

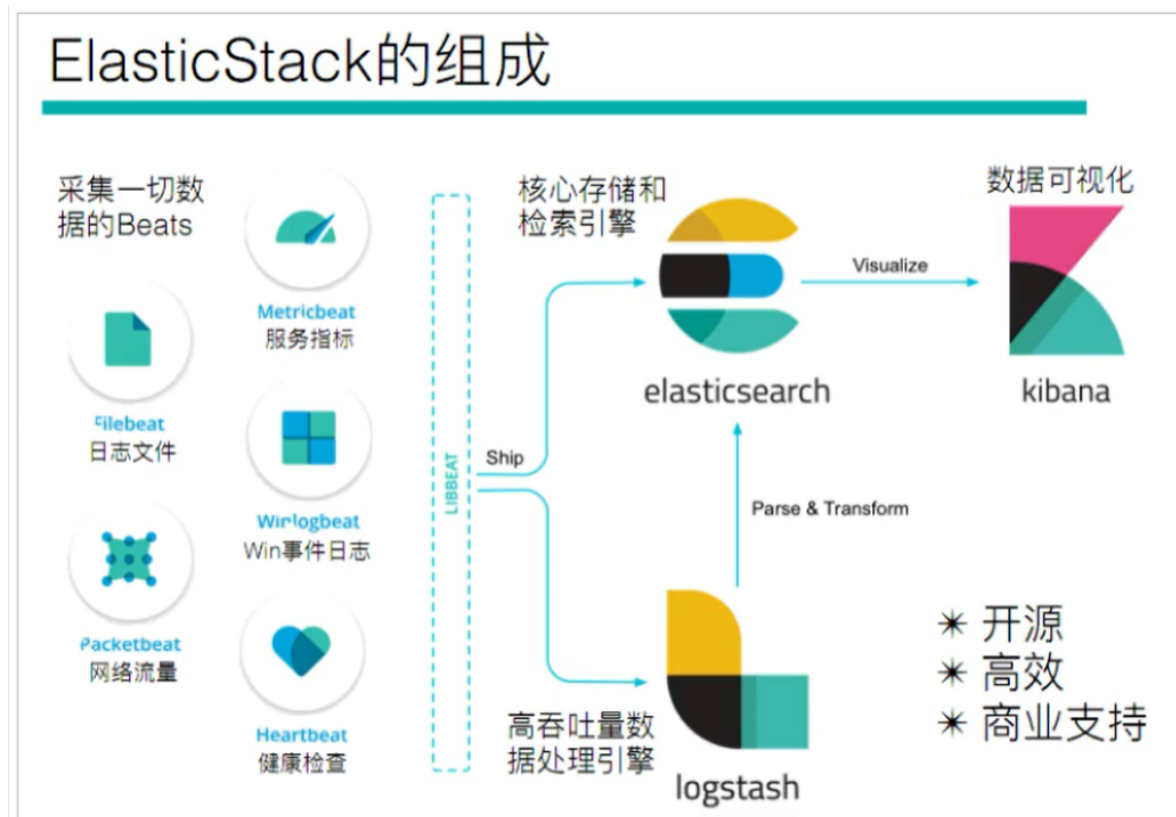
Logstash入门简介

介绍

Logstash是一个开源的服务器端数据处理管道，能够同时从多个来源采集数据，转换数据，然后将数据发送到最喜欢的存储库中（我们的存储库当然是ElasticSearch）

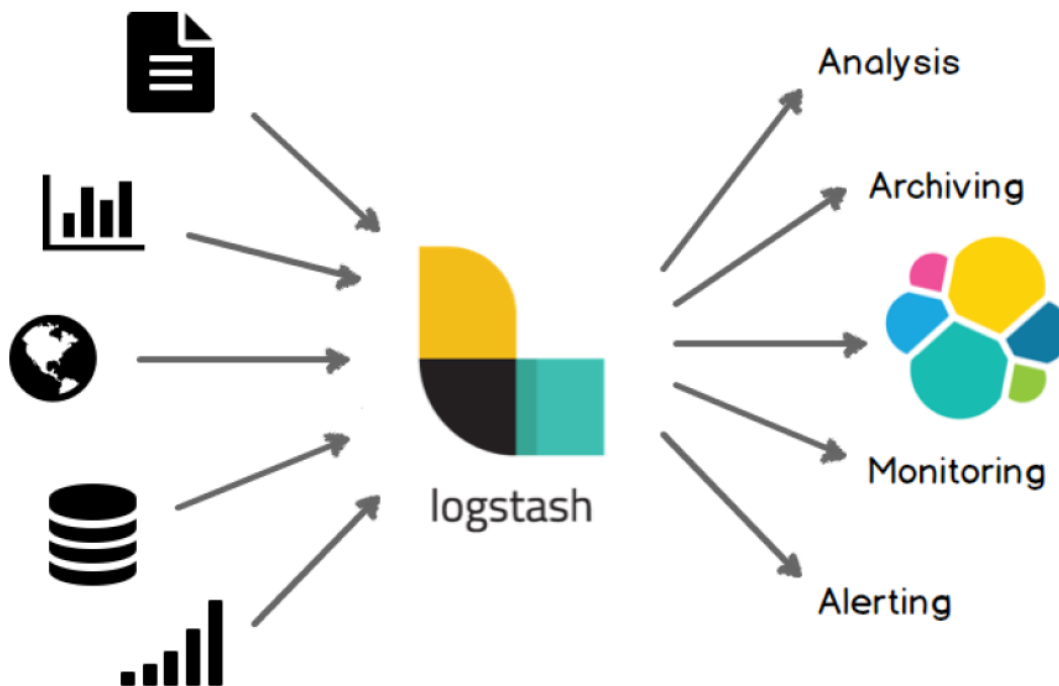


我们回到我们ElasticStack的架构图，可以看到Logstash是充当数据处理的需求的，当我们的数据需要处理的时候，会将它发送到Logstash进行处理，否则直接送到ElasticSearch中



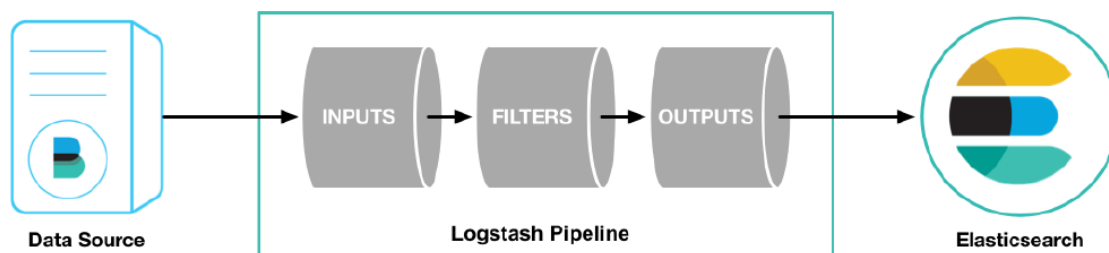
用途

Logstash可以处理各种各样的输入，从文档，图表中=，数据库中，然后处理完后，发送到



部署安装

Logstash主要是将数据源的数据进行一行一行的处理，同时还直接过滤切割等功能。



首先到官网下载logstash: <https://www.elastic.co/cn/downloads/logstash>

选择我们需要下载的版本:

Download Logstash

Want to upgrade? We'll give you a hand. [Migration Guide »](#)

Version: 7.9.1

Release date: September 04, 2020

License: [Elastic License](#)

Downloads: [TAR.GZ](#) [sha asc](#)

[DEB](#) [sha asc](#)

[ZIP](#) [sha asc](#)

[RPM](#) [sha asc](#)

Package Managers: Install with [yum](#)

Install with [apt-get](#)

Install with [homebrew](#)

Containers: Run with [Docker](#)

下载完成后，使用xftp工具，将其丢入到服务器中

```
#检查jdk环境, 要求jdk1.8+
java -version

#解压安装包
tar -xvf logstash-7.9.1.tar.gz

#第一个logstash示例
bin/logstash -e 'input { stdin { } } output { stdout { } }'
```

其实原来的logstash的作用, 就是为了做数据的采集, 但是因为logstash的速度比较慢, 所以后面使用beats来代替了Logstash, 当我们使用上面的命令进行启动的时候, 就可以发现了, 因为logstash使用java写的, 首先需要启动虚拟机, 最后下图就是启动完成的截图

```
telnetroot@elasticstack:~$ cd /opt/logstash/
telnetroot@elasticstack:~/opt/logstash$ bin/logstash -e 'input { stdin { } } output { stdout { } }'
Sending Logstash logs to /opt/logstash/logs which is now configured via log4j.properties
2020-09-24T14:09:06.013[INFO] [logstash.runner] Starting Logstash {"logstash.version":"7.9.1", "ruby.version":"ruby 2.7.1.0 (2.5.7) 2020-08-03 9a89c94bcc Java HotSpot(TM) 64-Bit Server VM 25.131-b11 on linux-x86_64"}
2020-09-24T14:09:07.460[INFO] [logstash.setting.writable_directory] Creating directory (settings->path.queue", .path=>/opt/logstash/data/queue")
2020-09-24T14:09:07.460[INFO] [logstash.setting.writable_directory] Creating directory (settings->path.dead_letter_queue", .path=>/opt/logstash/data/dead_letter_queue")
2020-09-24T14:09:09.360[WARN] [logstash.config.source.multilocal] Ignoring the 'pipeline.yml' file because modules or command line options are specified
2020-09-24T14:09:09.822[INFO] [logstash.agent] No persistent UUID file found. Generating new UUID [uuid=>"3780fec-887a-454c-9225-261c7a7d71e". .path=>/opt/logstash/data/uuid")
2020-09-24T14:09:12.442[INFO] [org.reflections.Reflections] Reflections took 25 ms to scan 1 urls, producing 22 keys and 45 values
2020-09-24T14:09:17.922[INFO] [logstash.javapipeline] [main] Starting pipeline (pipeline_id=>"main", pipeline_workers=>1, pipeline_batch_size=>125, pipeline_batch_delay=>50, pipeline_max_inflight=>125, pipeline.sources=>[{"config_string":"", .thread=>"ac
ad.0e575812b26 run"})
2020-09-24T14:09:20.267[INFO] [logstash.javapipeline] [main] Pipeline java execution initialization time ("seconds">2.3)
2020-09-24T14:09:20.425[INFO] [logstash.javapipeline] [main] Pipeline started ("pipeline.id">"main")
The stdin plugin is now waiting for input
2020-09-24T14:09:20.590[INFO] [logstash.agent] Pipelines running {:count=>1, :running_pipelines=>[main], :non_running_pipelines=>[]}
2020-09-24T14:09:21.301[INFO] [logstash.agent] Successfully started Logstash API endpoint {:port=>9600}
```

测试

我们在控制台输入 hello, 马上就能看到它的输出信息

```
[2020-09-24T14:09:21.301][INFO] [logstash.agent] Successfully started Logstash API endpoint
hello
{
  "host" => "ElasticStack",
  "@version" => "1",
  "message" => "hello",
  "@timestamp" => 2020-09-24T14:10:31.037Z
}
```

配置详解

Logstash的配置有三部分, 如下所示

```
input { #输入
  stdin { ... } #标准输入
}
filter { #过滤, 对数据进行分割、截取等处理
  ...
}
output { #输出
  stdout { ... } #标准输出
}
```

输入

- 采集各种样式、大小和来源的数据, 数据往往以各种各样的形式, 或分散或集中地存在于很多系统中。
- Logstash 支持各种输入选择, 可以在同一时间从众多常用来源捕捉事件。能够以连续的流式传输方式, 轻松地从您的日志、指标、Web 应用、数据存储以及各种 AWS 服务采集数据。



过滤

- 实时解析和转换数据
- 数据从源传输到存储库的过程中，Logstash 过滤器能够解析各个事件，识别已命名的字段以构建结构，并将它们转换成通用格式，以便更轻松、更快速地分析和实现商业价值。



输出

Logstash 提供众多输出选择，您可以将数据发送到您要指定的地方，并且能够灵活地解锁众多下游用例。



读取自定义日志

前面我们通过Filebeat读取了nginx的日志，如果是自定义结构的日志，就需要读取处理后才能使用，所以，这个时候就需要使用Logstash了，因为Logstash有着强大的处理能力，可以应对各种各样的场景。

日志结构

```
2019-03-15 21:21:21|ERROR|1 读取数据出错|参数: id=1002
```

可以看到，日志中的内容是使用“|”进行分割的，使用，我们在处理的时候，也需要对数据做分割处理。

编写配置文件

```
vim mogublog-pipeline.conf
```

然后添加如下内容

```
input {
  file {
    path => "/soft/beats/logs/app.log"
    start_position => "beginning"
  }
}
filter {
  mutate {
    split => {"message"=>"|"}
  }
}
output {
  stdout { codec => rubydebug }
}
```

启动

```
#启动
./bin/logstash -f ./mogublog-pipeline.conf
```

然后我们就插入我们的测试数据

```
echo "2019-03-15 21:21:21|ERROR|读取数据出错|参数: id=1002" >> app.log
```

然后我们就可以看到logstash就会捕获到刚刚我们插入的数据，同时我们的数据也被分割了

```
[2020-09-24T14:44:21.521][INFO ][logstash.agent] Successfully started Logstash API endpoint {:port=>9600}
{
  "path" => "/soft/beats/logs/app.log",
  "message" => [
    [0] "2019-03-15 21:21:21",
    [1] "ERROR",
    [2] "读取数据出错",
    [3] "参数: id=1002"
  ],
  "@timestamp" => 2020-09-24T14:46:36.974Z,
  "host" => "ElasticStack",
  "@version" => "1"
}
```

输出到Elasticsearch

我们可以修改我们的配置文件，将我们的日志记录输出到ElasticSearch中

```
input {
```

```

file {
    path => "/soft/beats/logs/app.log"
    start_position => "beginning"
}
}
filter {
    mutate {
        split => {"message"=>"|"}
    }
}
output {
    elasticsearch {
        hosts => ["127.0.0.1:9200"]
    }
}
}

```

然后在重启我们的logstash

```
./bin/logstash -f ./mogublog-pipeline.conf
```

然后向日志记录中，插入两条数据

```

echo "2019-03-15 21:21:21|ERROR|读取数据出错|参数: id=1002" >> app.log
echo "2019-03-15 21:21:21|ERROR|读取数据出错|参数: id=1002" >> app.log

```

最后就能够看到我们刚刚插入的数据了

The screenshot shows the Elasticsearch Kibana interface. The left sidebar lists various indices, with 'logstash-2020.09.24-000001' selected. The main panel displays a table of documents. The table has columns: _index, _type, _id, _score, host, @timestamp, path, and @version. Two documents are listed, both of type '_doc'. The first document has _id 'aoaiwHQBoKwHwRjx56f_' and the second has _id 'cYaiwHQBoKwHwRjx7qfu'. A modal window on the right shows the JSON source of the selected document, which is an error log entry.

_index	_type	_id	_score	host	@timestamp	path	@version
logstash-2020.09.24-000001	_doc	aoaiwHQBoKwHwRjx56f_	1	ElasticStack	2020-09-24T15:02:15.363Z	/soft/beats/logs/app.log	1
logstash-2020.09.24-000001	_doc	cYaiwHQBoKwHwRjx7qfu	1	ElasticStack	2020-09-24T15:02:17.474Z	/soft/beats/logs/app.log	1

```

{
  "_index": "logstash-2020.09.24-000001",
  "_type": "_doc",
  "_id": "aoaiwHQBoKwHwRjx56f_",
  "_version": 1,
  "_score": 1,
  "_source": {
    "host": "ElasticStack",
    "@timestamp": "2020-09-24T15:02:15.363Z",
    "message": [
      "2019-03-15 21:21:21",
      "ERROR",
      "读取数据出错",
      "参数: id=1002"
    ],
    "path": "/soft/beats/logs/app.log",
    "@version": "1"
  }
}

```