GoAccess

The real time web log analyzer

real-time web log analyzer

一 关于 GoAccess ____ 3
1 GoAccess 是什么？ ____ 3
2 特点 ____ 3
3 为什么用 GoAccess ____ 3
4 缺点 ____ 4
geoip 粒度太粗 ____ 4
日期粒度太粗 ____ 4
二 GoAccess 的安装 ____ 4
1 最新安装包的下载 ____ 4
2 需要安装的依赖 ____ 4
2.1 官方给出的相关依赖提示 ____ 5
2 编译安装 ____ 5
2.1 编译的参数 ____ 5
3 yum 安装（我们选择这种方式） ____ 6
三 GoAccess 参数说明 ____ 6
四 GoAccess 的使用 ____ 8
五 GoAccess 对于 access日志的自定义 ____ 9
1 如何自定义日志格式 ____ 9
2 官方自定义日志格式的参数 ____ 9
2.1 自定义参数及其对应 nginx 的 accesslog中的 fromat ____ 10
3 根据我们的 nginx 的 format 自定义日志格式 ____ 10
六 真正分析 access日志 ____ 11
键盘操作： ____ 12
七 使用 GoAccess 生成报表及一些管道的应用 ____ 12
八 两个问题的调研 ____ 13
1 关于设备识别 unknow 的问题 ____ 13
2 没有找到对错误日志的分析方法 ____ 16

一 关于 **GoAccess**

1 GoAccess 是什么？

GoAccess 是一款开源工具，它可以实时地去分析 web 的 log 并且提供了在 *nix 系统上通过交互式窗口来运行查看， 它通过可视化报表快速提供给系统管理员有价值的关于 HTTP 的统计信息。

2 特点

可以将指定的 web log 文件进行解析，通过终端把统计的数据展现出来

生成统计数据，带宽统计等
请求的时间（有利于查看慢的站点）
最高访问
请求的静态文件统计，例如图片， js , css 等
各个状态码的统计
Host，反向 DNS， ip 所在地
操作系统
浏览器 /蜘蛛
引用的网站
引用的 URLs
关键词组
地理位置 （大陆、国家、城市）
可输出 JSON 或 CSV
各种的颜色主题
支持大容量数据且支持大容量数据的持续分析
支持 IPv6
可生成 HTML 报告

3 为什么用 GoAccess

GoAccess 背后的主要思想是能够快速分析和查看 web 服务器统计数据， 虽然可以生成一个 HTML 、JSON、 CSV 报告 ,默认情况下它输出终端，并且你还可以用到他更多的监控命令工具。

4 缺点

geoip 粒度太粗

它是使用机器自带的 GeoIP，这个自带的 77k 左右的 IP 库只能判断出国籍，不能判断到城市。呃，这个就意味着你的报表中有 99% 的 IP 统计是来自： China 。你可以升级你的 GeoIp，但是，付费。网上有免费的 GeoLiteCity.dat 库，但是 goaccess 自身就不支持国籍 - 城市的分类，也不支持使用外部的 IP 库。

我能想到还有的办法就是用 goaccess 生成 json，然后自己写 python 也好， php 也好程序来加载城市的 Ip 库来解析地理位置。然后再生成报表。不过，这样，好像就不美了。

好消息是这个功能在 ISSUE 中有人提了， TODO LIST 。

日期粒度太粗

日期只能粒度到天， 如果需要统计一天每个小时的访问数据， 就没法了。也不是没办法，你可以先 grep 小时的数据，然后再用 goaccess 解析。不过，这样，好像很挫。

好消息是这个功能在 ISSUE 中有人提了， TODO LIST 。

二 **GoAccess**的安装

1 最新安装包的下载

通过 http://goaccess.io/download 可以下载到最新的 tar 包

2 需要安装的依赖

```
yum install glib2 glib2-devel GeoIP-devel ncurses-devel
```

根据官方给出的提示，这里我们先安装好相应的依赖，以便日后的使用

2.1 官方给出的相关依赖提示

Distro	NCurses	GLib >= 2.0.0	GeoIP (optional)	Tokyo Cabinet (optional)
Ubuntu/Debian	libncursesw5-dev	libglib2.0-dev	libgeoip-dev	libtokyocabinet-dev
Fedora/RHEL/CentOS	ncurses-devel	glib2-devel	geoip-devel	tokyocabinet-devel
Arch Linux	ncurses	glib2	geoip	compile from source
Gentoo	sys-libs/ncurses	dev-libs/glib:2	dev-libs/geoip	dev-db/tokyocabinet

2 编译安装

```
1 wget http://tar.goaccess.io/goaccess-0.9.2.tar.gz

2 tar -xzvf goaccess-0.9.2.tar.gz

3 cd goaccess-0.9.2/

4 ./configure --enable-geoip --enable-utf8

5 make

6 make install
```

2.1 编译的参数

```
--enable-debug
```

用于调试并且关闭编译器优化

```
--enable-utf8
```

用于支持更多的字符集，满足 Ncursesw 的需求

```
--enable-geoip
```

用于地理位置的支持 ．满足 MaxMind's GeoIP 需求

```
--enable-tcb=<memhash|btree>
```

用于对 memhash 和 btree 的支持

```
--disable-zlib
```

在 btee 数据库上禁用 zlib 压缩

```
--disable-bzip
```

在 btree 数据库上禁用 bzip 压缩

3 yum 安装（我们选择这种方式）

先安装 epel

```
Wget http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
wget http://rpms.famillecollet.com/enterprise/remi-release-6.rpm
sudo rpm -Uvh remi-release-6*.rpm epel-release-6*.rpm
```

安装 GoAccess

```
yum install goaccess-y
```

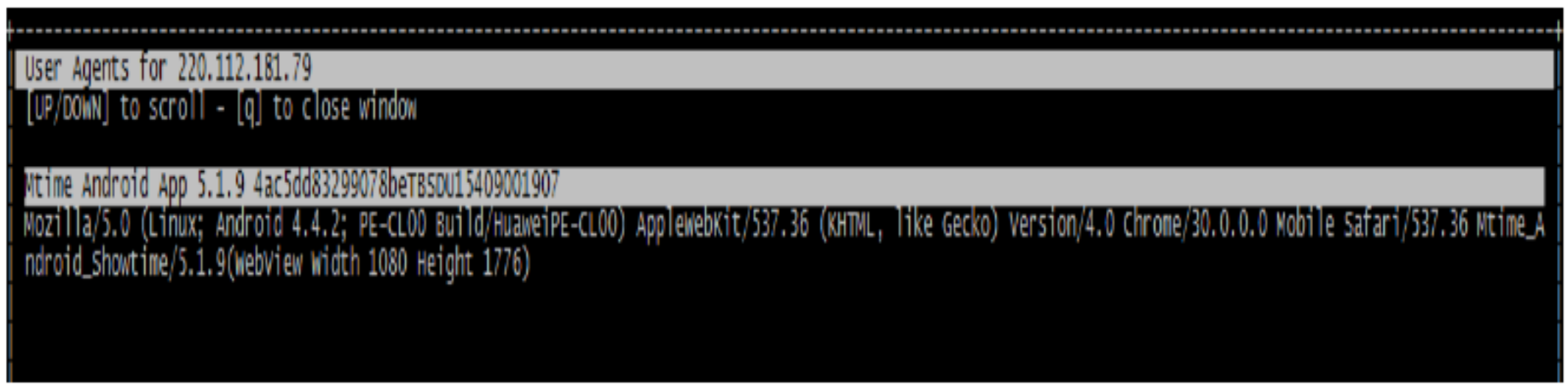
安装十分方便，到此就结束了

三 GoAccess参数说明

```
goaccess-f log [-c][-r][-m][-h][-q][-d][-g][-a][-o csv|json][-e IP_ADDRESS][...]
```

-a --agent-list

可以在 host 栏点击 ip 显示 user agent 如下图



-c --config-dialog

是否显示 log 和 format 配置对话窗口，也可以用于日后标准格式的调试，和重新设定格式

-d --with-output-resolver

生成 json 报表需要用到的参数

-e --exclude-ip <IP|IP-range>

排除某个 IP 或者一个范围例如 ., 192.168.0.1-192.168.0.10

-f --log-file <logfile>

用于解析日志文件，必加的参数

-g --std-geoip
支持标准的 geoip

-h --help
帮助

-H --http-protocol
包含 HTTP 请求的协议，格式中已设定无需加此参数

-m --with-mouse
支持鼠标的点击，点击相当于键盘的回车操作

-M --http-method
包含请求的方法和当前的请求，以后的格式中已设定，无需此参数

-o --output-format <json|csv>
生成 jsion 和 csv 报表需要加此参数导出

-p --config-file <configfile>
可以指定已经配置好格式的配置文件，优先于默认的格式

-q --no-query-string
忽略请求中疑问号后的 query 例如

www.google.com/page.htm?query => www.google.com/page.htm
官方提示移除这个可以减少内存的消耗，根据需求而定

-r --no-term-resolver
禁止 ip 在终端上的显示

-s --storage
显示当前的存储算法

-h --help
命令行的帮助

-V --version
显示版本号

--444-as-404
视 444code 为 404code

--ignore-panel=PANEL
忽略以下列出的数据

VISITORS

REQUESTS

REQUESTS_STATIC

NOT_FOUND

HOSTS

OS

BROWSERS

VISIT_TIMES

REFERRERS

REFERRING_SITES

KEYPHRASES

GEO_LOCATION

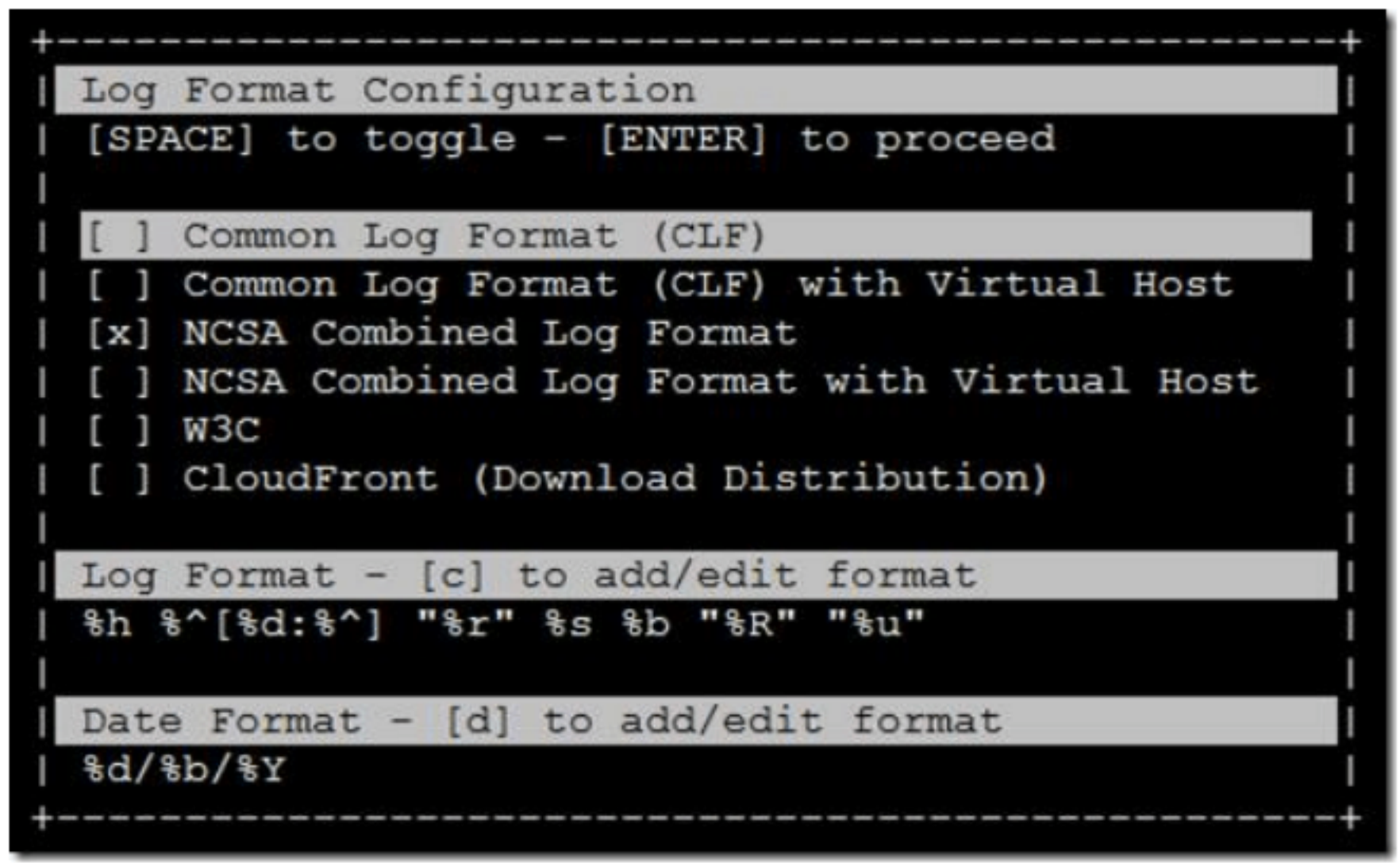
STATUS_CODES

--real-os

展示真实的操作系统。在 Operation System 模块中，是否展示更详细的操作系统信息。

四 GoAccess的使用

首次我们通过执行 goaccess -f access.log 解析得到以下窗口



Common Log Format （CLF） 此适用于 apache 标准日志

通用日志格式，例子：

127.0 . 0.1 - frank [10 /Oct/ 2000 : 13 : 55 : 36 - 0700] "GET /apache_pb.gif HTTP/1.0" 200 2326

主机 用户身份 作者 [日期] " 请求方法 请求路径 请求协议 " 状态码 字节数

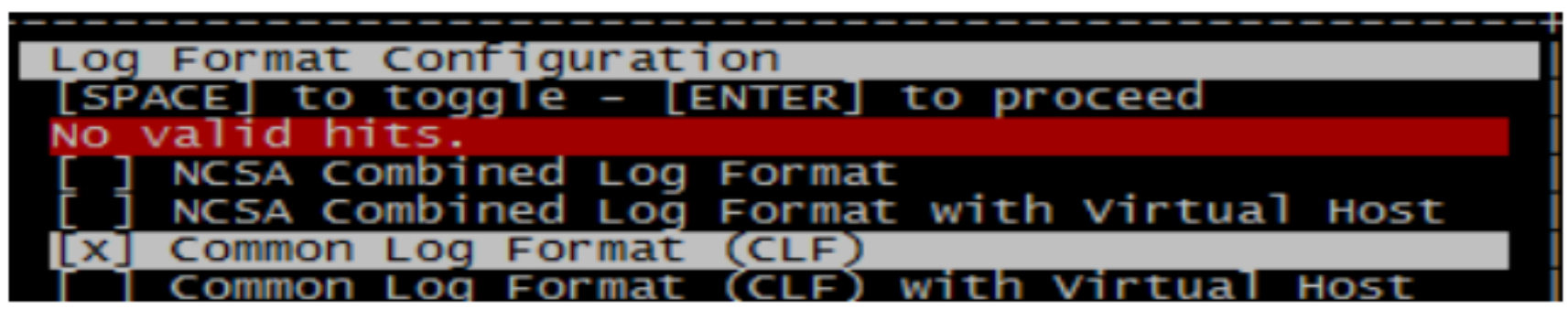
NCSA Commbined Log Format 适用于 nginx 标准日志

这个是 Common Log Format 的扩展，例子：

125.125 . 125.125 - dsmith [10 /Oct/ 1999 : 21 : 15 : 05 +0500] "GET /index.html HTTP/1.0" 200 1043 "http://www.ibm.com/" "Mozilla/4.05 [en] (WinNT; I)" "USERID=CustomerA;IMPID=01234"

主机 用户身份 作者 [日期] " 请求方法 请求路径 请求协议 " 状态码 字节数 referrer 客户端代理 cookie

需要注意的是



当空格选中模式的时候按回车会出现此报错，　　是由于你配置文档没有打开日志格式和日期格式的注释，或者说是有可能日志格式与系统给出的相应格式不符，系统无法读取

显然这里我们不能通过官方给出的标准格式来解析

五　GoAccess对于 access日志的自定义

1 如何自定义日志格式

1.1 可以通过 -c 的参数来调出上面系统自带的定义日志格式窗口，　　因为在你设置完日志格式后执行 goaccess -f access.log 就不会再调出此窗口，只有首次执行才会调用，所以要通过　　-c 参数

1.2 我们可以通过修改　/etc/, /usr/etc/ or /usr/local/etc/ 下的配置文件来进行自定义格式

Vim /etc/goaccess

```
# Apache log date format. The following date format works with any
# of the Apache's log formats below.
#
date-format %d/%b/%Y
#
# W3C (IIS) & AWS | Amazon CloudFront (Download Distribution)
#
#date-format %Y-%m-%d

# The log_format variable followed by a space or \t for
# tab-delimited, specifies the log format string.
#
# NCSA Combined Log Format
#
#log-format %h %^[%d:%^] "%r" %s %b "%R" "%u"
#
# NCSA Combined Log Format with Virtual Host
#
#log-format %^:%^ %h %^[%d:%^] "%r" %s %b "%R" "%u"
log-format %^ %^ %h %^[%d:%^] "%r" %s %b "%R" "%u" "%^" "%^:%^" "%^" "%T" "%^"
#log-format %^ %^ %h %^[%d:%^] %^
# Common Log Format (CLF)
```

按照上图，取消　data-format 和 log-format 的注释，定义自己的格式，或者自己加两行也可以

2 官方自定义日志格式的参数

time_format　在配置文件可以加入字段，后面加上时间的自定义参数，对时间格式做全局设

置

date_format 在配置文件可以加入此字段，后面加上日期的自定义参数，对日期格式做全局设置

log_format 在配置文件可以加入此字段，后面加上自定义的日志参数，对日志格式进行全局的设置

2.1 自定义参数及其对应 nginx 的 accesslog 中的 fromat

%x 匹配替代 time_format 和 date_format 的设定，可以同时调用两个的全局设置
%t 匹配替代 time_fromat 的设置
%d 匹配替代 date_format 的设置
%h 客户端 ip $remote_addr
%r 请求方法 $request
%m 请求算法 相当于 $request 中的 post 或 get 的匹配
%U 请求的 URL 路径 (包括任何查询字符串) 相当于 $request 中的 URL 匹配
%H 请求的协议 相当于 $request 中的 HTTP/1.1
%s 服务端返回客户端的状态 code $status
%b 返回客户端的 body size $body_bytes_sent
%R refer $http_referer
%u user-agent $http_user_agent
%D 服务请求的时间，以微秒为单位 $request_time
%T 服务请求的时间，以秒为单位 $request_time
%L 服务请求的时间，以毫秒为单位 $request_time
%^ 忽略官方没有对应参数的区域
以上是官方给出的所有匹配参数，原版见 http://www.goaccess.io/man

3 根据我们的 nginx 的 format 自定义日志格式

log_format main '$hostname $server_name $remote_addr - $remote_user [$time_local] "$request" '$status $body_bytes_sent "$http_referer" '
'"$http_user_agent" "$http_x_forwarded_for" "$upstream_addr" '
'"$upstream_response_time" "$request_time" "$http_cookie"';

例如：以上为 nginx 中的 format 配置

根据 nginx 中的 format 信息和真实访问日志内容，得出以下自定义格式

date-format %d/%b/%Y
log-format %^ %^ %h %^[%d:%^] "%r" %s %b "%R" "%u" "%^" "%^:%^" "%^" "%T" "%^"

注意　以上 **log-format** 每个参数之间有空格，跟真实访问日志中空格对应

下面对 **log-fomat** 后面匹配参数从左至右依次于　　**nginx** 的 **format** 和真实日志内容进行对应

Log-fromat 相应参数	**Nginx** 相应 **format**	
%^	'$hostname	
%^	$server_name	
%h	$remote_addr	
%^	- $remote_user	
[%d:%^]	[$time_local]	
“%r ”	"$request"	
%s	$status	
%b	$body_bytes_sent	
“%R ”	$http_referer	
"%u"	$http_user_agent	
"%^"	$http_x_forwarded_for	
"%^:%^"	$upstream_addr	
"%^"	$upstream_response_time	
"%T"	$request_time	
"%^"	$http_cookie	

date-format %d/%b/%Y　　对应的是　14/Jul/2015

%^[%d:%^]　　对应的是　- - [14/Jul/2015:18:13:06 +0800]　这部分是官方默认配置方法

这里的 %d 调用的就是　date-format 格式

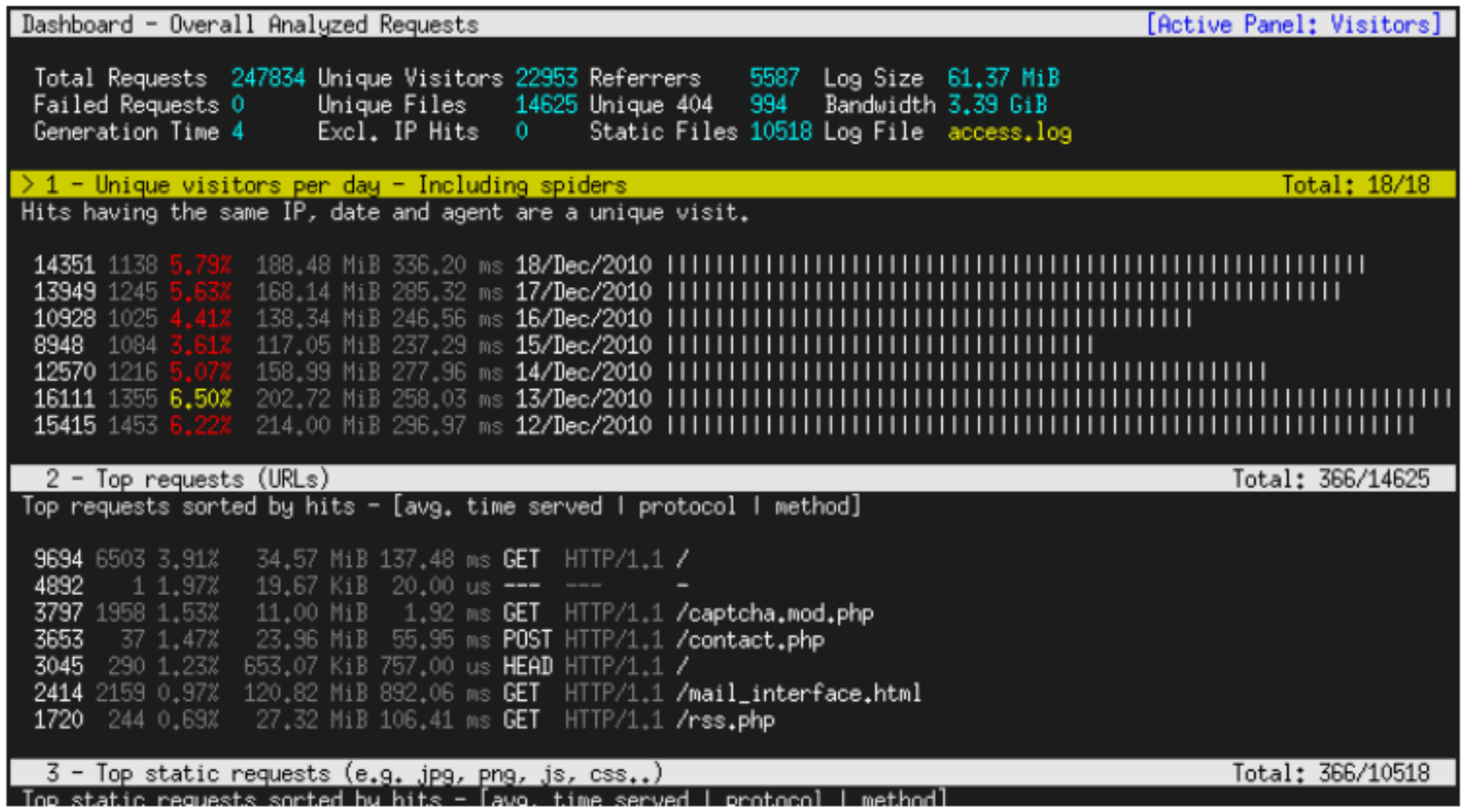
并且这里的匹配是官方默认方法，不用特意留意　　- -　：+ 这些符号和空格

通过 -c 参数调出的官方日志设置窗口可以分析出这点

到此日志格式就定义完成了

六　真正分析　**access**日志

此时通过　goaccess -f　log 日志名字　　直接可以分析，因为在配置文档设置过，不会再弹出之前官方的日志设置窗口，如果想弹出可以加　　-c 参数



键盘操作：

F1 或 **h**：帮助

F5 ：刷新主界面

q：退出程序 **/**当前窗口 **/**折叠当前模块

o 或 **Enter** ：展开选中的模块或窗口

0-9 以及 **Shift + 0** ：将选中的模块或窗口激活

k 和 **j**：模块内部移动

c：修改配色

^f 和 **^b**：模块中上下滚屏

tab shift+tab ：前后切换模块

s：模块内部排序选择

/：在所有模块中搜索 **(**支持正则 **)**

n：找到下个匹配

g 和 **G**：跳到第一项 **/**最后一项

七 使用 **GoAccess** 生成报表及一些管道的应用

使用 GoAccess 生成 html 报告：

goaccess -f access.log -a > report.html

生成 json 报告：

goaccess -f access.log -a -d -o json > report.json

CSV:

goaccess -f access.log -o csv > report.csv

由于 nginx 会自动压缩日志，以下命令可以直接分析压缩后的日志：

```
1zcat access.log.*.gz | goaccess
2#或者
3zcat -f access.log* | goaccess
```

一些其他关于管道的用法

```
zcat access.log.1.gz | goaccess
```

让 goaccess去分析已经打包压缩好的日志文件。
或者干脆分析目前下所有日志

```
zcat access.log* | goaccess
```

如果需要分析某天的日志，例如 10 月 5 号那天的日志，我们让 linux 管道命令来大显身手

```
sed -n '/05/Dec/2010/,$ p' access.log | goaccess
```

分析从 11 月 5 号到 12 月 5 号一个月内的日志

```
sed -n '/5/Nov/2010/,/5/Dec/2010/ p' access.log-s | goaccess
```

当你不希望在服务器上安装 goaccess程序，可以通过调用本地的 goaccess 程序来分析服务器上的日志

```
ssh user@server cat '/var/log/apache2/access.log' | goaccess
```

可以排除某个 IP 或者一个列表的 IP 交给 goaccess

```
grep -v "`cat exclude_vhost_list_file`" vhost_access.log | goaccess
```

八 两个问题的调研

1 关于设备识别 unknow的问题

此项应该是从 user_agent 判断，但是不清楚为什么识别不了 ios

分别取出 ios 的总数和 android 总数

```
root@131client:/server/scripts# cat api.m.mtime.cn_access.log |awk -F "[ ]+" '{print $15}'|grep "Android"|uniq -c
 17292 Android
root@131client:/server/scripts# cat api.m.mtime.cn_access.log |awk -F "[ ]+" '{print $15}'|grep "iOS"|uniq -c
 68494 iOS
```

（ios ）68494+（android ）17292=85786

用 68494/85786=79%

```
6 - Operating Systems                                                                                   Tot
Top Operating Systems sorted by unique visitors

 12495 76.88% Unknown   ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
 1579  9.72%  Android   ||||||||||||||||
 19    0.12%  Windows   |
 5     0.03%  Macintosh |
```

与报表中计算基本相似，没导致完全准确可能是由于下面截图导致

我对这列进行过滤发现会出现一些 (linux; 字段

```
iOS
Android
iOS
iOS
(Linux;
Android
iOS
Android
iOS
Android
Android
iOS
Android
iOS
Android
iOS
Android
Android
Android
Android
(Linux;
Android
```

专门对一条包含 IOS 的 accesslog 进行分析

日志内容如下：

XXXXXXXXXXXXX.com XXXXXXXXXXXXX 117.136.60.57 - - [14/Jul/2015:18:00:01 +0800] "GET /url HTTP/1.1" 200 1483 "-" "ios" "-" "XX.XX.XX.XX" "0.017" "0.017" "XXXXXXXXXXXXXXXXXXXXXXXXXX"

结果仍然是

```
 6 - Operating Systems
 Top Operating Systems sorted by unique visitors

  1 100.00% Unknown |||||||||||||||||||||||||||||
```

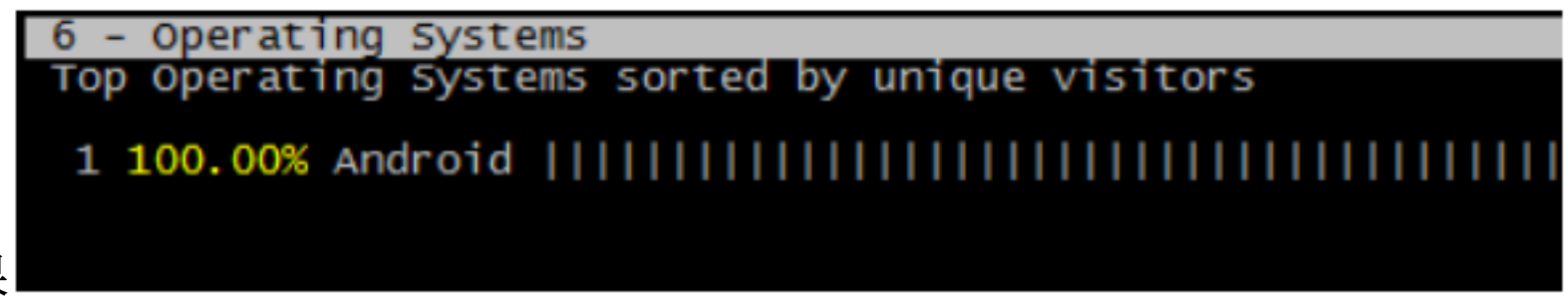
说明跟上面提到的 Linux 字段没有关系， goaccess 无法识别

然后再单独查看安卓字段的日志

日志内容如下：

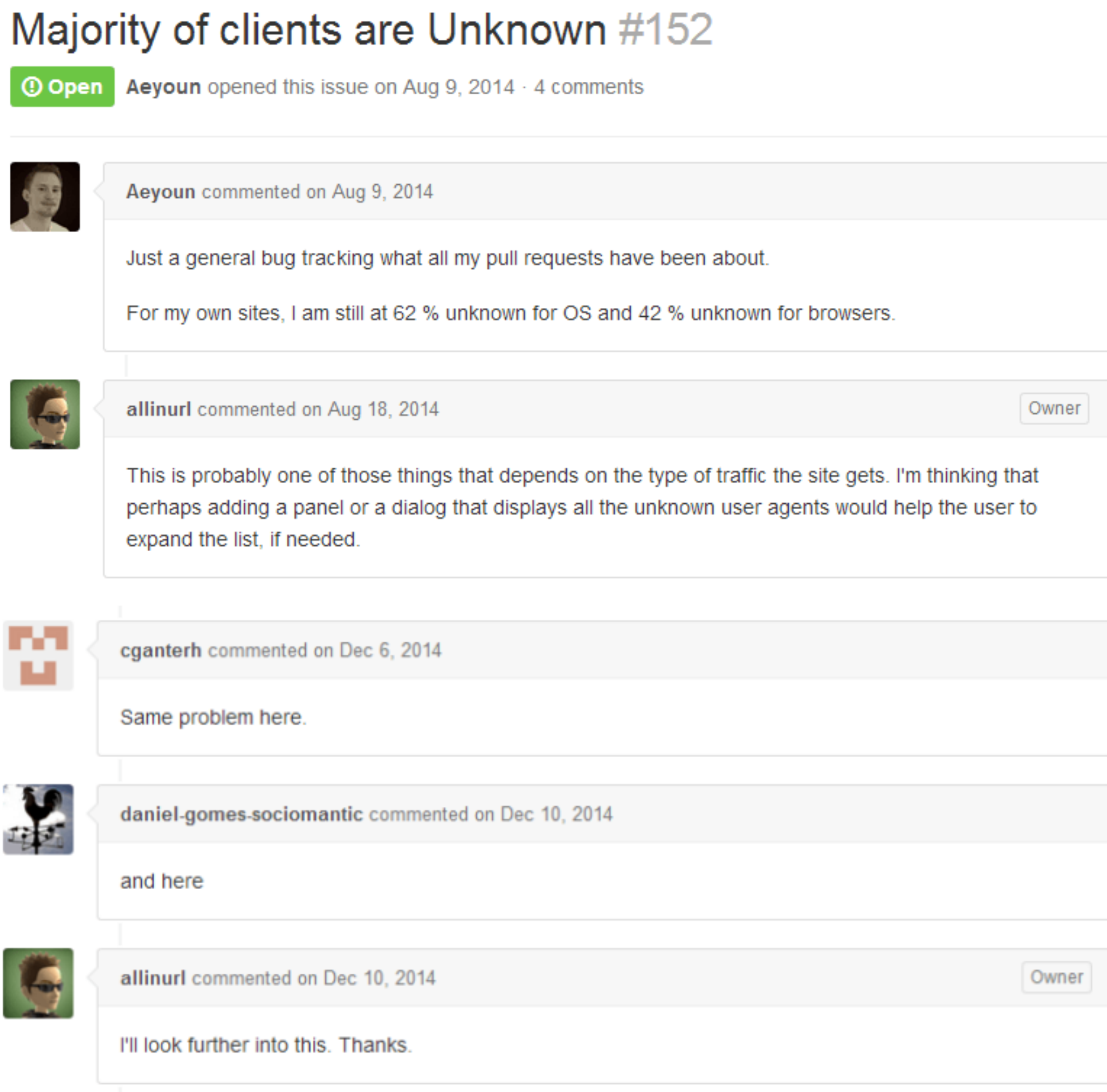
XXXXXXXXXXXX.com XXXXXXXXXXXXX 117.136.60.57 - - [14/Jul/2015:18:00:01 +0800] "GET /url HTTP/1.1" 200 1483 "-" "ios" "-" "XX.XX.XX.XX" "0.017" "0.017" "XXXXXXXXXXXXXXXXXXXXXXXXX"

结果



可以分析出安卓

Goaccess基本没什么详细文档，官方也没明确说法
这是我在 issue 查到一个相关的问题，如下图



Majority of clients are Unknown #152

Open **Aeyoun** opened this issue on Aug 9, 2014 · 4 comments

Aeyoun commented on Aug 9, 2014

Just a general bug tracking what all my pull requests have been about.

For my own sites, I am still at 62 % unknown for OS and 42 % unknown for browsers.

allinurl commented on Aug 18, 2014 Owner

This is probably one of those things that depends on the type of traffic the site gets. I'm thinking that perhaps adding a panel or a dialog that displays all the unknown user agents would help the user to expand the list, if needed.

cganterh commented on Dec 6, 2014

Same problem here.

daniel-gomes-sociomantic commented on Dec 10, 2014

and here

allinurl commented on Dec 10, 2014 Owner

I'll look further into this. Thanks.

2 没有找到对错误日志的分析方法

我首先用分析访问日志方法去匹配超时的那种　　errorlog

%d %t [%^] %^#%^: *%^ %^, %^:%h, %^:$^, %^:"%r", %^:"%R", %^:"%^"

这样是不成功的，然后我分析访问日志没逗号，我把逗号全部去掉也不行

然后我再以空格为分隔，所有未知区域用　　%^代替，还是不成功的，

然后我逐次去掉不同的符号去匹配也是不成功，然后就有了下面的想法

如果用 errorlog 去拼凑 accesslog

首先 accesslog 的解析是没问题的

log-format %^ %^ %h %^[%d:%^] "%r" %s %b "%R" "%u" "%^" "%^:%^" "%^" "%T" "%^"

我把后面都删除只剩下　　**log-format %^ %^ %h %^[%d:%^]**

想试试 **goaccess**有没有那么智能，只要匹配上正确的就能解析，显然不行

然后我又用这种方式　　**log-format %^ %^ %h %^[%d:%^] %^**

在后面加上了一个　**%^** 看看能不能全部匹配上后面的内容，显然还是不行

然后我又尝试

log-format %^ %^ %h %^[%d:%^] "%^" %^ %^ "%^" "%^" "%^" "%^:%^" "%^" "%^" "%^"

后面全部用 **%^** 忽略每个区域，数量都是对上的，但是显然还是不行

由此判断它对日志的分析有自己的一套格式，对于匹配要求很高

最后我把 **errorlog** 的所有未知区域都去掉，最小化分析

2015/07/17 19:10:20 61.158.152.98

"POST /Utility/PVCounter.api HTTP/1.1" "http://10.10.30.65:80/Utility/PVCounter.api"

log-format %d %H:%M:%S %h "%r" "%^"

还是没有成功