- iščemo ravnovesje med tem, da uporabljamo defenyivno programiranje (pohendlamo vse napake) in tem, da je stvar berljive, vzdržljiva
- včasih moramo uporabljati tujo kodo - kako jo narediti varno:
  - da "ščitimo kodo pred samo s sabo"; povemo, kako naj deluje in če od tega odstopa, se terminira
  - Landlock in seccomp
  - programe od drugih pustimo pri miru in pišemo okoli njih, da ne morejo delati bedarij

def_5:

- `fgets` prebere več, kot je velikost bufferja
- v exploit.txt smo prepisali return address z naslovom od `win`
- TODO: Protect program from running external programs
- seccomp:
  - INIT - definiraš nov ruleset, poveš kaj se zgodi, če se pravila kršijo
  - nato povemo, kaj dovolimo - seccomp_rule_add
  - seccomp omejuje sistemske klice (syscall)
  - dovoliti rabimo write, read, open, close ... - na pamet težko vemo, kaj vse je noter, začnemo s tistim, kar mislimo, da imamo
  - `./protect ./cat cat.c`
  - `./protect ./cat explot.txt`
  - `strace` izpisuje vse syscalle
  - `strace ./protect ./cat cat.c`
  - seccompa ne moreš kar dati na en drug program, ampak je narejen zato, da programi samega sebe omejujejo
  - imamo ogromno sistemskih klicev, ali je v redu, da vse to samo dovolimo?
    - moramo vedeti, do česa vsega rabi imeti program dostop
  - stack, heap sta read, write; text je read, execute
  - pri nas ni v redu, da vse dovolimo - lahko npr. nastavimo omejitve za open, ampak če ne omejimo na enak način readlinkat in openat, potem napadalec lahko vseeno pride mimo omejitve
  - SELinux - profili za programe, ko probaš zalaufat nekaj; moraš dati profil kaj sme delati, drugače bo vse denied; bolj advanced verzija seccomp:
    - če imaš npr. omogočen path treversal, to SELinux reši tako, da u eksplicitno poveš, kam vse sme
    - ne pišeš na roko vse dovoljenj - imaš stanja enabled, disabled, enabled ampak se ne terminira, ampak samo izpisuje

        - pogledaš error log od programa in potem sparšaš error log in avtomatsko napišeš pravila za ta program
        - `strace ./cat cat.c` - vidimo, kaj vse se uporablja
        - `strace ./cat cat.c 2&> seccomp.out`
    - ni nujno, da delamo v C-ju, včasih se seccomp da naloadati iz JSON-a in se potem iz tega spišejo pravila
    - naš exploit ne bo več delal, ker uporabljamo `system`, ki še neke stvari zraven naredi
    - če bi `system` spremenili v `execv`, bi lahko prižgali shell, tako da naša omejitev ni popolna
    - `execv` je pri nas najbolj problematičen, ker se lahko kličejo poljubni programi:
        - lahko samo omejimo argumente, ki jih `execve` lahko sprejme
        - želimo omejiti prvi argumente
        - seccomp nima podpore za primerjat stringe, torej ne moremo dati samo `SCMP_A0(SCMP_CMP_EQ, (long)argv[1])`, ker bo to samo naslov prvega znaka primerjalo - pri našem primeru bo to vseeno v redu, ker nato kličemo `execv` s točno tem stringom, ki je na `argv[1]`
        - še vedno je nevarno, če napadalec spremeni vrednost `argv[1]` - se da, ampak se bo moral napadalec fajn potruditi
        - če hočemo primerjati stringe, rabimo bolj high level rešitve, npr. AppArmor
    - dobro bi bilo, če razumemo vse rule, ki smo jih allowali, da točno vemo, do česa lahko programi dostopajo:
        - npr. pri `readlinkat` bi lahko omejili argumente, ker tako, kot imamo zdaj, lahko nekje naprej beremo datoteke, ki jih nismo hoteli pustiti
- landlock:
    - omejitve s stališča file systema in networka; omejitve do določenega resursa (omejitve na nivoju resourcov) - korak višje od seccomp
    - https://man7.org/linux/man-pages/man2/landlock_add_rule.2.html
    - zakaj rabimo `landlock_ruleset_attr` in `landlock_path_beneath_attr`:
        - prva je kaj bomo sploh omejevali - bolj pravilno je, da pri prvi omejimo vse, pri drugi pa samo povemo, kaj dovolimo delati - v našem primeru hočemo dovoliti read in execute v current direktoriju (podobno kot pri seccomp, najprej ničesar ne dovolimo in potem eksplicitno povemo, kaj bomo dovolili)
        - `O_PATH` pomeni, da naj bo to direktorij
    - na koncu closamo file descriptor
    - moramo najprej porunnati landlock in potem šele seccomp, da nam seccomp ne onemogoči loadanja landlocka? - vrstni red ni pomemben?
    - rule se da posebej pisati in applyjati - mu eksplicitno samo en file/samo en direktorij dovolimo

- smo naredilo, da se eksplicitno lahko samo `cat.c` poda kot argument; če damo za argument `/etc/passwd`, bo crashalo, ker nima dovoljenja