基于网格服务的网格环境下数据传输

刘 骥 王 茜
(重庆大学计算机学院 重庆 400044)

摘 要 网格环境下的数据访问需要一种能够跨异构平台进行数据传输的机制,但是,现有的 GridFTP 协议存在某些情况下无法跨防火墙进行数据传输的局限。为解决这一问题,论文提出了一种基于网格服务进行数据传输的方法,使用数据压缩、并行数据传输、缓冲区调整技术改善了该传输方法的性能;基于传输日志实现了故障恢复,提高了数据传输的可靠性。最后使用 Globus 工具集实现了这种数据传输方法,并进行了传输性能的分析。

关键词 网格,网格服务,Web 服务,GridFTP

Grid Service-Based Data Transfer in Grid Environment

LIU Ji WANG Qian
(Department of Computer Science, Chongqing University, Chongqing 400044)

Abstract In grid computing environment, data access needs a mechanism with capability of crossing heterogeneous platforms, but it is localized by one of the disabilities of GridFTP, which can not go through firewalls in some situations. To solve this issue, an approach to transfer data in grid environment using grid service was presented. Meanwhile, data compression, parallel data transfer, and buffer adjustment technologies enhances the performance of the transfer. Based on fault recovery by transfer log, the reliability of data transfer is increased. Finally, Globus toolkit was used to implement this approach and analysis of performance in transfer was also conducted.

Keywords Grid, Grid Service, Web Service, GridFTP

1 引言

在网格计算环境下,为实现在"动态多机构的虚拟组织中进行资源共享并协同问题解决"[1],大量网格应用需要对远程数据进行访问。例如在为了完成某项科学实验而构建的网格系统中,不同地区的科学家利用网格跨机构、跨地域对科研数据进行访问,并将数据集成到本地的实验环境。但是由于网格环境的特殊性,组成网格的主机来自不同的机构、组织和地区,各主机处于不同的网络环境、软硬件环境并且主机的情况也在动态地发生变化。因此,在这种完全异构的计算环境中,网格应用访问数据需要一种具备跨越异构平台进行数据传输能力的数据传输机制。

目前,广泛采用的网格数据传输协议是 GridFTP[2],该协议通过对 FTP 协议的网格化扩展,侧重于在异构的存储系统之上提供统一的访问接口,以及解决大量数据传输的性能和可靠性等问题。在实际应用时,GridFTP 并不具备跨越所有异构环境进行数据传输的能力。这主要体现在 GridFTP 在某些情况下无法顺利地跨越防火墙进行数据传输上。一方面由于 GridFTP 设计上的原因,基于 GridFTP 发送的数据在通过某些防火墙时存在丢失连接等问题[3];另一方面,网格的组成者和使用者来自地理上分布的机构或组织,各机构或组织出于安全的考虑通常都会设置防火墙,并采取不同的防火墙策略。在这种情况下可能由于防火墙配置的原因造成 GridFTP 传输的数据无法通过,例如防火墙被配置为只允许特定协议的数据通过,其他协议的数据就会被过滤掉。通过防火墙接入网格的主机由于权限的限制通常也无法修改防火墙的配置。鉴于网格环境下有跨越防火墙进行数据传输的实际需求,而采用 GridFTP 又无法满足这种要求,为了完整地解决跨异构平台数据传输的问题,就需要解决跨防火墙数据传输的问题。

开放网格服务体系结构 OGSA(Open Grid Services Architecture)[4] 所定义的核心概念网格服务(Grid Service)[5] 不仅具有跨异构平台与其他网格实体进行交互的能力,其采用的基于 HTTP 协议传递 SOAP 消息[6] 的方式也是一种能够跨越网络中各种防火墙进行通信的机制。本文提出了基于网格服务进行数据传输的方法,完整地解决了网格环境下跨异构平台进行数据传输的问题。该传输方法基于网格服务,首先将传输的数据进行字符编码,然后采用 SOAP 消息封装,最后以 HTTP 进行传输,能够跨越各种异构的平台,也能够跨越各种防火墙。但这种传输机制,在字符编码时会增加数据量,在编码、解码时会消耗时间,因而在传输效率上有一定的损失。同时该传输方式缺少故障恢复机制,可靠性差。针对这些问题,本文采取了数据压缩、并行数据传输、缓冲区调整技术改善了传输性能;为了增加传输的可靠性,实现了基于传输日志的故障恢复机制。最后,本文基于 Globus 工具集[7] 实现了这种数据传输方法,对其进行了传输性能的分析,并对今后进一步的性能改进提出了展望。

2 基于网格服务的数据传输

2.1 网格服务概述

刘 骥 硕士研究生,主要研究方向为分布式计算、电子商务。王 茜 副教授,硕士研究生导师,主要研究方向为分布式计算、电子商务和远程教育等。

开放网格服务体系结构 OGSA 提出了一种以服务为中心的模型，是最新的网格体系结构。在这种结构中，一切网格化实体都被抽象为网格服务，网格就是可扩展的网格服务的集合。这种抽象将资源、信息、数据等统一起来，十分有利于灵活、一致、动态共享机制的实现，使得网格化实体的管理有了标准的接口和行为。网格服务是 OGSA 的核心概念，是一种扩展的 Web 服务(Web Service)[8]，具有明确定义并遵守特定惯例的接口、服务发现、动态服务创建、生命周期管理、通知等能力。针对网格环境下大量的服务属于临时服务的特点，网格服务支持临时的服务实例，并且能够动态创建和删除。通过网格服务句柄 GSH(Grid Service Handle)[5] 可以转换为访问服务实例的网格服务引用 GSR(Grid Service Reference)[5]，从而调用服务实例。在访问服务实例的过程中，网格服务采用 HTTP 传送 SOAP 消息。在目前的网络环境下 HTTP 是网络上支持最广泛的协议，几乎所有的防火墙都不会过滤 HTTP 流。SOAP 消息是一种基于 XML[9] 的文档消息，无论是何种软件平台都可以解析这种消息，防火墙也可以检查 SOAP 消息中的内容，甚至对其中的内容进行过滤。网格服务的这些特性不仅解决了异构环境下网格实体之间的互操作问题，也满足了网格数据传输所需要的跨异构平台要求。

2.2 数据传输的基本原理

基于网格服务进行数据传输的基本思想是利用网格服务能够跨异构平台(包括各种防火墙)进行消息传递的特点，将数据封装在网格服务传递的 SOAP 消息中，实现数据的传递。同时利用网格服务动态创建和删除服务实例的特性，为每一次数据传输创建一个临时的数据传输服务实例，通过对应的网格服务句柄 GSH 进行调用，使得同一网格服务可以同时进行不同的数据传输。

在这种传输方式下，参与传输的双方，分为数据发送者和数据接收者。其中数据接收者需要运行数据传输网格服务，而数据发送者通过发送数据的客户端程序调用接收者的数据传输网格服务的对应数据传输服务实例(使用 GSH)。图 1 是基于网格服务进行数据传输的过程，该传递过程按照如下方式进行：

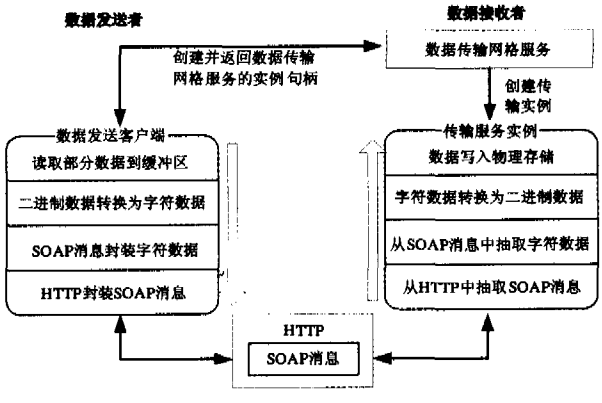

图 1 基于网格服务进行数据传输的过程

(1) 数据发送者调用数据接收者的数据传输网格服务，该服务创建一个传输服务实例，并向服务发送者返回相应的网格服务句柄。

(2) 数据发送者的数据发送客户端程序首先从本地数据中读取一部分到缓冲区，然后将二进制数据转换为字符数据，并生成对应的 SOAP 消息，最后将构造好的 SOAP 消息封装在 HTTP 中传输给数据接受者。

(3) 数据接收者端的传输服务实例在接受到 HTTP 传输的数据之后，从 HTTP 中抽取出 SOAP 消息，然后从 SOAP 消息中抽取出字符数据，并解码为二进制数据，最后将数据写入物理存储。

(4) 在传送大容量数据时，一次发送只能传输数据的一部分，此时数据接收者的传输服务实例应该通知数据发送者进行下一轮的数据传递，直到数据传递完毕为止。

2.3 传输方式存在的问题

虽然这种传输方式是一种完全跨异构平台进行数据传输的方式，但是这种方式却存在两个方面的问题。一方面，在这种方式中，SOAP 消息是一种基于 XML 的文档消息，无法直接保存二进制数据，因此需要将二进制数据转化为字符数据才能够将数据封装在 SOAP 消息中。在接收到数据之后，又需要将字符转化为二进制数据。编码、解码过程需要一定的时间。同时将二进制数据转换为字符数据会增加一定的数据量，例如采用 Base64 编码方式在最坏的情况下会增加近 33% 的数据量[10]。这两方面的开销都会损耗数据传输的性能。为解决这个问题，本文的第 3 部分提出了数据压缩、并行数据传输和缓冲区调整策略。综合这些策略可以改善数据传输的性能，本文在第 5 部分进行了改进前后的性能对比。另一方面，在网格这种分布式异构环境下，诸如断电、网络断线、主机故障等传输故障随时可能发生。但是这种传输方式却缺乏相应的故障恢复机制，在传输数据时，会因发生传输故障而导致数据重传。为解决这个问题本文在第 4 节提出了基于传输日志机制的故障恢复机制，提高了数据传输得可靠性。

3 传输性能的改进机制

3.1 数据压缩

采用将二进制数据编码为字符数据时会增加数据量，为了抵消这个数据量的增加，采用数据压缩是一种有效的方法。发送者可以先将二进制数据使用某种压缩算法进行数据压缩，然后再将压缩后的二进制数据进行字符编码。接收者在接收到数据之后，先将字符数据转化为二进制数据，然后再解压缩。在采取这种机制时需要注意两个问题：首先原始数据不能是已经被压缩的数据，如果是已经压缩后的数据，那么对其进行再次压缩没有任何意义。在传输时应该判断该传输的数据是否已经进行压缩，以确定是否使用数据压缩策略。其次，读取数据的缓冲区不能太小，太少的数据量会影响到压缩的效果。为此也可以采取先将数据全部进行压缩，然后传输的方式。

3.2 并行数据传输

在基于网格服务进行数据传输时，数据发送者和接收者在数据的编码、解码以及等待对方的应答上会消耗一定的时间。在使用单一传输服务实例进行数据传输时，发送者从读取数据到发送数据这段时间内，没有数据进行传输，网络端口处于闲置。在接收者收到数据直到返回信息给发送者这段时间内，网络端口同样处于闲置。图 2 是采用单一传输服务实例进行数据传输时一段时间内 CPU 使用时间和数据发送量的图像。

从图 2 可以看出，当进行数据发送时，CPU 又几乎处于空闲状态，如 A 点和 B 点两个数据发送的峰值时刻。而当 CPU 处于忙碌状态时，数据发送量很小，甚至为 0，如 C 点和 D 点所在时刻。并行数据传输的基本思想就是利用多个传输进程在同一时间内进行数据传输，充分利用这些闲置时间，从

而达到传输性能的提升。

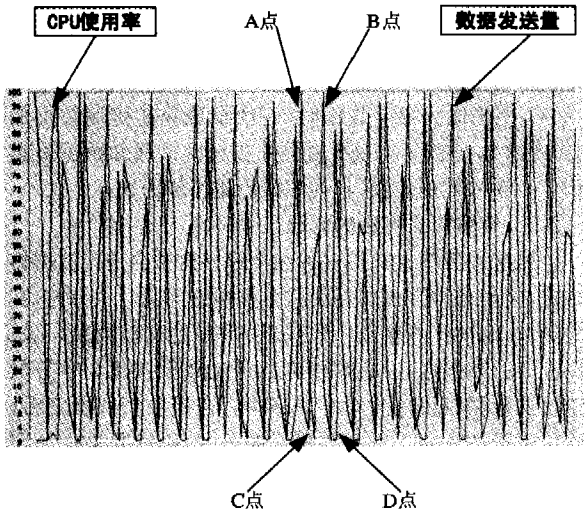

图 2　采用单一传输服务实例的 CPU 使用时间和数据发送量

采用并行数据传输时，数据发送者需要将数据分成多个数据块。数据接收者创建多个传输服务实例并返回相应的网格服务句柄。每一个传输服务实例只传输一个数据块的数据。传输时，数据发送者使用相应的网格服务句柄调用传输服务实例传输对应的数据块。

3.3　缓冲区调整

缓冲区调整是指对数据发送端读取数据的缓冲区进行大小的调整。此时如果缓冲区设置太大，就会造成更多的编码、解码时间，以及更大的内存消耗。在使用单一传输服务实例时，意味着更多的闲置时间。如果缓冲区设置太小，就会增加传输的次数，同样会降低传输的性能。如果采用了数据压缩机制，缓存区的大小将直接影响到数据的压缩效果。缓冲区的大小同样会影响到并行数据传输的数据传输效率。缓冲区调整机制将根据网络带宽、CPU 和内存资源利用率等软硬件状况对缓冲区的大小进行调整，以达到最佳的性能。

4　基于传输日志的故障恢复

4.1　传输日志的结构

为了提高基于网格服务进行数据传输的可靠性，本文引入了传输日志机制。对于一个运行着的数据传输任务，由一个或多个传输子任务组成。在传输日志中有两个部分与之对应。一部分是传输任务项，另一部分是传输子任务项。传输任务项标识了整个传输任务，传输子任务项标识了一个传输子任务。传输任务项包含以下几个部分：

(1) 传输任务编号。数据传输开始时被创建，作为数据传输任务的表示，全局唯一。

(2) 数据发送者标识。根据数据发送者来设定，通过这个标识可以确定传输任务的调用者。

(3) 数据名称。由数据发送者命名的数据逻辑名称，基于该名称和数据发送者标识可以确定一个传输任务。

(4) 数据存储的物理位置。数据在接收者端的本地物理存储位置。

(5) 传输数据大小。所传输数据的总大小。

(6) 传输任务失效期。任务在异常中止之后，多长时间可以删除该任务。

传输子任务项包含以下几个部分：

(1) 传输子任务项编号。在传输子任务被创建时创建，作为传输子任务的标识，全局唯一。

(2) 传输任务编号。与传输任务项中的传输任务编号相对应。

(3) 传输数据大小。传输子任务所需要传输的数据大小。

(4) 数据开始位置。指明传输子任务项所传输的数据在整个数据中的开始位置。

(5) 当前传输开始位置。每个传输子任务都需要进行多次数据传输才能够将该传输子任务所需传输的数据传输完毕，其中每一次传输称为“当前传输”。该项指明了当前传输的数据在整个数据中的位置。当这次传输完毕之后，应该根据当前传输的数据大小，更新这个项，新的“当前传输开始位置”等于旧的“当前传输开始位置”加当前传输的数据大小。

(6) 是否传输完毕。指明传输子任务是否已经结束。

在数据传输过程中，每个传输服务实例维护对应的传输任务子项，对传输日志不断进行更新，以反映最新的数据传输情况。当任务结束之后，删除相应的传输任务项和传输子任务项。

4.2　故障恢复

在传输过程中可能发生的错误主要有以下几类：

(1) 发送者发送的数据丢失。

(2) 接受者回复信息丢失。

(3) 发送者主机崩溃。

(4) 接收者主机崩溃。

(5) 网络连接中断。

由于数据传输基于网格服务，因此(1)和(2)两种情况都是通过网格服务本身的机制来解决的。网格服务自身的 XML-RPC 机制可以保证在交互过程中出现这两种错误的时候，数据能够被重新发送。当出现后三种情况时，传统的基于网格服务的数据传输就无法自动地恢复。

当出现情况(3)时，发送者在主机重启之后，重新开启程序进行传输。此时可以根据“数据发送者标识”以及要传输数据的“数据名称”来判断是否是一个因故障而中断的任务。如果发送者希望恢复传输，则检索相应的传输子任务项，并恢复相应的传输子任务。后两种情况(4)和(5)，如果发送者在无法发送的情况下，一直重试，而没有关闭程序，那么发送者还保留着“传输任务编号”和“传输子任务编号”。一旦与接收端的连接恢复，基于这两个信息通过查询传输日志，可以立刻恢复各个传输子任务的数据传输。如果发送者程序已经关闭，重新开启程序后可以按照处理情况(3)的方式进行数据传输的恢复。

由于某些原因，传输故障没有被恢复。此时接收者可以根据“任务失效期”决定何时删除这个传输任务，并清除相应的日志信息和已经传输的数据。

5　实现及传输性能测试

5.1　基于 Globus 的实现

Globus 工具集是网格应用开发中应用最广的开发工具，其核心部分基于网格服务，提供了众多的网格开发工具。Globus 工具集已经在 NASA 网格(NASA IPG)、欧洲数据网格(Data Grid)以及美国国家技术网格(NTG)等众多项目中得到了应用。基于 Globus 工具集提供的能力，本文开发了基于网格服务进行数据传输的原型系统。该系统包括了：数据发送客户端程序和数据接收服务器程序。其中数据接收服务

器程序包含了数据传输管理网格服务、数据传输网格服务以及传输日志。该应用实现了数据压缩、并行数据传输和缓冲区调整3种数据传输优化机制。

在本文的实现中，引入了新的数据传输管理网格服务。数据传输管理网格服务用于创建、监控并管理整个数据传输任务及其子任务。在这种模式下，数据发送客户端程序不再直接请求数据传输网格服务创建服务实例，而是请求数据传输管理网格服务来创建整个任务及其子任务，从而提高了数据传输的可靠性。图4是原型系统的实际结构，对应图3中的数字编号，整个传输过程的流程如下：

(1) 数据发送客户端首先请求数据传输管理网格服务创建其服务实例，并返回其句柄。

(2) 数据接收服务器程序所处的网格服务运行环境，创建数据传输管理网格服务的服务实例，并返回其句柄。

(3) 数据发送客户端程序通过数据传输管理网格服务的服务实例句柄访问该服务实例，并且提供各种传输信息(如传输的数据名称、采用多少实例进行并行传输等)，以创建传输任务及其子任务，并获取相应的数据传输网格服务实例句柄。

(4) 数据传输管理网格服务实例，根据数据发送客户端程序提供的信息创建数据传输任务及其子任务，并返回相应的数据传输网格服务实例句柄，建立相应的传输日志。在其后的传输过程中，数据传输管理网格服务实例还需要不断地更新数据传输日志。

(5) 数据传输网格服务在接受到创建服务实例的命令之后，创建相应的数据传输网格服务实例，每个实例对应一个数据传输子任务和传输日志中相应的数据传输子任务项。

(6) 数据发送客户端程序利用创建的数据传输网格服务实例进行数据传输。如果采用并行传输则使用多个传输实例。

(7) 传输过程中，每个数据传输网格服务实例维护传输日志中与之对应的数据传输子任务项，以便故障恢复。

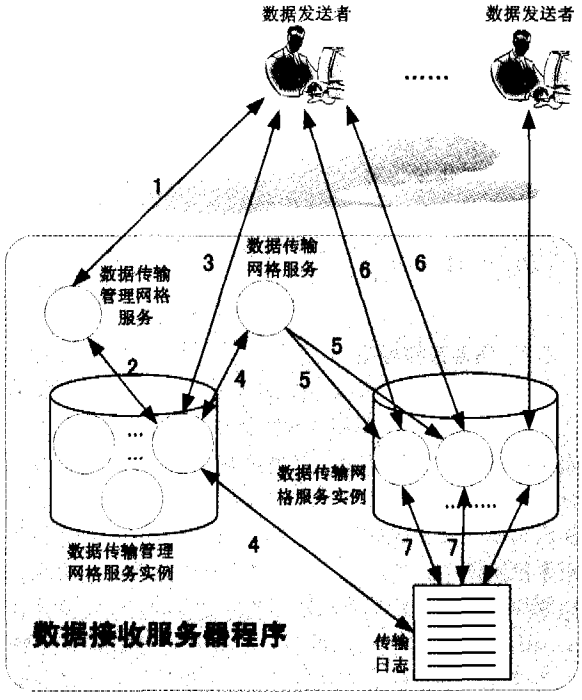

图3 基于网格服务的数据传输的Globus实现

5.2 传输性能测试

基于本文的原型系统，本文进行了传输性能的测试。测试条件是在局域网网络环境，网络带宽10Mbps，以及内存均为512MB、CPU为P3 1G的两台主机，分别安装数据发送客户端程序和数据接收服务器程序。两台主机均安装有防火墙软件，只开放8080端口(Globus的默认网格服务端口)，并过滤除HTTP以外的其他协议。测试数据为500MB的文档文件。测试分别使用32kB、64kB、128kB、256kB和512kB的传输缓冲区大小，并且测试单一传输实例、2个传输实例并行传输、4个传输实例并行传输、6个传输实例并行传输和8个传输实例并行传输。对于数据压缩机制进行了单独的测试。测试结果如下：

表1 测试结果(m:分，s:秒)

缓冲区大小 \ 传输实例数	单一实例	2	4	6	8
32kB	21m35s	15m48s	14m30s	13m36s	13m31s
64kB	18m23s	13m48s	13m34s	13m10s	13m09s
128kB	17m59s	15m21s	13m41s	13m47s	13m56s
256kB	17m28s	15m55s	14m09s	14m26s	14m37s
512kB	17m48s	15m37s	15m44s	16m37s	17m57s

在同等网络条件下，使用GridFTP进行网络传输测试，时间为7分40秒。根据上表的测试结果，基于网格服务的数据传输最好的传输速度是采用8个实例64kB缓存进行并行传输，时间为13分9秒。而未进行改进的数据传输(也就是1个实例的传输)，最好的情况也需要17分28秒。采用并行传输最慢的是17分57秒，仍然好于大多数的单一实例传输。图4是测试中传输速度最快一组的CPU使用率和数据发送量图像。从图中可以看出经过并行数据传输和缓冲区调整，CPU的使用率比图2有所下降，数据发送量比图2有了提高，并且曲线更加的连续。整个的传输性能得到了提升。

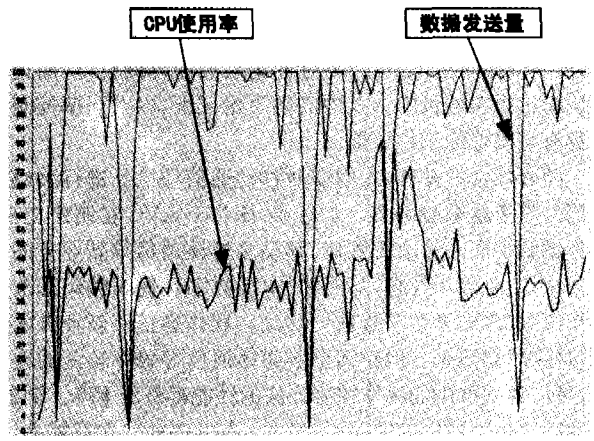

图4 64kB缓冲区8传输实例并行传输的CPU使用率和数据发送量图像

对于数据压缩机制，原型系统中采取用GZIP算法[11]。本文的测试数据由于全部是文本数据因此有很大的压缩比率。500MB的文件压缩为143MB，经过Base64编码后传输的数据量为157MB。压缩时间1分52秒，解压时间2分10秒。传输时间4分45秒。总时间为8分47秒。与采用GridFTP的时间接近。但是在对另一个已经压缩的637MB文件(AVI文件)进行测试时，压缩和解压时间就为5分17秒，且压缩后大小为623MB。因此在使用数据压缩机制前必

(下转第68页)

方案需要公布($n+p-t+1$)个信息;而对于 Yang 的方案,当 $p>t$ 时,需要公布($n+p-t+1$)个信息,但是当 $p\leqslant t$ 时需要公布($n+1$)个公共信息。很明显,当 $p<t$ 时,本文的方案和 Chien 的方案需要公布的信息量小于 Yang 的方案需要公布的信息量。尤其当 $p=1$ 和 $t=n$ 时,本文的方案和 Chien 的方案只需要公布 2 个公共信息,而 Yang 的方案仍需要公布 $n+1$ 个信息,这也是本文方案最为吸引人的地方之一。

通过以上分析,可以发现:Chien 的方案的优点是需要公布的信息量小,而缺点是秘密重构的计算复杂度大;Yang 的方案的优点是秘密构建的计算复杂小,而缺点是需要公布的信息量大,而且需要区分两种不同的情况并给以不同的实现。本文的方案正好综合了这两个的方案的优点,而避免了它们的缺点,因而是一个很有效的门限多重秘密共享方案。

结论 本文基于 Shamir 的秘密共享方案,提出了一个 (t,n) 多重秘密共享方案,一组秘密被 n 个参与者所共享,t 个或 t 个以上的参与者联合可一次性重构这些秘密,而少于 t 个参与者合作得不到秘密的任何信息。在每次秘密分发过程中,秘密分发者只需重新选取一个随机整数 r,而不必重新分配各参与者的秘密份额。比起 Chien 的方案,其秘密重构计算简单;比起 Yang 的方案,需要公布的公共信息量少且实现单一。安全性和性能分析表明,本文的方案是一个安全、有效的方案。

参 考 文 献

1 Shamir A. How to share a secret [J]. Communications of the ACM 22,1979. 612~613

2 Blakley G. Safeguarding cryptographic keys [A]. In: Proc. AFIPS 1979 Natl. Conf. [C], New York, 1979. 313~317

3 Asmuth C, Bloom J. A Modular approach to key safegrarding [J]. IEEE Transactions on Information Theory, 1983, 29:208~210

4 Karnin E D, Green J W, Hellman M E. On sharing secret system [J]. IEEE Transactions on Information Theory, 1983, 29:35~41

5 He J, Dawson E. Multisecret-sharing scheme based on one-way function [J]. Electronics Letters,1995;31(2):93~95

6 Harn L. Efficient sharing (broadcasting) of multiple secret [J]. IEE Proceedings-Computers and Digital Techniques, 1995, 142 (3):237~240

7 Chien H-Y, Jan J-K, Tseng Y-M. A practical (t, n) multi-secret sharing scheme [J]. IEICE Transactions on Fundamentals, 2000,E83-A (12):2762~2765

8 Yang Chou-Chen, Chang Ting-Yi, Hwang Min-Shiang. A (t,n) multi-secret sharing scheme [J]. Applied Mathematics and Computation,2004,151,(2):483~490

9 Crescenzo G D. Sharing one secret vs. sharing many secrets: Tight Bounds on the average improvement ratio [J]. Theoretical Computer Science,2003,295(1-3):123~140

10 ElGamal T. A public-key cryptosystem and a signature scheme based on discrete logarithms [J]. IEEE Transactions on Information Theory, 1985,IT-31:469~472

11 Tan K J, Zhu H-W, Gu S J. Cheater identification in (t, n) threshold scheme [J]. Computer Communications, 1999,22:762~765

12 Rosen K H. Elementary Number Theory and Its Applications [M]. Addison-Wesley, MA, 1993

13 Aho A, Hopcroft J, Ullman J. The Design and Analysis of Computer Algorithms [M]. Addison- Wesley, Reading, MA, 1974

14 Chang C C, Horug H J, Buehrer D J. A cascade exponentiation evaluation scheme based on the Lempel-Ziv-Welch compression algorithm [J]. Journal of Information Science and Engineering, 1995,11(3):417~431

(上接第 59 页)

须对文件是否进行过压缩进行判断,否则反而会增加数据传输的时间。

结论及未来工作 基于网格服务的数据传输具跨越防火墙进行数据传输的能力,是一种完整的跨异构平台进行数据传输的方式。采用数据压缩、并行数据传输和缓冲区调整机制可以提高这种传输方式的传输效率。本文提出的基于传输日志机制的故障恢复机制又提高了这种方式进行数据传输的可靠性。虽然在传输效率上这种方式与现有的 GridFTP 仍然有一定的差距,但是在需要跨防火墙进行数据传输的网格应用中,这种方式比 GridFTP 具有更强的适用范围。目前影响这种传输方式性能的关键因素在于二进制转化为字符过程中所增加的数据量。这是由于 SOAP 无法直接表示二进制所造成的。为了解决这一问题,国外已经展开了相应的研究工作。例如 SUN 的 Fast Infoset 项目[12],W3C 在 2005 年 1 月 25 日发布的 XOP(XML-binary Optimized Packaging)[13]、SOAP MTOM(SOAP Message Transmission Optimization Mechanism)[14]、RRSHB(Resource Representation SOAP Header Block)[15]规范。以及采用二进制方式传输 Web 服务消息的 Hessian 二进制 Web 服务协议(Hessian Binary Web Service Protocol)[16]。当前这些研究还处于起步阶段,未来的研究工作需要结合这些相关研究,将其应用到网格领域,彻底地解决基于网格服务进行数据传输所存在的效率问题。

参 考 文 献

1 Foster I, Kesselman C, Tuecke S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations [J]. Supercomputer Applications, 2001

2 Allcock B, Bester B, Bresnahan J, et al. Secure, Efficient Data Transport and Replica Management for High-Performance Data-Intensive Computing [J]. IEEE Mass Storage Conference, 2001

3 Mandrichenko I. GridFTP Protocol Improvements [S]. GridFTP WG, 2003

4 Kesselman F C, Nick J, Tuecke S. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration [J]. Ope Grid Service Infrastructure WG, Global Grid Forum, 2002

5 Czajkowski T K, Foster I, Frey J, et al. Open Grid Services Infrastructure (OGSI) Version 1. 0 [S]. Global Grid Forum Draft Recommendation, 2003

6 Simple Object Access Protocol [EB/OL]. http://www. w3. org/TR/soap/, 2005-03-05

7 Globus project [EB/OL]. http://www. globus. org/,2005-03-05

8 Web Service [EB/OL]. http://www. w3. org/2002/ws/, 2005-03-05

9 Extensible Markup Language [EB/OL]. http://www. w3. org/XML/,2005-03-05

10 Josefsson S. The Base16, Base32 and Base64 Data Encodings [S]. RFC 3548, 2003

11 Deutsch P. GZIP file format specification version 4. 3 [S]. RFC 1952, 1996

12 Fast Infoset [EB/OL]. http://java. sun. com/developer/technicalArticles/xml/fastinfoset/

13 Gudgin M, Mendelsohn N, Nottingham M, Ruellan H. XML-binary Optimized Packaging [S]. W3C Recommendation, 2005

14 Gudgin M, Mendelsohn N, Nottingham M, Ruellan H. SOAP Message Transmission Optimization Mechanism [S]. W3C Recommendation, 2005

15 Karmarkar A, Gudgin M, Lafon Y. Resource Representation SOAP Header Block [S]. W3C Recommendation, 2005

16 Hessian Binary Web Service Protocol [EB/OL]. http://www. caucho. com/hessian/