



# NEBULA LEVEL 06

Programmazione Sicura 2024/2025

Mario Lezzi 0522501840

Daniele Dello Russo 0522501766

# WORKFLOW

01

INTRODUZIONE

02

ALBERO DI ATTACCO

03

SOLUZIONE

04

DEBOLEZZE E VULNERABILITA'  
INDIVIDUATE

05



MITIGAZIONE



01

# Introduzione

Level 06



# Level 06

*" The **flag06** account credentials came from a legacy unix system.*

*To do this level, log in as the **level06** account with the  
password **level06**. Files for this level can be found in **/home/flag06**."*



# OBIETTIVI

- Recupero della password dell'utente *flag06*
- Autenticazione come utente *flag06*
- Esecuzione di */bin/getflag* con i privilegi di *flag06*







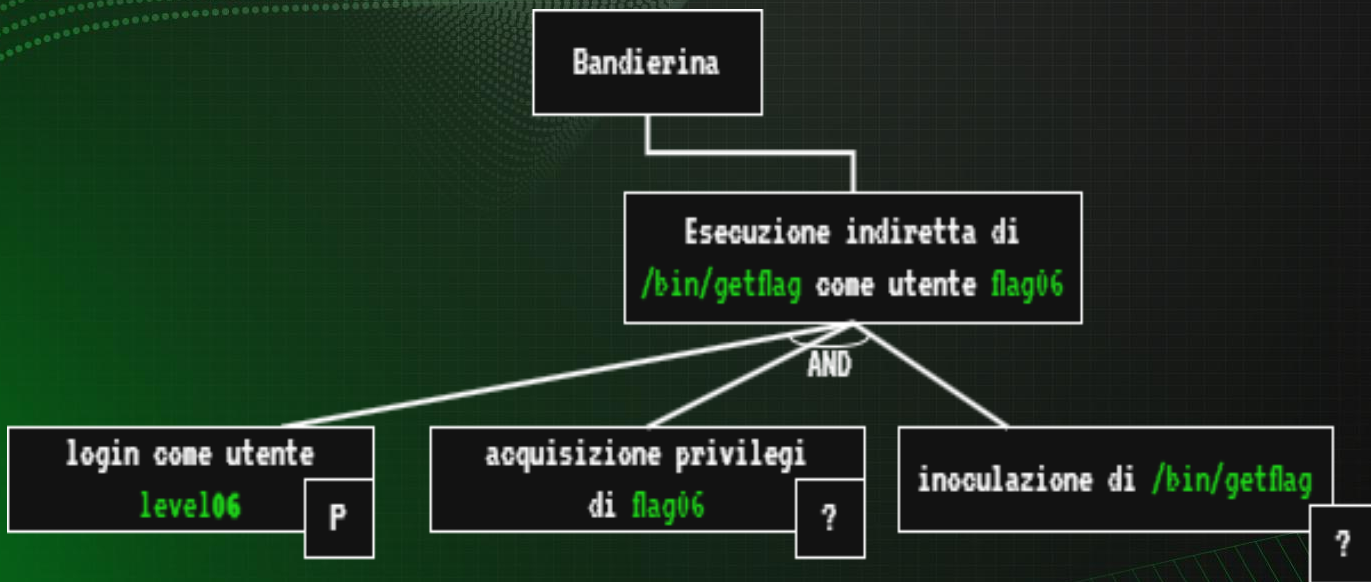
02

# Albero di attacco

Level 06



# Esecuzione indiretta



- La descrizione del livello su <https://exploit.education/nebula/level-06/> ci consiglia di cercare file utili alla sfida nella directory `/home/flag06`
- facciamo l'accesso con le credenziali di `level06` e cerchiamo file utili alla nostra sfida...



- Vediamo quali home directory sono a disposizione dell'utente `level06`:

```
ls /home/level*
```

```
ls /home/flag*
```

- L'utente `level06` può accedere solamente alle directory

```
/home/level06
```

```
/home/flag06
```

# /home/flag06

```
level06@nebula:~$ ls -la /home/flag06
total 5
drwxr-x--- 2 flag06 level06   66 2011-11-20 20:51 .
drwxr-xr-x 1 root    root     80 2012-08-27 07:18 ..
-rw-r--r-- 1 flag06 flag06  220 2011-05-18 02:54 .bash_logout
-rw-r--r-- 1 flag06 flag06 3353 2011-05-18 02:54 .bashrc
-rw-r--r-- 1 flag06 flag06   675 2011-05-18 02:54 .profile
```

## /home/flag06

- Nella directory ci sono solo file di configurazione di Bash, tutti leggibili ma nessuno è un binario o uno script eseguibile
- La descrizione della sfida ci ha però detto che all'interno di questi file potrebbero esserci indizi
- Esaminiamo il contenuto...

- Oltre alle configurazioni standard, troviamo questo blocco di codice, che prevede il caricamento opzionale di un file esterno:

`~/.bash_aliases`

```
if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi
```

- se questo file esiste, viene caricato automaticamente ogni volta che l'utente fa login oppure apre una shell
- Nel nostro caso, questo file non è ancora presente...



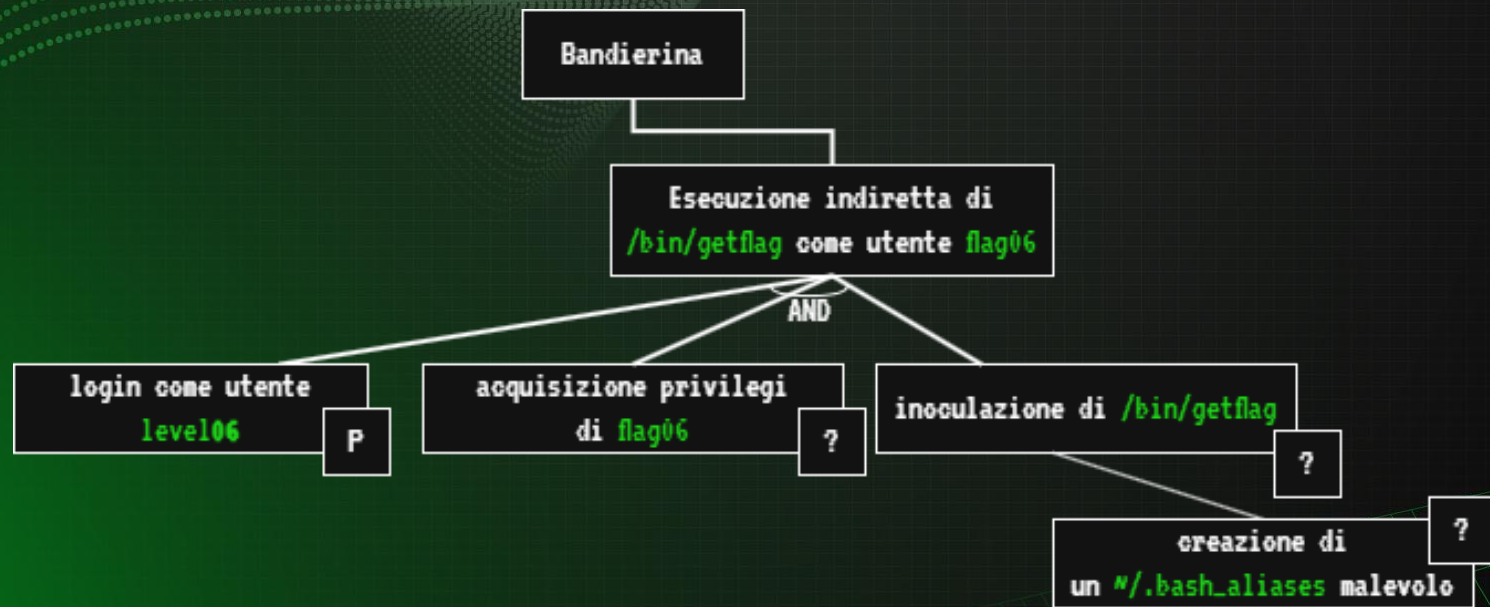
# Idea



creiamo un `~/.bash_aliases` contenente `/bin/getflag`

- Quando flag06 apre una shell, il file viene caricato automaticamente
- Se accade, `/bin/getflag` verrebbe eseguito con i suoi privilegi!

# Aggiornamento albero di attacco



# Iniezione di `/bin/getflag`



- Iniettiamo `/bin/getflag` nel file `.bash_aliases`

```
level06@nebula:~$ echo '/bin/getflag' > ~/.bash_aliases
level06@nebula:~$ ./bash_aliases
-sh: ./bash_aliases: Permission denied
```

- `.bash_aliases` è un file di configurazione Bash, non un eseguibile
- Non può essere lanciato con `./`
- Lo eseguiamo manualmente con `source`

# Iniezione di /bin/getflag



- `source` carica ed esegue il contenuto del file nella shell attuale

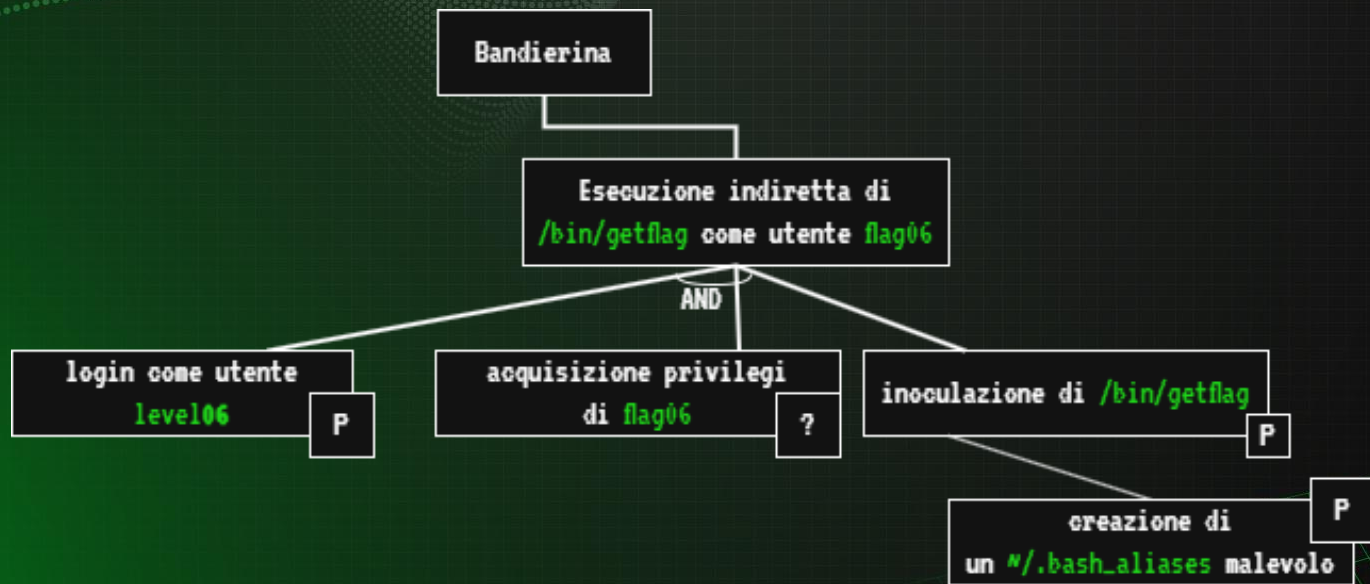
```
level06@nebula:~$ echo '/bin/getflag' > ~/.bash_aliases
level06@nebula:~$ ./bash_aliases
-sh: ./bash_aliases: Permission denied
level06@nebula:~$ source ~/.bash_aliases
getflag is executing on a non-flag account, this doesn't count
```

- Il comando viene eseguito con i permessi di `level06`!



- Anche se il file `.bash_aliases` viene eseguito, ciò accade dopo l'autenticazione, quindi i comandi girano con i permessi di `level06`
- Non possiamo forzare `flag06` ad aprire una shell (serve un suo login reale)
- Possiamo ottenere l'esecuzione privilegiata solo sfruttando un eseguibile con bit SETUID acceso
- Cerchiamo...

# Aggiornamento albero di attacco

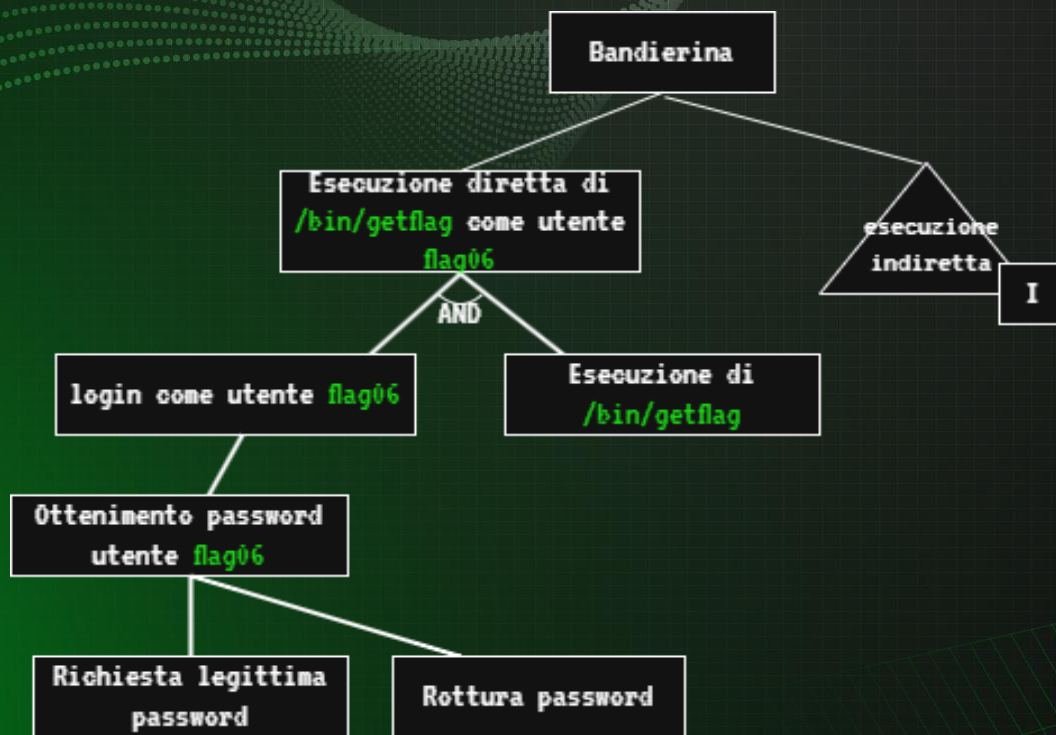


# Fallimento della strategia

- Abbiamo già visto che `/home/flag06` non contiene file con bit SETUID acceso
- Anche `/home/level06` non ha file vulnerabili
- Ricerca globale con `find / -perm /u+s 2>/dev/null` ma nulla



# Aggiornamento albero di attacco





# Richiesta password

- A chi si potrebbe chiedere la password dell'account **flag06**? Al legittimo proprietario (creatore della macchina virtuale Nebula)
- Il legittimo proprietario sarebbe disposto a darci la password? **NO**! Altrimenti che sfida sarebbe?
- Si deduce che la richiesta legittima della password non è una strada percorribile...

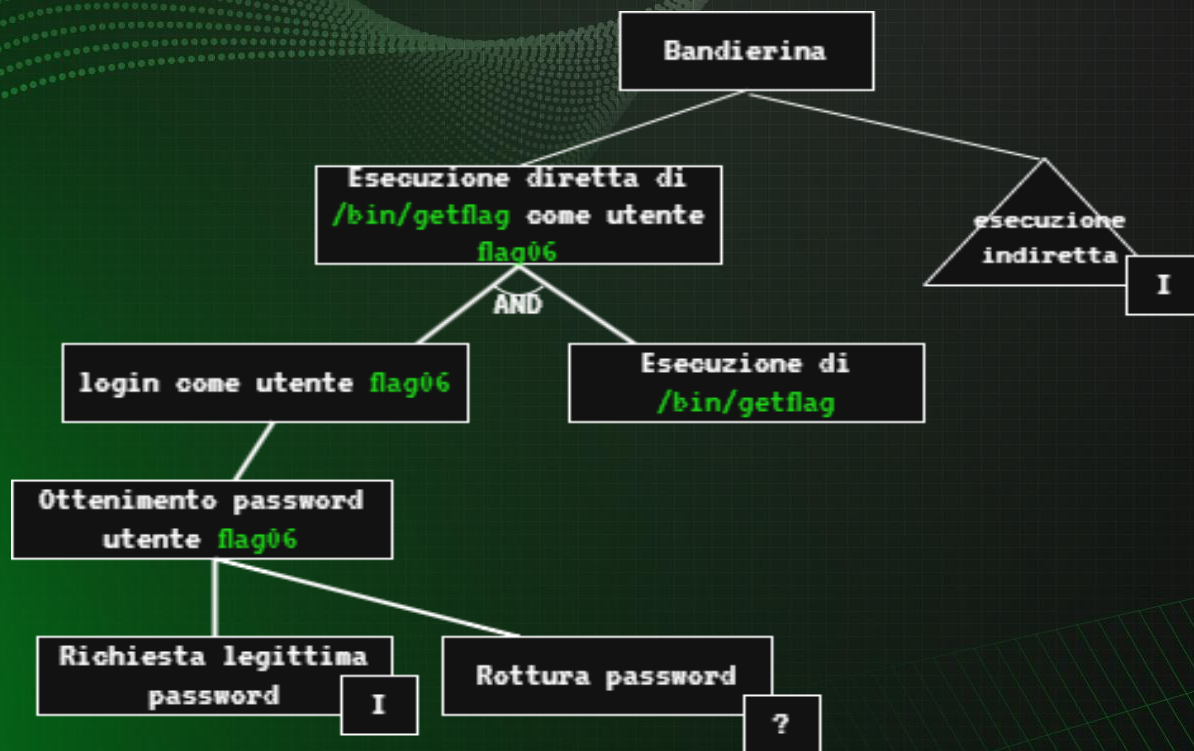
# Possibile rottura della password?

- Il sito ufficiale non ci ha solo suggerito di indagare in `/home/flag06/`, ma ci ha dato anche un altro indizio importante...

*"The **flag06** account credentials came from a legacy unix system."*



# Aggiornamento albero di attacco



# man passwd

«La sicurezza di una password dipende dalla forza dell'algoritmo di crittografia e dalla dimensione dello spazio key. Il metodo di crittografia legacy UNIX System si basa sull'algoritmo NBS DES. Ora sono consigliati metodi più recenti (vedere ENCRYPT\_METHOD).»

## FILES

[/etc/passwd](#)

Informazioni sull'account utente.

[/etc/shadow](#)

Protegge le informazioni  
sull'account utente.



# File Format conversion

man 5 passwd ci dice come è strutturato  
`/etc/passwd` :

- Nome utente
- Hash della password (opzionale)
- UID
- GID
- Comment field (info aggiuntive)
- Home directory path
- Shell utilizzata

«Se il campo password è una "x" minuscola, allora la password crittografata viene in realtà memorizzata nel file `shadow(5)`; deve esserci un record corrispondente nel file `/etc/shadow`, altrimenti l'account utente non è valido. Se il campo password è una stringa qualsiasi, allora verrà trattato come una password crittografata, come specificato da `crypt(3)`.»

# UNIX moderni

Gestione delle password separata tra:

- `/etc/passwd` contiene informazioni sugli account degli utenti (UID,GID), directory home, shell di default. Leggibile da tutti
- `/etc/shadow` contiene gli hash delle password degli utenti in formato cifrato. Accessibile solo da root

# UNIX LEGACY

- hash della password nel secondo campo di ogni record in `/etc/passwd`
- Accessibile a tutti
- Possibile rottura della password

# /etc/passwd

non possiamo leggere il contenuto di `/etc/shadow`. Vediamo cosa c'è in `/etc/passwd`:

```
flag04:x:995:995:~/home/flag04:/bin/sh
level05:x:1006:1006:~/home/level05:/bin/sh
flag05:x:994:994:~/home/flag05:/bin/sh
level06:x:1007:1007:~/home/level06:/bin/sh
flag06:ueqw0CnSGdsuM:993:993:~/home/flag06:/bin/sh
level07:x:1008:1008:~/home/level07:/bin/sh
flag07:x:992:992:~/home/flag07:/bin/sh
level08:x:1009:1009:~/home/level08:/bin/sh
flag08:x:991:991:~/home/flag08:/bin/sh
```



# /etc/passwd

Nel secondo campo di **flag06** compare l'hash della password...

```
level06@nebula:~$ grep flag06 /etc/passwd  
flag06:ueqw0CnSGdsuM:993:993::/home/flag06:/bin/sh
```



# /etc/passwd

Nome utente

UID

GID

Home directory path

```
flag06:ueqwOCnSGdsuM:993:993::/home/flag06:/bin/sh
```

Hash della password

Comment field  
(vuoto)

Shell utilizzata

# John The Ripper

In Kali Linux è incluso di default un software chiamato **John the Ripper**, progettato per il cracking delle password.

- supporta molteplici algoritmi di hashing: DES, MD5, SHA, bcrypt, ecc.
- supporta diverse tecniche di rottura delle password...



# Wordlist attack

- provare una lista di parole predefinite come possibili password
- Veloce ed efficace contro password comuni o deboli





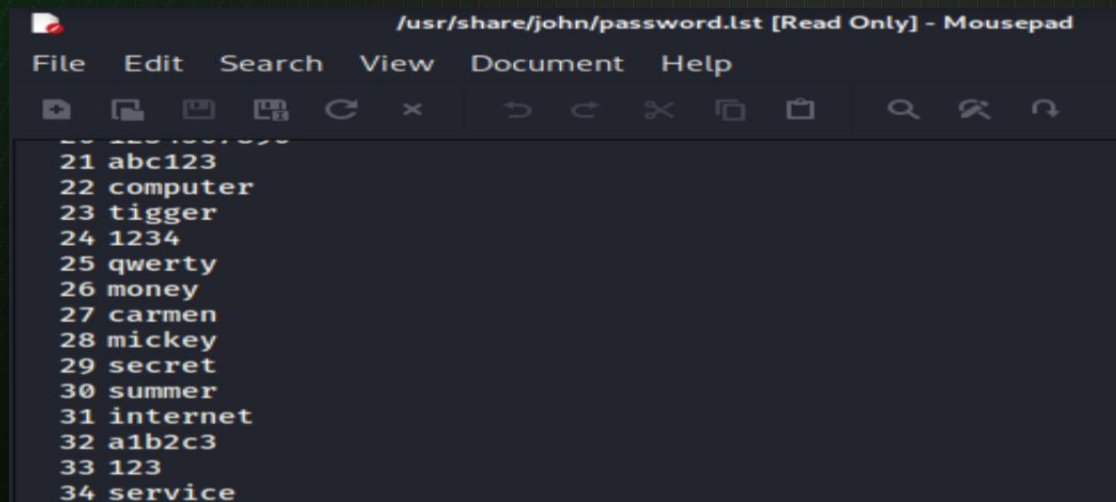
# Brute-force attack

- Prova tutte le combinazioni di possibili caratteri fino a trovare la password
- Estremamente lento
- Utile se si conosce la lunghezza della password



## Wordlist attack

- JTR ha una wordlist di default `password.lst`
- Centinaia di parole semplici e frequenti utilizzate come password



# Wordlist attack

```
(kali㉿kali)-[~]  
$ echo flag06:ueqwOCnSGdsuM > crackPasswd.txt  
  
(kali㉿kali)-[~]  
$ john --show crackPasswd.txt  
0 password hashes cracked, 1 left
```



# Wordlist attack

Cosa non ha funzionato?

- Abbiamo dato a JTR un file contenente un hash
- Non avendo specificato l'algoritmo di cifratura l'esecuzione fallisce
- Quale algoritmo è stato utilizzato per generare l'hash della password?





# man passwd

«La sicurezza di una password dipende dalla forza dell'algoritmo di crittografia e dalla dimensione dello spazio key. Il metodo di crittografia legacy UNIX System si basa sull'algoritmo NBS DES. Ora sono consigliati metodi più recenti (vedere ENCRYPT\_METHOD).»

La funzione di cifratura (**crypt()**) dei sistemi UNIX legacy utilizza l'algoritmo **NBS DES**...

# NBS DES

Versione modificata di DES utilizzata da `crypt()`. Approvato nel 1977 dall'ente statunitense National Bureau of Standards (NBS)

Principali differenze:

- Cifratura su `stringa fissa` (tutti zeri)
- `25 round` DES
- Utilizzo del `salt` per modificare la cifratura

# NBS DES - salt

stringa casuale di 2 caratteri scelta dall'insieme [a-zA-Z0-9./].

- Genera **4096 varianti** del DES
- rende unico l'output anche per password identiche
- Protegge da attacchi con rainbow table

# NBS DES - output

stringa di 13 caratteri ASCII:

ueqwOCnSGdsuM

salt

Hash della password



# Wordlist attack

```
(kali@kali)-[~]  
$ john --format=descrypt crackPasswd.txt  
Using default input encoding: UTF-8  
Loaded 1 password hash (descrypt, traditional crypt(3) [DES 128/128 SSE2])  
Will run 2 OpenMP threads  
Proceeding with single, rules:Single  
Press 'q' or Ctrl-C to abort, almost any other key for status  
Almost done: Processing the remaining buffered candidate passwords, if any.  
Proceeding with wordlist:/usr/share/john/password.lst  
hello (flag06)  
1g 0:00:00:00 DONE 2/3 (2025-04-13 15:28) 5.263g/s 68789p/s 68789c/s 68789C/s  
123456..Herman1  
Use the "--show" option to display all of the cracked passwords reliably  
Session completed.  
  
(kali@kali)-[~]  
$ john --show crackPasswd.txt  
flag06:hello  
  
1 password hash cracked, 0 left
```



La password è  
«hello»!!!

Siamo riusciti a rompere  
la password...





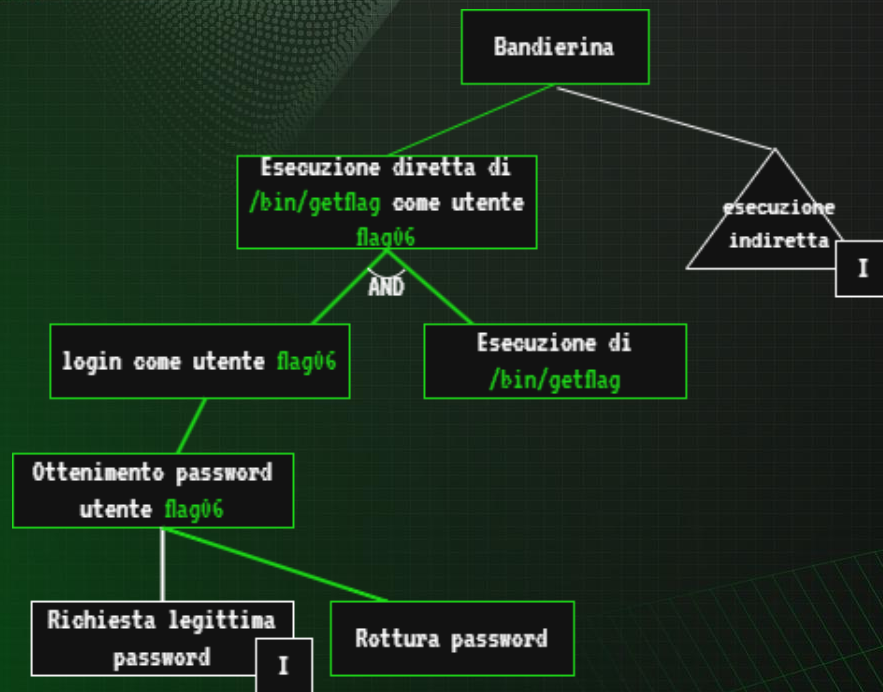
03

Soluzione

Level 06



# Aggiornamento albero di attacco





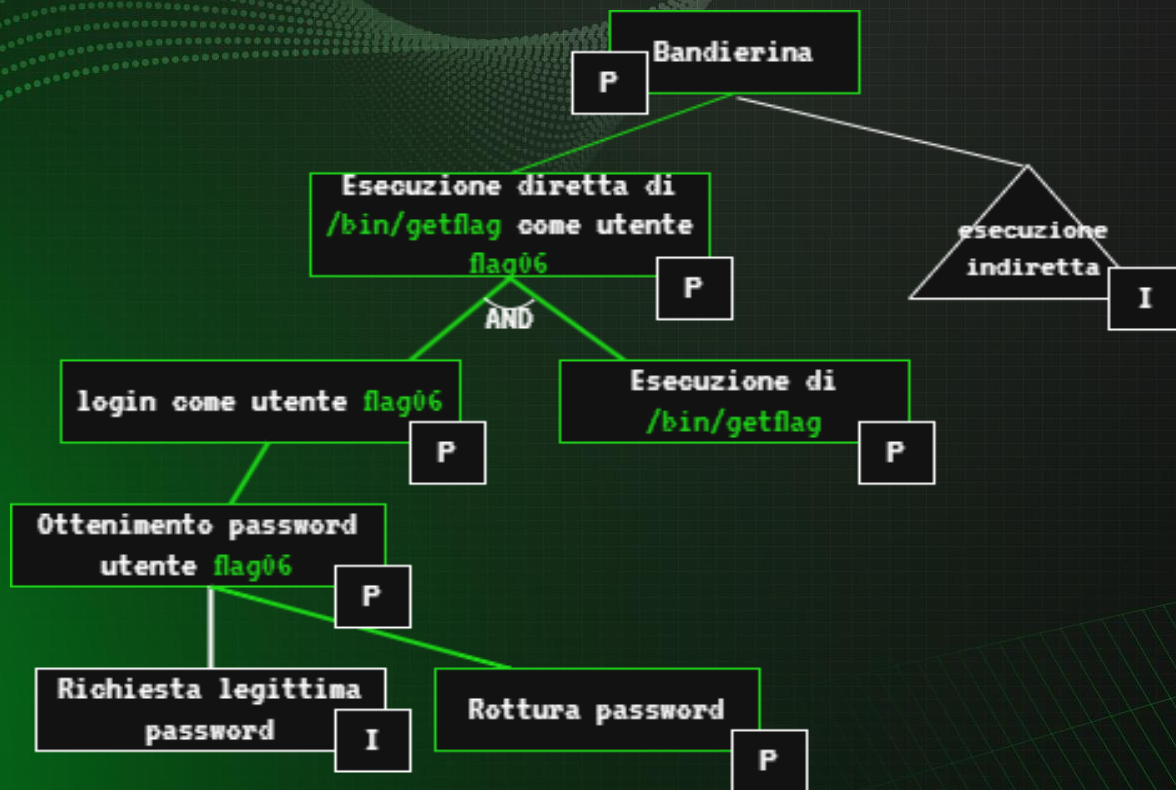
# Aggiornamento albero di attacco

- Nell'albero di attacco sono colorati in verde i nodi e gli archi che rappresentano le azioni da effettuare

Tali azioni sono eseguibili dall'utente level06?

- Ottenimento della password dell'utente flag06: SI
- Login come utente flag06: SI
- Esecuzione diretta di `/bin/getflag` come utente flag06: SI

# Aggiornamento albero di attacco



# Sfida vinta!

```
level06@nebula:~$ su flag06  
Password:  
sh-4.2$ whoami  
flag06  
sh-4.2$ getflag  
You have successfully executed getflag on a target account
```



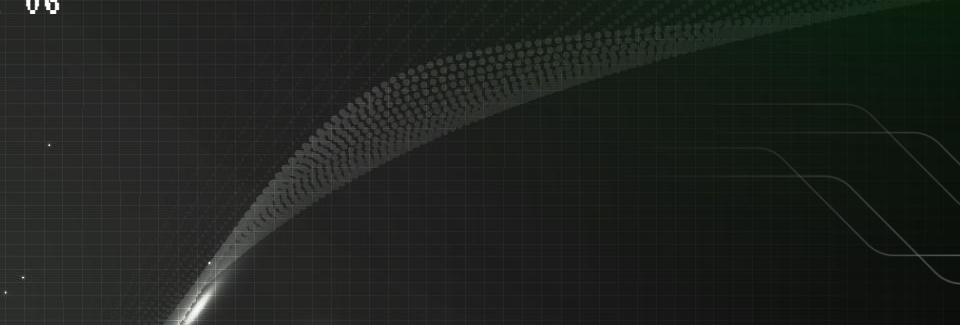




04

# Debolezze e vulnerabilità

Level 06





# Debolezze [1/2]

- CWE-200 Exposure of Sensitive Information to an Unauthorized Actor: l'utente level06 ha accesso all'hash di flag06
- CWE-312 Cleartext Storage of Sensitive Information  
l'hash è salvato in un file accessibile a tutti

## Debolezze [2/2]

- CWE-257 Storage of Passwords in a Recoverable Format:  
l'hash di flag06 è generato da un algoritmo di hashing vulnerabile
- CWE-328 Use of weak hash l'attaccante può risalire facilmente alla password partendo dall'hash
- CWE-521 Weak Password Requirements  
utilizzo di password facilmente indovinabile

# Vulnerabilità



- CVE-2025-27595 Weak hashing algorithm: il sistema utilizza un algoritmo di cifratura debole. Un aggressore può facilmente calcolare una password corrispondente



05

# Mitigazione

Level 06





# Aggiornamento password

- Comando `passwd` per aggiornare
- Scegliere password lunga e complessa
- Evitare parole comuni

```
flag06@nebula:~$ passwd
Changing password for flag06.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

# Aggiornamento password

- Generato nuovo hash
- Nebula utilizza **SHA-512**  
(sicuro, fino a 16  
caratteri per il salt)
- Hash ancora scoperto...

```
flag06@nebula:~$ grep ENCRYPT_METHOD /etc/login.defs
# This variable is deprecated. You should use ENCRYPT_METHOD.
ENCRYPT_METHOD SHA512
# Only used if ENCRYPT_METHOD is set to SHA256 or SHA512.
```

```
flag06@nebula:~$ grep flag06 /etc/passwd
flag06:$6$dJoBSOWT$jITtx0wZRrytajV4n0VuxiXg.n8pW51Lw3iWFBpTf4o4QMAde6QRUbI9mDed8
P9PS9s1RGtu09.wsfABuRQB21:993:993::/home/flag06:/bin/sh
```

Id SHA-512

salt

password

# Password shadowing

- `!` indica che la password non è ancora in `/etc/shadow`
- `pwconv` per spostare gli hash da `/etc/passwd` a `/etc/shadow`
- `pwconv` mette una `'x'` al posto dell'hash in `/etc/passwd`
- Hash protetto!!!




```
nebula@nebula:~$ grep flag06 /etc/passwd
flag06:$6$ezNBfUXp$50hu2HZSPSA05IhRdkVdI8ndulreuJe0VKCxnXGq9eZXknkb6/5AhGMKNaAwE
rDtJsJAR7IF8/ncJHfA6SoSx0:993:993::/home/flag06:/bin/sh
nebula@nebula:~$ sudo grep flag06 /etc/shadow
flag06:!:5299:::
```

```
nebula@nebula:~$ sudo pwconv
nebula@nebula:~$ grep flag06 /etc/passwd
flag06:x:993:993::/home/flag06:/bin/sh
nebula@nebula:~$ sudo grep flag06 /etc/shadow
flag06:$6$ezNBfUXp$50hu2HZSPSA05IhRdkVdI8ndulreuJe0VKCxnXGq9eZXknkb6/5AhGMKNaAwE
rDtJsJAR7IF8/ncJHfA6SoSx0:20197:::
```

# Nuovo attacco

- Proviamo ad effettuare un nuovo attacco
- Autentichiamoci come **level106**
- Proviamo a leggere il contenuto di **/etc/passwd**

```
level106@nebula:~$ grep flag06 /etc/passwd  
flag06:x:993:993::/home/flag06:/bin/sh
```

- Impossibile recuperare l'hash
- **x** nel secondo campo del record **flag06**
- Attacco fallito! 



YOU THANK  
FOR YOU  
YOUR FOR  
PATIENCE YOUR  
PATIENCE



- Dello Russo **Daniele**
- Lezzi **Mario**