

基于 Windows 的在线判题系统的安全性研究

李定才,瞿绍军,胡 争,段 兵,成幸毅,唐 强

(湖南师范大学 数学与计算机科学学院,湖南 长沙 410081)

摘 要:在线判题系统(OJ)是 ACM/ICPC 竞赛选手用来进行程序设计练习和比赛的平台,现有 OJ 在遇到恶意提交代码方面经常造成服务器故障、崩溃或硬盘阻塞等安全威胁。为参加 ACM/ICPC 竞赛选手提供安全、可靠和高性能的 OJ 平台,在保证安全性的同时又不影响使用性。论文从源码级控制、应用程序级控制与防止恶意提交方面作了深入研究,提出基于 sandbox(Windows 沙箱)、ACL(访问控制列表)、完美哈希和帐号保密等技术的安全体系结构,通过在 Windows 下搭建 OJ 平台验证了采用此体系结构的 OJ 彻底解决了前面的安全问题。OJ 安全可靠,性能优良。

关键词:国际大学生程序设计竞赛;在线判题;Windows;沙箱;访问控制表;完美哈希

中图分类号:TP309

文献标识码:A

文章编号:1673-629X(2011)09-0204-04

Research on the Safety of Online Judge System Based on Windows

LI Ding-cai, QU Shao-jun, HU Zheng, DUAN Bing, CHENG Xing-yi, TANG Qiang

(College of Mathematics and Computer Science, Hunan Normal University, Changsha 410081, China)

Abstract: Online Judge(OJ) system is the platform for ACM / ICPC programming players, the existing OJs now meet the server trouble, service stop, hard disk choke and other security threats because of malicious submits. To provide safe, reliable and high performance OJ platform for ACM / ICPC programming players, ensure both the safety and the usability at the same time. Study from source-level control, application-level control and preventing malicious submits, put forward measures that based on sandbox (Windows sandbox), ACL (Access Control List), the perfect Hash, account security and other technical that formed security technical architecture, by practicing this architecture based on Windows, this architecture can solve all problems above, and was verified safe, reliable, and with good performance.

Key words: ACM/ICPC; online judge; Windows; sandbox; access control list; perfect Hash

0 引言

ACM/ICPC 程序设计竞赛是目前全球影响力最大的计算机程序设计竞赛^[1]。ACM/ICPC Online Judge(一般简称 OJ,即在线判题)是选手们用来进行练习和比赛的平台。目前国外影响力较大的 OJ 有 uva(<http://acm.uva.es>)^[2], ural(<http://acm.timus.ru>); 国内影响力较大的 OJ 有 poj(<http://poj.org>), hdoj(<http://acm.hdu.edu.cn>)。作者所在学校的 HUNNU OJ(<http://acm.hunnu.edu.cn/online>)在国内也有一定的影响力。

OJ 的运行方式是:用户提交源代码至服务器,服务器编译用户的源代码,然后执行源代码生成的可执行文件(或用解释方式执行,或直接执行脚本文件^[3]),得到其输出的结果与正确结果比较,如果编译

正确,程序运行没有超出限定的时间和限定的内存,而且输出和正确答案一致,则认为用户提交的源代码正确,即 Accepted(AC);否则报出相应的错误,如 Presentation Error(PE), Wrong Answer(WA), Runtime Error(RE), Time Limit Exceeded(TLE), Memory Limit Exceeded(MLE), Output Limit Exceeded(OLE), Compile Error(CE)等。

由于使用 OJ 的大多是精于计算机编程的学生、老师和工程师,而 OJ 又需要执行用户提交的源代码,在安全方面有极大的特殊性,也十分具有挑战性。文中就 Windows 平台下 OJ 系统的安全性进行了深入的研究,构建出了一个安全,能应付多种攻击方法的 OJ。

1 安全体系结构

Windows 一直被认为是不安全的操作系统,实现基于 Windows 操作系统平台的安全性好的 OJ 也极其困难。但是,只要熟知 Windows 下编程技术,架设安全的 OJ 是可行的。文中方案使用 Windows 沙箱(sandbox)或说是作业(Job)技术来运行用户程序,以增强安

收稿日期:2011-02-28;修回日期:2011-06-24

基金项目:湖南师范大学教改项目(2008-24);2010 年湖南省大学生研究性学习和创新性实验计划项目(201012)

作者简介:李定才(1989-),男,湖南平江人,研究方向为数据库。

全性^[4],运行用户程序时,用新生成的用户帐号以保护系统。对于恶意重复提交的情况,也必须进行相应的处理。另外,对于用户帐号,也简要讨论了保护策略。

在 Windows 下的 OJ 安全性方案,目前没有哪一种方案能够一劳永逸,应当综合运用多种措施,多重设防,以达到安全目的^[5]。本系统安全体系结构如图 1 所示。

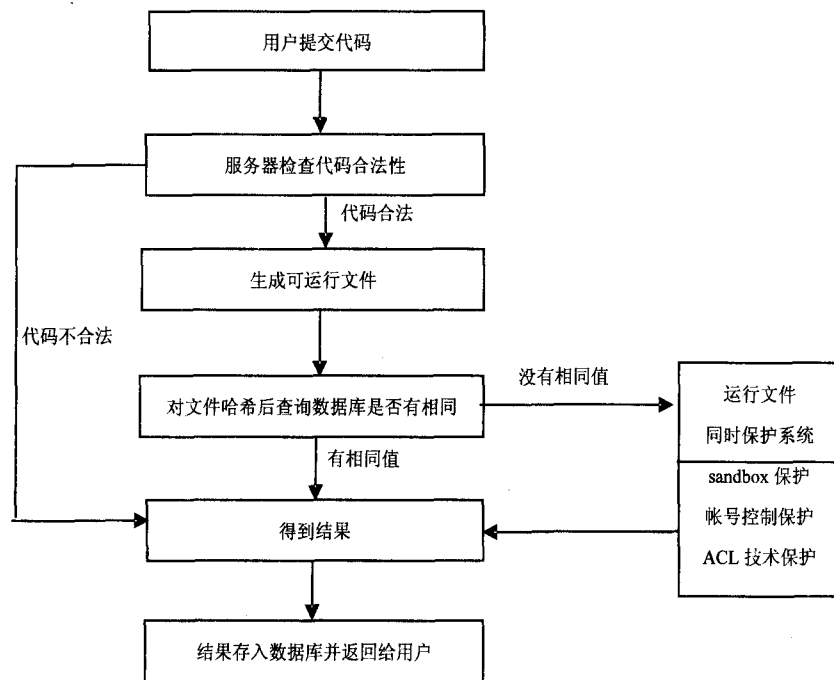


图1 系统安全体系结构

2 具体方法

2.1 源代码级限制

对于用户提交的代码,由于 OJ 只是用来测试算法程序,不必也不应该出现调用 Windows API 的代码,所以,应当将服务器的编译器中的 Windows 系统头文件全部删除。典型的,如 windows.h。如果用户提交的代码中调用了相关的系统 API,则直接报 Compile Error,起到很好的设防作用。当然,这并不能防止用户直接拷贝头文件所有内容,加在自己提交的程序的前面来避免 Compile Error。因此,有必要进一步限制用户提交的源代码文件的大小,以防拷贝头文件。要注意,这时候,不能直接将 Windows 相关的库也删掉,因为那样不只用户程序运行不了,服务器也会运行不了。另外,可能影响系统的调用也要删除,如 system(const char*) 系统调用,直接在相应的头文件里删除相应的函数声名即可。

为防止用户拷贝头文件,更安全的做法是对提交上来的源代码,首先进行检查,发现有限制函数的声名

或调用时,报出 Restricted Function(函数限制)错误。这能在很大程度增加系统的安全性,过程如图 2 所示。

```

ResultType CheckSourceCode(File SourceCode) //检查代码合法性
{
    for(all functions in source code) //得到代码中所有的用户自定义的函数声名
    if(any functions equals to restricted function) //如果某个声名出现在禁止列表中
        return RestrictedFunction; //直接返回有禁用函数错误
    return OK;
}
  
```

图2 源码级检查代码

2.2 应用程序级限制

对于用户提交的代码,无法从源代码判断结果,必须要编译,运行之后才知道结果。在执行生成的可执行程序(或解释脚本或字节码)时,用户程序可能执行恶意代码。所以,将应用程序放入一个 Windows 进程沙箱中,以限制进程的行为。Windows 沙箱能够限制进程的很多行为,这正是安全所需要的。

Windows 提供了一个作业(job)内核对象,它允许将进程组合在一起并创建一个“沙箱”来限制进程能够做什么。可以将作业对象想象成一个进程容器,但在 OJ 里,沙箱里只放入一个进程,而且,

还有必要限制沙箱里的进程不能再生成进程。

具体而言,将要运行的程序或命令创建进程,然后放入沙箱,在 OJ 里,Windows 提供的沙箱对进程进行以下限制^[12]:

PerProcessUserTimeLimit:限制沙箱中单个进程运行的最长时间。

MinimumWorkingSetSize/MaximumWorkingSetSize:限制沙箱中单个进程所能使用的最少/最多内存。

ActiveProcessLimit:限制沙箱中能运行的最多进程数目,在 OJ 中,这个值一定为 1,因为要限制用户代码再生成别的进程。

JOB_OBJECT_UILIMIT_EXITWINDOWS:阻止进程注销,关机,重启或断开系统电源。

JOB_OBJECT_UILIMIT_READCLIPBOARD:阻止进程读取剪贴板中的内容。

JOB_OBJECT_UILIMIT_WRITECLIPBOARD:阻止进程清除剪贴板中的内容。

JOB_OBJECT_UILIMIT_SYSTEMPARAMETERS:阻止进程更改系统参数。

JOB_OBJECT_UILIMIT_DISPLAYSETTINGS:阻止

进程更改显示设置。

JOB_OBJECT_UILIMIT_GLOBALATOMS: 为作业指定其专有的全局原子表,并限定作业中的进程只能访问此作业的表。

JOB_OBJECT_UILIMIT_DESKTOP: 阻止进程创建或切换桌面。

JOB_OBJECT_UILIMIT_HANDLES: 阻止作业中的进程使用同一个作业外部的进程所创建的用户对象(如 Hwnd)。

通过将进程放入沙箱中,使得用户程序无法进行危及系统安全和运行的操作。有沙箱的限制,用户程序已经失去了一个 Windows 程序的大半能力,只能作为算法程序来运行。

但要注意,这并不能防止用户程序进行垃圾行为,如向服务器硬盘中写入大量垃圾数据。

对此,有必要采用 windows 系统下 OS 级别的帐户控制与 ACL (Access Control List, 访问控制列表)^{[6][7][8]} 技术进行防范。在服务器上新建一个帐户,要特别注意不能使用控制面板新建,而应该用管理一>系统工具新建,因为新建的帐户要求安全,而且不应出现在 Windows 系统登录界面上。

对于新帐户的权限,设置其为允许登录,但是对于所有的驱动器或文件,都拒绝其读写权限。可以用 Windows 下的工具 cacls.exe 做到这一点。

最后,新建一个文件夹,这个文件夹对新建的帐户放开读写权限,而且,将用户程序标准输出重定向到这个文件夹里。另外,对于服务器中的数据文件,放开该用户的读权限。

用新建的帐号来运行用户程序。用户程序只能写指定的文件夹。每一次判题前,首先清空该文件夹即可。由于 ACM 的程序都是有时间限制的,在几秒钟或几十秒钟之间,用户不可能向服务器中写大量垃圾数据致其阻塞,而每次判题前都清空了这个文件夹,使得服务器中不会存有大量垃圾数据。同时,用户程序只能对这个文件夹进行写操作,对服务器其它文件已没有威胁。

2.3 防止恶意提交

恶意的用户可能重复提交同一份代码,或是几个,甚至若干个人同时提交同一份代码,要对此采取一定的防范措施(见图3)。

有两种方法可供选用。一是将用户提交的代码进行 perfect Hash(完美哈希),结果存入数据库中。二是将用户提交的代码生成可执行文件或字节码(当然也有脚本文件本身可运行的,如 python 语言^[3])进行 perfect Hash,Hash 值存入数据库中。当收到用户的提交代码时,首先进行 perfect Hash,得到 Hash 值,在数据库中查找有无此 Hash 记录,如有,则直接在数据库中调出相应的结果,否则,仍然进行一般的判题处理。

```

ResultType ProtectRun(File ExecutableFile)           //防范式运行
{
    HashValue=perfectHash(ExecutableFile);           //进行完美哈希
    if( find(HashValue record in Database) )           //查找数据库中是否有此哈希值
        return record.result;                         //如有, 则直接返回结果
    LoginOS(new ID,password);                         //用低权限帐户登录系统
    Process RunProcess=CreateProcess(ExecutableFile); //创建执行可运行文件的进程
    Sandbox RunSandbox=CreateSandbox();               //创建沙箱
    SetLimitToSandbox(Sandbox,Limit);                 //对沙箱施加限制
    Putinto(RunProcess,RunSandbox);                   //把创建好的进程放入沙箱
    RunProcess.start();                                //开始运行
    RunProcess.stop();                                //因运行时超出限制或已运行完, 进程结束
    CheckResult();                                    //比对结果
    return Result;                                     //返回结果
}

```

图3 应用程序级与防恶意提交代码

这么做的好处:一是防范了用户若干次重复提交恶意代码;二是对于相同的代码或生成的文件,总是有相同的结果,体现了 OJ 的公平性;三是由于利用了 Hash 机制,使得服务器运行速度加快^{[9][10][11]};四是由于使用了判重,可以用来防作弊。但要注意,保存源代码的 Hash 值与保存生成文件的 Hash 值的效果,是略有区别的。使用源代码的 Hash 值,如果用户仅仅更改变量名或注释等,是起不到太大作用的,其防作弊功能较弱。但保存生成的可执行文件的 Hash 值,由于编译器的优化机制,这种实质上相同的代码仍会被判为相同,有较好的防作弊功能。

2.4 用户帐号密码与数据保密

对于用户密码,在其注册与登录时,应当进行加密传送,在数据库中保存也应以密文保存,使其难于被盗。这里可以采用成熟的 MD5 加密^[12]。对于服务器数据库^[13],因安全因素,不宜采用 Access, Foxbase 等小型数据库,但由于成本等因素,亦不宜采用 Oracle, db2 等大型数据库,建议采用中型数据库 Mysql,在节约成本的同时很好地管理和保密信息。

2.5 服务器监控

以上解决了判题系统自身的安全问题,但判题系统是运行在 Windows 系统下的,需要对 Windows 进行相关的监控。主要是系统使用内存的监控,可以利用 Windows 下的相关工具完成^[14],也可以用 Windows API 核心编程来完成,此处不再详述。

表 1 OJ 测试评价对照表

测试方法	调用 windows API 攻击	调用 system 删去重要文件	拷贝头文件后调用 window API 或 system	恶意写垃圾数据	考试与比赛时拷贝他人代码作弊
实验使用 OJ	返回 Compiler Error	返回 Restricted Function	返回 Restricted Function	服务器不受影响	大部分能识别
现有普通 OJ	服务器仍能运行,但服务器已出现故障	服务器崩溃	服务器出故障或崩溃	服务器硬盘满,阻塞	完全不能识别

3 测试与评价

直接采用提交恶意代码的方式对整个 OJ 系统进行评测,并与目前普遍采用的 OJ 进行对照测试评估,结果见表 1。

4 结束语

综上所述,该方案大大增强了现有 OJ 的安全性,有效地防范了用户的多种恶意行为。系统采用源码级控制,sandbox 技术,帐户管理,ACL 技术,哈希等方法,多重设防,在 Windows 平台下搭建了十分安全的 ACM OJ 平台。方案的不足之处,正是因为使用了多种技术,操作较为复杂,要求技术人员对 Windows 的相关技术十分熟悉,方能达到理想的安全。

参考文献:

[1] 瞿绍军. 以学科竞赛为载体,培养大学生创新能力——以大学生程序设计竞赛为例[J]. 电脑知识与技术,2010,6(15):3980-3981.

[2] Revilla M A,Manzoor S,Liu Rujia. Competitive Learning in Informatics: The UVa Online Judge Experience[J]. Olympiads in Informatics,2008(2):131-148.

[3] 孙广磊. 征服 python——语言基础与典型应用[M]. 北京:人民邮电出版社,2007:3-6.

[4] Richter J,Nasarré C. Windows 核心编程[M]. 第 5 版. 葛子昂,周靖,廖敏译. 北京:清华大学出版社,2008:121-139.

(上接第 203 页)

(4):366-370.

[2] Still C K. Crowd Dynamics[D]. Warsick:University of Warsick, 2000.

[3] Helbing D. Traffic and related self-driven many-particle system[J]. Reviews of Modern Physics, 2001,73:1067-1141.

[4] 方正,卢兆明. 建筑物避难疏散的网格模型[J]. 中国安全科学学报,2001,11(4):10-14.

[5] 方正,卢兆明. 试论建筑物人员疏散的量化研究[J]. 武汉大学学报,2002,4(8):79-84.

[6] Foschini G J, Gans M J. On limits of wireless communications in a fading environment when using multiple antennas[J]. Wireless Pers. Commun., 1998,10(2):315-335.

[7] 田翠华,于天放. 基于 Agent 技术的交通流仿真研究[J]. 计算机技术与发展,2010,20(2):232-235.

[5] 杨志伟,曾艳珊. 基于 Linux 的 ACM 在线评测系统研究[J]. 计算机与现代化,2010(6):166-169.

[6] Ferraiol D,Kuhn D R. Role based access control[C]// Proceedings of the 15th Annual Conference on National Computer Security. Gaithersburg:MD National Institute of Standards and Technology,1992:554-563.

[7] Li Shoupeng, Wu Shizhong, Guo Tao. The Consistency of an Access Control List [C]//Information and Communications Security,4th International Conference, ICICS 2002. Singapore:[s. n.],2002:367-373.

[8] 李秋敬,刘广亮. 基于时间约束的角色访问控制模型研究[J]. 计算机技术与发展,2009,19(8):162-165.

[9] Cichelli R J. Minimal Perfect Hash Functions Made Simple [J]. Communications of the ACM,1980,23(1):17-19.

[10] Hagerup T,Tholey T. Efficient minimal perfect hashing in nearly minimal space[C]// Proceedings of the 18th Symposium on Theoretical Aspects of Computer Science (STACS ' 01),2001. [s. l.]:Springer,2001:317-326.

[11] 林雅榕,侯整风. 对哈希算法 SHA-1 的分析和改进[J]. 计算机技术与发展,2006,16(3):124-126.

[12] Touch J D. Performance Analysis of MD5 [J]. ACM SIGCOMM Computer Communication Review, 1995,25(4):77-86.

[13] Groff J R. Weinberg P N. SQL 完全手册[M]. 第 2 版. 章小莉,宁欣,汪永好,等译. 北京:电子工业出版社,2004:2-9.

[14] 姜博. 浅谈 Windows 环境下站点服务器的安全与攻防[J]. 计算机安全,2010(5):96-98.

[8] Sandhu R, Conye E. Role - based Access Control Models [J]. IEEE Computer,1996,10(7):125-130.

[9] 朱雅丽,熊前兴. 移动 Agent 在电子商务中的应用研究[J]. 计算机与数字工程, 2008(4):165-166.

[10] 陈建刚,王汝传. 基于模糊集合的网格资源访问的信任机制[J]. 计算机学报, 2009,32(8):1676-1682.

[11] 黄智维,倪子伟. 网格计算环境下资源管理的研究[J]. 计算机技术与发展,2009,19(3):200-204.

[12] 李钦,余谅. 基于免疫遗传算法的网格入侵检测模型[J]. 计算机技术与发展, 2009, 19(5):162-169.

[13] Chen J G, Wang R C, Wang H Y. The extended RBAC model based on grid computing[J]. The Journal of China Universities of Posts and Telecommunications, 2006,13(3):93-97.