基于云原生架构的企业级Elasticsearch平台研究

门玉森,韩 超,邓 罡,李 睿,李建林,曹铭轩,吕祷旋,杨 元

(中国民航信息网络股份有限公司,北京 100318)

摘 要: 为解决Elasticsearch在企业级应用场景下,基础资源利用率与运维效率低、容器的数据持久化及资源需求多样性等方面问题,基于Ceph和Kubernetes构建了云原生的企业级Elasticsearch平台。对Elasticsearch平台的架构高可用性、数据持久化、容器化部署及资源标准化交付进行关键设计,对架构的优势进行定性分析,并与传统部署架构进行性能方面的定量对比测试。实验结果表明,该平台在架构设计方面优势明显,在性能方面表现良好,在企业级应用中具有较好的运维与经济效益。

关键词: Elasticsearch;云原生;分布式存储;分布式计算;容器化

DOI: 10. 11907/rjdk. 231069 **开放科学(资源服务)标识码(OSID):**

中图分类号: TP391.3 **文献标识码:** A **文章编号:** 1672-7800(2024)007-0096-08

Enterprise-grade Elasticsearch Platform Research Based on Cloud-Native Architecture

MEN Yusen, HAN Chao, DENG Gang, LI Rui, LI Jianlin, CAO Mingxuan, LYU Yini, YANG Yuan

(*TravelSky Technology Limited*, *Beijing* 100318, *China*)

Abstract: In order to solve the problems of Elasticsearch in enterprise-level application scenarios, such as low utilization and O&M efficiency, data persistence of containers and diversity of resource requirements, an enterprise-grade and cloud-native Elasticsearch platform is built based on Ceph and Kubernetes. The architectural high availability, data persistence, containerized deployment and standardized resource delivery of the Elasticsearch platform are designed crucially. The advantages of the architecture are qualitatively analyzed and the performance is quantitatively tested with the traditional deployment architecture. The experimental results show that the platform has obvious advantages in architecture design, good performance, good O&M and economic benefits in enterprise-level applications.

Key Words: Elasticsearch; cloud-native; distributed storage; distributed computing; containerization

0 引言

Elasticsearch(ES)是一个建立在开源搜索引擎Apache Lucene™基础上的(近)实时分布式搜索和分析引擎,基于Lucene倒排索引技术实现搜索功能,而且通过标准的RESTful API隐藏了Lucene的复杂性,进行索引数据的增删改查操作,从而让全文搜索变得简单[1]。

ES主要应用于日志的存储、搜索分析、监控告警等场景,如物联网数据采集存储[2]、农业地理信息数据全文搜索[3]、遥感数据检索[4]、临床知识库构建与病案检索[5]及数据中心运维监控[6]等。在企业级应用中,随着业务量的增加和存储规模的日益增长,ES资源需求的多样性与基础设施资源不匹配且交付效率低之间的矛盾,以及逐步扩大

收稿日期: 2023-07-13

扫描二维码阅读全文:

作者简介: 门玉森(1991-),男,硕士,CCF会员,中国民航信息网络股份有限公司工程师,研究方向为模式识别与人工智能、大数据;韩超(1981-),男,硕士,中国民航信息网络股份有限公司工程师,研究方向为大数据、云计算、灾备、系统运维;邓罡(1979-),男,硕士,中国民航信息网络股份有限公司高级工程师,研究方向为大数据、云计算、灾备,存储技术;李睿(1994-),女,硕士,中国民航信息网络股份有限公司工程师,研究方向为机器学习、运维大数据分析;李建林(1982-),男,硕士,中国民航信息网络股份有限公司高级工程师,研究方向为大数据、工程测试;曹铭轩(1992-),女,硕士,中国民航信息网络股份有限公司工程师,研究方向为OLAP数据库;吕祷旋(1989-),女,硕士,中国民航信息网络股份有限公司工程师,研究方向为Hadoop技术、数据仓库;杨元(1991-),女,中国民航信息网络股份有限公司工程师,研究方向为流式处理、消息系统、微服务框架。本文通讯作者:门玉森。

的集群规模与人工运维效率低下之间的矛盾愈发突出,仅靠物理机或虚拟机这种传统的独立集群部署方式已无法满足业务场景需求。

存算分离的分层架构设计思想在ES容器化平台上的应用,能够实现对存储与计算资源的分别扩展,解决传统部署方式中存算资源不匹配的问题,提高存算资源利用率[7]。Ceph是一个高性能、高可用、高可扩展的分布式存储系统,能够进行数据的自动重均衡、自动恢复,不存在传统的单点故障问题,方便水平扩展[8],已作为高可靠的数据持久化解决方案在虚拟化[9]、云平台后端存储[10]及大数据量的视频图像存储[11]等业务场景下得到广泛应用。Ceph的块存储能力可以实现存储直接作为磁盘挂载,并且内置了容灾机制,其可靠性、可扩展及使用的便捷性非常适合作为ES的后端存储。

有了分布式存储,存算分离的另一大重点是分布式计算,容器技术的持续部署、弹性伸缩、环境标准化、隔离等特性,能够解决企业级ES平台在资源交付、资源统一管理及运营维护等方面的问题。一种ES容器化的实现途径是利用Docker容器技术,在运行容器前先编写Docker File,通过DockerFile生成镜像,然后运行Docker容器,适用于业务规模不大、容器数量较小的场景,显然无法满足企业级应用需求[12]。随着ES容器数量越来越多,Docker将面临如何协调、调度容器,如何进行容器监控和批量重启,如何保证在升级应用程序时不会中断服务等问题。解决这些问题需要用到容器编排技术,如Kubernetes、Mesos、Swarm,对计算资源进行更高层次的抽象,通过将ES容器进行细致的组合,将最终的集群服务交给用户,其核心特点是能够自主地管理容器以保证云平台中的容器按照用户所期望的状态稳定运行[13]。Kubernetes(以下简称K8S)作为新型分布式计算与资源调度框架,在云环境并行计算[14]、工业互联网资源调度[15]和大数据处理[16]等领域均有研究及应用。

针对ES服务在企业级应用中存在的资源交付与运维效率低下、容器的数据持久化及资源需求等方面的问题,基于Ceph分布式存储和K8S容器编排管理工具构建了云原生架构的企业级ES平台。该架构设计方法能够有效解决企业级业务场景多样性的需求,解决ES传统部署方式面临的资源交付效率及基础资源利用率低等问题,能够实现在容器化平台上数据的持久化及资源统一管理,提升运维效率。该方法已在企业业务场景下部署实施并投产应用,经测试验证,其性能和稳定性良好,能够满足企业级应用安全生产需求。

1　基于云原生架构的Elasticsearch平台架构设计

云原生表示平台服务从设计之初即考虑到云上环境,原生为云而设计,充分利用和发挥云平台弹性和分布式的优势[17]。主要包括容器、微服务、DevOps和持续交付4方面内容,即采用开源堆栈(Docker、Ceph及K8S)进行容器化,基于微服务架构提高灵活性和可维护性,借助敏捷方法和DevOps实现开发运维自动化,利用云平台特性实现弹性伸缩、动态调度和资源利用率提升等[18]。本文架构着重对容器化平台的高可用性、数据持久化、容器化调度与部署实施进行关键设计,以微服务的架构思想为指导,实现ES集群服务独立部署、独立交付、配置更新、弹性扩展等,并对资源服务进行规范化、标准化设计,以满足自动化发布、持续交付需求,实现生产环境快速部署和开发运维的协作(DevOps),进而达到应用持续迭代更新和开发运维自动化的目的。

1.1　ES容器化平台的高可用架构总体设计

基于Ceph和K8S构建ES容器化平台,利用Ceph的块存储能力部署存储资源池,提供存储盘挂载,通过K8S管理容器计算资源,以Deployment和Service的方式提供ES集群服务。相较于普通应用场景,企业级应用关注的一个重点技术指标是服务的可用性,当发生故障甚至灾难时,保证服务可用,不影响企业业务正常运行。因此,在企业级业务场景下,除功能性能力外,还需充分考虑架构的冗余性,以满足安全生产中的高可用需求。

本文创新性地将可用区(Availability Zone,AZ)[19]的概念和ElasticSearch的Zone机制相结合,进行了ES容器化平台跨AZ部署设计,具体架构如图1所示。考虑ES集群的副本机制,设计3个可用区AZ,每个AZ都由一组存储和计算资源(Ceph/K8S集群)组成,AZ之间是物理隔离,某个AZ故障,不会影响其它AZ,但AZ之间是连通的,且网络耗时低。这样可以利用AZ的隔离性和ES数据副本的Zone分配机制[20],对ES集群服务进行跨AZ部署,在3个AZ冗余下,能够充分保证ES集群服务的高可用。

以图1中的Elasticsearch集群1为例进行说明,分别设置集群中1/3 ES实例的节点属性为ZoneA、ZoneB和ZoneC,当该集群进行数据副本分配时(默认设置3副本),任一索引的任一主分片(Primary Shard)和副本分片(Replica Shard)将分别落到3个AZ可用区,当AZ故障,只要保证至少有1个AZ可用,就能保证集群中每份数据至少有1个副本数据可用,也就不会造成上层ES集群上索引数据丢失引起的服务不可用问题。当然,要实现ES集群服务的高可用,除数据节点(Data Node)的高可用外,还需保证每个AZ可用区至少有一个ES的主节点(Master Node)和协调节点(Coordinating Node)。为提高资源利用率,可以考虑节点角色复用。

1.2　基于Ceph的ES平台数据持久化关键设计

ElasticSearch是有状态的服务,涉及数据存储,而数据持久化问题一直是应用容器化部署的重点和难点问题,本文基于Ceph块存储(RBD)实现了与K8S PV/PVC集成,解决了数据持久化问题,实现了在ElasticSearch实例节点离

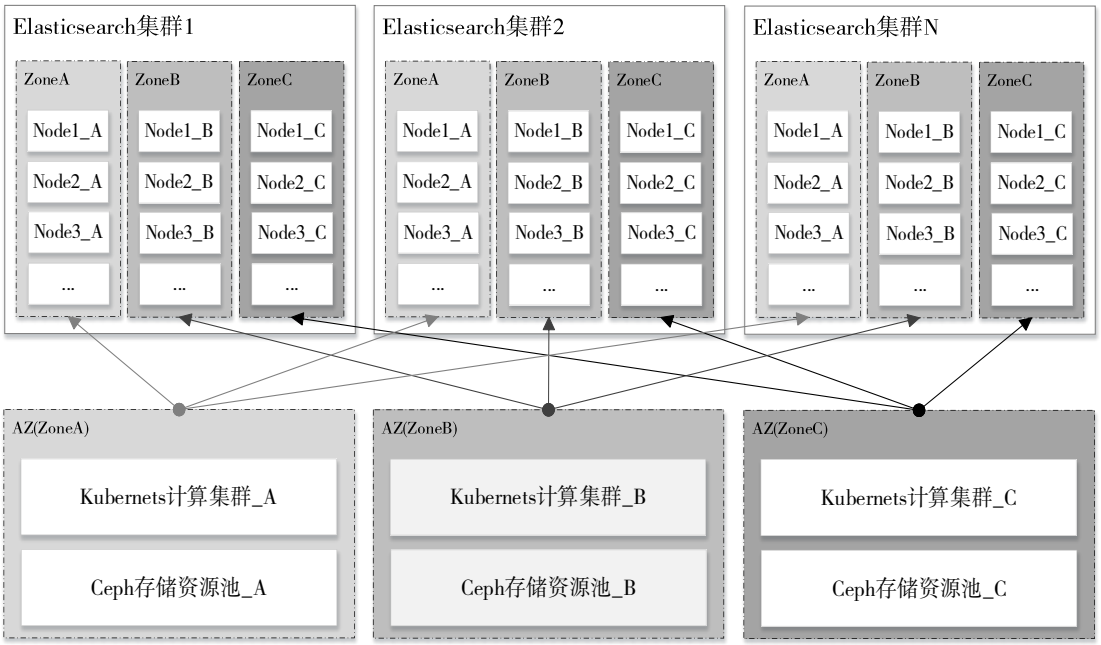

Fig. 1 High availability architecture of Easticsearch container platform based on Ceph and Kubernetes

图 1 基于 Ceph 与 Kubernetes 的 Elasticsearch 容器化平台高可用架构

线情况下，数据不丢失。

Ceph 是一个高性能、高可用和高可扩展性的分布式存储系统，支持对象存储、块设备存储和文件系统服务。在虚拟化领域，较常用到的是 Ceph 的块设备存储。Ceph 摒弃了传统集中式存储中的元数据寻址方案，充分利用了存储节点上的计算能力，在存储每一份数据时，均通过计算得出该数据存储位置，采用 CRUSH（Controlled Replication Under Scalable Hashing）算法、HASH 环等方法，使数据分布均衡，并行度高，不存在传统的单点故障问题[21]。Ceph 具有去中心化的特性，能够支持上千个存储节点的规模，且随着规模的扩大，性能并不会受到影响，支持 TB 到 PB 级的数据，能够满足 ES 在海量日志存储中大数据量的需求。

1.2.1 Ceph 分布式存储架构

Ceph 在云环境中通常作为可扩展的后端存储以提高数据转发效率，其分布式架构如图 2 所示[22]。

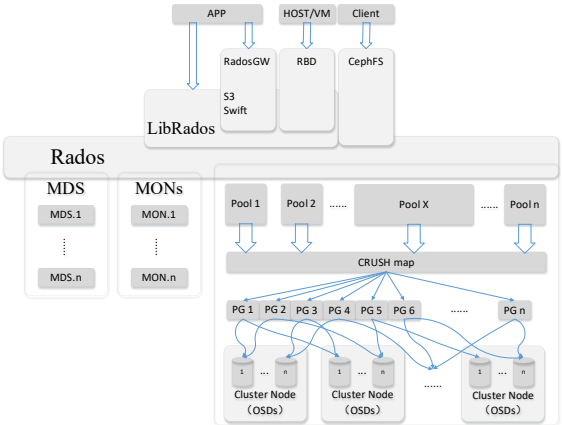

Fig. 2 Distributed architecture of Ceph

图 2 Ceph 分布式架构

（1）Rados（Reliable Autonomic Distributed Object Store）是 Ceph 存储集群的基础。在 Ceph 中，通过 Rados 对象存储系统对外提供服务，所有数据都以对象的形式存储以确保数据始终保持一致。

（2）Ceph 依赖 LibRados 库，对外提供直接访问 Rados、块存储接口（RBD）、对象存储网关接口（RadosGW）和文件系统接口（CephFS），供应用程序访问。

（3）Ceph 元数据 MDS（Metadata Server），主要保存的是 Ceph 文件系统的元数据，Ceph 的块存储和对象存储无需 MDS。

（4）Mons（Monitors 管理服务）负责维护集群状态的多种映射，同时提供认证和日志记录服务，包括 monitor 节点端到端的信息、当前版本信息及最新更改信息等。

（5）在 Pool 存储池中包含了很多 PG 归置组（Placement Group），然后通过 CRUSH 算法将 PG 中的数据存储到各 OSD（Object Storage Device）组中，从而消除了中心节点，寻址流程依次为 File->Objects 映射（oid）、Object->PG 映射（pgid）和 PG->OSD 映射（CRUSH），实现数据存储组织和位置映射。

（6）OSD 为对象存储守护程序，但是它并非针对对象存储，而是物理磁盘驱动器，将数据以对象的形式存储到集群中每个节点的物理磁盘上。Ceph 数据存储的绝大多数由 OSD Daemon 进程实现，包括存储数据读写和处理数据复制、恢复、回填（Backfilling）、再平衡等。此外，OSD 还对其它 OSD 进行心跳检测，检测结果汇报给 Monitors。OSD Map 机制负责执行 PG 在 OSD 上的分布和监控，其与 CRUSH 算法共同构成了 Ceph 分布式架构的基石。

1.2.2 Ceph RBD 与 K8S PV/PVC 集成

容器的生命周期是短暂的，为避免由于容器的销毁或

重建导致容器中的数据丢失，并满足不同容器间持久性数据共享需求，需采用数据持久化的方案解决。K8S 的 Volume 提供了较好的数据持久化方案，但在可管理性上尚有不足，存在应用和系统管理人员职责耦合问题。在生产环境中，考虑到效率和安全性，在 ES 容器化平台存储方案设计中，利用 K8S 的 PersistentVolume（PV）和 PersistentVolumeClaim（PVC）的解决方案[23]，将 Ceph 的块设备存储 RBD 与 K8S PV/PVC 对接，将后端块存储以磁盘的形式挂载到 Pod 上，考虑平台存储侧的高可用，部署 3 套存储 Pool，分别提供块存储资源，如图 3 所示。

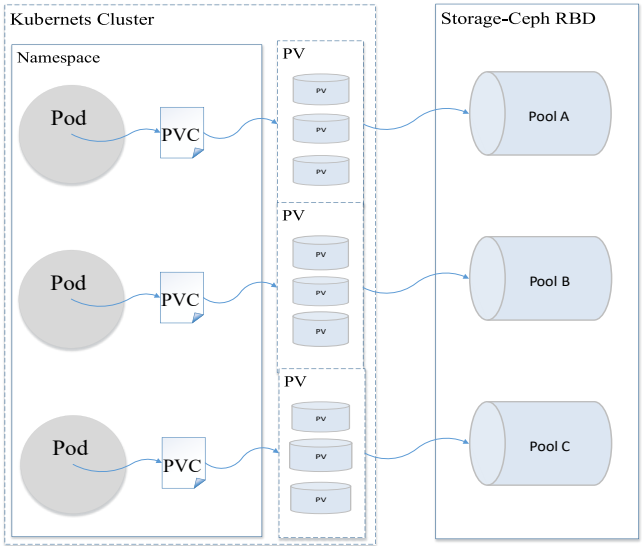

Fig. 3　Storage scheme of ES container platform

图 3　ES 容器化平台存储方案

PV 为分布式存储系统 Ceph 中的一块存储空间，由存储管理员创建和维护，与 Volume 一样，PV 具有持久性，生命周期独立于 Pod。PVC 是对 PV 的申请（Claim），由应用人员（ES 管理员）创建和维护，用于为 Pod 分配存储资源，并指明存储资源的容量大小和访问模式（比如只读）等信息，Kubernetes 会查找并提供满足条件的 PV，而无需关心真正的空间从哪里分配，如何访问等底层细节信息。

1.3　基于 K8S 的 ES 平台容器化部署关键设计

Kubernetes 是基于容器的集群编排引擎，能在实体机或虚拟机集群上调度和运行程序容器，从以主机为中心的架构跃至以容器为中心的架构，具备可扩展、弹性、自愈、在线升级、服务发现等特性[24]。

1.3.1　Kubernetes 的整体架构

由从宏观上来看 Kubernetes 的整体架构，包括 Master、Node 以及 Etcd，具体如图 4 所示。

Master：即主节点，负责控制整个 Kubernetes 集群，包括 Api Server、Scheduler、Controller 等组成部分，它们都需要与 Etcd 进行交互以存储数据。

Api Server：主要提供资源操作的统一入口，能够屏蔽与 Etcd 的直接交互，包括安全、注册与发现等功能。

Scheduler：负责按照一定的调度规则将 Pod 调度到

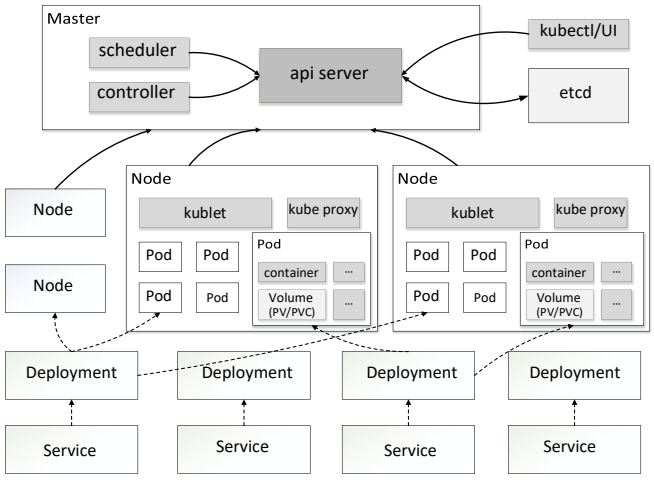

Fig. 4　Overall architecture of Kubernetes

图 4　Kubernetes 的整体架构

Node 上。

Controller：资源控制中心，确保资源处于预期工作状态。

Node：工作节点，为整个集群提供计算力，是容器真正运行的地方，包括 Pod、kubelet、kube-proxy。

Pod：最小调度及资源单元，是对一组容器的抽象，其包含一个或多个容器，Volume（PV/PVC）用来管理 K8S 的存储，被挂载在 Pod 中一个或者多个容器的指定路径下，在 ES 容器化平台中，后端存储对接的是 Ceph 块存储。

Kubelet：主要工作包括容器生命周期管理、监控、健康检查，以及节点状态定期上报。

Kube-proxy：主要利用 Service 提供集群内部的服务发现和负载均衡，同时监听 Service/Endpoints 变化并刷新负载均衡。

K8S 是通过创建 Deployment 请求进行 Pod 创建及其后续生命周期管理，通过 Service 微服务对外提供服务访问。

Deployment 是在 Pod 基础上更高层次的抽象，能够定义 Pod 的副本数及版本等，通过 Controller 维护 Deployment 中 Pod 的数目并自动恢复失败的 Pod。

Service 为 pod 提供用于外部访问的稳定地址，能够有效避免由于 Pod 众多或者异常重启导致的外部用户访问需频繁更新访问地址的问题。

1.3.2　ES 平台容器化部署关键设计

一个 ES 集群由多个节点构成，节点有角色上的区分，主要分为主节点（Master）、协调节点（Coordinating）和数据节点（Data），其中主节点负责参与选举投票、索引添加及删除、对分片进行分配、收集集群节点状态等；协调节点用于分发请求、收集结果及客户端读写分离设置等；数据节点主要负责对数据进行增、删、改、查、聚合等操作[25]。

在 ES 容器化平台部署设计中，基于 K8S 的 Service 和 Deployment 进行 Pod 创建和管理并构建 ES 集群，同时会根据业务团队或场景，创建不同的 NameSpace 用于资源间的逻辑隔离、配额管控及鉴权操作等[26]。每个 NameSpace 下

也可以根据业务需要创建不同的 ES 集群，平台管理员将按照 NameSpace 对 ES 资源进行管理和集群维护，如图 5 所示。

每组 Service 和 Deployment 对应创建 1 个 ES 服务实例，考虑到业务应用对性能的需求，对 ES 的主要角色进行分别独立部署，并引入 ES 中 Zone 的概念，集群中 ES 实例分别分到 3 个不同的 Zone 内，每个 Zone 对应一套 K8S 集群和后端 Ceph 存储池，显著提高了 ES 平台的可用性[27]。Kibana 是为 ES 设计的开源分析和可视化工具，可以用来搜索、查看存储在 ES 索引中的数据并与之交互。在 ES 容器化设计中，Kibana 同样基于 Service 和 Deployment 并按照 Zone 进行分别部署。

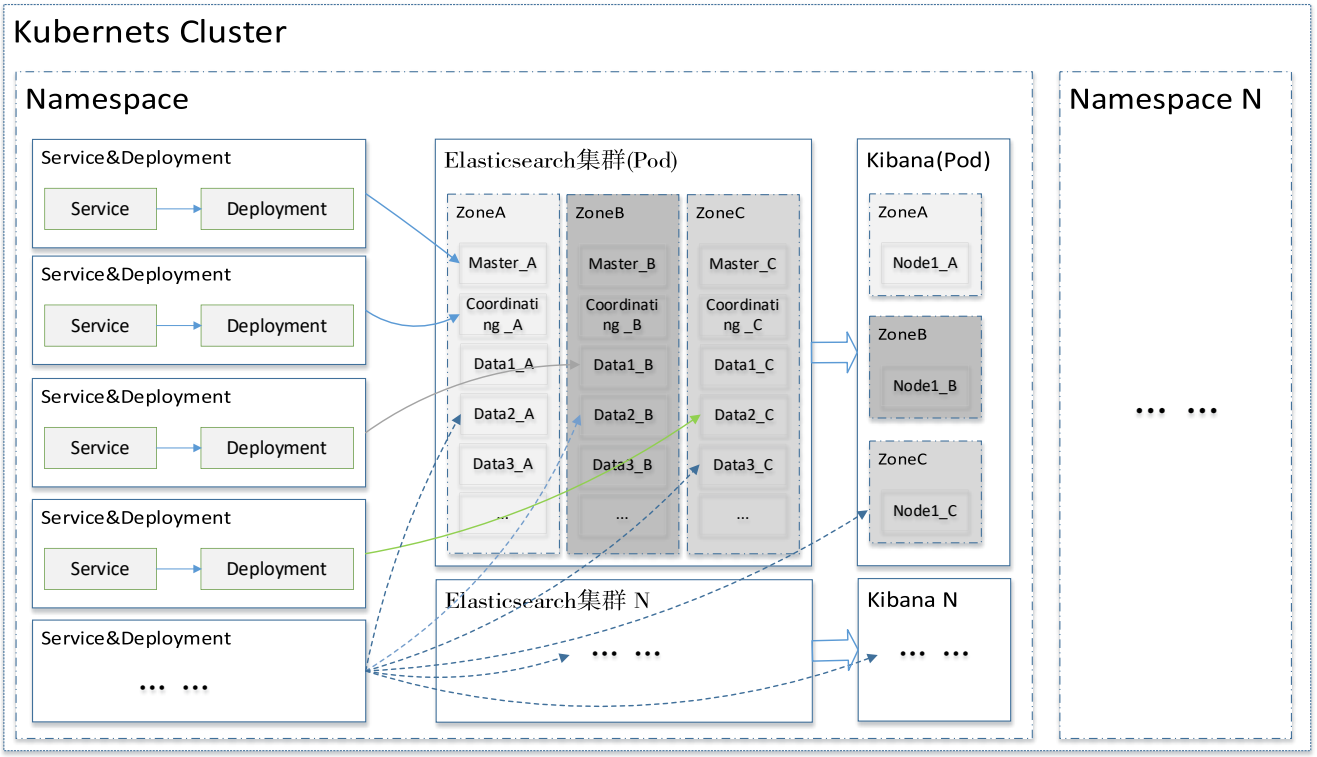

Fig. 5 Deployment design of ES container platform

图 5 ES 的容器化平台部署设计

1.4 ES 容器化资源标准化交付关键设计

面对企业内部不同产品线需求多样性与资源快速交付的矛盾，在企业级 ES 平台交付设计中，为保障应用的资源需求实现自动化快速交付，同时满足平台资源统一管理的运维侧要求，根据角色不同，结合 ES 最佳实践[28]，对 ES 集群节点的存储和计算资源进行规范化配置，具体如表 1 所示。

Table 1 Normalization configuration of ES role instance

表 1 ES 集群角色实例规范化配置

角色实例	计算配置	存储配置
主节点(Master)	8C/8 GB	50 GB
协调节点(Coordinating)	8C/32 GB	50GB
数据节点(Data)	8C/64 GB	50 GB+2*2 TB
Kibana 节点	4C/8 GB	50 GB

ES 实例的 jvm 配置为内存的 50%，并且不超过 32 GB，50 GB 的存储盘主要用于节点安装部署及日志存储，数据节点的数据存储盘为 2 块 2 TB 的存储，存储均依赖 Ceph 后端存储提供。

基于以上标准化节点配置，构建基于容量评估的 ES 容器化资源交付模型，根据用户对 ES 资源的存储需求 n_storages(单位 TB)及 ES 副本配置数 n_replica(大于等于 2，默认为 3)，确定集群各角色节点数，具体如下：

数据节点数：

$$num\_data = \lceil n\_storages/(2*2) \rceil * n\_replica \tag{1}$$

主节点数：

$$num\_master = \begin{cases} 3 & n\_storages <= 100 \\ 5 & n\_storages > 100 \end{cases} \tag{2}$$

协调节点数：

$$num\_coordinating = \begin{cases} 0 & n\_storages < 40 \\ 2 + \lfloor (n\_storages - 40)/20 \rfloor & 40 \leqslant n\_storages < 120 \\ 6 & n\_storages \geqslant 120 \end{cases} \tag{3}$$

举例说明，当应用的ES资源的存储需求为75 TB时，按照2副本评估，根据公式计算得到需要的数据节点数为38个，主节点数为3个，协调节点数为3个。应用提交资源申请后，将即时自动化创建一个44节点的ES标准化容器集群，且各角色的节点将尽可能地平均分配在不同的Zone中，以保证集群的高可用。

2　Elasticsearch容器化架构与传统架构比较

2.1　ES容器化架构优势

相较于物理机/虚拟机的传统部署架构，ES容器化平台更加轻量，在资源利用率、交付效率及运维效率方面具有明显优势。具体表现在以下方面：

（1）基于存算分离的架构，能够对存储和计算资源分别进行独立横向扩展，规避了传统架构下存储计算资源的强绑定关系，解决了存算资源不匹配导致的资源浪费问题，提高了资源利用率。

（2）依赖容器轻量、敏捷及K8S自由编排等特性，能够实现ES集群快速、标准化、模板化交付。此外，K8S提供了丰富的API以进行服务编排，支持资源交付流程全自动化管理，能够实现ES资源T+0的交付，相较于传统架构在资源交付效率方面有质的提升。

（3）基于容器弹性伸缩、统一编排及K8S友好的前端交互等特性，能够实现ES集群资源的弹性伸缩、集群配置统一分发、实例批量操作、快速版本升级等。而在传统架构下，不支持弹性伸缩，要实现配置统一分发和批量操作，需依赖人工开发相关自动化工具，版本升级更是一个繁杂且高风险的过程。因此，在容器化架构下，ES集群管理运维效率极大提升，同时降低了运维的人工与时间成本。

2.2　ES容器化性能分析

本文采用ES的官方压测工具Esrally[29]。将其运行在Docker容器上，管道策略使用benchmark-only的方式，对物理机、虚拟机和容器化这3种架构下的ES集群进行验证测试，开展集群性能方面的比较分析。

3种架构下，本次对比测试实验的ES集群配置信息如下：①集群规模均为6实例节点，其中3节点为ES的独立Master节点，其余3节点为数据（Data）节点；②集群Heap设置相同，Master节点JVM为8 GB，Data节点JVM为31 GB；③集群单个数据节点挂载的存储，配置为4 TB数据盘，由于物理机硬件配置较高，物理机架构使用单机多实例部署的方式，存储使用本地SATA盘；虚拟机架构后台挂载VSAN存储；容器化架构使用Ceph块存储；④各节点的操作系统、JDK等其它集群通用配置均相同。

Table 2　Test data set information of Ggeonames

表2　Geonames测试数据集信息

Track	压缩数据大小	解压数据大小	文档数
Geonames	252 MB	3.3 GB	11 396 505
Http_logs	1.2 GB	31 GB	247 249 096

测试分别使用Geonames和Http_logs测试数据集（Track），数据集信息如表2所示，使用append-no-conflicts测试策略，其中Geonames数据集测试包括index-stats、node-stats、default、term、phrase、scroll等10个性能测试任务（Task），Http_logs数据集测试包括default、term、range、scroll等6个测试任务。

Table 3　Comparison of performance test results under three different deployment architectures in the Geonames dataset

表3　Geonames数据集下3种不同部署架构性能测试结果比较

部署架构 测试任务/性能指标		物理机	虚拟机	容器化	容器化对比物理机	容器化对比虚拟机
index-stats	吞吐量	99.99	46.17	97.87	−2.12	51.70
	延时时间	19.25	905.07	271.29	252.04	−633.78
node-stats	吞吐量	81.52	61.03	61.11	−20.41	0.08
	延时时间	2 371.17	17 101.30	7 620.64	5 249.47	−9 480.70
default	吞吐量	50.04	49.92	49.89	−0.15	−0.03
	延时时间	65.40	17.92	20.09	−45.30	2.18
term	吞吐量	199.93	199.87	199.96	0.03	0.09
	延时时间	7.56	13.03	14.44	6.88	1.41
phrase	吞吐量	199.89	187.11	199.25	−0.64	12.14
	延时时间	39.82	322.82	28.79	−11.03	−294.03
country_agg_cached	吞吐量	98.04	96.95	95.96	−2.08	−0.99
	延时时间	64.98	21.83	12.06	−52.92	−9.77
scroll	吞吐量	25.06	25.02	25.05	−0.01	0.03
	延时时间	393.89	847.98	466.77	72.88	−381.21
large_terms	吞吐量	1.13	0.85	0.88	−0.25	0.03
	延时时间	114 823	202 005	192 179	77 356	−9 826
large_filtered_terms	吞吐量	1.13	0.85	0.88	−0.25	0.03
	延时时间	116 364	203 191	190 950	74 586	−12 241
large_prohibited_terms	吞吐量	1.15	0.85	0.88	−0.27	0.03
	延时时间	112 215	201 896	191 810	79 595	−10 086

实验测试结果分别如表3、表4所示，其中吞吐量为平均值（Mean Throughput），单位为ops/s，延时时间为完成100%测试的总的延时（100th percentile latency），单位为ms。

为便于进行3种不同部署架构下的性能测试结果比较分析，以容器化架构下的性能测试结果为基准，进行归一化处理。平均吞吐量及延时情况对比结果散点图，如图6—图9所示。

两种数据集下的测试实验结果均表明，ES集群在吞吐量（越大越好）和延时方面（越小越好）的表现，容器化架构普遍优于虚拟化架构，稍弱于物理机架构，个别测试任务下容器化架构的延时表现甚至优于物理机架构。

因此，在企业级应用中，推荐物理机架构和容器化架构互补的方式。针对个别大数据量或者性能要求较高的应用场景，使用物理机的交付架构，可以兼顾业务容量需求和性能要求，不足之处在于可能会造成一定的资源浪费和运维成本增加。而对于众多中小数据量需求场景，则适合使用容器化架构的方式交付，既能快速满足业务资源需

Table 4 Comparison of performance test results under three different deployment architectures in the Http_logs dataset

表 4 Http_logs 数据集下 3 种不同部署架构下的性能测试结果比较

部署架构 测试任务/性能指标		物理机	虚拟机	容器化	容器化对比物理机	容器化对比虚拟机
index-append	吞吐量	302 282	168 369	233 317	-68 965	64 948
	延时时间	3 444.76	4 529.81	4 883.83	1 439.07	354.02
default	吞吐量	8.01	8.01	8.01	0	0
	延时时间	10.29	25.63	13.60	3.31	-12.03
term	吞吐量	50.03	46.68	49.67	-0.36	2.99
	延时时间	15.15	26.47	12.48	-2.68	-13.99
range	吞吐量	1.51	1.51	1.51	0	0
	延时时间	45.51	137.96	95.97	50.46	-41.99
hourly_agg	吞吐量	0.20	0.19	0.20	0	0.01
	延时时间	38.32	64.36	26.71	-11.60	-37.64
scroll	吞吐量	25.09	25.03	25.06	-0.03	0.03
	延时时间	809.77	1 428.64	415.49	-394.28	-1 013.15

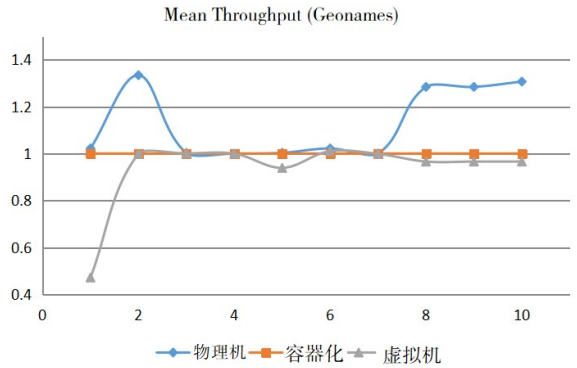

Fig. 6 Comparison of performance results of average throughput under different deployment architectures in Geonames dataset

图 6 不同部署架构下 Geonames 数据集测试平均吞吐量性能结果比较

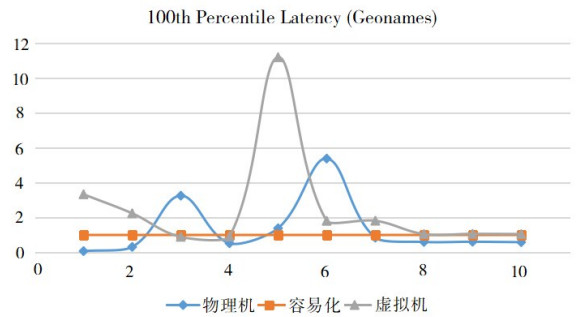

Fig. 7 Comparison of performance results of total delay under different deployment architectures in Geonames dataset

图 7 不同部署架构下 Geonames 数据集测试的总延时性能结果比较

求，又能在资源利用率和运维成本方面达到降本增效的目的。

3 结语

本文阐述了基于 Ceph 与 Kubernetes 的企业级 Elasticsearch 容器化平台总体架构，以及 ES 容器化存储与计算设

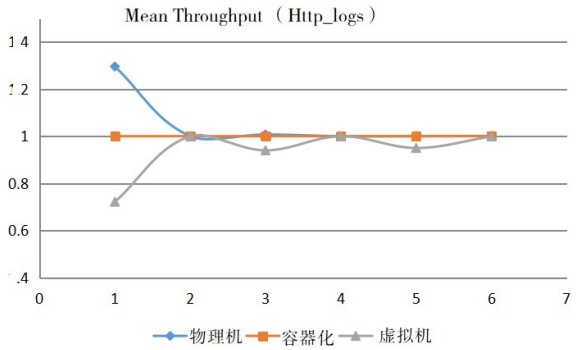

Fig. 8 Comparison of performance results of average throughput under different deployment architectures in Http_logs dataset

图 8 不同部署架构下 Http_logs 数据集测试平均吞吐量性能结果比较

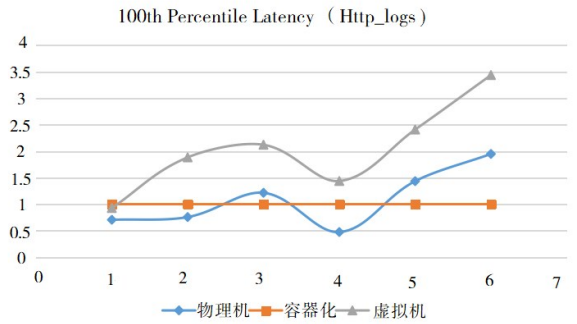

Fig. 9 Comparison of performance results of total delay under different deployment architectures in Http_logs dataset

图 9 不同部署架构下 Http_logs 数据集测试总延时性能结果比较

计及部署实施方案，对该平台架构的优势进行了定性分析，并与传统部署架构作了性能方面的定量对比分析。结果表明，该平台在架构设计方面优势明显，在性能方面表现良好，且平台具有高可用性，在企业级应用中具有较好的运维与经济效益。同时，该架构设计思想在数据库、大数据等领域的应用中也具有一定的借鉴和推广意义。

参考文献：

[1] WANG W, WEI L, LIU W Q, et al. Distributed full-text search system based on ElasticSearch[J]. Electronic Science and Technology, 2018, 31(8):56-59.
王伟，魏乐，刘文清，等. 基于ElasticSearch的分布式全文搜索系统[J]. 电子科技，2018，31(8)：56-59.

[2] QIN L, LIN X J, ZHOU X, et al. Technical solution of data collection and storage based on IOT[J]. Scientific and Technological Innovation, 2021(32):104-107.
秦磊，林希佳，周轩，等. 基于物联网的大数据采集与存储技术解决方案[J]. 科学技术创新，2021(32)：104-107.

[3] LIU J, SUN J Y, CHEN Z C, et al. Design and implementation of big data platform about agricultural geographic information based on ElasticSearch full-text retrieval[J]. Geomatics & Spatial Information Technology, 2021, 44(6):162-164.
刘吉，孙俊英，陈忠超，等. 基于ElasticSearch全文检索的农业地理信

息大数据平台设计与实现[J]. 测绘与空间地理信息,2021,44(6):162-164.

[4] LI F, SONG Y, WEI G Z, et al. Research on massive remote sensing data retrieval technology based on ElasticSearch[J]. Computer and Network, 2021,47(5):57-61.
李峰,宋宴,魏广泽,等. 基于ElasticSearch的海量遥感数据检索技术研究[J]. 计算机与网络,2021,47(5):57-61.

[5] CHENG B, ZHANG X M, RUAN C. Design and implementation of knowledge base and medical record retrieval service platform based on Elasticsearch[J]. Chinese Medical Record,2021,22(3):44-48.
程彪,张晓明,阮晨. 基于Elasticsearch的知识库和病案检索服务平台的设计与实现[J]. 中国病案,2021,22(3):44-48.

[6] PAN C H. Research and practice of multi-level monitoring platform for operation and maintenance of university data center based on ELK[J]. The Chinese Journal of ICT in Education,2020(7):93-96.
潘春华. 基于ELK的高校数据中心运维多层面监控平台研究与实践[J]. 中国教育信息化,2020(7):93-96.

[7] XU Q. Big data storage and calculation separation accelerate enterprise digital transformation [J]. Software and Integrated Circuit, 2020 (9): 98-99.
徐强. 大数据存算分离 加速企业数字化转型[J]. 软件和集成电路,2020(9):98-99.

[8] CHEN X D, PANG S L, ZENG D S, et al. Application of ceph storage system in cloud computing environment[J]. Electronic Technology, 2020, 49(8):40-42.
陈晓丹,庞双龙,曾德生,等. Ceph存储系统在云计算环境中的应用[J]. 电子技术,2020,49(8):40-42.

[9] XIE Y P. Design and application of virtual cloud desktop system based on hyperfusion architecture[J]. Computer and Information Technology, 2022, 30(2):30-32.
谢艺平. 基于超融合架构的虚拟云桌面系统设计及应用[J]. 电脑与信息技术,2022,30(2):30-32.

[10] XI L. Design and implementation of ceph distributed storage system on OpenStack cloud platform[J]. Research on Digital Media, 2021, 38(8): 37-43.
席磊. Ceph分布式存储系统在OpenStack云平台的设计与实现[J]. 数字传媒研究,2021,38(8):37-43.

[11] LIANG Z F, WANG X T, LIU X P. Research of key technologies on hydrological monitoring platform based on big data architecture[J]. Information Technology,2022(6):6-11.
梁志锋,王信堂,刘鑫鹏. 基于大数据架构的水文监测平台关键技术研究[J]. 信息技术,2022(6):6-11.

[12] ANDERSON C. Docker [software engineering] [J]. IEEE Software, 2015,32(3): 102-c3.

[13] ZHU Z Y. Research and implementation of distributed and integrated learning system in cluster environment[D]. Nanjing: Southeast University,2021.
朱泽宇. 集群环境下分布式集成学习动态系统的研究与实现[D]. 南京:东南大学, 2021.

[14] CHEN M J, LI J, ZHAI Z X, et al. Application and exploration of container cloud in seismic data processing[J]. Computer Simulation, 2022, 39(5):81-83.
陈明俊,李靖,翟中霞,等. 容器云在地震资料处理中的应用探索[J]. 计算机仿真,2022,39(5):81-83.

[15] ZHU L, LI J J, LIU Z J, et al. A multi-resource scheduling scheme of Kubernetes for IIoT[J]. Journal of Systems Engineering and Electronics, 2022,33(3):683-692.

[16] QI F, XU N, ZHOU H X, et al. Design of parallel computing test platform based on Kubernetes and its application in water system test[J]. Modern Information Technology,2022,6(12):74-77.
齐飞,许诺,周含笑,等. 基于Kubernetes的并行计算测试平台设计及其在水务系统测试中的应用[J]. 现代信息科技,2022,6(12):74-77.

[17] JIANG Q W, ZHAO Z Y. Construction and research of IPA audit robot ecological platform[J]. The Chinese Institute of Certified Public Accountants,2022(4):43-46.
江其玟,赵梓渊. IPA审计机器人生态平台构建研究[J]. 中国注册会计师,2022(4):43-46.

[18] XU Z L, HUO L S, CAO Y F, et al. Deployment scheme of AI model based on cloud native service grid[J]. Designing Techniques of Posts and Telecommunications,2021(3):32-36.
徐治理,霍龙社,曹云飞,等. 基于云原生服务网格的AI模型部署方案[J]. 邮电设计技术,2021(3):32-36.

[19] PARK Y, PARK Y. Data availability zone for backup system in Cloud computing service[C]//Proceedings of the Korean Institute of Information and Commucation Sciences Conference,2014: 366-369.

[20] SHUKLA P, KUMAR S. Learning elastic stack 7.0: distributed search, analytics, and visualization using elasticsearch, logstash, beats, and kibana[M]. Mumbai: Packt Publishing Ltd,2019.

[21] CUI H J. Research and implementation of optimization method about hadoop distributed file system based on network coding[D]. Beijing: Beijing Jiaotong University,2019.
崔海杰. 基于网络编码的hadoop分布式文件系统优化方法的研究与实现[D]. 北京:北京交通大学,2019.

[22] WEIL S A, BRANDT S A, MILLER E L, et al. Ceph: a scalable, high-performance distributed file system[C]//Proceedings of the 7th Symposium on Operating Systems Design and Implementation,2006: 307-320.

[23] NGUYEN N D, KIM T. Balanced leader distribution algorithm in kubernetes clusters[J]. Sensors,2021,21(3):869.

[24] LUKSA M. Kubernetes in action[M]. New York: Simon and Schuster, 2017.

[25] ZENG Y F. Research and implementation of distributed intelligent search engine based on Elasticsearch[D]. Chongqing: chongqing University, 2016.
曾亚飞. 基于Elasticsearch的分布式智能搜索引擎的研究与实现[D]. 重庆:重庆大学,2016.

[26] SAYFAN G. Mastering kubernetes[M]. Mumbai: Packt Publishing Ltd, 2017.

[27] GORMLEY C, TONG Z. Elasticsearch: the definitive guide: a distributed real-time search and analytics engine[M]. Sebastopol: O'Reilly Media, Inc. ,2015.

[28] HAUGERUD H, SOBHIE M, YAZIDI A. Tuning of elasticsearch configuration: parameter optimization through simultaneous perturbation stochastic approximation[J]. Frontiers in big Data,2022,5:686416.

[29] YAO J H, LI D Y. Research of stress testing on ES cluster for corporate log storage and analysis[J]. China CIO News,2022:141-144.
姚姜虹,李大勇. 企业日志存储分析ES集群压力测试研究[J]. 信息系统工程,2022(3):141-144.

(责任编辑:孙　娟)