

# SKPS laboratorium nr 3

Kinga Świderek (318 734) i Jakub Kowalczyk (318 676)

## 0. Przygotowanie stanowiska pracy i SDK

Przed rozpoczęciem pracy pobraliśmy SDK do folderu *openwrt-sdk* oraz pobraliśmy i rozpakowaliśmy archiwum *WZ\_W03\_przyklady.tar.xz*.

## 1. Pierwszy pakiet

W folderze *openwrt-sdk* wykonaliśmy komendę:

**export LANG=C**

W pliku zawierającym ścieżki do katalogów z pakietami *feeds.conf.default* dopisujemy na końcu:

*src-link skps /home/user/demo1\_owrt\_pkg*

Aktualizowanie listy pakietów:

**./scripts/feeds update -a**

**./scripts/feeds install -p skps -a**

**./scripts/feeds install -p packages -a**

W *make menuconfig* zaznaczamy opcje *demo1*, *demo1mak*

**Następnie, aby skompilować paczki wpisujemy:**

**make package/feeds/skps/demo1/compile**

**make package/feeds/skps/demo1mak/compile**

Przesłaliśmy pliki przez serwer *http* za pomocą komendy **wget**.

Na OpenWRT zainstalowaliśmy pomyślnie pakiet następująca komendą:

```
root@OpenWrt:/# opkg install --force-reinstall demo1.ipk
No packages removed.
Installing demo1 (1.0-1) to root...
Configuring demo1.
root@OpenWrt:/# demo1
dzien dobry
Komunikat z wątku A
Komunikat z wątku B
Komunikat z wątku B
Komunikat z wątku A
Komunikat z wątku B
Komunikat z wątku B
```

## 2. Pakiety “worms” i “buggy”

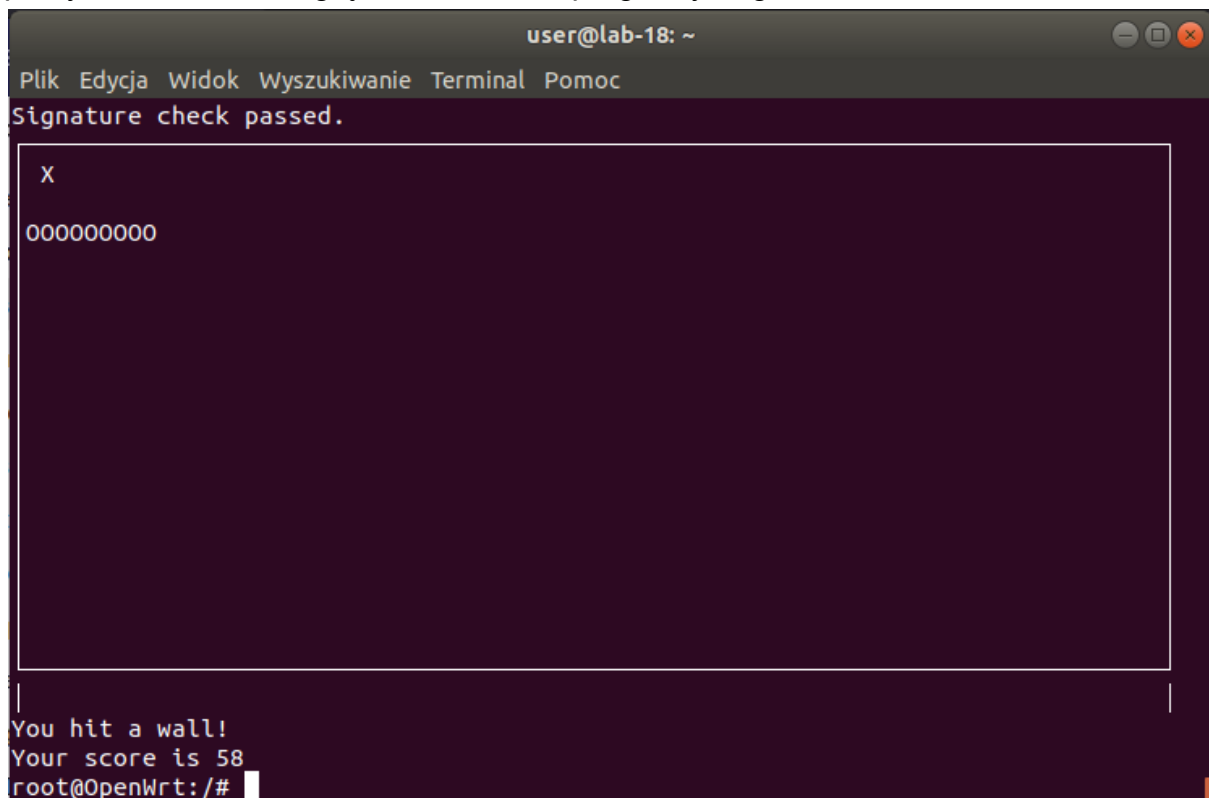
Pobraliśmy z moodle i rozpakowaliśmy paczkę z pakietami worms i buggy.

Następnie zainstalowaliśmy ncurses w SDK poleceniem:

**./scripts/feeds install libncurses**

Stworzyliśmy odpowiednie pliki makefile na podstawie tych z pierwszego zadania (zamieszczone na gitlabie).

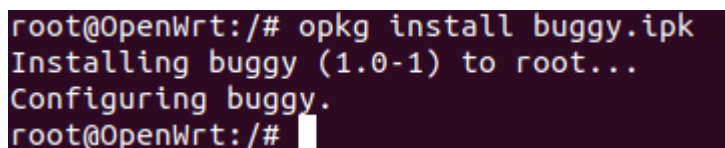
Następnie w analogiczny sposób, co w poprzednim podpunkcie zaktualizowaliśmy listy pakietów, zainstalowaliśmy nasze pakiety, oznaczając przy tym odpowiednie nazwy w menuconfig. Pozostało tylko przesłanie plików przez http i udało się pomyślnie uruchomić grę w Snake oraz programy bug1/2/3.



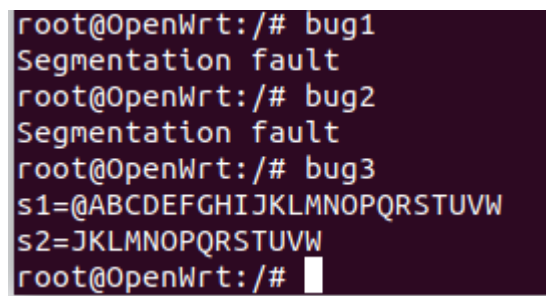
```
user@lab-18: ~
Plik Edycja Widok Wyszukiwanie Terminal Pomoc
Signature check passed.

X
000000000

You hit a wall!
Your score is 58
root@OpenWrt:/#
```



```
root@OpenWrt:/# opkg install buggy.ipk
Installing buggy (1.0-1) to root...
Configuring buggy.
root@OpenWrt:/#
```



```
root@OpenWrt:/# bug1
Segmentation fault
root@OpenWrt:/# bug2
Segmentation fault
root@OpenWrt:/# bug3
s1=@ABCDEFGHJKLMNOPQRSTUVWXYZ
s2=JKLMNOPQRSTUVWXYZ
root@OpenWrt:/#
```

### 3. Debuggowanie zdalne

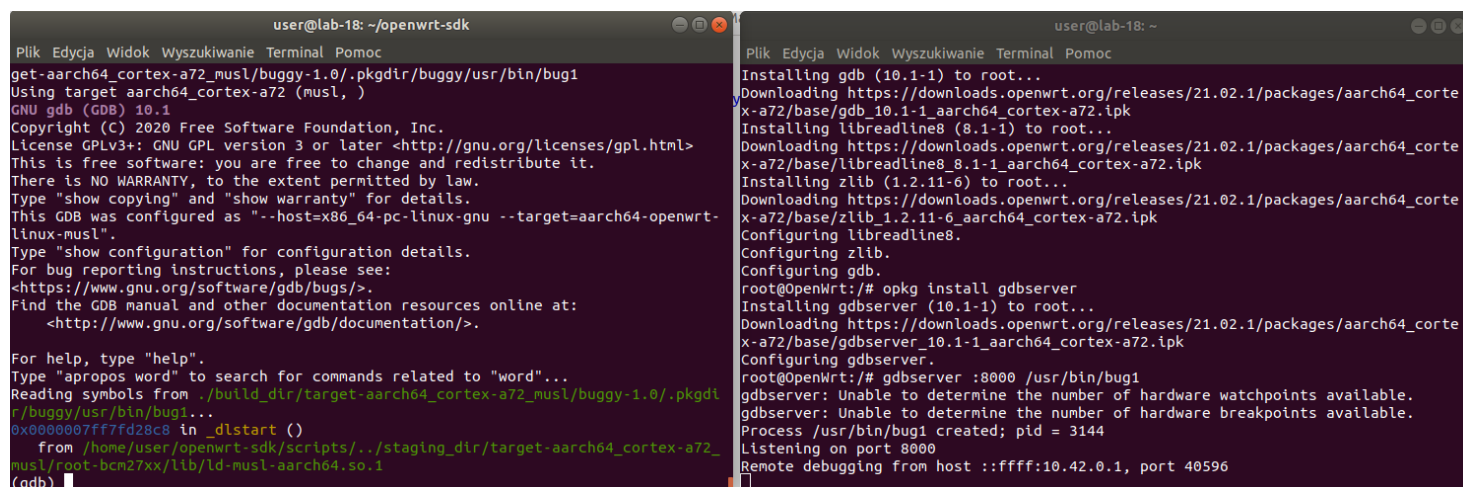
Zainstalowaliśmy pakiety gdb i gdbserver za pomocą **opkg install**. Następnie komendą **gdbserver :8000 /usr/bin/bug1** uruchomiliśmy serwer, a na komputerze hosta połączyliśmy się komendą **./scripts/remote-gdb 10.42.0.188:8000 ./build\_dir/target-aarch64\_cortex-a72\_musl/buggy-1.0/.pkgdir/buggy/usr/bin/bug1**

Włączyliśmy także S/W watchpoint poleceniami:

**set breakpoint auto-hw off**

**set can-use-hw-watchpoints 0**

Mogliśmy już rozpocząć debuggowanie.



```
user@lab-18: ~/openwrt-sdk
Plik Edycja Widok Wyszukiwanie Terminal Pomoc
get-aarch64_cortex-a72_musl/buggy-1.0/.pkgdir/buggy/usr/bin/bug1
Using target aarch64_cortex-a72 (musl, )
GNU gdb (GDB) 10.1
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=x86_64-pc-linux-gnu --target=aarch64-openwrt-linux-musl".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./build_dir/target-aarch64_cortex-a72_musl/buggy-1.0/.pkgdir/buggy/usr/bin/bug1...
0x0000007ff7fd28c8 in _dlstart ()
    from /home/user/openwrt-sdk/scripts/./staging_dir/target-aarch64_cortex-a72_musl/root-bcm27xx/lib/ld-musl-aarch64.so.1
(gdb)

user@lab-18: ~
Plik Edycja Widok Wyszukiwanie Terminal Pomoc
Installing gdb (10.1-1) to root...
Downloading https://downloads.openwrt.org/releases/21.02.1/packages/aarch64_cortex-a72/base/gdb_10.1-1_aarch64_cortex-a72.ipk
Installing libreadline8 (8.1-1) to root...
Downloading https://downloads.openwrt.org/releases/21.02.1/packages/aarch64_cortex-a72/base/libreadline8_8.1-1_aarch64_cortex-a72.ipk
Installing zlib (1.2.11-6) to root...
Downloading https://downloads.openwrt.org/releases/21.02.1/packages/aarch64_cortex-a72/base/zlib_1.2.11-6_aarch64_cortex-a72.ipk
Configuring libreadline8.
Configuring zlib.
Configuring gdb.
root@OpenWrt:~# opkg install gdbserver
Installing gdbserver (10.1-1) to root...
Downloading https://downloads.openwrt.org/releases/21.02.1/packages/aarch64_cortex-a72/base/gdbserver_10.1-1_aarch64_cortex-a72.ipk
Configuring gdbserver.
root@OpenWrt:~# gdbserver :8000 /usr/bin/bug1
gdbserver: Unable to determine the number of hardware watchpoints available.
gdbserver: Unable to determine the number of hardware breakpoints available.
Process /usr/bin/bug1 created; pid = 3144
Listening on port 8000
Remote debugging from host ::ffff:10.42.0.1, port 40596
```

W ramach doświadczeń z gdb wykonaliśmy:

- ustawienie breakpointu

```
(gdb) break main
Breakpoint 1 at 0x400460: file buggy-1.0/bug1.c, line 9.
(gdb)
```

- pracę krokową

```
Breakpoint 5, main () at buggy-1.0/bug3.c:12
12     for(i=0;i<24;i++) {
(gdb) next
13     s1[i]=i+64;
(gdb)
```

- podgląd wartości zmiennej (jednorazowy i przy każdym kroku)

```
(gdb) print i
$1 = 1
```

- backtrace

```
(gdb) backtrace
#0  main () at buggy-1.0/bug1.c:9
(gdb)
```

- podgląd stosu

```
(gdb) x/40x $sp
0x7fffffff50: 0xffffffffd60      0x00000007f      0xf7f93190      0x00000007f
0x7fffffff60: 0x000000000      0x000000000      0x000000000      0x000000000
0x7fffffff70: 0xffffffffd90      0x00000007f      0xf7fdca8      0x00000007f
0x7fffffff80: 0xf7ffde50      0x00000007f      0xf7ffde50      0x00000007f
0x7fffffff90: 0x000000001      0x000000000      0xffffffff69      0x00000007f
0x7fffffffda0: 0x000000000      0x000000000      0xffffffff77      0x00000007f
0x7fffffffdb0: 0xffffffff7f      0x00000007f      0xffffffff8a      0x00000007f
0x7fffffffdc0: 0xffffffff96      0x00000007f      0xffffffffa6      0x00000007f
0x7fffffffdd0: 0xffffffffb6      0x00000007f      0xffffffffc1      0x00000007f
0x7fffffffde0: 0xffffffffe4      0x00000007f      0x000000000      0x000000000
```

- wykorzystanie watchpoint'ów w programie bug3, aby sprawdzić kiedy następuje zapisanie wartości pod niewłaściwym adresem, np. w s1[10].

```
(gdb) watch s1[5]
Hardware watchpoint 6: s1[5]
(gdb) █
```

Znaleźliśmy następujące błędy:

- W pliku bug1 tablica "table" nie jest alokowana
- W pliku bug2 jest próba odwołania się poza tablicę
- W pliku bug3 nadpisuje się literę na końcowy NULL w ciągu znaków