So, we get a service in which we can see a message and decrypt a message. The only feedback we can get, after a bit of testing are invalid padding and Ack. This means that we are probably dealing with a padding oracle attack on AES-CBC.

But how did I arrive to that conclusion?

Well, first, looking at the ciphertext it is faintly reminiscent of AES, since AES ciphertexts are generally in base64. Then, given our feedback, it must be something concerning the padding, so a little googling led me to the conclusion we need to perform a padding oracle attack.

After looking through some tools I found https://github.com/stephenbradshaw/python-paddingoracle/blob/0af070cba117eec555a4157095d9474a603df616/README.rst, which seems to help, as it easily allows us to build the oracle logic. Just make sure to run the script from inside the git clone directory. I had a bit of trouble customizing it, but the end result is as follows:

```
from paddingoracle import BadPaddingException, PaddingOracle
from pwn import *
import base64
context.log_level="error"
class PadBuster(PaddingOracle):
    def __init__(self, **kwargs):
        super(PadBuster, self).__init__(**kwargs)
        self.counter=1
    def oracle(self, data):
        r.recvuntil(": ")
        r.sendline("2")
        r.recvuntil(":")
        r.sendline(base64.b64encode(data))
        inv = r.recvline().decode()
        if "invalid" in inv:
            raise BadPaddingException
        print(self.counter) # to check it's working and not stuck
        self.counter+=1
        return

r = remote("35.246.139.54", 32232)
r.recvuntil(": ")
r.sendline("1")
r.recvuntil(": ")
ciphertext = base64.b64decode(r.recvline().decode())
padbuster = PadBuster()
print(padbuster.decrypt(ciphertext, block_size=16))
```

Now, whether you made your own script or used the one above, it's going to take a while. So, extend the expire period, make yourself some tea, maybe eat a meal and wait for the program to finish execution. Once it's finished the flag will be printed. It took me about 45 minutes.

Made with love by: AndreiCat