面向算力网络的跨域数据管理方法

鲁蔚征[1]，戴奇志[2]，张策[3]
1. 中国人民大学大型科学仪器共享平台，北京 100083；
2. 联旌智能科技（上海）有限公司，上海 200051；
3. 华中科技大学网络与计算中心，湖北 武汉 430074

摘要
跨域算力网络希望整合多个算力中心的计算和数据资源，但现有的方案对跨域文件和数据管理关注不够。提出了一种轻量级的跨域算力网络数据管理方案：通过文件系统协议转换，接入远程算力中心的并行文件系统存储资源；算力中心内部的存储资源作为一种补充，应对高IOPS应用；通过容器绑定技术，将远程存储挂载并绑定到指定目录。基于该方案的原型系统已经在高校校级计算平台部署运行。实测数据和用户体验显示，该方案能够满足常见高性能计算应用需求。

关键词
算力网络；并行文件系统；数据管理；异构存储资源

中图分类号：TP316 文献标志码：A doi: 10.11959/j.issn.2096-0271.2023068

Cross-domain data management for computing power networks

LU Weizheng[1], DAI Qizhi[2], ZHANG Ce[3]
1. Office of Research Infrastructure, Renmin University of China, Beijing 100872, China
2. Lianjingzhineng Technology (Shanghai) Co., Ltd., Shanghai 200051, China
3. Network and Computing Center, Huazhong University of Science & Technology, Wuhan 430074, China

Abstract
Cross-domain computing power networks wish to integrate computational and data resources from multiple computing centers, but existing methods do not pay enough attention to cross-domain file and data management. In this paper, a lightweight data access scheme for cross-domain computing power networks was proposed: (1) accessing parallel file system storage resources of remote computing centers through file system protocol conversion; (2) local caching as a supplement to cope with high IOPS applications; and (3) mounting remote or local storage to specified directories through container binding technology. The prototype system based on this scheme had been deployed on high-performance computing centers in multiple universities. The measured data and user experience showed that the scheme in this paper could meet the requirements of common high-performance computing applications.

Key words
computing power network, parallel file system, data management, heterogeneous storage resource

0 引言

近年来，高性能计算应用场景不断增多，算力和数据需求出现爆炸性增长。传统的单机房或单集群的高性能计算中心已经无法满足飞速增长的计算和存储需求，跨域算力网络应运而生。跨域算力网络指在跨局域网或者跨广域网的不同算力中心或机房建设多套算力基础设施，多套算力基础设施通过管理软件为用户提供算力和数据服务。理论上，跨域算力网络解决了算力扩容问题，在大多数实际运营场景下，用户需要在多个相互独立的算力中心内分别注册账户、安装软件并上传数据，用户要适配不同算力中心提供的调度系统等软件服务。因此，整合多个中心的算力资源，向用户提供高效易用的服务仍然有诸多挑战。第一，系统必须方便易用。从用户使用的角度考虑，跨域算力网络应为用户提供一个统一的登录入口，用户能够在一个算力中心访问其他算力中心的文件系统，这就需要在用户认证、文件系统和数据管理等诸多层面对基础软件进行优化。第二，系统必须兼容并包。不同算力中心的技术栈千差万别，尤其是文件系统，几乎没有一套通用方案能够适配所有潜在场景。第三，系统必须保证性能。从用户的角度，高性能计算本身对性能要求高，在易用性和兼容性都满足的前提下，系统的性能必须足够好；从管理的角度，各个设备的利用率必须足够高。因此，需要在跨域算力基础设施上设计并部署统一的管理软件，该管理软件需要提供用户管理、网络连通、软硬件管理和数据管理的功能，让用户尽量无感知地调用多个数据中心的算力资源。

针对算力分散问题，各国自20世纪90年代就开始了探索[1]，例如我国的中国国家网格（CNGrid）项目研发了作业调度系统[2-3]和虚拟数据空间[4-6]，一些企业也提出了轻量级算力整合方案[7]，然而这些系统更关注算力资源在多个中心的调度问题，对跨域文件和数据管理关注甚少，这些方案在文件访问的易用性和多中心兼容性方面都有待提升。

本文提出了一种轻量级的跨域算力网络解决方案：高性能计算环境即服务（environment as a service，EaaS），该方案可以兼容不同算力中心的文件系统技术栈，实现了跨域文件访问和数据管理，用户可以像访问本地文件一样无感知地跨域访问其他算力中心的文件和数据。笔者团队在中国人民大学和华中科技大学搭建了EaaS原型系统，已经稳定运行两年以上，从用户反馈和性能实测来看，EaaS有效汇聚了跨中心多机房的算力资源，能够兼容不同的网络和文件系统环境，改善了用户体验，提高了资源利用率。

本文的主要贡献如下：

- 提出了一种跨域算力网络数据方案，针对高性能计算场景，开发或适配了相应的系统软件；
- 针对不同算力中心的文件系统，提出了跨域文件访问的解决方案，不同算力中心可互相挂载和访问远程的并行文件系统；
- 基于容器的绑定技术，向用户提供精心设计的目录结构，对用户屏蔽了下层文件系统差异，让用户无感知地完成跨域文件和数据操作。

1 研究背景

在算力基础设施中，一个算力中心或一个数据中心机房内部署着计算、网络和存储资源。管理员可以任意修改和重构中心内的局域网、计算和存储设备。

1.1 单中心高性能计算环境

高性能计算环境将用户与高性能计算集群相连，并对用户提供算力和存储资源，具体包括作业调度队列、个人空间目录、应用软件和数据，绝大多数高性能计算环境是单中心的集中式系统[1,8]。单中心的高性能计算资源通常由CPU节点和GPU节点提供。算力中心内一般设计多个相互隔离的网络，例如普通局域网络和高速网络，普通局域网络用于常见业务互联或节点管理，速度在1~100 Gbit/s，一般使用以太网；高速网络用于并行计算和数据读取，速度在40~200 Gbit/s，常见高速网络技术有InfiniBand、Omini-Path、RoCE（RDMA over Converged Ethernet）等[8]。存储部分由Lustre[9]或GPFS[10]等并行文件系统管理存储资源。所有计算节点上部署有并行文件系统客户端，并行文件系统客户端通过高速网络连接至文件系统服务器（元数据服务器、对象存储服务器），并行文件系统可以像本地文件系统一样挂载至计算节点，并行文件系统为用户提供了统一命名空间和POSIX（portable operating system interface）语义。**图1**展示了一个典型的单中心高性能计算集群的文件系统和数据管理架构：元数据服务器（metadata server，MDS）管理元数据目标（metadata target，MDT），MDT内存储着文件名、目录、访问权限等；对象存储服务器（object storage server，OSS）管理对象存储目标（object storage target，OST），OST内存储着数据本身，元数据服务器和对象存储服务器又被称为I/O节点。随着数据量的增多，元数据服务和对象存储服务均可以横向扩展到多个。由于并行文件系统提供了POSIX语义和统一命名空间，用户一般不需要关心底层文件系统的技术与实现，在任何节点上都可以使用POSIX语义读写某个路径下的文件。管理员或者用户将所需计算软件安装至某路径下。

用户一般使用作业调度软件（例如，SLURM[7]）将计算任务提交至计算节点。**图2**展示了SLURM组件与架构，其他作业调度软件与之类似。可以简单地认为一套SLURM集群由一个slurmctld和若干个slurmd组成：slurmctld守护进程运行在管理节点，负责整个集群的管理和调度；slurmd守护进程运行在计算节点，负责接受slurmctld分配的任务并执行；slurmdbd守护进程负责管理数据库，它记录着用户作业和机时信息。用户使用SLURM提供的客户端命令与SLURM集群交互。

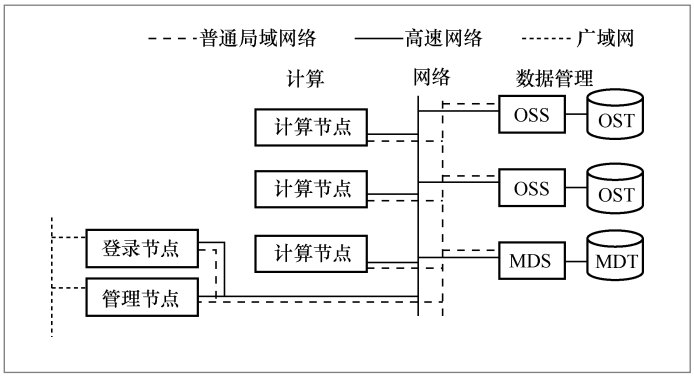

图1 单中心高性能计算集群系统架构

1.2 多中心高性能计算环境

（1）网格计算

20世纪90年代中期，网格计算（grid computing）的概念在美国兴起[1]，在网格计算的框架下，高性能计算环境应该将多个高性能计算中心的算力设施相连，并给用户提供一个集成的科研环境。一个典型的案例是CNGrid项目，CNGrid项目专门研发了CNGrid GOS、CNGrid Suite、

CNGrid SCE等系统软件，聚合了全国20个高性能计算中心的资源，实现了资源的互联互通和统一共享[2-3]。CNGrid分为运营中心和各个算力中心，运营中心部署了中心服务器和监控服务器，并将多个算力中心的高性能计算资源进行了整合，各个算力中心部署前置服务器与运营中心同步信息和数据。国家高性能计算环境虚拟数据空间系统[4-6]在CNGrid上实现了广域网上的异构存储资源的统一访问，形成了统一的虚拟数据空间。通过试用发现，CNGrid为用户提供了运营中心客户端，用户首先登录该操作客户端节点，在该客户端节点执行文件上传、作业提交等命令，将数据和计算任务提交至不同算力中心。例如，用户先将数据上传至客户端节点，再通过sceput2和sceget2等命令，将数据上传或下载到指定算力中心；在客户端节点上再通过bsub命令，提交作业到指定算力中心；其中sceput2和sceget2命令来自CNGrid SCE系统软件套件。该方案在一定程度上解决了算力跨域问题，但对用户来说，仍无法轻松访问不同算力中心的数据资源。此外，CNGrid系统软件体系复杂，且主要面向国家超级计算中心，中小型算力中心难以直接享受该方案的成果[4-6]。

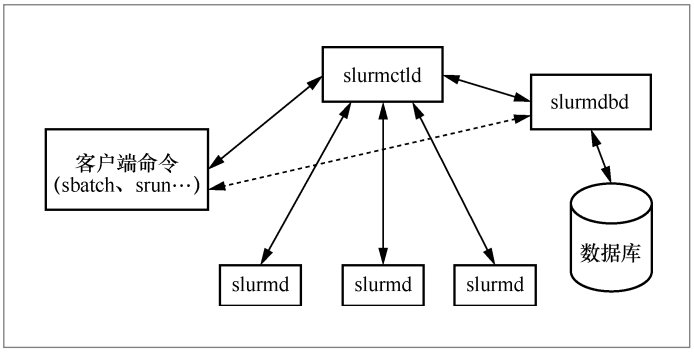

图 2 SLURM 组件与架构

（2）多集群与弹性上云

另一种更加轻量级的跨中心算力管理模式为联邦模式，如**图3（a）**的SLURM多集群和**图3（b）**的SLURM弹性上云所示，这种模式更适合自建高性能计算服务的企业和机构。对于**图3（a）**中的SLURM多集群模式，每套集群由管理节点的slurmctld向客户端提供服务，只需配置客户端能访问新增集群的slurmctld服务即可，同一

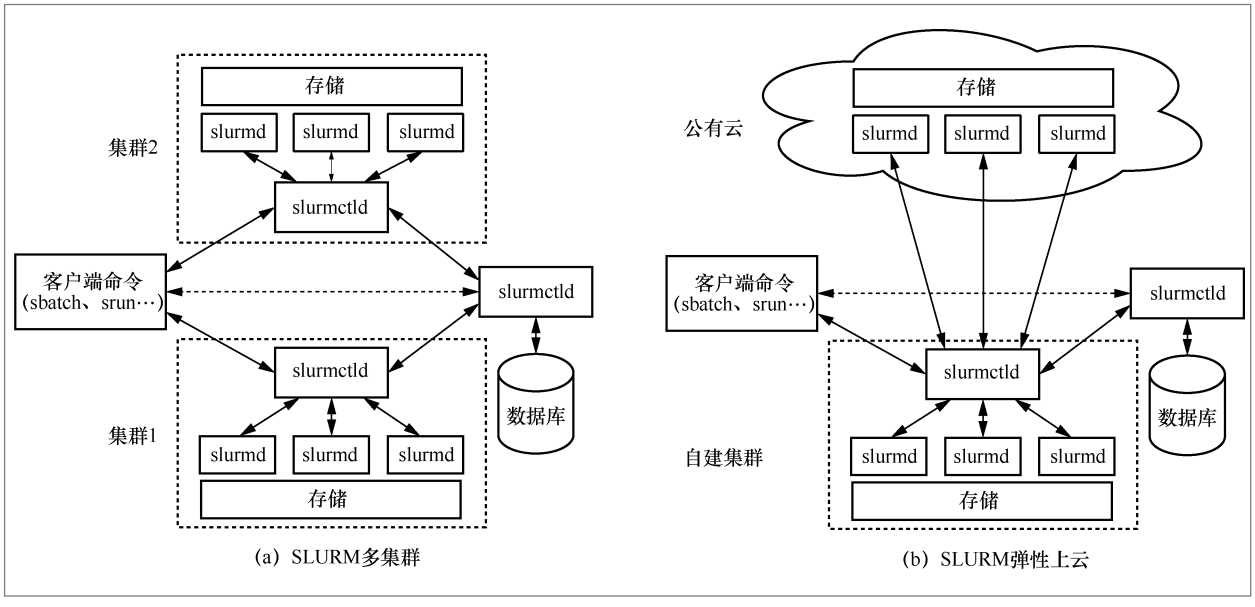

图 3 SLURM 多集群与弹性上云

个SLURM客户端可向不同的集群提交任务。对于**图3(b)**中的SLURM弹性上云模式，云上资源的slurmd被自建集群的slurmctld管理和调度。不管是多集群模式还是弹性上云模式，管理和运维相对简单，只需要跨域节点间开放对应服务的端口访问权限，例如slurmctld默认使用6817，slurmdbd默认使用6818，只要不同算力中心的守护进程通过TCP/IP访问对应端口号即可。需要注意的是，多集群模式虽然实现了统一的调度管理，但多个集群的存储是相互隔离的，用户对数据的访问是相互割裂的。例如，**图3(a)**中集群1和集群2需要分别建立一套存储，文件系统命名空间并不是统一的，需要用户和管理员手动或半自动地搬运数据。常用的搬运工具有scp、rsync等软件。

1.3 跨域异构文件系统

美国、欧洲以及我国一直以来非常重视跨域科学数据管理，试图将分散自治的计算和科学设施互联并共享。例如美国的TeraGrid[11]、XSEDE[12]和欧洲的EGI[13]：TeraGrid需使用专用网络；EGI缺乏全局统一能力；XSEDE采用松散的顶层元数据组织实现了异构存储资源的聚合，但顶层元数据组织极易成为性能瓶颈。Lustre-WAN[14]横跨3 500 km，提供Lustre文件系统服务，但该系统基于专用的100 Gbit/s广域网。CalvinFS[15]对元数据进行了横向切分，可以无限扩展，该系统对单文件进行了大量优化，但在多文件上的读写操作延迟和吞吐都相比传统文件系统高很多。Spanner[16]和PolarFS[17]提供了跨云的文件系统，但这些文件系统主要服务于云数据库，而非高性能计算工作负载。安全外壳协议文件系统(secure shell filesystem，SSHFS)[18]利用SSH协议，可挂载远程文件系统。OneData[19]提供了一种跨广域网的文件访问方式，客户端使用用户态文件系统(filesystem in user space，FUSE)挂载文件系统，数据分散在多个算力中心的存储上，每个算力中心部署独立的数据管理组件，并使用数据库存储文件的逻辑组件到物理位置之间的映射关系。

1.4 跨域数据管理关键问题

前文讨论了如何实现跨中心的算力互联互通，但在此基础上，针对文件系统共享或者跨域数据管理因为具有如下的特点，值得单独讨论。

(1)必要性

在构建算力系统及任务调度的时候，出于性能因素考虑，通常不会把紧耦合的任务(例如一个MPI任务)分拆到多个算力中心，因此高速网络远程直接内存访问(remote direct memory access，RDMA)不跨中心是合理的、可接受的设计。但另一方面，如果用户的数据无法做到跨算力数据中心的可访问，算力网络概念本身也就失去了绝大部分价值和意义，甚至无法被称为一张网。

(2)依赖性

通常在算力中心建设时，高速网络和普通局域网络会采用相对独立的两套组网，甚至采用完全不同的硬件设备和网络协议(例如，高速网络使用InfiniBand，普通局域网络使用以太网)，两者之间几乎不存在耦合和影响。但文件系统的访问很少单独拥有一套网络，这就导致了文件系统的流量要么运行在高速网络上，要么运行在普通局域网络上，进而导致文件系统与管理系统或计算系统产生了带宽竞争，而文件系统访问时产生的带宽占用可能非常大，在跨算力中心互连带宽受限的情况下显然需要特殊考虑。

(3)局域性

为了提升文件系统的访问性能，几乎所有成熟的文件系统都提供了RDMA的访问方式，通常在单算力中心内部也优先或者仅提供在RDMA上对文件系统访问的能力。另外，跨中心的互联依赖于IP网络，但RDMA协议与IP无法直接互通，在一个算力中心要实现对另一个算力中心的文件系统访问，必然要解决协议转换的问题。

(4)重要性

文件系统服务与管理类服务最大的区别在于，管理类服务的网络流量通常不大、带宽要求低，而随着大数据和人工智能类应用数据量的爆炸式增长以及数据密集型计算模式的兴起，即使在算力中心内部，文件系统访问的带宽和延迟也日渐成为制约整体性能的首要因素，程序瓶颈出现在数据的输入和输出阶段。因此如果只是简单地实现算力中心之间的文件互访，却无法提供足够低的延迟和足够高的读写带宽，则跨域算力网络的价值也大打折扣。这就要求我们必须根据不同类型应用的文件读写特性，借助缓存、镜像、目录结构组织等手段优化对文件系统的访问。

2 系统设计

2.1 总体设计

本文提出的跨域算力网络EaaS能够整合多个算力中心的算力资源，即使多个算力中心的网络情况和文件系统等技术栈差异较大，仍能兼容并包。**图4**列出了不同算力中心潜在的技术差异。网络层面，算力中心2有前置防火墙，算力中心4使用NAT网关，这两个算力中心与外界通信困难。数据管理层面，算力中心2和算力中心3均建有高可靠存储，用户的主要数据可挂载在算力中心内的存储上；算力中心4无自建存储，持久化数据可使用其他算力中心提供的存储。如果算力中心4的主要用户对I/O要求高，可使用存储服务器搭建网络文件系统(network file system，NFS)服务，

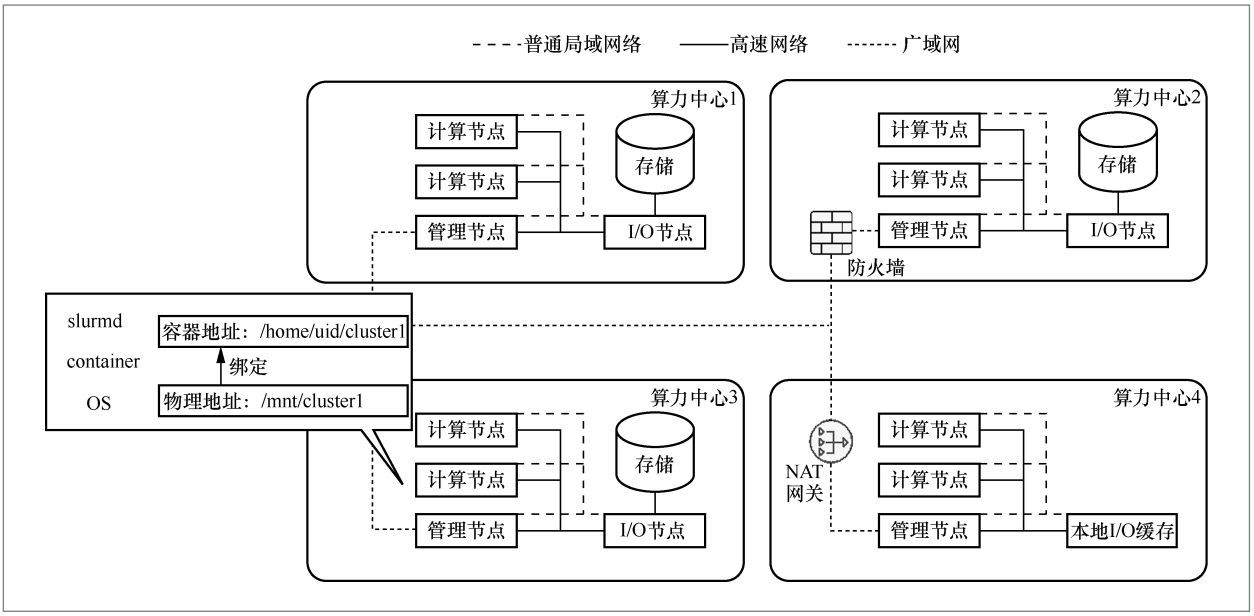

图 4 高性能计算环境 EaaS 总体设计

以较低的价格提供一个算力中心4的本地缓存。

系统有以下创新。

● 通用。系统能兼容多种存储环境，整合不同并行文件系统（Lustre、GPFS等）。

● 性能。采用跨域存储与本地存储相结合的方式。本地存储应对每秒输入输出操作数（input/output operations per second，IOPS）负载，可以是高价值的存储集群或磁盘阵列，也可以是廉价的单台存储服务器。

● 用户无感知。通过容器挂载技术，将主机目录挂载到特定路径下，用户可以无感知地读写跨域文件系统。管理员设计目录结构，进行公共软件安装和共享数据集管理。

2.2 用户体验

系统将不同算力中心的文件系统打通，协助用户进行跨域数据管理。以**图4**所示架构为例，假如用户登录算力中心1，系统将其他算力中心（算力中心2、3、4）的存储以文件形式挂载到算力中心1。算力中心2、3、4存储的挂载路径分别为/cluster2、/cluster3和/cluster4。用户在算力中心1内像访问本地存储一样，直接向/cluster2路径下读写数据，数据将被持久化到算力中心2的存储上。经过跨域数据管理后，可以实现以下效果。

● 算力整合。用户可以自由地在多个算力中心内切换，根据不同算力中心的计算资源利用情况，选择计算资源充裕的算力中心提交作业，减少资源闲置。

● 文件访问。用户登录一个算力中心即可访问多个中心的所有文件和数据。用户可以根据自身工作负载进行数据管理：可以运行时直接将输出数据读写到目标算力中心的路径下，也可以在计算任务完成后像在本地文件系统复制数据那样进行跨中心数据复制。文件的读写支持POSIX协议。

● 数据管理。常用软件和共享数据集只需要在一个主算力中心内部署，然后在多个算力中心共享，不需要在不同算力中心内反复部署，减少多个中心软件环境不一致带来的潜在问题。

表1为本文所提出的EaaS与其他现有面向高性能计算的跨域数据管理方案的对比。其中，CNGrid用户需要在操作客户端节点通过提供的sceput2、sceget2等上传下载命令，将数据上传到指定的算力中心；分散管理未使用任何管理工具，每个算力中心建设有独立的存储和调度基础设施。

3 跨域文件系统互通

3.1 协议转换

并行文件系统服务通常运行在RDMA

表1 现有跨域数据管理方案对比

方案	EaaS	CNGrid	分散管理
跨域文件操作	cp、mv、vim等文件操作命令	sceput2、sceget2等专用命令	scp、rsync等传输命令
数据管理	文件只需创建一次，并在多算力中心共享	文件需上传到多算力中心	文件需上传到多算力中心
软件管理	常用软件数据安装一次，多算力中心共享	每个算力中心安装自己的软件	每个算力中心安装自己的软件
算力调度	统一的调度软件	统一的调度软件	每个算力中心使用自己的调度软件

的局域网络上，跨域网络互联通常在IP网络上实现，要实现跨域的文件系统互访问必须解决协议转换的问题，将RDMA的流量转换封装到IP网络上，即算力中心1的计算节点能直接互联访问到算力中心2的文件系统。不同算力中心间使用的文件系统差异较大，**图1**已经提到：不同文件系统需要在计算节点上部署自己的客户端，不同文件系统的客户端之间往往互相不兼容，这给跨域访问带来了挑战。为实现跨域文件系统的互访问和互操作，本文提供了两种方案，分别适配不同算力中心的文件系统技术栈。

（1）文件协议模式

如果算力中心间使用不同的文件系统，则优先使用文件协议模式，基于NFS协议将算力中心内部的文件系统转换成运行在IP上的NFS文件系统对客户端提供服务。包括Lustre在内的常见的并行文件系统均内建了NFS协议服务能力。如**图5（a）**所示，两个算力中心之间通过隧道服务器（tunnel server）构建的物理或虚拟隧道实现了二层互通。算力中心1内部采用并行文件系统（例如Lustre）协议，算力中心1的计算节点在访问文件系统时，通过RDMA网络使用Lustre的原生客户端进行挂载。算力中心2内的计算节点在访问算力中心1的存储时，既无法直接通过RDMA网络进行连接，也无法使用并行文件系统的原生客户端。两个算力中心的隧道服务器通过TCP/IP建立隧道，数据在跨域网络上以NFS协议经由隧道互通。协议转换服务器（protocol gateway）既连接以太网络又连接RDMA网络，协议转换服务器安装有Lustre原生客户端，从而可以访问Lustre文件系统，同时作为NFS服务器端提供NFS服务，可以将运行在以太网络上的NFS的客户端的文件访问请求转换成RDMA上的Lustre文件系统的访问请求。这个方案可以说是一个万能的方案，能够解决任何场景下的文件系统互访问。

（2）网络传输模式

如果算力中心之间均使用Lustre文件系统，则可以使用网络传输模式基于LNet进行文件系统的互访问。LNet是Lustre networking的缩写，是Lustre的网络子系统，负责提供消息传递，在一个大型Lustre分布式系统中，可能存在多种类型的混合网络（例如以太网和InfiniBand等），为了使这些网络能够毫无阻碍地进

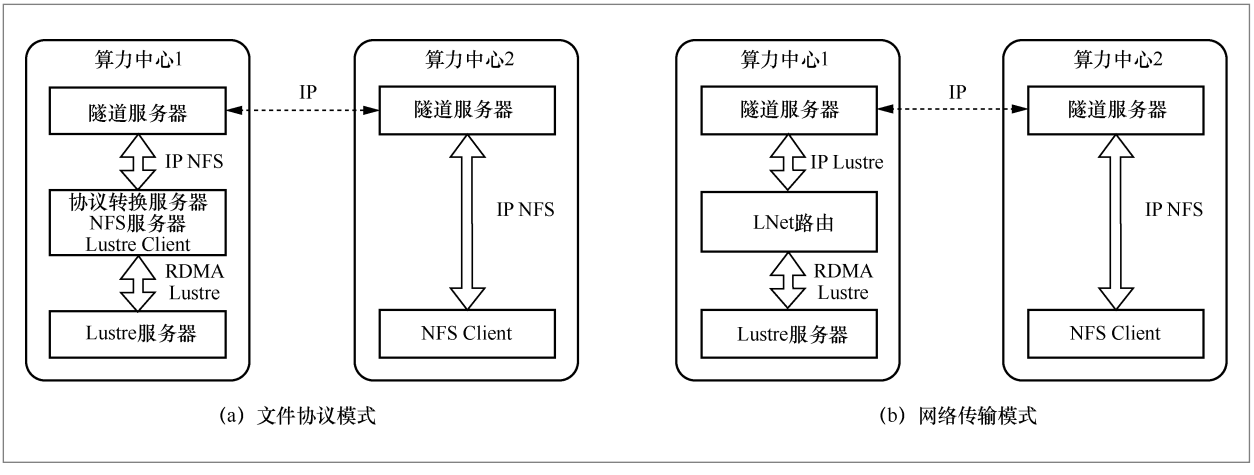

图 5 并行文件系统协议转换

行通信，LNet提供了路由（LNet router）功能。如图5（b）所示，与文件系统协议转换方式不同的是，算力中心2的计算节点在访问算力中心1的存储时，使用的依然是Lustre协议，运行在算力中心1的LNet路由可以同时访问以太网和RDMA高速网络，负责把以太网络上的Lustre客户端访问请求转换成RDMA协议，并转发至Lustre服务器端。采用LNet路由进行文件互连的方案好处在于可以保持所有的客户端的文件访问协议一致，限制在于通用性不够强，它要求所有算力中心均使用同一并行文件系统。

（3）性能及稳健性分析

文件系统访问是一个对网络传输性能极其敏感的服务，因此在承载文件系统服务的链路上要尽量提升传输效率（如避免IP切片）。同时文件访问流量是由用户操作触发或应用程序执行产生的，具有一定的不可预知性和不可控性。因此在实际应用中笔者为文件系统流量构建了一个单独的隧道，精确计算隧道协议开销并配置正确的最大传输单元（maximum transmission unit，MTU）。构建单独隧道的另一个好处是可以很方便地在该隧道上进行安全设置及流量控制等操作。

跨中心隧道的引入带来的另一个问题是网络的稳定性，与本地机房内网络设备简单可控的工作环境相比，长距离的传输以及外部物理环境的不可控会导致偶尔的网络波动或短暂中断，需要网络文件系统具备一定的容错和自我恢复能力。在实践中发现，因为NFSv3协议的无状态特性，停摆的Client端没有超时限制，使其在灾难恢复时具有极强的修复能力，甚至可以轻松应对服务端重启以及网络长时间中断的情况。Lustre也具备一定的自动恢复能力，可以应对分钟级别的网络中断，但长时间的网络中断可能会导致应用的异常退出。

（4）系统复杂度与运维成本

前文提到使用两种方式对异构存储进行管理：文件协议模式和网络传输模式。其中文件协议模式主要依赖NFS协议的通用性和兼容性，在高性能计算场景下，常见的生产可用文件系统包括：NFS、Lustre、GPFS或BeeGFS，以上文件系统均原生支持NFS协议转换，查阅文档即可实施。网络传输模式依赖Lustre文件系统提供的异构网络路由，可直接实现跨算力中心的路由。相比单算力中心，跨算力中心的数据管理确实更加复杂，但以上两种方式均基于文件系统提供功能，复杂度和运维成本均可控。

3.2 容器化目录挂载

容器提供了绑定（bind）技术，用于处理宿主机与容器的数据和目录管理的问题。本文提出的EaaS方案充分利用了容器的绑定技术，用于管理底层文件系统和上层用户所感知到的目录。通过协议转换，宿主机可以挂载不同算力中心的并行文件系统，再经过容器挂载，给用户所需的目录结构。调度软件（例如slurmd）和用户程序均运行在容器中，整个架构如图6所示。文件系统挂载与使用流程如下：

- 管理员将并行文件系统挂载至宿主机，例如基于Lustre客户端，将Lustre文件系统挂载至/mnt/cluster1/apps目录；
- 管理员将宿主机的/mnt/cluster1/apps目录通过容器绑定技术挂载至容器的/opt/app目录；
- 用户的计算工作负载在容器中，访问/opt/app获取所需文件。

管理员可以根据实际需求，确定挂载和绑定的目录结构。同一个存储资源可以被多个算力中心挂载并绑定，但并不需要

担心多个算力中心同时读写同一个文件而导致并发冲突，因为任何读写操作都将最终通过宿主机文件系统客户端发送至MDS和OSS，MDS负责一致性。用户以任意集群为入口，可以无感知地读写其他中心存储。容器挂载技术可以方便地控制物理机上的目录和容器内的目录的映射关系，从而调整容器内的目录组织，达到兼顾性能优化和管理方便的目的。

3.3 目录组织与数据管理

高性能计算的数据基本可以分为3类。第一类是平台的系统数据，例如操作系统等；第二类是公共软件和共享数据集；第三类是用户个人数据。第一类的数据通常通过安装到计算节点本地磁盘或者无盘启动的方式处理，与共享存储关系不大。下面着重讨论共享文件系统里存放的两类文件。

（1）公共软件和共享数据集

系统管理员通常会对公共软件和数据集进行集中安装和配置，一方面节省用户自己安装调试软件的时间；另一方面可以节省空间，避免同一软件被反复安装到不同目录。这部分数据的特点是更新和变化的频率不高。但大型的应用软件（如MATLAB）、存在大量小文件的应用软件（如Python和conda生态）以及大型共享数据集（如ImageNet、AlphaFold数据集）对读写性能有较高的要求。在跨域多算力中心的场景下，还需要考虑到保持多个算力中心之间文件目录的一致所需的开销。因此综合评估，采用如下策略。

- 对I/O性能不敏感的软件和数据集，采用全局共享的方式降低管理的复杂度，例如图6中的/opt/app目录。
- 对I/O性能敏感的共享数据以及使用频率很高的应用，采用每个算力中心存放本地复制、基于容器挂载技术映射到共享目录对应位置的方式使用，由管理员负责数据的更新和同步，从而提升访问性能，节省跨数据中心访问的带宽。例如图6中的/data目录，管理员可将共享数据集放置在/data下。

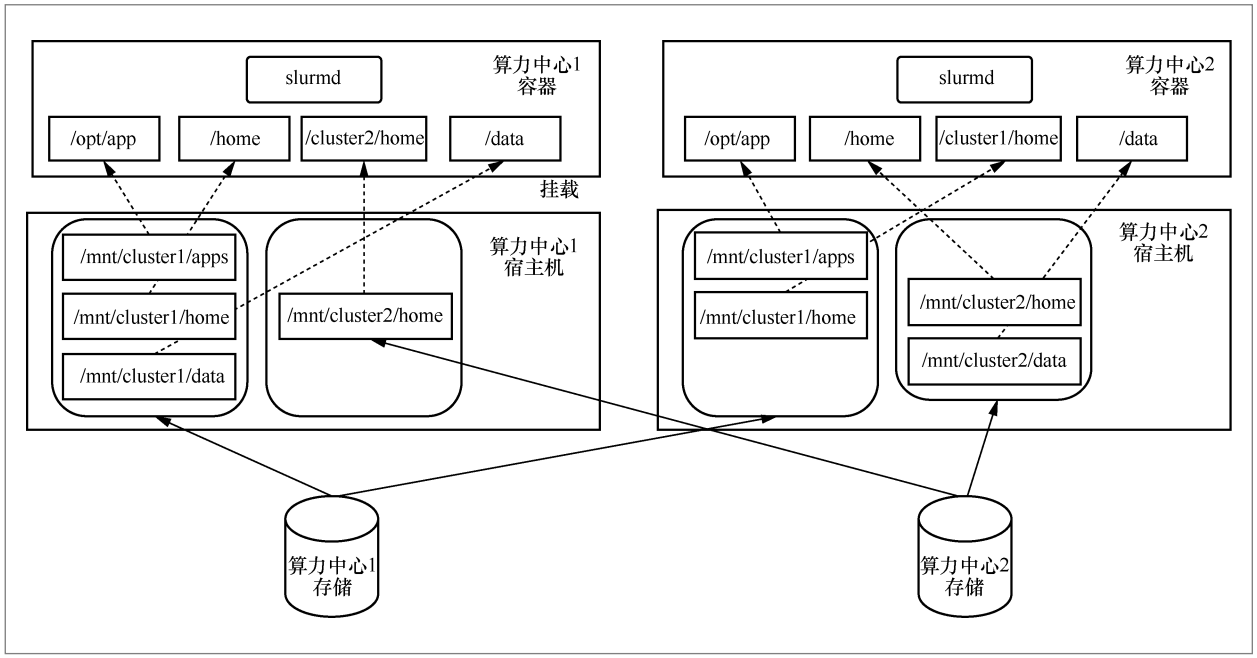

图 6 容器挂载与用户可见目录

(2)用户数据

与公共软件或数据集的低频率更新场景形成鲜明对比的是用户数据(通常放在用户的home目录),这部分数据几乎时时刻刻都在变化,无法通过文件同步的方式实时保持多算力中心之间的一致,只能采用前文所述的远程挂载的方案,以保证在一个算力中心对文件系统所做的修改在另外一个算力中心实时可见。但是,如果所有的算力中心均共享同一个用户目录,将会带来如下问题。

- 单点风险:跨中心隧道网络压力特别大,而且成为可靠性的单点风险。
- 性能受限:跨中心隧道网络提供的带宽与算力中心内的高速网络相比存在数量级上的差距。
- 无效流量:算力中心间存在大量无效的读写流量占用宝贵的跨中心隧道带宽。

因此,需要在用户使用的方便性和文件系统读写的性能之间取得平衡,让用户既可以方便透明地从一个算力中心直接读写在另外一个算力中心的文件,又能保证应用运行过程中低延迟、高带宽的要求。最终本文选择了本地存储和远程存储相融合的解决方案。

- 如果本地无法提供可靠的存储,共享使用其他算力中心存储:即在容器中将远程算力中心的home目录直接映射为本地的home目录。为了保证足够的读写性能,提供一个可靠性相对较低的本地存储用于满足计算过程中高IOPS要求。
- 如果本地有可靠的存储,优先使用本地的存储空间,在容器中将本地存储的用户目录映射为home目录,同时将远程其他算力中心的用户目录映射至容器中的/clusterX/home下,/clusterX是不同算力中心的代号。通过该方式,用户默认文件读取均在本地,不产生跨中心的网络流量,同时也可以直接通过/clusterX/home读取远程文件,或通过操作系统的各类文件工具(如mv、cp、vim等)进行文件跨中心的操作。

3.4 安全性

本文构建的系统既可以运行在局域网,也可以部署在广域网。对于局域网场景,网络环境相对安全可信,跨域数据访问的安全性相对可控。对于广域网场景,网络环境不可信,为了兼顾安全和性能,可以使用加密的网络隧道,例如WireGuard[20]。

4 案例分析与性能优化

4.1 案例介绍

中国人民大学和华中科技大学曾经面临着数据中心机房建设分散的问题,为整合多个算力中心的资源,笔者团队在两所高校校级高性能计算平台上部署了EaaS系统,并为用户持续提供服务两年有余,本文主要介绍华中科技大学高性能计算公共服务平台多算力中心接入案例与实践。目前平台自建公共集群一套,并通过远程接入方式先后完成了国家脉冲强磁场科学中心(强磁一号)、精密重力测量国家重大科技基础设施(引力一号)等多家单位的算力中心和资源的接入纳管,整合各集群资源后的理论峰值算力达2.5 PFLOPS,可用存储容量4 PB。其中,强磁一号集群为远程接入,该集群的机房与公共集群机房物理隔离,强磁一号集群机房与公共集群机房相距3 km,通过万兆光纤相连。强磁一号集群通过远程挂载方式使用公共集群机房的存储,并建设了机房内廉价的NFS缓存层。对于强磁一号集群,用户的home

目录挂载的是公共集群存储，用户在强磁一号的home目录所做任何文件操作（增加、修改和删除）都会直接写入公共集群存储上，并提供了实时的数据一致性。

4.2 容器技术性能实测

容器技术可以将宿主机的目录映射到容器中，用户实际在容器中进行高性能计算。为测试容器绑定技术对用户计算任务的性能影响，笔者分别使用HPL[21]和fio[22]对计算和存储系统进行了测试，主要测试容器内用户任务的计算性能相比裸金属物理机是否有损失。其中，HPL任务选择了5个计算节点，fio任务选择了2个计算节点对Lustre文件系统进行读写，随机读为小文件场景，顺序读与顺序写为大文件场景。以上任务主要模拟用户的多节点并行计算任务。性能对比见**表2**，所有实测任务均为运行5次的平均值，可以看到，容器相比裸金属物理机的性能损失在2.5%以内。

表 2 物理机与容器计算与存储性能实测对比

对比项	HPL/（GFLOPS）	随机读IOPS	顺序读带宽/（Gbit/s）	顺序写带/（Gbit/s）
裸金属物理机	4 980	184.5	14.53	9.43
容器	4 910	183	14.16	9.53
百分比	98.6%	99.2%	97.5%	101.1%

4.3 跨域数据访问性能实测

笔者使用fio对跨域文件系统进行了评测和分析，主要模拟了在不同算力中心进行跨域文件系统读写。根据常见I/O模式，笔者进行了两类评测：吞吐量与IOPS。对EaaS文件协议模式、EaaS网络传输模式、SSHFS与OneData进行了比较，即在强磁一号上，使用上述4种方案，对远程的公共集群存储进行读写测试。

（1）吞吐量

吞吐量指标主要面向数据归档、大数据分析等场景，例如很多科学数据被归档为HDF5大文件格式。笔者使用fio模拟了一个128 GB文件的顺序读和顺序写，使用不同大小的数据块，测试结果如**图7**所示。

随着块大小的增加，4种方式的吞吐量都有所增加，EaaS网络传输模式性能提升尤其明显，吞吐量可达到网络带宽上限（1 250 MB/s）的75%，能够充分利用算力中心间的跨域网络基础设施。相比SSHFS和OneData，EaaS文件协议模式在数据块较大时有一定优势。

（2）IOPS

IOPS指标面向包括人工智能在内的众多科学计算场景。笔者使用fio模拟了一个总大小为1 GB的随机读写场景，**图8**的测试结果显示，数据块较小时，EaaS网络传输模式的随机读性能更加出色。随着块大小的增加，几种方式的IOPS都有明显下降，这是因为块大小越大，读写单个文件的耗时越长，I/O操作次数越少。Eaas文件协议模式性能表现不佳，主要因为每次I/O的所有访问都要跨域执行，这些I/O访问包括访问MDS、MDT、OSS和OST；EaaS网络传输模式下，跨域I/O访问被先转化为NFS协议，以远程调用的方式读写跨域存储资源，而MDS、MDT、OSS、OST等访问都发生在算力中心内部。

综上发现，EaaS网络传输模式比较均衡，在大多数场景下能取得不错的性能，且有良好的适配性和可扩展性。

4.4 应用优化

比较适合运行时跨域数据访问的应用包括VASP、GROMACS、LAMMPS。这些

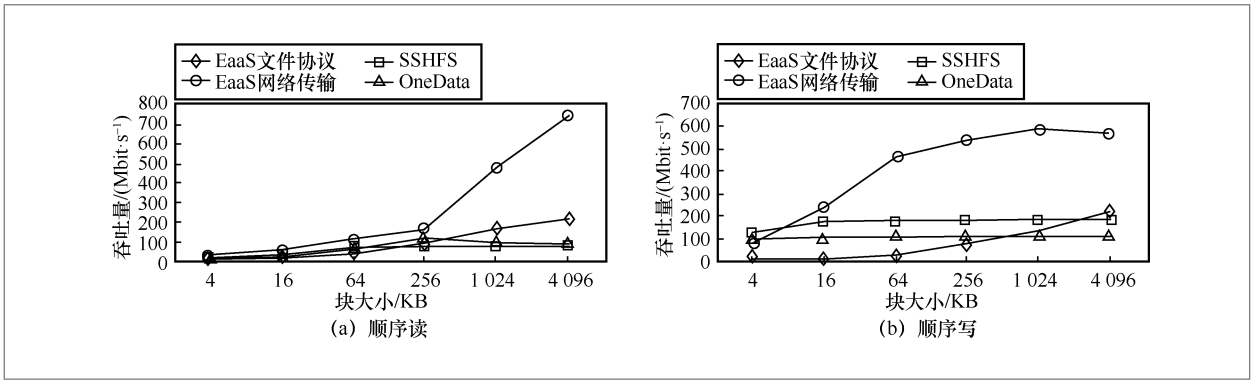

图 7 不同数据块大小下顺序读写性能

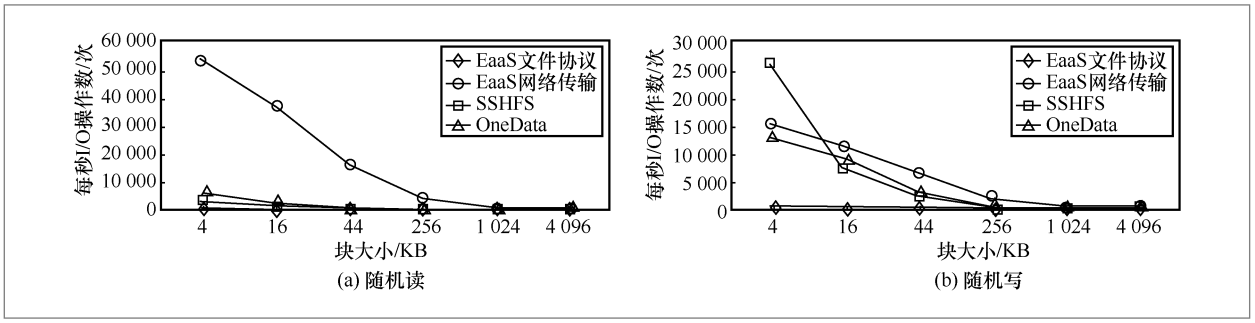

图 8 不同数据块大小下随机读写性能

应用是典型的计算密集型任务，输入文件和输出文件均为文本文件，整个过程对I/O性能要求低，对计算性能要求高，非常适合跨域算力调用，用户无感知。

不太适合运行时跨域数据访问的应用有PyTorch和WRF。PyTorch是当前典型的人工智能应用，输入数据量大，且计算机视觉任务需要处理海量小文件，对IOPS性能要求高，不适合直接远程访问。WRF的输入和输出都很大，且中间需要生成临时文件，也不适合直接远程访问。针对PyTorch和WRF这类对I/O要求高的应用，小文件随机读的性能瓶颈明显，数据应尽量在算力中心内的存储上，而非远程其他算力中心的存储上。算力中心内的存储可以是高价值的磁盘阵列或存储集群，也可以用廉价的NFS服务器做缓存，该缓存的可用性和可靠性不高，价格低廉，仅用来缓存IOPS密集型数据。

5 结束语

跨域算力网络希望整合多个算力中心的计算和数据资源，但现有的方案对跨域文件和数据管理关注不够。本文提出了一种跨域算力网络数据访问方案，该方案可通过文件系统协议转换，接入远程算力中心的并行文件系统存储资源；并通过容器绑定技术，将远程或算力中心内的存储挂载并绑定到指定目录，使用户无感知地访问数据。基于本方案的原型系统已经在高校校级计算平台部署运行，实测数据和用户体验显示，本文的方案能够满足常见的

高性能计算应用需求。未来将进一步优化数据密集型应用，有效管理跨域数据和元数据，以支持各类人工智能应用场景。

参考文献:

[1] FOSTER I T, KESSELMAN C. The grid: blueprint for a new computing infrastructure[M]. San Francisco: Morgan Kaufman Publishers, 1998.

[2] XU Z W, CHI X B, XIAO N. High-performance computing environment: a review of twenty years of experiments in China[J]. National Science Review, 2016, 3(1): 36-48.

[3] 钱德沛, 栾钟治, 刘轶. 从网格到“东数西算”：构建国家算力基础设施[J]. 北京航空航天大学学报, 2022, 48(9): 1561-1574.
QIAN D P,LUAN Z Z,LIU Y. From grid to “East-west Computing Transfer”: constructing national computing infrastructure[J]. Journal of Beijing University of Aeronautics and Astronautics, 2022, 48(9): 1561-1574.

[4] 秦广军, 肖利民, 张广艳, 等. 面向国家高性能计算环境的虚拟数据空间系统[J]. 大数据, 2021, 7(2): 101-122.
QIN G J, XIAO L M, ZHANG G Y, et al. Virtual data space system for national high-performance computing environment[J]. Big Data Research, 2021, 7(2): 101-122.

[5] 何小雨, 邓笋根, 栾海晶, 等. 国家高性能计算环境的虚拟数据空间运行支撑技术研究[J]. 大数据, 2021, 7(2): 158-171.
HE X Y, DENG S G, LUAN H J, et al. Study of technique support on the operation of virtual data space in national high-performance computing environment[J]. Big Data Research, 2021, 7(2): 158-171.

[6] 肖利民, 宋尧, 秦广军, 等. GVDS： 面向广域高性能计算环境的虚拟数据空间[J]. 大数据, 2021, 7(2): 123-146.
XIAO L M, SONG Y, QING J, et al. GVDS: a global virtual data space for wide-area high-performance computing environments[J]. Big Data Research, 2021, 7(2): 123-146.

[7] YOO A B, JETTE M A, GRONDONA M. SLURM: simple linux utility for resource management[C]//Proceedings of Job Scheduling Strategies for Parallel Processing. Seattle: Springer, 2003: 44-60.

[8] YIN F, SHI F. A comparative survey of big data computing and hpc: from a parallel programming model to a cluster architecture[J]. International Journal of Parallel Programming, 2022, 50(1): 27-64.

[9] BURROWS M. Lustre: building a file system for 1000-node clusters[C]//Proceedings of the 2003 Linux Symposium. [S.l.:s.n.], 2003: 380-386.

[10] SCHMUCK F B, HASKIN R L. GPFS: a shared-disk file system for largecomputing clusters[C]//The FAST 2002 Conferenceon File and Storage Technologies. Monterey: USENIX, 2002: 231-244.

[11] CATLETT C, ALLCOCK W E, ANDREWS P, et al. TeraGrid: analysis of organization, system architecture, and middleware enabling new types of applications[C]//Proceedings of the 2006 International Advanced Research Workshop on High Performance Computing and Grids. Amsterdam: IOS Press, 2006: 225-249.

[12] TOWN J, BOISSEAU J, ROSKIES J, et al. XSEDE: extreme science and engineering discovery environment (OAC 15-48562)[R]. 2020.

[13] NEWHOUSE S. Seeking new horizons: EGI's role in 2020 (EGI-1098-D230-V3)[R]. 2021.

[14] HENSCHEL R, SIMMS S, HANCOCK D, et al. Demonstrating lustre over a 100Gbps wide area network of 3500km[C]//Proceedings ofthe International Conference on High Performance Computing, Networking, Storage and Analysis. Piscataway: IEEE Press, 2012: 1-8.

[15] THOMSON A, ABADI D J. CalvinFS: consistent WAN replication and scalable metadata management for distributed

file systems[C]//Proceedings of the 13th USENIX Conference on File and Storage Technologies. Berkeley: USENIX, 2015: 1–14.
[16] CORBETT J C, DEAN J, EPSTEIN M, et al. Spanner: Google's globally-distributed database[J]. ACM Transactions on Computer Systems, 2013, 31(3): 8.
[17] CAO W, LIU Z J, WANG P, et al. PolarFS: an ultra-low latency and failure resilient distributed file system for shared storage cloud database[J]. Proceedings of the VLDB Endowment, 2018, 11(12): 1849–1862.
[18] A network filesystem client to connect to SSH servers[Z]. 2023.
[19] DUTKA U, SOTA R, WRZESZCZ M, et al. Uniform and efficient access to data in organizationally distributed environments[C]//Proceeding of eScience on Distributed Computing Infrastructure. Cham: Springer, 2014: 178–194.
[20] DONENFELD J A. Wireguard: next generation kernel network tunnel[C]// Proceeding of 24th Annual Network and Distributed System Security Symposium. San Diego: The Internet Society, 2017: 1–12.
[21] DONGARRA J, LUSZCZEK P, PETITET A. The LINPACK benchmark: past, present, and future[J]. Concurrency and Computation: Practice and Experience, 2003, 15(9): 803–820.
[22] Fio – flexible I/O tester rev. 3.25[Z]. 2020.

作者简介

鲁蔚征(1990–),男,中国人民大学大型科学仪器共享平台实验师,主要研究方向为高性能计算、数据科学。

戴奇志(1984–),男,联旌智能科技(上海)有限公司首席技术官,主要研究方向为高性能计算、云计算。

张策(1992–),男,华中科技大学网络与计算中心工程师,主要研究方向为高性能计算、数据中心信息化。

收稿日期: 2023-05-19
通信作者: 张策, cezhang@hust.edu.cn
基金项目: 国家重点研发计划资助项目(No.2020YFB1710004)
Foundation Item: The National Key Researchand Development Program of China (No. 2020YFB1710004)