基于 Linux 的 Web 服务器安全审计系统研究与实现

何建波[1] 刘国乐[1] 孟 正[2] 文伟平[2] / [1]国家保密科技测评中心 [2]北京大学软件与微电子学院

【摘 要】当前，Linux 下的 Web 服务器面临着严重的安全威胁，而 Linux 自身的审计机制存在着一些缺陷。本文首先介绍了 Linux 系统的日志信息和 Apache 的日志信息，然后提出了一种基于 Linux 的 Web 服务器安全审计方案，该方案将内核级的安全审计与应用程序级的安全审计有机结合起来，最后在仿真平台上对其予以实现。

【关键词】Linux Web 服务器 安全审计 Apache 日志

1 引言

Linux是一种常用的开源操作系统，被广泛地应用于嵌入式、服务器等领域[1]。由于Linux系统具有稳定性较好、安全性较高，网络功能强大等特点，Linux系统常被作为Web服务器的部署平台。

随着互联网技术的飞速发展，社交网络、电子商务、电子邮件等网络应用已经成为人们生活密不可分的一部分。与此同时，针对Web网站的攻击事件呈逐年上升趋势，Web网站安全问题也越来越引起人们的重视。近年来，针对Web网站的网络攻击技术不断更新，手段层出不穷，尤其是SQL注入、跨站攻击等针对Web应用的新型攻击手段的出现，给Web服务器带来了严重的安全威胁。这些攻击，轻则导致Web应用服务受影响，重则导致用户数据泄露，甚至Web服务器被完全控制。社会上，一些论坛、社交网站被攻陷，造成大量用户数据泄露的消息不时见诸报端，而某些单位的Web邮件系统被攻击，造成大量邮件被窃取的事件也时有发生。

针对这些威胁，人们主要是通过及时修补Web应用自身漏洞，部署Web应用防护系统（比如Web防火墙）等措施提高安全防护，加强入侵手段的检测，较少考虑通过加强和改进Linux平台安全审计功能来对入侵事件进行检测和预警。本文对Linux中最常见的Web服务器应用软件Apache所面临的安全威胁进行分析，以此为基础，提出了一种基于Linux的Web服务器安全审计方案，并在仿真平台下对其予以实现。

2 Linux 下 Web 服务器面临的安全威胁

2.1 Linux系统的安全问题

在安全性方面，Linux内核在一定程度上对POSIX标准草案中的capabilities安全机制提供了支持，它拥有传统的Unix自主访问控制机制（即root用户，用户ID，模式位安全），无法满足当前日益紧迫的信息安全问题的需要，对Linux系统的发展也造成了影响。另外，从实现的角度看，系统缺乏对访问控制的保护机制，具体表现为以下几个方面[2]：

（1）未对可加载模块的加载实施监控

模块使得Linux内核更加高效和灵活，将其加入内核后，它就会成为内核的一部分。如果超级用户（root）对模块随意加载，就有可能导致恶意代码被写成模块并加入到内核里，从而重定向系统调用，使系统处于不安全状态。

（2）系统管理未受保护

当入侵者获得root权限后，可以开展一些系统管理工作，如模块的装载/卸载，路由的设置，防火墙规则的设置等。

（3）进程未受保护

系统上运行的进程是为某些功能服务的，例如HTTPD用来满足远程客户端对于Web服务器的Web需求。对于Web服务器系统来说，应保护其进程不被非法终止。另外init进程是系统启动后全部运行进程的父进程，然而，当入侵者获得root权限以后，系统却无法对这些重要进程实施保护。

（4）文件系统未受到充分保护

如果发生入侵，系统中很多重要的文件将会被肆意修改，使得攻击者在系统中为所欲为。

2.2 Web服务器的安全威胁

Web服务器主要面临以下几个安全威胁：

（1）Apache版本信息泄露

通常Apache在安装时会显示版本号信息和模块信息等，它们可能被黑客利用，黑客也能够从中得知一些服务器的配置信息[3]。

（2）Apache以nobody的用户账号运行

有些Apache会使得服务器以nobody的用户运行，如果FTP服务器和Apache服务器都是以nobody的用户账号在同时运行，那么利用Apache发起的攻击就很有可能同时对FTP服务器产生攻击。

（3）Apache目录浏览

某些Apache服务器由于设置不当，允许用户在访问某个网站时，在后面增加相应的目录，从而浏览到Web网站目录，特别是可能的文件上传路径，有可能威胁到网站的安全性。

（4）Apache拒绝服务攻击

利用HTTP协议进行DOS攻击、UDP flood、ICMP flood、SYN flood等，大量伪造的连接请求向网络服务的端口发起攻击，使得服务器资源耗尽、系统停止响应，甚至系统瘫痪。

（5）Apache漏洞

利用Apache漏洞实施攻击，获取系统控制权。比如Apache1.3有一个远程缓冲区溢出漏洞，利用该漏洞可获得HTTPD服务运行者的权限[4]。

（6）攻击者获得ROOT权限

该安全缺陷产生的主要原因是Apache服务器一般是以root权限运行的，攻击者会通过Apache服务器获得root权限，进而完成对整个Linux系统的控制。

3 Linux 的审计机制

审计是安全操作系统中一项重要的安全机制。操作系统的安全审计是通过记录、查看和审核系统中与安全相关的活动来实现的，其主要目的是阻止非法用户入侵计算机系统。

Linux缺乏真正的安全审计机制，它当前的审计机制是通过以下三个模块实现的：应用程序日志、系统日志和记账日志[5]。Linux的审计机制记录了一些日志信息，如进程统计日志信息、用户登录退出信息以及内核与系统程序信息等。Linux的审计机制绝大部分是在应用程序级实现的，它利用独立于操作系统的程序syslogd对用户登录和相关操作的信息予以记录，将用户操作所产生的提示和

警告等信息按照统一的格式进行记录，其功能结构如图1所示。Linux的Shell命令是在.bash_history文件中被记录的，该文件位于用户工作目录下，但没有对Shell命令的时间予以记录。这种方式缺少对全局所有用户审计信息的记录，不便于对其进行管理。由于Linux系统是由应用程序进行安全审计的，因此，当入侵者获得相应的权限后，他可能会绕过syslogd，使系统不能记录入侵操作，甚至将所有的审计信息予以清除。

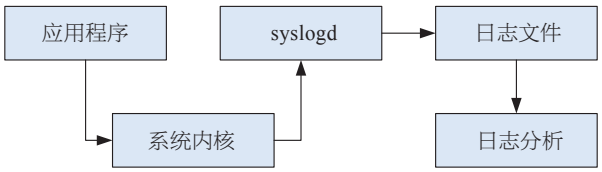

图1　Linux系统审计逻辑

总的来说，Linux自身的审计机制主要存在以下几方面的不足[6, 7]：

（1）审计记录没有针对性，不能很好地满足特定环境下的审计要求；

（2）获取的审计信息量较少，审计日志存储分散，审计记录内容不够详细，没有统一的记录格式，不利于对审计信息进行综合分析和管理；

（3）审计程序有可能会被黑客终止或绕过，从而使得审计记录不产生；

（4）产生的审计信息有可能会被黑客恶意篡改或删除，无法对审计信息的完整性提供保证；

（5）Linux自身的审计机制只提供一些必要的日志信息，包括进程统计日志信息、用户登录和退出信息以及内核与系统程序信息等，对于一款成熟的操作系统来说，仅提供这些日志信息是非常不完善的。

4 Linux下Web服务器日志信息

4.1 Linux系统的日志信息

Linux安全架构中的一项重要内容就是审计日志，它是表明攻击发生的唯一证据。Linux主要提供了主机、网络和用户级的日志信息[8]，可以对以下五个方面的内容进行记录：

（1）全部的内核和系统信息；

（2）网络连接的相关信息，包括源IP地址等；

（3）攻击者使用的操作系统和用户名等信息；

（4）远程用户访问的文件；

（5）用户使用的所有命令。

Linux系统的日志信息主要有以下两方面的作用：

（1）网络管理员和系统管理员可以通过日志对程序的行为信息或可疑用户进行分析，发现网络或系统中存在的安全漏洞，预防黑客的攻击，提升网络和系统的安全性。

（2）当系统和应用程序在运行过程中出现问题时，用户能够通过日志对问题的根源进行分析，从而解决问题、恢复系统。

Linux系统提供了一些必要的日志信息给系统管理员，包括进程统计日志信息、用户登录和退出信息以及内核与系统程序信息等。Linux系统将日志信息统一存放到某个特定的目录下，目录名为/var/log。

Linux通过一个名为syslogd的后台进程，记录内核或系统程序产生的警告、错误等信息。它首先按照消息的重要性和消息的写入者对消息进行分类，然后将其写入不同的文件中。管理员能够在配置文件/etc/syslog.conf中定制syslogd的行为。

在用户登录Linux系统的同时，系统会向/var/run/utmp和/var/log/wtmp文件中写入登录信息。同样，Linux系统还能够对进程的一些行为信息进行统计，如进程ID、IO传输信息、进程运行时间等，这些信息会被写入/var/log/pacct文件中。通过acct系统调用或accton命令，管理员也可以对进程统计信息进行重定向，将其写入其他文件中。

4.2 Apache日志信息

默认的情况下，Apache在服务器运行时就会生成两个日志文件，分别是error_log和access_log，它们存储在目录/usr/local/apache2/logs中[9]。

error_log表示错误日志，Apache httpd将在其中存放处理请求中出现的错误及诊断信息。错误日志的格式非常灵活，一个典型的例子是：

[Wed Oct 11 14:32:52 2000] [error] [client 127.0.0.1] client denied by server configuration: /export/home/live/ap/htdocs/test

其中，第一项表示错误发生的时间，第二项表示错误的严重性，第三项表示出现错误的IP地址，其后是信息本身。在该例中，客户的访问被服务器拒绝。服务器在对被访问文件进行记录时，没有使用Web路径，而是使用了文件系统路径。

访问日志access_log则记录了对Web服务器的全部访问，一个典型的例子是：

216.35.116.91 - - [19/Aug/2000:14:47:37 -0400] "GET / HTTP/1.0" 200 654

上述日志记录由7项组成，第一项信息表示远程主机的地址；第二项是空白，用一个“-”占位符表示，用于对浏览者的标识进行记录；第三项也是空白，用于对浏览者在进行身份认证时提供的名字进行记录；第四项是请求的时间；第五项信息是比较有用的，它表示服务器收到请求的类型；第六项是状态代码，它告诉我们请求是否成功，或者遇到了什么样的错误；日志记录的第七项表示向客户端发送的总字节数。

5 基于 Linux 的 Web 服务器安全审计工具总体设计

5.1 设计目标

本文对基于Linux的Web服务器安全审计工具进行设计和实现，其目的在于监控特定环境下的Linux主机和Web服务器，从审计数据中获得系统管理的相关信息，对Linux安全机制的实施情况进行详细记录，并以此为基础向系统中出现的问题做出反应。

基于Linux的Web服务器安全审计工具应具有以下两个特征：

（1）可以获得足够的审计信息，获取信息的过程不能被绕过且不会被黑客破坏；

（2）保证审计日志的完整性和可用性，防止审计日志被黑客恶意篡改。

5.2 总体架构

本文研究和实现的基于Linux的Web服务器安全审计工具是在应用程序级和内核级两个层次实现的，总体框架如图2所示。

在图2中，内核级安全审计子系统主要记录用户的各种操作，包括登录、通信、文件访问以及命令等信息，应用程序级安全审计子系统主要针对应用程序的特征以及系统的安全配置信息进行记录。将内核级安全审计子系统和应用程序级安全审计子系统记录的日志信息整合起来，会形成完整的审计日志，接下来对审计日志做进一步的处理，保证日志信息的完整性和可用性，最终将审计日志显示出来。内核级和应用程序级的两个安全审计子系统相辅相成，共同构成了基于Linux的Web服务器安全审计框架。

5.3 功能结构

基于Linux的Web服务器安全审计工具功能结

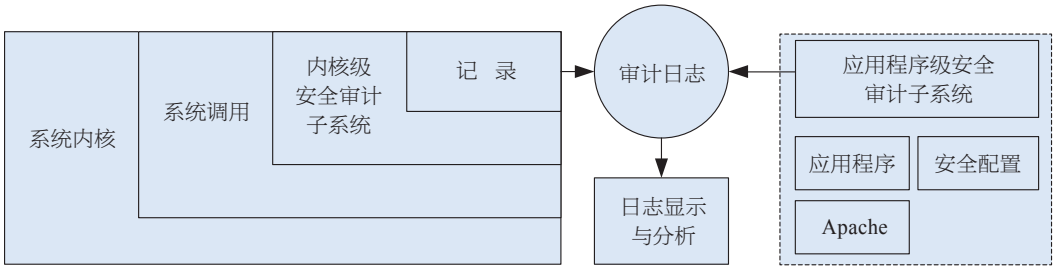

图2 工具总体架构

构如图3所示。

在图3中，内核级安全审计主要针对用户操作和用户权限等进行检测。而应用程序级安全审计主要包括对程序包的检测、对配置错误的检测、对安全问题的检测、对系统信息的检测以及对Apache日志的统计分析，应用程序级安全审计具体可以细化为37个审计点，分别是：系统信息，系统工具，启动项，内核，内存和进程，用户、组和认证，Shell，文件系统，存储，NFS，软件，网络，打印机服务，E-mail，防火墙，Web服务器，SSH，SNMP，数据库，轻量目录访问，PHP，Squid代理服务器，登录和文件，网络守护inetd，Banner，任务调度，账户，时间与同步，加密，虚拟化，安全框架，恶意扫描，文件权限，Home目录，内核固

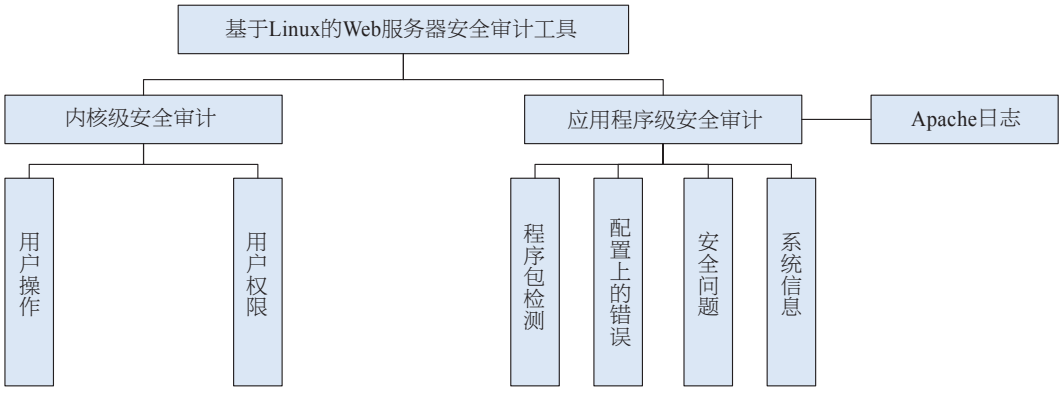

图3　工具功能结构

化以及编译固化。

6 基于 Linux 的 Web 服务器安全审计工具实现

6.1 内核级安全审计的实现

（1）设计思想

在Linux内核中，那些被其他操作所调用的最基本函数称为系统调用（System Call），所有进程与内核打交道的最终方式是系统调用[10]。系统调用是通过查找系统调用表（sys_call_table）实现的，从系统调用表可以找到内核函数的地址，然后对其进行调用。当函数返回时，需要做一些系统检查，接着返回用户进程。因此，如果想对某个系统调用的运作方式进行变更，只需要添加一些自己的代码，然后调用原函数，接着对系统调用表中的指针值进行更改，使其指向我们的函数。

（2）系统调用的劫持

用户软件调用中断int 0x80可以对Linux系统中的系统调用进行激发，执行int 0x80后，内核将获得对CPU的控制权，然后将程序交给system_call进行处理[11]。System_call所做的工作是：对寄存器进行保存，判断系统调用的编号是否合法，然后参照系统调用编号和系统调用表，对系统调用处理程序进行查找，最终执行系统调用。

目录的修改和创建、文件的读写一般都是通过系统调用完成的，因此，对系统调用进行控制，可以限制用户对文件的操作，从而实现审计功能。

（3）处理流程

内核级安全审计子系统的处理流程[12]描述如下：

a. 查找sys_call_table；

b. 将需要控制的系统调用的入口替换；

c. 当相应的系统调用出现时，先对审计模块的功能予以处理；

d. 当操作请求与预先设定的规则不匹配时，记录该操作；

e. 当操作请求合法时，转向原有的系统调用入

口，对正常操作予以执行；

f. 继续等待新的操作请求。

（4）修改查询文件信息的系统调用

在Linux中，对文件信息进行查询的系统调用是sys_getdents。如果对该系统调用进行修改，将查询结果中与文件相关的信息去掉，那么所有使用该系统调用的程序将不能发现该文件，从而达到隐藏的效果。系统调用sys_getdents的原型是：int sys_getdents(unsigned int fd, struct dirent* dirp, unsigned int count)。

其中，fd是一个文件描述符，它指向目录文件，sys_getdents函数根据fd指向的目录文件对相应的dirent结构进行读取，并将其放入到dirp中。第三个参数count表示dirp返回的数据量。该函数的返回值表示dirp中填充的字节数。修改后的系统调用sys_getdents执行流程如图4所示。

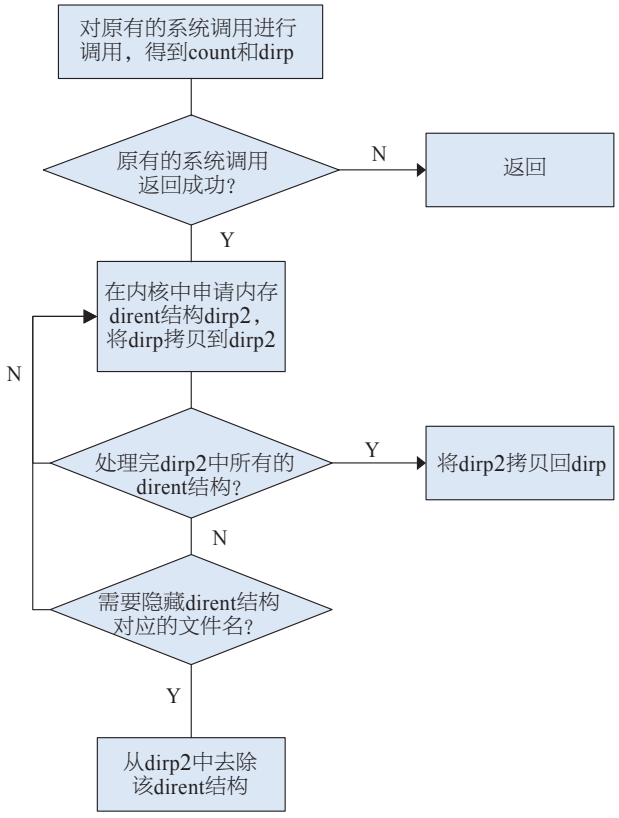

图4　修改后的系统调用sys_getdents执行流程

图4中的sys_getdents函数首先对原有的系统调用进行调用，然后在返回的结果dirent结构中去掉与文件相关的信息，使得应用程序从该系统调用返回后，文件是不可见的。

6.2 应用程序级安全审计的实现

（1）功能检测

应用程序级安全审计子系统的功能检测模块是通过Shell语言编写的，开发平台是Ubuntu 12.04。

a.系统类型检测

调用uname获取系统类型名，如果是Linux则调用uname –r命令获取内核版本信息，然后使用命令grep "^DISTRIB_ID=" /etc/lsb-release | cut -d '=' –f2获取系统名字，如图5所示。

```
-------------------------------------------------
Program version:          v 1.0
Operating system:         Linux
Operating system name:    Ubuntu
Operating system version: 12.04
Kernel version:           3.2.0-20-generic-pae
Hardware platform:        i686
Hostname:                 ubuntu
Profile:                  ./default.prf
Report file:              /usr/test_LMSCT/report/LMSCT_report.txt
Log file:                 /usr/test_LMSCT/report/LMSCT_log.txt
Report version:           1.0
-------------------------------------------------
```

图5　检测系统版本

b.检测系统的二进制可执行文件

在/bin、/sbin和/usr/bin等路径下对常见程序进行查找，确定常见程序是否存在，如图6所示。

```
[+] System Tools Checking
-------------------------------------
  - Scanning available tools...
  - Checking system binaries...
    - Checking /bin...                          [ FOUND ]
    - Checking /sbin...                         [ FOUND ]
    - Checking /usr/bin...                      [ FOUND ]
    - Checking /usr/sbin...                     [ FOUND ]
    - Checking /usr/local/bin...                [ FOUND ]
    - Checking /usr/local/sbin...               [ FOUND ]
    - Checking /usr/local/libexec...            [ NOT FOUND ]
    - Checking /usr/libexec...                  [ NOT FOUND ]
    - Checking /usr/sfw/bin...                  [ NOT FOUND ]
    - Checking /usr/sfw/sbin...                 [ NOT FOUND ]
    - Checking /usr/sfw/libexec...              [ NOT FOUND ]
    - Checking /opt/sfw/bin...                  [ NOT FOUND ]
    - Checking /opt/sfw/sbin...                 [ NOT FOUND ]
    - Checking /opt/sfw/libexec...              [ NOT FOUND ]
    - Checking /usr/xpg4/bin...                 [ NOT FOUND ]
    - Checking /usr/css/bin...                  [ NOT FOUND ]
    - Checking /usr/ucb...                      [ NOT FOUND ]
```

图6　检测系统的二进制可执行文件

c.检测审计工具的开启状态

脚本tests_accounting 检测不同系统下auditd审计工具的开启状态。

d.检测用户信息和认证

脚本tests_authentication检测用户信息和认证，并调用chkgrp检查超级用户的个数，如图7所示。

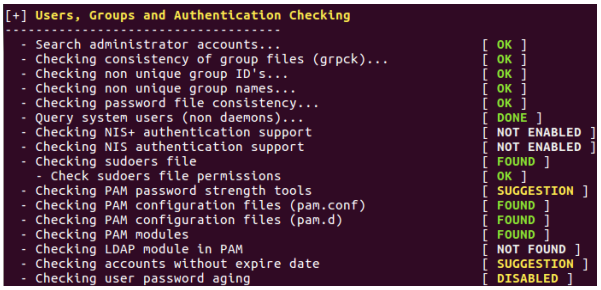

图7 检测用户信息和认证

e.检索系统的全部用户和用户id

从passwd中检索出系统的所有用户和用户id的命令： awk -F: '($3 > 500) && ($3 != 65534) || ($3 == 0) { print $1","$3 }' /etc/passwd。

f.检测是否存在重复的用户组

从/etc/group中检测是否存在重复的用户组的命令： cat /etc/group | grep -v '^#' | grep -v '^$' | awk -F: '{ print $3 }' | sort | uniq –d。

g.检测密码文件的数据完整性

调用pwck检测密码文件数据完整性的命令：/usr/sbin/pwck -q –r

h.检测含有系统banner的文件是否存在

脚本tests_banner检测含有系统banner的文件是否存在，如/etc/issue中保存了系统的版本信息。

i.检测系统启动项

脚本tests_boot_services 检测系统启动时运行的脚本，首先是grub boot loader文件/boot/grub/grub.conf 和grub.cfg。对于Ubuntu来说，可以检测/etc/rc2.d文件，检测文件/etc/init.d、/etc/rc、/etc/rc.local以及/etc/rc.d/rc.sysinit，如图8所示。

j.检测过期的SSL证书

脚本tests_crypto检测过期的SSL证书。

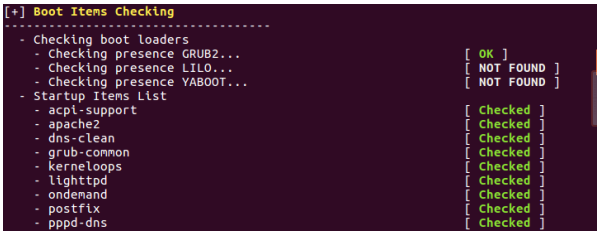

图8 检测系统启动项

k.检测是否有mysql进程存在

脚本tests_database首先检测是否有mysql进程存在，检测mysql 数据库是否设置密码，命令如下：${MYSQLCLIENTBINARY} -u root --password= --silent --batch --execute=""。

l.检测是否安装文件完整性检测工具

脚本test_file_integrity检测是否安装文件完整性检测工具，例如tripwire。

m.检测root/.ssh文件的权限设置

脚本tests_file_permission检测root/.ssh文件的权限设置。

n.检测/tmp目录和/home目录是否挂载在一个挂载点上

脚本tests_filesystem检测/tmp目录和/home目录是否挂载在一个挂载点上，检测文件系统类型是否是ext2，ext3或ext4。

o.检测iptable的开启状态和规则设置

脚本tests_firewall：检测iptable的开启状态和规则设置。

（2）Apache日志分析

系统的Apache日志存储在目录/usr/local/apache2/logs中，以perl作为编程语言，对Apache日志进行统计分析，并以直观的形式予以显示，如图9所示。

（3）审计日志

应用程序级的安全审计日志部分截图如图10所示。

6.3 审计日志的保护

程序在运行过程中生成了审计日志，审计日志

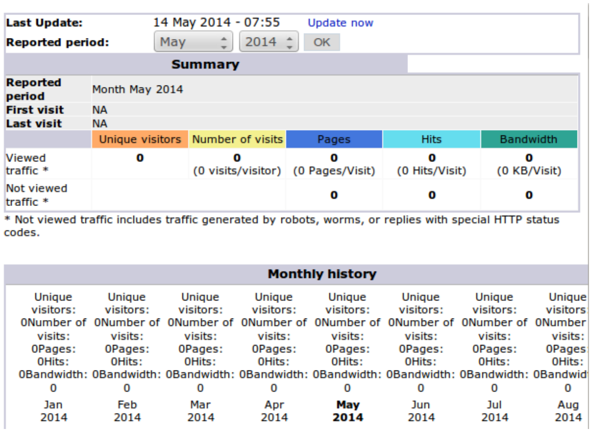

图9 Apache日志分析

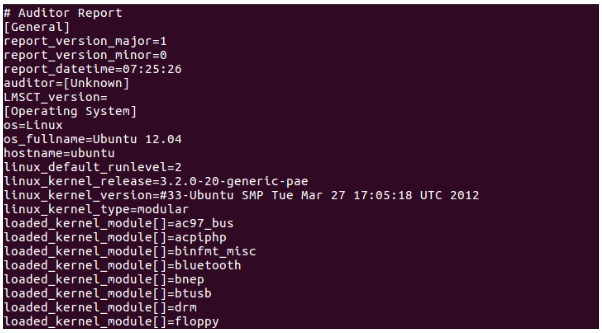

图10 安全审计日志

的安全级是由审计进程指定的。对于内核级的安全审计来说，创建的文件也是内核级的[13]。另外，系统会对审计员进行监督，从而降低风险[14]。

7 结语

本文首先描述了Linux下Web服务器面临的安全威胁，分析了Linux的审计机制，并针对其不足，提出了一种基于Linux的Web服务器安全审计方案，将内核级的安全审计与应用程序级的安全审计有机结合起来，保证了审计信息的完整性和可用性，最后在仿真平台上对其予以实现。END

参考文献：

[1] 刘嘉,訾小超,潘理,李建华.基于Linux操作系统的审计增强[J].计算机工程.2005,31(12):49~51.

[2] 汪立东,方滨兴.Linux Shell安全审计机制的扩展[J].软件学报,2002,13(01):80~84.

[3] Helm P, Liepins G. Statiscal Foundation of Audit Trail Analysis[J]. IEEE Transaction on Software Engineering,1993(19):886~901.

[4] Hoagland J, Wee C, Levitt K. Audit Log Analysis Using the Visual Audit Browser Toolkit[EB/OL]. http://seclab.cs.ucdavis.edu/awb/AuditWork-Bench.html.

[5] Descartes A, Bunce T. Programming the Perl DBI. 101 Morries Street Sebastopol[R]. CA 95427:O'Reilly&Associates, Inc.,2000.

[6] Wetmore B. Paradigms for the Reduction of Audit Trails[D]. Davis:University of California, 1993.

[7] 楼芳,谭安芬,陈志文,赵伟峰. Linux综合监控审计系统的设计与实现[J].计算机安全,2009(09):9~11.

[8] 卢纲,张丽芬.关于Linux审计信息的获取[J].北京理工大学学报,2001,21(01):69~72.

[9] 刘宏哲,鲍泓,须德.Linux安全操作系统的审计机制的研究与实现[J].计算机工程与应用,2005(17):50~52.

[10] Peter Loscocco, Stephen Smalley. Intergrating Flexible Support for Security Policy into the Linux Operation System[R]. NSA Technical Report, 2001,02.

[11] 魏立峰,陈松政.Kylin安全审计系统的设计与实现[J].计算机工程与设计,2010,31(15):3339~3342.

[12] 宋金玉,张镜森,梁小虎.一种查看Linux系统审计日志的图形工具[J].计算机工程,2005,31(10):222~224.

[13] 陈慧,石文昌,梁洪亮.通过扩展LSM框架构建审计支持机制[J].计算机科学,2005,32(03):144~147.

[14] 张小梅,李希.安全操作系统的审计子系统设计[J].兰州理工大学学报,2006,32(06):111~113.