# 数据安全行为分析底层抓包分类服务使用说明

工作文档 使用说明

### 1. 文件结构

- 主程序 **Flow-class**
- 配置文件 **flow_cfg.json**
- 依赖网络抓包库 libpcap (apt-get install libpcap-dev)

---

### 2.基本配置文件

配置文件为json格式，一个示例如下

```
{
        "device":"eth0",
        "polic_ip":"0.0.0.0",
        "polic_port":"8818",
        "report_ip":"172.16.2.198",
        "report_port":"8800"
   }
```

> device : 监听的抓包的网卡
> polic_ip,polic_port : 本机上监听的TCP地址，用于接收抓包策略
> report_ip,report_port : 上传抓包数据的tcp地址

---

### 3.程序的启动

将主程序Flow-class 拷贝到 /usr/bin 目录下，直接运行
或者拷贝到任意路径下使用绝对路径运行
通过 -c 参数指定运行配置文件

```
>$: Flow-class -c /path/to/your/config/flow-cfg.json
```

> 程序启动后通过 ps 命令查看进程是否存在
> 通过netstat 命令查看策略监听端口是否打开

#### 4 数据分类规则

| 服务器IP | 服务器端口 |
| --- | --- |
| 222.46.20.174 | 11980 |
| 222.46.20.174 | 12080 |

> 这两条信息表示抓取发往服务器 222.46.20.174 11980 和12080 端口的数据（目前只支持http）

#### 5.策略下发

策略规则的发送：
通过tcp发送一个json格式的字符串到策略监听地址更新规则
规则文件示例如下：

```
{
        "polic":
        [
            {"host":"222.46.20.174","port":"12080"},
            {"host":"222.46.20.174","port":"11980"}
        ]
 }
```

> polic 数组中每一个元素表示一个规则

#### 6.数据包上传协议

- 网络数据包过滤分类后是得到一个**二进制**的以太网数据帧（便于使用第三方工具进行二次分析）
- 使用tcp 封装数据发送到 数据接收服务端
- 封装数据包的格式

| version | total_len | src_ip | src_port | dst_ip | dst_port | time_stamp | data_len | data |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1字节 | 2字节 | 4字节 | 2 字节 | 4字节 | 2 字节 | 4字节 | 2字节 | data_len 字节数据 |

> version : 版本号，（0x01）
> total_len 数据总长度，减去1字节版本号和他本身，就是剩余的字段的总长
> src_ip: 网络格式的源地址信息，4字节
> src_port : 2 字节源端口号
> dst_ip: 网络格式的目的地址信息，4字节
> src_port : 2 字节目的端口号
> timestamp: 时间戳，4字节，unix时间格式（1970年1月1日之后的秒数）
> data_len : 网络数据长度
> data : 数据字段

- data 字段是一个二进制的以太网帧，包括 以太网帧头，IP头，TCP头和数据段

---

**7.完整启动流程**

1. 启动 Flow-class 主程序
2. 启动数据包接收程序
3. 发送策略(json格式) 到Flow-class的策略监听端口
4. 模拟网络行为（比如访问 策略配置中的服务）
5. 数据接收端接收到数据保存

**8. 测试程序**
为了方便测试，我们编写了一个数据包接收程序(应该由java端完成)

- srv [监听端口]

当接受到数据会在当前目录下生成一个 data.dat 的文件保存网络数据包
测试程序解析相关信息保存为文本，网络数据用Base64编码后保存