

云计算中虚拟机管理系统的研究与开发

许 波^{1,2}, 彭志平^{1,2}, 陈 珂¹, 朱兴统¹

(1. 广东石油化工学院计算机科学与技术系, 广东 茂名 525000)

(2. 广东高校石油化工故障诊断与信息化控制工程技术开发中心, 广东 茂名 525000)

[摘要] 如何有效地监控云计算中多个物理计算节点以及部署在物理计算节点上的大量虚拟机是一个非常重要的问题. 本文设计并开发了一个透明、相容、一致、易查的虚拟机管理系统. 该系统实现了查看主机和虚拟机的简要信息或者 XML 格式的详细信息, 能够对虚拟机进行开机、关机、暂停、恢复、重启、强制关闭、远程操控, 系统还实现了日志记录的功能. 测试运行表明该系统能有效地监控云计算中物理节点以及虚拟机的运行状态, 方便了对虚拟机的管理.

[关键词] 虚拟化, 云计算, 虚拟机管理, 监控

[中图分类号] TP393 **[文献标志码]** A **[文章编号]** 1001-4616(2014)01-0093-06

Research and Development of Virtual Machine Management System in Cloud Computing

Xu Bo^{1,2}, Peng Zhiping^{1,2}, Chen Ke¹, Zhu Xingtong¹

(1. Department of Computer Science and Technology, Guangdong University of Petrochemical Technology, Maoming 525000, China)

(2. Petrochemical Fault Diagnosis and Information Control Engineering Technology Development Center of Guangdong University, Guangdong University of Petrochemical Technology, Maoming 525000, China)

Abstract: How to monitor multiple physical computing nodes deployed and a large number of virtual machine on physical computing nodes in cloud computing effectively is a very important issue. A transparent, compatible, consistent and easy to check virtual machine management system is designed and developed. The system realizes the view hosts and virtual machines brief information or details in XML format, capable of virtual machine startup, shutdown, pause, resume, restart, forced shutdown, remote control, the system also enables logging function. Test runs show that the system can monitor the cloud physical nodes and virtual machines running effectively, facilitates the management of virtual machines.

Key words: virtualization, cloud computing, virtual machine management, monitoring

当今信息产业界最受关注两项新技术——虚拟化与云计算. 虚拟化是指在同一台物理机器上模拟多台虚拟机的能力^[1]. 每台虚拟机在逻辑上拥有独立的处理器、内存、硬盘和网络接口. 使用虚拟化技术能够提高硬件资源的利用率, 使得多个应用能够运行在同一台物理机上各自拥有彼此隔离的运行环境. 虚拟化实现了 IT 资源的逻辑抽象和统一表示, 在大规模数据中心管理和解决方案交付方面发挥着巨大的作用, 是支撑云计算伟大构想的最重要的技术基石^[2]. 虚拟化隔离了各种计算资源, 因此也就提供了重新分配与整合被隔离资源的机会, 以便更好、更高效地利用这些资源^[3].

云计算是对数据中心虚拟化的进一步封装. 在云计算这个层面, 虚拟机生命周期管理的权限被彻底下放给真正的普通用户, 但是也将资源池和物理机等概念从普通用户的视野中屏蔽了. 普通用户可以获得计算资源, 但是无需对其背后的物理资源有任何了解^[4]. 实质上, 云计算使计算资源管理的模式发生了改变, 最终用户不再需要系统管理员的帮助即可自助地获得和管理计算资源. 显而易见, 从虚拟化到云计算,

收稿日期: 2013-08-10.

基金项目: 国家自然科学基金(61272382)、广东省科技计划项目(2012B010100037)、广东高校石油化工故障诊断与信息化控制工程技术开发中心开放基金(512016)、茂名市科技计划项目(20120263).

通讯联系人: 许波, 博士研究生, 讲师, 研究方向: 云计算与计算智能. E-mail: xubo807127940@163.com

对物理资源的封装程度不断提高,虚拟机生命周期的管理权限逐步下放.云计算的目标是将各种IT资源以服务的方式通过互联网交付给用户.计算资源、存储资源、软件开发、系统测试、系统维护和各种丰富的应用服务,都将像水和电一样方便地被使用,并可按量计费^[5].

虚拟机管理系统是一个虚拟化和云计算两种技术相结合的产品,它是一个面向管理员的虚拟化管理工具,应该提供一个人性化的UI设计界面,方便快捷地进行各类虚拟机资源的统一操作和修改,方便对虚拟机进行管理.

1 虚拟机管理工具简介

在现阶段有三大主流的虚拟机管理工具,它们分别是VMware、Hyper-V和Xen^[6].

VMware是每家企业的首选.VMware提供可靠的虚拟机管理程序和可扩展的管理工具.这是一款性能、可靠性、扩展性和可管理性均很出众的虚拟机管理程序,所以是企业计算环境所要考虑的重要特性.

微软的Hyper-V相对来说是个新来者.从SharePoint取得的成功来看,作者认为微软搭配其服务器软件免费赠送的任何产品得到用户部署的可能性很大^[7].

Xen对已部署Linux服务器的关注预算的公司来说是个很好选择.成本与功能/效用之间总是存在取舍.当前许多公司更愿意为了降低成本而牺牲功能/效用.另外,如果贵企业的服务器基础架构高度依赖思杰,那么思杰对Xen的支持应当是采购时要考虑的一个重要因素^[8].

每一种虚拟机管理工具都有各自在某些方面上的优点,比如说,qemu的特点是可以虚拟不同的硬件平台架构,Xen则在高性能上比较突出,VMware是PC桌面领域上的霸主等等,它们都是各自相互独立的,要对创建的虚拟机进行管理的话,则必须使用相应的虚拟机管理器才可以实现这个功能.但是对于管理来说,是比较不便利的,使用了不同的虚拟化技术需要使用相应公司的管理工具,使投入和学习成本相应的增大了,因此,需要一个统一的虚拟机管理系统来解决这一问题.本系统可以解决使用不同虚拟机管理程序来管理虚拟机的问题,提供给管理者人性化的操作界面来管理虚拟机,使投入和学习成本降低.

2 搭建系统开发环境

本系统在理想情况下需要使用至少两台PC机,才能够完成系统的开发,也可以通过VMware虚拟出另外一台PC机.本系统需要的开发环境比较特殊,需要做许多配置.配置开发环境是开发这个系统的一个难点,下面将详细介绍开发环境的配置.

(1) 硬件配置

处理器需要支持虚拟化,支持64位,由于bios中默认没有开启,需要进入bios开启虚拟化,否则无法进行环境的配置.

(2) 软件配置

主机操作系统:Windows 7 Ultimate,32-bit 6.1.7601,Service Pack 1;

主机安装软件:VMware Workstation 9.0.0 build-81388;

主机开发工具:Qt Creator 2.4.1 base on Qt4.7.4(32 bit);

编译器:MinGW 4.7;

客户机操作系统:Ubuntu 12.04 桌面版 64 位(32 位无法安装成功 qemu,需要 64 位).

(3) 安装 Ubuntu 系统

在 windows 7 系统上使用 VMware 软件,创建一个虚拟机,在该虚拟机上安装 Ubuntu12.04 的系统,该系统用于安装 Qemu 和 Virt-Manager,还有运行 Libvirt 作为一个后台服务.该虚拟机的配置最好是:分配一个 CPU,配置 CPU 支持虚拟化,至少 1 G 的内存和至少 10 G 的硬盘空间.

(4) Libvirt 环境

该系统需要编译两个版本的 Libvirt,一个是 windows 下的,一个是 Linux 下的.在 windows 系统下,要使用 Libvirt 库提供的 API 的话,就需要 Libvirt 头文件和相应的动态链接库,并且在此系统下不支持把 Libvirt 作为服务在后台运行,并且 windows 下编译也比较麻烦.在 Linux 系统下编译,只需要下载官网上的源代码包,直接通过控制台的方式进行安装.

(5) Qemu 环境

选用 Qemu 作为测试的管理工具的一种,所以需要安装 Qemu. 打开 Ubuntu 的控制台,输入命令: `sudo apt-get install qemu` 即可安装. 修改 `/etc/group` 文件,将当前用户添加到 `qemu` 组里去,以便当前用户有足够的权限来控制 `qemu`.

(6) Virt-Manager 环境

Virt-manger 的安装:

解压 `virt-manager-0.9.1.tar.gz`

`cd virt-manager-0.9.1`

`./configure`

`make`

`sudo make install`

(7) Qt 项目的配置

拷贝 windows 下编译的 Libvirt 库的 `include` 文件夹到创建项目的根目录下,并将所有的 `libvirt` 的动态链接库拷贝到 `lib` 目录下,将该目录放置到影子构建目录的根目录下,最后在 Qt 项目里的 `pro` 文件添加下面的配置即可:

`INCLUDEPATH+=./include`

`LIBS+=-L./lib/-llibvirt-0`

3 系统的设计与实现

3.1 系统总体设计

根据需求分析设计出本系统的软件架构,并将各个模块功能进行相应的划分,站在全局高度上,实现整个系统的实现方案和软件结构,虚拟机管理系统的总体设计如图 1.

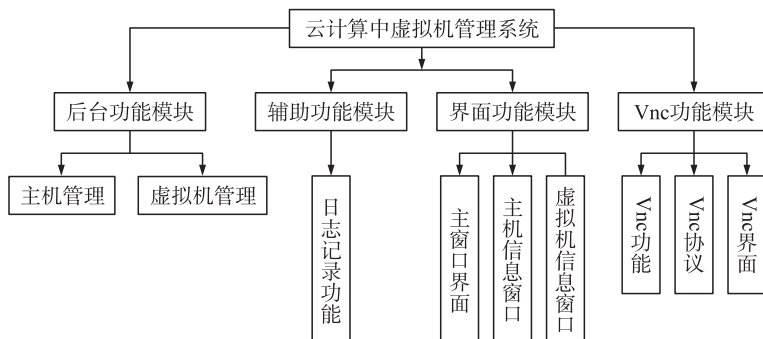


图 1 系统总体设计图

Fig.1 System overall design

后台管理模块:主要负责根据 Libvirt 提供的 API 获取远程主机的连接,取得远程主机的虚拟机管理权限,每隔 5 s,便重新获取远程主机和虚拟机的详细信息,以便远程主机或者虚拟机更新了配置等内容时,本地能够及时地更新信息给用户.

辅助功能模块:主要负责添加了一些辅助的功能,目前仅仅添加了日志记录的功能,能够将软件启动、运行到关闭过程执行的操作、出现的问题等记录到本地的日志文件中.

界面功能模块:主要负责将后台管理从远程主机获得到的主机和虚拟机信息的数据显示到界面上给管理员查看,管理员在界面上对主机和虚拟机进行的一系列操作,都会发送给后台管理模块,再由后台管理模块根据 `libvirt` API 提供的功能,发送到远程主机上进行远程操作.

Vnc 功能模块:主要负责虚拟机的远程桌面功能,可以在本地主机上,远程登录到远程主机上创建已经安装有系统的虚拟机上.

3.2 关键部分实现

虚拟机控制功能的实现:

启动虚拟机功能:

```
void VirtualMachine::startup()
{
    if(! this->isActive) {
        if( virDomainCreateWithFlags( this->domain, VIR_DOMAIN_NONE ) ) {
            this->valueStatus = VirtualMachine::Running;
            this->setVMStatus( QString( " Running" ) ); } } }
```

关闭虚拟机功能:

```
void VirtualMachine::shutdown()
{
    if( isActive ) {
        if( virDomainShutdownFlags( this->domain, VIR_DOMAIN_SHUTDOWN_DEFAULT ) ) {
            this->valueStatus = VirtualMachine::Close;
            this->setVMStatus( QString( " Close" ) );
        }
    }
}
```

强制关闭虚拟机功能:

```
void VirtualMachine::forceoff()
{
    if( isActive ) {
        if( virDomainDestroyFlags( this->domain, VIR_DOMAIN_DESTROY_DEFAULT ) ) {
            this->valueStatus = VirtualMachine::Close;
            this->setVMStatus( QString( " Close" ) );
        }
    }
}
```

暂停虚拟机功能:

```
void VirtualMachine::pause()
{
    if( isActive ) {
        if( virDomainSuspend( this->domain ) ) {
            this->valueStatus = VirtualMachine::Pause;
            this->setVMStatus( QString( " Pause" ) );
        }
    }
}
```

恢复虚拟机功能:

```
void VirtualMachine::resume()
{
    if( isActive ) {
        if( virDomainResume( this->domain ) ) {
            this->valueStatus = VirtualMachine::Running;
            this->setVMStatus( QString( " Running" ) );
        }
    }
}
```

日志功能的实现:

```
void LogRecorder::WriteLog( QString str)
```

```
{
    QDateTime now = QDateTime::currentDateTime();
    LogRecorder::GetInstance(&now);
    if( LogRecorder::GetInstance->logStream != 0)
    {
        * ( LogRecorder::GetInstance->logStream ) << now.
        toString( "[ HH:mm:ss ] " ) << str << endl;
    }
    else
    {
        qDebug() << "Log Recorder Failure";
    }
}
```

4 系统测试与运行

由于版面有限,只截取部分测试运行的界面.系统运行主界面如图 2 所示.

主机主要信息和详细信息界面如图 3 和图 4 所示.

虚拟机主要信息和详细信息界面如图 5 和图 6 所示.

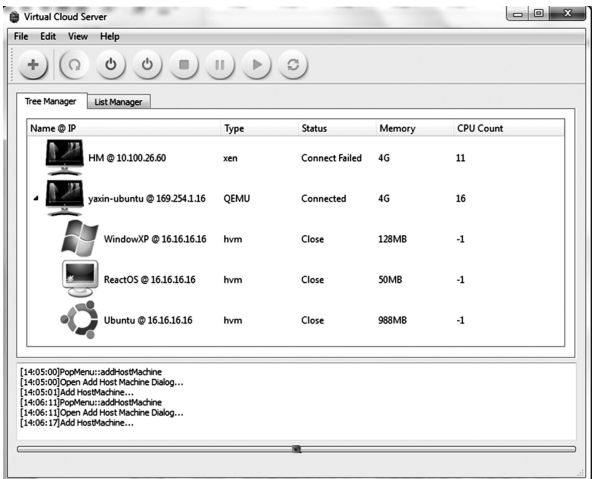


图 2 系统主界面
Fig.2 System main interface

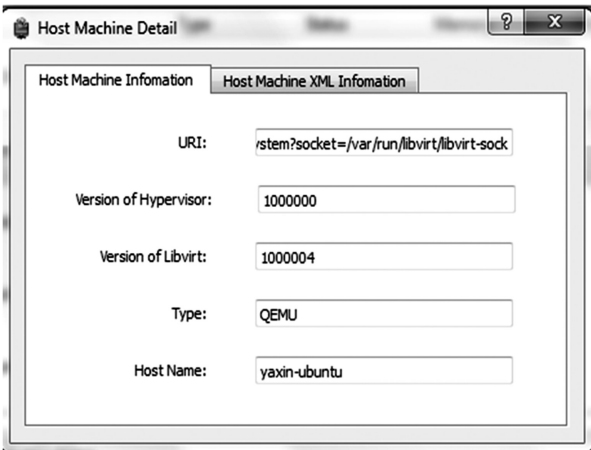


图 3 主机主要信息界面
Fig.3 Host main information interface

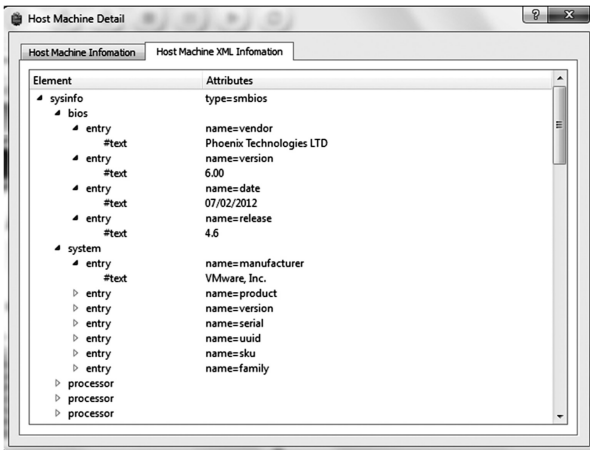


图 4 主机详细信息界面
Fig.4 Host more information interface

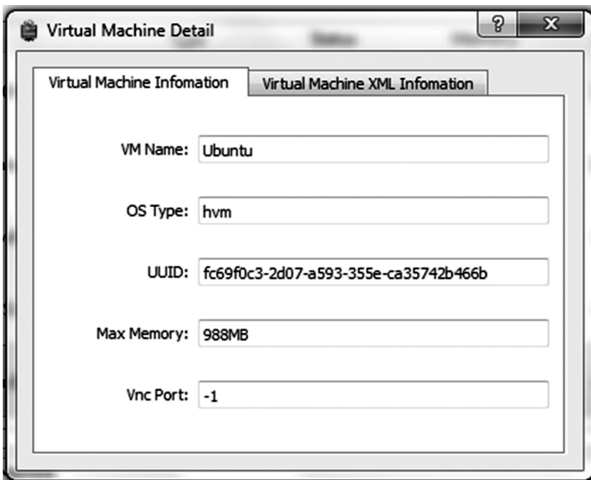


图 5 虚拟机主要信息界面
Fig.5 Virtual machine main information interface

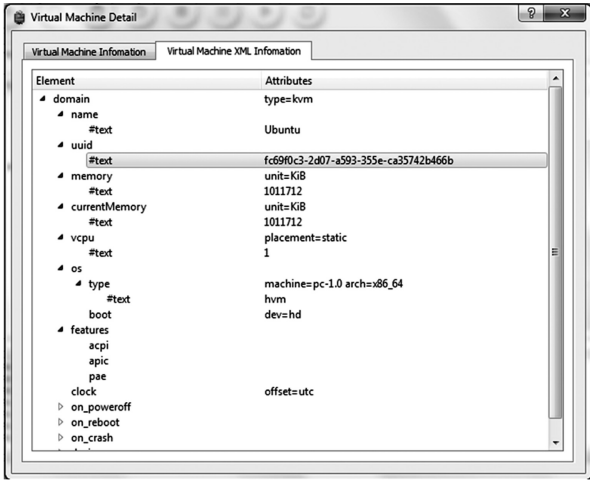


图 6 虚拟机详细信息界面
Fig.6 Virtual machine more information interface

系统是使用 Qt 来开发的,能够运行在多种平台下,包括 Windows 平台,虽然目前只实现了主要的功能,但是继续开发下去将会有不错的前景.

5 结语

本文设计并开发了一个透明、相容、一致、易查的虚拟机管理系统. 该系统实现了查看主机和虚拟机的简要信息或者 XML 格式的详细信息, 能够对虚拟机进行开机、关机、暂停、恢复、重启、强制关闭、远程操控, 系统还实现了日志记录的功能. 测试运行表明该系统能有效地监控云计算中物理节点以及虚拟机的运行状态, 方便了对虚拟机的管理.

[参考文献]

- [1] 田文洪, 赵勇. 云计算资源调度管理[M]. 北京: 国防工业出版社, 2011.
- [2] 王金波, 《虚拟化与云计算》小组编. 虚拟化与云计算[M]. 北京: 电子工业出版社, 2009.
- [3] Armbrust M, Fox A, Griffith R, et al. A view of cloud computing[J]. Communication of the ACM, 2010, 53(4): 50–58.
- [4] 刘晓茜. 云计算数据中心结构及其调度机制研究[D]. 合肥: 中国科学技术大学计算机科学与技术学院, 2011.
- [5] 谭浩宇. 多虚拟机管理平台中的监控系统[D]. 武汉: 华中科技大学计算机学院, 2008.
- [6] Mark Stillwell, David Schanzbach, Frederic Vivien, et al. Resource allocation algorithms for virtualized service hosting platforms[J]. Journal of Parallel and Distributed Computing, 2010, 70(9): 962–974.
- [7] Daniel Warneke, Odej Kao. Exploiting dynamic resource allocation for efficient parallel data processing in the cloud[J]. IEEE Transactions on Parallel and Distributed Systems, 2011, 22(6): 1 045–9 219.
- [8] 王忠儒. 云环境下的虚拟机监控和服务部署关键技术研究[D]. 长沙: 国防科学技术大学计算机学院, 2010.

[责任编辑: 丁 蓉]

(上接第 80 页)

- [5] Blond S L, Legout A, Dabbous W. Pushing Bittorrent locality to the limit[J]. Computer Networks, 2011, 55(3): 541–557.
- [6] Bindal R, Cao P, Chan W, et al. Improving traffic locality in Bittorrent via biased neighbor selection[C]//Proceedings of ICDCS 2006, Lisboa, 2006: 66–66.
- [7] 刘琼, 徐鹏, 杨海涛, 等. Peer-to-Peer 文件共享系统的测量研究[J]. 软件学报, 2006, 17(10): 2 131–2 140.
- [8] Hu C L, Lu Z X. Downloading trace study for Bittorrent P2P performance measurement and analysis[J]. Peer to Peer Networking and Applications, 2012, 5(4): 384–397.
- [9] IPOQUE. Internet Study 2008/2009[EB/OL]. http://www.ipoque.com/resources/internet-studies/internet-study-2008_2009,2009-2-18.
- [10] Liu J C, Wang H Y, Xu K. Understanding peer distribution in the global internet[J]. IEEE Network, 2011, 24(4): 40–44.
- [11] Wang H, Liu J, Xu K. On the locality of bittorrent-based video file swarming[C]//Proceedings of IPTPS 2009, Boston, 2009: 12–12.
- [12] Hobeld T, Lehrieder F, Hock D, et al. Characterization of Bittorrent swarms and their distribution in the Internet[J]. Computer Networks, 2011, 55(5): 1 197–1 215.
- [13] Zhang C, Dhungel P, Wu D, et al. Unraveling the Bittorrent ecosystem[J]. IEEE Transactions on Parallel and Distributed Systems, 2011, 22(7): 1 164–1 177.

[责任编辑: 顾晓天]