维普资讯 http://www.cqvip.com



# 基于曙光3000并行机的远程并行调试器*)

陈 勇 李春生 安 虹 郑启龙 陈志辉
(中国科学技术大学计算机科学技术系 合肥230027)
(国家高性能计算中心(合肥) 合肥230027)

**摘 要** 并行调试对并行程序开发非常重要,然而传统的远程并行调试方式是采用登录(telnet)命令通过命令行的文本界面进行,非常繁杂。本文介绍了为曙光3000系统设计实现的远程并行调试器RPB(Remote Parallel Debugger)。RPB实现了完全并行调试等功能,并且具备图形用户界面,用户界面采用Java语言和Swing工具包实现,具备平台独立的特点。RPB采用客户端/服务器模式,客户端和服务器之间的通信采用当今流行的CORBA中间件技术。RPB支持通过局域网或广域网远程调试并行机上的程序,屏蔽了客户平台的差异和并行机地理位置上的差异,因而大大提高了并行机的好用性。

**关键词** 调试器,好用性,Java,CORBA,曙光3000

## Remote Parallel Debugger Based on Dawning3000 Parallel Machine

CHEN Yong LI Chun-Sheng AN Hong ZHENG Qi-Long CHEN Zhi-Hui
(Department of Computer Science and Technology, University of Science and Technology of China, Hefei 230027)
(National High Performance Computing Center at Hefei, Hefei 230027)

**Abstract** This paper introduces the design and implementation of the remote parallel debugger (RPB) based on Java/CORBA, which is developed for Dawning3000 parallel machine. Parallel debugging is important to develop parallel applications. However, the traditional remote parallel debugging method is to use the command line interface by using telnet command, which makes users feel very inconvenient. So we implement a GUI-based debugger to support the remote parallel debugging. Here, GUI is implemented with Java language and Java GUI Swing package, which make it platform-independent. The RPB implements the completely parallel debugging technique. The RPB employs client/server mode and the communication between client and server is implemented with currently prevalent CORBA middleware. The RPB supports remote parallel debugging within both local area network and wide area network, and hides the difference of client platform and location of parallel machines, which highly improves the usability of parallel machines.

**Keywords** Debugger, Usability, Java, CORBA, Dawning3000

## 1 引言

长期以来,高性能并行计算机难于使用是用户普遍抱怨的一个主要缺点。这个缺点主要是因为高性能并行计算机一直缺乏一个完善的系统管理和开发环境。实际上,到目前为止,大多数并行计算机还没有这样功能完整的工具,而且已有的工具也大多仍然是难于使用的文本界面,这使得目前的并行机难以适应越来越多的并行应用的需要[1]。

正是在这样的背景下,我们以曙光3000超级计算机为目标系统,研制开发了一个功能完整的曙光并行机客户端集成环境工具DUET(Dawning User-side-integrated Environment and Tools)[2]。DUET的设计目标是解决并行计算机的"好用性(Usability)"问题,使用户通过浏览器像访问自己的桌面系统一样,在图形的用户界面下访问远程的超级计算机,并在客户端的图形界面系统内集成支持并行应用软件开发和使用全过程的工具,降低用户使用并行机的复杂度,提高并行机系统的使用效率,增强并行软件的好用性和互操作性。DUET还集成了有效的联机帮助和辅助教学培训等功能。

本文将详细探讨DUET中的重要组成构件远程并行调试器RPB(Remote Parallel Debugger)。调试是并行程序开发中一个重要的环节[3],由于并行程序的特殊性和复杂性,使得并行程序调试变得尤为困难,因此对并行程序调试的强有力支持显得尤为重要。然而目前的远程并行调试方式是采用登录命令通过命令行的文本界面进行,非常繁杂。因而我们在研发DUET时,致力于开发一个功能强大、易于使用的远程并行调试器。

中国科学技术大学高性能计算中心长期以来致力于并行调试的研究,取得了一定的成果,目前已经在曙光机群系统上实现了DCDB并行调试器[4],该调试器易于使用、界面友好、功能强大、性能优越,被集成到曙光系列并行机的用户环境工具中,目前又被移植到国产神威机群系统上。但DCDB尚不支持远程并行调试功能,本文探讨的RPB的设计与实现正是对DCDB功能的一个重要扩充。本文第2节将介绍远程并行调试器设计思想,第3节介绍主要技术,第4节介绍实现方法,第5节介绍性能测试,最后是介绍相关工作和小结。

## 2 远程并行调试器设计思想

### 2.1 传统的远程并行调试方法

*)基金项目:本文工作受国家863项目(No. 2001AA111041),中国科学技术大学青年科学基金(No. KA1109)和中国科学院高水平大学建设资助项目(No. KY2706)的资助。**陈 勇** 硕士生,主要研究领域为并行计算和分布式计算;**李春生** 实验师,主要研究领域为并行计算;**安 虹** 博士后,副教授,主要研究领域为计算机体系结构和高性能计算;**郑启龙** 博士研究生,研究方向为并行软件工具与环境和高性能计算;**陈志辉** 博士生,主要研究领域为高性能网络和分布式系统。

维普资讯 http://www.cqvip.com

并行调试最常见的一种方法是在源程序中加入某些输出语句作为测试指令，测试程序运行到此时变量值是否正确。这种方法对远程并行调试同样有效。这种方法虽为直观，但实际上是一种测试的方法，并不是一种完整的并行调试方法，其功能非常简单，而且需要反复修改源程序，并编译运行测试，使用起来比较困难。目前在并行计算机上完整的远程并行调试方法仍然主要是使用传统的方法，就是使用 telnet 命令登录到并行计算机上，然后用命令行的方式，通过启动带 gdb 或 dbx 参数的 mpirun 命令进行调试。如果发现程序有错，需要终止调试，然后再使用 vi 等文本编辑命令进行远程编辑修改源程序，再存盘进行编译、调试（或是通过在本地使用更易用的文本编辑工具修改源程序，再使用 ftp 命令上传至并行计算机，进行编译、调试）。不难发现，传统的远程并行调试仍然停留在文本界面的技术上，难于使用，并且相当繁琐。

### 2.2 远程并行调试器 RPB 设计思想

我们设计的 RPB 主要是解决传统远程并行调试方法难用、繁琐的问题，并且提供一个更为强大的并行调试技术。因而我们的主要设计思想是：①提供一个友好的图形化调试界面；②使用完全并行调试技术，对并行程序执行中各种不确定性因素提供全面的调试支持；③支持用户远程通过浏览器使用，用户可以抛开传统的 telnet 或是 telnet+ftp 繁琐的方法，并且用户可以随时随地在任何一台安装有浏览器的计算机上使用，无需安装客户端应用程序。

## 3 远程并行调试器主要技术

### 3.1 并行调试技术

在并行调试上，我们主要采用了两种技术：断点调试和记录/重放调试。断点调试是串行程序调试中的基本方式。在调试开始时，通常在语句级设置断点。在并行调试中，多任务取代了单任务，相应的多个串行断点取代了单个断点。记录/重放调试是为了解决并行程序的不确定性，从而使循环调试能有效地为并行调试服务[5,6]。记录/重放调试需要先对源程序进行插桩处理，然后在程序运行中可以对相关的消息传递进行记录，以便在重放时利用这些信息来控制同步通信，完全重现先前的通信过程，以此保证循环调试的有效性。在并行调试技术的实现上，我们是通过扩充现有的串行调试器的功能来完成。

### 3.2 开发语言

为了让 RPB 客户端具有平台独立性，使其可以在当前大部分主流的平台（包括 PCs，MACs 和 UNIXs 等平台）上运行，我们采用了 Java 语言和 Swing 包[7,8]。因为用 Java 语言写的应用程序具有跨平台的特性，并且 Java 小程序可以直接在 Web 浏览器里执行，这样，用户不需要安装任何客户端软件就可以直接在 Web 浏览器里面进行远程并行调试。而 Swing 组件技术使用户界面在不同的平台上都能保持组件的界面样式特性。

RPB 的服务器端采用了 Java 语言和 C 语言相结合的方式，使用 Java 语言使得这个系统具有良好的移植性，在某些关键的地方，为了提高系统的性能，我们采用了 Java 语言中的本地方法，这些本地方法使用 C 语言来实现，并编译成动态链接库的形式。

### 3.3 远程通信技术

远程用户和服务器之间的通信方式对开发远程并行调试器非常重要，常用的通信方式有 Socket、COM、CORBA 或自行设计等。我们采用当今流行的 CORBA 中间件技术，这样可以大大提高系统的可扩展性，而且 CORBA 提供了远程调用方式，使用起来非常方便。CORBA（Common Object Request Broker Architecture）是 OMG（Object Management Group）制定的面向对象分布式应用的一种规范[9]。基于 CORBA 的应用可以看作是一组 CORBA 对象的集合。每个 CORBA 对象都有一个接口，用以表示这个对象中可以被远程激发的操作。对象接口使用接口定义语言 IDL（Interface Definition Language）来描述。CORBA 的体系结构如图1所示。当客户端要激发远程对象的一个操作时，即以客户端的 IDL 桩为代理，把执行该操作的请求与参数发送给对象请求代理 ORB，ORB 则通过对象适配器和服务器端的 IDL 骨架激发对象执行该操作，并把其返回值送给客户。客户端的 IDL 桩和服务器端的 IDL 骨架都由 IDL 编译器自动生成。

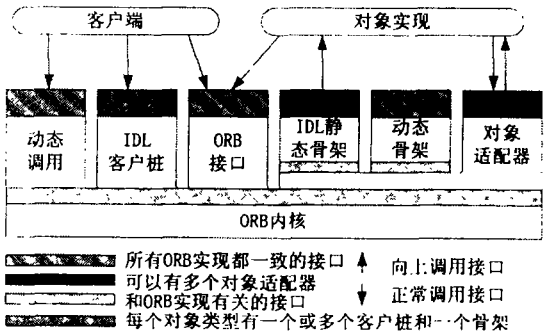


图1 CORBA 体系结构

### 3.4 我们选择的主要技术的好处

我们确定这样的主要技术的好处有：①系统具有良好的可移植性和可扩展性；②系统具有良好的图形界面，易于实现界面视图同步更新；③在串行调试器的基础上开发并行调试功能，能充分利用现有技术，简化设计和实现；④使用了许多用户已熟悉的串行调试器命令，用户无需花很多时间重新学习即可使用本调试系统；⑤应用软件的开发和网络通信技术的特性分离开来；⑥用户无需了解底层系统的实现细节。我们只需提供用户所需的各项服务的接口即可，这些接口可以采用多层抽象的方式，并使用统一的标准来减轻用户的负担。用户所看到的只是最上层的用户界面，其内部通信完全对用户透明。

## 4 远程并行调试器实现方法

RPB 远程并行调试器从上到下可分为五层：①用户界面层；②客户端/服务器网络通信层，负责客户端和服务器间的通信，利用 CORBA 实现；③服务器本地网络通信层，负责服务器和串行调试器层间的通信，利用 Socket 实现；④串行调试器层；⑤并行程序层。系统结构图如图2所示。

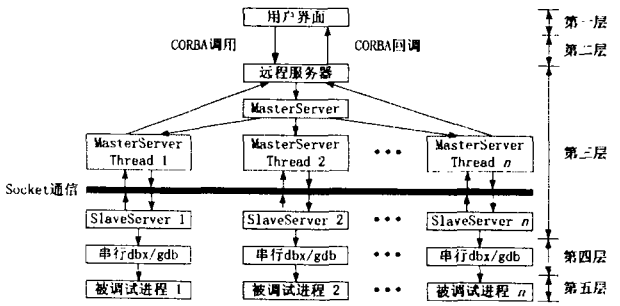


图2 RPB 系统结构图

用户所看到的只是最上层的用户界面，用户发出的调试

维普资讯 http://www.cqvip.com

命令将通过 CORBA 调用传递给远程服务器，服务器再经过 Socket 通道把调试命令传递给分布式的底层的串行调试器。MasterServer 是 Socket 的服务端，SlaveServer 是 Socket 的客户端。串行调试器执行调试命令的结果通过 Socket 通道传回，由服务器端通过 CORBA 回调将其显示到用户界面中。

### 4.1 接口定义

我们首先需要定义分布式对象间调用的接口，以实现客户端和服务器的交互。实际上，接口定义就是提供对象的接口信息，因为客户在向对象发送请求前必须知道对象提供哪些服务，因而对象的接口定义应该规定对象所支持的操作和类型，也就是定义客户向对象发送的请求的方法。在 CORBA 中，对象的接口定义由接口定义语言 IDL 实现。因为 RPB 远程并行调试器现已嵌入曙光3000客户端集成环境 DUET 中，它使用的接口定义是 DUET 接口定义的一部分，因而放在 Duet.idl 接口定义中，下面是 RPB 接口定义示例。

1. RPBMFrame 接口　该接口定义了服务器端对客户端的一些回调接口，用以控制服务器和客户端的交互。包括初始化客户端窗口、设置客户端窗口的一些属性、显示和清除断点、向客户端窗口的源程序区输出源程序以及向客户端窗口的调试信息区输出调试信息等。

```
interface RPBMFrame{
    void initDebugWindow(in long index,in string SearchPath,in
    string progname,in string filename,in string prog); //初始化客
        户端调试窗口
    void fileProc(in long index,in string srcfile); //在第 index 号调
        试窗口显示源程序
    void writeFrame(in long index,in string str); //在第 index 号调
        试窗口显示调试信息
    void setFileBreakPoint(in long index); //设置第 index 号调试窗
        口的断点
    void recordBreakPoint (in long index, in string text, in long
    lineNum); //记录第 index 号调试窗口用户设置的断点
...
}
```

2. RPBServer 接口　该接口定义了客户端对服务器的一些调用接口，用以控制客户端和服务器端的交互。包括获取服务器系统的一些属性、获取调试器的配置、设置服务器属性、设置调试器配置、获取进程数、写 tmp 文件、写 tracefile 文件、注册/注销 CORBA 服务对象、以及清除服务器端的调试状态、复位 PVM 等。

```
interface RPBServer{
    void init(in string firstHost,in long num,in long maxwindow);
        //初始化服务器端一些信息
    boolean register(in RPBMFrame MFrameClient); //向服务器注
        册一个 RPBMFrame 对象
    void release(); //释放客户端注册的对象
    string getSystemProperty(in string key); //获取服务器端的系
        统属性
    void startMasterServer(); //启动主服务程序
    void writeSlaveSocket(in long i,in string str); //向第 i 号 Socket
        通道写入信息
    void dealUseThread(in long i,in string str); //启动处理调试命
        令的线程
    void ReplayDone(); //进行重放
...
}
```

### 4.2 服务器端实现方法

在服务器端，主要工作是实现接口定义中的操作，根据客户端的请求，完成并行调试工作。为此，我们创建了 RPBServer 接口的实现对象 RPBServerImpl，这个对象是在用户发出请求(即点击 DUET 主控面板中的并行调试器按钮)后创建的。这个请求需要被认证后才能创建，但因为 RPB 是 DUET 中的一个构件，因而我们直接采用 DUET 中的用户认证，由 DuetFSImpl 对象认证后再创建 RPBServerImpl 对象，并把它挂在 ORB 总线上，然后向客户端返回对这个 RPBServerImpl 对象的引用。客户端获得这个引用后，就可以向 RPBServerImpl 对象发送请求，这些请求就是在 RPBServer接口中定义的功能。服务器端如果需要对客户端发送请求，则可以通过 CORBA 回调实现，这些请求是在 RPBMFrame 接口中定义的功能。实际上，服务器端的工作都是在客户端发出请求后完成的，这就好像客户端在操作一个本地的并行调试器一样，只不过这是一个远程的并行调试器，但一切网络通信的细节都被 CORBA 技术隐藏起来，网络通信对用户来讲完全透明。

服务器端并行调试的实现机制可以描述如下：例如用户要对启动 np 个进程运行的 MPI 程序进行远程调试，这将会从客户端通过 RPBServerImpl 对象向服务器端发送启动 MasterServer 进程的请求，随后再向服务器端发送“java RPB. SlaveServer 〈调试窗口标题〉-np〈并行进程数〉〈MPI 程序名〉〈参数〉-p4norem”这一请求。服务器端在接收到第一个请求后，将先启动 MasterServer 进程，监听来自本地节点的 Socket 连接请求。服务器端在接收到客户端的第二个请求后，将启动 np 个 SlaveServer 进程负责调试，然后一一和 MasterServer 进程连接，MasterServer 进程则会派生出 np 个 MasterServerThread 线程来处理 SlaveServer 进程的连接请求，并通过 Socket 通信，把 MasterServerThread 的输出流输入到相应的 SlaveServer 的输入流中，把 SlaveServer 的输出流输入到相应的 MasterServerThread 的输入流中，即把 MasterServerThread 发出的调试命令传递给 SlaverServer 调试进程，而把 SlaveServer 输出的调试信息传递给 MasterServerThread。而每个 MasterServerThread 则通过 CORBA 和客户端的每个调试窗口进程建立通信连接。这样服务器端在接受并处理完来自客户端的这两个请求后，就可以通过 CORBA 调用和 Socket 通道，来实现接收用户的调试命令，并传递给远程服务器端的调试进程，然后再将调试进程的输出信息传递给客户端的调试窗口。

### 4.3 客户端实现方法

客户端主要是实现以图形界面的方式提供对被调试进程的访问、控制和检查命令，这通过 Java 语言中的 Swing 包实现。如图3所示是从浏览器通过访问 DUET 调用并行调试器的界面，图4所示是用户调试 MPI 程序的情形。我们主要通过如下几方面实现图形化的并行调试窗口：①用一个窗口显示一个进程的调试信息，并在窗口名中显示调试进程号和所在节点；②将源程序名、源程序和调试信息分别输出在文本框和窗口区中。在源程序窗口区中，对源程序添加行号，当设置断点后，将显示断点标志，对于当前正在执行的语句使用加亮显示；③常用的调试命令，实现了快捷按钮，包括设置断点、删除断点、单步执行和重放等功能；④支持对调试进程组发出调试

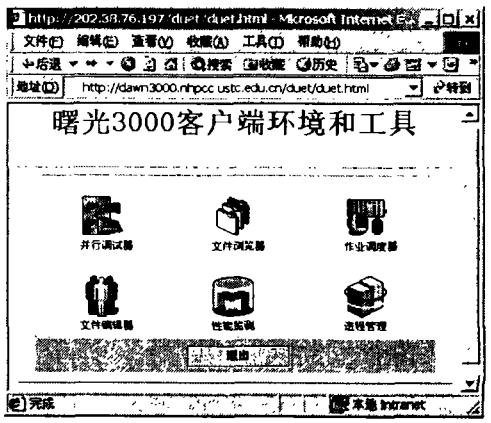


图3　DUET 界面

维普资讯 http://www.cqvip.com

命令，这可以从进程组下拉列表中选择需要发出调试命令的进程组；⑤提供命令行的支持，用户可以使用更加强大的调试功能；⑥对于集成调试环境的其它功能，例如对源文件的创建、编辑、编译运行等，这些在 DUET 集成环境中已经完全支持。

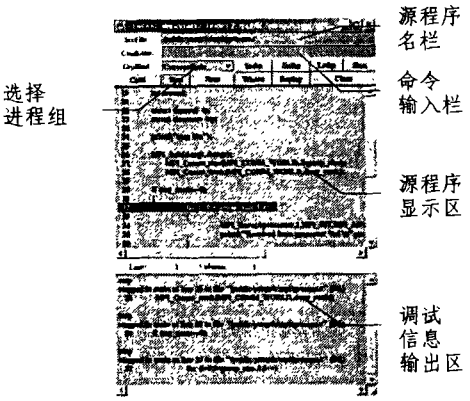


图4　RPB 并行调试窗口

客户端的实现机制向用户屏蔽了服务器端底层系统的实现细节，用户所看到的只是最上层的用户界面，其内部通信完全对用户透明。为此，在客户端，我们创建了 RPBMFrame 接口的实现对象 RPBMFrameImpl，这个对象是在客户端获得 RPBServerImpl 对象的引用后创建，并通过 RPBServerImpl 对象的 register(in RPBMFrame MFrameClient)请求向服务器端注册。这样使得服务器端可以通过 RPBMFrameImpl 对象对客户端进行 CORBA 回调，以此来控制客户端的信息输出和显示等。

## 5　远程并行调试器性能测试

我们在曙光3000系统上，通过使用 RPB 远程调试 MPI/PVM 并行程序，从四个方面对 RPB 进行了性能测试。

### 5.1　正确性测试

这部分的测试目标是检查 RPB 是否正确的实现所提供的全部功能，这包括是否支持设置用户参数并正确保存到服务器端、是否能正确读取服务器端用户参数设置、是否支持 MPI 和 PVM 程序的调试、是否同时支持 dbx 和 gdb、是否能正确显示服务器端文件列表和程序源码、是否能正确处理用户所有的调试命令、是否能在浏览器窗口正确反映程序执行到的位置和断点位置、是否能正确显示并行调试进程的状态和最终调试输出结果、是否能正确支持用户通过设置调试进程组进行组调试、是否能正确支持用户通过对程序插桩来进行记录/重放调试以及是否支持多用户同时调试等共计18项正确性测试。测试结果表明，RPB 完全正确地实现了所有测试目标。如图5所示，RPB 支持用户设置调试进程组，并对设置的进程组发出调试命令进行组调试。

### 5.2　容错性测试

这部分的测试目标是检查 RPB 是否健壮，是否能正确处理出错时的情况，这包括在设置最大进程时，如果输入不合法，是否能正确判断并给出出错信息；当在用户指定目录下未找到源程序，是否能正确给出出错信息；对 PVM 程序，是否能控制 spawn 的进程数，当 spawn 进程数超出最大进程数时是否能给出出错信息并正确处理；以及输入错误的调试命令，是否能不影响以后的调试等共计6项容错性测试。测试结果表明，RPB 完全能处理各种出错情况。如图6所示是 RPB 调试 PVM 程序时，一旦发现 spawn 进程数超出最大进程数时，将给出出错信息并清除所有调试进程，避免错误结果的出现。

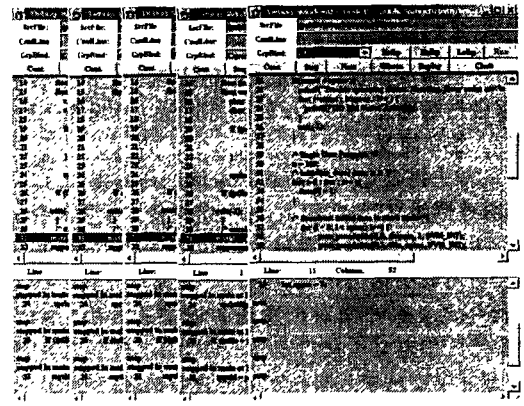

图5　RPB 支持设置进程组进行组调试

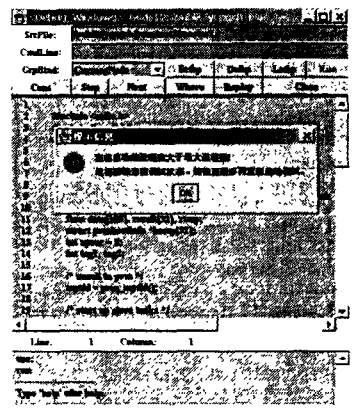

图6　RPB 容错性测试

### 5.3　网络传输性能测试

表1　RPB 网络传输性能数据

| 源程序 | 源程序文件大小(字节) | 源程序传输时间(ms) | 调试命令响应时间(ms) |
|---|---|---|---|
| hello.c | 595 | 20 | 30 |
| pimpi.c | 1110 | 50 | 20 |
| spmd.c | 4820 | 120 | 20 |

如表1所示是我们测试的 RPB 远程并行调试时的网络传输性能数据。其中源程序传输时间是指用户选择调试的源程序后到 RPB 的调试窗口完全显示源程序的时间；调试命令响应时间是指用户发出调试命令后到获得调试结果的时间。

### 5.4　并行调试性能测试

经测试，在一台64MB 内存的 PowerPC＋一台128MB 内存的 PowerPC 组成的简易机群系统上，以矩阵相乘 par 为例，RPB 最多能远程调试11个并行进程。如果有多用户同时远程调试，每个用户用3～4个并行进程，大约能支持3个用户同时分别调试。在由64个计算节点和6个服务节点组成的曙光3000系统上，以矩阵相乘 par 为例，RPB 大约能远程调试400个并行进程。如果有多用户同时远程调试，每个用户用4～5个并行进程，大约能支持100个用户同时分别调试并行应用程序。

## 6　相关工作

Cray 公司实现的 TotalView 是一个强大的并行调试工

（下转第200页）

维普资讯 http://www.cqvip.com

共享一块有限大小的存储资源，这块存储资源既可能构成一个隐蔽存储通道，也可能构成一个隐蔽时间通道。因此，TCSEC 要求，B2级以上安全的操作系统必须有隐蔽通道分析[8]。

访问控制模型，特别是 BLP 模型，是一类最实用的安全模型[20]。但随着信息处理系统越来越复杂，不论军事部门还是商业部门，都需要更加灵活的安全策略，需要更加细致地划分控制粒度，需要更加灵敏的信息流控制；而新的软件范型，如面向对象、超文本、虚拟机、移动代码、智能主体等，却引入了大粒度的实体；还有一些更加复杂的环境，很难区分可以独立分级标记的实体。因此，模仿"公文世界"的访问控制模型已经不能适应新的安全需求。现在已经明确，安全问题可归结为信息流的无干扰性质，而无干扰性质又可归结为系统的等价性问题[19]，后者是计算机科学中的中心问题之一。

## 参 考 文 献


1 Anderson J P. Computer Security Technology Planning Study Volume I:[ESD-TR-73-51]. Vol. I, Electronic Systems Division, Air Force Systems Command, Hanscom Air Force Base, Bedford, MA, USA, Oct. 1972

2 Anderson J P. Computer Security Technology Planning Study Volume Ⅱ: [ESD-TR-73-51]. Vol II, Electronic Systems Division, Air Force Systems Command, Hanscom Air Force Base, Bedford, MA, USA, Oct. 1972

3 Bell D E, LaPadula L J. Secure Computer Systems: Mathematical Foundations: [ESD-TR-73-278]. Vol. I, AD 770 768, Electronic Systems Division, Air Force Systems Command, Hanscom Air Force Base, Bedford, MA, USA, Nov. 1973

4 Bell D E, LaPadula L J. Secure Computer Systems: A Mathematical Model: [ESD-TR-73-278]. Vol. II, AD 771 543, Electronic Systems Division, Air Force Systems Command, Hanscom Air Force Base, Bedford, MA, USA, Nov. 1973

5 Bell D E, LaPadula L J. Secure Computer System: Unified Exposition and MULTICS Interpretation: [Technical Report ESD-TR-75-306]. MTR-2997 Rev. 1, The MITRE Corporation, Bedford, MA, USA, Mar. 1976

6 Castano S, Fugini M G, Martella G, Samarati P. Database Security, Addison-Wesley Publicating Company, 1995

7 Denning D E. A Lattice Model of Secure Information Flow. Communications of the ACM, 1976, 19(5): 236～243

8 Department of Defense: DOD Trusted Computer Security System Evaluation Criteria (The Orange Book), DOD 5200. 28-STD, 1985

9 Goguen J A, Meseguer J. Security policies and security models. In: Proc. of the 1982 IEEE Symposium on Security and Privacy, IEEE Computer Society Press, April, 1982. 11～20

10 Goguen J A, Meseguer J. Unwinding and Inference Control. In: Proc. of the 1984 IEEE Symposium on Security and Privacy, IEEE Computer Society Press, May 1984. 75～86

11 Harrison M A, Ruzzo W L, Ullman J D. Protection in Operating Systems Communications of the ACM, 1976, 19(8): 461～471

12 Lampson B W. Protection. ACM Operating Systems Reviews, 1974, 8(1): 18～24

13 Lampson B W. A Note on the Confinement Problem. Communications of the ACM, 1973, 16(10): 613～615

14 Landwehr C B. Formal Models for Computer Security. ACM Computing Surveys, 1981, 13(3): 247～278

15 McLean J. A Comment on the 'Basic Security Theorem' of Bell and LaPadula. Information Processing Letters, 1985, 20(2): 67～70

16 McLean J. Security models. in the Encyclopedia of Software Engineering (ed. John Marciniak), Wiley & Sons, Inc., 1994

17 Roscoe A W. CSP and determinism in security modelling. In: Proc. of the IEEE Symposium on Security and Privacy, IEEE Computer Society Press, 1995. 114～127

18 Roscoe A W. The theory and practice of concurrency. Prentice-Hall 1997

19 Ryan P Y A, Schneider S A. Process Algebra and Non-interference. Journal of Computer Security, 2001, 9(1-2): 75～103

20 Tanenbaum A S. Moden Operating Systems (second edition). China Machine Press, 2002


(上接第182页)

具[10]，它实现了图形用户界面、支持远程分布调试、支持多种断点以及提供了强大的变量跟踪和显示支持。但是 TotalView 的可移植性较差，而且是一个商业产品，使用起来的代价是一个不得不考虑的问题。清华大学实现了一个可移植的并行源级调试工具 Buster[11]，它支持多进程程序的源级交互调试，实现了图形用户界面，并能支持多种典型的工作站机群系统，但 Buster 不能支持远程调试。国防科学技术大学实现了一个远程并行调试工具 Pdebug[12]，但是他们采用的是自行设计的通信协议，功能有限，而且可移植性较差。

**小结** 并行调试对并行程序的开发非常重要，然而过去并行调试的主要方式是采用文本的命令行界面，远程用户一般通过 telnet 命令模拟终端来进行并行调试。

本文介绍了我们设计实现的远程并行调试器 RPB 的工作。RPB 实现了完全并行调试等强大的并行调试功能。同时 RPB 提供了友好的图形用户界面，并且使用 Java 语言和 Swing 图形工具包实现，具有良好的可移植性。RPB 通过 CORBA 技术实现了用户的远程调试，所有通信细节对用户完全透明，并且 RPB 的客户端嵌入到 Web 浏览器中，用户无需安装任何客户端程序，随时随地通过任何一个支持 Java 的浏览器就可以使用，极大地方便了用户使用并行调试工具的方式，使用户感觉不到平台的差异和并行机地理位置的差异，大大提高了并行机的好用性。RPB 的实现也较好地验证了为超级计算机在客户端提供支持远程使用的图形用户界面的可行性，让我们看到了未来超级计算机的使用方式将在新技术的支持下发生根本的变化。

## 参 考 文 献


1 吴明，陈国良，孙凝晖. 并行计算机用户环境的设计与实现. 计算机学报，2000，23(10)：1021～1027

2 安虹，陈国良，李宏，陈志辉. 曙光并行机客户端集成环境的设计. 计算机研究与发展，2001，38(增刊)：181～186

3 Lumetta S S, Culler D E. The Mantis Parallel Debugger. In: Proc. of the First Symposium on Parallel and Distributed Tools. Philadelphia, PA, 1996. 118～126

4 Wang Feng, Zheng Qilong, An Hong, Chen Guoliang. A Parallel and Distributed Debugger Implemented with Java. In: Proc. of the 31th Intl. Conf. on Technology of Object Oriented Languages and Systems (TOOLS Asia'99). IEEE Computer Society Press, 1999. 342～348

5 Netzer R H B, Miller B P. Optimal Tracing and Replay for Debugging Message-Passing Parallel Programs. In: Proc. of Supercomputing 92. Minneapolis, MN, 1992. 502～511

6 Wang Feng, An Hong, Chen Zhihui, Chen Guoliang. Completely Debugging Indeterminate MPI/PVM Programs. 软件学报，2001，12(3)

7 Sun Microsystems. Java Language—A White Paper. Sun Microsystems Computer Company, 1996

8 Geary D M 著，李建森等译. Java 2 图形设计卷Ⅰ：SWING. 北京：机械工业出版社，2000

9 Otte R, Patrick P, Roy M 著，李师贤等译. CORBA 教程：公共对象请求代理体系结构. 北京：清华大学出版社，1999

10 TotalView: http://www.etnus.com/

11 杜术，陈晓鹏，汪东升，郑纬民. Buster：一个可移植的并行调试器. 小型微型计算机系统，2001，22(10)

12 黄瑞芳，朱敏，张卫民. 远程调试的设计与实现. 计算机工程与应用，2001，27(1)