網路管理與系統管理 HW 11

B13902022 賴昱錡

2025 年 5 月 30 日

1. 三角準則的侵略者!?

(a)

1. 2017 年的勒索病毒 Wanacry 全球災情，被病毒感染時，電腦中檔案會被加密，必須支付比特幣才有可能解鎖。違反 CIA 三項，因為駭客能遠端存取、操作你的檔案 (違反 C)，電腦的檔案或服務也可能毀壞、變得不完整 (違反 IA)。
2. Crazy Hunter 組織透過 USB 的途徑感染電腦，駭得馬偕醫院的病患資料。違反 CIA 中的 Confidentiality，因為病患的病例與身份皆為相當敏感、重要，應被更多加密與保護的資訊。

(b)

Assumption:
筆電的硬體設備無故障 (且記憶體、硬碟容量夠)
Threat Model:

Threat Model	Countermeasure
有人嘗試從旁偷窺密碼	安裝螢幕防窺片
有人使用暴力破解密碼 (大量輸入)	採用指紋/臉部辨識解鎖

(c)

Assumption:
傳送簡訊時，必定可以成功發送
Threat Model:

Threat Model	Countermeasure
簡訊內容可以被偽造 (可以在 A 地點，傳送 B 地點條碼所生的簡訊)	使用 GPS 定位使用者的傳送地點
有人手動修改簡訊內容，誤導也浪費實名制系統的資源	對簡訊的格式嚴格過濾 (不符合格式的無法傳送)

(d)

Assumption:

1. 所有人的電腦、網路介面均能正常運作
2. 考生無法連上任何通訊軟體 (Gmail, Discord...etc)

Threat Model:

Threat Model	Countermeasure
有人透過工作站的 `/tmp2/` 存放作答資訊	令考生暫時只能修改 home，封鎖其特定 folder 的權限
有人試著偽造其他組的 token，幫他們作答	作答前組員會取得 OTP，表單須繳交該份 OTP 才算有效

2. 果汁店也有洞 (20 pts)

(a)

(i) DOM XSS

Observe that after we search for something, the query keyword we use will be embedded directly in the page. So we can type following text in the searching textbox, the alert function is then successfully triggered.

```
<iframe src="javascript:alert(`xss`)">
```

(v) View Basket

I open Burp Suite's browser and intercept its traffic, then I observe that when I click `Your Basket`, there's a GET request contains something like `GET /rest/basket/1 HTTP/1.1` (if we login admin), then try to change it to 2 or 3 and forward the request, the content of basket is someone else's.

(b)

3. R-SA！破密部

4. TESTING in the FUZZ (41 pts)

(a)

mutation-based fuzzing is about mutating the existing input values but in lack of understaning the format of the data. generation-based fuzzing is about generating input based on specific or expected format/structure. subsection*(b)

5. 敗北協定太多了！(28 pts)

(a)

攻擊者向 DNS resolver 發送大量的 UDP packet (參數可能是 ANY，讓之後得到的 response 越大越好)，將受害者的 IP 放在其中，因此 DNS resolver 將傳送大量的 response 給受害者，可能導致網路、性能遭到癱瘓。

這個攻擊可以透過減少 public DNS resolver 的數量，以及對 source IP 驗證來解決。

(b)

DNS server 為了加快之後回覆的速度，會存取某 domain 對應到的 IP 作為 cache。Attackers 可以向 DNS server 發出請求，然後當 DNS server 要向 authorative server 請求時，Attackers 可以偽裝成一個合法的 authorative server 向 DNS server 給出扭曲的回應/域名對應到的 IP，也因此 DNS server 之後對於某域名的 cache 都是被污染的。之後受害者去連某域名時，會被 redirect 到駭客提供的 IP。

解決方式可以是透過 DNSSEC 來檢查 DNS 封包的完整性與合法性，或是頻繁更新 DNS cache。

(c)

SPF (Sender Policy Framework)，這個協定會讓接收 email 的 server 去看郵件的來源 server 是否受到接收方 server 的授權 (來自特定的 domain)，如果允許就讓這個郵件通過，反之則自動擋掉。

(d)

(e)

(f)

(g)

6. 猫物語（赤）(34 pts)

(a)

我得到的 flag 是: `NASA_HW11{pseudorandomness_does_not_guarantee_unpredictability}`. 可以觀察到 fatcat.py 得到 random number 的公式是 $k = (ak + c) \mod m$, k 是當前的 state, 其中 a 和 c 是可以透過連續的 3 次 state output 得到的, 假設我們亂猜後得到 3 個正確的 state: k_1, k_2, k_3. 則他們會有以下關係式：

$$k_2 = (ak_1 + c) \mod m$$
$$k_3 = (ak_2 + c) \mod m$$

透過一些簡單的數學，我們可以得到 $a = (k_1 - k_2)^{-1}(k_2 - k_3)$, $c = (k_2 - ak_1) \mod m$ (在這裡 $(k_1 - k_2)^{-1}$ 代表其模 m 下的模逆元)。有了 a 和 c 我們就能預測接下來的每個數字，但 trust 要大於等於 100 才會有 flag1，所以我們就用 pwntools 猜 100 多次，再向 server 索取答案。我的 script 如下所示:

```python
from pwn import *

server = remote('140.112.91.4', 1234)

server.sendlineafter(b'Your choice: ', b'1')
server.sendlineafter(b'Guess a number: ', b'1')
k1 = int(server.recvuntil(b'.').decode().split(' ')[8].rstrip(','))

server.sendlineafter(b'Your choice: ', b'1')
server.sendlineafter(b'Guess a number: ', b'1')
k2 = int(server.recvuntil(b'.').decode().split(' ')[8].rstrip(','))

server.sendlineafter(b'Your choice: ', b'1')
server.sendlineafter(b'Guess a number: ', b'1')
k3 = int(server.recvuntil(b'.').decode().split(' ')[8].rstrip(','))

m = ... # too long to fit in, but it's given by fatcat.py

a = pow(k1-k2, -1, m) * (k2-k3)
c = (k2-a*k1)%m

cur = k3

for i in range(100):
```

```
    server.sendlineafter(b'Your choice: ', b'1')
    cur = (a*cur+c)%m
    server.sendlineafter(b'Guess a number: ', str(cur))

server.sendlineafter(b'Your choice: ', b'2')
flag = server.recvuntil(b'}')
print(flag)
```

(b)

The flag (FLAG2) is `NASA_HW11{`Z\iW^b8$s"f\I[P<"}`. Observe that the key is used in a cyclic manner. Since we know the prefix of flag must be `NASA_HW11{`, and its length is equal to the OTP key (it's 10). By xor the string we got in remote server and the prefix, we can obtain the key, thus the original string.

```
import binascii

tmp = # too long to fit in
tmp = binascii.unhexlify(tmp)

prefix = b'NASA_HW11{'
key = bytes([tmp[i] ^ prefix[i] for i in range(10)])

full_flag = bytes([tmp[i] ^ key[i % 10] for i in range(len(tmp))]).decode()
print(full_flag)
```

(c)

The flag (FLAG3) is `NASA_HW11{https://youtu.be/1GxwDuV5JMc}`. Just brute force since 2^{24} is not that big, so we can store all the answer and its corresponding key in dictionary. (for simplicity, I don't consider the collisions). My script is shown as below:

```
from pwn import *
import hashlib

mp = dict()
for i in range(2**24):
    mp[hashlib.md5(str(i).encode()).hexdigest()[0:8]] = i

server = remote('140.112.91.4', 1234)
```

```
server.sendlineafter(b'Your choice: ', b'4')

for i in range(10):
    prefix = server.recvuntil(b': ').decode().split('
    ↪  ')[7].strip('"').rstrip('"')
    ans = str(mp[prefix])
    server.sendline(ans)

ans1 = server.recvline()
ans2 = server.recvline()
print(ans1.decode(), ans2.decode())
```

(d)

FLAG4 是 `NASA_HW11{y0u_KN0w_r3F13C710n_4774cK}`，觀察 fatcat.py 可以發現 verify 的時候名字一點都不重要，所以只要讓 sha256 的結果是對的就行，而同時我們要根據 server 給的 nounce 輸入正確的結果，但似乎需要 shared key 欸？其實只要把 nounce 丟進 prover 然後把 server 輸出的 mac 複製下來，再回到 verify 的過程中，輸入 `richardlaiis||<mac>`，就可以取得 flag 了，好耶！