Семинар 16: управление процессами

9 апреля, 2020

Акт 1: mini GDB

ptrace

- ptrace — интерфейс ядра, предоставляющий возможность наблюдать и контролировать исполнение другого процесса
- tracer — это тот, *кто* контролирует выполнение
- tracee — это тот, *кого* контролируют

ptrace: attaching

```
pid_t pid = fork();
if (pid == 0) {
    ptrace(PTRACE_TRACEME, 0, NULL, NULL);
    execve(...);
} else {
    for (;;) {
        int status = 0;
        waitpid(pid, &status, 0);
        if (WIFSTOPPED(status)) {
            // процесс остановился, из-за ptrace
        } else if (WIFSIGNALED(status)) {
            // процесс был просигнализирован
        } else if (WIFEXITED(status)) {
            // процесс завершился
        } else {
            // никогда не выполняется
        }
    }
}
```

Как ставятся брейпоинты?

- В Linux поддерживается соглашение, что `int 3` будет вызывать **SIGTRAP**
- Команды PTRACE_PEEKUSER и PTRACE_POKEUSER используются, чтобы читать/писать машинные слова в child-процессе
- Идея — давайте вместо оригинальной инструкции запишем `int 3`, когда процесс остановится, посмотрим где остановились, восстановим инструкцию и повторим её

Как повторить инструкцию?

- В прерываниях процессора это делается автоматически (на самом деле, зависит от типа прерывания)
- Нам придётся вручную менять RIP для этого
- С помощью PTRACE_GETREGS и PTRACE_SETREGS можно изменять регистры

ptrace: struct pt_regs

```
struct pt_regs {
    unsigned long r15;
    unsigned long r14;
    unsigned long r13;
    unsigned long r12;

    /* ... */

    unsigned long rax;
    unsigned long orig_rax;

    /* ... */

    unsigned long rip;
    unsigned long cs;
    unsigned long eflags;
    unsigned long rbp;
    unsigned long rsp;
    unsigned long ss;
};
```

Акт 2: mini docker

Gratias ago!