Steps in the attack:

1. Control-flow hijack: overwriting a code pointer e.g., RA
2. Payload: (a) injected code
   (b) existing code (e.g. return-to-libc or ROP)
3. Bypass exploit mitigation techniques e.g. Stack canary
   (a) Partial overwrite
   (b) Double pointer attack

Exploit Goal: Increase attacker's capability (privilege)

Starting point: no login or ability to run code on victim
(but is able to connect and send data to victim)
↓ escalate
run code (as normal user)
↓
run code as root

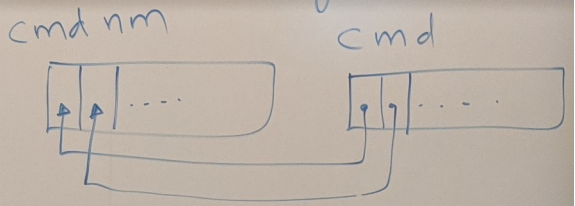
Partial overwrite:

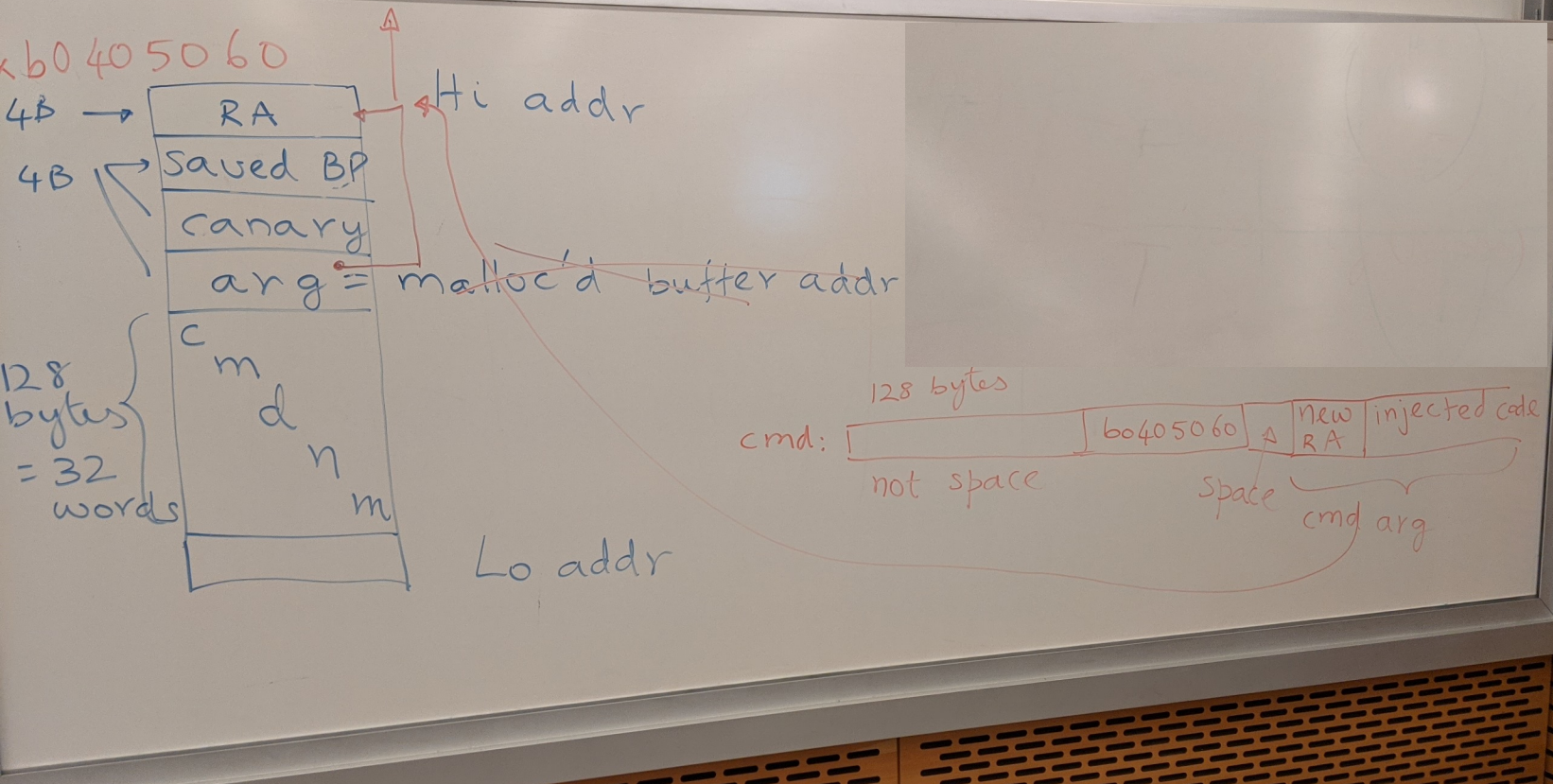
- Brute-forcing canary requires $\sim 2^{32}$ attempts.
- "Smart" search
  - overwrite the first reachable byte

1. wait for a client to connect
2. accept connection
3. fork a child process to handle connection

Double pointer overwrite: "skip past the canary"

```
void parseCmd(char *cmd) {
    char *arg = malloc(4096);
    char cmdnm[128];
    int i=0;
    while (!isspace(*cmd)) cmdnm[i++] = *cmd++;
    cmd++; // skip space
    ... copy (rest of) cmd into arg
```




0x b0 40 50 60
4B
RA
4B
Saved BP
canary
arg =
malloc'd buffer addr
Hi addr
128 bytes = 32 words
c
m
d
n
m
Lo addr
128 bytes
cmd:
b0405060
new RA
injected code
not space
space
cmd arg