

# Pwntools

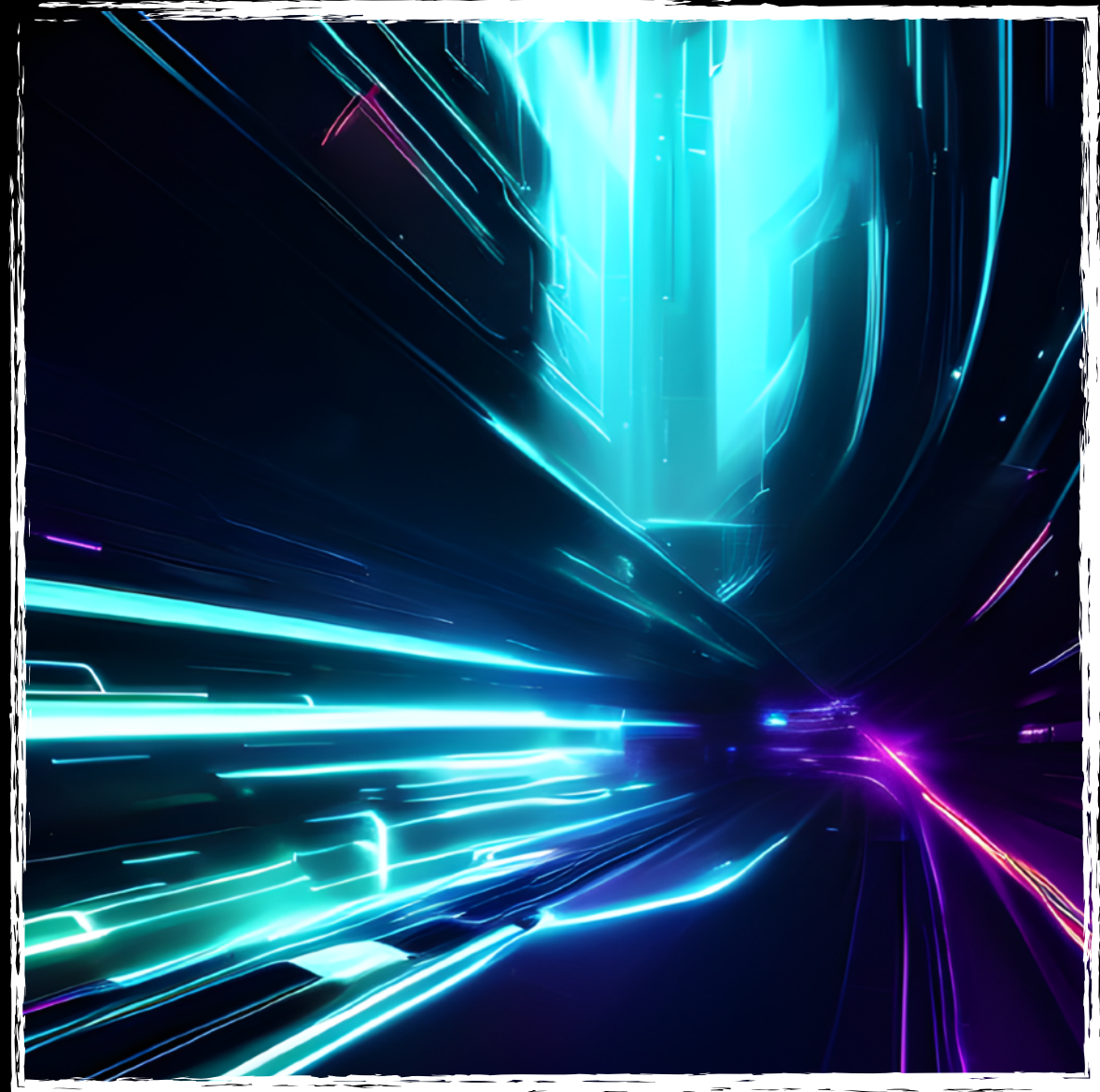
Curious @ NCYU Hackers

2023.09.26

## >\_ 目錄

- ▶ 先備知識
- ▶ PPC 介紹
- ▶ Pwntools

>\_ WhoAmI



- ▶ ID : Curious
- ▶ DC : curious\_lucifer
- ▶ SCIST 4rd 總召 / LoTuX CTF 創辦人

>\_ Lab

Lab : <http://172.104.90.38>

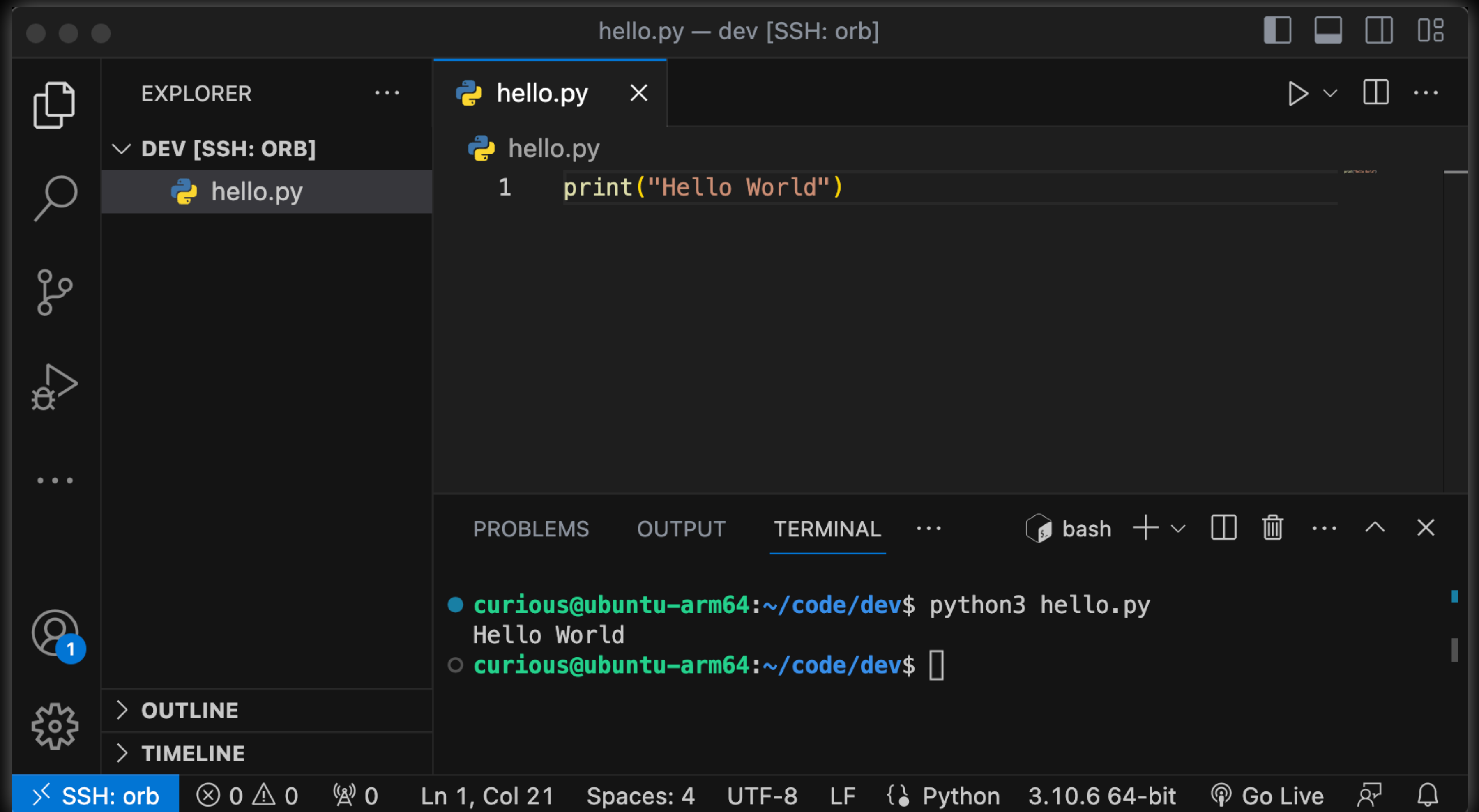
Flag Format : NCYU\_HACKERS{.\*}

先備知識

## >\_ 如何執行 Python

### ▶ 執行檔案

- ▶ Interactive Shell
- ▶ Interactive Shell Pro



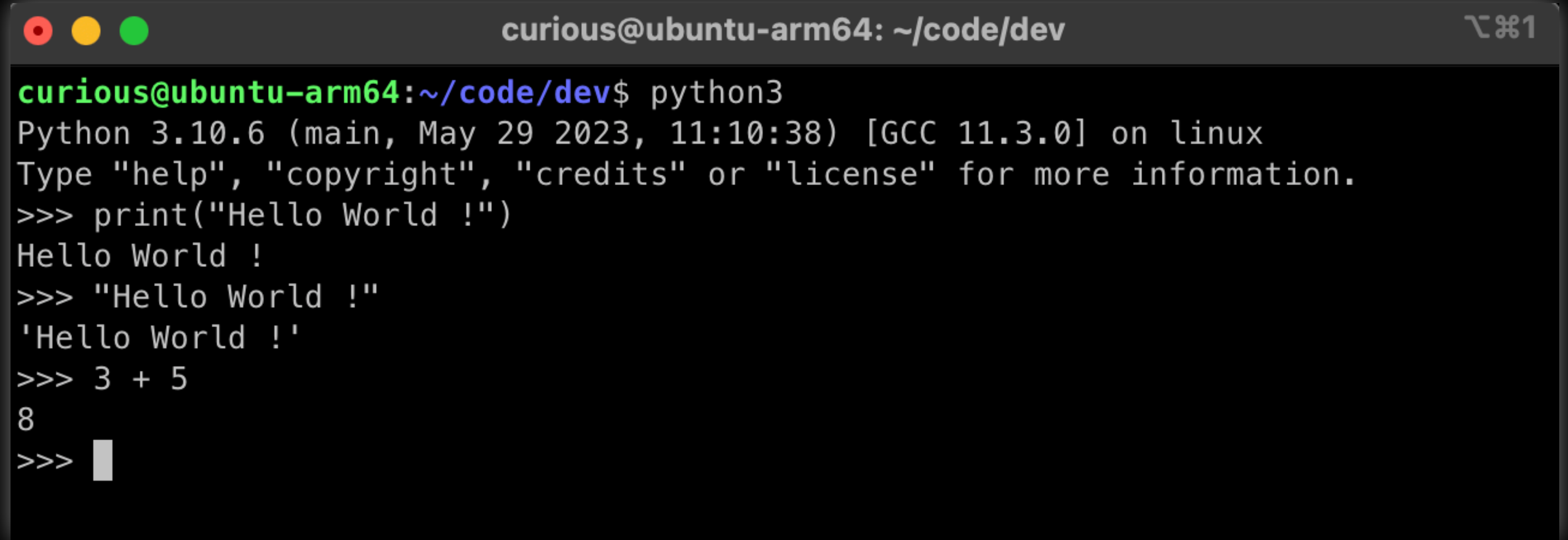
The screenshot shows the Visual Studio Code (VS Code) interface. The title bar indicates the file is 'hello.py' in a remote development environment 'dev [SSH: orb]'. The Explorer sidebar on the left shows the file 'hello.py' under the 'DEV [SSH: ORB]' workspace. The main editor area displays the code in 'hello.py':

```
1 print("Hello World")
```

Below the editor, the TERMINAL panel is active, showing a bash shell. The command 'python3 hello.py' has been executed, and the output 'Hello World' is displayed. The status bar at the bottom shows the current file is 'SSH: orb', with 0 errors, 0 warnings, and 0 suggestions. It also indicates the cursor is at line 1, column 21, with 4 spaces, using UTF-8 encoding and LF line endings. The Python version is 3.10.6 64-bit, and the 'Go Live' feature is enabled.

## >\_ 如何執行 Python


- ▶ 執行檔案
- ▶ **Interactive Shell**
- ▶ Interactive Shell Pro

A terminal window titled 'curious@ubuntu-arm64: ~/code/dev' with a standard Linux window header (red, yellow, green buttons). The terminal shows the execution of 'python3', which starts the Python 3.10.6 interpreter. The user enters 'print("Hello World !")' and the output is 'Hello World !'. Then, the user enters '"Hello World !"' and the output is ''Hello World !''. Finally, the user enters '3 + 5' and the output is '8'. The prompt '>>>' is visible at the end of the last line.

```
curious@ubuntu-arm64: ~/code/dev$ python3
Python 3.10.6 (main, May 29 2023, 11:10:38) [GCC 11.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello World !")
Hello World !
>>> "Hello World !"
'Hello World !'
>>> 3 + 5
8
>>> █
```

## >\_ 如何執行 Python

- ▶ 執行檔案
- ▶ Interactive Shell
- ▶ **Interactive Shell Pro**

A screenshot of a terminal window titled "IPython: code/dev". The terminal shows the execution of the command `ipython3` in a shell. The output displays the IPython version (7.31.1) and the Python version (3.10.6). The user then enters three lines of code: `print("Hello World !")`, `"Hello World !"`, and `3 + 5`. The terminal shows the corresponding outputs: `Hello World !`, `'Hello World !'`, and `8`. The fourth line shows the prompt `In [4]:` with a cursor.

```
IPython: code/dev
curious@ubuntu-arm64:~/code/dev$ ipython3
Python 3.10.6 (main, May 29 2023, 11:10:38) [GCC 11.3.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.31.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: print("Hello World !")
Hello World !

In [2]: "Hello World !"
Out[2]: 'Hello World !'

In [3]: 3 + 5
Out[3]: 8

In [4]:
```

Hint : 可以用 `sudo apt install ipython3` 來安裝



# >\_ ASCII

- ▶ 一張把字元和數字互相轉換的表
- ▶ 數字只包含 0 ~ 127
- ▶ 用 `chr(num)` 把數字轉成字元
- ▶ 用 `ord(char)` 把字元轉成數字

dec	hex	oct	char	dec	hex	oct	char	dec	hex	oct	char	dec	hex	oct	char
0	0	000	NULL	32	20	040	space	64	40	100	@	96	60	140	`
1	1	001	SOH	33	21	041	!	65	41	101	A	97	61	141	a
2	2	002	STX	34	22	042	"	66	42	102	B	98	62	142	b
3	3	003	ETX	35	23	043	#	67	43	103	C	99	63	143	c
4	4	004	EOT	36	24	044	\$	68	44	104	D	100	64	144	d
5	5	005	ENQ	37	25	045	%	69	45	105	E	101	65	145	e
6	6	006	ACK	38	26	046	&	70	46	106	F	102	66	146	f
7	7	007	BEL	39	27	047	'	71	47	107	G	103	67	147	g
8	8	010	BS	40	28	050	(	72	48	110	H	104	68	150	h
9	9	011	TAB	41	29	051	)	73	49	111	I	105	69	151	i
10	a	012	LF	42	2a	052	*	74	4a	112	J	106	6a	152	j
11	b	013	VT	43	2b	053	+	75	4b	113	K	107	6b	153	k
12	c	014	FF	44	2c	054	,	76	4c	114	L	108	6c	154	l
13	d	015	CR	45	2d	055	-	77	4d	115	M	109	6d	155	m
14	e	016	SO	46	2e	056	.	78	4e	116	N	110	6e	156	n
15	f	017	SI	47	2f	057	/	79	4f	117	O	111	6f	157	o
16	10	020	DLE	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM	57	39	071	9	89	59	131	Y	121	79	171	y
26	1a	032	SUB	58	3a	072	:	90	5a	132	Z	122	7a	172	z
27	1b	033	ESC	59	3b	073	;	91	5b	133	[	123	7b	173	{
28	1c	034	FS	60	3c	074	<	92	5c	134	\	124	7c	174	
29	1d	035	GS	61	3d	075	=	93	5d	135	]	125	7d	175	}
30	1e	036	RS	62	3e	076	>	94	5e	136	^	126	7e	176	~
31	1f	037	US	63	3f	077	?	95	5f	137	_	127	7f	177	DEL

>\_ ASCII

Welcome : Welcome

## >\_ Python 資料型別

- int : 整數
- str : 字串
- bytes : 位元組

```
curious@Ubuntu-22:~$ ipython3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.31.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: type(1)
Out[1]: int

In [2]: type("Hello")
Out[2]: str

In [3]: type(b"holá")
Out[3]: bytes

In [4]:
```

Hint : 可以用 `type(obj)` 來確認物件的型別

## >\_ Python 資料型別

- int : 整數
- str : 字串
- bytes : 位元組

```
curious@Ubuntu-22:~$ ipython3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.31.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: type(1)
Out[1]: int

In [2]: type("Hello")    # 由字元組成
Out[2]: str

In [3]: type(b"holá")     # 由數字組成，但顯示時如果該數字轉成字元可視就會以字元顯示
Out[3]: bytes            # 如果該數字不可視，則會以 \x?? 來表示，其中 ?? 是 16 進位的數字

In [4]:
```

Hint : 可以用 `type(obj)` 來確認物件的型別



## >\_ Python 資料轉換

- int -> str
- int -> bytes
- int -> hex
- int -> ASCII
- str -> int
- bytes -> int
- hex -> int
- ASCII -> int

```
curious@Ubuntu-22:~$ ipython3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.31.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: str(3)                # 注意這邊是轉成字串 '3'，但它的 ASCII 並不是 3
Out[1]: '3'

In [2]: bytes([3])           # 注意這邊轉換成 bytes 後，儲存的數字就是 3
Out[2]: b'\x03'

In [3]: hex(12)
Out[3]: '0xc'

In [4]: chr(97)
Out[4]: 'a'

In [5]:
```

## >\_ Python 資料轉換

- int -> str
- int -> bytes
- int -> hex
- int -> ASCII
  
- str -> int
- bytes -> int
- hex -> int
- ASCII -> int

```
curious@Ubuntu-22:~$ ipython3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.31.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: int('3')
Out[1]: 3

In [2]: b'\x03'[0]
Out[2]: 3

In [3]: int('0xc', 16)
Out[3]: 12

In [4]: ord('a')
Out[4]: 97

In [5]:
```

## >\_ Python 資料轉換

- ▶ str -> bytes
  - ▶ hex -> bytes
  - ▶ int -> bytes
  - ▶ int -> str -> bytes
- 
- ▶ bytes -> str
  - ▶ bytes -> hex
  - ▶ bytes -> int
  - ▶ bytes -> str -> int

```
curious@Ubuntu-22:~$ ipython3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.31.1 -- An enhanced Interactive Python. Type '?' for help.
```

```
In [1]: 'Hello'.encode()  
Out[1]: b'Hello'
```

```
In [2]: bytes.fromhex('437572696f7573')    # 可以發現兩個 hex 可以代表一個 bytes
Out[2]: b'Curious'
```

```
In [3]: bytes([7])           # 如果想要轉換多個數字，可以用 bytes([39, 219, 184]) 這樣
Out[3]: b'\x07'             # 注意這裡儲存起來的數字是 7
```

```
In [4]: str(7).encode()
Out[4]: b'7' # 注意這裡儲存起來的數字是 55
```

In [5]:

## >\_ Python 資料轉換

- str -> bytes
- hex -> bytes
- int -> bytes
- int -> str -> bytes
  
- bytes -> str
- bytes -> hex
- bytes -> int
- bytes -> str -> int

```
curious@Ubuntu-22:~$ ipython3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.31.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: b'Hello'.decode()      # 注意這裡如果 bytes 含有一些非 UTF-8 的數字的話會報錯
Out[1]: 'Hello'

In [2]: b'Curious'.hex()
Out[2]: '437572696f7573'

In [3]: b'\x07'[0]
Out[3]: 7

In [4]: int(b'7')              # 這邊可以不用先把 b'7' decode 成 str 再轉換
Out[4]: 7

In [5]:
```

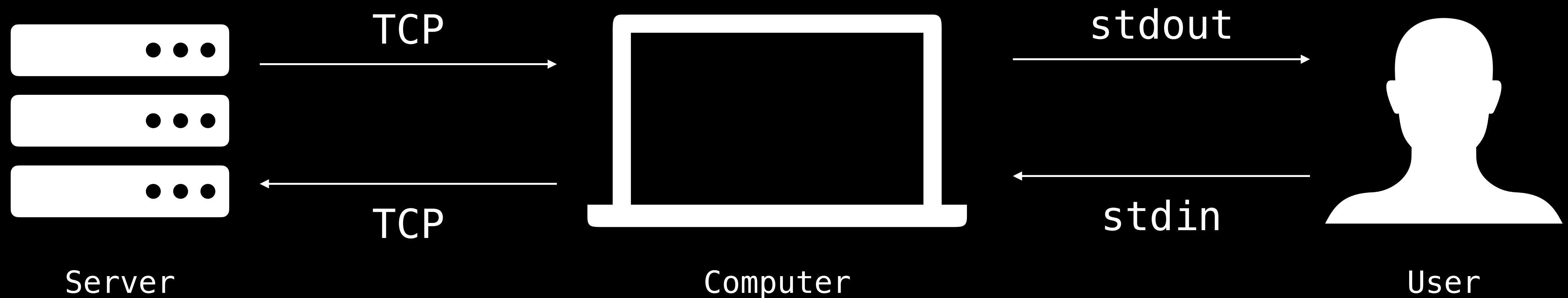


# PPC 介紹

## >\_ PPC 介紹

指令：`nc <IP/Domain> <Port>`

通常拿來和指定的 IP 或 Domain 上的 Port 建立連線，透過 terminal 上的 stdin 和 stdout 來和 server 互動



## >\_ PPC 介紹

指令：`nc <IP/Domain> <Port>`

通常拿來和指定的 IP 或 Domain 上的 Port 建立連線，透過 terminal 上的 stdin 和 stdout 來和 server 互動

```
curious@Ubuntu-22:~$ nc 172.104.90.38 10000
===== WELCOME TO PPC LAB =====
Please wait for a few second, let me calculate the flag
Flag : NCYU_HACKERS{[REDACTED]}

curious@Ubuntu-22:~$
```

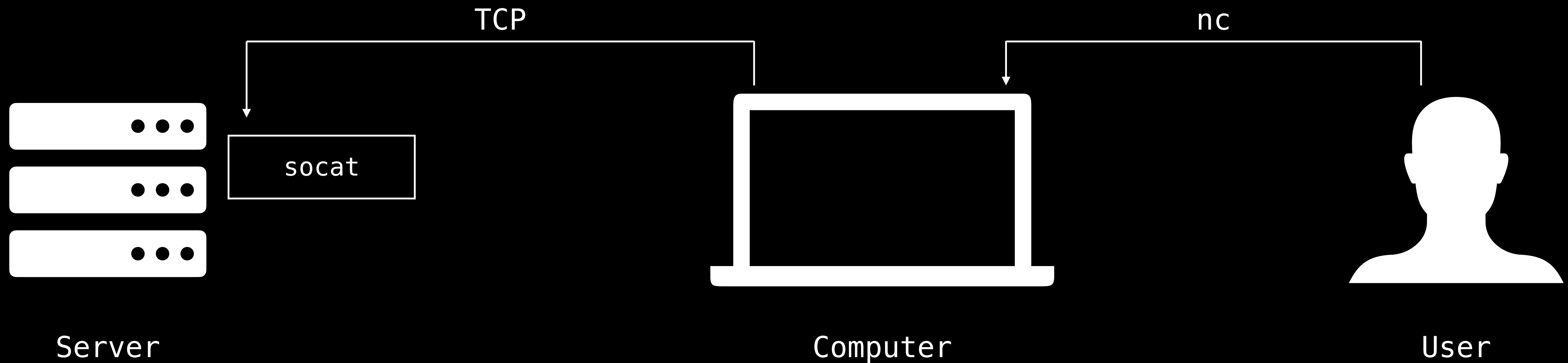
>\_ PPC 介紹

PPC Lab : Welcome

## >\_ PPC 介紹

指令：`socat TCP-LISTEN:<Port>,fork EXEC:"<cmd>"`

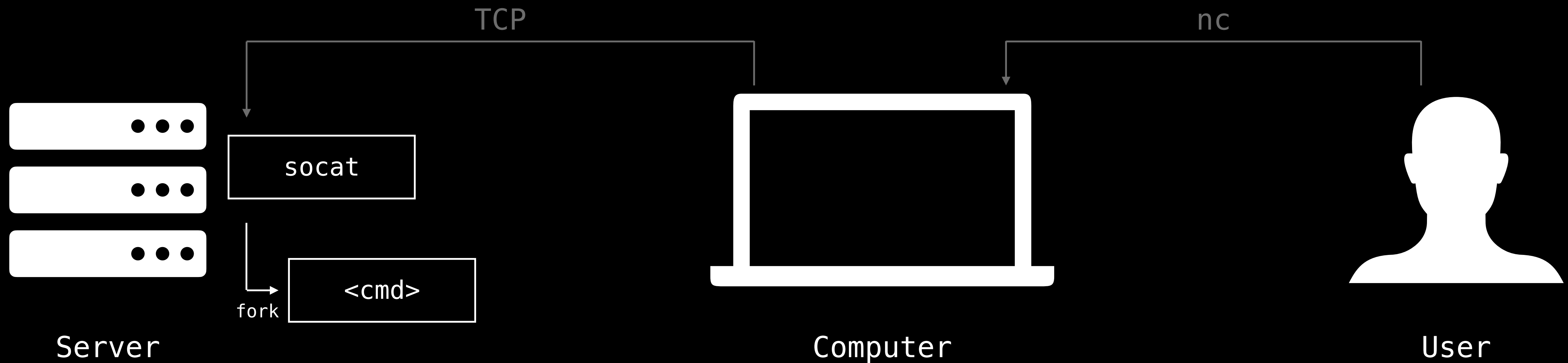
將指定指令的服務綁到本機 (127.0.0.1) 的 Port 上，當這個 Port 收到 TCP 請求後，執行指定的 cmd



## >\_ PPC 介紹

指令：`socat TCP-LISTEN:<Port>,fork EXEC:"<cmd>"`

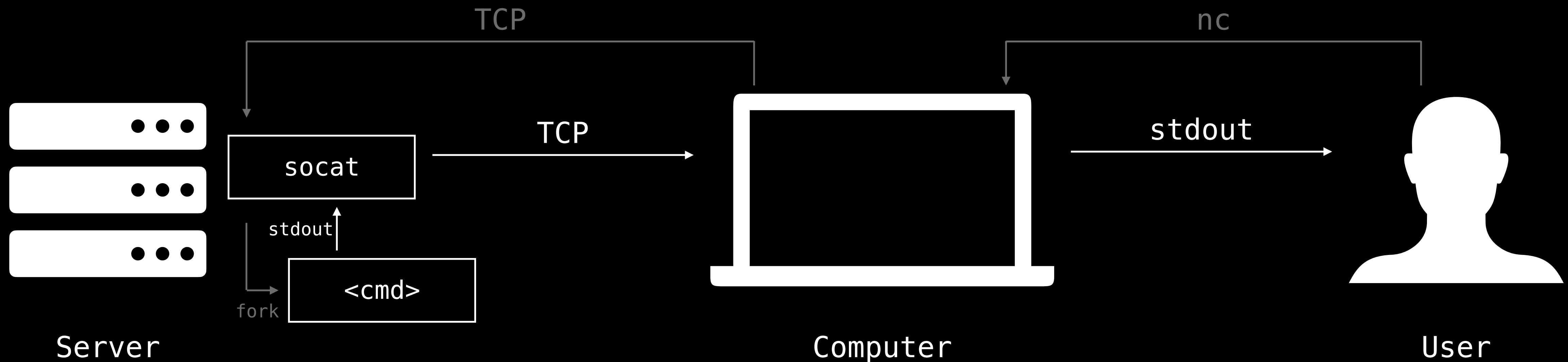
將指定指令的服務綁到本機 (127.0.0.1) 的 Port 上，當這個 Port 收到 TCP 請求後，執行指定的 cmd



## >\_ PPC 介紹

指令：`socat TCP-LISTEN:<Port>,fork EXEC:"<cmd>"`

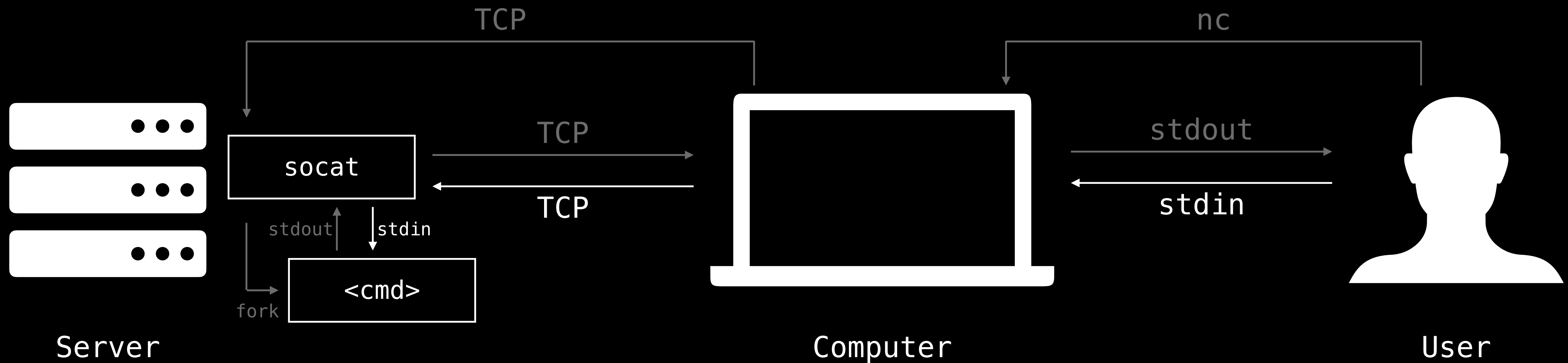
將指定指令的服務綁到本機 (127.0.0.1) 的 Port 上，當這個 Port 收到 TCP 請求後，執行指定的 cmd



## >\_ PPC 介紹

指令：`socat TCP-LISTEN:<Port>,fork EXEC:"<cmd>"`

將指定指令的服務綁到本機 (127.0.0.1) 的 Port 上，當這個 Port 收到 TCP 請求後，執行指定的 cmd





## >\_ PPC 介紹

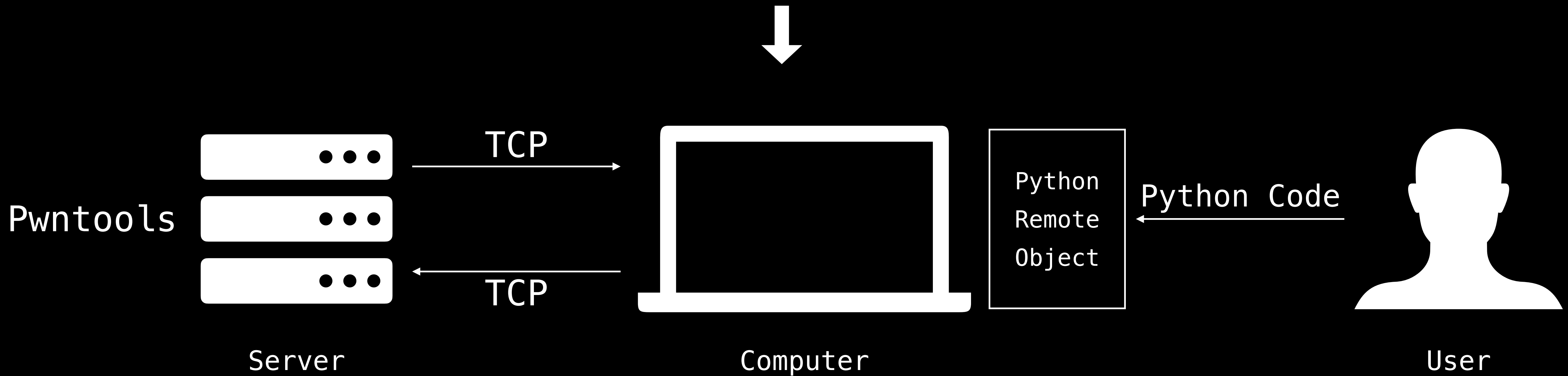
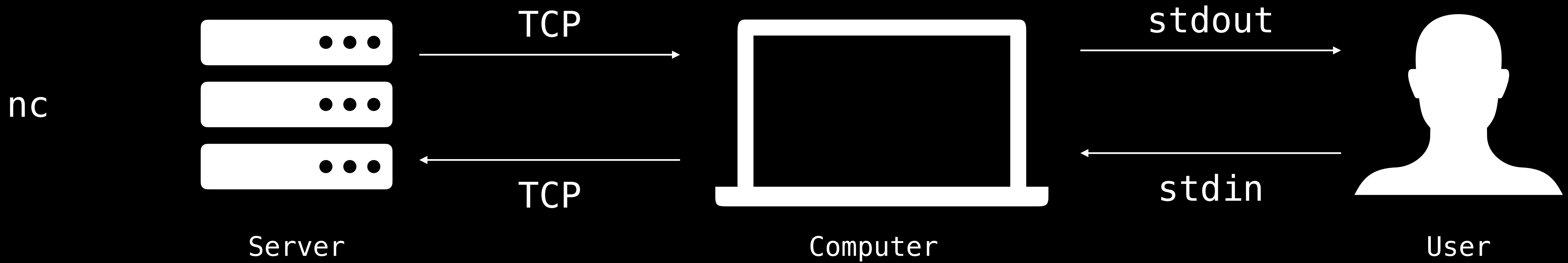
指令：`socat TCP-LISTEN:<Port>,fork EXEC:"<cmd>"`

將指定指令的服務綁到本機 (127.0.0.1) 的 Port 上，當這個 Port 收到 TCP 請求後，執行指定的 cmd

```
curious@Ubuntu-22:~/code/dev$ socat TCP-LISTEN:20000,fork EXEC:"python3 server.py"
```

# Pwntools

## >\_ Pwntools 原理



>\_ Pwntools Installaction

安裝說明

## >\_ Pwntools 基本用法

```
from pwn import *  
  
r = remote('172.104.90.38', 10000)  
  
# Do something with r (remote object)  
  
r.interactive()
```

## >\_ Pwntools 用法

### remote object 方法總結

#### Send Data :

- ▶ `r.send(payload)` : 送出 `payload`
- ▶ `r.sendline(payload)` : 送出 `payload` 並換行
- ▶ `r.sendafter(p1, p2)` : 接收到 `p1` 之後送出 `p2`
- ▶ `r.sendlineafter(p1, p2)` : 接收到 `p1` 之後送出 `p2` 並換行

#### Recv Data :

- ▶ `r.recv(n)` : 接收 `n` 個 bytes
- ▶ `r.recvuntil(payload)` : 接收到 `payload` 為止
- ▶ `r.recvline()` : 接收一行 (到換行為止)
- ▶ `r.recvlines(n)` : 接收 `n` 行

## >\_ Pwntools 用法

### remote object 方法總結

Other :

- ▶ `r.interactive()` : 切換到互動模式
- ▶ `r.close()` : 關閉連線

>\_ Pwntools

PPC Lab : Nth

PPC Lab : Count

PPC Lab : Guess

PPC Lab : Calculator



>\_ Pwntools

## remote object 錯誤

- ▶ 卡死：通常是因為 `recv` 相關的參數錯誤，導致 `server` 已經發完訊息，但 `Pwntools` 還在監聽
- ▶ EOF：通常是因為程式錯誤，`server` 那邊把連線斷掉

