转 入侵Tomcat服务器一次实战

2013年10月08日 23:50:37　阅读数：11921

到网上随便逛逛，我就会发现用JSP制作的电子商务网站多如牛毛，从JSP日渐繁荣的局面来看，适合于各种平台而且免费的Tomcat逐渐成为WEB服务器的一种选择。eBay.com与Dell计算机等知名网站都采用或者曾经采用Tomcat的Container容器执行Servlet　与JSP，可想而知这个服务器软件所具有的前途。那他们的安全性如何呢?一起来看看吧!

小知识：Tomcat是Sun的JSWDK(JavaServer Web Development Kit)中Servlet的运行环境(Servlet容器)。Tomcat的源代码被提供给Jakarta项目，在Open Source的模型下进行进一步的开发。Tomcat是一个Server容器，同样的，EJB运行在EJB的容器中。对于初学者来说，可以这样认为：当在一台机器上配置好Apache服务器，可利用它响应对HTML页面的访问请求。实际上Tomcat部分是Apache服务器的扩展，但它是独立运行的，所以当你运行Tomcat时，它实际上作为一个与Apache独立的进程单独运行的。也就是说当配置正确时，Apache为.HTML页面服务，而Tomcat实际上运行.jsp页面和Servlet。

先看看JSP的安全机制。JSP和PHP、ASP等语言的工作机制不一样，虽然它也是一种WEB编程语言，但首次调用JSP文件其实是执行一个编译为Servlet的过程——敏感部位暴露出来了，如果能够让JSP在编译前被浏览器当作一个文本或其它文件发送给客户端，或在JSP装载的时候不去执行编译好的Servlet，而直接读JSP的内容并发送给客户端，会出现什么问题呢?

对于Tomcat这样公布源代码的程序，若有更多的人使用并研究其代码，则可以找到并修补愈多的安全漏洞。但是，对从来不修补漏洞的人们来说，我就可以利用这些漏洞，做我喜欢做的事情了。

前一段时间应朋友之邀，我对他们托管的三台主机进行了测试，其中两台为Linux，一台为Windows系统，在路由器后面还有一台Cisco PIX 525对三台主机进行保护，只允许外部用户连接不同主机的部分端口，例如80，25，110。根据检测，Cisco PIX防火墙过滤规则设置比较严密，没有多余端口允许外部用户访问。细致分析后，发现目标网络的主机通过地址转换来提供对外访问，内部使用192.168.*.*地址段。先找个扫描软件来看看主机的安全情况。

我找来了X-Scan，在外部对这几台主机进行了端口扫描之后，生成了一份关于端口的报表，发现其中有一个Tomcat服务器，解释的自然就是JSP文件了。

尝试一下通过WEB服务进行间接攻击。首先检查TCP/80端口的服务，发现新闻搜索的功能是由端口8080提供的，输入http:// 202.103.*.168:8080/之后，得到了一个系统管理登录页面，简单地测试一下，输入test/test作为用户名/口令，似乎认证成功，但实际上并不能进入下一个页面。彷徨之间，我进行了JSP大小写的测试，因为JSP对大小写是敏感的，Tomcat只会将小写的JSP后缀的文件当作是正常的JSP文件来执行，如果大写了就会引起Tomcat将Index.jsp当作是一个可以下载的文件让客户下载。经过测试发现这个方法无效，可能管理员已经在服务器软件的网站上下载了最新的补丁。

找了一个朋友的JSP空间，仔细研究后发现大部分的JSP应用程序在当前目录下都会有一个WEB-INF目录，这个目录通常存放的是JavaBeans编译后的Class 文件，如果不给这个目录设置正常的权限，所有的Class就会曝光。而采用JAD软件对下载的Class文件反编译后，原始的Java文件甚至变量名都不会改变。如果网页制作者开始把数据库的用户名密码都写在了Java代码中，反编译后，说不定还能看到数据库的重要信息。那么，怎么得到这些文件呢?思考中……

Tomcat版本的缺省/admin目录是很容易访问的，不知道这个粗心大意的管理员是否做了相关处理。输入：http://202.103.*.168/admin/，管理员目录赫然在列。之后，我看到了“Tomcat WEB Server Administration Tool”的页面，默认情况下，他们的“User Name”应该是admin，“Password”应该是空，输入用户和密码后，并点击“Login”按钮，不能进入，陆续使用了几个比较常见的密码，也无济于事。

默认情况下，Tomcat打开了目录浏览功能，在对多个Resin或Tomcat服务器站点的测试中，我都碰到过管理员忽视这个问题的情况。

小提示：一般可以这么解释：当要求的资源直接映射到服务器上的一个目录时，由于在目录中缺少缺省的Index.jsp等文件，Tomcat将不返回找不到资源的404错误，而是返回HTML格式的目录列表。

我打开刚才用X-Scan扫描后生成的报表文件，找到“安全漏洞及解决方案”栏目，看到了几个可能会有CGI漏洞的目录。在地址栏输入其中之一，感觉是个比较重要的目录。

一些很典型的JSP文件、JS文件都列出来了!大喜之下，随便选择一个文件，点击右键，然后选择“用FlashGet下载全部链接”选项，于是，这个目录下的所有文件都被我下载到了本地，而其中尤其具有价值的是一个名字为Dbconn.js的文件!看来程序设计者是为了方便省事，把一些数据库连接的密码和连接地址都写在里面了(这是很多开发者可能会忽略的问题)。不过，我现在最关心的还是Tomcat的管理员密码!简单破解后，发现Tomcat系统中的admin用户使用了非常简单的口令：WEB123456，可能是管理员对进行口令统一的缘故。尽管外部用户无法直接访问Telnet服务，不过，在我看来，这些口令还是应当被重设为更为强壮的字符串。先不管别人怎么做了，利用这个漏洞，有了这个密码，我就要好好的研究研究Tomcat的管理界面了!

重新登陆Tomcat的管理界面，点击“Context (Admin)”这个链接，列出了WEB目录下的一些文件和目录的名称，我现在就可以对Tomcat的Context进行管理，例如查看、增加、删除Context。这个Context有点类似虚拟目录，于是我可以增加一个Context, 例如“/images”，这是为了避免管理员看出破绽，将它的文档根目录设置为“/”，这样当我访问http://202.103.*.168/admin/时看到的就是系统根目录的列表了!

既然是JSP文件，我就要考虑尝试一下XSS注入、HTML注入、SQL 注入以及命令注入的攻击钢丝了。许多网站都在防范XSS 攻击上存在缺陷，当然，实际的攻击可能并不像前面的示例那么简单，但是，只要在WEB应用程序中有将未过滤的输入回送给用户的缺陷，就可以设计出XSS 攻击的方式。含有容易受到XSS 攻击的WEB 应用程序可以用各种程序语言编写(包括Java)，并在任何操作系统上执行。这是一种一般且普遍的网页浏览器脚本问题，而此服务器端的问题主要来自于并未确认及过滤恶意的用户输入。能否在这方面做点文章呢?

回到Tomcat的管理界，无意中发现了一个上传文件的组件，而且这个站点下有个论坛，真乃天助我也!写一个JSP文件弄上去看看，也正好验证一下JSP用户过滤方面的技术。于是，我编写了一个JSP 网页，并将其当做一般的WEB 应用程序，这个JSP网页会接受用户的输入并显示少量的调试信息。通过上传的组件，我将自己精心准备的Input.jsp文件上传到对方的WEB目录里， 打开input.jsp这个页面。

网页上包括两个查询窗体和一个用户名称的输入窗体。前面的两个查询窗体基本是一样的，只不过一个使用HTTP GET，另一个使用HTTP POST。此外，其参数的名称也不同，所以可以一次测试两个窗体，其参数值不会彼此干扰。网页对查询窗体不会做任何输入验证，但是对用户名称的窗体则会。网页上的所有窗体都会自动填入上次传送来的值(如果没有任何上次的值，则会填入Null)。将数据填入窗体，来测试一下网页的漏洞，例子如下：

通过这些方法，我得到了一些论坛的用户信息，当然，这些都是针对JSP做的一些测试，以验证WEB应用程序中的所有输入字段。我的思路是这样的，如果能从第一个浏览器会话中取出URL，包括Jsessionid，然后粘贴到其它浏览器中进行浏览，就能够伪装成其他用户了!

几番轰炸，我制作出了一份精确的字符清单，以测试应用程序需要当成用户输入而接受的所有字符。如果应用程序会接受许多特殊字符，估计会有一些漏洞，这样我就可以获得更大的访问权限。有了用户信息，却没有密码，怎么办?在登陆时，我发现了一个8888端口，这会是个什么服务呢?

疲惫中，我将注意力转移到测试时发现的8888端口，难道是个聊天室?看来Linux下的平台确实有很多让人称奇的地方，打开地址后，我发现这个端口运行的是Apache+PHP，也就是说这台主机还可以编译运行PHP!不过从经验分析来看，管理员在JSP主机上同时安装PHP的主要目的可能是为了管理MySQL数据库，因此，这个端口很可能有phpMyadmin。再联想到前面发现的漏洞，这个端口上会不会有个数据库管理目录呢?测试后果然不出所料，在输入这个目录之后进入了一个phpMyadmin的管理界面，可以对MySQL数据库进行任意操作。它支持从本地操作系统读入或者写入数据。更不可理解的是，管理员居然用ROOT账户写在了数据库连接里面，想不控制这个数据库都不行了。

打开其中的一个数据库，在“SQL”中输入“SELECT * FROM `adminuser`”，Adminuser表中的数据全部显示出来了。和我前面用JSP探测的用户类型大致一致。至于他们的表和数据的删改权限，现在则完全在我的掌握之中了。

本来还想测试一下Tomcat下的Invoker Servlet的，不过朋友催地实在太急，也就暂时终止了。不过，从测试过程来看，这些问题都是Tomcat下的一些基本配置问题，很多问题来自安装目录下的Conf子目录下的WEB.xml文件。现在要做的，就是给他们提交一份详细的防范措施了。

文章标签： 入侵 Tomcat 黑客 漏洞

个人分类： 计算机网络