EMILIA STARCZYK
249005

SPRAWOZDANIE Z ZADAŃ 5,6,7

SCR - SYSTEMY OPERACYJNE

MGR INŻ. ARKADIUSZ MIELCZAREK

17.11.2020

Zadanie 5

Czytając dokumentację polecenia strace na Linuxie zwróciłam uwagę na kilka poszczególnych funkcji:

1) *Strace -i polecenie*
   Polecenie to wyświetla wskaźnik na daną instrukcję przed każdym wypisanym poleceniem systemowym.
2) *Strace – o name polecenie*
   Polecenie pozwala na wysłanie wszystkich informacji, które zostałyby wyświetlone w terminalu, do pliku o nazwie które wpiszemy w argument *name*.
3) *Strace -r polecenie*
   Polecenie to wyświetla różnicę czasową między kolejnymi wywołaniami systemowymi.
4) *Strace -t polecenie*
   Polecenie to pozwala wyświetlić przed każdym wywołaniem informacje o czasie wywołania.
5) *Strace -T polecenie*
   Polecenie pokazuje czas spędzony na danym wywołaniu systemowym od początku jego uruchomienia do zakończenia.
6) *Strace -c polecenie*
   Polecenie pozwala na wyświetlenie sumarycznych informacji w tabeli przy zakończeniu działania programu.
7) *Strace -e trace=open polecenie*
   Polecenie to ogranicza ilość wynikowych informacji. Wyświetlane są tylko logi, które ograniczają się do jednego polecenia – open(). Zamiast open możemy wpisać kilka innych opcji, przykładowo:
   - close,
   - read,
   - write,
   - unlink,
   - stat,
   - signal.

   Aby wyświetlać logi zawierające kilka różnych poleceń wpisujemy je po przecinku: -e trace=open,read….
8) *Strace -s sort_by polecenie*
   Polecenie pozwala na posortowanie informacji uzyskanych z samego polecenia strace.
   Pod sort_by możemy podstawić kilka rzeczy, przykładowo:
   - time,
   - max-time,
   - min-time,
   - calls,
   - erros,
   - name.

   Automatycznie ustawioną opcją jest time.
9) *Sudo strace -p pid*
   Polecenie pozwala na śledzenie kolejnych wywołań związanych z procesem o numerze PID podanym w argumencie *pid.*

Zadanie 6

Zadanie 6 składało się z 4 podpunktów, które wykonywałam osobno.

- Przeanalizuj wykonanie programu wyświetlającego napis Hello world na ekranie.
  Aby wykonać to polecenie utworzyłam program zadanie6.c i skompilowałam. Uruchomiłam program z poleceniem *strace -o zad6_out1 ./zadanie6.* Na czerwono dopisałam komentarze do odpowiednich fragmentów.

Oto zawartość pliku zad6_out1:
execve("./zadanie6", ["./zadanie6"], 0x7fffe81c1750 /* 48 vars */) = 0 uruchomienie programu
brk(NULL) = 0x56478a2fb000 zapytanie o miejsce zakończenia stosu pamięci – związane z malloc()
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffe97ed6020) = -1 EINVAL (Zły argument) ustawienie architektury procesu/wątku
access("/etc/ld.so.preload", R_OK) = -1 ENOENT (Nie ma takiego pliku ani katalogu) próba dostania się do biblioteki współdzielonej o podanej ścieżce i sprawdzenie czy jest pozwolenie do czytania. W_OK - pisanie X_OK wykonanie
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3 otwórz cache o ścieżce z argumentu drugiego na miejscu AT_FDCWD z parametrami z argumentu 3, zwraca numer file deskryptora
fstat(3, {st_mode=S_IFREG|0644, st_size=75219, ...}) = 0 zebranie statystyk o pliku określonym file descriptorem
mmap(NULL, 75219, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fd635146000
close(3) = 0 zamknięcie
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) =3 otworzenie biblioteki języka c
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360q\2\0\0\0\0\0"..., 832) = 832 przeczytanie zawartości
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784 czytanie deskryptorów plików zwraca liczbę bajtów z fd od początku pliku
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0", 32, 848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355Y\377\t\334"..., 68, 880) = 68
fstat(3, {st_mode=S_IFREG|0755, st_size=2029224, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fd635144000
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0", 32, 848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355Y\377\t\334"..., 68, 880) = 68
mmap(NULL, 2036952, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fd634f52000 mapowanie funkcji do pamięci programu
mprotect(0x7fd634f77000, 1847296, PROT_NONE) = 0
mmap(0x7fd634f77000, 1540096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x25000) = 0x7fd634f77000
mmap(0x7fd6350ef000, 303104, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x19d000) = 0x7fd6350ef000
mmap(0x7fd63513a000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7fd63513a000
mmap(0x7fd635140000, 13528, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fd635140000
close(3) = 0
arch_prctl(ARCH_SET_FS, 0x7fd635145540) = 0
mprotect(0x7fd63513a000, 12288, PROT_READ) = 0
mprotect(0x564788fff000, 4096, PROT_READ) = 0
mprotect(0x7fd635186000, 4096, PROT_READ) = 0
munmap(0x7fd635146000, 75219) = 0
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) = 0 sprawdzenie informacji wyjścia programu – tutaj o terminalu
brk(NULL) = 0x56478a2fb000 zapytanie o miejsce zakończenia stosu pamięci – malloc()
brk(0x56478a31c000) = 0x56478a31c000
write(1, "Hello world", 11) = 11 polecenie wypisania, które w programie napisane jest jako printf.

exit_group(0) = ? zakończenie działania
+++ exited with 0 +++

Całe wykonanie programu to tak naprawdę ostatnie linijki wyrzucone przez polecenie strace.
write(1, "Hello world", 11) = 11 Wypisanie 11 znaków na terminal(file descriptor 1)

- Wykorzystaj program strace do znalezienia wszystkich plików konfiguracyjnych, jakie powłoka próbuje odczytać przy starcie.
  Poleceniem *strace -e trace=read* wyświetliłam wszystkie pliki jakie powłoka odczytuje przy rozpoczynaniu jej pracy.

```
emilia@emilia-virtual-machine:~/Pulpit$ strace -e trace=read bash
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\240\346\0\0\0\0\0\0"..., 832) = 832
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0 \22\0\0\0\0\0\0"..., 832) = 832
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360q\2\0\0\0\0\0"..., 832) = 832
read(3, "# /etc/nsswitch.conf\n#\n# Example"..., 4096) = 542
read(3, "", 4096)                       = 0
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\3005\0\0\0\0\0\0"..., 832) = 832
read(3, "root:x:0:0:root:/root:/bin/bash\n"..., 4096) = 2797
read(3, "\36\2%\0&\0\17\0\235\1\356\5xterm-256color|xterm"..., 32768) = 3503
read(3, "", 28672)                      = 0
read(3, "# System-wide .bashrc file for i"..., 2319) = 2319
read(3, "# ~/.bashrc: executed by bash(1)"..., 3771) = 3771
read(3, "./vmware-install.pl \nsudo ./vmwa"..., 21563) = 21563
read(3, "export LESSOPEN=\"| /usr/bin/less"..., 128) = 86
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=2092, si_uid=1000, si_status=0, si_utime=0, si_stime=0} ---
read(3, "", 128)                        = 0
read(3, "LS_COLORS='rs=0:di=01;34:ln=01;3"..., 128) = 128
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=2096, si_uid=1000, si_status=0, si_utime=0, si_stime=0} ---
read(3, "a=30;41:tw=30;42:ow=34;42:st=37;"..., 128) = 128
read(3, "*.lzh=01;31:*.lzma=01;31:*.tlz=0"..., 128) = 128
read(3, ":*.lz=01;31:*.lzo=01;31:*.xz=01;"..., 128) = 128
read(3, "31:*.rpm=01;31:*.jar=01;31:*.war"..., 128) = 128
read(3, "=01;31:*.rz=01;31:*.cab=01;31:*."..., 128) = 128
read(3, "5:*.gif=01;35:*.bmp=01;35:*.pbm="..., 128) = 128
read(3, "=01;35:*.svg=01;35:*.svgz=01;35:"..., 128) = 128
read(3, "5:*.ogm=01;35:*.mp4=01;35:*.m4v="..., 128) = 128
read(3, "01;35:*.flc=01;35:*.avi=01;35:*."..., 128) = 128
read(3, "f=01;35:*.ogv=01;35:*.ogx=01;35:"..., 128) = 128
read(3, ":*.mpc=00;36:*.ogg=00;36:*.ra=00"..., 128) = 118
read(3, "", 128)                        = 0
read(3, "#                               "..., 74550) = 74550
read(3, "# \n# Apport bash-completion\n#\n##"..., 6636) = 6636
read(3, "# In git versions < 1.7.12, this"..., 439) = 439
read(3, "# bash/zsh git prompt support\n#\n"..., 16938) = 16938
read(3, "./vmware-install.pl \nsudo ./vmwa"..., 21563) = 21563
read(3, "./vmware-install.pl \nsudo ./vmwa"..., 21563) = 21563
read(3, "# /etc/inputrc - global inputrc "..., 1748) = 1748
```

Następnie przeanalizowałam je i uznałam, że pliki .bashrc plikami konfiguracyjnymi basha. Zatem wykonałam odpowiednie grupowanie i uzyskałam poniższe wyniki:

```
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB3$ strace -o zad6_2out -e trace=read bash
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB3$ cat zad6_2out | grep -E 'bashrc'
read(3, "# System-wide .bashrc file for i"..., 2319) = 2319
read(3, "# ~/.bashrc: executed by bash(1)"..., 3771) = 3771
```

- Sprawdź, czy plik edytowany w programie pico jest stale otwarty.
  Aby dokonać sprawdzenia wykonałam polecenie *strace -o zad6_3out pico nowyplik*. W programie pico zapisałam kilka linijek do pliku, zapisałam sam plik i zamknęłam program. Następnie wybrałam wszystkie linie z pliku zad6_3out, które zawierają słowo *nowyplik*. Na poniższym zdjęciu przedstawione są rezultaty tego grupowania zawartości pliku.

```
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB3$ strace -o zad6_3out pico nowyplik
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB3$ cat zad6_3out | grep nowyplik
execve("/usr/bin/pico", ["pico", "nowyplik"], 0x7ffc93dfaf48 /* 48 vars */) = 0
stat("nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
stat("./.nowyplik.swp", 0x7ffdcb9bd230) = -1 ENOENT (Nie ma takiego pliku ani katalogu)
stat("./.nowyplik.swp", 0x7ffdcb9bd110) = -1 ENOENT (Nie ma takiego pliku ani katalogu)
openat(AT_FDCWD, "./.nowyplik.swp", O_WRONLY|O_CREAT|O_EXCL|O_APPEND, 0666) = 3
stat("nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
stat("/home/emilia/Pulpit/SCR2/LAB3/nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
stat("/home/emilia/Pulpit/SCR2/LAB3/nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
openat(AT_FDCWD, "/home/emilia/Pulpit/SCR2/LAB3/nowyplik", O_RDONLY) = 3
access("nowyplik", W_OK)                = 0
stat("nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
stat("nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
stat("./.nowyplik.swp", {st_mode=S_IFREG|0664, st_size=1024, ...}) = 0
unlink("./.nowyplik.swp")               = 0
openat(AT_FDCWD, "./.nowyplik.swp", O_WRONLY|O_CREAT|O_EXCL|O_APPEND, 0666) = 3
stat("nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
stat("nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
stat("/home/emilia/Pulpit/SCR2/LAB3/nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
stat("nowyplik", {st_mode=S_IFREG|0664, st_size=34, ...}) = 0
openat(AT_FDCWD, "nowyplik", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 3
stat("nowyplik", {st_mode=S_IFREG|0664, st_size=40, ...}) = 0
unlink("./.nowyplik.swp")               = 0
emilia@emilia-virtual-machine:~/Pulpit/SCR2/LAB3$
```

Plik nie jest stale otwarty. Program korzysta z plików .swp – czyli plików wirtualnej pamięci i to na nich dokonuje zmian. Jako pierwszy zostaje otworzony plik .swp. Jest on potem zamykany i otwierany kilkukrotnie, co przedstawione jest na zdjęciu poniżej.

```
507 openat(AT_FDCWD, "./.nowyplik.swp", O_WRONLY|O_CREAT|O_EXCL|O_APPEND, 0666) = 3
508 fcntl(3, F_GETFL)                        = 0x8401 (flags O_WRONLY|O_APPEND|O_LARGEFILE)
509 fstat(3, {st_mode=S_IFREG|0664, st_size=0, ...}) = 0
510 write(3, "b0nano 4.8\0\0\0\0\0\0\0\0\0\0\0\0\0\0\273\v\0\0emil"..., 1024) = 1024
511 close(3)                                = 0
```

Oryginalny plik otwierany jest na początku tylko do odczytu, ale potem również jest zamykany. Na końcu otworzony jest do zapisu, przepisywana jest zawartość i odłączany jest plik pamięci wirtualnej.

- Odczytaj, jakie file deskryptory posiada uruchomiona aplikacja wyświetlająca napis Hello World na ekranie.

  Aby wykonać zadanie napisałam program zadanie6_2.c i użyłam polecenia strace -o output3 -e trace=desc ./zadanie6_2, aby wyświetlić wszystkie funkcje związane z file deskryptorami. Poniżej przedstawione są wyniki powyższego wywołania.

```
 1 openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
 2 fstat(3, {st_mode=S_IFREG|0644, st_size=75219, ...}) = 0
 3 mmap(NULL, 75219, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f40cb6a4000
 4 close(3)                              = 0
 5 openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
 6 read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360q\2\0\0\0\0\0"..., 832) = 832
 7 pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
 8 pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32
 9 pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355Y\377\t\334"..., 68, 880) = 68
10 fstat(3, {st_mode=S_IFREG|0755, st_size=2029224, ...}) = 0
11 mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f40cb6a2000
12 pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
13 pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32
14 pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355Y\377\t\334"..., 68, 880) = 68
15 mmap(NULL, 2036952, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f40cb4b0000
16 mmap(0x7f40cb4d5000, 1540096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x25000) = 0x7f40cb4d5000
17 mmap(0x7f40cb64d000, 303104, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x19d000) = 0x7f40cb64d000
18 mmap(0x7f40cb698000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f40cb698000
19 mmap(0x7f40cb69e000, 13528, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f40cb69e000
20 close(3)                              = 0
21 fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) = 0
22 write(1, "M\303\263j pid: 4817\n", 15)  = 15
23 write(1, "Hello\n", 6)                = 6
24 write(1, "Hello\n", 6)                = 6
25 write(1, "Hello\n", 6)                = 6
26 write(1, "Hello\n", 6)                = 6
27 write(1, "Hello\n", 6)                = 6
28 write(1, "Hello\n", 6)                = 6
29 write(1, "Hello\n", 6)                = 6
30 write(1, "Hello\n", 6)                = 6
31 write(1, "Hello\n", 6)                = 6
32 write(1, "Hello\n", 6)                = 6
33 write(1, "Hello\n", 6)                = 6
34 write(1, "Hello\n", 6)                = 6
35 write(1, "Hello\n", 6)                = 6
36 write(1, "Hello\n", 6)                = 6
37 write(1, "Hello\n", 6)                = 6
```

Jak możemy zauważyć używany jest file deskryptor numer 1 oraz 3. Pierwszy z nich używany jest do wypisywania napisu na terminal za pomocą funkcji write. Funkcja openat() zwraca numer drugiego file deskryptora. Oprócz nich istnieją jeszcze dwa file deskryptory: 0 raz 2. File deskryptor numer 0 jest swego rodzaju uchwytem do standardowego wejścia, którym w tym programie jest terminal. File deskryptor numer 2 jest uchwytem do standardowego wyjścia errorów, którym w tym programie jest terminal.

Zadanie 7

Zadanie 7 polegało na znalezieniu błędu w programie *program.c* i określeniu jaki sygnał zabił program. Dodatkowo należało określić czas wykonania poszczególnych elementów programu za pomocą funkcji strace. Przepisałam ten program i nazwałam go *zadanie7.c*.

Następnie uruchomiłam go z komendą *strace -o zad7_out ./zadanie7*. Już w samym bashu uzyskaliśmy informację, co zadziało się z programem: nastąpiło naruszenie ochrony pamięci.

Poniżej znajduje się zawartość pliku zad7_out:

```
execve("./zadanie7", ["./zadanie7"], 0x7ffed428bfe0 /* 48 vars */) = 0
brk(NULL)                               = 0x55c7024a2000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffc0dab2f90) = -1 EINVAL (Zły argument)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (Nie ma takiego pliku ani katalogu)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=75219, ...}) = 0
mmap(NULL, 75219, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f9db3c6f000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360q\2\0\0\0\0\0"..., 832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355Y\377\t\334"..., 68,
880) = 68
fstat(3, {st_mode=S_IFREG|0755, st_size=2029224, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f9db3c6d000
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355Y\377\t\334"..., 68,
880) = 68
mmap(NULL, 2036952, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f9db3a7b000
mprotect(0x7f9db3aa0000, 1847296, PROT_NONE) = 0
mmap(0x7f9db3aa0000, 1540096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x25000) = 0x7f9db3aa0000
mmap(0x7f9db3c18000, 303104, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x19d000) =
0x7f9db3c18000
mmap(0x7f9db3c63000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1e7000) = 0x7f9db3c63000
mmap(0x7f9db3c69000, 13528, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1,
0) = 0x7f9db3c69000
close(3)                                = 0
arch_prctl(ARCH_SET_FS, 0x7f9db3c6e540) = 0
mprotect(0x7f9db3c63000, 12288, PROT_READ) = 0
mprotect(0x55c700db4000, 4096, PROT_READ) = 0
```

```
mprotect(0x7f9db3caf000, 4096, PROT_READ) = 0
munmap(0x7f9db3c6f000, 75219)           = 0
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x2), ...}) = 0
brk(NULL)                               = 0x55c7024a2000
brk(0x55c7024c3000)                     = 0x55c7024c3000
write(1, "Witajcie moi mili ...\0\0\0\1\33\3;@\0\0\0"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 613) = 613
write(1, "\0\0\0\0\0\0\0\205\0\0\0\0\0\0\0\v\0\0\0\0\0\0\0\30\0\0\0\0\0\0\0\25"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\205\0\0\0\0\0\0\0\v\0\0\0\0\0\0\0\30\0\0\0\0\0\0\0\25"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024
--- SIGSEGV {si_signo=SIGSEGV, si_code=SEGV_MAPERR, si_addr=0x55c700db6000} ---
+++ killed by SIGSEGV (core dumped) +++
```

Istotne są dla nas ostatnie dwie linijki tego pliku. Zawierają one informację, jaki sygnał zabił program. Był to sygnał SIGSEGV – naruszenie pamięci. Spowodowane jest to pętlą for, która nie ma ograniczenia. Przez to już w kolejnym wywołaniu pętli - już poza ilością elementów zadeklarowanych w tablicy napis – spowodujemy błąd w dostępie do pamięci i zostanie wysłany sygnał do programu, aby go unicestwić.

Korzystając z polecenia strace możemy również określić czasy wykonania poszczególnych procedur programu. Uzyskujemy to przez komendę strace -T i przy każdym wywołaniu wypisanym przez strace, zapisane są czasy wykonania od początku do końca tej linijki.

```
execve("./zadanie7", ["./zadanie7"], 0x7fff12c5b258 /* 48 vars */) = 0 <0.000435>
brk(NULL)                               = 0x563136be8000 <0.000111>
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffe817f7440) = -1 EINVAL (Zły argument) <0.000121>
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (Nie ma takiego pliku ani katalogu) <0.000141>
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3 <0.000131>
fstat(3, {st_mode=S_IFREG|0644, st_size=75219, ...}) = 0 <0.000123>
mmap(NULL, 75219, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f3c7412d000 <0.000128>
close(3)                                = 0 <0.000103>
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3 <0.000089>
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360q\2\0\0\0\0\0"..., 832) = 832 <0.000105>
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784 <0.000106>
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32 <0.000078>
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355Y\377\t\334"..., 68,
880) = 68 <0.000064>
fstat(3, {st_mode=S_IFREG|0755, st_size=2029224, ...}) = 0 <0.000102>
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f3c7412b000
<0.000078>
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784 <0.000065>
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32 <0.000118>
```

```
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355Y\377\t\334"..., 68, 880)
= 68 <0.000785>
mmap(NULL, 2036952, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f3c73f39000 <0.000095>
mprotect(0x7f3c73f5e000, 1847296, PROT_NONE) = 0 <0.000110>
mmap(0x7f3c73f5e000, 1540096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x25000) = 0x7f3c73f5e000 <0.000125>
mmap(0x7f3c740d6000, 303104, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x19d000) =
0x7f3c740d6000 <0.000070>
mmap(0x7f3c74121000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1e7000) = 0x7f3c74121000 <0.000097>
mmap(0x7f3c74127000, 13528, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) =
0x7f3c74127000 <0.000075>
close(3)                          = 0 <0.000090>
arch_prctl(ARCH_SET_FS, 0x7f3c7412c540) = 0 <0.000063>
mprotect(0x7f3c74121000, 12288, PROT_READ) = 0 <0.000082>
mprotect(0x563136853000, 4096, PROT_READ) = 0 <0.000081>
mprotect(0x7f3c7416d000, 4096, PROT_READ) = 0 <0.000081>
munmap(0x7f3c7412d000, 75219)           = 0 <0.000076>
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) = 0 <0.000064>
brk(NULL)                         = 0x563136be8000 <0.000063>
brk(0x563136c09000)                   = 0x563136c09000 <0.000106>
write(1, "Witajcie moi mili ...\0\0\0\1\33\3;@\0\0\0"..., 1024) = 1024 <0.000081>
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024 <0.000092>
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024 <0.000123>
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 613) = 613 <0.000087>
write(1, "\0\0\0\0\0\0\0\205\0\0\0\0\0\0\0\v\0\0\0\0\0\0\0\30\0\0\0\0\0\0\0\25"..., 1024) = 1024 <0.000077>
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024 <0.000082>
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024 <0.000069>
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024 <0.000077>
write(1, "\0\0\0\0\0\0\0\205\0\0\0\0\0\0\0\v\0\0\0\0\0\0\0\30\0\0\0\0\0\0\0\25"..., 1024) = 1024 <0.000075>
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024 <0.000069>
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024 <0.000069>
write(1, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 1024) = 1024 <0.000069>
--- SIGSEGV {si_signo=SIGSEGV, si_code=SEGV_MAPERR, si_addr=0x563136855000} ---
+++ killed by SIGSEGV (core dumped) +++
```