

JASMINE C. OMANDAM

Hurray! You solved this challenge.

Challenge	Type	Difficulty	Solves	Completion %
Flag Hunters	Reverse Engineering	Easy	20,646 solves	89%
Quantum Scramb	Reverse Engineering	Easy	3,587 solves	
Tap into Hash	Reverse Engineering	Medium	3,249 solves	86%
perplexed	Reverse Engineering	Medium	2,279 solves	
packer	Reverse Engineering	Medium	8,190 solves	88%
FactCheck	Reverse Engineering	Medium	3,115 solves	

picoCTF Webshell

The ether's ours to conquer, picoCTF{70637h3r_f0r3v3r_b248b032}

[REFRAIN]
We're flag hunters in the ether, lighting up the grid,
No puzzle too dark, no challenge too hid.
With every exploit we trigger, every byte we decrypt,
We're chasing that victory, and we'll never quit.
Crowd:
Pico warriors rising, puzzles laid bare,
Solving each challenge with precision and flair.
With unity and skill, flags we deliver,
The ether's ours to conquer, picoCTF{70637h3r_f0r3v3r_b248b032}

[REFRAIN]
We're flag hunters in the ether, lighting up the grid,
No puzzle too dark, no challenge too hid.
With every exploit we trigger, every byte we decrypt,
We're chasing that victory, and we'll never quit.
Crowd:
Pico warriors rising, puzzles laid bare,
Solving each challenge with precision and flair.
With unity and skill, flags we deliver,
The ether's ours to conquer, picoCTF{70637h3r_f0r3v3r_b248b032}

[REFRAIN]
We're flag hunters in the ether, lighting up the grid,
No puzzle too dark, no challenge too hid.
With every exploit we trigger, every byte we decrypt,
We're chasing that victory, and we'll never quit.
Crowd:
;RETURN 0
urjas_mine-picoctf@webshell:~\$ ^C
urjas_mine-picoctf@webshell:~\$]

Flag Hunters

Easy Reverse Engineering picoCTF 2025 browser_webshell_solvable

AUTHOR: SYREAL

Description

Lyrics jump from verses to the refrain kind of like a subroutine call. There's a hidden refrain this program doesn't print by default. Can you get it to print it? There might be something in it for you.

The program's source code can be downloaded [here](#). Additional details will be available after launching your challenge instance.

This challenge launches an instance on demand. Its current status is: NOT_RUNNING

Launch Instance

Hints ?

1 2 3

When I first opened the challenge *Flag Hunters* on picoCTF, I noticed the description compared verses and refrains to subroutine calls. That immediately made me think: there's a hidden function or output that isn't printed by default, and my job is to coax it out.

I launched the instance and connected with netcat:

```
bash  
nc verbal-sleep.picoctf.net <port>
```

The program greeted me with poetic verses about hacking, exploits, and flag hunting. At the end of each stanza, it prompted me with:

Code
Crowd:

That prompt was my entry point. The hints reinforced my suspicion:

- “Unsanitized user input is always good, right?” → the program was evaluating my input directly.
- “Is there any syntax ripe for subversion?” → I should try calling functions or injecting code.
- “Lyrics jump from verses to the refrain...” → there must be a hidden function named something like `refrain`.

So I typed:

```
refrain()
```

Suddenly, the program printed new verses — different themes like forensics, binary reversing, cryptography, and web exploitation. Each time I called `refrain()`, it revealed another hidden stanza. I kept repeating the call, cycling through these hidden refrains.

Finally, after enough iterations, the program printed a special line that stood out from the lyrical text:

picoCTF{70637h3r_f0r3v3r_b248b032}

That was the flag. I copied it directly from the netcat output and submitted it on the picoCTF site.

Reflection

The challenge was a clever play on the idea of subroutine calls. By recognizing that my input was unsanitized and directly evaluated, I realized I could invoke hidden functions. Persistence repeatedly calling `refrain()` — eventually revealed the flag embedded in the poetic output. It was a simple but elegant exercise in spotting hidden functionality and exploiting unsanitized input.