计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2018,27(9):81−86 [doi: 10.15888/j.cnki.csa.006528]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 Filebeat 自动收集 Kubernetes 日志的分析系统①

翟雅荣[1,2], 于金刚[2]

[1](中国科学院大学, 北京 100049)
[2](中国科学院 沈阳计算技术研究所, 沈阳 110168)
通讯作者: 翟雅荣, E-mail: zhaiyarong@126.com

摘　要: Docker 容器产生的日志分散在不同的相互隔离的容器中, 并且容器具有即用即销的特点, 传统的解决方式是将日志文件挂载到宿主机上, 但是容器经常会漂移, 给日志的统一查看带来挑战, 并且传统的 Docker 容器集群日志分析系统存在扩展性弱、效率低下等问题. 本文采用 Kubernetes 实现容器管理、服务发现及调度, 使用 Filebeat 采集容器及宿主机上的日志文件, 并使用 Redis 作为缓存, Logstash 转发, 使用主流的开源日志收集系统 ELK 实现日志的存储、查看、检索. 该系统具有可靠性、可扩展性等特点, 提高运维人员的工作效率.

关键词: 日志收集与分析; Kubernetes; Filebeat; Docker; ELK

引用格式: 翟雅荣,于金刚.基于 Filebeat 自动收集 Kubernetes 日志的分析系统.计算机系统应用,2018,27(9):81–86. http://www.c-s-a.org.cn/1003-3254/6528.html

Analysis System Based on Filebeat Automated Collection of Kubernetes Log

ZHAI Ya-Rong[1,2], YU Jin-Gang[2]

[1](University of Chinese Academy of Sciences, Beijing 100049, China)
[2](Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang 110168, China)

Abstract: The logs generated by Docker containers are scattered in different isolated containers, and the container has the characteristics of “ready to use”. The traditional solution is to mount the log files to the host, but the containers often drift, bringing challenges to the unified view of the log, while the traditional Docker container log analysis system has the problems of weak expansibility and low efficiency. This study uses Kubernetes to implement container management, service discovery and scheduling, uses Filebeat to collect log files on containers and host computers, Redis as a cache, Logstash forwarding, and uses the mainstream open source log collection system ELK to store, view, and retrieve log. The system has the characteristics of real time, reliability and extensibility, and improves the efficiency of operation and maintenance personnel.

Key words: log collection and analysis; Kubernetes; Filebeat; Docker; ELK

日志是记录系统行为的一种方式, 具有非常重要的作用. 日志信息可作为系统排错和性能优化的依据; 通过日志, 可以有效的监控系统的运行状况, 如性能信息、故障检测、入侵检测等; 日志可以用于用户的行为信息分析, 发现潜在商机; 日志可以帮助开发人员找到 bug 的来源, 修复漏洞. 并且随着数据挖掘、大数据等技术的不断发展, 数据也愈发凸显重要, 日志作为数据分析的一大来源, 日志收集分析系统也愈发凸显重要. 但随着互联网、大数据等快速发展, 系统应用越来越复杂, 规模越来越大, 系统产生的日志急剧增加[1], 给日志的收集、存储和分析带来很多挑战.

传统的日志存储在本机磁盘上, 查看日志时需要

① 收稿时间: 2018-01-03; 修改时间: 2018-01-23; 采用时间: 2018-02-27; csa 在线出版时间: 2018-08-16

登陆到宿主机上, 用 grep 等工具进行查看分析, 这种方式的效率极低, 并且随着机器数量的增多, 排查问题所花费的时间随之增加, 并且这种方式很难进行比较复杂的分析, 不能充分利用起日志的价值. 随着 Docker 容器技术的不断发展, Docker 容器技术的应用范围也越来越广. 与传统的主机、虚拟机技术相比, Docker 容器产生的日志分散在不同相互隔离的容器中, 并且事先并不知道容器应用部署在哪一台机器上, 给日志的收集带来一定的挑战, 这也迫切需要一种方式来收集存储日志. 容器具有即用即销的特点, 容器中的存储会随着容器的关闭而被删除, 虽然可以将日志文件挂载到宿主机上, 但是容器会经常漂移, 给日志的统一查看带来挑战. 本文采用 Kubernetes 实现容器管理、服务发现及调度, 使用主流的开源日志收集系统 ELK 实现日志的存储、查看, 并通过缓存技术 Redis 消息队列提供可靠的数据传输, 将分散在不同容器中的日志统一收集存储, 提高运维人员的工作效率.

1 相关技术

1.1 Docker 技术

Docker[2]是一个开源基于 Linux 操作系统虚拟化技术的高级容器引擎, 可以将应用及开发环境打包到一个可移植的容器中. 为了避免启动和维护虚拟机的开销, Docker 使用 Linux 内核资源隔离技术 (如 cgroups、namespace)、联合文件系统 (如 aufs) 等技术实现多个容器在一个 Linux 内核中相互独立[3]. 与传统的虚拟机需要虚拟出一个完整的操作系统相比, Docker 是一种细粒度、轻量级的虚拟化技术, 属于操作系统级虚拟化. 图 1 显示 Docker 容器技术与虚拟机的原理对比.

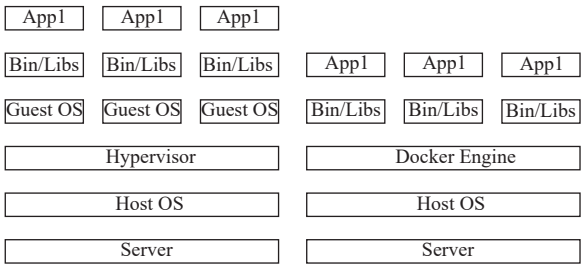

图 1 Docker 容器与虚拟机的原理对比

Docker 是秒级启动, 而虚拟机的启动速度是分钟级; Docker 容器作为一种轻量级的虚拟技术, 其单机支持上千容器, 而虚拟机由于需要虚拟出完整的操作系统, 单机一般只支持几十个; Docker 资源利用率比虚拟机高, 资源占用少[4].

1.2 Kubernetes 技术

Kubernetes (K8s)[5]是 Goolge 开源的软件, 提供了一个完备的分布式系统支撑平台, 是 Docker 生态圈中的重要一员. 它提供了强大的故障发现、负载均衡、自我修复、滚动升级、资源调度管理、自动扩缩容等多个功能, 具备的完善的集群管理的能力. 并且其对现有的平台侵入性较小, 现有平台容易升级改造迁移到 Kubernetes 上.

Kubernetes 集群由 Master 节点和 Node 节点组成, 其中 Master 节点运行集群管理方面的进程 (kube-apiserver、kuber-controller-manager 和 kube-scheduler), 负责实现整个集群的 Pod 调度、弹性伸缩、安全控制、资源管理等. Node 节点是集群的工作节点, 主要负责运行真正的应用程序, 通过运行 kublete、kube-proxy 服务进程, 来负责 Pod 的创建、启动、监控等, kube-proxy 还提供了负载均衡. 与传统的容器相比, Pod 是管理的最小运行单元, 而并非容器. 图 2 显示 Kubernetes 的系统架构图.

1.3 相关研究

目前, 针对 Kubernetes 的日志收集提出了多种方案. Kubernetes 官方推荐采用的方案是 Flunetd + Elasticsearch + Kibana, 通过使用 DaemonSet 的方式在每一个 Node 节点上启动一个 Flunetd 来收集日志文件 (包括/var/log、/var/lib/docker/containers), 常见的收集器还有 Filebeat、Logstash; 陈坚娟等人[6]提出 Logstash agent + Kafka + Logstash + Elasticsearch + Kibana 结构, 并将容器产生的日志挂载到宿主机的指定目录上, 通过 Node 上启动的 Logstash agent 来收集指定目录及/var/lib/docker 目录下的日志; 罗东锋等人[7]提出了基于 Docker 的大规模日志采集与分析系统, 主要使用 Flunetd + Kafka + Flunetd + Elasticsearch + Kibana 实现大规模日志采集; 周德永等人[8]提出了在 Node 节点上部署 Logspout 收集日志, 并 Logstash Shipper 转发送容器日志, 后转发给 Redis 做缓冲, 再有 Logstash 转发至 Elasticsearch 进行存储, Kibana 进行可视化.

Filebeat 具有占用内存较低, 性价比较高的特点, 本文采用 Filebeat 作为日志文件的采集器, 并针对两种

输出方式的日志文件采用不同的收集方式, 输出到控制台的日志文件, 这类文件主要保存在/var/lib/docker/containers/目录下, /var/log/container 软链接到该目录下, 该日志文件主要通过在 Node 节点上部署 Filebeat 进行收集; 未输出到控制台的日志文件, 这类文件主要根据业务需求保存在不同的目录下, 并且日志文件的格式也不尽相同, 此类日志文件通过在每一个 Pod 节点部署 Filebeat 来进行日志收集, 此方法可解决传统日志文件通过挂载到宿主机上进行收集时需要统一日志文件收集的规则、目录和输出方式的缺点.

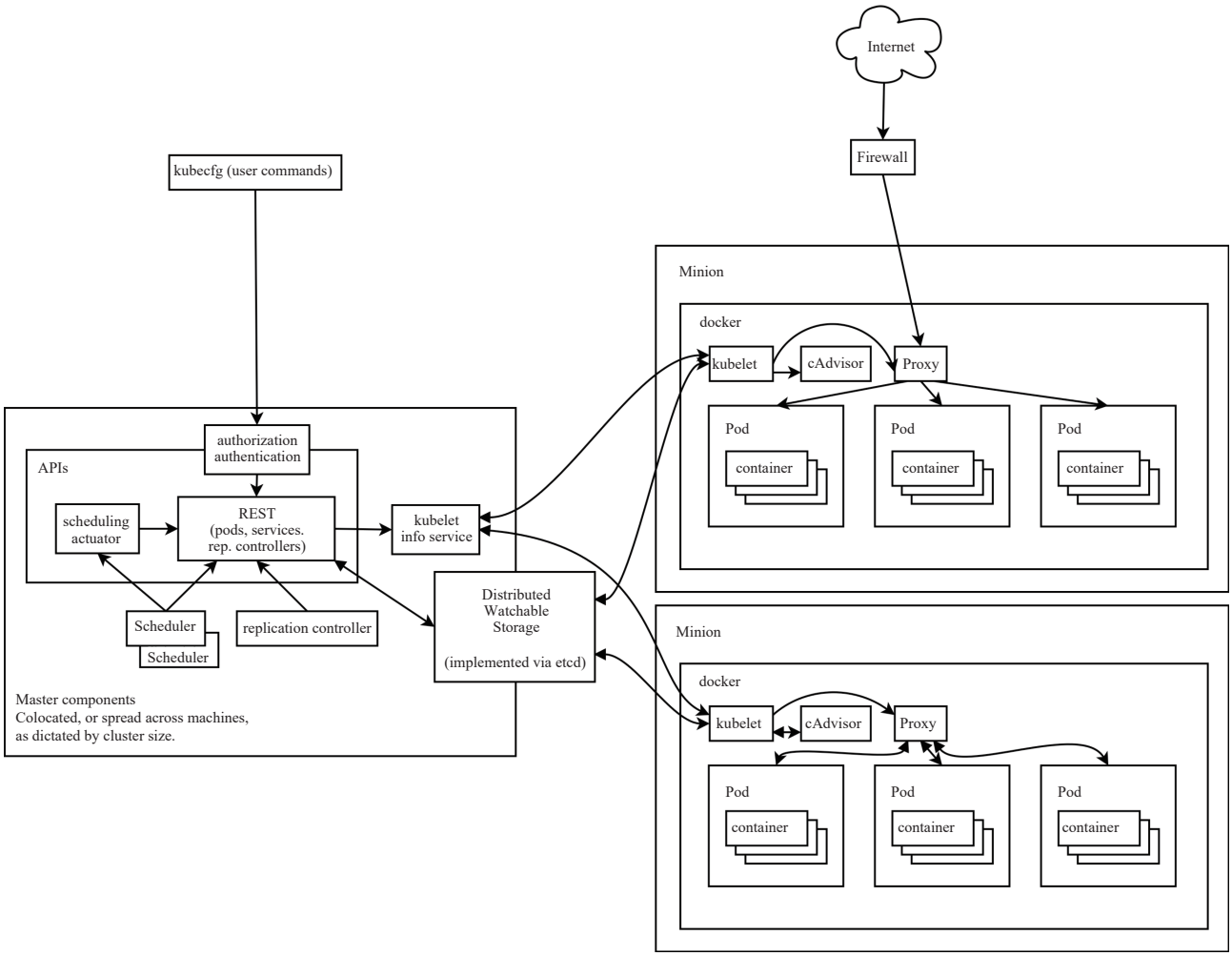

图 2 Kubernetes 系统架构

2 系统架构

该系统的整个架构 (如图 3) 从左到右共分为 5 层, 第 1 层数据采集层, 主要负责采集业务集群中的日志, 并将日志发送给 Redis 服务中, 采用的日志收集器为 Filebeat; 第 2 层为数据缓存层, 该层使用 Redis 作为缓存; 第 3 层为数据转发层, 该层主要负责去 Redis 集群拉数据, 转发到 Elasticsearch; 第 4 层为数据存储层, 该层使用 Elasticsearch 把收到的数据进行存储, 写入磁盘; 第 5 层主要使用 Kibana 处理数据检索请求, 并进行数据展示[7].

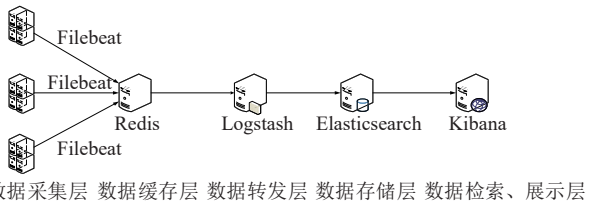

图 3 日志分析系统架构图

2.1 数据采集层

数据采集层主要负责采集各个服务器上不同的文

件, 进行预处理后将数据传送到数据缓存层. 目前主流的采集器主要有 Logstash、Fluentd、Filebeat, 本文采用 Filebeat 作为主要的采集器, Filebeat 相比与 Logstash、Fluentd 的内存使用较低, 性价比较高.

表 1 Logstash、Fluentd、Filebeat 对比

	Logstash	Fluentd	Filebeat
通用日志解析	支持 grok(基于正则表达式) 解析	支持正则表达式解析	支持正则表达式解析
数据发送压缩	插件支持	插件支持	插件支持
数据过滤	支持	支持	支持
运行环境	JRuby 实现, 依赖 JVM	CRuby、C 支持, 依赖 Ruby 环境	Go 语言开发
内存使用	较高	较低	较低
性价比	较低	较低	较高

本文采用 Kubernetes 来实现容器管理、服务发现及调度, 使用 Filebeat 作为采集器来采集日志文件. 日志文件按照输出的不同可以分为两种: 1) 输出到控制台的日志文件. 这类文件主要保存在/var/log/containers/目录下; 2) 应用日志 (未输出到控制台的日志) 输出到指定文件下的文件. 本文针对两种不同的日志文件, 使用不同的方式. 如图 4 为日志采集层.

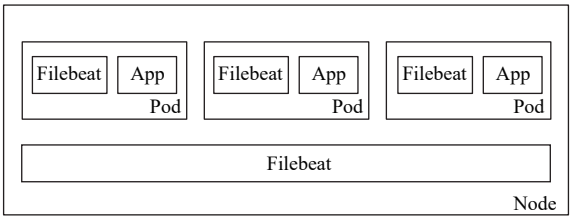

图 4 日志采集层

输出到控制台的日志文件, 这类文件都会保存在/var/lib/docker/containers/目录下, 以*-json.log 的命名, /var/log/containers 软链接到该目录下. 该类日志文件主要使用每个 Node 节点上个运行的一个 Filebeat 容器进行日志采集, 然后汇总到 Redis 缓存中. 本文采用以 DaemonSet 的方式将 Filebeat 作为 Kubernetes 的一种资源对象运行在 Node 节点上.

针对未输出到控制台的日志文件, 这些文件主要存储在容器中的指定目录下, 由于不同应用产生的日志文件格式不同, 位置不同, 如果采用将 Pod 的日志挂载到宿主机上, 进行日志收集, 需要统一日志收集规则、目录和输出方式, 而各个文件各有不同, 同时为了高扩展性、方便维护和升级, 本文针对这类文件采用以每个 Pod 为单位进行日志收集. 用户也可根据实际需求, 来决定是否需要在 Pod 中使用 Filebeat.

2.2 数据缓存层

为了防止发送和接收速率不同而造成数据丢失问题, 在数据采集层和数据转发层之间搭建了消息队列, 起到缓存作用. Redis[9]是一个开源、高性能、基于内存的键值对数据库, 支持数据持久化和订阅发布机制等高级特性, 能够满足本系统的要求. 为提高系统的性能和可靠性, 该层使用 Redis 消息队列作为缓存.

2.3 数据转发层

数据转发层作为 Redis 队列的消费者, 从 Redis 队列中拉取消息并转发到 Elasticsearch 数据存储层进行处理. 本文采用 Logstash 实现日志的拉取、处理、转发. 由于日志中存在字段不完整、格式不一致或者噪声数据, 为了降低这些对数据存储带来的冲击, 同时提高数据的质量, 在数据转发层需要对数据进行预处理, 使数据统一格式, 过滤掉噪声数据等.

输出到控制台的的日志文件, /var/log/containers/*-*-*_*_*.log 代表的命名规则为: podName_namespace_deploymentName-containerID.log, 针对日志文件名称解析出 deploymentName、podName、containerID、namespace 字段, 以下是 Logstash 部分配置信息:

```
filter{
……
if "stdoutLog" in [tags] {
ruby{
code => "
array=event.get('source').split('/')[4].split('_')
event.set('podName', array[0])
event.set('namespace', array[1])
index=array[2].rindex('-') event.set('containerID', array[2][index+1, index+13])
event.set('deployment', array[2][0, index])
```

```
"
}
}
……
}
```

2.4 数据存储层

数据存储层使用 Elasticsearch 集群, 数据转发层将日志逐条插入到数据存储层的 Elasticsearch 集群中. Elasticsearch[10]是一个实时分布式搜索和分析引擎, 基于 Apache Lucene(TM) 的开源搜索引擎, 使用 Lucene 作为实现所有索引和搜索的功能, 它通过简单的 RESTful API 来隐藏 Lucene 的复杂性, 使全文搜索变得简单.

2.5 数据检索和展示层

Kibana 是开源的数据分析和图像化展示的平台. Kibana 基于 Apache License2.0 开源协议, 可以对 Elasticsearch 索引中存储的数据进行搜索、分析、查看, 可以进行数据分析和统计, 并且可以绘制各种图表展示结果. Kibana 支持布尔运算符、通配符和字段筛选进行模糊匹配, 并且其提供了 Web 界面, 方便用户查看结果.

3 实验测试

本文重点验证系统整体的有效性. 通过部署在同一局域网的 5 台 PC 机搭建集群进行测试. 每台 PC 机的硬件换件为主频 3.1 GHz 的四核 CPU、内存 4 GB、磁盘 200 GB, 软件环境为 CentOS 7.2(64 bit)、Elasticsearch 6.0.0、Kibana 6.0.0、Filebeat 6.0.0、Redis、Kubernetes 1.6. 其中 1 台作为 Kubernetes 的 Master 节点, 3 台作为 Kubernetes 的 Node 节点, Kubernetes 的集群部署信息如表 2. 图 5、图 6 为实验结果.

表 2 Kubernetes 集群部署信息

IP 地址	安装的服务
192.168.139.220	etcd、registry、docker 、kube-apiserver、kube-controller-manager、kube-scheduler
192.168.139.141	kube-proxy、kubelet、etcd、flannel、docker、Filebeat
192.168.139.142	kube-proxy、kubelet、etcd、flannel、docker、Filebeat
192.168.139.143	kube-proxy、kubelet、etcd、flannel、docker、Filebeat
192.168.139.144	nfs

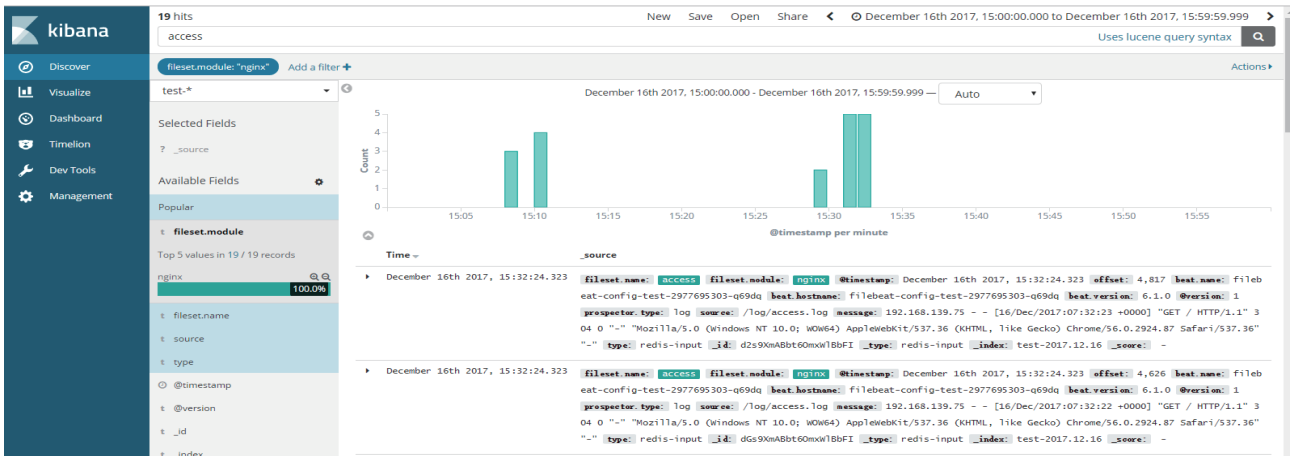

图 5 未输出到控制台日志

4 结论

本文将零散的日志统一收集存储, 并采用 Kubernetes 实现容器管理、服务发现及调度, 使用主流的开源日志收集系统 ELK 实现日志的存储、查看, 并通过缓存技术 Redis 消息队列提供可靠的数据传输, 提高运维人员的工作效率. 本系统可实现实时高效收集日志、处理分析, 并进行可视化展示. 由于使用 Kubernetes 作为容器的管理工具, 可实现快速部署、自动扩缩容、资源管理等. 本系统不仅解决了日志的统一收集, 针对不同的日志文件采用不同的收集策略, 用户灵活配置, 同

时也解决了容器的即用即销造成的漂移问题. 下一步工作将对采集到的日志进行挖掘, 发现有价值的信息.

图 6　输出到控制台日志

参考文献

1 宋密, 王劲松. 基于 Flume 的网络安全可视化系统. 天津理工大学学报, 2015, 31(2): 38–42. [doi: 10.3969/j.issn.1673-095X.2015.02.009]

2 李明, 郭洋, 蒋明. 基于 Docker 的虚拟化技术研究. 中国新通信, 2017, 19(9): 73–74. [doi: 10.3969/j.issn.1673-4866.2017.09.062]

3 刘思尧, 李强, 李斌. 基于 Docker 技术的容器隔离性研究. 软件, 2015, 36(4): 110–113. [doi: 10.3969/j.issn.1003-6970.2015.04.025]

4 陈清金, 陈存香, 张岩. Docker 技术实现分析. 信息通信技术, 2015, 9(2): 37–40.

5 Burns B, Grant B, Oppenheimer D, *et al*. Borg, omega, and kubernetes. Queue, 2016, 14(1): 70–93.

6 陈建娟, 刘行行. 基于 Kubernetes 的分布式 ELK 日志分析系统. 电子技术与软件工程, 2016, (15): 211–212, 214.

7 罗东锋, 李芳, 郝汪洋, 等. 基于 Docker 的大规模日志采集与分析系统. 计算机系统应用, 2017, 26(10): 82–88. [doi: 10.15888/j.cnki.csa.005997]

8 周德永, 王瑞刚, 梁小江. 基于 ELK 自动化收集 Docker 容器日志的分析系统. 电子设计工程, 2017, 25(19): 50–55. [doi: 10.3969/j.issn.1674-6236.2017.19.013]

9 Carlson JL. Redis in action. Shelter Island, NY: Manning Publications Company, 2013.

10 Gormley C, Tong Z. Elasticsearch: The definitive guide: A distributed real-time search and analytics engine. Sebastopol, CA: O'Reilly Media, Inc., 2015.