入侵检测系统中文件完整性检查的研究*)

Research on File Integrity Check in Intrusion Detection System

陈 昕 杨寿保

(中国科学技术大学计算机科学技术系 合肥230026)

Abstract File Integrity Check is not only an important intrusion detection methods, but also play an important role in helping detecting when using other methods. This paper introduces a method of designing File Integrity Check System, which will strengthen the intrusion detection system.

Keywords Intrusion detection, File integrity, Status database

1 引言

随着计算机技术特别是网络技术的发展,开放的互连的计算机系统为通信和资源共享提供了便利,但随之而来的是更多的系统安全问题。系统的安全是一个体系,建立在各种安全机制集成的基础上[1],入侵检测系统(Intrusion Detection System,简称 IDS)作为系统安全的一道重要防线,主要是通过多种手段监控系统来发现入侵。文件完整性检查是 IDS 中的关键部分之一,它通过检查系统关键资源的变化情况来为入侵的发现提供依据。然而目前这方面软件的检查方式并不多样,一般以单独使用的工具形式出现,不能形成一个系统,因而检查的效果也受到一定影响。本文首先介绍入侵检测系统和文件完整性检查的基本概念,接着对文件完整性检查的目标和主要内容进行了分析,然后从系统化的角度出发,提出了一个文件完整性检查系统的基本框架和组织方法,最后给出结论。

2 入侵检测技术和入侵检测系统概述

2.1 入侵检测技术

简单地说,入侵就是试图在未经授权的情况下进入他人系统或非法使用系统资源的行为。入侵检测则是用各种可能的技术手段来发现来自系统外部的入侵或系统内部合法用户滥用权限的行为。入侵检测的方法可分为两类:基于滥用(misuse-based)的入侵检测和基于异常情况(anomaly-based)的入侵检测。

基于滥用的检测技术根据入侵的形式或特征建立入侵模式,通过在被检测的数据流中匹配这些模式来检测入侵。这种技术的优点是原理简单,实现算法不复杂,缺点是与某些杀毒技术相似,只能检测到大部分或所有已知的入侵模式,对未知的入侵模式无能为力。

而基于异常的检测技术认为所有的入侵活动都是相异于正常的系统活动的。因此,如果预先建立安全情况下系统的状态档案,描述被保护资源的重要特征,如 CPU 利用率、一段时间内的网络连接数等(可通过统计的方法获得),再将系统运行时的状态与之比较,就可识别出入侵。这种检测技术的核心在于安全情况下系统的各种阈值的定义[2]。

2.2 入侵检测系统

入侵检测系统(IDS)是以入侵检测为核心机制的安全工具的集成,它运行在被保护系统的后台,监测系统资源和网络信息,分析其安全征兆,发现非法进入系统的外部入侵者或滥用系统资源的合法用户,通知系统管理员,并在必要时进行实时处理。IDS 是对防火墙的必要补充。根据检测的数据源,可将 IDS 分为:

·基于网络的 IDS(NIDS) 网络型入侵检测系统(NIDS)分布在网络上或设置在被监测的主机附近,检查网络通信情况以分析是否有异常活动,它能提供网段的整体信息,但由于要检测整个网段的大量信息,较易遭受拒绝服务(DOS)攻击。

·基于主机的 IDS(HIDS) 这种 IDS 运行在被保护的主机上,它使用审计日志,监控主机上的文件系统、安全日志,及其它重要文件,检查是否有可疑的操作。它能给主机提供较高程度的保护,但每台受保护主机都需安装相应的软件,且不能提供网段的整体信息。

另外,目前又出现一种较新的基于内核(Kernel)的 IDS 系统。它将 IDS 作为系统的一个模块来加载运行,这种方式在 Linux 系统下使用较多。采取防止缓存溢出,增加文件系统保护,采用阻塞信号等一些办法来增加入侵者攻克系统的难度。

不同种类的 IDS 系统由于适用范围不同,实现上也有差异,但都具有以下基本功能:

a. 监测并分析用户和系统的活动;

b. 核查系统配置和漏洞;

c. 评估系统关键资源和数据文件的完整性;

d. 识别已知的攻击行为;

e. 统计分析异常行为;

f. 操作系统日志管理,并识别违反安全策略的用户活动。

3 文件完整性检查概述

保证数据文件的完整性作为上述 IDS 的各项基本功能中的一项,同时也为其它功能如识别已知的攻击行为和统计分析异常行为提供了依据。

3.1 文件完整性检查的引入

一旦入侵者成功侵入系统后,为逃避检测和方便下次进入,首要的事就是更换系统文件,代之以后门程序。因此,发现入侵行为很重要的一个方面就是保证数据和系统的完整性。

*)本文受到国家自然科学基金重大研究计划项目(90104030)支持。陈 昕 硕士生,研究方向为计算机体系结构和网络安全。杨寿保 教授,博导,研究方向为计算机网络和信息安全。

一般对文件完整性的保护采用32位 CRC 校验(循环冗余校验),然而,高级入侵者已经可以使用特殊技术骗过 CRC 校验,需要我们采用更多更安全的方法来检测,如加密等[4]。

3.2 文件完整性检查的基本思想

文件完整性检查的根本思想就是:将被入侵的系统的状态和未被入侵的系统的状态相比较。在建立了正确安全的初始系统状态后,检查程序定期醒来,根据初始系统状态检查当前系统状态,发现可疑或非法的变动,触发警报和通知系统管理员。另外,正确的系统状态也被经常更新。由此,要实现一个完全的检查功能,至少需要以下几部分:正确安全的系统状态,预先定义的被检查项目,和一个进行定期检查的程序。单个的工具难以完成这样的功能,需要将各部分整合成一个文件完整性检查系统(FICS),下面我们将详细讨论这个问题。

4 文件完整性检查系统

4.1 文件完整性系统的目标

如前所述,文件完整性系统的目标在于:保证被保护主机的关键资源不被恶意更改。所谓关键资源是指系统中不应也不能被经常更改的关键成分,如:内核、配置文件、可执行程序、库等,以及那些由第三方提供的可能影响到主机运行的资源。由于保证的对象是主机上关键资源,又由于必须预定义一个正常安全的系统状态作为基准,所以,一个文件完整性检查系统,使用了基于主机基于异常的检测技术。

4.2 文件完整性检查的内容

具体对文件检查时,要考察哪些内容呢?这是涉及到检查有效性的核心问题。目前文件完整性检查方面的工具有很多,如 Tripwire,AIDE,Site Watcher,Sentinel,L5,Nannie 等[5,6],虽然它们有应用平台之分(UNIX 类/WINDOWS 类),也有开放源码与商业软件之分;但是它们在检查的方式方法上却有共通之处。在借鉴了这些思想的基础上,我们认为检查文件的完整性主要考虑以下内容:

文件的摘要　用密码学的方法来检验文件的完整性是大多数软件的共同考虑。在 Unix 系统中,仅用 sum 和 cksum 这两个命令来检验完整性是远远不够的,因为它们是用来检验偶然的修改,不足以阻止入侵者恶意产生特定的校验和行为。常用 MD5,SHA1等算法对文件产生信息摘要(message digest),由于这些算法中使用的单向散列函数的单向性,入侵者很难逆推,也极难找到两个随机信息使其产生相同的摘要,从而使产生的摘要具有高度安全性。特别是 MD5算法,它本身的直接安全性、简单性、紧凑性和速度都不错,适合于安全快速的产生摘要。

文件权限　文件的权限是系统安全的关键,它决定了文件的状态属性和访问控制。如果文件被赋予不正确的权限,如 shadow 文件可被任何用户读写,将带来严重的安全隐患。因此,记录关键文件的正确权限并定期检查,可以防范一些入侵。

文件权限中的 SUID 位(设置用户 ID)和 SGID 位(设置分组 ID)值得特别注意。设置 SUID root 可以让用户以 root 身份运行程序,执行以原有身份不能进行的操作。SGID 文件也是如此。这在给用户带来方便的同时,也引入了极大的安全问题:若入侵者创建并隐藏一个 SUID root 的 shell 拷贝,再调用该后门,就能取得 root 权限。使用 find 命令列出所有 suid 文件,或许能发现可疑的 SUID root 程序,但 find 命令也可能被篡改以逃避检测。所以,根本的解决方法是记录和检查系统中的所有 suid 程序,密切注意其改变情况[7]。

文件的其它属性　除权限外,对文件其它属性的逻辑判断也可用来检查文件的非法更改。例如,文件的 atime,ctime,mtime 属性之间,有确定的逻辑关系。ctime 的更改不一定影响 mtime,但文件的 mtime 的更改必使得 ctime 随之改变。如果一个文件的 mtime 晚于 ctime,那么该文件一定存在问题。又如,正常的文件总是有属主的,无属主的文件即使不是非法文件,也不是系统操作的正确结果,可以用 find/-nouser-o-nogroup-print 发现。此外,隐藏非法文件也是入侵者经常采用的方法,被隐藏的文件应特别注意。

4.3 文件完整性检查系统的组织

为保证完整性检查的效率,与保证文件完整性检查的有效性不同,需要使用一定的方式方法组织。作为入侵检测系统的一个子系统,文件完整性检查系统也由多个部分整合而成。我们认为,这个系统的构架和组织,有下面一些原则性的策略:

·使用状态数据库来保存文件系统的正确状态

完整性检查系统在运行之初,先要建立被保护系统安全的初始状态档案。这里的初始状态是指被保护系统已安装配置完毕,但尚未与网络连接或未启动网络服务时的状态。此时系统数据来源比较单一,易于保证其安全性。这时的初始状态就是系统正确的安全状态。但即使没有入侵发生,对系统的正常操作(正常的安装删除程序,增减用户,其他必要的管理)也使系统状态动态变化,这就要求先前保存的安全系统正确状态同步更新。为此,考虑使用数据库保存系统文件的正确状态。采用数据库可以使记录的保存结构化,迅速索引和定位,方便检查和更新。数据结构的设计应包括尽可能多文件属性,如文件路径,文件名,文件大小,文件权限,文件 inode 号,文件的连接数,文件属主(OWNER),文件属组(GROUP),文件 last access time,文件 status change time,文件 modified time,MD5值 SHA1值等。合理的数据结构也使各种检查易于编程实现。检查时根据路径和文件名可唯一定位到被检查的文件,更新时无需全部重写已有记录,只需有针对性的更新部分系统状态即可。

·选择需要进行完整性检查的内容　为数据库中数据来源进行选择的过程,也就是确定文件完整性检查的内容的过程。一方面,考察系统及其文件的安全性要尽量全面;另一方面,为保证检查效率和及时有效性,应使被检查的项目数量上最简。

根据4.2所述的内容检查策略,具体来说需保存入库的文件有:文件系统中关键的 binary 文件、库文件、系统头文件(.h)、其它配置文件、以及须保证安全的文件。需保存的文件属性包括:权限,i-node 号,用户,组,文件大小,atime,ctime,mtime,文件大小的增量和文件的连接数等。对某些特殊文件要进行针对性的检查,如:对/etc/passwd 文件,查看是否有未授权的,无密码的,uid 为0的可疑帐户;rhosts 文件应当是被禁用的,可以 find /home-ame ,rhosts-print 检查;对/etc/inetd.conf,查看是否有可执行 shell 程序的 entry,是否有本应关掉的服务被打开;inetd.conf 里提到的程序,都是完整性检查的对象;检查由 cron 和 at 运行的文件,它们很可能被留了后门,而且,那些直接或间接调用 cron 或 at 的,要检查其权限是否是 world-writable;查找所有 world-writable 的文件看是否有本不该对所有用户可写的;等等。对一些重要文件较多的目录,可以整个目录作为单位检查,比如 bin/sh,/bin/

time，因为那里常被放入非法的 suid，sgid 文件。

此外，特别注意某些文件不能作为需检查的对象，如某些日志文件，mail spools，用户自己的目录，临时目录等。它们都是临时的或迅速增长的文件，检查它们没有意义，只能增加系统的负担。

·使用特定的配置文件，针对不同内容采用不同检查方式

在考虑检查的具体策略时我们发现，一方面，许多有共同属性的文件其检查方式基本相同，而另一方面，对属性不同的文件所采取的检查方式会有较大差别。有的文件只需一般性检查即可，而有的文件却要附加特殊检查；有的文件发生了改变，仅仅说明有被入侵的可能，有的文件被改动却几乎可断定有入侵发生；另外对目录的处理也与文件会稍有不同。为此，考虑采用配置文件存储文件检查方式。首先，使用配置文件易于管理。它以条目方式记录规则，规则的内容是被检查的文件及相应的检查方法。对不同文件的不同检查只需制定不同的规则就可实现。对于有相同属性的文件使用一条通用规则，避免了大量规则的重复定义。规则的定义是灵活的，甚至可以自定义规则以适应具体系统状况。其次，配置文件的使用使得检查方便高效。检查程序识别配置文件中的规则，作出相应的检查行为，具体可用 Perl 语言来实现，这种语言在模式匹配方面有很多优点，快捷高效。第三，配置文件不仅存储检查方式的规则，也可存储文件完整性检查系统的重要配置，如状态数据库的参数等。一个典型的配置文件如下所示：

```
# known database
known=/root/databases/usr_known.cdb
# current database
current=/root/databases/usr_current.cdb
# Here are all the things we can check - these are the default rules
#p:permissions
#i:      inode no.
#n:      number of links
#u:      user
#g:      group
#s:      size
#m:      mtime
#a:      atime
#c:      ctime
#l:      links
#md5:    md5 checksum
#sha1:   sha1 checksum
#S:      check for growing size
#R:      p+i+n+u+g+s+m+c+md5
#L:      p+i+n+u+g
#E:      Empty group
#>:      Growing logfile p+u+g+i+n+S
#
# You can also create custom rules from above definitions- something goes
# like this:
#MyRule: p+i+n+u+g+s+b+m+c+md5+sha1
#
# Next decide what directories or files you want to put into the database
#
/etc p+i+u+g #check only permissions, inode, user and group for /etc.
/bin MyRule # apply the customed rule to the files in /bin
/sbin MyRule # apply the same custom rule to the files in /sbin
/var MyRule
!/var/log/.* # ignore spool dirs because they change too often
!/var/adm/utmp$ # ignore the log dir because it changes too often
!/var/spool/.* # ignore the file /var/adm/utmp
```

在这份配置文件中，可以配置状态数据库的逻辑路径，可以定义文件属性的标识以及几种固定的检查方式，可以根据基本的检查方式自定义组合甚至添加检查方式，还可确定目录的检查规则；总之，配置文件的使用可以非常灵活。

·文件完整性检查系统的整体运行方式　整个系统运行在被保护主机的后台，周期性地启动运行（周期的确定取决于系统的具体情况，可以在 crontab 文件中控制）。运行时首先读取配置文件，对规则中的每一项，根据指定方式进行检查，再将当前状态与库状态进行比较，并将结果录入日志。对于目录，检查目录下所有的文件，若有子目录，递归处理。若没有变动，在日志中记录正常，若有异动，在记录变动情况的同时，根据对系统可能的危害程度，发出报警，通常是发消息或 mail 给系统管理员。报警时也可一并给出事件紧急程度，但这需要事先对异常情况进行分类，并制定相应的规则。分类的标准可以是：

第一类：文件的权限发生变化及文件的属主发生变化（最严重）

第二类：文件大小变化及摘要值的变化（较严重）

第三类：atime、ctime、mtime 的变化等

报警和日志记录互为补充，缺一不可。报警可以及时提出问题，但在无法报警的严重情况下，如网络瘫痪，通信中断时，日志记录就为以后的分析处理提供了关键的信息。

此外，考虑到某些系统更新较快，相应的状态数据库更新频率也较高，简单的更新方法未必可行。我们认为要采取一种多库的机制，即有一个主库和多个更新库，主库在系统运行初期建立，系统每次检查文件系统后，所有的异常日志都进入更新库，并且提交报警信息，管理员可以根据实际情况自动或手动地更新主库中的信息。主库和更新库中的数据结构应该基本相同。

最后，也是最基本的一点，整个文件完整性检查系统，包括所有的库，日志，以及检查程序本身，都必须放置在安全的地方，比如可移动的存储介质，最好是可以启动系统的引导介质，能够恢复系统；或者从可保证高度安全的机器上通过只读 NFS 输出。

结束语　文件完整性检查系统是入侵检测系统中重要而基础的一部分，设计完整性检查的策略时需要对系统进行深入的分析；对整个完整性检查系统的设计更是要考虑到各个方面，运用各种技术来组织和架构。本文按照上述原则设计，提出一种文件完整性检查系统的框架组织方法，既增强了文件完整性检查的功能，同时也易于编程实现。

参 考 文 献

1 胡昌振，李贵涛. 面向21世纪网络安全与防护，“九五”国家重点电子出版物规划项目，北京希望电子出版社，1999

2 Sundaram A. An introduction to Intrusion Detection, ACM Crossroads, 1996

3 Marcus J. Ranum, Coverage in Intrusion Detection System. http://www.nfr.com/

4 http://www.sans.org/newlook/resources/IDFAQ/integrity_checker.htm

5 http://www.networkintrusion.co.uk/integrity.htm

6 http://www.securityportal.com/lskb/10000000/kben10000026.html

7 Bandel D A. Linux Security Toolkit, IDG Books Worldwide, Inc., 2000