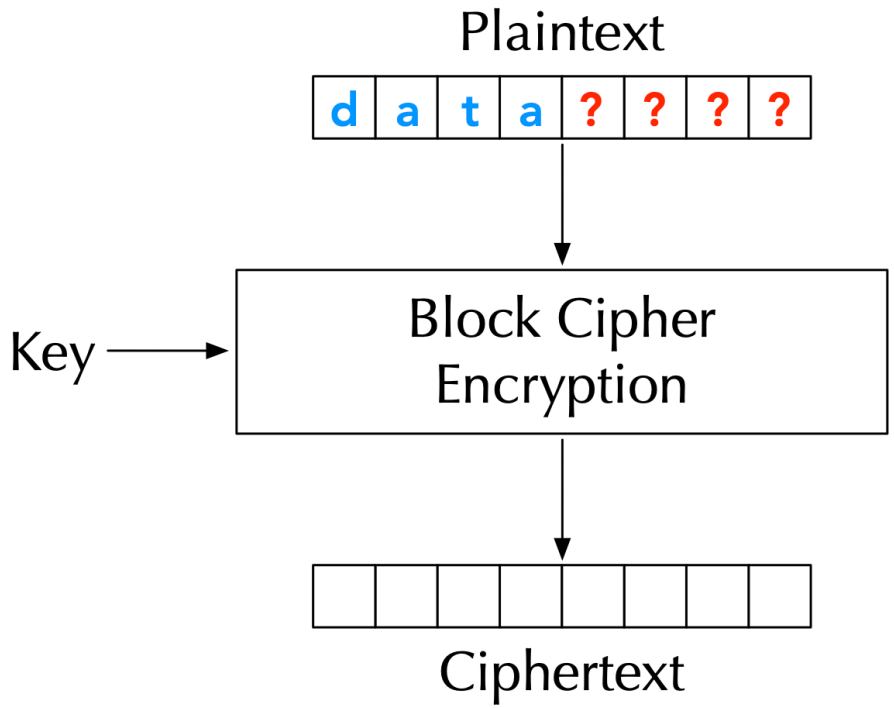
Padding

- Padding: *what is it?* and *why do we need it?*
- Experiments with openssl and padding

Padding

- Block cipher encryption modes divide plaintext into blocks; the size of each block should match the cipher's block size.
- No guarantee that the size of the last block matches the cipher's block size.
- The last block of the plaintext needs **padding**; i.e. before encryption, extra data needs to be added to the last block of the plaintext, so its size equals to the cipher's block size.
- Padding schemes (e.g., **PKCS#5**) need to clearly mark where the padding starts, **so decryption can remove the padded data.**



An Experiment w/ Padding (#1): What happens when data is smaller than the block size?

```
$ echo -n "123456789" > plain.txt
$ ls -ld plain.txt # plaintext is 9 bytes
-rw-rw-r-- 1 seed seed 9 Mar 18 20:49 plain.txt
```

Encrypt & examine size:

```
$ openssl enc -aes-128-cbc -e -in plain.txt -out cipher.bin \
    -K  00112233445566778899AABBCCDDEEFF \
    -iv 000102030405060708090A0B0C0D0E0F
```

```
$ ls -ld cipher.bin # ciphertext becomes 16 bytes!
-rw-rw-r-- 1 seed seed 16 Mar 18 20:49 cipher.bin
```

Decrypt & examine size:

```
$ openssl enc -aes-128-cbc -d -in cipher.bin -out plain2.txt \
    -K  00112233445566778899AABBCCDDEEFF \
    -iv 000102030405060708090A0B0C0D0E0F
```

```
$ ls -ld plain2.txt # decrypted ciphertext goes back to 9 bytes!
-rw-rw-r-- 1 seed seed 9 Mar 18 20:49 plain2.txt
```

An Experiment w/ Padding (#2):
How does decryption software know where padding starts?

Decrypt with removal of padding disabled (-nopad):

```
$ openssl enc -aes-128-cbc -d -in cipher.bin -out plain3.txt \
    -K  00112233445566778899AABBCCDDEEFF \
    -iv 000102030405060708090A0B0C0D0E0F -nopad
$ ls -ld plain3.txt
-rw-rw-r-- 1 seed seed 16 Mar 18 20:49 plain3.txt
```

Examine original and padded plaintext :

```
$ xxd -g 1 plain.txt
00000000: 31 32 33 34 35 36 37 38 39

$ xxd -g 1 plain3.txt
00000000: 31 32 33 34 35 36 37 38 39 07 07 07 07 07 07 07
```

7 bytes of 0x07 are added as the padding data

In general, for block size **B** and last block w/ **K** bytes,
B-K bytes of value **B-K** are added as the padding

An Experiment w/ Padding (#3 -> a special case)
What if the size of the plaintext is a multiple of the block size? And the last seven bytes are all 0x07?

```
$ xxd -g 1 plain3.txt # original plaintext (plain3.txt) is 16 bytes
00000000: 31 32 33 34 35 36 37 38 39 07 07 07 07 07 07 07
```

```
$ openssl enc -aes-128-cbc -e -in plain3.txt -out cipher3.bin \
    -K  00112233445566778899AABBCCDDEEFF \
    -iv 000102030405060708090A0B0C0D0E0F
$ openssl enc -aes-128-cbc -d -in cipher3.bin -out plain3_new.txt \
    -K  00112233445566778899AABBCCDDEEFF \
    -iv 000102030405060708090A0B0C0D0E0F -nopad
```

```
$ ls -ld cipher3.bin plain3_new.txt
-rw-rw-r-- 1 seed seed 32 Mar 18 21:07 cipher3.bin
-rw-rw-r-- 1 seed seed 32 Mar 18 21:07 plain3_new.txt
# decrypted output (plain3_new.txt) is 32 bytes -> A new block is added as padding!
```

```
$ xxd -g 1 plain3_new.txt
00000000: 31 32 33 34 35 36 37 38 39 07 07 07 07 07 07 07
00000010: 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
```

In PKCS#5, if the input length is already an exact multiple of the block size B, then B bytes of value B are added as the padding.