Durian

```
usage: durian [-h] [-t <target>] [-o <output>] [-r <rule_id>]

    ____    __  __    ____     ____     ___     _    __
   / __ \  / / / /   / __ \   / __/    /   |   / | / /
  / / / / / / / /   / /_/ /   / /      / /| |  /  |/ /
 / /_/ / / /_/ /   / _, _/  _/ /      / ___ | / /|  /
/_____/  \____/   /_/ |_|  /___/     /_/  |_|/_/ |_/

                                          v1.0-Durian
GitHub: https://github.com/DurianCoder/Durian.git
Durian is a static code analysis system that automates the detecting vulnerabilities and security issue.

optional arguments:
  -h, --help            show this help message and exit

Scan:
  -t <target>, --target <target>
                        file, folder
  -o <output>, --output <output>
                        vulnerability output MAIL
  -r <rule_id>, --rule <rule_id>
                        specifies rules e.g: CVI-100001,cvi-190001

Usage:
  python durian.py -t ~/Source
  python durian.py -t ~/Source -r cvi-100001,cvi-100002
```

0x01、Introduction（介绍）

`Durian`工具是一个基于`libclang`的个性化代码检测工具。

> 1）、忘记日志打印等较为较为明显的问题，可以通过正则表达式直接匹配出。可以通过xml文件添加匹配规则，对于不同类型的问题，进行不同的逻辑处理，如是否打印了日志，虚函数是否写了文档注释等问题。
> 2）、基于libclang，利用clang对C family 语言进行词法分析和语法分析。将源码生成对应的AST,通过遍历AST来分析函数内可能存在的变量重复赋值问题。

0x02、Features（特点）

Multi-language Supported（支持多种开发语言）

> 支持C系列(基于libclang)、JAVA、PHP、等开发语言以及伪代码检测，并支持数十种类型文件。

Multi-Vulnerabilities Supported（支持多种漏洞类型）

> 暂时开放几条漏洞规则匹配，用户可以自定义，后续将继续更新。

CLI/API Mode（命令行模式和API模式）

> 暂时提供CLI模式，用户可以自己添加API、WEB扩展。

0x03、什么是"源代码安全审计（白盒扫描）"？

> 由于开发人员的技术水平和安全意识各不相同，导致可能开发出一些存在安全漏洞的代码。 攻击者可以通过渗透测试来找到这些漏洞，从而导致应用被攻击、服务器被入侵、数据被下载、业务受到影响等等问题。 "源代码安全审计"是指通过审计发现源代码中的安全隐患和漏洞，而Durian可将这个流程自动化。

0x04、Durian为什么能从源代码中扫描到漏洞？

> 对于一些特征较为明显的可以使用正则规则来直接进行匹配出，比如硬编码密码、错误的配置等。 对于OWASP Top 10的漏洞，Durian基于clang将对应源代码解析为AST(Abstract Syntax Tree, 抽象语法树)，进而基于AST对代码进行词法分析和语法分析，对漏洞进行扫描检测（clang支持对C系列的语言进行解析）。

0x05、安装：(39.41直接到步骤4)

1、安装pip

> 检查linux是否有安装python-pip包：`yum install python-pip` 没有python-pip包就执行命令：`yum -y install epel-release` 执行成功之后，再次执行：`yum install python-pip` 对安装好的pip进行升级：`pip install --upgrade pip`

2、安装clang编辑器:

> yum安装：`yum install clang`
> apt安装：`sudo apt-get install clang`

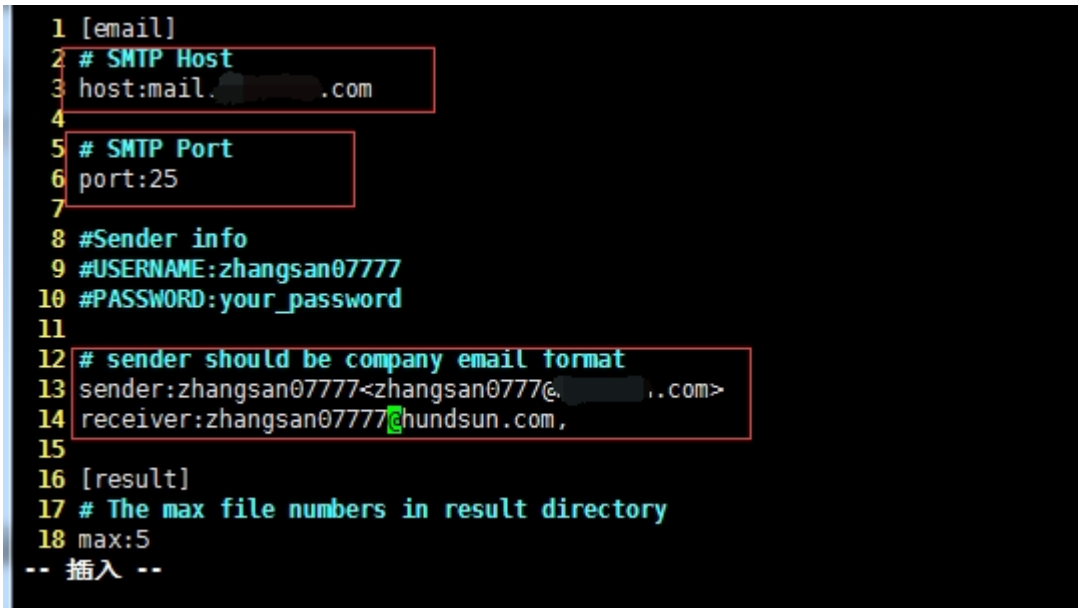
3、配置python bindings:

> 使用pip安装：`sudo pip install clang==版本号(如:3.4)`

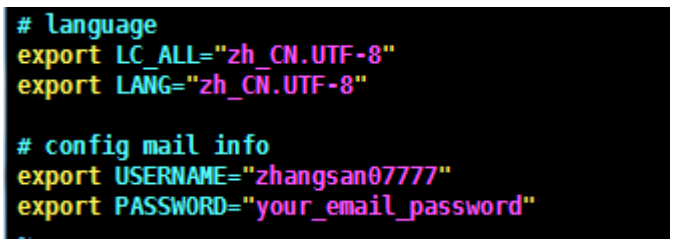
> **Tips:** 在代码检查工具`\libclang-packages`中提供了clang3.4的python绑定，如果服务器不能连接外网，可以将代码检查工具`\libclang-packages`中的两个文件添加到服务器的`/usr/lib/python2.7/site-packages`下（如服务器版本不是3.4，请自行安装相应的libclang）。
>
> **注意：clang的版本号和libclang的版本号必须一致。**

4、更改配置:

> 1）、在`Durian\config`文件中配置邮箱信息，发件人的账号密码可以在config中配置也可以在环境变量中配置(建议在环境变量中配置)。



> 2)、修改`.bash_profile`更改编码、配置发件人账号信息



5、运行

> 运行程序，扫描结果发送到配置的邮箱:
>
> **注意**：-t 扫描的目标文件夹路径必须为绝对路径
>
> 1）、扫描所有规则：`python durian.py -t target_file_absolute_path`
>
> 2）、扫描指定规则：`python durian.py -t target_file_absolute_path -r cvi-100001,cvi-1000002`
>
> 3）、添加收信人：
>
> ①、在config中receiver中添加多个邮箱以`,`分割。
>
> ②、-o参数中添加：`python durian.py -t target_file_absolute_path -o abc@qq.com,123@qq.com`

0x06、规则添加（二次开发）

> 1、静态规则：参考`docs\rule.md`和`docs\config.md`文件
>
> 2、非静态规则：参考`docs\develop.md`文件
>
> 3、项目结构：

```
## 项目结构文档树
Durian
.
├── config       配置文件，主要用于配置邮箱信息、result路径下结果文件最大个数
├── .gitignore   git版本控制忽略的文件
├── docs         项目文档
│   ├── config.md        配置规则文档
│   ├── develop.md       开发文档
│   ├── readme.txt       使用指南
│   ├── rule.txt         规则文件说明
│   ├── cron.txt         linux定时任务配置
│   └── tree.md          文档树
├── durian.py        程序入口
├── logs             日志文件
│   └── codechecktool.log
├── README.md        README
├── result           扫描结果，文件内文件个数可以在config中配置
│   ├── result.json
│   ├── s77345grk5mh
│   └── s77345tc2fjf
├── rules                扫描规则
│   ├── CVI-100001.xml
│   ├── CVI-100002.xml
│   ├── CVI-100003.xml
│   ├── CVI-100004.xml
│   ├── CVI-100005.xml
│   ├── CVI-110001.xml
│   ├── CVI-110002.xml
│   ├── CVI-120001.xml
│   ├── CVI-200001.xml
│   ├── languages.xml        语言
│   └── vulnerabilities.xml      缺陷类型
├── src              源码
│   ├── clangtool.py       libclang对AST的操作
│   ├── cli.py             cli扫描模式
│   ├── config.py          配置
│   ├── engine.py          后续api模式可以将方法抽象到engine
│   ├── __init__.py        解析参数，调用cli
│   ├── log.py             日志打印
│   ├── __pycache__
│   │   ├── cli.cpython-36.pyc
│   │   ├── __init__.cpython-36.pyc
│   │   ├── log.cpython-36.pyc
│   │   ├── send_mail.cpython-36.pyc
│   │   └── __version__.cpython-36.pyc

│   ├── rule.py              规则扫描操作
```

```
│   ├── send_mail.py        发送邮件
│   ├── util.py             工具类
│   ├── __version__.py      版本信息
└── test            测试
    ├── test_clang      libclang AST测试
    │   ├── AnalysisAST.py
    │   ├── asttree.py
    │   ├── get_func_detail.py
    │   ├── get_type.py
    │   ├── get_vars.py
    │   └── person.cpp
    ├── test_eamil.py
    ├── test_mulpro.py
    ├── test_re.py
    └── test_util.py
```