Crypto primitives

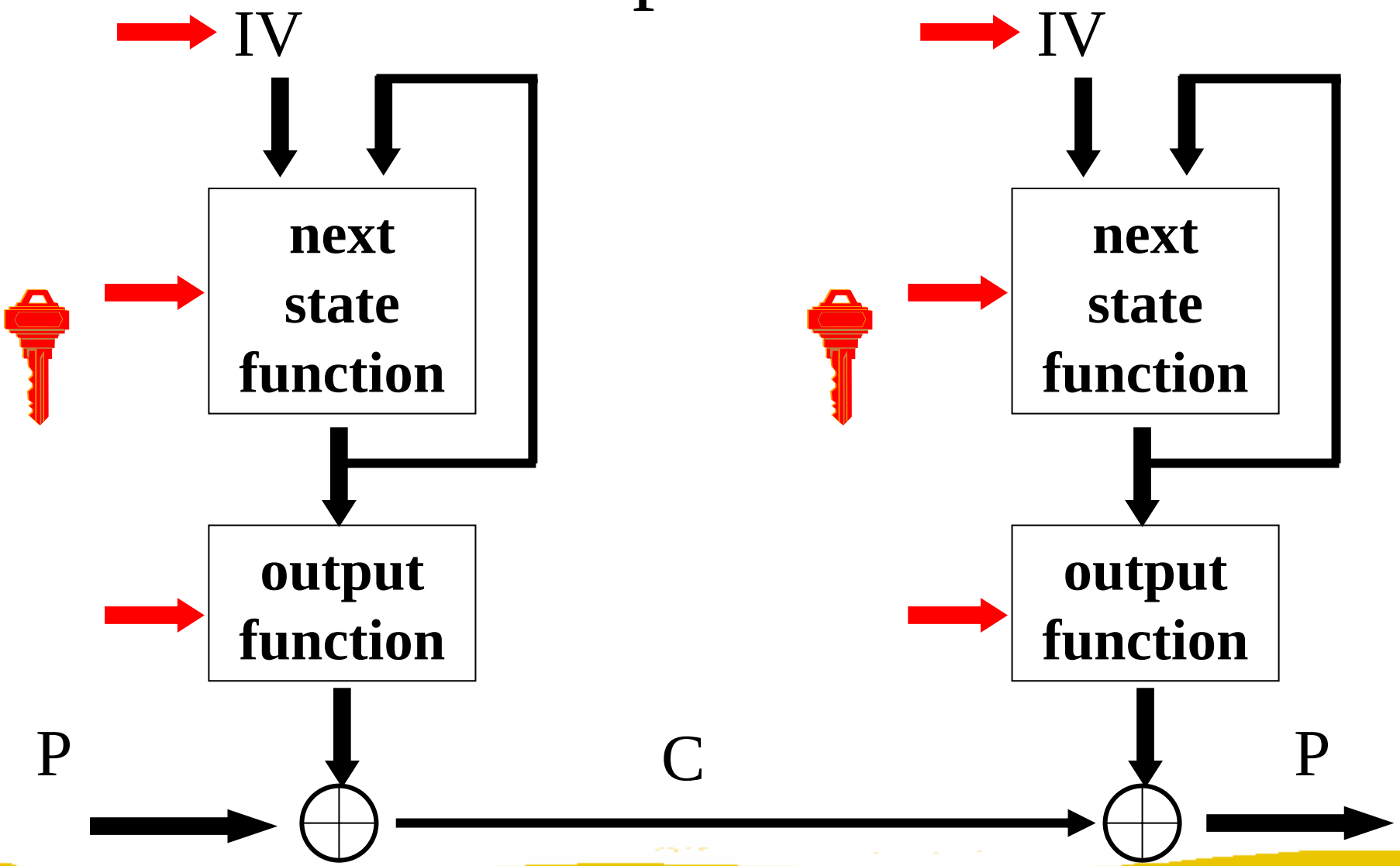
Today's talk includes slides from:

Bart Preneel and Dan Wallach

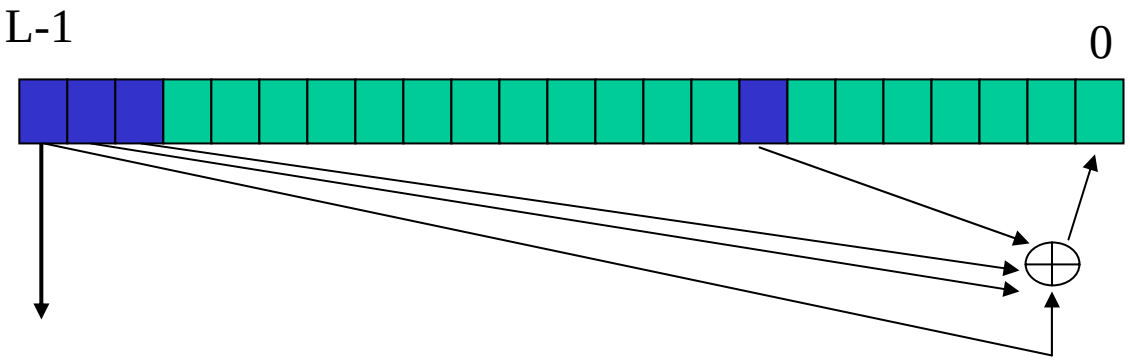
Crypto primitives

- The building blocks of everything else
  - Stream ciphers
  - Block ciphers (& cipher modes)
- Far more material than we can ever cover
  - In addition to your book…
    - Nice reference, lots of details:
    - http://home.ecn.ab.ca/~jsavard/crypto/jscrypt.htm

Model of a practical stream cipher



LFSR based stream cipher



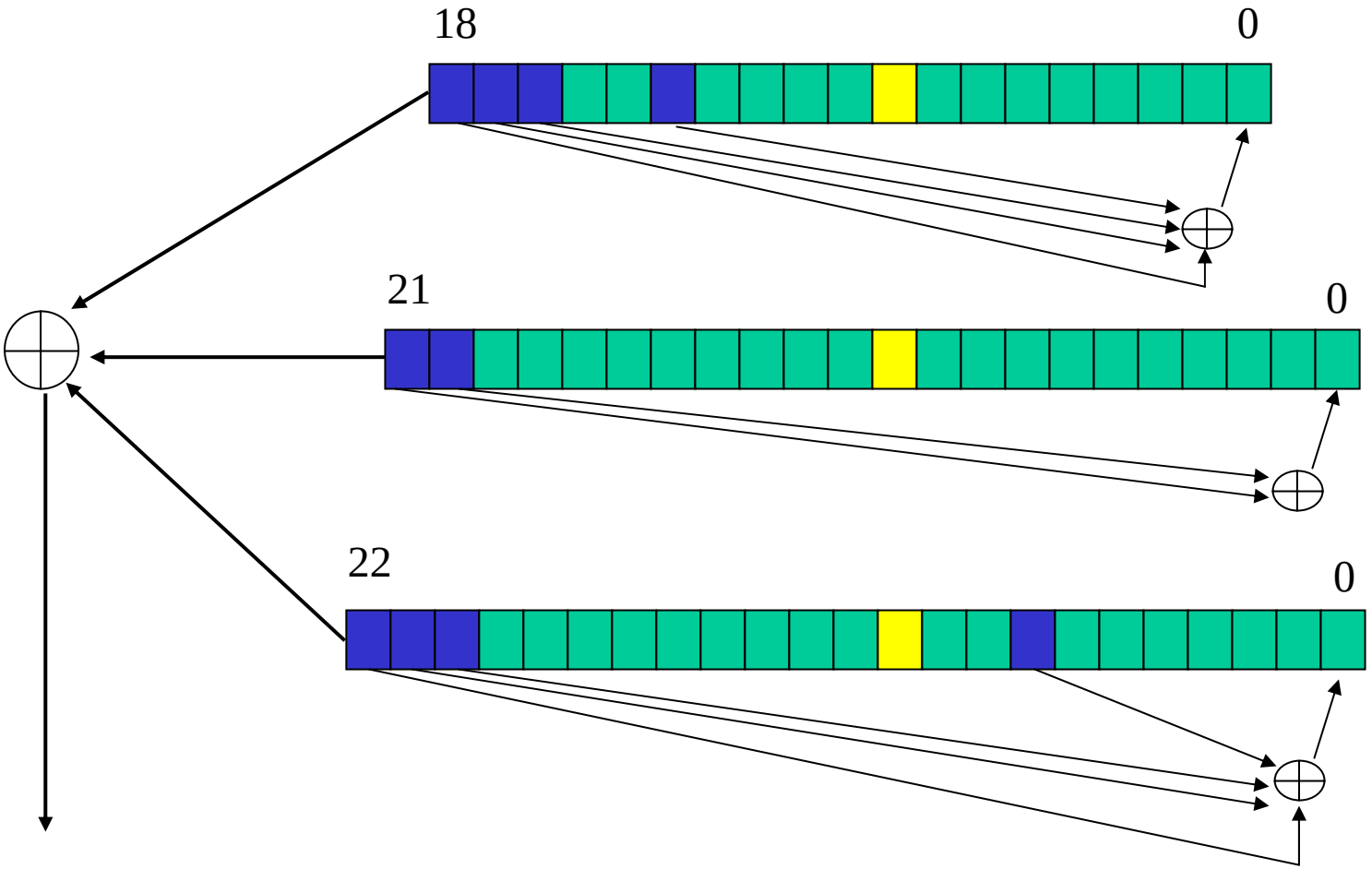
+ good randomness properties

+ mathematical theory

+ compact in hardware

- too linear: easy to predict after 2L output bits

A5/1 stream cipher (GSM)



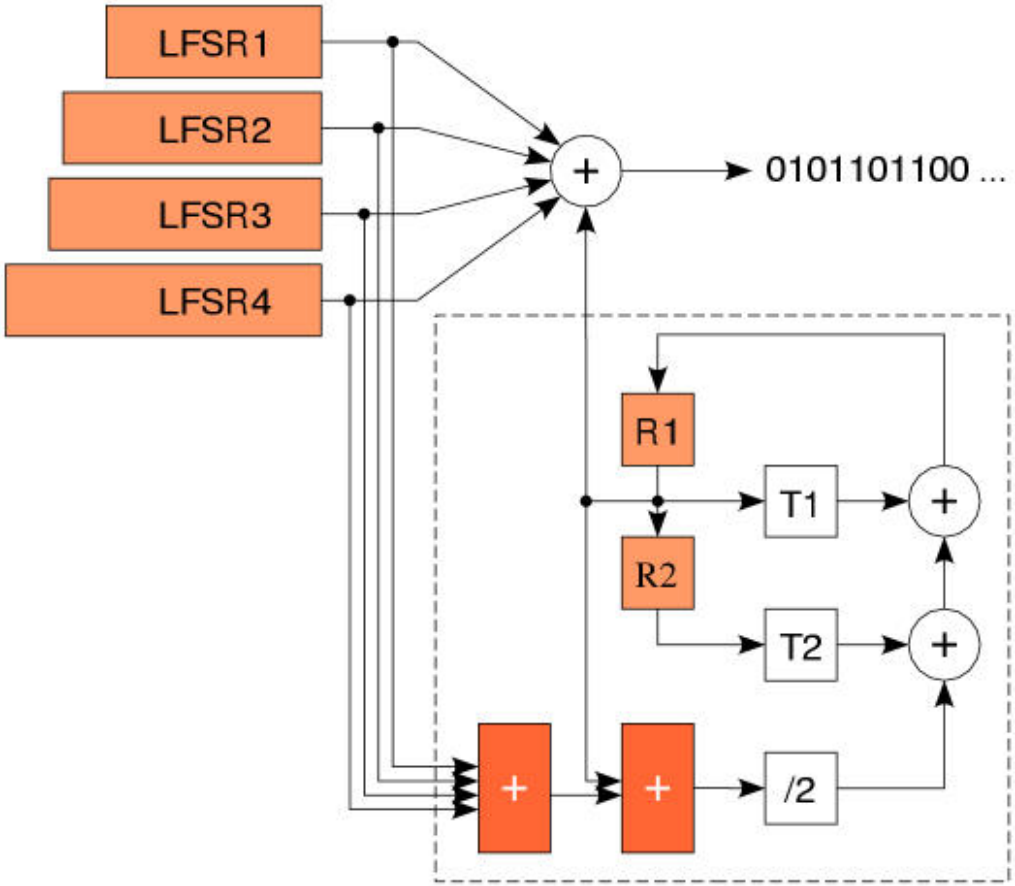
Clock control: registers agreeing with majority are clocked (2 or 3)

A5/1 stream cipher (GSM)

A5/1 attacks

- exhaustive key search: 2^{64} (or rather $2^{54)}$
- search 2 smallest registers: 2^{45} steps
- [BWS00] 2 seconds of plaintext: 1 minute on a PC
  - 2^{48} precomputation, 146 GB storage
- Real-time attack?
  - Barkan, Biham and Keller (Crypto ’03)
  - Ciphertext-only attack on A5/2, A5/1, etc.
    - A5/2: requires “a few dozen milliseconds” of traffic
    - Fetches key in 1 second of CPU on a normal PC

Bluetooth stream cipher



- best known shortcut attack: 2^{70} rather than 2^{128}

Cryptanalysis of stream ciphers

- exhaustive key search (key of k bits)
  - 2^k encryptions, about k known plaintext bits
- time-memory trade-off (memory of m bits)
  - 2^t short output sequences
  - 2^{m-t} precomputation and memory

- linear complexity
- divide and conquer
- fast correlation attacks (decoding problem)

A simple cipher: RC4 (1992)

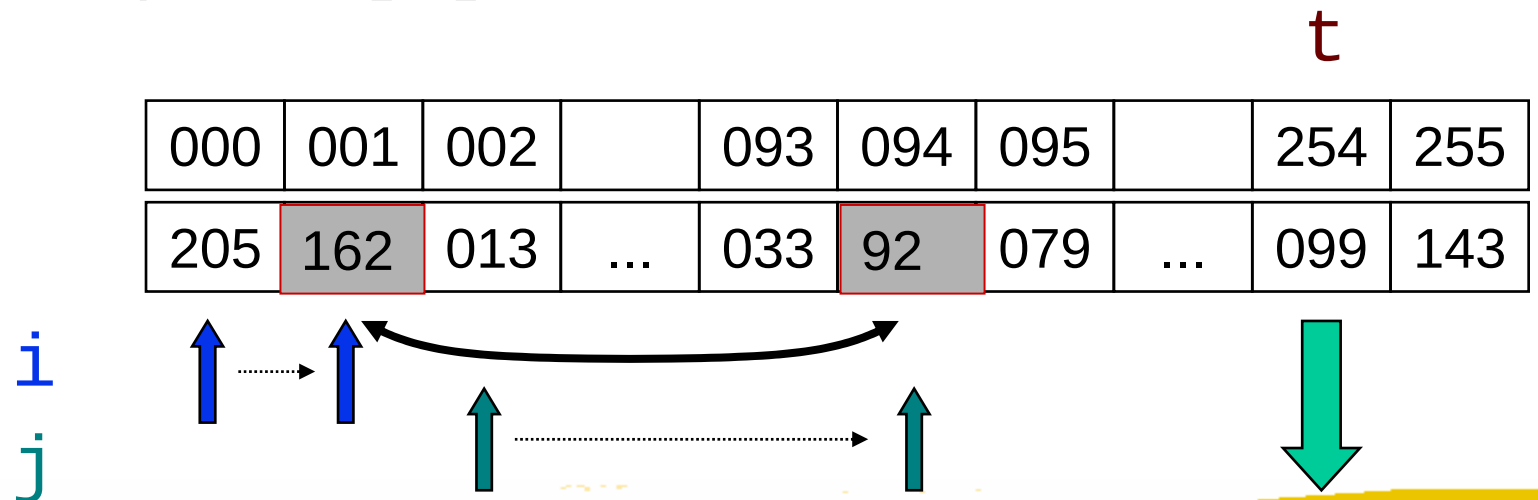
- designed by Ron Rivest (MIT)
- S[0..255]: secret table derived from user key K

```
for i=0 to 255 S[i]:=i
j:=0
for i=0 to 255
    j:=(j + S[i] + K[i]) mod 256
    swap S[i] and S[j]
i:=0,  j:=0
```

A simple cipher: RC4 (1992)

Generate key stream which is added to plaintext

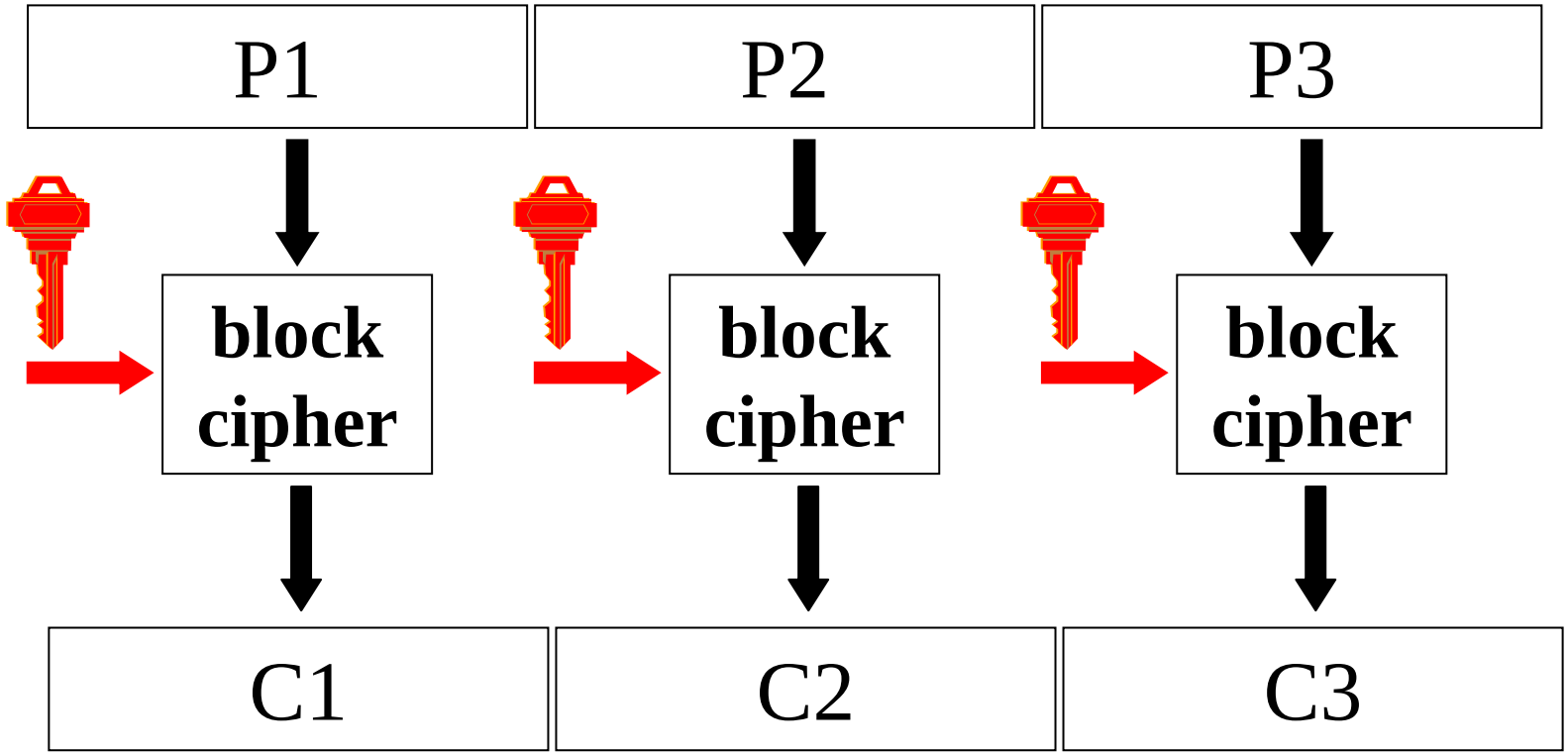
```
i:=i+1
j:=(j + S[i]) mod 256
swap S[i] and S[j]
t:=(S[i] + S[j]) mod 256
output S[t]
```



RC4: weaknesses

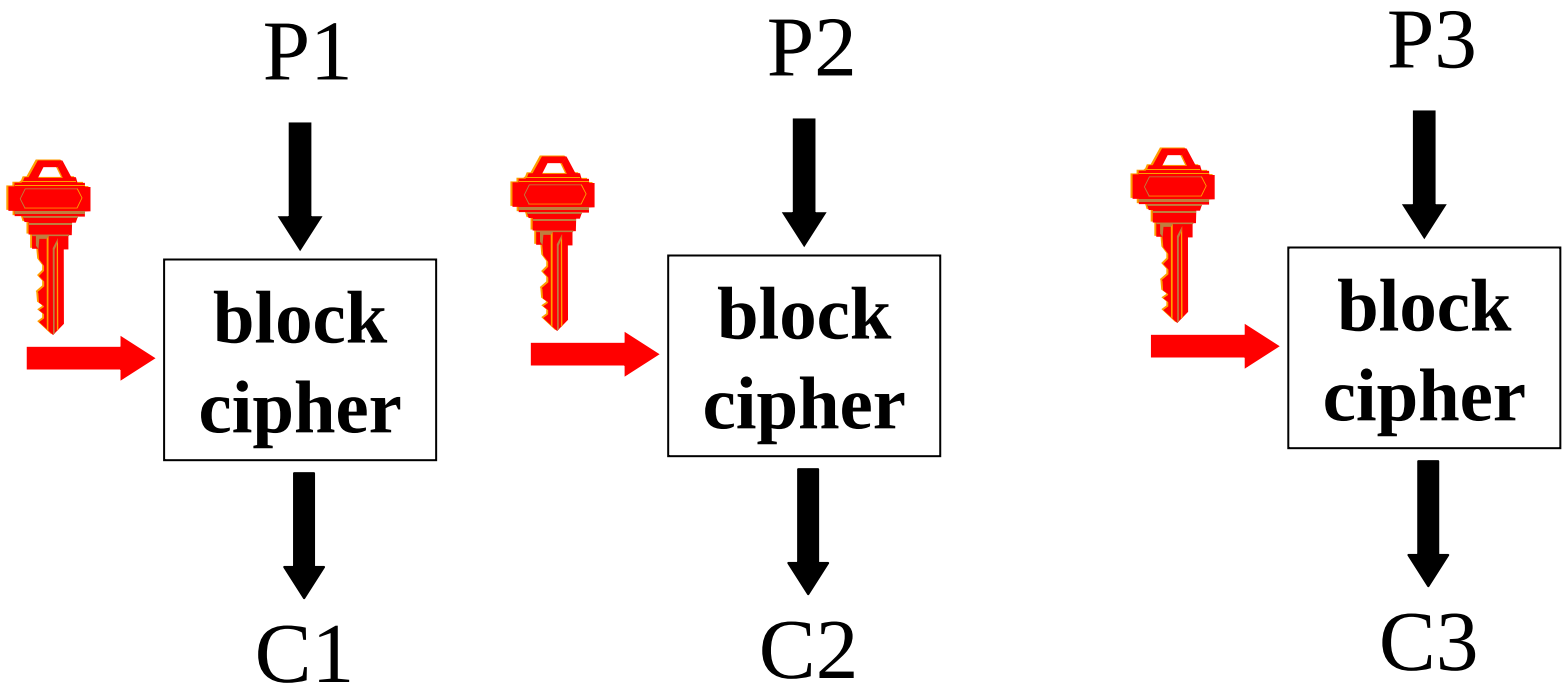
- often used with 40-bit key
  - US export restrictions until Q4/2000
- best known general shortcut attack: 2^{700}
- weak keys and key setup (shuffle theory)
- some statistical deviations
  - e.g., 2nd output byte is biased
  - solution: drop first 256 bytes of output
- problem with resynchronization modes (WEP)

Block cipher

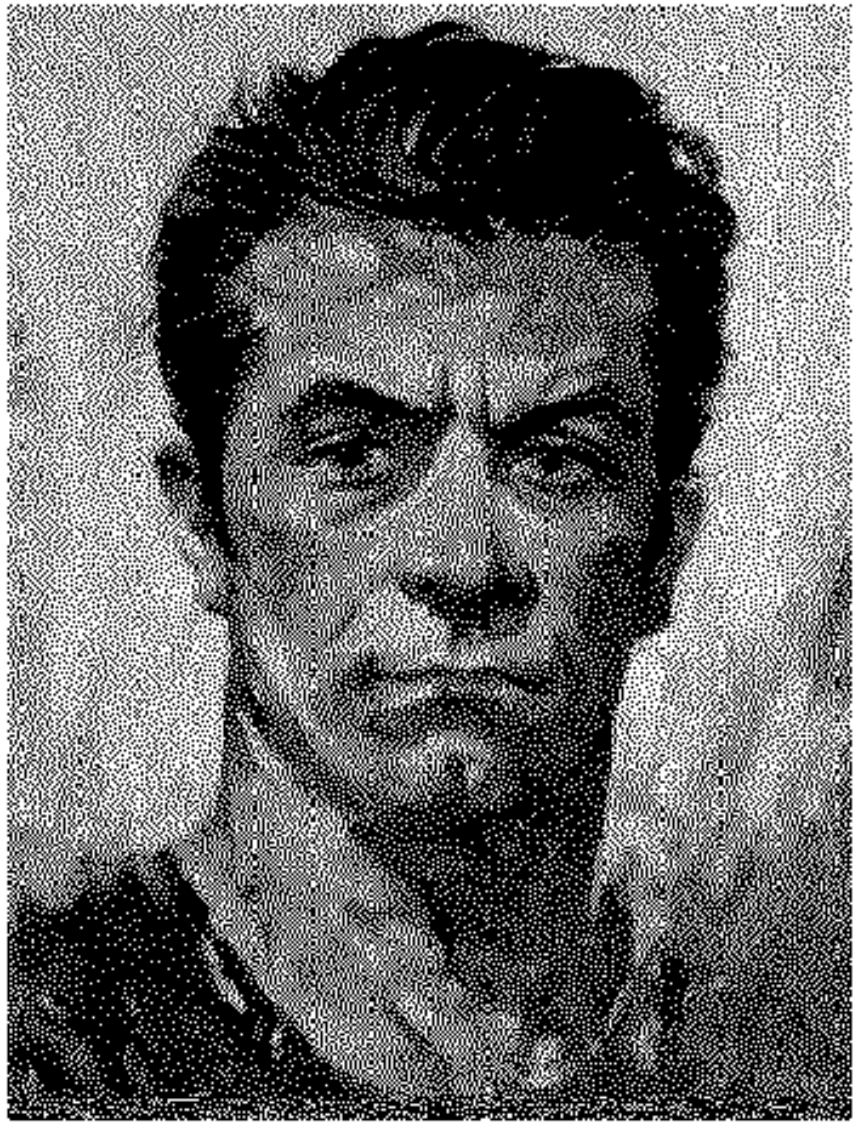


- larger data units: 64…128 bits
- memoryless
- repeat simple operation (round) many times

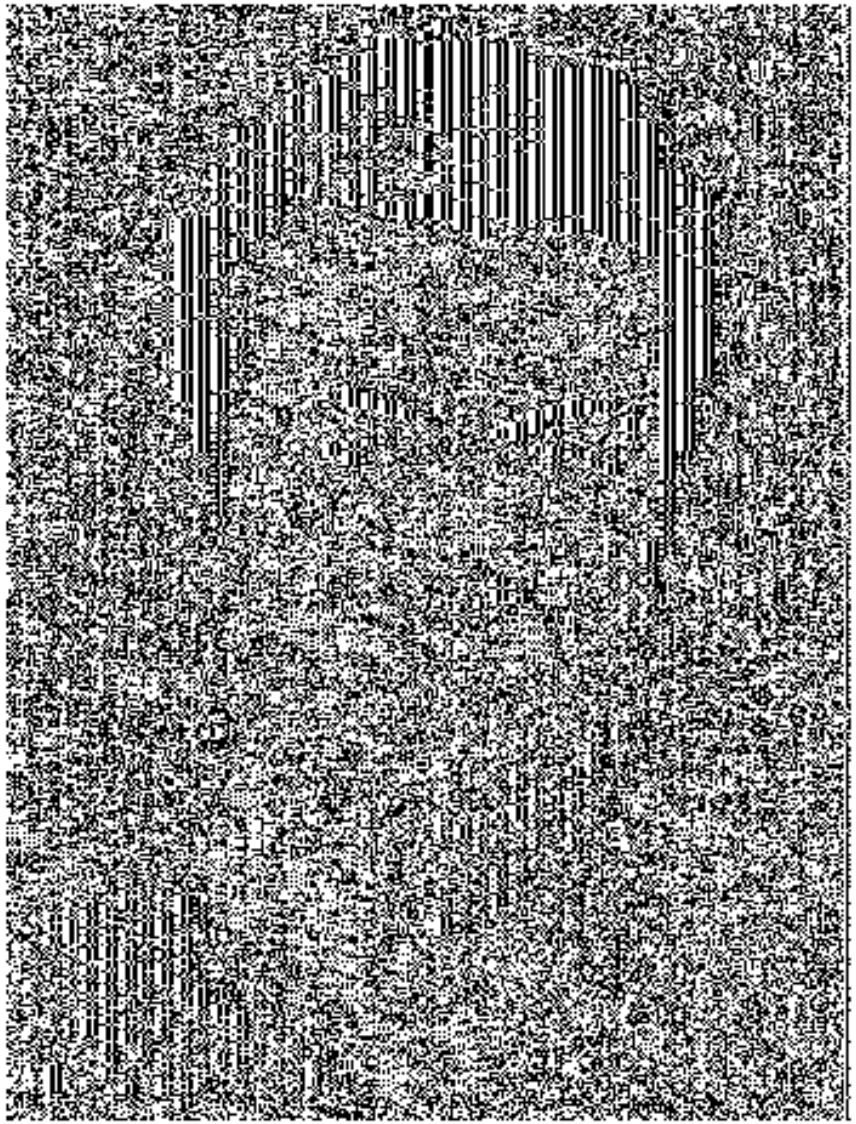
How NOT to use a block cipher: ECB mode



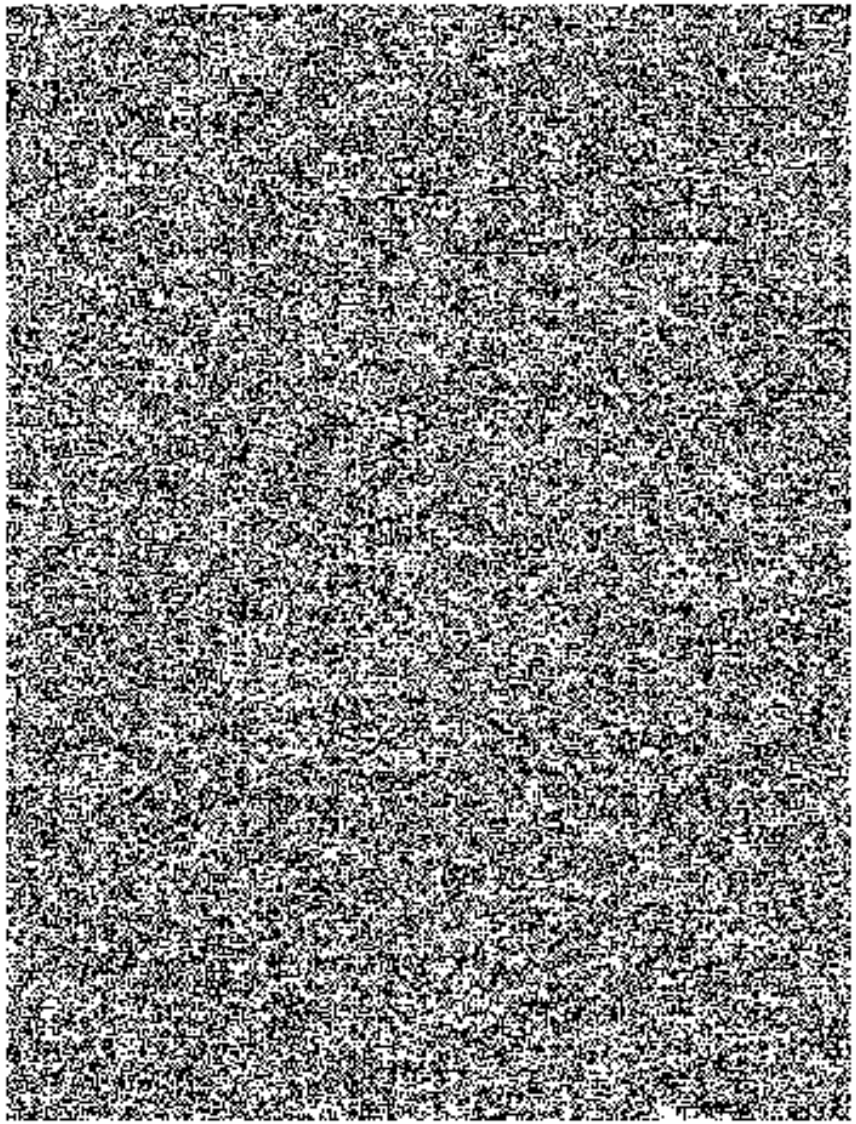
An example plaintext



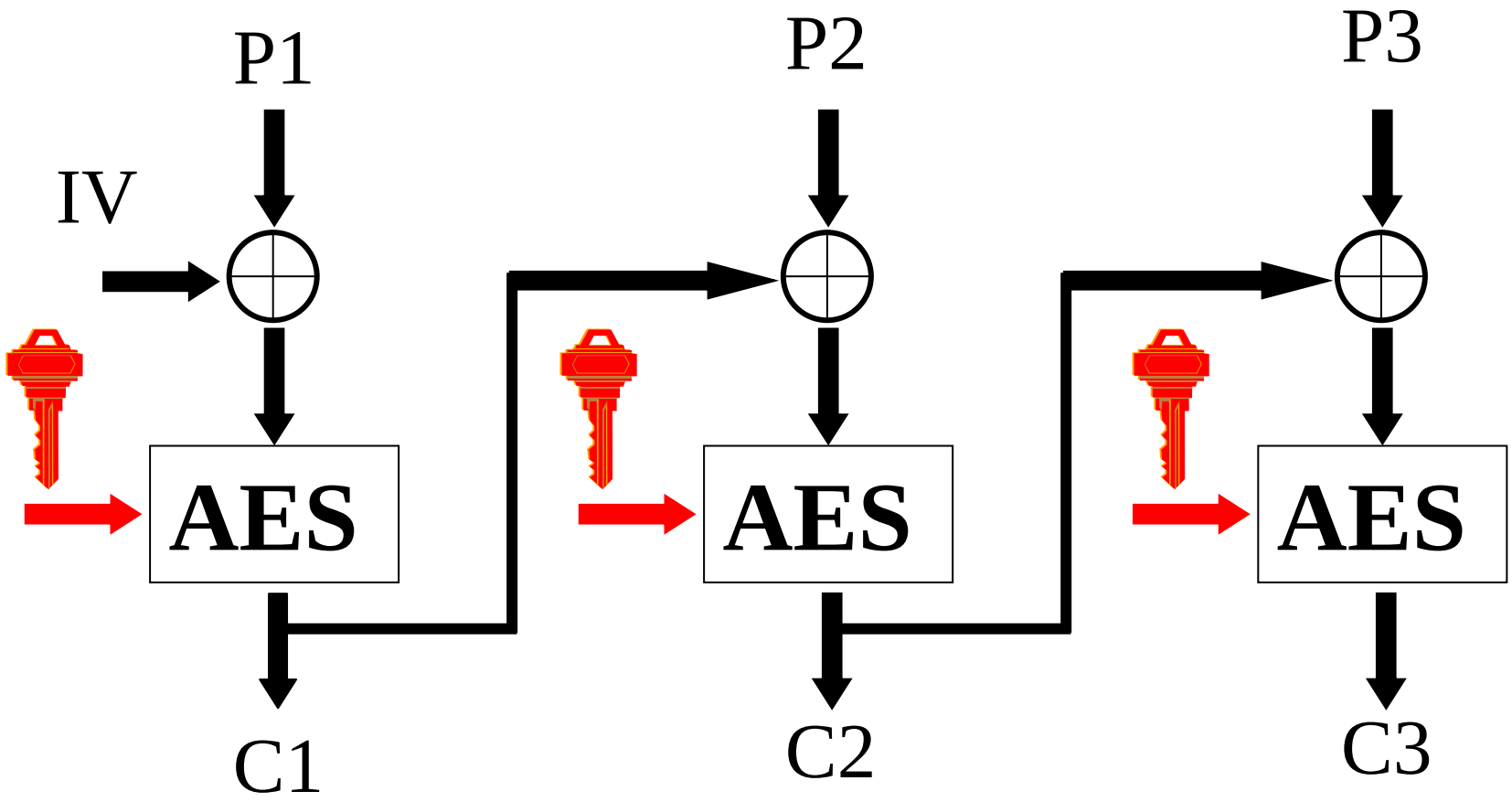
Encrypted with AES in ECB mode



Encrypted with AES in CBC mode

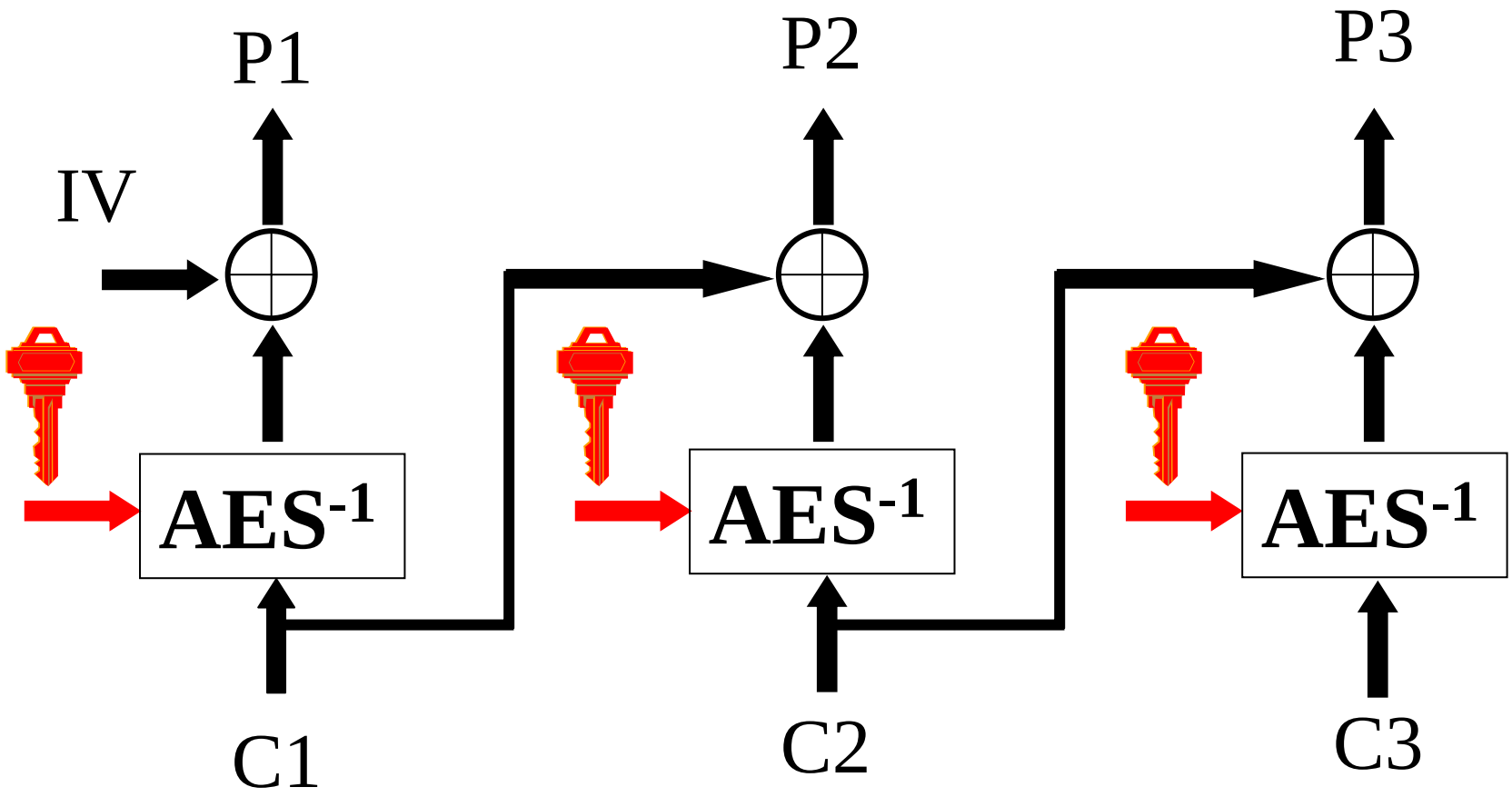


How to use a block cipher: CBC mode



need random IV

CBC mode decryption



Secure encryption

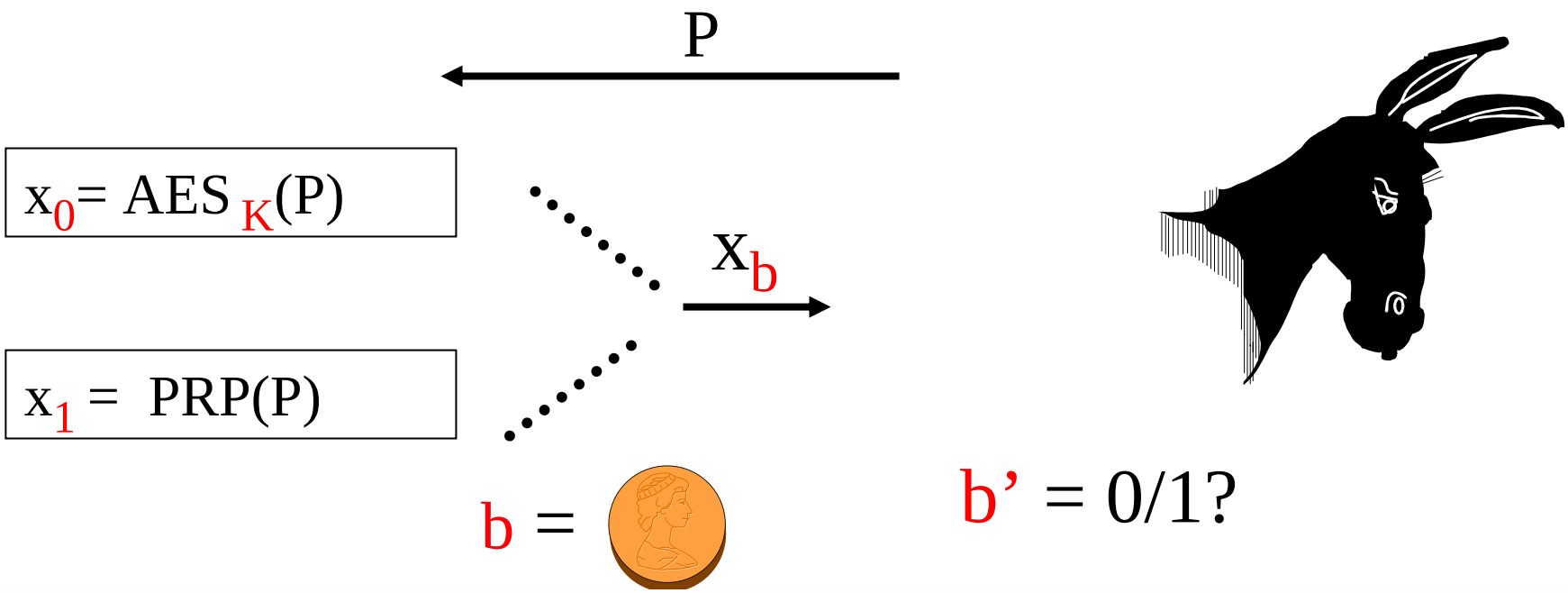
- What is a secure block cipher anyway?
- What is secure encryption anyway?

- 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 $\Rightarrow$ 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

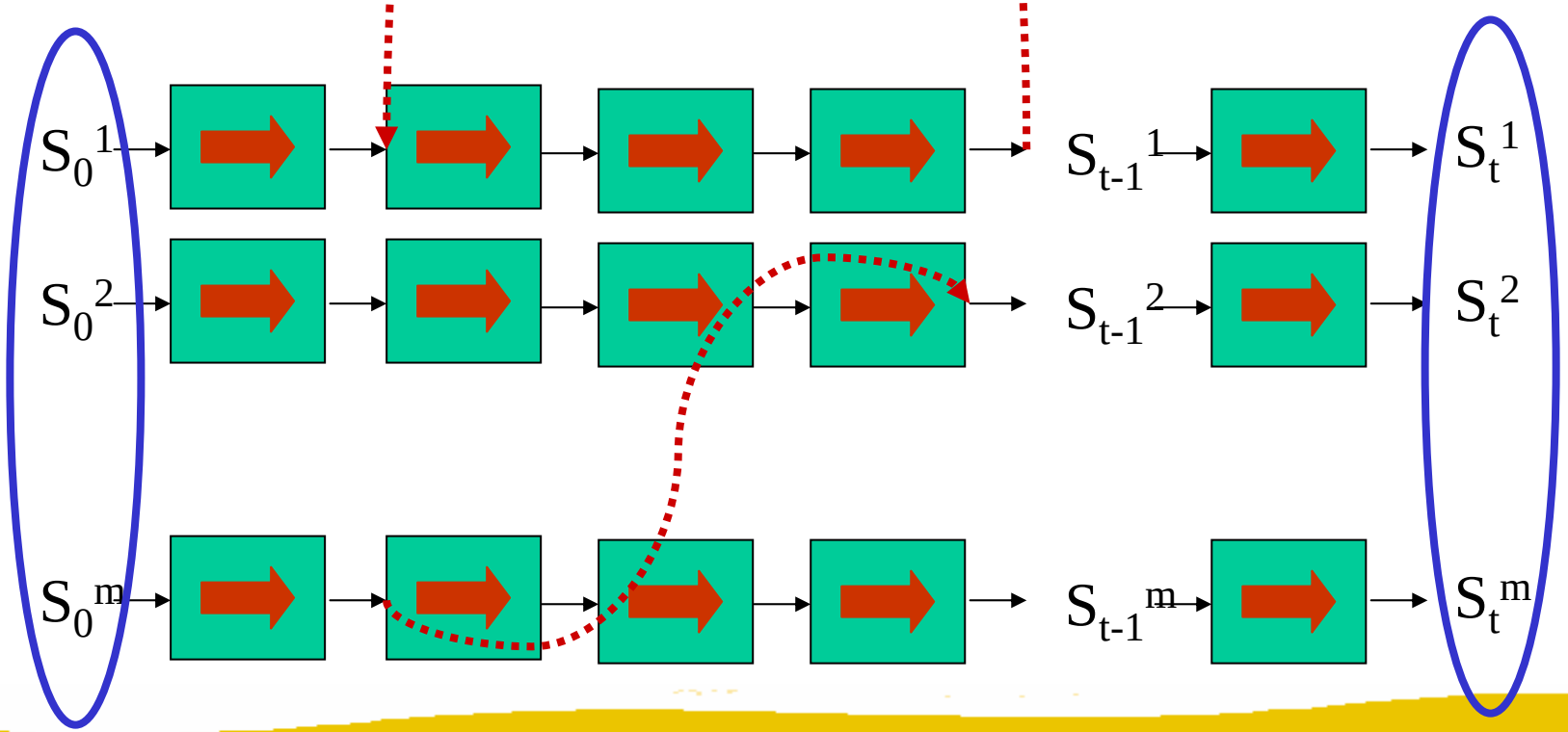
Time-memory trade-off [Hellman]

- 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 $\leq$ 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

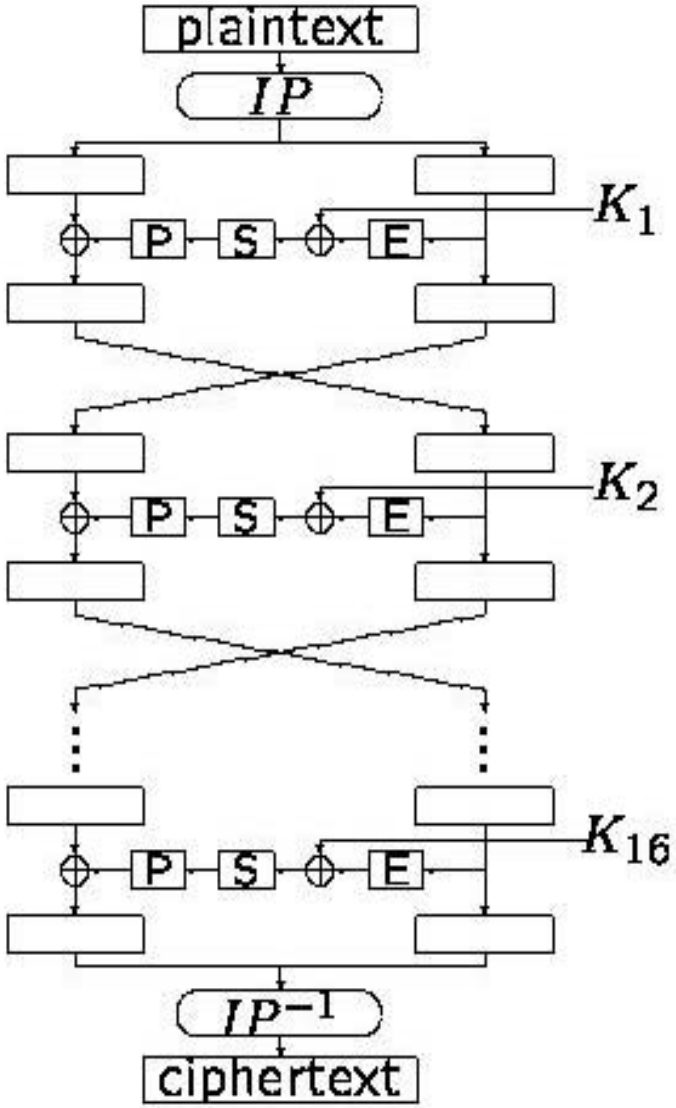
- Given a set with S elements
- Choose q elements at random (with replacements) with $q \ll S$
- The probability p that there are at least 2 equal elements is $1 - 2^{-q(q-1)/2S}$

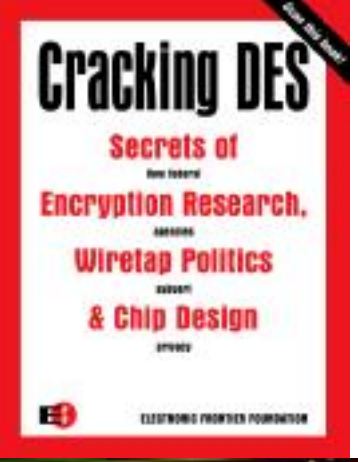
- S large, $q = \sqrt{S}$, $p = 0.39$
- $S = 365$, $q = 23$, $p = 0.50$

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

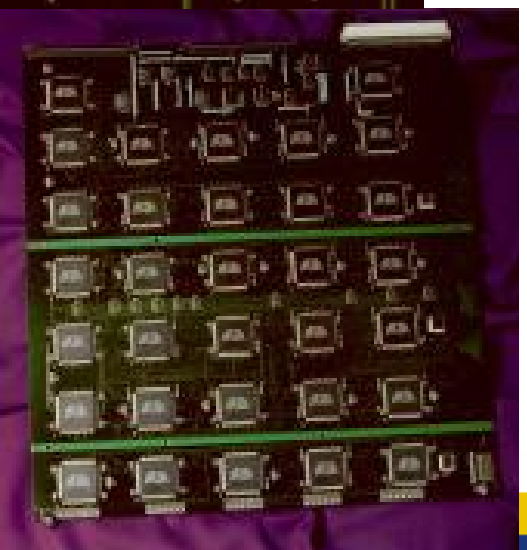
Data Encryption Standard





Security of DES (56-bit key)

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

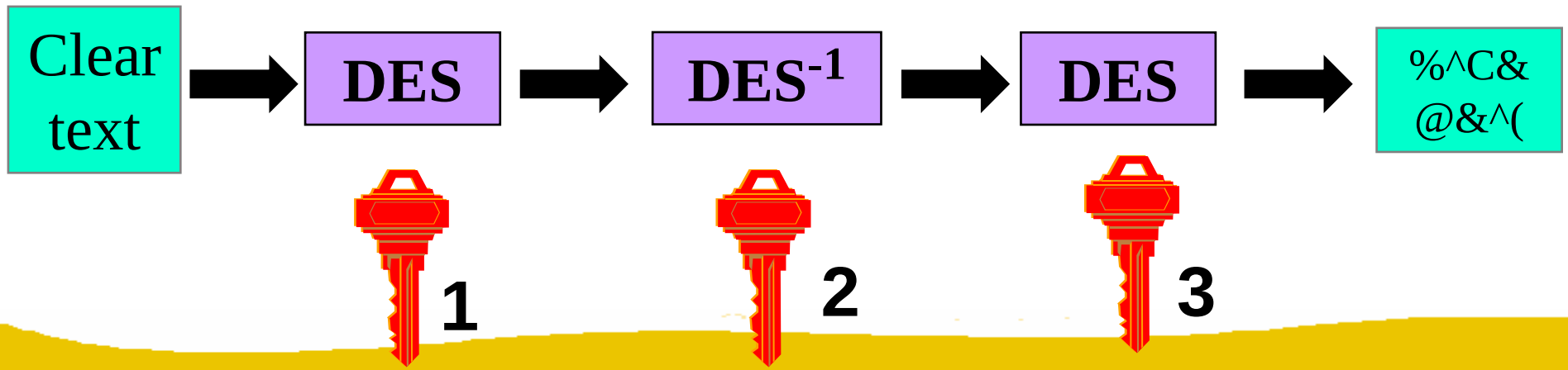


EFF Deep Crack (July 1999)

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):



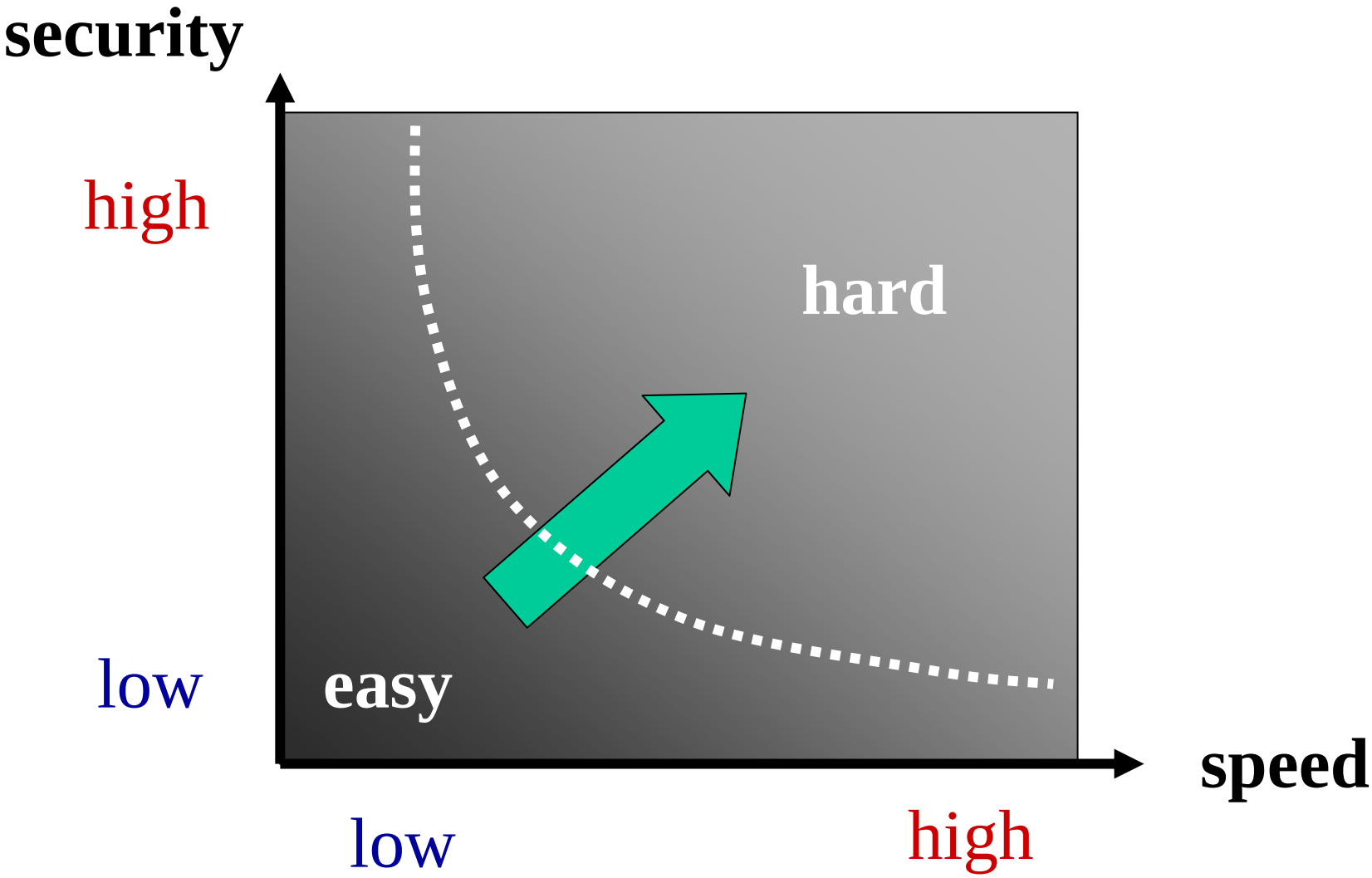
AES (Advanced Encryption Standard)

- 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