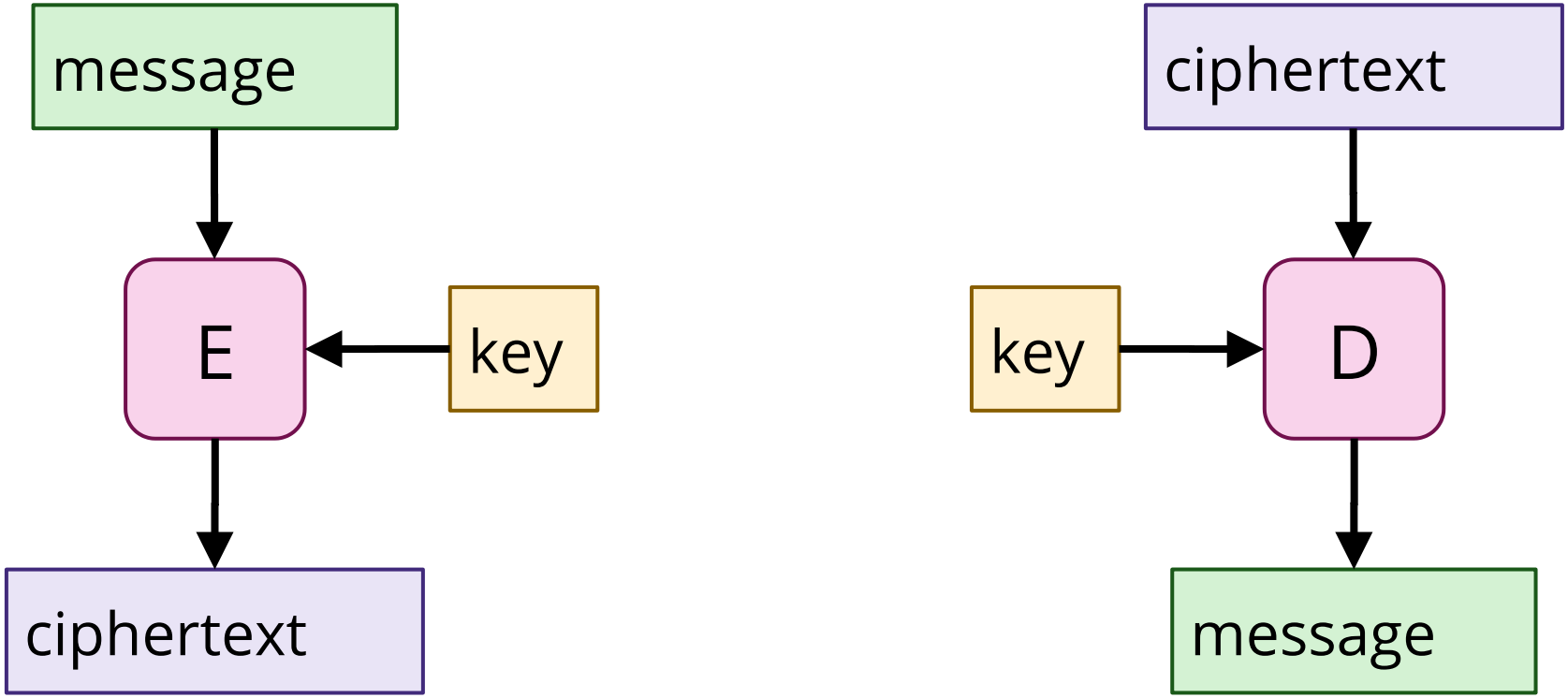
Cryptography

secret-key cryptography

Encryption scheme (symmetric)



‘Perfect’ secrecy

and why it’s not enough

Vigenère cipher

ALPN**ALP**CZIYI**AL**TH**AL**CDBKSHLGCTAGSPURPAZ
| 4 | 8 | 4 |

Depends on repetition of the key – how can we avoid?

Vigenère cipher

ALPN**ALP**CZIYI**AL**TH**AL**CDBKSHLGCTAGSPURPAZ
| 4 | 8 | 4 |

Depends on repetition of the key – how can we avoid?

message: KILLTHEEMPEROR
key: KISSTHECAPTAIN

If the key is as long as the message, no need to repeat!

Lessons from the Vigenère cipher

- Patterns in ciphertext reveal the keylength through Kasiski analysis

If the key is as long as the message, no need to repeat!

- Still vulnerable to analysis of patterns in the key material

What if we choose a key with no patterns?

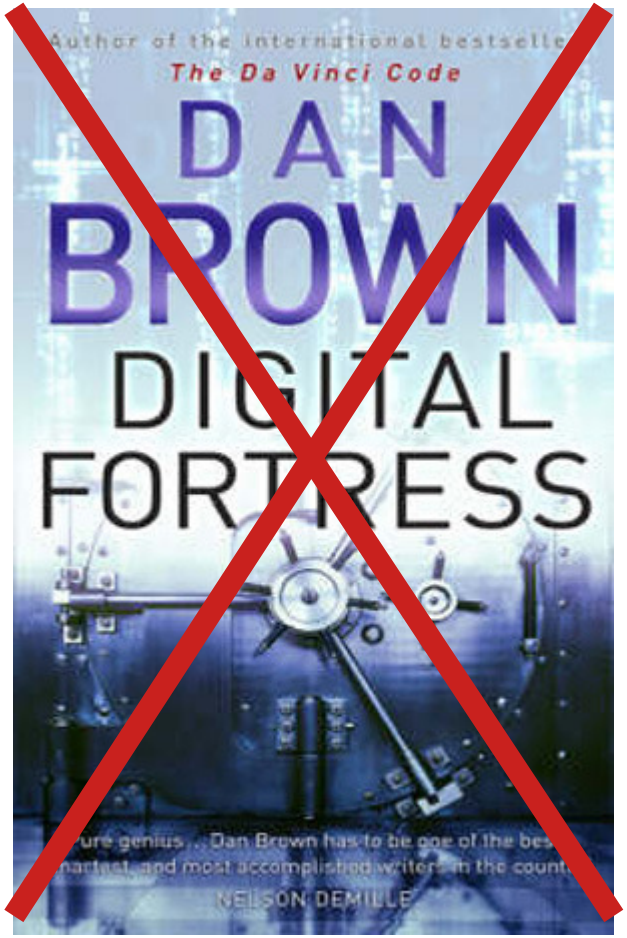
message: KILLTHEEMPEROR
key: *<random chars>*

One-time pad

If the key is:

1. **Truly** random (not **pseudo**random)
2. At least as long as the plaintext
3. Never reused
4. Kept completely secret

Then the resulting ciphertext is **impossible** to decrypt without the key.



One-time pad: brute force?

`RHGQIQLTYATSYLJWAMLERGMCNABGTQHUQRVGEELCRC`

- <random key 1>
`warmchocolatefudgesundaeshalfoffthissunday`

- <random key 2>
`warmchocolatefudgesundaestwicethepricehaha`

Which is the real message?

The one-time pad is information-theoretically secure

Information-theoretic security

Cannot be broken even with unlimited computing power.

The ciphertext *does not contain enough information* for any cryptanalysis to succeed.

Can you think of another system with this property?

Answer: VLQJOH XVH VBVWHP

The one-time pad

If the key is:

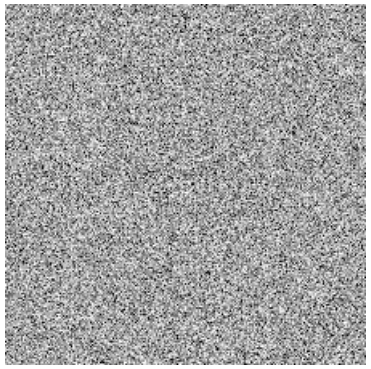
1. **Truly** random (not **pseudo**random)
2. At least as long as the plaintext
3. Never reused
4. Kept completely secret

Then the resulting ciphertext is
impossible to decrypt without the key.

So why aren't we using it?

One-time pads are used in some diplomatic and intelligence communications (i.e. by spies), but very rarely elsewhere.

The one-time pad



True Random is difficult and often slow to generate
(even small biases in randomness source have led to viable attacks)

More message == more key

Sufficient key material must be securely shared between communicating parties.

Nothing can be reused without risking compromise, so every byte you encrypt costs you a byte of key.

‘Perfect’ secrecy

isn’t good enough

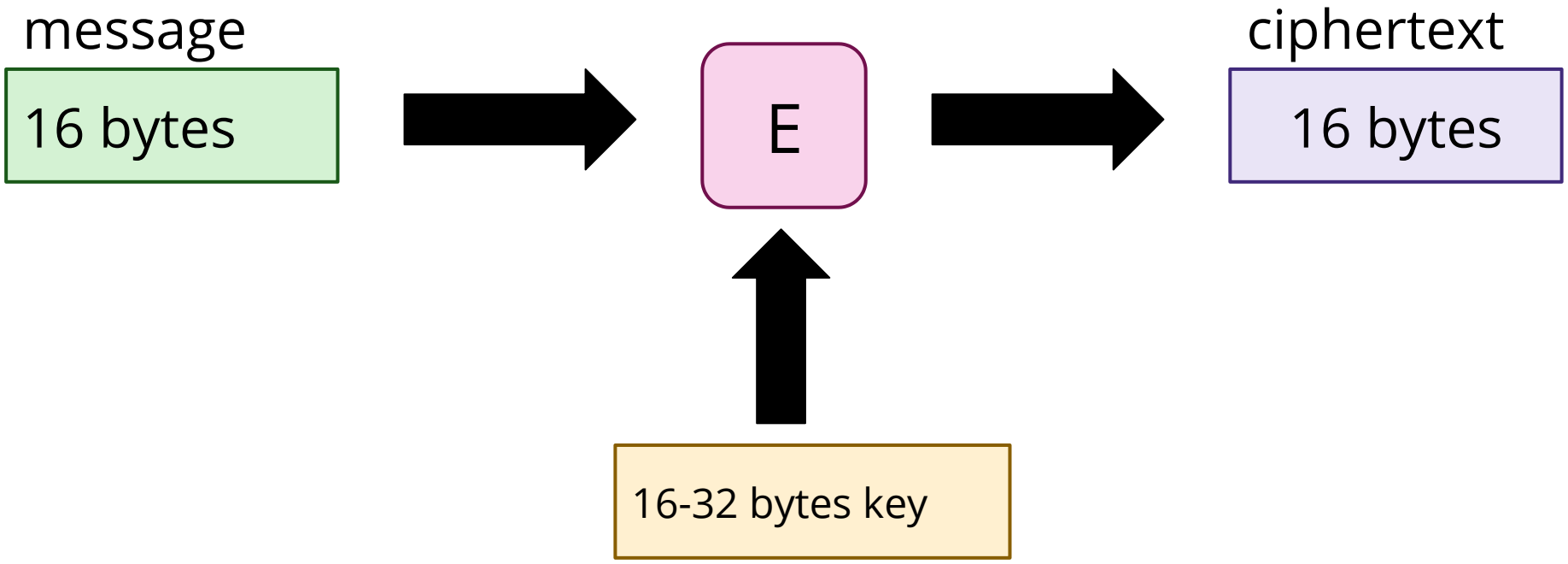
Modern symmetric ciphers focus on more ‘usable’ systems that allow you to make more use of key material, using cryptographically appropriate **pseudorandomness** and mathematically hard problems to prevent cryptanalysis of the ciphertext – but still being vulnerable to ‘implausibly-powerful’ brute force.

The key distribution problems are largely solved through **public-key** cryptography (next week’s lectures).

Block ciphers

Block cipher

For example, AES (advanced encryption standard).



Block ciphers

Once you've picked a key, a block cipher is a random-looking function from blocks to blocks.

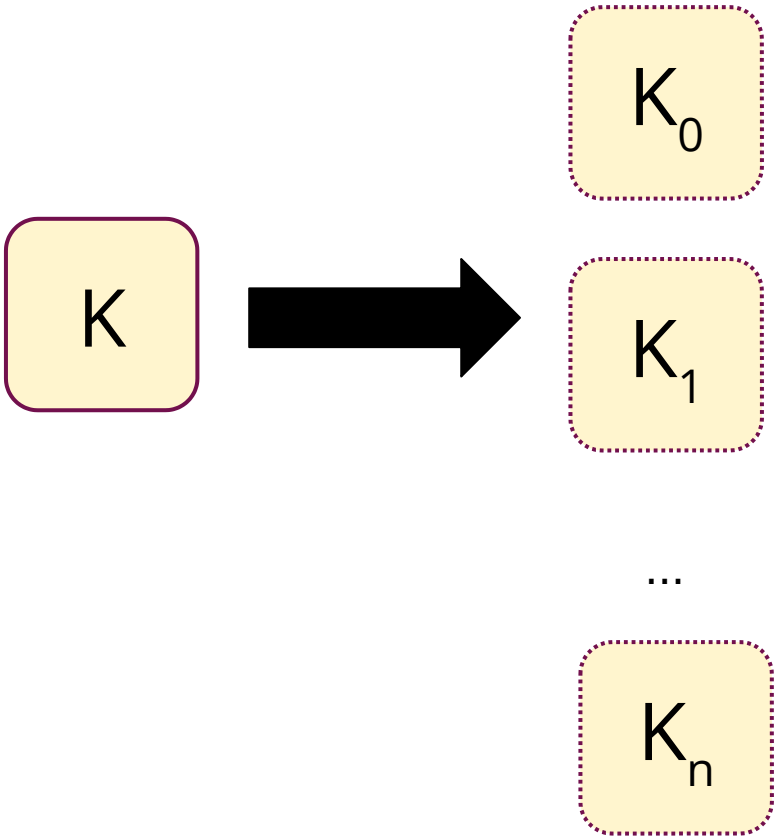
Just like the monoalphabetic cipher maps letters to letters.

It can't be completely random because you have to be able to undo it again to decrypt. The clever bit is making a usable but *random-looking* cipher.

Block ciphers: approaches

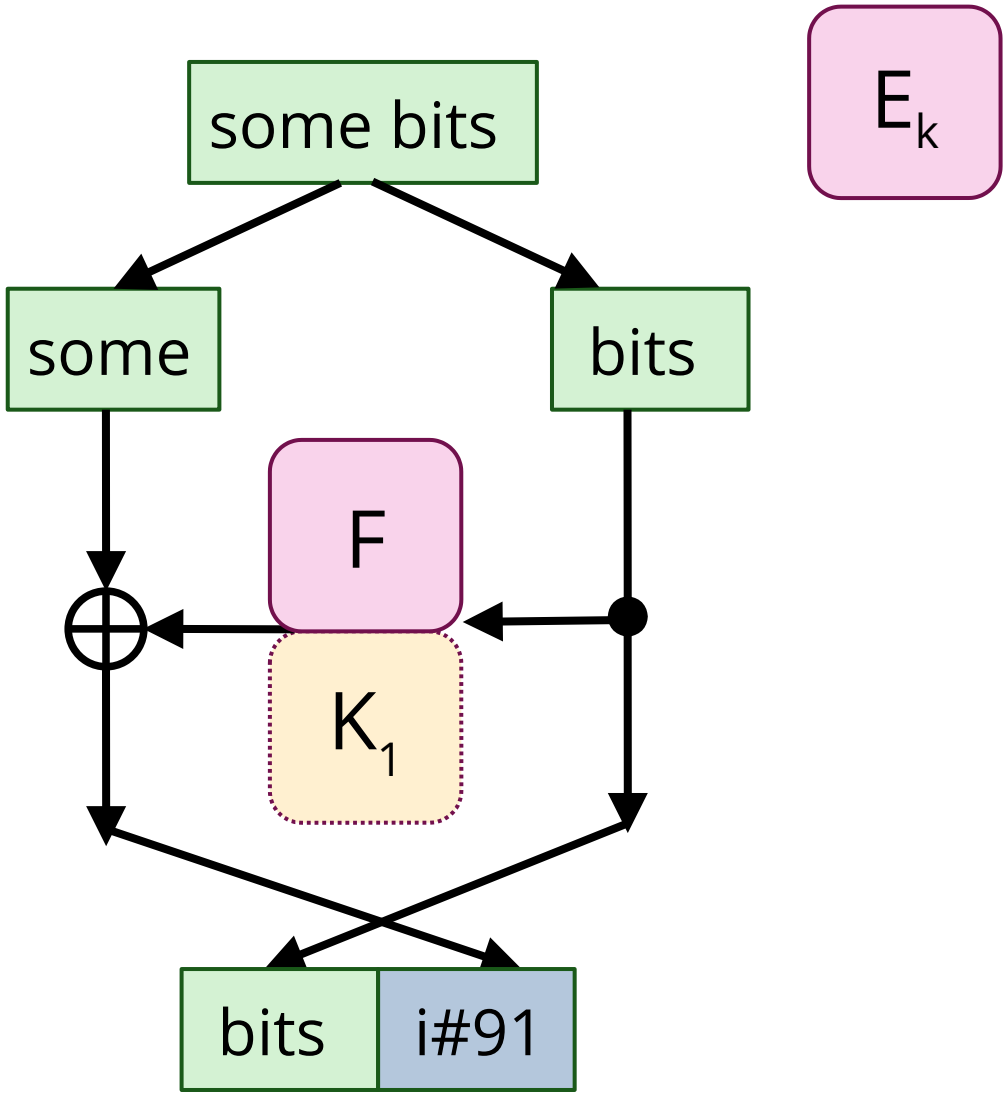
Feistel networks

E_k



Block ciphers: approaches

Feistel networks
for each subkey K_i in 0..N

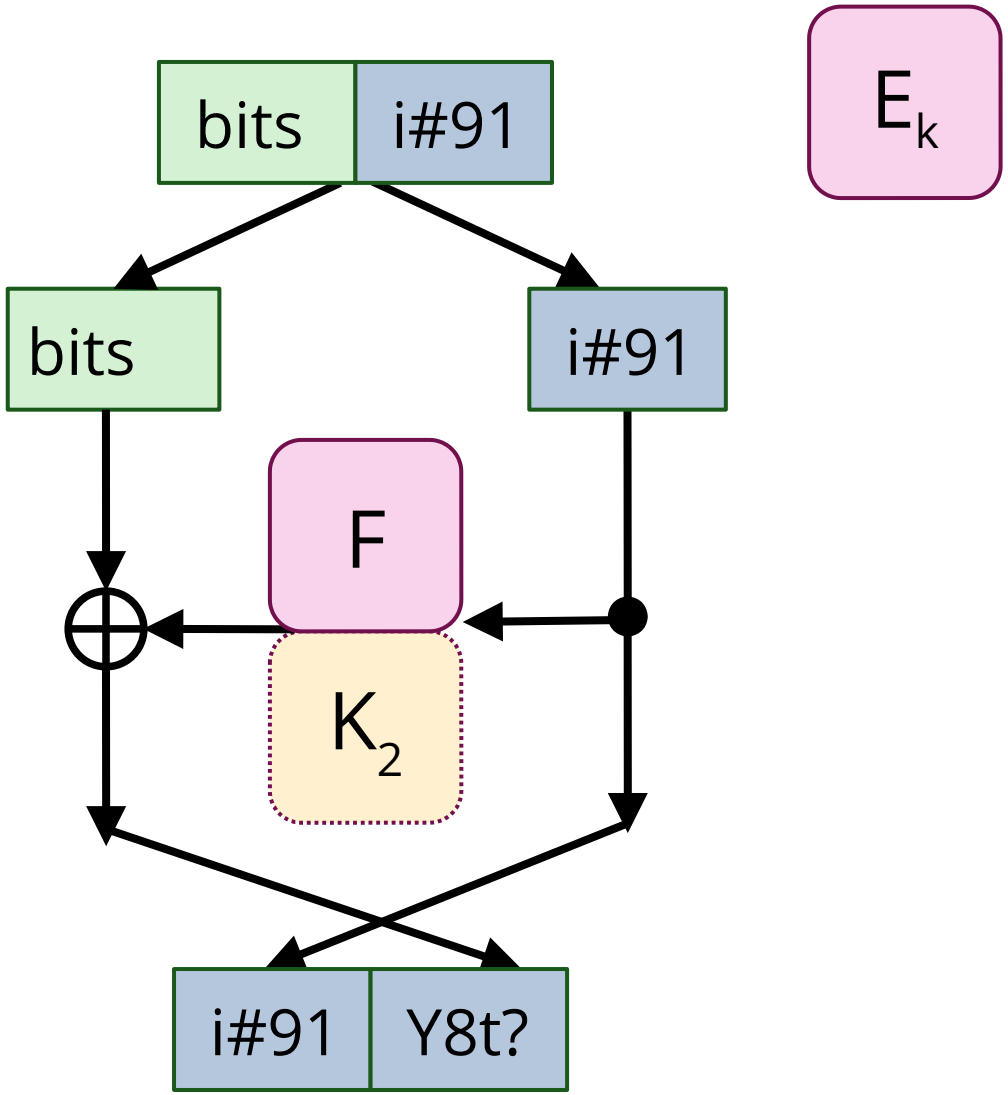


Block ciphers: approaches

Feistel networks
for each subkey K_i in 0..N

Used in DES, many other modern block ciphers

Easily reversible
(saves code/circuits)



Block ciphers: approaches

Substitution-permutation networks

Substitution: replace with something else

Permutation: shuffle the content around

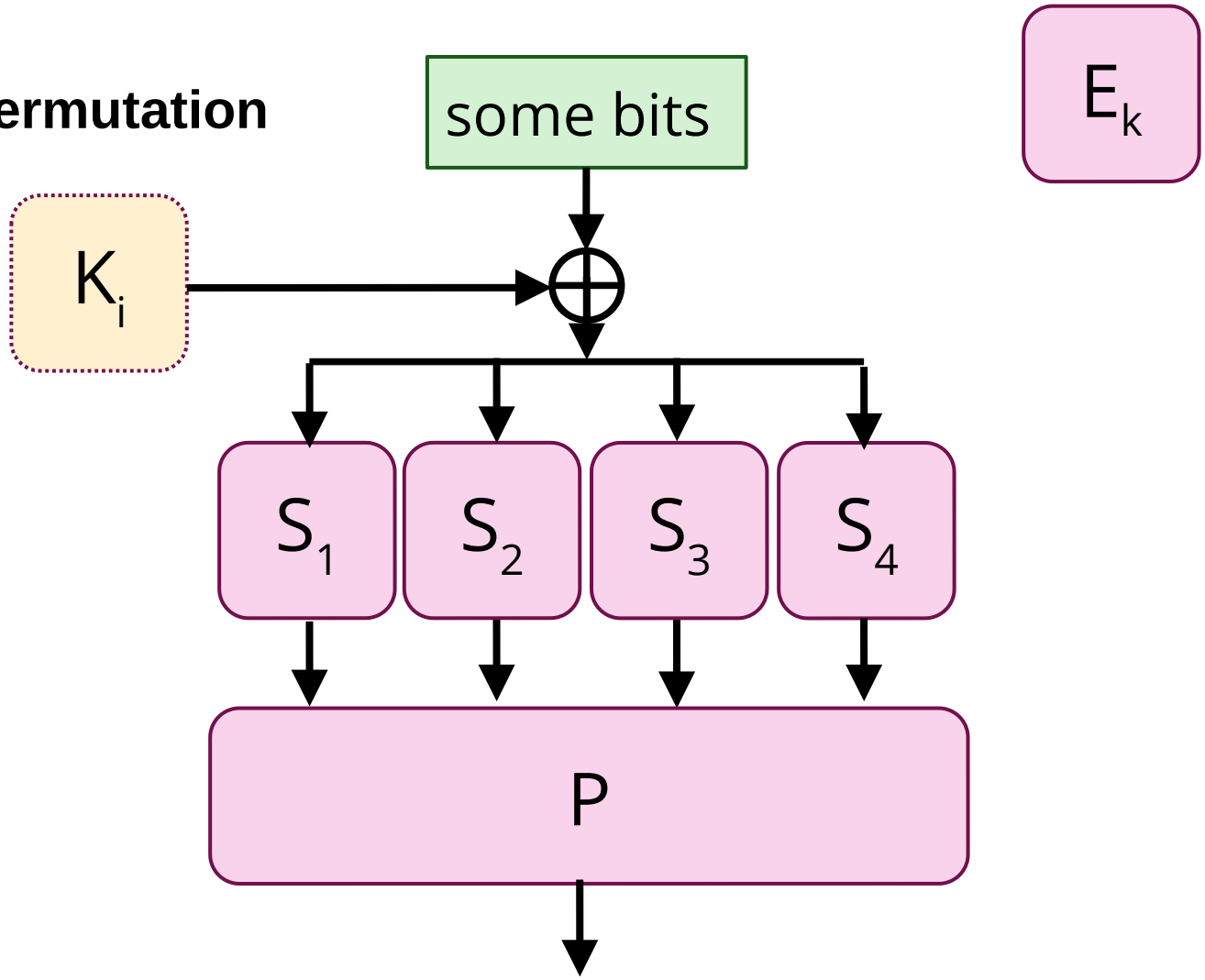
Aside: the Rail-Fence permutation cipher

```
killtheemperor


k   l   e   p   o
 i   t   e   e   r
  l   h   m   r


KLEPOITEERLHMR
```

Block ciphers: approaches

Substitution-permutation networks

As seen in AES



Modes of Operation

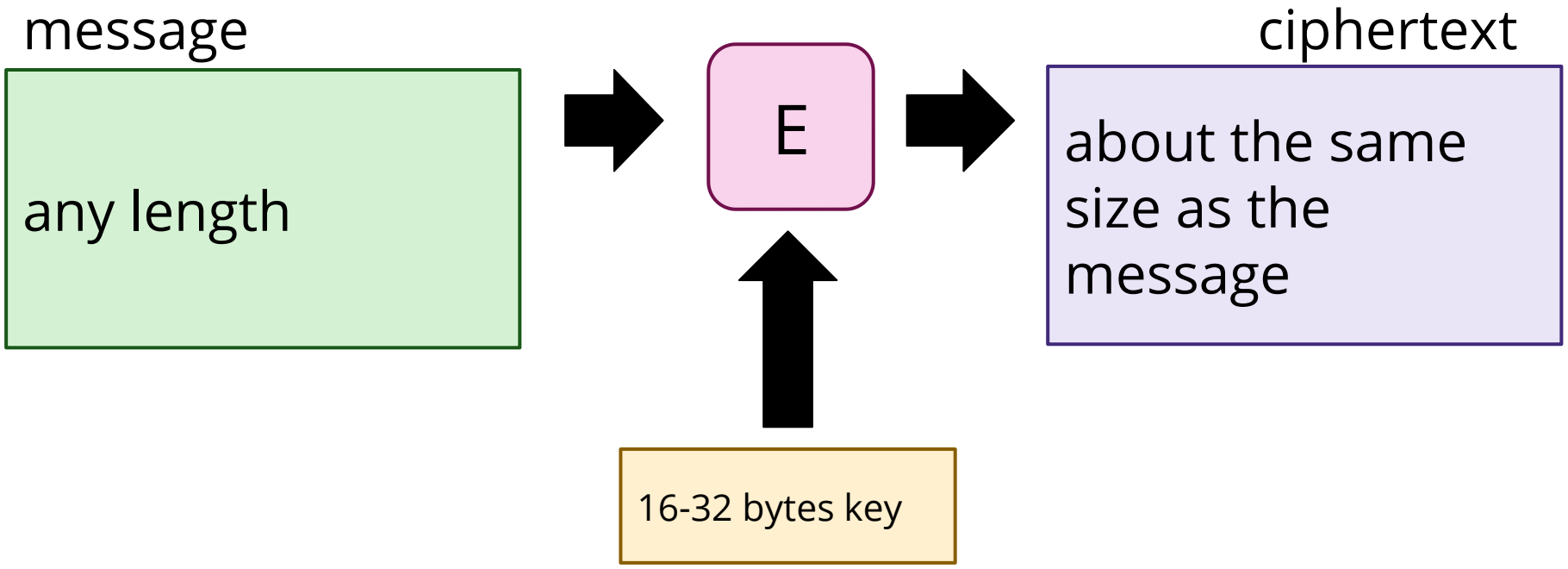
Block ciphers

AES: block size = 128 bits (16 bytes)

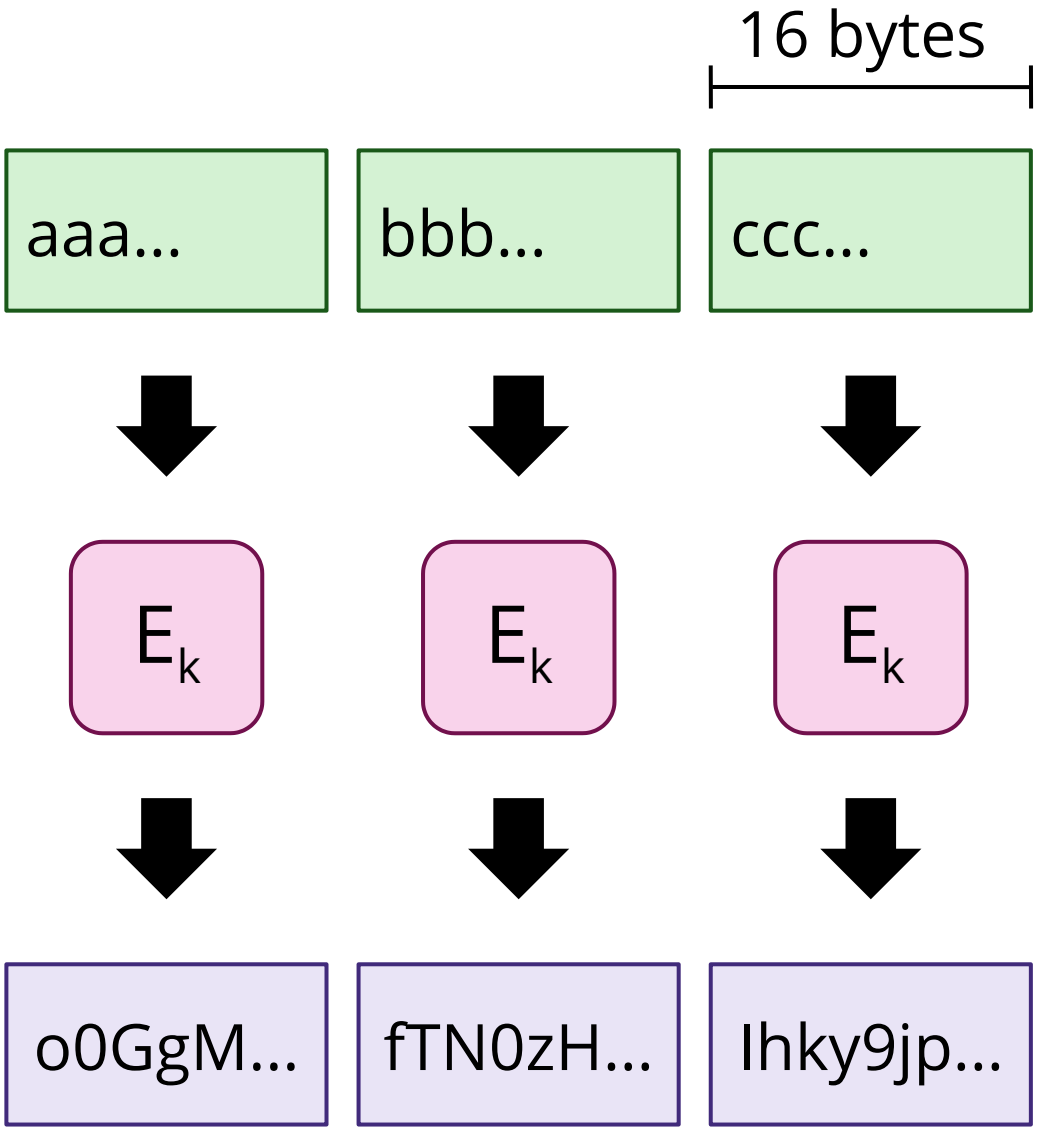
The problem with
simple ciphers i
s that they do n
ot hide patterns
...

?

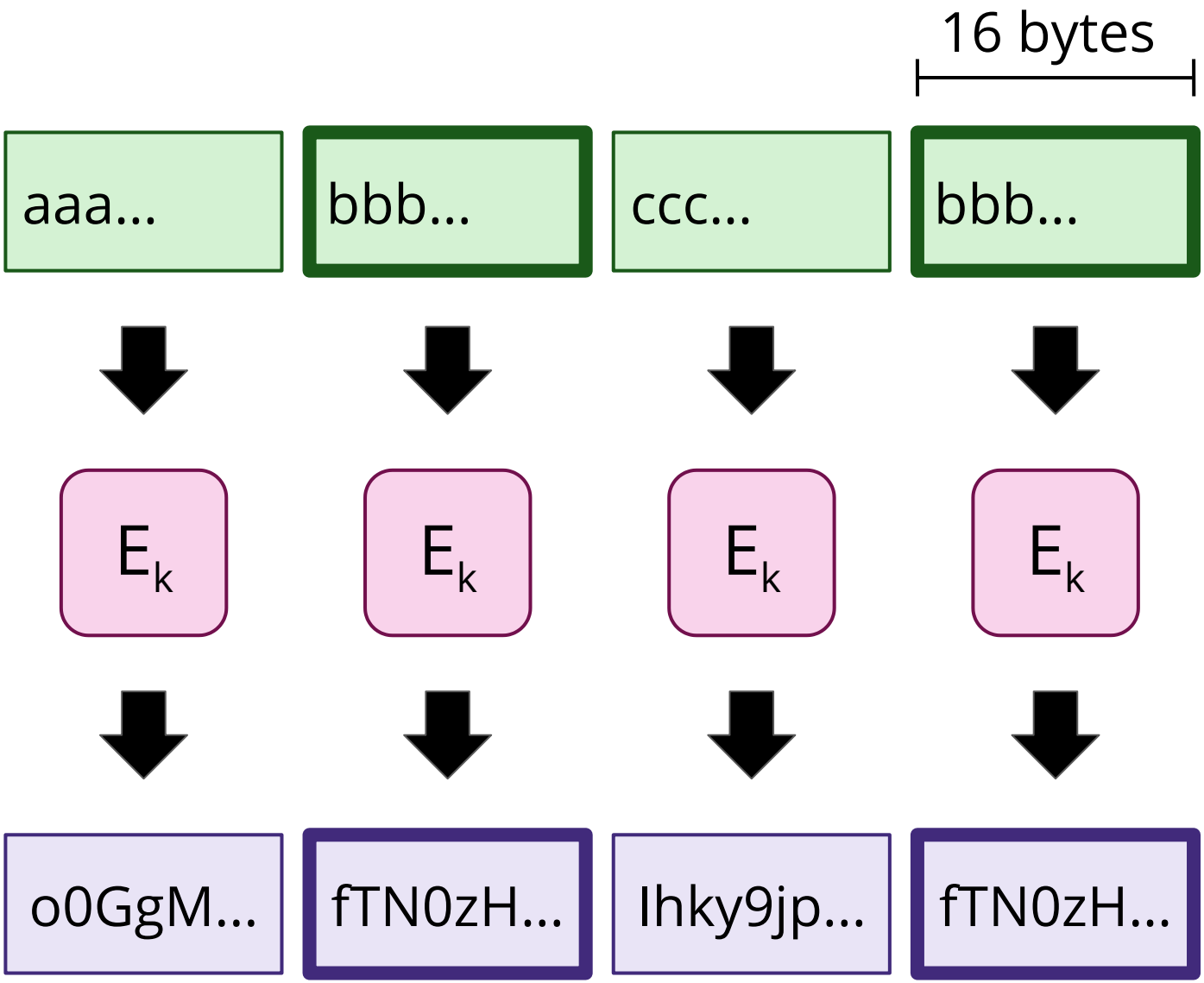
What we really want



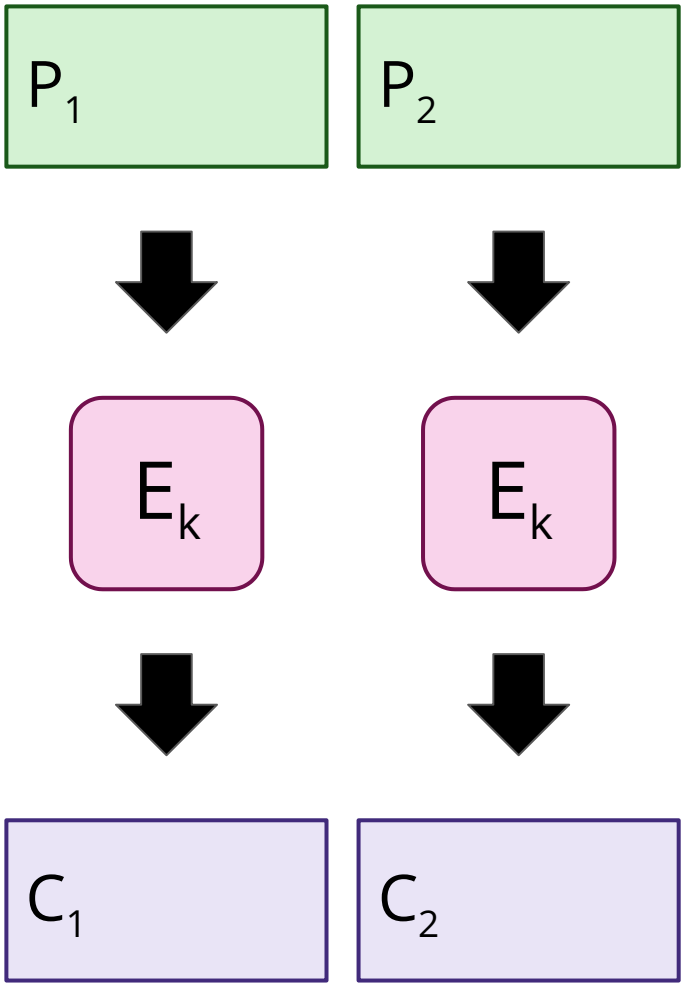
Bad idea!



Look - a pattern!



ECB mode



Using a block cipher like this is called ECB mode (electronic code book).

formula: $\mathbf{C_i = E_k(P_i)}$.

Never, ever do this!

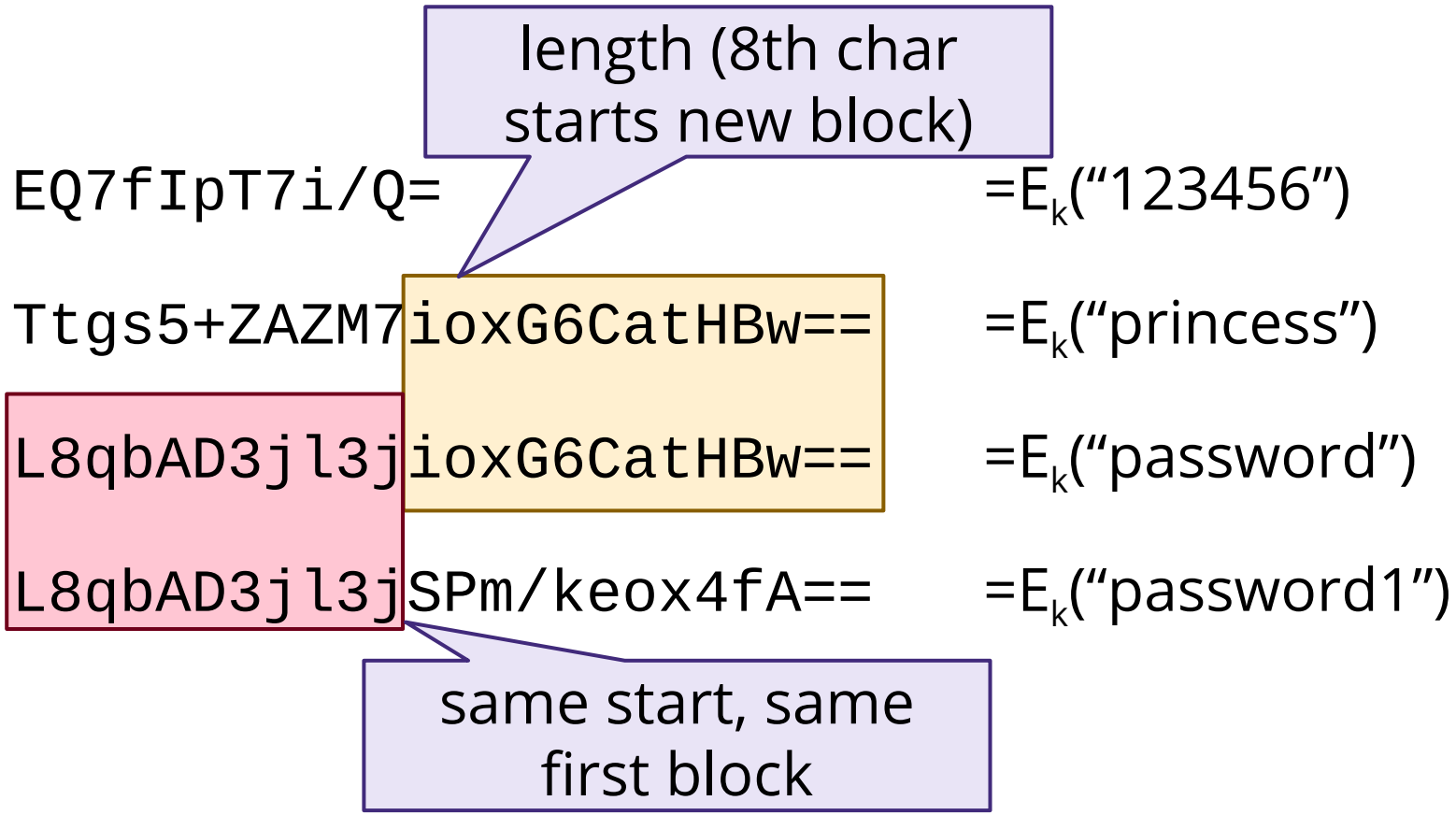
Real example

From the 2013 Adobe data breach:

`EQ7fIpT7i/Q=`	$=E_k$(“123456”)
`Ttgs5+ZAZM7ioxG6CatHBw==`	$=E_k$(“princess”)
`L8qbAD3jl3jioxG6CatHBw==`	$=E_k$(“password”)
`L8qbAD3jl3jSPm/keox4fA==`	$=E_k$(“password1”)

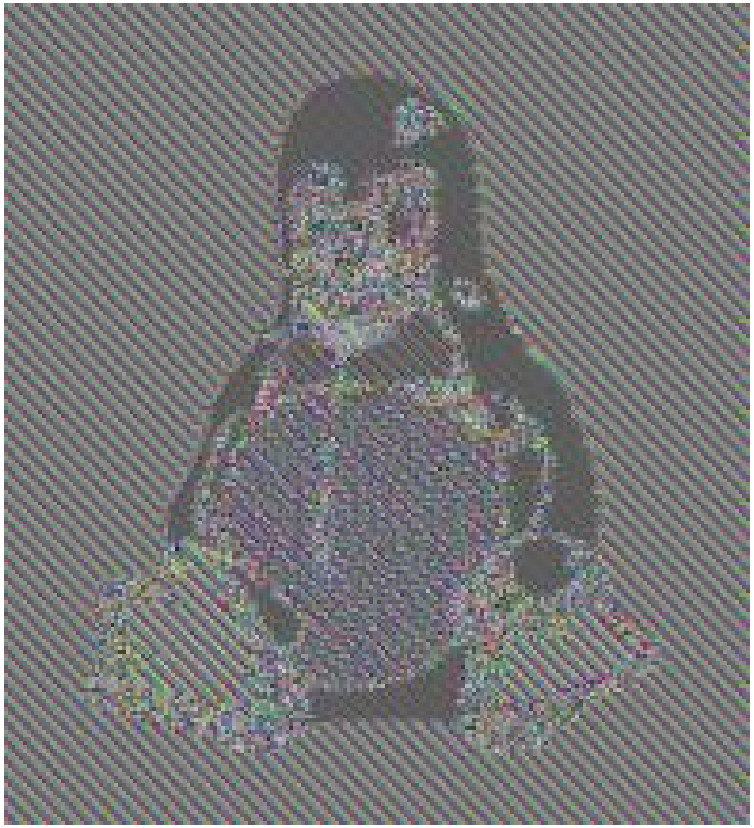
Real example

From the 2013 Adobe data breach:



Another example

Bitmap encrypted in ECB mode (source: wikipedia)



Modes of operation

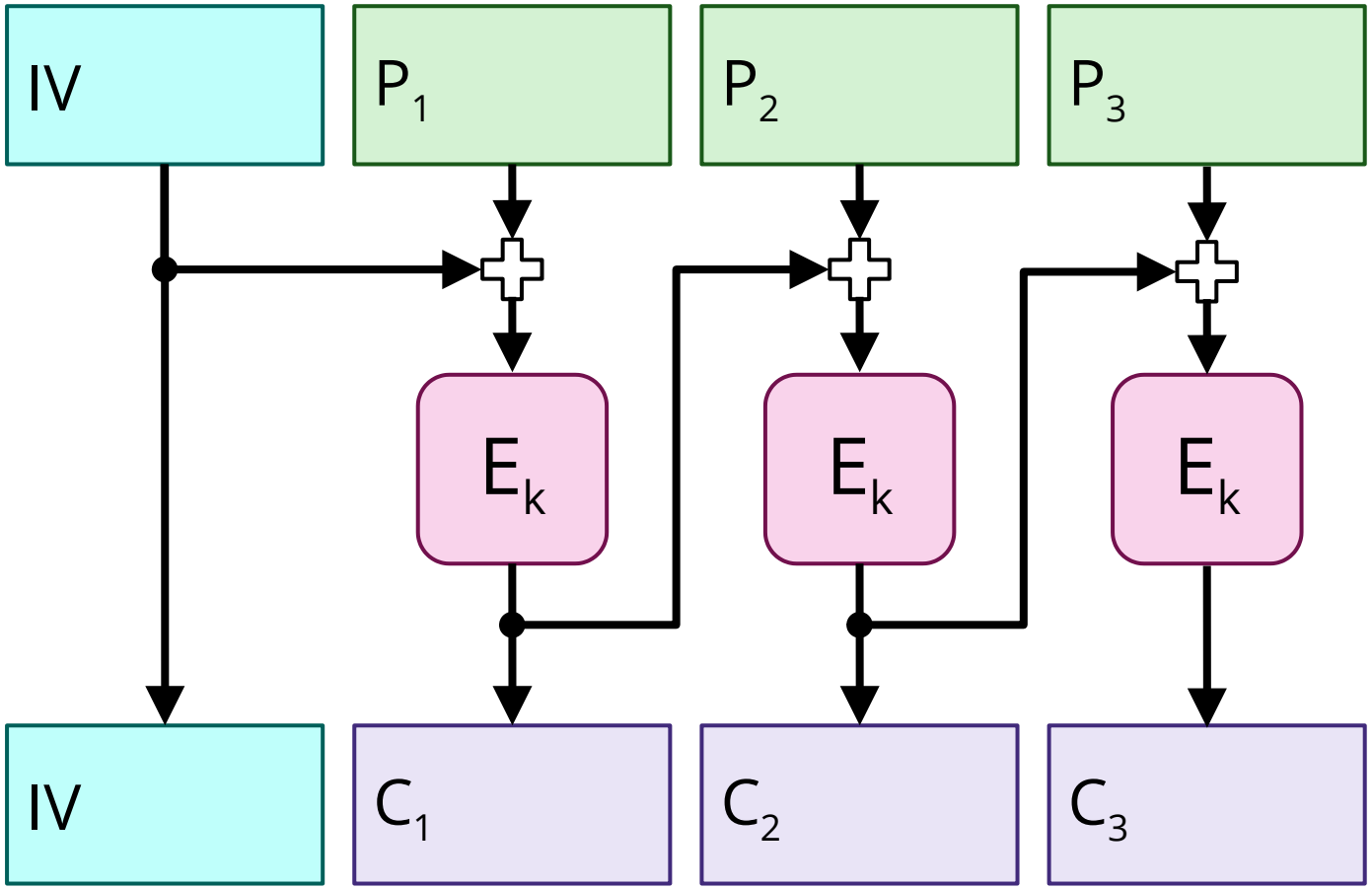
Exactly what makes a good mode is beyond this unit … that's what Crypto A/B is for.

The point is that we can prove that if the block cipher itself "looks random" on single blocks, a good mode "looks random" too.

Good modes include an extra, random block at the start called the *initialisation vector* so that you don't get the same thing if you encrypt the same message twice.

Example: CBC mode

CBC = cipher block chaining



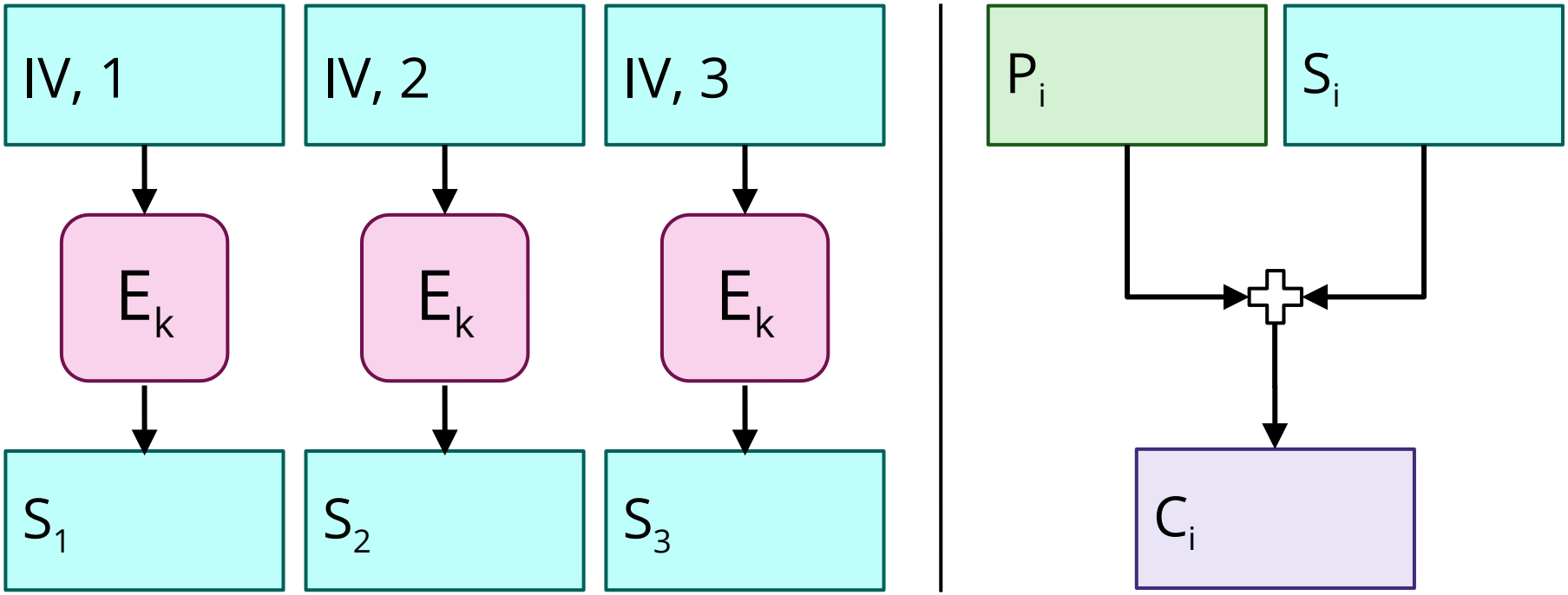
CBC mode

Let's try `IV = f`, addition a=0 … z=25 and key

```
ABCDE FGHIJ KLMNO PQRST UVWXY Z
SECUR ITYAB DFGHJ KLMNO PQVWX Z
```

```
 patterns and repetition must be hidden.

FPKUHFVAN HPN RQIGZYMPUY DWJC UX RZCIGO.
```

Example: CTR mode



counter mode (CTR) - stream cipher

$S_i = E_k(IV, i)$ **$C_i = P_i + S_i$**

A **(symmetric) encryption scheme** contains two algorithms E, D with **D(k, E(k, m)) = m**.

Without k, c = E(k, m) should tell you nothing about m.

A **block cipher** encrypts blocks of a fixed length. It needs to be used in a sensible **mode of operation** to encrypt longer messages.