AwesomeProxy

工作量

Total : 56 files, 5239 codes, 974 comments, 792 blanks, all 7005 lines

language	files	code	comment	blank	total
Go	48	5,082	974	757	6,813
Markdown	1	95	0	31	126
XML	4	29	0	0	29
JSON	1	23	0	0	23
Go Module File	1	6	0	3	9
Go Checksum File	1	4	0	1	5

预计功能

正向代理

- [x] HTTP、HTTPS、WS、WSS、SOCKET5多种协议数据包抓包代理
- [x] 支持请求数据和返回数据的拦截和自定义修改
- [x] 自动识别入站协议，按照消息头识别不同的协议进行处理

- [x] 正则匹配黑名单HOST过滤
- [ ] 用AC自动机算法去做一个模式串匹配，对访问的网站的返回数据进行数据包过滤+网站分析，排除隐患网站和非法网站以及钓鱼网站
- [x] 自动生成并安装RSA公私钥证书，代理加密数据包
- [x] 通过WindowsAPI，自动打开Windows系统代理，并设响应参数

反向代理

- [x] 多种算法实现负载均衡
  - [x] 基于随机算法的负载均衡
  - [x] 基于RoundRobin算法的负载均衡
  - [x] 基于带权重RoundRobin算法的负载均衡
  - [x] 基于一致性hash算法的负载均衡
  - [x] 基于洗牌算法的负载均衡
  - [x] 基于优化洗牌算法的负载均衡
- [x] 通过协议头和访问频率算法分析过滤爬虫，减轻服务器负担
- [ ] 通过流量分析防止DDOS流量攻击
- [ ] 违规IP封禁
- [ ] 分布式数据缓存，减轻服务器的计算压力，加快响应，提升用户交互

其他功能

- [x] 一个独立的轻量级Golang日志库
  - [x] 分布不同级别的日志
  - [x] 全局日志接口和实例化日志接口两种模式，方便切换和日志打印
  - [x] 根据不同日志级别在窗口打印不同颜色的日志
  - [x] 多种日志格式可选，方便定位到出错文件的位置
  - [x] 提供两种日志分割方式，按照日期分割和文件大小分割
  - [x] 线程安全，让日志库使用更加安全
- [ ] 生成流量分析报告
  - [ ] 反向代理分析恶意攻击和爬虫
  - [ ] 正向代理分析对用户上网行为分析，违禁网站访问等

config.json

先给一个简单的配置例子：

```
{
  "ProxyMethod":false,
  "ReProxy":{
    "port":"9090",
    "BalanceMethod":1,
    "backend":[{"host":"127.0.0.1:10001","Weight":1},
{"host":"127.0.0.1:10002","Weight":1},
{"host":"127.0.0.1:10003","Weight":1}],
```

```
    "Cache":{"start": "true","MaxSize": "100000"}
  },
  "CoProxy":{
    "port":"9090",
    "MultiListenNum":5,
    "nagle":true,
    "filt":[".*\\.csdn\\..*"]
  },
  "Logg": {
    "FileNameReProxy": "ReProxyLog.txt",
    "FileNameCoProxy": "CoProxyLog.txt",
    "SplitFormat": "DateSplit",
    "DateSplit": "MODE_DAY",
    "SizeSplit": {"LogSize": 300,"Unit": "MB","FileNum": 10}
  }
}
```

- `ProxyMethod` 表示的代理方式， `true` 为反向代理， `false` 为正向代理

ReProxy

> 这一部分是配置反向代理网关的

- `prot` 表示的是反向代理的代理端口
- `BalanceMethod` 表示的是负载均衡的方法，本项目支持六种负载均衡算法，分别是（配置文件从0开始算第一个算法）：

  1. 基于随机算法的负载均衡

2. 基于RoundRobin算法的负载均衡
3. 基于带权重RoundRobin算法的负载均衡
4. 基于一致性hash算法的负载均衡
5. 基于洗牌算法的负载均衡
6. 基于优化洗牌算法的负载均衡

- 项目默认基于RoundRobin算法的负载均衡，如果选用 `基于带权重RoundRobin算法的负载均衡` ，请配置 `config.json` 中的 `RePro xy.backend.Weghit` 部分，另外如果只有一台服务器，默认 `Rou ndRobin` 算法，可以提升效率
- `backend` 表示的是服务器集群，在这里可以配置多个服务器进行负载均衡
  - `host` 表示的是被代理服务器的ip+端口，或者是url
  - `weight` 表示的是被代理服务器的权重优先级，范围是[1,10]，值越小权重越大
- `Cache` 表示的是缓存配置（目前还未实现，暂时这样设计的）
  - `start` 表示是否开启， `true` 表示开启， `false` 表示关闭
  - `MaxSize` 表示总缓存的数据大小的最大值，单位目前是 `byte`

CoProxy

> 这一部分是配置正向代理网关的

- `port` 表示的是正向代理的代理端口
- `MultiListenNum` 表示的多线程数，可以设置为CPU的核心数
- `nagle` 表示是否开启nagle算法优化网络传输，在一些对时延要求较高的交互式操作环境中可以设置 `false` ，默认开启

- `filt` 表示的是正则过滤的网站，可以将一些违禁网站过滤，注意这里的正则匹配只支持Golang的regexp包规则，可以在这里查看语法规则：http://c.biancheng.net/view/5124.html

Logg

> 这一部分是配置日志文件的分割格式的，和日志文件名，
> 反向代理日志默认放在 `Log/Data/ReProxyData`
> 正向代理日志默认放在 `Log/Data/CoProxyData`

- `FileNameReProxy` 表示的是反向代理服务器的日志文件名
- `FileNameCoProxy` 表示的是正向代理服务器的日志文件名
- `SplitFormat` 表示的是日志文件分割格式，例如可以按照日期和文件大小进行分割成不同的文件
  - `DateSplit` 表示按照时间日期进行分割，`MODE_HOUR`、`MODE_DAY`、`MODE_MONTH` 分别表示文件按照小时、天、月进行分割
  - `SizeSplit` 表示按照大小进行分割
    - `LogSize` 表示的单个日志文件的大小
    - `Unit` 表示单个文件大小的单位
    - `FileNum` 表示日志文件的最大数量（只会保存最新的`FileNum`个）