apfree(wifidog)接口描述

ver 2.0　2014.1.6

ver 2.1　2014.1.14 @修改速率单位，添加主机策略“顺序”描述

ver 2.2　2014.2.20 @修正一个代码拼写错误（macwhite）

ver 2.3　2014.5.06 @增加认证服务器 login 接口新的提交 mac 参数

ver 2.5　2014.6.23 @补齐 wifidog 标准接口 by walkingsky

本文主要写给采用 ApFree 的 wifidog，自己做认证服务器端扩展的开发用户。

一. 配置文件接口

和标准版 wifidog 配置文件（wifidog.cfg）相比，增加了 ReqProcessMode（运行模式）参数

参数名：ReqProcessMode

值：　0　原始模式

　　1　单线程模式

　　2　优化后的模式

举例：ReqProcessMode 2

二. 认证服务器接口

标准版 wifidog 和认证服务器有 ping、login、auth、portal 4 个接口！ApFree 的 wifidog 也是基于这四个接口，只不过向服务器提交的参数或者认证服务器返回的数据有变化！

1. wifidog 提交参数的变化

a) ping 接口增加的参数：dev_id、cpu_usage、nf_conntrack_num

i. dev_id 设备 id，如：67869201000-780000541641387-35786613505096001

ii. cpu_usage cpu 利用率，单位%，值 0-100

iii. nf_conntrack_num 网络连接数，值为整数

b) login 接口增加的参数：dev_id、mac

i. mac 客户端 mac 地址

c) auth 接口增加的参数：dev_id、uprate、downrate

i. dev_id

ii. uprate 该客户端该时刻即时上行速率，单位 bps

iii. downrate 改客户端该时刻即时下行速率，单位 bps

d) portal 接口增加的参数：dev_id

2. 认证服务器返回数据的变化

主要是 ping 接口的返回改了很多！wifidog 根据这些返回做对应的流控策略和 wifidog 域名开放策略

先看一个典型的返回数据格式：

Pong

result={"rule":{"host":[{"id":"35","ip":"192.168.1.145","netmask":"255.255.192.0","up":"102400","down":"52428800","session":"0"}],"host_md5":"e74e2fb47bce1066ec02f9c125089d07","ipwhite":[{"id":"74","ip":"192.168.0.198","netmask":"255.255.255.255"},{"id":"77","ip":"192.168.0.169","netmask":"255.255.255.255"}],"ipwhite_md5":"5646ae26578019d406ef9d868bea2fe6","macblack":[],"macblack_md5":"d751713988987e9331980363e24189ce","mackwhite":[],"mackwhite_md5":"d751713988987e9331980363e24189ce","domain":[{"domain":"szshort.weixin.qq.com,www.apfree.net"}],"domain_md5":"119e64c7851aa20a52a06c12a9154c23"}}

开源 wifidog 只要求认证服务器返回 Pong
ApFree 的 wifidog 要求返回 Pong+空格+result=json 字符串

Json 格式如下

{
"rule":{
"host":[
{
"id":"35",
"ip":"192.168.1.145",
"netmask":"255.255.192.0",
"up":"102400",
"down":"52428800",
"session":"0"
}
],
"host_md5":"e74e2fb47bce1066ec02f9c125089d07",
"ipwhite":[
{
"id":"74",
"ip":"192.168.0.198",
"netmask":"255.255.255.255"
},
{
"id":"77",
"ip":"192.168.0.169",
"netmask":"255.255.255.255"
}
],
"ipwhite_md5":"5646ae26578019d406ef9d868bea2fe6",
"macblack":[
],
"macblack_md5":"d751713988987e9331980363e24189ce",
"macwhite":[
],

```
        "macwhite_md5":"d751713988987e9331980363e24189ce",
        "domain":[
            {
                "domain":"szshort.weixin.qq.com,www.apfree.net"
            }
        ],
        "domain_md5":"119e64c7851aa20a52a06c12a9154c23"
    }
}
```


host：为流控里主机（网段）限制规则，数组（可以有多条规则），规则内容分别为 ip、netmask、up、down、session，主要限制主机或网段内所有主机的上行（上传）、下行（下载）带宽，会话数限制不建议采用。本规则和 wifidog 的认证同时生效，即如果没有通过 wifidog 的认证，无论针对该主机的规则流量多大，也是不会有流量的（开放域名的除外）；执行顺序问题：主机策略都是有先后顺序的，采用串行模式（类似于程序里的 switch case break 模式），先匹配上的先执行，后面的都不再匹配和执行。所有下发多个主机策略的时候一定要注意先后顺序，在前面的先生效，后面的后生效。

注意事项：

- ip 和 netmask 组合使用，来决定是单主机规则还是整个网段的规则；
- 会话数限制 session 不建议使用；
- 不做限制数值要设置成 0；
- up、down 的单位为 Bps；
- 样例中的 id 无用处

ipwhite：为流控里的 ip 白名单规则，数组（可以有多条规则），配置的主机或网段不受主机规则影响。和 wifidog 共同起作用

注意事项：

- 本 ip 白名单是限速的白名单，不是 wifidog 里的白名单；
- ip 和 netmask 组合使用，来决定是单主机规则还是整个网段的规则；

macblack：为流控里的 mac 黑名单规则，加入规则的 mac 是不能上网的，和 wifdog 同时生效。

注意事项：

- mac 黑名单不再采用 json 数组格式，所有的 mac 列表都写在一起，中间用“,”隔开，如 mac1,mac2,mac3
- 加入 mac 黑名单后网络功能受很大影响，甚至不能分配 ip，不能 ping 通路由器

macwhite：为流控中 mac 白名单规则，mac 不经 wifidog 认证就可上网

domain：为 wifidog 域名白名单，专为 wifidog 设置，是专门给不经过认证就能访问的域名使用的

注意事项：

- 不支持泛域名，不支持通配符
- 不再采用 json 数组格式，所有的域名写在一起，如 domain1,domain2

特别说明：各个 md5（host_md5,ipwhite_md5,macblack_md5,macwhite_md5,domain_md5）是给对应的规则的 md5 字符串校验，算法是将 json 字符串中的“[”开始“]”结束做 md5 加密的结果。比如例子中的 ip 白名单

"ipwhite":[{"id":"74","ip":"192.168.0.198","netmask":"255.255.255.255"},{"id":"77","ip":"192.168.0.169","netmask":"255.255.255.255"}]

要将字符串：[{"id":"74","ip":"192.168.0.198","netmask":"255.255.255.255"},{"id":"77","ip":"192.168.0.169","netmask":"255.255.255.255"}]**进行 md5 加密，然后填充到** ipwhite_md5 里

三．Wifidog 标准流程描述（基于 2009 版本）

a) 流程描述

参见 http://blog.csdn.net/lxgwm2008/article/details/12782037

b) 认证流程描述

i. Wifidog 运行之后建立一系列的防火墙规则，主要规则起到如下作用：1.阻断所有内网到外网的访问。2.开通内网到外网的 dns 访问。3.开通内网到认证服务器以及域名白名单的访问。4.对内网到外网 80 端口的访问转向到 wifidog 自己的 http 服务（2060 端口）

ii. 手机、pc 连接上来后，app 或者系统（安卓、ios 会自己连接到各自的服务器上来验证网络的连通性）会发起对外网的访问请求，dns 请求会被放过，然后对应的 80 端口的访问会被指向 2060 的 http 服务，其他的请求都会被拦截

iii. Wifidog 的 http 接到 web 请求后，基本上都会被指向 404 页面，404 页面会给客户端一个重定向返回（302），要求客户端重定向访问认证服务器的 login 页面，附加参数 `gw_id、gw_address、gw_port、url`

iv. 手机、pc 客户端加载、显示认证服务器的 login 页面，用户根据页面内容做相关的认证操作（qq 登录、微博登录、用户名密码登录、手机短信登录等多种登录方式），原则只有一个认证不成功就仍然让用户停留在认证服务器继续认证操作，认证成功给客户端一个 302 重定向返回，根据 login 接口提交上来的参数 `gw_address、gw_port` 跳转套 wifidog web 服务的/wifidog/auth 页面上，附带 `token` 和 `url` 参数

v. Wifidog 的 web 服务收到手机、pc 客户端的/wifidog/auth 请求后，会主动对认证服务器的 auth 接口发起一个验证请求，附带参数 `ip、mac、token、stage=login`

vi. 认证服务器的 auth 接口收到 wifidog 的请求，要根据内部逻辑返回是否允许通

过的应答 `Auth: 0` 拒绝 `Auth: 1` 允许

vii. Wifidog 接收到验证结果后，如果拒绝访问，就会返回 302 给客户端，重定向到认证服务器的 gw_message 接口，附带 `message=denied` 参数，客户端的上网访问仍然会回到第二步骤；如果允许访问，则改动防火墙规则，开通改客户端的上网（至此客户端已经能够正常上网），然后返回 302 重点向给客户端，重定向到认证服务器的 portal 接口，附带参数 *gw_id*

viii. 认证服务器的的 portal 接口根据业务流成显示广告业或者做其他的跳转

ix. 整个认证流程完成

c) ping 心跳流程描述

i. ping 接口 wifidog 检测认证服务器访问是否正常、并向认证服务器提交 wifidog 的运行状态

ii. 定时 ping 认证服务器

iii. 提交的参数 *gw_id*、*sys_uptime*、*sys_memfree*、*wifidog_uptime*

d) auth 心跳流程描述

i. 和 ping 一样的频率定期请求认证服务器，并且有多少已认证客户端就发多少请求

ii. 用来向认证服务器提交客户端的状态以及执行认证服务的验证结果

iii. 提交的参数有：*ip*、*mac*、*token*、*incoming*、*outgoing* 、*stage=counters*

iv. 如果服务器返回拒绝，则 wifidog 改动防火墙规则，关闭该客户端的上网