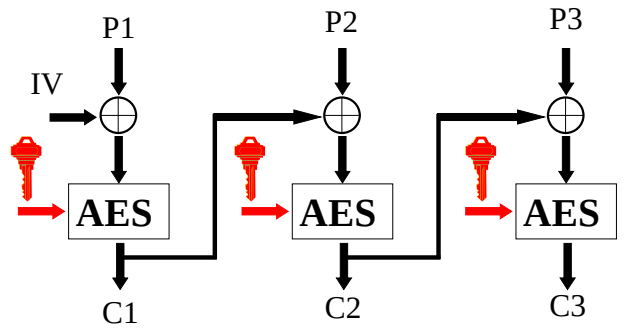
Encrypted with AES in ECB mode



Encrypted with AES in CBC mode

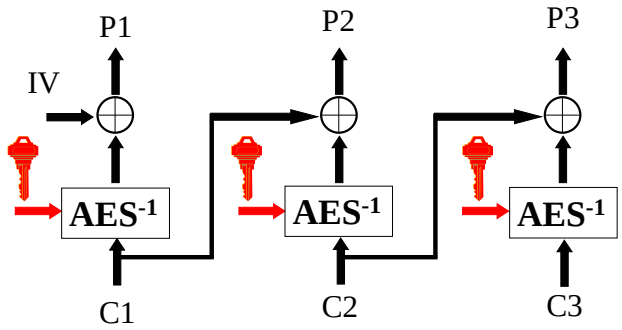


How to use a block cipher: CBC mode



need random IV

CBC mode decryption



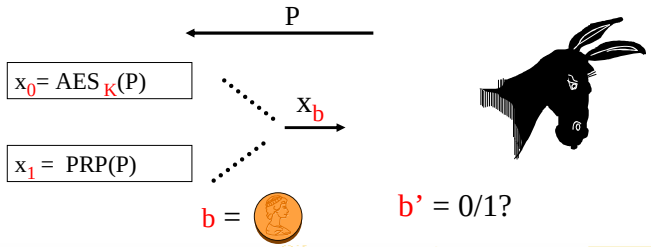
Secure encryption

- What is a secure block cipher anyway?
- What is secure encryption anyway?
- Definition of security
  - security assumption
  - security goal
  - capability of opponent

Security assumption:
the block cipher is a pseudo-random permutation

- It is hard to distinguish a block cipher from a random permutation
- Advantage of a distinguisher

$Adv_{AES/PRP} = Pr[b'=1|b=1] - Pr[b'=1|b=0]$



Security goal: "encryption"

- semantic security: adversary with limited computing power cannot gain any extra information on the plaintext by observing the ciphertext
- indistinguishability (real or random) [IND-ROR]: adversary with limited computing power cannot distinguish the encryption of a plaintext P from a random string of the same length
- IND-ROR ⇒ semantic security
  More on this in Comp527, later this month

Cryptanalysis of block ciphers

- exhaustive key search (key of *k* bits)
  - 2^k encryptions, *k/n* known plaintexts
- code book attack (block of *n* bits)
  - collect 2^n encryptions
- time-memory trade-off:
  - *k/n* chosen plaintexts
  - 2^k encryptions (precomputation)
  - on-line: $2^{2k/3}$ encryptions and memory
- differential cryptanalysis
- linear cryptanalysis

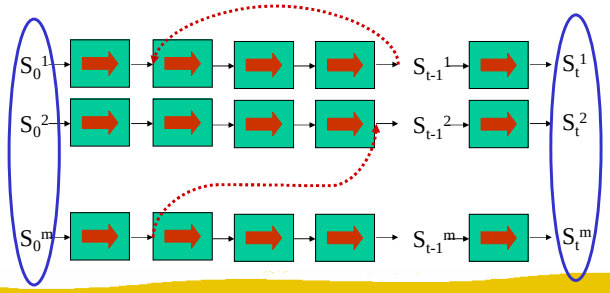
Time-memory trade-off [Hellman]

- f (x) is a one-way function: $\{0,1\}^n \rightarrow \{0,1\}^n$
- easy to compute, but hard to invert
- f(x) has (ε,t) preimage security iff
  - choose x uniformly in $\{0,1\}^n$
  - let M be an adversary that on input f(x) needs time ≤ t and outputs M(f(x)) in $\{0,1\}^n$
  - Prob{f(M(f(x))) = f(x) < ε}, where the probability is taken over x and over all the random choices of M
- t/ε should be large

Time-memory trade-off (3)

- Choose *m* different starting points and iterate for *t* steps (encrypt same message, new key)

! problem: collisions: m t « 2^n



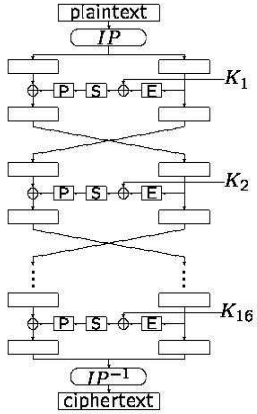
The birthday paradox

- Given a set with *S* elements
- Choose *q* elements at random (with replacements) with *q* « *S*
- The probability *p* that there are at least 2 equal elements is $1 - 2^{-q(q-1)/2S}$
- *S* large, $q = \sqrt{S}$, $p = 0.39$
- $S = 365$, $q = 23$, $p = 0.50$

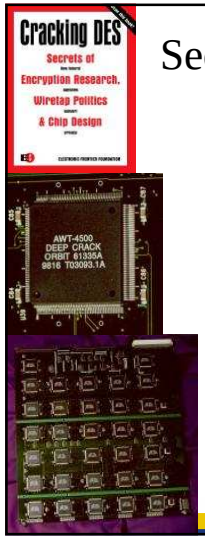
DES properties

- design: IBM + NSA (1977)
- 64-bit block cipher with a 56-bit key
- 16 iterations of a relatively simple mapping
- optimized for mid 1970ies hardware
- FIPS 41: US government standard for sensitive but unclassified data
- worldwide de facto standard since early 80ies
- surrounded by controversy: key length

Data Encryption Standard



Security of DES (56-bit key)

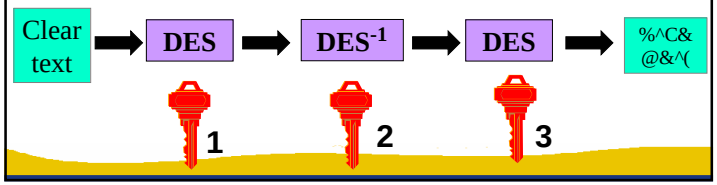


- PC: trying 1 DES key: 0.25 µs
- Trying all keys on 4000 PCs: 1 month: 2^{22} x 2^{16} x 2^{5} x 2^{12} = 2^{55}
- M. Wiener's estimate (1993): 1,000,000 $ machine: 35 minutes

EFF Deep Crack (July 1999)
250,000 $ machine: 50 hours…

Solution to DES key length

- Moore's "law": speed of computers doubles every 18 months
  - Conclusion: key lengths need to grow in time
- Use new algorithms with longer keys
- Or replace DES by triple-DES (168-bit key):



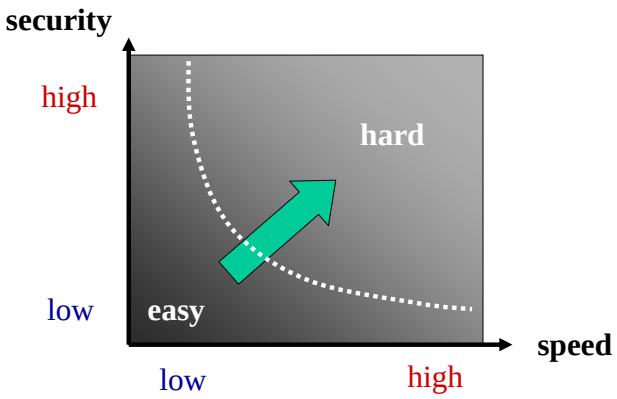
AES (Advanced Encryption Standard)

- Open competition launched by US government (‘97)
- 21 contenders, 15 in first round, 5 finalists
- decision October 2, 2000
- 128-bit block cipher with long key (128/192/256 bits)
- five finalists:
  - MARS (IBM, US)
  - RC6 (RSA Inc, US)
  - Rijndael (KULeuven/PWI, BE)
  - Serpent (DK/IL/UK)
  - Twofish (Counterpane, US)

AES properties

- Rijndael: design by V. Rijmen (COSIC) and J. Daemen (Proton World, ex-COSIC)
- 128-bit block cipher with a 128/192/256-bit key
- 10/12/14 iterations of a relatively simple mapping
- optimized for software for 8/16/32/64-bit machines, also suitable for hardware

Design trade-off



O’Connor versus Massey

- Luke O’Connor

  “most ciphers are secure after sufficiently many rounds”

- James L. Massey

  “most ciphers are too slow after sufficiently many rounds”

AES Status

- FIPS 197 published on 6 December 2001
- Revised FIPS on modes of operation
- Rijndael has more options than AES

- fast adoption in the market
  - early 2002, 74 products are using AES
  - standardization: ISO, IETF, …
- slower adoption in financial sector

Breaking news: is AES broken?

- *"AES may have been broken. Serpent, too. Or maybe not. In either case, there's no need to panic. Yet. But there might be soon. Maybe."* – Bruce Schneier
- New result [Courtois and Pieprzyk '02]
  - "express the entire algorithm as multivariate quadratic polynomials"
    - 2^{100}-ish attack against AES
    - 2^{200}-ish attack against Serpent
- Moral: algebraic structure is dangerous

AES/Rijndael: 1 round

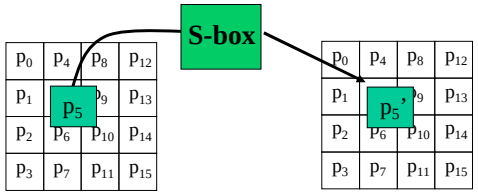
p_0	p_4	p_8	p_{12}
p_1	p_5	p_9	p_{13}
p_2	p_6	p_{10}	p_{14}
p_3	p_7	p_{11}	p_{15}

state: 16 bytes = 128 bits

1 round consists of 4 operations

SubBytes
ShiftRows
MixColumn
AddRoundKey

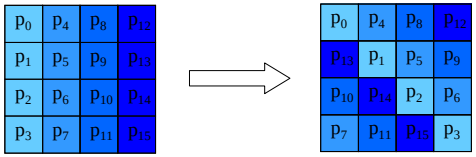
Rijndael round: SubBytes



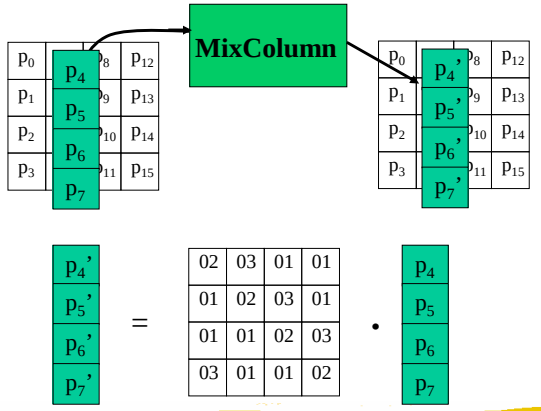
256 byte table

mapping x^{-1} over GF(2^8), plus some affine transformation over GF(2)

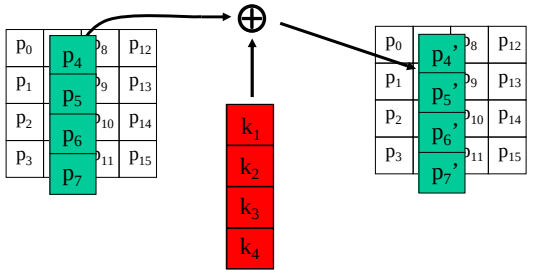
Rijndael round: ShiftRows



Rijndael round: MixColumn



Rijndael round: AddRoundKey



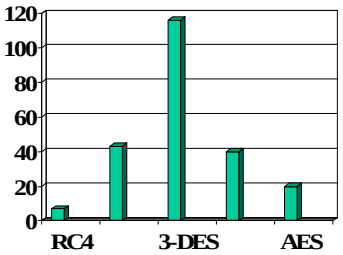
Rijndael design strategy

- simple and elegant
- no integer arithmetic
- wide trail strategy:
  - strong resistance against linear and differential attacks
  - over 4 rounds, sum of number of “active” input and output bytes equals 25
- diffusion based on (8,4) MDS code with minimum distance 5

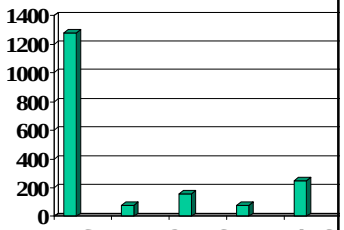
  [p1 p2 p3 p4 | p1’ p2’ p3’ p4’]

Performance reference data (Pentium)

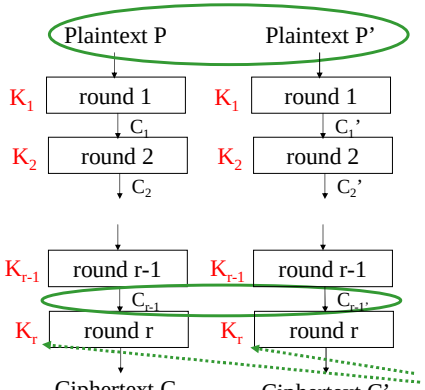
encryption speed (cycles/byte)



key setup (cycles)



Differential cryptanalysis [Biham Shamir90]

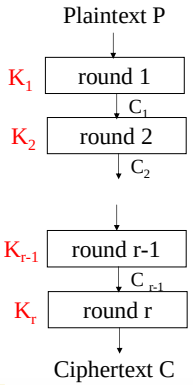


choose the difference P, P'

try to predict the difference C_{r-1}, C_{r-1}'

this leaks information on K_r

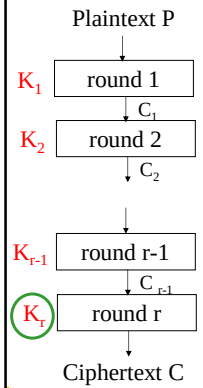
Linear cryptanalysis [Matsui93]



- Simple version: cipher can be approximated with XOR, with probability $p \neq ½$
- You learn one bit of the key.

$$\left(\bigoplus_{i\in\{1...64\}} \mathcal{P}^{(i)}\right) \oplus \left(\bigoplus_{j\in\{1...64\}} \mathcal{C}^{(j)}\right) = \bigoplus_{k\in\{1...56\}} \mathcal{K}^{(k)}$$

Linear cryptanalysis [Matsui93]



- Complex version: assume final subkey is broken
- You learn one bit of the key.

$$\left(\bigoplus_{i\in\{1...64\}} \mathcal{P}^{(i)}\right) \oplus \left(\bigoplus_{j\in\{1...64\}} \mathcal{C}^{(j)}\right) \oplus \left(\bigoplus_{m\in\{1...32\}} f(\mathcal{C},\mathcal{K}_{16})^{(m)}\right) = \bigoplus_{k\in\{1...56\}} \mathcal{K}^{(k)}$$

Equations from Junod's PhD thesis ('00)

Linear and differential cryptanalysis

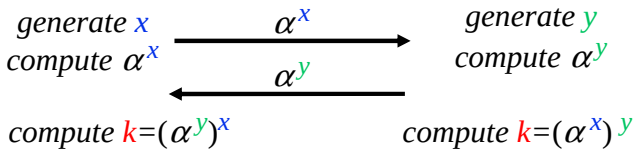
- hard to find good linear or differential attacks
  - it is even harder to prove that it is impossible to find good linear or differential attacks
  - for some ciphers, this proof exists
- there exist many optimizations and generalizations
  - it is even harder to show that none of these work for a particular cipher
- analysis requires some heuristics
- DES: linear analysis needs 2^{43} known texts and differential analysis needs 2^{47} chosen texts

Public key primitives

- Diffie-Hellman
  - Hard problem: Discrete logarithms
- RSA
  - Hard problem: Factoring composite numbers
- Field: integers modulo a large prime number (numbers wrap around)

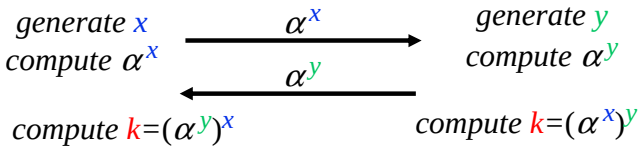
A public-key distribution protocol: Diffie-Hellman

- Before: Alice and Bob have never met and share no secrets; they know a public system parameter α



- After: Alice and Bob share a short term key k
  - Eve cannot compute k : in several mathematical structures it is hard to derive x from α^x (this is known as the discrete logarithm problem)

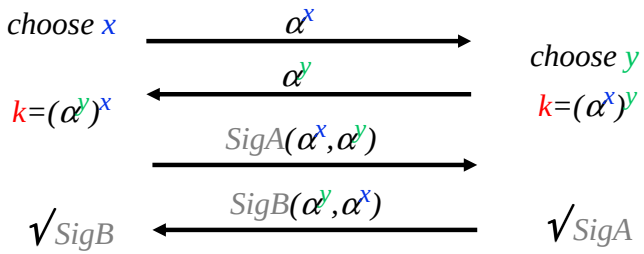
Diffie-Hellman (continued)



- BUT: How does Alice know that she shares this secret key k with Bob?
- Answer: Alice has no idea at all about who the other person is! The same holds for Bob.

Station to Station protocol (STS)

- The problem can be fixed by adding digital signatures
- Many variations on this theme used in practice



RSA ('78)

- Choose 2 "large" prime numbers p and q
- modulus $n = p.q$
- compute $\lambda(n) = \text{lcm}(p-1,q-1)$
- choose e relatively prime w.r.t. $\lambda(n)$
- compute $d = e^{-1} \mod \lambda(n)$

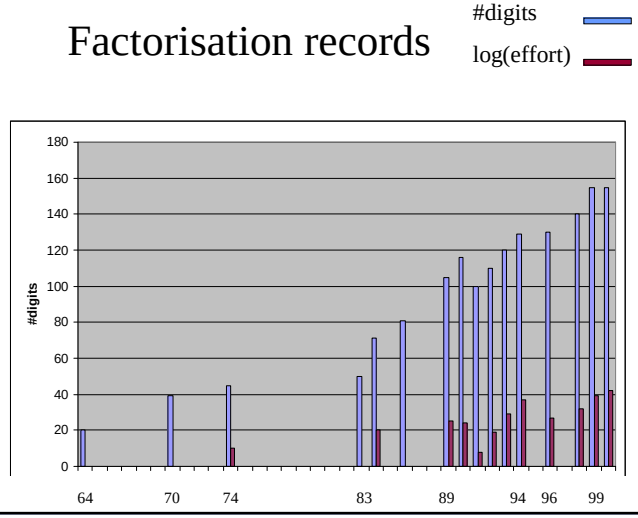
- public key = (e,n)
- private key = (d,p,q)

The security of RSA is based on the "fact" that it is easy to generate two large primes, but that it is hard to factor their product

- encryption: $c = m^e \mod n$
- decryption: $m = c^d \mod n$

try to factor 2419

Factorisation records



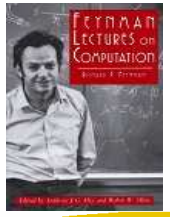
What about quantum computers?

- exponential parallelism

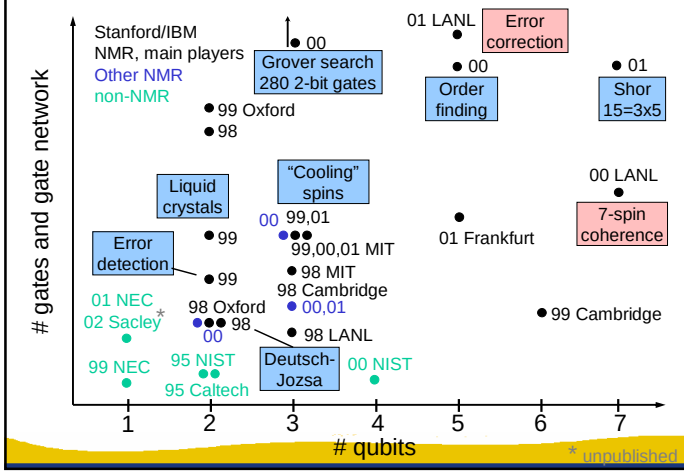
n coupled quantum bits

2^n degrees of freedom !

- Shor 1994: perfect for factoring
- But: can a quantum computer be built?



State of the art in coherent qubit control



Advantages of public-key cryptology

- Reduce protection of information to protection of authenticity of public keys
- Confidentiality without establishing secret keys
  - extremely useful in an open environment
- Data authentication without shared secret keys: digital signature
  - sender and receiver have different capability
  - third party can resolve dispute between sender and receiver

Disadvantages of public-key cryptology

- Calculations in software or hardware two to three orders of magnitude slower than symmetric algorithms
- Longer keys: 1024 bits rather than 56…128 bits
- What if factoring is easy?