普通高等教育“十五”国家级规划教材

信息安全概论

An Introduction to Information Security

段云所　魏仕民　唐礼勇　陈　钟

高等教育出版社

ISBN 7-04-012314-2
9 787040 123142 >
定价：21.20元

普通高等教育“十五”国家级规划教材

信息安全概论

段云所　魏仕民
唐礼勇　陈　钟

高等教育出版社

内容提要

本书被列为普通高等教育“十五”国家级规划教材。本书系统地论述了信息安全的理论、原理、技术和应用。主要内容有：对称加密算法（DES、AES）、公钥密码算法（RSA、ECC）安全散列算法（SHA）、数字签名（DSS）、数字证书、认证机构 CA、身份认证、访问控制、安全审计、安全威胁分析、安全扫描、入侵检测、防火墙、IPSec 协议、SSL 协议、安全评估标准（TCSEC、CC、GB17859）、Web 安全、Email 安全（PGP、S/MIME）、电子商务安全（SET 协议）等。

本书适合作为高等院校本科或研究生教材使用，也可供研究或开发人员参考。

图书在版编目(CIP)数据

信息安全概论 / 段云所等编. —北京：高等教育出版社，2003.9

ISBN 7-04-012314-2

Ⅰ. 信… Ⅱ. 段… Ⅲ. 信息系统-安全技术-高等学校-教材 Ⅳ. TP309

中国版本图书馆 CIP 数据核字（2003）第 045277 号

出版发行	高等教育出版社	购书热线	010-64054588
社　　址	北京市西城区德外大街 4 号	免费咨询	800-810-0598
邮政编码	100011	网　　址	http://www.hep.edu.cn
总　　机	010-82028899		http://www.hep.com.cn
经　　销	新华书店北京发行所		
印　　刷	涿州市星河印刷厂		
开　　本	787 × 1092　1/16	版　　次	2003 年 9 月第 1 版
印　　张	16.75	印　　次	2003 年 9 月第 1 次印刷
字　　数	330 000	定　　价	21.20 元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换。

前　言

随着网络应用的发展和普及，网络与信息安全的重要性日益突出。国内外有关信息安全的研究与开发力度都在不断加大。学术界研究力量明显增加，许多大学和科研机构都设立了信息安全研究室（所、院）。科技主管部门也加大支持力度，如“十五”期间，国家“863”计划专门设立信息安全主题，重点支持信息安全领域关键技术的研究和产业化。有关部门也出台了相应法规、标准、指南，成立专门的测评认证机构，加强对信息安全的监管。产业界涌现了大批信息安全公司，仅防火墙一个产品，国内就有几百家公司在研制生产。行业用户也加大投入，很多行业纷纷制定技术规范和总体方案，并组织产品选型。实施安全建设。普通个人用户也十分关心自己计算机的安全和隐私。总之，信息安全引起了社会各界的广泛关注，面对这样的局面，高等院校开始将信息安全纳入主修课程，本书正是为适应这样的需求而编写的。

本书是作者在北京大学开设信息安全课程的讲义的基础上完成的，比较全面地论述了信息安全的基础理论和技术原理，包括密码理论与应用、身份认证、访问控制、审计、安全脆弱性分析、入侵检测、防火墙、安全协议等。为了让学生能更好地将理论和原理与应用结合起来，还安排了安全标准和应用安全等内容。在具体编排上既考虑内容的完整性，又考虑到课时的限制，因此，大部分章节适合一周（3~4 学时）内讲授，少数章节需要 5~6 学时，但根据具体情况可删节。全部课程约需 50~60 学时，适合一学期讲授。

本书被列为普通高等教育“十五”国家级规划教材。本书的编写得到高等教育出版社的大力支持。北京大学计算机系的研究生刘迎、胡嵩、张锦懋、武勇、张明、徐鹏为本书的编写做了很多工作。在此一并表示衷心感谢。

由于编者水平有限，时间仓促，书中难免有错误和不当之处，敬请读者和同行专家批评指正。

编　者

2003 年 5 月于燕园

策划编辑　韩　飞
责任编辑　韩　飞
封面设计　于文燕
版式设计　陆瑞红
责任校对　杨雪莲
责任印制　陈伟光

目　录

第一章　概　述

1.1　信息安全的目标

通信、计算机和网络等信息技术的发展大大提升了信息的获取、处理、传输、存储和应用能力，信息数字化已经成为普遍现象。互联网的普及更方便了信息的共享和交流，使信息技术的应用扩展到社会经济、政治、军事、个人生活等各个领域。因此，信息安全的重要性可以上升到国家安全的高度。

无论在计算机上存储、处理和应用，还是在通信网络上传输，信息都可能被非授权访问而导致泄密，被篡改破坏而导致不完整，被冒充替换而导致否认，也可能被阻塞拦截而导致无法存取。这些破坏可能是有意的，如黑客攻击、病毒感染；也可能是无意的，如误操作、程序错误等。

那么，信息安全究竟关注哪些方面呢？尽管目前说法不一，但普遍被接受的观点认为，信息安全的目标是保护信息的机密性、完整性、抗否认性和可用性；也有观点认为是机密性、完整性和可用性，即 CIA(Confidentiality，Integrity，Availability)。

- 机密性(Confidentiality)

机密性是指保证信息不被非授权访问；即使非授权用户得到信息也无法知晓信息内容，因而不能使用。通常通过访问控制阻止非授权用户获得机密信息，通过加密变换阻止非授权用户获知信息内容。

- 完整性(Integrity)

完整性是指维护信息的一致性，即信息在生成、传输、存储和使用过程中不应发生人为或非人为的非授权篡改。一般通过访问控制阻止篡改行为，同时通过消息摘要算法来检验信息是否被篡改。

- 抗否认性(Non-repudiation)

抗否认性是指能保障用户无法在事后否认曾经对信息进行的生成、签发、接收等行为，是针对通信各方信息真实同一性的安全要求。一般通过数字签名来提供抗否认服务。

- 可用性(Availability)

可用性是指保障信息资源随时可提供服务的特性，即授权用户根据需要可以随时访问所需信息。可用性是信息资源服务功能和性能可靠性的度量，涉及物理、网络、系统、数

据、应用和用户等多方面的因素，是对信息网络总体可靠性的要求。

1.2　信息安全的研究内容

信息安全是一门交叉学科，涉及多方面的理论和应用知识。除了数学、通信、计算机等自然科学外，还涉及法律、心理学等社会科学。本书只从自然科学的角度介绍信息安全的研究内容。

信息安全研究大致可以分为基础理论研究、应用技术研究、安全管理研究等。基础理论研究包括密码研究、安全理论研究；应用技术研究包括安全实现技术、安全平台技术研究；安全管理研究包括安全标准、安全策略、安全测评等。各部分研究内容及相互关系如图 1.1 所示。

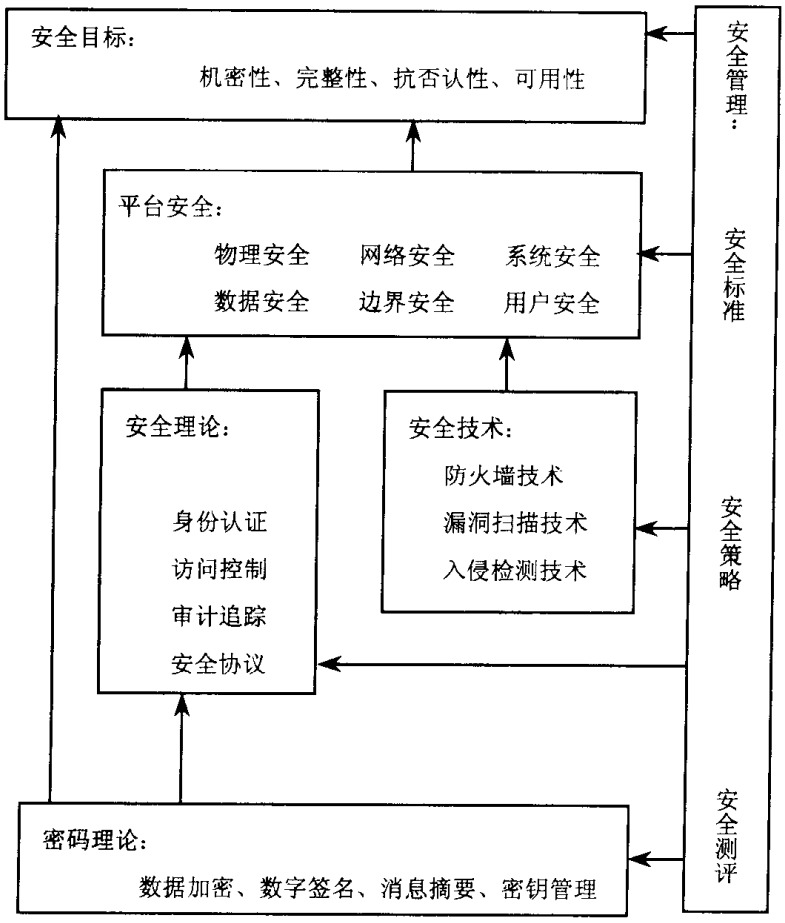


图 1.1　信息安全研究内容及相互关系

密码理论的研究重点是算法，包括数据加密算法、数字签名算法、消息摘要算法及相应的密钥管理协议等。这些算法提供两方面的服务：一方面，直接对信息进行运算，保护信息的安全特性，即通过加密变换保护信息的机密性，通过消息摘要变换检测信息的完整性，通过数字签名保护信息的抗否认性；另一方面，提供对身份认证和安全协议等理论的支持。

安全理论的研究重点是单机或网络环境下信息防护的基本理论，主要有访问控制(授权)、身份认证、审计追踪(这三者常称为 AAA，即 Authorization , Authentication, Audit)、安全协议等。这些研究成果为建设安全平台提供理论依据。

安全技术的研究重点是在单机或网络环境下信息防护的应用技术，目前主要有防火墙技术、入侵检测技术、漏洞扫描技术、防病毒技术等。其研究思路与具体的平台环境关系密切，研究成果直接为平台安全防护和检测提供技术依据。

平台安全是指保障承载信息产生、存储、传输和处理的平台的安全和可控。平台由网络设备、主机(服务器、终端)、通信网、数据库等有机组合而成，这些设备组成网络并形成特定的连接边界。平台安全不仅涉及物理安全、网络安全、系统安全、数据安全和边界安全，还包括用户行为的安全。

此外，安全管理也是很重要的。普遍认为，信息安全三分靠技术，七分靠管理，可见管理的分量。管理应该有统一的标准、可行的策略和必要的测评，因此，安全管理包括安全标准、安全策略、安全测评等。这些管理措施作用于安全理论和技术的各个方面。

1.2.1 信息安全基础研究

信息安全基础研究的主要内容包括密码学研究和网络信息安全基础理论研究。

1. 密码理论

密码理论(Cryptography)是信息安全的基础，信息安全的机密性、完整性和抗否认性都依赖于密码算法。通过加密可以保护信息的机密性；通过信息摘要可以检测信息的完整性；通过数字签名可以保护信息的抗否认性。加密变换需要密钥参与，因而密钥管理也是十分重要的研究内容。因此，密码学的主要研究内容是加密算法、消息摘要算法、数字签名算法以及密钥管理。

- 数据加密(Data Encryption)

数据加密算法是一种数学变换，在选定参数(密钥)的参与下，将信息从易于理解的明文加密为不易理解的密文，同时也可以将密文解密为明文 。加、解密时用的密钥可以相同，也可以不同。加、解密密钥相同的算法称为对称算法，典型的算法有 DES、AES 等；加、解密密钥不同的算法称为非对称算法，通常一个密钥公开，另一个密钥私藏，因而也

称为公钥算法。典型的算法有 RSA、ECC 等。

• 消息摘要(Message Digest)

消息摘要算法也是一种数学变换，通常是单向(不可逆)的变换，它将不定长度的信息变换为固定长度(如 16 字节)的摘要，信息的任何改变(即使是 1bit)也能引起摘要面目全非，因而可以通过消息摘要检测消息是否被篡改。典型的算法有 MD5、SHA 等。

• 数字签名(Digital Signature)

数字签名主要是消息摘要和非对称加密算法的组合应用。从原理上讲，通过私有密钥用非对称算法对信息本身进行加密，即可实现数字签名功能。用私钥加密只能用公钥解密，使得接受者可以解密信息，但无法生成用公钥解密的密文，从而证明此密文肯定是拥有加密私钥的用户所为，因而是不可否认的。实际实现时，由于非对称算法加/解密速度很慢，通常先计算消息摘要，再用非对称加密算法对消息摘要进行加密而获得数字签名。

• 密钥管理(Key Management)

密码算法是可以公开的，但密钥必须严格保护。如果非授权用户获得加密算法和密钥，则很容易破解或伪造密文，加密也就失去了意义。密钥管理研究就是研究密钥的产生、发放、存储、更换和销毁的算法和协议等。

2. 安全理论

• 身份认证(Authentication)

身份认证是指验证用户身份与其所声称的身份是否一致的过程。最常见的身份认证是口令认证。口令认证是在用户注册时记录下其用户名和口令，在用户请求服务时出示用户名和口令，通过比较其出示的用户名和口令与注册时记录下的是否一致来鉴别身份的真伪。复杂的身份认证则需要基于可信的第三方权威认证机构的保证和复杂的密码协议来支持，如基于证书认证中心(CA)和公钥算法的认证等。

身份认证研究的主要内容包括认证的特征(知识、推理、生物特征等)和认证的可信协议及模型。

• 授权和访问控制(Authorization and Access Control)

授权和访问控制是两个关系密切的概念，常常替换使用。它们的细微区别在于，授权侧重于强调用户拥有什么样的访问权限，这种权限是系统预先设定的，并不关心用户是否发起访问请求；而访问控制是对用户访问行为进行控制，它将用户的访问行为控制在授权允许的范围之内，因此，也可以说，访问控制是在用户发起访问请求时才起作用的。打个形象的比喻，授权是签发通行证，而访问控制则是卫兵，前者规定用户是否有权出入某个区域，而后者检查用户在出入时是否超越了禁区。

授权和访问控制研究的主要内容是授权策略、访问控制模型、大规模系统的快速访问

控制算法等。

• 审计追踪(Auditing and Tracing)

审计和追踪也是两个关系密切的概念，审计是指对用户的行为进行记录、分析和审查，以确认操作的历史行为。追踪则有追查的意思，通过审计结果追查用户的全程行踪。审计通常只在某个系统内进行，而追踪则需要对多个系统的审计结果综合分析。

审计追踪研究的主要内容是审计素材的记录方式、审计模型及追踪算法等。

• 安全协议 (Security Protocol)

安全协议指构建安全平台时所使用的与安全防护有关的协议，是各种安全技术和策略具体实现时共同遵循的规定，如安全传输协议、安全认证协议、安全保密协议等。典型的安全协议有网络层安全协议 IPSec、传输层安全协议 SSL、应用层安全电子商务协议 SET 等。

安全协议研究的主要内容是协议的内容和实现层次、协议自身的安全性、协议的互操作性等。

1.2.2 信息安全应用研究

信息安全的应用研究是针对信息在应用环境下的安全保护而提出的，是信息安全基础理论的具体应用，它包括安全技术研究和平台安全研究。

1. 安全技术

安全技术是对信息系统进行安全检查和防护的技术，包括防火墙技术、漏洞扫描技术、入侵检测技术、防病毒技术等。

• 防火墙技术(Firewall)

防火墙技术是一种安全隔离技术，它通过在两个安全策略不同的域之间设置防火墙来控制两个域之间的互访行为。隔离可以在网络层的多个层次上实现，目前应用较多的是网络层的包过滤技术和应用层的安全代理技术。包过滤技术通过检查信息流的信源和信宿地址等方式确认是否允许数据包通行；而安全代理则通过分析访问协议、代理访问请求来实现访问控制。

防火墙技术的主要研究内容包括防火墙的安全策略、实现模式、强度分析等。

• 漏洞扫描技术(Venearbility Scanning)

漏洞扫描是针对特定信息网络中存在的漏洞而进行的。信息网络中无论是主机还是网络设备都可能存在安全隐患，有些是系统设计时考虑不周而留下的，有些是系统建设时出现的。这些漏洞很容易被攻击，从而危及信息网络的安全。由于安全漏洞大多是非人为的、隐蔽的，因此，必须定期扫描检查、修补加固。操作系统经常出现的补丁模块就是为加固

发现的漏洞而开发的。由于漏洞扫描技术很难自动分析系统的设计和实现，因此很难发现未知漏洞。目前的漏洞扫描更多的是对已知漏洞检查定位。

漏洞扫描技术研究的主要内容包括漏洞的发现、特征分析以及定位、扫描方式和协议等。

• 入侵检测技术(Intrusion　Detection)

入侵检测是指通过对网络信息流提取和分析发现非正常访问模式的技术。目前主要有基于用户行为模式、系统行为模式和入侵特征的检测等。在实现时，可以只检测针对某主机的访问行为，也可以检测针对整个网络的访问行为，前者称为基于主机的入侵检测，后者称为基于网络的入侵检测。

入侵检测技术研究的主要内容包括信息流提取技术、入侵特征分析技术、入侵行为模式分析技术、入侵行为关联分析技术和高速信息流快速分析技术等。

• 防病毒技术(Anti-Virus)

病毒是一种具有传染性和破坏性的计算机程序。自从 1988 年出现 morris 蠕虫以来，计算机病毒成为家喻户晓的计算机安全隐患之一。随着网络的普及，计算机病毒的传播速度大大加快，破坏力也在增强，出现了智能病毒、远程控制病毒等。因此，研究和防范计算机病毒也是信息安全的一个重要方面。

病毒防范研究的重点包括病毒的作用机理、病毒的特征、病毒的传播模式、病毒的破坏力、病毒的扫描和清除等。

2. 平台安全

• 物理安全(Physical　Security)

物理安全是指保障信息网络物理设备不受物理损坏，或是损坏时能及时修复或替换。通常是针对设备的自然损坏、人为破坏或灾害损坏而提出的。目前常见的物理安全技术有备份技术(热备、冷备、同城、异地)、安全加固技术、安全设计技术等。如，保护 CA 认证中心，采用多层安全门和隔离墙，核心密码部件还要用防火、防盗柜保护。

• 网络安全(Network Security)

网络安全的目标是防止针对网络平台的实现和访问模式的安全威胁。在网络层，大量的安全问题与连接的建立方式、数据封装方式、目的地址和源地址等有关。如，网络协议在建立连接时要求三次应答，就导致了通过发起大量半连接而使网络阻塞的 SYN-flooding 攻击。

网络安全研究的内容主要有：安全隧道技术、网络协议脆弱性分析技术、安全路由技术、安全 IP 协议等。

• 系统安全(System Integrity)

系统安全是各种应用程序的基础。系统安全关心的主要问题是操作系统自身的安全性问题。信息的安全措施是建立在操作系统之上的，如果操作系统自身存在漏洞或隐蔽通道，就有可能使用户的访问绕过安全机制，使安全措施形同虚设。因此系统自身的安全性非常重要。现在商用操作系统自身的安全级别都不高，并且存在大量漏洞，研究系统安全就更为重要。

系统安全研究的主要内容包括安全操作系统的模型和实现、操作系统的安全加固、操作系统的脆弱性分析、操作系统与其他开发平台的安全关系等。

- 数据安全(Application Confidentiality)

数据是信息的直接表现形式，数据安全的重要性则不言而喻。数据安全主要关心数据在存储和应用过程中是否会被非授权用户有意破坏，或被授权用户无意破坏。数据通常以数据库或文件形式来存储，因此，数据安全主要是数据库或数据文件的安全问题。数据库系统或数据文件系统在管理数据时采取什么样的认证、授权、访问控制及审计等安全机制，达到什么安全等级，机密数据能否被加密存储等，都是数据的安全问题。

数据安全研究的主要内容有：安全数据库系统、数据存取安全策略和实现方式等。

- 用户安全(User Security)

用户安全问题有两层含义：一方面，合法用户的权限是否被正确授权，是否有越权，访问，是否只有授权用户才能使用系统资源。如，一个普通的合法用户可能被授予了管理员的身份和权限。另一方面，被授权的用户是否获得了必要的访问权限，是否存在多业务系统的授权矛盾等。

用户安全研究的主要内容包括用户账户管理、用户登录模式、用户权限管理、用户的角色管理等。

- 边界安全(Boundary Protection)

边界安全关心的是不同安全策略的区域边界连接的安全问题。不同的安全域具有不同的安全策略，将它们互连时应该满足什么样的安全策略，才不会破坏原来的安全策略，应该采取什么样的隔离和控制措施来限制互访，各种安全机制和措施互连后满足什么样的安全关系，这些问题都需要解决。

边界安全研究的主要内容是安全边界防护协议和模型、不同安全策略的连接关系问题、信息从高安全域流向低安全域的保密问题、安全边界的审计问题等。

1.2.3 信息安全管理研究

1. 安全策略研究

安全策略是安全系统设计、实施、管理和评估的依据。针对具体的信息和网络的安全，

应保护哪些资源，花费多大代价，采取什么措施，达到什么样的安全强度，都是由安全策略决定的。不同的国家和单位针对不同的应用都应制定相应的安全策略。如，什么级别的信息应该采取什么保护强度，针对不同级别的风险能承受什么样的代价，这些问题都应该制定策略。

安全策略研究的内容包括安全风险的评估、安全代价的评估、安全机制的制定以及安全措施的实施和管理等。

2．安全标准研究

安全标准研究是推进安全技术和产品标准化、规范化的基础。各国都非常重视安全标准的研究和制定。主要的标准化组织都推出了安全标准，著名的安全标准有可信计算机系统的评估准则(TCSEC)、通用准则 (CC)、 安全管理标准 ISO17799 等。

安全标准给出了技术发展、产品研制、安全测评、方案设计等多方面的技术依据。如 TCSEC 将安全划分为 7 个等级，并从技术、文档、保障等方面规定了各个安全等级的要求。

安全标准研究的主要内容包括安全等级划分标准、安全技术操作标准、安全体系结构标准、安全产品测评标准和安全工程实施标准等。

3．安全测评研究

安全测评是依据安全标准对安全产品或信息系统进行安全性评定。目前开展的测评有技术评测机构开展的技术测评，也有安全主管部门开展的市场准入测评。测评包括功能测评、性能测评、安全性测评、安全等级测评等。

安全测评研究的内容有测评模型、测评方法、测评工具、测评规程等。

1.3　信息安全的发展

信息安全已经历了漫长的发展过程。某种意义上说，从人类开始信息交流，就涉及信息的安全问题。从古代烽火传信到今天的通信网络，只要存在信息交流，就存在信息欺骗。信息安全的发展可以划分为经典信息安全阶段和现代信息安全阶段。经典信息安全阶段主要是通过对文字信息进行加密变换来保护信息；现代信息安全阶段则充分应用了计算机、通信等现代科技手段。

1.3.1　经典信息安全

在这一阶段，人们似乎更关注信息通信的保密性，通常采用一些简单的替代或置换来

保护信息。这些变换是密码学的雏形。这一阶段发展了很多密码算法，但基本的方法都是将字母编号后平移、旋转、置换、扩展等，如将字母编号平移产生了凯撒密码。其他的算法还有单表置换算法、Vigenere 算法、Wernam 算法、Hill 算法等。此外，还发展了密码分析和破译方法。

1.3.2 现代信息安全

随着数学、计算机和通信技术的发展，信息的处理能力和传输能力大大提高，传统的密码变换已不能满足信息化的要求。因此，信息安全加速发展，出现了现代密码理论、计算机安全和通信安全的新理论、新技术。

1. 现代密码理论

现代密码理论起源于 20 世纪 70 年代，但其理论基础可以追溯到 1949 年 Shanon 的论文“保密通信的理论基础”。现代密码理论充分结合了数学理论基础和计算机计算能力，提出了密码算法的框架结构。其标志性的成果首推 DES 算法和 RSA 算法。

数据加密标准(DES)是 1977 年美国国家标准局正式公布实施的。该算法在后来的 20 年一直作为国际最通用的分组加密算法在使用。虽然后来出现了其改进算法 3DES，但除了增加了 DES 加解密的运算次数和顺序外，没有本质的突破。DES 算法将数据按 64 比特分组进行加密，其密钥长度也是 64 比特，其中每 8 比特中有一位校验位，因此 DES 的有效密钥长度为 56 比特。DES 不仅仅是一个加密算法，它还代表了现代对称密码算法的一般性结构，后来很多算法都是在 DES 结构上发展起来的。

现代密码的另一个标志就是公钥密码体制的提出。Diffie 和 Hellman 在《密码学的新方向》中首次提出了非对称密码算法的思想。两年后 Rivest、Shamir 和 Adleman 提出的 RSA 算法体现了公钥算法的思想。RSA 算法至今仍然是公钥密码算法的典型代表。

目前，密码学的研究依然炙手可热，美国花巨资历时 3 年遴选了代替 DES 算法的 AES 算法，欧洲也正在制定新的欧洲密码体制。在公钥体制方面，椭圆曲线算法 ECC 是目前研究的热点。

2. 计算机安全

第一台计算机出现于 1946 年，那时人们都在关注提高计算机的功能和性能，还没有意识到计算机安全的重要性，只考虑到物理安全问题。随着多用户、多进程计算机的出现，众多用户在同一台计算机上工作，产生了计算机账户管理和资源分配等需求，因此出现了身份认证和访问控制，开始在操作系统中设置专门的用户口令文件和用户账户文件，并在用户登录时引发身份认证进程。计算机还为不同的用户设置专用目录和公用目录，根据预

先分配用户的权限来控制其访问的范围。20 世纪 70 年代初出现的 Unix 操作系统就具备了这样的安全机制。

到了 1988 年，出现了第一个计算机病毒 Morris 蠕虫，防病毒开始成为计算机安全的主要任务之一。至今，防止网络环境下的病毒扩散仍然是计算机安全的一项重要内容。

3. 网络安全

计算机网络的发展向信息安全提出了新的挑战，尤其是 Internet 的出现使信息安全在学术界、产业界和主管部门都掀开了新的一页。从 20 世纪 90 年代开始，计算机网络安全研究进一步加强。尽管信息安全的学科建设还不完善，但信息安全作为一个独立的交叉学科，其地位越来越重要。

首先，安全通信协议的研究成果显著，出现了 IPSec、SSL、SHTTP、SET 等安全协议；

其次，安全技术研究也开始加强，出现了漏洞扫描技术、入侵检测技术、防火墙技术、VPN 技术等，并开发出相应的产品；

第三，安全操作系统的需求也越来越明显，出现商用 C2、B1 级操作系统；

第四，安全标准的制定也开始加速，出现了 CC、ISO17799 等安全标准。

4. 信息保障

目前，在国际研究前沿，信息安全已上升到信息保障的高度，提出了计算环境安全、通信网安全、边界安全及安全支撑环境和条件的概念，并开始研究信息网络的生存性等课题。

总之，信息安全还没有形成完整的学科概念，但其发展速度正在加快，研究人员正在增加，信息安全作为独立产业的形态开始显现，主管部门也在加大管理力度，并加紧制定信息安全法律、法规。信息安全学科正应时代需要发展和完善。

1.4　本书内容安排

从以上论述中不难看出，信息安全学科不仅内容庞杂，而且发展迅速，要在一本教材中悉数介绍，在一个学期的教学中全部讲授，是不现实的。因此本书定位于向学生介绍信息安全的基本原理和技术，并结合实例讲解原理和技术的应用，以帮助学生学以致用。课程的内容比例大致是：基本理论和技术占 70%，应用、标准、策略和管理占 30%。

习 题 一

1. 信息安全的目标是什么？
2. 简述信息安全的学科体系。
3. 信息安全的理论、技术和应用是什么关系？如何体现？

第二章　密码学概论

2.1　密码学的基本概念

密码学是研究如何实现秘密通信的科学，包括两个分支，即密码编码学和密码分析学。密码编码学是对信息进行编码实现信息保密性的科学；而密码分析学是研究、分析、破译密码的科学。

对需要保密的消息进行编码的过程称为加密，编码的规则称为加密算法。需要加密的消息称为明文，明文加密后的形式称为密文。将密文恢复出明文的过程称为解密，解密的规则称为解密算法。加密算法和解密算法通常在一对密钥控制下进行，分别称为加密密钥和解密密钥。

一个密码系统(体制)包括所有可能的明文、密文、密钥、加密算法和解密算法。密码系统的安全性基于密钥而非加密和解密算法的细节，这意味着算法可以公开，甚至可以当成一个标准加以公布。

密码系统从原理上可分为两大类，即单密钥系统和双密钥系统。单密钥系统又称为对称密码系统或秘密密钥密码系统，单密钥系统的加密密钥和解密密钥或者相同，或者实质上等同，即易于从一个密钥得出另一个，如图 2.1 所示。

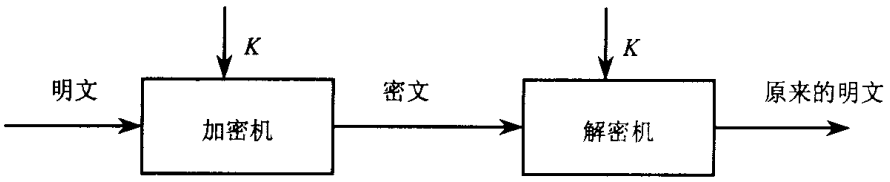


图 2.1　单钥密码的加密、解密过程

对明文的加密有两种形式，一种是对明文按字符逐位加密，称之为流密码或序列密码；另一种是先对明文消息分组，再逐组加密，称之为分组密码。

双密钥系统又称为非对称密码系统或公开密钥密码系统。双密钥系统有两个密钥，一个是公开的，用 K_1 表示，谁都可以使用；另一个是私人密钥，用 K_2 表示，只由采用此系统的人自己掌握。从公开的密钥推不出私人密钥，如图 2.2 所示。

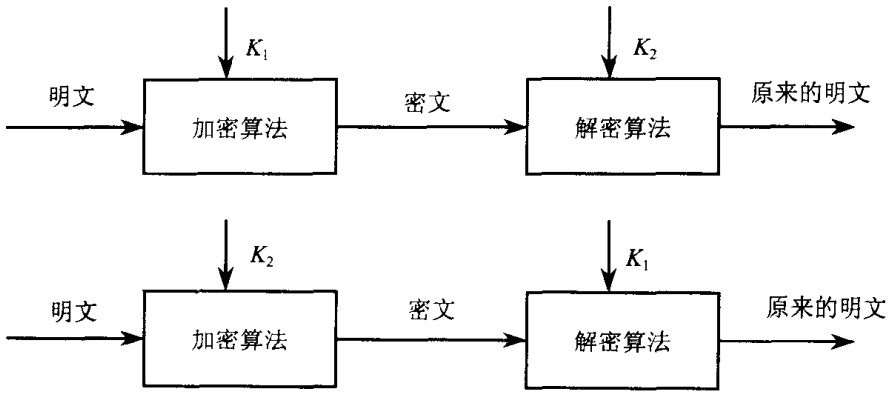


图 2.2 双钥密码的加密、解密过程

双钥密码系统的主要特点是将加密和解密密钥分开。即用公开的密钥 K_1 加密消息，发送给持有相应私人密钥 K_2 的人，只有持有私人密钥 K_2 的人才能解密；而用私人密钥 K_2 加密的消息，任何人都可以用公开的密钥 K_1 解密，此时说明消息来自持有私人密钥的人。前者可以实现公共网络的保密通信，后者则可以实现对消息进行数字签名。

2.2 经典密码体制

经典密码体制采用手工或机械操作实现加/解密，相对简单。回顾和研究这些密码的原理和技术，对于理解、设计和分析现代密码仍然具有借鉴的价值。经典密码大体上可分为三类：单表代换密码、多表代换密码和多字母代换密码。

2.2.1 单表代换密码

在经典密码体制中，最典型的是替换密码，其原理可以用一个例子来说明：

将字母 a，b，c，d，…，w，x，y，z 的自然顺序保持不变，但使之与 D，E，F，G，…，Z，A，B，C 分别对应(即将字母表中的每个字母用其后的第三个字母进行循环替换)。若明文为 student，则对应的密文为 VWXGHQW(此时密钥为 3)。这就是凯撒(Kaesar)密码，也称为移位代换密码。凯撒密码仅有 26 个可能的密钥，非常不安全。如果允许字母表中的字母用任意字母进行替换，即上述密文能够是 26 个字母的任意排列，则将有 26！或多于 4×10^{26} 种可能的密钥。这样的密钥空间甚至对计算机来说穷举搜索密钥也是不现实的。

这里有一个由加密函数组成的“随机”置换：

明文：a b c d e f g h i j k l m n o p q r s t u v w x y z

密文：X N Y A H P O G Z Q W B T S F L R C V M U E K J D I

解密函数是如下的一个逆置换：

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

d l r y v o h e z x w p t b g f j q n m u s k a c i

作为一个练习，读者可以试着利用解密函数解密下列密文：

MGZVY ZLGHC MHJMY XSSFM NHAHY CDLMHA

由于英文字母中各字母出现的频度早已有人进行过统计，所以根据字母频度表可以很容易对替换密码进行破译。替换密码是对所有的明文字母都用一个固定的代换进行加密，因而称为单表代换密码。为了抗击字母频度分析，随后产生了多表代换密码和多字母代换密码。

2.2.2　多表代换密码

多表代换密码中最著名的一种密码称为维吉尼亚(Vigenère)密码。这是一种以移位代换为基础的周期代换密码，m 个移位代换表由 m 个字母组成的密钥字确定(这里假设密钥字中 m 个字母不同，如果有相同的，则代换表的个数是密钥字中不同字母的个数)。如果密钥字为 deceptive，明文 wearediscoveredsaveyourself 被加密为：

明文：w e a r e d i s c o v e r e d s a v e y o u r s e l f

密钥：d e c e p t i v e d e c e p t i v e d e c e p t i v e

密文：Z I C V T W Q N G R Z G V T W A V Z H C Q Y G L M G J

其中，密钥字母 a，b，…，y，z 对应数字 0，1，…，24，25。密钥字母 d 对应数字 3，因而明文字母 w 在密钥字母 d 的作用下向后移位 3，得到密文字母 Z；明文字母 e 在密钥字母 e 的作用下向后移位 4，得到密文字母 i，以此类推。解密时，密文字母在密钥字母的作用下向前移位。

在维吉尼亚密码中，如果密钥字的长度是 m，明文中的一个字母能够映成这 m 个可能的字母中的一个。容易看出维吉尼亚密码中长度为 m 的可能密钥字的个数是 26^m，甚至对于一个较小的 m 值，如 m=5，密钥空间超过了 1.1×10^7，这个空间已经足以阻止手工穷举密钥搜索。

为方便记忆，维吉尼亚密码的密钥字常常取于英文中的一个单词、一个句子或一段文章。因此，维吉尼亚密码的明文和密钥字母频率分布相同，仍然能够用统计技术进行分析。要抗击这样的密码分析，只有选择与明文长度相同并与之没有统计关系的密钥内容。1918 年美国电报电话公司的 G．W．Vernam 提出这样的密码系统：明文英文字母编成 5 比特二元数字，称之为五单元波多代码(Baudot Code)，选择随机二元数字流作为密钥，加密通过执行明文和密钥的逐位异或操作，产生密文，可以简单地表示为：

$$C_i=p_i \oplus k_i$$

其中，p_i表示明文的第 i 个二元数字，k_i表示密钥的第 i 个二元数字，C_i表示密文的第 i 个二元数字，$\oplus$ 表示异或操作。解密仅需执行相同的逐位异或操作：

$$p_i=C_i \oplus k_i$$

Vernam 密码系统的密钥若不重复使用，就能得到一次一密密码。若密钥有重复，尽管使用长密钥增加了密码分析的难度，但只要有了足够的密文，使用已知的或可能的明文序列，或二者相结合也能够破译。

2.2.3 多字母代换密码

前面介绍的密码都是以单个字母作为代换的对象，对多于一个字母进行代换，就是多字母代换密码。它的优点是容易将字母的频度隐蔽，从而抗击统计分析。首先介绍 Hill 密码，它是数学家 Lester Hill 于 1929 年研制的。虽然这类密码由于加密操作复杂而未能广泛应用，但仍在很大程度上推进了经典密码学的研究。

Hill 密码将明文分成 m 个字母一组的明文组，若最后一组不够 m 个字母就用字母补足，每组用 m 个密文字母代换，这种代换由 m 个线性方程决定，其中字母 a，b，…，y，z 分别用数字 0，1，…，24，25 表示。若 $m=3$，该系统可以描述如下：

$$C_1=(k_{11}p_1+k_{12}p_2+k_{13}p_3)\bmod 26$$
$$C_2=(k_{21}p_1+k_{22}p_2+k_{23}p_3)\bmod 26$$
$$C_3=(k_{31}p_1+k_{32}p_2+k_{33}p_3)\bmod 26$$

可用列向量和矩阵表示为

$$\begin{pmatrix}C_1\\C_2\\C_3\end{pmatrix}=\begin{pmatrix}k_{11}&k_{12}&k_{13}\\k_{21}&k_{22}&k_{23}\\k_{31}&k_{32}&k_{33}\end{pmatrix}\begin{pmatrix}p_1\\p_2\\p_3\end{pmatrix}$$

或

$$\boldsymbol{C}=\boldsymbol{KP}$$

其中，$\boldsymbol{C}$ 和 $\boldsymbol{P}$ 分别是密文和明文向量，$\boldsymbol{K}$ 是密钥矩阵，操作要执行模 26 运算。例如，用密钥

$$\boldsymbol{K}=\begin{pmatrix}11&3\\8&7\end{pmatrix}$$

来加密明文 july。将明文分成两个组 ju 和 ly，分别为(9，20)和(11，24)，计算如下：

$$\begin{pmatrix}11&3\\8&7\end{pmatrix}\begin{pmatrix}9\\20\end{pmatrix}=\begin{pmatrix}99+60\\72+140\end{pmatrix}=\begin{pmatrix}3\\4\end{pmatrix}$$

$$\begin{pmatrix}11 & 3\\ 8 & 7\end{pmatrix}\begin{pmatrix}11\\ 24\end{pmatrix}=\begin{pmatrix}121+72\\ 88+168\end{pmatrix}=\begin{pmatrix}11\\ 22\end{pmatrix}$$

因此，july 的加密结果为 DELW。为了解密，必须先计算密钥矩阵 $\boldsymbol{K}$ 的逆矩阵

$$\boldsymbol{K}^{-1}=\begin{pmatrix}7 & 23\\ 18 & 11\end{pmatrix}$$

然后计算

$$\begin{pmatrix}7 & 23\\ 18 & 11\end{pmatrix}\begin{pmatrix}3\\ 4\end{pmatrix}=\begin{pmatrix}9\\ 20\end{pmatrix}$$

$$\begin{pmatrix}7 & 23\\ 18 & 11\end{pmatrix}\begin{pmatrix}11\\ 22\end{pmatrix}=\begin{pmatrix}11\\ 24\end{pmatrix}$$

最后，得到正确的明文 july。

当 Hill 密码的密钥矩阵为一置换矩阵时，相应的密码就是置换密码，也称为换位密码。它对明文 m 长字母组中的字母位置重新排列，而不改变明文字母。在置换密码系统中，使用字母比使用数字更方便，密钥使用置换而不使用矩阵。如 m=6，用密钥置换

$$\boldsymbol{K}=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 3 & 5 & 1 & 6 & 4 & 2\end{pmatrix}$$

对明文 shesellsseashellsbytheseashore 进行加密。首先将明文分成 6 个字母长的明文组

shesel | lsseas | hellsb | ythese | ashore

然后将每个 6 字母长的明文组按密钥置换 $\boldsymbol{K}$ 重新排列如下：

EESLSH | SALSES | LSHBLE | HSYEET | HRAEOS

所以，密文是 EESLSHSALSESLSHBLEHSYEETHRAEOS。密文能利用密钥置换的逆置换

$$\boldsymbol{K}^{-1}=\begin{pmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 3 & 6 & 1 & 5 & 2 & 4\end{pmatrix}$$

以类似的方式来解密。

2.2.4　转轮密码

使用密码机可以使前面介绍的密码系统更复杂、更安全，这些机器也可加速密码系统的加/解密过程，同时提供大量可选择的密钥。转轮密码是一组转轮或接线编码轮所组成的机器，用于实现长周期的多表代换密码，它是经典密码最杰出的代表，曾经被广泛应用于军事和外交保密通信。最有名的两类密码机是 Enigma 和 Hagelin。Enigma 密码机由德国 Arthur Scherbius 发明，在二次世界大战中，曾经装备于德军。Hagelin 密码机由瑞典 Boris Caesar Wilhelm Hagelin 发明，二次大战中，Hagelin C-36 曾经装备于法军；Hagelin C-40，

即 M-209 转换机，曾经装备于美军，一直沿用到 50 年代。另外，在二次世界大战中，美国的 SIGABA 和日本的 RED 和 PURPLE 都是转轮密码机。

2.3 密 码 分 析

对通信双方而言，通信的一方将信息加密后发送给另一方，是为了使攻击者即使得到密文也无法读懂。对于攻击者来说，在不知道密钥的情况下，要想读懂密文，就要根据他的知识以及掌握的情报来进行密码分析。

密码分析是研究密钥未知的情况下恢复明文的科学。通常假设攻击者知道正在使用的密码体制，称为 Kerckhoff 原则。当然，如果分析者不知道正在使用的密码体制，分析起来将更加困难。成功的密码分析可能直接恢复明文或密钥，也可能找出保密系统的弱点来恢复明文或密钥。根据攻击者具有的知识和掌握的情报，可以将密码分析分为以下几种类型：

(1) 只有密文的攻击。攻击者有一些密文，它们是使用同一加密算法和同一密钥加密的。

(2) 已知明文的攻击。攻击者不仅得到一些密文，而且能得到这些密文对应的明文。

(3) 选择明文的攻击。攻击者不仅得到一些密文和对应明文，而且能选择用于加密的明文。

(4) 选择密文的攻击。攻击者可以选择不同的密文来解密，并能得到解密后的明文。

在每种情况下，攻击者的目标是确定正在使用的密钥或待破译密文所对应的明文。上述几种攻击方式是以强度递增排列的，只有密文的攻击是最容易防护的攻击。然而，目前最常见的是已知明文和选择明文的攻击。事实上，攻击者欲得到一些明、密文对或加密一些选择好的明文并不困难，攻击者可以知道很多信息具有标准的头部和尾部。一般来说，密码系统应该经得起已知明文的攻击。

密码系统的安全依赖于破译该系统的困难程度。如果破译一个密码系统的费用超过了被加密数据本身的价值，或者破译所需的时间超过了被加密数据所需保密的时间，或者由一个密码系统加密的数据少于破译该算法所需的数据量，就可以认为这个系统是安全的。

如果攻击者无论得到多少密文，都没有足够的信息去恢复明文，那么该密码系统就是无条件安全的。在理论上，只有一次一密的系统才能真正实现这一点。除此之外的系统都至少可以使用只有密文的攻击来破译。

设计密码系统必须至少满足下面的一个准则：

(1) 破译该密码的成本超过被加密信息的价值；

(2) 破译该密码的时间超过被加密信息的生命周期。

如果一个密码系统能够满足上述准则，就认为该系统在计算上是安全的。然而困难之处就在于难以估计成功破译密码的努力有多大。只能使设计的密码系统抗击已知的攻击，尤其要能抗击穷举密钥攻击，即强力攻击。

穷举密钥攻击方法是对截获的密文用所有可能的密钥解密，直到得到有意义的明文为止。从理论上来说，除了一次一密密码系统，其他任何密码系统都可以用穷举密钥攻击法破解，但实际上，任何一个实用的密码设计都会使这一方法在计算上失去可行性。

密码分析方法可以分为确定性和统计性两类。确定性分析法是利用若干已知数学关系式表示出所求的未知量，这种关系式是从加密和解密算法得来的；统计分析法则是利用明文的已知统计规律进行破译的方法。

经典密码分析的许多技术利用了英文语言的统计特性。英文中字母的相对标准概率分布如下：

A	0.082	B	0.015	C	0.028	D	0.043	E	0.127
F	0.022	G	0.020	H	0.061	I	0.070	J	0.002
K	0.008	L	0.040	M	0.024	N	0.067	O	0.075
P	0.019	Q	0.001	R	0.060	S	0.063	T	0.091
U	0.028	V	0.010	W	0.023	X	0.001	Y	0.020
Z	0.001								

在密码分析中，考虑英文中两字母组或三字母组是有用的。30 个最常见的两字母组是 th、he、in、er、an、re、ed、on、es、st、en、at、to、nt、ha、nd、ou、ea、ng、as、or、ti、is、et、it、ar、te、se、hi 和 of。12 个最常见的三字母组是 the、ing、and、her、ere、ent、tha、nth、was、eth、for 和 dth。上述字母组是按出现次数递减排列的。

替换密码虽然能抗击穷举密钥攻击，但密码分析者可以利用英文语言的统计规律来破译。为了说明如何利用英文的统计数据来破译替换密码，来看如下例子。假设从替换密码中获得的密文是：

YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJNDIFEFMDZCDMQ
ZKCEYFCJMYRNCWJCSZREXCHZUNMXZNZUCDRJXYYSMRTMEYIFZWDYVZVYFZ
UMRZCRWNZDZJJXZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR

作为破译的第一步，统计密文的频数如下：

A	0	B	1	C	15	D	13	E	7	F	11	G	1
H	4	I	5	J	11	K	1	L	0	M	16	N	9
O	0	P	1	Q	4	R	10	S	3	T	2	U	5
V	5	W	8	X	6	Y	10	Z	20				

由于 Z 出现的频数最高，可以假设密文字母 Z 对应的明文是 e。出现 10 次以上的密文字母是 C、D、F、J、M、R、Y，可能对应的明文字母是 t、a、o、i、n、s、h、r，但实际的频数还不能告诉我们它们之间的对应关系。

此时，需要考虑－Z 和 Z－的两字母组合。这种类型的最一般的两字母是：DZ 和 ZW 出现 4 次；NZ 和 ZU 出现 3 次；RZ、HZ、XZ、FZ、ZR、ZV、ZC、ZD 和 ZJ 出现 2 次。由于 ZW 出现 4 次而 WZ 一次都没有出现，同时 W 出现次数较少(8 次)，所以可以假设 W 对应的明文字母是 d。由于 DZ 出现 4 次而 ZD 出现 2 次，同时 D 出现次数较多(13 次)，所以可以假设 D 对应的明文字母是 r、s、t 中之一。

现在已假设 Z、W 对应的明文字母是 e、d。由于 RW 出现 2 次而 WR 没有出现，同时 R 出现次数较多(10 次)，所以我们可以假设 R 对应的明文字母是 n。注意到 NZ 出现 3 次而 ZN 没有出现，同时 N 出现次数是 9 次，所以我们可以假设 N 对应的明文字母是 h。如果这个假设成立，则密文 RZCRWNZ 可以译为明文 ne-ndhe，因此我们可以试着将 C 译为 a。

注意到密文段 RNM 解密为 nh-，这说明 h-是一个词的开头，所以 M 可能是一个元音。由于 M 出现频率是次高的，而元音 a、e 已被使用，所以 M 对应的明文字母可能是 i 或 o。因为 ai 是比 ao 出现次数更多的两字母组，所以密文字母段 CM 建议先试着将 M 译为 i。于是有

－－－－－iend－－－－－a－i－e－a－inedhi－e－－－－－－a－－

YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDD

－i－h－－－－－i－ea－i－e－a－－－a－i－nhad－a－en－－a－

UMJNDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCH

e－hi－ehe－a－n－－－－－in－i－－－－ed－－－e－－－e－ine

ZUNMXZNZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZ

andhe－e－－－ed－a－－inhi－－hai－－a－e－i－－ed－－－

CRWNZDZJJXZWGCHSMRNMDHNCMFQCHZJMXJZWIEJ

－－a－d－－he－－n

YUCFWDJNZDIR

由于密文中 MD 出现 4 次而 DM 出现一次，结合前面假设 D 对应的明文字母是 r、s、t 中之一，所以 D 对应的明文字母最可能的是 s。剩下的密文字母中频率最高的三个字母是 F、J、Y，它们可能解密为 r、t、o。首先考虑 o 对应的密文字母，Y 似乎是最可能的，否则从 CFM 或 CJM 中得到长串元音 aoi。因此，假设 Y 解密为 o。密文段 HNCMF 可能是 chair 的加密，即 H、F 可能分别解密为 c、r。这样 J 可能解密为 t。现在有

o－r－riend－ro－－arise－a－inedhise－－t－－－ass

Y I F Q F M Z R W Q F Y V E C F M D Z P C V M R Z W N M D Z V E J B T X C D D
— i t h s — r — r i s e a s i — e — a — o r a t i o n h a d t a — e n — — a c
U M J N D I F E F M D Z C D M Q Z K C E Y F C J M Y R N C W J C S Z R E X C H
e — h i — e h e — a s n t — o o — i n — i — o — r e d s o — e — o r e — i n e
Z U N M X Z N Z U C D R J X Y Y S M R T M E Y I F Z W D Y V Z V Y F Z U M R Z
a n d h e s e t t — e d — a c — i n h i s c h a i r — a c e t i — t e d — — t
C R W N Z D Z J J X Z W G C H S M R N M D H N C M F Q C H Z J M X J Z W I E J
o — a r d s t h e s — n
Y U C F W D J N Z D I R

最终容易确定出明文和密钥，完整的解密如下：

Our friend from Paris examined his empty glass with surprise, as if evaporation had taken place while he wasn't looking. I poured some more wine and he settled back in his chair, face tilted up towards the sun.

从以上的分析可以看出，单表代换根本经不起统计密码分析；多表代换使得明文的统计特性在密文中被弱化，破译起来更困难，但也使得密文的统计特性明显不同于明文。分析多表代换密码的关键是确定识别多表代换密码的参数，而 Friedman 引入的重合指数是识别多表代换密码的有效方法，Kasiski 提出的重码分析法对确定代换表的个数十分有效。有了这些，剩下的工作就相对容易了。

一般来说，对于多字母代换密码，很难用只有密文攻击的方法破译，但采用已知明文攻击就很容易破译。

习　题　二

1. 概念解释：分组密码、流密码、对称密码、非对称密码。

2. 设 a~z 的编号为 1~26，空格为 27，采用凯撒(Kaesar)密码算法为 $C=k_1M+k_2$，取 $k_1=3$, $k_2=5$, $M=$ Peking University, 计算密文 C。

3. 设 a~z 的编号为 1~26，空格为 27，采用 Vigenere 方案，密钥长度与消息相同，给出密文：

ANKYODKYUREPFJBYOJDSPLREYIUNOFDOIUERFPLUYTS

分别找出对应下列两组明文的密钥：

a) MR MUSTARD WITH THE CANDLESTICK IN THE HALL

b) MISS　SCARLET WITH　THE KNIFE IN THE LIBRARY

4. 构造一个用选择明文破译 Hill 算法的例子。

第三章　对称密码体制

本章先介绍对称分组密码的一般原理，然后介绍典型分组密码算法 DES 和 AES，最后简要介绍流密码。

3.1　分组密码原理

对称密码体制根据对明文加密方式的不同而分为分组密码和流密码。前者按一定长度(如 64 字节、128 字节等)对明文进行分组，然后以组为单位进行加/解密；后者则不进行分组，而是按位进行加/解密。

分组密码系统对不同的组采用同样的密钥 k 来进行加/解密。设密文组为 $y=y_1y_2\cdots y_m$，则对明文组 $x=x_1x_2\cdots x_m$ 用密钥 k 加密可得到 $y=e_k(x_1)e_k(x_2)\cdots e_k(x_m)$。

流密码的基本思想是利用密钥 k 产生一个密钥流 $z=z_0z_1\cdots$，并使用如下规则加密明文串 $x=x_0x_1x_2\cdots$，$y=y_0y_1y_2\cdots=e_{z_0}(x_0)e_{z_1}(x_1)e_{z_2}(x_2)\cdots$。密钥流由密钥流发生器 f 产生：$z_i=f(k,\sigma_i)$，这里的 σ_i 是加密器中的记忆元件(存储器)在时刻 i 的状态，f 是由密钥 k 和 σ_i 产生的函数。

分组密码与流密码的区别就在于记忆性(如图 3.1)。流密码的滚动密钥 $z_0=f(k,\sigma_0)$ 由函数 f、密钥 k 和指定的初态 σ_0 完全确定。此后，由于输入加密器的明文可能影响加密器中内部记忆元件的存储状态，因而 $\sigma_i(i>0)$ 可能依赖于 $k,\sigma_0,x_0,x_1,\cdots,x_{i-1}$ 等参数。

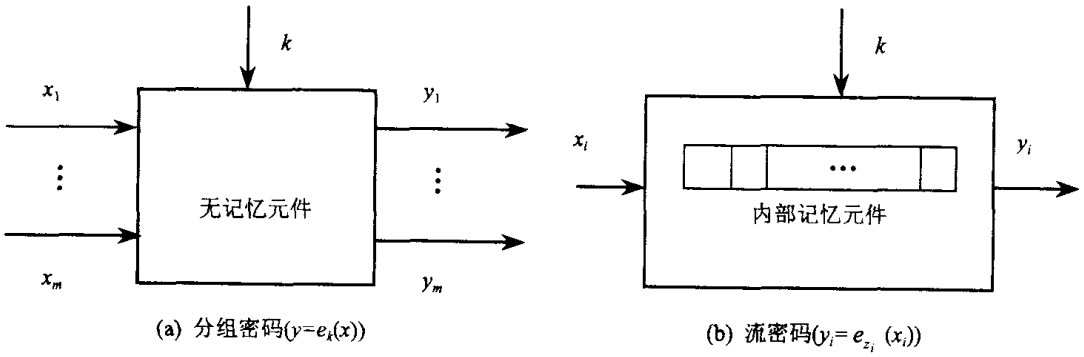


图 3.1　分组密码和流密码加密

3.1.1　分组密码设计原理

分组密码是将明文消息编码表示后的数字(简称明文数字)序列 $x_0, x_1, x_2, \cdots$，划分成长度为 n 的组 $x=(x_0, x_1, \cdots, x_{n-1})$ (可看成长度为 n 的矢量)，每组分别在密钥 $k=(k_0, k_1, \cdots, k_{m-1})$ 的控制下变换成等长的输出数字(简称密文数字)序列 $y=(y_0, y_1, \cdots, y_{n-1})$，其加密函数是 $E: V_n \times K \to V_n$，V_n 是 n 维矢量空间，K 为密钥空间，如图 3.2 所示。在相同的密钥 k 的控制下，加密函数可以看成是函数 $E(\circ, k): V_n \to V_n$。这实质上是对字长为 n 的数字序列进行置换。在二元的情况下，x 和 y 都是二元序列，共有 2^n 个不同的明文分组。为了使加密运算可逆，从而解密运算可行，每个明文分组应对应惟一一个密文分组，即置换 $E(\circ, k)$ 是可逆的。众所周知，V_n 上这样的置换共有 $2^n!$ 个，因而密钥个数最多为 $2^n!$ 个。实际应用中的许多分组密码，如 DES、IDEA 等，所用的置换只不过是上述置换集中一个很小的子集。

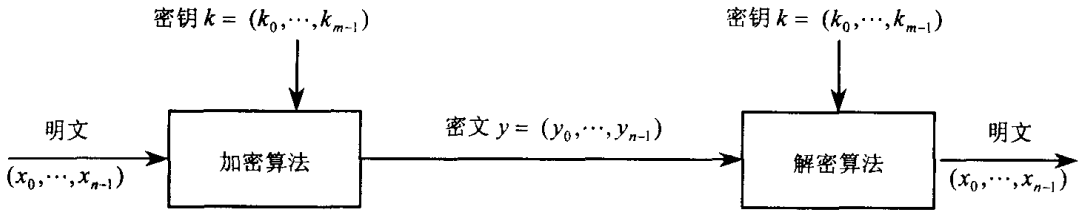


图 3.2　分组密码模型

分组密码设计就是要找到一种算法，能在密钥的控制下，从一个足够大、足够好的置换子集中简单迅速地选出一个置换，对当前输入的明文数字组进行加密变换。因此，设计的算法应满足下述安全性和软/硬件实现的要求：

(1) 分组长度应足够大，使得不同明文分组的个数 2^n 足够大，以防止明文被穷举法攻击。如 $n=64$，则在进行攻击时用 2^{32} 个分组密文成功的概率为 1/2，同时需要 $2^{32} \cdot 64$ bit $=2^{15}$ MB 的存储空间，因而采取穷举法攻击是不可行的。新的算法标准一般要求 $n=128$。

(2) 密钥空间应足够大，尽可能消除弱密钥，从而使所有密钥同等概率，以防穷举密钥攻击。同时，密钥不能太长，以利于密钥管理。DES 采用 56 比特有效密钥，现在看来显然不够长。今后一段时间内，128 比特密钥应该是足够安全的。

(3) 由密钥确定的算法要足够复杂，充分实现明文与密钥的扩散和混淆，没有简单关系可循，要能抵抗各种已知的攻击，如差分攻击和线性攻击等；另外，还要求有较高的非线性阶数。

(4) 软件实现的要求：尽量使用适合编程的子块和简单的运算。密码运算在子块上进行，因此要求子块的长度能适应软件编程，如 8、16、32 比特等。应尽量避免按比特置换，

在子块上进行的密码运算应尽量采用易于软件实现的运算。最好是使用处理器的基本运算，如加法、乘法、移位等。

(5) 硬件实现的要求：加密和解密应具有相似性，即加密和解密过程的不同应仅仅在于密钥的使用方式上，以便采用同样的器件来实现加密和解密，以节省费用和体积。尽量采用标准的组件结构，以便能在超大规模集成电路中实现。

需要指出的是，混淆和扩散是 Shannon 提出的设计密码系统的两种基本方法。Shannon 认为，在理想密码系统中，密文的所有统计特性都应与所使用的密钥独立，然而实用的密码系统都很难达到这个目标。在扩散中，要求明文的统计结构扩散消失到密文的统计特性中。要做到这一点，必须让明文的每个比特影响到密文许多比特的取值，即每个密文比特被许多明文比特影响。所有的分组密码都包含明文分组到密文分组的代换，而具体代换依赖于密钥。混淆则是试图使得密文的统计特性与密钥的取值之间的关系尽量复杂。扩散和混淆的目的都是为了挫败推测出密钥的尝试，从而抗击统计分析。

迭代密码是实现混淆和扩散原则的一种有效方法。合理选择的轮函数经过若干次迭代后能够提供必要的混淆和扩散。所以本书中讨论的分组密码只是迭代分组密码。分组密码由加密算法、解密算法和密钥扩展算法三部分组成。解密算法是加密算法的逆过程，由加密算法惟一确定，因而主要讨论加密算法和密钥扩展算法。

3.1.2 分组密码的一般结构

分组密码的结构一般可以分为两种：Feistel 网络结构和 SP 网络结构。

1. Feistel 网络结构

Feistel 网络是由 Horst Feistel 在设计 Lucifer 分组密码时发明的，并因为被 DES 采用而流行。许多分组密码采用了 Feistel 网络，如 FEAL、Blowfish、RC5 等。

一个分组长度为 n(偶数)比特的 l 轮 Feistel 网络的加密过程如下。给定明文 P，将 P 分成左右长度相等的两半分别记为 L_0 和 R_0，从而 $P=L_0R_0$。进行 l 轮完全类似的迭代运算后，再将左、右长度相等的两半合并产生密文分组。可以根据下列规则计算 L_iR_i，$1\leqslant i\leqslant l$：

$$L_i=R_{i-1},\quad R_i=L_{i-1}\oplus F(R_{i-1},K_i)$$

在进行 l 轮迭代运算后，将 L_l 和 R_l 再进行交换，最后输出的密文分组是 $C=R_lL_l$；其中 $\oplus$ 表示两个比特串的“异或”，$F:V_{n/2}\times V_N\rightarrow V_{n/2}$ 是轮函数(后面将详细介绍)，K_i 是由种子密钥 K 生成的子密钥，N 为子密钥的长度，如图 3.3 所示。

每轮中都对数据的左半部分进行代换，方法是先对右边一半数据应用轮函数，然后将输出的数据与左半部分进行异或。轮函数 F 的结构一直不变，但作为控制参数的每轮子密

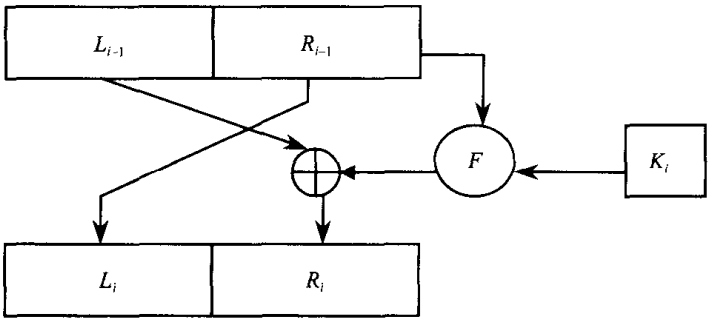


图 3.3　一轮 Feistel 网络加密过程

钥一般不同。全部代换完成后，再做一次置换操作交换左、右两半数据。

Feistel 网络的安全性和软、硬件实现速度取决于下列参数：

(1) 分组长度：分组长度越大，则安全性越高(其他条件相同时)，但同时加、解密速度也越慢。64 比特的分组目前也可以使用，但最好采用 128 比特。

(2) 密钥长度：密钥长度越大，则安全性越高(其他条件相同时)，但同时加、解密速度也越慢。64 比特密钥现在已不够安全，128 比特是一个折中的选择。

(3) 循环次数：Feistel 网络结构的一个特点是循环次数越多，则安全性越高，通常选择 16 次。

(4) 子密钥算法：子密钥算法越复杂，安全性越高。

(5) 轮函数：轮函数越复杂，安全性越高。

(6) 快速的软件实现：有时候客观条件不允许用硬件实现，算法被镶嵌在应用程序中。此时，算法的执行速度是关键。

(7) 算法简洁：通常算法越复杂越好，但采用简洁算法也不无裨益。如果算法比较容易解释清楚，就能通过分析算法而获知算法抗各种攻击的能力，将有助于设计高强度的算法。

上述这些要求并不能保证完全一致，有时甚至是互相制约的，应根据具体应用情况折中处理。

现在简单讨论一下 Feistel 网络的解密过程。Feistel 网络解密过程与其加密过程实质是相同的。以密文分组作为算法的输入，但以相反的次序使用子密钥，即第一轮使用 K_l，第二轮使用 K_{l-1}，直至第 l 轮使用 K_1。这意味着可以用同样的算法来进行加、解密。

可以证明，明文分组等于在上述算法中输入相应的密文分组而得到的输出结果。类似加密过程，解密过程可以概括为：先将密文分组 $C=R_lL_l$ 分成左、右长度相等的两半，分别记为 L_0' 和 R_0'，可以根据下列规则计算 $L_i'R_i'$，$1 \leqslant i \leqslant l$：

$$L_1' = R_{i-1}', \quad R_i' = L_{i-1}' \oplus F(R_{i-1}', K_i')$$

最后输出的分组是 $R_l'L_l'$；这里 $K_i' = K_{l-1}$。这样，只要证明 $R_l' = L_0$ 和 $L_l' = R_0$ 即可。显然，$L_0' = R_l$ 且 $R_0' = L_l$，根据加、解密规则，有

$$L_1' = R_0' = L_l = R_{l-1}, \quad R_1' = L_0' \oplus F(R_0', K_1') = R_l \oplus F(L_l, K_{l-1}) = L_{l-1}$$

$$L_2' = R_1' = L_{l-1} = R_{l-2}, \quad R_2' = L_1' \oplus F(R_1', K_2') = R_{l-1} \oplus F(L_{l-1}, K_{l-2}) = L_{l-2}$$

递归，有

$$L_{l-1}' = R_{l-2}' = L_2 = R_1, \quad R_{l-1}' = L_{l-2}' \oplus F(R_{l-2}', K_{l-1}') = R_2 \oplus F(L_2, K_1) = L_1$$

$$L_l' = R_{l-1}' = L_1 = R_0, \quad R_l' = L_{l-1}' \oplus F(R_{l-1}', K_l') = R_1 \oplus F(L_1, K_0) = L_0$$

这就验证了解密过程的正确性。注意，以上解密过程并不要求轮函数 F 是可逆的。

Feistel 网络有几种推广的形式。在 Feistel 网络中，L_i' 和 R_i' 的长度是相等的，如果 L_i' 和 R_i' 的长度不相等，则称为非平衡 Feistel 网络。在非平衡 Feistel 网络中，每一轮中的 F 函数都是相同的，只是密钥不同，如果每一轮中的 F 函数也是变化的，则称为非齐次非平衡 Feistel 网络，如 Khufu、MD4 等算法均采用了这种结构。

2. SP 网络结构

SP 网络是分组密码的另一种重要结构，SAFER、SHARK、AES 等著名的密码算法都采用此结构。在这种结构中，每一轮的输入首先被一个由子密钥控制的可逆函数 S 作用，然后再对所得结果用置换(或可逆线性变换)P 作用。S 和 P 被分别称为混淆层和扩散层，起混淆和扩散的作用，如图 3.4 所示。设计者可以根据 S 和 P 的某些密码指标来估计 SP 型密码对抗差分密码分析和线性密码分析的能力。与 Feistel 网络相比，SP 网络密码可以得到更快的扩散，但加、解密通常不相似。

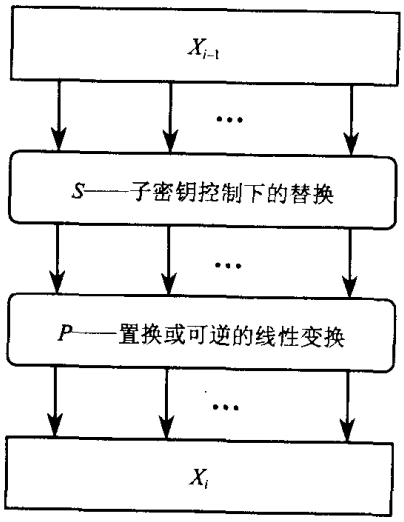


图 3.4 一轮 SP 网络加密过程

3.2 数据加密标准(DES)

数据加密标准(Data Encryption Standard，DES)是迄今为止使用最为广泛的加密算法。1973 年 5 月 13 日美国国家标准局 NBS(National Bureau of Standards)公布了一项公告，征求国家密码标准方案。IBM 提交了他们研制的一种密码算法，该算法是由早期的 LUCIFER 密码改进而得的。在经过大量的公开讨论之后该密码算法于 1977 年 1 月 15 日被正式批准为美国联邦信息处理标准，即 FIPS-46，同年 7 月 15 日开始生效。规定每隔 5 年由美国国

家保密局(National Security Agency)重新评估它是否继续作为联邦加密标准。最近的一次评估是在 1994 年 1 月，当时决定 1998 年 12 月以后，DES 不再作为联邦加密标准。新的美国联邦加密标准称为高级加密标准 AES(Advanced Encryption Standard)。尽管如此，DES 对推进密码理论的发展和应用仍起了重要作用，并对学习和研究分组密码的基本理论、设计思想和实际应用有着珍贵的参考价值。

3.2.1 DES 描述

DES 是分组长度为 64 比特的分组密码算法，密钥长度也是 64 比特，其中每 8 比特有一位奇偶校验位，因此有效密钥长度为 56 比特。DES 算法是公开的，其安全性依赖于密钥的保密程度。DES 结构框图如图 3.5 所示。

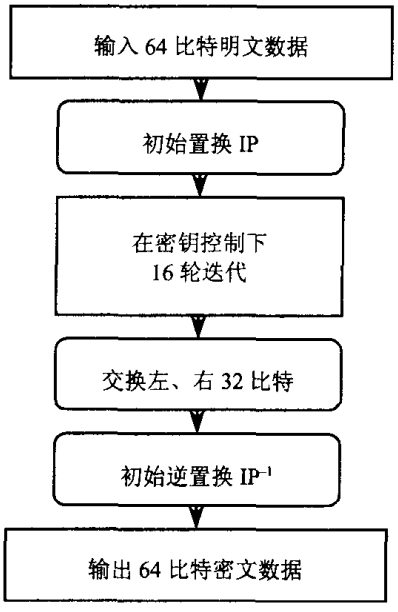


图 3.5　DES 算法框图

初始置换 IP 和初始逆置换 IP^{-1}：将 64 比特明文数据用初始置换 IP 置换，得到一个乱序的 64 比特明文分组，然后分成左、右等长的 32 比特，分别记为 L_0 和 R_0。进行 16 轮完全类似的迭代运算后，将所得左、右长度相等的两半 L_{16} 和 R_{16} 交换得到 64 比特数据 $R_{16}\ L_{16}$，最后再用初始逆置换 IP^{-1} 进行置换，产生密文数据组。置换表自左向右、自上而下的 64 个位置对应 64 比特数据组，置换表中的数字表示将 64 比特数据组中该数字所在位置的比特置换为该数字表示的位置的比特。初始置换 IP 和初始逆置换 IP^{-1} 如下所示：

初始置换 IP								初始逆置换 IP^{-1}							
58	50	42	34	26	18	10	2	40	8	48	16	56	24	64	32
60	52	44	36	28	20	12	4	39	7	47	15	55	23	63	31
62	54	46	38	30	22	14	6	38	6	46	14	54	22	62	30
64	56	48	40	32	24	16	8	37	5	45	13	53	21	61	29
57	49	41	33	25	17	9	1	36	4	44	12	52	20	60	28
59	51	43	35	27	19	11	3	35	3	43	11	51	19	59	27
61	53	45	37	29	21	13	5	34	2	42	10	50	18	58	26
63	55	47	39	31	23	15	7	33	1	41	9	49	17	57	25

例如将 64 比特数据表示为 $M=(m_1,m_2,\cdots,m_{64})$，则在初始置换 IP 的作用下变为

$IP(M)=(m_{58},m_{50},m_{42},m_{34},m_{26},m_{18},m_{10},m_2,\cdots,m_{63},m_{55},m_{47},m_{39},m_{31},m_{23},m_{15},m_7)$。如果将 $IP(M)$ 用初始逆置换 IP^{-1} 作用，将会得到 M。例如 M 中的第 60 个比特 m_{60}，在 $IP(M)$ 中位于第 9 个比特位，$IP(M)$ 在初始逆置换 IP^{-1} 作用下，第 9 个比特移至第 60 个比特位。这说明 IP 和 IP^{-1} 互逆。

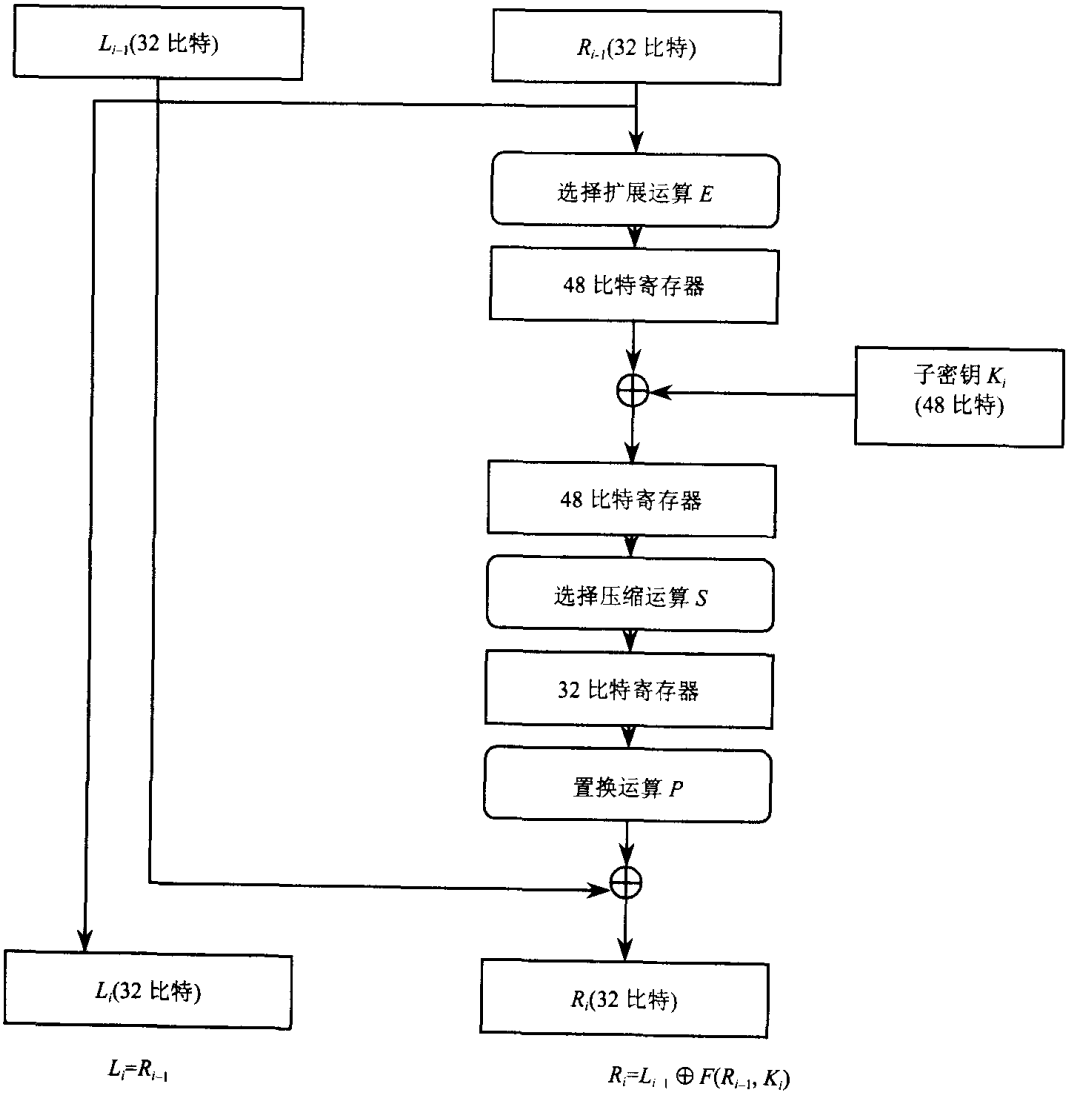


图 3.6 DES 的一轮迭代

迭代变换：迭代变换是 DES 算法的核心部分，如图 3.6 所示。每轮开始时将输入的 64 比特数据分成左、右长度相等的两半，右半部分原封不动地作为本轮输出的 64 比特数据的左半部分，同时对右半部分进行一系列的变换，即用轮函数 F 作用右半部分，然后将所

得结果(32 比特数据)与输入数据的左半部分进行逐位异或，最后将所得数据作为本轮输出的 64 比特数据的右半部分。

从图 3.6 可以看出，轮函数 F 由选择扩展运算 E、与子密钥的异或运算、选择压缩运算 S 和置换 P 组成。下面分别介绍这几种运算。

选择扩展运算 E：将输入的 32 比特数据扩展为 48 比特的输出数据，变换表如下：

E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

如果将输入的 32 比特数据按 E 中所标位置顺序读出，则可得到 48 比特的输出数据。可以看出 1、4、5、8、9、12、13、16、17、20、21、24、25、28、29、32 这 16 个位置上的数据都被读了两次。

与子密钥的异或运算：将选择扩展运算的 48 比特输出数据与子密钥 K_i(48 比特)进行异或运算。

选择压缩运算：将输入的 48 比特数据从左至右分成 8 组，每组 6 比特。然后输入 8 个 S 盒，每个 S 盒为一非线性代换，有 4 比特输出，如图 3.7 所示。

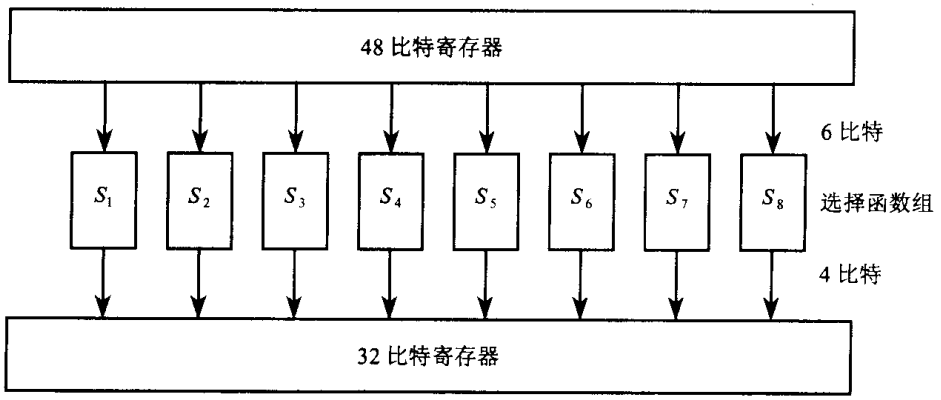


图 3.7　选择压缩运算 S

盒S_1、S_2、S_3、S_4、S_5、S_6、S_7、S_8的选择函数关系分别如下：

	行\列	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_1	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	1	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

对每个盒S_i，6比特输入中的第1和第6比特组成的二进制数确定S_i的行，中间4位二进制数用来确定S_i的列。S_i中相应行、列位置的十进制数的4位二进制数表示作为输出。

例如，S_2 的输入为 101001，则行数和列数的二进制表示分别是 11 和 0100，即第 3 行和第 4 列，S_2 的第 3 行第 4 列的十进制数为 3，用 4 位二进制数表示为 0011，所以 S_2 的输出为 0011。

置换 P：置换 P 如下所示：

P			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

最后需要描述的是用密钥 K 来产生 16 个 48 比特的子密钥 K_i，$1 \leqslant i \leqslant 16$。

子密钥的产生：给定 64 比特的密钥 K，用置换选择 1(PC-1)作用，去掉了输入的第 8、16、24、31、40、48、56、64 位，这 8 比特是奇偶校验位，并重排实际 56 比特的密钥。将得到的 56 比特数据分成左、右等长的 28 比特，分别记为 C_0 和 D_0。对 $1 \leqslant i \leqslant 16$，计算 $C_i = LS_i(C_{i-1})$ 和 $D_i = LS_i(D_{i-1})$。将每轮 56 比特数据 C_i D_i 用置换选择 2(PC-2)作用，去掉第 9、18、22、25、35、38、43、54 位，同时重排剩下的 48 比特，输出作为 K_i，产生子密钥的密钥编排算法如图 3.8 所示。这里 LS_i 表示循环左移 1 位(当 i=1, 2, 9, 16 时)或 2 位(其他情况)，置换选择 1(PC-1)和置换选择 2(PC-2)如下：

PC-1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

PC-2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

进行 16 轮完全类似的迭代运算后，将所得左、右长度相等的两半 L_{16} 和 R_{16} 交换，得 64 比特数据 R_{16} L_{16}，用初始逆置换 IP^{-1} 进行置换，产生密文数据组。置换表自左向右、

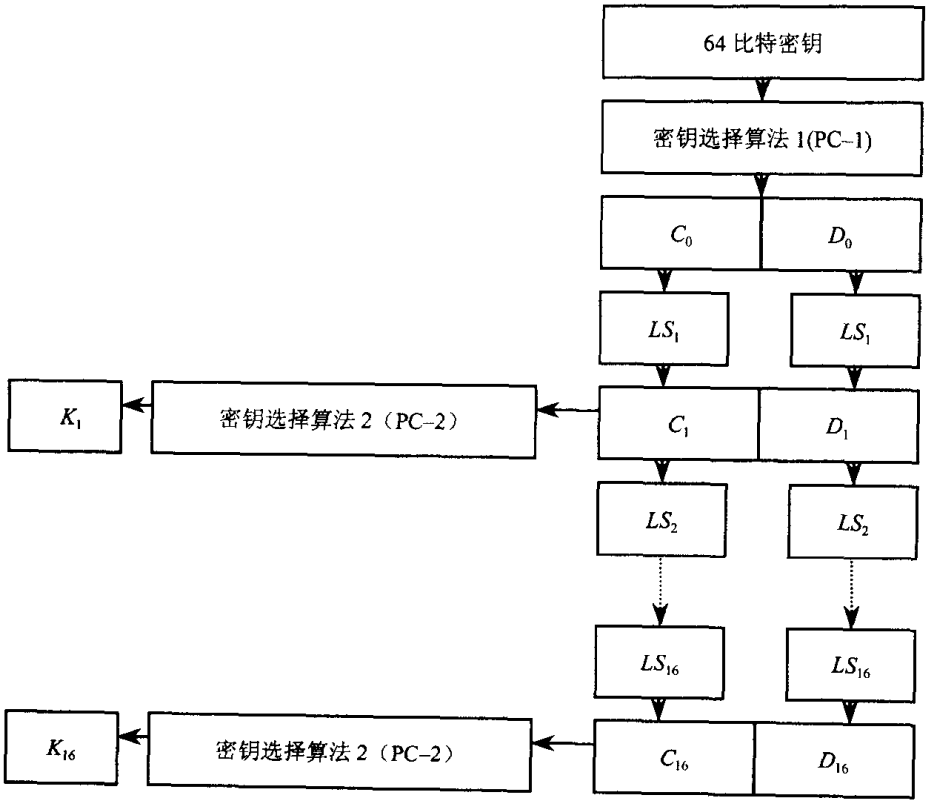


图 3.8 密钥编排算法

自上而下的 64 个位置对应 64 比特数据组，置换表中的数字表示将 64 比特数据组中该数字所在位置的比特置换为该数字表示的位置的比特。

解密：由于 DES 算法是在 Feistel 网络结构的输入和输出阶段分别添加初始置换 IP 和初始逆置换 IP^{-1} 而构成的，所以它的解密使用与加密同样的算法，只是子密钥的使用次序相反。

3.2.2 DES 问题讨论

当 DES 算法被建议作为一个标准时，曾出现过很多批评，其中最有争议的问题之一就是 S 盒。S 盒是 DES 的安全核心，因为在 DES 算法中，除了 S 盒外，所有计算都是线性的。然而 S 盒的设计准则并没有完全得到规范，有人认为 S 盒可能存在陷门。不能排除这种说法，但至今没有迹象表明 S 盒中存在陷门。

1976 年，美国国家安全局公布了下列几条 S 盒的设计原则：

(1) S 盒的每一行是整数 0, 1, …, 15 的一个置换；

(2) 没有一个 S 盒是它输入的线性或仿射函数；

(3) 改变S盒的一个输入比特至少要引起两比特的输出改变；

(4) 对任何一个S盒$S_i(1\leqslant i\leqslant 8)$和任何一个输入$x$(6比特串)，$S_i(x)$与$S_i(x\oplus 001100)$至少要有两个比特不同；

(5) 对任何一个S盒$S_i(1\leqslant i\leqslant 8)$和任何一个比特对$(e,f)$，$S_i(x)\neq S_i(x\oplus 11ef00)$；

(6) 对任何一个S盒$S_i(1\leqslant i\leqslant 8)$，如果固定一个输入比特，考察一个特点输出比特的值，该输出比特值为0的输入个数与输出比特值为1的输入个数接近。

据说后面两个特性是为通过美国国家安全局要求的设计准则而专门导出的。目前仍然不知道在S盒的构造中是否使用了进一步的设计准则。

关于DES算法的另一个最有争议的问题就是担心实际56比特的密钥长度不足以抵御穷举攻击，因为密钥量只有$2^{56}\approx 10^{17}$个。事实证明确实如此。早在1977年，Diffie和Hellman就建议制造每秒能测试10^6个密钥的VLSI芯片。每秒测试10^6个密钥的机器大约需要一天就可以搜索完整个DES密钥空间。他们估计制造这样的机器大约需要20 000 000美元。在CRYPTO'93大会上，Session和Wiener给出了一个非常详细的密钥搜索机器的设计方案，这个机器基于并行运算的密钥搜索芯片，16次加密能同时完成。此芯片每秒能测试5×10^7个密钥，用5 760个芯片组成的系统需要花费100 000美元，平均用1.5天左右就可找到DES密钥，如果一个机器使用10个这样的系统将花费1 000 000美元，但可将平均搜索时间降到3.5小时左右。1997年1月28日，美国的RSA数据安全公司在RSA安全年会上公布了一项“秘密密钥挑战”竞赛，其中包括悬赏1万美元破译密钥长度为56比特的DES。美国科罗拉多州的程序员Verser从1997年3月13日起，用了96天时间，在Internet上数万名志愿者的协同工作下，于6月17日成功找到了DES的密钥，赢得了1万美元。这一事件表明，依靠Internet的分布式计算能力，用穷举式攻击方法破译DES已成为可能，这意味着随着计算能力的增长，必须相应地增加算法密钥的长度。1998年7月电子前沿基金会(EFF)使用一台25万美元的电脑在56小时内破译了56比特密钥的DES。1999年1月RSA数据安全会议期间，电子前沿基金会用22小时15分钟就宣告破解了DES的密钥。

3.2.3 DES的变形

由于已经决定1998年12月以后，DES将不再作为联邦加密标准。新的美国联邦加密标准被称为高级加密标准AES(Advanced Encryption Standard)。1997年9月12日美国联邦登记处(FR)公布了征集AES候选算法的通告。在新的加密标准实施之前，由于DES容易受到穷举式攻击，人们已着手研究替代的加密算法，为了使已有的DES算法投资不浪费，人们尝试用DES和多个密钥进行多次加密，其中三重DES已被广泛采用。

1. 双重 DES

最简单的多次加密形式是用两个密钥进行两次加密，如图 3.9 所示。已给一个明文 P 和两个加密密钥 K_1 和 K_2，密文为 $C=E_{K_2}(E_{K_1}(P))$。解密要求密钥以相反的次序使用：$P=D_{K_1}(D_{K_2}(C))$。对 DES 来说，这个算法的密钥长度是两个密钥长度的和，即 112 比特，因此密码强度似乎增加了一倍，但问题并不这么简单，需要严格的数学证明。

假设对于 DES 和所有 56 比特密钥，任意给定两个密钥 K_1 和 K_2，都能找到一个密钥 K_3，使得 $E_{K_2}(E_{K_1}(P))=E_{K_3}(P)$。如果假设成立，则 DES 的两重加密或者多重加密都将等价于用一个 56 比特密钥的一次加密。

直观上看，上面的假设不可能为真。因为 DES 加密事实上就是从一个 64 比特分组到另一个 64 比特分组的置换， 而 64 比特分组共有 2^{64} 种可能的状态，因而可能的置换个数为

$$2^{64}!>10^{347380000000000000000}>10^{3\cdot10^{20}}$$

另一方面，DES 的每个密钥确定了一个置换，因而总的置换个数为 $2^{56}<10^{17}$。虽然有许多证据支持上面的假设不成立，但直到 1992 年才有人证明了这个结果。

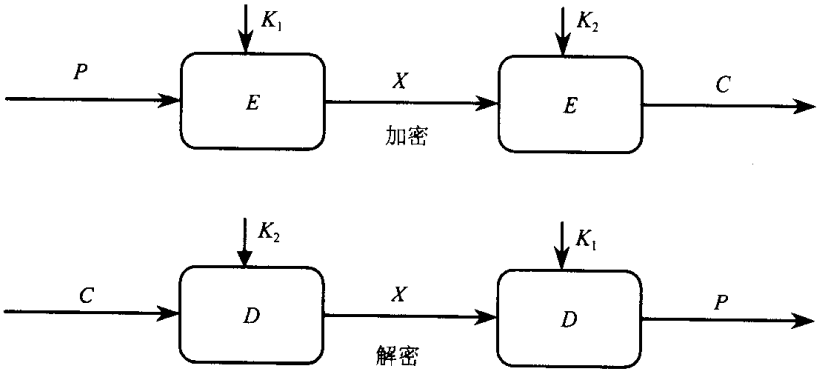


图 3.9 双重加密

2. 中途攻击

现在已经知道使用 DES 算法用不同的密钥进行两次加密并不等价于一次 DES 加密。但是用另一种方法可以攻击这种算法，这种方法不依赖于任何 DES 的特殊属性，它对任何分组密码都有效，通常称其为中途攻击算法。

中途攻击算法基于以下观察，如果有 $C=E_{K_2}(E_{K_1}(P))$，则 $X=E_{K_1}(P)=D_{K_2}(C)$(见图 3.9)。给定一个已知明、密文对 (P,C)，攻击方法如下：首先用所有 2^{56} 个可能的密钥加密

P，得到2^{56}个可能的值，把这些值从小到大存在一个表中；然后再用所有2^{56}个可能的密钥对C进行解密，每次做完解密都将所得的值与表中值进行比较，如果发现与表中的一个值相等，则它们对应的密钥可能分别是K_1和K_2。现在用一个新的明、密文对检测所得到的两个密钥，如果满足$E_{K_1}(P)=D_{K_2}(C)$，则把它们接受为正确的密钥。

对于任意一个给定的明文P，双重 DES 产生的密文值有2^{64}种可能。双重 DES 实际使用了一个 112 比特的密钥，因此有2^{112}种可能的密钥。这样平均来说对于一个给定的明文P，将产生一个给定密文C的不同的 112 比特的密钥个数是$2^{112}/2^{64}=2^{48}$，因而上述过程对于第一对明、密文有2^{48}次加、解密结果相等。如果再加上一对已知明、密文，误报率为$2^{48}/2^{64}=2^{-16}$。因此，已知两对明、密文，实施中途攻击检测到正确密钥的概率为$1-2^{-16}$。攻击的工作量为2^{56}，这与攻击 DES 的工作量2^{55}差不多。

3. 三重 DES

对付中途攻击的有效方法是用三个密钥进行三次加密。这将把已知明文攻击的工作量提高到2^{112}，这个密钥长度大大提高了抗攻击强度。其缺点是要使用$3\times56=168$比特的密钥。作为替代方案，Tuchman 建议使用两个密钥进行加密—解密—加密(EDE)的方案，即加密为$C=E_{K_1}(D_{K_2}(E_{K_1}(P)))$，解密为$P=D_{K_1}(E_{K_2}(D_{K_1}(C)))$，如图 3.10 所示。

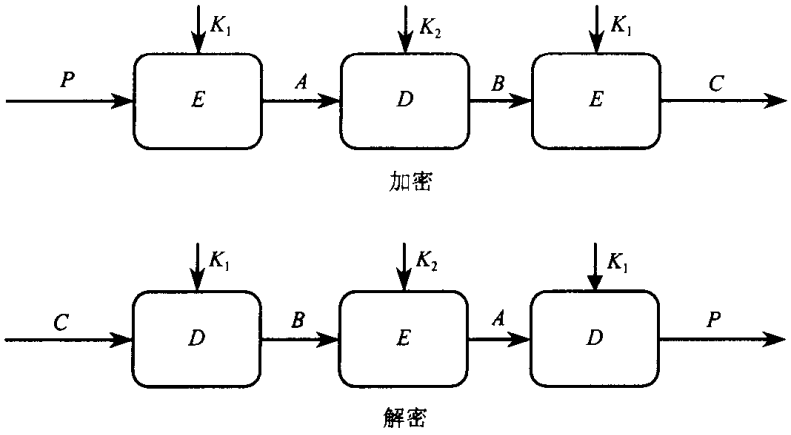


图 3.10　三重加密

第二个步骤使用解密并没有密码编码上的考虑，相对于使用加密，它的惟一优点是可以使三重 DES 的用户能够解密原来仅用一重 DES 加密的数据，即$P=D_{K_1}(E_{K_1}(D_{K_1}(C)))=D_{K_1}(C)$。

使用两个密钥的三重 DES 是比较受欢迎的一个算法，已被密钥管理标准 ANS X9.17

和 ISO 8732 采用。

目前还没有针对两个密钥三重 DES 的实用攻击方法，但有一些设想，限于篇幅这里不做介绍。

虽然针对两个密钥三重 DES 并没有实用的攻击方法。但许多研究人员感到具有三个密钥的三重 DES 是更好的选择。其加、解密过程如下：

$$C = E_{K_3}(D_{K_2}(E_{K_1}(P)))，\quad P = D_{K_1}(E_{K_2}(D_{K_3}(C)))$$

与一重 DES 的兼容性可以通过取 $K_1 = K_2$ 或 $K_3 = K_2$ 得到。

3.3 高级加密标准 AES

1997 年 4 月 15 日，(美国)国家标准技术研究所(NIST)发起征集高级加密标准 AES(Advanced Encryption Standard)的活动，目的是确定一个非保密的、可以公开技术细节的、全球免费使用的分组密码算法，以作为新的数据加密标准。1997 年 9 月 12 日，美国联邦登记处公布了正式征集 AES 候选算法的通告。作为进入 AES 候选过程的一个条件，开发者必须承诺放弃被选中算法的知识产权。许多个人和公司积极响应。AES 的基本要求是：比三重 DES 快；至少与三重 DES 一样安全；数据分组长度为 128 比特；密钥长度为 128/192/256 比特。1998 年 8 月 12 日，在首届 AES 会议上指定了 15 个候选算法。1999 年 3 月 22 日第二次 AES 会议上，全球各机构和个人对 15 个候选算法的分析结果进行了研讨，并于 1999 年 8 月将候选名单减少为 5 个，这 5 个算法是 RC6、Rijndael、SERPENT、Twofish 和 MARS。2000 年 4 月 13 日，在第三次 AES 会议上，又对这 5 个候选算法的各种分析结果进行了讨论。2000 年 10 月 2 日，NIST 宣布了获胜者——Rijndael 算法。至此，历时三年的遴选过程宣告结束，并确定比利时研究者 Vincent Rijmen 和 Joan Daemen 研制的 Rijndael 算法为新的数据加密标准 AES。

AES 算法的设计考虑到如下三条准则：

(1) 能抵抗所有已知的攻击；

(2) 在多个平台上同时运行时速度要快并且编码紧凑；

(3) 设计简单。

在大多数分组密码中，轮变换中有 Feistel 结构，通常将中间状态的部分比特不加改变简单转置到下一轮的其他位置。AES 算法的轮变换中没有 Feistel 结构，轮变换是由三个不同的可逆一致变换组成，称之为层，这里的一致是指状态的每个比特都是用类似的方法进行处理的。不同层的选择建立在宽轨迹策略的应用基础上，这是提供抗差分密码分析和线

性密码分析的一种设计方法，在宽轨迹策略中，每层都有它自己的函数：

(1) 线性混合层：确保多轮之上的高度扩散。

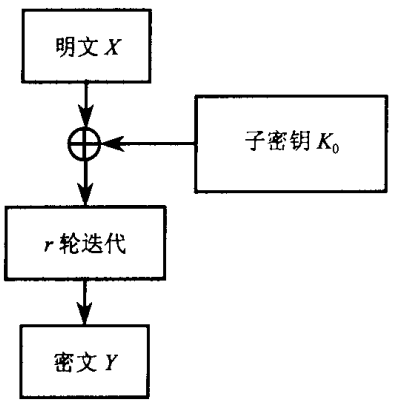
(2) 非线性层：具有最优–最差情形的非线性 S 盒的并行应用。

(3) 密钥加层：轮密钥简单地异或到中间状态上。

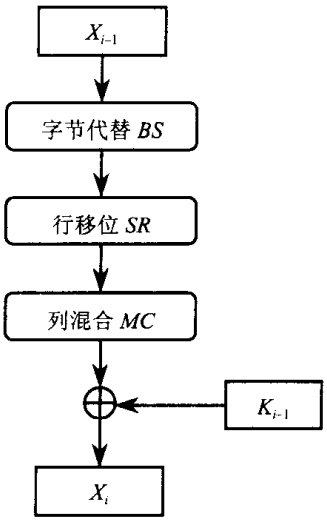
AES 是一个迭代分组密码，其分组长度和密钥长度都可以改变。分组长度和密钥长度可以独立地设定为 128 比特、192 比特或者 256 比特。第一轮之前，应用了一个密钥加层，它对密码的安全性不做任何贡献。为了使加密和解密在结构上更为相似，最后一轮的线性混合层与其他的混合层不同，可以证明它不会影响算法的安全性。图 3.11(a)是 AES 加密算法的框图。

1. 一轮 AES 结构

每一轮变换由 4 个不同的可逆变换组成：字节代替 *BS*、行移位 *SR*、列混合 *MC* 和轮密钥异或。这四个变换构成 3 层：非线性层(字节代替 *BS*)、线性混合层(行移位 *SR* 和列混合 *MC*)和密钥加层，如图 3.11(b)所示。最后一轮稍微有点不同，比其他轮少了变换列混合 *MC*。



(a) AES 算法框图



(b) 一轮 AES 结构

图 3.11　AES 算法结构

如果记与轮密钥 K_i 异或的变换为 $O_{K_i}(0 \leqslant i \leqslant r)$，则加密过程可表示为

$$Y = O_{K_r} \circ SR \circ BS \circ O_{K_{r-1}} \circ MC \circ SR \circ BS \circ \cdots \circ O_{K_1} \circ MC \circ SR \circ BS \circ O_{K_0}(X)$$

其中“∘”表示变换的复合，$K_i(0 \leqslant i \leqslant r)$是由密钥 K 产生的轮密钥。

2. 状态、密钥和轮数

各个不同的变换都在称之为状态的中间结果上运算。状态可以用一个字节矩形阵列来表示，该阵列有 4 行，列数记为 Nb 并且等于分组长度除以 32。

密钥类似地用一个 4 行的字节矩形阵列表示，列数记为 Nk 并且等于密钥长度除以 32，如图 3.12 所示。

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$

图 3.12 状态(Nb=6)和密钥(Nk=4)布局的例子

有时这些分组被看成是 4 字节矢量的一维阵列，其中每个矢量由矩形阵列中相应的列组成。因而一维阵列的长度有 4、6 或 8，分别用 0 至 3、0 至 5 或 0 至 7 标记。4 字节矢量有时被看成是字。

约定用 (a,b,c,d) 表示 4 字节矢量或字的时候，a、b、c、d 分别是列中位置在 0、1、2、3 的字节。

在与外界交互的时候，输入/输出被看成是 8 比特字节的一维阵列，从 0 到 $4Nb$–1。因而分组的长度有 16、24 或 32 字节，标记范围分别为 0 至 15、0 至 23 或 0 至 31。

密钥也被看成是 8，比特字节的一维阵列，从 0 到 $4Nk$–1。因而分组的长度有 16、24 或 32 字节，标记范围分别为 0 至 15、0 至 23 或 0 至 31。

输入的明文字节按 $a_{0,0},a_{1,0},a_{2,0},a_{3,0},a_{0,1},a_{1,1},a_{2,1},a_{3,1},\cdots$ 的顺序映射到状态字节上。密钥按 $k_{0,0},k_{1,0},k_{2,0},k_{3,0},k_{0,1},k_{1,1},k_{2,1},k_{3,1},\cdots$ 的顺序映射到阵列上。密码运算结束时，密文的输出是以同样的次序从状态中取状态字节而获得的。如果一个字节的一维标记是 n 并且二维标记是 (i,j)，则有

$$i = n \bmod 4,\quad j = \lfloor n/4 \rfloor,\quad n = 4j + i$$

标记 i 是4字节矢量或字中的第n字节数，而 j 正是对该矢量或字的标记。

轮数与 Nb 和 Nk 的值有关。如果记轮数为 r，则它们的关系如下：

轮数 r 与 Nb 和 Nk			
r	Nb=4	Nb=6	Nb=8
Nk=4	10	12	14
Nk=6	12	12	14
Nk=8	14	14	14

3. 字节代替变换

字节代替变换是一个非线性的字节代替，它在每个状态字节上独立进行运算。代替表(或S盒)是可逆的，并且是由两个可逆的变换复合而成：

- 首先在有限域 $GF(2^8)$ 中取逆元，00 的逆元规定为 00;
- 其次，将所得逆元经过下面定义的($GF(2)$ 上的)仿射变换的作用：

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

S盒对状态所有字节的变换记为 BS，图 3.13 描述了变换 BS 在状态上作用的效果。

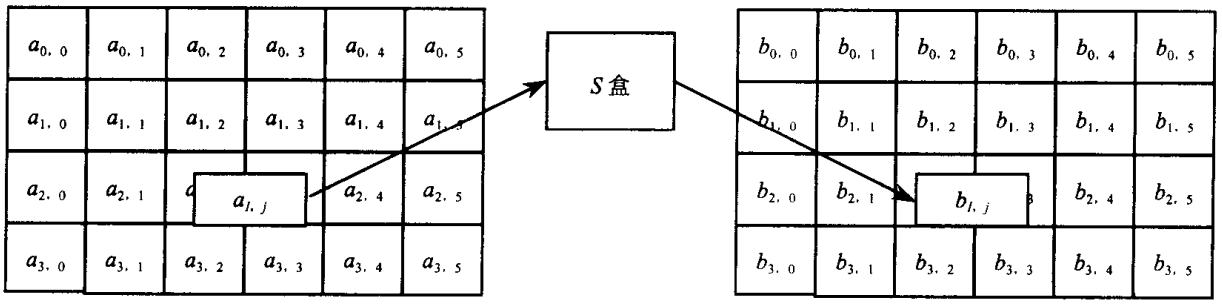


图 3.13　BS 作用在状态的单个字节上

4. 行移位变换

在行移位变换中，状态的第一行没有任何变化，第二行循环移位 C1 字节，第三行循

环移位 $C2$ 字节，第四行循环移位 $C3$ 字节。位移量 $C1$、$C2$ 和 $C3$ 与分组长度 Nb 有关，如下表：

对应于不同分组长度的位移量			
Nb	$C1$	$C2$	$C3$
4	1	2	3
6	1	2	3
8	1	3	4

所有状态的位移变换记为 SR。图 3.14 描述了变换 SR 在状态行上作用的效果。

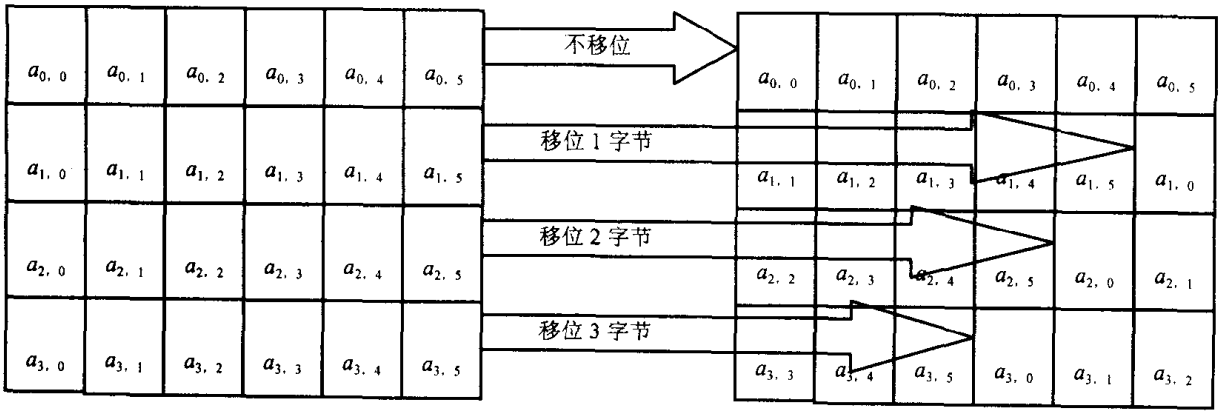


图 3.14 SR 作用在状态行上

5. 列混合变换

在列混合变换中，将状态的列视为有限域 $GF(2^8)$ 上的 4 维向量并且与 $GF(2^8)$ 上的一个固定的可逆方阵 $\boldsymbol{A}$ 相乘：

$$\begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix}$$

这里为了简便，将矩阵中的元素用十六进制表示。所有状态列的混合记为 MC。图 3.15 描述了变换 MC 在状态列上作用的效果。

6. 轮密钥加

用简单的比特异或将一个轮密钥作用在状态上，如图 3.16 所示。轮密钥通过密钥调度

算法从密钥中产生，其长度等于分组长度。

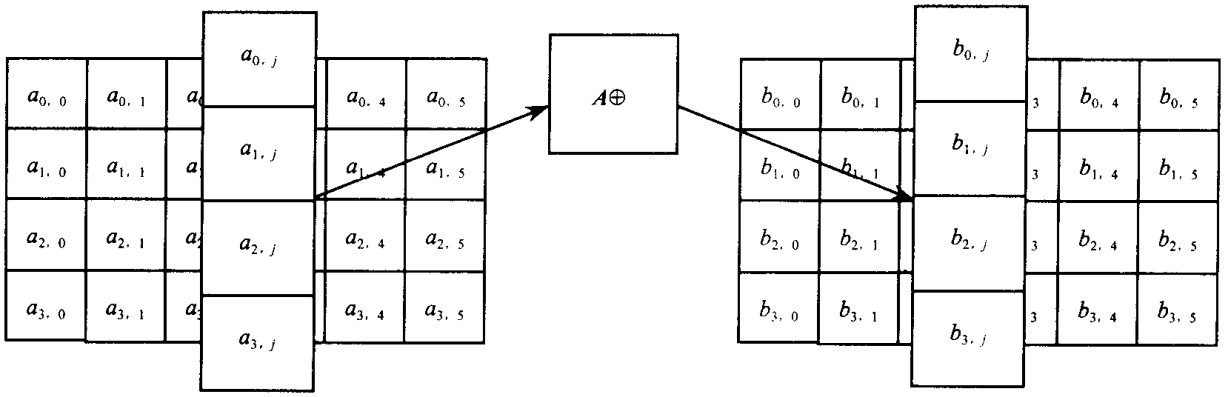


图 3.15　*MC* 作用在状态列上

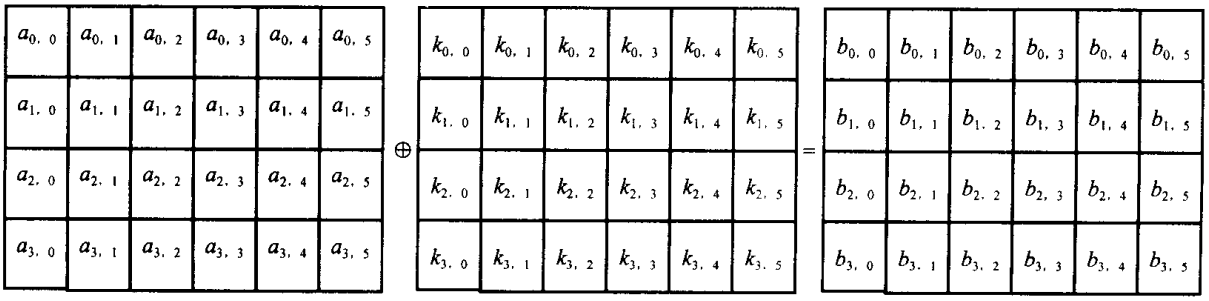


图 3.16　轮密钥比特异或到状态上

7. 密钥调度算法

轮密钥是通过密钥调度算法从密钥中产生的，这其中包括两个组成部分：密钥扩展和轮密钥选取。基本原理如下：

- 所有轮密钥比特的总数等于轮数加 1 乘以分组长度 (如 128 比特的分组长度和 10 轮迭代，共需要 1 408 比特的密钥)；
- 将密码密钥扩展成一个扩展密钥；
- 轮密钥按下述方式从扩展密钥中选取：第一个轮密钥由一开始的 *Nb* 个字组成，第二个轮密钥由接下来的 *Nb* 个字组成，如此继续下去。

密钥扩展：扩展密钥是一个 4 字节字的直线阵列，记为 $\boldsymbol{W}=(w_0,w_1,\cdots,w_{Nb(r+1)})$，开始 *Nk* 个字由密钥组成，其他字由前面字的递归来定义。*Nk* 小于或等于 6 的情况与 *Nk* 大于 6 的情况扩展算法是不同的。

对于 $Nk \leqslant 6$，有

$$w_i = \begin{cases} k_i & \text{如果} i \leqslant Nk \\ w_{i-Nk} \oplus S(R(w_{i-1})) \oplus c_{i/Nk} & \text{如果} i \text{是} Nk \text{的倍数} \\ w_{i-Nk} \oplus w_{i-1} & \text{否则} \end{cases}$$

可以看出，开始 Nk 个字是由密钥填充的，随后的每个字 w_i 等于前面的字 w_{i-1} 和 Nk 个位置之前的字 w_{i-Nk} 的异或。对于 Nk 的整数倍处的字，在异或之前，要对 w_{i-1} 进行变换，并且还要异或一个轮常数，这个变换由函数 S 和 R 复合而得，函数 $S(\circ)$ 是把 AES 的 S 盒作用到输入字的每个字节上，函数 $R(\circ)$ 是把输入字的 4 个字节循环移位一个字节，如输入字为 (a,b,c,d)，则输出字为 (b,c,d,a)。轮常数与 Nk 无关，定义为

$$c_i = (c_{0,i},00,00,00),$$

其中 $c_{0,0} = 01$，$c_{0,i} = x \oplus c_{0,i-1}$。

对于 $Nk > 6$，有

$$w_i = \begin{cases} k_i & \text{如果} i \leqslant Nk \\ w_{i-Nk} \oplus S(R(w_{i-1})) \oplus c_{i/Nk} & \text{如果} i \text{是} Nk \text{的倍数} \\ w_{i-Nk} \oplus S(w_{i-1}) & \text{如果} i-4 \text{是} Nk \text{的倍数} \\ w_{i-Nk} \oplus w_{i-1} & \text{否则} \end{cases}$$

可以看出 $Nk > 6$ 与 $Nk \leqslant 6$ 时密钥扩展算法的区别在于，当 i 满足 $i-4$ 是 Nk 的整数倍时，在异或之前，要把 AES 的 S 盒作用到 w_{i-1} 的每个字节。

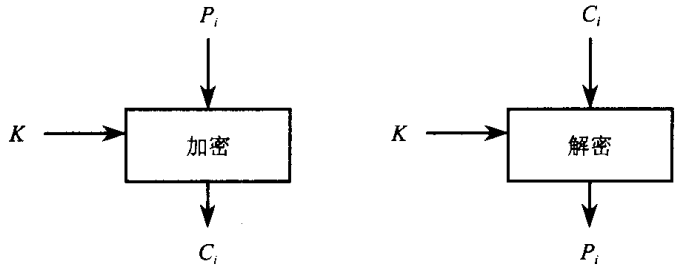
3.4 分组密码的工作模式

分组密码在加密时明文分组的长度是固定的。而实用中待加密消息的数据量是不定的，数据格式也可能是多种多样的，为了能在各种应用场合安全地使用分组密码，通常对不同的使用目的运用不同的工作模式。所谓分组密码的工作模式就是以该分组密码为基础构造的一个密码系统。目前已提出许多种分组密码的工作模式，如电码本(ECB)、密码分组链接(CBC)、密码反馈(CFB)、输出反馈(OFB)、级连(CM)、计数器、分组链接(BC)、扩散密码分组链接(PCBC)、明文反馈(PFB)、非线性函数输出反馈(OFBNLF)等模式。下面以 DES 算法为例介绍其中最重要和最基本的四种工作模式。

3.4.1 电码本模式(ECB)

ECB(Electronic CodeBook)模式是最简单的运行模式，它一次对一个 64 比特长的明文

分组进行加密，而且每次的加密密钥都相同，如图 3.17 所示。当密钥取定时，对于每一个明文分组，都有惟一的一个密文分组与之对应。因此可以想象有一个非常大的电码本，对每一个可能的明文分组，在电码本中都有惟一与之对应的密文分组。



(a) 加密　　(b) 解密

图 3.17　ECB 模式

对大于 64 比特的报文，需将其分为长为 64 比特的分组，最后一个分组可能需要填充。解密过程也是一次对一个密文分组进行解密，而且每次解密都使用同一个密钥。如图 3.17 所示，明文是由长为 64 比特的分组序列 $P_1,P_2,\cdots,P_N$ 构成的，相应的密文分组序列是 $C_1,C_2,\cdots C_N$，$i=1,2,\cdots,N$。

ECB 用于短数据(如加密密钥)时非常理想，但如果同一明文分组在消息中反复出现，产生的密文分组就会相同，因此，用于长消息时可能不够安全。如果消息有固定的结构，密码分析者就有可能会利用这种规律。例如，如果已知消息总是以某个事先规定的字段开始，那么分析者就有可能会得到许多明文，密文对。如果消息有重复的成分，而重复的周期是 64 比特的倍数，那么这些成分都有可能会被密码分析者识别出来。

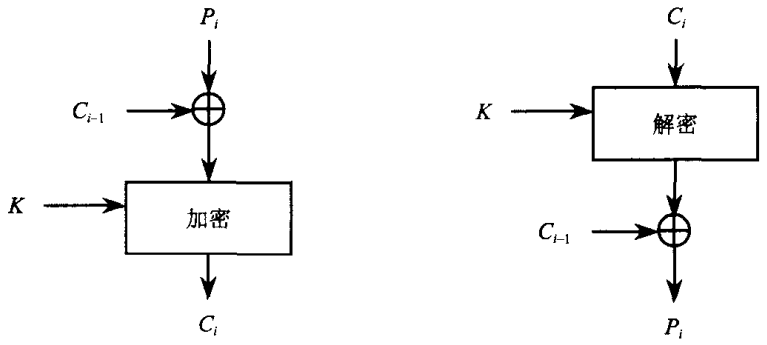
3.4.2　密码分组链接模式(CBC)

为了克服 ECB 的缺陷，希望设计一种方案使同一明文分组重复出现时产生的密文分组不同。一种简单的方案就是密码分组链接(Cipher Block Chaining)模式，如图 3.18 所示。每次加密使用同一密钥，加密算法的输入是当前明文组和前一次密文组的异或。因此加密算法的输入与明文分组之间不再有固定的关系，所以重复的明文分组不会在密文中暴露。

解密时，每一个密文分组解密后，再与前一个密文分组异或来产生出明文分组，即

$$D_K[C_i]\oplus C_{i-1}=D_K[E_K[C_{i-1}\oplus P_i]]\oplus C_{i-1}=C_{i-1}\oplus P_i\oplus C_{i-1}=P_i$$

这里 $C_i=E_K[C_{i-1}\oplus P_i])$，$i=1,2,\cdots,N$。



(a) 加密　　(b) 解密

图 3.18　CBC 模式

为了产生第一个密文分组，需要一个初始向量 $C_0 = V_I$ 与第一个明文分组异或。解密时，V_I 与解密算法的第一输出进行异或，以恢复第一个明文分组。

V_I 对于收、发双方都应是已知的。为使安全性程度最高，V_I 应像密钥一样被保护。保护 V_I 的原因如下：如果攻击者能欺骗接收方使用不同的 V_I 值，攻击者就能够在第一个明文分组中改变某些选定的比特，这是因为：

$$C_1 = E_K[V_I \oplus P_1]$$

$$P_1 = V_I \oplus D_K[C_1]$$

用 $X(i)$ 表示 64 比特分组 X 的第 i 个比特，那么 $P_1(i) = V_I(i) \oplus D_K[C_1](i)$，由异或的性质，可得

$$P_1(i)' = V_I(i)' \oplus D_K[C_1](i)$$

其中撇号表示比特补。上式意味着如果攻击者篡改 V_I 中的某些比特，则接收方收到的 P_1 中相应的比特也会发生变化。

由于 CBC 模式的链接机制，它对加密大于 64 比特的消息非常合适。CBC 模式除了能够获得保密性外，还能用于认证，可以识别攻击者在密文传输中是否做了数据篡改，比如组的重放、嵌入和删除等，但同时也会导致错误传播，密文传输中任何一组发生错误不仅会影响该组的正确译码，也会影响其下一组的正确译码。

3.4.3　密码反馈模式(CFB)

若待加密的消息必须按字符(如电传电报)处理时，可以采用 CFB(Cipher FeedBack)模式或 OFB(Output FeedBack)模式，这样做事实上是将 DES 转换成为流密码。流密码不需要对消息进行填充，而且运行是实时的。因此，如果只是传送字母流，就可使用流密码对每个字母直接加密并传送。

流密码的密文和明文一样长，因此，如果需要发送的每个字符长为 8 比特，就应按每次 8 比特来加密。如果超过 8 比特，就会造成浪费。图 3.19 所示的就是 CFB 模式。设传送的每个单元(如一个字符)是 j 比特长，通常取 j=8，与 CBC 模式一样，明文单元被链接在一起，使得密文是前面所有明文的函数。

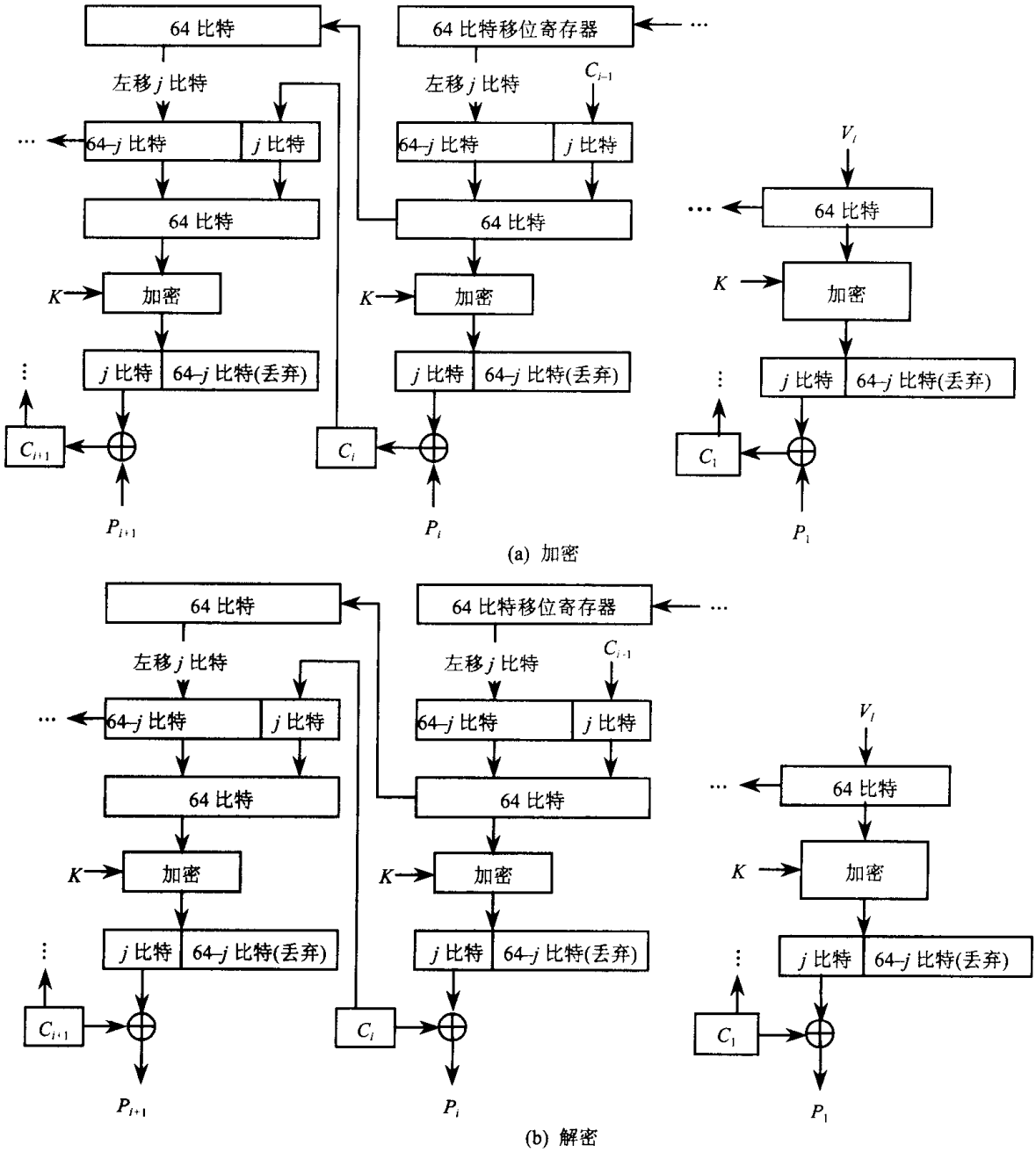


图 3.19　CFB 模式

加密时，加密算法的输入是 64 比特移位寄存器，其初值为某个初始向量V_I。加密算法输出的最左边(最高有效位) j 比特与明文的第一个单元 P_1 异或，产生出密文的第一个单元 C_1，并传输该单元。然后将移位寄存器中的内容左移 j 位并将 C_1 送入移位寄存器的最右边(最低有效位)j 位。这一过程持续进行直到明文的所有单元都被加密为止。

解密时，除了将收到的密文单元与加密函数的输出进行异或以产生明文单元外，其他与加密采用相同的方案。应当注意，这里仍然使用加密算法而不是解密算法，因为

$$P_1 = C_1 \oplus S_j(E(V_I))$$

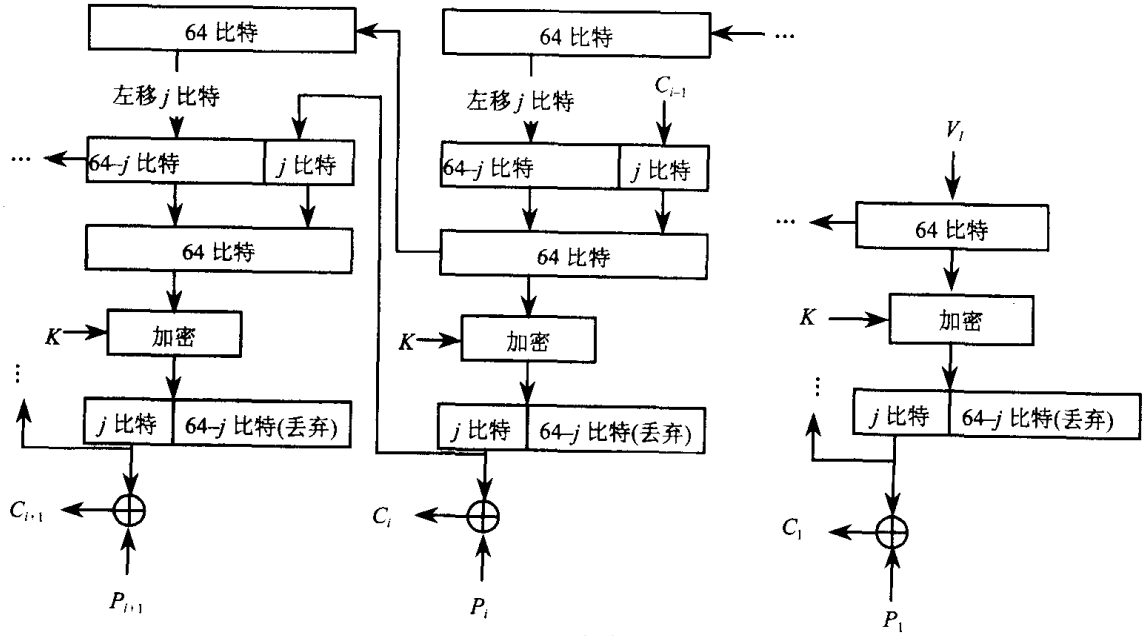
这里$S_j(X)$是X的第j个最高有效位，$C_1 = P_1 \oplus S_j(E(V_I))$。可以证明以后各步也有类似关系。

除了拥有 CBC 模式的优点外，CFB 模式还能适应用户不同的数据格式的需要。当然，它也拥有 CBC 模式的缺点，即也会导致错误传播。另外，它还有降低数据加密速度的缺点。

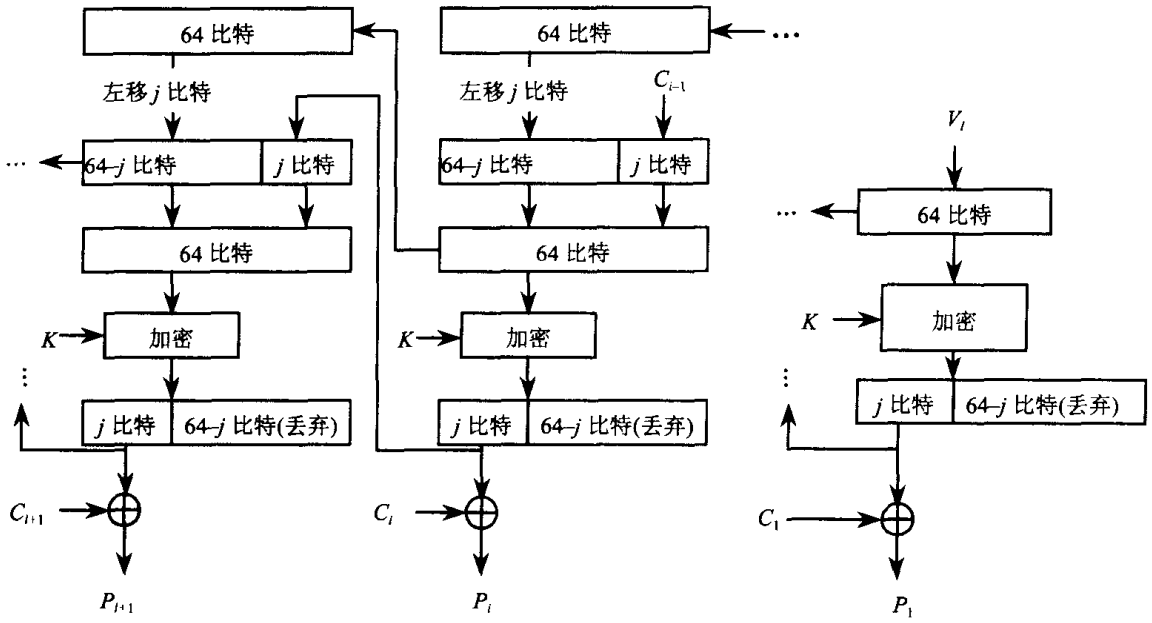
3.4.4 输出反馈模式(OFB)

OFB(Output FeedBack)模式在结构上类似于 CFB，如图 3.20 所示。不同之处是 OFB 模式将加密算法的输出反馈到移位寄存器，而 CFB 模式是将密文单元反馈到移位寄存器。

与 CFB 模式相比，OFB 模式的优点是传输过程中的比特错误不会被传播。例如，C_1 中出现 1 比特错误，在解密结果中只有 P_1 会受到影响，以后各明文单元则不受影响；而在 CFB 中，C_1 也作为移位寄存器的输入，因此它的 1 比特错误会影响到解密结果中各明文单元的值。



(a) 加密



(b) 解密

图 3.20　OFB 模式

OFB 模式的缺点是难于检测密文是否被篡改。如在密文中取 1 比特的补，那么在恢复后的明文中相应位置的比特也为原比特的补。因此攻击者有可能通过对数据部分和校验部分同时进行篡改，导致纠错码无法检测。

3.5　流密码简介

前面已经介绍了流密码的基本思想，利用密钥 k 生成一个密钥流 $z = z_0 z_1 \cdots$，密钥流由密钥流生成器 f 产生：$z_i = f(k, \sigma_i)$，这里的 σ_i 是加密器中的记忆元件(存储器)在时刻 i 的状态，f 是由密钥 k 和 σ_i 生成的函数，而 $\sigma_i (i > 0)$ 可能依赖于 $k, \sigma_0, x_0, x_1, \cdots, x_{i-1}$ 等参数。根据加密器中记忆元件的存储状态 σ_i 是否依赖于输入的明文字符，流密码可进一步分成同步和自同步两种。σ_i 独立于明文字符的叫做同步流密码，否则叫做自同步流密码。由于自同步流密码密钥流的生成与明文有关，因而较难从理论上进行分析。目前大多数研究成果都是关于同步流密码的。

3.5.1　同步流密码

在同步流密码中，由于 $z_i = f(k, \sigma_i)$ 与明文字符无关，因而密文字符 $y_i = e_{z_i}(x_i)$ 也不

依赖于此前的明文字符。因此，可将同步流密码的加密器分成密钥流生成器和加密变换器两个部分。如果与上述加密变换对应的解密变换为 $x_i = d_{z_i}(x_i)$，则可给出同步流密码的模型如图 3.21 所示。

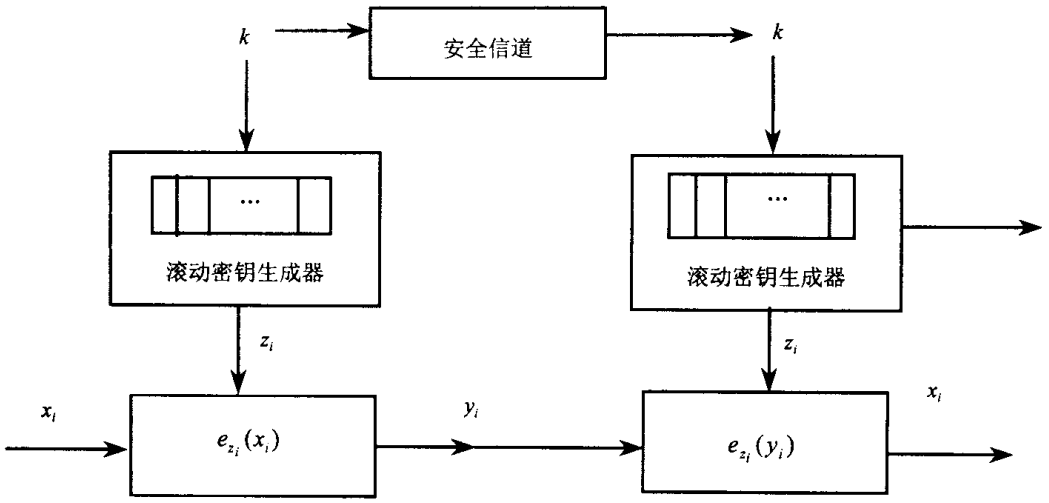


图 3.21 同步流密码体制模型

同步流密码的加密变换 e_{z_i} 可以有多种选择，只要保证变换是可逆的即可。实际使用的数字保密通信系统一般都是二元系统，因而在有限域 $GF(2)$ 上讨论的二元加法流密码(如图 3.22 所示)是最受欢迎的流密码体制，其加密变换可表示为 $y_i = z_i + x_i$。

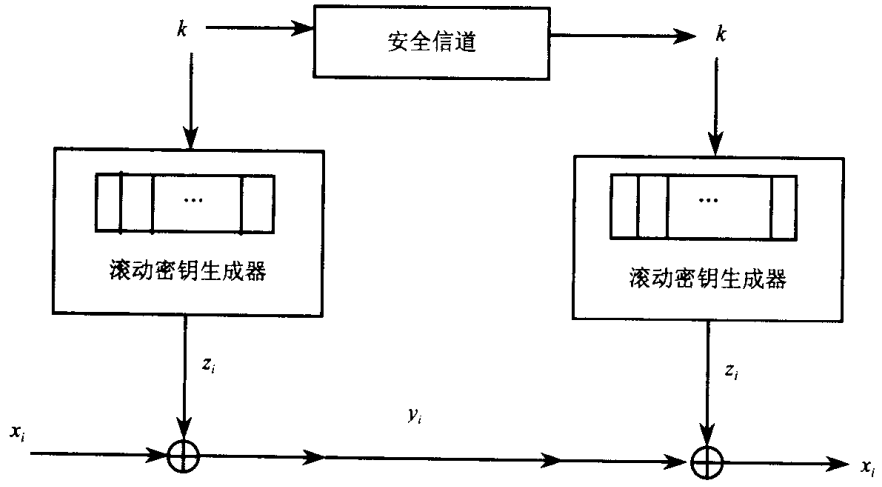


图 3.22 加法流密码体制模型

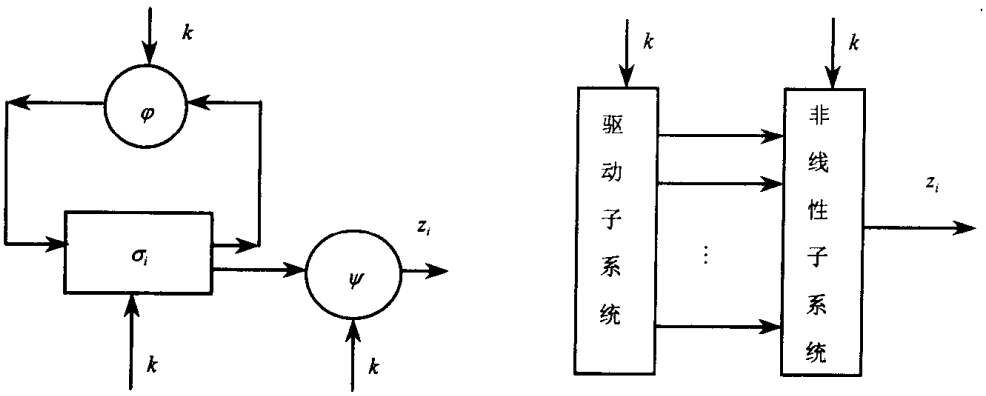
一次一密密码是加法流密码的原型。事实上，如果 $z_i = k_i$(即密钥用做滚动密钥流)，

则加法流密码就退化成一次一密密码。实际使用中，密码设计者的最大愿望是设计出一个滚动密钥生成器，使得密钥 k 经其扩展成的密钥流序列 z 具有如下一些性质：极大的周期、良好的统计特性、抗线性分析、抗统计分析等。

3.5.2　密钥流生成器

同步流密码的关键是密钥流生成器。一般可将其看成是一个参数为 k 的有限状态自动机，由一个输出符号集 Z 、一个状态集 Σ 、两个函数 φ 和 ψ 以及一个初始状态 σ_0 所组成，如图 3.23(a)所示。状态转移函数 $\varphi:\sigma_i \to \sigma_{i+1}$，将当前状态 σ_i 变为一个新状态 σ_{i+1}；输出函数 $\psi:\sigma_i \to z_i$，将当前状态 σ_i 变为输出符号集中的一个元素 z_i。这种密钥流生成器设计的关键在于找出适当的状态转移函数 φ 和输出函数 ψ，使得输出序列 z 满足极大的周期、良好的统计特性、抗线性分析和抗统计分析等要求，并且要求在设备上是节省的和容易实现的。为了实现这一目标，必须采用非线性函数。

由于非线性的 φ 的有限状态自动机理论很不完善，相应的密钥流生成器的分析工作受到极大的限制。而采用线性的 φ 和非线性的 ψ 时，却能够进行深入的分析并可以得到好的生成器。为方便，可将这类生成器分成驱动部分和非线性组合部分，如图 3.23(b)所示。驱动部分控制状态转移，并为非线性组合部分提供统计性能良好的序列；而非线性组合部分利用这些序列组合出满足要求的密钥流序列。



(a) 作为有限状态自动机的密钥流生成器　　(b) 密钥流生成器的分解

图 3.23　同步流密码密钥流生成器

密钥流事实上是一个无限长序列。由于一个有限状态机在确定的逻辑连接下，迟早要进入周期状态。因而实际得到的密钥流本质上是一个周期序列。用 s 表示序列 $s_0, s_1, \cdots$，如果存在正整数 p，使得 $s_{i+p} = s_i$，i=0, 1, 2, …，则称序列 s 为周期序列，满足上式的最

小正整数 p 称为序列 s 的周期。

设计流密码系统一般可以从两个方面进行考虑：一是从系统自身的复杂性度量出发，如输出密钥序列的周期、线性复杂度、随机性等；另一方面从抗已知攻击出发，如线性逼近、统计分析等。目前设计密钥流生成器主要有如下准则：

(1) 密钥量足够大：一般密钥长度为 128 比特或 256 比特；

(2) 周期要足够长：一般为 2^{128} 或 2^{256}；

(3) 线性复杂度：一般要求密钥序列的线性复杂度大于周期长度的一半，同时要有好的线性复杂度稳定性、好的线性复杂度曲线等；

(4) 良好的统计特性：如均匀的 0、1 分布和游程分布；

(5) 混淆和扩散：如每个密钥流比特是全部或大多数密钥比特的一个复杂的变换子结构；

(6) 组合函数：良好的密码学性质等。

上述设计准则都是设计安全流密码生成器所必须要考虑的。

一般来说，流密码总是比分组密码要快，通常使用的代码也比分组密码少得多。如最常用的流密码 RC4，比最快的分组密码至少快两倍。RC4 可以只用 30 行代码写成，而大多数分组密码则需要数百行代码。

习 题 三

1. 证明 DES 解密过程是加密过程的逆过程。

2. 编制一个 DES 算法，设密钥为 SECURITY，明文为 NETWORK INFORMATION SECURITY，计算密文，并列出每一轮的中间结果。

3. 设 M′是 M 的逐位补，证明 $Y'=\mathrm{DES}_K(X')$，即明文、密钥取补后加密结果等于原密文的补。

4. AES 算法采用什么结构？与 DES 算法结构有何区别？

5. 如果在 8 比特的 CFB 方式下密文字符的传输中发生 1 比特的差错，这个差错会传播多远？

6. 描述流密码的密钥生成过程。

第四章　公钥密码体制

公钥密码体制的发现是密码学发展史上的一次革命。从古老的手工密码，到机电式密码，直至运用计算机的现代对称密码，这些编码系统虽然越来越复杂，但都建立在基本的替代和置换工具的基础上，而公钥密码体制的编码系统是基于数学中的单向陷门函数。更重要的是，公钥密码体制采用了两个不同的密钥，这对在公开的网络上进行保密通信、密钥分配、数字签名和认证有着深远的影响。

本章从公钥密码学的基本原理出发，具体讲述了三种常用的公开密钥密码系统，RSA密码体制、ElGamal 密码体制和椭圆曲线密码体制。

4.1　公钥密码体制的基本原理

对称密码体制的一个缺点是：在进行保密通信时，发送者与接收者需要使用一个安全的信道来建立会话密钥。这可以通过两种方式解决：已经有了一个共享密钥，通过共享密钥加密传输会话密钥；或者通过一个密钥分配中心获得会话密钥。前者可以用人工等方式传递共享密钥；后者需要通信双方都与密钥分配中心有一个共享密钥，共享密钥同样需要用人工等方式传递。因而密钥分配成本很高，并且依赖于信使或密钥分配中心的可靠性。在某些情况下，甚至无法及时获得建立会话密钥所需的合理安全信道。

1976 年，Diffie 和 Hellman 指出，密码学可以利用 NP 复杂性理论，通过检验找出 NP-完全问题中可以用于密码学的问题。基本方法是将信息通过编码加密在一个 NP-完全问题中，使得用普通的方法破译这种密码等价于解这个 NP-完全问题。但如果使用解密密钥就有简便的解密方法，这就需要所谓的陷门单向函数。如果对一个函数 f 的定义域上的任意一个 x 都容易计算 $f(x)$，但对 f 的值域上的任意一个 y，$f^{-1}(y)$ 都在计算上不可行的，就称 f 是单向函数。如果进一步给定某些辅助信息，计算 $f^{-1}(y)$ 又变得容易，则称 f 是陷门单向函数。为了构造这样的密码算法，秘密的“陷门”必须被嵌在单向函数求逆计算的问题之中。这一陷门信息也就是私人密钥。

单向函数是密码学中的一个重要研究内容，虽然有许多函数被认为是单向的，但遗憾的是目前还没有一个函数被证明是单向的。这里给出一个相信是单向函数的例子：假设 n 是两个大素数 p 和 q 的乘积，分解 n 被认为是一个非常困难的问题(NP-完全问题)。设 b

是一个正整数，定义函数 $f:Z_n \to Z_n$，$f(x)=x^b \bmod n$，f 被认为是一个单向函数。当知道 n 的因子是 p 或 q 时，计算逆是容易的，因而 f 是陷门单向函数。

4.2 RSA 算法

最早实现 Diffie 和 Hellman 的想法的是 Rivest、Shamir 和 Adleman 3 人。他们于 1977 年研制并于 1978 年发表了一种密码算法，一般称为 RSA(3 个研制者姓氏的首字母)算法，实现了公钥密码的基本思想。

4.2.1 RSA 算法描述

RSA 算法正是利用了陷门单向函数的一种可逆模指数运算。它的安全性基于大整数分解因子的困难性。

1. RSA 密码体制的建立

建立一个 RSA 密码体制的过程如下：

(1) 选择两个大素数 p 和 q；

(2) 计算乘积 $n=pq$ 和 $\phi(n)=(p-1)(q-1)$；

(3) 选择大于 1 小于 $\phi(n)$ 的随机整数 e，使得 $\gcd(e,\phi(n))=1$；

(4) 计算 d 使得 $de=1 \bmod \phi(n)$；

(5) 对每一个密钥 $k=(n,p,q,d,e)$，定义加密变换为 $E_k(x)=x^e \bmod n$，解密变换为 $D_k(x)=y^d \bmod n$，这里 $x,y \in Z_n$；

(6) 以 $\{e,n\}$ 为公开密钥，$\{p,q,d\}$ 为私有密钥。

这样就建立了一个明文空间 P 和密文空间 C 为 $P=C=Z_n$，密钥空间为 $K=\{(n,p,q,e,d): n=pq$，p 和 q 是大素数，$1<e,d<\phi(n)$：$de=1 \bmod \phi(n)\}$ 的 RSA 密码体制。

假设接收方 B 公开了他的公钥，而 Alice 希望向他发送一个消息 M，那么发送方 A 计算密文 $C=M^e \bmod n$ 并将 C 传送给 B，B 收到这个密文后通过计算 $M=C^d \bmod n$ 进行解密。

这个算法的基础是欧拉定理：对任意的 $e \in Z_n^*$，有 $e^{\phi(n)}=1 \bmod n$，其中 $Z_n^*=\{x \in Z_n \mid \gcd(n,x)=1\}$，函数 ϕ 是欧拉函数。

由于这里选择的 e 和 d 满足 $ed \equiv 1 \bmod \phi(n)$，因而 ed 具有 $k\phi(n)+1$ 的形式。因此给定满足 $n=pq$ 的两个素数 p 和 q，对于整数 M，$0<M<n$，有：

$$C^d = M^{ed} = M^{k\phi(n)+1} = (M^{\phi(n)})^k \cdot M \equiv M \bmod n$$

2. RSA 算法实例

下面用两个小素数 7 和 17 来建立一个简单的 RSA 算法：

(1) 选择两个素数 $p=7$ 和 $q=17$；

(2) 计算 $n=pq=7\times17=119$，计算 $\phi(n)=(p-1)(q-1)=6\times16=96$；

(3) 选择一个随机整数 $e=5$，它小于 $\phi(n)=96$ 并且与 96 互素；

(4) 求出 d，使得 $de=1\bmod 96$ 且 $d<96$，此处求出 $d=77$，因为 $77\cdot5=385=4\cdot96+1$；

(5) 输入明文 $M=19$，计算 19 模 119 的 5 次幂，$M^e=19^5=66\bmod 119$，传送密文 $C=66$；

(6) 接收密文 66，计算 66 模 119 的 77 次幂；$C^d=66^{77}\equiv19\bmod 119$ 得到明文 19。

当然，在实际应用中 p、q、n、e 和 d 都要大得多。

4.2.2　RSA 算法中的计算技巧

1. 加密和解密

在 RSA 的加密和解密中都涉及到求一个大整数的幂，然后模 n。如果先对整数作幂运算然后再模 n，中间结果将是天文数字，但由于模运算的性质

$$[(a \bmod n)\cdot(b \bmod n)] \bmod n = (ab) \bmod n$$

我们可以在计算中间结果时就取模，即使进行了这样的改进，要计算一个大整数的大整数次幂还是相当慢的(标准的 RSA 要求 p 和 q 是 128 位以上，e 和 d 也接近这样的数量级)。考虑计算 x^{16}，如果用普通的计算乘幂的方法，需要 15 次乘法计算。但可以通过 $4(=\log_2 16)$ 次乘法得到结果，即对每次的部分结果取平方，依次得到 x^2,x^4,x^8,x^{16}。

更一般地，有一种计算模指数的有效算法“平方-乘”算法。如果要计算 M^e，将 e 写成二进制形式 $b_k b_{k-1}\cdots b_0$，这里 $b_k=1$，则有 $e=\sum_{b_i\neq0}2^i$。因此

$$M^e = M^{\left(\sum_{b_i\neq0}2^i\right)} = (((\cdots((M^{b_k})^2 M^{b_{k-1}})^2 M^{b_{k-2}}\cdots)^2 M^{b_2})^2 M^{b_1})^2 M^{b_0}$$

这样可以构造一个计算 $M^e \bmod n$ 的算法：

$d=1$；

for $i=k$ downto 0 do

　　$d=d^2 \bmod n$；

　　if $b_i=1$ then $d=(d\times M)\bmod n$；

return d ;

作为一个练习，可以利用上述算法计算 $7^{560} \bmod 561$ 的结果。

2. 密钥的选取

要实现 RSA 算法，每个设计者还要面对一个比较困难的问题，就是密钥的选取。这包括以下两个方面的内容：

(1) 挑选大素数 p 和 q；

(2) 选择 e 或 d 并且计算另外一个。

首先考虑对素数 p 和 q 的选取。为了防止敌手通过穷举方法发现 p 和 q，p 和 q 必须从很大的集合中选取，因而要求 p 和 q 必须是大数。当然，我们需要有一个有效的方法来找到大素数。

遗憾的是目前我们并没有产生随机大素数的有效的方法。采用的方法是随机选取一个足够大的奇数并检验它是否是素数。如果不是，重复上述过程，直到找到能通过检验的素数为止。我们称上述过程为素性检测。选取一个素数的过程如下：

(1) 随机选一个奇数 n；

(2) 随机选一个整数 $a < n$；

(3) 完成素性检测，如下面介绍的 Miller-Rabin 算法；

(4) 如果 n 没有通过检验，舍弃 n 并转向步骤 1，如果 n 通过了足够多次的检验，就接受它为素数，否则转向步骤 2。

素性检测已经出现了许多种方法，但几乎所有的方法都是概率性的，即给出通过检验的奇数只有一定的概率是素数。这些方法的缺点是通过检验的奇数可能仍然不是素数，但我们可以按照需要做到使得通过检验的奇数不是素数的概率尽可能接近 0。下面介绍一个相当流行的素性检测算法，即所谓的 Miller-Rabin 算法。它依赖于如下数学定理：如果 p 是一个奇素数，则 $x^2 \equiv 1 \bmod p$ 只有平凡解 $x \equiv \pm 1 \bmod p$。这是由于

$$x^2 \equiv 1 \bmod p \Rightarrow p \mid (x^2 - 1) \Rightarrow p \mid (x-1)(x+1)$$
$$\Rightarrow p \mid (x-1) \text{ 或者 } p \mid (x+1) \Rightarrow x \equiv 1 \bmod p \text{ 或者 } x \equiv -1 \bmod p$$

因此当 $x^2 = 1 \bmod n$ 有非平凡解时，可以断定 n 不是一个素数。

选择随机大奇数 n，将 $n-1$ 表示成二进制数 $b_k b_{k-1} \cdots b_0$，随机选择 $a < n-1$。Miller-Rabin 算法如下：

$d = 1$;
for $i = k$ downto 0 do
    $x = d$;

$d = d^2 \bmod n$；
if $d = 1$ and $x \neq 1$ and $x \neq n-1$ then return TRUE;
if $b_i = 1$ then $d = ad \bmod n$；
if $d \neq 1$ then return TRUE;
return FALSE;

注意在该算法中，函数返回 TRUE 表示 n 不是一个素数，而返回 FALSE 则 n 有可能是一个素数，因此在实际检测中要多次返回 FALSE 才能以较大的概率判断 n 是一个素数。

现在考虑选择 e 或 d 并且计算另外一个。比如选择的是 e 并且计算 d，流程如下：选择一个随机整数 e，使得 $\gcd(\phi(n),e)=1$，然后计算 $d = e^{-1} \bmod \phi(n)$。利用“推广的欧几里得算法”可以简化计算。

4.2.3 RSA 算法的安全性

防范 RSA 算法受到强力攻击的措施与其他密码体制一样，即采用大的密钥，以增大密钥空间。因此，e 和 d 的比特数越多越好。但这样密钥的产生和加/解密速度也就越慢。因此，在安全性和有效性之间需要折中。

对 RSA 算法的数学攻击实际上等效于对模数 n 乘积因子的分解。如果 RSA 算法的模数 n 被分解为 p 和 q，则可以得到 $\phi(n)=(p-1)(q-1)$，从而得到 e 模 $\phi(n)$ 的乘法逆 $d = e^{-1} \bmod \phi(n)$。

有趣的是，计算解密指数 a 的任何算法都可以作为分解 n 的一个概率算法的子程序或预言(oracle)。这个结果告诉我们，如果 a 被泄露，将会危及 n 的分解。选择一个新的解密指数已不能保证系统的安全，必须重新选择大素数，以产生新的模数。这也说明，计算加密指数 a 的难度与分解 n 的难度应该相当。

随着计算能力的提高和计算方法的改进，原来被认为不可能分解的大数已能够被成功分解。考虑到算法的安全性和有效性，目前密钥长度介于 1 024 比特和 2 048 比特之间的 RSA 算法是安全的。考虑到现有攻击 RSA 的方法，对素数 p 和 q 的选取还有其他一些限制，如 p 和 q 的长度相差不能太大，$p-1$ 和 $q-1$ 都应该有大的素数因子，$\gcd(p-1,q-1)$ 应该偏小，等等。

4.3 ElGamal 密码体制

ElGamal 算法是在密码协议中有着大量应用的一类公钥密码算法，它的安全性是基于求解离散对数问题(discrete logarithm problem)的困难性。在一个有限域 Z_p(p 是素数)上的求

解离散对数问题可以表述为：给定 Z_p 的一个本原元 α，对 $\beta \in Z_p^*$，寻找惟一的整数 $a(0 \leqslant a \leqslant p-2)$，使得 $\alpha^a \equiv \beta \bmod p$，记为 $a = \log_\alpha \beta$。一般的，如果仔细选择 p，则认为该问题是困难的。目前还没有找到计算离散对数问题的多项式时间算法。为了抗击已知的攻击，p 应该至少是 150 位以上的十进制整数，并且 $p-1$ 至少有一个大的素因子。

要设计一个 ElGamal 密码体制，需要完成如下步骤：

(1) 选择足够大的素数 p 使得求解离散对数问题在 Z_p 上是困难的；

(2) 在 Z_p^* 中选择一个本原元 α；

(3) 随机选择整数 a 使得 $0 \leqslant a \leqslant p-2$，并计算 $\beta = \alpha^a \bmod p$；

(4) 对密钥 $k=(p,\alpha,a,\beta)$，定义加密变换 $e_k(x,r)=(y_1,y_2)$，这里明文 $x \in Z_p^*$，$r(0 \leqslant r \leqslant p-2)$ 是每次加密前随机选择的随机数，$y_1 = \alpha^r \bmod p$，$y_2 = x\beta^r \bmod p$；定义解密变换为 $d_k(y_1,y_2) = y_2(y_1^a)^{-1} \bmod p$，这里密文 $(y_1,y_2) \in Z_p^* \times Z_p^*$；

(5) 以 $\{p,\alpha,\beta\}$ 为公开密钥，a 为私有密钥。

这样就建立了一个明文空间 $P = Z_p^*$、密文空间 $C = Z_p^* \times Z_p^*$、密钥空间 $K = \{(p,\alpha,a,\beta): p$ 是大素数，α 是 Z_p^* 的一个本原元，$0 \leqslant a \leqslant p-2$，$\beta = \alpha^a \bmod p\}$ 的 ElGamal 密码体制。

ElGamal 密码体制是非确定性的，因为每次加密都要选择一个随机数，相同的明文随着加密前随机数 r 的不同而产生不同的密文。这也是 ElGamal 密码体制的工作原理：明文 x 通过乘以 β^r 来隐藏，产生 y_2，值 α^r 也作为密文的一部分进行传送。知道秘密密钥 a 的收方能够从 α^r 中计算出 β^r，并可以通过 y_2 除以 β^r 来去掉隐藏而得到明文 x。

例子：假设 $p = 2\,579$，$\alpha = 2$，$a = 765$，则 $\beta = 2^{765} = \bmod 2\,579 = 949$。如果 A 要发送消息 $x = 1\,299$ 给 B，则 A 选择随机数 $r = 853$ 并计算 $y_1 = 2^{853} \bmod 2\,579 = 435$ 和 $y_2 = 1\,299 \times 949^{853} \bmod 2\,579 = 2\,396$。A 发送密文 $y = (435, 2\,396)$ 给 B，B 收到密文后，计算 $x = 2\,396 \times (435^{765})^{-1} \bmod 2\,579 = 1\,299$，得到 A 加密的明文。

4.4 椭圆曲线密码(ECC)体制

ElGamal 密码体制能够在任何离散对数难处理的有限群中实现。我们已经使用了乘法群 Z_p^*，但其他群也是合适的候选者，如椭圆曲线群。

椭圆曲线在代数学和几何学上已广泛研究了 150 多年之久，有丰富而深厚的理论积累。椭圆曲线密码体制(Elliptic Curve Cryptosystem，ECC)在 1985 年由 Koblitz 和 Miller 提出，

不过一直没有像 RSA 等密码系统一样受到重视。纵观目前的发展趋势，椭圆曲线已经逐渐被采用，很可能是一个重要的发展方向。

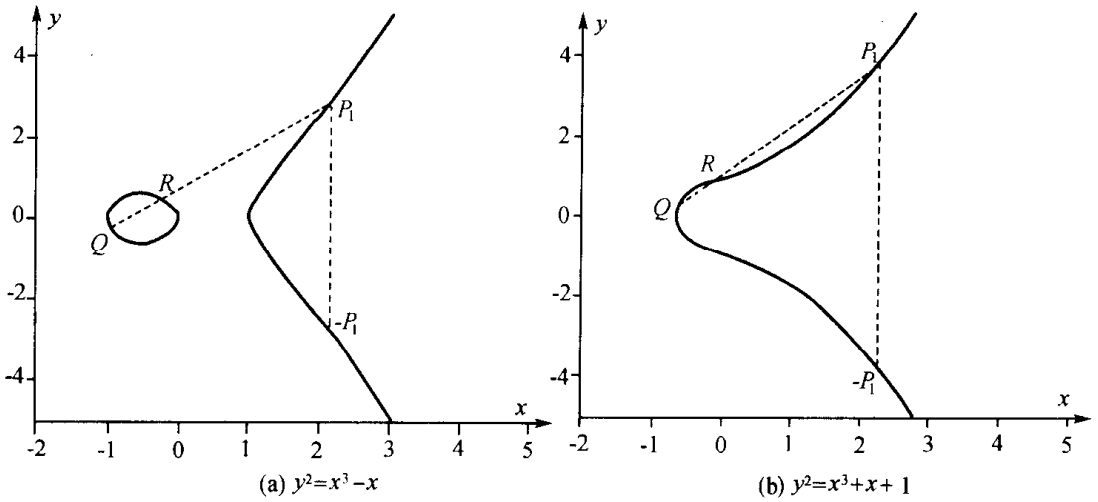


图 4.1　椭圆曲线的例子

4.4.1　一般椭圆曲线

椭圆曲线并非椭圆，这么命名是因为它们是由三次方程描述的，而这些三次方程类似于计算椭圆周长的方程。一般的，描述椭圆曲线方程的形式是

$$y^2 + axy + by = x^3 + cx^2 + dx + e$$

其中 a、b、c、d 和 e 是满足一些简单条件的实数。

一般来说，椭圆曲线还包含了一个特殊的点，即称为无穷远点(Point at Infinity)或零点(Zero Point)的 O(后面将进行讨论)。下面给出两个椭圆曲线的例子，这些公式有时会产生看上去很奇怪的曲线：

对于椭圆曲线上的点可以定义一种形式的加法：如果一个椭圆曲线上的三个点处于一条直线上，那么它们的和为 O。从这个定义可以导出椭圆曲线上点的加法法则。

(1) O 是加法的单位元，因而 $O=-O$；对于椭圆曲线上的任何一点 P，有 $P+O=P$。

(2) 一条与 x 轴垂直的线和曲线相交于两个 x 坐标相同的点 $P_1=(x,y)$ 和 $P_2=(x,-y)$，同时它也和曲线相交于无穷远点，因此 $P_1+P_2+O=O$。因而一个点的负值是与其有着相同 x 坐标和相反的 y 坐标的点，如图 4.1(a)所示。

(3) 要对具有不同 x 坐标的两个点 Q 与 R 进行相加，先在它们之间画一条直线并求出第三个交点 P_1。容易看出这种交点是惟一的。注意到 $Q+R+P_1=O$，有 $Q+R=-P_1$。图 4.1(b)说明了这一过程。特别地，当 $Q=R$ 时，相当于对一个点 Q 加倍，只需画出一条切

线并求出另一个交点 S，那么 $Q+Q=2Q=-S$。

显然，根据定义，此类加法满足交换率和结合率。而一个点的倍乘定义为

$$nP=\underbrace{P+P+P+\cdots+P}_{n\text{个}P}$$

4.4.2 有限域上的椭圆曲线

密码学中关心的是有限域 F 上的椭圆曲线。讨论比较多的是素域 F_p 上的椭圆曲线，这里 p 是一个素数。选择两个小于 p 的非负整数 a 和 b 满足

$$4a^3+27b^2(\text{mod}\,p)\neq 0$$

用 $E_p(a,b)$ 表示如下模 p 的椭圆群中的点(或如下有限域 F_p 上的椭圆曲线的点)，再加上一个无穷远点 O。设 $(x,\ y)$ 是 $E_p(a,\ b)$ 中的点，x 和 y 是小于 p 的非负整数，则有如下椭圆曲线方程：

$$y^2\equiv x^3+ax+b(\text{mod}\,p)$$

如取 $p=23$，$a=b=1$，有 $4\times1^3+27\times1^2(\text{mod}23)=8\neq0$，则 $y^2=x^3+x+1$ 是椭圆曲线。因此 $E_{23}(1,\ 1)$ 是一个模 23 的椭圆群。

注意到 $y^2=x^3+x+1$ 正是图 4.1(b)中的一个方程。我们只关心这个图中满足方程并且在以(0，0)和(p,p)为顶点的正方形中坐标为整数的点，如图 4.2 所示。

产生 $E_{23}(1,\ 1)$ 中点的过程如下：

(1) 对 $x=0,\ 1,\ 2,\ \cdots,\ p-1$，计算 $x^3+x+1(\text{mod}\,p)$；

(2) 对于上一步骤得到的每个结果确定它是否有一个模 p 的平方根，如果没有，则 $E_{23}(1,\ 1)$ 中没有具有与该结果相应的 x 坐标的点。如果有，就有两个平方根 y 和 $p-y$，从而点 (x,y) 和 $(x,p-y)$ 是 $E_{23}(1,\ 1)$ 中的点(特别情况下，如果结果是 0，只有一个点 $(x,0)$)。

(0，1)	(6，4)	(12，19)
(0，22)	(6，19)	(13，7)
(1，7)	(7，11)	(13，16)
(1，16)	(7，12)	(17，3)
(3，10)	(9，7)	(17，20)
(3，13)	(9，16)	(18，3)
(4，0)	(11，3)	(18，20)
(5，4)	(11，20)	(19，5)
(5，19)	(12，4)	(19，18)

图 4.2 椭圆曲线 $E_{23}(1,\ 1)$上的点

$E_p(a,b)$ 上的加法规则：

- $P+O=P$；
- 如果 $P=(x,y)$，则 $P+(x,-y)=O$，点 $(x,-y)$ 是 P 的加法逆元，记为 $-P$；
- 如果 $P=(x_1,y_1)$，$Q=(x_2,y_2)$，并且 $P\neq-Q$，则 $P+Q=(x_3,y_3)$ 由下列规则确定：

$$x_3\equiv\lambda^2-x_1-x_2(\text{mod}\,p)$$

$$y_3 \equiv \lambda(x_1 - x_3) - y_1 (\mathrm{mod}\, p)$$

其中：

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{如果} P \neq Q \\ \dfrac{3x_1^2 - a}{2y_1} & \text{如果} P = Q \end{cases}$$

考虑 $P=(3,\ 10)$，$Q=(9,\ 7)$，则

$$\lambda = \frac{7-10}{9-3} = \frac{-3}{6} = \frac{-1}{2} \equiv 11 \bmod 23$$
$$x_3 = 11^2 - 3 - 9 = 109 \equiv 17 \bmod 23$$
$$y_3 = 11(3-(-6)) - 10 = 89 \equiv 20 \bmod 23$$

因而 $P+Q=(17,\ 20)$。计算 $2P$，

$$\lambda = \frac{3(3^2)+1}{2\times 10} = \frac{5}{20} = \frac{1}{4} \equiv 6 \bmod 23$$
$$x_3 = 6^2 - 3 - 3 = 30 \equiv 7 \bmod 23$$
$$y_3 = 6(3-7) - 10 = -34 \equiv 12 \bmod 23$$

因此 $2P=(7,\ 12)$。

实际中常用的还有在有限域 F_{2^m} 上的椭圆曲线群，其运算规则与 F_p 上的椭圆曲线群稍有不同，这里就不作介绍了。

前面介绍过离散对数的概念，椭圆曲线群上也有相应的概念，而且椭圆曲线群中的离散对数也一样属于难解问题。与通常理解的对数概念不同，由于椭圆曲线群中的运算是加法，加法的倍数对应于原来乘法的指数，因而椭圆曲线群中的离散对数问题是指已知群中的 Q 和 R，求解方程 $R=kQ$ 中 k 值的问题。

对基于 F_{23} 的椭圆群 $y^2 = x^3 + 9x + 17$，求 R=(4，5)对于 Q=(16，5)的离散对数，最直接的方法就是计算 Q 的倍数，直到找到 R。

Q=(16，5)，$2Q$=(20，20)，$3Q$=(14，14)，$4Q$=(19，20)，$5Q$=(13，10)，
$6Q$=(7，3)，$7Q$=(8，7)，$8Q$=(12，17)，$9Q$=(4，5)

因此 R 关于 Q 的离散对数是 9，对于大素数构成的群 F_p，这样计算离散对数是不现实的，事实上现在也没有更好的(非指数级的)算法来解离散对数问题。

4.4.3 椭圆曲线密码算法

基于上面讲的椭圆群上的离散对数问题，可以用椭圆曲线密码(ECC)算法加密，具体方法如下：

首先选择一个点 G 和一个椭圆群 $E_p(a,b)$ 作为参数，用户 A 选择一个私有密钥 n_A 并产生一个公开密钥 $P_A = n_A G$。发送者要加密并发送一个报文 P_m 给 A，可选择一个随机整数 k，并产生由如下点对组成的密文 $C_m = (kG, P_m + kP_A)$。

注意这里使用了 A 的公开密钥 P_A。要解密这个报文，A 用这个点对的第一个点乘以 A 的私有密钥，再从第二个点中减去这个值

$$P_m + kP_A - n_A(kG) = P_m + k(n_A G) - n_A(kG) = P_m$$

发送者通过对 P_m 加上 kP_A 来保护 P_m。除了发送者之外没有人知道 k 的值，因此即便 P_A 是公开密钥也没有人能去掉 kP_A，当然只有知道 n_A 的人才可以去掉 kP_A。攻击者在不知道 n_A 的情况下要想得到报文只能在知道 G 和 kG 的情况下计算出 k，这归结为求解椭圆曲线离散对数问题，是非常困难的。

注意到以上的过程并没有说明怎样将作为字符串(当然可以看成分段的整数)的消息编码嵌入到椭圆群的点中(将明文嵌入椭圆曲线)，实际中的转化方式多种多样，关键的步骤与其正确性证明都涉及到复杂的数学推导，可以参看相关文献。

4.4.4 椭圆曲线密码体制的安全性

椭圆曲线密码体制的安全性依赖于求解椭圆曲线离散对数问题的困难性，即已知椭圆曲线上的点 P 和 kP 计算 k 的困难程度。图 4.3 比较了目前计算椭圆曲线对数和分解大数因子的难度。可以看出，与 RSA 相比 ECC 可以用小得多的密钥取得与 RSA 相同的安全性。另外，在密钥大小相等时，ECC 与 RSA 所需要的计算量相当。因此，在安全性相当的情况下，使用 ECC 比使用 RSA 具有计算上的优势。两者都可以用于加/解密、密钥交换和数字签名。

密钥大小	MIPS 年
150	3.8×10^{10}
205	7.1×10^{18}
234	1.6×10^{28}

(a) 用 Pollard rho 法计算椭圆曲线对数

密钥大小	MIPS 年
512	3×10^{4}
768	2×10^{8}
1024	3×10^{11}
1280	1×10^{14}
1536	3×10^{16}
2048	3×10^{20}

(b) 用通用数域筛对整数进行因子分解

图 4.3 椭圆曲线和 RSA 安全性比较

习　题　四

1．应用 RSA 算法对下列情况进行加/解密，并比较计算结果：

a) p=3, q=11, d=7；M=5；

b) p=5, q=11, e=3；M=9；

c) p=7, q=11, e=17；M=8；

d) p=11, q=13, e=11；M=7；

e) p=17, q=31, e=7；M=2。

2．设截获 e=5，n=35 的用户密文 C=10，请问明文 M 是多少？

3．对于 RSA 算法，已知 e=31，n=3 599，求 d。

4．在 RSA 算法中，如果经过有限的几次重复编码之后又得到明文，那么，可能的原因是什么？

5．对于椭圆曲线 $y=x^3+x+6$，考虑点 G=(2,7)，计算 $2G$ 到 $13G$ 的各倍数值。

6．对于椭圆曲线 $y=x^3+x+6$，考虑点 G=(2,7)，已知秘密密钥 n=7，计算：

a) 公开密钥 P_b；

b) 已知明文 P_m=(10,9)，并选择随机数 k=3，确定密文 C_m。

第五章　消息认证与数字签名

公钥密码体制最重要的一种应用是数字签名。数字签名通常需要与散列函数结合起来使用。本章先简要介绍信息认证，然后介绍散列函数，最后介绍数字签名。

5.1 信 息 认 证

前面介绍的对称密码和公钥密码体制，都是围绕着信息的保密性，即防止第三方获取明文消息而展开的，但信息的完整性和抗否认性也是信息安全的重要内容。保证信息的完整性和抗否认性主要通过信息认证和数字签名来实现。

一般说来，产生认证符的方法可以分为以下三类：

(1) 信息加密：将明文加密后以密文作为认证符；

(2) 消息认证码(MAC)：用一个密钥控制的公开函数作用后，产生固定长度的数值作为认证符，也称为密码校验和。

(3) 散列函数：定义一个函数将任意长度的消息映射为定长的散列值，以散列值作为认证符。

5.1.1 加密认证

信息加密能够提供一种认证措施，具体又分为对称密码体制加密认证和公钥密码体制加密认证。

1．对称密码体制加密认证

设想发送者 A 使用只有他与接收者 B 知道的密钥 K 加密信息并发送给接收者。因为 A 是除 B 以外惟一拥有密钥的一方，而攻击者不知道如何通过改变密文来产生明文中所期望的变化，所以，B 能够知道解密得到的明文是否被改变。这样，对称加密就提供了消息认证功能。

现在考虑如下情况：给定解密函数 D 和密钥 K，接收者接受输入 X 并产生输出 $Y=D_K(X)$，如果 X 是合法信息 M 的密文，则 Y 是信息 M 的明文，否则，Y 将是毫无意义的二进制比特序列。接收方需要某些自动化措施，以确定 Y 是否是合法的明文。

假定信息 M 的明文可以是任意比特的组合。在这种情况下，接收方将没有自动化的方法来确定收到的 X 是否是合法信息的密文。因此，需要从所有可能的比特模式集的一个子集中来考虑合法的信息，任何可疑的密文不可能产生合法的明文。如考虑在 10^6 种组合中只有一种是合法的，则从中随机选择一个比特组合作为密文能够产生合法的明文的概率是 10^{-6}。

为了用自动化的方法来确定收到的 Y 是否合法信息的密文，可以采用的方法之一是强制明文有某种结构。这种结构易于识别，但不能复制并且不求助于加密。例如，在加密前对消息附加检错码。

2. 公钥密码体制加密认证

使用公开密钥加密明文只能提供保密而不能提供认证。因为任何人都可以得到公钥。为了提供认证，发送者 A 用私钥对信息的明文进行加密，任意接收者都可以用 A 的公钥解密。这种方式提供的认证措施与对称密码体制加密的情形在原理上是相同的。与前面一样，在明文中也要求有某种内部结构，因此，接收者能够识别正常的明文和随机的比特串。

采用这样的结构既可提供认证，也可提供数字签名。因为只有 A 才能够产生密文，其他任何一方都不能产生密文。从效果上看，A 已经用私钥对信息的明文进行了签名。

应当注意，只用私钥加密不能提供保密性。因为任何人只要拥有 A 的公开密钥，就能够对该密文进行解密。

5.1.2　消息认证码

消息认证码(MAC)(或称密码检验和)是在密钥的控制下将任意长的消息映射到一个简短的定长数据分组，并将它附加在消息后。设 M 是变长的消息，K 是仅由收发双方共享的密钥，则 M 的 MAC 由如下的函数 C 生成

$$\mathrm{MAC}=C_K(M)$$

这里 $C_K(M)$ 是定长的。发送者每次将 MAC 附加到消息中。接收者通过重新计算 MAC 来对消息进行认证。如果收到的 MAC 与计算得出的 MAC 相同，则接收者可以认为：

(1) 消息未被更改过。因为任意更改消息而没有更改 MAC 的行为，都将导致收到的 MAC 与用更改过的消息计算出的 MAC 不相同。而且，由于使用的密钥只有收发双方知道，其他方无法更改 MAC 使之与更改后的消息对应。

(2) 消息来自其他共享密钥的发送者。由于使用的密钥只有收发双方知道，其他方无法产生对应的 MAC。

MAC 函数类似于加密函数，主要区别在于 MAC 函数不需要可逆而加密函数必须是可逆的。因此，认证函数比加密函数更不易破解。

应当注意，由于收发双方共享相同的密钥，上述过程只能提供认证而不能提供保密，也不能提供数字签名。

前面谈到对称密码体制的加密能够提供认证，为什么还要使用独立的消息认证码呢？主要有如下一些理由：

(1) 一些应用要求将相同的消息对许多终端进行广播，而仅使用一个终端负责消息的认证，这种方法既经济又实用。负责认证的终端有相应的密钥，并执行认证操作。如果认证不正确，其他终端将收到它发来的警告

(2) 接收方有繁重的任务，无法负担大量的解密任务，因此仅进行有选择的认证，对消息进行随机检查。

(3) 有一些应用，只关心信息的完整性而不需要保密性。

(4) 认证与保密的分离能够提供结构上的灵活性。

(5) 有些应用场合期望在超过接收时间后继续延长保护期限，同时允许处理消息的内容。如果使用了加密，解密后保护就失效了，这样，消息只能在传输过程中得到完整性保护，但在目标系统中却办不到。

5.2 散列(Hash)函数

散列函数(又称杂凑函数)是对不定长的输入产生定长输出的一种特殊函数：

$$h = H(M)$$

其中 M 是变长的消息，$h = H(M)$ 是定长的散列值(或称为消息摘要)。散列函数 H 是公开的，散列值在信源处被附加在消息上，接收方通过重新计算散列值来确认消息未被篡改。由于函数本身公开，传送过程中对散列值需要另外的加密保护(如果没有对散列值的保护，篡改者可以在修改消息的同时修改散列值，从而使散列值的认证功能失效)。

5.2.1 散列函数的性质

散列函数的目的是为文件、消息或其他的分组数据产生“指纹”。用于消息认证的散列函数 H 必须具有如下性质：

(1) H 能用于任何大小的数据分组，都能产生定长的输出。

(2) 对于任何给定的 x，$H(x)$ 要相对易于计算。

(3) 对任何给定的散列码 h，寻找 x 使得 $H(x) = h$ 在计算上不可行(单向性)。

(4) 对任何给定的分组 x，寻找不等于 x 的 y，使得 $H(x) = H(y)$ 在计算上不可行(弱抗冲突)。

(5) 寻找任何的(x,y)，使得$H(x)=H(y)$在计算上不可行(强抗冲突)。

注意到前两个性质使得散列函数用于消息认证成为可能。第 2 和第 3 个性质保证 H 的单向性：给定消息产生散列值很简单，反过来由散列值产生消息在计算上不可行，这保证了攻击者无法通过散列值恢复消息。第 4 个性质保证了攻击者无法在不修改散列值的情况下替换消息而不被察觉。第 5 个性质比第 4 个性质更强，保证了一种被称为生日攻击的方法无法奏效。

散列码不同的使用方式可以提供不同要求的消息认证。

(1) 使用对称密码体制对附加了散列码的消息进行加密。这种方式与用对称密码体制加密再附加检错码的消息在结构上是一致的。认证的原理也相同，而且这种方式也提供保密性。

(2) 使用对称密码体制仅对附加的散列码进行加密。在这种方式中，如果将散列函数与加密函数合并成一个整体函数实际上就是一个 MAC 函数。

(3) 使用公钥密码体制，但发送方的私有密钥仅对散列码进行加密。这种方式与第二种方式一样不仅提供认证，而且还提供数字签名。

(4) 发送者将消息 M 与通信各方共享的一个秘密值 S 串接，然后计算出散列值，并将散列值附在消息 M 后发送出去。由于秘密值 S 并不发送，攻击者将无法产生假消息。

5.2.2　散列函数的结构

为了对不定长的输入产生定长的输出，并且最后的结果要与所有的字节相关，大多数安全的散列函数都采用了分块填充链接的模式，其结构是迭代型的。这种散列函数模型最早由 Merkle 于 1989 年提出，在 Ron Rivest 1990 年提出的 MD_4 中也采用了这种模型。下面介绍这种模型的一般结构。

这种散列函数将输入数据分为 L 个固定长度为 b 比特的分组。输入数据除了消息和附加的填充数据外，还附加了消息的长度值。附加的这个长度值将增加攻击者攻击的难度。攻击者要么找出两个具有相同长度的消息，使得它们加上各自长度值后散列值相同；要么找出两个不同长度的消息，其加上各自的长度值后也必须散列成相同的值。

散列算法中重复使用一个压缩函数 f，f 产生一个 n 比特的输出。f 有两个输入：一个是前一步的 n 比特输出，称为链接变量；另一个是 b 比特的分组。算法开始时，要指定一个初始的链接变量值 V_1。最终的链接变量值便是散列值。通常有 $b>n$，因此称 f 为压缩函数。该散列算法可以表达如下：

$$
\begin{aligned}
&CV_0 = V_1 \\
&CV_i = f(CV_{i-1}, Y_{i-1}),\quad 1 \leqslant i \leqslant L \\
&h = H(M) = CV_L
\end{aligned}
$$

这里的 M 由分组 $Y_0,Y_1,\cdots,Y_{L-1}$ 组成，如图 5.1 所示。

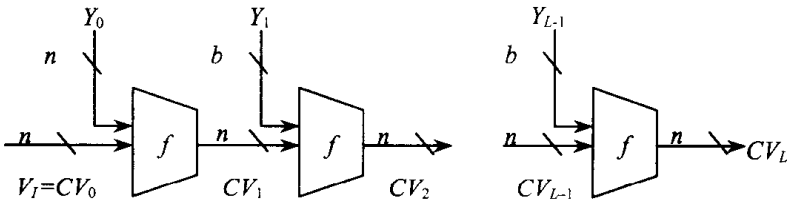


图 5.1 迭代型散列函数的结构

已经证明如果压缩函数是无碰撞的，则上述方法得到的 Hash 函数也是无碰撞的。因此，Hash 函数的核心技术是设计无碰撞的压缩函数。同样，攻击者对算法的攻击重点也是对 f 内部结构的分析。与分组密码一样，f 也是由若干轮处理过程组成的，因而对 f 的分析需要通过对各轮之间比特模式的分析来进行，常常需要先找出 f 的碰撞。

由于 f 是压缩函数，因而一定存在碰撞。这就要求在设计 f 时尽量使得找出 f 的碰撞在计算上是不可行的。

5.2.3 安全散列算法(SHA)

Ron Rivest 于 1990 年提出了一个称为 MD_4 的散列函数。它的设计没有基于任何假设和密码体制，这种直接构造的方法受到人们的广泛关注。不久，它的一些缺点也被提出。为了增强安全性和克服 MD_4 的缺陷，Rivest 于 1991 年对 MD_4 作了六点改进，并将改进后的算法称为 MD_5。MD_5 曾经是使用最普遍的安全散列算法。然而，随着对 MD_5 分析的深入，从密码分析和强力攻击的角度来看，MD_5 也被认为是易受攻击的。因此，有必要用一个具有更长散列码和更能抗击已知密码分析攻击的散列函数来代替现在流行的 MD_5。目前，已经有两个散列码长度为 160 比特的替代者 SHA-1 和 RIPEMD-160。本书只介绍 SHA-1。

安全散列算法 SHA(Secure Hash Algorithm)是由美国国家标准和技术协会(NIST)提出的，并作为联邦信息处理标准(FIPS PUB 180)在 1993 年公布。1995 年又发布了一个修订版 FIPS PUB 180-1，通常称为 SHA-1。SHA 是基于 MD_4 算法的。

1. SHA-1 描述

如图 5.2 所示，SHA-1 算法的输入为不超过 2^{64} 比特长的任意消息，输出为一个 160 比特长的消息摘要。如果输入按 512 比特长的分组进行处理，其处理过程如下：

(1) 对消息进行填充：在原始的消息后面附加填充比特串，使得数据长度(比特数)与 448 模 512 同余。即使得填充后的数据长度为 512 的整数倍减去 64。填充是必须的，即使消息

长度已满足要求，仍然需要填充。如消息长度为 448 比特，则需要填充 512 比特，使其长度变为 960。因此，填充的比特数从 1 到 512。规定填充比特串的第 1 位是 1，其余各位均是 0。

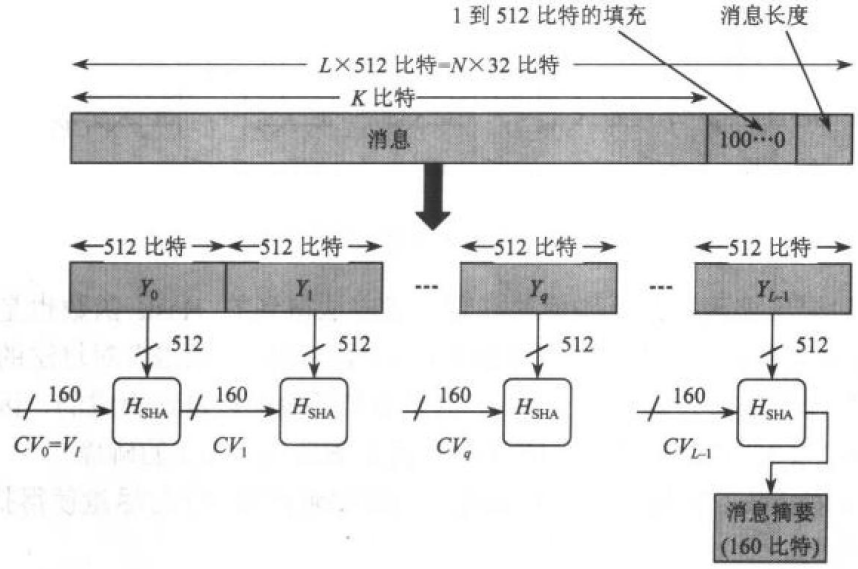


图 5.2　SHA-1 的框图

(2) 附加消息的长度：用 64 比特的二进制数表示原始消息的长度，再将所得的 64 比特数据附加在第 1 步所得的数据后面(高位字节优先)。

(3) 初始化 MD 缓存：用一个 160 比特的缓存来存放该散列函数的中间及最终结果。该缓存可以表示为 5 个 32 比特的寄存器 (A,B,C,D,E)。这些寄存器被初始化为以下 32 比特长的整数(十六进制表示)：

A=67452301

B=EFCDAB89

C=98BADCFE

D=10325476

E=C3D2E1F0

注意这些值要以高位字节优先的方式存储，因而初始化的值(十六进制表示)存储如下：

字 A=67 45 23 01

字 B=EF CD AB 89

字 C=98 BA DC FE

字 D=10 32 54 76

字 E=C3 D2 E1 F0

(4) 处理 512 比特分组序列：核心是一个包含 4 个循环的模块，每个循环由 20 个处理

步骤组成，如图 5.3 所示。4 个循环结构相似，但使用不同的原始逻辑函数，分别为 f_1、f_2、f_3 和 f_4。

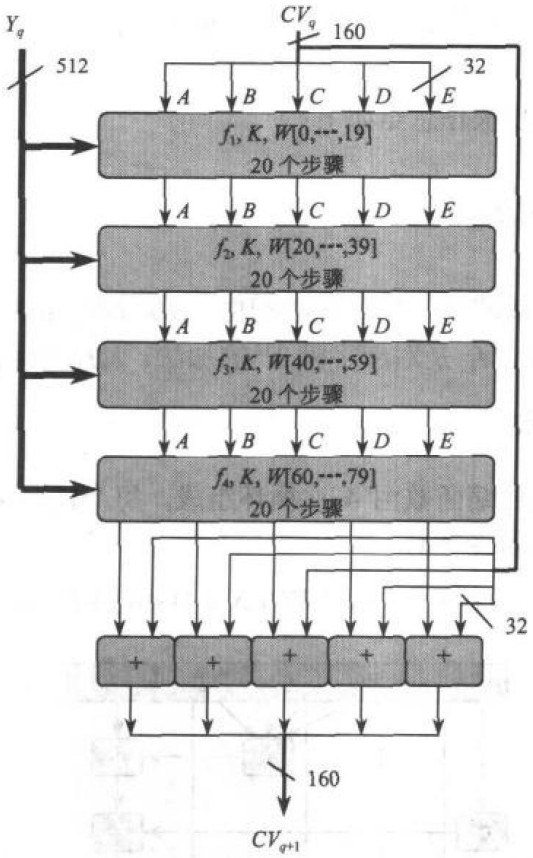


图 5.3 SHA-1 的一个 512 比特分组处理过程

每个循环以当前处理的 512 比特分组 Y_q 和 160 比特的缓存值 *ABCDE* 作为输入，然后更新缓存的内容。每个循环也需要使用一个额外的常数值 K_t，其中 $0 \leqslant t \leqslant 79$。$K_t$ 的值用十六进制表示如表 5.1 所示。

表 5.1 SHA 的加法常量 K_t

迭 代 次 数	常量 K_t(十六进制)	常量 K_t(十进制)
$0 \leqslant t \leqslant 19$	*5A827999*	$\lfloor 2^{30} \times \sqrt{2} \rfloor$
$20 \leqslant t \leqslant 39$	*6ED9EBA1*	$\lfloor 2^{30} \times \sqrt{3} \rfloor$
$40 \leqslant t \leqslant 59$	*8F1BBBCDC*	$\lfloor 2^{30} \times \sqrt{5} \rfloor$
$60 \leqslant t \leqslant 79$	*CA62C1D6*	$\lfloor 2^{30} \times \sqrt{10} \rfloor$

第四个循环的输出加到第一个循环的输入(CV_q)上产生 CV_{q+1}，这里的相加是缓存中的 5 个字分别与 CV_q 中对应的 5 个字以模 2^{32} 相加。

(5) 输出：所有 L 个 512 比特的分组处理完成之后，第 L 阶段产生的输出便是 160 比特的报文摘要。

SHA-1 的第三到第五步操作总结如下：

$CV_0=V_I$;

$CV_q=SUM_{32}(CV_q, ABCDE_q)$;

$MD=CV_L$。

其中，V_I 是第三步定义的缓冲区 $ABCDE$ 的初值，$ABCDE_q$ 是第 q 个分组经过四轮处理后的输出，L 是分组个数，SUM_{32} 是对应字模 2^{32} 的加法，MD 是最终摘要值。

2. SHA-1 的压缩函数

上面说过，SHA-1 的压缩函数由 4 次循环组成，每个循环又由 20 个处理步骤组成，每个处理步骤的形式为(参看图 5.4)：

$$A,B,C,D,E \leftarrow (E+f(t,B,C,D)+S^5(A)+W_t+K_t),A,S^{30}(B),C,D$$

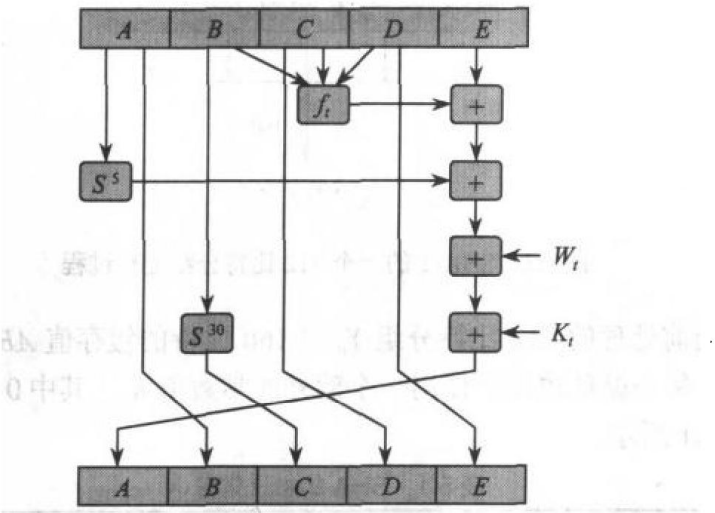


图 5.4　SHA-1 的压缩函数中一次迭代

其中，A、B、C、D、E 是上面提到的缓存中的 5 个字，$f(t,B,C,D)$ 是步骤 t 的原始逻辑函数，S^k 为 32 比特的参数循环左移 k 位，W_t 是由当前 512 比特分组导出的一个 32 比特字(导出方式见表 5.2 所示)，K_t 如表 5.1 的定义。

每个基本逻辑函数有 3 个 32 比特字的输入并产生一个 32 比特字的输出。每个函数执

行一组按位的逻辑操作，即第 n 位的输出是这三个输入中第 n 个比特的一个函数。函数的定义如表 5.2 所示。

表 5.2 SHA-1 中基本逻辑函数的定义

步 骤	函 数 名	函 数 值
$0 \leqslant t \leqslant 19$	$f_1 = f(t,B,C,D)$	$(B \wedge C) \vee (\overline{B} \wedge D)$
$20 \leqslant t \leqslant 39$	$f_2 = f(t,B,C,D)$	$B \oplus C \oplus D$
$40 \leqslant t \leqslant 59$	$f_3 = f(t,B,C,D)$	$(B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$
$60 \leqslant t \leqslant 79$	$f_4 = f(t,B,C,D)$	$B \oplus C \oplus D$

逻辑运算与、或、非、异或分别用符号∧，∨，—，⊕表示。函数的真值如表 5.3 所示。

表 5.3 SHA-1 基本逻辑函数真值表

B	C	D	f_1	f_2	f_3	f_4
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	0	1	0	1
0	1	1	1	0	1	0
1	0	0	0	1	0	1
1	0	1	0	0	1	0
1	1	0	1	0	1	0
1	1	1	1	1	1	1

32 比特字 W_t 的值是通过如下过程从 512 比特的分组中导出的：W_t 的前 16 个字的值直接取自当前分组中前 16 个字的值。余下的字定义如下：

$$W_t = S^1(W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3})$$

因此，在前 16 步处理中，W_t 的值等于分组中对应字的值。余下的 64 步中 W_t 的值由 4 个前面的 W_t 值异或之后再循环左移一位得出，如图 5.5 所示。SHA-1 将 16 个分组字扩展为 80 个字以供压缩函数使用，这将使得碰撞的寻找非常困难。

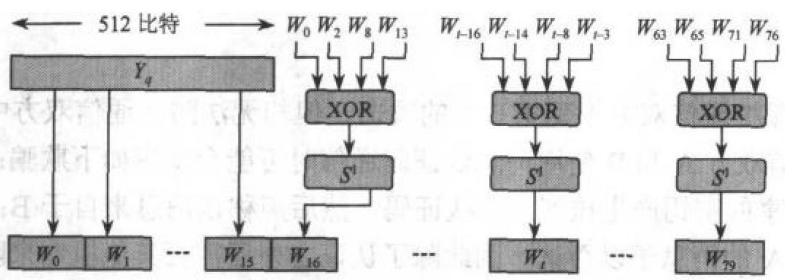


图 5.5 SHA-1 处理单个分组时所需 80 个字的产生过程

3. 与 MD_5 和 RIPEMD-160 的比较

SHA-1 在许多方面都与 MD_5 和 RIPEMD-160 相似，因为这 3 个算法都是由 MD_4 导出的。表 5.4 展示了它们的异同。

表 5.4　SHA-1 与 MD_5 和 RIPEMD-160 的对比

算法 / 设计指标	MD_5	SHA-1	RIPEMD-160
消息摘要长度	128 比特	160 比特	160 比特
分组长度	512 比特	512 比特	512 比特
步骤数	64(4 个 16 步的循环)	80(4 个 20 步的循环)	160(5 对 16 步的循环)
消息最大长度	∞	2^{64}-1 比特	2^{64}-1 比特
基本逻辑函数个数	4	4	5
加法常数个数	64	4	9
数据存储方式	底位字节优先	高位字节优先	底位字节优先

从上面的设计指标和目前的分析结果来看，它们之间的优缺点比较如下：

(1) 抗强力攻击的能力：对于弱碰撞攻击，这 3 个算法都是无懈可击的。MD_5 很容易遭遇强碰撞的生日攻击，而 SHA-1 和 RIPEMD-160 目前在这方面是安全的。

(2) 抗密码分析攻击的能力：对 MD_5 的密码分析已经取得了很大的进展。RIPEMD-160 设计时就考虑了如何对抗已知密码分析的攻击。SHA-1 也有很高的抗密码分析攻击的能力。一般来说，RIPEMD-160 应该比 SHA-1 有更强的抗密码分析攻击的能力。

(3) 计算速度：3 个算法的主要运算都是模 2^{32} 加法和按位逻辑运算，因而都易于在 32 位的结构上实现，但 SHA-1 和 RIPEMD-160 的迭代次数较多，复杂性较高，因此速度较 MD_5 慢些。

(4) 存储方式：三个算法在低位字节优先与高位字节优先方面都没有明显的优势。

5.3　数字签名体制

消息认证能保护通信双方不受第三方的攻击，但却无法防止通信双方中的一方对另一方的欺骗。如通信双方 A 和 B 使用消息认证码通信时可能会发生如下欺骗：A 伪造一个消息并使用与 B 共享的密钥产生该消息的认证码，然后声称该消息来自于 B；同样，B 也可以对自己发送给 A 的消息予以否认。因此除了认证之外还需要其他机制来防止通信双方的抵赖行为，最常见的解决方案就是数字签名。

5.3.1 数字签名原理

数字签名体制是以电子签名形式存储消息的方法，所签名的消息能够在通信网络中传输。数字签名与传统的手写签名有如下几点不同：

(1) 签名：手写签名是被签文件的物理组成部分；而数字签名不是被签消息的物理部分，因而需要将签名连接到被签消息上。

(2) 验证：手写签名是通过将它与真实的签名进行比较来验证；而数字签名是利用已经公开的验证算法来验证。

(3) 数字签名消息的复制品与其本身是一样的；而手写签名纸质文件的复制品与原品是不同的。

与手写签名类似，一个数字签名至少应满足以下 3 个基本条件：

(1) 签名者不能否认自己的签名；

(2) 接收者能够验证签名，而其他任何人都不能伪造签名；

(3) 当关于签名的真伪发生争执时，存在一个仲裁机构或第三方能够解决争执。

在这些性质的基础上，结合手写签名与数字签名的区别，可以归纳出如下数字签名的需求：

(1) 依赖性：数字签名必须依赖于要签名消息的比特模式(类似于笔迹签名与被签文件的不可分离性)；

(2) 惟一性：数字签名必须使用对签名者来说是惟一的信息，以防伪造和否认(类似于笔迹签名的独特性)；

(3) 可验证：数字签名必须是在算法上可验证的。

(4) 抗伪造：伪造一个数字签名在计算上不可行，无论是通过以后的数字签名来构造新的消息，还是对给定的消息构造一个虚假的数字签名(类似于笔迹签名不可模仿性)；

(5) 可用性：数字签名的产生、识别和验证必须相对简单，并且可以将其备份在存储设备上(显然签名不能太长)。

目前已经提出了许多种数字签名体制，但大致可以分成两类：直接数字签名和需要仲裁的数字签名。

- 直接数字签名

直接数字签名仅涉及通信方。它假定接收方知道发送方的公开密钥。数字签名通过使用发送方的私有密钥对整个消息进行加密或使用发送方的私有密钥对消息的散列码进行加密来产生。

目前，所有的直接数字签名体制都有一个共同的弱点：方案的有效性依赖于发送方私有密钥的安全性。如果发送方随后想否认发送过某个签名消息，发送方可以声称用来签名

的私钥丢失或被盗用，并有人伪造了他的签名。如果 A 的私有密钥真的被盗，盗用者便能够发送带有 A 签名的消息，并附加上盗用前的任何时刻作为时间戳。

- 需仲裁的数字签名

为了解决直接数字签名中存在的问题，我们可以引入仲裁者。需要仲裁的数字签名体制一般流程如下：发送方 A 对消息签名后，将附有签名的消息发送给仲裁者 C，C 对其验证后，连同一个通过验证的证明发送给接收方 B。在这个方案中，A 无法对自己发出的消息予以否认，但仲裁者必须是得到所有用户信任的负责任者。

5.3.2　RSA 数字签名体制

用 RSA 实现数字签名的方法中，将要签名的消息作为一个散列函数的输入，产生一个定长的安全散列码。使用签名者的私有密钥对这个散列码进行加密就形成签名，然后，将签名附在消息后。验证者根据消息产生一个散列码，同时使用签名者的公开密钥对签名进行解密。如果计算得出的散列码与解密后的签名匹配，那么签名就是有效的。因为只有签名者知道私有密钥，所以只有签名者才能产生有效的签名。RSA 的签名与验证过程如图 5.6 所示。

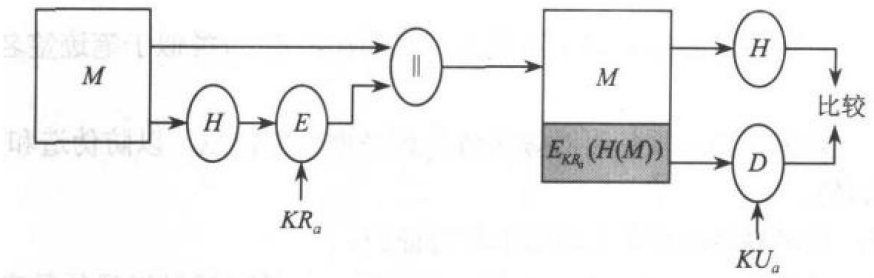


图 5.6　RSA 数字签名体制

对照数字签名的各项要求：

(1) 散列函数取得消息的信息摘要，从而使对之进行加密而产生的签名依赖于消息；

(2) RSA 私钥的保密性使得签名是惟一的；

(3) 信息摘要的字节数很少(如 SHA 的 160 字节)，因而 Hash 函数输出产生签名相对简单，同样地，签名的识别与证实也相对简单。

(4) 散列函数的单向性与抗冲突性以及 RSA 私钥的保密性，使得伪造数字签名不可行。

可以看出，散列函数在数字签名中发挥了产生消息摘要、减少签名与认证工作量的作用。

5.3.3 ElGamal 数字签名体制

ElGamal 于 1985 年基于离散对数问题提出了一个数字签名体制，通常称为 ElGamal 数字签名体制。它与 ElGamal 密码体制一样也是非确定性的，即一个确定的消息可能有许多合法的签名。这里介绍一个更一般的数字签名体制，称它为 ElGamal 型数字签名体制。

设计一个 ElGamal 型数字签名体制的过程如下：

(1) 选择足够大的素数 p 使得求解离散对数问题在 Z_p 上是困难的，q 是 $p-1$ 的大素因子或者 $q=p$。

(2) 随机选择 Z_p^* 中一个阶为 q 的元素 α；当 $q=p$ 时，在 Z_p^* 中随机选择一个本原元 α。

(3) 随机选择整数 a 使得 $0\leqslant a\leqslant p-2$，并计算 $\beta=\alpha^a \bmod p$。

(4) 对每个密钥 $k=(p,q,\alpha,a,\beta)$，定义签名变换 $Sig_k(x,r)=(\gamma,\delta)$，这里的 r 是每次签名前随机选择的随机数，$\gamma=(\alpha^r \bmod p)\bmod q$。$\delta$ 的取值如下：当 $q<p$ 时，δ 满足方程 $rf(\gamma,x,\delta)+ag(\gamma,x,\delta)+h(\gamma,x,\delta)\equiv 0 \bmod p$；当 $q=p$ 时，δ 满足方程 $rf(\gamma,x,\delta)+ag(\gamma,x,\delta)+h(\gamma,x,\delta)\equiv 0 \bmod (p-1)$，这里的 f，g，h 是公开知道的函数，并且使得从方程中求解 δ 是容易的。

(5) 对 $x\in Z_p^*,\gamma\in Z_q^*,\delta\in Z_q$，定义验证函数为 $Ver(x,\gamma,\delta)=T$，当且仅当 $\gamma^{f(\gamma,x,\delta)}\cdot\beta^{g(\gamma,x,\delta)}\cdot\alpha^{h(\gamma,x,\delta)}\equiv(1 \bmod p)\bmod q$。

(6) 以 $\{p,q,\alpha,\beta\}$ 为公开密钥，a 为私有密钥。

这样就建立了 ElGamal 型数字签名体制：其消息空间 $P=Z_p^*$，签名空间 $S=Z_q^*\times Z_q(q<p)$ 或 $S=Z_p^*\times Z_{p-1}(q=p)$，密钥空间为 $K=\{(p,q,\alpha,a,\beta):\beta=\alpha^a \bmod p\}(q<p)$ 或 $K=\{(p,\alpha,a,\beta):\beta=\alpha^a \bmod p\}(q=p)$。其中 p 是大素数，$q=p$ 或者 q 是 $p-1$ 的大素因子，α 是 Z_p^* 的一个本原元(取 $q=p$)或 Z_p^* 中一个阶为 q 的元素(取 $q<p$)，$0\leqslant a\leqslant p-2$，$\beta=\alpha^a \bmod p$。

如果签名是正确构造的，则验证将是成功的，因为

$$(\gamma^{f(\gamma,x,\delta)}\cdot\beta^{g(\gamma,x,\delta)}\cdot\alpha^{h(\gamma,x,\delta)} \bmod p)\bmod q=(\alpha^{rf(\gamma,x,\delta)}\cdot\alpha^{ag(\gamma,x,\delta)}\cdot\alpha^{h(\gamma,x,\delta)} \bmod p)\bmod q=1$$

当 f，g，h 取不同的函数时，就能得到不同的数字签名体制。

在上面的签名体制中，取 $q=p$，得到的就是 ElGamal 数字签名体制。此时，签名算法为 $Sig_k(x,r)=(\gamma,\delta)$，$\gamma=\alpha^r \bmod p$，$\delta=(x-a\gamma)r^{-1} \bmod (p-1)$；而验证算法为 $Ver(x,\gamma,\delta)=T$，当且仅当 $\gamma^\delta\cdot\beta^\gamma\equiv\alpha^x \bmod p$，$x,\gamma\in Z_p^*,\delta\in Z_{p-1}$。

5.3.4 数字签名标准 DSS

DSS 数字签名标准由美国国家标准技术研究所(NIST)于 1991 年提出，在 1994 年公布

的联邦信息处理标准 FIPS PUB 186 中采用为数字签名标准，它利用了前面介绍的安全散列算法(SHA)并提出了一种新的数字签名技术，即数字签名算法(DSA)。

在 DSS 数字签名方法中，签名方先利用安全散列算法产生消息的信息摘要，然后将信息摘要和一个专用于该签名的随机数 k 作为签名函数的输入，该签名函数还依赖于签名方的私有密钥(KR_a)和一个全局公开密钥(KU_G，事实上是一组相关联的参数)。签名由两个分量组成，记为 s 和 r。验证方计算消息的散列码，该散列码和签名做为验证函数的输入。验证函数同时还依赖于全局公开密钥和与签名方的私钥配对的验证方的公钥。如果签名是有效的，验证函数的输出等于签名分量 r，其工作流程如图 5.7 所示。

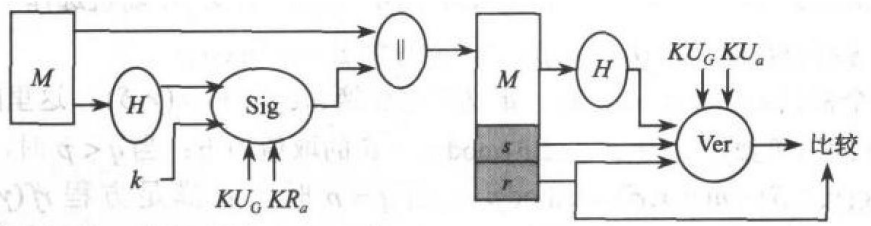


图 5.7　DSS 数字签名体制

签名算法 DSA 是基于 Z_p^* 上求解离散对数问题的难度，是 ElGamal 型数字签名体制的变形。在 ElGamal 型数字签名体制中，当 $q<p$，$f(\gamma,x,\delta)=\delta$，$g(\gamma,x,\delta)=-\gamma$，$h(\gamma,x,\delta)=-H(x)$ 时，就成为数字签名算法 DSA，其中 H 是一个散列算法。在 DSA 中，称公钥 $\{p,q,\alpha,\beta\}$ 中的 $\{p,q,\alpha\}$ 为全局公钥分量，β 为用户公钥。

设计一个 DSA 算法的步骤如下：

(1) 全局公钥的选择：首先选择一个 160 比特的素数 q，接着选择一个长度在 512 到 1 024 比特之间的素数 p，使得 $p-1$ 能被 q 整除，最后选择 $g=h^{(p-1)/q} \bmod p$，其中 h 是大于 1 小于 $p-1$ 的整数，从而使 g 大于 1。

(2) 选择用户私钥公钥对：选择 1 到 $p-1$ 之间的随机数或者伪随机数 x 作为用户私钥，计算 $y=g^x \bmod p$ 作为公钥。

(3) 生成签名：用户选择随机整数 k，对消息 M，计算两个分量 r 与 s 产生签名 (r,s)：

$$r=f_2(k,p,q,g)=(g^k \bmod p) \bmod q$$

$$s=f_1(H(M),k,x,r,q)=(k^{-1}(H(M)+xr) \bmod q$$

(4) 验证签名：接收方先根据收到的消息 M'、签名 (r',s')、公钥 y、p、q、g 等值计算

$$w=f_3(s',q)=(s')^{-1} \bmod q$$

$$v=f_4(y,q,g,H(M'),w,r')=((g^{(H(M')w) \bmod q} y^{r'w \bmod q} \bmod p) \bmod q$$

将 v 与 r 进行比较，若相等则接受签名。

签名和验证的过程如图 5.8 所示。

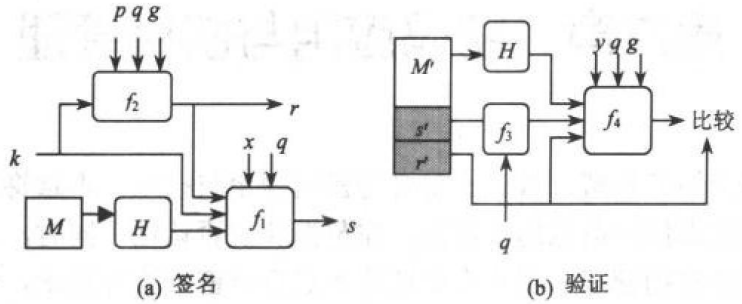


图 5.8 DSS 签名和验证

习 题 五

1. 散列函数应该满足哪些性质？
2. 给出一种利用 DES 构造散列函数的算法。
3. 编制一个程序，用 SHA-1 计算自选文件的散列值。
4. 比较 MD5 与 SHA-1。
5. 比较 DSA 和 RSA 算法。
6. 在 DSA 中，如果计算结果 $s=0$，则必须重新选 k，重新计算，为什么？

第六章　密码应用与密钥管理

前面几章分别介绍了加密、认证和签名等密码理论与技术，本章将论述如何在信息网络环境中使用这些技术保护信息的机密性、完整性和抗否认性。同时，为了有效地应用密码技术，还必须了解密钥管理，因此本章还将重点介绍密钥管理技术，包括公钥基础设施PKI。

6.1　密 码 应 用

6.1.1　信息加密、认证和签名流程

加密、认证和签名是保护信息的机密性、完整性和抗否认性的主要技术措施。

加密是保障信息机密性的主要措施。加密机制的应用一般需要考虑如下几个因素：首先选择一种密码体制(对称密码体制或公钥密码体制)，然后从相应的密码体制中选择一个算法，再确定加密工作模式。有时，数据的大小不适合算法的分组需要，则需要填充附加数据。

完整性检测也称消息认证，主要通过信息摘要来实现。相应的机制主要有消息认证码、签名、加密、序列完整性、复制、完整性恢复等。序列完整性检测是为了检测数据项的重放、重排、丢失，一般可通过两种方法来实现：一种是在封装、签名或加密之前，给数据附加一个完整性序列号；另一种是在封装、签名或加密过程中利用数据项上的链产生一个加密链。

信息否认有两种情况：起源的否认和传递的否认。前者的目的是否认产生特定的数据和/或产生数据的时间；后者的目的是否认传递特定的数据和/或传递数据的时间。抗起源否认可以采用如下机制：

(1) 发送者签名：发送者本地签署数据项并将签名与相应的数据一起传递给接收者，接收者收到后验证数字签名并保存签名。若以后发生发送者否认数据项起源的事件，接收者可以出示保存的数字签名作为证据。

(2) 可信第三方签名：可以用可信第三方的数字签名代替发起者自己的签名。此时可信第三方被看成了发起者的担保。发起者传递数据项给可信第三方，可信第三方对数据项、

发起者身份以及其他必要的信息(如时间戳)产生一个数字签名，传送给接收者用于验证和保存证据。

抗传递否认可以采用如下机制：

(1) 接收者签名确认：接收者对接收到的信息或内容摘要及其他必要的信息进行签名，作为确认信息回复发送者。

(2) 可信传递代理确认：为了防止接收者收到信息之后否认，可以采用可信的第三方作为传递代理介入发送者与接收者的通信线路中。当发送者发送的数据项经过传递代理时，由它生成签名确认收到的消息。一般情况下传递代理只有在消息到达接收者才会生成签名确认。

信息在网络中传输时，发送方、接收方、可信第三方及敌方的关系如图 6.1 所示。

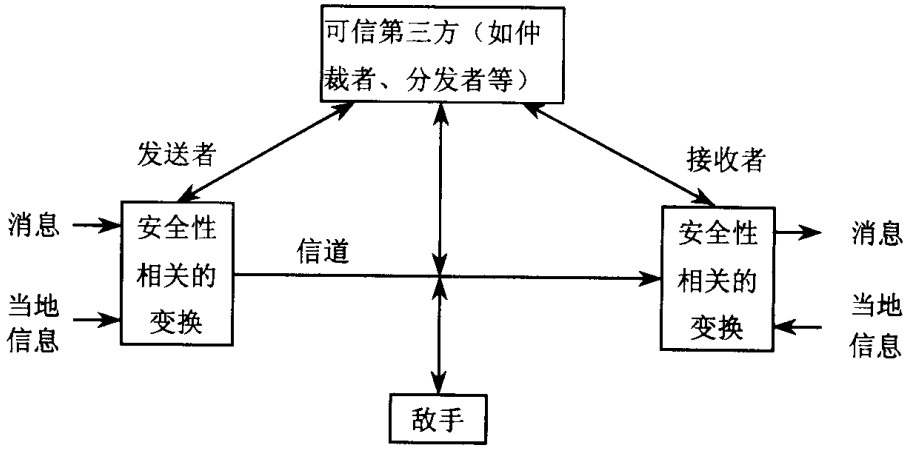


图 6.1 网络通信安全模型

其中，安全相关变换主要是加密、认证和签名，其流程如图 6.2 所示。发送者首先用自己的私钥对要发送的消息(附加时间戳)签名附在消息后，再用只与接收者共享的密钥加密，就得到要在信道上传输的数据。接收者收到这个数据后，用只与发送者共享的密钥解密，再用发送者的公钥验证签名，从而验证发送者和消息的完整性。如果验证成功，说明消息传输到接收者时保证了机密性、完整性和源非否认性。如果接收者发给发送者确认签名，则保证了传递的非否认性。

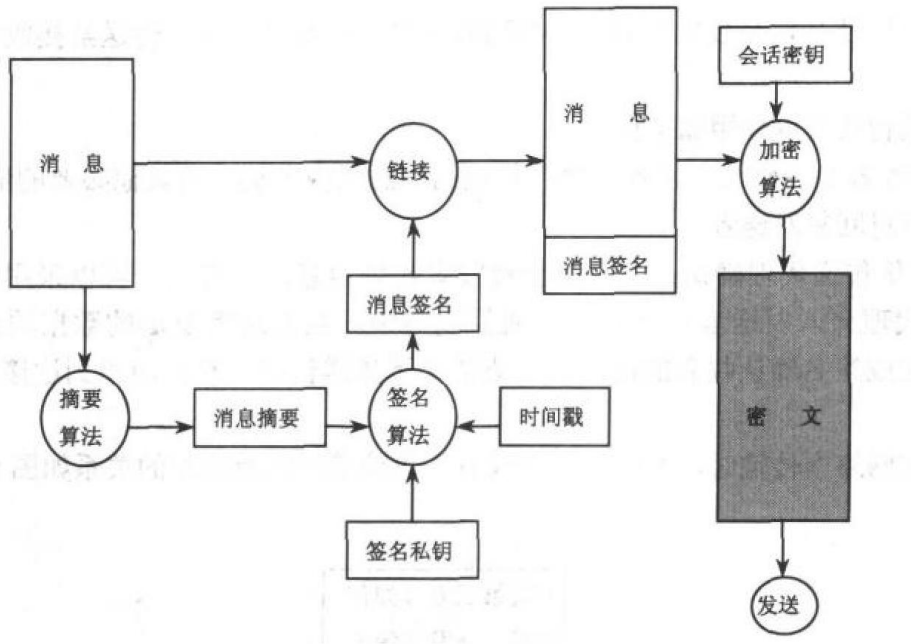


图 6.2　加密、认证和签名流程

6.1.2　加密位置

在通信网络中加密，首先要知道哪些数据需要加密以及在网络的哪些环节进行加密。根据加密物理位置的不同，可以分为端—端加密和链路加密。

链路加密是指每个易受攻击的链路两端都使用加密设备进行加密，图 6.3 中加密机 A 和加密机 B 之间的加密就是链路加密。链路加密中整条链路上的传输都是安全的。但链路加密也有缺点：数据报每进入一个分组交换机后都需要解密，因为交换机必须读取数据报报头中的地址以便为数据报选择路由，这样，在交换机中数据报易受攻击。链路加密时，每一链路两端的一对节点都应该共享一个密钥，不同结点对共享不同的密钥。因而需要提供很多密钥，每个密钥仅仅分配给一对节点。

端—端加密是指仅在一对用户的通信线路两端进行加密，图 6.3 中客户终端 A 和客户终端 B 或者客户终端 A 和服务器 A 间的加密都属于端—端加密。端—端加密数据是以加密的形式从源结点通过网络传送到目标结点，目标结点用与源结点共享的密钥对数据解密。所以，端—端加密可以防止对网络上链路和交换机的攻击。端-端加密还能提供一定程度的认证，因为源结点与目标结点共享同一密钥，因而目标结点相信收到的数据是源结点传送来的。

根据在网络中加密逻辑层次的不同，可分为链路层加密、网络层加密、会话层加密和应用层加密。链路层加密主要采用流密码技术，在链路层加密与通信链路关系密切，因此

通用性差；网络层加密和会话层加密类似，都以分组加密算法为主，其典型的协议分别是IPsec 和 SSL，本书将在第 13 章“网络安全协议”中专门介绍；应用层加密则与应用关系密切，如邮件加密协议 PGP、电子商务安全协议 SET 等，将在第 15 章“应用安全”中介绍。

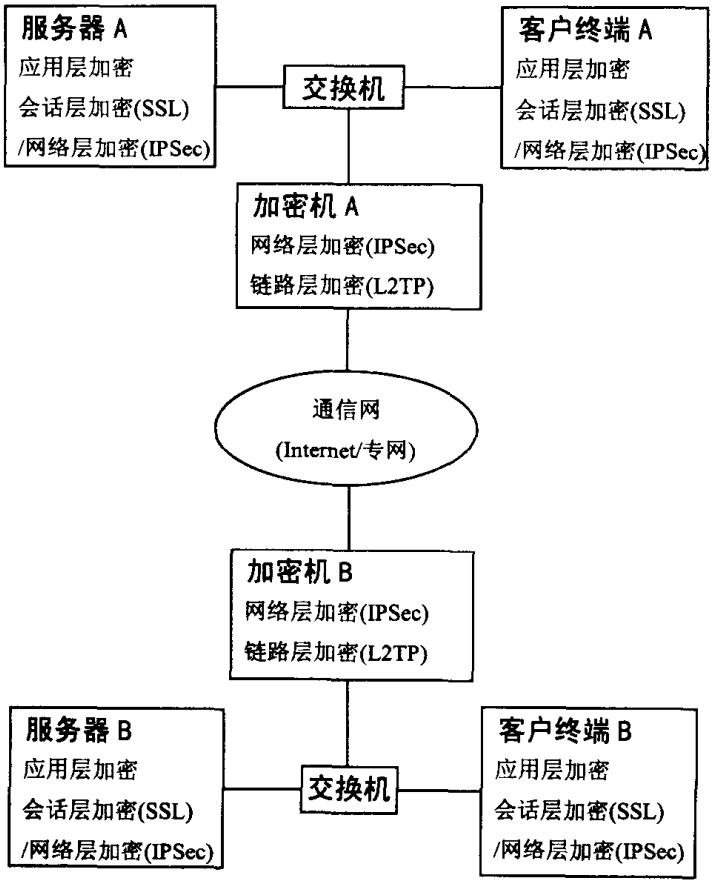


图 6.3 加密位置

6.2 密 钥 管 理

6.2.1 概述

在采用密码技术保护的现代通信系统中，密码算法通常是公开的，因此其安全性就取决于对密钥的保护。一旦密钥丢失或出错，要么合法用户不能提取保密的信息，要么非法

用户可能窃取信息。因此，密钥生成算法的强度、密钥的长度、密钥的保密和安全管理是保证系统安全的重要因素。

密钥管理的任务就是管理密钥的产生到销毁全过程，包括系统初始化、密钥的产生、存储、备份、恢复、装入、分配、保护、更新、控制、丢失、吊销和销毁等。

所有的密钥都有生命周期。这是因为拥有大量的密文有助于密码分析。一个密钥使用时间太长，为攻击者收集大量密文提供了机会。破译一个密钥需要时间，限制密钥的使用时间也就限制了密钥的破译时间，降低了密钥被破译的可能性。

6.2.2　密钥的分类

从网络应用来看，密钥一般分为以下几类：基本密钥、会话密钥、密钥加密密钥和主机密钥等。

(1) 基本密钥：基本密钥又称初始密钥，是由用户选定或由系统分配，可在较长时间内由一对用户专门使用的秘密密钥，也称用户密钥。基本密钥既要安全，又要便于更换。基本密钥与会话密钥一起用于启动和控制密钥生成器，从而生成用于加密数据的密钥流，如图 6.4 所示。

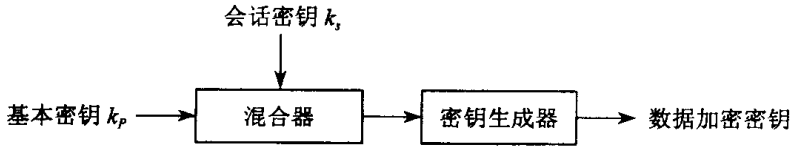


图 6.4　几类密钥之间的关系

(2) 会话密钥：会话密钥即两个通信终端用户在一次通话或交换数据时所用的密钥。当用于对传输的数据进行保护时称为数据加密密钥，而用于保护文件时称为文件密钥。会话密钥的作用是使人们不必太频繁地更换基本密钥，有利于密钥的安全和管理。这类密钥可由用户双方预先约定，也可由系统通过密钥建立协议动态地生成并赋予通信双方，它为通信双方专用，故又称专用密钥。

(3) 密钥加密密钥：用于对传送的会话或文件密钥进行加密时采用的密钥，也称为次主密钥、辅助密钥或密钥传送密钥。每个节点都分配有一个这类密钥。为了安全，各节点的密钥加密密钥应该互不相同。每个节点都须存储有关到其他各节点和本节点范围内各终端所用的密钥加密密钥，而各终端只需要一个与其节点交换会话密钥时所需要的密钥加密密钥，称为终端主密钥。

(4) 主机主密钥：是对密钥加密密钥进行加密的密钥，存于主机处理器中。

除了上述几种密钥外，还有一些密钥，如用户选择密钥、算法更换密钥等，这些密钥

的某些作用可以归入上述几类中。

6.2.3 密钥长度的选择原则

密钥长度的选择与加密数据的重要性和保密期限有关。当加密算法除穷举外无其他破译捷径时，密钥长度和每秒可实现的搜索密钥数决定了密码体制的安全性。

当密钥的长度为 n 比特时，有 2^n 个可能的穷举对象。密钥搜索速度由计算资源决定。1991 年 Cryptologia 杂志第 3 期报导，用当时的技术，5 年内可做出 400 万密钥/秒的 ASIC 芯片，10 年内可做出 2.56 亿密钥/秒的 ASIC 芯片。若用 4 000 个 2.56 亿密钥/秒的 ASIC 芯片，则 56 比特密钥的 DES 可在一天内破译。若用 80 万个这类芯片，则 64 比特密钥的算法可在一天内破译。1999 年 1 月 RSA 数据安全会议期间，电子前沿基金会用 22 小时 15 分钟就宣告破解了一个 DES 的密钥。

应当注意，密钥长度的选择与破译的代价有关。

总之，密钥长度的选择与具体的应用有关，如数据的重要性、保密期限的长短、可能破译者的计算能力大小等。目前，长度在 128 比特以上的密钥是安全的。

6.2.4 密钥的产生和装入

密钥的产生必须考虑具体密码体制的公认的限制。

网络系统中加密需要大量的密钥，以分配给各主机、节点和用户。产生好的密钥是非常重要的。可以用手工的方法，也可以用密钥产生器产生密钥。用密钥产生器产生密钥不仅可以减轻繁琐的劳动，而且可以消除人为的差错。

不同种类的密钥产生的方法不同。基本密钥是控制和产生其他加密密钥的密钥，而且长期使用，其安全性非常关键，需要保证其完全随机性、不可重复性和不可预测性。而任何密钥产生器产生的密钥都有周期性和被预测的危险，不适宜作主机密钥。基本密钥量小，可以用掷硬币等方法手工产生。密钥加密密钥可以用伪随机数产生器、安全算法等产生。如在主机主密钥控制下，由 X9.17 安全算法生成。会话密钥、数据加密密钥可在密钥加密密钥控制下通过安全算法产生。

密钥的装入有以下几种情形：

(1) 主机主密钥的装入：主密钥由可信赖的保密员在非常安全的条件下装入主机，一旦装入，就不能读取。输入环境要防电磁辐射、防窜扰、防人为出错。并且要用可靠的算法检验证实装入的密钥是否正确。

(2) 终端主密钥的装入：可由保密员在安全的条件下装入，也可以用专用的密钥注入设备装入，以后不能读取。同样要验证装入密钥的正确性。

(3) 会话密钥的获取：如主机与某终端通信，主机产生会话密钥，以相应的终端主密

钥对其进行加密，将加密结果送给相应的终端，终端收到后，解密得到会话密钥。

6.2.5　对称密码体制的密钥分配

任何密码系统的强度都依赖于密钥分配技术，密钥分配研究密码系统中密钥的分发和传送中的问题。

对称密码的密钥分配的方法归纳起来有两种：利用公钥密码体制实现和利用安全信道实现。利用公钥密码体制实现将下节介绍。利用安全信道实现直接面议或通过可靠的信使传递。传统的方法是通过邮递或信使传送密钥，密钥可用打印、穿孔纸带或电子形式记录。这种方案的安全性完全取决于信使的忠诚和素质。这种方式成本较高，有人估计分配密钥的费用占密码系统费用的三分之一。为了减少费用可采用分层的方法，信使只传送密钥加密密钥，这种方法只适宜于高安全级密钥，如主密钥的传递。也可采用某种隐蔽的方法，如将密钥分拆成几部分分别递送，除非敌手可以截获密钥的所有部分。这只适宜于少量密钥的情况，如主密钥、密钥加密密钥。会话密钥可以用主密钥加密后通过公用网络传送。

在局部网络中，每对用户可以共享一个密钥，如图 6.5 所示。两个用户 A 和 B 要建立会话密钥，须经过以下 3 步：

(1) A 向 B 发出建立会话密钥的请求和一个一次性随机数 N_1；

(2) B 用与 A 共享的主密钥对应答的消息加密，并发送给 A，应答的消息中包括 B 选取的会话密钥、B 的身份、$f(N_1)$和另一个一次性随机数 N_2；

(3) A 用新建立的会话密钥加密 $f(N_2)$并发送给 B。

在大型网络中，不可能每对用户共享一个密钥。因此采用中心化密钥分配方式，由一个可信赖的联机服务器作为密钥分配中心(KDC)来实现，图 6.6 所示的是中心化密钥管理方式的一个实例。用户 A 和 B 要建立共享密钥，可以采用如下 5 个步骤：

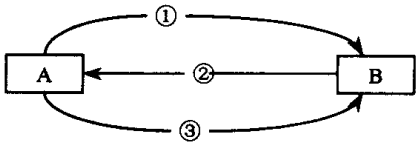


图 6.5　无中心密钥分配模式

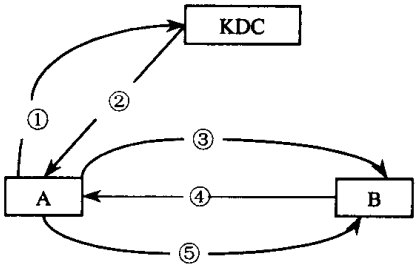


图 6.6　中心化密钥分配模式

(1) A 向 KDC 发出会话密钥请求。该请求由两个数据项组成，一是 A 与 B 的身份；二是一次性随机数 N_1。

(2) KDC 为 A 的请求发出应答。应答是用 A 与 KDC 的共享主密钥加密的，因而只有

A 能解密这一消息，并确信消息来自 KDC。消息中包含 A 希望得到的一次性会话密钥 K_s 以及 A 的请求，还包括一次性随机数 N_1。因此 A 能验证自己的请求没有被篡改，并通过一次性随机数 N_1 可知收到的应答不是过去的应答的重放。消息中还包含 A 要转发给 B 的部分，这部分包括一次性会话密钥 K_s 和 A 的身份。它们是用 B 与 KDC 的共享主密钥加密的。

(3) A 存储会话密钥，并向 B 转发从 KDC 的应答中得到的应该转发给 B 的部分。B 收到后，可得到会话密钥 K_s，从 A 的身份知会话的另一方为 A。

(4) B 用会话密钥 K_s 加密另一个一次性随机数 N_2，并将加密结果发送给 A。

(5) A 用会话密钥 K_s 加密 $f(N_2)$，并将加密结果发送给 B。

应当注意前 3 步已完成密钥的分配，后两步结合第(2)、(3)步完成认证功能。

6.2.6 公钥密码体制的密钥分配

公钥密码体制的一个重要用途就是分配对称密码体制使用的密钥，本节将介绍此内容和公钥的分配。

由于公钥加密速度太慢，常常只用于加密分配对称密码体制的密钥，而不用于保密通信。

当 A 要与 B 通信时，A 产生一对公钥、私钥对，并向 B 发送产生的公钥和 A 的身份。B 收到 A 的消息后，产生会话密钥 K_s，用产生的公钥加密后发送给 A。A 用私钥解密得到会话密钥 K_s。此时，A 和 B 可以用会话密钥 K_s 进行保密通信。A 销毁此次产生的公钥、私钥对，B 销毁从 A 得到的公钥。

这个方法虽然简单，但易受敌手的攻击。下面介绍一种具有保密性和认证性的分配方法。假定 A 和 B 已完成公钥交换，则

(1) A 用 B 的公钥加密 A 的身份和一个一次性随机数 N_1 后发给 B；

(2) B 解密得 N_1，并用 A 的公钥加密 N_1 和另一个一次性随机数 N_2 后发给 A；

(3) A 用 B 的公钥加密 N_2 后发给 B；

(4) A 选一个会话密钥 K_s，用 A 的私钥加密后再用 B 的公钥加密，发送给 B；

(5) B 用 A 的公钥和 B 的私钥解密得 K_s。

公钥的分配方法有以下几种：

(1) 公开发布：用户将自己的公钥发给所有其他用户或向某一团体广播。如将自己的公钥附加在消息上，发送到公开的区域，如因特网的邮件列表。

这种方法简单但有一个非常大的缺陷，别人能容易地伪造这种公开的发布。

(2) 公钥动态目录表：指建立一个公用的公钥动态目录表，表的建立和维护以及公钥的分布由某个公钥管理机构承担，每个用户都可靠地知道管理机构的公钥。公钥的分配步

骤如下：

a) 用户 A 向公钥管理机构发送一带时戳的请求，请求得到用户 B 当前的公钥；

b) 管理机构为 A 的请求发出应答，应答中包含 B 的公钥以及 A 向公钥管理机构发送的带时戳的请求；

c) A 用 B 的公钥加密一个消息并发送给 B，这个消息由 A 的身份和一个一次性随机数 N_1 组成；

d) B 用与 A 同样的方法从公钥管理机构得到 A 的公钥；

e) B 用 A 的公钥加密一个消息并发送给 A，这个消息由 N_1 和 N_2 组成，这里 N_2 是 B 产生的一个一次性随机数；

f) A 用 B 的公钥加密 N_2，并将加密结果发送给 B。

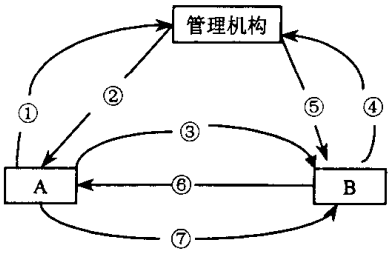


图 6.7　公钥动态目录表分配模式

用上述方案分配公钥也有缺点，由于每一用户要想与他人通信都要求助于公钥管理机构，因而公钥管理机构有可能成为系统的瓶颈，而且公钥目录表也容易被窜扰。

分配公钥的一种安全有效的方法是采用公钥证书，用户通过公钥证书相互之间交换自己的公钥而无需与公钥管理机构联系。

(3) 公钥证书：公钥证书由证书管理机构 CA 为用户建立，其中的数据项有该用户的公钥、用户的身份和时戳等。所有的数据经 CA 用自己的私钥签字后就形成证书。证书中可能还包括一些辅助信息，如公钥使用期限、公钥序列号或识别号、采用的公钥算法、使用者的住址或网址等，后面将详细介绍。

6.2.7　密钥托管

密钥托管密码系统是具有备份解密能力的密码系统，它允许授权者在特定的条件下，借助于一个以上持有专用数据恢复密钥的、可信赖的委托方所提供的信息来解密密文。数据恢复密钥不同于通常的加解密密钥，但由它们可以恢复加解密密钥。

密钥托管技术提供了一个备用的解密途径，政府机构在需要时，可通过密钥托管技术解密用户的信息，而用户的密钥若丢失或损坏，也可通过密钥托管技术恢复出自己的密钥。

一个密钥托管系统由用户安全组件 USC、密钥托管组件 KEC 和数据恢复组件 DRC 三个部分组成，如图 6.8 所示。USC 使用密钥 K 加密明文数据，并把数据恢复域 DRF 联接到密文上；DRC 则利用 KEC 提供的信息和包含于数据恢复域 DRF 中的信息恢复出明文。

1. 用户安全组件 USC

USC 是硬件设备或软件程序，用于加密和解密，同时也支持密钥托管功能。USC 可以用于通信，法律执行机构在获得法庭的授权后采用应急解密介入通信，即所谓的搭线窃

听。数据的拥有者可使用应急解密恢复丢失或损坏的密钥，法律执行机构经法庭许可后，可以使用应急解密，解密法庭所允许的数据。USC 也可用于数据存储。

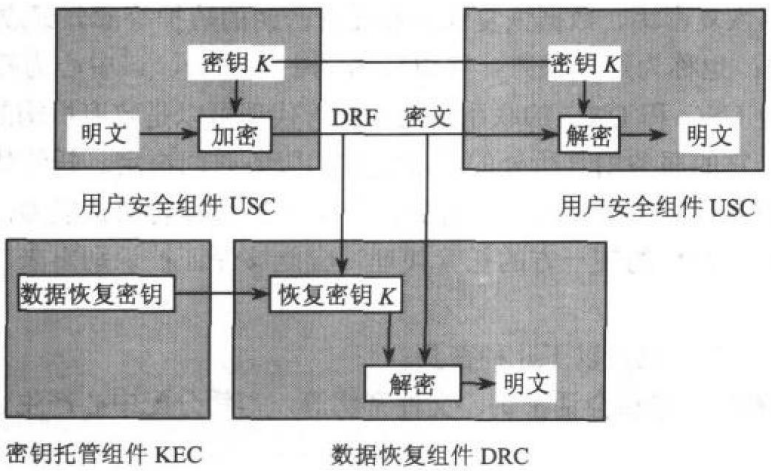


图 6.8 密钥托管系统

USC 标有数据加密算法的名称、运行模式、密钥长度等。存储应急解密所需要的识别符，如用户、USC、密钥、KEC、托管代理等的识别符。还存储各种密钥，如用户的密钥、USC 的密钥等。

当用密钥 K 对数据加密时，USC 必须把密文和 K 与一个或多个数据恢复密钥链接在一起，通常把 DRF 连接到加密数据上。这种连接包括如下几个方面：

(1) 密钥 K 与用户(发方、收方或者双方)的密钥托管代理所保存的数据恢复密钥的联系。

(2) DRF 和链接机制可以集入将密钥 K 传送给预定接收者的协议中，这时发送者必须传送一个有效的 DRF，以使预定的接收者能获得密钥。

(3) DRF 一般包括一个以上用数据恢复密钥(如产品的密钥、发送者或接收者的公钥、KEC 的主公钥等)加密的 K。在某些情况下，通过 DRF 只能恢复 K 的一部分，其余部分必须用穷举搜索来确定。DRF 还包括一些标识信息，可以识别数据恢复密钥、KEC 或密钥托管代理、加密算法和模式以及 DRF 的生成方法。整个 DRF 可由与 DRC 相关的族密钥加密，以保护 DRF 中的标识信息，加密算法可以用公钥算法也可以用单钥算法。DRF 的有效性通过接收者验证 DRF 中所包含的托管认证符(EA)，以确定 DRF 的完整性。

USC 可以用硬件、软件、固件或其他组合方式实现。一般硬件比较安全，硬件实现包括专用密码处理器、随机数产生器、高精度时钟等。USC 设备有时也称为托管加密设备。

2. 密钥托管组件 KEC

KEC 用于存储所有的数据恢复密钥，可以看成是公钥证书管理系统的组成部分，也可

以看成是一般的密钥管理基础设施的组成部分，向 DRC 提供所需的数据和服务。KEC 由密钥托管代理进行操作，在公钥结构下，托管代理可以作为一个公钥证书签发机构。KEC 由托管代理、数据恢复密钥、数据恢复业务和托管密钥的防护等部分组成。

(1) 托管代理：也称为可信赖方，可以在密钥中心注册，该中心为托管代理制定操作规则，也可以作为 USC 和 DRC 的联系机构。托管代理可以是政府机构的一个部门或者是一个独立的实体，它的服务由其所处的地理位置和服务时间决定。托管代理的安全性是指防止被托管密钥的泄露、丢失和滥用的能力，包括可靠性和防窜扰能力；对于破坏数据恢复或将密钥泄露给非授权的某一方的托管代理，应能够保证被识别出来。托管代理应该有政府的许可证。

(2) 数据恢复密钥：包括以下几种密要：

1) 数据加密密钥：包括会话密钥、文件密钥等，密钥分配中心产生、托管并分配这些密钥；

2) 产品密钥：对 USC 是惟一的；

3) 用户密钥：通常是公钥/私钥对，用于建立数据加密密钥。KEC 可以作为用户的公钥证书签发机构，负责签发用户公钥证书；

4) 主密钥：与 KEC 相关，可由多个 USC 共享。

数据恢复密钥可采用密钥分拆(如秘密共享或门限方案)进行托管，把数据恢复密钥分拆为多个密钥分量，每个分量都分别由不同的托管代理保存。若要恢复被分拆的密钥，需要所有密钥托管代理的参与，或在采用(n，k)门限方案时，至少需要有 k 个密钥托管代理的参与。

数据恢复密钥可由 KEC、USC 或者两者的结合来产生和分配。如果由 USC 产生密钥，可采用可验证秘密共享方案进行分拆和托管，这样托管代理在不知数据恢复密钥的情况下，能够验证自己托管的密钥分量的有效性。

数据恢复密钥的托管可以在产品的制作阶段、系统和产品的初始化阶段或用户注册期间进行。如果托管的是用户的公钥/私钥对中的私钥，则应该在相应的公钥进入公钥结构和证书发放时进行托管。USC 只能把加密的数据送给有公钥证书的用户，该证书上有获得许可的托管代理的签字。数据恢复密钥可以采用部分或全部托管，在部分托管的情况下进行数据恢复时，通过穷举搜索可以确定其余未托管部分的密钥。

某些体制允许数据恢复密钥的更新，但只能根据要求或定期进行。

(3) 数据恢复业务：KEC 所提供的服务还包括向 DRC 披露信息。首先要进行授权，使操作和使用 DRC 的人员可以利用 KEC 的业务，包括建立身份证明和获得接入已加密数据的合法授权证明。所提供的业务有以下几种：

1) 披露数据恢复密钥：一般在数据恢复密钥是会话密钥或用户产品密钥的情况下采用此方法(不暴露主密钥)，密钥可以与有效期一起披露，然后自动销毁；

2) 披露派生密钥：KEC 披露数据恢复密钥的派生密钥，如时限密钥，该密钥只允许加密在规定时间段内加密的数据；

3) 解密密钥：在主数据恢复密钥用于加密 DRF 中的数据加密密钥时，一般采用这种方法，这样 KEC 不必把主密钥披露给 DRC；

4) 实现门限解密：每个托管代理分别将其解密的“片段”送给 DRC，由 DRC 将这些碎片合成。数据恢复业务中可以人工或自动地将数据传入或传出 DRC。

(4) 托管密钥的防护：KEC 采取措施防止密钥的泄漏或丢失，这些措施包括技术保护、操作保护和法律保护等。如审计、责任分离、知识划分、秘密分拆、多人控制、物理安全、密码技术、信赖体制、独立验证、证书、鉴定、结构管理以及对滥用的惩罚等。

3. 数据恢复组件 DRC

DRC 由算法、协议和必要的设备组成，用来从密文和 KEC 所提供的包含于 DRF 中的信息中恢复出明文。它只有在执行规定的合法数据恢复时才能使用。

为了解密数据，DRC 必须获得数据加密密钥 K，可以采用如下方法：

(1) 从发送者或接收者接入：当只能利用发送者的托管代理持有的密钥才能获取 K 时，DRC 就必须得到对某一特定用户传送消息的所有各方的密钥托管数据，这可能会阻碍实时解密，尤其是在各个用户使用不同的托管代理时。同样地，当只利用接收者的托管代理持有的密钥才能获取 K 时，就不可能实时解密特定用户传送的所有消息。如果利用托管代理的子集所持有的密钥可以进行数据恢复，则一旦获得 USC 的加密密钥 K，DRC 就可以实时解密从 USC 进出的消息。对两方实时通信(如电话)的情况，要实时恢复加密数据，要求通信双方使用同样的密钥。

(2) 与 KEC 的交换频度：可以规定 DRC 针对每一个数据加密密钥或 USC 用户都和 KEC 进行一次交互，其中，对前者要求交互是联机的。这样，在每次会话密钥改变时，仍然可以做到实时解密。

(3) 穷举搜索：当 DRC 从托管代理只能得到部分密钥时，密钥的其余部分只有通过穷举搜索来确定。

DRC 可以使用保护手段以便控制哪些内容可以被解密。这些手段有技术保护、操作保护和法律保护等。如对数据恢复进行时间限制等。也可以使用认证机制来防止 DRC 使用自己所需要的密钥产生伪造的消息。

4. 美国托管加密标准简介

1993 年 4 月，美国政府提出了一项新的建议，倡导使用新的具有密钥托管功能的联邦加密标准，称为托管加密标准 EES(Escrowed Encryption Standard)。该标准使用的托管加密

技术不仅提供强加密功能，而且提供实施法律授权下的监听。该托管加密技术通过一个防窜扰的芯片(Clipper)实现，分为两部分：

(1) Skipjack 加密算法：该算法由 NSA 设计，用于加解密用户间的通信消息。

(2) 法律强制访问域 LEAF(Law Enforcement Access Field)：通过这部分，法律实施部门可以在法律授权下，实施对用户通信的解密。

Skipjack 算法是对称分组密码算法，密钥长度为 80 比特，输入和输出的分组长度为 64 比特。该算法于 1985 年由 NSA 开始设计，于 1990 年完成评价，结果认为算法的强度高于 DES，并且没有陷门。

Clipper 芯片装有 Skipjack 算法、80 比特长的族密钥 FK(Family Key)(同一批芯片的族密钥相同)、芯片单元识别符 UID(Unique Identifier)、80 比特的芯片单元密钥 UK(Unique Key)，它是两个 80 比特的芯片单元密钥分量的异或及控制软件。这些成分固化在芯片上，编程过程在安全密封信息设备 SCIF(Secure Compartmented Information Facility)中进行，有两个托管机构监控，一段时间一批；如图 6.9 所示。

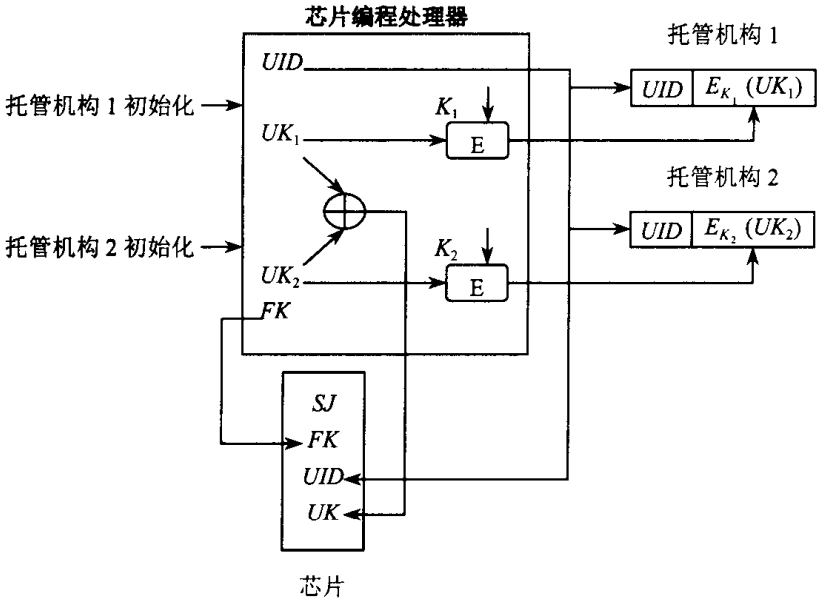


图 6.9　托管加密芯片的编程过程

两个托管机构的代表通过向芯片编程处理器输入两个参数(随机数)进行初始化。芯片编程处理器对每个芯片，分别计算以上两个初始参数和 *UID* 的函数，作为单元密钥的两个分量 UK_1 和 UK_2，以二者的异或作为单元密钥 *UK*。用分配给托管机构的密钥 K_1 和 K_2 分别加密 UK_1 和 UK_2，并将(UID，$E_{K_1}(UK_1)$)和(UID，$E_{K_2}(UK_2)$)分别给托管机构 1 和托管机构 2 以托管形式保存。

编程过程结束后，处理器被清除，只能从托管机构获得加密了的单元密钥分量，并且只能用政府解密设备解密。

5. 托管加解密和授权监听

法律强制访问域 LEAF 是 EES 的一个主要特征。为了监视一个 Clipper 芯片，必须产生一个 LEAF。如用户 A 使用 Clipper 芯片加密发送消息 M 给另一个用户 B，A 与 B 用任一个密钥协商的方法(如 Diffie-Hellman 密钥协商协议)商定一个 80 比特的会话密钥 K，然后 A 将 M 和 K 输入芯片，与芯片产生的初始化向量 V_I 一起产生 LEAF 并计算出密文 $C = E_K(M)$。LEAF 结构中包含 32 比特长的芯片识别符、对 80 比特长的会话密钥的加密结果和 16 比特长的认证码 MAC，因此 $LEAF = E_{FK}(E_{UK}(K)\|UID\|MAC)$，这里 $MAC = E_{FK}(A,K,V_I)$。最后 A 发送密文 C 和法律强制访问域 LEAF 给 B。

B 收到密文 C 和法律强制访问域 LEAF 后，将它们与会话密钥 K 一起输入芯片。芯片用族密钥 FK 解密 LEAF 得到 $E_{UK}(K)$、UID 和 MAC，并验证 MAC。如果验证不能通过，则停止；否则，用会话密钥解密密文得消息 M。

在上述解密过程中，认证码的验证对 EES 的应用是关键。因为 EES 被封装在一个防窜扰的芯片中，没有人能够绕过认证码的验证。只有在这个验证通过之后，解密才能进行。因此，A 要使得他发送的密文被合法接收者解密，就必须产生一个 LEAF。

如果一个政府机构要监听用户 A 给用户 B 发送的消息，首先要获得法庭的批准。电信部门验证法庭批准文件后，出租线路给该政府部门用于监听。被截获的通信数据首先被输入一个政府机构控制的解密设备，解密设备可识别由托管芯片加密的通信，取出密文 $E_K(M)$、法律强制访问域 LEAF 和初始化向量 V_I，并用族密钥 FK 解密得到的 LEAF 获取芯片的识别符 UID 和加了密的会话密钥 $E_{UK}(K)$。然后向 A 的两个密钥托管机构出示法庭批准监听用户 A 的文件和芯片识别符 UID。在验证了法庭的批准文件之后，密钥托管机构向政府专门解密设备提供加了密的单元密钥分量 $E_{K_1}(UK_1)$ 和 $E_{K_2}(UK_2)$，政府控制的解密设备解密得到单元密钥分量 UK_1 和 UK_2，并计算它们的异或得单元密钥 UK。用 UK 解密 $E_{UK}(K)$得到会话密钥 K。最后解密设备用会话密钥 K 解密密文 C 得到消息 M。

6.3 公钥基础设施 PKI

6.3.1 PKI 概述

公钥基础设施 PKI(Public Key Infrastructure)是一种标准的密钥管理平台，它为网络应

用透明地提供加密和数字签名等密码服务所必需的密钥和证书管理。PKI 由证书颁发机构 CA、注册认证机构 RA、证书库、密钥备份及恢复系统、证书作废处理系统、PKI 应用接口系统等部分组成，如图 6.10 所示。

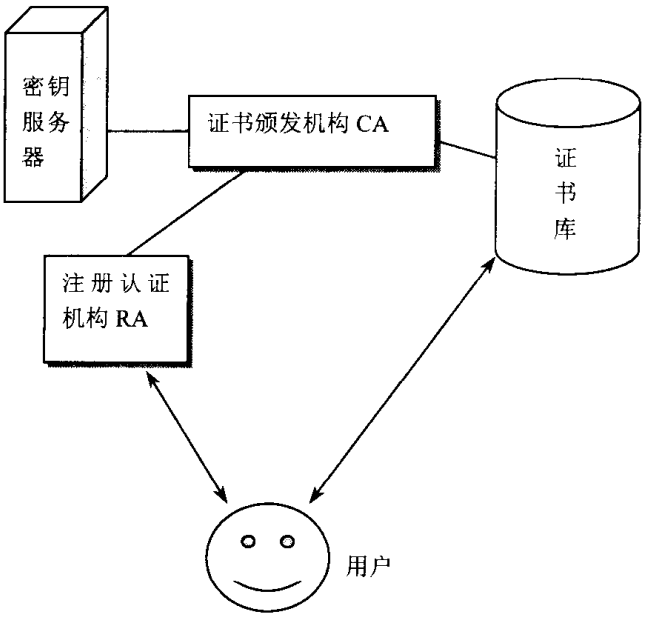


图 6.10　PKI 的组成

1. 证书颁发机构 CA

CA 是提供身份验证的第三方机构，也是公钥证书的颁发机构，由一个或多个用户信任的组织或实体组成。在 PKI 中，CA 负责颁发、管理和吊销最终用户的证书，认证用户并在分发证书之前对证书信息签名。公钥证书是公开密钥体制的一种密钥管理媒介，是一种权威性的电子文档，用于证明某一主体的身份以及公开密钥的合法性，证书结构如图 6.11 所示。

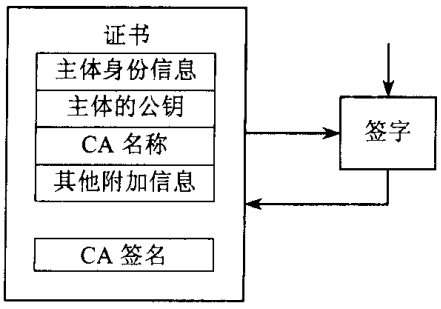


图 6.11　公钥证书结构

在使用公钥体制的网络中，必须向公钥的使用者证明公钥的真实合法性。因此，必须有一个可信的机构对密钥进行公证，证明密钥主人的身份及与公钥的关系。CA 正是这样的可信机构，它的职责有：

(1) 验证并标识证书申请者的身份；

(2) 确保 CA 用于签名证书的非对称密钥的质量；

(3) 确保整个签证过程和签名私钥的安全性；

(4) 证书材料信息(如公钥证书序列号、CA 等)的管理；

(5) 确定并检查证书的有效期限；

(6) 确保证书主人的标识的惟一性，防止重名；

(7) 发布并维护作废的证书表；

(8) 对整个证书签发过程做日志记录；

(9) 向申请人发通知。

其中 CA 自己的一对密钥的管理尤为重要。必须确保 CA 的私钥的机密性，防止它方伪造证书。CA 在提供服务的过程中，必须向所有由它认证的最终用户和可能使用这些认证信息的可信主体提供自己的公钥。但在 CA 自己的公钥证书中，主体和 CA 的名称是一样的。因此，CA 自己的证书是自签名的。

一个主体可以通过从证书库中获取或由另一主体传送的方式来得到另一主体的证书。CA 的数字签名保证了证书的合法性和权威性。证书中主体(用户)的公钥的产生方式有两种：

(1) 主体自己产生密钥对，并将公钥传送给 CA，该过程必须保证主体公钥的可验证性和完整性。

(2) CA 替主体生成密钥对，并将其安全地传送给主体，该过程必须保证主体密钥的机密性、可验证性和完整性。这种方式对 CA 的信任要求更高。

公钥一般有两类用途：

(1) 验证数字签名：消息接收者用发送者的公钥对消息的数字签名进行验证。

(2) 加密信息：消息发送者用接收者的公钥加密用于加密数据的密钥，进行数据加密密钥的传送。

相应于公钥的用途，系统中应配置用于数字签名/验证的密钥对和用于数据加密/脱密的密钥对，分别称为签名密钥对和加密密钥对。对这两种密钥的管理有所不同：

(1) 签名密钥对由签名私钥和验证公钥组成。为了保证签名私钥的惟一性，签名私钥不能备份和存档，遗失后只需要重新生成新的签名密钥对，旧的签名可由旧验证公钥的备份来验证。验证公钥需要存档，以便验证旧的数字签名。因此，签名密钥对可以有较长的生命周期。

(2) 加密密钥对由加密公钥和脱密私钥组成。为了防止密钥丢失时丢失数据以及在任何时候都能脱密历史密文数据，脱密私钥应该进行备份和存档。加密公钥不用备份和存档，当加密公钥丢失时，只需重新产生加密密钥对。

2. 注册机构 RA

可以将 RA 看成是 PKI 的一个扩展部分，但它已逐渐成为 PKI 的一个必不可少的组成

部分。RA 充当了 CA 和它的最终用户之间的桥梁，分担了 CA 的部分任务，协助 CA 完成证书处理服务。RA 一般有如下功能：

(1) 接收和验证新注册人的注册信息；

(2) 代表最终用户生成密钥对；

(3) 接收和授权密钥备份和恢复请求；

(4) 接收和授权证书吊销请求；

(5) 按需分发或恢复硬件设备，如令牌。

当终端用户数量增加或分散时， CA 负荷将随之增加。RA 可以减轻 CA 负担，方便用户。

3. 证书库

证书库是公开的信息库，用于证书的集中存放，供用户查询其他用户的证书和公钥。

4. 密钥备份及恢复系统

为了防止用户丢失密钥后，密文数据无法脱密，从而造成数据丢失，PKI 应该提供脱密密钥的备份和恢复的机制。脱密密钥的备份和恢复应该由可信机构来完成，如 CA。

5. 证书作废处理系统

证书在 CA 为其签署的有效期内也可能需要作废，为实现这一点，PKI 必须提供证书作废机制。作废证书一般通过将证书列入作废证书表(CRL)来完成。CRL 存放在目录系统中，由 CA 创建、更新和维护。当用户验证证书时负责检查该证书是否在 CRL 之列。

6. PKI 应用接口系统

也可称为客户端证书处理系统。为了使用户能够方便地使用加密、数字签名等安全服务，PKI 必须提供良好的应用接口系统，使各种应用能够以安全、一致、可信的方式与 PKI 交互，保证所建立起来的网络环境的可信性，降低维护和管理成本。

为此，PKI 需要满足如下一些要求：

(1) 透明性：PKI 必须尽可能地向应用屏蔽密码实现服务的实现细节，向用户屏蔽复杂的安全解决方案，使密码服务对用户而言简单易用，并且便于用户完全控制其信息资源；

(2) 可扩展性：为满足系统不断发展的需要，证书库和 CRL 应有良好的可扩展性；

(3) 支持多种用户：提供文件传送、文件存储、电子邮件、电子表单、WEB 应用等的安全服务；

(4) 互操作性：不同用户的 PKI 的实现可能是不同的，因此必须支持异构平台 PKI 的

互操作性；

6.3.2 公钥证书

公钥证书按包含的信息分为两种：一种是身份证书，能够鉴别一个主体与它的公钥关系，证书中列出了主体的公钥；另一类是属性证书，是包含了实体属性的证书，属性可以是成员关系、角色、许可证或其他访问权限。使用属性证书可以鉴别许可证、凭据或其他属性。现在讨论最多的是身份证书，这类证书提供了认证、数据完整性和机密性。但是它虽然解决了安全传输问题，但还不足以提供授权。因为在授权中，更多考虑的是主体的属性而不是身份，根据这些属性决定授权。为了管理方便、安全，可互操作，通常将这些授权信息从身份中提取出来，与公钥证书同样的方式加以保护，这就是属性证书。

现实中有各种各样的证书，如 PGP 证书、SET 和 IPSec 证书。最广泛采用的证书格式是国际电信联盟(ITU)提出的 X.509 版本 3 的格式。有时，一种证书可以被定义为几种不同的版本，如 X.509 公钥证书就有三种版本。版本 2 是版本 1 的扩张，版本 3 又是版本 2 的扩张，而版本 3 又有多种可选的不同的扩展。图 6.12 所示的是 X.509 版本 3 的证书结构。

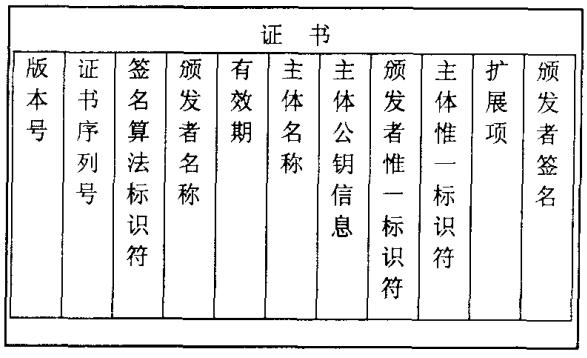


图 6.12 X.509 版本 3 的证书结构

证书中签名算法标识符是用来标识签署证书所用的数字签名算法和相关参数。如 SHA-1 和 RSA 的对象标识符就用来说明该数字签名是利用 RSA 对 SHA-1 杂凑加密。主体公钥信息包括主体的公钥及所用的加密算法。可选的扩展项包括：机构密钥标识符(用来区分同一个颁发者的多对证书签名密钥)、主体密钥标识符(用来区分同一个证书拥有者的多对密钥)、密钥用途(指明运用证书中的公钥可完成的各项功能和服务)、扩展密钥用途(说明证书中的公钥的特别用途)、CRL 分布点、私钥使用期、证书策略(一系列的与证书颁发和使用有关的策略对象标识符和可选的限定符)、主体别名(如主体的邮件地址、IP 地址等)、CA 别名(如 CA 的邮件地址、IP 地址等)、主体目录属性(证书拥有者的一系列属性)等。

有必要对 CRL 分布点进一步解释。当一个给定的 PKI 系统的 CRL 变得非常大时，就要创建许多小的 CRL 用于分发，而不是使用单一的大 CRL。这些较小的 CRL 可以更容易检索和处理。为了使用 CRL 分布点，CA 需要提供一个指向位于颁发证书中 CRL 分布点扩展项中的位置的指针。该指针包括一个 DSN 名字、一个 IP 地址或者一个 Web 服务器上的特定文件名，可以使依托主体定位 CLR 分布点。

6.3.3　公钥证书的发放和管理

公钥证书的发放一般经历两个过程：终端用户注册、证书的产生和发放。

CA 可以有多种方式让终端用户注册，用户的选择在很大程度上取决于应用环境。许多终端用户使用浏览器通过互联网向 CA 或 RA 注册。注册是 PKI 中最重要的过程之一。此时，终端用户与 CA 之间建立起信任。终端用户可以查看 CA 公布的证书策略和认证实施说明。而 CA 可以通过面对面的交流方式要求终端用户提供各种身份信息。

注册完成后，信任关系就在 CA 与终端用户之间建立了。此后，终端用户可以发起证书请求。发起证书请求有两种方式：终端用户自己生成密钥对，并向 CA 提供其公钥；或者由 RA 或 CA 代表终端用户生成一个密钥对。无论密钥对由谁产生，证书的建立是由 CA 完成的。

请求并取回证书需要有一个安全的协议机制。如 X.509 公开密钥基础设施证书管理协议(RFC2510)和 X.509 证书请求报文格式(RFC2511)等。

公钥证书发放以后，需要进行有效的管理。公钥证书的管理一般包括三个方面：

(1) 证书的检索：有两种不同的需求可能要求检索一个最终用户的公钥证书，加密发送数据或验证收到的数字签名。

(2) 证书的验证：验证一个证书的有效性。

(3) 证书取消：有两种方式导致证书的取消，即证书的自然过期，或证书在有效期内被作废。

6.3.4　PKI 的信任模型

PKI 的信任模型用来描述终端用户、依托主体和 CA 之间的关系。这里介绍最基本的 PKI 的信任模型：CA 的严格层次结构、分布式信任结构、Web 模型、以用户为中心的信任模型以及交叉认证模型。

1. CA 的严格层次结构

随着 PKI 的规模的扩大，证书的数量不断增多，一个单一的 CA 可能会成为认证过程的瓶颈。采用认证层次结构是解决这个问题的办法之一。在这个结构中，根 CA 将它的权利授给多个子 CA，这些子 CA 再将它们的权利授给它们的子 CA，这个过程直至某个 CA 实际颁发了某一证书。

图 6.13 所示的是 CA 的严格层次结构，可以看出它是一个树形的结构。每个实体(包括 CA 和终端用户)都信任根 CA，因而都必须拥有根 CA 的公钥。

一个终端实体 A 可以如下检验另一个终端实体 B 的证书。假设 B 的证书由子 CA_3(公钥为 k_3)签发，子 CA_3 的证书由子 CA_2(公钥为 k_2)签发，子 CA_2 的证书由子 CA_1(公钥为 k_1)

签发，子 CA_1 的证书由根 CA(公钥为 k)签发。拥有 k 的终端实体 A 可以利用 k 来验证子 CA_1 的公钥 k_1，然后利用 k_1 来验证子 CA_2 的公钥 k_2，再利用 k_2 来验证子 CA_3 的公钥 k_3，最终利用 k_3 来验证 B 的证书。

2. CA 的分布式结构

分布式信任结构把信任分散到两个或更多个 CA 上。采用严格层次结构的 PKI 系统往往在一个企业或部门实施，为了将这些 PKI 系统互联起来，可以采用下列两种方式建立分布式信任结构：

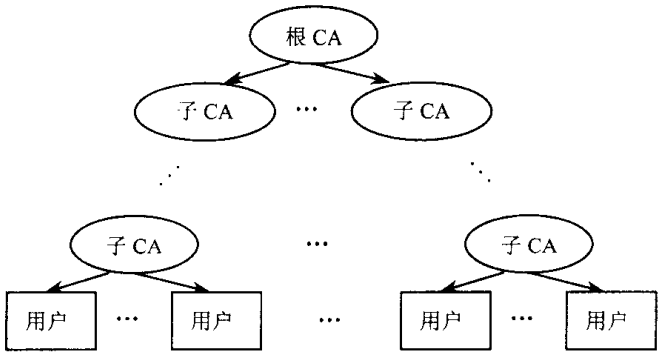


图 6.13 CA 的严格层次结构

(1) 中心辐射配置：在这种配置中，有一个中心地位的 CA，每个根 CA 都与这个中心 CA 进行交叉认证；

(2) 网状配置：在所有根 CA 之间进行交叉认证。

在分布式信任结构的中心辐射配置中，中心 CA 并不能被看成根 CA，如图 6.1.4 所示。在这个结构中，可能有许多个根 CA，每个实体都信任自己的根 CA，它们只拥有自己的根 CA 的公钥。一个终端实体 A 可以如下检验另一个终端实体 B 的证书。如果它们拥有同一个根 CA 的公钥，认证过程层次结构一样。否则，A 可以利用自己根 CA 的公钥来验证中心 CA 的公钥，然后利用中心 CA 的公钥来验证 B 的根 CA 的公钥，再利用 B 的根 CA 的公钥向下验证，直至验证终端实体 B 的证书。

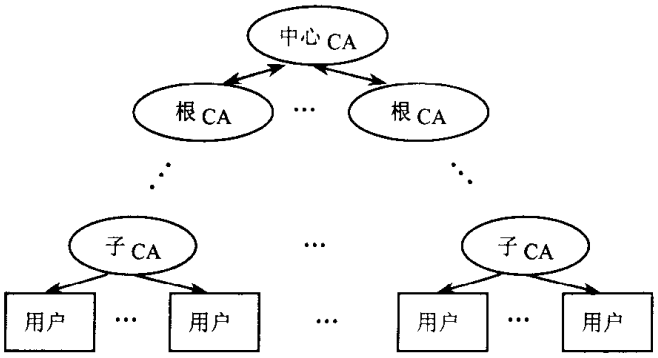


图 6.14 CA 的分布式信任结构之中心辐射配置

3. Web 模型

Web 模型依赖于浏览器，如 Navigator 和 Internet Explorer。这种模型将一些 CA 的公

钥预装在使用的浏览器上，这些公钥确定了一组 CA，浏览器的用户最初信任这些 CA 并把它们作为根 CA。这些根 CA 是通过物理地嵌入软件来发布的，这样就将 CA 的名字和它的公钥安全地绑定，如图 6.15 所示。

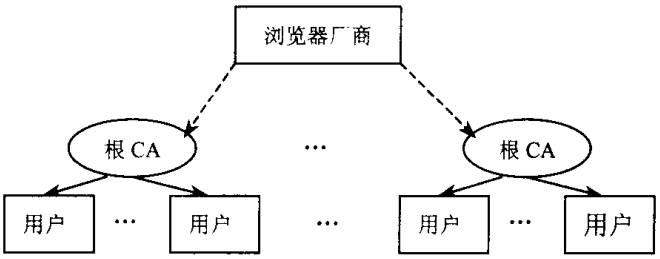


图 6.15　Web 模型

4．以用户为中心的信任模型

在以用户为中心的信任模型中，每个用户都对决定信赖或拒绝哪个证书负责。最初可信的密钥集可能只有朋友、家人同事等，如图 6.16 所示。

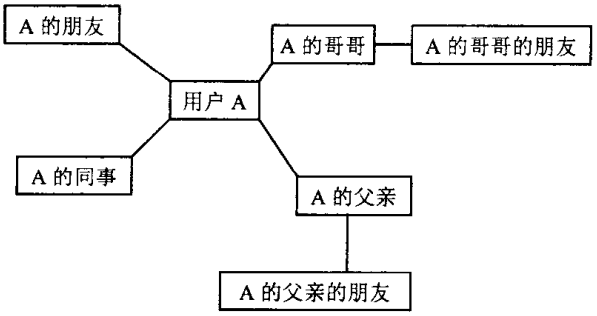


图 6.16　以用户为中心的信任模型

5．交叉认证模型

交叉认证是一种把各个 CA 连接在一起的有用的机制，它可以是单向的，如在 CA 的严格层次结构中，上层 CA 对下层 CA 的认证；也可以是双向的，如在分布式信任结构之中心辐射配置中，根 CA 与中心 CA 之间的互相认证。

在两个 CA 之间的交叉认证是：一个 CA 承认另一个 CA 在一个名字空间中被授权颁发的证书。例如，假设实体 A 已被 CA_1 认证并且拥有 CA_1 的公钥 k_1，而实体 B 已被 CA_2 认证并且拥有 CA_2 的公钥 k_2。在交叉认证之前，A 只能验证 CA_1 颁发的证书，不能验证 CA_2 颁发的证书；而 B 只能验证 CA_2 颁发的证书，不能验证 CA_1 颁发的证书。在 CA_1 与 CA_2 互相交叉认证之后，A 就能验证 CA_2 的公钥，从而验证 CA_2 颁发的证书；而 B 也能验证 CA_1

的公钥，从而验证 CA_1 颁发的证书。

习 题 六

1. 试述并图示采用 RSA、DES 及 SHA-1 算法保护信息的机密性、完整性和抗否认性的原理。

2. 说明密钥的分类和作用。

3. 设进行一次解密的时间是 1 微秒，计算用穷举法破译 64 比特、128 比特和 256 比特密钥长度的密码分别需要多少年。

4. 为什么常用对称算法加密数据，而用非对称算法分配密钥？

5. 授权监听是如何实现的？

6. 说明 CA 层次模型中的信任建立过程。

7. 试述一次完整的数字证书的颁发和使用过程。

第七章 身份认证

身份认证是信息安全理论的重要组成部分。以密码理论为基础的身份认证是访问控制和审计的前提，因此对网络环境下的信息安全尤其重要。本章重点讲述身份认证的作用、原理、实现和发展。

7.1 身份认证基础

正确识别主体(用户、节点或终端等)的身份十分重要。如，银行系统的自动取款机(ATM)，用户可以直接从取款机中提取现款，但 ATM 首先要确认用户的身份，否则恶意欺诈会给银行和用户造成损失。身份认证是指证实主体的真实身份与其所声称的身份是否相符的过程。这一过程是通过特定的协议和算法来实现的。

7.1.1 物理基础

身份认证的物理基础可以分为以下三种：

(1) 用户所知道的(something the user knows)；

(2) 用户所拥有的(something the user possesses)；

(3) 用户的特征(something the user is or how he/she behaves)。

第一类的例子是最常用的密码和口令；第二类的例子有身份证、护照、密钥盘等；第三类包括用户的生物特征，如指纹、红膜、DNA、声纹，还包括用户的下意识行为，比如签名。

这三类物理基础各有利弊。第一类方法最简单，系统开销最小，但是最不安全；第二类泄漏秘密的可能性比较小，因而安全性比第一类高，但是认证系统相对复杂；第三类的安全性最高，比如想窃取一个人的指纹是很困难的，但是涉及更复杂的算法和实现技术。

针对前两类基础的技术起步较早，目前相对成熟，应用也比较广泛。有关第三类的技术也由于它在安全性上的优势正在迅速发展。

7.1.2 数学基础

用 P 表示示证者，用 V 表示验证者，P 试图向 V 证明自己知道某信息。一种方法是 P

说出这一信息使得 V 相信，这样 V 也知道了这一信息，这是基于知识的证明；另一种方法是使用某种有效的数学方法，使得 V 相信他掌握这一信息，却不泄露任何有用的信息，这种方法被称为零知识证明问题。

零知识证明可以分为两大类：最小泄露证明(Minimum Disclosure Proof)和零知识证明(Zero Knowledge Proof)。

最小泄露证明需要满足：

(1) P 几乎不可能欺骗 V：如果 P 知道证明，他可以使 V 以极大的概率相信他知道证明；如果 P 不知道证明，则他使得 V 相信他知道证明的概率几乎为零。

(2) V 几乎不可能知道证明的知识，特别是他不可能向别人重复证明过程。

零知识证明除了要满足以上两个条件之外，还要满足第三个条件：

(3) V 无法从 P 那里得到任何有关证明的知识。

零知识证明最通俗的例子是图 7.1 中所示的山洞问题。图中的山洞里 C、D 两点之间有一扇上锁的门，P 知道打开门的咒语，按照下面的协议 P 就可以向 V 证明他知道咒语但是不需要告诉 V 咒语的内容：

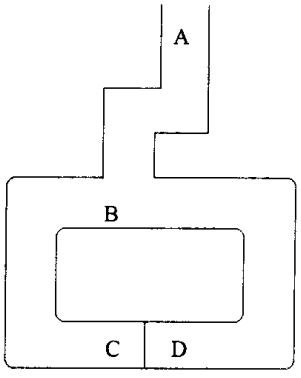


图 7.1 零知识问题

① V 站在 A 点；

② P 进入山洞，走到 C 点或 D 点；

③ 当 P 消失后，V 进入到 B 点；

④ V 指定 P 从左边或者右边出来；

⑤ P 按照要求出洞(如果需要通过门，则使用咒语)；

⑥ P 和 V 重复步骤①~⑤n 次。

如果 P 知道咒语，他一定可以按照 V 的要求正确地走出山洞 n 次；如果 P 不知道咒语并想使 V 相信他知道咒语，就必须每次都事先猜对 V 会要求他从哪一边出来，猜对一次的概率是 0.5，猜对 n 次的概率就是 0.5^n，当 n 足够大时，这个概率接近于零。因此山洞问题满足零知识证明的所有条件。

7.1.3 协议基础

认证协议可以分为双向认证协议(mutual authentication protocol)和单向认证协议(one-way authentication protocol)。双向认证协议是最常用的协议，它使得通信双方互相认证对方的身份。单向认证协议是通信的一方认证另一方的身份，比如服务器在提供用户申请的服务之前，先要认证用户是否是这项服务的合法用户，但是不需要向用户证明自己的身份。

基于对称密钥的认证协议往往需要双方事先已经通过其他方式(比如电话、信函或传递等物理方法)拥有共同的密钥。而基于公钥的认证协议的双方一般需要知道对方的公钥。公钥的获得相对于对称密钥要简便，比如通过 CA 获得对方的数字证书，这是基于公钥认证协议的优势。但是，公钥的缺点是加/解密速度慢、代价大，所以认证协议的最后，双方要协商出一个对称密钥作为下一步通信的会话密钥。产生会话密钥还有安全性方面的考虑：一旦会话密钥泄漏，则只会泄漏本次通信的内容，而不会带来更大的损失，而且通信者一旦发现会话密钥泄漏后，就可以用原有的密钥协商出新的会话密钥，重新开始新的会话。所以，无论是基于公钥还是基于对称密钥的认证协议，都要经常产生对称会话密钥。

在针对协议的攻击手法中，消息重放攻击(replay 攻击)是一种很常用的主动攻击。消息重放攻击最典型的情况是：A 与 B 通信时，第三方 C 窃听并获得了 A 过去发给 B 的消息 M，然后冒充 A 的身份将 M 发给 B，希望能够以 A 的身份与 B 建立通信，并从中获得有用信息。在最坏情况下，重放攻击可能导致被攻击者误认对方身份，暴露密钥和其他重要信息，至少也可以干扰系统的正常运行。

防止重放攻击的常用方式有时间戳方式和提问/应答方式两种。

时间戳(Time Stamp)方式的基本思想是：A 接受一个新消息当且仅当该消息包含一个时间戳，并且该时间戳在 A 看来是足够接近 A 所知道的当前时间。这种方法要求不同参与者之间的时钟同步。

时间戳方法不能用于面向连接的应用，最大的困难是时钟同步问题：

(1) 由于某一方的时钟机制故障可能导致临时失去同步，这将增大攻击成功的机会；

(2) 由于变化的和不可预见的网络延迟，不能期望分布式时钟保持精确的同步。

因此，任何基于时间戳的过程必须采用时间窗的方式来处理：一方面，时间窗应足够大，以包容网络延迟；另一方面，时间窗应足够小，最大限度地减小遭受攻击的机会。

另一种提问/应答(Challenge/Response)方式不需要时钟同步。提问/应答方式的基本原理是：A 期望从 B 获得一个新消息，首先发给 B 一个临时值(challenge)，并要求后续从 B 收到的消息(response)中都包含这个临时值或是由这个临时值进行某种事先约定的计算后的正确结果。这个临时值往往是一个随机数，称为现时(nonce)。

提问/应答方式的缺点是不适应非连接性的应用，因为它要求在传输开始之前先有握手，要求实时性较高。

7.2 身份认证协议

认证协议按照认证的方向可以分为双向认证协议和单向认证协议，按照使用的密码技

术可以分为基于对称密码的认证协议和基于公钥密码的认证协议。

本节基于这两种分类介绍并分析几种基本的认证协议。

7.2.1 双向认证协议

双向认证就是使通信双方确认对方的身份，适用于通信双方同时在线的情况。

1. 基于对称密码的双向认证协议

Needham/Schroeder 是一个基于对称加密算法的协议，它要求有可信任的第三方 KDC 参与，采用 Challenge/Response 的方式，使得 A、B 互相认证对方的身份。协议过程是：

(1) $A \rightarrow KDC$：$ID_A \| ID_B \| N_1$；

(2) $KDC \rightarrow A$：$E_{K_a}[K_S \| ID_B \| N_1 \| E_{K_b}[K_S \| ID_A]]$；

(3) $A \rightarrow B$：$E_{K_b}[K_S \| ID_A]$；

(4) $B \rightarrow A$：$E_{K_S}[N_2]$；

(5) $A \rightarrow B$：$E_{K_S}[f(N_2)]$。

其中，KDC 是密钥分发中心；ID_A 表示 A 身份的惟一标识；密钥 K_a 和 K_b 分别是 A 和 KDC、B 和 KDC 之间共享的密钥；N_1 和 N_2 是两个 nonce；$f(N)$是对 N 进行一个运算，比如 $f(N)=N+1$。本协议的目的是认证 A 和 B 的身份后，安全地分发一个会话密钥 K_S 给 A 和 B。

第(1)步，A 向 KDC 申请要和 B 通信。A 在第(2)步安全地得到了一个新的会话密钥。第(3)步，只能由 B 解密并理解，B 也获得会话密钥。第(4)步，B 告诉 A 已知道正确的 K_S 了，从而证明了 B 的身份。第(5)步表明 B 相信 A 也知道 K_S，从而认证了 A 的身份。

第(4)、第(5)步的目的是为了防止某种类型的重放攻击，特别是如果攻击方能够在第(3)步捕获该消息并重放之，这将在某种程度上干扰破坏 B 方的运行操作。

由于除了 KDC 之外只有 A 知道 K_a，也只有 B 知道 K_b，所以双方都可以向对方证明自己的身份。

但是，这个协议仍然有漏洞。假定攻击方 C 已经掌握 A 和 B 之间通信的一个老的会话密钥，C 可以在第(3)步冒充 A，利用老的会话密钥欺骗 B。除非 B 记住所有以前使用的与 A 通信的会话密钥，否则 B 无法判断这是一个重放攻击。然后，如果 C 可以中途阻止第(4)步的握手信息则可以冒充 A 在第(5)步响应。从这一点起 C 就可以向 B 发送伪造的消息，而 B 认为是在与 A 进行正常的通信。

Denning 结合了时间戳的方法，进行了改进：

(1) $A \rightarrow KDC$：$ID_A \| ID_B$；

(2) KDC → A: $E_{K_a}[K_S \| ID_B \| T \| E_{K_b}[K_S \| ID_A \| T]]$;

(3) A → B: $E_{K_b}[K_S \| ID_A \| T]$;

(4) B → A: $E_{K_S}[N_1]$;

(5) A → B: $E_{K_S}[f(N_1)]$;

$|Clock - T| < \Delta t_1 + \Delta t_2$。

其中，T 是时间戳；Δt_1 是 KDC 时钟与本地时钟(A 或 B)之间差异的估计值；Δt_2 是预期的网络延迟时间。

Denning 协议比 Needham/Schroeder 协议在安全性方面增强了一步，然而这又提出了新的问题，即必须依靠时钟同步。

如果发送者的时钟比接收者的时钟要快，攻击者就可以从发送者处窃听消息，并等待时间戳对接收者来说成为当前时刻时重放给接收者，这种重放将会得到意想不到的后果。这类攻击称为抑制重放攻击。

2. 基于公钥密码的双向认证协议

使用公钥密码算法，可以克服基于对称密码的认证协议中的一些问题。但同样需要有可信的第三方参与，现以 WOO92b 协议为例来说明：

请看 WOO92b 协议，同样需要有可信的第三方参与：

(1) A → KDC: $ID_A \| ID_B$;

(2) KDC → A: $E_{KR_{auth}}[ID_B \| KU_b]$;

(3) A → B: $E_{KU_b}[N_a \| ID_A]$;

(4) B → KDC: $ID_B \| ID_A \| E_{KU_{auth}}[N_a]$;

(5) KDC → B: $E_{KR_{auth}}[ID_A \| KU_a] \| E_{KU_b}[E_{KR_{auth}}[N_a \| K_S \| ID_A \| ID_B]]$;

(6) B → A: $E_{KU_a}[E_{KR_{auth}}[N_a \| K_S \| ID_A \| ID_B] \| N_b]$;

(7) A → B: $E_{K_S}[N_b]$。

其中，KU_a 是 A 的公钥；KR_a 是 A 的私钥；KU_{auth} 是 KDC 的公钥；KR_{auth} 是 KDC 的私钥。

第(1)步，A 向 KDC 提出和 B 通信。第(2)步，A 得到 B 的公钥。第(3)步，A 向 B 提出通信要求，包含一个现时 N_a。第(4)步，B 向 KDC 询问 A 的公钥。第(5)步，B 得到 A 的公钥和一段 KDC 签名的消息。第(6)步，B 将这段消息和现时 N_b 发给 A，A 在 KDC 签名的消息中找到 N_a，知道这不是一个重放。第(7)步，A 使用刚得到的会话密钥回答 B。

协议中，A 和 B 都向 KDC 索取对方的公钥，如果对方能正确解密用其公钥加密的消

息，就能够证明对方的身份。

7.2.2 单向认证协议

许多单向认证的应用(比如 email)不需要双方同时在线。一方在向对方证明自己身份的同时，即可发送数据；另一方收到后，首先验证发送方的身份，如果身份有效，就可以接受数据。

单向认证协议中只有一方向另一方证明自己的身份，因此过程一般都相对简单。下面是一个不需要第三方的基于对称密码的单向认证协议，要求 A 和 B 事先拥有共享密钥。

(1) $A \rightarrow B$: $ID_A \| N_1$；

(2) $B \rightarrow A$: $E_{K_{ab}}[K_S \| ID_B \| f(N_1) \| N_2]$；

(3) $A \rightarrow B$: $E_{K_S}[f(N_2)]$, 即 $f(x) = x + 1$。

第(1)步，A 将自己的身份和一个 nonce 发给 B，希望和 B 通信。第(2)步，B 生成一个会话密钥 K_S，连同对 Anonce 的应答和一个新的 nonce，一起用两人的共享密钥加密后发给 A；A 收到后，如果能用共享密钥解密得到 nonce 的正确应答，则相信 B 的身份，同时得到 K_S。第(3)步，A 计算出第 2 个 nonce 的应答，用 K_S 加密，发给 B；B 收到后如果能正确解密出应答，则相信 A 的身份，因为只有 A 拥有两人的共享密钥。

这种方案的缺点是双方都必须等待对方的回答，不适用于双方不同时在线的情况。

基于第三方的方案可以避免这一问题，第三方一般长期在线，双方都可以及时和第三方进行通信。下述方案要求通信双方都事先与第三方有共享密钥。

(1) $A \rightarrow KDC$: $ID_A \| ID_B \| N_1$；

(2) $KDC \rightarrow A$: $E_{K_a}[K_S \| ID_B \| N_1 \| E_{K_b}[K_S \| ID_A]]$；

(3) $A \rightarrow B$: $E_{K_b}[K_S \| ID_A] \| E_{K_S}[M]$。

第(1)步，A 向 KDC 要求和 B 通信，同时发给 KDC 一个 nonce。第(2)步，KDC 发给 A 一个用 A 的密钥加密的消息，包括一个会话密钥、A 发的 nonce、一段用 B 的密钥加密的消息；同时 A 解密得到 K_S。第(3)步，A 将用 B 的密钥加密的那段消息和用 K_S 加密的数据一起发给 B；B 收到后首先解密得到 A 的身份标识和 K_S，然后就可以解密 A 发来的数据了。

以上方案中，A 向 B 认证自己身份的同时将数据一同发给 B，并不要求 B 当时在线。但是这种方案不能抵挡重放攻击，第(3)步中 B 收到消息不能判定这是一个真实的消息还是一个重放。

基于公钥密码的单向身份认证协议可以非常简单，例如如下方案：

$A \rightarrow B$: $E_{KU_b}[K_S] \| E_{K_S}[M \| E_{KR_a}[H(M)]]$

这种方案要求 A 和 B 互相知道对方的公钥。首先，A 用 B 的公钥加密一个会话密钥，然后 A 用会话密钥加密数据和用自己私钥签名的消息摘要，A 将这些内容一起发送给 B。B 收到后，首先用自己的私钥解密出会话密钥，然后解密消息和 A 的签名，最后计算出消息摘要以验证 A 的签名，从而确定 A 的身份。

7.3　身份认证的实现

本节讲述实际应用中的几种身份认证协议。

7.3.1　拨号认证协议

在拨号环境中，要进行两部分的认证：首先是用户的调制解调器和网络访问服务器(NAS)之间使用 PPP 认证协议认证，这类认证协议包括 PAP、CHAP 等；然后是 NAS 和认证服务器(AS)之间进行认证，这类协议包括 TACACS+、RADIUS 等。整个认证过程可以用图 7.2 表示。

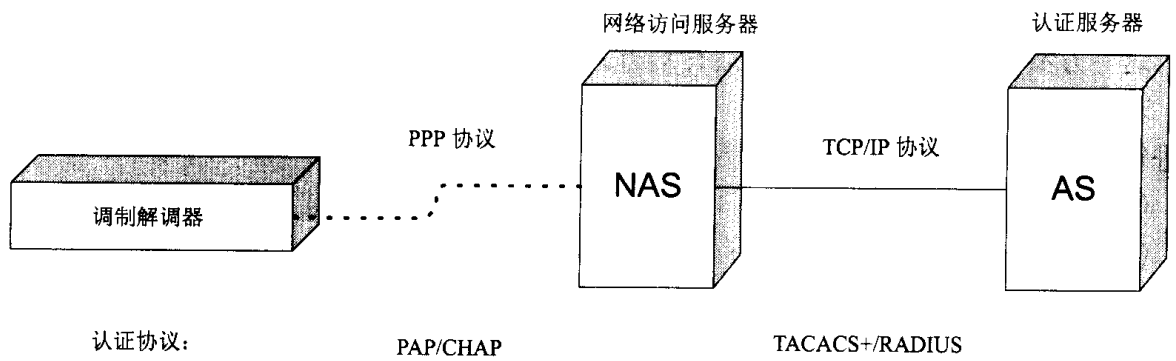


图 7.2　拨号网络认证

PPP(Point to Point Protocol)是点对点链路上的标准化 Internet IP 封装协议。在建立了链路之后，开始网络层协议之前，PPP 提供了一个可选的认证阶段。PPP 的帧格式如图 7.3 所示。

标志	地址	控制	协议	信息	FCS	标志

图 7.3　PPP 帧格式

其中，各个字段的含义如下：

(1) 标志：单字节，值为 01111110，指示一帧的开始或结尾；
(2) 地址：单字节，值为 11111111，标准广播地址；
(3) 控制：单字节，值为 00000011，请求无序列帧中用户数据的传输；
(4) 协议：双字节，标识封装在信息字段中的协议；
(5) FCS：Frame Check Sequence，即帧检查序列。

1. PPP 口令认证协议

口令认证协议(Password Authentication Protocol，PAP)结构简单，使对等实体通过双向握手进行认证。PPP PAP 建立初始链路的帧格式如图 7.4 所示。

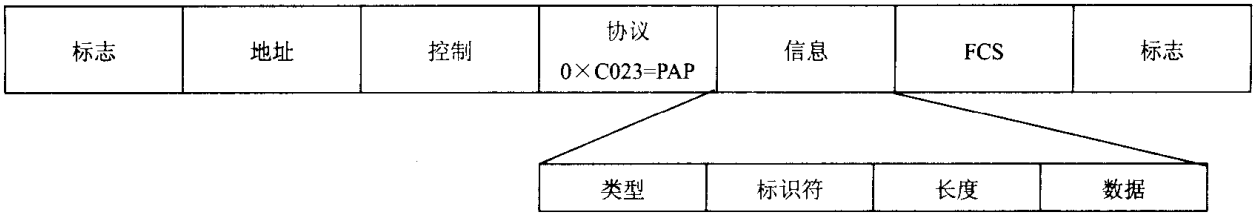


图 7.4 PPP PAP 帧格式

将 PPP 帧中的协议字段设置为 0xC023，信息字段内为 PAP 协议的内容。其中，各字段的解释是：

（1）类型：a)认证要求； b)认证确认； c)认证否认。
（2）标识符：单字节，帮助匹配请求和应答。
（3）长度：双字节，指出 PAP 包的长度，包括代码、标识符、长度和数据字段。
（4）数据：0 或多个字节。

链路建立起来以后，使用包含对等实体名称和口令的认证请求包初始化 PAP 认证。认证请求包内的信息字段格式如图 7.5 所示。

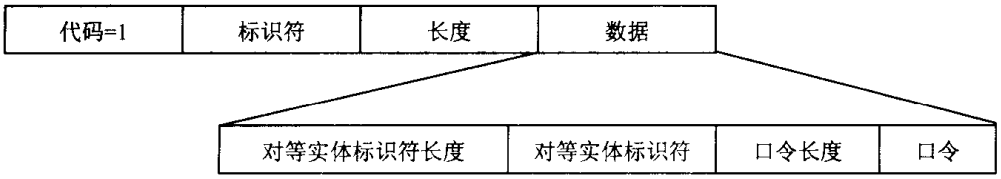


图 7.5 PPP PAP 认证请求帧格式

请求者不停地发送认证请求包，直到收到一个回复包或者重试计数器过期。认证者收到请求者发来的一个对等 ID/口令(Peer-ID/Password)对后对其进行比较。如果这个对等 ID/口令对是可识别、可接受的，认证者回复 Authenticate-Ack(Authenticate Acknowledge)；如果不

可识别或者不可接受，认证者回复 Authenticate-Nak(Authenticate Negative Acknowledge)。

PAP 不是强健的认证协议，主要有以下几点缺点：

(1) PAP 只认证对等实体；

(2) 口令在电路中以明文传输；

(3) 没有防止重放攻击或者反复的攻击。

2. PPP 质询–握手协议

质询–握手协议(Challenge-Handshake Authentication Protocol，CHAP)使用三次握手的方式，周期性地认证被认证者的身份。一旦连接建立起来，协议便可执行，并且可以在任意时刻重复执行。

CHAP 共有四种帧类型，图 7.6 是 CHAP 的帧格式。

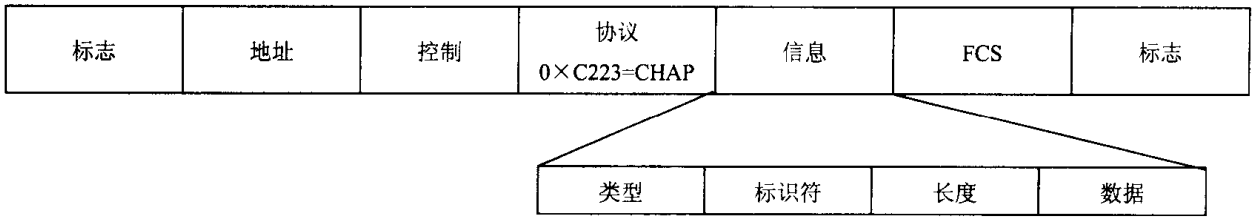


图 7.6　PPP CHAP 帧格式

将 PPP 帧中的协议字段设置为 0xC223，信息字段内为 CHAP 协议的内容。其中，各字段的解释是：

(1) 类型：a)质询；b)响应；c)失败；d)成功；

(2) 标识符：单字节，帮助匹配请求和应答；

(3) 长度：双字节，指出 CHAP 包的长度，包括代码、标识符、长度和数据字段；

(4) 数据：0 或多个字节。

CHAP 采用基于 nonce 的 Challenge/Response 算法，要求两个相互认证的实体事先共享一个密钥。如果需要双向认证，可以将 CHAP 按两个相反的方向执行两次，两次执行可以使用同一个共享密钥，但是更好的做法是使用两个不同的共享密钥。

CHAP 的具体认证过程是：

(1) 链路建立完成后，认证者发给被认证者一个 challenge，这个 challenge 具有惟一的标识符。

(2) 被认证者以 challenge 作为一个单向 Hash 函数的输入，计算出 response，发回给认证者，消息中还包括 challenge 的标识符。因为有可能认证者收不到 response，所以 response 是可以重发的，同时为了说明这个 response 是哪个 challenge 的应答，其中要包含 challenge

的标识符。

(3) 认证者比较收到的 response 和自己的计算结果是否相同，然后发送一个成功或者失败的消息给被认证者。被认证者发送 response 之后如果一定时间以后仍旧收不到成功或失败的结果，就会重发 response。

3. TACACS+协议

TACACS 协议是一种基于 UDP 的协议，最初由 BBN 开发，之后被 Cisco 多次扩展。TACACS+是 TACACS 的最新版本，基于 TCP 协议。

TACACS+是客户机/服务器型协议：TACACS+客户机通常是一个 NAS；而 TACACS+服务器是一个监控程序，监听 49 号端口。

TACACS+的基本设计组件有认证、授权和记账。

(1) TACACS+认证

TACACS+的认证是可选的，可以根据需要进行设置。TACACS+允许 TACACS+客户端采用任意认证协议(PPP PAP、PPP CHAP、Kerberos 等)，在拨号网络环境中，通常使用 PPP PAP 或 PPP CHAP。另外，TACACS+的认证交换包可以包括任意的长度和内容。

TACACS+认证用到三种类型的包：

- START，由 TACACS+客户机发送给 TACACS+服务器；
- CONTINUE，由 TACACS+客户机发送给 TACACS+服务器；
- REPLY，由 TACACS+服务器发送给 TACACS+客户机。

TACACS+的认证过程有以下 3 步：

- 客户机发送一个 START 包给服务器。START 包的内容包括被执行的认证类型(明文口令、PPP PAP 或 PPP CHAP 等)，还可能包括用户名等其他认证数据。START 包只在一个认证会话开始时使用一次，序列号永远是 1。
- 服务器收到 START 包以后，回送一个 REPLY 包，指示认证继续还是结束。如果认证继续，REPLY 包还需要指出需要哪些新信息。
- 如果客户机收到认证结束的 REPLY 包，则认证结束；如果收到认证继续的 REPLY 包，则向服务器发送 CONTINUE 包，其中包括服务器要求的信息，如此反复，直到认证结束。

(2) TACACS+授权

认证结束后，TACACS+客户机可以启动授权过程。如果一个未被认证的用户请求授权，由授权代理决定是否为其提供服务。

TACACS+授权过程分为以下两步：

- 客户端向服务器发送一个 REQUEST 包，内容包括用户和过程的真实性、请求授权

的服务和选项。

• 服务器回复客户端一个 RESPONSE 包。

(3) TACACS+记账

TACACS+在完成认证和授权之后通常要进行记账。

TACACS+有三种类型的记账记录：

• “开始记录”(Start Records)：指示一个服务即将开始；

• “停止记录”(Stop Records)：指示一个服务已经停止；

• “更新记录”(Update Records)：指示一个服务仍在进行。

TACACS+记账记录不仅包括了授权记录的全部信息，还包括了开始时间、结束时间等额外信息。

4．RADIUS 协议

RADIUS(Remote Address Dial-In User Service)是基于 UDP 的访问服务器认证和记账的客户机/服务器型协议，由 Livingston Enterprises 公司开发。RADIUS 客户机通常是一个 NAS，而服务器是一个监控程序。RADIUS 服务器同时可以作为其他 RADIUS 服务器的代理客户机，也可以作为其他任意类型的认证服务器的客户机。

RADIUS 的基本设计组件有认证、授权和记账。

(1) RADIUS 认证与授权

RADIUS 的认证与授权结合在一起,可以支持多种认证方法，包括 PPP PAP、PPP CHAP、UNIX 登陆等。RADIUS 的认证、授权过程有以下两步：

• RADIUS 客户机向 RADIUS 服务器发送 Access-Request 包，内容包括用户名、加密口令、客户机的 IP 地址和端口号，以及用户想要启动的会话类型；

• RADIUS 服务器收到 Access-Request 包后，在数据库中查询是否有此用户名的记录。如果没有，则加载一个默认的配置文件，或者返回一个 Access-Reject 消息，内容为拒绝访问的原因。如果数据库中有此用户名并且口令正确，服务器返回一个 Access-Accept 消息，内容包括用于该次会话的参数(包括服务类型、协议类型、分配给用户的 IP 地址、应用的访问列表或者 NAS 路由表中的静态路由)属性。

(2) RADIUS 记账

RADIUS 的记账独立于 RADIUS 的认证和授权。RADIUS 的记账功能使得在会话开始或结束时发送数据，用来指示会话期间所使用的资源量(时间、字节、包的数量等)。

7.3.2　Kerberos 认证协议

Kerberos 是由美国麻省理工学院(MIT)最初提出的，现在最常用的版本是第 4 版和第 5

版，第5版主要是修补了第4版中的一些安全漏洞。

Kerberos 是为 TCP/IP 网络设计的第三方鉴别协议。本节主要介绍它的设计动机、模型和第5版的认证过程。

1. Kerberos 的设计动机

Kerberos 需要解决的问题是：在一个公开的分布式网络中，工作站上的用户需要访问分布在网络中的服务器，所以希望服务器能够限制授权用户的访问，并能鉴别服务请求。Kerberos 的解决方案是调用每项服务时都需要用户证明自己的身份，同时也需要服务器向用户证明自己的身份。

第一个有关 Kerberos 的公开发表的报告列举了如下的 Kerberos 需求：

(1) 安全：网络窃听者不能获得必要信息以假冒其他用户。并且，Kerberos 应足够强壮以至于潜在的攻击者无法找到它的脆弱的连接。

(2) 可靠：Kerberos 应高度可靠，应该使用分布式服务器体系结构，并且能够使得一个系统备份另一个系统。

(3) 透明：理想的情况下，除了输入口令以外用户应感觉不到认证的发生。

(4) 可伸缩：系统应能够支持大数量的客户机和服务器。

为了满足这些要求，Kerberos 的总体方案是通过协议实现可信的第三方认证服务，这个协议是以 Needham/Schroeder 协议为基础的。假设 Kerberos 的协议设计得足够好，而且 Kerberos 服务器是安全的，那么认证服务就应该是安全的。

Kerberos 采用对称加密，其中第4版要求使用 DES 算法。

2. Kerberos 模型

Kerberos 模型中，实体包括客户机和服务器。客户机可以是用户，也可以是需要处理事物的独立的软件程序，例如下载文件、发送消息、访问数据库、访问打印机等。

Kerberos 系统中有一个数据库用来保存所有客户的秘密密钥。需要鉴别的网络服务以及希望使用这些服务的客户机，都需要向 Kerberos 注册秘密密钥。

由于 Kerberos 系统知道每个客户的秘密密钥，所以它可以向一个实体证实另一个实体的身份；然后可以产生一个会话密钥供双方通信使用；当这次会话结束后，Kerberos 会自动销毁会话密钥。

在 Kerberos 的认证过程中，需要使用两种凭证：票据(ticket)和鉴别码(authenticator)。

票据用于客户秘密地向服务器发送自己的身份标识，同时，服务器还可以通过票据中的信息证实使用票据的和发送票据的是同一个客户。鉴别码是另外一个凭证，与票据一同发送。

票据的格式是

$$T_{c,s} = s \| E_{K_s}[c, a, v, K_{c,s}]$$

其中，c 是客户机的标识，s 是服务器的标识，$T_{c,s}$ 是客户机 c 使用服务器 s 的票据，K_s 是 s 的秘密密钥，v 是票据的有效起止时间，$K_{c,s}$ 是 c 和 s 的会话密钥。

客户一旦获得票据，就可以多次使用它来访问服务器，直到票据过期。

鉴别码用来证明票据的发送方就是票据的拥有者，格式是

$$A_{c,s} = E_{K_{c,s}}[c, t, key]$$

其中，$A_{c,s}$ 是客户机 c 到服务器 s 的鉴别码，t 是时间戳，key 是可选附加会话密钥。

每个鉴别码只能使用一次，但是客户可以根据需要产生鉴别码。鉴别码的作用是：它以会话密钥进行加密，表明发送者也知道密钥；加封的时间戳使得攻击者无法重放。

在 Kerberos 系统中，有两种专门的服务器却用于认证：鉴别服务器(AS)和票据许可服务器(TGS)。鉴别服务器只有一个，票据许可服务器却可以有多个。由于协议中使用了时间戳，所以系统中应有一套同步机制。

3. Kerberos 认证过程

在第 5 版的整个认证过程中，消息有以下 5 个：

(1) $c \rightarrow AS$：c, tgs

(2) $AS \rightarrow c$：$E_{K_c}[K_{c,tgs}] \| E_{K_{tgs}}[T_{c,tgs}]$

(3) $c \rightarrow TGS$：$E_{K_{c,tgs}}[A_{c,s}] \| E_{K_{tgs}}[T_{c,tgs}]$

(4) $TGS \rightarrow c$：$E_{K_{c,tgs}}[K_{c,s}] \| E_{K_s}[T_{c,s}]$

(5) $c \rightarrow s$：$E_{K_{c,s}}[A_{c,s}] \| E_{K_s}[T_{c,s}]$

c 是客户，AS 是鉴别服务器，TGS 是票据许可服务器，s 是 c 要使用的服务器，其余的标记与前面相同。

这一过程可以分为三个阶段，参见图 7.7 所示。

(1) 票据许可票据的获取

客户给 Kerberos 鉴别服务器发送一个消息，包含客户名和票据许可服务器名。在实际实现中，一般是客户将自己的客户名输入系统，然后由注册程序发送该请求。

鉴别服务器在数据库中查找客户信息，如果客户在数据库中，鉴别服务器就为客户和 TGS 生成一个会话密钥，并用客户的秘密密钥加密。另外，鉴别服务器为客户产生一个票据许可票据(Ticket Granting Ticket，TGT)，并用 TGS 的秘密密钥加密，用来让客户向 TGS 证明自己的身份。鉴别服务器将这两部分一起发送给客户。

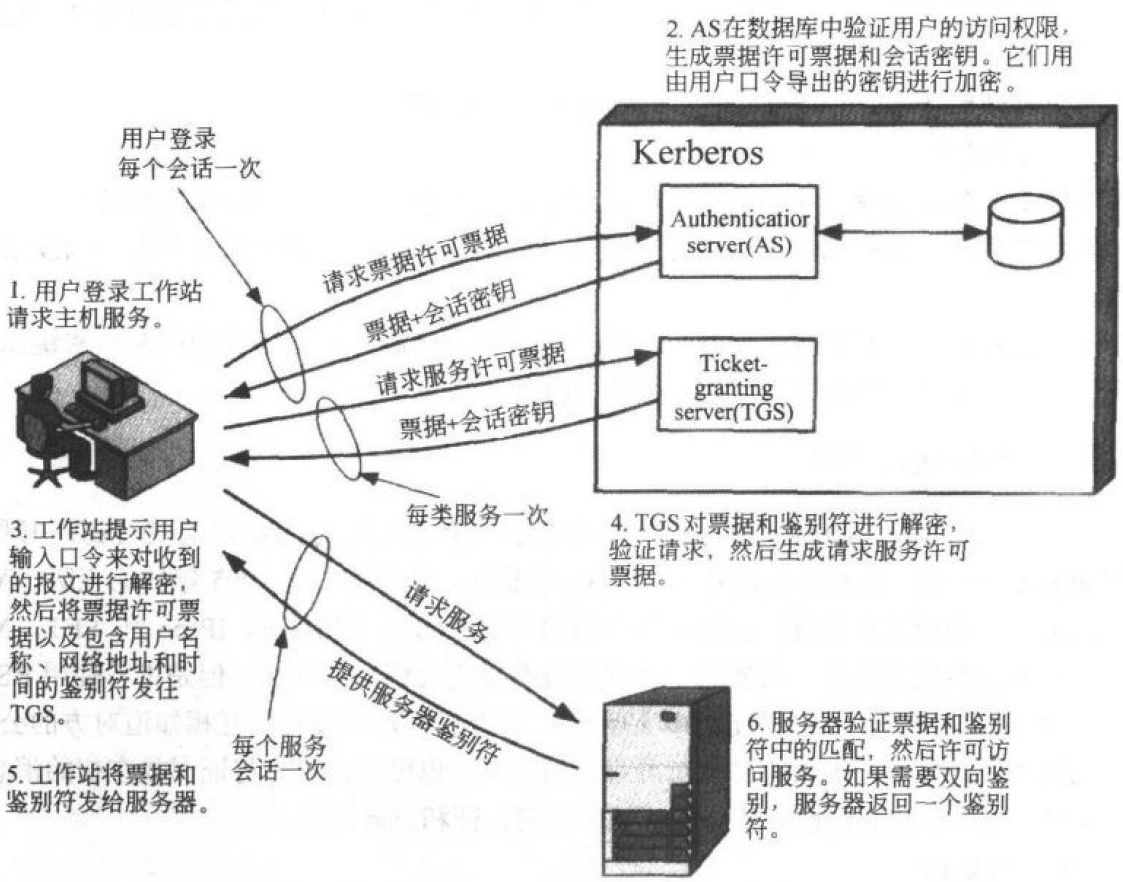


图 7.7 Kerberos 认证过程

客户收到后，解密第一部分，获得与 TGS 的会话密钥，并将 TGT 直接保存起来。这样，客户可以在 TGT 的有效期内向 TGS 证明自己的身份。

(2) 请求许可票据的获取

客户必须为他想使用的每一项服务申请请求许可票据，TGS 就负责给每个服务器分配票据。

客户需要使用一个自己没有票据的新服务时，就将 TGT 和一个用会话密钥加密的鉴别码发给 TGS。

TGS 收到消息后，首先用自己的秘密密钥解开 TGT，得到与客户的会话密钥；再用会话密钥解开鉴别码；然后比较鉴别码和票据里的信息、客户的网络地址和发送的请求地址、时间戳与当前时间是否一致，如果一致，则允许处理该请求。

TGS 为客户和服务器生成一个会话密钥，并用自己和客户的会话密钥加密；然后生成

一个针对该服务的请求许可票据，并用服务器的秘密密钥加密；最后将这两部分一起发给客户。

客户收到后解密第一部分，获得与服务器的会话密钥。

(3) 服务请求

本质上，TGS 也是一个服务器，所以这个阶段的消息与上一阶段非常类似。

客户产生一个鉴别码，用与服务器的会话密钥加密，连同请求许可票据一同发送给服务器。

服务器解密并检查票据、鉴别码、用户地址和时间戳，无误后返回用会话密钥加密的时间戳，表明客户通过认证，可以开始使用服务。

7.3.3 X.509 认证协议

X.509 协议(也称 ISO 鉴别框架)是基于公钥证书和 CA 的认证协议，有关公钥证书和 CA 的知识参见前一章。X.509 协议最早于 1988 年发布，最新版本是 1995 年的第三版。X.509 定义的证书结构和认证协议已经得到了广泛的应用，比如 S/MIME、IPsec 和 SET。X.509 基于公开密钥和数字签名。虽然这个协议并没有指定公钥加密算法，但是推荐使用 RSA。

在 X.509 中规定了几个可选的认证过程。所有过程都假设双方互相知道对方的公钥。一方可以通过访问 CA 提供的目录获得对方的证书，也可以由对方将证书放在初始消息里。

X.509 的三种认证过程支持单向认证、双向认证和三向认证。

(1) 单向认证

单向认证只需要一次通信。用户 A 向用户 B 发送一条消息，证明自己的身份。消息内容如下：

$\mathrm{A} \rightarrow \mathrm{B}$：$\mathrm{A}\{t_{\mathrm{A}}, r_{\mathrm{A}}, B, \mathrm{sgn}\, Data, E_{KU_b}[K_{ab}]\}$

其中时间戳 t_{A} 包含起始时间和终止时间，用来防止消息的延迟传送；r_{A} 是一个现时，在一个有效期内是惟一的，B 需要保存 r_{A} 直到有效期结束；B 是 B 的身份信息；sgnData 是签名数据；最后项是用 B 的公钥加密的会话密钥。整个消息还要由 A 签名，以保证完整性和抗抵赖性。

这种方式只认证了 A 的身份。由于使用了时间戳，需要时钟同步。

如果用于纯粹的认证，消息可以只用于向 B 提交证书。

(2) 双向认证

双向认证在单向认证的基础上增加了一次通信，过程如下：

(1) $\mathrm{A} \rightarrow \mathrm{B}$：$A\{t_{\mathrm{A}}, r_{\mathrm{A}}, B, \mathrm{sgn}\, Data, E_{KU_b}[K_{ab}]\}$；

(2) $\mathrm{B} \rightarrow \mathrm{A}$：$B\{t_{\mathrm{B}}, r_{\mathrm{B}}, A, r_{\mathrm{A}}, \mathrm{sgn}\, Data, E_{KU_a}[K_{ba}]\}$。

第二条消息与第一条相似，但是增加了现时 r_A 的回应，要由 B 签名。

这种方式允许双方相互认证，但是两条消息中有冗余信息。双向认证也需要时间同步。

(3) 三向认证

三向认证又在最后增加了一个 A 向 B 的回答。过程如下：

(1) $A \rightarrow B$: $A\{t_A, r_A, B, \text{sgn } Data, E_{KU_b}[K_{ab}]\}$；

(2) $B \rightarrow A$: $B\{t_B, r_B, A, r_A, \text{sgn } Data, E_{KU_a}[K_{ba}]\}$；

(3) $A \rightarrow B$: $A\{r_B\}$。

最后一条消息包含现时 r_B 的签名备份，这样就无需检查时间戳，因为每一端都可以通过检查返回的现时来探测重放攻击。所以三向认证不需要时钟同步，在没有时钟同步条件时，必须采用这种方法。

习 题 七

1. 试述零知识证明的原理。
2. 在身份认证中如何对抗重放攻击？在基于时间戳的认证中，当时钟不同步时，如何实现身份欺骗？
3. 安全的口令应该满足哪些原则？
4. 试述采用 challenge/response 与一次性口令认证的区别。
5. 描述采用 CHAP 和 RADIUS 进行拨号接入的完整的身份认证流程。
6. Kerberos 认证协议实现身份认证的什么特点？如何实现由多个 TGS 组成的分布式认证？
7. 如何用 X.509 证书实现基于 challenge/response 的双向认证。

第八章　访 问 控 制

访问控制是在保障授权用户能获取所需资源的同时拒绝非授权用户的安全机制。访问控制也是信息安全理论基础的重要组成部分。本章讲述访问控制的原理、作用、分类和研究前沿，重点介绍较典型的自主访问控制、强制访问控制和基于角色的访问控制。

8.1　访问控制原理

访问控制与其他安全措施之间的关系可以用图 8.1 来简要说明。在用户身份认证(如果必要)和授权之后，访问控制机制将根据预先设定的规则对用户访问某项资源(目标)进行控制，只有规则允许时才能访问，违反预定的安全规则的访问行为将被拒绝。资源可以是信息资源、处理资源、通信资源或者物理资源，访问方式可以是获取信息、修改信息或者完成某种功能，一般情况可以理解为读、写或者执行。

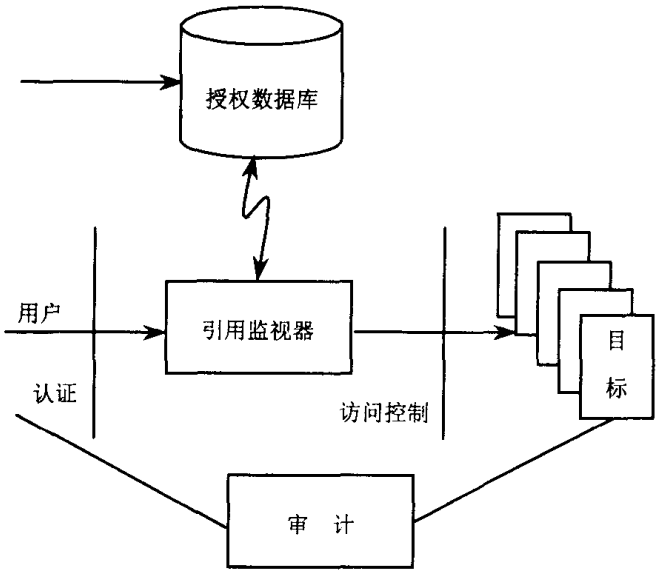


图 8.1　访问控制与其他安全措施的关系模型

访问控制的目的是为了限制访问主体对访问客体的访问权限，从而使计算机系统在合法范围内使用；它决定用户能做什么，也决定代表一定用户身份的进程能做什么。其中主

体可以是某个用户，也可以是用户启动的进程和服务。为达到此目的，访问控制需要完成以下两个任务：

(1) 识别和确认访问系统的用户；

(2) 决定该用户可以对某一系统资源进行何种类型的访问。

实现访问控制的一般模型是访问矩阵。访问控制机制可以用一个三元组来表示：(*S*，*O*，*A*)。其中，*S* 代表主体集合(Subject set)；*O* 代表客体集合(Object set)；*A* 代表属性集合(Attribution set)。

对于任意一个 $s_i \in S, o_j \in O$，都存在相应的一个 $a_{ij} \in A$，且 $a_{ij} = P(s_i, o_j)$，其中 P 是访问权限的函数。a_{ij} 就代表了 s_i 可以对 o_j 执行什么样的操作。这一关系可以用一个访问控制矩阵来表示：

$$A = \begin{bmatrix} a_{00} & a_{01} & \cdots & a_{0n} \\ a_{10} & a_{11} & \cdots & a_{1n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m0} & a_{m1} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} S_0 \\ S_1 \\ \cdots \\ S_m \end{bmatrix} = [O_0 \quad O_1 \quad \ldots \quad O_n]$$

其中，$S_i(i = 0,1,\cdots,m)$ 是主体 S_i 对所有客体的权限集合，$O_j(j = 0,1,\cdots,n)$ 是客体 o_j 对所有主体的访问权限集合。

访问控制一般包括三种类型：自主访问控制、强制访问控制和基于角色的访问控制。下面各小节将分别进行介绍。

8.2 自主访问控制

自主访问控制(Discretionary Access Control，简称 DAC)是一种常用的访问控制方式，它基于对主体或主体所属的主体组的识别来限制对客体的访问，这种控制是自主的。自主是指主体能够自主的(可能是间接的)将访问权或访问权的某个子集授予其他主体。简单来说，自主访问控制就是由拥有资源的用户自己来决定其他一个或一些主体可以在什么程度上访问哪些资源。

自主访问控制是一种比较宽松的访问控制，一个主体的访问权限具有传递性。比如大多数交互系统的工作流程是这样的：用户首先登陆，然后启动某个进程为该用户做某项工作，这个进程就继承了该用户的属性，包括访问权限。这种权限的传递可能会给系统带来安全隐患，某个主体通过继承其他主体的权限而得到了它本身不应具有的访问权限，就可能破坏系统的安全性。这是自主访问控制方式的缺点。

为了实现完整的自主访问系统，访问控制矩阵内的内容必须以某种形式进行保存和管理。如果把整个矩阵保存下来，不仅实现起来不方便，而且效率很低。由前面访问控制矩阵的公式可以看出，矩阵中的一行表示一个主体的所有权限；一列则是关于一个客体的所有权限；矩阵中的元素是该元素所在行对应的主体对该元素所在列对应的客体的访问权限。根据这一特点，具体实现时，往往是基于矩阵的行或者列来表达访问控制信息。下面将分别介绍这两种方法。

8.2.1　访问控制表

访问控制表(Access Control List，简称 ACL)是基于访问控制矩阵中列的自主访问控制。它在一个客体上附加一个主体明晰表，来表示各个主体对这个客体的访问权限。明晰表中的每一项都包括主体的身份和主体对这个客体的访问权限。如果使用组(group)或者通配符(wildcard)的概念，可以有效地缩短表的长度。

访问控制表是实现自主访问控制比较好的方式，下面通过例子进行详细说明。

对系统中一个需要保护的客体 o_j 附加的访问控制表的结构如图 8.2 所示。

O_j	s_0.re	s_1.r	s_2.e	…	s_m.rew

图 8.2　访问控制表举例

在上图的例子中，对于客体 o_j，主体 s_0 具有读(r)和执行(e)的权利；主体 s_1 只有读的权利；主体 s_2 只有执行的权利；主体 s_m 具有读、写(w)和执行的权利。

但是，在一个很大的系统中，可能会有非常多的主体和客体，这就导致访问控制表非常长，占用很多的存储空间，而且访问时效率下降。解决这一问题就需要分组和使用通配符。

在实际的多用户系统中，用户可以根据部门结构或者工作性质被分为有限的几类。一般来说，一类用户使用的资源基本上是相同的。因此，可以把一类用户作为一个组，分配一个组名，简称“GN”，访问是可以按照组名判断。通配符“*”可以代替任何组名或者主体标识符。这时，访问控制表中的主体标识为：

主体标识=ID.GN。

其中，ID 是主体标识符，GN 是主体所在组的组名。

请看图 8.3 的示例。

O_j	liu.INFO.rew	*.INFO.re	zhang. *.r	*.*.n

图 8.3　带有组和通配符的访问控制表示例

在上图的访问控制表中，属于 INFO 组的所有主体都对客体 o_j 具有读和执行的权利；但是只有 INFO 组中的主体 liu 才额外具有写的权限；无论是哪一组中的 zhang 都可以读客体 o_j；最后一个表项说明所有其他的主体，无论属于哪个组，都不具备对 o_j 有任何访问权限。

在访问控制表中还需要考虑的一个问题是缺省问题。缺省功能的设置可以方便用户的使用，同时也避免了许多文件泄露的可能。最基本的，当一个主体生成一个客体时，该客体的访问控制表中对应生成者的表项应该设置成缺省值，比如具有读、写和执行权限。另外，当某一个新的主体第一次进入系统时，应该说明它在访问控制表中的缺省值，比如只有读的权限。

8.2.2 访问能力表

访问能力表(Access Capabilities List)是最常用的基于行的自主访问控制。能力(capability)是为主体提供的、对客体具有特定访问权限的不可伪造的标志，它决定主体是否可以访问客体以及以什么方式访问客体。主体可以将能力转移给为自己工作的进程，在进程运行期间，还可以添加或者修改能力。能力的转移不受任何策略的限制，所以对于一个特定的客体，不能确定所有有权访问它的主体。因此，访问能力表不能实现完备的自主访问控制，而访问控制表是可以实现的。利用访问能力表实现的自主访问控制系统不是很多，其中只有少数系统试图通过增加其他措施实现完备的自主访问控制。

图 8.4 举例说明了访问能力表的样式。

图 8.4 访问能力表示例

上图是主体 s_i 的访问能力表。图中每一表项包括客体的标识和 s_i 对该客体的访问能力。如图所示，s_i 是客体 o_0 的拥有(own，简写为 o)者，并对它具有最大的访问能力(读、写、执行)；s_i 对客体 o_1 只有读的能力；s_i 对客体 o_n 具有读和写的能力。

能力机制的最大特点是能力的拥有者可以在主体中转移能力。在转移的能力中有一种叫做“转移能力”，它允许接受能力的主体继续转移能力。比如，进程 A 将某个能力的拷贝转移给进程 B，B 又将能力的拷贝传递给进程 C。如果 B 不想让 C 继续转移这个能力，就在转移给 C 的能力拷贝中去掉转移能力，这样 C 就不能转移能力了。主体为了在能力取消时从所有主体中彻底清除自己的能力，需要跟踪所有的转移。

8.3 强制访问控制

自主访问控制的最大特点是自主，即资源的拥有者对资源的访问策略具有决策权，因此是一种限制比较弱的访问控制策略。这种方式给用户带来灵活性的同时，也带来了安全

隐患。

在一些系统中，需要更加强硬的控制手段，强制访问控制(Mandatory Access Control，简称 MAC)就是其中的一种机制。

强制访问控制系统为所有的主体和客体指定安全级别，比如绝密级、机密级、秘密级和无密级。不同级别标记了不同重要程度和能力的实体。不同级别的主体对不同级别的客体的访问是在强制的安全策略下实现的。

在强制访问控制机制中，将安全级别进行排序，如按照从高到低排列，规定高级别可以单向访问低级别，也可以规定低级别可以单向访问高级别。这种访问可以是读，也可以是写或修改。在 Bell Lapadula 模型中，信息的完整性和保密性是分别考虑的，因而对读、写的方向进行了反向规定，如图 8.5 所示。

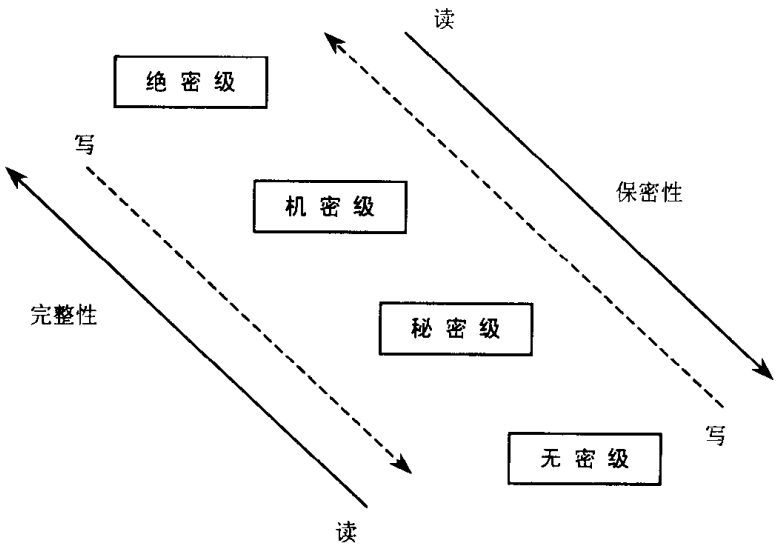


图 8.5　MAC 模型

(1) 保障信息完整性策略：为了保障信息的完整性，低级别的主体可以读高级别客体的信息(不保密)，但低级别的主体不能写高级别的客体(保障信息完整)，因此采用的是上读/下写策略。即属于某一个安全级的主体可以读本级和本级以上的客体，可以写本级和本级以下的客体。比如机密级主体可以读绝密级、机密级的客体，可以写机密级、秘密级、无密级的客体。这样，低密级的用户可以看到高密级的信息，因此，信息内容可以无限扩散，从而使信息的保密性无法保障；但低密级的用户永远无法修改高密级的信息，从而保障信息的完整性。

(2) 保障信息机密性策略：与保障完整性策略相反，为了保障信息的保密性，低级别的主体不可以读高级别的信息(保密)，但低级别的主体可以写高级别的客体(完整性可能破

坏)，因此采用的是下读/上写策略。即属于某一个安全级的主体可以写本级和本级以上的客体，可以读本级和本级以下的客体。比如机密级主体可以写绝密级、机密级的客体，可以读机密级、秘密级、无密级的客体。这样，低密级的用户可以修改高密级的信息，因此，信息完整性得不到保障；但低密级的用户永远无法看到高密级的信息，从而保障信息的保密性。

实体的安全级别是由敏感标记(sensitivity label)来表示的。敏感标记，简称标记，是表示实体安全级别的一组信息，在安全机制中把敏感标记作为强制访问控制决策的依据。当输入未加安全级别的数据时，系统应该向授权用户要求这些数据的安全级别，并对收到的安全级别进行审计。

自主访问控制较弱，而强制访问控制又太强，会给用户带来许多不便。因此，实际应用中，往往将自主访问控制和强制访问控制结合在一起使用。自主访问控制作为基础的、常用的控制手段；强制访问控制作为增强的、更加严格的控制手段。某些客体可以通过自主访问控制保护，重要课题必须通过强制访问控制保护。

下面介绍一个实例：Unix 文件系统强制访问控制机制的 Multics 方案。

Multics 方案是 Unix 文件系统强制访问控制机制的多种方案之一，其他的还有 Linus IV 方案、Secure Xenix 方案、Tim Thomas 方案等。

在 Multics 方案中，文件系统和 Unix 文件系统一样，是一个树形结构，所有的用户和文件(包括目录文件)都有一个相应的安全级。用户对文件的访问需要遵守下述安全策略：

① 仅当用户的安全级别不低于文件的安全级别时，用户才可以读文件；

② 仅当用户的安全级别不高于文件的安全级别时，用户才可以写文件。

这就是前面详细介绍过的保密性策略。

一个文件的创建和删除被认为是对文件所在目录(文件的父目录)的写操作，所以当一个用户创建或者删除文件时，他的安全级一定不能高于文件父目录的安全级。这种限制与 Unix 文件系统是不相容的，因为在 Unix 文件系统中，有些用户可以读安全级别比自己低的文件。例如，在 Unix 文件系统中有一个共享的/TMP 目录用于存放临时文件，为使用户能读他们存放在/TMP 下的文件，用户的安全级别应该不低于/TMP 的安全级。这就与 Multics 方案中的强制访问控制策略矛盾，因为在 Multics 方案中，用户如果想写/TMP 目录，他的安全级别就必须不高于/TMP 目录的安全级别。

8.4 基于角色的访问控制

在传统的访问控制中，主体始终是和特定的实体捆绑对应的。例如，用户以固定的用

户名注册，系统分配一定的权限，该用户将始终以该用户名访问系统，直至销户。其间，用户的权限可以变更，但必须在系统管理员的授权下才能进行。然而在现实社会中，这种访问控制方式表现出很多弱点，不能满足实际需求。主要问题在于：

(1) 同一用户在不同的场合需要以不同的权限访问系统，按传统的做法，变更权限必须经系统管理员授权修改，因此很不方便。

(2) 当用户量大量增加时，按每用户一个注册账号的方式将使得系统管理变得复杂、工作量急剧增加，也容易出错。

(3) 传统访问控制模式不容易实现层次化管理。即按每用户一个注册账号的方式很难实现系统的层次化分权管理，尤其是当同一用户在不同场合处在不同的权限层次时，系统管理很难实现。除非同一用户以多个用户名注册。

基于角色的访问控制模式(Role Based Access Control，　RBAC)就是为克服以上问题而提出来的。在基于角色的访问控制模式中，用户不是自始至终以同样的注册身份和权限访问系统，而是以一定的角色访问，不同的角色被赋予不同的访问权限，系统的访问控制机制只看到角色，而看不到用户。用户在访问系统前，经过角色认证而充当相应的角色。用户获得特定角色后，系统依然可以按照自主访问控制或强制访问控制机制控制角色的访问能力。

8.4.1　角色的概念

在基于角色的访问控制中，角色(role)定义为与一个特定活动相关联的一组动作和责任。系统中的主体担任角色，完成角色规定的责任，具有角色拥有的权限。一个主体可以同时担任多个角色，它的权限就是多个角色权限的总和。基于角色的访问控制就是通过各种角色的不同搭配授权来尽可能实现主体的最小权限(最小授权指主体在能够完成所有必需的访问工作基础上的最小权限)。

例如，在一个银行系统中，可以定义出纳员、分行管理者、系统管理员、顾客、审计员等角色。其中，担任系统管理员的用户具有维护系统文件的责任和权限，无论这个用户具体是谁。系统管理员可能是由某个出纳员兼任，他就具有两种角色。但是出于责任分离的考虑，需要对一些权利集中的角色组合进行限制，比如规定分行管理者和审计员不能由同一个用户担任。

基于角色的访问控制可以看做是基于组的自主访问控制的一种变体，一个角色对应一个组。

8.4.2　基于角色的访问控制

明确了角色的概念与作用，容易理解，基于角色的访问控制就是通过定义角色的权限，

为系统中的主体分配角色来实现访问控制的，其一般模型如图 8.6 所示。用户先经认证后获得一定角色，该角色被分派了一定的权限，用户以特定角色访问系统资源，访问控制机制检查角色的权限，并决定是否允许访问。

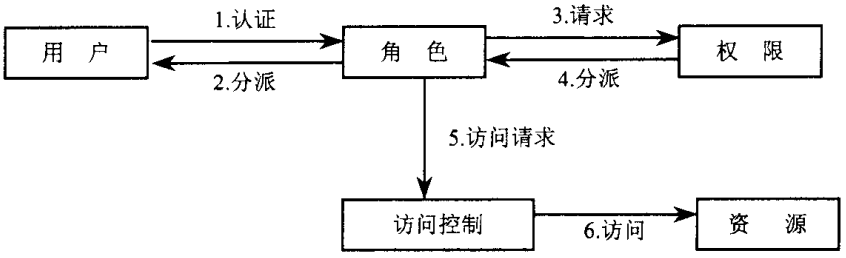


图 8.6 RBAC 模型

这种访问控制方法的具体特点有：

(1) 提供了三种授权管理的控制途径：

a) 改变客体的访问权限，即修改客体可以由哪些角色访问以及具体的访问方式；

b) 改变角色的访问权限；

c) 改变主体所担任的角色。

(2) 系统中所有角色的关系结构可以是层次化的，便于管理。角色的定义是从现实出发，所以可以用面向对象的方法来实现，运用类和继承等概念表示角色之间的层次关系非常自然而且实用。

(3) 具有较好的提供最小权利的能力，从而提高了安全性。由于对主体的授权是通过角色定义，因此调整角色的权限粒度可以做到更有针对性，不容易出现多余权限。

(4) 具有责任分离的能力。定义角色的人不一定是担任角色的人，这样，不同角色的访问权限可以相互制约，因而具有更高的安全性。

下面通过一个具体实例来说明基于角色的访问控制策略。例如，前面已经定义了角色的银行系统，可是设计如下的访问策略：

(1) 允许出纳员修改顾客的账号记录(包括存款、取款、转账等)，并允许出纳员询问所有账号的注册项；

(2) 允许分行管理者修改顾客的账号记录(包括存款、取款，但不包括规定的资金数目的范围)，并允许分行管理者查询所有账号的注册项，还可以创建和取消账号；

(3) 允许一个顾客询问自己的注册项，但不能询问其他任何的注册项；

(4) 允许系统管理员询问系统注册项和开关系统，但不允许读或修改顾客的账号信息；

(5) 允许审计员阅读系统中所有的信息，但不允许修改任何信息。

这种策略陈述具有很明显的优势，包括：

(1) 表示方法和现实世界一致，使得非技术人员也容易理解；

(2) 很容易映射到访问矩阵和基于组的自主访问控制，便于实现。

随着面向对象方法进一步推广，对于系统易用性需求更高，基于角色的访问控制的优势会越来越突出，将具有非常广阔的前景。

8.5　常用操作系统中的访问控制

前面讲述了最为常用的三种访问控制方法，本节将在这些理论的基础上介绍实际操作系统中访问控制的实现机制。这里以目前最为流行的 Windows NT 和 Linux 作为代表。

8.5.1　Windows NT 中的访问控制

Windows NT 4.0 分为服务器版(Server)和工作站版(Workstation)。这两个版本面向不同的需求，但是在核心特性、安全系统和网络设计上非常相似。它们最主要的区别是：服务器版的用户账户数据库可以用于整个域；而工作站版的用户账户只能用于本地。

1999 年 12 月 9 日，Windows NT 4.0 的服务器版和工作站版全部通过了 C2 安全等级的测试。C2 安全等级最重要的特点就是自主访问控制，它要求资源的所有者必须能够控制对资源的访问。

这里，我们以服务器版为对象进行介绍，工作站版的访问控制大体相同。

1．Windows NT 的安全模型与基本概念

Windows NT 采用的是微内核(Microkernel)结构和模块化的系统设计。有的模块运行在底层的内核模式上，有的模块则运行在受内核保护的用户模式上。Windows NT 的安全子系统根植于核心(kernel)层。

Windows NT 的安全模型由以下几个关键部分构成：

(1) 登录过程(Logon Process，LP)：接受本地用户或者远程用户的登录请求，处理用户信息，为用户做一些初始化工作。

(2) 本地安全授权机构(Local Security Authority，LSA)：根据安全账号管理器中的数据处理本地或者远程用户的登录信息，并控制审计和日志。这是整个安全子系统的核心。

(3) 安全账号管理器(Security Account Manager，SAM)：维护账号的安全性管理数据库(SAM 数据库，又称目录数据库)。

(4) 安全引用监视器(Security Reference Monitor，SRM)：检查存取合法性，防止非法存取和修改。

这几部分在访问控制的不同阶段发挥了各自的作用，在后面的访问控制过程中将详细讲述。

在 Windows NT 中，有关安全的概念主要有以下几个：

(1) 安全标识(Security Identifier，SID)：安全标识和账号惟一对应，在账号创建时创建，账号删除时删除，而且永不再用。安全标识与对应的用户和组的账号信息一起存储在 SAM 数据库里。

(2) 访问令牌(Access Token)：当用户登录时，本地安全授权机构为用户创建一个访问令牌，包括用户名、所在组、安全标识等信息。以后，用户的所有程序都将拥有访问令牌的拷贝。

(3) 主体：用户登录到系统之后，本地安全授权机构为用户构造一个访问令牌，这个令牌与该用户所有的操作相联系，用户进行的操作和访问令牌一起构成一个主体。

(4) 对象、资源、共享资源：对象的实质是封装了数据和处理过程的一系列信息集合体。资源是用于网络环境的对象。共享资源是在网络上共享的对象。

(5) 安全描述符(Security Descriptor)：Windows NT 系统会为共享资源创建安全描述符，包含了该对象的一组安全属性，分为 4 个部分：

a) 所有者安全标识(Owner Security ID)：拥有该对象的用户或者用户组的 SID；

b) 组安全标识(Group Security ID)：只有 POSIX 子系统使用，系统中其他部分将忽略；

c) 自主访问控制表(Discretionary Access Control List，DAC)：该对象的访问控制表，由对象的所有者控制；

d) 系统访问控制表(System Access Control List，ACL)：定义操作系统将产生何种类型的审计信息，由系统的安全管理员控制；

e) 访问控制项(Access Control Entries，ACEs)：安全描述符中的每一个访问控制表(ACL)都由访问控制项组成，用来描述用户或者组对对象的访问或审计权限。ACEs 有三种类型：Access Allowed、Access Denied 和 System Audit，前两种用于自主访问控制，后一种用于记录安全日志。

2. Windows NT 的访问控制过程

当一个账号被创建时，Windows NT 系统为它分配一个 SID，并与其他账号信息一起存入 SAM 数据库。

每次用户登录时，登录主机(通常为工作站)的系统首先把用户输入的用户名、口令和用户希望登录的服务器/域信息送给安全账号管理器，安全账号管理器将这些信息与 SAM 数据库中的信息进行比较，如果匹配，服务器发给工作站允许访问的信息，并返回用户的安全标识和用户所在组的安全标识。工作站系统为用户生成一个进程。服务器还要记录用

户账号的特权、主目录位置、工作站参数等信息。

然后，本地安全授权机构为用户创建访问令牌，包括用户名、所在组、安全标识等信息。此后用户每新建一个进程，都将访问令牌复制作为该进程的访问令牌。

当用户或者用户生成的进程要访问某个对象时，安全引用监视器将用户/进程的访问令牌中的 SID 与对象安全描述符中的自主访问控制表进行比较，从而决定用户是否有权访问对象。

在这个过程中应该注意 SID 对应账号的整个有效期，而访问令牌只对应某一次账号登录。

图 8.7 以 NTFS 文件对象的访问控制为例说明了上述过程。

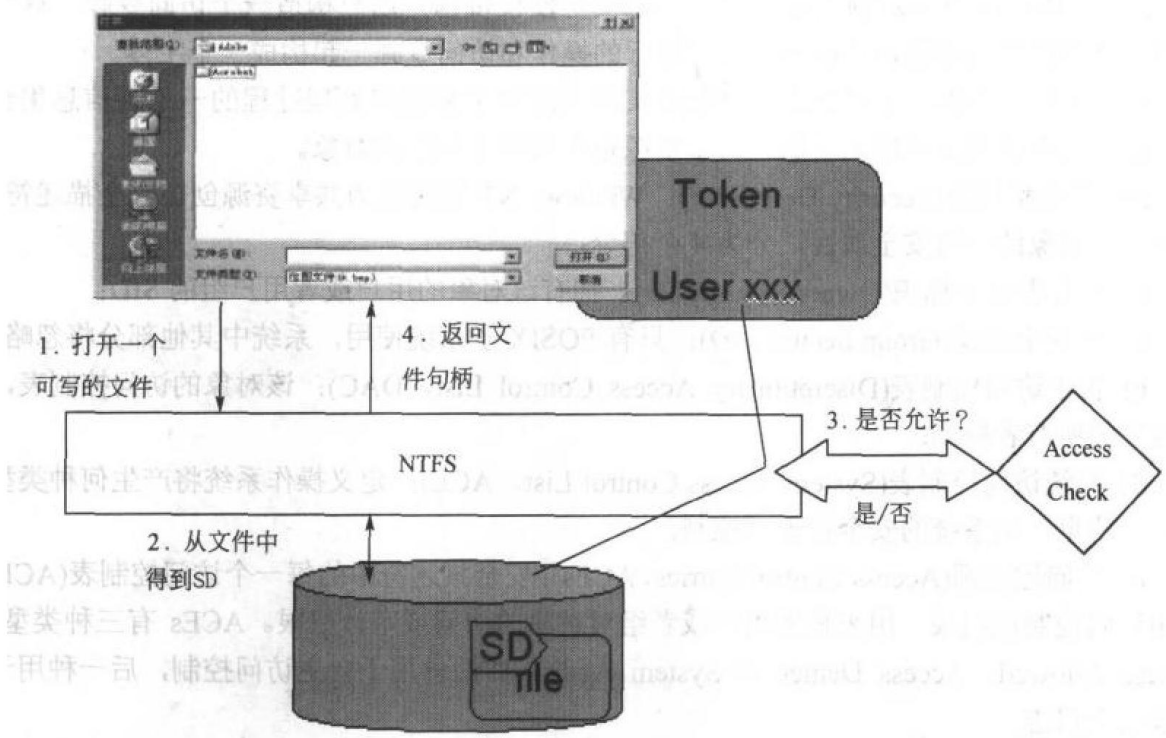


图 8.7　Windows NT 中的 NTFS 对象访问

用户发出一个打开文件的请求，NTFS 文件系统在 SAM 数据库中查找到该文件的安全描述符(SD)，将其与文件的访问令牌一起送到安全引用监视器进行访问检查，如果可以访问，则返回文件句柄。

Windows NT 系统中共享对象的访问权限是由对象所有者决定的，这体现了自主访问控制的思想。如果用户想共享某个对象，他首先要为对象选择惟一的名字，然后为其他的用户和组分配访问权限。然后，系统会以此为共享对象创建安全描述符。一个没有访问控

制表的对象可以被任何用户以任何方式访问。

共享资源的访问权限共有以下四种：

(1) 完全控制(Full Control)；

(2) 拒绝访问(No Access)；

(3) 读(Read)；

(4) 更改(Change)。

8.5.2 Linux 中的访问控制

Linux 系统将设备和目录都看作文件，有很多与普通文件相同的操作，这为简洁的访问控制奠定了基础。

在 linux 中，主体对文件具有三种权限：

(1) 读(r)：用户可以读文件；

(2) 写(w)：用户可以创建或修改文件；

(3) 执行(x)：用户可以运行可执行程序。

系统将所有用户分为四类：

(1) 根用户(root)：在系统里具有最大的权利，几乎可以控制整个系统，独有很多重要的能力；

(2) 所有者(owner)：文件的所有者。一般可以读写执行文件，比其他用户具有更高的权限。多数情况下是文件的创建者，但并不完全如此；

(3) 组(user group)：所有者所在组，一个用户可以同时在多个组中。组可以按不用的需求设定，便于设置访问权限；

(4) 其他用户(other users)：不属于前三类的用户。

Linux 系统中每个文件有 10 位权限位，含义如下：

(1) 第 1 位：d(目录)， b(块系统设备)， c(字符设备)， .(普通文件)；

(2) 第 2-4 位：所有者的读、写、执行权限；

(3) 第 5-7 位：所有者所在组的读、写、执行权限；

(4) 第 8-10 位：其他用户的读、写、执行权限。

例如，可以在 shell 中输入如下命令查看文件的权限：

```
$ ls –ld test
drwxr-x--x    2   lucy   work        1024   Jun 25 22:53 text
```

ls 命令打开 ld 开关用来查看目录文件 text 的资料，第一位 d 说明这是一个目录文件，后 9 位说明所有者具有所有权限、所有者所在组拥有读和执行权限、其他用户只有执行的权限。根用户自动拥有所有权限，所有没有必要说明。后面的“2”说明文件的所有结点

有两个目录连接。然后是所有者登录名 lucy 和共享该文件访问的用户组组名 work。之后是文件所占的字节数和最后修改的日期和时间。最后是文件名。

用户可以使用 chmod 命令修改自己所拥有文件的访问权限。根用户可以修改所有用户的文件权限。例如：

```
$chmod  777  test
```

就将目录文件 test 的权限改为对所有用户都可读可写可执行(drwxrwxrwx)。

在 Linux 系统中，正常情况下，一个程序运行的时候，它的进程属于当前用户。但是 SUID(Set User Identification)程序运行时，它的进程不属于启动用户，而是属于该程序的所有者用户。所以，运行者的权限在程序运行时会暂时提升，运行结束时又恢复为正常的授权级。这样 SUID 程序的漏洞就可能为入侵者提供机会，所以应该通过可靠的机制进行保护。

根用户具有很高的权限，因此也容易被利用而成为系统隐患。

Linux 访问控制机制的一个缺点是控制粒度比较大，这可以通过安装相关的模块来加以改善。

习　题　八

1. 访问控制机制有哪几类？有何区别？
2. 访问控制表和访问能力表有何区别？
3. 安全标记有什么作用？如何实现？
4. Bell-LaPadula 模型能否同时保证机密性和完整性？为什么？
5. 基于角色的访问控制是如何实现的？有什么优点？
6. Windows NT 采用什么访问控制模型？如何体现？

第九章 安全审计

9.1 安全审计的原理

9.1.1 安全审计的目标

审计技术出现在计算机技术之前，是产生、记录并检查按时间顺序排列的系统事件记录的过程。安全审计是计算机和网络安全的重要组成部分。安全审计提供的功能服务于直接和间接两方面的安全目标：直接的安全目标包括跟踪和监测系统中的异常事件，间接的安全目标是监视系统中其他安全机制的运行情况和可信度。

所有审计的前提是有一个支配审计过程的规则集。规则的确切形式和内容随审计过程具体内容的改变而改变。在商业与管理审计中，规则集包括对确保商业目标的实现有重要意义的管理控制、过程和惯例。这些商业目标包括资源的合理使用、利率最大化、费用最小化、符合相应的法律法规和适当的风险控制。在计算机安全审计的特殊情况下，规则集通常以安全策略的形式明确表述。并且，为了能完成合理的审计数据分析，策略表中还需要增加一些不容易明确表述的规则。

尽管从抽象意义上讲，传统的金融和管理审计与计算机安全审计的过程是完全相同的，但它们各自关注的问题有很大不同。计算机安全审计是通过一定的策略，利用记录和分析历史操作事件发现系统的漏洞并改进系统的性能和安全。计算机安全审计需要达到的目的包括：对潜在的攻击者起到震慑和警告的作用；对于已经发生的系统破坏行为提供有效的追究责任的证据；为系统管理员提供有价值的系统使用日志，帮助系统管理员及时发现系统入侵行为或潜在的系统漏洞。

James P. Anderson 在 1980 年写的一份报告中对计算机安全审计机制的目标作了如下阐述：

(1) 应为安全人员提供足够多的信息，使他们能够定位问题所在；但另一方面，提供的信息应不足以使他们自己也能够进行攻击。

(2) 应优化审计追踪的内容，以检测发现的问题，而且必须能从不同的系统资源收集信息。

(3) 应能够对一个给定的资源(其他用户也被视为资源)进行审计分析，分辨看似正常的

活动，以发现内部计算机系统的不正当使用；

(4) 设计审计机制时，应将系统攻击者的策略也考虑在内。

概括而言，审计系统的目标至少包括：确定和保持系统活动中每个人的责任；确认重建事件的发生；评估损失；临测系统问题区；提供有效的灾难恢复依据；提供阻止不正当使用系统行为的依据；提供案件侦破证据。

9.1.2　安全审计系统的组成

审计是通过对所关心的事件进行记录和分析来实现的，因此审计过程包括审计发生器、日志记录器、日志分析器和报告机制几部分，如图 9.1 所示。

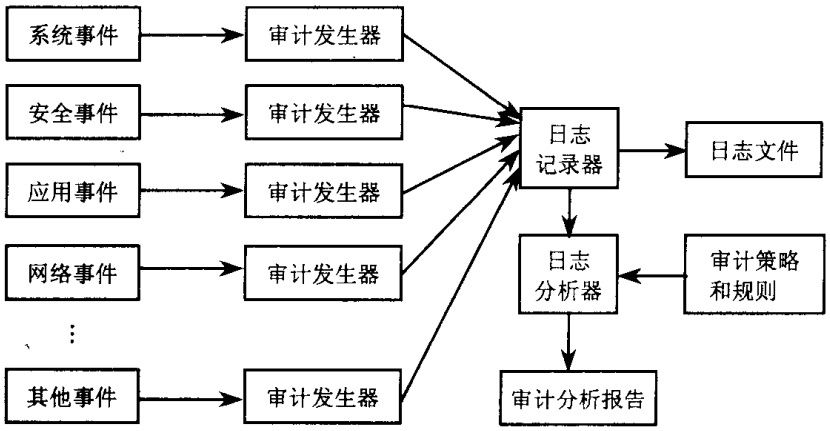


图 9.1　审计系统基本结构

审计发生器的作用是在信息系统中各种事件(如系统事件、安全事件、应用事件、网络事件等)发生时将这些事件的关键要素进行抽取并形成可记录的素材。日志记录器将审计发生器抽取的事件素材记录到指定的位置(如本机硬盘、磁带或专用记录主机)上，从而形成日志文件。日志分析器根据审计策略和规则对已形成的日志文件进行分析，得出某种事件发生的事实和规律，并形成日志审计分析报告。

9.1.3　日志的内容

在理想情况下，日志应该记录每一个可能的事件，以便分析发生的所有事件，并恢复任何时刻进行的历史情况。然而，这样做显然是不现实的，因为要记录每一个数据包、每一条命令和每一次存取操作，需要的存储量将远远大于业务系统，并且将严重影响系统的性能。因此，日志的内容应该是有选择的。一般情况下，日志记录的内容应该满足以下原则：

(1) 日志应该记录任何必要的事件，以检测已知的攻击模式；

(2) 日志应该记录任何必要的事件，以检测异常的攻击模式；

(3) 日志应该记录关于记录系统连续可靠工作的信息。

在这些原则的指导下，日志系统可根据安全要求的强度选择记录下列事件的部分或全部：

(1) 审计功能的启动和关闭；

(2) 使用身份鉴别机制；

(3) 将客体引入主体的地址空间；

(4) 删除客体；

(5) 管理员、安全员、审计员和一般操作人员的操作；

(6) 其他专门定义的可审计事件。

通常，对于一个事件，日志应包括事件发生的日期和时间、引发事件的用户(地址)、事件和源和目的的位置、事件类型、事件成败等。

9.1.4 安全审计的记录机制

不同的系统可采用不同的机制记录日志。日志的记录可能由操作系统完成，也可以由应用系统或其他专用记录系统完成。但是，大部分情况都可用系统调用 Syslog 来记录日志，也可以用 SNMP 记录。下面简单介绍 Syslog。

Syslog 由 Syslog 守护程序、Syslog 规则集及 Syslog 系统调用三部分组成，如图 9.2 所示。

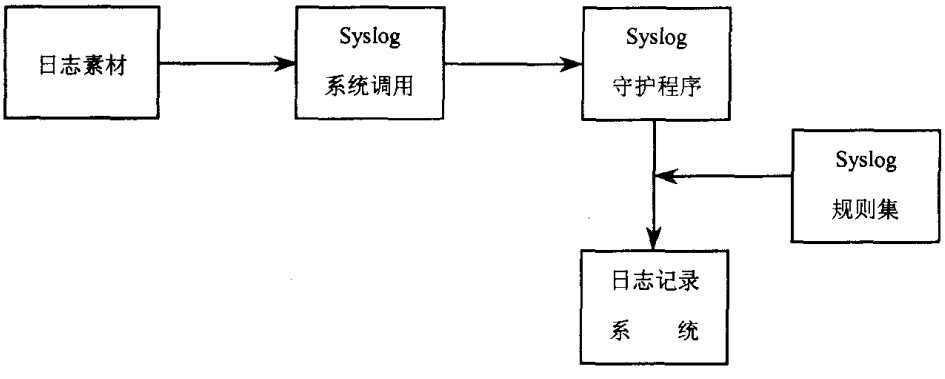


图 9.2 Syslog 记录机制

记录日志时，系统调用 Syslog 将日志素材发送给 Syslog 守护程序，Syslog 守护程序临听 Syslog 调用或 Syslog 端口(UDP514)的消息，然后根据 Syslog 规则集对收到的日志素材进行处理。如果日志是记录在其他计算机上，则 Syslog 守护程序将日志转发到相应的日志服务器上。Syslog 规则集(对于 Unix，常放在文件/etc/syslog.conf 中)是用来配置 Syslog

守护程序如何处理日志的规则。通常的规则可以是以下情况：

(1) 将日志放进文件中；

(2) 通过 UDP 将日志记录到另一台计算机上；

(3) 将日志写入系统控制台；

(4) 将日志发给所有注册的用户。

在记录日志时，为了便于管理，通常将一定时段(如 1 天或 1 周)的日志存为一个文件。这样，就需要在 0：00 时刻切换日志文件，如图 9.3 所示。图中假设日志文件在 2003 年 5 月 15 日零点切换，此前的文件名为 logfile.20030515，此后为 logfile.20030516，那么 0：00 日志临护程序会生成新文件，关闭旧文件，同时将新日志写入新文件。

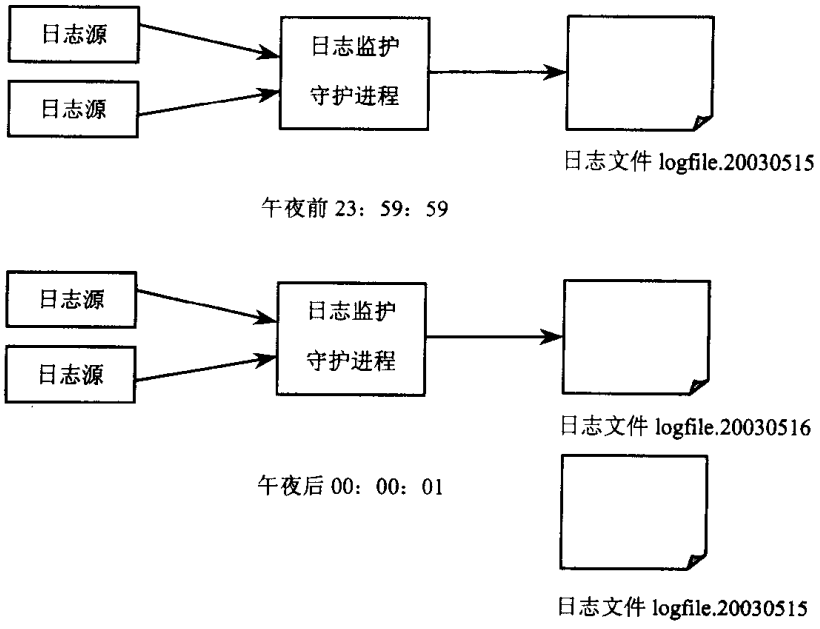


图 9.3　日志文件切换

在日志文件切换时，一种适合切换的算法是每次写文件之前打开文件，写完后关闭，程序如下：

```
While(okToRun){
    Message=getlogmessage( );
    FILE=openForAppending(logfile);
    Append(FILE,message);
    Close(FILE)
    }
```

值得注意的是，由于文件的打开和关闭时间较写的时间慢得多，因此可能会导致有些事件丢失。为此，可以将一个文件永久打开，供日志读写。程序如下：

```
FILE=openForAppending(logfile);
While(okToRun){
        Message=getlogmessage( );
        Append(FILE,message);
    }
Close(FILE);
```

但这样又会影响日志文件的切换。因此比较好的做法是将 Syslog 监护程序打开的文件作为原始日志文件，另外增加一个日志整理进程，专门负责日志的整理和归档。

9.1.5 安全审计分析

通过对日志进行分析，发现所需事件信息和规律是安全审计的根本目的。因此，审计分析十分重要。日志分析就是在日志中寻找模式，主要内容如下：

(1) 潜在侵害分析：日志分析应能用一些规则去监控审计事件，并根据规则发现潜在的入侵。这种规则可以是由已定义的可审计事件的子集所指示的潜在安全攻击的积累或组合，或者其他规则。

(2) 基于异常检测的轮廓：日志分析应确定用户正常行为的轮廓，当日志中的事件违反正常访问行为的轮廓，或超出正常轮廓一定的门限时，能指出将要发生的威胁。

(3) 简单攻击探测：日志分析应对重大威胁事件的特征有明确的描述，当这些攻击现象出现时，能及时指出。

(4) 复杂攻击探测：要求高的日志分析系统还应能检测到多步入侵序列，当攻击序列出现时，能预测其发生的步骤。

9.1.6 审计事件查阅

由于审计系统是追踪、恢复的直接依据，甚至是司法依据，因此其自身的安全性十分重要。审计系统的安全主要是查阅和存储的安全。

审计事件的查阅应该受到严格的限制，不能篡改日志。通常通过以下不同的层次保证查阅的安全：

(1) 审计查阅：审计系统以可理解的方式为授权用户提供查阅日志和分析结果的功能。

(2) 有限审计查阅：审计系统只能提供对内容的读权限，因此应拒绝具有读以外权限的用户访问审计系统。

(3) 可选审计查阅：在有限审计查阅的基础上限制查阅的范围。

9.1.7　审计事件存储

审计事件的存储也有安全要求，具体有如下几种情况：

(1) 受保护的审计踪迹存储：即要求存储系统对日志事件具有保护功能，防止未授权的修改和删除，并具有检测修改/删除的能力。

(2) 审计数据的可用性保证：在审计存储系统遭受意外时，能防止或检测审计记录的修改，在存储介质存满或存储失败时，能确保记录不被破坏。

(3) 防止审计数据丢失：在审计踪迹超过预定的门限或记满时，应采取相应的措施防止数据丢失。这种措施可以是忽略可审计事件、只允许记录有特殊权限的事件、覆盖以前记录、停止工作等等。

9.2　安全审计应用实例

流行的商业操作系统都提供审计的功能。下面以 Windows NT 和 Unix/Linux 操作系统为例，说明安全审计的应用实例。

9.2.1　Windows NT 中的安全审计

1. NT 审计子系统结构

几乎 Windows NT 系统中的每一项事务都可以在一定程度上被审计，在 Windows NT 中可以在 Explorer 和 User manager 两个地方打开审计。在 Explorer 中，选择 Securtiy，再选择 Auditing 以激活 Directory Auditing 对话框，系统管理员可以在这个窗口选择跟踪有效和无效的文件访问。在 User manager 中，系统管理员可以根据各种用户事件的成功和失败选择审计策略，如登录和退出、文件访问、权限非法和关闭系统等。Windows NT 使用一种特殊的格式存放它的日志文件，这种格式的文件可以被事件查看器 Event viewer 读取。事件查看器可以在 Administrative tool 程序组中找到。系统管理员可以使用事件查看器的 Filter 选项根据一定条件选择要查看的日志条目。查看条件包括类别、用户和消息类型。

Windows NT 的日志文件很多，但主要是系统日志、安全日志和应用日志三个。这三个审计日志是审计一个 Windows NT 系统的核心。默认安装时安全日志不打开。Windows NT 中所有可被审计的事件都存入了其中的一个日志。

(1) Application Log：包括用 NT Security authority 注册的应用程序产生的信息。

(2) Security Log：包括有关通过 NT 可识别安全提供者和客户的系统访问信息。

(3) System Log：包含所有系统相关事件的信息。

不幸的是，Windows NT 中的审计缺乏足够的深度和广度。系统或安全管理员用于浏览审计日志的工具——事件浏览器(Event Viewer)只有非常有限的灵活性，并且对大型日志的浏览速度很慢。这些日志并不能以域作为中心存储。每个服务器和工作站都有自己的日志集。这些日志分散在 Windows NT 网络的成千上万个服务器上，没有复杂的自动操作工具和数据转储工具，管理和利用这些日志是非常困难的。

2. 审计日志和记录格式

Windows NT 的审计日志由一系列的事件记录组成。每一个事件记录分为三个功能部分:头、事件描述和可选的附加数据项。表 9.1 显示了一个事件记录的结构。安全日志的入口通常由头和事件描述组成。

表 9.1 Windows NT 事件记录格式

记录头	数据	时间	用户名	计算机名
	事件 ID	源	类型	种类
事件描述	可变内容，依赖于事件。可以使问题的文本解释和纠正措施的建议			
附加数据	附加域。如果采用的话，包含可以字节或字显示的二进制数据及事件记录的源应用产生的信息			

事件记录头由下列域组成：

(1) 日期：事件的日期标识。

(2) 时间：事件的时间标识。

(3) 用户名：标识事件是由谁触发的。这个标识可以是初始用户 ID、某个客户 ID 或两者同时具有。具体是哪一个用户 ID，由 Windows NT 的扮演功能是否触发来决定。当操作系统允许一个进程继承另一个进程的安全属性时，扮演被触发。当扮演发生时，安全日志机制将同时反映用户 ID 和扮演 ID。

(4) 计算机名：事件所在的计算机名。当用户在整个企业范围内集中安全管理时，该信息大大简化了审计信息的回顾。

(5) 事件 ID：事件类型的数字标识。在事件记录描述中，这个域通常被映射成一个文本标识(事件名)。

(6) 源：用来响应产生事件记录的软件。源可以是一个应用程序、一个系统服务或一个设备驱动程序。

(7) 类型：事件严重性指示器。在系统和应用日志中，类型可以是错误、警告或信息，按重要性降序排列。在安全日志中，类型可能是成功审计或失败审计。

(8) 种类：触发事件类型，主要用在安全日志中指示该类事件的成功或失败审计已经

被许可。

3. NT 事件日志管理特征

Windows NT 提供了大量特征给系统管理员去管理操作系统事件日志机制。例如：管理员能限制日志的大小并规定当文件达到容量上限时，如何去处理这些文件。选项包括：用新记录去冲掉最老的记录，停止系统直到事件日志被手工清除。

当系统开始运行时，系统和应用事件日志也自动开始。当日志文件满并且系统配置规定它们必须被手工清除时，日志停止。另一方面，安全事件日志必须由具有管理者权限的人启动。利用 NT 的用户管理器，可以设置安全审计规则。要启用安全审计的功能，只需在规则菜单下选择审计，然后通过查看 NT 记录的安全事件日志中的安全性事件，即可以跟踪所选用户的操作。

4. NT 安全日志的审计策略

NT 安全日志由审计策略支配。审计策略可以通过配置审计策略对话框中的选项来建立。审计策略规定日志的事件类型并可以根据动作、用户和目标进一步具体化。安全事件记录包括动作的时间和日期、已执行的动作和执行响应的动作。成功和失败的动作都能在安全日志中产生条目。日志条目也记录企图执行被策略禁止的动作的活动。

NT 的审计规则如下(既可以审计成功的操作，又可以审计失败的操作)：

(1) 登录及注销：登录及注销或连接到网络。

(2) 用户及组管理：创建、更改或删除用户账号或组，重命名、禁止或启用用户号，设置和更改密码。

(3) 文件及对象访问：访问设置用于文件或目录审计的目录或文件的用户，向设置用于打印机审计的打印机发送打印作业的用户。

(4) 安全性规则更改：对用户权利、审计或委托关系规则的改动。

(5) 重新启动、关机及系统级事件：用户重新启动或关闭计算机，或者发生了一个影响系统安全性或安全日志的事件。

(6) 进程追踪：这些事件提供了关于事件的详细跟踪信息，如程序活动、某些形式句柄的复制、间接对象的访问和退出进程。对于“文件及对象访问”中的文件和目录的审计还需要在资源管理器中对要审计的目录或文件进行具体设置。

(7) 文件和目录审计：允许跟踪目录和文件的用法。对于一个具体的文件或目录，可以指定要审计的组、用户或操作。既可以审计成功的操作，又可以审计失败的操作。审计目录可以选择读、写、执行、删除、更改权限或者获得所有权等事件。审计文件也可以选择读、写、可执行、删除、更改权限、获得所有权等事件。

5. 管理和维护 NT 审计

通常情况下，Windows NT 不是将所有的事件都记录日志，而需要手动启动审计的功能。这时需要首先从开始菜单中选择程序，然后再选择管理工具。 从管理工具子菜单选择用户管理器，显示出用户管理器窗口。然后从用户管理器的菜单中单击 policies(策略)，再单击 audit(审计)，审计策略窗口就出现了。接着选择单选框“audit thest events”(审计这些事件)。最后选择需要启动事件的按 OK，然后关闭用户管理器。值得注意的是在启动 Windows NT 的审计功能时，需要仔细选择审计的内容。审计日志将产生大量的数据，因此较为合理的方法是首先设置进行简单的审计，然后在监视系统性能的情况下逐步增加复杂的审计要求。当需要审查审计日志以跟踪网络或机器上的异常事件时，采用一些第三方提供的工具是一个较有效率的选择。

最后介绍一下Windows NT的三个日志文件的物理位置。

系统日志：%systemroot%\system32\config\sysevent.evt

安全日志：%systemroot%\system32\config\secevent.evt

应用程序日志：%systemroot%\system32\config\appevent.evt

9.2.2 Unix/Linux 中的安全审计

和 Windows NT 审计日志物理位置的统一相比，不同版本的 Unix 日志文件的目录是不同的，最常用的目录是：

/usr/adm　　早期版本的 Unix

/var/adm　　较新版本的 Unix

/var/log　　用于 Solaris,Linux,BSD 等

/etc　　Unix system V 早期版本

在这些目录或其子目录下，可以找到以下日志文件(也许是其中的一部分)：

lastlog：记录用户最后一次成功登录时间；

loginlog：不良的登陆尝试记录；

messages：记录输出到系统主控台以及由 syslog 系统服务程序产生的消息；

utmp：记录当前登录的每个用户；

utmpx：扩展的 utmp；

wtmp：记录每一次用户登录和注销的历史信息 wtmpx 扩展的 wtmp；

vold.log：记录使用外部介质出现的错误；

xferkig：记录 Ftp 的存取情况，sulog 记录 su 命令的使用情况；

acct：记录每个用户使用过的命令；

aculog：拨出自动呼叫记录。

所有这些日志文件可以大致分为三个日志子系统。

1．连接时间日志

连接时间日志由多个程序执行，把记录写入到/var/log/wtmp 和/var/run/utmp 中并通过 login 等程序更新 wtmp 和 utmp 文件，使系统管理员能够跟踪谁在何时登录到系统。下面对该子系统中的日志文件逐一进行详细的说明。

lastlog 文件：Unix 在 lastlog 日志文件中记录每一个用户注册进入系统的最后时间，在每一次进入系统时，系统会显示出这个时间。例如：

```
login: blackeyes
password: ******
Last login :Tue Jul 27 09:55:50 on tty01
```

Lastlog 告诉用户，要核对一下最后注册进入系统的时间是否正确，若系统显示的时间与上次进入系统的时间不符，说明发生了非授权用户注册。如果这种情况发生了，用户应该马上修改账户口令，并通知管理员。在每次注册时，lastlog 新的内容冲掉老的内容。

loginlog 文件：Unix system V 版本中，可以把不成功的登录行为记录在/var/adm/loginlog 中。要登记不成功的注册行为，可以用下列命令建立/var/adm/loginlog 文件：

```
#touch /var/adm/loginlog
#chmod 600 /var/adm/loginlog
#chown root /var/adm/loginlog
```

如果某个入侵者知道一个系统的用户名，同时又想猜出密码，/var/adm/loginlog 就会记录他的失败的登录尝试。

管理员可以查看/var/adm/loginlog 的内容，命令如下：

```
#cat /var/adm/loginlog
hacker: from 202.88.88.xx: Tue Jul 27 02:40:50 1999
hacker: from 202.88.88.xx: Tue Jul 27 02:41:50 1999
hacker: from 202.88.88.xx: Tue Jul 27 02:42:50 1999
```

utmp、wtmp 和 lastlog 日志文件是多数 Unix 日志子系统的关键——记录用户登录和退出的信息。有关当前登录用户的信息记录在文件 utmp 中，登录和退出记录在文件 wtmp 中，最后一次登录文件可以用 lastlog 命令查看。数据交换、关机和重启也记录在 wtmp 文件中。所有的记录都包含时间戳。这些文件(lastlog 通常不大)在具有大量用户的系统中增长十分迅速。例如 wtmp 文件可以无限增长，除非定期截取。许多系统以一天或一周为单位把 wtmp 配置成循环使用。它通常由 cron 运行的脚本来修改。这些脚本重新命名并循环使用 wtmp

文件。通常，wtmp 在第一天结束后命名为 wtmp.1，第二天后 wtmp.1 变为 wtmp.2，等等，直到 wtmp.7。

wtmp 和 utmp 文件都是二进制文件，它们不能被诸如 tail 命令剪贴或合并(使用 cat 命令)。用户需要通过 who、w、users、last 和 ac 来使用这两个文件包含的信息。更具体的命令使用方法，读者可以去阅读有关的书籍。

2. 进程统计日志

进程统计日志由系统内核执行。当一个进程终止时，系统往进程统计文件(pacct 或 acct)中写一个记录。进程统计的目的是为系统中的基本服务提供命令使用统计。

Unix 可以跟踪每个用户运行的每条命令。如果想知道昨晚弄乱了哪些重要的文件，进程统计子系统可以告诉你。它还对跟踪一个侵入者有帮助。与连接时间日志不同，进程统计子系统缺省为不激活，它必须通过启动来激活。在 Unix/Linux 系统中启动进程统计使用 accton 命令，必须用 root 身份来运行。accton 命令的形式为 accton file，file 必须先存在。然后运行 accton：accton/var/log/pacct。一旦 accton 被激活，就可以使用 lastcomm 命令监测系统中任何时候执行的命令。若要关闭统计，可以使用不带任何参数的 accton 命令。

进程统计的一个问题是 pacct 文件可能增长得十分迅速。这时需要交互式或经过 cron 机制运行 sa 命令来保持日志数据在系统控制的范围内。sa 命令报告、清理并维护进程统计文件，它能把/var/log/pacct 中的信息压缩到摘要文件/var/log/savacct 和/var/log/usracct 中。这些摘要包含按命令名和用户名分类的系统统计数据。sa 在缺省情况下先读它们，然后读 pacct 文件，使报告能包含所有的可用信息。

3. 错误日志

错误日志由 syslog 执行。各种系统守护进程、用户程序和内核通过 syslog 向文件/var/log/messages(也有可能是/var/adm/messages)报告值得注意的事件。另外，有许多 Unix/Linux 程序也会创建日志。像 Http 和 Ftp 这样提供网络服务的服务器也都保持详细的日志。

syslog 采用可配置的、统一的系统登记程序，随时从系统各处接受 log 请求，然后根据/etc/syslog.conf 中的预先设定把 log 消息写入相应文件中并邮寄给特定用户或者直接以消息的方式发往控制台。任何程序都可以通过 syslog 记录事件。syslog 可以记录系统事件，也能记录本地事件或通过网络记录另一个主机上的事件。syslog 设备依据两个重要的文件：/etc/syslogd(守护进程)和/etc/syslog.conf 配置文件。习惯上，多数 syslog 信息被写到/var/adm 或/var/log 目录下的信息文件中(messages.*)。一个典型的 syslog 记录包括生成程序的名字和一个文本信息。它还包括一个设备和一个优先级范围(但不在日志中出现)。

每个 syslog 消息被赋予下面的主要设备之一。

LOG_AUTH——认证系统：login、su、getty 等；

LOG_AUTHPRIV——同 LOG_AUTH，但只登录到所选择的单个用户可读的文件中；

LOG_CRON——cron 守护进程；

LOG_DAEMON——其他系统守护进程，如 routed；

LOG_FTP——文件传输协议：ftpd、tftpd；

LOG_KERN——内核产生的消息；

LOG_LPR——系统打印机缓冲池：lpr、lpd；

LOG_MAIL——电子邮件系统；

LOG_NEWS——网络新闻系统；

LOG_SYSLOG——由 syslog 产生的内部消息；

LOG_USER——随机用户进程产生的消息；

LOG_UUCP——UUCP 子系统；

LOG_LOCAL0~LOG_LOCAL7——为本地使用保留。

syslog 为每个事件赋予几个不同的优先级。

LOG_EMERG——紧急情况；

LOG_ALERT——应该被立即改正的问题，如系统数据库破坏；

LOG_CRIT——重要情况，如硬盘错误；

LOG_ERR——错误；

LOG_WARNING——警告信息；

LOG_NOTICE——不是错误情况，但是可能需要处理；

LOG_INFO——情报信息；

LOG_DEBUG——包含情报的信息，通常只在调试一个程序时使用。

syslog.conf 文件指明 syslog 程序记录日志的行为，syslog 程序在启动时查询配置文件。该文件由不同程序或消息分类的单个条目组成，每个占一行。对每类消息提供一个选择域和一个动作域。这些域由 tab 隔开，选择域指明消息的类型和优先级，动作域指明 syslog 接收到一个与选择标准相匹配的消息时所执行的动作。每个选项是由设备和优先级组成。当指明一个优先级时，syslog 将记录一个拥有相同或更高优先级的消息。所以如果指明 crit，那所有标为 crit、alert 和 emerg 的消息都将被记录。每行的行动域指明当选择域选择了一个给定消息后应该把它发送到哪里。

通过审计/var/log/messages 中的一些记录，可以找出入侵者的痕迹，例如：

- Sep 8 09:08:03 xxx login: REPEATED LOGIN FAILURES ON /dev/pts/3 FROM xxxxx

说明入侵者采用暴力破解口令命令失败；

• Sep 8 09:08:03 xxx su: 'su root' failed for xxxxx on /dev/pts/2 说明入侵者想利用 su 命令成为 root 失败。

习 题 九

1. 审计系统的目标是什么？如何实现？
2. 审计的主要内容包括哪些？
3. Windows 的审计系统是如何实现的？采用什么策略？
4. Unix 的日志分哪几类？有何作用？

第十章　安全脆弱性分析

信息系统不安全的主要原因是系统自身存在安全脆弱点，因此信息系统安全脆弱性分析是评估信息系统安全强度和设计安全体系的基础。本章首先分析安全威胁的基本原理和类型，然后介绍一种目前用于安全脆弱性分析的主要技术，安全扫描技术。

10.1　安全威胁分析

10.1.1　入侵行为分析

黑客的入侵和攻击行为很难界定，也很难被发现。怎样才算是受到了黑客的入侵和攻击呢？狭义的定义认为：攻击仅仅发生在入侵行为完成且入侵者已经在其目标网络中。广义的定义是：使网络受到入侵和破坏的所有行为都应被称为“攻击”。本书采纳广义的定义，即认为当入侵者试图在目标机上“工作”的那个时刻起，攻击就已经发生了。

下面将从几个方面分析系统中发生的各种入侵行为，包括入侵者的目的、实施入侵的人员、入侵过程中的各个阶段和每个阶段不同的特点。

入侵者的目的各不相同。善意的入侵者只是想探索互联网，看看未知的部分是什么。而恶意的入侵者可能读取特权数据、进行非授权修改或者破坏系统。不幸的是,一般很难分清入侵者的行为是善意的还是恶意的，而且即使某入侵者的目的是善意的，他也可能在不经意间给系统造成极大的损失，或者为别的恶意攻击行为提供方便。然而分析黑客的目的还是有助于了解入侵者的行为，特别是有助于了解系统的哪些部分最容易受到攻击。

大体说来入侵者在入侵一个系统者时会想达到以下一种或几种目的：

(1) 执行进程：攻击者在成功入侵目标主机后，或许仅仅是为了运行一些程序，而且这些程序除了消耗了系统资源外，对目标机器本身是无害的。举例来说，一些扫描程序只能在 Unix 系统中运行，这时候，攻击者就可能入侵一台 Unix 系统的工作站。尽管这种情况对目标主机本身并无多大坏处，但是潜在的危害是不能被忽视的，不但消耗资源，而且有可能将责任转移到目标主机，这时后果是难以估计的。

(2) 获取文件和数据：入侵者的目标是系统中的重要数据。入侵者可以通过登录目标主机，使用网络监听程序进行攻击。监听到的信息可能含有重要的信息，如关于用户口令

的信息等。

(3) 获取超级用户权限：在多用户的系统中，超级用户可以进行任何操作，因此获取超级拥护权限是每一个入侵者都梦寐以求的。

(4) 进行非授权操作：很多用户都会去尝试尽量获得超出许可的一些权限，如寻找管理员设置中的一些漏洞，或者寻找一些工具来突破系统的防线。

(5) 使系统拒绝服务：这种攻击将使目标系统中断或者完全拒绝对合法用户、网络、系统或其他资源的服务。任何这种攻击的意图都是邪恶的，而这种攻击往往不需要复杂技巧，只需借助很容易找到的工具即可实现。

(6) 篡改信息：包括对重要文件的修改、更换、删除等。不真实或者错误的信息往往会给用户造成巨大的损失。

(7) 披露信息：入侵者将目标站点的重要信息与数据发往公开的站点，造成信息的扩散，由于公开站点往往会有许多人访问，其他用户就可能得到这些信息，并将其再次扩散出去。

入侵者大致上可以分为三种类型：伪装者、违法者及秘密用户。未经授权使用计算机者或绕开系统访问控制机制获得合法用户账户权限者，都称为伪装者。未经授权访问数据、程序或资源的合法用户，或者具有访问授权但错误使用其权利的人，称为违法者。拥有账户管理权限者，利用这种控制来逃避审计和访问数据，或者禁止收集审计数据，称其为秘密用户。伪装者很可能是外部人员；违法者一般是内部人员；而秘密用户可能是外部人员，也可能是内部人员。

入侵和攻击需要一个时间过程，可以把这个过程大致分成窥探设施、攻击系统、掩盖踪迹三个不同的步骤。

窥探设施，顾名思义也就是对目标系统的环境的了解。没有这种了解，入侵者想要突破防线是很难的。窥探的目的是想了解目标采用是什么操作系统——Unix、Windows NT 还是 NetWare？哪些信息是公开的，有何价值？运行的 WEB 服务器是什么类型——apache 还是 IIS？其版本如何？这些问题都要经过对目标环境的窥探后才能回答，而这些问题的答案对入侵者以后将要发动的攻击起着至关重要的作用。

窥探设施需要得到的信息至少有以下几种：

(1) 通过收集尽可能多的关于一个系统的安全态势的各个方面的信息，进而构造出关于该目标机构的因特网、远程访问及内联网/外联网之间的结构。

(2) 通过使用 ping 扫描、端口扫描以及操作系统检测等工具和技巧，进一步掌握关于目标环境所依赖的平台、服务等相关信息。

(3) 从系统中抽取有效账号或者导出资源名，这时会涉及到往目标系统的主动连接和定向查询。这类信息看起来并无危害。然而一旦查出一个有效用户名或共享资源，攻击者

猜出对应的密码或者标示与资源共享协议关联的某些脆弱点就很容易了。

在窥探设施的工作完成后，入侵者将根据得到的信息对系统发起攻击。攻击系统可以分为针对操作系统的攻击、针对应用软件的攻击及针对网络的攻击三个层次。

入侵者一旦通过攻击成功获得某个系统的特权账号，会千方百计地避免自己被检测出来。当从目标系统上获得所有感兴趣的信息后，往往会安置后门并藏匿一个工具箱，以保证将来可以再次轻易的获得访问权，而且便于对其他的系统发动攻击。因此系统管理员在发现自己的系统被入侵之后，所要做的绝不是仅仅删除一两个账号。必须仔细审查系统，以确保黑客所安装的后门被完全删除，并对已知的系统漏洞打上补丁，以防被黑客再次入侵。

10.1.2　安全威胁分析

本节将从系统的角度出发，详细分析计算机系统所面临的各种安全威胁，包括安全威胁的来源、分类和特点等。

1. 威胁来源

计算机系统面临的安全威胁有来自计算机系统外部的，也有来自计算机系统内部的。来自计算机系统外部的威胁主要有：

(1) 自然灾害、意外事故；

(2) 计算机病毒；

(3) 人为行为，比如使用不当、安全意识差等；

(4)“黑客”行为：由于黑客的入侵或侵扰，比如非法访问、拒绝服务、非法连接等；

(5) 内部泄密；

(6) 外部泄密；

(7) 信息丢失；

(8) 电子谍报，比如信息流量分析、信息窃取等；

(9) 信息战。

计算机系统内部存在的安全威胁主要有：

(1) 操作系统本身所存在的一些缺陷；

(2) 数据库管理系统安全的脆弱性；

(3) 管理员缺少安全方面的知识，缺少安全管理的技术规范，缺少定期的安全测试与检查；

(4) 网络协议中的缺陷，例如 TCP/IP 协议的安全问题；

(5) 应用系统缺陷，等等。

本书关注的重点是来自计算机系统外部的黑客的攻击和入侵。

2. 攻击分类

攻击的分类方法是多种多样的。这里根据入侵者使用的手段和方式，将攻击分为口令攻击、拒绝服务攻击、利用型攻击、信息收集攻击以及假消息攻击等5类。

(1) 口令攻击

抵抗入侵者的第一道防线是口令系统。几乎所有的多用户系统都要求用户不但提供一个名字或标识符(ID)，而且要提供一个口令。口令用来鉴别一个注册系统的个人 ID。在实际系统中，入侵者总是试图通过猜测或获取口令文件等方式来获得系统认证的口令，从而进入系统。入侵者登录后，便可以查找系统的其他安全漏洞，来得到进一步的特权。为了避免入侵者轻易地猜测出口令，用户应该避免使用不安全的口令。不安全的口令类型有:

a) 用户名或用户名的变形;

b) 电话号码、执照号码等;

c) 一些常见的单词;

d) 生日;

e) 长度小于5的口令;

f) 空口令或默认口令;

g) 上述词后加上数字。

有时候即使有好的口令也是不够的，尤其是当口令需要穿过不安全的网络时将面临极大的危险。很多的网络协议中是以明文的形式传输数据，如果攻击者监听网络中传送的数据包，就可以得到口令。在这种情况下，一次性口令是有效的解决方法。

入侵者获取了目标系统的初始访问权后，往往试图获取系统的口令文件，以便进一步控制目标系统甚至整个目标网络。在 Windows(Windows NT 和 Windows 2000/XP)平台上，口令文件是%systemroot%\system32\config 中一个名为“SAM”的文件；在 Unix/Linux 平台上，口令文件可能是/etc/passwd 或者/etc/shadow。一旦攻击者获得了以上信息，就会尝试各种口令攻击工具来进行破解。常用的两个口令破解工具如“John the Ripper”和“Lophtcrack”，分别用于破解 Unix 系统和 Windows NT/2000/XP 下的口令文件。

(2) 拒绝服务攻击

拒绝服务站 DOS(Denial of Services)使得目标系统无法提供正常的服务，从而可能给目标系统带来重大的损失。值得关注的是，现实情况中破坏一个网络或系统的运行往往比取得访问权容易的多。像 TCP/IP 之类的网际互连协议是按照在彼此开放和信任的群体中使用来设计的，在现实环境中表现出这种理念的内在缺陷。此外，许多操作系统和网络设备的网络协议栈也存在缺陷，从而削弱了抵抗 DOS 攻击的能力。

下面介绍 4 种常见的 DOS 攻击类型及原理。

a) 带宽耗用(bandwidth-consumption)：其本质是攻击者消耗掉通达某个网络的所有可用带宽。一种情况是攻击者有比受害者网络的带宽更大的可用带宽，从而可以造成受害者的网络拥塞；另一种情况是攻击者通过征用多个站点集中拥塞受害者的网络连接来放大 DOS 攻击效果，后一种情况被称为分布式拒绝服务攻击 DDOS(Distributed Denial of Services)，DDOS 攻击的第一步是瞄准并获得尽可能多的系统管理员访问权。获得了对系统的访问权后，攻击者将 DDOS 软件上传并运行，大多数的 DDOS 服务器程序运行的方式是监听发起攻击的指令，这样攻击者只需将需要的软件上传到被控制的系统，然后等待适当的时机发起攻击命令。DDOS 攻击工具的数量增长极快，比较典型的工具有 TFN、Trinoo、Stachedraht、TFN2K 以及 WinTrinoo。

图 10.1 是多个系统汇总攻击的图示。

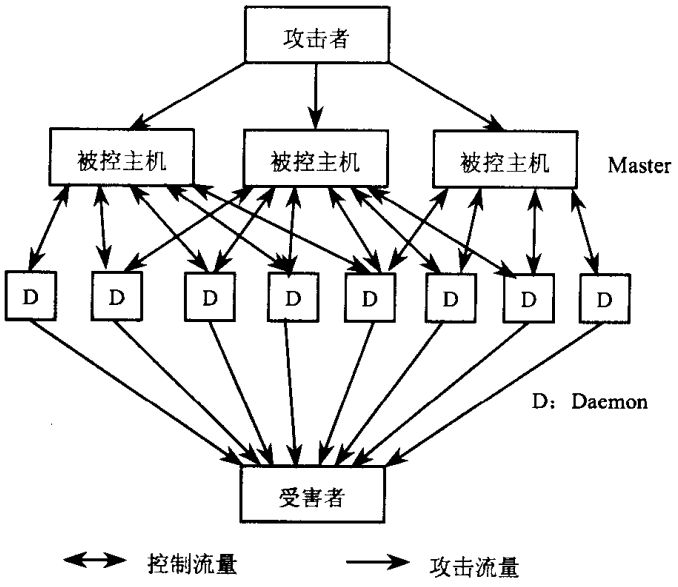


图 10.1　DDOS 攻击汇总示意图

b) 资源衰竭(resource-starvation)：一般地说，它涉及诸如 CPU 利用率、内存、文件系统限额和系统进程总数之类系统资源的消耗。攻击者往往拥有一定数量系统资源的合法访问权限。然而滥用这种访问权将消耗额外的资源，合法用户被剥夺了原来享有的资源份额。

c) 编程缺陷(programming flaw)：是应用程序、操作系统或嵌入式 CPU 在处理异常条件上的失败。这些异常条件往往是用户往脆弱的元素发送不期望的数据造成的。攻击者往往向目标系统发送非法的、与 RFC 不相容的数据包来确认其网络协议栈是否会处理这种异常，或则是否会导致系统崩溃。teardrop、newtear、ping of death、boink 等工具就利用系

统编程缺陷进行攻击。

d) 路由和 DNS 攻击：基于路由的 DOS 攻击涉及攻击者操纵路由表项以拒绝对合法系统或网络提供服务。很多较早版本的路由协议没有或者只有很弱的认证机制，这给攻击者修改合法路径创造了条件，攻击者往往通过假冒源 IP 地址就能达到目的。这种攻击的后果是去往受害者网络的数据包或者经由攻击者的网络路由，或者被路由到不存在的网络地址。基于域名系统(DNS)的攻击往往将受害者域名服务器高速缓存中的信息修改成虚假的地址信息。这样当合法用户请求某台 DNS 服务器执行查找请求时，服务请求就被重定向到了攻击者指定的站点。

(3) 利用型攻击

利用型攻击是一种试图直接对主机进行控制的攻击。它有两种主要的表现形式：特洛伊木马和缓冲区溢出攻击。

a) 特洛伊木马：表面看是有用的软件工具，而实际上却在启动后暗中安装破坏性的软件。许多远程控制后门往往伪装成无害的工具或文件，使得用户在不知不觉间进行了安装。恶意的木马程序有 NetBus、BackOrifice 和 B02K 等。为了防止受到木马程序的攻击，应该遵守一些准则，如避免下载可疑的程序并执行，用网络扫描软件定期监测内部主机的监听 TCP 服务。

b) 缓冲区溢出攻击(Buffer Overflow)：通过往程序的缓冲区写超出其长度的内容，造成缓冲区的溢出，从而破坏程序的堆栈，使程序转而执行一段恶意代码，以达到攻击的目的。据统计，通过缓冲区溢出进行的攻击占所有系统攻击总数的 80%以上。在 C 语言中，指针和数组越界没有得到保护是缓冲区溢出的根源，如在 C 语言的标准库中有许多能提供溢出的函数，如 strcat( )、strcpy( )、sprintf( )和 scanf( )等。在 UNIX 平台上，通过缓冲区溢出攻击，攻击者可以得到一个交互式的 Shell。在 Windows 平台上，攻击者可以执行任意的恶意代码。尽管缓冲区溢出漏洞的发掘需要很高的技术和知识背景，然而一旦有人写出了溢出代码，使用起来就非常简单。在缓冲区溢出攻击面前，防火墙往往显得很无力。1988 年的 Morris 病毒，在很短的时间内就感染了 6000 多台 Unix 平台的机器。该蠕虫病毒所使用的两个漏洞之一就是利用 Unix 服务 finger 中的缓冲区溢出攻击来获得访问权限。为了防止缓冲区溢出攻击，开发软件时应该尽量使用带有边界检查的函数版本，或者主动进行边界检查。

一个原理性的缓冲区溢出程序如下：

```
Void  sub(char *str)
   {
     char buf[16];
     strcpy(buf,str);
```

```
    return;
  }
Void  main()
  {
    char   large_str[256]
    int i;
    for(i=0;i<255;i++)
          large_str[i]='A';
     sub(large_str)
   }
```

堆栈结构：

```
    16     4     4        4
  [buf]  [ebp] [ret]  [large_str]
```

在该程序中，子程序 sub( )为字符串变量 buf 分配了 16 个字节的内存空间，而主程序 Main( )调用 sub( )要求将 256 个自己赋值给 buf,因此，子程序的堆栈将被 AAAA…覆盖，从而导致程序返回出错。

(4) 信息收集型攻击

消息收集型攻击并不直接对目标系统本身造成危害，它是为进一步的入侵提供必须的信息。这种攻击手段大部分在黑客入侵三步曲中的第一步——窥探设施时使用。

扫描技术就是一种常用的消息收集型攻击方法。扫描技术大致可以分为 ping 扫描、端口扫描以及操作系统检测这三类工具和技巧。通过使用 ping 扫描工具，入侵者可以标示出存活的系统，从而指出潜在的目标。通过使用各种端口扫描工具，入侵者进而可以标示出正在监听着的潜在服务，并将目标系统的暴露程度做出一些假设。最基本的端口扫描技术是 TCP connect()扫描，但是使用这种扫描方式时极易被系统监测出来。因此攻击者发展了一些新的扫描技巧，如 TCP SYN 扫描(半开连接扫描)，TCP Fin 扫描，Tcp ftp bouncce 扫描，用 IP 分片进行 SYN/FIN 扫描，UDP recvfrom 扫描以及 Reverse-ident 扫描等。使用操作系统检测软件可以相当准确地确定目标系统的特定操作系统。从而为以后的攻击系统的活动提供重要的信息。在 10.2 节“安全扫描技术”中将详细分析这一技术。

入侵者往往还利用系统提供的一些信息服务来收集它们所需要的服务。如，在 Unix 系统中入侵者可以使用 finger 服务来得到一些该系统的用户信息。在 Windows 平台下，入侵者往往使用 LDAP 协议来窥探目标网络内部的系统及用户信息。入侵者经常使用的另一个服务是 DNS 查询服务，如果一个 DNS 服务器被错误的配置了允许区域传送的选项，攻

击者就可以轻易获得一个目标系统内的IP地址和一些服务的重要信息。

(5) 假消息攻击

攻击者用配置不正确的消息来欺骗目标系统，以达到攻击的目的被称为假消息攻击。常见的假消息攻击形式有电子邮件欺骗、IP 欺骗、Web 欺骗及 DNS 欺骗等，以下分别解释。

a) 电子邮件欺骗：对于大部分普通因特网用户来说，电子邮件服务是他们使用的最多的网络服务之一。然而由于 SMTP 服务并不强制要求对发送者的身份进行认证，而恶意的电子邮件往往是攻击者恶意攻击用户的有效措施。常见的通过电子邮件的攻击方法有：隐藏发信人的身份，发送匿名或垃圾信件；使用用户熟悉的人的电子邮件地址骗取用户的信任；通过电子邮件执行恶意的代码。

b) IP 欺骗：IP 欺骗的主要动机是隐藏自己的 IP 地址，防止被跟踪。有些网络协议仅仅以 IP 地址作为认证手段，此时伪造 IP 就可以轻易地骗取系统的信任。对于设置了防火墙的目标系统，将自己的 IP 伪装成该系统的内部 IP 有可能使得攻击者冲破目标系统的防火墙。为了防止 IP 欺骗，应该尽量将路由器配制成不允许带源路由选项的 IP 报通过。并在路由器上设置欺骗过滤器，以防止外来的包带有内部的IP地址或者内部的包带有外部的IP地址。

c) Web 欺骗：由于 Internet 的开放性，任何用户都可以建立自己的 Web 站点，同时并不是每个用户都了解 Web 的运行规则。这就使得攻击者的 Web 欺骗成为可能。常见的 Web 欺骗的形式有：使用相似的域名、改写 URL、Web 会话劫持等。

d) DNS 欺骗：修改上一级 DNS 服务记录，重定向 DNS 请求，使受害者获得不正确的IP地址。

10.2 安全扫描技术

10.2.1 安全扫描技术概论

安全扫描技术指手工地或使用特定的软件工具——安全扫描器，对系统脆弱点进行评估，寻找可能对系统造成损害的安全漏洞。扫描主要分为系统扫描和网络扫描两个方面，系统扫描侧重主机系统的平台安全性以及基于此平台的应用系统的安全，而网络扫描则侧重于系统提供的网络应用和服务以及相关的协议分析。

扫描的主要目的是通过一定的手段和方法发现系统或网络存在的隐患，以利于己方及时修补或发动对敌方系统的攻击。

同时，自动化的安全扫描器要对目标系统进行漏洞检测和分析，提供详细的漏洞描述，并针对安全漏洞提出修复建议和安全策略，生成完整的安全性分析报告，为网络管理员完善系统提供重要依据。

安全扫描器主要有两种类型：

(1) 本地扫描器或系统扫描器：扫描器和待检查系统运行于同一结点，进行自身检查。

(2) 远程扫描器或网络扫描器：扫描器和待检查系统运行于不同结点，通过网络远程探测目标结点，寻找安全漏洞。

攻击者在选定攻击目标后首先就是对目标系统进行扫描侦察，搜集目标系统信息，查找可以利用的安全漏洞。目前，可以获得的自动安全扫描器的数目在日益增多。

网络系统的安全漏洞也就是系统防护最弱的部分，不管其他部分如何强壮，一旦此漏洞被利用，就会给网络带来灾难。如何找到这些漏洞并加以保护，是系统管理的重要任务。然而，系统管理员面对大量的服务器、用户和网络设备，工作非常繁重，如果没有自动化工具的辅助，很难检测出所有的安全问题。而且大多数系统管理员缺乏专业技能去跟踪攻击者行踪，查找安全漏洞。如果能够模拟攻击者的行为，扫描系统的安全漏洞，并加以保护和改进，攻击者的可乘之机就会越来越少。

安全扫描器可以通过两种途径提高系统安全性，一是提前警告存在的漏洞，从而预防入侵和误用；二是检查系统中由于受到入侵或操作失误而造成的新漏洞。

10.2.2　安全扫描的内容

安全扫描器需要对目标系统进行全面的检查，以发现可能的安全漏洞。但两大类安全扫描器在检查项目方面又各有所侧重。

1．本地扫描器

本地扫描器分析文件的内容，查找可能存在的配置问题。由于本地扫描器实际上是运行于目标结点上的进程，具有以下几个特点：

(1) 可以在系统上任意创建进程。为了运行某些程序，检测缓冲区溢出攻击，要求扫描器必须做到这一点。

(2) 可以检查到安全补丁一级，以确保系统安装了最新的安全解决补丁。

(3) 可以通过在本地查看系统配置文件，检查系统的配置错误。

而除非能够攻入系统，远程扫描器无法达到上述要求，所以本地扫描器可以查出许多远程扫描器检查不出来的问题。

本地扫描器通常针对特定的宿主操作系统如 Unix、Windows NT/2000 等进行。对 Unix 系统，需要进行以下项目的检查：

(1) 系统完整性检查；
(2) 关键系统文件变化检测；
(3) 用户账户变化检测；
(4) 黑客入侵标记检测；
(5) 未知程序版本；
(6) 不常见文件名；
(7) 可疑设备文件；
(8) 未经授权服务；
(9) 弱口令选择检测；
(10) 有安全漏洞程序版本检测；
(11) 标记可被攻击程序；
(12) 报告需要安装的安全补丁；
(13) 检查系统配置安全性；
(14) 全局信任文件；
(15) crontab 文件；
(16) rc 系统启动文件；
(17) 文件系统 mount 权限；
(18) 打印服务；
(19) 账户配置；
(20) 组配置；
(21) 检查网络服务安全性；
(22) 是否允许 IP 转发；
(23) 标记有风险服务；
(24) ftp 配置；
(25) news 服务器配置；
(26) NFS 配置；
(27) 邮件服务器配置；
(28) Web 服务器配置；
(29) 检查用户环境变量安全性；
(30) 系统文件属主；
(31) 系统文件权限许可；
(32) 文件属主及权限许可；
(33) shell 启动文件；

(34) 用户信任文件；
(35) 应用程序配置文件；
(36) 其他。

对 Windows NT/2000 系统，还有一些特定的安全检查项目：

(1) 是否允许建立 guest 账户；
(2) guest 账户有无口令；
(3) 口令构造和过时原则；
(4) 弱口令选择；
(5) 登录失败临界值；
(6) 注册表权限许可；
(7) 允许远程注册访问；
(8) 独立的注册设置；
(9) 对系统文件和目录不正确的分配许可权；
(10) 非 NT 缺省配置的未知服务；
(11) 运行易遭到攻击的服务，如运行在 Web 服务器上的 SMB 服务等；
(12) 带有许可访问控制设置的共享，可能给远程用户全部访问权；
(13) 其他。

2．远程扫描器

远程扫描器检查网络和分布式系统的安全漏洞。与本地服务器通过运行程序、查看文件来检查安全漏洞不同的是，远程扫描器通过执行一整套综合的穿透测试(Penetration Testing)程序集，发送经精心构造的数据包来检测目标系统。被测系统和扫描器的操作系统可以是同一类型，也可以是不同类型的。本地扫描和远程扫描这两种类型的扫描都能发现目标系统的配置问题和网络安全漏洞。

本地扫描和远程扫描二者相辅相成，在系统环境中对二者加以综合利用非常有必要。有时，它们会以不同的方式检测出同一问题，这一特征对实现多种层次的安全性是很有用的。

远程扫描需要检查的项目很多，这些项目又可以分为以下几大类：

(1) 网络端口扫描；
(2) 系统信息搜集和侦查漏洞：可用于明确目标站点有关信息；
(3) 身份认证机制漏洞；
(4) 拒绝服务攻击；
(5) 防火墙、包过滤及应用程序代理漏洞；

(6) 包过滤规则确认；
(7) WWW、 HTTP 和 CGI 漏洞；
(8) SMTP、POP、IMAP 及邮件传输漏洞；
(9) FTP 安全漏洞；
(10) NFS 安全漏洞；
(11) RPC 远程过程调用服务；
(12) 网络协议欺骗；
(13) Windows NT 网络安全漏洞；
(14) Windows NT 信息搜集和侦查；
(15) 错误配置和系统后门、特洛伊木马检测；
(16) 硬件外设安全漏洞：如打印机、路由器、交换机等设备；
(17) 其他。

10.2.3 安全扫描系统的选择

用户选择安全扫描系统时应注意以下问题：

(1) 升级问题：发现安全漏洞并不是一个静止的过程，安全专家、黑客都在不停地寻找系统分析、设计、实现及配置中可能存在的安全问题，新的安全漏洞不停地被发现，这就决定了安全扫描系统也必须相应地更新自己的安全漏洞库，不可能一劳永逸。优秀的安全扫描系统必须有良好的可扩充性和迅速升级的能力。因此，在选择产品时，首先要注意产品是否能直接从 Internet 升级、升级方法是否能够被非专业人员掌握，同时要注意产品制造者有没有足够的技术力量来保证对新出现漏洞做出迅速的反应。

(2) 可扩充性：对具有比较深厚的网络知识，并且希望自己扩充产品功能的用户来说，可首选有功能模块或插件技术的产品。这样，用户可以针对自己的应用系统设计专用的扫描模块，使扫描器能更好地满足自己机构的实际安全需要。

(3) 局限性：安全扫描系统不是万能的，不能完全依赖它。首先，它不能弥补由于认证机制薄弱带来的问题，也不能弥补协议本身存在的问题；此外，它不能处理所有的数据包攻击，当网络繁忙时无法分析所有的数据流；当受到攻击后要进行调查，离不开安全专家的参与。

10.2.4 安全扫描技术分析

扫描器的工作过程通常包括 3 个阶段：(1)发现目标主机或网络。(2)发现目标后，进一步发现目标系统类型及配置等信息，包括确认目标主机操作系统类型、运行的服务及服务软件版本等等，如果目标是一个网络，那么还可以进一步发现该网络的拓扑结构、路由设

备及各网络主机等信息。(3)测试哪些服务具有安全漏洞。

扫描器是一把双刃剑，系统管理员使用它可以发现自己的系统存在的问题，而攻击者使用它则可以破坏被攻击对象的安全。不管是出于防护的目的，还是出于攻击的目的，二者同样关心如何更多地发现目标主机或网络中存在的安全漏洞。由于未经同意的网络扫描行为往往意味着网络攻击的开始，所以攻击者还需要考虑如何避免扫描动作被发现。安全扫描技术的技术含量更多地体现在这一点上，而从防护的角度来说，也必须了解对手可能用到的手段，才能实施更有效的防范措施。

这里将重点介绍端口扫描技术和系统类型检测技术，这两种技术主要应用于扫描器工作过程的前两个阶段。第三个阶段，测试哪些服务具有安全漏洞，虽然也很重要，但都是一些简单日常检查项目的累积，主要反映在数量上，并不需要特别的技术。

1．端口扫描技术

根据扫描方法的不同，端口扫描技术可分为全开扫描(open scanning)、半开扫描(half-open scanning)、秘密扫描(stealth scanning)、区段扫描(sweeps scanning)等。

这几种技术都可用于查找服务器上打开和关闭的端口，从而发现该服务器上对外开放的服务。但并不是每一种技术都保证能得到正确的结果，它能否成功还依赖于远程目标的网络拓扑结构、IDS(入侵检测系统)、日志机制等配置。**全开扫描**最精确，但它会引起大量的日志记录，同时也很容易被检测到。而使用**秘密扫描**虽然可能避开某些 IDS 和可能绕过防火墙规则，但它发出的带特殊标志的数据包却很可能在网络传输过程中被丢弃，从而造成误报。

图 10.2 给出了目前常见的端口扫描技术及其所属分类。

下面以典型的 TCP connect、SYN flag、SYN|ACK 扫描为例，分析端口扫描的工作原理。

(1) 全开扫描(open scan, TCP connect)

全开扫描技术通过直接同目标主机通过一次完整的 3 次 TCP/IP 握手过程，建立标准 TCP 连接来检查目标主机的相应端口是否打开。

标准的 TCP/IP 连接建立过程如下，如果 server 的端口是打开的，在 client 和 server 间的 TCP 数据包将包括以下握手过程：

client　→　SYN
server　→　SYN | ACK
client　→　ACK

而如果 server 的端口没有打开，上述流程将如下：

client　→　SYN
server　→　RST | ACK

client → RST

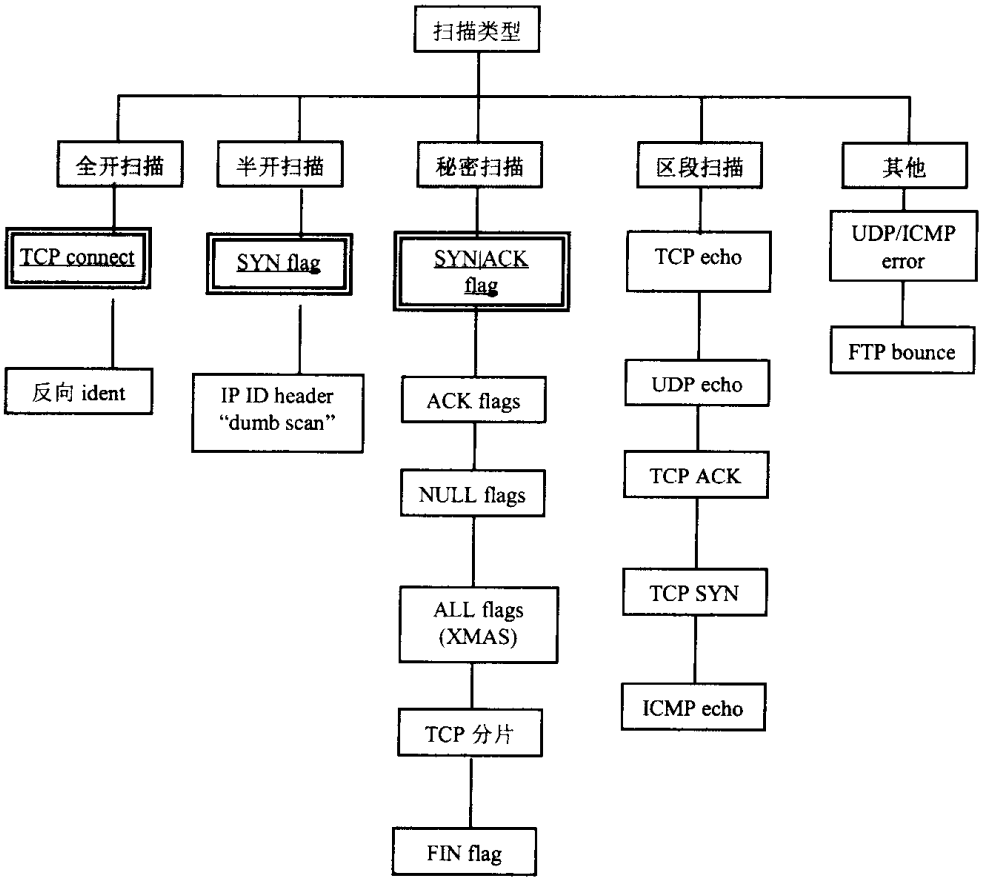


图 10.2 常见端口扫描技术及其分类

全开扫描的优点是快速、精确、不需要特殊用户权限。缺点也是不能进行地址欺骗并且非常容易被检测到。

(2) SYN 扫描

术语“半开”的意思是指 client 端在 TCP 3 次握手尚未完成就单方面中止了连接过程。由于完整的连接还没有建立起来，这种扫描方法常常可以不会被 server 方记录下来，同时也可以避开基于连接的 IDS 系统。同时，半开扫描仍然可以提供相当可靠的端口是否打开的信息。

SYN 扫描是半开扫描方法的一种，它的工作流程同完整的 3 次 TCP/IP 握手类似，只是在第三步时不是发送 ACK 包，而代之以 RST 包要求中止连接。对于打开的端口，SYN 扫描流程如下所示：

client → SYN

server　→　SYN | ACK

client　→　RST

对关闭的端口，流程同全开扫描相同，只是这时不需要发送第三个包了：

client　→　SYN

server　→　RST | ACK

由于半开扫描方式在外在表现上同 SYN flood 拒绝服务攻击方法类似，现在很多 IDS 都可以轻易地检测到这种扫描方法。

半开扫描的优点是快速、可绕开基本的 IDS、避免了 TCP 3 次握手。缺点是需要超级用户权限，IDS 系统可能阻止大量的 SYN 扫描，从而造成误报。

(3) SYN|ACK 扫描

术语“秘密扫描”的意思最早是用来描述避开 IDS 和日志记录的扫描，当时实际上是指的“半开扫描”。但现在它包括了符合下面这些特征的扫描技术：

a) 设置单独的标志位（如 ACK、FIN、RST 等）；

b) NULL 标志位（不设任何标志位）；

c) ALL 标志位（设置所有标志位）；

d) 绕开过滤规则、防火墙、路由器；

e) 表现为偶然的网络数据流；

f) 变化的数据包发散率。

下面以 SYN|ACK 扫描为例来说明秘密扫描方法。

SYN|ACK 扫描技术省掉了 TCP 3 次握手的第一步，client 直接发送 SYN|ACK 包到 server 端，根据 server 的应答，client 可以猜测 server 的端口是否打开。

如果交互过程如下：

client　→　SYN | ACK

server　→　RST

通常说明了 server 的端口是关闭的。

如果 client 收不到 server 的应答，说明 server 的端口可能是开着的。但考虑到收不到应答的原因还可能是超时、发出包被状态检测防火墙过滤掉等原因，这个信息不如 TCP connect 扫描可靠。

SYN|ACK 扫描的优点是快速、可绕开基本的 IDS、避免了 TCP 3 次握手。缺点是需要超级用户权限、不可靠。

2. 系统类型检测技术

由于许多安全漏洞是同操作系统紧密相关的，因此，检测系统类型对于攻击者具有很

重要的作用。攻击者可以先扫描一段网络地址空间，搜集主机类型以及这些主机打开/关闭的端口信息，然后先暂时把这些数据放在一边。当某一系统的新漏洞被发现后，攻击者就可以在已经搜集到的情报中寻找相匹配的系统，从而实施攻击。

检测系统类型主要有 3 种手段，即利用系统旗标(Banner Grabbing)、利用 DNS 信息及利用 TCP/IP 堆栈指纹。

(1) 利用系统旗标

很多时候，利用系统服务给出的信息，立刻就可以知道该系统的信息。这些服务包括 TELNET、FTP、WWW、Mail 等。最简单的检测方法就是直接登陆到该系统提供的服务，检查系统给出的内容。

下面给出了一个利用 TELNET 服务获得系统信息的例子。

```
% telnet 192.168.0.3
HP-UX hpux B.10.01 A 9000/715 (ttyp2)
login:
```

即使关掉了上面的标志，其他服务也可能会将系统类型暴露无遗。如：

```
% telnet 192.168.0.3 smtp
220 target ESMTP Sendmail 8.9.3/8.9.3; Wed, 6 Jun 2001 14:03:59 GMT
```

有些应用可能在响应数据中暴露信息，如：

```
% telnet 192.168.100.4 www
GET /adafasfasf HTTP/1.1
HTTP/1.1 501 Method Not Implemented
Date: Wed, 06 Jun 2001 14:08:43 GMT
Server: Apache/1.3.12 (Unix) ApacheJServ/1.1
Allow: GET, HEAD, OPTIONS, TRACE
Connection: close
Content-Type: text/html; charset=iso-8859-1
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML><HEAD>
<TITLE>501 Method Not Implemented</TITLE>
</HEAD><BODY>
<H1>Method Not Implemented</H1>
get to /adafasfasf not supported.<P>
Invalid method in request get /adafasfasf http/1.1<P>
</BODY></HTML>
```

(2) 利用 DNS 信息

主机信息有时还可能通过 DNS 记录泄露出去。如：

```
www      IN   HINFO   "Sparc Ultra 5" "Solaris 2.6"
solaris  IN   A       192.168.10.32
```

不过由于很少再有管理员配置 HINFO 记录了，这种技术相对说来并不是很有效。

(3) 利用 TCP/IP 堆栈指纹

TCP/IP 堆栈指纹指的是操作系统在实现 TCP/IP 协议时的一些特有的实现特征。由于管理员可能会隐藏应用系统所报的信息，也可能会伪造信息来愚弄攻击者，利用系统旗标的方法有时并不是很可靠。而利 TCP/IP 堆栈作为特殊的“指纹”来确定系统身份的准确性相当高，因为管理员不太可能去修改系统底层的网络的堆栈参数。

目前，利用这种技术实现的工具很多，比较著名有 nmap、queso、checkos 等。可以利用的手段主要有：FIN 探测、BOGUS 标记探测、TCP ISN 采样、“Don't Fragment”位、TCP 的初始化窗口大小、ACK 值、ICMP 错误信息抑制、ICMP 消息引用、ICMP 错误消息回应完整性、服务类型(TOS)、分段控制、TCP 选项等。

习　题　十

1. 入侵行为的目的主要是哪些？
2. 常见的攻击有哪几类？采用什么原理？
3. 如何预防 DDOS 攻击？
4. 编制一个缓冲区溢出攻击程序，在攻击成功后显示“Buffer Overflow Success!”。
5. 安全扫描的目标是什么？如何分类？
6. 半开扫描是怎么实现的？
7. 简述系统类型检测的原理和步骤。

第十一章 入侵检测

网络互连互通后，入侵者可以通过网络实施远程入侵。而入侵行为与正常的访问或多或少有些差别，通过收集和分析这种差别可以发现很大部分的入侵行为，入侵检测技术就是应这种需求而诞生的。经入侵检测发现入侵行为后，可以采取相应的安全措施，如报警、记录、切断或拦截等，从而提高网络的安全应变能力。

11.1 入侵检测原理与技术

入侵检测(Intrusion Detection)是对入侵行为的发觉。它从计算机网络或计算机系统的关键点收集信息并进行分析，从中发现网络或系统中是否有违反安全策略的行为和被攻击的迹象。负责入侵检测的软/硬件组合体称为入侵检测系统(IDS)。

下面讨论入侵检测的方法及体系结构，并对有关入侵检测的测试和标准化问题进行阐述，同时讨论该领域存在的问题及未来的研究方向。

11.1.1 入侵检测的起源

入侵检测的概念最早由 Anderson 在 1980 年提出，他提出了入侵检测系统的三种分类方法。Denning 对 Anderson 的工作进行了扩展，她详细探讨了基于异常和误用检测方法的优缺点，于 1987 年提出了一种通用的入侵检测模型。这个模型独立于任何特殊的系统、应用环境、系统脆弱性和入侵种类，因此提供了一个通用的入侵检测专家系统框架，并由 IDES 原型系统实现。

IDES 原型系统采用的是一个混合结构，包含了一个异常检测器和一个专家系统，如图 11.1 所示。异常检测器采用统计技术刻画异常行为，专家系统采用基于规则的方法检测已知的危害行为。异常检测器对行为的渐变是自适应的，因此引入专家系统能有效防止逐步改变的入侵行为，提高准确率。该模型为入侵检测技术的研究提供了良好的框架结构，为后来各种模型的发展奠定了基础，导致了随后几年内一系列系统原型的研究，如 Discovery、Haystack、MIDS、NADIR、NSM、Wisdom and Sense 等。

直到 1990 年，大部分入侵检测系统还都是基于主机的，它们对活动性的检查局限于操作系统审计跟踪数据以及其他以主机为中心的信息源泉。1988 年 Internet 蠕虫事件的发

生使人们开始对计算机安全高度关注，分布式入侵检测系统(DIDS)随之产生。它最早试图将基于主机和网络监视的方法集成在一起，解决了大型网络环境中跟踪网络用户和文件以及从发生在系统不同的抽象层次的事件中发现相关数据或事件的两大难题。

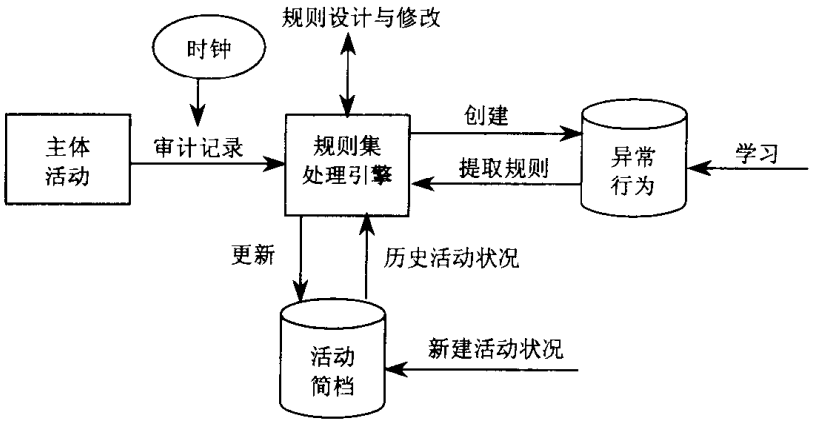


图 11.1　IDES 原型系统

从 20 世纪 80 年代后期开始，数家机构开发了入侵检测工具，其中有一些是对信息安全新技术的尝试。如 ISS RealSecure™ 、CMDS™、NetProwler™、 NetRanger™等。

11.1.2　入侵检测系统的需求特性

一个成功的入侵检测系统至少要满足以下 5 个要求：

(1) 实时性要求：如果攻击或者攻击的企图能够被尽快发现，就有可能查出攻击者的位置，阻止进一步的攻击活动，就有可能把破坏控制在最小限度，并记录下攻击过程，可作为证据回放。实时入侵检测可以避免管理员通过对系统日志进行审计以查找入侵者或入侵行为线索时的种种不便与技术限制。

(2) 可扩展性要求：攻击手段多而复杂，攻击行为特征也各不相同。所以必须建立一种机制，把入侵检测系统的体系结构与使用策略区分开。入侵检测系统必须能够在新的攻击类型出现时，可以通过某种机制在无需对入侵检测系统本身体系进行改动的情况下，使系统能够检测到新的攻击行为。在入侵检测系统的整体功能设计上，也必须建立一种可以扩展的结构，以便适应扩展要求。

(3) 适应性要求：入侵检测系统必须能够适用于多种不同的环境，比如高速大容量的计算机网络环境。并且在系统环境发生改变，比如增加环境中的计算机系统数量，改变计算机系统类型时，入侵检测系统应当依然能够正常工作。适应性也包括入侵检测系统本身对其宿主平台的适应性，即：跨平台工作的能力，适应其宿主平台软、硬件配置的不同情况。

(4) 安全性与可用性要求：入侵检测系统必须尽可能的完善与健壮，不能向其宿主计算机系统以及其所属的计算机环境中引入新的安全问题及安全隐患。并且入侵检测系统在设计和实现时，应该考虑可以预见的、针对该入侵检测系统类型与工作原理的攻击威胁，及其相应的抵御方法。确保该入侵检测系统的安全性与可用性。

(5) 有效性要求：能够证明根据某一设计所建立的入侵检测系统是切实有效的。即：对于攻击事件的错报与漏报能够控制在一定范围内。

11.1.3 入侵检测原理

入侵检测系统是根据入侵行为与正常访问行为的差别来识别入侵行为的，根据识别采用的原理不同，可以分为异常检测、误用检测和特征检测三种。

1. 异常检测

进行异常检测(Anomaly Detection)的前提是认为入侵是异常活动的子集。异常检测系统通过运行在系统或应用层的监控程序监控用户的行为，通过将当前主体的活动情况和用户轮廓进行比较。用户轮廓通常定义为各种行为参数及其阈值的集合，用于描述正常行为范围。当用户活动与正常行为有重大偏离时即被认为是入侵。如果系统错误地将异常活动定义为入侵，称为错报(false positive)；如果系统未能检测出真正的入侵行为则称为漏报(false negative)。这是衡量入侵检测系统性能很重要的两个指标模型如图 11.2 所示。

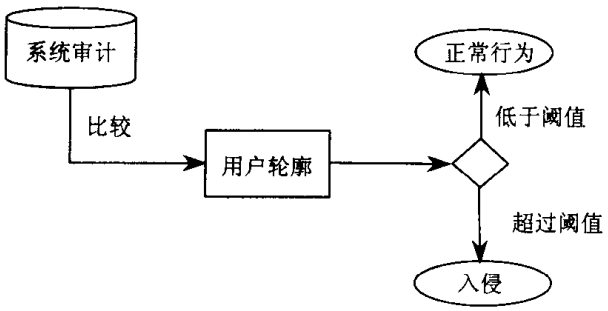


图 11.2 异常检测模型

异常检测系统的效率取决于用户轮廓的完备性和监控的频率。因为不需要对每种入侵行为进行定义，因此能检测未知的入侵。同时系统能针对用户行为的改变进行自我调整和优化，但随着检测模型的逐步精确，异常检测会消耗更多的系统资源。

常见的异常检测方法包括统计异常检测、基于特征选择异常检测、基于贝叶斯推理异常检测、基于贝叶斯网络异常检测、基于模式预测异常检测、基于神经网络异常检测、基于贝叶斯聚类异常检测、基于机器学习异常检测等。目前一种比较流行的方法就是采用数据挖掘技术，来发现各种异常行为之间的关联性，包括源 IP 关联、目的 IP 关联、特征关联、时间关联等。

2. 误用检测

进行误用检测(Misuse Detection)的前提是所有的入侵行为都有可被检测到的特征。误

用检测系统提供攻击特征库，当监测的用户或系统行为与库中的记录相匹配时，系统就认为这种行为是入侵。如果入侵特征与正常的用户行为匹配，则系统会发生错报；如果没有特征能与某种新的攻击行为匹配，则系统会发生漏报。模型如图 11.3 所示。

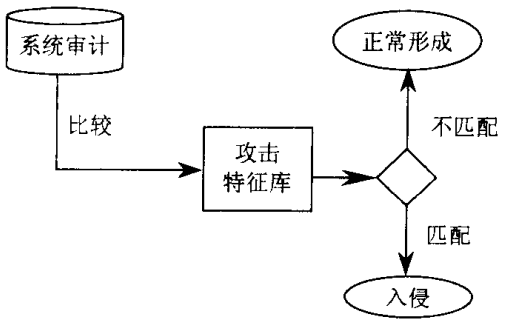


图 11.3　误用检测模型

采用特征匹配，误用模式能明显降低错报率，但漏报率随之增加。攻击特征的细微变化，会使得误用检测无能为力。

常见的误用检测方法包括基于条件概率的误用入侵检测、基于专家系统的误用入侵检测、基于状态迁移的误用入侵检测、基于键盘监控的误用入侵检测、基于模型的误用入侵检测等。

以基于条件概率的误用入侵检测方法为例，该方法将入侵方式对应于一个事件序列，然后通过观测事件发生情况来推测入侵出现。这种方法的依据是外部事件序列，根据贝叶斯定理进行推理检测入侵。令 ES 表示事件序列，先验概率为 $P(\text{Intrusion})$，后验概率为 $P(\text{ES} \mid \text{Intrusion})$，事件出现的概率为 $P(\text{ES})$，则

$$P(\text{Intrusion} \mid \text{ES}) = P(\text{ES} \mid \text{Intrusion}) \frac{P(\text{Intrusion})}{P(\text{ES})}$$

通常可以给出先验概率 $P(\text{Intrusion})$，对入侵报告数据进行统计处理得出 $P(\text{ES} \mid \text{Intrusion})$ 和 $P(\text{ES} \mid -\text{Intrusion})$，于是可以计算出：

$$P(\text{ES}) = ((P(\text{ES} \mid \text{Intrusion} - P(\text{ES}) \mid -\text{Intrusion})) \times P(\text{Intrusion}) + P(\text{ES} \mid -\text{Intrusion})$$

因此可以通过对事件序列的观测，推算出 $P(\text{Intrusion} \mid \text{ES})$。基于条件概率的误用入侵检测方法是在概率理论基础上的一个普遍的方法。它是对贝叶斯方法的改进，其缺点就是先验概率难以给出，而且事件的独立性难以满足。

3. 特征检测

和以上两种检测方法不同，特征检测(Specification-based Detection)关注的是系统本身的行为。定义系统行为轮廓，并将系统行为与轮廓进行比较，对未指明为正常行为的事件定义为入侵。特征检测系统常采用某种特征语言定义系统的安全策略。

这种检测方法的错报与行为特征定义准确度有关，当系统特征不能囊括所有的状态时就会产生漏报。

特征检测最大的优点是可以通过提高行为特征定义的准确度和覆盖范围，大幅度降低漏报和错报率；最大的不足是要求严格定义安全策略，这需要经验和技巧，另外为了维护动态系统的特征库通常是很耗时的事情。

由于这些检测各有优缺点，许多实际入侵检测系统通常同时采用两种以上的方法实现。

11.1.4 入侵检测分类

根据入侵检测系统所检测对象的区别可分为基于主机的入侵检测系统和基于网络的入侵检测系统。

1. 基于主机的入侵检测系统

基于主机的入侵检测系统通过监视与分析主机的审计记录检测入侵。这些系统的实现不全在目标主机上，有一些采用独立的外围处理机，如 Haystack。另外 NIDES 使用网络将主机信息传到中央处理单元。但它们全部是根据目标系统的审计记录工作。能否及时采集到审计记录是这些系统的难点之一，从而有的入侵者会将主机审计子系统作为攻击目标以避开入侵检测系统。

典型的基于主机的入侵检测系统结构如图 11.4 所示。

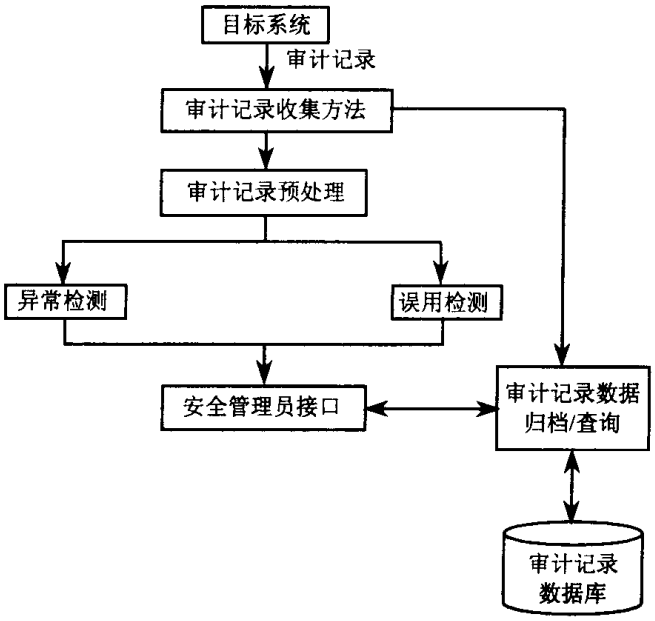


图 11.4 基于主机的入侵检测系统模型

基于主机的入侵检测系统具有检测效率高，分析代价小，分析速度快的特点，能够迅速准确地定位入侵者，并可以结合操作系统和应用程序的行为特征对入侵进行进一步分析。目前，基于主机日志分析的入侵检测系统很多。基于主机的入侵检测系统存在的问题

是：首先它在一定程度上依赖于系统的可靠性，它要求系统本身应该具备基本的安全功能并具有合理的设置，然后才能提取入侵信息；即使进行了正确的设置，对操作系统熟悉的攻击者仍然有可能在入侵行为完成后及时地将系统日志抹去，从而不被发觉；并且主机的日志能够提供的信息有限，有的入侵手段和途径不会在日志中有所反映，日志系统对有的入侵行为不能做出正确的响应。在数据提取的实时性、充分性、可靠性方面基于主机日志的入侵检测系统不如基于网络的入侵检测系统。

2. 基于网络的入侵检测系统

基于网络的入侵检测系统如图 11.5 所示，通过在共享网段上对通信数据进行侦听采集数据，分析可疑现象。与主机系统相比，这类系统对入侵者而言是透明的。由于这类系统不需要主机提供严格的审计，因而对主机资源消耗少，并且由于网络协议是标准的，它可以提供对网络通用的保护，而无需顾及异构主机的不同架构。基于网关的检测系统可以认为是这类系统的变种。

网络入侵检测系统能够检测那些来自网络的攻击，它能够检测到超过授权的非法访问。一个网络入侵检测系统不需要改变服务器等主机的配置。由于它不会在业务系统的主机中安装额外的软件，从而不会影响这些机器的 CPU、I/O 与磁盘等资源的使用，不会影响业务系统的性能。网络入侵检测系统不和路由器、防火墙等关键设备以同样的方式工作，因此也不会成为系统中的关键路径，其发生故障不会影响正常业务的运行。

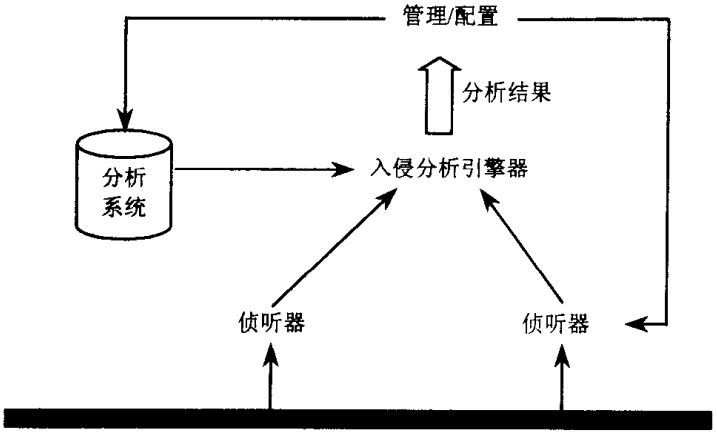


图 11.5　基于网络的入侵检测系统模型

网络入侵检测系统只检查它直接连接网段的通信，不能检测在不同网段的网络包。在使用交换以太网的环境中就会出现监测范围的局限。而安装多台网络入侵检测系统的传感器会使部署整个系统的成本大大增加。同时，网络入侵检测系统为了性能目标通常采用特

征检测的方法，它可以检测出一些普通的攻击，而很难实现一些复杂的需要大量计算与分析时间的攻击检测。

网络入侵检测系统可能会将大量的数据传回分析系统中。在一些系统中监听特定的数据包会产生大量的分析数据流量。有些系统在实现时采用一定方法来减少回传的数据量，对入侵判断的决策由传感器实现，而中央控制台成为状态显示与通信中心，不再作为入侵行为分析器。这样的系统中的传感器协同工作能力较弱。

按照控制方式,入侵检测系统的可以分类以下两类:

(1) 集中式控制：一个中央节点控制系统中的所有入侵检测要素。集中式控制要求在系统组件间提供保护消息的机制，能灵活方便地启动和终止组件，能集中控制状态信息并将这些信息以一种可读的方式传给最终用户。

(2) 与网络管理工具相结合：将入侵检测简单地看作是网络管理的子功能。网管软件包搜集的一些系统信息流可以作为入侵检测的信息源。因此可以将这两个功能集成到一起，便于用户使用。

根据系统的工作方式还可以将入侵检测系统分为离线(off-line)和在线(on-line)检测系统。离线系统是一种非实时的事后分析系统，能通过集中化和自动化节省成本，也能分析大量历史事件。在线系统是实时联机检测系统，它能对入侵迅速作出反应。在大规模的网络环境中保证检测的实时性是目前研究的热点。

11.1.5 入侵检测现状

目前大多数商业 IDS 都只使用了入侵检测的部分方法，这导致攻击者能采用新的攻击方法突破入侵检测系统的保护。传统的入侵检测面临以下的问题：

(1) 随着能力的提高，入侵者会研制更多的攻击工具，以及使用更为复杂精致的攻击手段，对更大范围的目标类型实施攻击；

(2) 入侵者采用加密手段传输攻击信息；

(3) 日益增长的网络流量导致检测分析难度加大；

(4) 缺乏统一的入侵检测术语和概念框架；

(5) 不适当的自动响应机制存在着巨大的安全风险；

(6) 存在对入侵检测系统自身的攻击；

(7) 过高的错报率和误报率，导致很难确定真正的入侵行为；

(8) 采用交换方法限制了网络数据的可见性；

(9) 高速网络环境导致很难对所有数据进行高效实时的分析。

目前入侵检测领域要解决的一些难点包括：

(1) 更有效的集成各种入侵检测数据源，包括从不同的系统和不同的传感器上采集的

数据，以减少虚假报警；

(2) 在事件诊断中结合人工分析；

(3) 提高对恶意代码的检测能力，包括 email 攻击，Java，ActiveX 等；

(4) 采用一定的方法和策略来增强异种系统的互操作性和数据一致性；

(5) 研制可靠的测试和评估标准；

(6) 提供科学的漏洞分类方法，尤其注重从攻击客体而不是攻击主体的观点出发；

(7) 提供对更高级的攻击行为如分布式攻击、拒绝服务攻击等的检测手段。

11.2　入侵检测的数学模型

数学模型的建立有助于更精确地描述入侵问题，特别是异常入侵检测。Dennying 提出了可用于入侵检测的 5 种统计模型：

11.2.1　实验模型

实验模型(Operational Model)基于这样的假设：若变量 x 出现的次数超过某个预定的值，就有可能会出现异常的情况。此模型最适用于入侵活动与随机变量相关的方面，如口令失效次数。

11.2.2　平均值和标准差模型

平均值和标准差模型(Mean and Standard Deviation Model)根据已观测到随机变量 x 的样值 $X_i(i=1, 2, \ldots, n)$以及计算出这些样值的平均值 mean 和标准方差 stddev, 若新的取样值 X_{n+1}不在可信区间[mean–d×stddev, m + d×stddev]内时，则出现异常，其中 d 是标准偏移均值 mean 的参数。这个模型适用于事件计数器、间隔计时器、资源计数器三种类型的随机变量处理。该模型的优点在于不需要为了设定限制值而掌握正常活动的知识。相反，这个模型从观测中学习获取知识，可信区间的变动就反映出知识的增长过程。另外，可信区间依赖于观测到的数据，这样对于用户正常活动定义不同可能差异较大。此模型可加上权重的计算，如最近取样的值权重大些，就会更准确反映出系统的状态。

11.2.3　多变量模型

多变量模型(Multivariate Model)基于两个或多个随机变量的相关性计算，适合于根据多个随机变量的综合结果来识别入侵行为，而不仅仅是单个变量。例如一个程序的使用 CPU 时间和 I/O、用户注册频度、通信会话时间等多个变量来检测入侵行为。

11.2.4 马尔可夫过程模型

马尔可夫过程模型(Markov Process Model)将离散的事件(审计记录)看作一个状态变量，然后用状态迁移矩阵刻画状态之间的迁移频度。若观察到一个新事件，而根据先前的状态和迁移检测矩阵得到新的事件的出现频率太低，则表明出现异常情况。对于通过寻找某些命令之间的转移而检测出入侵行为，这个模型比较适合。

11.2.5 时序模型

时序模型(Time Series Model)通过间隔计时器和资源计数器两种类型的随机变量来描述入侵行为。根据 x_1, x_2, …,x_n之间的相隔时间和它们的值来判断入侵，若在某个时间内 x 出现的概率太低，则表示出现异常情况。这个模型有利于描述行为随时间变化的趋势，缺点在于计算开销大。

11.3 入侵检测的特征分析和协议分析

11.3.1 特征分析

IDS 要有效地检测入侵行为，必须拥有一个强大的入侵特征库。本节将对入侵特征的概念、种类以及如何创建特征进行介绍，并举例如何创建满足实际需要的特征数据模板。

1. 特征的基本概念

IDS 中的特征(signature)是指用于识别攻击行为的数据模板，常因系统而异。不同的 IDS 系统具有的特征功能也有所差异。例如：有些网络 IDS 系统只允许少量地定制存在的特征数据或者编写需要的特征数据，另外一些则允许在很宽的范围内定制或编写特征数据，甚至可以是任意一个特征；一些 IDS 系统只能检查确定的报头或负载数值，另外一些则可以获取任何信息包的任何位置的数据。以下是一些典型的入侵识别方法：

(1) 来自保留 IP 地址的连接企图：可通过检查 IP 报头的来源地址识别；

(2) 带有非法 TCP 标志的数据包：可通过参照 TCP 协议状态转换来识别；

(3) 含有特殊病毒信息的 Email：可通过比较 Email 的主题信息或搜索特定附件来识别；

(4) DNS 缓冲区溢出企图：可通过解析 DNS 域及检查每个域的长度来识别；

(5) 针对 POP3 服务器的 DOS 攻击：通过跟踪记录某个命令的使用频率，并和设定的

阈值进行比较而发出报警信息；

(6) 对 FTP 服务器文件的访问攻击：通过创建具备状态跟踪的特征样板以监视成功登录的 FTP 对话，及时发现未经验证的使用命令等入侵企图。

从以上分类可以看出特征涵盖的范围很广，有简单的报头和数值，也有复杂的连接状态跟踪和扩展的协议分析。

2. 报头值特征

报头值(Header Values)的结构比较简单，而且可以很清楚地识别出异常报头信息，因此，特征数据首先选择报头值。异常报头值的来源大致有以下几种：

(1) 大多数操作系统和应用软件都是在假定 RFC 被严格遵守的情况下编写的，没有添加针对异常数据的错误处理程序，所以许多包含报头值的漏洞利用都会故意违反 RFC 的标准定义；

(2) 许多包含错误代码的不完善软件也会产生违反 RFC 定义的报头值数据；

(3) 并非所有的操作系统和应用程序都能全面拥护 RFC 定义，会存在一些方面与 RFC 不协调；

(4) 随着时间的推移，新的协议可能不被包含于现有 RFC 中。

由于以上几种情况，严格基于 RFC 的 IDS 特征数据就有可能产生漏报或误报效果。对此，RFC 也随着新出现的信息而不断进行更新。

另外合法但可疑的报头值也同样要重视。例如，如果检测到存在到端口 31337 或 27374 的连接，就可初步确定有特洛伊木马在活动，再附加上其他更详细地探测信息，就能够进一步地判断其真假。

3. 确定报头值特征

为了更好地理解如何发现基于报头值的特殊数据报，下面通过分析一个实例的整个过程进行详细阐述。

Synscan 是一个流行的用于扫描和探测系统的工具，其执行过程很具有典型性，它发出的信息包具有多种特征，如不同的源 IP ，源端口 21，目标端口 21 ，服务类型 0，IP ID39426，设置 SYN 和 FIN 标志位，不同的序列号集合，不同的确认号码集合，TCP 窗口尺寸 1 028。可以对以上这些特征进行筛选，查看比较合适的特征数据。以下是特征数据的候选对象：

(1) 只具有 SYN 和 FIN 标志集的数据包，这是公认的恶意行为迹象；

(2) 没有设置 ACK 标志，却具有不同确认号的数据包，而正常情况应该是 0；

(3) 源端口和目标端口都被设置为 21 的数据包，经常与 FTP 服务器关联；

(4) TCP 窗口尺寸为 1 028，IPID 在所有的数据包中为 39426。根据 IP RFC 的定义，这两类数值应有所变化，因此，如果持续不变就表明可疑。

从以上 4 个候选对象中，可以单独选出一项作为基于报头的特征数据，也可以选出多项组合作为特征数据。选择一项数据作为特征有很大的局限性，选择以上 4 项数据联合作为特征也不现实，尽管能够精确地提供行为信息，但是缺乏效率。实际上，特征定义就是要在效率和精确度间取得折中。大多数情况下，简单特征比复杂特征更倾向于误报，因为前者很普遍；复杂特征比简单特征更倾向于漏报，因为前者太过全面，攻击的某个特征会随着时间的推进而变化，完全应由实际情况决定。例如，我们想判断攻击可能采用的工具是什么，那么除了 SYN 和 FIN 标志以外，还需要知道什么其他属性？源端口和目的端口相同虽然可疑，但是许多工具都使用到它，而且一些正常通信也有此现象，因此不适宜选为特征。TCP 窗口尺寸 1 028 尽管可疑，但也会自然地发生。IP ID 为 39426 也一样。没有 ACK 标志的 ACK 数值很明显是非法的，因此非常适于选为特征数据。

接下来我们真正创建一个特征，用于寻找并确定 Synscan 发出的每个 TCP 信息包中的以下属性：

(1) 只设置了 SYN 和 FIN 标志；

(2) IP 鉴定号码为 39426；

(3) TCP 窗口尺寸为 1 028。

第一个项目太普遍，第二个和第三个项目联合出现在同一数据包的情况不很多，因此，将这三个项目组合起来就可以定义一个详细的特征了。再加上其他的 Synscan 属性不会显著提高特征的精确度，只能增加资源的耗费。到此，特征就创建完成。

4. 特征的广谱性

以上创建的特征可以满足对普通 Synscan 软件的探测了。但 Synscan 可能存在多个变种，上述建立的特征很难适用于这些变种的工具，这时就需要结合特殊特征和通用特征。

首先看一个变种 Synscan 所发出的数据信息特征：

(1) 只设置了 SYN 标志，这纯属正常的 TCP 数据包特征；

(2) TCP 窗口尺寸总是 40 而不是 1 028。40 是初始 SYN 信息包中一个罕见的小窗口尺寸，比正常的数值 1 028 少见得多；

(3) 端口数值为 53 而不是 21。

以上 3 种特征与普通 Synscan 产生的数据有很多相似，因此可以初步推断产生它的工具或者是 Synscan 的不同版本，或者是其他基于 Synscan 代码的工具。显然，前面定义的特征已经不能将这个变种识别出来。这时，可以结合普通异常行为的通用特征和一些专用

的特征进行检测。通用特征可以创建如下：

(1) 没有设置确认标志，但是确认数值却非 0 的 TCP 数据包；

(2) 只设置了 SYN 和 FIN 标志的 TCP 数据包；

(3) 初始 TCP 窗口尺寸低于一定数值的 TCP 数据包。

使用以上的通用特征，上面提到过的两种异常数据包都可以有效地识别出来。如果需要更加精确的探测，可再在这些通用特征的基础上添加一些个性数据。

从上面讨论的例子中，我们看到了可用于创建 IDS 特征的多种报头信息。通常，最有可能用于生成报头相关特征的元素为以下几种：

(1) IP 地址：保留 IP 地址、非路由地址、广播地址；

(2) 端口号：特别是木马端口号；

(3) 异常信息包片断：特殊 TCP 标志组合值；

(4) 不应该经常出现的 ICMP 字节或代码。

知道了如何使用基于报头的特征数据，接下来要确定的是检查何种信息包。确定的标准依然是根据实际需求而定。因为 ICMP 和 UDP 信息包是无状态的，所以大多数情况下，需要对它们的每一个包都进行检查。而 TCP 信息包是有连接状态的，因此有时候可以只检查连接中的第一个信息包。其他特征如 TCP 标志会在对话过程的不同数据包中有所不同，如果要查找特殊的标志组合值，就需要对每一个数据包进行检查。检查的数量越多，消耗的资源和时间也就越多。

另外，关注 TCP、UDP 或者 ICMP 的报头信息要比关注 DNS 报头信息更方便。因为 TCP、UDP 以及 ICMP 的报头信息和载荷信息都位于 IP 数据包的载荷部分，比如要获取 TCP 报头数值，首先解析 IP 报头，然后就可以判断出这个载荷采用的是 TCP。而要获取 DNS 的信息，就必须更深入才能看到其真面目，而且解析此类协议还需要更多更复杂的编程代码。实际上，这个解析操作也正是区分不同协议的关键所在，评价 IDS 系统的好坏也体现在是否能够很好地分析更多的协议。

11.3.2　协议分析

以上关注 IP、TCP、UDP 和 ICMP 报头中的值作为入侵检测的特征。现在来看看如何通过检查 TCP 和 UPD 包的内容(其中包含其他协议)来提取特征。首先我们必须清楚某些协议如 DNS 是建立在 TCP 或 UDP 包的载荷中，且都在 IP 协议之上。所以我们必须先对 IP 头进行解码，看它负载是否包含 TCP、UDP 或其他协议。如果负载是 TCP 协议，那么我们就需要在得到 TCP 负载之前通过 IP 协议的负载来处理 TCP 报头的一些信息。

入侵监测系统通常关注 IP、TCP、UDP 和 ICMP 特征，所以它们一般都能够解码部分或全部这些协议的头部。然而，只有一些更高级的入侵监测系统才能进行协议分析。这些

系统的探针能进行全部协议的解码，如 DNS、HTTP、SMTP 和其他一些广泛应用的协议。由于解码众多协议的复杂性，协议分析需要更先进的 IDS 功能，而不能只进行简单的内容查找。执行内容查找的探针只是简单地在包中查找特定的串或字节流序列，并不真正知道它正在检查的是什么协议，所以它只能识别一些明显的或简单特征的恶意行为。

协议分析表明入侵监测系统的探头能真正理解各层协议是如何工作的，而且能分析协议的通信情况来寻找可疑或异常的行为。对于每个协议，分析不仅仅是建立在协议标准的基础上(如 RFC)，而且建立在实际的实现上，因为许多协议事实上的实现与标准并不相同，所以特征应能反映现实状况。协议分析技术观察包括某协议的所有通信并对其进行验证，当不符合预期规则时进行报警。协议分析使得网络入侵监测系统的探头可以检测已知和未知的攻击方法。

以涉及 FTP 协议的众多弱点和漏洞为例。Internet 上可以发现与 FTP 相关的漏洞至少上百个，关于攻击这类漏洞的程序和代码更是数不胜数，而且这些程序之间的差异也很大，确立一个建立在攻击程序基础上的特征集合显然是不可能的。另一个问题是时效性，在已知漏洞的基础上来确定特征，意味着该入侵监测系统不能在新漏洞被公开之前对其进行报警。而在大多数情况下，在漏洞第一次被利用到入侵监测系统能检测这种入侵行为之间会有相当长一段时间的延迟。

1. FTP 的协议解码

下面将以 FTP 协议分析为例，讨论协议分析技术如何进行入侵检测。

一个典型的关于 FTP 命令 MKD 的缓冲区溢出漏洞，可以通过发送包含 shellcode 的代码加以利用。标准的包内容查找的特征包含 shellcode 的序列(通常是 10~20 字节)，而且必须同 FTP 包中的数据完全匹配。协议分析允许我们检测到 MKD 命令的参数并确认它的长度和是否包含二进制数据，通过检查可以发现各种不同的尝试性攻击，而不仅仅发现已知的那些漏洞。

FTP 命令"SITE EXEC"用来在 FTP 服务器上执行命令，它被用于许多攻击中。"SITE"是事实上的 FTP 命令，而"EXEC"是参数。对于包内容查找在包中寻找"SITE EXEC"，试图找到一个大小写无关的匹配。入侵者在"SITE"和"EXEC"之间加入一些空格就可以避免被检测到，而许多 FTP 服务器忽略多余的空格。显然，只通过内容查找并不能查到后一个命令。一个协议分析的特征理解为如何分解"SITE EXEC"，或其他变种，所以仍然可以准确地检测到该攻击。

以上只是一个有限的协议分析，在基于状态的协议分析中，可以检查一个会话的所有通信过程，如整个 FTP 会话，包括初始的 TCP 连接、FTP 认证、FTP 命令和应答、FTP 数据连接的使用、TCP 连接的断开等，以期获得更为精确的检测能力。

2. HTTP 的协议解码

许多攻击者使用的一种简单的 IDS 逃避方法是路径模糊。这种技术的中心思想是改变路径，使它可以在不同的出现方式下做同样的事情。这种技术在 URL 中频繁使用，用来隐藏基于 HTTP 的攻击。攻击者通常利用反斜线符号、单点顺序、双点顺序等来模糊路径。

协议分析可以处理这些技术，因为它完成 WEB 服务器同样的操作，在监听 HTTP 通信时，IDS 从 URL 中析取路径并进行分析。查找反斜线、单点和双点目录，并进行适当的处理，在完成“标准化”URL 操作后，IDS 搜索合法的目录内容以确定异常。

下面展示 IDS 识别并“标准化”URL 的实现步骤：

(1) 解码 IP 报头来监测有效负载包含的协议。在本例中，协议段为 6，它与 TCP 相对应；

(2) 解码 TCP 报头，查找 TCP 目的端口号。假定 WEB 服务器监听端口为 80，若目的端口为 80，则表明用户正在发送 http 请求给服务器；

(3) 依靠 HTTP 协议分析将该请求进行解析成，包括 URL 路径；

(4) 通过处理路径模糊、Hex 编码、双重 Hex 编码和 Unicode 来处理 url 路径；

(5) 分离模糊路径，并进行异常匹配，在匹配成功时发出警告。

这是协议分析真实工作的例子，网络入侵监测识别所有这种攻击的惟一方法就是执行协议分析。

11.4　入侵检测响应机制

11.4.1　对响应的需求

一次完整的入侵检测包括准备、检测和响应三个阶段。响应是一个入侵检测系统必须的部分，没有它入侵检测就失去了存在的价值。通常在准备阶段制订安全策略和支持过程，包括如何组织管理和保护网络资源，以及如何对入侵进行响应等。

在设计入侵检测系统的响应特性时，需要考虑各方面的因素。某些响应要设计得符合通用的安全管理或事件处理标准；而另一些响应则要设计来反映本地管理的重点和策略。因此，一个完好的入侵检测系统应该提供这样一个性能：用户能够裁剪定制其响应机制以符合其特定的需求环境。

在设计响应机制时，必须综合考虑以下几个方面的因素：

(1) 系统用户：入侵检测系统用户可以分为网络安全专家或管理员、系统管理员、安

全调查员。这三类人员对系统的使用目的、方式和熟悉程度不同，必须区别对待；

(2) 操作运行环境：入侵检测系统提供的信息形式依赖其运行环境；

(3) 系统目标：为用户提供关键数据和业务的系统，需要部分地提供主动响应机制；

(4) 规则或法令的需求：在某些军事环境里，允许采取主动防御甚至攻击技术来对付入侵行为。

11.4.2 自动响应

自动响应是最便宜、最方便的响应方式，这种事故处理形式广泛实行。只要它能得到明智小心的实施，也还是比较安全的。但其中存在两个问题：第一个问题是，既然入侵检测系统有产生误报警的问题，就有可能错误地针对一个从未攻击过我们的网络节点进行响应。另一个问题是如果攻击者判定系统有自动响应，他可能会利用这一点来攻击系统。想像一下，他可能与 2 个带自动响应入侵检测系统的网络节点建立起一个与 echo-chargen 等效的反馈环，再对那 2 个节点进行地址欺骗攻击。或者攻击者可从某公司的合作伙伴/客户/供应商的地址发出虚假攻击，使得防火墙把一个公司与另一个公司隔离开，这样两者之间就有了不能逾越的隔离界限。

基于网络的入侵检测系统通常是被动式的，仅分析比特流，它们通常不能做出响应(RESETs 和 SYN | ACK 明显例外)。在大多数商业实现中，都是将入侵检测系统与路由器或防火墙结合起来，用这些设备来完成响应单元的功能。

以下是几种常见的自动响应方式。

(1) 压制调速：对于端口扫描、SYN Flood 攻击技术，压制调速是一种巧妙的响应方式。其思想是在检测到端口扫描或 SYN Flood 行为时就开始增加延时，如果该行为仍然继续，就继续增加延时。这可挫败几种由脚本程序驱动的扫描，例如对 0~255 广播地址 ping 映射，因为它们要靠计时来区分 Unix 和非 Unix 系统的目标。这种方式也被广泛地应用于防火墙，作为其响应引擎(尽管对其使用还存在争议)。

(2) SYN | ACK 响应：设想入侵检测系统已知某个网络节点用防火墙或过滤路由器对某些端口进行防守，当入侵检测系统检测到向这些端口发送的 TCP SYN 包后，就用一个伪造的 SYN | ACK 进行回答。这样的话，攻击者就会以为他们找到了许多潜在的攻击目标，而实际上他们得到的只不过是一些误报警。最新一代的扫描工具以其诱骗功能给入侵检测带来很多的问题和麻烦，而 SYN | ACK 响应正是回击它们的最好办法。

(3) RESETs：对使用这一技术应该持慎重的保留态度。RESETs 可能会断开与其他人的 TCP 连接。这种响应的思想是如果发现一个 TCP 连接被建立，而它连接的是你要保护的某种东西，就伪造一个 RESET 并将其发送给发起连接的主机，使连接断开。尽管在商用入侵检测系统中很可能得到这一响应功能，但它不是经常被用到。一旦与误报警联系在

一起，这个技术就变得很有意思。另外攻击者可能很快就会修补他们的 TCP 程序使其忽略 RESET 信号。当然，还有一种方式是向内部发送 RESET。

11.4.3　蜜罐

高级的网络节点可以采用路由器把攻击者引导到一个经过特殊装备的系统上，这种系统被称为蜜罐(Honeypot)。蜜罐是一种欺骗手段，它可以用于错误地诱导攻击者，也可以用于收集攻击信息，以改进防御能力。蜜罐能采集的信息量由自身能提供的手段以及攻击行为的数量决定。

下面简单介绍几种常见的蜜罐：

(1) BOF：由 NFR 公司的 Marcus Ranum 和 crew 开发，能运行于大多数 Windows 平台。通过模拟有限的几种服务，它能够记录针对这些端口的攻击场景，并通过伪造数据包对攻击者进行欺骗。

(2) Deception Toolkit：由 Fred Cohen 开发，结合 Perl 和 C 两种语言编写，可模仿大量服务程序。DTK 是一个状态机，实际上它能虚拟任何服务，并可方便地利用其中的功能直接模仿许多服务程序。

(3) Specter：Specter 是商业产品，运行在 Windows 平台上。和 BOF 相比，它能模拟多种操作系统上较大范围的端口，同时提供信息自动收集和处理功能

(4) Mantrap：由Recourse公司开发的商用软件。它能模拟最多达四种操作系统，并允许管理员动态配置，并能在模拟平台上加载应用程序，收集包括从网络层到应用层的各种攻击信息。但这个产品也存在一些限制，如能模拟的操作系统产品有限，只能在 Solaris 上运行等。

(5) Honeynets：这是一个专门设计来让人“攻陷”的网络，一旦被入侵者所攻破，入侵者的一切信息、工具等都将被用来分析学习。其想法与 honeypot 相似，但两者之间还是有些不同点：

1) Honeynet 是一个网络系统，而并非某台单一主机，这一网络系统是隐藏在防火墙后面的，所有进出的数据都受到关注、捕获及控制。这些被捕获的数据可以用来研究分析入侵者们使用的工具、方法及动机。在 Honeynet 中可以使用各种不同的操作系统及设备，如 Solaris, Linux, Windows NT, Cisco Switch, 等等。这样建立的网络环境看上去会更加真实可信，同时还可以在不同的系统平台上面运行着不同的服务，比如 Linux 的 DNS server，Windows NT 的 webserver 或者一个 Solaris 的 FTP server，使用者可以学习不同的工具以及不同的策略——或许某些入侵者仅仅把目标定位于几个特定的系统漏洞上，而我们这种多样化的系统，就可能更多地揭示出他们的一些特性。

2) 在 Honeynet 中的所有系统都是标准的机器，上面运行的都是真实完整的操作系统

及应用程序——就像你在互联网上找到的系统一样。在 Honeynet 里面找到的存在风险的系统，与在互联网上一些公司组织的毫无二致。可以简单地把各种操作系统放到 Honeynet 中，并不会对整个网络造成影响。

传统意义上的信息安全，一般都是防御性质的，比如防火墙、入侵检测系统、加密等等，它们都是用来保护我们的信息资产，其策略是，先考虑系统可能出现哪些问题，然后对问题一一进行分析解决。而 Honeynet 希望做到的是改变这一思路，使其更具交互性——它的主要功能是用来学习了解敌人(入侵者)的思路、工具、目的。通过获取这些技能，互联网上的组织将会更好地理解他们所遇到的威胁，并且理解如何防止这些威胁。但这种技术的使用必须要考虑到法律上的因素。

11.4.4 主动攻击模型

更为主动的响应是探测到进攻时发起对攻击者的反击；但这个方法是非常危险的，它不但是非法的，也会影响到网络上无辜的用户。这个方法如图 11.6 所示。

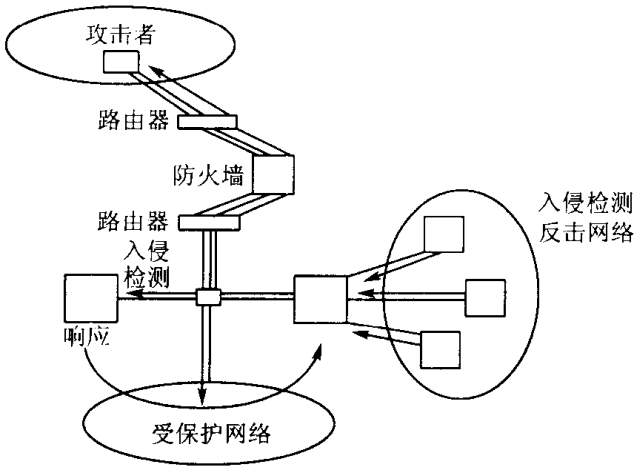


图 11.6 主动攻击模型

11.5 绕过入侵检测的若干技术

入侵检测系统的应用日益广泛，但也受到很多技术条件的限制，本节将结合网络入侵检测系统(NIDS)的特点，探讨一些典型的技术和方法。掌握这些技术和方法，能对入侵检测系统的局限性有进一步的认识，有助于提高系统的检测和防护能力。

目前市场上有大量的网络入侵检测产品，这些产品来自不同的厂商，但按照所采用的

检测技术主要可以分为两大类。一类是特征分析法，系统捕获网络数据包，并将这些包和特征库中的攻击特征进行匹配，从而确定潜在的攻击行为。特征分析通常不对数据包进行任何处理，只是简单地进行特征字符串匹配。这类系统提供的检测能力有限，因此很难应用于重载荷和高带宽的网络环境中。另一类是协议分析法，系统同样捕获和分析网络数据包，但具有一定的协议理解能力，能通过模拟主机和应用环境的协议转换流程来进行检测。这类系统能检测较复杂的攻击，有效降低误报率和错报率，但其应用也受处理能力的限制。

11.5.1　对入侵检测系统的攻击

1．直接攻击

网络入侵检测系统通常运行在一定的操作系统之上，其本身也是一个复杂的协议栈实现，这就意味着 NID 本身可能受到 smurf、synflood 或 jolt2 等攻击。如果安装 IDS 的操作系统本身存在漏洞或 IDS 自身防御力差，此类攻击很有可能造成 IDS 的探测器丢包、失效或不能正常工作。但是随着 IDS 技术的发展，一些 NIDS 采用了双网卡技术，一个绑定 IP 用于控制台通信，一个则无 IP，用于收集网络数据包。对于没有 IP 的 NIDS 则无法直接攻击，而且新的 IDS 一般采用了协议分析技术，提高了 IDS 捕获和处理数据的性能，所以直接攻击的方法会失效。

2．间接攻击

一般的 NIDS 都有入侵响应的能力，攻击者可以利用其响应进行间接攻击，如使入侵次数迅速增多，发送大量的报警信息，并占用大量的 CPU 资源；或使防火墙错误配置，造成一些正常的 IP 无法访问等。

11.5.2　对入侵检测系统的逃避

1．针对 HTTP 请求

URL 编码：将 URL 进行编码，可以避开一些采用规则匹配的 NIDS。如十六进制编码、Unicode/Wide 编码、斜线 (包括"/"和 "\")、增加目录以及各种等价命令替换等。另外，还包括各种不规则方式的编码，如用 Tab 替换空格、NULL 方式、虚假的请求结束以及长 URL 等。

会话组合：如果将请求放在不同的报文中发出，IDS 就可能不会匹配出攻击了，但这种攻击行为无法逃避采用协议分析和会话重组技术的 NIDS。

大小写敏感：DOS/Windows 和 Unix 不同，它对大小写不敏感。这种手段可能造成一

些老式的 IDS 匹配失败。

2. 针对缓冲区溢出

一些 NIDS 检测远程缓冲区溢出的主要方式是通过检测数据载荷里是否包含“/bin/sh”或是否含有大量的 NOP。针对这种识别方法，某些溢出程序的 NOP 考虑用“eb 02”代替。另外，目前出现了一种多形态代码技术，使攻击者能潜在的改变代码结构来欺骗许多 NIDS，但它不会破坏最初的攻击程序。经过伪装的溢出程序，每次攻击所采用的 shellcode 都不相同，这样降低了被检测到的可能。有些 NIDS 能依据长度、可打印字符等判断这种入侵，但会造成大量的错报。

3. 针对木马

IDS 对木马和后门程序一般是通过端口来判断的。如果按照木马的默认端口进行连接，很容易就能识别。目前大部分的木马都使用浮动端口，且采用加密方式传输数据，这样 NIDS 就很难检测。

11.5.3 其他方法

(1) 慢扫描：网络入侵检测系统按照特定数据源的访问频率来判断网络扫描。扫描器可以通过延长扫描时间，降低扫描频率来躲避这种检测。

(2) 分片：数据报分片是 TCP/IP 协议适应各种网络环境应用的机制之一。很多入侵检测系统为了避免载荷过大，一般都不进行分片重组，因此，一些攻击数据可以通过分布在不同的数据分片中，逃避检测。

(3) 地址欺骗：利用代理或者伪造 IP 包进行攻击，隐藏攻击者的 IP，使 NIDS 不能发现攻击者。目前的 NIDS 只能根据异常包中的地址来判断攻击源。

(4) 利用 LLKM 处理网络通信：利用 LLKM 简单、临时改变 TCP/IP 协议栈的行为，如更改出现在网络传输线路上的 TCP 标志位，就可以躲避一些 NIDS。

11.6 入侵检测标准化工作

目前的入侵检测系统大部分是基于各自的需求和设计独立开发的，不同系统之间缺乏互操作性和互用性，这对入侵检测系统的发展造成了障碍。网络安全的发展要求 IDS 能够与其他组件如访问控制、防火墙等系统共享信息和相互通信，相互协作，形成一个整体有效的安全保障系统。为此 DARPA 在 1997 年 3 月开始着手 CIDF (Common Intrusion Detection

Framework，公共入侵检测框架)标准的制定。现在加州大学 Davis 分校的安全实验室已经完成了 CIDF 标准，IETF(Internet Engineering Task Force, Internet 工程任务组)成立了 IDWG(Intrusion Detection Working Group，入侵检测工作组)负责建立 IDEF(Intrusion Detection Exchange Format，入侵检测数据交换格式)标准，并提供支持该标准的工具，以便更高效率地开发 IDS 系统。

CIDF 是一套规范，它定义了 IDS 表达检测信息的标准语言以及 IDS 组件之间的通信协议。符合 CIDF 规范的 IDS 可以共享检测信息，相互通信，协同工作，还可以与其他系统配合实施统一的配置响应和恢复策略。CIDF 的主要作用在于集成各种 IDS 使之协同工作，实现各 IDS 之间的组件重用，所以 CIDF 也是构建分布式 IDS 的基础。

CIDF 的规格文档由四部分组成，分别为：

(1) 体系结构(The Common Intrusion Detection Framework Architecture)；

(2) 规范语言(A Common Intrusion Specification Language)；

(3) 内部通信(Communication in the Common Intrusion Detection Framework)；

(4) 程序接口(Common Intrusion Detection Framework APIs)。

其中体系结构阐述了一个标准 IDS 的通用模型；规范语言定义了一个用来描述各种检测信息的标准语言；内部通信定义了 IDS 组件之间进行通信的标准协议；程序接口提供了一整套标准的应用程序接口(API 函数)。以下将分别对 CIDF 的 4 个组成部分进行结构分析。

11.6.1 CIDF 体系结构

CIDF 的体系结构文档阐述了一个 IDS 的通用模型。它将一个 IDS 分为以下 4 个组件：

(1) 事件产生器(Event generators)；

(2) 事件分析器(Event analyzers)；

(3) 事件数据库(Event databases)；

(4) 响应单元(Response units)。

CIDF 将 IDS 需要分析的数据统称为事件(event)，它可以是基于网络的 IDS 从网络中提取的数据包，也可以是基于主机的 IDS 从系统日志等其他途径得到的数据信息。

CIDF 组件之间是以通用入侵检测对象(generalized intrusion detection objects，以下简称为 GIDO)的形式交换数据的，一个 GIDO 可以表示在一些特定时刻发生的一些特定事件，也可以表示从一系列事件中得出的一些结论，还可以表示执行某个行动的指令。

CIDF 中的事件产生器负责从整个计算环境中获取事件，但它并不处理这些事件，而是将事件转化为 GIDO 标准格式提交给其他组件使用，显然事件产生器是所有 IDS 所需要

的，同时也是可以重用的。CIDF 中的事件分析器接收 GIDO，分析它们，然后以一个新的 GIDO 形式返回分析结果。CIDF 中的事件数据库负责 GIDO 的存贮，它可以是复杂的数据库，也可以是简单的文本文件。CIDF 中的响应单元根据 GIDO 做出反应，它可以是终止进程、切断连接、改变文件属性，也可以只是简单的报警。

CIDF 将各组件之间的通信划分为三个层次结构：GIDO 层(GIDO layer)、消息层(Message layer)和传输层(Negotiated Transport layer)。其中传输层不属于 CIDF 规范，它可以采用很多种现有的传输机制来实现。消息层负责对传输的信息进行加密认证，然后将其可靠地从源传输到目的地，消息层不关心传输的内容，它只负责建立一个可靠的传输通道。GIDO 层负责对传输信息的格式化，正是因为有了 GIDO 这种统一的信息表达格式，才使得各个 IDS 之间的互操作成为可能。

CIDF 定义了 IDS 系统和应急系统之间通过交换数据方式，共同协作来实现入侵检测和应急响应。CIDF 的互操作有下面三类：

(1) 配置互操作，可相互发现并交换数据；

(2) 语法互操作，可正确识别交换的数据；

(3) 语义互操作，可相互正确理解交换的数据。

同时，CIDF 定义了 IDS 系统的六种协同方式，即分析、互补、互纠、核实、调整、响应等。

11.6.2 CIDF 规范语言

CIDF 的规范语言文档定义了一个公共入侵标准语言(Common Intrusion Specification Language，以下简称为 CISL)，各 IDS 使用统一的 CISL 来表示原始事件信息、分析结果和响应指令，从而建立了 IDS 之间信息共享的基础。CISL 是 CIDF 的最核心也是最重要的内容。CISL 设计的目标如下：

(1) 表达能力：CISL 语言应当具有足够的词汇和复杂的语法来实现广泛的表达，主要针对事件的因果关系、事件的对象角色、对象的属性、对象之间的关系、响应命令或脚本等几个方面。

(2) 表示的惟一性：要求发送者和接收者对协商好的目标信息能够相互理解；

(3) 精确性：两个接收者读取相同的消息不能得到相反的结论；

(4) 层次化：语言当中有一种机制能够用普通的概念定义详细而又精确的概念；

(5) 自定义：消息能够自我解析说明；

(6) 效率：任何接收者对语言的格式理解开销不能成倍增加；

(7) 扩展性：语言里有一种机制能够让接收者理解发送者使用的词汇，或者是接收者能够利用消息的其余部分说明解析新的词汇的含义；

(8) 简单：不需理解整个语言就能接收和发送消息；

(9) 可移植性：语言的编码不依赖于网络的细节或特定主机的消息；

(10) 容易实现。

为了能够满足以上的要求，CISL 使用了一种类似 Lisp 语言的 S 表达式，它的基本单位由语义标志符 SID、数据和圆括号组成。例如：(HostName'first.example.com')，其中 HostName 为 SID，表示后面的数据是一个主机名，'first.example.com'为数据，括号将两者关联。多个 S 表达式的基本单位递归组合在一起，构成整个 CISL 的 S 表达式。在 CISL 中，所有信息(原始事件、分析结果、响应指令等)均是用这种 S 表达式来表示的。例如下面的 S 表达式:

```
(Delete
    (Context
        (HostName 'first.example.com')
        (Time '16:40:32 Jun 14 1998')
    )
    (Initiator
        (UserName 'joe')
    )
    (Source
        (FileName '/etc/passwd')
    )
)
```

表示用户名为'joe'的用户在 1998 年 6 月 14 日 16 点 40 分 32 秒删除了主机名为'first.example.com'的主机上的文件'/etc/passwd'。这是一个事件描述，其中的 Delete、Context、HostName、Time 等均为 SID。

这些 SID、S 表达式以及 S 表达式的组合、递归、嵌套等构成了 CISL 的全部表达能力，也即 CISL 本身。在计算机内部处理 CISL 时，为节省存储空间提高运行效率，必须对 ASCII 形式的 S 表达式进行编码，将其转换为二进制字节流的形式，编码后的 S 表达式就是 GIDO。GIDO 是 S 表达式的二进制形式，是 CIDF 各组件统一的信息表达格式，也是组件之间信息数据交换的统一形式。

11.6.3 CIDF 的通信机制

CIDF 要实现协同工作，还要解决构件之间通信方面的问题：

(1) CIDF 的一个构件怎样才能安全地联系到其他构件，其中包括构件的定位和构件的

认证；

(2) CIDF 如何保证构件之间达到安全有效地通信。

为了解决以上两个问题，CIDF 采用了一种称为 Matchmaker 的通信架构。它提供了一个标准的、统一的方法，使得 CIDF 的构件之间互相识别和定位，让它们能够共享信息。这样极大地提高构件间的互操作能力，从而使入侵检测和应急系统的开发变得容易。Matchmaker 支持目录服务，提供多种方式的查询构件。Matchmaker 体系结构的中间件是为 CIDF 的构件提供查询服务。

CIDF 的内部通信文档描述了两种 CIDF 组件之间通信的机制，一种为匹配服务(Matchmaking Service)法，另一种为消息层(Message Layer)法。

CIDF 的匹配服务(也叫做匹配器)，为 CIDF 各组件之间的相互识别、定位和信息共享提供了一个标准的统一机制。匹配器的实现是基于轻目录存取协议(The Lightweight Directory Access Protocol，以下简称为 LDAP)的，每个组件通过目录服务注册，并公告它能够产生或能够处理的 GIDO，这样组件就被分类存放，其他组件就可以方便地查找到那些它们需要通信的组件。目录中还可以存放组件的公共密钥，从而实现对组件接收和发送 GIDO 时的身份认证。CIDF 的匹配器由四部分构成，它们分别为：

(1) 通信模块(the communications module)；

(2) 匹配代理(the broker)；

(3) 认证和授权模块(the authentication/authorization module)；

(4) 客户端缓冲区(the client-side cache)。

在 CIDF 的内部通信文档中还包括匹配器的目录数据格式与组织、组件查找协议以及匹配协议等。CIDF 的消息层在易受攻击的环境中实现了一种安全(保密、可信、完整)并可靠的信息交换机制。使用消息机制主要是为了达到以下目的：

(1) 使通信与阻塞和非阻塞处理无关；

(2) 使通信与数据格式无关；

(3) 使通信与操作系统无关；

(4) 使通信与编程语言无关；

利用消息格式中的选择项，消息层提供了路由信息追踪、数据加密和认证等功能，这些功能是在客户端与服务器之间的握手阶段完成的，从而既保证了数据的安全性又最大程度地减少了传输开销。应用程序可根据实际需要来决定是否使用这些选择项。默认情况下消息传输是基于 UDP 的，且使用端口 0x0CDF 作为 CIDF 消息传输的服务端口。在 CIDF 的内部通信文档中还描述了各种情况下的消息处理过程，其中包括与传输层协议无关的标准消息处理过程，基于可信 UDP 的消息处理过程，基于 CIDF 加密认证机制的消息处理过程等。

11.6.4　CIDF 程序接口

CIDF 的程序接口文档描述了用于 GIDO 编解码以及传输的标准应用程序接口(以下简称为 API)，它包括以下几部分内容：

(1) GIDO 编码和解码 API(GIDO Encoding/Decoding API Specification)；

(2) 消息层 API(Message Layer API Specification)；

(3) GIDO 动态追加 API(GIDO Addendum API)；

(4) 签名 API(Signature API)；

(5) 顶层 CIDF 的 API(Top-Level CIDF API)。

GIDO 有两种表现形式：一种为逻辑形式，表现为 ASCII 文本的 S 表达式，它是用户可读的；另一种为编码形式，表现为二进制的与机器相关的数据结构。GIDO 编解码 API 定义了 GIDO 在这两种形式之间进行转换的标准程序接口，它使应用程序可以方便地转换 GIDO 而不必关心其具体技术细节。每类 API 均包含数据结构定义、函数定义和错误代码定义等。

目前，CIDF 小组准备加入 Internet 工程任务组(IETF)，有可能使得 CIDF 成为入侵检测领域的标准。上述 CIDF 的内容仅仅是 Internet 草案。不过，CIDF 的重要贡献在于将软件构件理论应用到入侵检测系统中，定义构件之间的接口方法，从而使得不同的构件能够互相通信和协作。事实上，今天已有不少 IDS 研究原型和产品，但是都未考虑重用和环境变化。

习 题 十 一

1. 简述入侵检测的目标和分类。
2. 异常检测和误用检测有何区别？
3. 标准差检测模型如何实现？举例说明。
4. 如何实现 SYN-Flooding 攻击检测？
5. 如何通过协议分析实现入侵检测？
6. 如何逃避缓冲区溢出检测？

第十二章 防 火 墙

12.1 防火墙概述

12.1.1 什么是防火墙

防火墙的原义是指古代修筑在房屋之间的一道墙，当某一房屋发生火灾的时候，它能防止火势蔓延到别的房屋。这里所说的防火墙是指目前一种广泛应用的网络安全技术。它用来控制两个不同安全策略的网络之间互访，从而防止不同安全域之间的相互危害。

防火墙一般是指在两个网络间执行访问控制策略的一个或一组系统。防火墙现在已成为将内部网接入外部网(如 Internet)时所必需的安全措施。防火墙可能在一台计算机上运行，也可能在计算机群上运行。

对防火墙的明确定义来自 AT&T 的两位工程师 William Cheswick 和 Steven Bellovin，他们将防火墙定义为置于两个网络之间的一组构件或一个系统，具有以下属性：

- 它是不同网络或网络安全域之间信息流通过的惟一出入口，所有双向数据流必须经过它；
- 只有被授权的合法数据，即防火墙系统中安全策略允许的数据，才可以通过；
- 该系统应具有很高的抗攻击能力，自身能不受各种攻击的影响。

简言之，防火墙将网络分隔为不同的物理子网，限制威胁从一个子网扩散到另一子网，正如传统意义的防火墙能防止火势蔓延一样。防火墙用于保护可信网络免受非可信网络的威胁，同时仍有限制地允许双方通信。通常，这两个网络称为内部网和外部网。虽然现在许多防火墙用于 Internet 和内部网之间，但是也可以在任何网络之间和企业网内部使用防火墙。防火墙通常安装在内部网络和外部网络的连接点上，如图 12.1 所示。

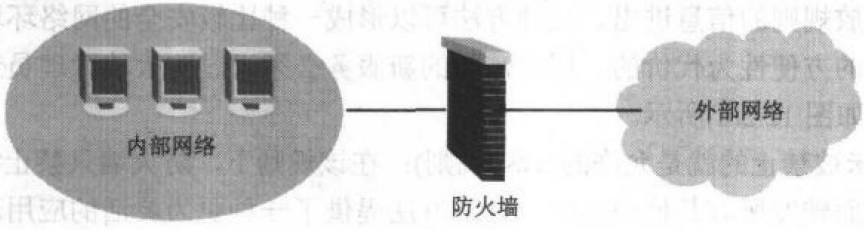


图 12.1 防火墙系统的位置

12.1.2　防火墙的功能

本质上来讲，防火墙是两个网络间的隔断，只允许符合规则的一些数据通过。同时防火墙自身应当足够安全，不易被攻入。

早期的防火墙主要用来提供服务控制，现在已经扩展为多种服务，还包括方向控制、用户控制、行为控制等。

- 服务控制：确定可以访问的因特网服务的类型；
- 方向控制：确定特定的服务请求通过防火墙流动的方向；
- 用户控制：控制用户对特定服务的访问；
- 行为控制：控制怎样使用特定的服务；例如：可以使外部只能访问一个本地服务器的部分信息。

归纳起来，防火墙主要作用如下：

(1) 防火墙对内部网实现了集中的安全管理，可以强化网络安全策略，比分散的主机管理更经济易行；

(2) 防火墙能防止非授权用户进入内部网络；

(3) 防火墙可以方便地监视网络的安全并及时报警；

(4) 使用防火墙，可以实现网络地址转换(Network Address Translation，即 NAT)，利用 NAT 技术，可以缓解地址资源的短缺，隐藏内部网的结构；

(5) 利用防火墙对内部网络的划分，可以实现重点网段的分离，从而限制安全问题的扩散；

(6) 所有的访问都经过防火墙，因此它是审计和记录网络的访问和使用的理想位置。

通过选择市场上优秀的防火墙产品，制定出合理的安全策略，内部网络就可以在很大程度上避免遭受攻击。

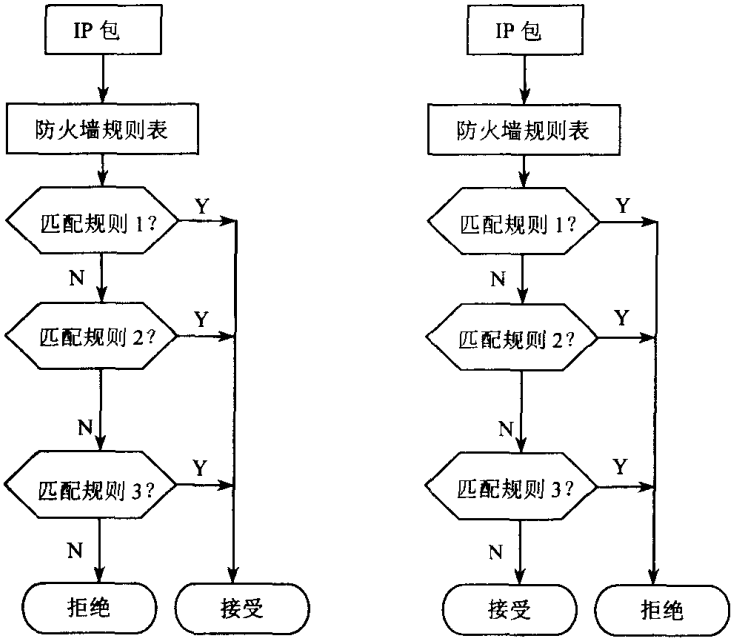
12.1.3　防火墙的基本规则

配置防火墙有两种基本规则：

(1) 一切未被允许的就是禁止的(No 规则)：在该规则下，防火墙封锁所有的信息流，只允许符合开放规则的信息进出。这种方法可以形成一种比较安全的网络环境，但这是以牺牲用户使用的方便性为代价的，用户需要的新服务必须通过防火墙管理员逐步添加，其规则检查策略如图 12.2(a)所示。

(2) 一切未被禁止的就是允许的(Yes 规则)：在该规则下，防火墙只禁止符合屏蔽规则的信息进出，而转发所有其他信息流。这种方法提供了一种更为灵活的应用环境，但很难提供可靠的安全防护，其规则检查策略如图 12.2(b)所示。具体选择哪种规则，要根据实际

情况决定，如果出于安全考虑就选择第一条准则，如果出于应用的便捷性考虑就选用第二条准则。



(a) No 规则　　(b) Yes 规则

图 12.2　防火墙基本规则

12.2　防火墙技术

防火墙采用了两种基本的技术：数据包过滤和代理服务。

12.2.1　数据包过滤技术

数据包过滤是在网络的适当位置，根据系统设置的过滤规则。对数据包实施过滤，只允许满足过滤规则的数据包通过并被转发到目的地，而其他不满足规则的数据包被丢弃。当前大多数的网络路由器都具备一定的数据包过滤能力，很多情况下，路由器除了完成路由选择和转发的功能之外，还可进行数据包过滤。

在使用 TCP 协议的 IP 数据包中，每个数据包的报头信息大致包括以下内容：

- IP 源地址；
- IP 目的地址；

- IP 协议字段；
- TCP 源端口；
- TCP 目的端口；
- TCP 标志字段。

在使用 UDP 协议的 IP 数据包中，每个数据包的报头信息大致包括以下内容：

- IP 源地址；
- IP 目的地址；
- IP 协议字段；
- UDP 源端口；
- UDP 目的端口。

数据包过滤器通过检查数据包的报头信息，根据数据包的源地址、目的地址和以上的其他信息相组合，按照过滤规则来决定是否允许数据包通过。数据包过滤器在接收数据包时一般不判断数据包的上下文，只根据目前的数据报的内容作决定(这避免不了重放攻击)。Internet 上的服务一般都与特定的端口号有关，如 FTP 一般工作在 21 端口，TELNET 工作在 23 端口，WEB 服务在 80 端口，因此可通过包过滤器来禁止某项服务，例如：可通过包过滤器禁止所有通过 80 端口的数据包来禁止 Web 服务。

通过对普通路由器和包过滤路由器进行比较，可以进一步了解包过滤的工作原理。普通路由器只检查一下每个数据包的目的地址，为数据包选择它所知道的最佳路由，将这个数据包发送到目的地址。而包过滤路由器除了执行普通路由器的功能外，还根据设定的包过滤规则决定是否转发数据包。

12.2.2　代理服务

代理服务是在防火墙主机上运行的专门的应用程序或服务器程序，这些程序根据安全策略处理用户对网络服务的请求，代理服务位于内部网和外部网之间，处理其间的通信以替代相互直接的通信。

代理服务具有两个部件：一个是服务器端代理，一个是客户端代理。所谓服务器端代理，是指代表客户处理在服务器连接请求的程序。当服务器端代理得到一个客户的连接请求时，它们将核实客户请求，并经过特定的安全化的 Proxy 应用程序处理连接请求，将处理后的请求传递到真实的服务器上，然后接受服务器应答，并做进一步处理，最后将答复交给发出请求的最终客户。代理服务器在外部网络向内部网络申请服务时发挥了中间转接的作用。服务器端代理可以是一个运行代理服务程序的网络主机，客户端代理可以是经过配置的普通客户程序，例如 FTP、Telnet、IE。客户和客户端代理通信，服务器和服务器端代理通信，这两个代理相互之间直接通信，代理服务器检查来自客户端代理的请求，根

据安全策略认可或否认这个请求。

代理服务器的工作机理如图 12.3 所示。

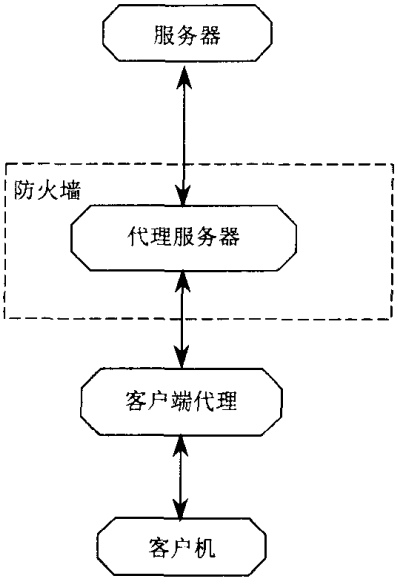


图 12.3 代理技术示意图

12.3 过滤型防火墙

包过滤通过检查每个 IP 网络包，取得包头信息，包括到达的物理网络接口、源 IP 地址、目标 IP 地址、传输层类型(TCP、UDP、ICMP)、源端口和目的端口等，根据这些信息判别是否与规则集中的某条目相匹配，并对匹配包执行规则中指定的动作(禁止或允许)。

包过滤防火墙通常还可以重置网络包地址，从而使从防火墙流出的数据包源地址不同于原始发出数据包的主机地址。这种重置机制称为网络地址转换(NAT)。通过 NAT 可以隐藏内部网络拓扑和地址表。

包过滤防火墙有一组接受和禁止规则，这些规则明确定义了哪个包将被允许通过或被禁止通过网络接口。输入包和输出包的过滤分别对应不同的规则列表。

输入包的过滤一般包括：

- 远程源地址过滤：识别 IP 包发送者的依据是数据包的源地址，因此在防火墙中检查数据包的源地址，并根据预设的规则可以允许或禁止该数据包通过；
- 本地目的地址过滤；

- 远程端口过滤；
- 本地目的端口过滤。

输出包的过滤一般包括：

- 本地源地址的过滤；
- 远程目的地址的过滤；
- 本地源端口过滤；
- 远程目的端口过滤。

一般说来，包过滤类型的防火墙具有以下优点：

(1) 其性能要优于其他类型的防火墙，因为它执行较少的计算，并且可以很容易地以硬件方式实现；

(2) 规则设置简单，通过禁止内部计算机和特定 Internet 资源的连接，单一规则即可保护整个网络；

(3) 不需对客户端计算机进行专门配置；

(4) 通过 NAT(网络地址转换)，可以对外部用户屏蔽内部 IP。

其缺点也是明显的：

(1) 无法识别应用层协议，也无法对协议子集进行约束，甚至最基本的服务，如 ftp 中的 put 和 get 命令也无法识别；

(2) 处理包内信息的能力也很有限，通常不能提供其他附加功能，如 http 的对象缓存，URL 过滤及认证等；

(3) 一般无法约束由内部主机到防火墙服务器上的信息，只能控制哪些信息可以通过，从而入侵者可能访问到防火墙主机的服务，带来安全隐患；

(4) 由于对众多网络服务的“广泛”支持所造成的复杂性，很难对规则有效性进行测试；

(5) 另外，多数包过滤因为不保存源自会话层或应用层的信息，是无状态的。

包过滤型防火墙又可分为静态包过滤型和状态检测型两种。

12.3.1　静态包过滤防火墙

静态包过滤防火墙工作在 TCP/IP 协议的 IP 层，如图 12.4 所示。依据系统事先设定好的过滤规则，检查数据流中的每个数据包。根据数据包的源地址、目的地址、所用端口号、数据的对话协议及数据包头中的各种标志位或它们的组合等因素来确定是否允许该数据包通过。具体过滤要素如下：

- 数据包协议类型：TCP、UDP、ICMP、IGMP 等；
- 源、目的 IP 地址；

- 源、目的端口：FTP、HTTP、DNS 等；
- IP 选项：源路由、记录路由等；
- TCP 选项：SYN、ACK、FIN、RST 等；
- 其他协议选项：ICMP ECHO、ICMP ECHO REPLY 等；
- 数据包流向：in 或 out；
- 数据包流经网络接口：eth0、eth1。

例子：设内部网地址为 192.168.0.0/24，防火墙内网卡 eth1 地址为 192.168.0.1，防火墙外网卡 eth0 地址为 10.11.12.13，DNS 地址为 10.11.15.4，要求允许内部网所有主机能访问外网 WWW、FTP 服务，外部网不能访问内部主机，那么，防火墙规则表可以配置如下：

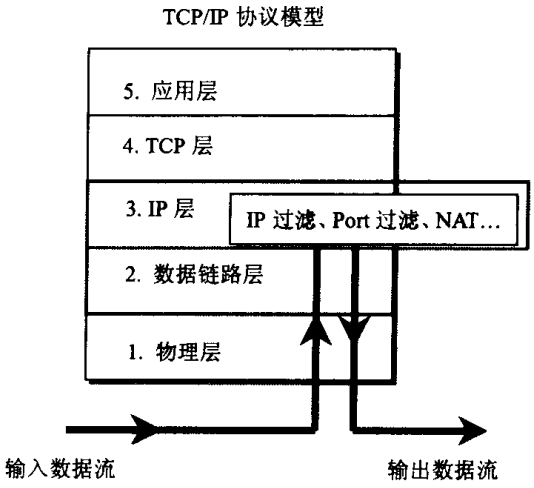


图 12.4 静态包过滤防火墙工作层示意图

```
Set internal=192.168.0.0/24
Deny ip from $internal to any in via eth0
Deny ip from not $internal to any in via eth1
Allow udp from $internal to any dns
Allow udp from any dns to $internal
Allow tcp from any to any established
Allow tcp from $internal to any www in via eth1
Allow tcp from $internal to any ftp in via eth1
Allow tcp from any ftp-data to $internal in via eth0
Deny ip from any to any
```

包过滤防火墙的优点是逻辑简单，对网络性能的影响较小，有较强的透明性。此外，它的工作与应用层无关，无须改动任何客户机和主机上的应用程序，易于安装和使用。他的弱点是：配置基于包过滤方式的防火墙，需要对 IP、TCP、UDP、ICMP 等各种协议有深入的了解，否则容易出现因配置不当带来的问题；过滤依据只有网络层和传输层的有限信息，各种安全要求不能得到充分满足；数据包的地址及端口号都在数据包的头部，不能彻底防止地址欺骗；允许外部客户和内部主机的直接连接；不提供用户的鉴别机制。

12.3.2 状态监测防火墙

静态包过滤防火墙最明显的缺陷是，为了实现期望的通信，它必须保持一些端口永久

开放，这就为潜在的攻击提供了机会。为了克服这一弱点，发展出了动态包过滤技术，它根据数据包的头信息打开或关闭端口。当一组数据包通过打开的端口到达目的地，防火墙就关闭这些端口。状态监测技术就是在动态包过滤技术的基础上发展而来，是对动态包过滤技术的增强。这种防火墙采用了一个在网关上执行网络安全策略的软件引擎，称之为监测模块。监测模块工作在链路层和网络层之间，如图 12.5 所示，对网络通信的各层实施监测分析，提取相关的通信和状态信息，并在动态连接表中进行状态及上下文信息的存储和更新，这些表被持续更新，为下一个通信检查积累数据。

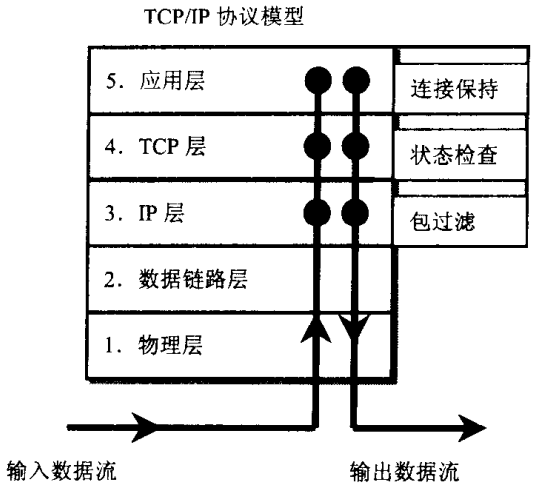


图 12.5　状态监测防火墙工作示意图

状态监视器的另一个优点是：能够为基于无连接的协议(UDP)的应用(如 DNS、WAIS、etc)及基于端口动态分配的协议(RPC)的应用(如 NFS、NIS)提供安全支持，静态的包过滤和代理网关都不支持此类应用。总之，这类防火墙的优点是减少了端口的开放时间，提供了对几乎所有服务的支持，缺点是它也允许外部客户和内部主机的直接连接，不提供用户的鉴别机制。

例如，针对前例的配置和安全要求，采用状态检测防火墙，相应的规则可如下配置：

```
Set internal=192.168.0.0/24
Deny ip from $internal to any in via eth0
Deny ip from not $internal to any in via eth1
Allow $internal access any dns by udp keep state
Allow $internal acess any www by tcp keep state
Allow $internal access any ftp by tcp keep state
Deny ip from any to any
```

12.4　代理型防火墙

代理型防火墙又可分为应用级网关(application level gateways)和电路级网关(circuit level gateways)两大类。

12.4.1 应用级网关防火墙

应用级网关防火墙通常也称为应用代理服务器，一般说的代理级防火墙就是这一种。其工作原理如图 12.6 所示。

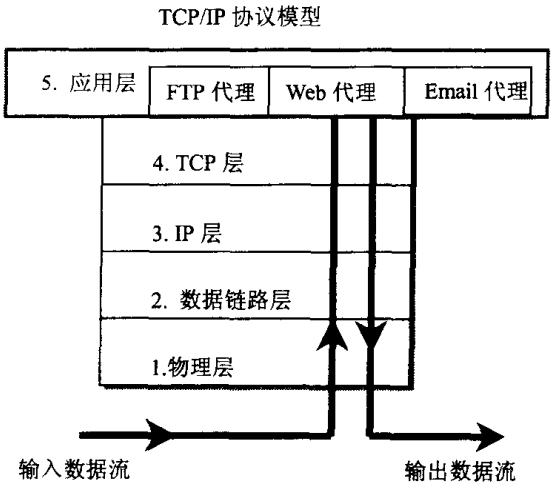


图 12.6 应用级网关防火墙工作示意图

这种防火墙通过代理(Proxy)技术参与了 TCP 连接的全过程。从内部发出的数据包经过防火墙处理，就好像源于防火墙外部网卡，从而可以达到隐藏内部网结构的作用。它的核心技术就是代理服务技术。它通常被配置为“双宿主网关”，具有两个网络接口卡，跨接内部和外部网。网关可以与两个网络通信，因此是安装传递数据软件的理想位置。这种软件就称为“代理”，通常是为其所提供的服务定制的。代理服务不允许直接连接，而是强制检查和过滤所有的网络数据包。用户并不直接与真正的服务通信，而是与代理服务器通信(用户的默认网关指向代理服务器)。各个应用代理在用户和服务之间处理所有的通信，能够对通过它的数据进行详细的审计追踪。

代理级防火墙主要具有以下优点：

(1) 代理服务可以识别并实施高层的协议，如 http 和 ftp 等；

(2) 代理服务包含通过防火墙服务器的通信信息，可以提供源于部分传输层、全部应用层和部分会话层的信息；

(3) 代理服务可以用于禁止访问特定的网络服务，并允许使用其他服务；

(4) 代理服务也能够处理数据包；

(5) 代理服务不允许外部和内部主机间直接通信，因此内部主机名对外部是不可知的。也就是说，代理服务很容易将内部地址与外部屏蔽开来；

(6) 通过提供透明服务，可以让使用代理的用户感觉在直接与外部通信；

(7) 代理服务还可以转送内部服务，也就是外部对内部的服务请求，例如，可以将 http 服务器转到另一台主机；

(8) 代理服务可以提供如前所述的屏蔽路由器不具备的附加功能；

(9) 代理服务具有良好的日志记录，从而可以建立有效的审计追踪机制，允许管理员监视可能危害到安全策略的企图。

当然，代理服务也有一些缺点，主要包括：

(1) 通常不可以在防火墙服务器上开放本机的网络服务，因为代理服务器需要侦听这些服务端口，但这同时也加强了其作为堡垒机的安全性。

(2) 代理服务会有性能上的延迟，因为流入数据需要被处理两次(应用程序和其代理)。

(3) 通常，需要为通过防火墙的每个协议加入一个新的代理，因为功能完善的代理需要很好地支持应用协议，不存在真正意义上的通用代理。从而其伸缩性和可用网络服务的数量会有所限制。一般来说，一个新的开发应用到其代理服务可能会有一定的滞后周期。也就是说，用户必须对其新型应用的代理具有耐心。

(4) 应用级防火墙一般不能提供 UDP、RPC 等特殊协议类族的代理。

(5) 代理防火墙一般需要修改或设置客户端，因此配置过程较繁琐。

(6) 由于对操作系统和应用层协议具有依赖性，代理服务具有这方面的脆弱性因素。多数包过滤防火墙对操作系统的支持机制没有太大依赖，而是通常依赖于设备驱动等。但是大多应用层防火墙需要操作系统的支持以保证其正确运行，诸如对 NDIS、TCP/IP 和标准 C 库等的支持。如果一个与安全相关的漏洞隐藏在这些库中，就可能对防火墙产生不可预料的影响。应用层防火墙有时会忽略那些底层的网络包信息。

(7) 代理通常需要附加的口令和认证，从而造成延迟，可能会给某些用户带来不便。

代理类型防火墙的最突出优点就是安全性高。由于每一个内、外网络之间的连接都要通过代理服务器的介入和转换，通过专门为特定的服务如 Http 编写的安全化应用程序进行处理，然后由防火墙本身提交请求和应答，没有给内外网络的计算机以任何直接会话的机会，从而避免了入侵者使用数据驱动类型的攻击方式入侵内部网。包过滤类型的防火墙是很难彻底避免这一漏洞的。

代理防火墙的最大缺点就是速度相对比较慢，当用户对内外网络网关的吞吐量要求比较高时，代理防火墙就会成为内外网络之间的瓶颈。在现实环境中，大部分高速网(ATM 或千兆位以太网等)之间的防火墙要考虑使用包过滤类型防火墙来满足速度的要求。

12.4.2　电路级网关防火墙

电路级网关是一个通用代理服务器，通常可以认为它工作于 OSI 互联模型的会话层或是 TCP/IP 协议的 TCP 层，如图 12.7 所示。它适用于多个协议，但它不需要识别在同一个协议栈上运行的不同应用，当然也就不需要对不同的应用设置不同的代理模块，但这种代理需要

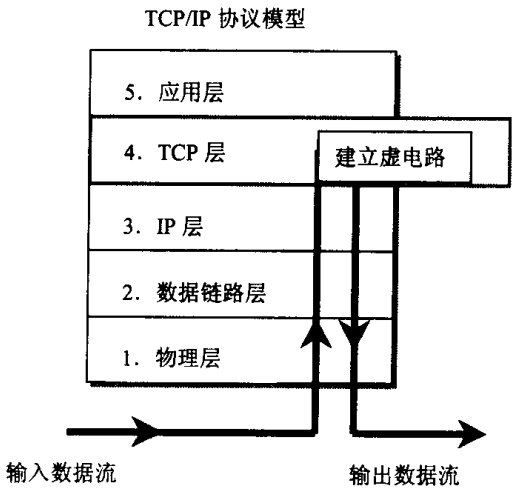


图 12.7　电路级网关防火墙工作示意图

对客户端做适当修改。它接受客户端的连接请求，代理客户端完成网络连接，建立起一个回路，对数据包起转发作用，数据包被提交给用户的应用层来处理。通过电路级网关传递的数据似乎起源于防火墙，隐藏了被保护网络的信息。

电路级网关比较通用，同应用程序网关技术相比，其优点是不需要对不同的应用设置不同的代理模块，但同样具有资源占用大、速度慢的缺点。同数据包过滤技术相比，其优点是可以在转发数据包之前完成身份认证的工作，但同样地，由于它也不分析上下文，包过滤技术中与此相关的缺点它同样具有。

电路级网关的实现典型是 Socks 协议，目前已发展到第 5 版，支持多种认证方式。

12.5 防火墙连接模式

常见的防火墙在网络中的连接模式有三种典型结构：双宿/多宿主机模式、屏蔽主机模式、屏蔽子网模式。

在介绍这几种结构前，先来了解一下堡垒主机(bastion host)的概念，堡垒主机是一种配置了安全防范措施的网络计算机，堡垒主机为网络之间的通信提供了一个阻塞点，也就是说如果没有堡垒主机，网络之间将不能相互访问。

12.5.1 双宿/多宿主机模式

双宿/多宿主机防火墙(Dual-Homed /Multi-Homed Host Firewall)又称为双宿/多宿网关防火墙，它是一种拥有两个或多个连接到不同网络的网络接口的防火墙，通常是一台装有两块或多块网卡的堡垒主机，两块或多块网卡各自与受保护网络和外部网络相连，其配置结构如图 12.8 所示。

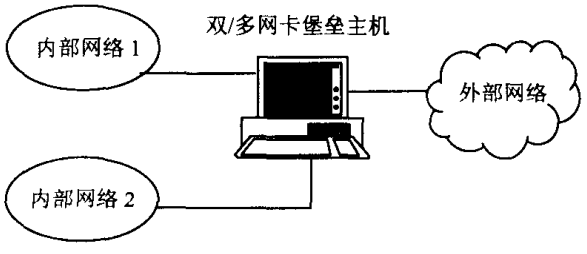


图 12.8 双宿/多宿主机模式防火墙配置

由于堡垒主机具有两个以上的网卡，可以连接两个以上的网络，所以计算机系统可以充当这些网络之间的防火墙，从一个网络到另一个网络发送的 IP 数据包必须经过双宿主机的检查。双宿主机检查通过的数据包，并根据安全策略进行处理。

双宿/多宿主机模式下，堡垒主机可以采用包过滤技术，也可以采用代理服务技术。在使用代理服务技术的双宿主机中，主机的路由功能通常是被禁止的，两个网络之间的通信通过应用层代理服务来完成。如果一旦黑客侵入堡垒主机并使其具有路由功能，防火墙将失去作用。

12.5.2　屏蔽主机模式

屏蔽主机防火墙(Screened Host Firewall)由包过滤路由器和堡垒主机组成，其配置如图 12.9 所示。在这种方式的防火墙中，堡垒主机安装在内部网络上，通常在路由器上设立过滤规则，并使这个堡垒主机成为从外部网络惟一可直接到达的主机，这确保了内部网络不受未被授权的外部用户的攻击。屏蔽主机防火墙实现了网络层和应用层的安全，因而比单独的包过滤或应用网关代理更安全。

在这一方式下，过滤路由器是否配置正确是这种防火墙安全与否的关键，如果路由表遭到破坏，堡垒主机就可能被绕过，使内部网络完全暴露。

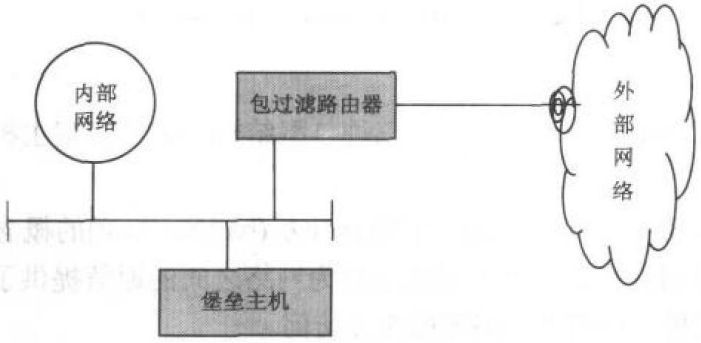


图 12.9　屏蔽主机模式防火墙配置

12.5.3　屏蔽子网模式

屏蔽子网防火墙(Screened Subnet mode)的配置如图 12.10 所示，采用了两个包过滤路

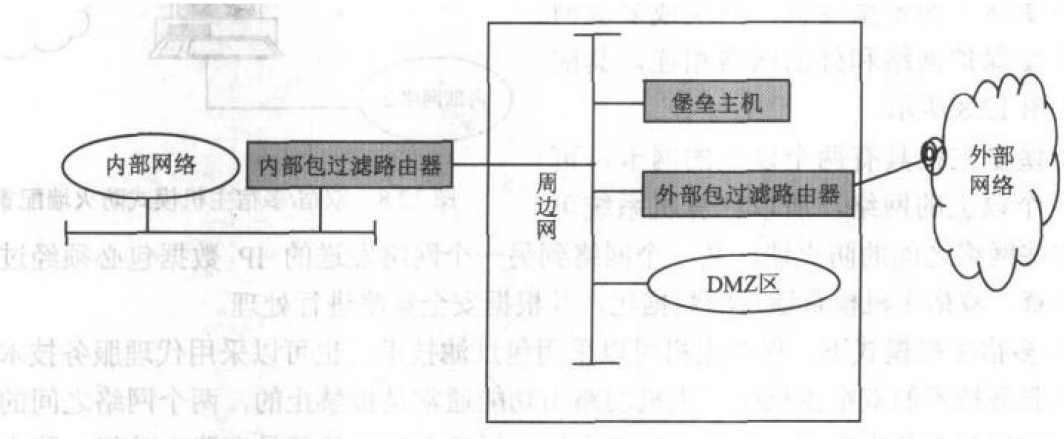


图 12.10　屏蔽子网模式防火墙配置

由器和一个堡垒主机，在内、外网络之间建立了一个被隔离的子网，定义为“非军事区(Demilitarized Zone，DMZ)”网络，有时也称做周边网(Perimeter Network)。网络管理员将堡垒主机、Web 服务器、Mail 服务器等公用服务器放在非军事区网络中。内部网络和外部网络均可访问屏蔽子网，但禁止它们穿过屏蔽子网通信。在这一配置中，即使堡垒主机被入侵者控制，内部网络仍受到内部包过滤路由器的保护。

12.6 防火墙产品的发展

防火墙技术越来越多地应用于专用网络与公用网络的互联环境之中，尤以 Internet 网络最甚。Internet 的迅猛发展，使得防火墙产品在短短几年内异军突起，很快形成了一个产业。

在防火墙产品的开发中，广泛采用了网络拓扑技术、计算机操作系统技术、路由技术、加密技术、访问控制技术、安全审计技术等，不同阶段的防火墙采用的技术也有区别，其发展可分为四个阶段：

第一阶段：基于路由器的防火墙。

由于多数路由器本身就包含有分组过滤的功能，故网络访问控制功能可通过路由控制来实现，从而使具有分组过滤功能的路由器成为第一代防火墙产品。

第一代防火墙产品的特点是：

- 利用路由器本身对分组的解析，以访问控制表(access list)方式实现对分组的过滤；过滤判决的依据可以是：地址、端口号及其他网络特征；
- 只有分组过滤的功能，且防火墙与路由器是一体的，对安全要求低的网络采用路由器附带防火墙功能的方法，对安全性要求高的网络则可单独使用一台路由器作防火墙。

第一代防火墙产品的不足之处主要有：

- 路由协议十分灵活，本身具有安全漏洞，外部网络要探寻内部网络十分容易。
- 路由器上的分组过滤规则的设置和配置存在安全隐患。对路由器中过滤规则的设置和配置十分复杂，它涉及到规则的逻辑一致性、作用端口的有效性和规则集的正确性，一般的网络系统管理员难于胜任，加之一旦出现新的协议，管理员就得加上更多的规则去限制，这往往会带来很多错误。
- 路由器防火墙的最大隐患是：攻击者可以“假冒”地址，由于信息在网络上是以明文传送的，黑客可以在网络上伪造假的路由信息以欺骗防火墙。
- 路由器防火墙的本质性缺陷是：由于路由器的主要功能是为网络访问提供动态的、灵活的路由，而防火墙则要对访问行为实施静态的、固定的控制，这是一对难以调和的矛

盾，防火墙的规则设置会大大降低路由器的性能。

可以说：基于路由器的防火墙只是网络安全的一种应急措施，用这种权宜之计去对付黑客的攻击是十分脆弱的。

第二阶段：用户化的防火墙。

为了弥补路由器防火墙的不足，很多大型用户纷纷要求以专门开发的防火墙系统来保护自己的网络，从而推动了用户化防火墙的出现。

作为第二代防火墙产品，用户化的防火墙工具具有以下特征：

- 将过滤功能从路由器中独立出来，并加上审计和报警功能；
- 针对用户需求，提供模块化的软件包；
- 软件可通过网络发送，用户可自己动手构造防火墙；
- 与第一代防火墙相比，安全性提高了。

由于是纯软件产品，第二代防火墙产品无论在实现还是在维护上都对系统管理员提出了更高的要求，并带来以下一些问题：

- 配置和维护过程复杂、费时；
- 对用户的技术要求高；
- 全软件实现，安全性和处理速度均有局限；
- 实践表明，使用中出现差错的情况很多。

第三阶段：建立在通用操作系统上的防火墙。

基于软件的防火墙在销售、使用和维护上的问题迫使防火墙开发商很快推出了建立在通用操作系统上的商用防火墙产品，近年来在市场上广泛使用的就是这一代产品，它具有以下特点：

- 是批量上市的专用防火墙产品；
- 包括分组过滤或者借用路由器的分组过滤功能；
- 装有专用的代理系统，监控所有协议的数据和指令；
- 保护用户编程空间和用户可配置内核参数的设置；
- 安全性和速度大为提高。

第三代防火墙有以纯软件实现的，也有以硬件方式实现的，都已得到广大用户的认可。但随着安全需求的变化和使用时间的推延，仍表现出不少问题，比如：

- 作为基础的操作系统及其内核往往不为防火墙管理者所知，由于源代码的保密，其安全性无从保证。
- 由于大多数防火墙厂商并非通用操作系统的厂商，通用操作系统厂商不会对操作系统的安全性负责；
- 从本质上看，第三代防火墙既要防止来自外部网络的攻击，还要防止来自操作系统

厂商的攻击。

第四阶段：具有安全操作系统的防火墙。

防火墙技术和产品随着网络攻击和安全防护手段的发展而演进，到 1997 年初，具有安全操作系统的防火墙产品面市，使防火墙产品步入了第四个发展阶段。

具有安全操作系统的防火墙本身就是一个操作系统，因而在安全性上较第三代防火墙有质的提高。获得安全操作系统的办法有两种：一种是通过许可证方式获得操作系统的源码；另一种是通过固化操作系统内核来提高可靠性，由此建立的防火墙系统具有以下特点：

- 防火墙厂商具有操作系统的源代码，并可实现安全内核；
- 对安全内核实现加固处理：去掉不必要的系统特性，加上内核特性，强化安全保护；
- 对每个服务器、子系统都作了安全处理，一旦黑客攻破了一个服务器，它将会被隔离在此服务器内，不会对网络的其他部分构成威胁；
- 在功能上包括了分组过滤、应用网关、电路级网关，且具有加密与鉴别功能；
- 透明性好，易于使用。

在第四代防火墙产品的设计与开发中，安全内核、代理系统、多级过滤、安全服务器和鉴别与加密是关键所在。

12.7 防火墙的局限性

需要指出的是，防火墙是网络安全的重要一环，但并非全部，认为所有的网络安全问题都可以通过简单地配置防火墙来解决是不切实际的。

防火墙有自身的缺陷和不足，主要有：

(1) 为了提高安全性，限制或关闭了一些有用但存在安全缺陷的网络服务，给用户带来使用的不便。

(2) 目前防火墙对于来自网络内部攻击的防范能力有限。

(3) 防火墙不能防范不经过防火墙的攻击，如内部网用户通过拨号直接进入 Internet。

(4) 防火墙对用户不完全透明，可能带来传输延迟、瓶颈及单点失效。

(5) 防火墙也不能完全防止受病毒感染的文件或软件的传输，由于病毒的种类繁多，如果要在防火墙完成对所有病毒代码的检查，防火墙的效率就会降到不能忍受的程度。

(6) 防火墙不能有效地防范数据驱动式攻击。

在实际应用中，可以根据对安全的需求选择合适的防火墙技术及相应的配置方式。防火墙最早只是提供了一种简单的数据流访问控制策略，随着各项信息技术的迅速发展，新的软件技术、信息安全思想和技术不断地应用于防火墙的开发上，比如现代密码技术、一

次口令系统、智能卡等。随着各种新的安全问题的出现，防火墙的概念和内涵也随着需求的发展而不断丰富，比如支持 VPN 构建、远程集中管理、对内容的过滤、阻止脚本程序、具有一定的病毒检测功能等。如今个人防火墙已得到了广泛的应用，操作系统大多也提供了许多在传统意义上属于防火墙的功能，作为构建安全网络环境的第一道屏障，防火墙还是目前比较有效的措施。

习 题 十 二

1. 简述防火墙的功能和分类。

2. 静态包过滤防火墙和状态检测防火墙有何区别？如何实现状态检测？

3. 状态检测防火墙的优点是什么？为什么？

4. 如何实现 M-N 地址转换？

5. 应用网关防火墙是如何实现的？与包过滤防火墙比较有什么优缺点？

6. 双宿连接和屏蔽子网连接各有何优缺点？

7. 配置一个防火墙的规则，实现允许 Email 通过，拒绝来自 162.105.0.0 网段的 FTP 请求，允许除主机 202.112.9.7 外的 WWW 服务。

第十三章　网络安全协议

13.1　安全协议概述

在信息网络中，可以在 ISO 七层协议中的任何一层采取安全措施，图 13.1 给出了每一层可以利用的安全机制。大部分安全措施都采用特定的协议来实现，如在网络层加密和认证采用 IPSec 协议，在传输层加密和认证采用 SSL 协议等。

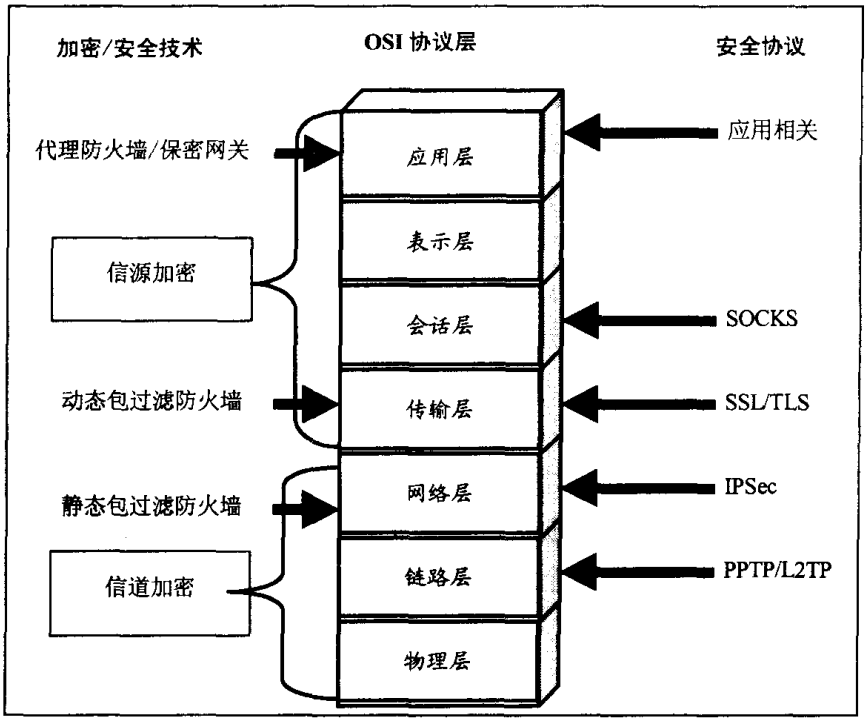


图 13.1　OSI 七层协议与信息安全

安全协议本质上是关于某种应用的一系列规定，包括功能、参数、格式、模式等，通信各方只有共同遵守协议，才能互操作。本章先简要介绍常见的安全协议，然后重点介绍安全传输协议 IPSec 和 SSL。

13.1.1　应用层安全协议

1. 安全 Shell (SSH)

在实际工作中，SSH 协议通常是替代 TELNET 协议、RSH 协议来使用的。它类似于 TELNET 协议，允许客户机通过网络连接到远程服务器并运行该服务器上的应用程序，被广泛用于系统管理中。该协议可以加密客户机和服务器之间的数据流，这样可以避免 TELNET 协议中口令被窃听的问题。该协议还支持多种不同的认证方式。

除了支持终端类应用，SSH 协议还可用于加密包括 FTP 数据外的多种情况。

2. SET

SET(安全电子交易)协议是电子商务中用于安全电子支付最典型的代表协议。它是由 MasterCard 和 VISA 制定的标准，这一标准的开发得到了 IBM、Microsoft、Netscape、SAIC、Terisa 和 Verisign 的投资以及其他信用卡和收费卡发行商的支持。

SET 是在一些早期协议(如 MasterCard 的 SEPP、VISA 和 Microsoft 的 STT)的基础上整合而成的，它定义了交易数据在卡用户、商家、发卡行、收单行之间的流通过程，以及支持这些交易的各种安全功能(数字签名、Hash 算法、加密等)。

为了进一步加强安全，SET 以两组密钥对分别用于加密和签名。SET 不希望商家得到顾客的账户信息，同时也不希望银行了解到交易内容，但又要求能对每一笔单独的交易进行授权。SET 通过双签名(dual signature)机制将订购信息和账户信息链在一起签名，巧妙地解决了这一矛盾。

尽管优点很多，SET 协议也存在不足之处：(1)它是目前最为复杂的保密协议之一，整个规范有 3000 行以上的 ASN.1 语法定义，交易处理步骤很多，在不同实现方式之间的互操作性也是一大问题；(2)每个交易涉及到 6 次 RSA 操作，处理速度很慢。

有关 SET 的介绍详见“应用安全”一章。

3. S-HTTP

WWW 是在超文本传输协议(HTTP)基础上建立起来的，但 HTTP 中不包含安全性机制，因此提出了安全 HTTP(即 S-HTTP)协议，它是对 HTTP 进行的扩展，描述了一种使用标准加密工具来传送 HTTP 数据的机制。S-HTTP 是一个非常完整的实现，几乎包括了在今后相当长一段时间内可能需要的安全 HTTP 访问应该具有的全部特征。它工作在应用层，同时对 HTML 进行了扩展，服务器方可以在需要进行安全保护的文档中加入加密选项，控制对该文档的访问以及协商加密、解密、签名算法等。但由于缺乏厂商支持，S-HTTP 协

议现在已经几乎不再使用。

4. PGP

PGP(Pretty Good Privacy)主要用于安全电子邮件，它可以对通过网络进行传输的数据创建和检验数字签名、加密、解密以及压缩。除电子邮件外，PGP 还被广泛用于网络的其他功能之中。PGP 的一大特点是源代码免费使用、完全公开。

有关 PGP 的介绍详见“应用安全”一章。

5. S/MIME

S/MIME 在 MIME(多用途 Internet 邮件扩展)规范中加入了获得安全性的一种方法，提供了用户和认证方的形式化定义，支持邮件的签名和加密。

有关 S/MIME 的介绍详见“应用安全”一章。

13.1.2 传输层安全协议

1. SSL

SSL(安全套接字层，Secure Socket Layer)是 Netscape 开发的安全协议，它工作在传输层，独立于上层应用，为应用提供一个安全的点-点通信隧道。SSL 机制由协商过程和通信过程组成，协商过程用于确定加密机制、加密算法、交换会话密钥服务器认证以及可选的客户端认证，通信过程秘密传送上层数据。虽然现在 SSL 主要用于支持 HTTP 服务，但从理论上讲，它可以支持任何应用层协议，如 Telnet、FTP 等。

2. PCT

PCT(私密通信技术，Private Communication Technology)是 Microsoft 开发的传输层安全协议，它与 SSL 有很多相似之处。现在 PCT 已经同 SSL 协议合并为 TLS(传输层安全)协议，只是习惯上仍然把 TLS 协议称为 SSL 协议。

13.1.3 网络层安全协议

为开发在网络层保护 IP 数据的方法，IETF 成立了 IP 安全协议工作组(IPSec)，定义了一系列在 IP 层对数据进行加密的协议，包括：

- IP 验证头(AH)协议；
- IP 封装安全载荷协议(ESP)；
- Internet 密钥管理协议(IKMP)。

在本章中，重点讨论网络层安全协议 IPSec 和传输层安全协议 SSL。

13.2　IPSec

13.2.1　IPSec 综述

在传统的TCP/IP协议中，IP数据包在传输过程中并没有考虑安全需求，人们很容易伪造IP包的地址，修改其内容，重播以前的包并在传输途中拦截并查看包的内容。因此，不能保证收到的IP数据包来自正确的发送方，不能肯定数据是发送方所发送的原始数据，也不能确认原始数据在传输中途未被窃听。

IPSec 的目的，就是要有效地保护 IP 数据包的安全。它提供了一种标准的、强大的以及包容广泛的机制，为 IP 及上层协议(如 UDP 和 TCP)提供安全保证；并定义了一套默认的、强制实施的算法，以确保不同的实施方案相互间可以共通，而且很方便扩展。IPSec 为保障 IP 数据包的安全，定义了一个特殊的方法，规定了要保护什么数据，如何保护以及将该数据发给何人。IPSec 可保障主机之间、安全网关(如路由器或防火墙)之间或主机与安全网关之间的数据包的安全。由于受 IPSec 保护的数据包本身只是另一种形式的 IP 包，所以完全可以嵌套提供安全服务，同时在主机之间提供端到端的验证，并通过一个安全通道，将那些受 IPSec 保护的数据传送出去(通道本身也通过 IPSec 受到安全网关的保护)。

IPSec 是一个工业标准网络安全协议，为 IP 网络通信提供透明的安全服务，保护 TCP/IP 通信免遭窃听和篡改，可以有效抵御网络攻击，同时保持易用性。IPSec 有两个基本目标：保护 IP 数据包安全；为抵御网络攻击提供防护措施。IPSec 结合密码保护服务、安全协议组和动态密钥管理，三者共同实现这两个目标，它不仅能为局域网与拨号用户、域、网站、远程站点以及 Extranet 之间的通信提供有效且灵活的保护，而且还能用来筛选特定数据流。IPSec 基于一种端对端的安全模式。这种模式有一个基本前提假设，就是假定数据通信的传输媒介是不安全的，因此通信数据必须经过加密，而掌握加解密方法的只有数据流的发送端和接收端，两者各自负责相应的数据加解密处理，而网络中其他只负责转发数据的路由器或主机无须支持 IPSec。

IPSec 提供三种不同的形式来保护通过公有或私有 IP 网络传送的私有数据。

(1) 认证：通过认证可以确定所接受的数据与所发送的数据是否一致，同时可以确定申请发送者在实际上是真实的，还是伪装的发送者；

(2) 数据完整验证：通过验证，保证数据在从原发地到目的地的传送过程中没有发生任何无法检测的丢失与改变；

(3) 保密：使相应的接收者能获取发送的真正内容，而无关的接收者无法获知数据的真正内容。

IPSec 由三个基本要素来提供以上三种保护形式：验证头(AH)、封装安全载荷(ESP)和互联网密钥管理协议(IKMP)。认证协议头和安全加载封装可以通过分开或组合使用来实现所预期的保护目标。

13.2.2 IPSec 结构

RFC2401给出了IPSec的基本结构定义，并为所有具体的实施方案建立了基础。它定义了IPSec提供的安全服务，并说明了它们如何使用及在哪里使用、数据包如何构建及处理以及IPSec处理同策略之间如何协调等问题。IPSec协议既可用来保护一个完整的IP载荷，亦可用来保护某个IP载荷的上层协议。这两方面的保护分别由IPSec两种不同的“模式”来提供，即传送模式和通道模式，如图13.2所示。其中，传送模式用来保护上层协议；而通道模式用来保护整个IP数据报。在传送模式中，IP头与上层协议头之间需插入一个特殊的IPSec头；而在通道模式中，要保护的整个IP包都需封装到另一个IP数据报里，同时在外部与内部IP头之间插入一个IPSec头。两种IPSec协议(AH和ESP)均能以传送模式或通道模式工作。对传送模式所保护的数据包而言，其通信终点必须是一个加密的终点，这是由构建方法决定的。有时可用通道模式来取代传送模式，并由安全网关使用，来保护与其他联网实体(比如一个虚拟专用网络)有关的安全服务。在后一种情况下，通信终点是由受保护的内部IP头指定的地点，而加密终点则是那些由外部IP头指定的地点。在IPSec处理结束后，安全网关会剥离出内部IP包，再将该包转发到它最终的目的地。

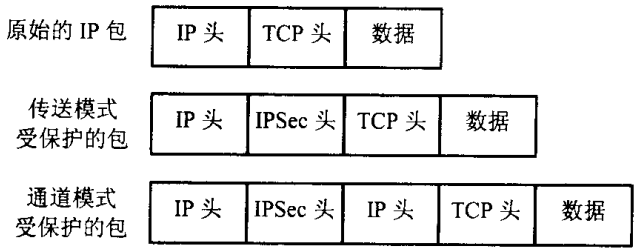


图 13.2 位于传送模式和通道模式下的数据包

IPSec既可在终端系统上实现，也可在安全网关上实现(如路由器和防火墙)。要正确封装和提取IPSec数据包，必须解决如何保护通信数据、保护什么样的通信数据以及由谁来实行保护等问题。这种构建方案称为“安全关联(Security Association ， SA)”。IPSec的SA是单向进行的。也就是说，它仅在一个方向上定义安全服务，要么对通信实体收到的包进行“进入”保护，要么对实体外发的包进行“外出”保护。具体采用什么方式，要由三方面

的因素决定：第一个是“安全参数索引(SPI)”，该索引存在于IPSec协议头内；第二个是IPSec协议值；第三个是其应用SA的目标地址，它同时决定了方向。通常，SA以成对的形式存在，分别指向一个方向。SA驻留在“安全关联数据库(SADB)”内，既可人工创建，也可动态创建。若用人工方式创建，SA便没有“存活时间”，除非再用人工方式将其删除，否则便会一直存在下去；若用动态方式创建，则SA有一个存活时间与其关联在一起，这个存活时间通常是由密钥管理协议在IPSec通信双方之间协商而确立的。存活时间(TTL)非常重要，因为受一个密钥保护的通信量(或者类似地，一个密钥保持活动以及使用的时间)必须谨慎管理。若超时使用一个密钥，会为攻击者入侵系统提供更多的机会。IPSec的基本架构定义了用户设定自己的安全策略的精度。这样一来，便可简单地为某些通信设置某一级的基本安全措施；而对其他通信谨慎对待，为其应用完全不同的安全级别。举例来说，我们可在一个安全网关上制订IPSec策略，对在其本地保护的子网与远程网关的子网间通信的所有数据全部采用DES加密，并用HMAC-MD5进行验证；另外，从远程子网发给一个邮件服务器的所有Telnet数据均用3DES进行加密，同时用HMAC-SHA进行验证；最后，对于需要加密的、发给另一个服务器的所有Web通信数据，则用IDEA满足其加密要求，同时用HMAC-RIPEMD进行验证。IPSec策略由“安全策略数据库(Security Policy Database ，SPD)”维护。在SPD这个数据库中，每个条目都定义了要保护什么通信、怎样保护它以及和谁共享这种保护。对进入或离开IP堆栈的每个包，都必须检索SPD数据库，调查可能的安全应用。对一个SPD条目来说，它可能定义了下述几种行为：丢弃、绕过以及应用。其中，“丢弃”表示不让这个包进入或外出；“绕过”表示不对一个外出的包应用安全服务，也不指望一个进入的包进行保密处理；而“应用”是指对外出的包应用安全服务，同时要求进入的包已应用了安全服务。定义了“应用”行为的SPD条目，均会指向一个或一套SA，表示要将其应用于数据包。

IPSec通信到IPSec策略的映射关系是由“选择符(Selector)”来建立的。选择符标识通信的一部分组件，它的定义可以是粗略的，也可以非常细致。IPSec选择符包括目标IP地址、源IP地址、名字、上层协议、源和目标端口以及一个数据敏感级(假如也为数据流的安全提供了一个IPSec系统)。这些选择符的值可能是特定的条目、一个范围或者是“不透明”的。在策略规范中，选择符之所以会出现“不透明”的情况，是由于在某时刻，相关的信息不能提供给系统。举个例子来说，假定一个安全网关和另一个安全网关建立了IPSec通道，它可指定在该通道内传输的(部分)数据是网关背后的两个主机之间的IPSec通信。在这种情况下，两个网关都不能访问上层协议或端口，因为它们均被终端主机进行了加密。“不透明”亦可作为一个通配符使用，表明选择符可为任意值。假定某个SPD条目将行为定义为“应用”，但并不指向SADB数据库内已有的任何一个SA，那么在进行任何实际的通信之前，必须先创建SA。如果这个规则用于自外入内的“进入(Inbound)”通信，而且SA尚不存在，

则按照IPSec基本架构的规定，数据包必须丢弃。假如该规则用于自内向外的“外出(Outbound)”通信，则通过Internet 密钥交换，便可动态地创建SA。

IPSec结构定义了SPD和SADB这两种数据库之间如何沟通，这是通过IPSec处理功能——封装与拆封来实现的。此外，它还定义了不同的IPSec实施方案如何共存。然而，它却没有定义基本IPSec协议的运作方式。这方面的信息包含在另外两个不同的文件中，一个定义了“封装安全载荷(ESP, RFC2406)”，另一个对“验证头(AH, RFC2401)”进行了说明。两种IPSec协议均提供抗重播服务(Anti-Replay)。

为了抵抗重放攻击，IPSec数据包专门使用了一个序列号，以及一个“滑动”的接收窗口。每个IPSec头内，都包含了一个惟一且单调递增的序列号。创建好一个SA后，序列号便会初始化为零，并在进行IPSec输出处理前，令这个值递增。新的SA必须在序列号回归为零之前创建，由于序列号的长度为32位，所以序号值最大为2^{32}，即4G位。接收窗口的大小可为大于32的任何值，一般推荐为64。窗口最左端对应于窗口起始位置的序列号，而最右端对应于将来的第“窗口长度”个数据包。接收到的数据包必须是新的，且必须落在窗口内部或靠在窗口右侧，否则，便应丢弃。那么，如何判断一个数据库是“新”的呢？只要它在窗口内从未出现过，便可认为是新的。假如收到的一个数据包在窗口右侧，如未能通过真实性测试，便会将其丢弃；如通过了真实性检查，窗口便会向右移动，将其包含进来。注意数据包的接收顺序可能被打乱，但不妨碍正确处理。还要注意，那些接收迟的数据包(在一个有效的数据包之后接收，但其序列号大于窗口的长度的数据包)会被丢弃！重播窗口的结构如图13.3所示。尽管由于长度仅为16位，是不符合规定的，但如仅用于演示，它还是非常适合的。图13.3窗口最左端的序列号为N，最右端的序列号自然为N+15 。编号为N、N+7、N+9、N+16和N+18以及在此之后的数据包尚未收到(以阴影表示)。

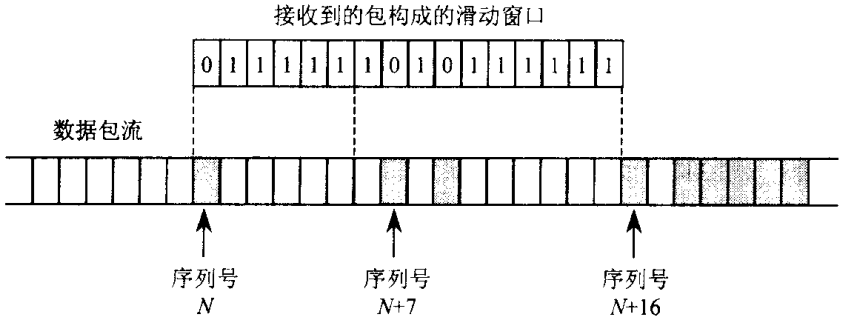


图 13.3 一个 16 位的滑动重播窗口

如果最近收到的数据包N+17通过了真实性检查，窗口便会向右滑动一个位置，使窗口左侧变成N+2，右侧变成N +17。这样数据包N就变为靠在滑动接收窗口的左侧，以致造

成数据包 N 无可挽回地被丢弃。但是，倘若包N+23 已接收到，而且通过了真实性检查，N+7这个包仍会被收到。要注意的是，除非造成窗口向前滑动的那个包通过了真实性检查，否则窗口是不会前进的。不然，攻击者便可生成伪造的包，为其植入很大的序列号，令窗口错误移至有效序列号的范围之外，造成有效数据包错误地丢失。

13.2.3　封装安全载荷(ESP)

ESP(Encapsulating Security Payload，封装安全载荷)作为基于IPSec的一种协议，可用于确保IP数据包的机密性、完整性以及对数据源的身份验证。此外，它也要负责抵抗重播攻击。具体做法是在IP头(以及任何IP选项)之后，在要保护的数据之前，插入一个新头，亦即ESP头。受保护的数据可以是一个上层协议，或者是整个IP数据包。最后，还要在最后追加一个ESP尾。ESP是一种新的IP协议，对ESP数据包的标识是通过IP头的协议字段来进行的。假如它的值为50，就表明这是一个ESP包，而紧接在IP头后面的是一个ESP头。

如图13.4所示，RFC2406文件中对ESP进行了详细的定义。此外，RFC1827还定义了一个早期版本的ESP，该版本的ESP没有提供对数据完整性的支持，IPSec工作组强烈反对继续使用它。目前，已有几套以RFC1827为基础的具体实施方案，但随着发展都可能被最新的ESP定义所取代。

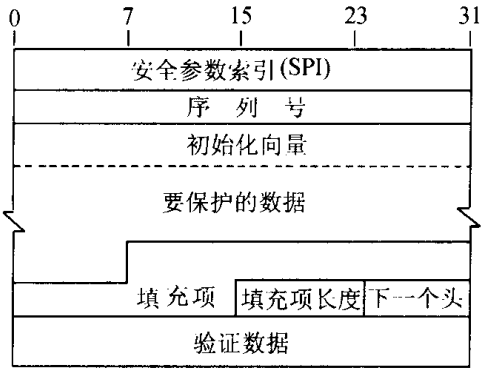


图 13.4　ESP 头

由于ESP同时提供了机密性和身份验证机制，所以在其SA中必须定义两套算法：用来确保机密性的算法称为加密器(Cipher)，而负责身份验证的算法称为验证器(Authenticator)。每个ESP的SA都至少有一个加密器和一个验证器。亦可定义NULL加密器或NULL验证器，分别表示ESP不做加密或不做验证。但在单独一个ESP SA内，不能同时定义一个NULL加密器和一个NULL验证器。这样做不仅为系统带来了无谓的负担，也毫无安全保证可言。特别强调，假如用一种安全协议来做不安全的事情，甚至比一开始就不用安全协议还要糟！因为这可能造成安全的假象，使人疏忽大意。ESP具有安全保密特性，所以千万不要以明显不安全的形式使用。ESP头并未加密，但ESP尾的一部分却进行了加密。由于采用了SPI，同时这个包的IP头还附有目标IP地址，所以为标识一个SA，必须采用明文形式。此外，序列号和验证数据也必须是明文，不可加密。这是由ESP包的指定处理顺序所决定的：首先查验序列号，然后查验数据的完整性，最后对数据进行解密。由于解密放在最后一步进行，所以序列号和验证数据自然要采取明文形式。

随ESP使用的所有加密算法必须以CBC模式工作。CBC要求加密的数据量刚好是加密

算法的块长度的整数倍。进行加密时，可在数据尾填充适当的数据，来满足这项要求。随后，填充数据会成为包密文的一部分，并在完成了IPSec处理后，由接收端予以删除。假如数据已经是加密算法的块长度的一个整数倍，便不需增加填充内容。按照标准规定，每一个IPSec实现必须支持DES加密算法。

CBC模式中的加密算法也要求有一个初始矢量(IV)来启动加密过程。这个IV包含在载荷字段内，通常是第一批字节。然而，最终还是要由特定的算法规范来做出决定，定义IV包含在什么地方，同时定义IV的大小。对DES-CBC来说，IV是受保护数据字段的头8个字节。

如前所述，ESP既有头，也有尾，头和尾之间封装了要保护的数据，如图13.5所示。其中，头部包含了SPI和序列号；而尾部则包含了填充数据(如果有的话)、与填充数据的长度有关的一个指示符、ESP后的数据采用的协议以及相关的验证数据。验证数据的长度取决于采用的验证算法。按照标准规定，此时需采用恰当的实施方案，以同时提供对HMAC-MD5 和HMAC-SHA这两种“验证器”的正确支持，输出数据的长度是96位。然而，这两个MAC会产生长度不同的摘要。其中，HMAC-MD5产生一个128位的摘要，而HMAC-SHA产生一个160位的摘要。这并没有什么不妥，因为只有摘要的高96位才被用做ESP的验证数据。之所以决定使用96位，是由于它能与IPv6很好地协调。至于将MAC的输出截去一部分是否安全，目前还颇有争议。大多数人认为，这种做法并不存在先天性的安全问题。

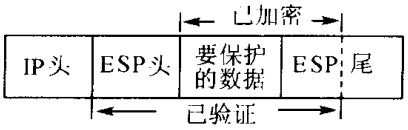


图 13.5 一个受 ESP 保护的 IP 包

目前，有一份文件用来描述如何将DES-CBC作为ESP的加密算法使用；另外两份文件则描述如何利用对HMAC-MD5 和HMAC-SHA输出的截取，来实现对ESP的验证。其他加密算法文档包括Blowfish-CBC、CAST-CBC以及3DES-CBC(均可选为最终的实施方案)。

13.2.4 验证头(AH)

与ESP类似，AH也提供了数据完整性、数据源验证以及抗重播攻击的能力，但不能以此保证数据的机密性。因此AH头比ESP简单得多，它只有一个头，而非头、尾皆有。除此以外，AH头内的所有字段都是一目了然的。

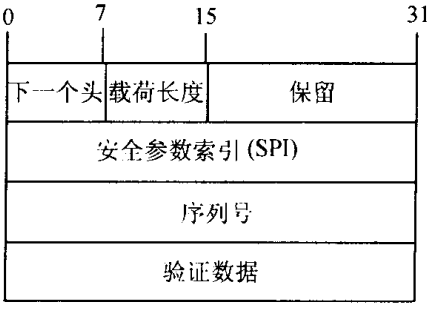


图 13.6 AH 头

RFC2402定义了最新版本的AH，如图13.6所示。而RFC1826定义的是AH的一个老版本，现已明确不再推荐对它提供支持。此外，自RFC1826问世以来，还出现了一些对特性和概念的新的澄清，

它们都已加入新文件。例如，抗重播攻击现已成为规范不可分割的一部分，同时增加了在通道模式中使用AH的定义。RFC1826和RFC1827一样，也存在着几种具体的实施方案。随着新的IPSec RFC出台，这些不再赞成使用的老方案也被新方案所取代。

AH头和ESP头相似，也包含：一个SPI，为要处理的包定位SA；一个序列号，提供对重播攻击的抵抗；另外还有一个验证数据字段，包含着用于保留数据包的加密MAC的摘要。和ESP相同，定义了如何将MAC应用于ESP的同样两份文件——定义HMAC-MD5-96的RFC2403以及定义HMAC-SHA-96的RFC2404——也定义了如何将它们应用于AH。由于AH并不通过CBC模式下的一个对称加密算法来提供对数据机密性的支持，所以没有必要强行填充数据来满足长度要求。有些MAC可能需要填充(如DES-CBC-MAC)，但具体的填充技术资料，则留待对MAC本身进行描述的文件加以定义。AH的验证范围与ESP有区别，AH验证的是IPSec包的外层IP头。因此，AH文件对IPv4及IPv6那些不定的字段进行了说明，比如，在包从源传递到目的地的过程中，可能会由路由器进行修改。在对验证数据进行计算之前，这些字段首先必须置零。AH文件还分别定义了AH头的格式、采用传送模式或通道模式时头的位置、对输出数据的处理等一些相关信息，比如对分段及重新装配的控制等，一个受AH保护的IP包如图13.7所示。

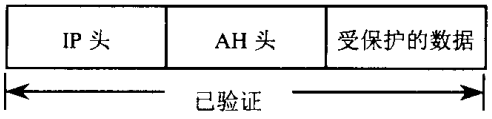


图 13.7　一个受 AH 保护的 IP 包

13.2.5　Internet 密钥交换

在IPSec中，用“安全关联”来定义对一个特定的IP包进行的处理。对一个外出的数据包而言，它会“命中”SPD，而且SPD条目指向一个或多个SA。如果没有SA来指示SPD的策略，则必须自行创建一个，此时便需用到Internet密钥交换(IKE)了。IKE惟一的用途，就是在IPSec通信双方之间建立起共享安全参数及验证过的密钥(亦即建立“安全关联”关系)。IKE协议是Oakley和SKEME协议的混合，并在由ISAKMP(Internet安全关联和密钥管理协议，Internet Security Association and Key Management Protocol) 规定的框架内运作。ISAKMP定义了包括数据包格式、重发计数器以及消息构建要求等在内的整套加密通信语言。Oakley和SKEME定义了通信双方建立一个共享的验证密钥所必须采取的步骤。IKE利用ISAKMP语言对这些步骤以及其他信息交换措施进行表述。IKE实际上是一种常规用途的安全交换协议，可用于磋商策略，以及建立验证加密材料，适用于多方面的需求，如SNMPv3 、OSPFv2等。

IKE采用的规范在RFC2407(“解释域”，Domain of Interpretation ，DOI)”中制订,DOI规定了IKE具体如何与IPSec SA进行协商。如果其他协议要用到IKE，每种协议都要定义各自的DOI。同IPSec相似，IKE也采用了“安全关联”的概念，但IKE SA的物理构建方式与IPSec

SA并不相同。IKE SA定义了双方的通信形式。如用哪种算法来加密IKE通信，怎样对远程通信方的身份进行验证，等等。

构建好IKE的SA后，便可用IKE SA在通信双方之间提供任何数量的IPSec SA 。假如一个SPD条目含有一个NULL SADB 指针，那么IPSec方案采取的行动就是将来自SPD的安全要求传达给IKE，并指示它建立IPSec SA 。

由IKE建立的IPSec SA 有时也会为密钥带来“完美向前保密(PFS)”特性，而且如果愿意，亦可使通信对方的身份具有同样的特性。通过一次IKE密钥交换，可创建多对IPSec SA 。正是由于提供了多样性的选择，IKE才会一方面具有广泛的包容面，另一方面具有高度的复杂性。IKE协议由打算执行IPSec的每一方执行；IKE通信的另一方也是IPSec通信的另一方。换言之，假如想随远程通信实体一同创建IPSec SA，必须将IKE传达给那个实体，而非传达给一个不同的IKE实体。协议本身具有“请求响应”特性，要求同时存在一个“发起者(Initiator)”和一个“响应者(Responder)”。其中，发起者要从IPSec那里接收指令，以便建立一些SA，这是某个外出的数据包同一个SPD条目相符所产生的结果，它负责为响应者对协议进行初始化。IPSec的SPD会向IKE指出要建立的是什么，但却不会指示IKE怎样做。至于IKE以什么方式来建立IPSec SA，取决于它自己的策略设定。IKE以“保护套件(Protection suite)”的形式来定义策略。每个保护套件至少都需要定义采用的加密算法、散列算法、Diffie-Hellman组以及验证方法。IKE的策略数据库则列出了所有保护套件(按各个参数的顺序)。

只有当通信双方决定了一个特定的策略套件后，它们以后的通信才能根据它进行，因此两个IKE通信实体第一步所需要做的就是某种形式的协商。双方需要建立一个共享的秘密，这一点尽管有多种方式可以做到，但实际上IKE只使用一个Diffie-Hellman交换，这是不可协商改变的。但是，其中采用的具体参数却是可以商量的。IKE从Oakley文档中借用了五个组。其中三个是传统交换方法，是对一个大质数进行乘幂模数运算；另外两个则是椭圆曲线组。

Diffie-Hellman交换以及共享秘密的建立是IKE协议的第二步。Diffie-Hellman交换完成后，通信双方已建立了一个共享的秘密，只是尚未验证通过。它们可利用该秘密(在IKE的情况下，则使用自它衍生的一个秘密)来保护双方的通信。但在这种情况下，却不能保证远程通信方事实上的确是自己所信任的。因此，IKE交换的下一个步骤便是对Diffie-Hellman共享秘密进行验证，同时还要对IKE SA本身进行验证。IKE定义了五种验证方法：

(1) 预共享密钥；
(2) 数字签名(使用数字签名标准，即 DSS)；
(3) 数字签名(使用 RSA 公共密钥算法)；
(4) 用 RSA 进行加密的 nonce 交换；

(5) 用加密 nonce 进行的一种“校订”验证方法，它与其他加密的 nonce 方法稍有区别。

“nonce”是一种随机数字。IKE交换牵涉到的每一方都会对交换的状态产生影响。

IKE SA 的创建称为“阶段一”。阶段一完成后，阶段二便开始了。在这个阶段中，要创建IPSec SA 。针对阶段一，可选择执行两种交换。一种叫主模式(Main Mode)交换，另一种叫野蛮模式(Aggressive Mode)交换。其中，野蛮模式的速度较快，但主模式更显灵活。而阶段二仅能选择一种交换模式，即Quick Mode (快速模式)。

为正确实施IKE，需遵守三份文件(文档)的规定，它们分别是：

(1) 基本 ISAKMP 规范(R F C 2 4 0 8)；

(2) IPSec 解释域(R F C 2 4 0 7)；

(3) IKE 规范(R F C 2 4 0 9)。

13.3　传输层安全协议 SSL

SSL(Secure Sockets Layer)是由 Netscape 公司开发的一套 Internet 数据安全协议，目前已广泛用于 Web 浏览器与服务器之间的身份认证和加密数据传输。SSL 协议位于 TCP/IP 协议与各种应用层协议之间，为数据通信提供安全支持，如图 13.8 所示。

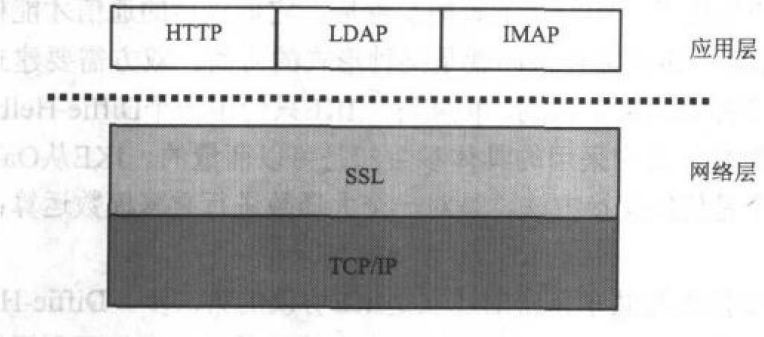


图 13.8　SSL 在 TCP/IP 协议栈中的位置

13.3.1　SSL 体系结构

SSL 被设计成使用 TCP 来提供一种可靠的端到端的安全服务。SSL 分为两层协议，如 13.9 所示。

可以看到，SSL 握手协议、修改密文协议、告警协议位于上层，SSL 记录协议为不同的更高层协议提供了基本的安全服务，可以看到 HTTP 可以在 SSL 上运行。

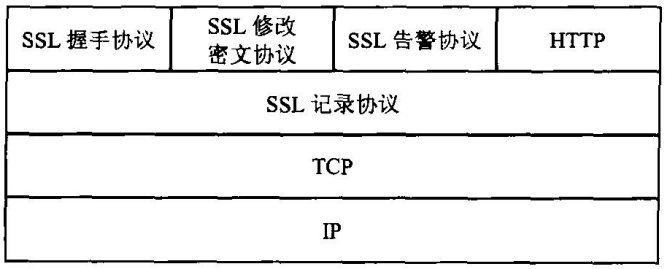


图 13.9 SSL 协议体系结构

SSL 中有两个重要概念：SSL 会话和 SSL 连接，在协议中定义如下：

(1) SSL 连接：连接是提供恰当类型服务的传输。SSL 连接是点对点的关系，每一个连接与一个会话相联系。

(2) SSL 会话：SSL 会话是客户和服务器之间的关联，会话通过握手协议来创建。会话定义了加密安全参数的一个集合，该集合可以被多个连接所共享。会话可以用来避免为每个连接进行昂贵的新安全参数的协商。

13.3.2 SSL 记录协议

SSL 记录协议为 SSL 连接提供两种服务：

(1) 机密性：握手协议定义了共享的、可以用于对 SSL 有效载荷进行常规加密的密钥；

(2) 报文完整性：握手协议定义了共享的、可以用来形成报文的 MAC 码和密钥。

SSL 记录协议接受传输的应用报文，将数据分片成可管理的块，可选地压缩数据，应用 MAC，加密，增加首部，在 TCP 报文段中传输结果单元；被接受的数据被解密、验证、解压和重新装配，然后交给更上层的应用。

SSL 记录协议的操作步骤如下(如图 13.10 所示)：

(1) 分片：每个上层报文被分成 16KB 或更小的数据块。

(2) 压缩：压缩是可选的应用，压缩的前提是不能丢失信息，并且增加的内容长度不能超过 1 024 字节，在 SSLv3 中没有说明任何压缩算法，因此缺省的压缩算法为空。

(3) 增加 MAC 码：这一步需要用到共享的

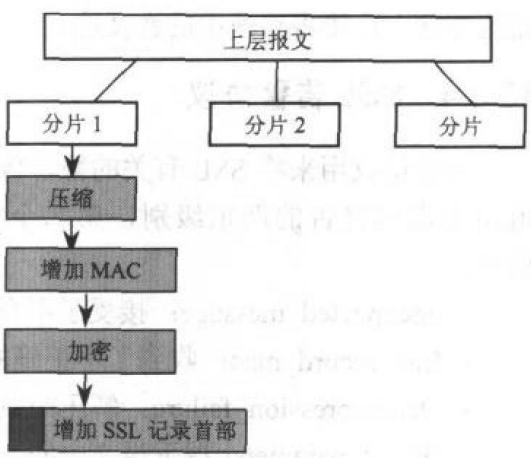


图 13.10 SSL 记录协议的操作

密钥。

(4) 加密：使用同步加密算法对压缩报文和 MAC 码进行加密。加密对内容长度的增加不可超过 1 024 字节。表 13.1 所示为一些可选的加密算法。

(5) 增加 SSL 首部。该首部由以下字段组成：

a) 内容类型(8 比特)：用来处理这个数据片更高层的协议；

b) 主要版本(8 比特)：指示 SSL 的主要版本，例如：SSLv3 的本字段值为 3；

c) 次要版本(8 比特)：指示使用的次要版本；

d) 压缩长度(16 比特)：明文数据片以字节为单位的长度，最大值是 16K＋2K。

表 13.1　可选的加密算法

分组密文		流密文	
算法	密钥长度(比特)	算法	密钥长度(比特)
IDEA	128	RC4-40	40
RC2-40	40	RC4-128	128
DES-40	40		
DES	56		
3DES	168		
Fortezza	80		

13.3.3　SSL 修改密文规约协议

SSL 修改密文规约协议是 SSL 协议体系中最简单的一个，它由单个报文构成，该报文由值为 1 的单个字节组成。这个报文的惟一目的就是使得挂起状态被复制到当前状态，从而改变这个连接将要使用的密文族。

13.3.4　SSL 告警协议

告警协议用来将 SSL 有关的警告传送给对方实体。它由两个字节组成，第一个字节的值用来表明警告的严重级别，第二个字节表示特定告警的代码。下面列出了一些告警信息：

- unexpected_message：接受了不合适的报文；
- Bad_record_mac：收到了不正确的 MAC；
- Decompression_failure：解压函数收到不适当的输入；
- Illegal_parameter:握手报文中的一个字段超出范围或与其他字段不兼容；
- Certificate_revoked：证书已经被废弃；

• Bad_certificate：收到的证书是错误的。

13.3.5 SSL 握手协议

SSL 协议中最复杂的部分是握手协议。这个协议使得服务器和客户能相互鉴别对方的身份、协商加密和 MAC 算法以及用来保护在 SSL 记录中发送数据的加密密钥。在传输任何应用数据前，都必须使用握手协议。

握手协议由一系列在客户和服务器之间交换的报文组成。所有这些报文具有三个字段：

(1) 类型(1 字节)：指示 10 种报文中的一个；

(2) 长度(3 字节)：以字节为单位的报文长度；

(3) 内容(>=1 字节)：和这个报文有关的参数。

表 13.2 所示为 SSL 握手协议报文的类型和参数。

表 13.2 SSL 握手协议报文类型和参数

报文类型	参数
Hello_request	空
Client_hello	版本、随机数、会话 ID、密文族、压缩方法
Server_hello	版本、随机数、会话 ID、密文族、压缩方法
Certificate	X.509v3 证书链
Server_key_exchange	参数、签名
Certificate_request	类型、授权
Serverhello_done	空
Certificate_verify	签名
Client_key_exchange	参数、签名
Finished	散列值

握手协议的动作可以分为 4 个阶段，整个过程如图 13.11 所示：

阶段一：建立安全能力，包括协议版本、会话 ID、密文族、压缩方法和初始随机数。这个阶段将开始逻辑连接并且建立和这个连接相关联的安全能力；

阶段二：服务器鉴别和密钥交换；

阶段三：客户鉴别和密钥交换；

阶段四：结束，这个阶段完成安全连接的建立。

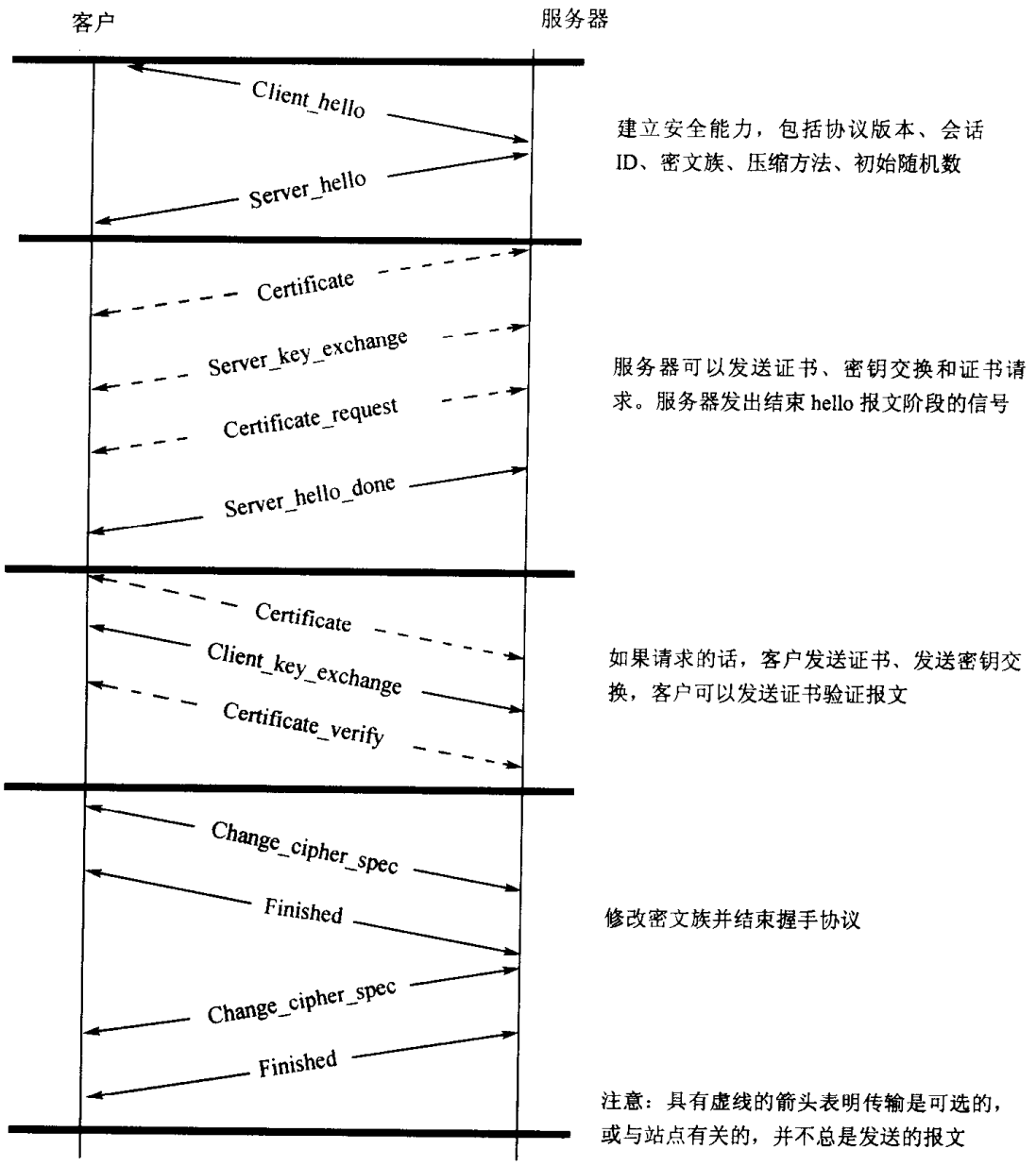


图 13.11　SSL 握手协议的动作

13.4　安全协议的应用

目前，网络安全协议已得到广泛的应用，出现了许多基于安全协议的产品和技术。例

如，基于 IPSec 的 VPN(虚拟专用网)技术（如图 13.12 所示），在 Windows2000/XP 下也内置了 IPSec 的实现。SSL 的应用则更广泛，例如 Web 浏览器，可以通过使用 SSL 来达到网页传输的安全性。此外，许多安全产品(例如防火墙)也采用了网络安全协议，为了保障安全产品的互通互连，必须采用统一的协议规范。

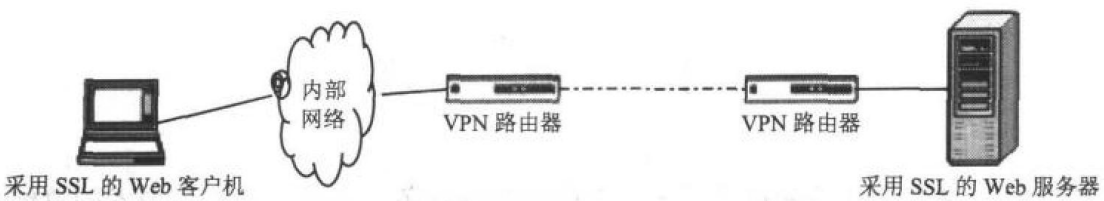


图 13.12 SSL 和 IPSec 的一个简单应用

习 题 十 三

1. 简述不同网络协议层常用的安全协议。
2. IPSec 协议的隧道模式和传输模式有何区别？如何实现？
3. IPSec 中 ESP 和 AH 分别有什么作用？能否同时使用？
4. 简述 IPSec 的密钥交换原理。
5. SSL 协议的目标是什么？
6. 简述 SSL 协议建立安全连接的过程。

第十四章　安全评估标准

14.1　概　　述

安全标准是安全理论和技术用于实践的纲领性规范，是安全理论和技术的总结，对安全产业发展具有指导性作用。安全标准对安全产品的功能、结构及互操作都提出了要求。安全标准的制定也是一个国家科研水平、技术能力的体现，反映了一个国家的综合实力。安全标准还是加入 WTO 的国家保护自己利益的重要手段。因此，各国都很注重安全标准的研究、制定和推广工作。

目前已有很多安全标准，包括安全体系结构的标准、密码技术标准、安全认证标准、安全产品标准、安全评估标准等等。在各种安全标准中，研究最多、影响最广泛的当属安全评估标准，因此本章将重点介绍这方面的发展状况。

安全评估标准最早起源于美国。1970 年，美国国防部(DoD)在国家安全局(NSA)建立了计算机安全中心(NCSC)，开始进行有关计算机安全评估标准的研究工作。美国国防部于 1984 年公布了“可信计算机系统评估准则(Trusted Computer System Evaluation Criteria，TCSEC)”，由于该书封面是桔色包装，因此俗称桔皮书。桔皮书论述的重点是通用的操作系统，为了使它的评判方法适用范围更广，NCSC 于 1987 年出版了一系列增补解释，如“可信计算机数据库解释”(黄皮书)、“可信计算机网络解释”(红皮书 TNI)等，俗称彩虹系列。其中，TNI 在桔皮书的基础上，增加了与网络安全评估有关的解释与说明；TNI 从网络安全的角度出发，解释了准则中的观点，从用户登录、授权管理、访问控制、审计跟踪、隐通道分析、可信通道建立、安全检测、生命周期保障、文本写作、用户指南等各个方面提出了规范性要求。

TCSEC 是应用最早和影响最大的计算机安全评估标准，它带动了国际计算机安全的评估研究。但是随着时间推移，TCSEC 暴露出严重的局限性，它偏重于信息安全的保密性，而对于完整性与可用性没有给予足够的重视。20 世纪 90 年代西欧四国(英、法、荷、德)在各自安全评估准则的基础上联合提出了“信息技术安全评估准则”(Information Technology Security Evaluation Criteria，ITSEC)，ITSEC(又称欧洲白皮书)除了吸收 TCSEC 的成功经验外，首次提出了信息安全的保密性、完整性、可用性，把可信计算机的概念提高到可信信息技术的高度上来认识。它推荐从八个方面来评价信息系统的安全性，分

别是识别和认证、访问控制、责任、审计、客体复用、精确性、服务的可靠性和数据交换。这些工作成为欧共体信息安全计划的基础，并对国际信息安全的研究和实施带来深刻的影响。

加拿大 1988 年开始制订“加拿大可信计算机产品评估准则，(CTCPEC)”，1989 年 5 月公布第一版，并于 1993 年 1 月在结合 TCSEC 与 ITSEC 的基础上，公布了第三版[CTCPEC93]。

1993 年，美国对 TCSEC 做了补充和修改，制定了“组合的联邦标准”(简称 FC)[FC92]，对保护框架和安全目标作了定义，明确规定了由用户提供其系统安全保护需求的详细框架，厂商定义产品的安全功能、安全目标等，但该标准有很多缺陷，只是一个过渡标准。

1993 年 6 月，CTCPEC、FC、TCSEC 和 ITSEC 的发起者联合起来，将各自独立的标准组合成一个单一的、能被广泛使用的 IT 安全准则。这一行动被称为 CC 项目，目的是解决原标准中出现的概念和技术上的差异，并把结果作为国际标准提交给 ISO。这些发起组织来自六国七方：加拿大、法国、德国、荷兰、英国、美国国家标准及技术研究所(NIST)和国家安全局(NSA)。

CC 的基础是欧洲的 ITSEC、美国的包括 TCSEC 在内的新的联邦评价标准、加拿大的 CTCPEC 以及国际标准化组织 ISO :SC27 WG3 的安全评价标准。1995 年颁布 CC0.9 版，1996 年 1 月更新为 CC1.0 版， 1997 年 8 月更新为 CC2.0 Beta 版，1998 年 5 月发布 CC 2.0 版。

CC 2.1 版于 1999 年发布，并被 ISO 采纳作为国际标准 ISO 15408 发布[CC99]。CC 在 ISO 行文中的正式名称应为“信息技术安全评估准则”，但出于历史的和连续性的目的，ISO/IEC 第一联合技术委员会已经同意在文档中继续使用“通用准则(CC)”这一术语。

图 14.1 展示了安全评估标准的发展历程。

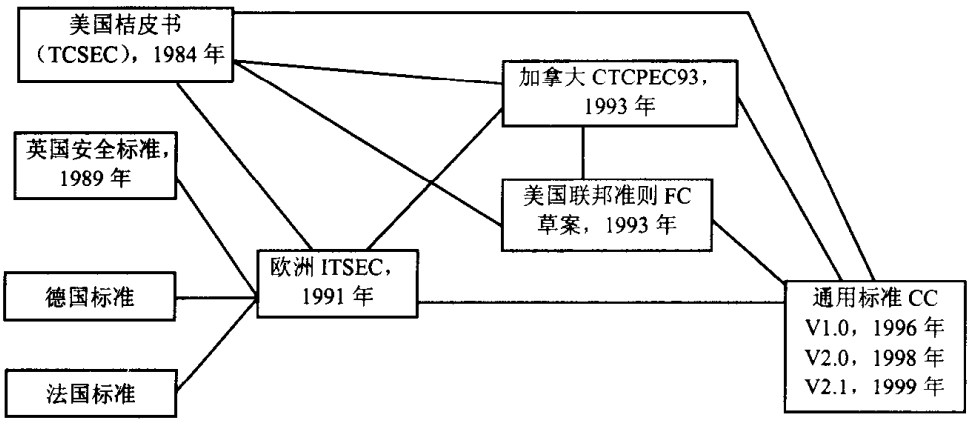


图 14.1 安全评估标准的发展历程

制订 CC 标准的目的是建立一个各国都能通用的信息安全产品和系统的安全性评估准则，国家与国家之间可以通过签订互认协议，决定相互接受的认可级别。这样能使大部分的基础性安全机制，在任何一个地方通过了 CC 准则评估并得到许可进入国际市场时，就不需要再作评价，使用国只需测试与国家主权和安全相关的安全功能从而大幅度节省评价支出并迅速推向市场。CC 标准吸收了各先进国家关于现代信息系统安全的经验与知识，将对未来信息安全的研究与应用带来重大影响。

由于信息系统和安全产品的安全性评估具有特殊性，各国都不会无条件地接受由他国所作的评估结果，大多数国家都要通过本国标准的测试才给予认可。因此，很少有国家会把信息安全产品和系统的安全性基于在国际的评估标准、评估体系和评估结果，而是在充分借鉴国际标准的前提下，制定本国的安全评估标准。

在我国，20 世纪 90 年代开始起草国内的安全评估标准，1999 年 9 月 13 日发布了《计算机信息系统安全保护等级划分准则》，并于 2001 年 1 月 1 日起实施。该标准相关配套标准也在制定和修订中。

14.2　国际安全标准

14.2.1　TCSEC

1984 年，美国国防部(DoD)发布了《可信计算机系统评估准则》(Trusted Computer System Evaluation Criteria，缩写为 TCSEC)，即所谓的桔皮书。该准则于 1985 年修订后重新公布，1989 年又发布了新的版本。

TCSEC 的发布主要有以下三个目的：

(1) 为制造商提供一个安全标准，使他们在开发商业产品时加入相应的安全因素，为用户提供广泛可信的应用系统；

(2) 为国防部各部门提供一个度量标准，用来评估计算机系统或其他敏感信息的可信程度；

(3) 在分析、研究规范时，为制定安全需求提供基础。

TCSEC 采用等级评估的方法，将计算机安全分为 A、B、C、D 四个等级八个级别。其中 D 等的安全级别最低，A 等的安全级别最高。各级名称如表 14.1：

TCSEC 从安全策略、可审计性、保证和文档四个方面对不同安全级别的系统提出了不同强度的要求，随着安全等级的提高，系统的安全性增加，风险降低。有关各级别具体要求如表 14.2 所示。

表 14.1 TCSEC 中的等级划分

等级分类	保护等级
D 类：最低保护等级	D 级：无保护级
C 类：自主保护级	C1 级：自主安全保护级
	C2 级：控制访问保护级
B 类：强制保护级	B1 级：标记安全保护级
	B2 级：结构化保护级
	B3 级：安全域保护级
A 类：验证保护级	A1 级：验证设计级
	超 A1 级

表 14.2 系统安全等级评估准则

类别	评估	D	C1	C2	B1	B2	B3	A1	A2
安全策略	自主访问控制	*	*	—	—	*	—	—	—
	客体重用			*	—	—	—	—	—
	标记								
	标记的完整性								
	多级设备输出				*	—	—	—	—
	单级设备输出				*	—	—	—	—
	标记的硬拷贝输出				*	—	—	—	—
	输出					*	—	—	—
	主体安全标记					*	—	—	—
	设备标记				*	*	—	—	—
	强制访问控制								
可审计性	标识与验证		*	*	*	—	—	—	—
	可信路径					*	*	—	—
	安全审计			*	*	*	*	—	—
保证	操作保证								
	系统结构		*	*	*	—	—	—	—
	系统完整性			*	—	—	—	—	—
	隐蔽信道分析					*	*	*	—
	可信设施管理					*	*	—	—
	可信恢复						*	—	—
	生命周期保证								
	安全性测试		*	*	*	*	*	*	*
	设计规范与验证				*	*	*	*	*
	配置管理					*	—	*	—
	可信分发							*	—
文档	安全特性用户指南		*	—	—	—	—	—	—
	可信设施手册		*	*	*	*	*	—	—
	测试文档		*	—	—	*	—	—	—
	设计文件		*	—	*	*	*	*	*

注：表中空白表示无要求；* 表示增加或有修改；— 表示与前一等级相同。

TCSEC 提出了可信计算基(Trusted Computing Base，缩写为 TCB)概念，即计算机系统负责执行安全策略的保护机制的全体由硬件、软件、固件组合而成。一个 TCB 由一个或多个在产品或系统上一同执行统一的安全策略的部件组成。

下面，对每个安全等级的内容和要求作简要的说明。

1．无保护级

D 等是最低保护等级，即非保护级。它是为那些经过评估，但不满足较高评估等级要求的系统设计的，只具有一个级别。该等级是指不符合要求的那些系统，这种系统不能用于在多用户环境下处理敏感信息。

2．自主保护级

C 等为自主保护级，它具有一定的保护能力，主要通过身份认证、自主访问控制和审计等安全措施来保护系统。C 等一般只适用于具有一定等级要求的多用户环境，通过对主体责任和主体初始动作的控制和审计，对主体及其产生的动作负责。这一等级分为 C1 和 C2 两个级别。

(1) 自主安全保护级(C1 级)

C1 级通过提供用户与数据的隔离，满足 TCB 自主安全要求。在这一级，TCB 应在命名用户和命名客体之间加以定义并进行访问控制。采用的机理(如个人 / 组 / 公共控制，访问控制表)应允许客体拥有者指定、控制客体是由自己使用，还是由用户组或公共使用。该级需要在进行任何活动之前，TCB 能确认用户身份(如采用口令)，并保护确认数据，以免未经授权对确认数据访问和修改。通过自主定义和控制客体拥有者，可以防止客体被其他用户有意或无意地读出、篡改、干涉和破坏。同时 TCB 定期检查运行的正确性。为保证系统的完整性，还要求硬件和软件提供保证 TCB 连续有效操作的特性。

目前生产的大多数计算机系统都能达到这一等级。

(2) 可控安全保护级(C2 级)

在 C2 级，计算机系统比 C1 级具有更细粒度的自主访问控制，通过注册过程控制、审计安全相关事件以及资源隔离，使单个用户为其行为负责。

C2 级的访问控制粒度细化到单个用户而不是组，因此在注册时就要求按单个用户进行鉴别和审计，提供授权服务，防止存取权限扩散，能对确定用户的动作或默认客体提供保护，避免非授权存取。具体来说可指定哪些用户能够访问哪些客体，未经授权用户不得访问已指定访问权的客体。同时，它还提供了客体重用功能，即对一个未使用的存储客体，TCB 应该能够保证该客体不包含未授权主体的数据。

在审计方面，除具有 C1 级全部功能外，C2 级还提供惟一的识别系统中各个用户身份

的能力；提供将这种身份与该客体用户发生的所有审计动作相联系的能力。C2 级系统能与该识别符合，可审计所有主体进行的各种活动；能对 TCB 进行建立和维护，对客体存取的审计进行跟踪，并保护审计信息，防止其被修改、毁坏或未经授权访问。

TCB 还能记录下列类型的事件：确认和识别安全机理的使用，将客体引入用户地址空间，客体删除，以及操作人员、系统管理人员和安全管理人员进行的各种与安全相关的活动。

对每个审计事件，审计记录包括：用户名、事件发生时间、事件类型、事件的成功或失败等。对于确认事件，审计记录还应包括请求源(如终端 ID)；对于客体进行访问的事件，审计记录应包括客体名。系统管理人员应能通过识别符，有选择地审计任一用户或多个用户的活动。

目前主要的几个商业操作系统都达到这一等级，如 Windows NT。

3. 强制安全保护级

B 等为强制保护级。这一等级比 C 等级的安全功能有大幅增强。它要求对客体实施强制访问控制，并要求客体必须带有敏感标记，可信计算机利用敏感标记去施加强制访问控制。这一部分可分为 B1、B2 和 B3 三级。

(1) 标记安全保护级(B1)

该级具有 C2 级的全部功能，并增加了标记、强制访问控制、责任、审计和保证功能。具体要求如下：

1) 标记

B1 级要求为每个主体和存储客体指定敏感标记，并由 TCB 维护，这些标记在实施强制访问控制时使用。为了引导非标记信息，TCB 将授权用户请求并接收数据安全级。主体的各种活动能通过 TCB 审计。这一级标记有如下要求：

a) 标记的完整性：安全标记应能准确地体现主体和客体的安全级别。当敏感标记由 TCB 输出时，它应该能够准确体现内部标记，并与输出信息相关联。

b) 标记信息的输出：TCB 应该能够指明，每个通信的信道和 I/O 设备，是作为单级还是多级使用。其指定都应人工完成，并由 TCB 对这种活动进行审计。

c) 多级设备输出：当 TCB 输出一客体到多级 I/O 设备时，敏感标记应与客体一起输出，并以同样的形式与输出信息一起驻留在同一物理介质上。当 TCB 通过多级通信信道输出和输入一个客体时，使用的协议应在敏感标记和发送(或接受)的信息间提供明确的对应关系。

d) 单级设备输出：不要求对单级 I/O 设备和单级通信信道所处理信息保留敏感标记。然而，TCB 应该有合理机制，能够让授权用户经由单级通信信道或 I/O 设备来安全传输标

有单级安全级信息。

e) 硬拷贝标记输出：系统管理员应该能够指定与输出敏感标记相关联的可打印标记名。TCB 标记出硬拷贝敏感标记(人们可以读的)输出的开始和结束。敏感标记可以是秘密、机密和绝密等字样。

2) 强制访问控制

TCB 应对它自己控制下的所有主体和客体施加强制访问控制策略。主体和客体要指定敏感标记，这些标记是分级保护和非分级保护结合而成，是强制访问控制判断的依据。TCB 应支持两个以上的安全级。在由 TCB 控制的主体和客体之间的访问必须满足以下要求：只有在主体的安全级中保护级小于或等于客体安全级中保护级时，才允许主体对客体进行操作。

3) 可审计性

当用户开始完成任何由 TCB 干预的活动时，TCB 将要求对他们进行识别，并且 TCB 要保存确认数据，以防止未经授权的用户访问。用于识别和确认的这些数据包括验证用户身份信息(如口令)、检查用户鉴别信息和用户授权信息。

4) 审计

B1 级的审计，除包括 C2 级的全部功能外，还需要可以对任何滥用职权的用户输出标记，对安全级记录的事件进行审计，并能对基于安全级的用户活动进行有选择的审计。

(2) 结构保护级(B2)

该级着重强调实际操作中的评价手段。为此，B2 级增加了以下功能：

1) 安全策略

加强了强制访问功能。将强制访问控制对象从主体和客体扩展到 I/O 设备等所有资源，并要求各种系统资源必须与安全标记相联系。

2) 可审计性

在责任方面，提高了连续保护和防渗透能力。保证了和用户之间开始注册和确认时的通路是可信的，提高了系统连续保护和防渗入的能力。要求审计功能得到加强，能审计使用隐蔽存储信道的标记事件。所谓隐蔽信道，是指用违反系统安全策略的方法传输信息的通道，如，一个进程直接或间接地对一个存储单元写，而另一个进程直接或间接地对该存储单元读，就形成一个隐蔽通道。

3) 结构

在结构上应支持操作人员和管理人员相分离，并执行最小特权原则。也就是说，每个主体进行授权任务时，应被授予完成任务所必需的最小存取权；此外还应划分与保护有关和无关部分，并把它的执行维持在一个固定的区域，以防止外部干预和篡改，其设计和实现要利用检测和评估，以实现操作人员和管理人员相分离。

(3) 强制安全区域级(B3)

B3 级除满足 B2 级的要求外，还需要能监督所有主体对客体的访问，防篡改，并提供分析和测试；并将审计机理扩展到能报知与安全有关的事件。系统要有恢复能力，为此，B3 级增加了一个安全策略。

1) 安全策略

采用访问控制表进行控制，允许用户指定和控制对客体的共享，也可以指定命名用户对命名客体的访问方式。

2) 可审计性

能监视安全审计事件的发生和积累，当超过一定阀值时，能立即报知安全管理人员，进行处理。

3) 保证

只能完成与安全有关的管理功能，对其他完成非安全功能的操作，要严加限制。在系统出现故障或灾难性事件后，要提供过程和机理，以保证在不损害保护的条件下，恢复系统的正常。

4. 验证安全保护级

A 类系统的特点是，使用形式化的安全验证方法来保证系统的自主，并强制安全控制措施有效地保护系统中存储和处理的秘密信息或其他敏感信息。为证明 TCB 满足设计、开发及实现等各个方面的安全要求，采用形式化验证方法，系统应提供丰富的文档信息。

该等级分为 A1 和超 A1 两个级。

(1) 验证设计级(A1 级)

A1 级系统在功能上和 B3 级系统是相同的，没有增加体系结构特性和策略要求。其最显著的特点是，用形式化设计规范和验证方法来对系统进行分析，确保 TCB 按设计要求实现。从本质上说，这种保证是发展的，它从一个安全策略的形式化模型和设计的形式化高层规约(FTLS)开始。针对 A1 级系统设计验证，有 5 种独立于特定规约语言或验证方法的重要准则。

1) 安全策略的形式化模型必须明确标识并文档化，以提供该模型与其公理一致的、能够对安全策略给予足够支持的数学证明。

2) 应提供形式化的高层规约，包括 TCB 功能的抽象定义，以及用于隔离执行域的硬件/固件机制的抽象定义。

3) 应通过形式化的技术(如果可能的话)和非形式化的技术，证明 TCB 的形式化高层规约(FTLS)与模型是一致的。

4) 通过非形式化的方法，证明 TCB 的实现(硬件、固件、软件)与形式化的高层规约

(FTLS)是一致的。应证明，FTLS 的元素与 TCB 的元素是一致的，FTLS 表达应用于满足安全策略的一致的保护机制，这些保护机制的元素应映射到 TCB 的要素。

5) 应使用形式化的方法标识和分析隐蔽信道，非形式化的方法可以用来标识时间隐蔽信道，必须对系统中存在的隐蔽信道进行解释。

A1 级系统要求更严格的配置管理，要求建立系统安全分发的程序。支持系统安全管理员的职能。

(2) 超 A1 级

本级在 A1 级基础上增加了许多安全措施，超出了目前的技术发展，这些讨论只是为了对今后的工作提供一些指导，在未来，随着更多、更好的分析技术出现，本级系统的要求才会变得更加明确。今后，形式化的验证方法将应用到源码一级，时间隐蔽信道也将得到全面的分析。在这一级，设计环境将变的更重要，形式化高层规约的分析将对测试提供帮助，TCB 开发中使用的工具的正确性及 TCB 运行的软硬件功能的正确性，将得到更多的关注。超 A1 级系统涉及的范围包括系统体系结构、安全测试、形式化规约与验证、可信设计环境等。

14.2.2　通用准则 CC

1. CC 简介

CC 作为国际标准，对信息系统的安全功能、安全保障给出了分类描述，并综合考虑信息系统的资产价值、威胁等因素后，对被评估对象提出了安全需求(保护轮廓 PP)及安全实现(安全目标 ST)等方面的评估。本节概要地介绍 CC 的主要思想。

(1) CC 的范围

CC 重点考虑人为的信息威胁，无论是有意的还是无意的，不过，CC 也可用于非人为因素导致的威胁。

CC 适用于硬件、固件和软件实现的信息技术安全措施，而某些内容因涉及特殊专业技术或仅是信息技术安全的外围技术，因此不在 CC 的范围内，例如：

1) CC 不包括那些与信息技术安全措施没有直接关联的、属于行政性管理安全措施的安全评估准则。在评估对象(Target of Evaluation, TOE)的运行环境中，这类管理安全措施被认为是 TOE 安全使用的前提条件。

2) CC 不专门针对信息技术安全性的物理方面(诸如电磁辐射控制)的评估。

3) CC 不涉及评估方法学，也不涉及评估机构使用 CC 的管理模式或法律框架。

4) 评估结果用于产品和系统认可的过程不在 CC 的范围之内。

5) CC 不包括密码算法固有质量评价准则。

(2) CC 的结构

CC 全文包括三个部分：

第一部分：简介和一般模型。这一部分介绍了 CC 的一般概念和格式，描述了 CC 的结构和适用范围，描述了安全功能、保证需求的定义，并给出了保护轮廓(PP)和安全目标(ST)的结构。

第二部分：安全功能要求。这一部分为用户和开发者提供了一系列安全功能组件，作为表述评估对象(TOE)功能要求的标准方法，在保护轮廓(PP)和安全目标(ST)中将使用这些功能组件进行描述。

第三部分：安全保证要求。这一部分为开发者提供了一系列安全保证组件，作为表述评估对象(TOE)保证要求的标准方法，同时还提出了七个评估保证级别(Evaluation Assurance Levels：EALs)。各保障级别与 TCSEC 级别的大致对应关系如表 14.3 所示。

表 14.3 评估保证级别(EAL)与 TCSEC 的对照

CC 保证级	CC 保证级名称	相当的 TCSEC 级
EAL1	功能测试	
EAL2	结构测试	C1
EAL3	系统测试和检查	C2
EAL4	系统设计、测试和复查	B1
EAL5	半形式化设计和测试	B2
EAL6	半形式化验证的设计和测试	B3
EAL7	形式化验证的设计和测试	A1

2. CC 的一般模型

(1) 一般安全上下文

CC 认为，安全就是保护资产不受威胁，威胁可依据滥用被保护资产的可能性进行分类。所有的威胁类型都应该考虑到，但是在安全领域内，被高度重视的威胁是和人们的恶意攻击及其他人类活动密切联系的。图 14.2 说明了相关概念和关系。

图中说明，保护资产是资产所有者的责任，而实际或假定的威胁者试图以与所有者初衷相反的方式滥用资产。所有者会意识到这种威胁可能致使资产损坏，对所有者而言，资产的价值将会降低。安全性损坏一般包括但又不仅仅包括以下几项：资产破坏性地暴露于未授权的接收者(失去保密性)；资产由于未授权的更改而损坏(失去完整性)；授权用户无法访问资产(失去可用性)。

资产所有者必须分析可能的威胁并确定哪些存在于他们的环境中，从而导致风险。这种分析有助于选择对策，把风险降低到一个可接受的水平。

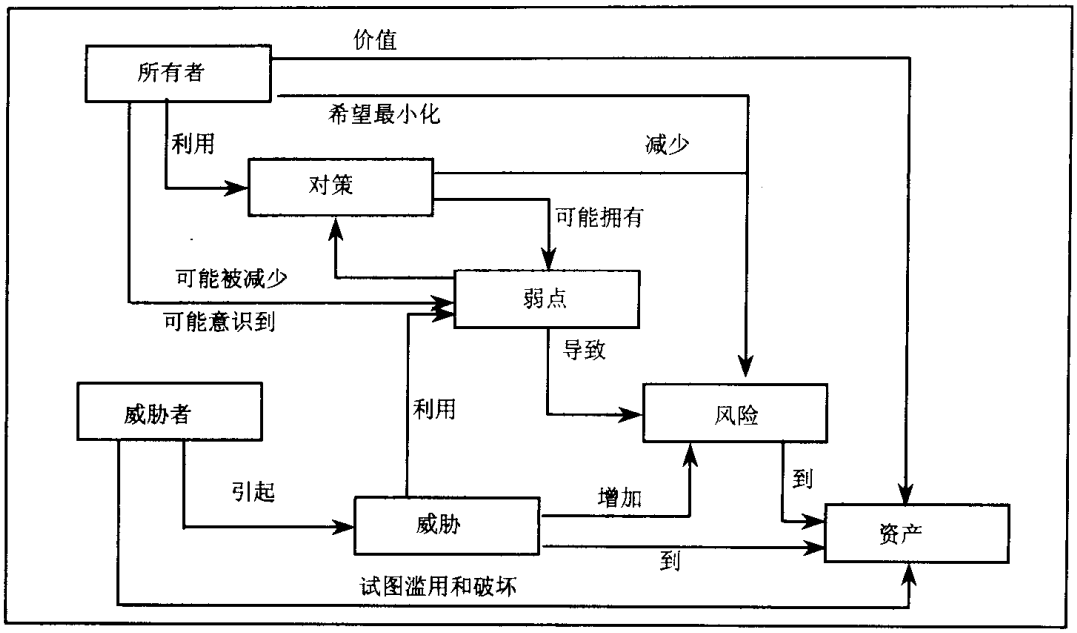


图 14.2　安全概念和关系

对策用以(直接或间接地)减少脆弱性，并满足资产所有者的安全策略。在安全策略使用后仍会有残留的弱点，这些弱点仍可以被威胁者利用，从而造成了资产的残余风险。资产所有者将通过给出其他的约束来寻求残留风险的最小化。

在将资产暴露于特定威胁之前，所有者需要确信其对策足以应付所面临的威胁。所有者自身可能没有能力判断对策的所有方面，但可以寻求对策的评估。评估结果对安全保证可达程度做出描述，即对策能否被信任用于降低风险、保护资产。该描述还将对策的保证进行分级。保证是对策的特性，对策是正确操作时获得信心的基础。资产所有者可以根据此描述决定是否接受将资产暴露给威胁者所冒的风险。图 14.3 说明了这种关系。

(2) TOE 评估

图 14.4 描述的 TOE 评估过程可能与开发过程同步进行，或有所滞后。TOE 评估过程的主要输入内容有：

1) 一系列 TOE 证据，包括评估过的 ST 作为 TOE 评估的基础；

2) 需要评估的 TOE；

3) 评估准则、方法和方案；

另外，说明性材料(例如 CC 的使用说明书)和评估者及评估组织的 IT 安全专业知识也常常是评估过程的输入内容。

评估过程的期望结果是对 TOE 满足 ST 中安全要求的确认，即一个或多个由评估准则规定的评估者对 TOE 的裁决报告。这些报告对 TOE 所代表产品或系统的用户和潜在用户

非常有用，对开发者也同样有用。

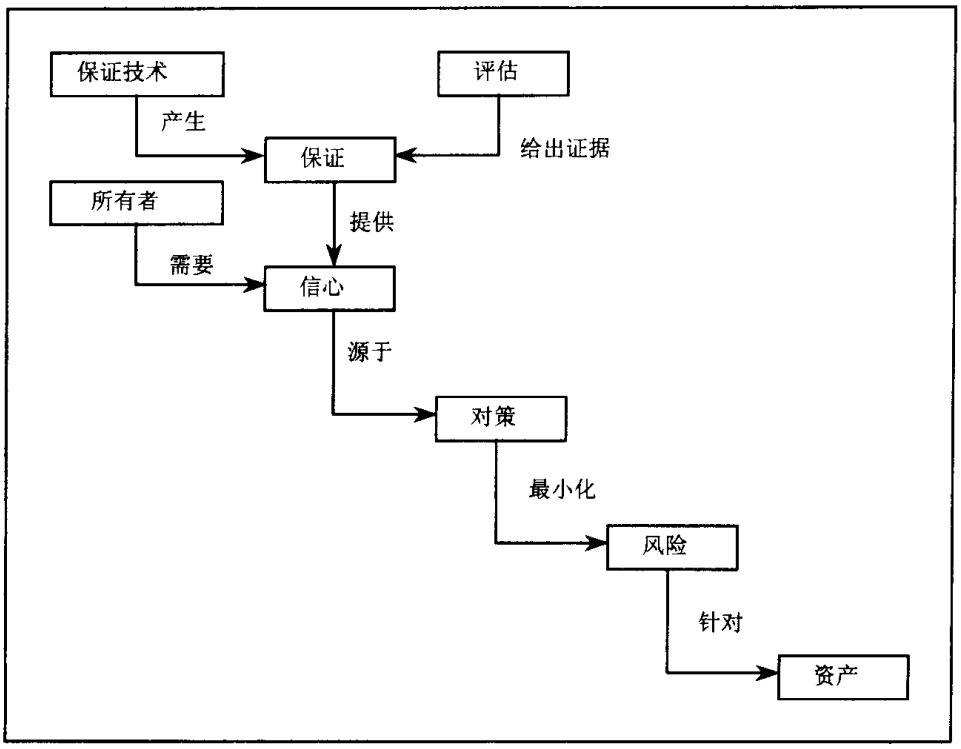


图 14.3 评估概念和关系

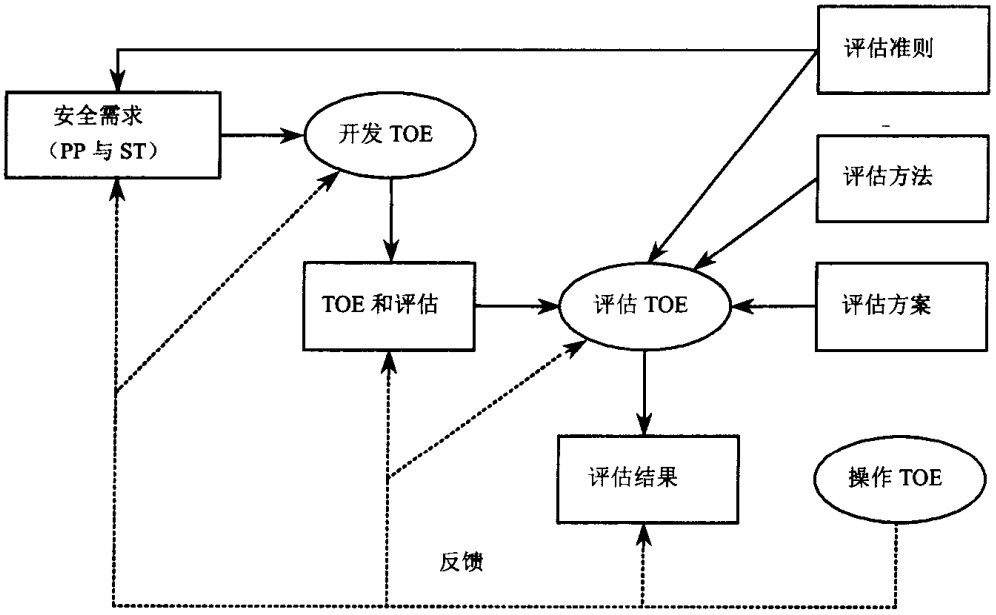


图 14.4 TOE 评估过程

通过评估获得的信任度依赖于所达到的保证要求(即评估保证级别)。

评估过程通过两种途径改进安全产品。首先，评估过程能发现开发者可以纠正的 TOE 错误或弱点，从而减少将来操作中安全失效的可能性；其次，为了通过严格的评估，开发者在 TOE 设计和开发时也将更加细心。总之，评估过程对最初需求、开发过程、最终产品以及操作环境都将产生积极的影响。

(3) CC 安全概念

只有在 IT 环境中考虑 IT 组件保护资产的能力时，CC 才是可用的。为了表明资产是安全的，必须在各个层面考虑安全性，包括从最抽象的设计到最终的 IT 实现。

CC 要求在某层次上的表述包含在该层次上 TOE 描述的基本原理。也就是说，这个层次必须包含合情合理、令人信服的论据，表示它与更高层次是一致的，而且它自己也是全面的、正确的、内部一致的。通过陈述与安全目标的符合程度及其基本原理，可以表明 TOE 在对抗威胁、执行安全策略时的有效性。

如图 14.5 所示，CC 将表述分成不同的层次，包括安全环境、安全目的、安全需求、安全规范等，它阐明了一种在开发 PP 或 ST 时得出安全需求和规范的方法。所有的 TOE 安全要求根本上均来源于对 TOE 目的和环境的考虑。这个图并不限制 PP 和 ST 的开发方法，只是阐明一些分析方法的结果是怎么与 PP 和 ST 的内容相联系的。

• 安全环境

安全环境包括所有相关的法规、组织性安全策略、习惯、专门技术和知识。因此它定义了 TOE 使用的上下文，同时也包括环境里出现的安全威胁。

为建立安全环境，PP 或 ST 的作者必须考虑以下几点：

1) TOE 物理环境，指所有与 TOE 安全相关的 TOE 运行环境，包括已知的物理环境和人事的安全安排。

2) 需要根据安全策略由 TOE 元素实施保护的资产。这可能包括直接相关的资产，如文件和数据库，也包括受安全要求间接支配的资产，如授权凭证和 IT 实现本身。

3) TOE 目的，说明产品类型和可能的 TOE 用途。

• 安全目的

安全环境的分析结果用来阐明如何对抗已标识的威胁，说明组织性安全策略和假设的安全目的。安全目的应和已说明的 TOE 运行目标、产品目标以及有关的物理环境知识一致。

确定安全目的的意图是为了阐明所有的安全考虑，并指出安全方面的各种问题是由 TOE 直接处理还是由它的环境来处理。这种划分需要工程判断、安全策略、经济因素和可接受风险决策之间的协作。

环境安全目的在 IT 领域内将用非技术的或程序化的方法来实现。

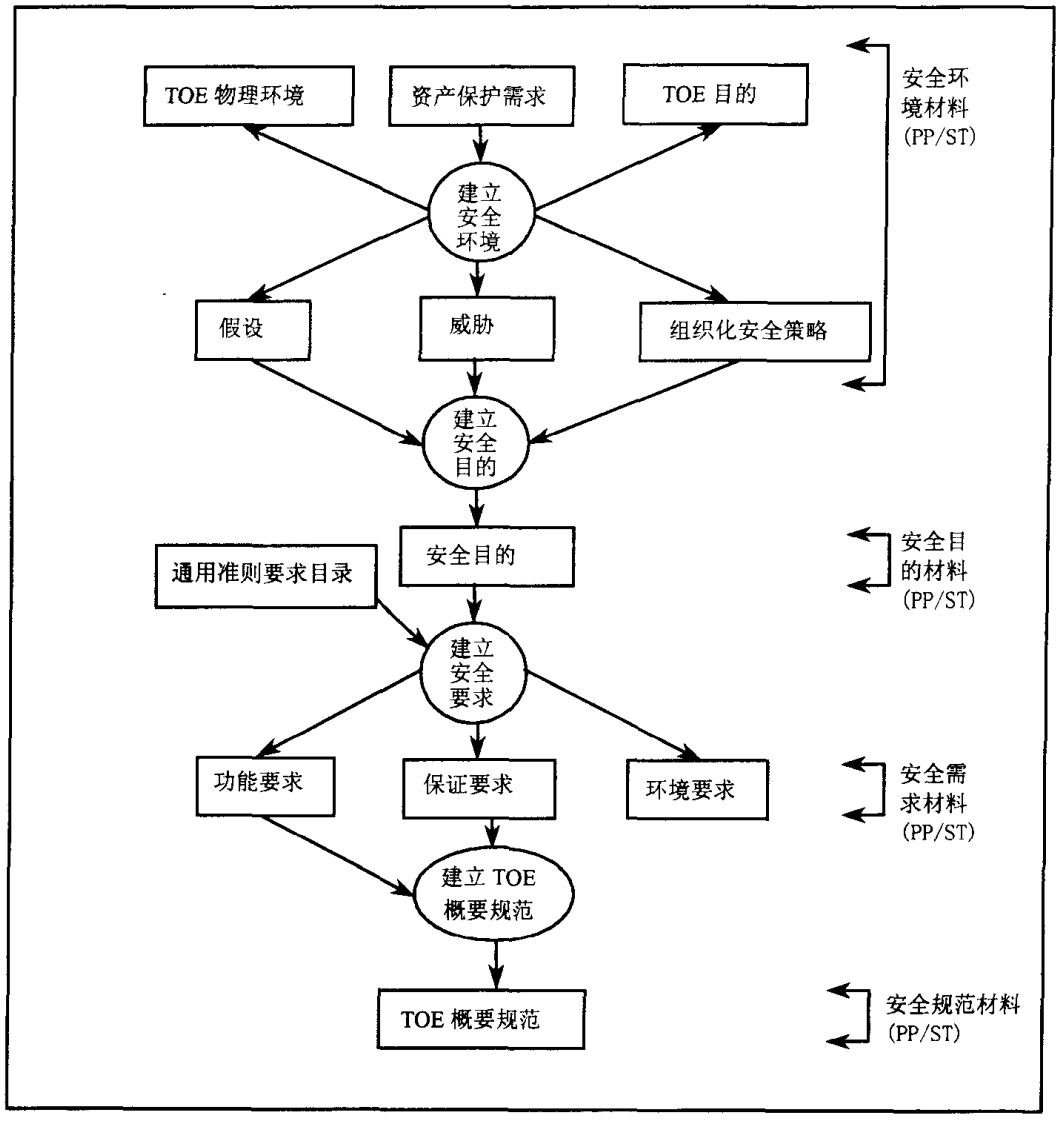


图 14.5 要求和规范的派生关系

IT 安全需求只涉及 TOE 安全目的和它的 IT 环境。

• IT 安全要求

IT 安全要求是将安全目的细化为一系列 TOE 及其环境的安全要求，一旦这些要求得到满足，就可以保证 TOE 达到其安全目的。

CC 在不同种类功能要求和保证要求下提出安全要求。

CC 的第二部分定义了功能要求，例如识别、鉴别、安全审计以及抗否认功能等。

当 TOE 包括由概率或置换机制(例如口令和散列函数)实现的安全功能时，保证要求应

满足与宣称的安全目的一致的最小强度等级，这种等级可以是基本功能强度、中等功能强度、高等功能强度中的任意一个。而功能要求应满足最小的级别或至少是选择定义的特定级别。

对给定的一组功能要求，其保证程度可以是变化的，这种变化表现为保证组件级别严格增加。CC 的第三部分使用这些组件定义了保证要求和一个评估保证级别的尺度。保证要求涉及开发者的行为、要产生的证据以及评估者的行为，例如：对开发过程的严格性约束，对查找并分析潜在安全弱点影响的需求。

通过选择合理的安全功能，可以确保达到一定的安全目的，这种保证来源于以下两个因素：

1) 对安全功能正确实现的信任，也就是评估它们是否被正确实现。

2) 对安全功能有效性的信任，也就是评估它们是否确实满足所陈述的安全目的。

安全要求通常包括出现期望行为和避免不期望行为。通过使用和检验，一般可以证明存在的期望行为，但不太可能明确证明不期望行为是不存在的。检验、设计审查、实现审查都有助于减少存在不期望行为的风险。另外，基本原理的陈述也有助于证明不存在不期望的行为。

• TOE 概要规范

ST 中提供的 TOE 概要规范定义了 TOE 安全要求的实现方法，它提供分别满足功能需求和保证需求的安全功能以及保证措施的高层定义。

3. CC 中安全要求的描述方法

CC 安全要求是以“类—族—组件”这种层次方式组织而成的(如图 14.6 所示)，这种组织方式可帮助用户定位特定的安全要求。在描述安全功能和保证要求时，CC 使用相同的风格、组织方式和术语。

(1) 类

类是最通用安全要求的组合，其所有的成员关注共同的安全焦点，但覆盖不同的安全目的。

类的成员被称为族。

(2) 族

族是若干组安全要求的组合，这些要求有共同的安全目的，但侧重点和严格性有所区别。

族的成员被称为组件。

(3) 组件

组件描述一组特定的安全要求集，它是 CC 定义结构中所包含的最小的可选安全要求集。族内具有相同目标的组件可以以安全要求强度(或能力)逐步增加的顺序排列，也可以部分地按相关非层次集合的方式组织。在某些实例中，一个族只有一个组件，因而是不可

能排序的。

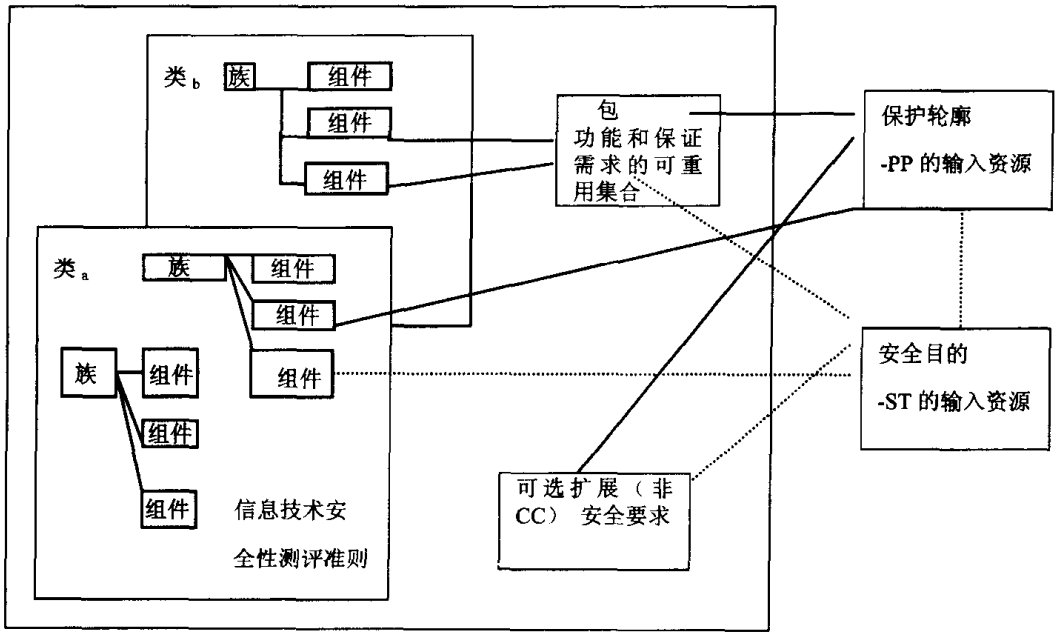


图 14.6 安全要求的组织和结构

组件由单个元素组成，元素是安全需求最低层次的表达，也是能被评估验证的不可分割的安全要求。

1) 组件间的依赖

组件之间可能存在依赖关系。当一个组件无法充分表达安全要求并且依赖于另一个组件的存在时，依赖关系就产生了。依赖关系可以存在于功能组件之间、保证组件之间，以及功能和保证组件之间。

组件间的依赖关系描述是 CC 组件定义的一部分。为了保证达到 TOE 要求的完备性，把组件加入到适当的 PP 和 ST 中时，应满足相应的依赖关系。

2) 组件允许的操作

CC 组件可以像在 CC 中定义一样来使用，或者通过使用组件允许的操作，对组件进行裁剪，以满足特定的安全策略或对付特定的威胁。每一个 CC 组件标识并定义了组件允许的“赋值”和“选择”操作、在哪些情况下可对组件使用这些操作以及使用这些操作的后果。任何一个组件均允许“反复”和“细化”操作。这四个操作如下：

a) 反复：在不同操作时，允许组件多次使用；

b) 赋值：当组件被应用时，允许规定所赋予的参数；

c) 选择：允许从组件给出的列表中选定若干项；

d) 细化：当组件被应用时，允许对组件增加细节。

一些需要的操作可以在 PP 内完成(整体或部分地)，或者留在 ST 内完成，不过所有操作必须在 ST 内完成。

4. CC 中安全需求的描述方法

CC 定义了三种类型的需求结构：包、PP 和 ST。

关注信息系统或产品的人员可以粗略比分作三类：消费者、开发者与评估者。借助这三类人员的不同使命，可以说明 PP 与 ST 的作用。

CC 为所有的安全需求提供了一种通用的组件库，即根据具体的安全需求，从库中选择所需的组件——安全功能组件与安全保证组件。因此，可以说 CC 是知识库，是材料基地，包含了常用的安全技术。但一个具体的信息系统或产品有特定的安全需求，这些安全需求只是知识库的子集，如何将从知识库中选取的组件子集合理、有序地表达出来，使这种表达能够符合既定的安全目标，需要一种描述方式，PP 与 ST 正是这种描述方式。因此，如果说 CC 是一个完备的字典，则 PP 与 ST 就是组织字词的语法规则，按照这种规则组织字词所形成的语言，能充分表达自己的需求，并使这些需求覆盖既定的安全目标。

一般说来，PP 是消费者表达自己需求的描述方式，针对的是一类 TOEs，而 ST 是开发者表达自己方案的描述方式，针对具体的(Identified)TOE。因此，PP 与安全功能的实现无关，它回答的问题是“我们在安全方案中需要什么”，是“目标”的说明；ST 依赖于实现，回答的问题是“我们在解决方案中提供了什么(提供了什么安全措施)”，是“建造”的说明。CC 对 PP 与 ST 描述的格式与内容表述均做出了结构化的规定。

(1) 包

组件的中间组合被称为包。包是对功能或保证需求集合的描述，这个集合能够满足一个安全目标的可标识子集。包可重复使用，可用来定义那些公认有用的、能够有效满足特定安全目标的要求。包可用于构造更大的包、PP 和 ST。

评估保证级别(EAL)，是在 CC 第三部分中预先定义的保证包。一个保证级别是评估中保证要求的基线集。每个评估保证级别定义一套一致的保证要求，合起来构成一个预定义 CC 保证级别尺度。

(2) 保护轮廓 PP

保护轮廓(PP)，包含一套来自 CC(或明确阐述)的安全要求，它应包括一个评估保证级别(EAL)(可能增加附加的保证组件)。PP 是关于一系列满足一个安全目标集的 TOE 的、与实现无关的描述。PP 可反复使用，还可用来定义那些公认有用的、能够有效满足特定安全目标的 TOE 要求。PP 也包括安全目的和安全要求的基本原理。

PP 的开发者可以是用户团体、IT 产品开发者或其他对定义这样一系列通用要求有兴趣的团体。PP 为消费者提供了一套参照特定安全需求集的方法以及进一步评估这些安全需求的措施。

PP 的描述结构如图 14.7 所示。

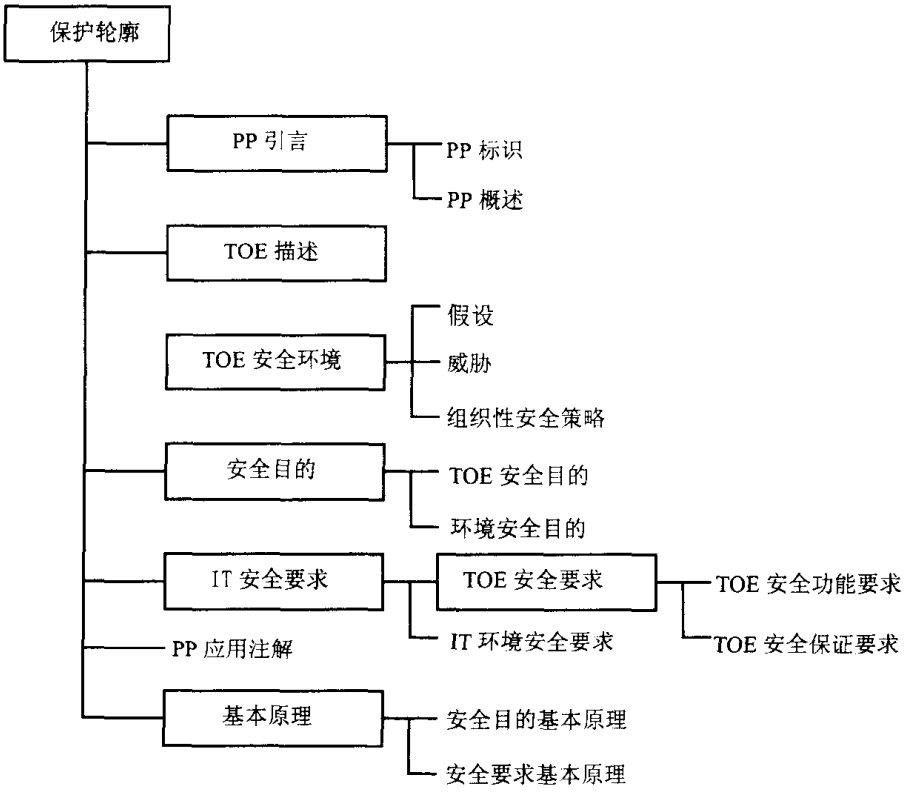


图 14.7 保护轮廓 PP 的描述结构

(3) 安全目标 ST

安全目标(ST)包括一系列安全要求，这些要求可以引用 PP，也可以直接引用或明确说明 CC 中的功能或保证组件。ST 是针对特定 TOE 安全要求的描述，通过评估可以证明这些安全要求对满足指定目的是有用和有效的。

一个 ST 包含 TOE 的概要规范，同时还包括安全要求、目的以及它们的基本原理。ST 是所有团体就 TOE 应提供何种安全性达成一致的基础。

安全目标的描述结构如图 14.8 所示。

5. CC 框架下的评估类型

(1) PP 评估

PP 评估是依照 CC 第三部分的 PP 评估准则进行的。这样的评估目标是为了证明 PP 是完备的、一致的、技术合理的，同时适于作为一个可评估 TOE 的安全要求的声明。

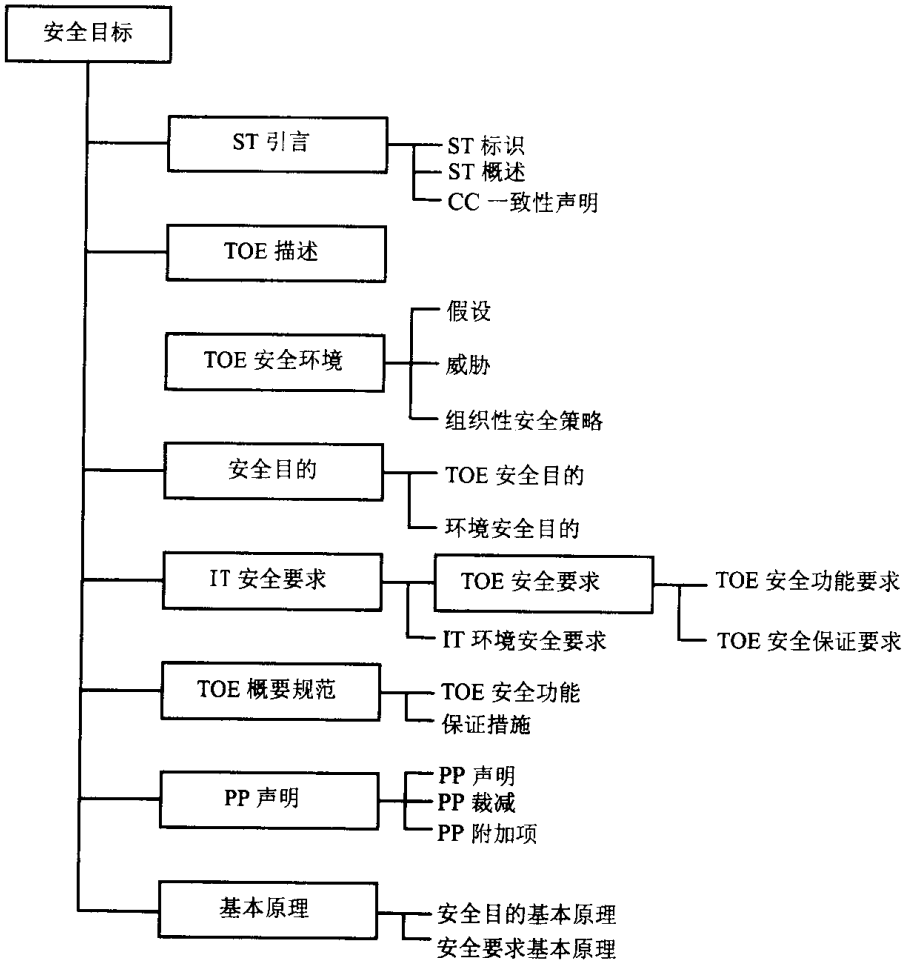


图 14.8　安全目标描述结构

(2) ST 评估

针对 TOE 的 ST 评估是依照 CC 第三部分的 ST 评估准则进行的。ST 评估具有双重目标：首先是为了证明 ST 是完备的、一致的、技术合理的，而且适合于用作相应 TOE 评估的基础；其次，当某一 ST 宣称与某一 PP 一致时，用来证明 ST 满足该 PP 的要求。

(3) TOE 评估

TOE 评估，使用一个已评估过的 ST 作为基础，依照 CC 第三部分的评估准则来进行。这样一个评估的目标是为了证明 TOE 满足 ST 中的安全要求。

以上三种评估的关系如图 14.9 所示。

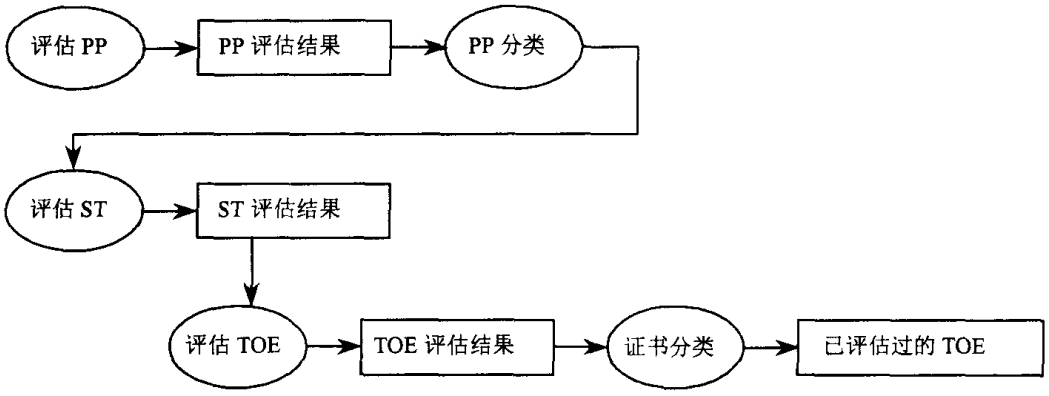


图 14.9 评估结果

6. CC 安全要求

CC 的第二部分是安全功能要求，这对满足安全需求的诸安全功能提出了详细的要求。另外，如果有超出第二部分的安全功能要求，开发者可以根据“类—族—组件—元素”的描述结构表达其安全要求，并附加在其 ST 中。详细了解各功能类，需要仔细阅读 CC 并在实际中加以应用。CC 共包含 11 个安全功能类分别是：

(1) FAU 类：安全审计；
(2) FCO 类：通信；
(3) FCS 类：密码支持；
(4) FDP 类：用户数据保护；
(5) FIA 类：标识与鉴别；
(6) FMT 类：安全管理；
(7) FPR 类：隐秘；
(8) FPT 类：TFS 保护；
(9) FAU 类：资源利用；
(10) FTA 类：TOE 访问；
(11) FTP 类：可信信道/路径。

CC 的第三部分是评估方法，提出了 PP、ST、TOE 三种评估，共 10 个类，但其中的 APE 类与 ASE 类分别介绍了 PP 与 ST 的描述结构及其评估准则；对于已评估过的受测系统或产品，保证维护类 AVA 旨在保证当运行环境发生变化后，TOE 仍然能够满足安全目标。因此，只有 7 个安全保证类是 TOE 的评估类别，分别是：

(1) ACM 类：配置管理；

(2) ADO 类：分发与操作；
(3) ADV 类：开发；
(4) AGD 类：指导性文档；
(5) ALC 类：生命周期支持；
(6) ATE 类：测试；
(7) AVA 类：脆弱性评定。

7. CC 认可协议

1998 年 1 月，经过两年的密切协商，来自美国、加拿大、法国、德国以及英国的政府组织签订了历史性的安全评估互认协议：《IT 安全领域内 CC 认可协议》。根据该协议，在协议签署国范围内，在某个国家进行的基于 CC 的安全评估将在其他国家内得到承认。对 IT 产品及保护轮廓的安全评估来说，此协议的签订代表着该领域的一个重要进步，该协议的参与者在这个领域内有着共同的目的，即：

(1) 确保 IT 产品及保护轮廓的评估遵循一致的标准，为这些产品及保护轮廓的安全提供足够的信心；

(2) 在国际范围内提高那些经过评估的、安全性增强的 IT 产品及保护轮廓的可用性；

(3) 消除 IT 产品及保护轮廓的重复评估，改进安全评估的效率及成本，改进 IT 产品及保护轮廓的证明/确认过程。

根据 CC 认可协议，已经获得 CC 证书的 IT 产品及保护轮廓在使用前不必再经过评估及证明/确认。协议中规定了在何种情况下，协议方将接受或承认其他协议方进行的 IT 安全评估及相关证明/确认。由签约各方代表组成的管理委员会将负责有关该协议的执行及其他相关事务。目前该加入该协议的国家有：澳大利亚、新西兰、加拿大、芬兰、法国、德国、希腊、以色列、意大利、荷兰、挪威、西班牙、英国及美国等。

14.3　国内安全标准

14.3.1　GB17859-1999

在国内，我国制定了强制性国家标准《计算机信息系统安全保护等级划分准则》(GB17859-1999)。该准则于 1999 年 9 月 13 日由国家质量技术监督局发布，于 2001 年 1 月 1 日起实施。《计算机信息系统安全保护等级划分准则》是建立安全等级保护制度，实施安全等级管理的重要基础性标准。它将计算机信息系统安全保护等级划分为五个级别，

通过规范、科学和公正的评定和监督管理，为计算机信息系统安全等级保护管理法规的制定和执法部门的监督检查提供依据，为计算机信息系统安全产品的研制提供技术支持，同时为安全系统的建设和管理提供技术指导。

该标准规定了计算机系统安全保护能力的五个等级，计算机信息系统安全保护能力随着安全保护等级的提升而增强。

目前，该标准的 20 多个配套标准正在制定和修订中。

1. 等级划分的思想

GB17859-1999 划分安全等级的思路，始终围绕着信息系统主客体间访问过程中安全机制所提供的不同安全功能和保障来进行的。在该标准中，要求 TCB 提供的安全功能和保证共有十类：自主访问控制、身份鉴别、数据完整性、客体重用、审计、标记、强制访问控制、隐蔽信道分析、可信路径、可信恢复。这些安全功能和保证在主体和客体间的组合关系如图 14.10 所示。

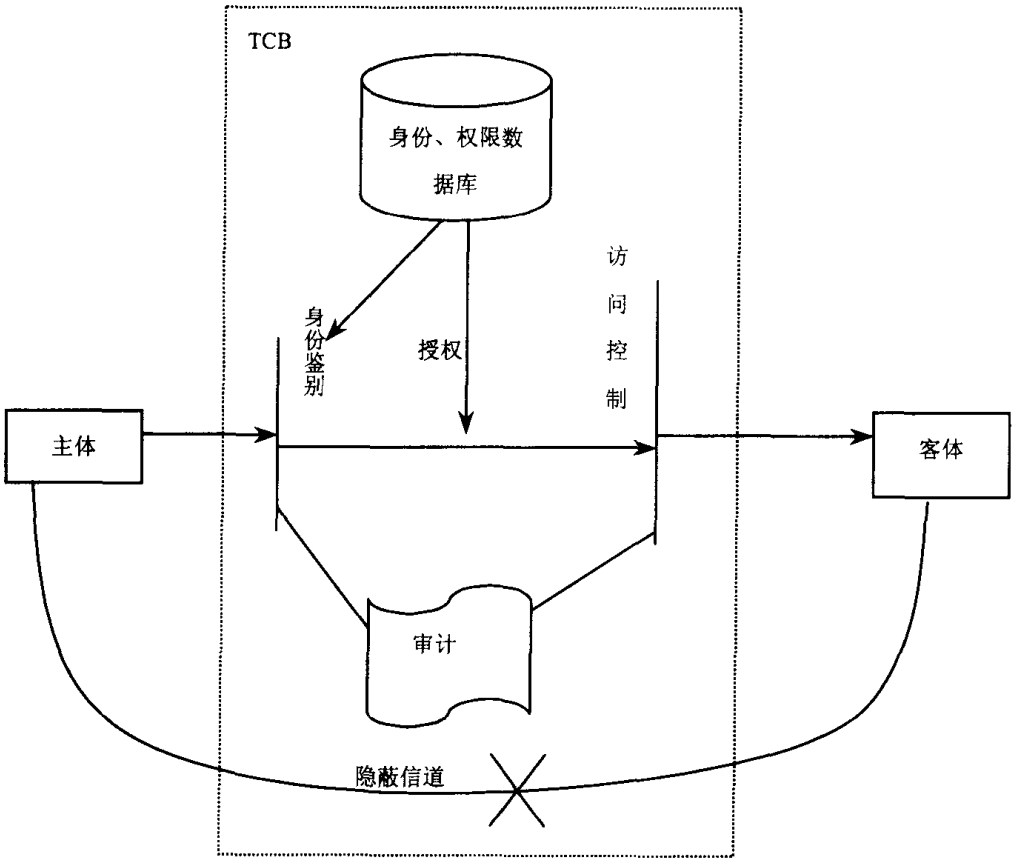


图 14.10 TCB 安全模型

不同的安全等级要求 TCB 提供不同的安全功能和保障强度，各级间的安全强度差别简要总结如表 14.4 所示。

表 14.4　信息系统安全等级划分递进关系

级别 / 要求	第一级：用户自主保护级	第二级：系统审计保护级	第三级：安全标记保护级	第四级：结构化保护级	第五级：访问验证保护级
自主访问控制	用户以用户(组)为单位控制	以单用户为单位控制，并能阻止权限扩散	同左	同左	同左
身份鉴别	能标识和鉴别用户身份	能惟一标识用户，并可用于审计	TCB 维护授权数据	同左	同左
数据完整性	自主完整性策略，阻止非授权用户修改或破坏敏感信息	同左	在传输中用敏感标记校验	同左	同左
审计		能审计初始化、删除、安全管理等事件	可审计客体安全级别，并审计更改可读输出记号	可审计隐蔽存储信道事件	能监控积累，超过阈值告警或终止
客体重用		在指定、分配客体前清除残留信息	同左	同左	同左
标记			维护与主体及其控制的存储客体相关的敏感标记	对所有直接或间接可访问资源进行标记	同左
强制访问控制			对所有主体及其所控制的客体实施强制访问控制	对外部主体能直接或间接访问的所有客体实施强制访问控制	同左
隐蔽信道分析				搜索隐蔽存储信道、测量带宽	同左
可信路径				TCB 与发起用户间建立可信路径	可信路径逻辑隔离
可信恢复					能无损地恢复安全保护性能

下面简要介绍各级的主要思想。

2. 第一级　用户自主保护级

本级的计算机信息系统可信计算基通过隔离用户和数据，使用户具备自主安全保护的能力。它具有多种形式的控制能力，对用户实施访问控制，即为用户提供可行的手段，保护用户和用户组的信息，避免其他用户对数据非法读写与破坏。

3. 第二级　系统审计保护级

与用户自主保护级相比，本级的计算机信息系统可信计算基实施了粒度更细的自主访

问控制，它通过登录规程、审计安全性相关事件和隔离资源，使用户对自己的行为负责。

4. 第三级 安全标记保护级

本级的计算机信息系统可信计算基具有系统审计保护级的所有功能。此外，还提供有关安全策略模型、数据标记以及主体对客体强制访问控制的非形式化描述，能够准确地标记输出信息的能力，并消除通过测试发现的任何错误。

5. 第四级 结构化保护级

本级的计算机信息系统可信计算基建立于一个明确定义的形式化安全策略模型之上，它要求将第三级系统中的自主和强制访问控制扩展到所有主体与客体，此外还要考虑隐蔽通道。本级的计算机信息系统可信计算基必须结构化为关键保护元素和非关键保护元素。计算机信息系统可信计算基的接口也必须明确定义，使其设计与实现能经受更充分的测试和更完整的复审。本级加强了鉴别机制，支持系统管理员和操作员的职能，提供可信设施管理，增强了配置管理控制。总之，系统具有了相当的抗渗透能力。

6. 第五级 访问验证保护级

本级的计算机信息系统可信计算基满足访问监控器需求。访问监控器仲裁主体对客体的全部访问。访问监控器本身是抗篡改的，同时必须足够小，能够分析和测试。为了满足访问监控器需求，计算机信息系统可信计算基在构造时，务必排除那些对实施安全策略来说并非必要的代码，在设计和实现时，应从系统工程角度将其复杂性降低到最小程度。此外，它还支持安全管理员职能，扩充了审计机制，当发生与安全相关的事件时发出信号，并提供系统恢复机制。从而，系统具有很高的抗渗透能力。

14.3.2 GB/T15408

国家技术监督局还发布了推荐标准 GB/T15408,该标准是 CC 的等同采用版本，可直接参考 CC 的有关介绍。

习 题 十 四

1. 简述国际安全评估标准的发展历程。
2. TCSEC 如何划分安全等级？
3. C 级安全和 B 级安全的主要区别是什么？

4. CC 的评估思想是什么？
5. CC 如何描述安全功能和安全保障？
6. PP 和 ST 的作用是什么？有何区别？
7. CC 为什么只对安全保障进行分级，而没有对安全功能分级？
8. 简述 GB17859 的主要思想。

第十五章　应 用 安 全

应用系统的安全是安全建设最主要的目的。因为信息总是通过应用系统来存取的，所以，应用系统的安全是确保信息安全的根本。本章介绍常见应用的安全，包括 Web、Email 和电子商务的安全问题。

15.1　Web 安全

由于 Web 业务是开放的交互业务，而且其实现支撑平台也越来越复杂，因此 Web 业务的安全性面临着比其他业务更大的挑战。

15.1.1　Web 安全目标

Web 安全的主要目标如下：

(1) 服务器安全

- 保护服务器不被非授权者访问、不被篡改或阻塞；
- 保护服务软件不访问系统文件，从而避免引起系统混乱；
- 只允许授权用户访问 Web 发布的信息；
- 确保用户上载的数据安全可靠。

(2) 传输安全

确保重要的 Web 服务信息在传输中不被窃听和篡改。

(3) 客户机安全

确保浏览器不被恶意代码侵袭，尤其是不受到病毒和木马的侵袭。

15.1.2　Web 安全措施

针对以上安全目标，目前采取的防护措施主要有以下几类：

(1) 定期扫描加固

Web 服务器被入侵通常是由于存在漏洞，其中大部分漏洞都是已知的，并且有相应的补丁程序发布。因此，只要采用专门的漏洞扫描系统定期进行扫描，发现漏洞打上补丁并进行加固，就可以避免很多威胁。

(2) 安装 Web 服务器保护系统

目前有很多公司开发了专用的 Web 服务器保护软件，这些软件通过控制 Web 服务器的 I/O 操作或定时检查主页的完整性来防止 Web 页面被篡改。

(3) 采用完善的认证和加密措施

对于重要的业务系统，必须对每一位访问者进行足够强度的身份认证，并对传输信息进行加密。在这方面，前面介绍过的 IPsec 协议和 SSL 协议都可以提供相应的措施，尤其是 SSL 协议主要是保护 Web 服务的。还有安全 http(s_http)协议，是专门针对 http 的安全而提出来的，它通过在客户端设立本地代理与服务器建立安全通道，实现加密和认证。由于 s-http 应用不是很广泛，这里不再做专门介绍。

(4) 设立代理服务器

在真正的 web 服务器前设立代理服务器，对 web 请求进行分析，只有那些合理的、安全的请求才被转发到真正的服务器，也是保护 Web 服务器的一项有效措施。代理服务器可以只检查和转发 Web 的请求和响应，也可以在真实服务器和代理服务器间建立映射关系，将真实服务器隐藏。

(5) 谨慎设置浏览器安全选项

客户端浏览器的安全也是很重要的，除了在与服务器连接时采用 SSL 等协议进行认证，正确设置浏览器自身的安全选项也很重要，如谨慎使用 Active-X 控件，java 脚本，不随意执行下载的程序等等措施，都有利于提高安全性。

15.2　Email 安全

E-mail 是网络中最广泛的应用之一，除了常见的安全因素外，Email 还涉及个人隐私等敏感问题，因此其安全性备受关注。

15.2.1　Email 系统组成

Email 系统主要由邮件分发代理、邮件传输代理、邮件用户代理及邮件工作站组成。各部分作用如图 15.1 所示。

(1) 邮件分发代理 MDA：负责将邮件数据库中的邮件分发到用户的邮箱中。在分发邮件时，MDA 还承担邮件自动过滤、邮件自动回复和邮件自动触发等任务。常见的 MDA 开放原代码程序有 binmail 和 promail 等。

(2) 邮件传输代理 MTA：负责邮件的接收和发送，通常采用 SMTP 协议传输邮件。常见的 MTA 程序有 sendmail、qmail 和 Postfix 等。

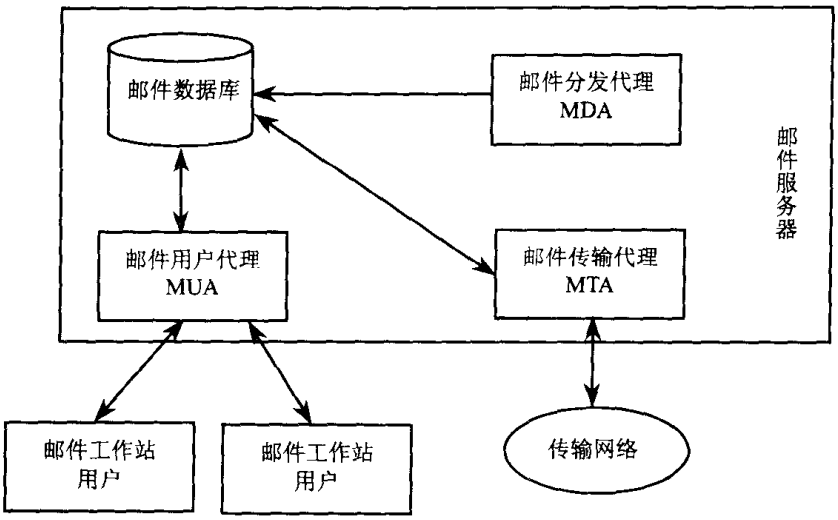


图 15.1 Email 系统组成

(3) 邮件用户代理 MUA：MUA 并不接收邮件，而是负责将邮箱中的邮件显示给用户。MUA 常用的协议有 POP3 和 IMAP，常见的程序有 pine、kmail 等。

(4) 邮件工作站：是邮件用户直接操作的计算机，负责显示、撰写邮件等。

15.2.2 Email 安全目标

根据邮件系统的组成，可以将邮件安全的目标总结如下：

(1) 邮件分发安全：邮件分发时，可能遇到垃圾邮件、邮件病毒、开放转发等威胁，所以邮件分发安全应能阻止垃圾邮件和开放转发，并查杀已知病毒。

(2) 邮件传输安全：邮件在传输过程中可能被窃听、篡改，因此必须保障邮件传输的机密性和完整性。同时，邮件在传输中采用 SMTP 协议，该协议允许远程查询邮件账户，因此，如何在高安全要求的系统中保护邮件账户的状态(如存在、可用等)也是安全的目标。

(3) 邮件用户安全：邮件用户通过工作站，采用 POP3 或 IMAP 等协议浏览邮件，这个过程中需要确认用户的身份，否则将导致邮件被非授权访问。同时，邮件在用户工作站上显示时，可能需要在本地执行显示软件，因而容易使病毒或其他有害代码发作。所以在工作站端，也要能支持病毒查杀功能。

15.2.3 Email 安全措施

针对以上安全目标，常用的安全措施如下：

(1) 身份认证：包括邮件转发认证、邮件收发认证等。即在要求转发邮件时，必须经过认证，而不是开放转发。而在用户要求接收或发送邮件时，必须经过身份认证，以避免

邮件在邮箱中被窃取。要特别强调的是，认证的口令要有足够强度，以防在线口令被破解。

(2) 加密、签名：在传输中，必须采用加密和签名措施来保障重要邮件的机密性和完整性。目前，电子邮件已渐渐成为商务信函的重要形式，因此，必要时还要进行发送和接收签名，以防止否认。在这方面，已有成熟的安全协议 PGP 和 S/MIME 等，将在后面详细介绍。

(3) 协议过滤：为了防止邮件账号远程查询，要对 SMTP 的协议应答进行处理，如对 VERY、EXPN 等命令不予应答，或无信息应答。

(4) 设立防火墙：经常设立内、外邮件服务器，在内、外服务器间设立防火墙。外服务器负责对外邮件的传输收发，而内服务器才是真正的用户邮件服务器。所有来自公网上的邮件操作均止于外服务器，再由外服务器转发。这样可以将真正的邮箱和公网隔离。

(5) 安装邮件病毒过滤系统：在邮件服务器上安装邮件病毒过滤软件，使大部分邮件病毒在邮件分发时被分检过滤。同时在邮件客户端也安装防病毒软件，在邮件打开显示前查杀病毒。

15.2.4 PGP

PGP 是采用对称加密算法和非对称加密算法相结合来对电子邮件提供安全服务的协议，它通过加密、签名和认证来保护邮件内容的完整性、机密性和抗抵赖性。

PGP 是 Phil Zimmermann 的贡献，于 1991 年推出了 1.0 版，1994 年推出了 2.6 版，目前最新的版本是 7.1 版。现在 PGP 已经得到了广泛的应用，其成功的主要原因在于：

(1) 支持版本多：PGP 支持各种系统平台和不同商业版本。

(2) 选择众所周知的算法，避免算法的安全性争议：如公钥加密，包括 RSA、DSS、Diffie-Hellman，对称加密包括 CAST-128、IDEA、3DES、AES，以及 SHA-1 散列算法等。

(3) 适用性强，既可用于机构，也可用于个人。

(4) 可自主使用，不受政府或标准化组织所控制。

1. PGP 的功能

PGP 的主要功能就是对邮件进行加密、签名和认证，具体实现特点如表 15.1 所示。

表 15.1　PGP 的功能

服　务	采用的算法	说　明
数字签名	DSS/SHA 或RSA/SHA	用SHA-1创建散列码，用发送者的私钥和DSS或RSA加密消息摘要
消息加密	CAST 或 IDEA 或 3DES、AES 及 RSA 或 D-F	消息用一次性会话密钥加密，会话密钥用接收方的公钥加密
压缩	ZIP	消息用 ZIP 算法压缩
邮件兼容性	Radix 64	邮件应用完全透明，加密后的消息用 Radix 64 转换
数据分段		为了适应邮件的大小限制，PGP 支持分段和重组

2. PGP 运行流程

为了描述 PGP 的运行过程，先定义以下符号：

K_s	:	session key
KR_a	:	用户 A 的私钥
KU_a	:	用户 A 的公钥
EP	:	公钥加密
DP	:	公钥解密
EC	:	常规加密
DC	:	常规加密
H	:	散列函数
$\|\|$	:	连接
Z	:	用 ZIP 算法数据压缩
R64	:	用 radix64 转换到 ASCII 格式

(1) 认证

PGP 认证的步骤如图 15.2 所示。

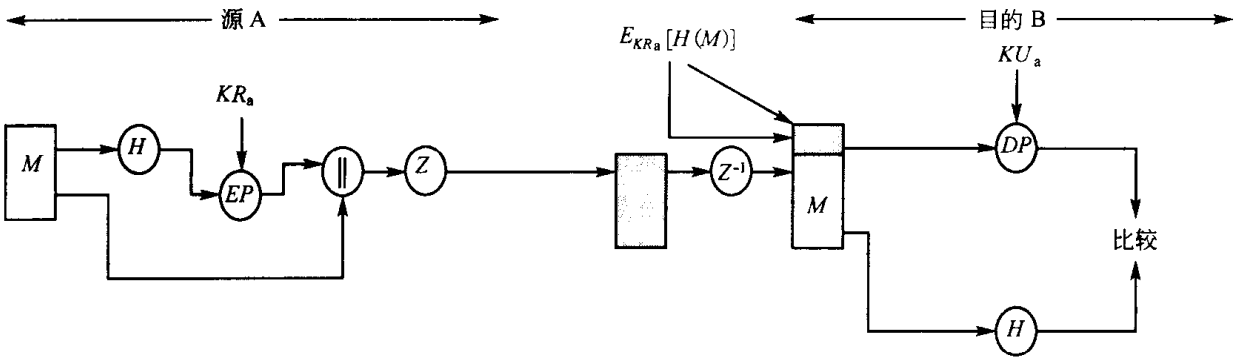


图 15.2 PGP 认证过程

1) 发送方生成报文 M；

2) 使用 SHA-1 生成 M 的一个 160 位的散列码 H；

3) 使用发送者的私钥，采用 RSA 算法对 H 加密，并与 M 连接；

4) 接收方使用发送者的公钥，采用 RSA 算法解密，并恢复散列码 H；

5) 接收方对报文 M 生成一个新的散列码与 H 比较，如果一致，则报文 M 通过认证。

身份认证中，RSA 算法的强度确保了发送方的身份，SHA-1 的强度保证了签名的有效

性。此外，签名也可以采用 DSS/SHA-1 替代方案。

(2) 加密

邮件加密流程如图 15.3 所示。

1) 发送方生成报文 M 并为该报文生成一个随机数作为会话密钥；

2) 采用 CAST-128(或 IDEA 或 3DES)算法，用会话密钥加密 M；

3) 采用 RSA 算法，用接收者的公钥加密会话密钥，并与消息 M 结合；

4) 接收方采用 RSA 算法，用自己的私钥解密恢复会话密钥；

5) 接收方使用会话密钥解密恢复消息 M。

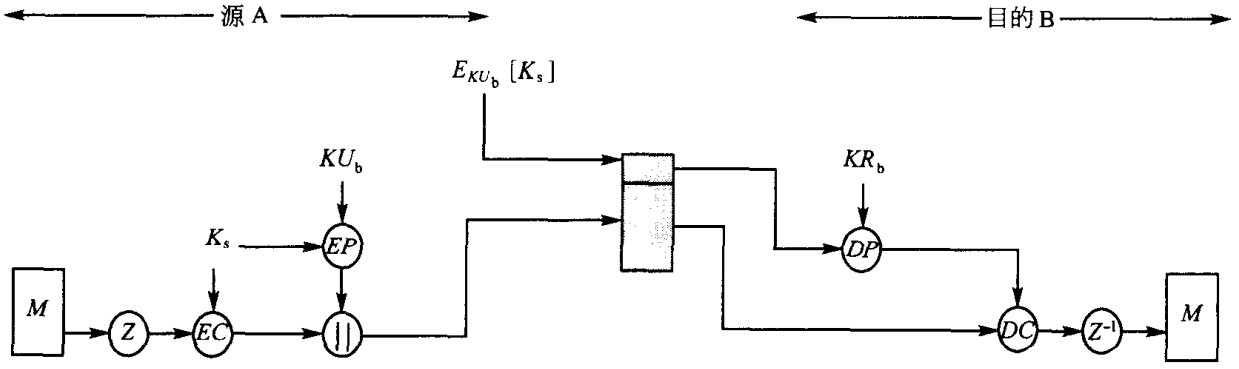


图 15.3　PGP 加密过程

在 PGP 的加密流程中，通过用称密算法加密报文和用非对称算法交换密钥，较好地提高了性能，也解决了邮件作为非实时通信系统中单向密钥交换的问题；同时，每次消息加密都采用一次性会话密钥，进一步增强了保密的强度。

当同时需要加密和认证两种服务时，发送者先用自己的私钥签名，然后用会话密钥加密消息，再用接收者的公钥加密会话密钥，如图 15.4 所示。

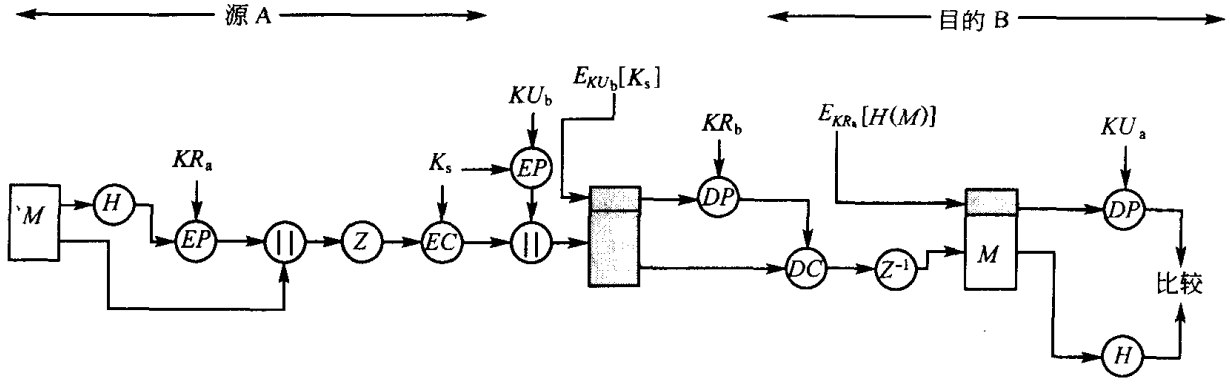


图 15.4　PGP 加密和认证过程

(3) 数据压缩

在 PGP 中，先对报文签名再压缩，然后对压缩后的报文进行加密。压缩有利于节省邮件传输和存储的空间。之所以先对报文签名然后再压缩，是因为这样不需要为检验签名而保留压缩版本的消息，而且为了检验而做压缩不能保证一致性，压缩算法不同，实现版本可能会产生不同的结果。而压缩之后再做加密，原因在于，压缩后的消息冗余小，增加密码分析的难度，相反若先加密，则压缩难以见效。

(4) 兼容性

PGP 处理后的报文，部分或者全部是加密后的字节流，为任意的 8 位字节。某些邮件系统只允许使用 ASCII 字符组成的块，所以 PGP 采用了 Radix-64 转换方案提供了转换到 ASCII 格式的功能。

(5) 分段与重组

电子邮件设备对报文通常有最大长度的限制，因此 PGP 提供了自动处理的功能，发送方将超过限制的报文分割成合适的长度，再进行发送；接收方则自动将分割好的报文重新装配完整。

3. PGP 密钥管理

(1) PGP 密钥需求

PGP 使用四种类型的密钥：一次性会话传统密钥、公钥、私钥以及基于口令的传统密钥。

1) 会话密钥的生成

为了生成不可预知的会话密钥，PGP 使用了一种复杂的随机密钥生成算法(一定的真随机性)。以 CAST-128 为例，128 位的随机数是由 CAST-128 自己生成的，CAST128 以一个 128 位的密钥加密两个 64 位的明文数据块，使用 CFB 方式，加密结果产生两个 64 位的密文数据块，这两个数据块的结合构成 128 位的会话密钥(算法基于 ANSI X12.17)。其中，作为明文被加密的两个 64 位明文数据块，是从一个 128 位的随机数流中导出的。这些随机数流基于用户的键盘击键时间和输入内容来产生。

2) 密钥标识符

一般来说，用户都拥有多个公钥/私钥对，为了辨别不同的公钥/私钥对，PGP 给每个公开密钥定义了一个密钥标识符(密钥 ID),即公开密钥的低 64 位。这样通过密钥 ID，接收方就可以知道发送方使用的是哪一对公钥/私钥。

因此，一个完整的 PGP 报文由三个部分组成：消息段、签名(可选)段和会话密钥段(可选)。具体格式如图 15.5 所示。

其中，消息段包括要存储或传输的实际数据、文件名以及说明创建时间的时间戳；签

名段包括消息摘要、消息摘要的前两个 8 位组、发送方公开密钥的密钥 ID 和签名构造的时间戳；会话密钥段包括会话密钥和接收方公开密钥的密钥 ID。

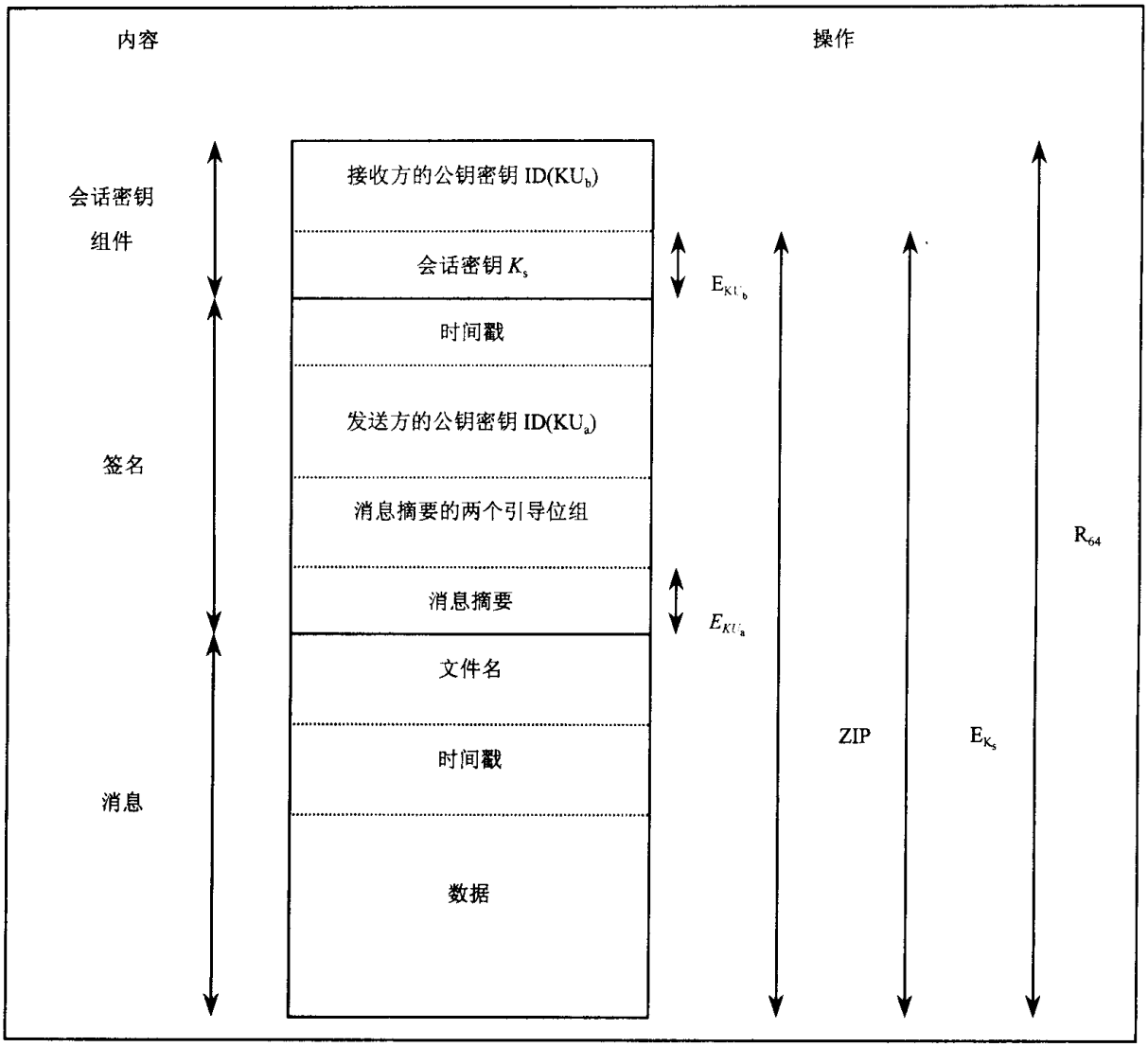


图 15.5　PGP 报文结构

(2) 密钥环

为了管理密钥 ID，PGP 在每个节点创建了一对数据结构，一个用来存储该节点所拥有的公钥/私钥对，称为私钥环，另一个用来存储该节点所知的其他节点的公钥，称为公钥环，如图 15.6 所示。

1) 私钥环

私钥环

时间戳	密钥ID	公开密钥	加密了的私有密钥	用户ID
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•
T_i	$KU_i \bmod 2^{64}$	KU_i	$E_{H(P_i)}[KR_i]$	用户i
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•

公钥环

时间戳	密钥ID	公开密钥	属主信用	用户ID	密钥合法性	签名 (s)	签名确认(s)
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
T_i	$KU_i \bmod 2^{64}$	KU_i	trust_flag_i	用户i	trust_flag_i		
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•

图 15.6 PGP 密钥环结构

私钥环中所包含的数据项有时间戳、密钥 ID、公开密钥、私有密钥以及用户 ID。私有密钥可以通过用户 ID 或密钥 ID 查找。为了保证私有密钥的安全性，PGP 采用以下方式保存私有密钥：

a) 用户选择一个口令短语用于加密私钥。

b) 当系统用 RSA 生成一个新的公钥/私钥对时，要求用户输入口令，使用 SHA-1 对该口令生成一个 160 位的散列码，然后销毁口令。

c) 系统用散列码中，128 位作为密钥用 CAST-128 加密私钥，然后销毁这个散列码，并将加密后的私钥存储到私钥环中。

d) 当用户要访问私钥环中的私钥时，必须提供口令。PGP 将取出加密后的私钥，生成散列码，解密私钥。

2) 公钥环

公钥环中所包含的主要数据项有时间戳、密钥 ID、公开密钥和用户 ID。公钥环可以通过用户 ID 或密钥 ID 进行查找。

3) 密钥环的应用

在发送方签名和加密消息、接收方解密和验证消息的过程中，密钥环的应用流程如图

15.7 所示。

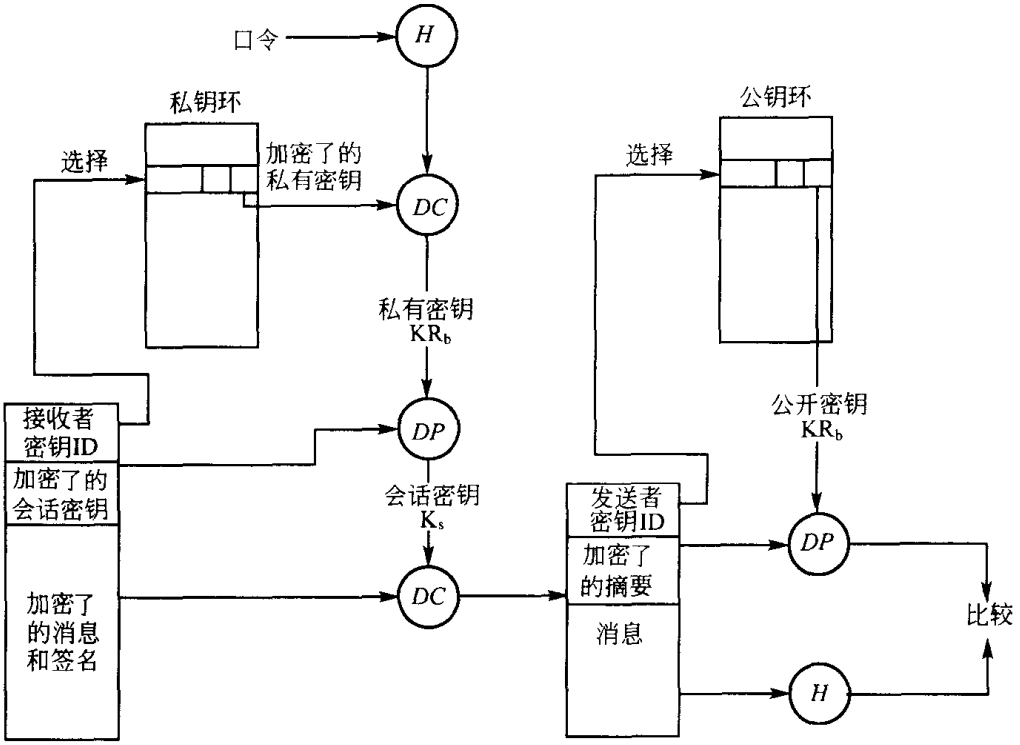


图 15.7　密钥环的应用流程

其中发送方处理消息过程如下：

- 消息签名：

1) 从私钥环中得到私钥，利用 user_id 作为索引；

2) PGP 提示输入口令，恢复私钥；

3) 构造签名部分。

- 消息加密：

1) PGP 产生一个会话密钥，并加密消息；

2) PGP 用接收者 use_id 从公钥环中获取其公钥；

3) 构造消息的会话密钥部分。

接收方处理消息过程如下：

- 消息解密：

1) PGP 用会话密钥段中的 Key_ID 为索引，从私钥环中获取私钥；

2) PGP 提示输入口令短语，恢复私钥；

3) PGP 恢复会话密钥，并解密消息。

• 消息验证：

1) PGP 用签名段中的 Key_ID 作为索引，从公钥环中获取发送者的公钥；

2) PGP 恢复消息摘要；

3) PGP 对收到的消息作摘要，并与上一步的结果作比较。

(3) 公开密钥管理

公开密钥算法在 PGP 应用中起到了非常关键的作用，为了避免用户使用公钥时受欺骗，通常可以采用的方法有：

1) 直接获取公钥，如通过软盘拷贝。这种方式可靠，但存在一定局限性。

2) 通过电话验证公钥的合法性。

3) 从双方都信任的第三方获得公钥。

4) 从一个信任的 CA 中心得到公钥。

由于 PGP 注重在正式或非正式环境下广泛应用，没有建立严格的公钥管理模式。尽管 PGP 没有包含任何建立认证权威机构或建立信任体系的规范，但它提供了一个利用信任关系的方法，将信任关系与公钥联系起来。

每个公钥有三个相关的属性：

• 合法性字段(Key legitimacy field)：表明 PGP 对“此用户公钥是合法的”的信任程度。信任级别越高，这个 userID 与该公钥的绑定越强。这个字段由 PGP 计算。

• 签名信任字段(Signature trust field)：每一个公钥项都有一个或者多个签名，这是公钥环主人收集到的、能够认证该公钥项的签名。每一个签名与一个签名信任字段关联，表明这个 PGP 用户对“签名人对公钥签名”的信任程度。合法性字段是由多个签名信任字段导出的。

• 属主信任字段(Owner trust field)：表明该公钥被用于对其他公钥证书签名时的信任程度，这个信任程度由用户给出。

信任处理的过程如下：

1) 当 A 向公钥环中插入一个新公钥时，PGP 必须向 trust flag 赋值，该标志与该公钥的主人相关。如果主人是 A，则该值为最高信任(ultimate trust)。否则，PGP 询问用户，让用户给出信任级别。用户可以选择如下级别：该公钥的主人是不认识(unknown)、不信任(untrusted)、接近信任(marginally trusted)或完全信任(complete trusted)

2) 当新公钥插入时，可能有一个或多个签名跟随其后。许多签名可以随后再加入。当一个签名插入到一个条目中时，PGP 查找该公钥环中公钥主人是否已经包含了该公钥的作者。如果包含，则这个主人的 ownertrust 值被赋予该签名的 sigtrust 字段。否则，赋予 unknown user 值。

3) key legitimacy field 的值根据该条目中 signature trust field 来计算。如果至少有一个

签名的信任值为 ultimate，则 key legitimacy field 的值设为 complete。否则，PGP 将计算一个信任值的权重和。

需要指出的是，一个密钥被认为是可信任的，并不意味着由它所签名的其他密钥同样可信任。

最后，为了处理密钥泄漏或定时更新，必须注销公开密钥。通常的注销方法是由私钥所有者签发一个密钥注销证书，私钥所有者应尽可能快和尽可能广地散布这个证书，以使有关人员更新他们的公钥环。

15.2.5 S/MIME

S/MIME(Secure/Multipurpose Internet Mail Extension)是基于密码学的诸多成果对 MIME Internet 电子邮件格式的安全扩充。S/MIME 与 PKI 结合，采用了 X.509 证书以及 PKCS 标识。S/MIME 从一般功能上讲与 PGP 非常相似，同样提供对报文进行签名和加密的功能。但 S/MIME 很有可能在未来成为商业或组织使用的工业标准，而 PGP 更面向个人用户。

1. S/MIME 主要功能

S/MIME 的主要功能包括：

• 数据封装(Enveloped data)：对邮件加密，并附上一个或多个接收者的加密密钥，形成封装。

• 数据签名(Signed data)：对邮件内容作摘要，然后用签名者的私钥对摘要加密，从而形成一个数字签名。邮件内容与签名使用 radix-64 编码方式进行编码，此时，邮件内容只能被使用 S/MIME 的接收者查看。

• 明文签名(Clear-signed data)：只有签名部分用 radix-64 方式编码，即使接收者没有 S/MIME 能力，也能查看报文内容，只是不能验证该签名。

• 数据签名和封装(Signed and enveloped data)：签名和加密相结合，加密数据被签名或者签名数据被加密。

2. S/MIME 加密算法

S/MIME 使用的主要加密算法如表 15.2 所示。

发送代理必须决定接收代理能否对给定的加密算法进行解密，如果接收代理只能接收弱的加密内容，则发送代理必须决定使用强加密算法能否接收。发送代理应该依次遵循以下规则：

(1) 如果发送代理获得了接收代理的解密能力表，则应该选择表中最为安全的算法；

(2) 如果发送代理没有接收代理的解密能力表，但曾经接收过接收代理发送的报文，则应该使用与最新的接收报文相同的算法；

表 15.2 S/MIME 使用的加密算法

功 能	需 求
创建报文摘要	必须支持 SHA-1 和 MD5
	应该使用 SHA-1
对报文摘要进行数字签名	发送和接收代理必须支持 DSS
	发送代理应该支持 RSA
	接收代理应该支持使用长度为 512－1024 比特的密钥验证 RSA 签名
加密会话密钥和报文一起传输	发送和接收代理必须支持 Diffie-Hellman
	发送代理应该支持使用长度为 512－1024 比特的密钥的 RSA
	接收代理应该支持 RSA 解密
使用一次性会话密钥加密传输的报文	发送代理应该支持三重 DES 和 RC2/40 加密
	接收代理必须支持 RC2/40 解密
	应该支持三重 DES 解密

(3) 如果发送代理对于接收代理的解密能力没有任何了解，但是愿意冒接收代理无法解密报文的风险，则应该使用三重 DES；

(4) 如果发送代理对于接收代理的解密能力没有任何了解，并且不愿意冒接收代理无法解密报文的风险，则应该使用 RC2/40。

15.3 电子商务安全

电子商务是在开放的 Internet 环境中实现交易应用，涉及资金流和物流信息的传送，因此其安全性更重要。

针对电子商务的业务特点，目前最有影响的安全协议是安全电子交易协议(Secure Electronic Transaction，SET)。该协议由世界两大信用卡商 Master Card 和 Visa 于 1997 年 5 月联合制定。

SET 协议是一种基于信用卡网上电子商务交易的安全标准，主要是为了解决用户、商家和银行之间通过信用卡支付的交易安全而设计的，用以保证支付信息的机密、支付过程的完整、商户及持卡人的合法身份以及互操作性。SET 协议的核心技术包括电子安全证书、电子数字签名、电子信封等。

SET 协议得到许多著名 IT 公司的支持，如 IBM，Microsoft，RSA，Netscape 等。

SET 协议的详细内容很复杂，本书只能概要介绍，有兴趣的读者可参看有关论著。

15.3.1 SET 的安全目标

在一次完整的电子商务交易中，SET 涉及到消费者(卡用户)、商家、收单银行、发卡

银行和公证书中心(CA)等实体，如图 15.8 所示。

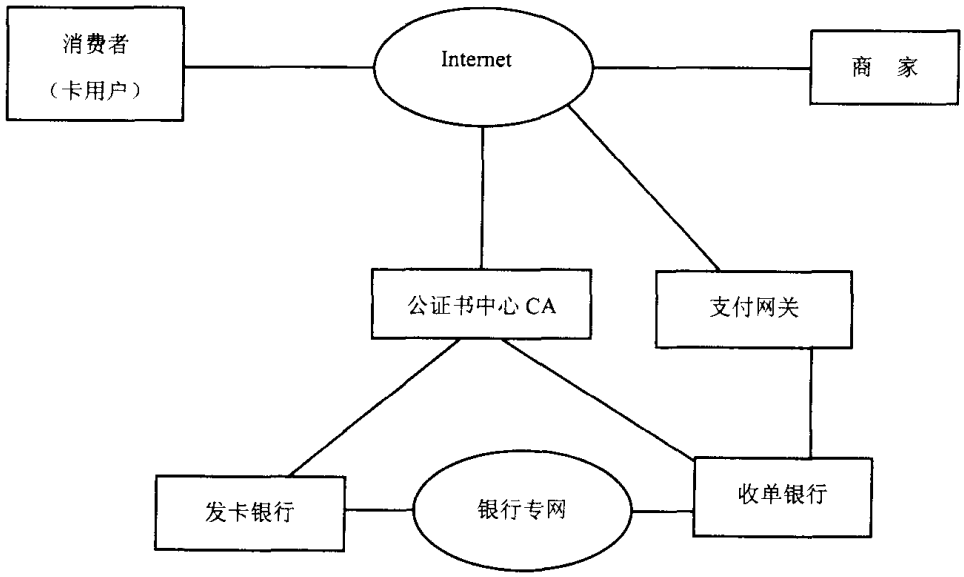


图 15.8　电子商务模型

- 卡用户：持有信用卡的消费者；
- 商家：提供网络购物的电子商店与提供电子交易服务的企业组织；
- 收单银行：使用支付网关为电子商家提供处理卡的授权及支付服务的金融机构；
- 发卡银行：负责信用卡的发放、账目管理、付款清算等业务的银行；
- 证书中心(CA)：通常由一些发卡机构共同委派，负责向卡用户、电子商家发行 X.509v3 公开密钥证书的机构。

SET 的安全目标就是保障信息流在各实体间安全流动。具体如下：

(1) 保护信息的机密性：卡用户的账号以及支付信息必须通过安全的途径在 Internet 上传输。值得注意的是，卡用户的某些信息对于商家而言应该是不可见的；

(2) 保护数据的完整性：在 Internet 上传输的过程中，卡用户对商家的支付信息不能被篡改；

(3) 身份鉴别：卡用户与商家相互认证，确定彼此身份的合法性。

15.3.2　SET 的工作流程

SET 的工作流程分为以下几个步骤：

(1) 卡用户向商家发送订购和支付信息，其中支付信息被加密，商家无从得知；

(2) 商家将支付信息发送到收单银行，收单银行解密信用卡号，并通过认证验证签名；

(3) 收单银行向发卡银行查问，确认用户信用卡是否属实；
(4) 发卡银行认可并签证该笔交易；
(5) 收单银行认可商家并签证此交易；
(6) 商家向用户传送货物和收据；
(7) 交易成功后，商家向收单银行请求支付；
(8) 收单银行按合同将货款划给商家；
(9) 发卡银行定期向用户寄去信用卡消费账单。

15.3.3 SET 交易类型

本小节先简单介绍 SET 协议的主要交易类型，然后介绍购买请求、支付认可和支付获取这三种类型交易的具体细节。

1. SET 主要交易类型

SET 的主要交易类型包括：
(1) 卡用户注册：卡用户向商家发送 SET 报文前，必须在认证中心注册；
(2) 商家注册：商家与卡用户和收单银行交换 SET 报文前，必须在认证中心注册；
(3) 购买请求：卡用户发送商家的报文，包含给商家的订购信息和给银行的支付信息；
(4) 支付兑现：在商家与收单银行间传送的报文，用来验证卡用户信用卡账户的合法性；
(5) 支付获取：商家向收单银行请求支付；
(6) 证书调查和状态：认证中心无法立刻完成证书请求时，指示请求者以后再查看；
(7) 购买调查：卡用户检查订购处理的状态；
(8) 认可撤销：商家可以更正以前的认可请求；
(9) 获取撤销：商家可以更正获取请求中的误差；
(10) 信用：商家向卡用户的账号发送一个信用；
(11) 信用撤销：商家可以更正以前的请求信用；
(12) 收单银行证书请求：商家可以请求收单银行证书；
(13) 批管理：商家可以成批的与收单银行通信；
(14) 差错信息：指出响应者报文在格式或内容检测中失败，并因此拒绝该报文；

2. SET 支付处理

在介绍支付处理前，有必要先介绍 SET 中的双签名。在 SET 协议中，卡用户要将订购信息(OI)与支付信息(PI)分别发送给商家与银行，为了避免因为 OI 与 PI 分离而产生 OI

与 PI 的错误匹配，必须将 OI 与 PI 以某种方式连接起来。在 SET 协议中，卡用户分别取得 OI 与 PI 的散列数据，将其连接起来，然后取得连接结果的散列数据，再用卡用户的私有签名密钥对最终的散列数据进行加密，最终创建双签名。

如果商家获得双签名(DS)、OI 和 PI 的报文摘要(PIMD)以及卡用户的公开密钥 KU，就可以计算 H(PIMD||H(OI))和 DKU(DS)，如果两者相等，则商家验证了该签名。如果银行获得了 DS、PI 和 OI 的报文摘要(OIMD)以及卡用户的公开密钥，则同样可以计算 H(H(PI)||OIMD)和 DKU(DS)，如果二者相等，则通过签名验证。这样既避免了商家获得卡用户支付信息中的有关私有信息，又保证了 OI 与 PI 的匹配。

支付处理主要涉及购买请求、支付授权和支付兑现 3 个阶段，介绍如下：

(1) 购买请求(Purchase Request)

购买请求交易主要可以分为 4 步。

第一步，卡用户向商家发送请求报文，请求商家和收单银行证书。

第二步，商家生成响应报文，包括商家的签名证书和收单银行的密钥交换证书，并用私钥加密。

第三步，卡用户验证有关证书，创建 OI 和 PI，生成一次性对称加密密钥 *Ks*，并且生成购买请求报文。购买请求报文包括以下信息：

a) 与支付有关的信息：PI、双签名、OI 报文摘要(OIMD)以及数字信封(通过对 *Ks* 和收单银行的公开密钥交换的密钥进行加密而成)。商家无法读取这一部分的信息。

b) 与订购有关的信息：OI、双签名、PI 报文摘要(PIMD)。

c) 卡用户的证书。

第四步，商家验证卡用户证书以及双签名，然后处理订购信息并将支付信息转送收单银行，最后向卡用户发送购买响应。

(2) 支付授权(Payment Authorization)

在处理卡用户的订购时，商家向收单银行请求授权该项交易，主要可以分为以下两步。

第一步，商家向收单银行发送认可请求报文。该报文主要包含以下内容：

a) 与支付有关的信息：PI、双签名、OI 报文摘要(OIMD)、数字信封等。

b) 与认可有关的信息：一个认可数据块(使用商家生成的一次性对称密钥进行了加密的交易 ID,并用商家私有密钥签名)、一个数字信封(使用收单银行的公开密钥交换的密钥对一次性密钥进行加密而形成的)等。

c) 证书：卡用户的签名密钥证书、商家的签名密钥证书和商家的密钥交换证书等。

第二步，收单银行先验证所有证书，接着解密并验证认可数据块以及支付数据块的双签名，然后向发卡银行请求和接收一个认可，最后向商家发送认可响应报文。该报文包括认可数据块、数字信封、获取权标信息以及收单银行的证书。

商家获得授权后，即可向卡用户提供相应的服务。

(3) 支付兑现(Payment Capture)

首先，商家发送兑现请求报文给收单银行。该报文主要包括兑现请求数据块、兑现权标以及商家的签名密钥和密钥交换密钥的证书。

收单银行收到兑现请求报文后，解密并验证兑现请求数据块以及兑现权标，验证兑现请求和兑现权标是否一致，接着通过专用网络向发卡银行发送清算请求，请求发卡银行将资金划入商家账号，然后发送兑现响应报文给商家，通知有关支付情况。而商家将保留该报文并与从发卡银行所得到的支付相匹配。

习 题 十 五

1. Web 安全的目标是什么？如何实现？
2. PGP 协议是如何保护 Email 的安全？
3. PGP 的密钥如何管理？
4. 简述电子商务的实现和流程。
5. 在 SET 协议的支付流程中如何体现对电子商务的安全保护？

参 考 文 献

1 Stallings W 著. 密码编码学与网络安全：原理与实践（第二版）. 杨明等译. 北京：电子工业出版社，2001

2 Adams C 等. 公开密钥基础设施——概念、标准和实施. 冯登国等译. 北京：人民邮电出版社，2001

3 Burnett S 等著. 密码工程实践指南. 冯登国等译. 北京：清华大学出版社，2001

4 冯登国等. 分组密码的设计与实现. 北京：清华大学出版社，2000

5 Saloman A 著. 公钥密码学. 丁存生等译. 北京：国防工业出版社，1998

6 Schneier B 著. 应用密码学——协议、算法与 C 源程序. 吴世忠等译. 北京：机械工业出版社，2000

7 Adams C. Understanding Public-Key Infrastructure. Macmillan Technical Publishing，1999

8 卢开澄. 计算机密码学. 北京：清华大学出版社，1998

9 王育民. 通信网的安全——理论与技术. 西安：西安电子科技大学出版社，2000

10 张汉亭. 计算机病毒与反病毒技术. 北京：清华大学出版社，1996

11 陈爱民等. 计算机的安全与保密. 北京：电子工业出版社，1992

12 王锡林等. 计算机安全. 北京：人民邮电出版社，1995

13 龚俭. 计算机网络安全导论. 南京：东南大学出版社，2000

14 赵战生等. 信息安全技术浅谈. 北京：科学出版社，1988

15 Kaeo M 著. 网络安全性设计. 潇湘工作室译. 北京：人民邮电出版社，2000

16 Stallings W 著. 网络安全要素——应用与标准. 潇湘工作室译. 北京：人民邮电出版社，2000

17 Ravi S, Sandhu. Access Control: Principle and Practice. IEEE Communication Magazine. 1994

18 微软公司著. Microsoft Windows 2000 网络安全设计. 袁勤勇等译. 北京：北京希望电子出版社，2001

19 Cornes P 著. Linux 从入门到精通. 童寿彬等译. 北京：电子工业出版社，1998

20 黎萍等. 网络安全管理与 Windows 2000. 北京：人民邮电出版社，2000

21 Scambray J 等著. 黑客大曝光. 钟向群等译. 北京：清华大学出版社，2002

22 林东如. 实战黑客不求人. 北京: 人民邮电出版社, 2001

23 梁上. 黑客攻防技术速查. 北京: 中国铁道出版社, 2002

24 Klander L 著. 挑战黑客——网络安全最终解决方案. 陈永剑译. 北京: 电子工业出版社, 2000

25 唐正军等. 网络入侵检测系统的设计与实现. 北京: 电子工业出版社, 2002

26 Northcutt S 著. 网络入侵检测分析员手册. 余青霓等译. 北京: 人民邮电出版社, 2000

27 段云所等. 个人防火墙. 北京: 人民邮电出版社, 2002

28 Scott Fuller 等著. Internet 防火墙. 董春等译. 北京: 电子工业出版社, 1997

29 Robert L. Ziegler 著. Linux 防火墙. 余青霓等译. 北京: 人民邮电出版社, 2000

30 Doraswamy M 著. 京京工作室译. IPSec 新一代因特网安全标准. 北京: 机械工业出版社, 2000

31 梁晋等. 电子商务核心技术——安全电子交易协议的理论与设计. 西安: 西安电子科技大学出版社, 2000

32 中华人民共和国国家标准. GB/T15408 信息技术安全评估通用准则

33 GB17859-1999 计算机信息系统安全保护等级划分准则

34 公安部计算机安全监察司. 中华人民共和国计算机信息系统安全法规汇编. 北京: 群众出版社, 1998

郑重声明

{
  "filename": "MTExNTUxMTYuemlw",
  "filename_decoded": "11155116.zip",
  "filesize": 26989776,
  "md5": "71b65d9a7787dc4932da3587285d5a47",
  "header_md5": "529aad5352ddc53fcc3303ea30a28620",
  "sha1": "cfb1bd403efb0052d396021c09621bb09594fcb8",
  "sha256": "f0e511fb4f30ba8971bc1c112725dabfb8f3a0a030d3957c2cf55094d4f137d6",
  "crc32": 2813123566,
  "zip_password": "",
  "uncompressed_size": 29608370,
  "pdg_dir_name":
"\u00ed\u2562\u2568\u253c\u2567\u00f3\u2591\u2593\u255a\u00bd\u2555\u253c\u252c\u2588\u00ed\u2556_11155116",
  "pdg_main_pages_found": 257,
  "pdg_main_pages_max": 257,
  "total_pages": 268,
  "total_pixels": 1565115520,
  "pdf_generation_missing_pages": false
}