使用CC标准开发的高保证安全信息系统*)

袁春阳　贺也平　潘学俭　梁洪亮
(中国科学院软件研究所　北京 100080)

摘　要　通用标准(Common Criteria)提供了衡量系统安全性的流行准则。本文主要提出通过各类保证措施,如何构建符合CC标准的高保证安全信息系统。文中首先给出了CC的评估模型、评估过程和安全保证的具体要求。然后以开发安全审计系统为例,分析了系统安全功能和保证要求的产生、审计系统的实现框架以及为达到标准要求而在系统开发过程中使用的各种保证证据和保证措施。最后,又分析了审计系统对整个系统的性能影响因素,并提出了改进办法。本文通过深入剖析通用标准中各个保证要求的内涵,为开发具有高保证要求的信息系统提供了理论指导和实现方法。

关键词　通用标准CC,审计,安全保证

Developing High-assurance Secure Information System According to Common Criteria

YUAN Chun-Yang　HE Ye-Ping　PAN Xue-Jian　LIANG Hong-Liang
(Institute of Software, Chinese Academy of Sciences, Beijing 100080)

Abstract　Common Criteria provides a popular criteria to measure the security of system. This paper mainly poses how to build a high-assurance secure information system according with Common Criteria by kindes of assurance measures. Firstly, the evalution model, evalution procedure of Common Criteria and concrete security assurance requirements are provided. Then taking example of developing secure audit system, this paper discusses the production of security function and assurance requirements of system, the implementation framework of audit system, the asurrance evidences and measures during the development process of system. At last, the perfomance effect elements of audit system are analysed and some improvement measures are provided. According to analyse the meanings of assurance requiremens deeply, the instruction in theory and implement method to develop high-assurance information system are provided.

Keywords　Common criteria, Audit, Security assurance

1　概述

当今信息系统面临着不同的安全威胁,尤其是随着计算机的普及和网络的发展,安全问题显得日益突出,并已引起了人们的重视。并且针对不同的应用环境和安全目标,开发出了不同的安全信息系统。如何评估安全信息系统确实已经到达了预定的安全目标,成为安全领域研究的重点之一。这便是安全保证的概念,即信息系统满足用户安全需求的可信度,主要解决的问题是如何确保系统能够达到其所宣称的安全程度[1]。安全保证依靠保证论据(argument)来实现。保证论据解释了系统的安全措施是有效的、可靠的。一般要由开发者构建、并由独立的第三方测评机构进行评估。保证论据应至少包括以下3个方面[2]:

• 保证声明(Assurance Claim):开发者宣称系统应有的某些安全相关的属性。这些声明要包含系统可能遇到的所有威胁。

• 保证证据(Assurance Evidence):对可信性做出判断或结论所要依赖的数据,并说明系统是否存在可被利用的缺陷。

• 环境证据(Circumstantial Evidence):证实主要证据可靠性的证据,包括用于创建系统的过程、人员和环境。

保证过程中最重要的便是确定系统需要达到的目标。由于应用环境不同,信息系统面临的安全威胁各有所异,为了便于统一评价,国际上出现了不同的安全评估标准[3],如上世纪八十年代美国国防部颁布的TCSEC标准[4],以后陆续出现的美国信息技术安全评估联邦标准Federal Criteria[5]、欧洲的ITSEC标准[6]和加拿大可信计算机产品评估标准CTCPEC[7]。目前最为流行的标准便是六国七方互认的信息技术安全评估通用标准(Common Criteria,简称CC标准)[8]。我国亦发布了与CC等同的标准,即GB/T 18336-2001[9]。CC标准的一个显著特点是将安全信息系统的功能要求与保证要求分离。

除了构建保证目标外,近年来,研究者提出和发展了很多科学方法来规范、构建和认证高安全保证的系统,其中包括形式化规范技术、形式化模型、设计方法、严格的验证及确认技术[10]。而保证技术根据其应用方式可以分为3类:过程、设计和构建、检验。过程保证技术是通过在开发过程中使用过程方式进行保证,如开发时采取严格、合理的设计过程和软件开发方法,包括遵从良好的编码风格和实施严格代码审查。

*)本文研究得到国家自然科学基金项目(No. 60373054)和中国科学院研究生院创新资金的资助。**袁春阳**　博士研究生,CCF会员,主要研究方向为系统软件与信息安全;**贺也平**　博士,研究员,博士生导师,主要研究方向为计算机安全与系统软件;**潘学俭**　硕士研究生,主要研究方向为计算机安全与系统软件;**梁洪亮**　博士,副研究员,硕士生导师,主要研究方向为计算机安全与系统软件。

这类保证如系统安全能力成熟度模型标准 SSE-CMM[11]。设计和构建保证主要在建立和构建组件时提高可信任级别。它与过程保证的不同是，强调了组件本身的设计和构建，而不是创建组件的过程。这类保证技术如测试、形式化技术。检验保证是在系统建立后，提高可信任级别。如产品的操作历史状况、第三方评估、形式化评估、开发时的各类文档、依据安全标准进行相应的测试等。保证检验的另外一个方法，是对系统进行风险评估，评价系统能否抵御某些攻击和风险。CC 标准就属于检验保证。但是，其整个保证过程同时可以指导系统的开发和构建。系统开发者可以使用 CC 方法建立一种保证，即按照该标准开发的产品或系统能够满足特定安全性能标准，从而通过到达国际上共同认可的标准来减少安全风险。

我们依据 CC 标准设计和开发了符合评估保证级 EAL4 要求的安全操作系统。作为其中的一个重要子系统，安全审计系统也是完全遵循安全标记级保护轮廓(LSPP)[12]的要求而实现的。在整个开发过程中，采用了各种安全保证措施，从而保证系统确实能够达到了所期望的安全目标。

目前，国内已有学者对 CC 标准进行了介绍。文[13]给出了通用标准的结构模型，并给出了根据其模型研制操作系统安全核心系统的基本作法。文[14]通过 CC 标准来确定红旗 Linux 的安全可信度。但是两文中均未给出具体的保证证据，以及如何利用保证证据和措施来确定系统的安全可信度。而本文根据开发安全操作系统过程中的实践经验和保证方法，提出如何根据 CC 标准的要求设计和实现符合高保证要求的信息系统，明确开发中的各类保证证据和措施。安全审计系统是各类安全系统必备的安全组件，具有通用性，而且又与其他系统紧密相关，所以我们将重点以其为例进行介绍。而对安全操作系统中的其他模块，如自主访问控制、强制访问控制、特权处理等，不再详细赘述。

本文第 2 部份提出了 CC 标准中的保证模型和保证过程。然后概述根据 CC 标准如何开发安全信息系统，并以安全审计系统的开发为例，详细阐述开发过程中的各种保证方法和措施。本文还强调了审计系统的性能要求，提出了性能改进的办法。最后总结全文。

2 CC 标准的评估模型和保证过程

2.1 评估模型

CC 标准将安全要求分为功能要求和保证要求。保证要求又逐步细分为保证类、保证族和组件。系统所达到的评估保证级 EAL 是根据其能够满足的保证要求而确定的。而评估时重点主要围绕评估对象 TOE 和保护轮廓 PP。第三方认证中心首先针对不同类型的系统发布标准的安全目标 ST，由开发者进一步细化此安全目标，并建立保护轮廓 PP。系统评估时，便根据 PP、TOE 以及其他保证证据来评测系统是否满足 PP 所预定的评估保证级 EAL。

CC 标准的评估模型如下：

$$EAR=ES\cdot\sum_1^n RC_i=ES\cdot\sum(EM,EV,AR)$$

$$EAL_i\rightarrow AR,i=1,\cdots,7;\ SR=\{AR,\ FR\}$$

其中：

• *EAR* 为最终的评估保证结果，由对每个保证组件的评估结果 *RC* 的交所得。每个保证组件的评估结果包括 3 种情况“通过”、“不通过”和“无法确定”，即 $RC\in\{pass,\ inconclusive,\ fail\}$。但是，最终评估只有两种结果，即所有的 *RC* 均为“通过”时，*EAR* 才为“通过”。当有一个 *RC* 为“不通过”或“无法确定”时，*EAR* 为“不通过”。造成单个保证组件的评估结果为“无法确定”的原因可能是保证证据不够充分或保证声明不够具体。此时，便需要修改保证声明或补充保证证据，重新进行评估。

• 评估机制 *ES* 是指评估者应用 CC 标准的管理和规定框架对 TOE 进行评估。

• *SR* 是系统的安全要求，分为功能要求 *FR* 和保证要求 *AR*。*FR* 来自于评估对象 TOE 的安全目标 ST 和保护轮廓 PP。而 *AR* 来自于安全保证级 EAL_i，即 $EAL_i\rightarrow AR$。不同的安全保证级提出了不同的安全保证组件，这些保证组件确定了保证要求，标准中没有提及的保证，可以在 PP 中进行补充。

• 评估方法 *EM*，对不同的保证要求或组件采用的评估方法并不相同，如可采取测试、形式化验证等方法。

• 评估证据 *EV*，能够证明 TOE 满足 *AR* 的证据，包括各类文档(如用户使用手册)、TOE 的实现表示、策略模型与 TOE 的对应性说明等。主要由系统开发者和测试员提供给第三方认证中心使用。

CC 标准的评估模型如图 1 所示。根据该评估模型，我们可以很清楚的看到，系统达到的评估保证级 EAL 是由其能够满足的保证要求所确定的，而不是依据功能的多少来决定。

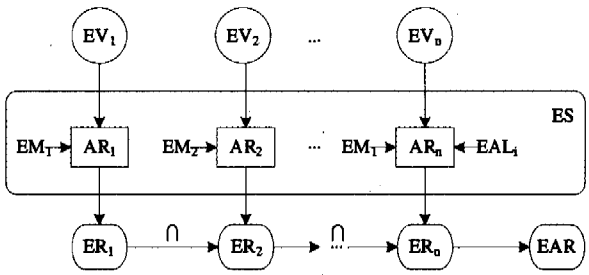

图 1 CC 标准的评估模型

2.2 保证过程

开发系统时一般经过分析、设计、实现、运行和维护阶段。根据系统开发的一般过程，我们将 CC 标准的保证过程大致分为 4 步，如图 2 所示。

(1)确定安全目的：从 TOE 当前面临的威胁、采取的假设和策略出发，确定安全操作系统应当到达的安全目的，安全目的又决定了安全功能要求。这些内容应在 TOE 的 ST 中有所体现。

(2)设计和实现过程保证：在确定了安全目的和安全功能要求以后，通过对功能要求的逐步细化，明确开发过程中的功能规范、高层设计和低层设计，并最终实现 TOE。为保证每次细化的完整性和完备性，每次细化过程中都要有高层与低层之间的对应性表示说明。

(3)运行和维护保证：对实现的系统进行测试，包括深度分析和覆盖分析，确保系统的功能和接口的正确性。配置管理、交付运行、生命周期支持保证类确保系统操作和维护阶段的正常使用。保证维护能够确保系统在评估后依旧能够维护其评估保证级的要求。

(4)认可：通过脆弱性评估，确信系统是否存在漏洞，并能达到某种安全需求，从而认可该系统符合 EAL 级。

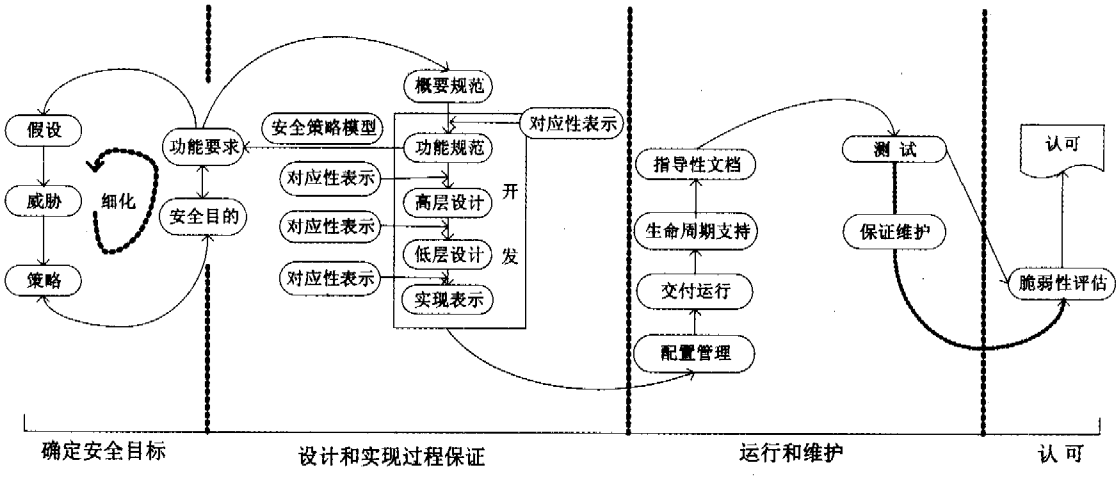

图 2　CC 标准的保证过程

3　安全审计系统的安全要求和保证要求概述

本节概述如何根据 CC 标准产生审计系统的具体安全要求，以及系统设计和实现、操作和维护过程中各个保证类的不同保证要求。

3.1　安全要求的产生

安全要求有安全功能要求和安全保证要求两部份。安全功能要求主要来自于 TOE 的使用环境要求。通过说明安全使用时的假设条件、分析 TOE 所面临的威胁，以及采取的组织安全策略，总结 TOE 和 IT 环境下的安全目的。不同的安全功能组件满足一个或多个安全目的要求。安全保证要求来自于系统要达到的评估保证级 EAL。

系统只有在特定的环境下才能正常工作，需要明确系统的正常工作情景，在各种前提假设下，保证系统发挥正常的效能。对使用环境的假设，主要来自于物理、人员、工作程序和连接四个方面。物理假设要求 TOE 必须能够从物理上在受控的条件下使用，如要求的使用场所、能够防止物理上对系统的破坏；人员假设提出管理和使用 TOE 的用户必需具备的素质和能力；工作程序要求 TOE 的操作必须遵守规定的流程；连接假设说明 TOE 是否属于网络或分布式系统，与网络的连接状况。

TOE 在运行期间会遇到来自各种方面的安全威胁，如恶意代码、隐蔽信道等。这些安全威胁应当是 TOE 实现时要解决的。为保护敏感数据，对操作这些数据的动作施加一组特定的规则或程序，即组织安全策略。如“策略. 责任”是指用户在系统中的活动，包括对敏感数据的修改等，都应该是可追责的。

TOE 在设计初期，明确 TOE 要达到的要求是必要的。TOE 安全目的列出了对 TOE 的要求。这些目的解决或减缓了 TOE 所面临的威胁、保证组织安全策略。对于环境方面的假设，则通过环境方面的目的来映射。目的与威胁、组织安全策略和环境假设之间的这种映射都是多对多的关系，即一个目的可能会解决一个或多个威胁，一个威胁可能需要一个或多个目的来降低。依据对 TOE 所处环境、面临的安全威胁，分析需要达到的安全目的，便可以从 CC 本身提供的很多预定安全要求（包括功能要求和保证要求）中选取 TOE 的具体要求。安全要求具有充分必要性，即每一个安全目的都至少有一个安全要求组件与其对应，每一个安全要求都至少解决了一个安全目的。

3.2　保证要求

在确定了系统的安全功能要求和保证要求后，便进入了 TOE 的开发阶段。整个开发过程中包含了保证过程的一部分，但保证过程长于开发过程，渗透了系统生命周期的每个阶段。

保证类（ADV）贯穿着 TOE 的设计到最终实现。根据 TOE 的功能和保证要求，开发者提出系统的概要规范，描述这些要求的具体内容。功能规范（ADV_FSP）进一步依据功能要求，建立安全策略模型（ADV_SPM），细化功能要求组件。高层设计（ADV_HLD）体现了系统实现安全功能要求的方案和实现路线。低层设计（ADV_LLD）是对高层设计的细化和求精，更贴切于系统的实现。而实现表示（ADV_IMP）则是 TOE 真正的实现。从功能规范起，到实现表示终止，每一次都是进一步的细化。为保证这种从略到详的过程的一致性和完备性，可以使用对应性表示（ADV_RCR）组件确保求精的同时，不会产生遗漏和偏差。

配置管理（ACM）保证类对 TOE 及其他信息（如开发文档等）的修改过程进行规范和控制，确保 TOE 的完整性。交付运行类（ADO）提供了安全交付、安装和运行 TOE 的措施和程序，确保 TOE 所提供的保护在传递、安装和运行时不会被减弱。生命周期支持类（ALC）定义 TOE 开发过程的生命周期模型，从而明确保证要求。指导性文档类（AGD）要求针对用户和管理员，提供相应的指导性文档，是保证 TOE 安全运行的重要因素之一。测试类（ATE）对 TOE 进行特定范围内的测试，论证 TSF 满足了 TOE 安全功能要求。保证维护类（AMA）确保 TOE 或其环境发生变化时，TOE 仍能维护其保证级别，满足安全目标。脆弱性评估类（AVA）标识出开发过程中可能引入的潜在缺陷，判断在实际应用时其是否会被利用。

4　安全审计系统的实现和保证方法

本节以安全审计系统的具体实现和保证方法，说明如何根据 CC 标准提供的保证机制，开发高保证的安全信息系统。

4.1　功能要求

系统面临着用户权限滥用的威胁，记为“威胁. 用户权限滥用”。为了提供个人责任追究的能力，要求审计系统能够记录每个用户的具体动作，这种目的记为“目的. 审计用户”。TSF 必须对 TOE 用户的涉及安全性的行为进行记录，并向具有相应权限的管理员提供这类记录信息，记为“目的. 审计

产生”。只有具有相应权限的管理员才能查看审计信息，并提供能够选择性地查看审计信息的能力，记为“目的.审计查阅”。

TOE的审计部件应提供如下基本功能：记录安全相关的所有重要事件。管理员用户可以根据需要选择审计事件，同时通过配置管理工具查看审计记录。防止非授权用户对审计记录文件的访问和破坏。对不能由TOE独立分辨的审计事件，提供审计记录接口，由授权主体调用。从CC标准的安全审计类(FAU)中六个功能族中选取四个：审计数据生成族(FAU_GEN)给出了记录安全相关事件的一些要求；审计事件选择族(FAU_SEL)定义了预先或在安全操作系统运行过程中从可审计事件全集之中选择部分或全部事件进行审计的要求；审计查看族(FAU_SAR)对授权用户使用的审计工具查看审计数据提出了要求；审计事件存储族(FAU_STG)对审计踪迹的建立、保护和维护提出了要求。由于系统尚未实现入侵检测功能，所以没有选取审计分析族(FAU_SAA)和安全自动响应族(FAU_ARP)。

4.2 实现框架

我们以FreeBSD系统为基础，已开发了满足EAL4评估保证级要求的安全操作系统。作为该安全操作系统的一部分，安全审计子系统在满足EAL4保证级所有要求的同时，又符合业界流行的规范，即Sun公司的BSM(Basic Security Module)[15]，并支持对其他模块(如强制访问控制)行为的审计。具体实现时，可以分为两部份：用户态部分和内核部分。用户态部分由向用户态程序提供接口的libbsm库函数、审计管理命令、负责接收并处理来自管理命令和内核审计线程的信号的守护进程auditd，以及审计的相关配置和日志文件。

内核部分负责采集审计事件信息并记录到日志文件中，主要包括以下几个部件：

·审计事件采集点

扩充系统调用，确定能够获取审计事件相关信息的最佳位置。

·审计事件管理模块

根据审计配置不同，预选所需审计事件，设定审计屏蔽字。

·审计记录队列

存放审计记录的双向链表，以减少对磁盘的频繁写操作。

·内核审计线程

从审计记录队列中获取审计记录，并写入到审计日志文件中。

·审计日志管理模块

管理审计日志文件，如当审计日志占满整个磁盘空间时，发出报警等。

·审计的安全管理

对审计日志文件提供安全保护，审计操作只能由授权用户执行。

安全审计系统的整体框架如图3所示。

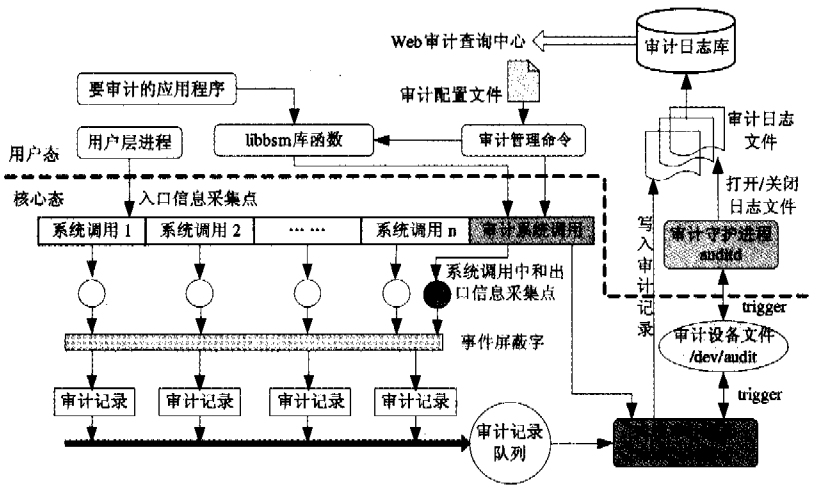

图3 安全审计系统的整体框架

4.3 评估证据和保证措施

CC标准中，不同的保证类有不同的保证族。根据EAL4评估保证级的要求，我们对各安全保证族采用的评估证据如表1所示。保证族ADV_INT、ALC_FLR和AVA_CCA不做要求，故未列出。

(1)配置管理保证类ACM

ACM_AUT.1的保证措施是使用协同开发软件CVS来控制对源码的修改，脚本和CVS提供产生TOE时的自动支持。ACM_CAP.4为每一个版本的TOE有一个唯一的标识名称。TOE所有的组件都与配置项目相关联，每一个版本的配置项目都被唯一标识。所有对配置项目的修改都是可复查的和被认可的。ACM_SCP.2不仅源码，而且也应包括文档、产品缺陷和开发工具等都使用严格的版本控制机制，从而可以重现配置管理项目的以往版本，并能确定任何配置管理项目之间的差别。

(2)交货与使用保证类ADO

ADO_DEL.2和ADO_IGS.1确保TOE被安全地交货给了用户，而且用户能够安全地安装并启动TOE。措施主要为用户提供规范、明了的安装说明。

(3)开发保证类ADV

开发保证类中的保证族符合一般的开发过程，即首先描述系统功能，然后逐步细化，进行概要、详细设计以及编码实现。ADV_FSP.2描述TOE安全功能和TSF外部的用户可见接口。ADV_FSP.2为记录TSF的概要设计。ADV_LLD.1记录TSF的详细设计。ADV_IMP.1提供源码实现表示。ADV_RCR.1描述所有的设计抽象层之间的对应关系，采取的保证措施可有形式化对应表示、非形式化说明，确保从高层功能描述到概要设计、详细设计直至最终源码的实

现之间能够进行完全映射。

(4)指南资料保证类 AGD

AGD_ADM.1 和 AGD_USR.1 采取的保证措施分别是为提供管理员和用户指南,包括详尽的安装和使用说明,如 man 手册等。

(5)生命周期支持保证类 ALC

ALC_DVS.1 要说明在 TOE 开发环境中保护 TOE 设计和实现的安全性的所有物理、过程、私人以及其他安全措施。ALC_LCD.1 使用标准的生命周期管理对 TOE 的修改,明确分工、开发进度和监控机制,保证系统能够按照预定进度正常进行。ALC_TAT.1 定义了开发工具,包括协同开发工具(如 CVS)、编码规范等。

表 1 EAL4 的保证要求和评估证据

保证类	保证族	评估证据	保证类	保证族	评估证据
配置管理 ACM	ACM_AUT.1	系统的配置管理文档 系统的生命周期支持文档	生命周期支持 ALC	ALC_DVS.1	系统的生命周期支持文档
	ACM_CAP.4			ALC_LCD.1	
	ACM_SCP.2			ALC_TAT.1	
交货与使用 ADO	ADO_DEL.2	系统的交付和运行文档	安全测试 ATE	ATE_COV.2	系统的测试文档(包括对系统的功能、接口进行的各种黑盒与白盒测试)
	ADO_IGS.1			ATE_DPT.1	
开发 ADV	ADV_FSP.2	系统功能规范		ATE_FUN.1	
	ADV_HLD.2	系统概要设计说明书		ATE_IDN.2	
	ADV_IMP.1	系统实现代码表示			
	ADV_LLD.1	系统详细设计说明书	脆弱性评估 AVA	AVA_MSU.2	配置管理、指南性和生命周期支持文档
	ADV_RCR.1	系统概要和详细设计说明书;功能规范		AVA_SOF.1	系统遵从的 ST
	ADV_SPM.1	系统的安全策略模型		AVA_VLA.2	脆弱性分析
指南资料 AGD	AGD_ADM.1	系统的管理员手册			
	AGD_USR.1	审计系统用户手册			

(6)安全测试保证类 ATE

ATE_COV.2 采取的保证措施为每个 TSF 接口提供多种测试。对 ATE_DPT.1 采取的保证措施为很多测试实例测试了每个子系统,对测试深度的分析说明了哪些测试实例测试了哪个子系统和模块。ATE_FUN.1 要说明每个测试的目的、方法和期望的结果。ATE_IND.2 则说明测试实例安装和运行的方法。

对系统的功能、接口进行黑盒与白盒测试,并根据测试目的、方法、实际结果和期望结果,形成测试文档。将测试中发现的问题直接提交 bugzilla,通过开发人员的确认、修正和关闭以及测试人员的复查等过程,实施严格的代码质量控制管理。

(7)脆弱性评估保证类 AVA

脆弱性评估包括标识并试图利用系统中存在的弱点。脆弱性可能来自于模块设计和实现时的漏掉或者与其他模块的交互。AVA_MSU.2 要求文档列出了操作的所有模式、IT 环境的所有假设和外部安全措施的所有要求,这些指南性文档是完全的、一致的和清晰的。AVA_SOF.1 主要针对安全系统中的口令机制,如密码猜测进行安全功能强度分析,确认其强度满足了 ST 和 LSPP 的要求。AVA_VLA.2 采取的保证措施为系统地搜索 TOE 中所存在的安全漏洞。搜索漏掉的主要方法是渗透测试。通过对系统的试图入侵,发现潜在的漏洞。

5 安全审计系统的性能分析

审计系统需要记录系统活动和多次的磁盘写入操作,会对给系统带来较大的运行负载,因而对于审计系统而言,除了安全功能和保证要求外,性能要求也是非常重要的一方面。常用的性能测试工具是 Lmbench[16],它会针对系统的各种应用程序接口进行测试,包括基本系统调用、上下文切换效率、文件系统延迟和内存带宽等[17]。在测试环境为 CPU P4 2.4GHz、内存 512DDR、硬盘 IDE 80G 7200 转的情况下,对我们开发的安全审计系统的性能进行测试,部分结果见表 2,表中省略了影响在 5%以下的其他非重要性能测试参数。

表 2 安全审计系统的性能测试部份结果

Item		Standard FreeBSD	FreeBSD only with Audit Lowering Rate(%)	FreeBSD with MLS and Audit Lowering Rate(%)
Processor, Processes - times in microseconds - smaller is better	Null call	0.52	0.55(5.8)	0.55(5.8)
	null I/O	0.82	0.85(3.7)	0.85(3.7)
	stat	6.61	6.87(3.9)	6.88(4.1)
	open clos	9.14	15.0(64.1)	15.1(65.2)
	sig inst	0.77	0.79(2.5)	0.82(6.5)
	sig hndl	2.57	2.82(9.7)	2.84(10.5)
	fork proc	268.	287.(7.1)	287.(7.1)
	exec proc	1274	1484(16.5)	1495(17.3)
	sh proc	3775	4293(13.7)	4303(14.0)

从整个测试结果来看,无强制访问控制与加入强制访问控制的审计系统相比而言,对系统性能影响不大。而审计系

(下转第 47 页)

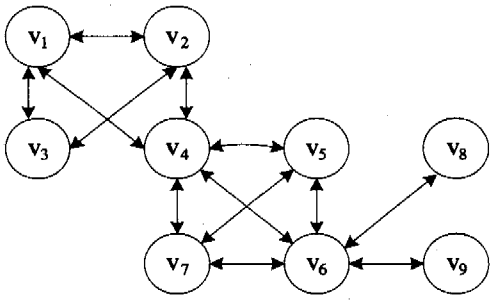

图 3 VPN 拓扑发现

结论 本文利用在 BGP/MPLS VPNs 中存储的配置信息 VRF 和 RT，描述了使 VPN 拓扑发现过程自动化的算法。我们通过分析 VPN 结构，并利用简单的原子组件和分子组件表示 VPN，设计了 VPN 拓扑发现算法。我们的算法使用矩阵模型表现了 VPN 并识别简单的组件。使用该算法，服务提供者能使用当前网络配置信息自动地发现 VPN 拓扑。

参 考 文 献

1 Rosen E, Rekhter Y. RFC2547: BGP/MPLS VPN. IETF, March 1999

2 HP Openview Network Services Management Solution for MPLS Networks. http://www.hp.com

3 Kim Youngtak, Choi Hyung-Woo, Kim Hyo-Sung. A QoS-guaranteed DiffServ-aware-MPLS VPN and its Network Management System. SNPD, 2003

4 Ould-Brahim H, Rosen E C, Rekhter Y. Using BGP as an Auto-Discovery Mechanism for Layer-3 and Layer-2 VPNs. IETF draft, June 2005

5 Tomsu P, Wieser G. MPLS-Based VPNs Designing Advanced Virtual Networks. Pearson Education, December 2001

6 Semeria C. RFC2547bis: BGP/MPLS VPN Fundamentals. Juniper Networks Inc White Paper, 2001

7 Sangli S R, Tappan D. BGP Extended Communities Attribute. Internet Draft, draft-ietf-idr-bgp-ext-communities-07. txt, March 2004

8 White R, McPherson D, Srihari S. Practical BGP. Addison Wesley Professional, July 06, 2004

(上接第 21 页)

统对整个操作系统造成的性能影响主要体现在系统调用和文件系统访问操作上。原因是审计系统在系统调用处采集审计信息，并将这些信息写入磁盘，这是一种比较耗时的操作。提高审计系统性能的方法，包括选择合适的审计点、减少审计记录的内容、使用审计记录缓冲区以较少写磁盘的次数、采取合理的审计配置、优化冗余代码等。开发审计系统时应采取以上各种措施，以尽量减少审计系统所带来的系统负载。

结论 通用标准 CC 已成为了当今评估安全信息系统安全性的常用标准。本文以安全审计系统开发为例，描述了如何使用各种保证措施，以满足 CC 标准对安全保证的要求，为今后开发高保证的安全信息系统，尤其是安全操作系统，提供了理论指导和实现思路。

参 考 文 献

1 National Institute of Standards and Technology. Invitational Workshop on Information Technology (IT) Assurance and Trustworthiness. March 1995

2 Williams J R, Schaefer M, Landoll D J. Pretty Good Assurance. In: Proc. of the New Security Paradigms Workshop, 1995

3 蔡昱，张玉清，孙铁，等. 安全评估标准综述. 计算机工程与应用，2004，40：129～132

4 CSC-STD-001-83, Department of Defense Standard. Department of Defense Trusted Computer System Evaluation Criteria. DoD Computer Security Center, August 1983

5 DoD 5200. 28-STD, Department of Defense Standard. Department of Defense Trusted Computer System Evaluation Criteria. National Computer Security Center, Ft Meade, MD, USA, Dec. 1985

6 France, Germany, the Netherlands, the United Kingdom. Information Technology Security Evaluation Criteria, Version 1. 2. Office for Official Publications of the European Communities, Jun. 1991

7 Canadian System Security Centre. The Canadian Trusted Computer Product Evalution Criteria, Draft Version 3. 0e. Government of Canada, April 1992

8 Joint Technical Committee 1. Evaluation Criteria for IT Security? Part 1, 2, 3. ISO/IEC 15408-1：1999(E), The International Organization for Standardization and the International Electrotechnical Commission, 1999

9 中国信息安全产品测评认证中心. GB/T 18336-2001、GB/T 18336-2001. 信息技术安全技术信息技术安全性评估准则

10 McLean J, Heitmeyer C. High Assurance Computer Systems: A Research Agenda. America in the Age of Information, National Science and Technology Council Committee on Information and Communications Forum, Bethesda, 1995

11 Ferraiolo K, Gallagher L, Thompson V. Building a Case for Assurance from Process. In: Proceeding of the 21 National Information Systems Security Conference, October 1999

12 LSPP. Labeled Security Protection Profile, Version 1. b. National Security Agency. October 1999

13 石文昌，孙玉芳. 信息安全国际标准 CC 的结构模型分析. 计算机科学，2001，28(1)：8～11

14 石文昌，孙玉芳. 通过 CC 标准的思想确定 RS-Linux 的安全可信度. 广西科学，2001，8(1)：15～18

15 BSM. SunSHIELD Basic Security Module Guide (Solaris 8). Sun Microsystems, Inc, February 2000

16 LMbench - Tools for Performance Analysis. http://lmbench.sourceforge.net/

17 McVoy L, Staelin C. lmbench: Portable tools for performance analysis. In: Proc. Winter 1996 USENIX, San Diego, CA, January 1996. 279～284