

[Top](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/ARCHITECTURE/DAY02/CASE/01/index.html" \l "page_top_case)

**NSD ARCHITECTURE DAY02**

1. [案例1：安装Logstash](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/ARCHITECTURE/DAY02/CASE/01/index.html#case1)
2. [案例2：编写logstash配置文件](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/ARCHITECTURE/DAY02/CASE/01/index.html#case2)
3. [案例3：Logstash input插件](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/ARCHITECTURE/DAY02/CASE/01/index.html#case3)
4. [案例4：Web日志解析实验](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/ARCHITECTURE/DAY02/CASE/01/index.html#case4)
5. [案例6：部署beats与filebeat](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/ARCHITECTURE/DAY02/CASE/01/index.html#case5)
6. [案例7：实时分析日志案例](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/ARCHITECTURE/DAY02/CASE/01/index.html#case6)

**1 案例1：安装Logstash**

**1.1 问题**

本案例要求：

* 创建虚拟机并安装 logstash
* 最低配置： 2cpu，2G内存，10G硬盘
* 虚拟机IP： 192.168.1.47 logstash
* 创建虚拟机并安装 apache
* 最低配置： 1cpu，1G内存，10G硬盘
* 默认首页： hello world
* 虚拟机IP： 192.168.1.48 web

**1.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：安装apache

1）安装apache

1. [root@es-0005 ~]# yum -y install httpd
2. [root@es-0005 ~]# echo hello world > /var/www/html/index.html
3. [root@es-0005 ~]# systemctl restart httpd
4. [root@es-0005 ~]# systemctl enable httpd

步骤二：安装logstash

1）配置主机名，ip和yum源，配置/etc/hosts

1. [root@logstash ~]# vim /etc/hosts
2. 192.168.1.41 es-0001
3. 192.168.1.42 es-0002
4. 192.168.1.43 es-0003
5. 192.168.1.44 es-0004
6. 192.168.1.45 es-0005
7. 192.168.1.46 kibana
8. 192.168.1.47 logstash

2）安装java-1.8.0-openjdk和logstash

1. [root@logstash ~]# yum -y install java-1.8.0-openjdk
2. [root@logstash ~]# yum -y install logstash
3. [root@logstash ~]# java -version
4. openjdk version "1.8.0\_161"
5. OpenJDK Runtime Environment (build 1.8.0\_161-b14)
6. OpenJDK 64-Bit Server VM (build 25.161-b14, mixed mode)
7. [root@logstash ~]# touch /etc/logstash/logstash.conf
8. [root@logstash ~]# /opt/logstash/bin/logstash --version
9. logstash 2.3.4
10. [root@logstash ~]# /opt/logstash/bin/logstash-plugin list //查看插件
11. ...
12. logstash-input-stdin    //标准输入插件
13. logstash-output-stdout    //标准输出插件
14. ...
15. [root@logstash ~]# vim /etc/logstash/logstash.conf
16. input{
17. stdin{
18. }
19. }
20. filter{
21. }
22. output{
23. stdout{
24. }
25. }
26. [root@logstash ~]# /opt/logstash/bin/logstash -f /etc/logstash/logstash.conf
27. //启动并测试
28. Settings: Default pipeline workers: 2
29. Pipeline main started
30. aa        //logstash 配置从标准输入读取输入源,然后从标准输出输出到屏幕
31. 2018-09-15T06:19:28.724Z logstash aa

**2 案例2：编写logstash配置文件**

**2.1 问题**

本案例要求：

* 编写 logstash 配置文件
* 标准输入采用 json 编码格式
* 标准输出采用 rubydebug 编码格式
* 启动 logstash 验证

**2.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：codec类插件

1）codec类插件

1. [root@logstash ~]# vim /etc/logstash/logstash.conf
2. input{
3. stdin{ codec => "json" }
4. }
5. filter{ }
6. output{
7. stdout{ codec => "rubydebug" }
8. }
9. [root@logstash ~]# /opt/logstash/bin/logstash -f /etc/logstash/logstash.conf
10. Settings: Default pipeline workers: 2
11. Pipeline main started
12. a
13. {
14. "message" => "a",
15. "tags" => [
16. [0] "\_jsonparsefailure"
17. ],
18. "@version" => "1",
19. "@timestamp" => "2020-05-23T12:34:51.250Z",
20. "host" => "logstash"
21. }

**3 案例3：Logstash input插件**

**3.1 问题**

本案例要求：

* 编写 logstash 配置文件
* 从文件中读取数据，并在屏幕显示
* 启动 logstash 验证

**3.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：file模块插件

1）file模块插件

1. [root@logstash ~]# vim /etc/logstash/logstash.conf
2. input{
3. file {
4. path => [ "/tmp/a.log", "/tmp/b.log" ]
5. sincedb\_path => "/var/lib/logstash/sincedb"    //记录读取文件的位置
6. start\_position => "beginning"                //配置第一次读取文件从什么地方开始
7. type => "testlog"                    //类型名称
8. }
9. }
10. filter{
11. }
12. output{
13. stdout{
14. codec => "rubydebug"
15. }
16. }
17. [root@logstash ~]# touch /tmp/a.log
18. [root@logstash ~]# touch /tmp/b.log
19. [root@logstash ~]# /opt/logstash/bin/logstash -f /etc/logstash/logstash.conf

另开一个终端：写入数据

1. [root@logstash ~]# echo a1 > /tmp/a.log
2. [root@logstash ~]# echo b1 > /var/tmp/b.log

之前终端查看：

1. [root@logstash ~]# /opt/logstash/bin/logstash -f /etc/logstash/logstash.conf
2. Settings: Default pipeline workers: 2
3. Pipeline main started
4. {
5. "message" => "a1",
6. "@version" => "1",
7. "@timestamp" => "2019-03-12T03:40:24.111Z",
8. "path" => "/tmp/a.log",
9. "host" => "logstash",
10. "type" => "testlog"
11. }
12. {
13. "message" => "b1",
14. "@version" => "1",
15. "@timestamp" => "2019-03-12T03:40:49.167Z",
16. "path" => "/tmp/b.log",
17. "host" => "logstash",
18. "type" => "testlog"
19. }

**4 案例4：Web日志解析实验**

**4.1 问题**

本案例要求：

* Web日志解析实验
* 复制一条 web 日志添加到文件中
* 使用 grok 匹配出日志的各个字段含义转化成 json 格式

**4.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：filter grok模块插件

grok插件：

解析各种非结构化的日志数据插件

grok使用正则表达式把飞结构化的数据结构化

在分组匹配，正则表达式需要根据具体数据结构编写

虽然编写困难，但适用性极广

解析Apache的日志，之前已经安装过的可以不用安装

1. [root@es-0005 ~]# yum -y install httpd
2. [root@es-0005 ~]# systemctl restart httpd

浏览器访问网页，在/var/log/httpd/access\_log有日志出现

1. [root@es-0005 ~]# cat /var/log/httpd/access\_log
2. 192.168.1.254 - - [12/Mar/2019:11:51:31 +0800] "GET /favicon.ico HTTP/1.1" 404 209 "-" "Mozilla/5.0 (X11; Linux x86\_64; rv:52.0) Gecko/20100101 Firefox/52.0"
3. [root@logstash ~]# vim /etc/logstash/logstash.conf
4. input{
5. file {
6. path => [ "/tmp/a.log", "/tmp/b.log" ]
7. sincedb\_path => "/var/lib/logstash/sincedb"
8. start\_position => "beginning"
9. type => "testlog"
10. }
11. }
12. filter{
13. grok{
14. match => [ "message", "(?<key>reg)" ]
15. }
16. }
17. output{
18. stdout{
19. codec => "rubydebug"
20. }
21. }

复制/var/log/httpd/access\_log的日志到logstash下的/tmp/a.log

1. [root@logstash ~]# vim /tmp/a.log
2. 192.168.1.254 - - [15/Sep/2018:18:25:46 +0800] "GET / HTTP/1.1" 403 4897 "-" "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0"
3. [root@logstash ~]# /opt/logstash/bin/logstash -f /etc/logstash/logstash.conf
4. //出现message的日志，但是没有解析是什么意思
5. Settings: Default pipeline workers: 2
6. Pipeline main started
7. {
8. "message" => ".168.1.254 - - [15/Sep/2018:18:25:46 +0800] \"GET / HTTP/1.1\" 403 4897 \"-\" \"Mozilla/5.0 (Windows NT 6.1; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0\"",
9. "@version" => "1",
10. "@timestamp" => "2018-09-15T10:26:51.335Z",
11. "path" => "/tmp/a.log",
12. "host" => "logstash",
13. "type" => "testlog",
14. "tags" => [
15. [0] "\_grokparsefailure"
16. ]
17. }

若要解决没有解析的问题，同样的方法把日志复制到/tmp/a.log，logstash.conf配置文件里面修改grok

查找正则宏路径

1. [root@logstash ~]# cd /opt/logstash/vendor/bundle/ \
2. jruby/1.9/gems/logstash-patterns-core-2.0.5/patterns/
3. [root@logstash ~]# vim grok-patterns //查找COMBINEDAPACHELOG
4. COMBINEDAPACHELOG %{COMMONAPACHELOG} %{QS:referrer} %{QS:agent}
5. [root@logstash ~]# vim /etc/logstash/logstash.conf
6. ...
7. filter{
8. grok{
9. match => ["message", "%{COMBINEDAPACHELOG}"]
10. }
11. }
12. ...

解析出的结果

1. [root@logstash ~]# /opt/logstash/bin/logstash -f /etc/logstash/logstash.conf
2. Settings: Default pipeline workers: 2
3. Pipeline main started
4. {
5. "message" => "192.168.1.254 - - [15/Sep/2018:18:25:46 +0800] \"GET /noindex/css/open-sans.css HTTP/1.1\" 200 5081 \"http://192.168.1.65/\" \"Mozilla/5.0 (Windows NT 6.1; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0\"",
6. "@version" => "1",
7. "@timestamp" => "2018-09-15T10:55:57.743Z",
8. "path" => "/tmp/a.log",
9. ZZ "host" => "logstash",
10. "type" => "testlog",
11. "clientip" => "192.168.1.254",
12. "ident" => "-",
13. "auth" => "-",
14. "timestamp" => "15/Sep/2019:18:25:46 +0800",
15. "verb" => "GET",
16. "request" => "/noindex/css/open-sans.css",
17. "httpversion" => "1.1",
18. "response" => "200",
19. "bytes" => "5081",
20. "referrer" => "\"http://192.168.1.65/\"",
21. "agent" => "\"Mozilla/5.0 (Windows NT 6.1; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0\""
22. }
23. ...

**5 案例6：部署beats与filebeat**

**5.1 问题**

本案例要求：

* 打通ELK全流程
* 在 logstash 上安装配置 beats 插件
* web 服务器上安装 filebeat
* 使用 filebeat 收集 web 日志，并发送给 logstash
* 将日志转化为json格式存入elasticsearch

**5.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：filter grok模块插件

1）在之前安装了Apache的主机上面安装filebeat

1. [root@es-0005 ~]# yum -y install filebeat
2. [root@es-0005 ~]# vim/etc/filebeat/filebeat.yml
3. paths:
4. - /var/log/httpd/access\_log //日志的路径，短横线加空格代表yml格式
5. document\_type: apachelog //文档类型
6. elasticsearch:        //加上注释
7. hosts: ["localhost:9200"]                //加上注释
8. logstash:                    //去掉注释
9. hosts: ["192.168.1.57:5044"]     //去掉注释,logstash那台主机的ip
10. [root@se5 ~]# systemctl start filebeat

**6 案例7：实时分析日志案例**

**6.1 问题**

本案例要求：

* 在web服务器上部署 filebeat
* 搭建ELK集群，并导入filebeat的数据
* 实现web日志流量实时分析

**6.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：接上个案例继续操作

1）更改配置

1. [root@logstash ~]# vim /etc/logstash/logstash.conf
2. input{
3. stdin{ codec => "json" }
4. beats{
5. port => 5044
6. }
7. file {
8. path => [ "/tmp/a.log", "/tmp/b.log" ]
9. sincedb\_path => "/var/lib/logstash/sincedb"
10. start\_position => "beginning"
11. type => "testlog"
12. }
13. filter{
14. if [type] == "apachelog"{
15. grok{
16. match => ["message", "%{COMBINEDAPACHELOG}"]
17. }}
18. }
19. output{
20. stdout{ codec => "rubydebug" }
21. if [type] == "filelog"{
22. elasticsearch {
23. hosts => ["192.168.1.41:9200", "192.168.1.42:9200"]
24. index => "filelog"
25. flush\_size => 2000
26. idle\_flush\_time => 10
27. }}
28. }
29. [root@logstash logstash]# /opt/logstash/bin/logstash \
30. -f /etc/logstash/logstash.conf

打开另一终端查看5044是否成功启动

1. [root@logstash ~]# netstat -antup | grep 5044
2. tcp6 0 0 :::5044 :::\* LISTEN 23776/java
3. [root@se5 ~]# firefox 192.168.1.55 //ip为安装filebeat的那台机器

回到原来的终端，有数据

2）修改logstash.conf文件

1. [root@logstash logstash]# vim logstash.conf
2. ...
3. output{
4. stdout{ codec => "rubydebug" }
5. if [type] == "apachelog"{
6. elasticsearch {
7. hosts => ["192.168.1.51:9200", "192.168.1.52:9200"]
8. index => "apachelog"
9. flush\_size => 2000
10. idle\_flush\_time => 10
11. }}
12. }

浏览器访问Elasticsearch，有apachelog，如图-1所示：

1. firefox http://192.168.1.45:9200/\_plugin/head

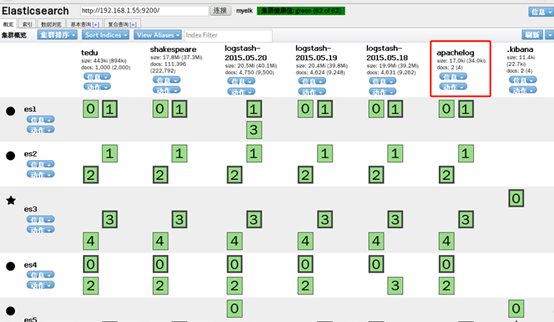


图-1

Kibana: 导入日志,展示图形

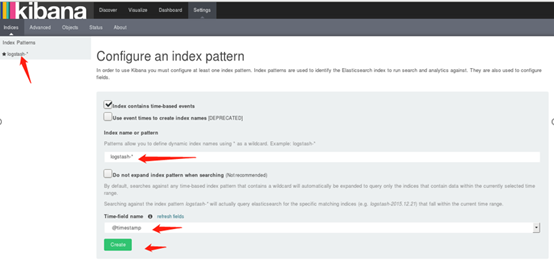


图-2



图-3