第32卷 第22期 计 算 机 工 程 2006年11月
Vol.32 No.22 Computer Engineering November 2006
·软件技术与数据库· 文章编号：1000—3428(2006)22—0094—02 文献标识码：A 中图分类号：TP311

性能测试中脚本捕获的方法研究与应用

刘 严[1]，胡 敏[2]
（1. 华东工程软件测评中心，上海 200233；2. 上海理工大学计算机工程学院，上海 200093）

摘 要：脚本捕获是性能测试的重要组成部分之一，该文介绍了脚本捕获的基本原理和过程，并利用API挂接技术实现了在Windows操作系统上对Oracle数据库访问过程的脚本捕获，并由此提出了一种通用的性能测试脚本捕获方法。
关键词：性能测试；脚本捕获；输入节；挂接API

Research and Application of Script Capture During Performance Test

LIU Yan[1], HU Min[2]
(1. Testing Center of East China Engineering Software, Shanghai 200233;
2. School of Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093)

【Abstract】 Script capture is a very important part of performance test. This paper introduces the basic theory and procedure of script capture, and implements a simple model to capture scripts during Oracle accessing on MS Windows OS. Based on this model, a common method of script capture is proposed.
【Key words】 Performance test; Script capture; Import table; API hook

随着网络技术的迅速发展，分布式应用软件得以广泛开发和应用。在分布式应用环境中，服务器性能对整个系统的成功与否有着决定性的影响。因此，性能测试逐渐成为软件验收测试的重要组成部分之一。其中，脚本开发是不可或缺的一环，一般的自动化性能测试工具都会包含脚本开发模块。该模块应该向用户提供脚本捕获的功能。所谓脚本捕获，就是将用户在客户端的操作记录下来并形成脚本，这些操作包括对服务器资源的请求和访问等。脚本形成之后，测试人员根据需要对其进行修改，通过对修改后脚本的大规模调度执行来达到模拟多个用户并发访问服务器的效果。

目前，在性能测试领域，国外已有一些商业化产品，而国内则处于摸索和交流阶段。本文以Oracle数据库服务器负载性能测试为例，提出并实现了一种通用的脚本捕获方法。

1 脚本开发模块简述

自动化性能测试工具的脚本开发模块的功能是产生满足特定要求的测试脚本，一般由脚本捕获、脚本转换和脚本建模3部分组成。其中脚本捕获是整个开发过程的基础，它捕获用户的操作行为，生成测试脚本的骨架。通过脚本开发模块得到的脚本是测试过程中虚拟用户的行为依据，大量的虚拟用户通过对一个或一组脚本的回放，模拟出生产环境下有意义的场景。

最初的性能测试中没有明确的脚本开发环节，想要实现对场景的模拟，往往是由研发人员根据系统提供的开放编程接口编写各个测试用例的模拟程序，然后通过批量运行这些模拟程序，以达到测试系统负载性能的目的。这种方法不仅要求测试人员对被测系统有十分深刻的理解，而且还需要较丰富的程序设计经验，最重要的是，该方法不具有通用性，代码的可重用率差。同时，最后的测试过程也比较难以控制。如果能把一个普通用户对系统的访问全程记录下来，那接下来的测试无疑将会便捷得多。

2 脚本捕获原理

2.1 脚本捕获的目标

脚本既然是对用户行为的记录，那么首先要界定“用户行为”的范围。以访问Oracle数据库为例，登录数据库、查询操作等都是用户行为，属于需要捕获的目标，而在捕获过程中如果用户打开IE浏览器并访问了一个与该数据库无关的Web服务器，则不属于捕获的范围，在脚本中将不予记录。换句话来说，脚本捕获前，用户必须选择此次会话所用到的协议，捕获过程中，只会记录与所选择协议有关的操作。

Oracle本身提供了一套编程接口工具，称为Oracle调用层接口(Oracle Call Interface, OCI)。由头文件和库函数组成，支持Windows NT和Windows 95/98操作系统，并支持多种C语言编译器，使程序员能够使用已熟悉的第3代程序设计语言(如C、C++等)中的编程技术和开发环境，并通过Oracle的OCI函数调用快速开发Oracle数据库应用程序。用户访问Oracle数据库的任何操作，都有OCI调用与之对应。所以脚本捕获的目标，就是将这些OCI接口调用全部记录下来。

以Oracle8.0为例，它所有OCI的实现，都作为输出函数封装在两个动态链接库中：ORA805.dll和OCI.dll。只要可以记录下应用程序对它们的输出函数的调用情况，最基本的脚本就产生了。为了实现这一点，需要深入了解一些关于动态链接库的知识。

2.2 DLL与模块的输入节

在创建一个从DLL模块输入函数和变量的可执行模块的过程中，链接程序需要对输入符号进行转换，此时它就将

作者简介：刘 严(1982－)，男，硕士生，主研方向：软件工程，软件测试理论；胡 敏，硕士生
收稿日期：2005-12-04 **E-mail：**fateditor@hotmail.com

一个称为输入节的特殊的节嵌入产生的可执行模块。输入节列出了该模块需要的 DLL 模块以及由每个 DLL 模块引用的符号。由于 DLL 本身也可能引用到其它 DLL 模块中的符号，因此每个 DLL 模块自身也包含一个输入节。

使用 Visual Studio6.0 提供的 Dependency Walker 或者 DumpBin.exe 实用程序可以很方便地观察可执行模块或 DLL 的输入节信息。通过这种方法观察 Oracle8.0 提供的客户端程序 PLUS80W.exe 可以看到，它的输入节为其所需要的每个 DLL 设置了一个项目，包括 ORA805.dll、CORE40.dll、NLSRTL33.dll、sqllib80.dll、KERNEL32.dll、USER32.dll、GDI32.dll、comdlg32.dll 和 MSVCRT.dll。在每个 DLL 的模块名下面，都有一个 PLUS80W.exe 从该特定模块输入的符号列表。在 ORA805.dll 项目中，可以看到 PLUS80W.exe 引用到的所有 OCI 接口函数。

当试图运行 PLUS80W.exe 这个可执行模块时，操作系统的加载程序将为新进程创建一个虚拟地址空间。可执行模块被映射到新进程的地址空间。加载程序对可执行模块的输入节进行分析。对于该节中列出的每个 DLL 名字，加载程序要找出用户系统上的 DLL 模块，再将该 DLL 映射到进程的地址空间。DLL 模块本身也拥有自己的输入节，进程初始化过程中还需要分析每个模块的输入节并将所有需要的 DLL 映射到进程的地址空间。当所有的 DLL 加载完毕之后，PLUS80W.exe 的初始化就结束了，此时进程中的所有线程都可以无限制地调用 DLL 中的函数，就好像它们本来就属于这个可执行模块一样。

那么如何得知应用程序在执行过程中调用了哪些 OCI 接口函数呢？一个可行的方法就是，在程序初始化时改写其输入节中的地址，使其指向我们设计的伪 OCI 函数(Hook 函数)。当模块调用一个输入函数时，线程实际上要从模块的输入节中获取需要的输入函数的地址，然后转移到该地址。一旦伪 OCI 函数被调用，就可以判断程序本来是准备调用哪个 OCI 函数，并予以记录。这就是 API Hook。

2.3 API Hook

要改写可执行模块的输入节，并不是一件十分容易的事情。在 Microsoft Windows 操作系统中，每个进程都有自己的私有地址空间，任何进程都不能创建引用属于其它进程的内存指针。可执行模块的输入节驻在该程序主进程的地址空间中，要获得对该块内存的修改权限，必须要把我们的代码挂接到该进程中去。Windows 提供了好几种将 DLL 插入到另一进程的地址空间的方法，我们使用的是 Windows 挂钩(hook)。

Hook 是 Windows 操作系统中非常重要的一种系统接口，是消息处理机构的一部分。它设计的本意是用来监视系统中某个或所有线程处理的消息，这些消息可以是线程接收到的鼠标、键盘的信息等。很多木马程序就是通过它来窃取用户的密码的。本质上，它是一段处理系统消息的程序，通过系统调用，将其消息处理函数所在的 DLL 插入到线程所在进程的地址空间中，在消息达到目的地之前，该段代码会拦截并处理消息，并选择是否将消息发向目的地。

Hook 的使用十分简单而功能却十分强大，它几乎可以拦截系统中所有的消息并加以处理，但是我们要用到的却不是它的消息处理功能。如上所述，hook 挂接入线程之后，会将 DLL 插入线程所在的进程中，所以只要将改写输入节的代码放入该 DLL，代码就可以获得访问内存的权限。

Hook 是通过下面的函数来安装的：

HHOOK SetWindowsHookEx(int idHook,HOOKPROC lpfn, HINSTANCE hMod,DWORD dwThreadId);

其中，idHook 指定 hook 类型；lpfn 为指向 hook 函数，也就是消息处理函数的指针；hMod 指定钩子函数所在的 DLL 的句柄；dwThreadId 为要挂接线程的 ID 号，如果为 0，则表示为系统中所有线程安装 Hook。当 Hook 被安装到某个线程中后，系统会将 Hook 的消息处理函数所在的 DLL 强行加载给该线程所在的进程。所以将修改可执行模块输入节的代码也放入这个 DLL 中，并设法令此段代码被调用。有 3 种方法可以实现代码的自动调用：(1)写在 DllMain 中；(2)在 dll 中定义全局变量，将代码写在其构造函数中；(3)将代码放在消息处理函数中调用。前两种方法中，改写输入节的代码都是在 DLL 初始化的时候被调用的，相比起来可靠性更强。我们采用第(1)种方法来实现。

2.4 改写可执行模块的输入节

现在已经获得了修改进程内部数据的权限，那么如何修改输入节呢？首先介绍一种实际的情形。比如说，运行 PLUS80W.exe，该模块中的代码调用 ORA805.dll 中包含的 OCIInitialize 函数，但是想要让它实际调用我们的 HOOKORA805.dll 模块中包含的 MyOCIInitialize 函数。为了完成这个操作，需要做下面几步工作：

(1)获得 PLUS80W.exe 的输入节描述符。可以通过调用函数 ImageDirectoryEntryToData 来实现。该函数的原型如下：

PVOID ImageDirectoryEntryToData (PVOID Base, BOOLEAN MappedAsImage, USHORT DirectoryEntry, PULONG Size);

将第 3 个参数设置为 IMAGE_DIRECTORY_ENTRY_IMPORT，即可获得指定模块的输入节描述符指针。

(2)定位 PLUS80W.exe 的输入符号地址表。该表存放的是模块所引用到的所有 DLL 输出函数名称以及当它被调用时程序的跳转地址。它的位置可以由模块在内存中的基地址和输入节描述符指针所指示的偏移地址共同确定。

(3)遍历输入符号地址表，找到 OCIInitialize 项，将该项的跳转地址改为 MyOCIInitialize 函数的地址。修改的工作可以由 WriteProcessMemory 函数来完成。

需要注意的是，在脚本捕获的过程中，希望用户感觉不到任何异常。所以在 MyOCIInitialize 函数中除了记录下函数调用情况外，最终仍会调用到原始的 OCIInitialize 函数，以完成用户想要的操作。

3 对 Oracle8i 数据库的实现及实验结果

3.1 实现

我们在开发自动化测试工具时，实现了对 Oracle 数据库访问过程中用户行为的捕获并生成了文本文件。Oracle 数据库版本是 8.0，客户端使用 SQL Plus8.0，也就是 PLUS80W.exe。我们替换了 ORA805.dll 中 69 个 OCI 接口函数，基本覆盖了数据库访问过程中全部可能被调用到的 OCI 函数。实验证明该工具对用户访问 Oracle 数据库操作的捕获是有效和可靠的。为了更精确地记录用户的行为，还捕获了操作与操作间的间隔。这是通过一个全局时间变量来实现的。

3.2 实验结果

使用 VC++开发的脚本捕获函数，捕获一次对 Oracle8 数据库操作后生成的文本文件部分内容如下：

OCIInitialize|0|2|0|0|0|0

OCIEnvInit|0|16182256|0|0|0

OCIHandleAlloc|0|16182256|16603832|2|0|0

OCIHandleAlloc|0|16182256|16602676|2|0|0 (下转第 123 页)