链滴

我的网站被暴力破解后之 ssh 加固与 Nginx 流量控制

作者：chenteng

原文链接：https://ld246.com/article/1638077455034

来源网站：链滴

我的网站被暴力破解后之ssh加固与Nginx流量控制

一、发现

当我登录我的服务器看到下面这行英文时，我就知道没那么简单

There were 622 failed login attempts since the last successful login.

我们查看一下/var/log/secure这个文件，查看ssh登陆纪录

```
Nov 28 10:48:47 website sshd[12326]: Received disconnect from 139.59.243.74 port 35042:11: Normal Shutdown, Thank you for playing [preauth]
Nov 28 10:48:47 website sshd[12326]: Disconnected from 139.59.243.74 port 35042 [preauth]
Nov 28 10:50:48 website sshd[12504]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=139.59.243.74  user=root
Nov 28 10:50:48 website sshd[12504]: pam_succeed_if(sshd:auth): requirement "uid >= 1000" not met by user "root"
Nov 28 10:50:49 website sshd[12504]: Failed password for root from 139.59.243.74 port 57840 ssh2
Nov 28 10:50:49 website sshd[12504]: Received disconnect from 139.59.243.74 port 57840:11: Normal Shutdown, Thank you for playing [preauth]
Nov 28 10:50:49 website sshd[12504]: Disconnected from 139.59.243.74 port 57840 [preauth]
Nov 28 10:52:12 website sshd[12638]: Invalid user yarn from 20.83.249.93 port 51352
Nov 28 10:52:12 website sshd[12638]: input_userauth_request: invalid user yarn [preauth]
Nov 28 10:52:12 website sshd[12638]: pam_unix(sshd:auth): check pass; user unknown
Nov 28 10:52:12 website sshd[12638]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=20.83.249.93
Nov 28 10:52:14 website sshd[12638]: Failed password for invalid user yarn from 20.83.249.93 port 51352 ssh2
Nov 28 10:52:14 website sshd[12638]: Received disconnect from 20.83.249.93 port 51352:11: Bye Bye [preauth]
Nov 28 10:52:14 website sshd[12638]: Disconnected from 20.83.249.93 port 51352 [preauth]
Nov 28 10:52:46 website sshd[12669]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=139.59.243.74  user=root
Nov 28 10:52:46 website sshd[12669]: pam_succeed_if(sshd:auth): requirement "uid >= 1000" not met by user "root"
Nov 28 10:52:48 website sshd[12669]: Failed password for root from 139.59.243.74 port 52378 ssh2
Nov 28 10:52:48 website sshd[12669]: Received disconnect from 139.59.243.74 port 52378:11: Normal Shutdown, Thank you for playing [preauth]
Nov 28 10:52:48 website sshd[12669]: Disconnected from 139.59.243.74 port 52378 [preauth]
Nov 28 10:53:50 website sshd[12755]: Accepted password for root from 112.231.27.0 port 50049 ssh2
Nov 28 10:53:50 website sshd[12755]: pam_unix(sshd:session): session opened for user root by (uid=0)
Nov 28 10:53:50 website sshd[12766]: Accepted password for root from 112.231.27.0 port 50062 ssh2
Nov 28 10:53:50 website sshd[12766]: pam_unix(sshd:session): session opened for user root by (uid=0)
Nov 28 10:54:45 website sshd[12868]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=139.59.243.74  user=root
Nov 28 10:54:45 website sshd[12868]: pam_succeed_if(sshd:auth): requirement "uid >= 1000" not met by user "root"
Nov 28 10:54:47 website sshd[12868]: Failed password for root from 139.59.243.74 port 46912 ssh2
Nov 28 10:54:47 website sshd[12868]: Received disconnect from 139.59.243.74 port 46912:11: Normal Shutdown, Thank you for playing [preauth]
Nov 28 10:54:47 website sshd[12868]: Disconnected from 139.59.243.74 port 46912 [preauth]
Nov 28 10:56:21 website sshd[13020]: Invalid user cbwx from 46.101.131.26 port 39504
Nov 28 10:56:21 website sshd[13020]: input_userauth_request: invalid user cbwx [preauth]
Nov 28 10:56:21 website sshd[13020]: pam_unix(sshd:auth): check pass; user unknown
Nov 28 10:56:21 website sshd[13020]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=46.101.131.26
Nov 28 10:56:24 website sshd[13020]: Failed password for invalid user cbwx from 46.101.131.26 port 39504 ssh2
Nov 28 10:56:24 website sshd[13020]: Connection closed by 46.101.131.26 port 39504 [preauth]
Nov 28 10:56:35 website sshd[13040]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=219.255.172.20  user=root
Nov 28 10:56:35 website sshd[13040]: pam_succeed_if(sshd:auth): requirement "uid >= 1000" not met by user "root"
Nov 28 10:56:37 website sshd[13040]: Failed password for root from 219.255.172.20 port 44562 ssh2
Nov 28 10:56:37 website sshd[13040]: Received disconnect from 219.255.172.20 port 44562:11: Bye Bye [preauth]
Nov 28 10:56:37 website sshd[13040]: Disconnected from 219.255.172.20 port 44562 [preauth]
Nov 28 10:56:41 website sshd[13042]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=139.59.243.74  user=root
Nov 28 10:56:41 website sshd[13042]: pam_succeed_if(sshd:auth): requirement "uid >= 1000" not met by user "root"
```


果然发现大量肉鸡ssh我们的服务器，这些IP全部来源于海外，我们截取其中的一段进行分析

```
Nov 28 10:56:35 website sshd[13040]: pam_unix(sshd:auth): authentication failure; logname=
id=0 euid=0 tty=ssh ruser= rhost=219.255.172.20  user=root
Nov 28 10:56:35 website sshd[13040]: pam_succeed_if(sshd:auth): requirement "uid >= 1000"
not met by user "root"
Nov 28 10:56:37 website sshd[13040]: Failed password for root from 219.255.172.20 port 445
2 ssh2
Nov 28 10:56:37 website sshd[13040]: Received disconnect from 219.255.172.20 port 44562:11
 Bye Bye [preauth]
Nov 28 10:56:37 website sshd[13040]: Disconnected from 219.255.172.20 port 44562 [preauth
```

这段话的意思是219.255.172.20 这个ip使用错误的密码尝试登陆我的服务器，也就是俗称的ssh暴力解！

二、处理

注意：公有云环境可以通过安全组限制源IP，因为笔者有远程运维的需求，所以这种方案被我PASS了，其他小伙伴可以采用这种方案，也是最安全有效的

2.1、更改ssh默认端口

我这里是将22端口更改为2222端口，使用其他把下面命令中的2222更改即可

```
sed -i 's/#Port 22/Port 2222/g' /etc/ssh/sshd_config    ##更改为2222端口
systemctl restart sshd    ##重启ssh服务，当前连接不会断开，下次连接需要使用2222端口
```

注意：如果是公有云环境，需要在安全组放通上面开放的端口

2.2、用/etc/hosts.allow 和/etc/hosts.deny来控制

思路：我们可以写一个脚本，通过/var/log/secure中，通过统计访问失败的ip，超过四次就将他加入/etc/hosts.deny中，加入之后就不能登录到服务器了，即使密码正确，必须要管理员来把他删掉才以进入。把这个脚本加入crontab里。

2.2.1、添加常用ip到/etc/hosts.allow中

先写/etc/hosts.allow。这个很关键。可以把我们日常登录的ip都写上，以免登录出错多了，登录不去

比如我的ip是33.33.33.33,我们可以这样写

```
vim /etc/hosts.allow

#######################
sshd:33.33.33.33:allow
```

2.2.2、编写脚本

脚本如下：

```
#!/bin/bash
```

```
cat /dev/null > /root/black.deny

awk '/Failed/{time[$(NF-3)]++}END{for(ip in time) if(time[ip]> 4)print ip}' /var/log/secure >> /
oot/black.deny

for i in `cat /root/black.deny`
do
      grep $i /etc/hosts.deny > /dev/null
      if [ $? -ne 0 ]
      then
            echo "sshd:$i:deny"
            echo "sshd:$i:deny" >> /etc/hosts.deny
      fi
done
```

脚本解读:

通过awk命令中的for循环于if语句，判断登陆失败的times是否超过4，如果超过4，那么我们将他的i进行组合存入hosts.deny文件中，实现拒绝该IP，上面还做了一个判断，已经存在的就不再添加了。

2.2.3、写入定时任务中

1分钟执行一次，并把日志存到/var/log/ssh_deny.log中

```
crontab -e

0 */1 * * * sh /root/cron_sh/sshd_failed_4.sh >> /var/log/ssh_deny.log ##wq保存

systemctl restart sshd ##重启ssh服务
```

三、Nginx 配置防御DDos，cc等流量攻击

3.1、限制同一时间段ip访问次数

nginx可以通过ngx_http_limit_conn_module和ngx_http_limit_req_module配置来限制ip在同一时段的访问次数.

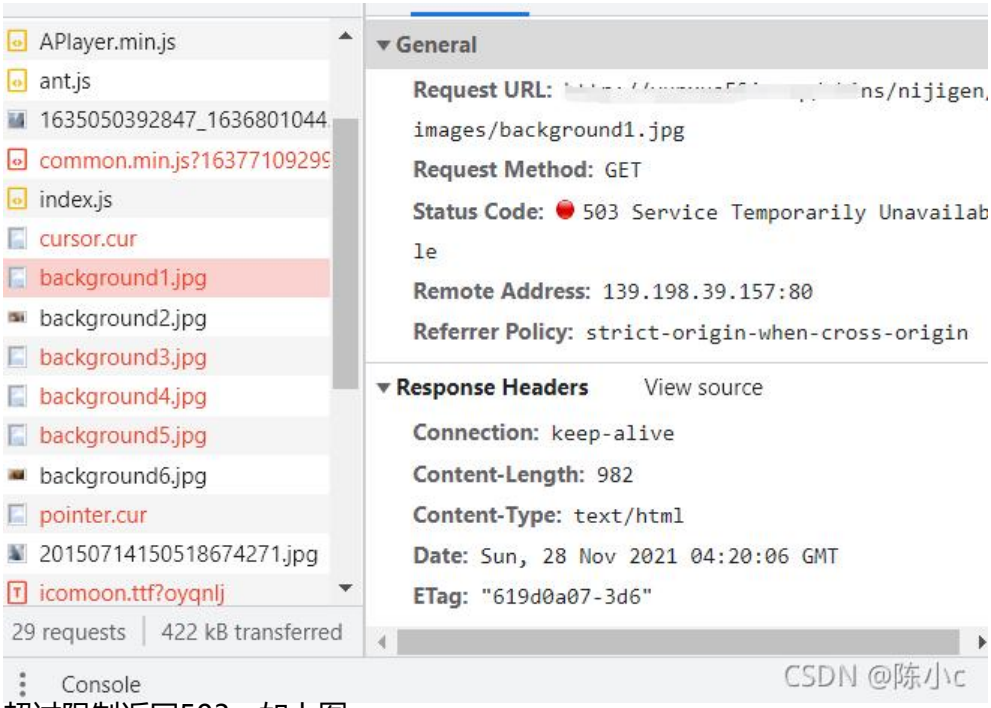
● ngx_http_limit_conn_module：该模块用于限制每个定义的密钥的连接数，特别是单个IP地址的接数．使用limit_conn_zone和limit_conn指令．

● ngx_http_limit_req_module：用于限制每一个定义的密钥的请求的处理速率，特别是从一个单一的P地址的请求的处理速率。使用“泄漏桶”方法进行限制．指令：limit_req_zone和limit_req．

配置用法实例如下:

```
http {
  limit_conn_zone $binary_remote_addr zone=addr:10m;
      #定义一个名为addr的limit_req_zone用来存储session，大小是10M内存，
  #以$binary_remote_addr 为key,
  #nginx 1.18以后用limit_conn_zone替换了limit_conn,
  #且只能放在http{}代码段．
  ...
  server {
```

```
    ...
    location /download/ {
      limit_conn addr 1;        #连接数限制
     #设置给定键值的共享内存区域和允许的最大连接数。超出此限制时，服务器将返回503（服务临
不可用）错误.
                    #如果区域存储空间不足，服务器将返回503（服务临时不可用）错误
    }
```



超过限制返回503，如上图

可能有几个limit_conn指令,以下配置将限制每个客户端IP与服务器的连接数，同时限制与虚拟服务器总连接数：

```
http {
  limit_conn_zone $binary_remote_addr zone=perip:10m;
  limit_conn_zone $server_name zone=perserver:10m
  ...
  server {
    ...
    limit_conn perip 10;            #单个客户端ip与服务器的连接数.
    limit_conn perserver 100;       #限制与服务器的总连接数
    }
```

参考文档：http://nginx.org/en/docs/http/ngx_http_limit_conn_module.html

3.2、限制某一时间内，单一IP的请求数.

ngx_http_limit_req_module：限制某一时间内，单一IP的请求数.

配置如下：

```
http {
 limit_req_zone $binary_remote_addr zone=one:10m rate=1r/s;
 ...
      #定义一个名为one的limit_req_zone用来存储session，大小是10M内存，
```

```
    #以$binary_remote_addr 为key,限制平均每秒的请求为1个，
    #1M能存储16000个状态，rete的值必须为整数，

 server {
   ...
   location /search/ {
     limit_req zone=one burst=5;
                    #限制每ip每秒不超过1个请求，漏桶数burst为5,也就是队列.
                    #nodelay，如果不设置该选项，严格使用平均速率限制请求数，超过的请求被延
处理.
                    #举个栗子:
                    #设置rate=20r/s每秒请求数为２０个，漏桶数burst为5个，
                    #brust的意思就是，如果第1秒、2,3,4秒请求为19个，第5秒的请求为25个是被允
的，可以理解为20+5
                    #但是如果你第1秒就25个请求，第2秒超过20的请求返回503错误.
                    #如果区域存储空间不足，服务器将返回503（服务临时不可用）错误
                    #速率在每秒请求中指定（r/s）。如果需要每秒少于一个请求的速率，则以每分钟
请求（r/m）指定。

   }
```

3.3、禁止ip或ip网段

统计每个ip的登陆次数

```
awk '{print $1}' access.log |sort |uniq -c|sort -n
```

在nginx的安装目录下面,新建屏蔽ip文件，命名为guolv_ip.conf，以后新增加屏蔽ip只需编辑这个文即可。 加入如下内容并保存:

```
deny 66.249.79.84 ;
```

在nginx的配置文件nginx.conf中加入如下配置，可以放到http, server, location, limit_except语句，需要注意相对路径，本例当中nginx.conf，guolv_ip.conf在同一个目录中。

```
include guolv_ip.conf;
```

保存nginx.conf文件，然后测试现在的nginx配置文件是否是合法的:

```
nginx -t
```

如果配置没有问题，就会输出:

```
the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok
configuration file /usr/local/nginx/conf/nginx.conf test is successful
```

如果配置有问题就需要检查下哪儿有语法问题，如果没有问题，需要执行下面命令，重载 nginx 配置件:

```
service nginx  reload
```

屏蔽ip的配置文件既可以屏蔽单个ip，也可以屏蔽ip段，或者只允许某个ip或者某个ip段访问。

```
//屏蔽单个ip访问

deny IP;
```

```
//允许单个ip访问

allow IP;

//屏蔽所有ip访问

deny all;

//允许所有ip访问

allow all;

//屏蔽整个段即从123.0.0.1到123.255.255.254访问的命令

deny 123.0.0.0/8

//屏蔽IP段即从123.45.0.1到123.45.255.254访问的命令

deny 124.45.0.0/16

//屏蔽IP段即从123.45.6.1到123.45.6.254访问的命令

deny 123.45.6.0/24

//如果你想实现这样的应用，除了几个IP外，其他全部拒绝，
//那需要你在guolv_ip.conf中这样写

allow 1.1.1.1;
allow 1.1.1.2;
deny all;
```

单独网站屏蔽IP的方法，把include guolv_ip.conf; 放到网址对应的在server{}语句块，
所有网站屏蔽IP的方法，把include guolv_ip.conf; 放到http {}语句块。

参考网站：

https://my.oschina.net/u/2607135/blog/1844590

https://blog.csdn.net/qq_41201816/article/details/88136947