Cloud Search Service

Service Overview

Issue 14

Date 2023-05-30

HUAWEI

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

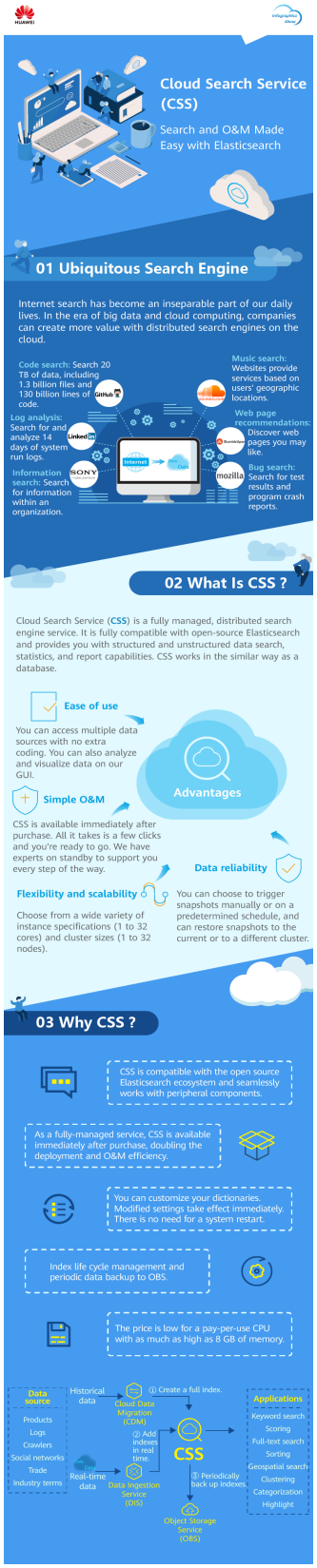
Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process.* For details about this process, visit the following web page:
https://www.huawei.com/en/psirt/vul-response-process
For vulnerability information, enterprise customers can visit the following web page:
https://securitybulletin.huawei.com/enterprise/en/security-advisory

Contents

1 CSS Infographics



2 What Is Cloud Search Service?

CSS

Cloud Search Service (CSS) is a fully hosted distributed search service based on Elasticsearch. You can use it for structured and unstructured data search, and use AI vectors for combine search, statistics, and reports. CSS is a fully managed cloud service of the ELK Stack and is compatible with open-source Elasticsearch, Kibana, and Cerebro.

Elasticsearch is an open-source distributed search engine that can be deployed in standalone or cluster mode. As the heart of the ELK Stack, Elasticsearch clusters support multi-condition search, statistical analysis, and create visualized reports of structured and unstructured text. For details about Elasticsearch, see the **Elasticsearch: The Definitive Guide**.

CSS can be automatically deployed, allowing you to quickly create Elasticsearch clusters. It provides the search engine optimization practices and does not require your O&M. Additionally, it has a robust monitoring system to present you key metrics, including clusters and query performance so that you can focus on the business logic.

Functions

- Compatible with Elasticsearch

  Freely use native Elasticsearch APIs and other software in the ecosystem, such as Beats and Kibana.
- Support various data sources

  A few simple configurations can allow you to smoothly connect to multiple data sources, such as FTP, OBS, HBase, and Kafka. No extra coding is required.
- One-click operation

  One-click cluster application, capacity expansion, and restart from small-scale testing to large-scale rollout
- Flexible dictionary management

  You can custom your dictionaries. Modified settings take effect immediately without system restart.
- User-defined snapshot policies

  Trigger backup snapshots manually or configure an automated schedule.

Access Mode

You can access the public cloud platform with HTTPS-based application programming interfaces (APIs) or from a web-based console.

- Using APIs

  If you need to integrate CSS into a third-party system for secondary development, you can use API calls to access CSS. For details, see the ***Cloud Search Service API Reference***.

- Web-based console

  If no secondary development is involved, the CSS management console is a convenient way to access CSS. If you have registered with the public cloud, log in to the management console and search for **Cloud Search Service** in the service list. If you do not sign up for the public cloud, click **Register**. In the displayed window, specify your basic information and click **Register**.

3 Advantages

CSS has the following features and advantages.

Efficient and Ease of Use

You can get insights from terabyte-scale data in milliseconds. In addition, you can use the visualized platform for data display and analysis.

Flexible and Scalable

You can request resources as needed and perform capacity expansion online with zero service interruption.

Easy O&M

CSS is a fully-managed, out-of-the-box service. You can start using it with several clicks, instead of managing clusters.

Kernel Enhancement

- **Vector search**

  When you search for unstructured data, such as images, videos, and corpuses, the nearest neighbors or approximate nearest neighbors are searched based on feature vectors. For details, see **Vector Retrieval**.

- **Decoupled storage and compute**

  CSS provides an API for freezing indexes. Hot data stored on SSD can be dumped to OBS to reduce data storage costs and decouple compute from storage. For details, see **Storage-Compute Decoupling**.

- **Flow control**

  CSS can control traffic at the node level. You can configure the blacklist and whitelist, the maximum concurrent HTTPS connections, and the maximum HTTP connections for a node. Each function has an independent control switch. For details, see **Flow Control**.

- **Large query isolation**

  CSS allows you to separately manage large queries. You can isolate query requests that consume a large amount of memory or take a long period of time. For details, see **Large Query Isolation**.

- **Index monitoring**

  CSS monitors various metrics of the running status and change trend of cluster indexes to measure service usage and handle potential risks in a timely manner, ensuring that clusters can run stably. For details, see **Index Monitoring**.

- **Enhanced monitoring**

  CSS supports enhanced cluster monitoring. It can monitor the P99 latency of cluster search requests and the HTTP status codes of clusters. For details, see **Enhanced Monitoring**.

High Reliability

You can choose to trigger snapshots manually or on a periodic basis for backup and restore snapshots to the current or other clusters. Snapshots of a cluster can be restored to another cluster to implement cluster data migration. For details, see **Index Backup and Restoration**.

- Automatic backup using snapshots

  CSS provides the backup function. You can enable the automatic backup function on the CSS management console and set the backup period based on the actual requirements.

  Automatic backup is to back up the index data of a cluster. Index backup is implemented by creating cluster snapshots. For backup of the first time, you are advised to back up all index data.

  CSS allows you to store the snapshot data of Elasticsearch instances to OBS, thereby achieving cross-region backup with the cross-region replication function of OBS.

- Restoring data using snapshots

  If data loss occurs or you want to retrieve data of a certain period, click **Restore** in the **Operation** column in the **Snapshots** area to restore the backup index data to the specified cluster by using existing snapshots.

High Security

CSS ensures secure running of data and services from the following aspects:

- Network isolation

  The network is divided into two planes, service plane and management plane. The two planes are deployed and isolated physically to ensure the security of the service and management networks.

  - Service plane: refers to the network plane of the cluster. It provides service channels for users and delivers data definition, index, and search capabilities.
  - Management plane: refers to the management console. It is used to manage CSS.
  - VPC security groups or isolated networks ensure the security of hosts.

- Access control
  - Using the network access control list (ACL), you can permit or deny the network traffic entering and exiting the subnets.

  - Internal security infrastructure (including the network firewall, intrusion detection system, and protection system) can monitor all network traffic that enters or exits the VPC through the IPsec VPN.
  - User authentication and index-level authentication are supported. CSS also supports interconnection with third-party user management systems.
- Data security
  - In CSS, the multi-replica mechanism is used to ensure user data security.
  - Communication between the client and server can be encrypted using SSL.
- Operation audit

  Cloud Trace Service (CTS) can be used to perform auditing on key logs and operations.

High Availability

To prevent data loss and minimize the cluster downtime in case of service interruption, CSS supports cross-AZ cluster deployment. When creating a cluster, you can select two or three AZs in the same region. The system will automatically allocate nodes to these AZs. If an AZ is faulty, the remaining AZs can still run properly, significantly enhancing cluster availability and improving service stability. For more information, see **Deploying a Cross-AZ Cluster**.

4 Product Components

CSS supports Kibana and Cerebro.

Kibana

Kibana is an open-source data analytics and visualization platform that works with Elasticsearch. You can use Kibana to search for and view data stored in Elasticsearch indexes and display data in charts and maps. For details about Kibana, visit **https://www.elastic.co/guide/en/kibana/current/index.html**.

By default, the Elasticsearch cluster of CSS provides the access channel to Kibana. You can quickly access Kibana without installing it. CSS is compatible with Kibana visualizations and Elasticsearch statistical and analysis capabilities.

- Over 10 data presentation modes
- Nearly 20 data statistics methods
- Classification in various dimensions, such as time and tag

Cerebro

Cerebro is an open-source Elasticsearch web admin tool built using Scala, Play Framework, AngularJS, and Bootstrap. Cerebro allows you to manage clusters on a visualized page, such as executing REST requests, modifying Elasticsearch configurations, monitoring real-time disks, cluster loads, and memory usage.

By default, the Elasticsearch cluster of CSS provides the access channel to Cerebro. You can quickly access Cerebro without installing it. CSS is fully compatible with the open-source Cerebro and adapts to the latest 0.8.4 version.

- Elasticsearch visualized and real-time load monitoring
- Elasticsearch visualized data management

5 Scenarios

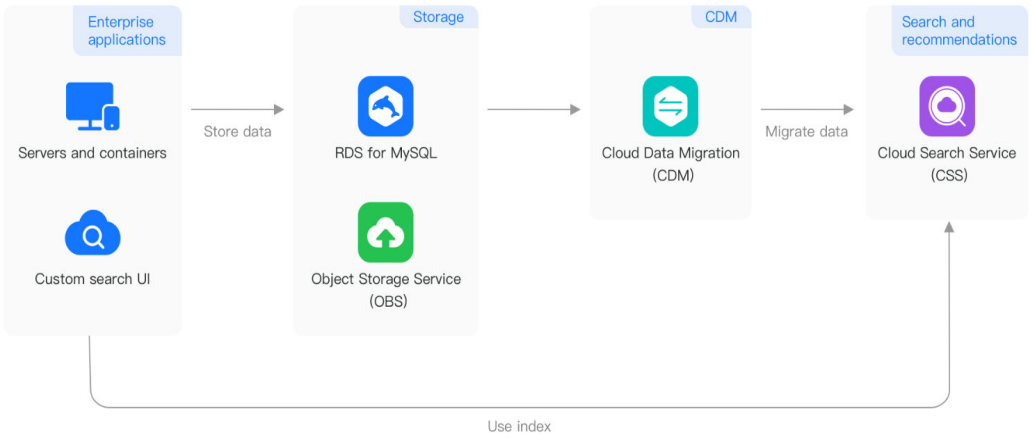
CSS can be used to build search boxes for websites and apps to improve user experience. You can also build a log analysis platform with it, facilitating data-driven O&M and business operations. CSS vector search can help you quickly build smart applications, such as AI-based image search, recommendation, and semantic search.

Site Search

CSS can be used to search for website content by keyword as well as search for and recommend commodities on e-commerce sites.

- Real-time search: When site content is updated, you can find the updated content in your search within minutes, or even just seconds.
- Categorized statistics: You can apply search filters to sort products by category.
- Custom highlight style: You can define how the search results are highlighted.

Figure 5-1 Site search



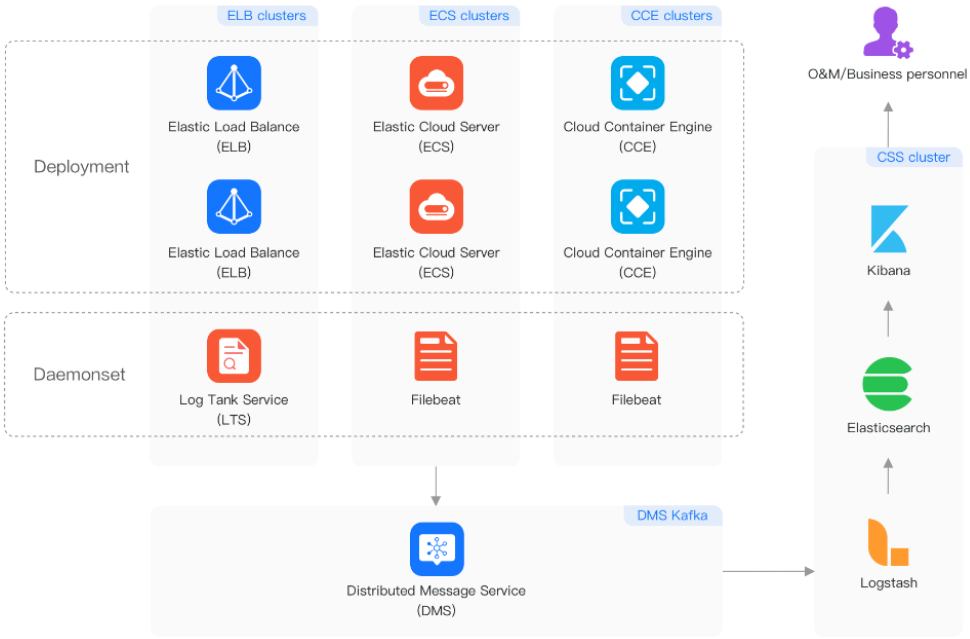
All-Scenario Log Analysis

Analyze the logs of Elastic Load Balance (ELB), servers, containers, and applications. In CSS, the Kafka message buffer queue is used to balance loads in

peak and off-peak hours. Logstash is used for data extract, transform and load (ETL). Elasticsearch retrieves and analyzes data. The analysis results are visualized by Kibana and presented to you.

- High cost-effectiveness: CSS uses the Kunpeng computing power, separates cold and hot storage, and decouples computing and storage resources, achieving high performance and reducing costs by over 30%.
- Ease of use: Perform queries in a GUI editor. Easily create reports using drag-and-drop components.
- Powerful processing capability: CSS can import hundreds of terabytes of data per day, and can process petabytes of data.

Figure 5-2 All-scenario log analysis

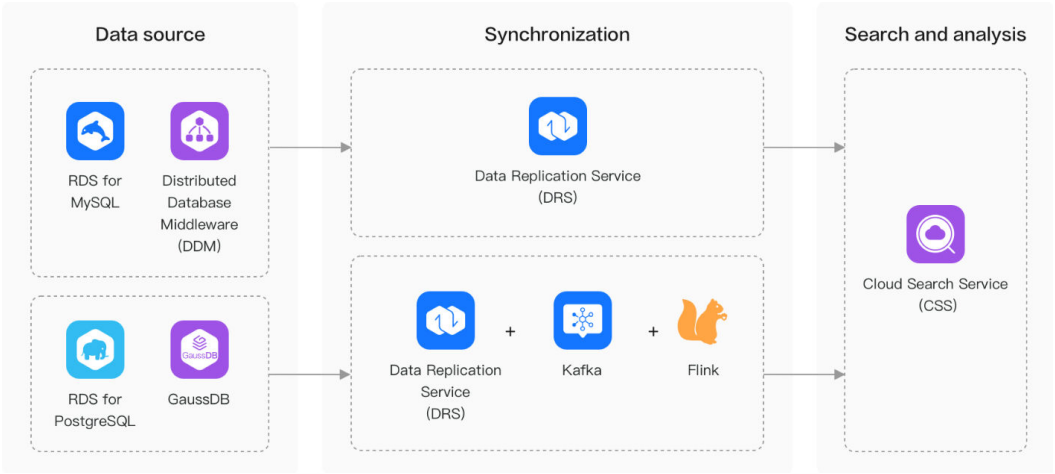


Database Query Acceleration

CSS can be used to accelerate database queries. E-commerce and logistics companies have to respond to a huge number of concurrent order queries within a short period of time. Relational databases, although having good transaction atomicity, are weak in transaction processing, and can rely on CSS to enhance OLTP and OLAP capabilities.

- High performance: Retrieve data from hundreds of millions of records within milliseconds. Text, time, numeric, and spatial data types are supported.
- High scalability: CSS can be scaled to have over 200 data nodes and over 1000 columns.
- Zero service interruption: The rolling restart and dual-copy mechanisms can avoid service interruption in case of specifications change or configuration update.

Figure 5-3 Database query acceleration

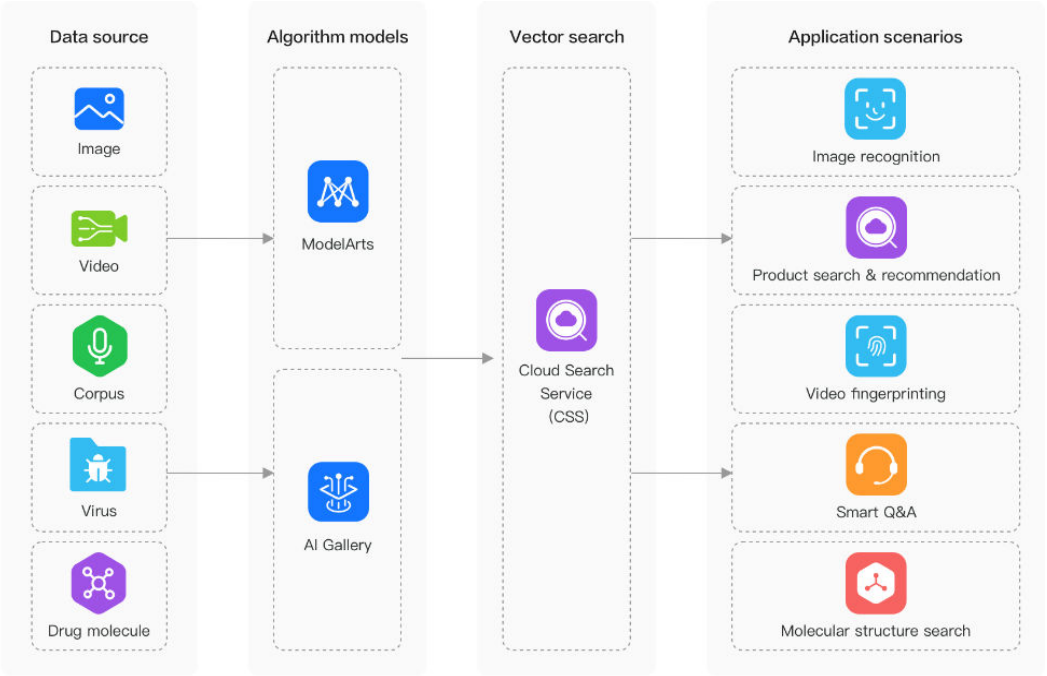


Vector Search

When you search for unstructured data, such as images, videos, and corpuses, the nearest neighbors or approximate nearest neighbors are searched based on feature vectors. This has the following advantages:

- Efficiency and reliability: The Huawei Cloud vector search engine provides ultimate search performance and distributed disaster recovery capabilities.
- Abundant indexes: Multiple indexing algorithms and similarity measurement methods are available and can meet diverse needs.
- Easy learning: CSS is fully compatible with the open-source Elasticsearch ecosystem.

Figure 5-4 Vector search



6 Billing

CSS supports two billing modes: pay-per-use and yearly/monthly. The latter is more cost-effective. For pricing details, see **CSS Price Calculator**.

Billing Items

CSS bills you for your selected instance specifications and usage duration.

Table 6-1 CSS billing

Billing Item	Description
Node Specifications	Specify the instance type and specifications (vCPUs and memory), required duration, and the number of purchased instances. You can purchase node instances in the pay-per-use or yearly/monthly mode.
Node storage	Specifies the disk type. You can select disks of a type based on your business scenario. The billing standard varies depending on the disk type. Billing mode: pay-per-use or yearly/monthly. CSS provides the following types of disks: • **Common I/O** • **High I/O** • **Ultra-High I/O**
Bandwidth	Specifies the bandwidth. When you enable the public IP address function or Kibana public IP address for a cluster, you will be billed for the bandwidth. Billing mode: pay-per-use or yearly/monthly. CSS provides the following types of bandwidth: • **Low bandwidth** (1 to 5 Mbit/s) • **High bandwidth** (6 to 2,000 Mbit/s) The billing automatically falls into low or high bandwidth based on the bandwidth you select.

Billing Modes

- Pay-per-use

  In pay-per-use mode, you are billed for a full hour even though you use CSS for less than 1 hour. You can enable or disable CSS as you like. This mode is suitable if you want more flexibility and control on resource usage.

- Yearly/Monthly billing

  In yearly/monthly mode, you need to pay for the service duration you selected in one-off mode. The service duration range is one month to three years. This mode provides a larger discount than pay-per-use and is recommended for long-term users.

Changing the Billing Mode

- Change the billing mode

  Changing from pay-per-use to yearly/monthly: After the billing mode is changed from pay-per-use to yearly/monthly, a new order is then generated for you, and the new billing mode takes effect immediately after you pay for the order.

  Changing from yearly/monthly to pay-per-use: The pay-per-use billing mode will take effect after the original yearly/monthly subscription expires.

- Change the node storage and quantity

  If the existing cluster adopts the pay-per-use billing mode, the modified nodes and node storage in the cluster will also be billed in the pay-per-use mode by default.

  For the modified nodes and storage capacity in a yearly/monthly cluster, their billing period starts from their provisioning time and ends with the yearly/monthly subscription of the cluster. For example, assume that your cluster is billed on a monthly basis and is scaled out on January 20. If the monthly subscription renews on January 30, you will have to pay for the fees generated by the new resources from January 20 to January 30.

- Change node specifications

  After node specifications are modified, nodes are billed based on the new specifications.

- Change bandwidth

  If you change the Internet access bandwidth for a cluster or Kibana, you will be billed based on the new bandwidth.

Renewal

You can renew a resource package upon its expiration, or you can set auto-renewal rules for a resource package. For more information about renewing resource packages, see **Renewal Management**.

Expiration and Overdue Payment

If your account is in arrears, you can view the arrears details in the Billing Center. To prevent related resources from being stopped or released, top up your account in a timely manner. If your account is in arrears, top up your account within the specified period. For details, see **Topping Up an Account**.

7 Security

7.1 Shared Responsibilities

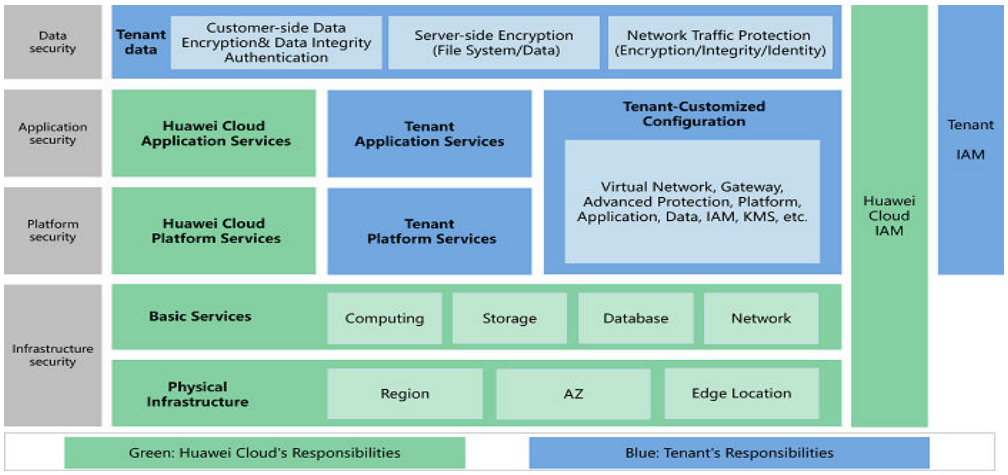
Huawei Cloud guarantees that its commitment to cyber security will never be outweighed by the consideration of commercial interests. To cope with emerging cloud security challenges and pervasive cloud security threats and attacks, Huawei Cloud builds a comprehensive cloud service security assurance system for different regions and industries based on Huawei's unique software and hardware advantages, laws, regulations, industry standards, and security ecosystem.

Figure 7-1 illustrates the responsibilities shared by Huawei Cloud and users.

- Huawei Cloud: ensures the security of cloud services and provides secure clouds. Huawei Cloud's security responsibilities include ensuring the security of our IaaS, PaaS, and SaaS services, as well as the physical environments of the Huawei Cloud data centers where our IaaS, PaaS, and SaaS services operate. Huawei Cloud is responsible for not only the security functions and performance of our infrastructure, cloud services, and technologies, but also for the overall cloud O&M security and, in the broader sense, the security compliance of our infrastructure and services.
- Tenant: uses the cloud securely. Tenants of Huawei Cloud are responsible for the secure and effective management of the tenant-customized configurations of cloud services including IaaS, PaaS, and SaaS. This includes but is not limited to virtual networks, the OS of virtual machine hosts and guests, virtual firewalls, API Gateway, advanced security services, all types of cloud services, tenant data, identity accounts, and key management.

Huawei Cloud Security White Paper introduces in detail the building ideas and measures of Huawei cloud security, including cloud security strategy, responsibility sharing model, compliance and privacy, security organization and personnel, infrastructure security, tenant service and tenant security, engineering security, O&M and operation security, and ecosystem security.

Figure 7-1 Huawei Cloud shared security responsibility model



7.2 Authentication and Access Control

CSS uses Identity and Access Management (IAM) and cluster security mode to perform authentication and access control for service resources and security clusters, respectively. The two modules are independent of each other.

IAM is used to control resource operation permissions on the CSS management plane. If you need to assign different permissions to employees in your organization to access your CSS resources, IAM is a good choice for fine-grained permissions management. IAM provides identity authentication, permissions management, and access control, helping you secure access to your CSS resources. For details about CSS permission management, see **Permissions Management**.

If the security mode is enabled for a cluster, identity authentication is required when users access the cluster. You can also authorize other users to access Kibana of the security cluster. For details, see **Clusters in Security Mode**. CSS supports identity authentication and access control only for clusters in security mode.

7.3 Data Protection Technologies

CSS uses network isolation, in addition to various host and data security measures.

- Network isolation

  The entire network is divided into two planes: service plane and management plane. The two planes are deployed and isolated physically to ensure the security of the service and management networks.

  - Service plane: This is the network plane of the cluster. It provides service channels for users and delivers data definitions, indexing, and search capabilities.
  - Management plane: This is the management console, where you manage CSS.

- Host security

  CSS provides the following security measures:

  - The VPC security group ensures the security of the hosts in a VPC.

  - Network access control lists (ACLs) allow you to control what data can enter or exit your network.
  - The internal security infrastructure (including the network firewall, intrusion detection system, and protection system) monitors all network traffic that enters or exits the VPC through an IPsec VPN.
- Data security

  Multiple replicas, cross-AZ deployment of clusters, and third-party (OBS) backup of index data ensure the security of user data.

7.4 Audit and Log

Auditing

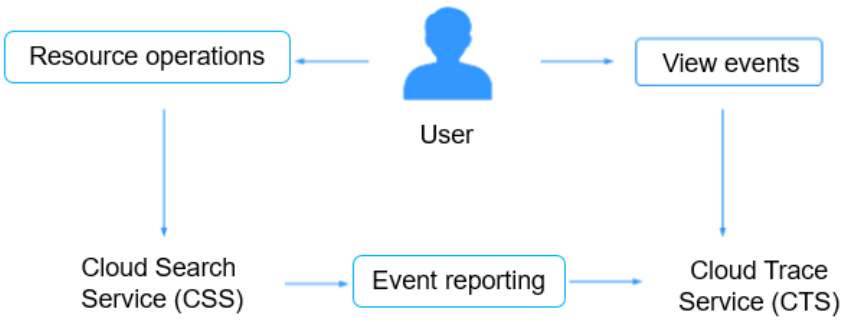
Cloud Trace Service (CTS) records operations on the cloud resources in your account. You can use the logs generated by CTS to perform security analysis, track resource changes, audit compliance, and locate faults.

After you enable CTS and create a tracker, CTS starts to record CSS operations for audit.

For details about how to enable and configure CTS, see **Enabling CTS**.

For details about CSS operations that can be recorded by CTS, see **Key Operations Recorded by CTS**.

Figure 7-2 CTS



After CTS is enabled, CTS starts recording CSS operations. The CTS management console stores the operation records generated in the last seven days. For details about how to view operation records for the last seven days on the CTS console, see **Viewing Audit Logs**.

Logging

CSS allows you to back up and query logs for locating faults. You can back up cluster logs in OBS buckets and download required log files from OBS for fault analysis and locating. For details about CSS log management, see **Managing Logs**.

7.5 Security Risk Monitoring

Cloud Eye is a monitoring platform for Huawei Cloud resources. It provides capabilities such as real-time monitoring, alarm reporting, resource grouping, and website monitoring. Cloud Eye can monitor metrics of CSS clusters and nodes and visualize the monitoring information in reports.

For details about CSS metrics that can be monitored by Cloud Eye, see **Supported Metrics**. Cloud Eye allows you to configure threshold-crossing alarms for specified monitoring metrics.

7.6 Security Notice

7.6.1 Notice on Apache Log4j Remote Code Execution Vulnerability (CVE-2021-44228)

Apache Log4j2 has a remote code execution vulnerability (CVE-2021-44228). This notice describes the impact of the vulnerability and its fix.

Vulnerability Impact

Apache Log4j2 has a remote code execution vulnerability (CVE-2021-44228). When Apache Log4j2 processes user input during log processing, attackers can construct special requests to trigger remote code execution. The POC has been disclosed and the risk is high. For details, see **Apache Log4j2 Remote Code Execution Vulnerability (CVE-2021-44228 and CVE-2021-45046)**.

Elasticsearch uses the Log4j framework to record logs and uses Java security manager, so it is not affected by this remote code execution vulnerability. Attackers can exploit the information leakage vulnerability in Log4j to obtain environment variables and some environment data through DNS, but cannot access data in Elasticsearch clusters, so there is no risk of data leakage.

Vulnerability Fix

CSS has installed a patch package for existing clusters to fix this vulnerability. If the cluster has not been restarted since March 30, 2022, restart it to make the patch take effect.

8 Permissions Management

If you need to assign different permissions to employees in your organization to access your CSS resources, IAM is a good choice for fine-grained permissions management. IAM provides identity authentication, permissions management, and access control.

If the current account has met your requirements, you do not need to create an independent IAM user for permission management. Then you can skip this section. This will not affect other functions of CSS.

With IAM, you can use your account to create IAM users for your employees and assign permissions to the users to control their access to your resources. IAM is free of charge. You pay only for the resources you purchase. For more information about IAM, see **IAM Service Overview**.

Permissions Management

New IAM users do not have any permissions assigned by default. You need to first add them to one or more groups and attach policies or roles to these groups. The users then inherit permissions from the groups and can perform specified operations on cloud services based on the permissions they have been assigned.

CSS is a project-level service deployed in specific physical regions. Therefore, CSS permissions are assigned to projects in specific regions and only take effect in these regions. If you want the permissions to take effect in all regions, you need to assign the permissions to projects in each region. When accessing CSS, the users need to switch to a region where they have been authorized to use cloud services.

You can use roles and policies to grant users permissions.

- Roles: A type of coarse-grained authorization mechanism that defines permissions related to user responsibilities. This mechanism provides only a limited number of service-level roles for authorization. When using roles to grant permissions, you need to also assign dependency roles. Roles are not ideal for fine-grained authorization and secure access control.
- Policies: A type of fine-grained authorization mechanism that defines the permissions for performing operations on specific cloud resources under certain conditions. This mechanism allows for more flexible authorization. Policies allow you to meet requirements for more secure access control. For example, CSS administrators can only grant CSS users the permissions needed

for managing a particular type of CSS resources. Most policies define permissions based on APIs. For the API actions supported by CSS, see **Permissions Policies and Supported Actions**.

Table 8-1 lists all the system-defined roles and policies supported by CSS.

- **CSS Administrator** depends on the roles of other services to execute its permissions. Therefore, if you assign the **Elasticsearch Administrator** role to a user, assign its dependency roles at the same time.
- **CSS FullAccess** and **CSS ReadOnlyAccess** can be used to control the resources that users can access. For example, if you want your software developers to use CSS resources but not delete them or perform any high-risk operations, you can create IAM users for these software developers and assign them only the permissions required for using CSS resources.

Table 8-1 CSS system permission

Role/Policy Name	Type	Description	Dependency
CSS Administrator	System-defined role	Full permissions for CSS. This role depends on the **Tenant Guest**, **Server Administrator**, and **IAM ReadOnlyAccess** roles in the same project.	The **VPCEndpoint Administrator** system role is required for accessing a cluster through a VPC endpoint. Some operations depend on the following permissions: • View the agency list: iam:agencies:listAgencies iam:permissions:listRolesForAgency iam:permissions:listRolesForAgencyOnProject • Automatically create an agency: iam:agencies:listAgencies iam:agencies:createAgency iam:permissions:grantRoleToAgency • Display enterprise projects and predefined tags on the console: eps:enterpriseProjects:list tms:predefineTags:list • Use the snapshot, word dictionary, and log management functions: obs:bucket:Get* obs:bucket:List* obs:object:List* obs:object:Get* obs:bucket:HeadBucket obs:object:PutObject obs:object:DeleteObject
CSS FullAccess	System-defined policy	Full CSS permissions granted through policies. Users with these permissions can perform all operations on CSS. Some functions depend on corresponding permissions. To use certain functions, you need to enable the dependent permissions in the same project.	

Role/Policy Name	Type	Description	Dependency
			• Create a yearly/monthly order and pay for it: bss:order:update bss:order:pay
CSS ReadOnlyAccess	System-defined policy	Read-only permissions for CSS. Users with these permissions can only view CSS data. Some functions depend on corresponding permissions. To use certain functions, you need to enable the dependent permissions in global services.	Some operations depend on the following permissions: • View the agency list: iam:agencies:listAgencies iam:permissions:listRolesForAgency iam:permissions:listRolesForAgencyOnProject • Display enterprise projects and predefined tags on the console: eps:enterpriseProjects:list tms:predefineTags:list • Use the snapshot, word dictionary, and log management functions: obs:bucket:Get* obs:bucket:List* obs:object:List* obs:object:Get* obs:bucket:HeadBucket

Table 8-2 lists the common operations supported by each system permission of CSS. Please choose proper system permissions according to this table.

Table 8-2 Common operations supported by each system-defined policy

Operation	CSS FullAccess	CSS ReadOnlyAccess	CSS Administrator	Remarks
Creating a cluster	√	x	√	-
Querying the cluster list	√	√	√	-
Querying the cluster details	√	√	√	-
Deleting a cluster	√	x	√	-
Restarting a cluster	√	x	√	-
Expanding cluster capacity	√	x	√	-
Adding instances and expanding instance storage capacity	√	x	√	-
Querying tags of a specified cluster	√	√	√	-
Querying all tags	√	√	√	-
Loading a custom word dictionary	√	x	√	Depends on OBS and IAM permissions
Querying the status of a custom word dictionary	√	√	√	-

Operation	CSS FullAccess	CSS ReadOnlyAccess	CSS Administrator	Remarks
Deleting a custom word dictionary	√	x	√	-
Automatically setting basic configurations of a cluster snapshot	√	x	√	Depends on OBS and IAM permissions
Modifying basic configurations of a cluster snapshot	√	x	√	Depends on OBS and IAM permissions
Setting the automatic snapshot creation policy	√	x	√	-
Querying the automatic snapshot creation policy	√	√	√	-
Manually creating a snapshot	√	x	√	-
Querying the snapshot list	√	√	√	-
Restoring a snapshot	√	x	√	-
Deleting a snapshot	√	x	√	-

Operation	CSS FullAccess	CSS ReadOnlyAccess	CSS Administrator	Remarks
Disabling the snapshot function	√	x	√	-
Modifying specifications	√	x	√	-
Scaling in clusters	√	x	√	-

Helpful Links

- **IAM Service Overview**
- **Creating a User and Granting Permissions**
- **Permissions Policies and Supported Actions**

9 Specifications

When creating a CSS cluster, you can select specifications as required. For details about the specifications and application scenarios, see **Table 9-1**.

Table 9-1 Node specifications

CPU Architecture	Specification Type	CPU/Memory Ratio	Application Scenario
x86	Compute-intensive	1:2	This specification provides strong CPUs and is suitable for search scenarios that require high computing performance and low latency, such as e-commerce and app search. It is recommended when ultra-high I/O disks are used. This specification is expensive and has higher reliability than that of the NVMe local disk cluster.
	Disk-intensive	1:8	This specification uses local SAS pass-through disks that have large space. It is applicable to scenarios where a large amount of data is stored, such as searching for logs and public opinions. Generally, it is recommended for cold nodes.
	General computing	1:4	This is the default specification, which is widely used in diverse scenarios and can meet your common requirements.
	Memory-optimized	1:8	This specification provides large memory. It is recommended for scenarios that require large memory and do not need quick response, such as multi-aggregation (in the **filedata** heap), sorting, and column storage format **DocValue** (out-of-heap memory).

10 Constraints

Restrictions on Clusters and Nodes

The following table describes restrictions on clusters and nodes in CSS.

Table 10-1 Restrictions on Elasticsearch clusters and nodes

Cluster and Node	Restriction
Maximum number of nodes in a cluster	Default: **32**. Maximum: 200. To change the default value, contact technical support.
Minimum number of nodes in a cluster	1

Restrictions on Browsers

- You are advised to use the following browsers to access the CSS management console:
  - Google Chrome 36.0 or later
  - Mozilla Firefox 35.0 or later
  - Microsoft Edge: the latest three stable versions
- You are advised to use the following browsers to access Kibana and Cerebro integrated in CSS:
  - Google Chrome 36.0 or later
  - Mozilla Firefox 35.0 or later
  - Microsoft Edge: the latest three stable versions

11 Performance Metrics

11.1 Overview

This section describes the result of testing performance of CSS clusters (in version 7.6.2) by using Rally 1.0.0 provided by Elasticsearch.

The data in the following tables is based on **geonames** that contains 11,396,505 documents with a total size of 3.2 GB. The index uses six shards. By default, an index uses five shards. For details about performance metrics, see the official document **https://esrally.readthedocs.io/en/stable/summary_report.html#summary-report**.

The performance test results of CSS clusters with the following specifications are provided:

- **Performance Test of a Cluster with Three Nodes of the ess.spec-2u8g Specification**
- **Performance Test of a Cluster with Three Nodes of the ess.spec-4u16g Specification**

The performance test result comparison between the cluster with three nodes of the **ess.spec-2u8g** specification and that with three nodes of the **ess.spec-4u16g** specification is also provided. For details, see **Performance Comparison Between Clusters of the ess.spec-2u8g and ess.spec-4u16g Specification**.

11.2 Performance Test of a Cluster with Three Nodes of the ess.spec-2u8g Specification

The following table lists the performance test result of a cluster with three nodes of the **ess.spec-2u8g** specification.

Metric	Task	Value	Unit
Cumulative indexing time of primary shards	-	11.48263333	min

Metric	Task	Value	Unit
Min cumulative indexing time across primary shards	-	0	min
Median cumulative indexing time across primary shards	-	2.313783333	min
Max cumulative indexing time across primary shards	-	2.401766667	min
Cumulative indexing throttle time of primary shards	-	0	min
Min cumulative indexing throttle time across primary shards	-	0	min
Median cumulative indexing throttle time across primary shards	-	0	min
Max cumulative indexing throttle time across primary shards	-	0	min
Cumulative merge time of primary shards	-	6.466066667	min
Cumulative merge count of primary shards	-	85	-
Min cumulative merge time across primary shards	-	0	min
Median cumulative merge time across primary shards	-	1.257475	min

Metric	Task	Value	Unit
Max cumulative merge time across primary shards	-	1.417283333	min
Cumulative merge throttle time of primary shards	-	1.089583333	min
Min cumulative merge throttle time across primary shards	-	0	min
Median cumulative merge throttle time across primary shards	-	0.200458333	min
Max cumulative merge throttle time across primary shards	-	0.28265	min
Cumulative refresh time of primary shards	-	3.641266667	min
Cumulative refresh count of primary shards	-	530	-
Min cumulative refresh time across primary shards	-	0	min
Median cumulative refresh time across primary shards	-	0.725791667	min
Max cumulative refresh time across primary shards	-	0.74775	min
Cumulative flush time of primary shards	-	0.3056	min
Cumulative flush count of primary shards	-	11	-

Metric	Task	Value	Unit
Min cumulative flush time across primary shards	-	0	min
Median cumulative flush time across primary shards	-	0.059858333	min
Max cumulative flush time across primary shards	-	0.09155	min
Total Young Gen GC	-	11.519	s
Total Old Gen GC	-	0	s
Store size	-	3.045436038	GB
Translog size	-	2.791873856	GB
Heap used for segments	-	15.81298065	MB
Heap used for doc values	-	0.037128448	MB
Heap used for terms	-	14.63806534	MB
Heap used for norms	-	0.073120117	MB
Heap used for points	-	0.272666931	MB
Heap used for stored fields	-	0.791999817	MB
Segment count	-	95	-
Min Throughput	index-append	41705.19	docs/s
Median Throughput	index-append	46911.27	docs/s
Max Throughput	index-append	47765.4	docs/s
50th percentile latency	index-append	642.339781	ms
90th percentile latency	index-append	1114.672936	ms

Metric	Task	Value	Unit
99th percentile latency	index-append	1733.648438	ms
99.9th percentile latency	index-append	4770.059011	ms
100th percentile latency	index-append	7045.246771	ms
50th percentile service time	index-append	642.339781	ms
90th percentile service time	index-append	1114.672936	ms
99th percentile service time	index-append	1733.648438	ms
99.9th percentile service time	index-append	4770.059011	ms
100th percentile service time	index-append	7045.246771	ms
error rate	index-append	0	%
Min Throughput	index-stats	90.05	ops/s
Median Throughput	index-stats	90.07	ops/s
Max Throughput	index-stats	90.12	ops/s
50th percentile latency	index-stats	2.834653556	ms
90th percentile latency	index-stats	3.527868712	ms
99th percentile latency	index-stats	4.332674769	ms
99.9th percentile latency	index-stats	8.392195267	ms
100th percentile latency	index-stats	9.692270112	ms
50th percentile service time	index-stats	2.766648	ms
90th percentile service time	index-stats	3.448194001	ms
99th percentile service time	index-stats	4.26309684	ms

Metric	Task	Value	Unit
99.9th percentile service time	index-stats	8.322068306	ms
100th percentile service time	index-stats	9.624071001	ms
error rate	index-stats	0	%
Min Throughput	node-stats	90.06	ops/s
Median Throughput	node-stats	90.1	ops/s
Max Throughput	node-stats	90.35	ops/s
50th percentile latency	node-stats	3.205233055	ms
90th percentile latency	node-stats	3.595145422	ms
99th percentile latency	node-stats	4.469114152	ms
99.9th percentile latency	node-stats	8.306063762	ms
100th percentile latency	node-stats	8.748160444	ms
50th percentile service time	node-stats	3.1379455	ms
90th percentile service time	node-stats	3.5278055	ms
99th percentile service time	node-stats	4.397312671	ms
99.9th percentile service time	node-stats	8.236949997	ms
100th percentile service time	node-stats	8.680502	ms
error rate	node-stats	0	%
Min Throughput	default	50.03	ops/s
Median Throughput	default	50.05	ops/s
Max Throughput	default	50.09	ops/s
50th percentile latency	default	2.354736001	ms

Metric	Task	Value	Unit
90th percentile latency	default	2.7983462	ms
99th percentile latency	default	4.59134772	ms
99.9th percentile latency	default	13.97301623	ms
100th percentile latency	default	16.199022	ms
50th percentile service time	default	2.286799	ms
90th percentile service time	default	2.7289099	ms
99th percentile service time	default	4.511846871	ms
99.9th percentile service time	default	13.90608139	ms
100th percentile service time	default	16.130242	ms
error rate	default	0	%
Min Throughput	term	150.07	ops/s
Median Throughput	term	150.1	ops/s
Max Throughput	term	150.15	ops/s
50th percentile latency	term	2.316147835	ms
90th percentile latency	term	2.610932901	ms
99th percentile latency	term	5.968978318	ms
99.9th percentile latency	term	10.37105939	ms
100th percentile latency	term	12.147341	ms
50th percentile service time	term	2.249188999	ms
90th percentile service time	term	2.5313585	ms

Metric	Task	Value	Unit
99th percentile service time	term	5.32149807	ms
99.9th percentile service time	term	9.589421289	ms
100th percentile service time	term	11.204094	ms
error rate	term	0	%
Min Throughput	phrase	150.07	ops/s
Median Throughput	phrase	150.1	ops/s
Max Throughput	phrase	150.16	ops/s
50th percentile latency	phrase	2.350160666	ms
90th percentile latency	phrase	2.689091867	ms
99th percentile latency	phrase	4.606508314	ms
99.9th percentile latency	phrase	11.32920839	ms
100th percentile latency	phrase	11.53972367	ms
50th percentile service time	phrase	2.283426499	ms
90th percentile service time	phrase	2.6023857	ms
99th percentile service time	phrase	4.073278879	ms
99.9th percentile service time	phrase	11.26236945	ms
100th percentile service time	phrase	11.471612	ms
error rate	phrase	0	%
Min Throughput	country_agg_uncached	4	ops/s
Median Throughput	country_agg_uncached	4.01	ops/s

Metric	Task	Value	Unit
Max Throughput	country_agg_uncached	4.01	ops/s
50th percentile latency	country_agg_uncached	154.036113	ms
90th percentile latency	country_agg_uncached	160.160262	ms
99th percentile latency	country_agg_uncached	217.9470218	ms
100th percentile latency	country_agg_uncached	270.401061	ms
50th percentile service time	country_agg_uncached	153.9164235	ms
90th percentile service time	country_agg_uncached	160.0393962	ms
99th percentile service time	country_agg_uncached	217.8203381	ms
100th percentile service time	country_agg_uncached	270.314704	ms
error rate	country_agg_uncached	0	%
Min Throughput	country_agg_cached	100.04	ops/s
Median Throughput	country_agg_cached	100.06	ops/s
Max Throughput	country_agg_cached	100.07	ops/s
50th percentile latency	country_agg_cached	1.772262999	ms
90th percentile latency	country_agg_cached	1.943878399	ms
99th percentile latency	country_agg_cached	2.796966468	ms
99.9th percentile latency	country_agg_cached	6.427875642	ms
100th percentile latency	country_agg_cached	14.575363	ms
50th percentile service time	country_agg_cached	1.7050655	ms

Metric	Task	Value	Unit
90th percentile service time	country_agg_cached	1.878483099	ms
99th percentile service time	country_agg_cached	2.689127631	ms
99.9th percentile service time	country_agg_cached	4.762661218	ms
100th percentile service time	country_agg_cached	14.506126	ms
error rate	country_agg_cached	0	%
Min Throughput	scroll	20.05	pages/s
Median Throughput	scroll	20.06	pages/s
Max Throughput	scroll	20.07	pages/s
50th percentile latency	scroll	387.0272235	ms
90th percentile latency	scroll	400.7843767	ms
99th percentile latency	scroll	452.1627557	ms
100th percentile latency	scroll	478.26665	ms
50th percentile service time	scroll	386.143462	ms
90th percentile service time	scroll	399.8976064	ms
99th percentile service time	scroll	451.295933	ms
100th percentile service time	scroll	477.360055	ms
error rate	scroll	0	%
Min Throughput	expression	2	ops/s
Median Throughput	expression	2	ops/s
Max Throughput	expression	2	ops/s
50th percentile latency	expression	285.121047	ms

Metric	Task	Value	Unit
90th percentile latency	expression	292.0323929	ms
99th percentile latency	expression	336.1215281	ms
100th percentile latency	expression	389.221478	ms
50th percentile service time	expression	284.883145	ms
90th percentile service time	expression	291.78961	ms
99th percentile service time	expression	335.9078465	ms
100th percentile service time	expression	388.982388	ms
error rate	expression	0	%
Min Throughput	painless_static	1.5	ops/s
Median Throughput	painless_static	1.5	ops/s
Max Throughput	painless_static	1.5	ops/s
50th percentile latency	painless_static	414.4142772	ms
90th percentile latency	painless_static	428.3021712	ms
99th percentile latency	painless_static	551.0764984	ms
100th percentile latency	painless_static	586.564512	ms
50th percentile service time	painless_static	414.134189	ms
90th percentile service time	painless_static	428.0409987	ms
99th percentile service time	painless_static	550.7989791	ms
100th percentile service time	painless_static	586.432656	ms
error rate	painless_static	0	%
Min Throughput	painless_dynamic	1.5	ops/s

Metric	Task	Value	Unit
Median Throughput	painless_dynamic	1.5	ops/s
Max Throughput	painless_dynamic	1.5	ops/s
50th percentile latency	painless_dynamic	387.1022877	ms
90th percentile latency	painless_dynamic	402.260061	ms
99th percentile latency	painless_dynamic	472.1731577	ms
100th percentile latency	painless_dynamic	480.22595	ms
50th percentile service time	painless_dynamic	386.7965725	ms
90th percentile service time	painless_dynamic	401.955634	ms
99th percentile service time	painless_dynamic	471.9657896	ms
100th percentile service time	painless_dynamic	479.91248	ms
error rate	painless_dynamic	0	%
Min Throughput	decay_geo_gauss_function_score	1	ops/s
Median Throughput	decay_geo_gauss_function_score	1	ops/s
Max Throughput	decay_geo_gauss_function_score	1	ops/s
50th percentile latency	decay_geo_gauss_function_score	364.5783855	ms
90th percentile latency	decay_geo_gauss_function_score	369.3249541	ms
99th percentile latency	decay_geo_gauss_function_score	376.4548957	ms
100th percentile latency	decay_geo_gauss_function_score	402.051915	ms
50th percentile service time	decay_geo_gauss_function_score	364.0542175	ms

Metric	Task	Value	Unit
90th percentile service time	decay_geo_gauss_function_score	368.6669817	ms
99th percentile service time	decay_geo_gauss_function_score	375.7975505	ms
100th percentile service time	decay_geo_gauss_function_score	401.399591	ms
error rate	decay_geo_gauss_function_score	0	%
Min Throughput	decay_geo_gauss_script_score	1	ops/s
Median Throughput	decay_geo_gauss_script_score	1	ops/s
Max Throughput	decay_geo_gauss_script_score	1	ops/s
50th percentile latency	decay_geo_gauss_script_score	388.6800445	ms
90th percentile latency	decay_geo_gauss_script_score	404.632834	ms
99th percentile latency	decay_geo_gauss_script_score	450.7542979	ms
100th percentile latency	decay_geo_gauss_script_score	538.551451	ms
50th percentile service time	decay_geo_gauss_script_score	388.0335405	ms
90th percentile service time	decay_geo_gauss_script_score	403.9975599	ms
99th percentile service time	decay_geo_gauss_script_score	450.1032284	ms
100th percentile service time	decay_geo_gauss_script_score	537.919936	ms
error rate	decay_geo_gauss_script_score	0	%
Min Throughput	field_value_function_score	1.5	ops/s
Median Throughput	field_value_function_score	1.5	ops/s
Max Throughput	field_value_function_score	1.51	ops/s

Metric	Task	Value	Unit
50th percentile latency	field_value_function_score	147.6084107	ms
90th percentile latency	field_value_function_score	161.4163745	ms
99th percentile latency	field_value_function_score	218.4858815	ms
100th percentile latency	field_value_function_score	223.5476993	ms
50th percentile service time	field_value_function_score	147.071556	ms
90th percentile service time	field_value_function_score	160.8855899	ms
99th percentile service time	field_value_function_score	217.9465422	ms
100th percentile service time	field_value_function_score	223.080105	ms
error rate	field_value_function_score	0	%
Min Throughput	field_value_script_score	1.5	ops/s
Median Throughput	field_value_script_score	1.5	ops/s
Max Throughput	field_value_script_score	1.51	ops/s
50th percentile latency	field_value_script_score	208.4922433	ms
90th percentile latency	field_value_script_score	213.0348423	ms
99th percentile latency	field_value_script_score	256.5748294	ms
100th percentile latency	field_value_script_score	274.4188643	ms
50th percentile service time	field_value_script_score	208.058553	ms
90th percentile service time	field_value_script_score	212.5744289	ms
99th percentile service time	field_value_script_score	256.1503058	ms

Metric	Task	Value	Unit
100th percentile service time	field_value_script_score	274.185904	ms
error rate	field_value_script_score	0	%
Min Throughput	random_function_score	1.5	ops/s
Median Throughput	random_function_score	1.5	ops/s
Max Throughput	random_function_score	1.5	ops/s
50th percentile latency	random_function_score	244.4104887	ms
90th percentile latency	random_function_score	257.7793149	ms
99th percentile latency	random_function_score	323.8163443	ms
100th percentile latency	random_function_score	376.470245	ms
50th percentile service time	random_function_score	243.9546325	ms
90th percentile service time	random_function_score	257.3440943	ms
99th percentile service time	random_function_score	323.3741708	ms
100th percentile service time	random_function_score	376.091853	ms
error rate	random_function_score	0	%
Min Throughput	random_script_score	1.5	ops/s
Median Throughput	random_script_score	1.5	ops/s
Max Throughput	random_script_score	1.5	ops/s
50th percentile latency	random_script_score	265.276135	ms
90th percentile latency	random_script_score	276.8986875	ms

Metric	Task	Value	Unit
99th percentile latency	random_script_score	327.6141767	ms
100th percentile latency	random_script_score	339.1401533	ms
50th percentile service time	random_script_score	264.845466	ms
90th percentile service time	random_script_score	276.4729421	ms
99th percentile service time	random_script_score	327.2584587	ms
100th percentile service time	random_script_score	338.704812	ms
error rate	random_script_score	0	%
Min Throughput	large_terms	1.5	ops/s
Median Throughput	large_terms	1.5	ops/s
Max Throughput	large_terms	1.5	ops/s
50th percentile latency	large_terms	474.347426	ms
90th percentile latency	large_terms	482.346874	ms
99th percentile latency	large_terms	521.4118005	ms
100th percentile latency	large_terms	529.6919453	ms
50th percentile service time	large_terms	474.1270145	ms
90th percentile service time	large_terms	482.1388748	ms
99th percentile service time	large_terms	521.2451771	ms
100th percentile service time	large_terms	529.479614	ms
error rate	large_terms	0	%
Min Throughput	large_filtered_terms	1.5	ops/s

Metric	Task	Value	Unit
Median Throughput	large_filtered_terms	1.5	ops/s
Max Throughput	large_filtered_terms	1.5	ops/s
50th percentile latency	large_filtered_terms	475.7995187	ms
90th percentile latency	large_filtered_terms	486.3646669	ms
99th percentile latency	large_filtered_terms	565.6174992	ms
100th percentile latency	large_filtered_terms	585.669044	ms
50th percentile service time	large_filtered_terms	475.580755	ms
90th percentile service time	large_filtered_terms	486.1421912	ms
99th percentile service time	large_filtered_terms	565.483224	ms
100th percentile service time	large_filtered_terms	585.452311	ms
error rate	large_filtered_terms	0	%
Min Throughput	large_prohibited_terms	1.5	ops/s
Median Throughput	large_prohibited_terms	1.5	ops/s
Max Throughput	large_prohibited_terms	1.5	ops/s
50th percentile latency	large_prohibited_terms	474.8867557	ms
90th percentile latency	large_prohibited_terms	483.007269	ms
99th percentile latency	large_prohibited_terms	540.355679	ms
100th percentile latency	large_prohibited_terms	574.8374467	ms
50th percentile service time	large_prohibited_terms	474.6650815	ms

Metric	Task	Value	Unit
90th percentile service time	large_prohibited_terms	482.7923966	ms
99th percentile service time	large_prohibited_terms	540.1352455	ms
100th percentile service time	large_prohibited_terms	574.674312	ms
error rate	large_prohibited_terms	0	%
Min Throughput	desc_sort_population	1.5	ops/s
Median Throughput	desc_sort_population	1.51	ops/s
Max Throughput	desc_sort_population	1.51	ops/s
50th percentile latency	desc_sort_population	49.97947483	ms
90th percentile latency	desc_sort_population	52.97220567	ms
99th percentile latency	desc_sort_population	65.81446927	ms
100th percentile latency	desc_sort_population	68.243857	ms
50th percentile service time	desc_sort_population	49.3373975	ms
90th percentile service time	desc_sort_population	52.3443909	ms
99th percentile service time	desc_sort_population	65.17446437	ms
100th percentile service time	desc_sort_population	67.595051	ms
error rate	desc_sort_population	0	%
Min Throughput	asc_sort_population	1.5	ops/s
Median Throughput	asc_sort_population	1.51	ops/s
Max Throughput	asc_sort_population	1.51	ops/s

Metric	Task	Value	Unit
50th percentile latency	asc_sort_population	50.29814734	ms
90th percentile latency	asc_sort_population	54.12596357	ms
99th percentile latency	asc_sort_population	57.9221302	ms
100th percentile latency	asc_sort_population	69.35533	ms
50th percentile service time	asc_sort_population	49.667352	ms
90th percentile service time	asc_sort_population	53.4878858	ms
99th percentile service time	asc_sort_population	57.2779194	ms
100th percentile service time	asc_sort_population	68.714241	ms
error rate	asc_sort_population	0	%
Min Throughput	desc_sort_geoname id	1.5	ops/s
Median Throughput	desc_sort_geoname id	1.51	ops/s
Max Throughput	desc_sort_geoname id	1.51	ops/s
50th percentile latency	desc_sort_geoname id	49.2601545	ms
90th percentile latency	desc_sort_geoname id	53.48767223	ms
99th percentile latency	desc_sort_geoname id	69.43293772	ms
100th percentile latency	desc_sort_geoname id	72.512932	ms
50th percentile service time	desc_sort_geoname id	48.6107425	ms
90th percentile service time	desc_sort_geoname id	52.839748	ms
99th percentile service time	desc_sort_geoname id	68.79282147	ms

Metric	Task	Value	Unit
100th percentile service time	desc_sort_geonameid	71.872758	ms
error rate	desc_sort_geonameid	0	%
Min Throughput	asc_sort_geonameid	1.5	ops/s
Median Throughput	asc_sort_geonameid	1.51	ops/s
Max Throughput	asc_sort_geonameid	1.51	ops/s
50th percentile latency	asc_sort_geonameid	47.071104	ms
90th percentile latency	asc_sort_geonameid	50.264151	ms
99th percentile latency	asc_sort_geonameid	57.9888054	ms
100th percentile latency	asc_sort_geonameid	96.39665433	ms
50th percentile service time	asc_sort_geonameid	46.427649	ms
90th percentile service time	asc_sort_geonameid	49.6192723	ms
99th percentile service time	asc_sort_geonameid	57.75922607	ms
100th percentile service time	asc_sort_geonameid	95.751176	ms
error rate	asc_sort_geonameid	0	%

11.3 Performance Test of a Cluster with Three Nodes of the ess.spec-4u16g Specification

The following table lists the performance test result of a cluster with three nodes of the **ess.spec-4u16g** specification.

Metric	Task	Value	Unit
Cumulative indexing time of primary shards	-	11.95073333	min
Min cumulative indexing time across primary shards	-	0	min
Median cumulative indexing time across primary shards	-	2.339941667	min
Max cumulative indexing time across primary shards	-	2.470116667	min
Cumulative indexing throttle time of primary shards	-	0	min
Min cumulative indexing throttle time across primary shards	-	0	min
Median cumulative indexing throttle time across primary shards	-	0	min
Max cumulative indexing throttle time across primary shards	-	0	min
Cumulative merge time of primary shards	-	4.21495	min
Cumulative merge count of primary shards	-	65	-
Min cumulative merge time across primary shards	-	0	min
Median cumulative merge time across primary shards	-	0.813216667	min
Max cumulative merge time across primary shards	-	0.974483333	min

Metric	Task	Value	Unit
Cumulative merge throttle time of primary shards	-	0.83345	min
Min cumulative merge throttle time across primary shards	-	0	min
Median cumulative merge throttle time across primary shards	-	0.157775	min
Max cumulative merge throttle time across primary shards	-	0.24605	min
Cumulative refresh time of primary shards	-	2.164983333	min
Cumulative refresh count of primary shards	-	291	-
Min cumulative refresh time across primary shards	-	0	min
Median cumulative refresh time across primary shards	-	0.425391667	min
Max cumulative refresh time across primary shards	-	0.450516667	min
Cumulative flush time of primary shards	-	0.1559	min
Cumulative flush count of primary shards	-	11	-
Min cumulative flush time across primary shards	-	0	min
Median cumulative flush time across primary shards	-	0.0248	min

Metric	Task	Value	Unit
Max cumulative flush time across primary shards	-	0.043433333	min
Total Young Gen GC	-	6.421	s
Total Old Gen GC	-	0	s
Store size	-	3.124213032	GB
Translog size	-	2.790678718	GB
Heap used for segments	-	15.03110981	MB
Heap used for doc values	-	0.043689728	MB
Heap used for terms	-	13.85075188	MB
Heap used for norms	-	0.077697754	MB
Heap used for points	-	0.266856194	MB
Heap used for stored fields	-	0.792114258	MB
Segment count	-	99	-
Min Throughput	index-append	92446.94	docs/s
Median Throughput	index-append	92935.55	docs/s
Max Throughput	index-append	93217.68	docs/s
50th percentile latency	index-append	176.7329985	ms
90th percentile latency	index-append	285.5450693	ms
100th percentile latency	index-append	333.228537	ms
50th percentile service time	index-append	176.7329985	ms
90th percentile service time	index-append	285.5450693	ms
100th percentile service time	index-append	333.228537	ms
error rate	index-append	0	%

Metric	Task	Value	Unit
Min Throughput	index-stats	90.04	ops/s
Median Throughput	index-stats	90.06	ops/s
Max Throughput	index-stats	90.11	ops/s
50th percentile latency	index-stats	3.6713165	ms
90th percentile latency	index-stats	3.919960223	ms
99th percentile latency	index-stats	4.500246093	ms
99.9th percentile latency	index-stats	20.14171663	ms
100th percentile latency	index-stats	21.36778278	ms
50th percentile service time	index-stats	3.604376499	ms
90th percentile service time	index-stats	3.8517339	ms
99th percentile service time	index-stats	4.36148177	ms
99.9th percentile service time	index-stats	20.0748024	ms
100th percentile service time	index-stats	21.300971	ms
error rate	index-stats	0	%
Min Throughput	node-stats	90.05	ops/s
Median Throughput	node-stats	90.09	ops/s
Max Throughput	node-stats	90.32	ops/s
50th percentile latency	node-stats	4.056046	ms
90th percentile latency	node-stats	4.256959922	ms
99th percentile latency	node-stats	7.993649534	ms
99.9th percentile latency	node-stats	15.0162469	ms

Metric	Task	Value	Unit
100th percentile latency	node-stats	18.79192022	ms
50th percentile service time	node-stats	3.989104	ms
90th percentile service time	node-stats	4.1902188	ms
99th percentile service time	node-stats	7.39785926	ms
99.9th percentile service time	node-stats	14.95028028	ms
100th percentile service time	node-stats	15.226284	ms
error rate	node-stats	0	%
Min Throughput	default	50.03	ops/s
Median Throughput	default	50.04	ops/s
Max Throughput	default	50.09	ops/s
50th percentile latency	default	2.890284501	ms
90th percentile latency	default	3.054330301	ms
99th percentile latency	default	3.41013575	ms
99.9th percentile latency	default	4.536945459	ms
100th percentile latency	default	5.063877001	ms
50th percentile service time	default	2.82345	ms
90th percentile service time	default	2.987489999	ms
99th percentile service time	default	3.34539951	ms
99.9th percentile service time	default	4.466092296	ms
100th percentile service time	default	4.996857	ms
error rate	default	0	%

Metric	Task	Value	Unit
Min Throughput	term	150.06	ops/s
Median Throughput	term	150.09	ops/s
Max Throughput	term	150.14	ops/s
50th percentile latency	term	2.822069666	ms
90th percentile latency	term	2.927460233	ms
99th percentile latency	term	3.585279107	ms
99.9th percentile latency	term	9.586351776	ms
100th percentile latency	term	13.36534567	ms
50th percentile service time	term	2.755832	ms
90th percentile service time	term	2.8613018	ms
99th percentile service time	term	3.4037467	ms
99.9th percentile service time	term	4.571924473	ms
100th percentile service time	term	13.301659	ms
error rate	term	0	%
Min Throughput	phrase	149.99	ops/s
Median Throughput	phrase	150.07	ops/s
Max Throughput	phrase	150.13	ops/s
50th percentile latency	phrase	3.207932333	ms
90th percentile latency	phrase	3.514073	ms
99th percentile latency	phrase	26.65015757	ms
99.9th percentile latency	phrase	38.92041855	ms

Metric	Task	Value	Unit
100th percentile latency	phrase	40.044182	ms
50th percentile service time	phrase	3.1409695	ms
90th percentile service time	phrase	3.3666699	ms
99th percentile service time	phrase	9.39342965	ms
99.9th percentile service time	phrase	18.80974216	ms
100th percentile service time	phrase	21.417291	ms
error rate	phrase	0	%
Min Throughput	country_agg_uncached	4.01	ops/s
Median Throughput	country_agg_uncached	4.01	ops/s
Max Throughput	country_agg_uncached	4.01	ops/s
50th percentile latency	country_agg_uncached	153.726532	ms
90th percentile latency	country_agg_uncached	156.0977097	ms
99th percentile latency	country_agg_uncached	167.696362	ms
100th percentile latency	country_agg_uncached	198.43754	ms
50th percentile service time	country_agg_uncached	153.606521	ms
90th percentile service time	country_agg_uncached	155.9869715	ms
99th percentile service time	country_agg_uncached	167.5793267	ms
100th percentile service time	country_agg_uncached	198.325432	ms
error rate	country_agg_uncached	0	%

Metric	Task	Value	Unit
Min Throughput	country_agg_cached	100.04	ops/s
Median Throughput	country_agg_cached	100.05	ops/s
Max Throughput	country_agg_cached	100.07	ops/s
50th percentile latency	country_agg_cached	2.7020445	ms
90th percentile latency	country_agg_cached	2.783604899	ms
99th percentile latency	country_agg_cached	3.03382523	ms
99.9th percentile latency	country_agg_cached	3.635769276	ms
100th percentile latency	country_agg_cached	4.106574	ms
50th percentile service time	country_agg_cached	2.6356045	ms
90th percentile service time	country_agg_cached	2.717349899	ms
99th percentile service time	country_agg_cached	2.93948264	ms
99.9th percentile service time	country_agg_cached	3.567144201	ms
100th percentile service time	country_agg_cached	4.039871999	ms
error rate	country_agg_cached	0	%
Min Throughput	scroll	20.04	pages/s
Median Throughput	scroll	20.05	pages/s
Max Throughput	scroll	20.07	pages/s
50th percentile latency	scroll	421.9468245	ms
90th percentile latency	scroll	433.3017323	ms
99th percentile latency	scroll	450.0724775	ms

Metric	Task	Value	Unit
100th percentile latency	scroll	505.502723	ms
50th percentile service time	scroll	421.0948965	ms
90th percentile service time	scroll	432.4389587	ms
99th percentile service time	scroll	449.2045264	ms
100th percentile service time	scroll	504.653479	ms
error rate	scroll	0	%
Min Throughput	expression	2	ops/s
Median Throughput	expression	2	ops/s
Max Throughput	expression	2	ops/s
50th percentile latency	expression	270.920167	ms
90th percentile latency	expression	277.4334041	ms
99th percentile latency	expression	286.5631326	ms
100th percentile latency	expression	293.09254	ms
50th percentile service time	expression	270.662187	ms
90th percentile service time	expression	277.1779957	ms
99th percentile service time	expression	286.3073191	ms
100th percentile service time	expression	292.826178	ms
error rate	expression	0	%
Min Throughput	painless_static	1.5	ops/s
Median Throughput	painless_static	1.5	ops/s
Max Throughput	painless_static	1.5	ops/s
50th percentile latency	painless_static	360.9218617	ms

Metric	Task	Value	Unit
90th percentile latency	painless_static	368.2584616	ms
99th percentile latency	painless_static	382.3877013	ms
100th percentile latency	painless_static	425.989704	ms
50th percentile service time	painless_static	360.5910995	ms
90th percentile service time	painless_static	367.9205895	ms
99th percentile service time	painless_static	382.0613883	ms
100th percentile service time	painless_static	425.659728	ms
error rate	painless_static	0	%
Min Throughput	painless_dynamic	1.5	ops/s
Median Throughput	painless_dynamic	1.5	ops/s
Max Throughput	painless_dynamic	1.5	ops/s
50th percentile latency	painless_dynamic	354.4270103	ms
90th percentile latency	painless_dynamic	362.9108269	ms
99th percentile latency	painless_dynamic	409.7732626	ms
100th percentile latency	painless_dynamic	410.1049017	ms
50th percentile service time	painless_dynamic	354.0901565	ms
90th percentile service time	painless_dynamic	362.5730453	ms
99th percentile service time	painless_dynamic	409.4442952	ms
100th percentile service time	painless_dynamic	409.777646	ms

Metric	Task	Value	Unit
error rate	painless_dynamic	0	%
Min Throughput	decay_geo_gauss_function_score	1	ops/s
Median Throughput	decay_geo_gauss_function_score	1	ops/s
Max Throughput	decay_geo_gauss_function_score	1	ops/s
50th percentile latency	decay_geo_gauss_function_score	354.387216	ms
90th percentile latency	decay_geo_gauss_function_score	358.9124798	ms
99th percentile latency	decay_geo_gauss_function_score	363.9485787	ms
100th percentile latency	decay_geo_gauss_function_score	371.780245	ms
50th percentile service time	decay_geo_gauss_function_score	353.7158425	ms
90th percentile service time	decay_geo_gauss_function_score	358.2845019	ms
99th percentile service time	decay_geo_gauss_function_score	363.275623	ms
100th percentile service time	decay_geo_gauss_function_score	371.114045	ms
error rate	decay_geo_gauss_function_score	0	%
Min Throughput	decay_geo_gauss_script_score	1	ops/s
Median Throughput	decay_geo_gauss_script_score	1	ops/s
Max Throughput	decay_geo_gauss_script_score	1	ops/s
50th percentile latency	decay_geo_gauss_script_score	379.4620745	ms
90th percentile latency	decay_geo_gauss_script_score	383.2876548	ms
99th percentile latency	decay_geo_gauss_script_score	389.7544834	ms

Metric	Task	Value	Unit
100th percentile latency	decay_geo_gauss_script_score	395.75293	ms
50th percentile service time	decay_geo_gauss_script_score	378.8137045	ms
90th percentile service time	decay_geo_gauss_script_score	382.6389076	ms
99th percentile service time	decay_geo_gauss_script_score	389.1097136	ms
100th percentile service time	decay_geo_gauss_script_score	395.100654	ms
error rate	decay_geo_gauss_script_score	0	%
Min Throughput	field_value_function_score	1.5	ops/s
Median Throughput	field_value_function_score	1.5	ops/s
Max Throughput	field_value_function_score	1.51	ops/s
50th percentile latency	field_value_function_score	142.4418055	ms
90th percentile latency	field_value_function_score	146.0292471	ms
99th percentile latency	field_value_function_score	149.4448299	ms
100th percentile latency	field_value_function_score	154.4188467	ms
50th percentile service time	field_value_function_score	141.8792295	ms
90th percentile service time	field_value_function_score	145.4722711	ms
99th percentile service time	field_value_function_score	148.8731825	ms
100th percentile service time	field_value_function_score	153.87006	ms
error rate	field_value_function_score	0	%
Min Throughput	field_value_script_score	1.5	ops/s

Metric	Task	Value	Unit
Median Throughput	field_value_script_score	1.5	ops/s
Max Throughput	field_value_script_score	1.51	ops/s
50th percentile latency	field_value_script_score	200.310233	ms
90th percentile latency	field_value_script_score	206.2690364	ms
99th percentile latency	field_value_script_score	216.7453505	ms
100th percentile latency	field_value_script_score	252.6694313	ms
50th percentile service time	field_value_script_score	199.886616	ms
90th percentile service time	field_value_script_score	205.7897592	ms
99th percentile service time	field_value_script_score	216.2602712	ms
100th percentile service time	field_value_script_score	252.180659	ms
error rate	field_value_script_score	0	%
Min Throughput	random_function_score	1.5	ops/s
Median Throughput	random_function_score	1.5	ops/s
Max Throughput	random_function_score	1.5	ops/s
50th percentile latency	random_function_score	242.6018717	ms
90th percentile latency	random_function_score	251.1366288	ms
99th percentile latency	random_function_score	290.9842466	ms
100th percentile latency	random_function_score	307.5584597	ms
50th percentile service time	random_function_score	242.149128	ms

Metric	Task	Value	Unit
90th percentile service time	random_function_score	250.6830153	ms
99th percentile service time	random_function_score	290.5378949	ms
100th percentile service time	random_function_score	307.111375	ms
error rate	random_function_score	0	%
Min Throughput	random_script_score	1.5	ops/s
Median Throughput	random_script_score	1.5	ops/s
Max Throughput	random_script_score	1.5	ops/s
50th percentile latency	random_script_score	258.3288777	ms
90th percentile latency	random_script_score	262.5996219	ms
99th percentile latency	random_script_score	276.7350459	ms
100th percentile latency	random_script_score	278.8234443	ms
50th percentile service time	random_script_score	257.8902625	ms
90th percentile service time	random_script_score	262.1680452	ms
99th percentile service time	random_script_score	276.3056912	ms
100th percentile service time	random_script_score	278.384714	ms
error rate	random_script_score	0	%
Min Throughput	large_terms	1.5	ops/s
Median Throughput	large_terms	1.5	ops/s
Max Throughput	large_terms	1.5	ops/s
50th percentile latency	large_terms	429.023917	ms

Metric	Task	Value	Unit
90th percentile latency	large_terms	438.5573247	ms
99th percentile latency	large_terms	468.2661402	ms
100th percentile latency	large_terms	494.4412297	ms
50th percentile service time	large_terms	428.772941	ms
90th percentile service time	large_terms	438.29435	ms
99th percentile service time	large_terms	468.0068679	ms
100th percentile service time	large_terms	494.168992	ms
error rate	large_terms	0	%
Min Throughput	large_filtered_terms	1.5	ops/s
Median Throughput	large_filtered_terms	1.5	ops/s
Max Throughput	large_filtered_terms	1.5	ops/s
50th percentile latency	large_filtered_terms	433.0397738	ms
90th percentile latency	large_filtered_terms	443.241508	ms
99th percentile latency	large_filtered_terms	460.8045067	ms
100th percentile latency	large_filtered_terms	486.396965	ms
50th percentile service time	large_filtered_terms	432.7802525	ms
90th percentile service time	large_filtered_terms	442.9739873	ms
99th percentile service time	large_filtered_terms	460.7444745	ms
100th percentile service time	large_filtered_terms	486.145846	ms

Metric	Task	Value	Unit
error rate	large_filtered_terms	0	%
Min Throughput	large_prohibited_terms	1.5	ops/s
Median Throughput	large_prohibited_terms	1.5	ops/s
Max Throughput	large_prohibited_terms	1.5	ops/s
50th percentile latency	large_prohibited_terms	430.1467708	ms
90th percentile latency	large_prohibited_terms	436.8730103	ms
99th percentile latency	large_prohibited_terms	484.5697929	ms
100th percentile latency	large_prohibited_terms	492.75088	ms
50th percentile service time	large_prohibited_terms	429.8833325	ms
90th percentile service time	large_prohibited_terms	436.6196592	ms
99th percentile service time	large_prohibited_terms	484.3087876	ms
100th percentile service time	large_prohibited_terms	492.492977	ms
error rate	large_prohibited_terms	0	%
Min Throughput	desc_sort_population	1.5	ops/s
Median Throughput	desc_sort_population	1.51	ops/s
Max Throughput	desc_sort_population	1.51	ops/s
50th percentile latency	desc_sort_population	45.9402765	ms
90th percentile latency	desc_sort_population	49.01190953	ms
99th percentile latency	desc_sort_population	58.5120831	ms

Metric	Task	Value	Unit
100th percentile latency	desc_sort_population	60.027354	ms
50th percentile service time	desc_sort_population	45.2962825	ms
90th percentile service time	desc_sort_population	48.3757462	ms
99th percentile service time	desc_sort_population	57.86711494	ms
100th percentile service time	desc_sort_population	59.377354	ms
error rate	desc_sort_population	0	%
Min Throughput	asc_sort_population	1.5	ops/s
Median Throughput	asc_sort_population	1.51	ops/s
Max Throughput	asc_sort_population	1.51	ops/s
50th percentile latency	asc_sort_population	46.02105783	ms
90th percentile latency	asc_sort_population	48.79212977	ms
99th percentile latency	asc_sort_population	55.94577758	ms
100th percentile latency	asc_sort_population	72.898199	ms
50th percentile service time	asc_sort_population	45.37886	ms
90th percentile service time	asc_sort_population	48.1426418	ms
99th percentile service time	asc_sort_population	55.30153109	ms
100th percentile service time	asc_sort_population	72.260339	ms
error rate	asc_sort_population	0	%
Min Throughput	desc_sort_geonameid	1.5	ops/s

Metric	Task	Value	Unit
Median Throughput	desc_sort_geonameid	1.51	ops/s
Max Throughput	desc_sort_geonameid	1.51	ops/s
50th percentile latency	desc_sort_geonameid	52.22274167	ms
90th percentile latency	desc_sort_geonameid	69.4325779	ms
99th percentile latency	desc_sort_geonameid	79.57920996	ms
100th percentile latency	desc_sort_geonameid	80.11872267	ms
50th percentile service time	desc_sort_geonameid	51.6055115	ms
90th percentile service time	desc_sort_geonameid	68.801679	ms
99th percentile service time	desc_sort_geonameid	79.41158055	ms
100th percentile service time	desc_sort_geonameid	79.465491	ms
error rate	desc_sort_geonameid	0	%
Min Throughput	asc_sort_geonameid	1.5	ops/s
Median Throughput	asc_sort_geonameid	1.51	ops/s
Max Throughput	asc_sort_geonameid	1.51	ops/s
50th percentile latency	asc_sort_geonameid	51.35154333	ms
90th percentile latency	asc_sort_geonameid	52.2966503	ms
99th percentile latency	asc_sort_geonameid	55.33079961	ms
100th percentile latency	asc_sort_geonameid	55.520544	ms
50th percentile service time	asc_sort_geonameid	50.7138335	ms

Metric	Task	Value	Unit
90th percentile service time	asc_sort_geonameid	51.6588923	ms
99th percentile service time	asc_sort_geonameid	54.68967127	ms
100th percentile service time	asc_sort_geonameid	54.874135	ms
error rate	asc_sort_geonameid	0	%

11.4 Performance Comparison Between Clusters of the ess.spec-2u8g and ess.spec-4u16g Specification

The following table lists the performance comparison between the cluster with three nodes of the **ess.spec-2u8g** specification and that with three nodes of the **ess.spec-4u16g** specification.

Metric	Task	Baseline	Contender	Diff	Unit
Cumulative indexing time of primary shards	-	11.48263333	11.95073333	-0.468099997	min
Min cumulative indexing time across primary shards	-	0	0	0	min
Median cumulative indexing time across primary shards	-	2.313783333	2.339941667	-0.026158334	min
Max cumulative indexing time across primary shards	-	2.401766667	2.470116667	-0.06835	min
Cumulative indexing throttle time of primary shards	-	0	0	0	min

Metric	Task	Baseline	Contender	Diff	Unit
Min cumulative indexing throttle time across primary shards	-	0	0	0	min
Median cumulative indexing throttle time across primary shards	-	0	0	0	min
Max cumulative indexing throttle time across primary shards	-	0	0	0	min
Cumulative merge time of primary shards	-	6.466066667	4.21495	2.251116667	min
Cumulative merge count of primary shards	-	85	65	20	-
Min cumulative merge time across primary shards	-	0	0	0	min
Median cumulative merge time across primary shards	-	1.257475	0.813216667	0.444258333	min
Max cumulative merge time across primary shards	-	1.417283333	0.974483333	0.4428	min
Cumulative merge throttle time of primary shards	-	1.089583333	0.83345	0.256133333	min

Metric	Task	Baseline	Contender	Diff	Unit
Min cumulative merge throttle time across primary shards	-	0	0	0	min
Median cumulative merge throttle time across primary shards	-	0.200458333	0.157775	0.042683333	min
Max cumulative merge throttle time across primary shards	-	0.28265	0.24605	0.0366	min
Cumulative refresh time of primary shards	-	3.641266667	2.164983333	1.476283334	min
Cumulative refresh count of primary shards	-	530	291	239	-
Min cumulative refresh time across primary shards	-	0	0	0	min
Median cumulative refresh time across primary shards	-	0.725791667	0.425391667	0.3004	min
Max cumulative refresh time across primary shards	-	0.74775	0.450516667	0.297233333	min
Cumulative flush time of primary shards	-	0.3056	0.1559	0.1497	min
Cumulative flush count of primary shards	-	11	11	0	-

Metric	Task	Baseline	Contender	Diff	Unit
Min cumulative flush time across primary shards	-	0	0	0	min
Median cumulative flush time across primary shards	-	0.059858333	0.0248	0.035058333	min
Max cumulative flush time across primary shards	-	0.09155	0.043433333	0.048116667	min
Total Young Gen GC	-	11.519	6.421	5.098	s
Total Old Gen GC	-	0	0	0	s
Store size	-	3.045436038	3.124213032	-0.078776994	GB
Translog size	-	2.791873856	2.790678718	0.001195138	GB
Heap used for segments	-	15.81298065	15.03110981	0.781870842	MB
Heap used for doc values	-	0.037128448	0.043689728	-0.00656128	MB
Heap used for terms	-	14.63806534	13.85075188	0.787313458	MB
Heap used for norms	-	0.073120117	0.077697754	-0.004577637	MB
Heap used for points	-	0.272666931	0.266856194	0.005810737	MB
Heap used for stored fields	-	0.791999817	0.792114258	-0.000114441	MB
Segment count	-	95	99	-4	-
Min Throughput	index-append	41705.19	92446.94	-50741.75	docs/s
Median Throughput	index-append	46911.27	92935.55	-46024.28	docs/s

Metric	Task	Baseline	Contender	Diff	Unit
Max Throughput	index-append	47765.4	93217.68	-45452.28	docs/s
50th percentile latency	index-append	642.339781	176.732998 5	465.6067825	ms
90th percentile latency	index-append	1114.672936	285.545069 3	829.1278669	ms
99th percentile latency	index-append	1733.648438	-	1733.648438	ms
99.9th percentile latency	index-append	4770.059011	-	4770.059011	ms
100th percentile latency	index-append	7045.246771	333.228537	6712.018234	ms
50th percentile service time	index-append	642.339781	176.732998 5	465.6067825	ms
90th percentile service time	index-append	1114.672936	285.545069 3	829.1278669	ms
99th percentile service time	index-append	1733.648438	-	1733.648438	ms
99.9th percentile service time	index-append	4770.059011	-	4770.059011	ms
100th percentile service time	index-append	7045.246771	333.228537	6712.018234	ms
error rate	index-append	0	0	0	%
Min Throughput	index-stats	90.05	90.04	0.01	ops/s
Median Throughput	index-stats	90.07	90.06	0.01	ops/s
Max Throughput	index-stats	90.12	90.11	0.01	ops/s
50th percentile latency	index-stats	2.834653556	3.6713165	-0.836662944	ms
90th percentile latency	index-stats	3.527868712	3.919960223	-0.392091511	ms

Metric	Task	Baseline	Contender	Diff	Unit
99th percentile latency	index-stats	4.332674769	4.500246093	-0.167571324	ms
99.9th percentile latency	index-stats	8.392195267	20.14171663	-11.74952136	ms
100th percentile latency	index-stats	9.692270112	21.36778278	-11.67551267	ms
50th percentile service time	index-stats	2.766648	3.604376499	-0.837728499	ms
90th percentile service time	index-stats	3.448194001	3.8517339	-0.403539899	ms
99th percentile service time	index-stats	4.26309684	4.36148177	-0.09838493	ms
99.9th percentile service time	index-stats	8.322068306	20.0748024	-11.75273409	ms
100th percentile service time	index-stats	9.624071001	21.300971	-11.6769	ms
error rate	index-stats	0	0	0	%
Min Throughput	node-stats	90.06	90.05	0.01	ops/s
Median Throughput	node-stats	90.1	90.09	0.01	ops/s
Max Throughput	node-stats	90.35	90.32	0.03	ops/s
50th percentile latency	node-stats	3.205233055	4.056046	-0.850812945	ms
90th percentile latency	node-stats	3.595145422	4.256959922	-0.6618145	ms
99th percentile latency	node-stats	4.469114152	7.993649534	-3.524535382	ms
99.9th percentile latency	node-stats	8.306063762	15.0162469	-6.710183138	ms

Metric	Task	Baseline	Contender	Diff	Unit
100th percentile latency	node-stats	8.748160444	18.79192022	-10.04375978	ms
50th percentile service time	node-stats	3.1379455	3.989104	-0.8511585	ms
90th percentile service time	node-stats	3.5278055	4.1902188	-0.6624133	ms
99th percentile service time	node-stats	4.397312671	7.39785926	-3.000546589	ms
99.9th percentile service time	node-stats	8.236949997	14.95028028	-6.713330283	ms
100th percentile service time	node-stats	8.680502	15.226284	-6.545782	ms
error rate	node-stats	0	0	0	%
Min Throughput	default	50.03	50.03	0	ops/s
Median Throughput	default	50.05	50.04	0.01	ops/s
Max Throughput	default	50.09	50.09	0	ops/s
50th percentile latency	default	2.354736001	2.890284501	-0.5355485	ms
90th percentile latency	default	2.7983462	3.054330301	-0.255984101	ms
99th percentile latency	default	4.59134772	3.41013575	1.18121197	ms
99.9th percentile latency	default	13.97301623	4.536945459	9.436070774	ms
100th percentile latency	default	16.199022	5.063877001	11.135145	ms
50th percentile service time	default	2.286799	2.82345	-0.536651	ms
90th percentile service time	default	2.7289099	2.987489999	-0.258580099	ms

Metric	Task	Baseline	Contender	Diff	Unit
99th percentile service time	default	4.511846871	3.34539951	1.166447361	ms
99.9th percentile service time	default	13.90608139	4.466092296	9.439989092	ms
100th percentile service time	default	16.130242	4.996857	11.133385	ms
error rate	default	0	0	0	%
Min Throughput	term	150.07	150.06	0.01	ops/s
Median Throughput	term	150.1	150.09	0.01	ops/s
Max Throughput	term	150.15	150.14	0.01	ops/s
50th percentile latency	term	2.316147835	2.822069666	-0.505921831	ms
90th percentile latency	term	2.610932901	2.927460233	-0.316527332	ms
99th percentile latency	term	5.968978318	3.585279107	2.383699211	ms
99.9th percentile latency	term	10.37105939	9.586351776	0.784707617	ms
100th percentile latency	term	12.147341	13.36534567	-1.218004668	ms
50th percentile service time	term	2.249188999	2.755832	-0.506643001	ms
90th percentile service time	term	2.5313585	2.8613018	-0.3299433	ms
99th percentile service time	term	5.32149807	3.4037467	1.91775137	ms
99.9th percentile service time	term	9.589421289	4.571924473	5.017496816	ms

Metric	Task	Baseline	Contender	Diff	Unit
100th percentile service time	term	11.204094	13.301659	-2.097565	ms
error rate	term	0	0	0	%
Min Throughput	phrase	150.07	149.99	0.08	ops/s
Median Throughput	phrase	150.1	150.07	0.03	ops/s
Max Throughput	phrase	150.16	150.13	0.03	ops/s
50th percentile latency	phrase	2.350160666	3.207932333	-0.857771667	ms
90th percentile latency	phrase	2.689091867	3.514073	-0.824981133	ms
99th percentile latency	phrase	4.606508314	26.65015757	-22.04364926	ms
99.9th percentile latency	phrase	11.32920839	38.92041855	-27.59121016	ms
100th percentile latency	phrase	11.53972367	40.044182	-28.50445833	ms
50th percentile service time	phrase	2.283426499	3.1409695	-0.857543001	ms
90th percentile service time	phrase	2.6023857	3.3666699	-0.7642842	ms
99th percentile service time	phrase	4.073278879	9.39342965	-5.320150771	ms
99.9th percentile service time	phrase	11.26236945	18.80974216	-7.547372708	ms
100th percentile service time	phrase	11.471612	21.417291	-9.945678999	ms
error rate	phrase	0	0	0	%
Min Throughput	country_agg_uncached	4	4.01	-0.01	ops/s

Metric	Task	Baseline	Contender	Diff	Unit
Median Throughput	country_agg_uncached	4.01	4.01	0	ops/s
Max Throughput	country_agg_uncached	4.01	4.01	0	ops/s
50th percentile latency	country_agg_uncached	154.036113	153.726532	0.309581	ms
90th percentile latency	country_agg_uncached	160.160262	156.097097	4.062552299	ms
99th percentile latency	country_agg_uncached	217.9470218	167.696362	50.25065978	ms
100th percentile latency	country_agg_uncached	270.401061	198.43754	71.963521	ms
50th percentile service time	country_agg_uncached	153.9164235	153.606521	0.3099025	ms
90th percentile service time	country_agg_uncached	160.0393962	155.9869715	4.052424699	ms
99th percentile service time	country_agg_uncached	217.8203381	167.5793267	50.24101142	ms
100th percentile service time	country_agg_uncached	270.314704	198.325432	71.989272	ms
error rate	country_agg_uncached	0	0	0	%
Min Throughput	country_agg_cached	100.04	100.04	0	ops/s
Median Throughput	country_agg_cached	100.06	100.05	0.01	ops/s
Max Throughput	country_agg_cached	100.07	100.07	0	ops/s
50th percentile latency	country_agg_cached	1.772262999	2.7020445	-0.929781501	ms
90th percentile latency	country_agg_cached	1.943878399	2.783604899	-0.8397265	ms
99th percentile latency	country_agg_cached	2.796966468	3.03382523	-0.236858762	ms

Metric	Task	Baseline	Contender	Diff	Unit
99.9th percentile latency	country_agg_cached	6.427875642	3.635769276	2.792106366	ms
100th percentile latency	country_agg_cached	14.575363	4.106574	10.468789	ms
50th percentile service time	country_agg_cached	1.7050655	2.6356045	-0.930539	ms
90th percentile service time	country_agg_cached	1.878483099	2.717349899	-0.8388668	ms
99th percentile service time	country_agg_cached	2.689127631	2.93948264	-0.250355009	ms
99.9th percentile service time	country_agg_cached	4.762661218	3.567144201	1.195517017	ms
100th percentile service time	country_agg_cached	14.506126	4.039871999	10.466254	ms
error rate	country_agg_cached	0	0	0	%
Min Throughput	scroll	20.05	20.04	0.01	pages/s
Median Throughput	scroll	20.06	20.05	0.01	pages/s
Max Throughput	scroll	20.07	20.07	0	pages/s
50th percentile latency	scroll	387.0272235	421.9468245	-34.919601	ms
90th percentile latency	scroll	400.7843767	433.3017323	-32.5173556	ms
99th percentile latency	scroll	452.1627557	450.0724775	2.090278199	ms
100th percentile latency	scroll	478.26665	505.502723	-27.236073	ms
50th percentile service time	scroll	386.143462	421.0948965	-34.9514345	ms

Metric	Task	Baseline	Contender	Diff	Unit
90th percentile service time	scroll	399.8976064	432.4389587	-32.5413523	ms
99th percentile service time	scroll	451.295933	449.2045264	2.091406559	ms
100th percentile service time	scroll	477.360055	504.653479	-27.293424	ms
error rate	scroll	0	0	0	%
Min Throughput	expression	2	2	0	ops/s
Median Throughput	expression	2	2	0	ops/s
Max Throughput	expression	2	2	0	ops/s
50th percentile latency	expression	285.121047	270.920167	14.20088	ms
90th percentile latency	expression	292.0323929	277.4334041	14.5989888	ms
99th percentile latency	expression	336.1215281	286.5631326	49.55839553	ms
100th percentile latency	expression	389.221478	293.09254	96.128938	ms
50th percentile service time	expression	284.883145	270.662187	14.220958	ms
90th percentile service time	expression	291.78961	277.1779957	14.6116143	ms
99th percentile service time	expression	335.9078465	286.3073191	49.60052738	ms
100th percentile service time	expression	388.982388	292.826178	96.15621	ms
error rate	expression	0	0	0	%
Min Throughput	painless_static	1.5	1.5	0	ops/s
Median Throughput	painless_static	1.5	1.5	0	ops/s

Metric	Task	Baseline	Contender	Diff	Unit
Max Throughput	painless_static	1.5	1.5	0	ops/s
50th percentile latency	painless_static	414.4142772	360.9218617	53.49241547	ms
90th percentile latency	painless_static	428.3021712	368.2584616	60.04370963	ms
99th percentile latency	painless_static	551.0764984	382.3877013	168.6887971	ms
100th percentile latency	painless_static	586.564512	425.989704	160.574808	ms
50th percentile service time	painless_static	414.134189	360.5910995	53.5430895	ms
90th percentile service time	painless_static	428.0409987	367.9205895	60.1204092	ms
99th percentile service time	painless_static	550.7989791	382.0613883	168.7375908	ms
100th percentile service time	painless_static	586.432656	425.659728	160.772928	ms
error rate	painless_static	0	0	0	%
Min Throughput	painless_dynamic	1.5	1.5	0	ops/s
Median Throughput	painless_dynamic	1.5	1.5	0	ops/s
Max Throughput	painless_dynamic	1.5	1.5	0	ops/s
50th percentile latency	painless_dynamic	387.1022877	354.4270103	32.67527737	ms
90th percentile latency	painless_dynamic	402.260061	362.9108269	39.34923413	ms
99th percentile latency	painless_dynamic	472.1731577	409.7732626	62.39989507	ms
100th percentile latency	painless_dynamic	480.22595	410.1049017	70.1210483	ms

Metric	Task	Baseline	Contender	Diff	Unit
50th percentile service time	painless_dynamic	386.7965725	354.0901565	32.706416	ms
90th percentile service time	painless_dynamic	401.955634	362.5730453	39.3825887	ms
99th percentile service time	painless_dynamic	471.9657896	409.4442952	62.5214944	ms
100th percentile service time	painless_dynamic	479.91248	409.777646	70.134834	ms
error rate	painless_dynamic	0	0	0	%
Min Throughput	decay_geo_gauss_function_score	1	1	0	ops/s
Median Throughput	decay_geo_gauss_function_score	1	1	0	ops/s
Max Throughput	decay_geo_gauss_function_score	1	1	0	ops/s
50th percentile latency	decay_geo_gauss_function_score	364.5783855	354.387216	10.1911695	ms
90th percentile latency	decay_geo_gauss_function_score	369.3249541	358.9124798	10.4124743	ms
99th percentile latency	decay_geo_gauss_function_score	376.4548957	363.9485787	12.50631705	ms
100th percentile latency	decay_geo_gauss_function_score	402.051915	371.780245	30.27167	ms
50th percentile service time	decay_geo_gauss_function_score	364.0542175	353.7158425	10.338375	ms
90th percentile service time	decay_geo_gauss_function_score	368.6669817	358.2845019	10.3824798	ms

Metric	Task	Baseline	Contender	Diff	Unit
99th percentile service time	decay_geo_gauss_function_score	375.7975505	363.275623	12.52192747	ms
100th percentile service time	decay_geo_gauss_function_score	401.399591	371.114045	30.285546	ms
error rate	decay_geo_gauss_function_score	0	0	0	%
Min Throughput	decay_geo_gauss_script_score	1	1	0	ops/s
Median Throughput	decay_geo_gauss_script_score	1	1	0	ops/s
Max Throughput	decay_geo_gauss_script_score	1	1	0	ops/s
50th percentile latency	decay_geo_gauss_script_score	388.6800445	379.4620745	9.21797	ms
90th percentile latency	decay_geo_gauss_script_score	404.632834	383.2876548	21.3451792	ms
99th percentile latency	decay_geo_gauss_script_score	450.7542979	389.7544834	60.99981453	ms
100th percentile latency	decay_geo_gauss_script_score	538.551451	395.75293	142.798521	ms
50th percentile service time	decay_geo_gauss_script_score	388.0335405	378.8137045	9.219835999	ms
90th percentile service time	decay_geo_gauss_script_score	403.9975599	382.6389076	21.3586523	ms
99th percentile service time	decay_geo_gauss_script_score	450.1032284	389.1097136	60.99351485	ms

Metric	Task	Baseline	Contender	Diff	Unit
100th percentile service time	decay_geo_gauss_script_score	537.919936	395.100654	142.819282	ms
error rate	decay_geo_gauss_script_score	0	0	0	%
Min Throughput	field_value_function_score	1.5	1.5	0	ops/s
Median Throughput	field_value_function_score	1.5	1.5	0	ops/s
Max Throughput	field_value_function_score	1.51	1.51	0	ops/s
50th percentile latency	field_value_function_score	147.6084107	142.4418055	5.166605167	ms
90th percentile latency	field_value_function_score	161.4163745	146.0292471	15.38712737	ms
99th percentile latency	field_value_function_score	218.4858815	149.4448299	69.04105157	ms
100th percentile latency	field_value_function_score	223.5476993	154.4188467	69.12885263	ms
50th percentile service time	field_value_function_score	147.071556	141.8792295	5.1923265	ms
90th percentile service time	field_value_function_score	160.8855899	145.4722711	15.4133188	ms
99th percentile service time	field_value_function_score	217.9465422	148.8731825	69.07335967	ms
100th percentile service time	field_value_function_score	223.080105	153.87006	69.210045	ms

Metric	Task	Baseline	Contender	Diff	Unit
error rate	field_value_function_score	0	0	0	%
Min Throughput	field_value_script_score	1.5	1.5	0	ops/s
Median Throughput	field_value_script_score	1.5	1.5	0	ops/s
Max Throughput	field_value_script_score	1.51	1.51	0	ops/s
50th percentile latency	field_value_script_score	208.4922433	200.310233	8.182010333	ms
90th percentile latency	field_value_script_score	213.0348423	206.2690364	6.765805933	ms
99th percentile latency	field_value_script_score	256.5748294	216.7453505	39.82947894	ms
100th percentile latency	field_value_script_score	274.4188643	252.6694313	21.74943303	ms
50th percentile service time	field_value_script_score	208.058553	199.886616	8.171937	ms
90th percentile service time	field_value_script_score	212.5744289	205.7897592	6.784669699	ms
99th percentile service time	field_value_script_score	256.1503058	216.2602712	39.89003456	ms
100th percentile service time	field_value_script_score	274.185904	252.180659	22.005245	ms
error rate	field_value_script_score	0	0	0	%
Min Throughput	random_function_score	1.5	1.5	0	ops/s
Median Throughput	random_function_score	1.5	1.5	0	ops/s
Max Throughput	random_function_score	1.5	1.5	0	ops/s
50th percentile latency	random_function_score	244.4104887	242.6018717	1.808616967	ms

Metric	Task	Baseline	Contender	Diff	Unit
90th percentile latency	random_function_score	257.7793149	251.1366288	6.642686066	ms
99th percentile latency	random_function_score	323.8163443	290.9842466	32.83209765	ms
100th percentile latency	random_function_score	376.470245	307.5584597	68.9117853	ms
50th percentile service time	random_function_score	243.9546325	242.149128	1.8055045	ms
90th percentile service time	random_function_score	257.3440943	250.6830153	6.661078999	ms
99th percentile service time	random_function_score	323.3741708	290.5378949	32.83627594	ms
100th percentile service time	random_function_score	376.091853	307.111375	68.980478	ms
error rate	random_function_score	0	0	0	%
Min Throughput	random_script_score	1.5	1.5	0	ops/s
Median Throughput	random_script_score	1.5	1.5	0	ops/s
Max Throughput	random_script_score	1.5	1.5	0	ops/s
50th percentile latency	random_script_score	265.276135	258.3288777	6.947257301	ms
90th percentile latency	random_script_score	276.8986875	262.5996219	14.29906557	ms
99th percentile latency	random_script_score	327.6141767	276.7350459	50.87913078	ms
100th percentile latency	random_script_score	339.1401533	278.8234443	60.31670903	ms
50th percentile service time	random_script_score	264.845466	257.8902625	6.9552035	ms
90th percentile service time	random_script_score	276.4729421	262.1680452	14.3048969	ms

Metric	Task	Baseline	Contender	Diff	Unit
99th percentile service time	random_script_score	327.2584587	276.3056912	50.95276753	ms
100th percentile service time	random_script_score	338.704812	278.384714	60.320098	ms
error rate	random_script_score	0	0	0	%
Min Throughput	large_terms	1.5	1.5	0	ops/s
Median Throughput	large_terms	1.5	1.5	0	ops/s
Max Throughput	large_terms	1.5	1.5	0	ops/s
50th percentile latency	large_terms	474.347426	429.023917	45.323509	ms
90th percentile latency	large_terms	482.346874	438.5573247	43.7895493	ms
99th percentile latency	large_terms	521.4118005	468.2661402	53.14566029	ms
100th percentile latency	large_terms	529.6919453	494.4412297	35.25071563	ms
50th percentile service time	large_terms	474.1270145	428.772941	45.3540735	ms
90th percentile service time	large_terms	482.1388748	438.29435	43.8445248	ms
99th percentile service time	large_terms	521.2451771	468.0068679	53.23830923	ms
100th percentile service time	large_terms	529.479614	494.168992	35.310622	ms
error rate	large_terms	0	0	0	%
Min Throughput	large_filtered_terms	1.5	1.5	0	ops/s
Median Throughput	large_filtered_terms	1.5	1.5	0	ops/s
Max Throughput	large_filtered_terms	1.5	1.5	0	ops/s

Metric	Task	Baseline	Contender	Diff	Unit
50th percentile latency	large_filtered_terms	475.7995187	433.0397738	42.75974487	ms
90th percentile latency	large_filtered_terms	486.3646669	443.241508	43.1231589	ms
99th percentile latency	large_filtered_terms	565.6174992	460.8045067	104.8129925	ms
100th percentile latency	large_filtered_terms	585.669044	486.396965	99.272079	ms
50th percentile service time	large_filtered_terms	475.580755	432.7802525	42.8005025	ms
90th percentile service time	large_filtered_terms	486.1421912	442.9739873	43.1682039	ms
99th percentile service time	large_filtered_terms	565.483224	460.7444745	104.7387495	ms
100th percentile service time	large_filtered_terms	585.452311	486.145846	99.306465	ms
error rate	large_filtered_terms	0	0	0	%
Min Throughput	large_prohibited_terms	1.5	1.5	0	ops/s
Median Throughput	large_prohibited_terms	1.5	1.5	0	ops/s
Max Throughput	large_prohibited_terms	1.5	1.5	0	ops/s
50th percentile latency	large_prohibited_terms	474.8867557	430.1467708	44.73998487	ms
90th percentile latency	large_prohibited_terms	483.007269	436.8730103	46.13425867	ms
99th percentile latency	large_prohibited_terms	540.355679	484.5697929	55.78588612	ms
100th percentile latency	large_prohibited_terms	574.8374467	492.75088	82.08656667	ms
50th percentile service time	large_prohibited_terms	474.6650815	429.8833325	44.781749	ms

Metric	Task	Baseline	Contender	Diff	Unit
90th percentile service time	large_prohibited_terms	482.7923966	436.6196592	46.1727374	ms
99th percentile service time	large_prohibited_terms	540.1352455	484.3087876	55.82645786	ms
100th percentile service time	large_prohibited_terms	574.674312	492.492977	82.181335	ms
error rate	large_prohibited_terms	0	0	0	%
Min Throughput	desc_sort_population	1.5	1.5	0	ops/s
Median Throughput	desc_sort_population	1.51	1.51	0	ops/s
Max Throughput	desc_sort_population	1.51	1.51	0	ops/s
50th percentile latency	desc_sort_population	49.97947483	45.9402765	4.039198334	ms
90th percentile latency	desc_sort_population	52.97220567	49.01190953	3.960296137	ms
99th percentile latency	desc_sort_population	65.81446927	58.5120831	7.30238617	ms
100th percentile latency	desc_sort_population	68.243857	60.027354	8.216503	ms
50th percentile service time	desc_sort_population	49.3373975	45.2962825	4.041115	ms
90th percentile service time	desc_sort_population	52.3443909	48.3757462	3.968644701	ms
99th percentile service time	desc_sort_population	65.17446437	57.86711494	7.307349431	ms
100th percentile service time	desc_sort_population	67.595051	59.377354	8.217697001	ms
error rate	desc_sort_population	0	0	0	%
Min Throughput	asc_sort_population	1.5	1.5	0	ops/s

Metric	Task	Baseline	Contender	Diff	Unit
Median Throughput	asc_sort_population	1.51	1.51	0	ops/s
Max Throughput	asc_sort_population	1.51	1.51	0	ops/s
50th percentile latency	asc_sort_population	50.29814734	46.02105783	4.277089506	ms
90th percentile latency	asc_sort_population	54.12596357	48.79212977	5.333833798	ms
99th percentile latency	asc_sort_population	57.9221302	55.94577758	1.976352622	ms
100th percentile latency	asc_sort_population	69.35533	72.898199	-3.542868999	ms
50th percentile service time	asc_sort_population	49.667352	45.37886	4.288492001	ms
90th percentile service time	asc_sort_population	53.4878858	48.1426418	5.345244	ms
99th percentile service time	asc_sort_population	57.2779194	55.30153109	1.976388311	ms
100th percentile service time	asc_sort_population	68.714241	72.260339	-3.546097999	ms
error rate	asc_sort_population	0	0	0	%
Min Throughput	desc_sort_geonameid	1.5	1.5	0	ops/s
Median Throughput	desc_sort_geonameid	1.51	1.51	0	ops/s
Max Throughput	desc_sort_geonameid	1.51	1.51	0	ops/s
50th percentile latency	desc_sort_geonameid	49.2601545	52.22274167	-2.96258717	ms
90th percentile latency	desc_sort_geonameid	53.48767223	69.4325779	-15.94490567	ms
99th percentile latency	desc_sort_geonameid	69.43293772	79.57920996	-10.14627224	ms

Metric	Task	Baseline	Contender	Diff	Unit
100th percentile latency	desc_sort_geonameid	72.512932	80.11872267	-7.60579067	ms
50th percentile service time	desc_sort_geonameid	48.6107425	51.6055115	-2.994769001	ms
90th percentile service time	desc_sort_geonameid	52.839748	68.801679	-15.961931	ms
99th percentile service time	desc_sort_geonameid	68.79282147	79.41158055	-10.61875908	ms
100th percentile service time	desc_sort_geonameid	71.872758	79.465491	-7.592732999	ms
error rate	desc_sort_geonameid	0	0	0	%
Min Throughput	asc_sort_geonameid	1.5	1.5	0	ops/s
Median Throughput	asc_sort_geonameid	1.51	1.51	0	ops/s
Max Throughput	asc_sort_geonameid	1.51	1.51	0	ops/s
50th percentile latency	asc_sort_geonameid	47.071104	51.35154333	-4.280439331	ms
90th percentile latency	asc_sort_geonameid	50.264151	52.2966503	-2.0324993	ms
99th percentile latency	asc_sort_geonameid	57.9888054	55.33079961	2.658005793	ms
100th percentile latency	asc_sort_geonameid	96.39665433	55.520544	40.87611033	ms
50th percentile service time	asc_sort_geonameid	46.427649	50.7138335	-4.286184501	ms
90th percentile service time	asc_sort_geonameid	49.6192723	51.6588923	-2.039619999	ms
99th percentile service time	asc_sort_geonameid	57.75922607	54.68967127	3.069554799	ms
100th percentile service time	asc_sort_geonameid	95.751176	54.874135	40.877041	ms

Metric	Task	Baseline	Contender	Diff	Unit
error rate	asc_sort_geonameid	0	0	0	%

12 Quotas

CSS uses the following resources:

- Instance
- CPU
- Memory (GB)
- Disk quantity
- Disk size (GB)

For details about how to view and modify the quota, see **Quotas**.

13 Related Services

Figure 13-1 shows the relationships between CSS and other services.

Figure 13-1 Relationships between CSS and other services

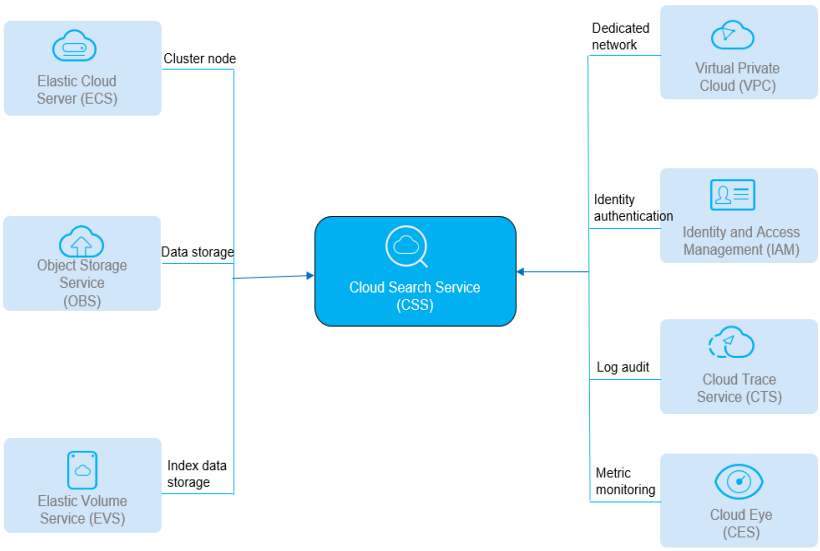


Table 13-1 Relationships between CSS and other services

Service	Description
Virtual Private Cloud (VPC)	CSS clusters are created in the subnets of a VPC. VPCs provide a secure, isolated, and logical network environment for your clusters. For details, see **Creating a VPC User Guide**.
Elastic Cloud Server (ECS)	In a CSS cluster, each node represents an ECS. When you create a cluster, ECSs are automatically created.
Elastic Volume Service (EVS)	CSS uses EVS to store index data. When you create a cluster, EVSs are automatically created for cluster data storage.

Service	Description
Object Storage Service (OBS)	Snapshots of CSS clusters are stored in OBS buckets. For details, see **Object Storage Service User Guide**.
Identity and Access Management (IAM)	IAM authenticates access to CSS. For details, see **Identity and Access Management User Guide**.
Cloud Eye	CSS uses Cloud Eye to monitor cluster metrics in real time. The supported CSS metrics include the disk usage and cluster health status. You can learn about the disk usage of the cluster based on the disk usage metric. You can learn about the health status of a cluster based on the cluster health status metric. For details, see **Cloud Eye User Guide**.
Cloud Trace Service (CTS)	With CTS, you can record operations associated with CSS for query, audit, and backtracking operations. For details, see **Cloud Trace Service Guide**.

14 Basic Concepts

Cluster

CSS provides functions on a per cluster basis. A cluster represents an independent search service that consists of multiple nodes.

Index

An index stores Elasticsearch data. It is a logical space in which one or more shards are grouped.

Shard

An index can potentially store a large amount of data that can exceed the hardware limits of a single node. To solve this problem, Elasticsearch provides the ability to subdivide your index into multiple pieces called shards. When you create an index, you can simply define the number of shards that you want. Each shard is in itself a fully-functional and independent "index" that can be hosted on any node in the cluster.

You need to specify the number of shards before creating an index and cannot change the number after the index is successfully created.

Replica

A replica is a copy of the actual storage index in a shard. It can be understood as a backup of the shard. Replicas help prevent single point of failures (SPOFs). You can increase or decrease the number of replicas based on your service requirements.

Document

An entity for Elasticsearch storage. Equivalent to the row in the RDB, the document is the basic unit that can be indexed.

Document Type

Similar to a table in the RDB, type is used to distinguish between different data.

In versions earlier than Elasticsearch 7.*x*, each index can contain multiple document types. Elasticsearch defines a type for each document.

Elasticsearch 7.*x* and later versions only support documents of the .doc type.

Mapping

A mapping is used to restrict the type of a field and can be automatically created based on data. It is similar to the schema in the database.

Field

The field is the minimum unit of a document. It is similar to the column in the database.

15 Change History

Released On	What's New
2023-12-26	Updated **Scenarios**
2023-08-29	• Deleted the billing description. • Updated CSS system permission dependency in **Permissions Management**.
2023-05-29	Added the description of the enhanced feature in **Advantages**.
2023-05-29	Added **Specifications**.
2023-03-21	Added the description about yearly/monthly billing mode in the "Billing" section.
2023-02-02	Modified the product introduction outline based on user experience.
2022-10-30	Added **Security**.
2022-06-30	• Added the description about Elasticsearch 7.10.2 and deleted the description about Elasticsearch 7.9.3. • Optimized **Permissions Management**.
2022-06-09	Added **Notice on Apache Log4j Remote Code Execution Vulnerability (CVE-2021-44228)**.
2022-04-02	Added the application scenario of vector search.
2021-10-12	Deleted the description about Logstash clusters.
2021-03-02	Added the description about Logstash clusters.
2021-01-30	Added the description about cluster version 7.9.3.
2020-08-25	Supported fine-grained authorization in **Permissions Management**.
2019-09-12	Added the introduction about Cerebro.

Released On	What's New
2019-04-30	This issue is the first official release.

