一个安全操作系统 SLinux 隐蔽通道标识与处理

刘文清[1] ,韩乃平[2] ,陈 [illegible][1]
(1. 解放军信息工程大学电子技术学院 ,河南郑州 450004 ;2. 上海中标软件有限公司 ,上海 200120)

摘 要: 标识与处理隐蔽通道是美国橘皮书 TCSEC 对 B2 及以上级别安全产品的关键评估要求 ,也是国际标准 CC 评估 EAL5 及以上系统的关键指标. 本文基于安全操作系统 SLinux 的设计开发实践 ,给出一个安全系统隐蔽通道的标识方案、分析流程、分析结果 ,论述了隐蔽通道带宽计算的关键点 ,探讨了隐蔽通道标识与处理方案.

关键词: 隐蔽通道; 安全操作系统设计; 标识与处理
中图分类号: TN393 **文献标识码**: A **文章编号**: 0372-2112 (2007) 01-0153-04

Identifying and Dealing with Covert Channel of the Secure OS-SLinux

LIU Wen-qing[1] ,HAN Nai-ping[2] ,CHEN Zhe[1]
(1. *Institute of Electronic Technology , the PLA Information Engineering University , Zhengzhou , Henan* 450004 , *China ;*
2. *China Standard Software Co , LTD.* 24/ *F Neiwailian Building , NO.* 518 , *Shangcheng Road , Shanghai* 200120 , *China)*

Abstract: Recognizing and dealing with covert channel is the key requirement of TCSEC to evaluate the B2 or above products ,as well as the key requirement of international standard CC to evaluate the EAL5 or above products. Now covert channel analysis is the main bottleneck of development of security information system with high security level. Based on the secure OS (SLinux) designing and implementing ,this paper emphasis describes the identification processing , mainstream methods , bandwidth computation and analysis report of covert channel. It can give important edification on advanced secure information system research.

Key words: covert channel ; designing for security OS ; recognizing and dealing with

1 引言

隐蔽通道是指“给定强制安全模型 M 及其在操作系统中的解释 I(M) ,两个主体 I(Si) 和 I(Sj) 任何潜在通信是隐蔽的 ,当且仅当模型 M 相应主体 Si 和 Sj 之间的任何通信在 M 中是非法的 ”,即通常所说的“允许进程以危害系统安全策略的方式传输信息的通信信道 ”[1]. 美国橘皮书 TCSEC 对 B2 级以上安全产品要求进行隐蔽通道分析[2] ,国际标准 CC 评估第五级 (EAL5) 及以上系统也对隐蔽通道分析提出了明确的要求. GB17859-1999 第四级 (结构化保护级)[3] 明确要求 :“系统开发者应彻底搜索隐蔽存储通道 ,并根据实际测量或工程估算确定每一个被标识通道的最大带宽 ”.

隐蔽通道标识与分析是一个众所周知的困难问题 ,也是一个不断发展中的课题. 自 1973 年开始认识到隐蔽通道以来 ,国际上先后提出了多种隐蔽通道标识方案 ,比较有代表性的有信息流法 (Information Flow)、无干扰法 (Noninterference)、共享资源矩阵法 (SRM)、语义信息流法 (Semantic Information Flow Analysis)、隐蔽流树法 (CFT 法)、改进 SRM 法 (Extended SRM)[4]、回溯搜索法等. 在搜索隐蔽通道方面 ,为减少人工干预 ,设计了 IITRE 流分析器、Gypsy 流分析器等信息流分析工具. 在消除隐蔽通道方面 ,近几年出现了模糊时间技术、模糊传送技术、数据泵技术等新技术成果[5].

SLinux 是一个基于 Linux 资源自主开发的安全操作系统 ,设计目标是符合 GB17859-1999 第四级安全功能要求. 本文则借助其隐蔽通道标识与处理的实践过程 ,详细给出了一个安全系统的隐蔽通道标识的流程 ,分析结果以及隐蔽通道处理方案.

2 隐蔽通道分析流程

安全操作系统隐蔽通道分析包括标识通道、计算带宽、处理通道 ,测试和编写文档等四个阶段.

1. 标识隐蔽通道. 产生隐蔽通道的根源在于系统内部出现了不应该出现的非法信息流 ,因此 ,标识隐蔽通道就是要从系统描述中找出所有潜在的信息流 ,然后再从这些信息流中排除合法的信息流.

2. 计算隐蔽通道带宽. 理论估计和工程测量隐蔽通道的最大可达带宽. 隐蔽通道的带宽与下面因素有关 : ①执行一次隐蔽通道场景所能传送的信息比特数 ; ②执行隐蔽通信所需时间 ; ③其它系统动作对信息传输效率的影响 :通常取决于工作环境、系统载荷等等因素. 在保守估算时 ,通常假定其它系统动作对隐蔽通道传输没有影响.

收稿日期 :2005-12-21 ;修回日期 :2006-08-08
基金项目 :国家科技攻关计划 (No. 2005BA112A02)

3. 隐蔽通道处理. 处理通道的常用手段包括清除信道、降低带宽和审计使用等. 确定处理措施的因素除了通道带宽外,还要考虑受保护的信息本身的数据特性与重要性,以及系统本身的特征等因素. 一般采用策略是: ①采用审计法对信息流进行监督; ②在具体环境允许的情况下,尽可能使用消除法对非法信息流进行控制; ③不得已的情况下采用降低带宽法,即设法降低隐蔽通道的信息传输速率,使得传输结果难以预测.

4. 测试及编写分析文档. 通过测试和编写隐蔽通道分析文档,清楚地说明已经标识出的隐蔽通道,以及采取的处理措施及效果.

安全操作系统从最初提出设计要求,到最终完整的代码实现,要经过建立模型、编写规范说明、编写代码和测试等多个阶段. 由于系统内部的一致性,高层的设计漏洞肯定会被带到低层中,因此设计漏洞发现的越早,纠正错误所需的代价就越小. 但是设计规范不可能包括数据结构等实现细节,无法确保规范与代码具有严格的一致性,因此要完全找出系统中所有隐蔽信道,就必须对全部源代码进行搜索和分析. 基于源代码标识隐蔽通道还具有两个明显的优点: ①能标示所有隐蔽存储通道(除了硬件导致的通道以外); ②能找出放置审计代码、延迟和噪音的位置. 然而,但由于源代码规模庞大和结构复杂,分析源代码极其费时费力.

3 隐蔽通道标识与分析

目前隐蔽通道分析的主流观点是:只有采用信息流分析技术,才能发现隐蔽通道. 标识隐蔽通道,本质上就是分析系统中的非法信息流.

3.1 SLinux 隐蔽通道标识方案

SLinux 隐蔽通道标识方案首先是继承了 Tsai 的语义信息流法的基本思想[1],包括: ①利用信息流分析发现间接读写关系; ②共享变量包含 extern 和 static 两种类型的变量; ③确定共享变量的安全级之后,再使用信息流规则; ④将分析结果分为 P、N、L、E 四种情况.

针对 Linux 核心代码对 Tsai 方法做相应调整,包括: ①不局限于非特权系统调用,面向所有系统调用进行分析; ②分析所有核心源代码; ③深入到系统实现的共享变量,而不是从每一个系统调用入手. ④编写辅助分析工具,如共享变量搜寻器、共享资源矩阵传递闭包生成器等. ⑤按照图 1 所示的流程对 SLinux 进行隐蔽通道分析.

从标识方案可以看出,如果把用户进程可见的系统调用称为"外",把系统的内核变量称为"内",则可以把 Tsai 法称为"自外而内"的分析方案,而 SLinux 采用的是"自内而外"的分析方案. 在这种方案中,不是从系统调用一端着手开始分析,直至找出该系统调用能够读写的共享变量,而是从共享变量一端着手,直至找出能读写该共享变量的所有系统调用. 这就使得 SLinux 隐蔽通道分析方案与 Tsai 方案相比,摆脱了对特定的信息流分析工具依赖,同时减少了工作量.

3.2 SLinux 隐蔽通道分析流程

基于改进的 Tsai 隐蔽通道标识方案,Slinux 隐蔽通道的分析流程如图 1 所示.

3.3 隐蔽通道分析结果

在 SLinux 系统信息流分析的结果上,再通过手工智能分析,共构造出 SLinux 的 13 个隐蔽通道场景,如表 1 所示.

表 1 SLinux 系统隐蔽通道场景

序号	所在函数	所在程序	场景描述	处理建议
1	get_pid()	kernel/fork.c	通过顺序增加进程号进行隐蔽信息传递	对 last_pid 进行噪音处理(加随机数)
2	find_task_by_pid()	include/linux/sched.h	通过探知指定进程存在与否来传递隐蔽信息	加入安全控制策略来消除该信道
3	mac_sb_statfs()	Security/mac/hook.c	通过查询文件系统的空闲文件数、空闲数据数等信息传递隐蔽信息	加入安全控制策略来控制文件系统的信息输出
4	mac_task_kill()	security/mac/hook.c	通过试图向指定进程发信号来传递隐蔽信息	加入安全控制策略来消除该信道
5	mac_ptrace()	security/mac/hook.c	通过跟踪指定进程状态或控制进程执行来传递隐蔽信息	加入安全控制策略来消除该信道
6	ext2_new_block()	fs/ext2/balloc.c	通过资源耗尽指定文件系统的数据区空间传递隐蔽信息	调用隐蔽通道处理函数,根据带宽进行延时或审计处理
7	ext2_new_inode()	fs/ext2/ialloc.c	通过资源耗尽指定文件系统的 Inode 空间传递隐蔽信息	调用隐蔽通道处理函数,根据带宽进行延时或审计处理
8	get_empty_filp()	fs/file_table.c	通过资源耗尽系统的可允许打开文件数来传递隐蔽信息	调用隐蔽通道处理函数,根据带宽进行延时或审计处理
9	do_fork()	kernel/fork.c	通过资源耗尽每个用户可创建进程的数来传递隐蔽信息	调用隐蔽通道处理函数,根据带宽进行延时或审计处理
10	newque()	ipc/msg.c	通过资源耗尽系统的消息队列空间来传递隐蔽信息	调用隐蔽通道处理函数,根据带宽进行延时或审计处理
11	newary()	ipc/sem.c	通过资源耗尽系统的信号量空间传递隐蔽信息	调用隐蔽通道处理函数,根据带宽进行延时或审计处理
12	newseq()	ipc/shm.c	通过资源耗尽系统共享内存空间传递隐蔽信息	调用隐蔽通道处理函数,根据带宽进行延时或审计处理
13	sys_sync()	fs/buffer.c	通过调用同步更新文件系统操作传递隐蔽信息	调用隐蔽通道处理函数,根据带宽进行延时或审计处理

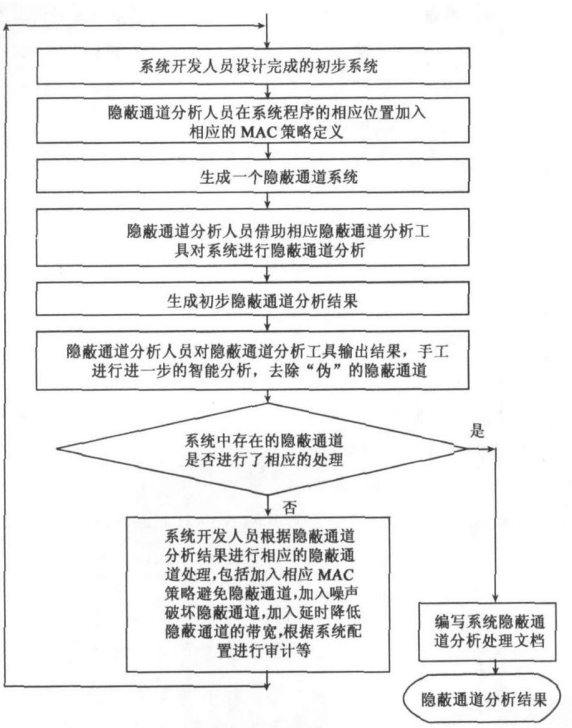

图 1　SLinux 隐蔽通道分析流程

4　隐蔽通道带宽计算

针对隐蔽通道的带宽计算,Tsai 提出一种用 Markov 模型计算带宽的方法[6],并得到一个可用于工程估计带宽的公式 $B = b * (T_r + T_s + 2T_{cs}) - 1$. 其中,$b$ 为信息编码因子,T_r 为平均设置一次读环境并读一个 0 或 1 所用时间,T_s 为平均设置一个 0 或 1 所用时间,T_{cs} 为上下文切换时间. J. Millen 基于信息论的方法,采用有限状态图描述通信的过程,从理论提出了求解隐蔽通道最大带宽计算公式[7]:$B = \log_2 X$. 其中 X 为差分方程 $1 - X^{-d1} - X^{-d2} = 0$ 的解,$d1$ 和 $d2$ 为发送一个 0 和 1 所用平均时间.

由于隐蔽通道的带宽就是“每秒传送的信息比特数(bps)”,为便于编程实现,SLinux 隐蔽通道带宽计算所采用的公式为 B = TotalBits/ DeltaTime. 其中,DeltaTime 是一个采样周期,TotalBits 是一个采样周期内传递的信息总比特数. 确定隐蔽通道带宽问题的关键在于:①确定一个采样周期:该时间周期的确定要考虑系统的实际运行状况,如硬件配置、用户数、负载等. 这个周期应可以进行配置调整,既不能太长也不能太短. ②确定一个事件所代表的信息数:一般来说发生一个事件就意味着可以传递一个比特数的信息,SLinux 根据系统中每个事件的实际情况,进行分析确定. ③计算过程中需要重复多次取平均值,才能得到较为合理的带宽估值.

隐蔽通道带宽计算需要考虑如下因素:①噪音和延迟;②编码和符号的分布;③TCB 原语的选择;④进程和上下文切换时间.

SLinux 隐蔽通道带宽计算采用了一种保守估算方法,忽略了系统动作对隐蔽通道传输所造成的影响. 即其计算公式具体为(采样周期内发生的该类事件的计数 * 一次事件所传递的信息 bit 数)/ 采样周期时间.

5　隐蔽通道处理

处理隐蔽通道,首先应该针对系统产生隐蔽通道的条件,分别采用增加相应的安全策略控制、选取随机数、修改函数的返回信息或错误号等方法,避免隐蔽通道的产生;然后针对系统不可避免的隐蔽通道分类处理,较大带宽的隐蔽通道需要加入延时和审计机制,较小带宽的隐蔽通道仅需要进行审计处理,或采用在文档中进行标识.

5.1　增加安全策略控制避免隐蔽通道

在系统的设计和实现过程中,可以在核心程序和用户层的命令中加入安全策略控制,避免可能产生的隐蔽通道.

SLinux 针对表 1 中的系统隐蔽通道类型 1 和 4,在向指定进程发送信号时(相当于写操作,比如 kill() 系统调用),采用 MAC 方法,实现调用进程的安全级必须等于接收信号进程安全级的策略,来避免隐蔽通道类型 1 和 4 的产生.

5.2　修改函数返回信息避免隐蔽通道

通过修改函数返回信息,可有效避免表 1 中隐蔽通道类型 2、3、5 的产生.

SLinux 需要修改的函数返回信息包括:①对于系统调用的返回值,由于权限不允许,用户访问不到某个进程或文件,一般应该返回 EACCES,但出于避免隐蔽通道的考虑,不能让用户知道该进程或文件是否存在,因此修改函数返回值为 ESRCH,表示该进程或文件并不存在. 这样可以避免类型 2、5 的隐蔽通道产生. ②在文件系统的状态统计(statfs)操作中,如果当前进程的安全级不支配该文件系统的最高安全级,而又没有相应特权时,则需要将输出的该文件系统“空闲块总数”和“空闲文件结点总数”清零,. 这样可以避免类型 3 的隐蔽通道产生.

5.3　加入噪音机制限制隐蔽通道

为避免隐蔽通道产生,设计中需要为系统的一些值选取随机数.

在 SLinux 中实施的噪音机制,如在 fork()/ clone() 系统调用产生子进程中,函数 get_pid() 将为子进程分配一个唯一的 pid. SLinux 是将开始查找空闲 ID 的起点改为一个随机的位置,而不是传统操作系统的上次停止位置,这样可以限制类型 1 隐蔽通道的产生.

5.4　加入延时机制避免隐蔽通道

潜在的类型 6、7、8、9、10、11、12、13 隐蔽通道,在操作系统实现过程中一般是不能避免的. 对于这些类型的隐蔽通道,则需要加入延时机制来最大程度地限制其带宽.

SLinux 在其核心的相应程序中,通过标识隐蔽通道事件记录,来激发隐蔽通道处理函数进行延时或审计处理.

5.4　应用层命令中隐蔽通道的避免与处理

应用层命令中隐蔽通道的产生,主要集中在对系统设备资源竞争和文件锁资源的竞争方面. 为避免隐蔽通道产生,SLinux 也采用了类似内核所采取的方法,包括增加安全策略控制、修改函数返回信息等. 本文以基于文件锁的隐蔽通道处理为示例进行说明.

1. 场景分析. 在操作系统中,同一个文件通常可以被多个

进程读打开,但是只有一个进程可以写打开.当一个进程读打开某个文件时,为了避免另外一个进程写打开该文件导致该文件的内容出现前后不一致,系统会在第一个进程读打开该文件的同时给文件加上读锁.这样,试图写打开该文件的进程通常会得到系统的反馈信息:该文件正在被读.假设系统中只有两个进程在运行,进程 A 创建文件 a,根据系统 MAC 规则,a 的安全级应该等于 A 的安全级.这时一个进程 B 要打开文件 a.假设 B 的安全级支配 A 的安全级,根据 MAC,B 可以读打开 a,然后 A 写打开 a,这时系统就会给 A 提示信息:文件 a 正在被读打开.这样,进程 A 可以推断是进程 B 打开的该文件.如果 B 没有读打开 a,则 A 写打开 a 时将不会收到任何反馈信息.这样,高安全级进程 B 和低安全级进程 A 可以用 B 是否读打开 a 编码实现隐蔽通信,进程 B 对 a 进行一次读打开操作表示 B 发送 0,不做这个读打开动作表示 B 发送 1,进程 A 反复尝试写打开 a,没有反馈信息表示接收到一个 1,接到该文件正在被读打开的反馈表示接收到一个 0.

2.隐蔽通道处理:禁止系统向 A 发出反馈信息,则进程 A 无法知道进程 B 是否读打开文件 a.

6 结束语

标识与处理隐蔽通道是安全操作系统设计的关键技术,也是安全操作系统评测的重要指标.从对 SLinux 安全操作系统的隐蔽通道分析处理过程可以看出:①隐蔽通道是与系统所采用的 MAC 策略紧密相关的,不同的 MAC 策略需要处理的隐蔽通道可能是不同的;②隐蔽通道是与具体的安全操作系统的实现机制密切关联的,同样的安全策略在不同系统中实现时,可能引起的隐蔽通道或需要处理的隐蔽通道是不相同的;③隐蔽通道的标识、分析流程与处理的基本原理与方法,具有通用的借鉴意义,但由于隐蔽通道的标识与搜索,要涉及到具体系统的源代码,所以本文提出的标识与分析隐蔽通道方法,应用于不同版本的 Linux 核心代码和非 Linux 类操作系统时,需要重新做标识、分析与处理.

参考文献:

[1] Tsai C R, Gligor V D, Chandersekaran C S. A formal method for the identification of covert channels in source code[J]. IEEE trans on software engineering, 1990, 16(6): 569 - 580.

[2] DoD 5200.28-STD 1985, Trusted Computer System Evaluation Criteria[S].

[3] GB/T 17859-1999, 计算机信息系统安全保护等级划分准则[S].

[4] Porras P A, Kemmerer R A. covert flow trees: technique for identifying and analyzing covert channe[J]. IEEE Transaction on Software enineering, 1991, 17(11): 36 - 51

[5] 卿斯汉. 高安全等级安全操作系统的隐蔽通道分析[J]. 软件学报, 2004, 15(12) 1837 - 1848.

[6] Tsai C R, Gligor V D. A bandwidth computation model for covert storage channels and its applications[A]. In Proc of the IEEEE Symp on Security and Privacy[C]. 1998. 108 - 121.

[7] Millen J K. Finite-State noiseless vovert channels[A]. In Proc of the Cpmputer Security Foundations Workshop[C]. 1989. 81 - 85.

作者简介:

刘文清 男,1967 年 7 月生于河南虞城,博士,副研究员,主要研究领域为操作系统安全、网络安全. E-mail: wenqing.liu@cs2c.com.cn

韩乃平 男,1969 年 2 月生于山东即墨,高工,主要研究领域为系统软件及安全. E-mail: naiping.han@cs2c.com.cn