|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 子模块名称 | 核心功能 | 相关技术 & 说明 | |
| 预  处  理  模  块 | 输入处理 | 分析处理输入的字符串,转换编码 | Encode decode  Idna字符集  [Python idna库](https://pypi.org/project/idna/)  [国际化域名-维基百科](https://zh.wikipedia.org/wiki/%E5%9B%BD%E9%99%85%E5%8C%96%E5%9F%9F%E5%90%8D) | |
| 类型判别 | 判断输入的字符串是IP还是非IP | 正则匹配 | |
| IP反查域名 | 查询IP对应的域名  (最近经常出现此类需求或者场景) | Host nslookup | |
| 粗过滤 | 对得到的FQDN进行粗过滤(去除协议字符串,/后面的内容,空格,及其他无关数据) |  | |
| 域名提取  与分析 | 获取标准域名 | [Tldextract](https://github.com/john-kurkowski/tldextract) (此[python库](https://pypi.org/project/tldextract/2.2.0/)也是基于Public Suffix List数据进行操作的)  [Public Suffix List](https://github.com/publicsuffix/list)  进行最长匹配 | |
| 域  名  W  H  O  I  S  获  取  模  块 | WHOIS服务器选择 | 选择TLD所对应的WHOIS服务器 | 这里同时要获取WHOIS服务的特定请求格式与编码类型及转发WHOIS服务器数据(可选) | |
| TLD锁 | 判别该TLD是否能与此时发出请求 | 单机锁实现:thread lock  共享锁实现:数据库,redis,同步机制 | |
| SOCKS代理 | 是否使用socks代理来代理请求 | [Pysocks](https://github.com/Anorov/PySocks)  Python默认的socket不支持socks代理, 需要借助其他方式实现代理功能  这里最好不要与数据库有关联,通过配置文件或者函数参数的方式指定此次过程的代理.,方便程序的部署 | |
| WHOIS服务器通信 | 核心模块 基于[RFC 3912](https://github.com/h-j-13/WHOIS-theory.zh-cn/blob/master/RFC/NOW/RFC%203912.zh-cn.md) 向WHOIS服务器发出请求,获取数据. | [Pysocks](https://github.com/Anorov/PySocks) (注意超时的设置和异常错误的处理方式)  Socket  [nscd](http://www.361way.com/linux-nscd-dns-cache/4265.html)-DNS缓存机制(利用此来替换去除WHOIS服务器ip模块,精简系统) | |
| 通信管理 | 控制整个通信流程  根据链接是否成功,是否存在二级WHOIS,数据是否完完成,是否可以请求TLD,etc  来控制整个请求的过程 | 通信过程flag设计与文档. | |
| WHOIS二级服务器 | 根据TLD及WHOIS一级数据判断是否存在二级WHOIS服务器 | 已知规则+正则匹配 | |
| 数据分析 | WHOIS原始数据解析 | 提取函数 | 正则匹配  继承方式改写基础提取函数 | |
| 规范化与特征分析 | 时间规范化 | 包括规范化为YY-MM-DD HH:mm:ss  及时区解析及处理 | Pytz  dateutil | |
| 注册电话规范化 | 规范格式  +86.1XXXXXXXXXX | 已有经验与规则 | |
| 地理位置分析 | 地理位置字段提取  Tel 邮编 定位 | 第三方地理位置数据  字典树(前缀匹配) | |
| 异常数据修复 | 例如: province : Shangdong修复为 Shandong  其次就是有部分尚未被提取函数获取的字段,可以通过文本分类进行判别 | Levenshtein距离/文本分类 | |
| 邮箱验证 | 邮箱真实性验证 | Smtp协议  mx记录 | |
| 输入模块 | 函数调用方式 | 探测点方式  分布式方式 | Api调用方式 | |
| 输出模块 | 域名相关信息 | WHOIS数据 | 解析数据 | 拆分信息 |