# THE DYNAMIC LINKER

## (A.K.A. LOADER)

CALLED AT

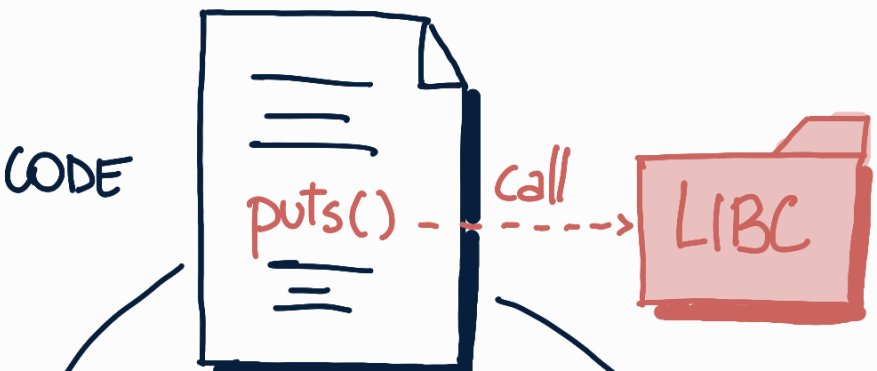
- **STARTUP**: Loads all required libraries (.so) transitively
- **RUNTIME**: The first time a lib functions gets called

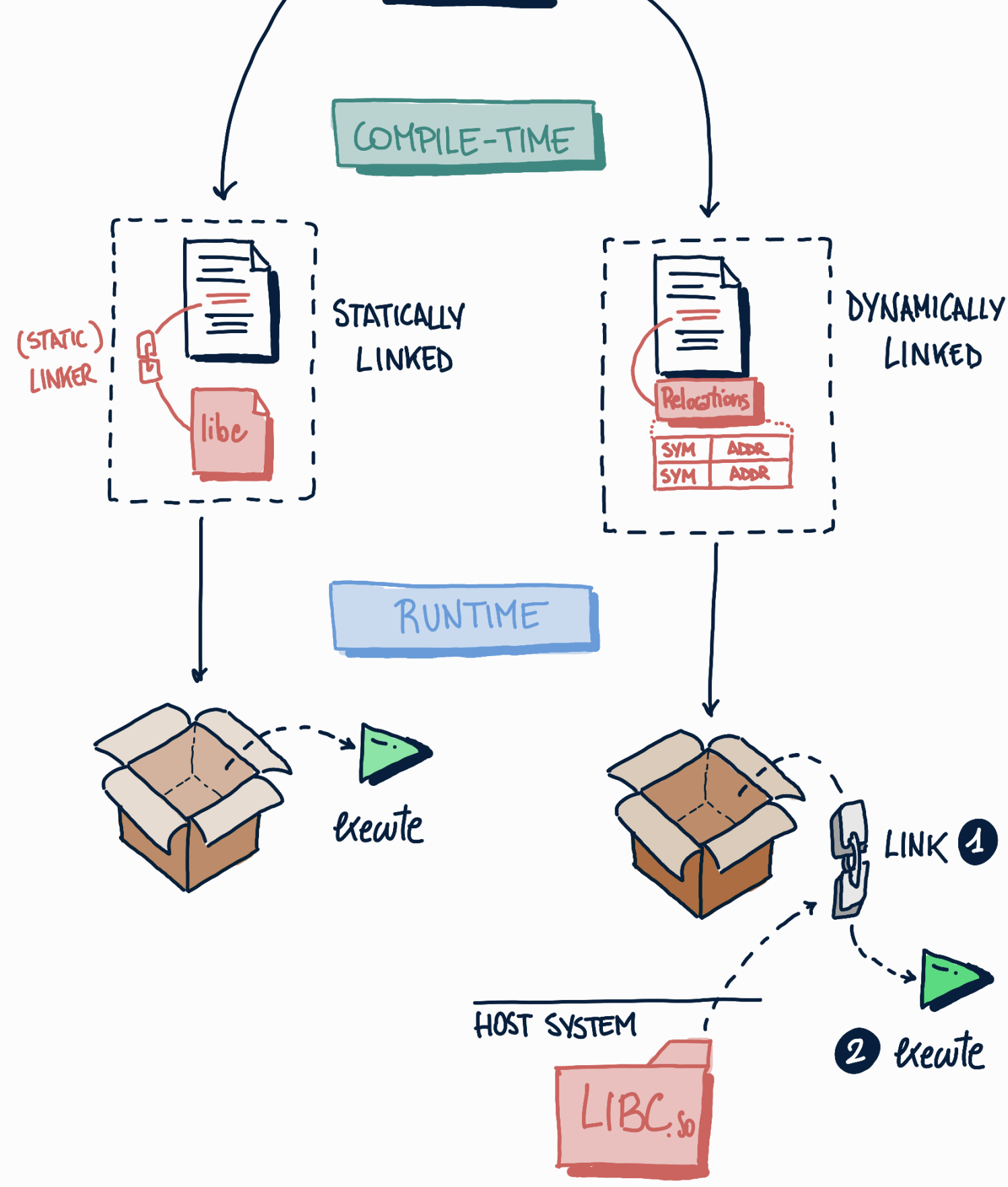
SECTIONS

- **GOT**: Global Offset Table
  *Tells the program where to find external library functions*
- **PLT**: Procedure Linkage Table
  *Stubs for library functions*

# LINKING

## (STATIC VS DYNAMIC)



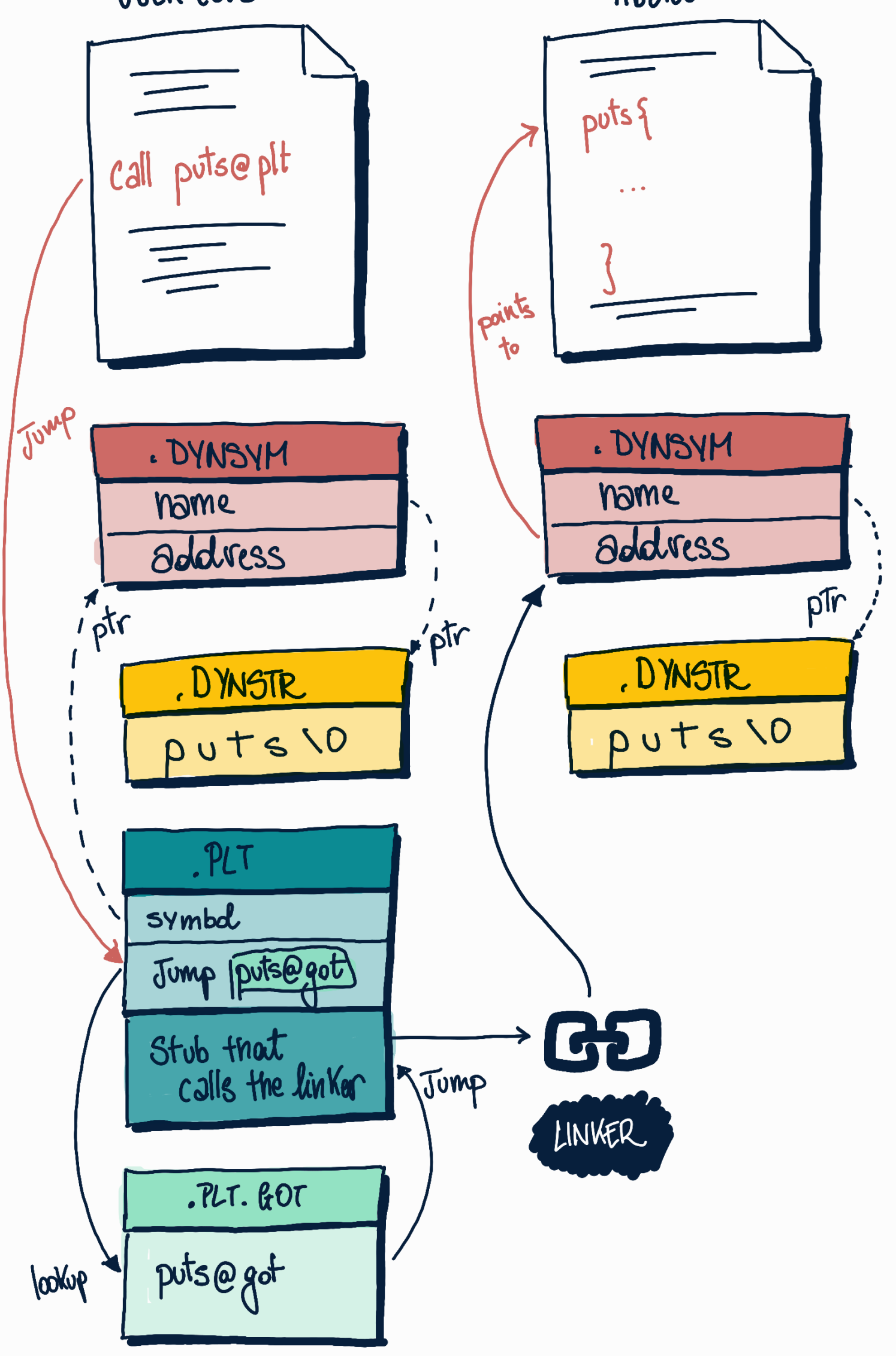



# DYNAMIC LINKING

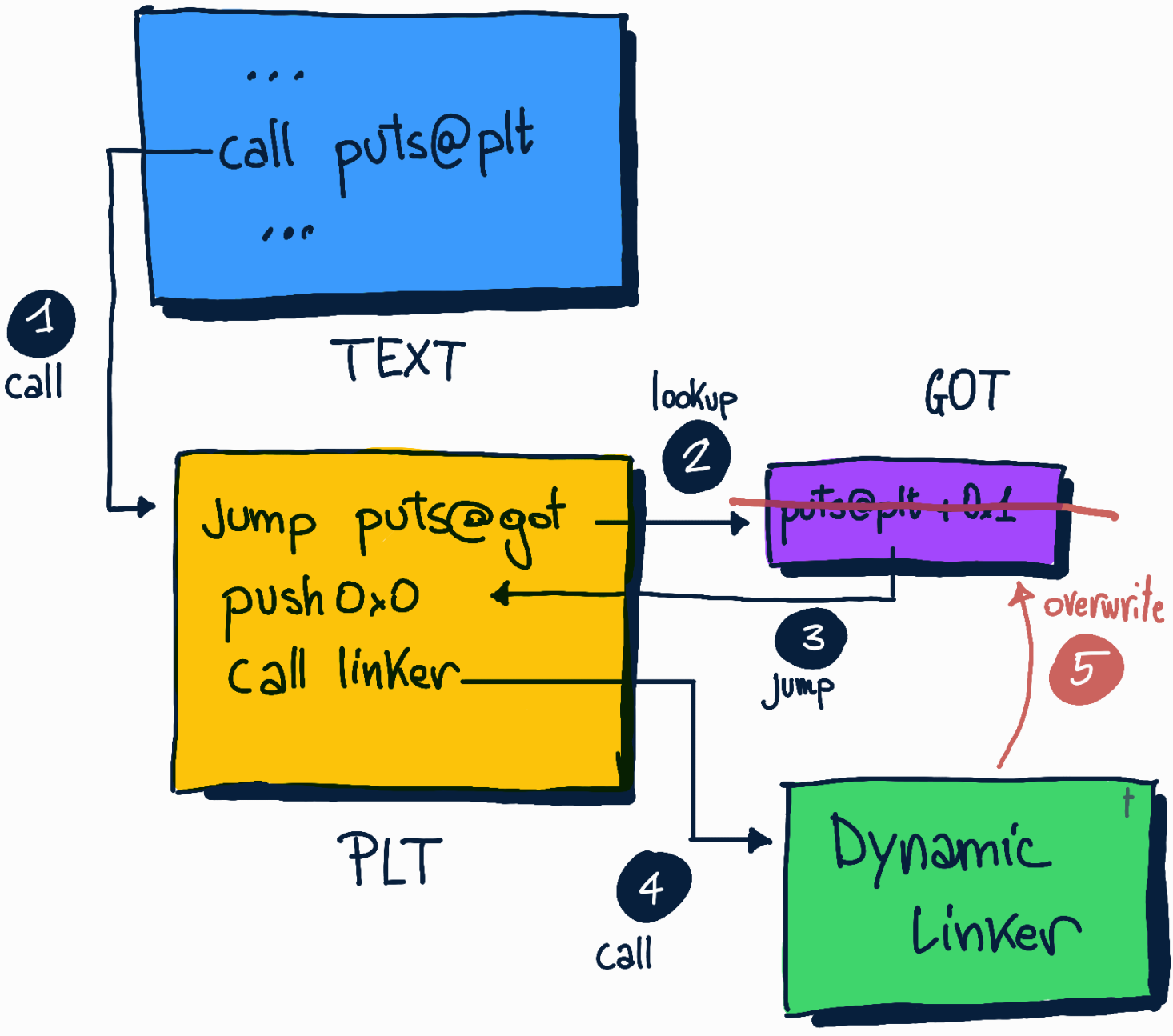
## (ELF SECTIONS)

USER CODE

libc.so


call puts@plt
puts {
...
}
points to
Jump
.DYNSYM
name
address
ptr
ptr
.DYNSTR
puts \0
.DYNSYM
name
address
ptr
.DYNSTR
puts \0
.PLT
symbol
Jump puts@got
Stub that calls the linker
Jump
LINKER
.PLT.GOT
puts@got
lookup

# RELOCATIONS
## (LAZY LOADING)




1. User code calls PLT
2. PLT jumps to pointer in the GOT
3. The first time, the GOT points to a stub in the PLT

4. The stub calls the linker

5. The linker loads the real address in the GOT
   ↳ next time, the PLT will jump directly to the lib

## RELRO

(RELOCATIONS READ-ONLY)

- PARTIAL `-z,relro`
  ↳ some sections marked read-only
- FULL `-z,relro -z,now`
  ↳ no lazy loading