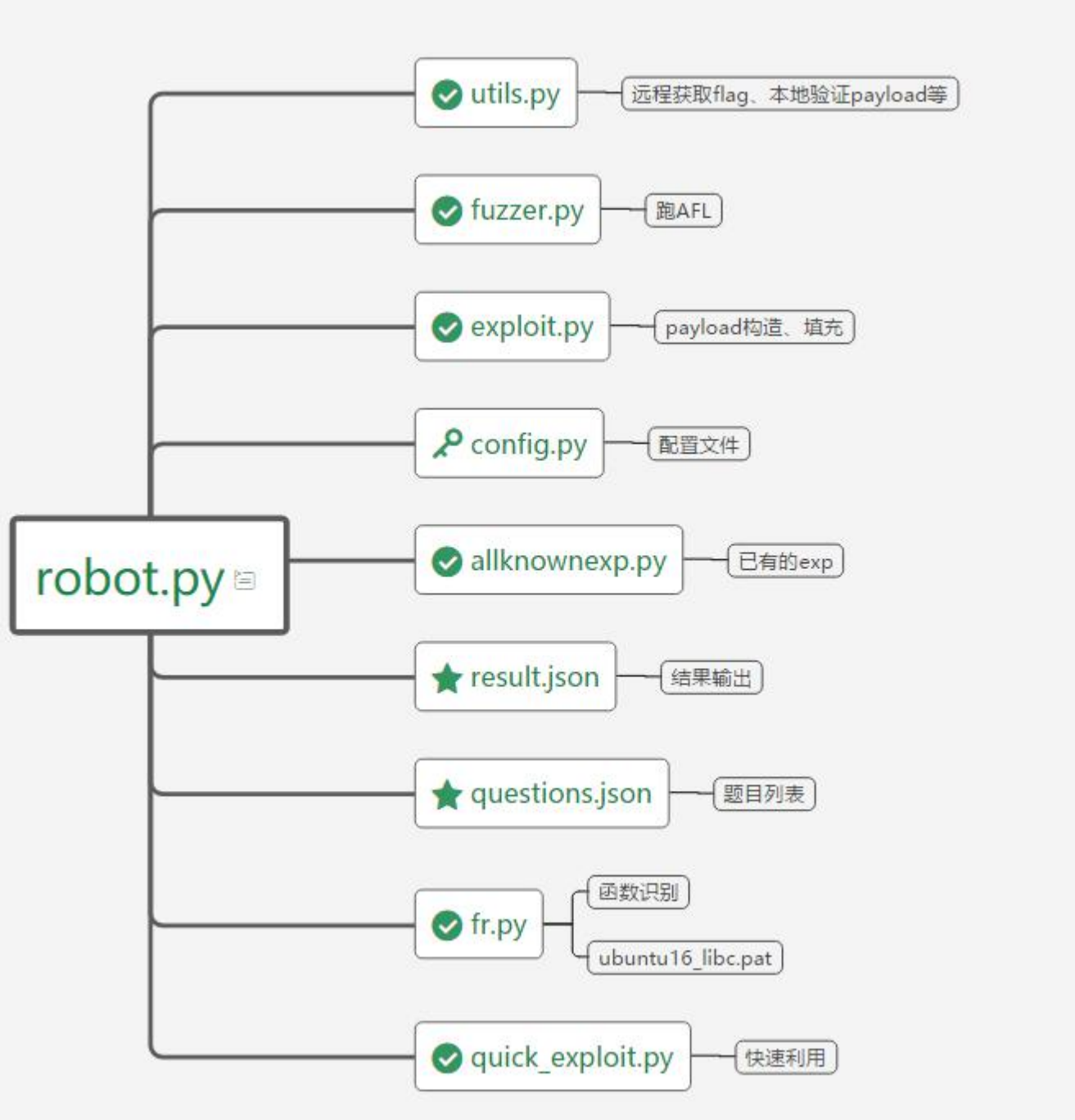
Introduction

项目主要文件：


robot.py
utils.py
远程获取flag、本地验证payload等
fuzzer.py
跑AFL
exploit.py
payload构造、填充
config.py
配置文件
allknownexp.py
已有的exp
result.json
结果输出
questions.json
题目列表
fr.py
函数识别
ubuntu16_libc.pat
quick_exploit.py
快速利用

robot.py主要功能

```
def download_questions_info(download_binary=True):
```

```
def download_binarys(questions):
```

```
def check_same_binary():
```

```
def brute_same_binary():
```

```
def quick_exploit_in_fuzz(binary):
```

```
def submit_flag():
```

```
def fuzz_and_exploit():
```

- 主函数：

```
504 def main():
505     init()
506
507     download_questions_info()
508
509     check_same_binary()
510
511     submit_flag()
512
513     quick_exploit_before_fuzz()
514
515     fuzz_and_exploit()
```

- results.json

robot.py | 签到.py | results.json | utils.py | fuzzer.py | exploit.py | Find Results


```
1 {"binary": "/aictf/bin/bin8", "flag": "flag{f521a588-3d7f-11e8-98ae-5254003be4ba}", "payload": "\n\n1\n\n\u00001\
```

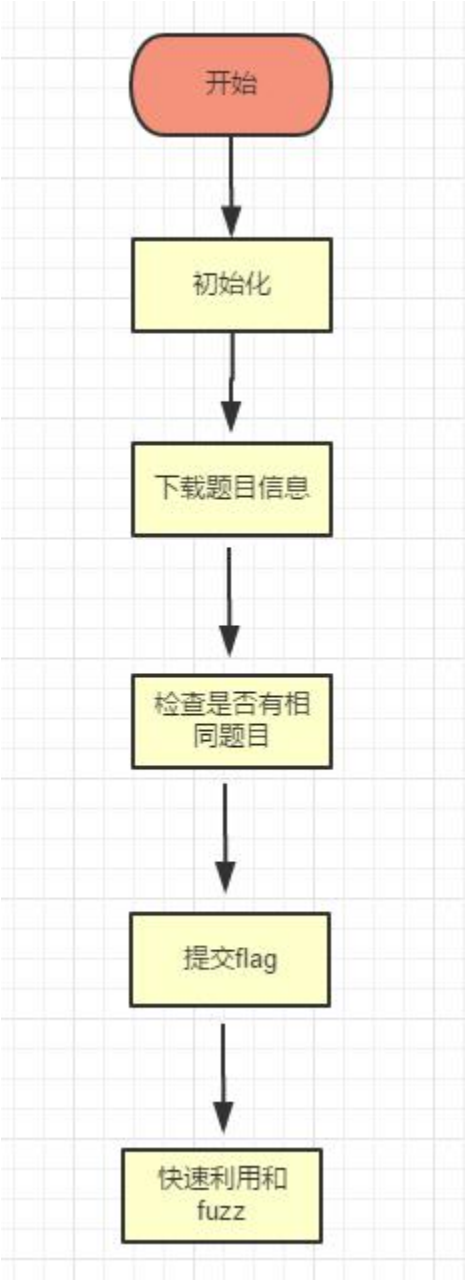

- questions.json

robot.py | 签到.py | questions.json | utils.py | fuzzer.py | exploit.py


```
1 [{"vm_name": "pwn1", "score": 0, "vm_ip": "111.206.245.37", "challengeID": 1, "binaryUrl": "http
```

运行流程

- 线程池：
  提高效率
- 死循环：
  不停歇



Review

具体实现

主函数：

```
504 def main():
505     init()
506
507     download_questions_info()
508
509     check_same_binary()
510
511     submit_flag()
512
513     quick_exploit_before_fuzz()
514
515     fuzz_and_exploit()
```

➢ init()：两个作用

```
def init():
    if not os.path.isdir(config.work_path):
        os.mkdir(config.work_path)

    utils.kill_process('afl-fuzz')
```

python >
os.path.isdir()函数 判断某一路径是否为目录
os.mkdir()函数 创建目录

- config.py 配置文件

```
robot.py    config.py
# 题目下载目录
work_path = '/aictf/bin'
```

```
from config import work_path, questions_file, result_file
questions_path = os.path.join(os.path.abspath('.'), questions_file)
result_path = os.path.join(os.path.abspath('.'), result_file)
```

- utils.py 杀掉进程

```
def kill_process(name):
    cmd = "ps aux | grep " + name + " | grep -v grep | awk '{print $2}' | xargs kill -9"
    os.system(cmd)
```

➢ Linux命令：ps/grep/awk/kill

- PID

```
Jack_t0m@ubuntu:~$ps aux
USER       PID %CPU %MEM
root         1  0.1  0.1
root         2  0.0  0.0
root         3  0.0  0.0
```

- kill -9 pid

主函数：

```
504 def main():
505     init()
506
507     download_questions_info()
508
509     check_same_binary()
510
511     submit_flag()
512
513     quick_exploit_before_fuzz()
514
515     fuzz_and_exploit()
```

◻download_questions_info() 下载题目信息和题目

```
等待开题，死循环下载题目信息，当比赛开始，成功下载题目信息和binary后返回
"""
def download_questions_info(download_binary=True):
    while True:
        try:
            r = requests.get(config.questions_url, auth=(config.user, config.password), headers=headers)

            if r.status_code == 200:
                data = r.json()
                #print data

                questions = data['AiChallenge']
                open(questions_path, 'wb').write(json.dumps(questions))

                if not download_binary:
                    return True

                if download_binarys(questions):
                    print "Donwload questions and binarys success!"
                    return True

        except Exception as e:
            print ("download_questions_info error: " + str(e))

        time.sleep(SLEEP_TIME)
```

- question(.json)


```
[{"vm_name": "pwn1", "score": 0, "vm_ip": "111.206.245.29", "challengeID": 1, "binaryUrl":
"http://ai.defcon.ichunqiu.com/resources/file/bin1", "flag_path": "/home/flag1.txt", "question_port": "9001"},
{"vm_name": "pwn1", "score": 0, "vm_ip": "111.206.245.29", "challengeID": 2, "binaryUrl":
"http://ai.defcon.ichunqiu.com/resources/file/bin2", "flag_path": "/home/flag2.txt", "question_port": "9002"},
{"vm_name": "pwn1", "score": 0, "vm_ip": "111.206.245.29", "challengeID": 3, "binaryUrl":
"http://ai.defcon.ichunqiu.com/resources/file/bin3", "flag_path": "/home/flag3.txt", "question_port": "9003"},
{"vm_name": "pwn1", "score": 0, "vm_ip": "111.206.245.29", "challengeID": 4, "binaryUrl":
"http://ai.defcon.ichunqiu.com/resources/file/bin4", "flag_path": "/home/flag4.txt", "question_port": "9004"},
{"vm_name": "pwn1", "score": 0, "vm_ip": "111.206.245.29", "challengeID": 5, "binaryUrl":
"http://ai.defcon.ichunqiu.com/resources/file/bin5", "flag_path": "/home/flag5.txt", "question_port": "9005"},
{"vm_name": "pwn1", "score": 0, "vm_ip": "111.206.245.29", "challengeID": 6, "binaryUrl":
"http://ai.defcon.ichunqiu.com/resources/file/bin6", "flag_path": "/home/flag6.txt", "question_port": "9006"},
{"vm_name": "pwn1", "score": "64", "vm_ip": "111.206.245.29", "challengeID": 7, "binaryUrl":
"http://ai.defcon.ichunqiu.com/resources/file/bin7", "flag_path": "/home/flag7.txt", "question_port": "9007"},
{"vm_name": "pwn1", "score": "128", "vm_ip": "111.206.245.29", "challengeID": 8, "binaryUrl":
"http://ai.defcon.ichunqiu.com/resources/file/bin8", "flag_path": "/home/flag8.txt", "question_port": "9008"},
```

◻download_binarys()下载题目

```
下载题目，添加执行权限，并创建每个题目的工作目录，目录名格式为binary二进制文件内容的md5
工作目录存放字典，fuzz信息等
"""
def download_binarys(questions):
    try:
        for question in questions: #TODO: requests frequence limit?
            r = requests.get(question['binaryUrl'], auth=(config.user, config.password), headers=headers)
            binary = question['binaryUrl'].split('/')[-1]
            binary_path = os.path.join(work_path, binary)
            open(binary_path, 'wb').write(r.content)
            os.system("chmod +x {}".format(binary_path))
            job_dir = os.path.join(config.work_path, utils.uniq_binary_name(binary_path))
            if not os.path.isdir(job_dir):
                os.mkdir(job_dir)
            try:
                cmd = "rm -rf {}".format(os.path.join(job_dir, "sync"))
                os.system(cmd)
            except:
                pass

    except Exception as e:
        print "Download_binarys error:" + str(e)
        return False

    return True
```

◻ utils.uniq_binary_name() 计算md5

```
def uniq_binary_name(binary_path):
    m = hashlib.md5()
    content = open(binary_path).read()
    m.update(content)
    return os.path.basename(binary_path) + '-' + m.hexdigest()
    #os.path.basename(path) 返回path最后的文件名
```

- bin7-33a08423963d753c99a6554d7fa880cd

□小结

```
504 def main():
505     init()
506
507     download_questions_info()
508
509     check_same_binary()
510
511     submit_flag()
512
513     quick_exploit_before_fuzz()
514
515     fuzz_and_exploit()
```