第3章

Linux操作系统的攻防

作为类 UNIX 操作系统家族中的一员，Linux 操作系统已经成为服务器应用领域的首选。相较于 Windows 操作系统，Linux 在很多方面都具有一定的优势，尤其在安全方面。由于 Linux 的开源性，其安全漏洞的发现与补丁的发布效率都要比 Windows 系统高。然而，Linux 并非一个绝对安全的操作系统，也存在大量的安全漏洞，并且攻击者凭借其开源性可以从源码中发现更多的系统内核和开源软件的漏洞。本章将从网络攻防的角度介绍 Linux 操作系统的安全机制、攻击技术及对应的防范方法。

3.1 Linux 操作系统的工作机制

Linux 是源自于 UNIX 的开放源代码的多用户、多任务、支持多线程和多 CPU 的操作系统，主要应用于安全性要求较高的服务器、网络设备和移动终端。

3.1.1 Linux 操作系统概述

Linux 操作系统诞生于 1991 年，最初是由芬兰大学生 Linus Torvalds 为在 Intel x86 架构计算机上运行自由免费的类 UNIX 操作系统，而用 C 语言编写的开放源代码的操作系统。目前，Linux 存在着许多不同的版本，主要包括 Ubuntu、Fedora、RedHat、CentOS、OpenSUSE 等，但不同版本都使用了相同的 Linux 内核。

Linux 的基本思想是：一切都是文件，即系统中包括命令、硬件和软件设备、进程等所有对象对于操作系统内核而言都被视为拥有各自特性或类型的文件。Linux 是一款免费的操作系统，用户可以通过网络或其他途径免费获得，并可以根据需要对其源代码进行修改。目前，Linux 可以运行在多种硬件平台上，如 x86(32 位和 64 位)、680x0、SPARC、Alpha 等处理器的平台。此外，Linux 还是一种嵌入式操作系统，可以运行在智能手机、机顶盒、路由器等设备上，如 Google 基于 Linux 内核开发了 Android 移动智能终端操作系统与开发

环境。

Linux通过自带的防火墙、入侵检测和安全认证等工具，可以及时发现和修复系统的漏洞，以提高系统的安全性。在桌面应用中，Linux的用户数要少于Windows操作系统的用户数，较少的用户群使专门针对Linux的恶意代码和渗透攻击要比Windows操作系统少。同时，Linux内核源代码是以标准规范的32位或64位计算机进行优化设计的，良好的稳定性使得一些安装了Linux的主机像UNIX主机一样可以常年不关机或宕机。在网络功能方面，Linux内置了免费网络服务器软件、数据库和Web开发工具，如Apache、Sendmail、SSH、MySQL、PHP、JSP、VSFTP等。丰富而强大的网络功能为用户提供了安全可靠的网络服务，使得Linux成为安全性、稳定性和可靠性要求较高的服务器操作系统的首选。

3.1.2 Linux操作系统的结构

Linux操作系统采用宏内核(monolithic kernel)架构，整个操作系统是一个运行在核心态的单一的进程文件，这个二进制文件包含进程管理、内存管理、文件管理等。而Linux的前身Minix采用的是微内核(micro kernel)架构，基于该架构的操作系统大部分都运行在单独的进程中，而且多数在内核之外，进程之间通过消息传递来通信，内核的任务是处理消息传递、中断处理、底层的进程管理及可能的I/O。

1. Linux操作系统的内核结构

如图3-1所示，从体系结构来看，Linux操作系统的体系架构分为用户态和内核态，也称为用户空间和内核。内核从本质上看是一种软件，用于控制计算机的硬件资源，并提供上层应用程序运行的环境。用户态即上层应用程序的活动空间，应用程序的执行必须依托于内核提供的资源，包括CPU资源、存储资源、I/O资源等。为了使上层应用能够访问到这些资源，内核必须为上层应用提供访问的接口，即系统调用。系统调用是操作系统的最小功能单位，根据不同的应用场景可以进行扩展和裁剪，不同的Linux版本提供的系统调用数量各不相同。系统调用功能通过系统调用接口实现。

在Linux内核中，位于硬件抽象层中的各类设备驱动程序可以完全访问硬件设备，并以模块化形式进行设置，而且系统运行期间可以直接通过LKM(Loadable Kernel Module，可装载内核模块)机制装载或卸载。内核服务功能模块位于硬件抽象层之上，包括进程与线程管理、内存管理、文件系统管理、设备控制与网络5个子系统。这些内核服务功能模块通过系统调用接口向用户态的GNU运行库/工具、命令行Shell、X窗口及应用软件提供服务。

Shell是一个被称为“命令行”的特殊的应用程序，其实质是一个命令解释器，它负责将上层的各种应用与系统调用连接起来，以便让不同程序能够以一个清晰的接口协同工作，从而增强各个程序的功能。同时，Shell是可编程的，它可以执行符合Shell语法的文本，即Shell脚本。为了方便用户和系统交互，一般一个Shell对应一个终端，终端是一个硬件设备，呈现给用户的是一个图形化窗口，用户可以通过这个窗口输入或者输出文本，这个文本直接传递给Shell进行分析解释，然后执行。

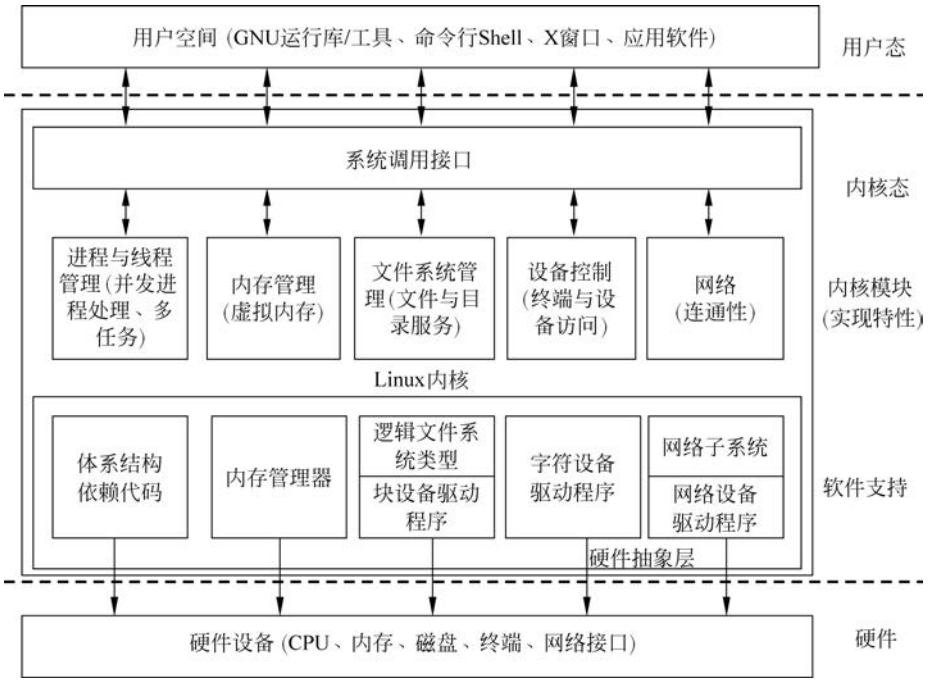

图 3-1　Linux 操作系统内核结构

2. Linux 的工作机制

Linux 操作系统在进程与线程管理、内存管理、文件系统管理、设备控制、网络、系统调用等方面都形成了特有的工作机制，掌握这些工作机制对全面学习 Linux 操作系统的功能及应用特点是非常有帮助的。

（1）进程与线程管理。进程是一个动态的概念，进程运行过程实际上是进程的一个生存周期，通常分为实际占用 CPU 的运行状态、进程可运行（但暂时挂起）的就绪状态和资源不可用的阻塞（中断）状态 3 种状态。线程可以理解为同一进程中相互独立执行的上下文，线程是“多任务”的进程。线程的概念较为适用于紧密耦合的一组处理流程。在传统进程的概念中，一个进程有一条执行线索，进程之间空间、时间独立，互不干扰。而线程在逻辑上是一个事务的不同线索，线索之间有着种种联系，可能是共享一段缓存或协调执行一组任务。

Linux 内核采用抢占式多用户多进程（multiprocessing）模式。在该模式下，多个进程可并发活动，具体由内核进程管理模块负责调度并分配硬件资源。进程作为最基本的调度单元，维护着一个进程控制块（Processing Control Block，PCB）结构，由内核 schedule()进程调度函数根据进程优先级和 CPU、内存、外设等资源情况来选择进程的运行。

（2）内存管理。内存管理在操作系统中不仅非常重要，而且非常复杂。利用虚拟存储技术，Linux 使得一个拥有有限内存资源的计算机可以为每个进程提供多达 4GB 的虚拟内存空间。其基本实现思路是通过进程映像和分页机制在内存和二级存储之间传送数据，以充分利用有限的内存资源。另外，Linux 虚拟内存管理机制把用户空间和核心空间分开，这样不仅有效地保护了核心空间，而且各个进程之间也互不影响。

（3）文件系统管理。Linux使用了虚拟文件管理（Virtual File System，VFS）机制，从而使得它能够支持多种不同类型的逻辑文件系统（主要包括ext2/ext3/ext4、vfat、NTFS等），通过设备驱动程序访问特定硬件设备（如磁盘、打印机等）。而VFS为用户进程提供了一组通用的文件系统调用函数（如open、close、read、write等），可以对不同文件系统中的文件进行统一的操作。ext2/ext3/ext4是Linux中默认的文件系统格式，使用索引节点来记录文件信息，作用类似于Windows系统中的FAT32文件系统的目录项，包含了文件长度、创建及修改时间、权限、所属关系、磁盘中的位置等信息。

（4）设备控制。设备驱动程序（device driver）是操作系统中一种可以使计算机和设备进行通信的特殊软件。Linux抽象了设备的处理，它将所有的硬件设备都视为常规的文件来处理。Linux支持3种类型的硬件设备：字符设备、块设备和网络设备。其中，字符设备直接读/写，没有提供缓冲区，如系统的串行端口/dev/cua0和/dev/cua1；块设备只能按照一个块（一般是512字节或1024字节）的倍数进行读/写，块设备通过bcache（buffer cache）访问，进行随机存取；网络设备则通过BSD（Berkeley Software Distribution，伯克利软件套件）Socket网络接口进行访问。大多数的设备驱动程序都采用LKM（Loadable Kernel Modules，可装载内核模块）机制，在需要的时候作为核心模块加载，在不需要的时候进行卸载，以提高对系统资源的利用率。

（5）网络。Linux中的网络模块提供了对各种网络标准的访问，并支持各种网络硬件设备。网络接口可以分为网络协议栈和网络驱动程序。其中，网络协议栈负责实现每种可能的网络传输协议，包括网络接口层的协议（如以太网、PPP、SLIP等）、TCP/IP协议层，以及为上层网络应用提供的Socket接口，如图3-2所示。网络设备驱动程序负责与硬件设备之间的通信。

Linux通过以上5种工作机制实现了操作系统基本的硬件管理和系统功能，这些功能模块全部运行在CPU的核心态。而应用程序运行在用户态，不能直接访问内存空间，也不能直接调用内核函数。Linux提供了系统调用接口，应用程序通过该接口可以访问硬件设备和其他系统资源，以增加系统的安全性、稳定性和可靠性。同时，Linux为用户空间提供了一种统一的抽象接口，有助于应用程序的跨平台移植。

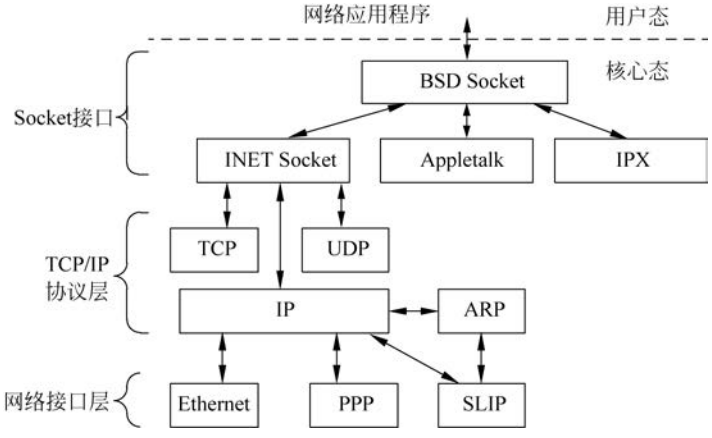

图3-2　Linux操作系统的网络功能

3.2 Linux 操作系统的安全机制

与 Windows 操作系统类似，Linux 同样通过身份认证、授权访问与安全审计等机制来实现对系统的安全管理。

3.2.1 用户和组

Linux 通过基于角色的身份认证方式实现对不同用户（user）和组（group）的分类管理，来确保多用户多任务环境下操作系统的安全性。

1. 用户

用户是 Linux 操作系统中执行进程完成特定操作任务的主体，根据不同的角色定位，可以将用户分为以下 3 种类型。

（1）root 用户。root 用户是 Linux 系统中唯一拥有系统管理员（超级用户）权限的用户，可以对系统进行任何的操作。由于 root 用户对系统拥有最高的控制和管理权限，因此成为网络攻击的主要目标。

（2）普通用户。普通用户是由系统使用者根据需要创建的一种用户类型，其基本功能是登录系统并执行基本的计算任务，该类用户在系统中的操作被限制在自己的目录内，且执行权限受到限制。

（3）系统用户。系统用户不具有登录系统的能力，但却是系统运行中不可缺少的用户。例如，当启动网络服务时使用的 Daemon、Apache 等用户，以及匿名访问时使用的 Nobody、FTP 等用户。

Linux 的用户信息保存在系统的“/etc/passwd”文件中，主要包括用户名、用户唯一的标识（UID）、使用 Shell 类型、用户初始目录等，而被加密后的口令则存放在“/etc/shadow”文件中，只有 root 用户可以读取其信息。

2. 组

Linux 通过组来简化对系统中用户的管理。根据管理的需要，可以将具有相同权限的用户集中纳入同一个组中，通过对组设置权限来使该组中的所有用户自动继承组的权限。在权限设置上，对组的权限设置会自动传递给组中的所有用户，因此组也称为用户组。

Linux 组信息保存在系统的“/etc/group”文件中，包括组名称、组标识（GID）及组所包含的用户名列表，组被加密后的口令保存在“/etc/gshadow”文件中。可以使用 id-a 命令查询和显示当前用户所属的组，并通过 groupadd 命令添加组，使用 usermod-G group_name user-name 命令向组中添加用户。


视频讲解

3.2.2 身份认证

Linux 分别为本地登录和远程登录用户提供了身份认证方式，同时还为不同的应用软件和网络服务提供了用于统一身份认证的 PAM（Pluggable Authentication Modules，可插

入身份认证模块)中间件。

1. 本地身份认证

本地身份认证对从本地计算机通过 Linux 控制台登录的用户身份的合法性进行认证,基本的认证流程是:由 init 进程启动 getty,产生 tty1、tty2 等一组虚拟控制台。在虚拟控制台上为用户提供了登录方式,在用户输入用户名和密码后,getty 执行登录(Login)进程,并开始对用户身份的合法性进行认证。当身份认证通过后,登录进程会通过 fork()函数复制一份该用户的界面(Shell),从而完成登录过程,用户可以在该界面下进行相应的操作。

登录进程通过 Crypt()函数对用户输入的口令进行验证,并通过引入在用户设置密码时随机产生的 salt 值来提高身份认证的安全性。salt 值和用户密码被一起加密后形成密文,连同 salt 值保存在"/etc/shadow"文件中。当用户登录系统时,Crypt()函数会对用户输入的口令和 shadow 文件中的 salt 值进行加密处理,再将处理后的密文与保存在 shadow 文件中的密文进行比对,以确定用户身份的真实性。

Linux 的口令加密机制源于 DES(Data Encryption Standard,数据加密标准),通常使用 56 位密钥加密的 64 位的文本块,抵抗暴力破解的能力较弱。为提高身份认证的可靠性,较新版本的 Linux 开始采用 MD5、SHA-256、SHA-512、blofish 等高强度的加密算法,同时增加了 salt 的编码长度。

2. 远程身份认证

UNIX 系统中提供的 Rlogin(远程登录)、RSh(Remote Shell,远程界面)和 Telnet 登录用户和终端登录与访问服务在 Linux 系统中得到了继承,但由于这些服务信息都以明文方式在网络中传输,传输口令和控制命令极易被攻击者获取并利用,如典型的中间人(man in the middle)攻击。为解决此问题,目前 Linux 系统普遍采用 SSH(Secure Shell)服务来实现对远程访问的安全保护。

使用 SSH 具有两大明显的优势:数据加密和数据压缩。利用数据加密功能可以对所有传输的数据进行加密,以避免中间人攻击或网络欺骗;利用数据压缩功能,可以对传输的数据进行压缩,以提高数据传输的效率。

SSH 协议由传输层协议(Transport Layer Protocol,TLP)、用户认证协议(User Authentication Protocol,UAP)和连接协议(Connection Protocol,CP)三部分组成。每层提供自己类型的保护,并且可以与其他方式一起使用,SSH 协议的体系结构如图 3-3 所示。

(1) 传输层协议。SSH 传输层协议提供了高强度的数据通信加密处理、加密的主机身份认证、数据完整性校验及可供用户选择的数据压缩等多种安全服务。通信双方所需要的密钥交换方式、公钥密码算法、对称密钥密码算法、消息认证算法和 Hash 算法等都可以进行协商。传输层协议的主要功能是为两种主机之间的认证和通信提供安全数据传输通道,通常运行于 TCP/IP 之上。

需要说明的是,SSH 传输层协议中的认证是基于主机的,而不是针对客户端用户的身份认证。用户身份认证可以通过基于传输层协议之上、单独设计的协议来完成。这样既保证了通信的安全性,又提供了协议的灵活性和扩展性。

(2) 用户认证协议。在传输层构建了一个连接客户端与服务器端的安全通道后,服务器将告诉客户端它所支持的认证算法,客户端将用服务器支持的算法向服务器证明自己的

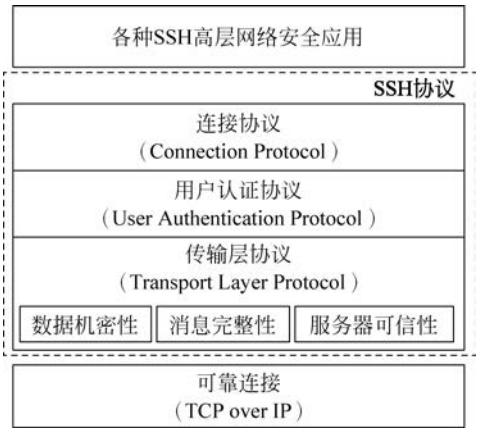

图 3-3　SSH 协议的体系结构

身份。认证由服务器主导，客户端可以根据服务器提供的方法列表自由进行选择。这样一方面使服务器对认证有完全的控制权，同时也给客户端足够的灵活度。

SSH 提供了基于口令的安全验证和基于密钥的安全验证两种方式。其中，基于口令的安全验证方式可以使用 Linux 系统内建的用户账户（用户名＋口令）进行远程登录；基于密钥的安全验证方式使用公钥密钥机制对用户身份的合法性进行认证。系统生成的一对密钥，私钥由用户自己保存，而公钥保存在远程访问服务器上。当用户通过 SSH 方式连接服务器时，客户端软件首先向服务器发出连接请求，服务器在接收到请求后就利用请求用户的公钥加密“质询”（challenge）并将其发送给客户端软件，客户端软件收到被加密的“质询”后利用私钥进行解密，再发送给服务器端，完成了基于公钥密钥机制的身份认证过程。

（3）连接协议。SSH 连接协议的主要功能是完成用户请求的各种具体的网络服务，而这些服务的安全性由 SSH 传输层协议和用户认证协议实现。在 SSH 传输层成功认证后，通过信道复用方式同时打开客户端与服务器之间的多个信道连接，每个信道处理不同的终端会话。通过连接协议提供的信道，扩展了 SSH 协议的应用范围和灵活性。

SSH 使用多种加密方式和认证方式，解决了传统服务的数据加密和身份认证问题。SSH 成熟的公钥密钥体系为客户端和服务端之间的会话提供加密通道，解决了口令、控制命令和用户数据在网络上以明文传输的不安全问题。SSH 还支持 CA、智能卡等多种认证方式，有效解决了身份认证问题，并克服了重放攻击和中间人攻击等不安全因素。

3. 可插入身份认证模块（PAM）

为了能够为不同的应用程序和服务提供统一的身份认证机制，1995 年 Sun 公司提出了 PAM（Pluggable Authentication Modules，可插入身份认证模块）技术，并充分利用互联网中已成功应用的分层思想。采用分层的体系结构，将不同应用程序的认证功能从应用程序中分离出来，形成一个额外的相对独立的认证层，使得系统管理员和软件开发人员可以根据需要灵活地配置或开发应用程序，而无须具体了解应用程序自身的认证机制。

PAM 是要求对其服务进行身份认证的应用程序与提供认证服务的认证模块之间的中间件。PAM 提供了对所有服务进行认证的中央机制，应用于 Login、Telnet、FTP、SU（Switch User）等应用程序中。系统管理员通过 PAM 配置文件来制定不同应用程序的不同

认证策略；应用程序开发者通过在服务程序中使用 PAM API 来实现对认证方式的调用；PAM 服务模块(Service Module)的开发者则利用 PAM SPI(Service Module API，服务编程接口)来编写认证模块，将不同的认证机制(Kerberos、智能卡等)加入系统中；PAM 接口库(Libpam)则读取配置文件，以此为根据将服务程序和相应的认证方法联系起来，为各种服务提供身份认证服务。

如图 3-4 所示，PAM 体系结构分为应用层、接口层和服务层。其中，应用层由需要进行身份认证的应用程序组成；接口层由 PAM API 函数和配置文件组成，用于服务模块的管理和配置；服务层则由具体的认证服务模块组成，是进行身份认证的核心。PAM API 是应用程序和认证服务模块之间的接口，应用程序通过调用 PAM API 访问一组配置文件，按照配置文件的规定加载相应的认证服务模块。PAM API 函数把认证请求及参数传递给底层的认证服务模块，认证服务模块根据要求执行具体的认证操作。当认证服务模块执行完相应的操作后，将结果返回给 PAM API，由 PAM API 将配置文件认证结果返回给应用程序。

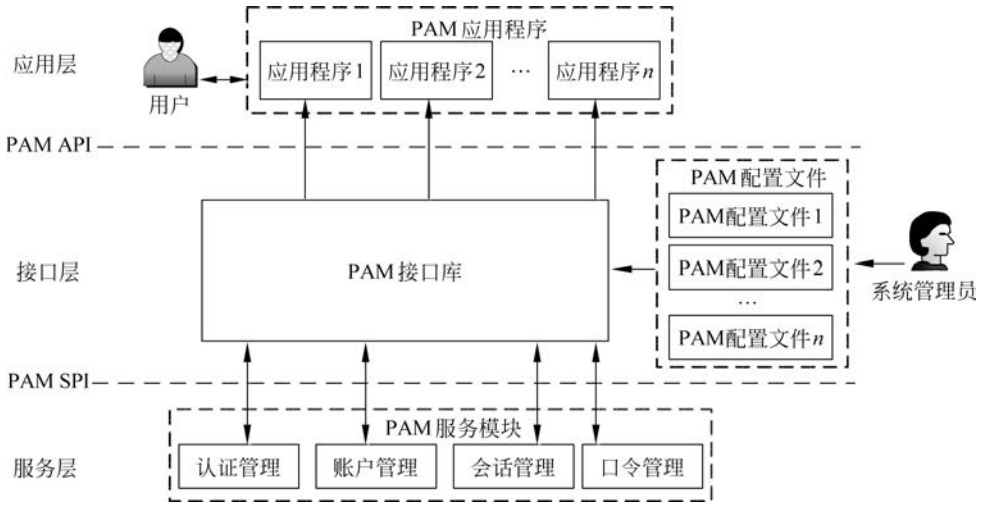

图 3-4　PAM 体系结构

PAM 支持认证管理、账户管理、会话管理和口令管理 4 种服务模块。其中，认证(authentication)管理主要接收用户名和密码，进而对该用户的密码进行认证，并负责设置用户的一些涉密信息；账户(account)管理主要是检查账户是否被允许登录系统、账号是否已经过期、账号的登录时间是否存在限制等；会话(session)管理主要用来定义用户登录前后所要进行的操作，如登录连接信息、用户数据的打开与关闭、挂载文件系统等；口令(password)管理主要用来修改用户的密码。

Linux 的 PAM 配置可以在“/etc/pam.conf”文件或“/etc/pam.d/”目录下进行，系统管理员可以根据需要进行灵活配置。不过，这两种配置方式不能同时起作用，即只能使用其中一种方式对 PAM 进行配置，一般为 etc/pam.d 优先。

4. SELinux

SELinux(Security Enhanced Linux，安全强化 Linux)是美国国家安全局(NSA)于 2000 年正式发布的，是对于 MAC(Mandatory Access Control，强制访问控制)的一种实现，目的

在于明确地指明某个进程可以访问哪些资源（文件、网络端口等）。SELinux 默认安装在 Fedora、RHEL（Red Hat Enterprise Linux）等服务器操作系统上，也可以通过安装包的形式安装到其他发行版的 Linux 服务器系统上。

通过使用 MAC 可以有效地抵御 0day 攻击。例如，目前很多互联网上的 Web 服务通过在 Linux 系统上安装 Apache 服务来实现，如果 Apache 存在漏洞，那么攻击者就可以访问 Web 服务器上的敏感文件，如通过“/etc/passwd”来获得系统中已有用户。但是，修复存在的安全漏洞需要由 Apache 开发商提供补丁程序，这需要一段时间。此时 SELinux 可以发挥其功能来弥补由该漏洞引起的安全攻击，因为“/etc/passwd”不具有 Apache 的访问标签，所以 Apache 对于“/etc/passwd”的访问会被 SELinux 阻止。

SELinux 可以从进程初始化、继承和程序执行 3 个方面通过安全策略进行控制，控制范围覆盖文件系统、目录、文件、文件启动描述符、端口、消息接口和网络接口等。SELinux 安全策略的配置文件为“/etc/sysconfig/selinux”。

3.2.3 访问控制

访问控制技术的基本目标是防止非法用户进入系统和合法用户对系统资源的非法使用。为了达到这个目标，访问控制通常以用户身份认证为前提，在此基础上实施各种访问控制策略来控制和规范合法用户在系统中的行为。

1. 虚拟文件系统（VFS）

Linux 支持 Ext/Ext2/Ext3/Ext4、XIA、MINIX、MSDOS、FAT32、NTFS、PROC、STUB、NCP、HPFS、AFFS 等多种文件系统，不同的物理文件系统具有不同的组织结构和不同的处理方式。为了能够处理各种不同的物理文件系统，操作系统必须把它们所具有的特性进行抽象，并建立一个面向各种物理文件系统的转换机制。通过这个转换机制，把各种不同的物理文件系统转换为一个具有统一共性的虚拟文件系统。VFS（Virtual File System，虚拟文件系统）实际上向 Linux 内核和系统中运行的进程提供了一个处理各种物理文件系统的公共接口，通过这个接口使不同的物理文件系统看起来都是相同的。

VFS 和各种物理文件系统之间的关系如图 3-5 所示。从图中可以看出，VFS 并不是一种物理文件系统，它仅是一套转换机制，在系统启动时建立，在系统关闭时消失，并且仅存在于内存空间。所以，VFS 并不具有一般物理文件系统的实体。在 VFS 提供的接口中包含向各种物理文件系统转换用的一系列数据结构（如 VFS 超级块），同时还包含对不同物理文件系统进行处理的各种操作函数的转换入口。

2. Linux 的权限分配与访问控制机制

在 Linux 操作系统中，不仅仅是普通的文件，包括目录、字符设备、块设备、套接字等在内的所有类型都以文件形式被对待，即“一切皆是文件”。在 Linux 操作系统中对所有文件与设备资源的访问控制都通过 VFS 来实现，所以在 Linux 的虚拟文件系统安全模型中，可以通过设置文件的相关属性来实现系统的授权和访问控制。

为便于对 Linux 文件属性的理解，图 3-6 是对以 root 的身份运行 ls-al 命令后显示结果的内容说明。下面针对授权和访问控制功能的实现，主要介绍文件属性字段的定义。

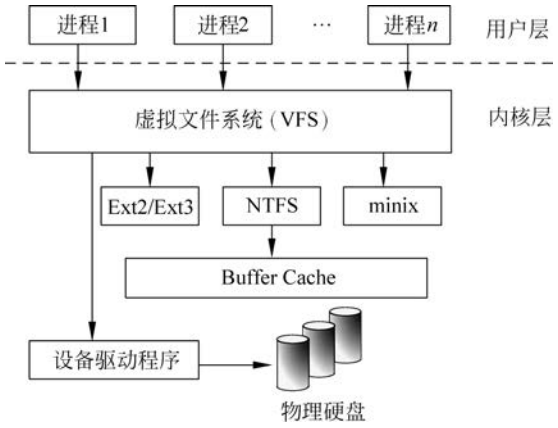

图 3-5　VFS 和各种物理文件系统之间的关系

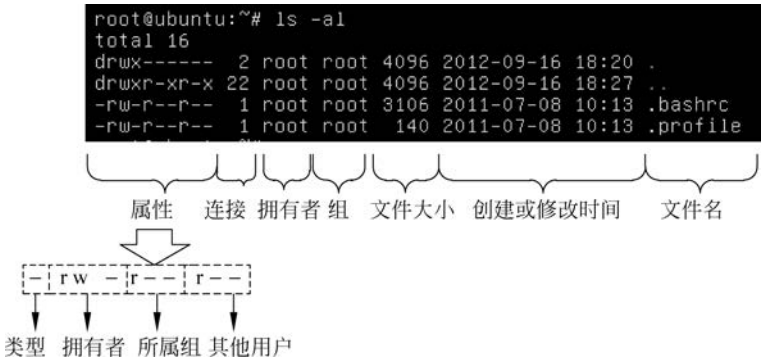

图 3-6　对 Linux 文件显示内容的说明

(1) 类型。其中，“d”表示目录，“-”表示文件，“l”表示连接文件，“b”表示设备文件中可供存储的接口设备(即块设备文件)，“c”表示设备文件中的串行设备(如键盘、鼠标)。

(2) 拥有者(ownership)。每个 Linux 文件都有一个拥有者，表明这一文件归谁所有。拥有者以用户 ID(User ID)及文件拥有者所在的组 ID(Group ID)来标识。在用户创建一个文件时，文件系统将自动设置新文件的拥有者及其所在的组，并自动分配给文件拥有者读/写(r/w)权限。文件的拥有者可以通过 chown 命令进行修改。

文件拥有者、拥有者所属组和其他用户对文件都可以被分配读(read)、写(write)和执行(execute)权限。其中，“r”对文件来说具有读取文件内容的权限，对目录而言具有浏览目录的权限；“w”对文件来说具有新增或修改文件内容的权限，对目录而言具有删除或移动目录内文件的权限；“x”对文件来说具有执行文件的权限，对目录而言具有进入目录的权限。

需要特别注意的是“x”权限。如果文件名为一个目录，当需要对其他用户开放该目录时，首先要开放该目录的“x”权限。如果开放了“r”权限，而没有开放“x”权限，该用户同样无法进入该目录并读取目录中的内容。

需要说明的是，在 Windows 系统中，由文件的扩展名决定了该文件的性质，如以. exe、. bat 和. com 为扩展名的文件都是可执行文件；而在 Linux 系统中，文件是否能够执行，则

是通过是否拥有执行“x”权限来决定的。

(3) 所属组和其他用户。在 Linux 系统中，UID 是代表拥有者的唯一标识。根据管理需要，一个 UID 可以指派到一个或多个 GID 中进行管理。除拥有者和拥有者所属组中的用户之外的用户，称为其他用户。例如，GID 为 studentgroup 的组中存在 student1、student2 和 student3 共 3 个用户(拥有者)，如果 student1 对某一文件拥有“-rwxrwx---”属性，那么 student2 和 student3 同样会拥有该属性，因为这 3 个用户同属于一个组 studentgroup。其他用户将不会拥有对该文件的“-rwxrwx---”属性。

例如，通过 ls -l 命令来显示 myfile 文件的信息，显示为“-rwxr-x--- 1 foot staff 7734 Apr 08 14:27 myfile”。

其中，myfile 表示普通文件，文件的拥有者是 foot 用户，而 foot 用户属于 staff 组，文件只有一个连接，文件长度是 7734 字节，最后修改时间为 4 月 8 日 14:27。拥有者 foot 对文件有读、写和执行权限，staff 组的成员对文件有读和执行权限，其他的用户对这个文件没有权限。

通过以上介绍可以看出，Linux 系统中的每一个文件在具有了拥有者和访问权限之后，系统将会通过 VFS 来对每次针对该文件的操作请求进行访问控制。通过获取该文件的拥有者及访问权限信息，来决定该操作请求者是否拥有读、写和执行权限。如果请求得到许可，则依据具体的权限分配对该文件进行相关的操作；否则，显示“Permission deny”(权限受限)提示。

(4) SUID 和 SGID。SUID(Set User ID)和 SGID(Set Group ID)表示对文件属性的拥有者或拥有者所属组的执行(x)权限的“特殊”设置，即将原来的执行(x)位修改为“s”位，如“-rwsr-xr-x”表示 SUID 和拥有者权限中可执行位被设置，“-rw-r-sr---”表示 SGID 被设置，但所属组中的用户权限中的执行位没有被设置。

SUID 权限允许可执行文件在运行时从运行者的当前身份提权至文件拥有者的权限，可以任意访问文件拥有者的全部资源。例如，当某一程序的文件拥有者为 root，且设置了 SUID 和拥有权限中可执行位时，该程序就拥有了 root 所具有的特权权限，“/bin/login”文件就是设置了 SUID 位且为 root 拥有的可执行程序。

针对 SUID 和 SGID 的特点，一旦一些程序存在安全漏洞且被利用，系统就容易受到攻击。尤其是攻击者在获得 root 访问特权后，可以用 root 的身份对系统进行任意操作。例如，攻击者在提权到 root 后，可以对系统植入木马，并将木马程序设置上 SUID 位和 root 拥有，随时发起对系统的攻击。

SGID 位与 SUID 位的功能类似，设置了 SGID 位的程序执行时是以拥有者所属组的权限运行的，该程序可以任意访问整个组能够使用的资源。

3.2.4 Linux 的日志

Linux 提供了丰富的日志功能用于记录和查看应用程序的各种信息。大部分发行版本中，Linux 系统的日志服务由日志守护进程 syslog 管理，syslog 位于“/etc/syslog/”“etc/syslogd”或“/etc/rsyslog.d”中，默认配置文件为“/etc/syslog.conf”或“rsyslog.conf”，当某一程序要生成日志时都需要通过配置向 syslog 发送信息。例如，Linux 系统内核和许多程

序会产生各种错误信息、警告信息和其他的提示信息，这些信息对系统管理员了解系统的运行状态是非常有帮助的，一般都需要通过 syslog 将其记录到日志文件中。默认配置下，日志文件通常都保存在“/var/log”目录下。表 3-1 列出了由 syslog 管理的常用日志文件及其说明。

表 3-1 由 syslog 管理的常用日志文件及其说明

日 志 文 件	功 能 说 明
/var/log/boot.log	记录了系统在引导过程中发生的事件，即 Linux 系统开机自检过程显示的信息
/var/log/lastlog	记录了最后一次用户成功登录的时间、登录时使用的 IP 等信息
/var/log/messages	记录 Linux 操作系统常见的系统和服务错误信息
/var/log/secure	Linux 系统安全日志，记录了用户和组变化情况、用户登录认证情况等
/var/log/btmp	记录 Linux 登录失败的用户、时间及尝试登录时使用的 IP 地址等信息
/var/log/syslog	只记录警告信息，主要是系统出问题的信息，可通过 lastlog 命令查看
/var/log/wtmp	永久记录了每个用户登录、注销及系统的启动、停机的事件，可使用 last 命令查看
/var/run/utmp	记录了有关当前登录的每个用户的信息，如 who、w、users、finger 等需要访问这个文件

另外，“/var/log/syslog”或“/var/log/messages”文件用于存储所有的全局系统活动数据，包括开机信息等。其中，基于 Debian 的系统（如 Ubuntu）在“/var/log/syslog”文件中存储日志信息，而基于 RedHat 的系统（如 RHEL、CentOS 等）则在“/var/log/messages”文件中存储日志信息。“/var/log/auth.log”或“/var/log/secure”文件存储来自可插入身份认证模块（PAM）的日志，包括已成功的登录、失败的登录尝试和认证方式等。Ubuntu 和 Debian 在“/var/log/auth.log”文件中存储认证信息，而 RedHat 和 CentOS 则在“/var/log/secure”文件中存储该信息。

多数基于 Linux 环境的应用程序都提供了功能丰富的日志记录，用于记录主要事件与出错信息，以加强对程序运行的监管。例如，Apache 程序的访问日志（access_log）记录了 HTTP 访问的相关信息，通过漏洞扫描可以从中发现系统存在的安全缺陷，通过对这些安全缺陷的分析可以降低远程入侵的可能性。

3.3 Linux 系统的远程攻防技术

与针对 Windows 系统的攻防相似，针对 Linux 系统的网络攻防技术同样包括收集目标 Linux 主机的信息、发现安全漏洞、利用安全漏洞远程获取 Linux 主机的 Shell 访问权、提权至 root 用户权限、实施攻击行为等步骤。本节主要介绍各个步骤的主要实现方法。

3.3.1 Linux 主机账户信息的获取

由于 Linux 系统所具有的可靠性和稳定性，互联网上的 FTP、邮件、Web 等大量的应用服务多采用 Linux 系统来提供。针对这些应用服务的网络攻击，多通过收集目标主机的远程登录账户信息（用户名＋口令）来实现。

1. 远程登录账户信息的获取

获取远程登录用户的账户信息是实施远程入侵的关键，为此，攻击者在确定了被攻击的目标后，需要通过各种方法获得登录的用户名和密码。为实现这一目的，最高效的办法是在直接获取保存远程登录账户信息的文件（etc/passwd 和 etc/shadow）后，从文件中取得用户名和密码。显然这一过程是很难实现的，因为出于安全考虑，Linux 系统对保存用户账户信息的文件从存储和访问控制等方面都设置了严格的管理权限，只有 root 用户才能读取，而要获取 root 用户的权限则需要获得其密码。

在具体网络攻防中，多通过口令猜测或暴力破解等攻击手段来获取远程登录账户的信息。一般过程是：首先利用 Linux 系统上的 rusers、sendmail、finger 等服务来获取被攻击 Linux 主机上的用户名，然后通过猜测（针对弱口令）、字典攻击、暴力破解等方式来获得对应的密码。其中，由于 root 账户的重要性，利用该方法获得其登录密码几乎成为所有攻击者的关注目标。

除了系统账户信息外，HTTP/HTTPS、FTP、SNMT、POP3/SMTP、MySQL 等基于 Linux 系统的各类网络服务所拥有的管理账户信息也是攻击者关注的焦点。不过，与系统账户不同的是，这些网络服务的管理账户的操作一般会被限制在一定的范围内。例如，Apache 是 Linux 系统上使用最为广泛的 HTTP 服务，攻击者在获得管理员账户信息后，可以对发布的 Web 站点目录文件进行读取或修改，利用可以上传 PHP 后门程序的权限，达到修改 Web 主页或上传木马的目的。

2. 远程登录账户的防范方法

与 Windows 系统中用户账户信息的安全管理类似，在 Linux 系统中要防御针对远程登录账户的攻击，仍然需要从用户名和密码两方面入手，加强对用户账户信息的管理。远程登录账户的防范方法主要包括以下几方面。

（1）为不同的管理员分配不同的管理账户，而不是共同使用 root 账户。

（2）限制 root 等特权账户的远程登录功能，只允许其本地登录。如果部分特权账户需要进行远程登录，可使用普通账户登录后再通过 su 命令提权，su 命令的密码功能可以增强登录账户的安全性。

（3）限制尝试登录次数，对多次登录失败的账户进行锁定并记录其信息。

（4）密码设置符合复杂性要求，即密码不少于 8 个字符，字符包含字母、数字和特殊符号（如 $、@、_等）。

最有效的安全防范方法是利用基于 PKI（Public Key Infrastructure，公钥密钥基础设施）技术的身份认证机制来替代传统的“用户名＋口令”方式，同时将一些安全风险较高的服务（如 SSH）设置到非熟知端口上，以减少口令攻击的可能性。

3.3.2 Linux 主机的远程渗透攻击

远程渗透攻击的实现主要依赖于目标主机上存在的各类安全漏洞。当攻击者要对某一目标主机进行远程渗透攻击前，首先要收集目标主机的相关信息，然后分析是否存在安全漏洞。如果存在安全漏洞，再考虑如何去利用。为此，从攻防角度分析，发现安全漏洞是实施

攻击的前提，而及时修补安全漏洞是进行防范的基础。

1. Linux 安全漏洞及利用

漏洞的普遍性及其后果的严重性促使研究人员将更多注意力集中于漏洞相关技术的研究上，包括漏洞检测（发现/挖掘）、漏洞特性分析、漏洞定位、漏洞利用、漏洞消控等。Linux作为一个开放源代码的操作系统，较之闭源的 Windows 操作系统，研究人员可以从源代码分析过程中发现漏洞，并利用其开源性进行及时修复。然而，也存在一些漏洞在发现之后未能及时发布补丁程序，而是被用于渗透攻击的现象。

与 Windows 系统相比较，Linux 系统的安全漏洞相对较少，但由于 Linux 系统在网络服务应用领域占有较高的比例，所以其安全漏洞存在的风险和威胁更为严重。例如，RHEL Linux 系统内核网络协议栈实现（net/ipv4/udp. c）中存在一个远程拒绝服务安全漏洞（CVE-2010-4161），攻击者通过向目标主机上任意开放的 UDP 端口发送一个特殊构造的 UDP 数据包，就可以发起对目标主机的 DoS 攻击。Linux 系统的每个网络服务都依赖于内核中的网络协议栈实现，一旦这些实现代码中存在具有远程代码执行危害后果的安全漏洞，不管 Linux 系统开放什么服务，都可能被攻击者用于实施远程渗透攻击。

LAMP（Linux/Apache/MySQL/PHP）是目前互联网上应用最为广泛的 Web 站点解决方案，即以 Linux 操作系统作为网站运行的服务器基础平台，以 Apache 提供的 HTTP/HTTPS 作为 Web 服务，以 MySQL 数据管理系统作为 Web 应用程序的后台数据库，以 PHP 语言作为 Web 应用程序的开发语言。在这种高效的 LAMP 组合中，一旦任何一个组件存在安全漏洞，都会被利用进行目标主机的远程渗透攻击。例如，早期 Apache mod_rewrite 模块中存在对 LDAP 协议 URL 处理过程中的溢出漏洞（CVE-2006-3747），可以对 Web 服务器通过 TCP 80 端口进行远程溢出攻击，从而获得 Web 服务器的本地访问权。又如，MySQL：sha256_password 认证长密码拒绝服务式攻击漏洞（CVE-2018-2696），该漏洞源于 MySQL sha256_password 认证插件，该插件没有对认证密码的长度进行限制，而直接传给 my_crypt_genhash()，用 SHA256 对密码加密求哈希值。该过程需要大量的 CPU 计算，如果传递一个很长的密码时，会导致 CPU 耗尽。还有，利用 PHP HTTP_PROXY 环境变量安全漏洞（CVE-2016-5385）中 HTTP_PROXY 环境变量未能过滤构造的客户端数据这一缺陷，远程攻击者通过构造 HTTP 请求的 Proxy 标头，可以将 HTTP 数据流重定向到任意的代理服务器。除此之外，运行于 Linux 平台的 FTP、Samba、Sendmail 等服务对应的各类软件，都被发现存在不同程度的安全漏洞。

2. 针对远程渗透攻击的防范方法

由于远程渗透攻击的实现主要利用了被攻击目标主机存在的安全漏洞，所以加强安全漏洞的检测和修补是防范攻击的基础。

（1）只开启需要的服务。网络远程渗透攻击中需要借助主机上开启的服务，即利用服务存在的漏洞实施攻击。开启服务需要同时启用服务进程，并打开对应的端口。对于互联网上的服务器来说，只需要开启与业务相关的最基本的网络服务，其他的服务应全部禁用。

（2）使用安全性高的服务软件。互联网上的一个协议一般会同时对应多款服务软件，虽然每一款软件的基本功能都是基于相同的协议标准来开发的，但代表各自特点的扩展功能可能不尽相同。同时每一款软件的应用表现也不完全一致，有些软件注重操作的友好性

却忽视了安全性，而有些软件有可能在强调安全性的同时使易操作性不尽如人意。例如，Linux 系统中可以分别通过 Apache、Nginx、Tomcat 来提供 HTTP 网络服务，这 3 款软件虽然都提供 Web 服务功能，但其应用特点不尽相同。其中，Apache 不但可以跨平台运行（可运行于 UNIX、Linux 和 Windows 系统中），还具有较高的安全性，以及拥有快速、可靠、简单的 API 扩展，是互联网上使用最多的 Web 服务软件；Nginx 作为一款轻量级的网站服务软件，具有很好的稳定性和丰富的功能，且占用系统资源较少；Tomcat 属于轻量级的 Web 服务软件，一般用于开发和调试 JSP 代码，通常认为 Tomcat 是 Apache 的扩展程序。作为 Web 服务器，如果追求性能可以选择 Nginx，而如果强调安全性则选择 Apache。出于安全考虑，在应用功能能够满足需求的前提下，应尽可能选择安全性高的服务软件。

（3）及时更新软件。及时更新软件可以增加软件自身的新功能，解决以前版本的漏洞或缺陷，增加软件的稳定性和对新的操作系统提供更好的支持等，尤其是对发现的软件安全漏洞需要进行及时修补。例如，在 RHEL、CentOS、Fedora Core 等 Red Hat 系列 Linux 发行版本中，可以通过 yum update 命令来将软件更新到最新版本，并通过 chkconfig-level 3 yum on 命令来激活“/etc/cron. daily/yum. cron”，再通过 Crond 服务来配置系统的自动更新时间。需要注意的是，在进行软件版本升级前，需要对服务软件在新版本环境中进行测试，测试无误后再升级，因为在旧版本下运行良好的软件不一定会适应新版本的要求。

（4）设置访问控制机制。Linux 系统在启动时根据需要开启相应的服务，并禁用不需要的服务，这样不但可以有效地利用系统的资源，而且更有利于系统的安全。Linux 系统提供了一个被称为“超级守护进程”的 xinetd（eXtended InterNET Daemon）工具。在系统启动时由 xinetd 负责统一管理需要启动的进程，在系统启动后，当相应请求到来时需要通过 xinetd 的转接来唤醒被 xinetd 管理的进程。同时，xinetd 内建了基于远程主机地址、网段及域名的访问控制机制，并支持分时间段的访问控制。另外，xinetd 还能够限制服务并发运行数、服务进程数和同一主机的最大网络连接数，此功能可以有效地缓解对主机的 DoS 攻击。还有，xinetd 支持将网络服务绑定到指定的网络接口与监听端口上，以降低被扫描和攻击的风险。除 xinetd 工具之外，Linux 系统集成的 netfilter/iptables 防火墙解决方案可以有效地加强对网络边界的安全管理。

3.3.3 DNS 服务器的攻防

DNS（Domain Name System，域名系统）是互联网中绝大多数应用的实际寻址方式，域名是互联网上的身份标识，是不可重复的唯一标识资源。DNS 以其操作的便捷性在丰富了互联网应用的同时，因其在互联网应用中的重要性，已成为网络攻击的主要对象。

1. BIND 介绍

BIND（Berkeley Internet Name Domain）是互联网上使用最为广泛的域名解析软件，目前有 90％以上的域名服务器都使用 BIND 来解析。BIND 由加州大学伯克利分校开发，目前有 BIND 8. x 和 BIND 9. x 这两个不同发展方向的版本，BIND 8. x 中融合了许多提高效率、增强稳定性和安全性的技术；而 BIND 9. x 则增加了一些新的应用功能，如支持 IPv6、提供公开密钥加密、支持多处理器、提供线程安全操作、提供增量区传送等。从 BIND 9. x 开始支持 View 功能，利用 BIND 9. x 中的 View，在具体配置中通过 View 与 ACL 的协同工

作，以实现根据用户源 IP 地址智能解析对应服务器的 IP 地址的功能。如果要使同一个域名指向不同的 3 个网域，只需要在 named. conf 文件中通过 ACL 定义 3 个不同的网域，即 View 分别指向同一个域名的 3 个不同的网域，之后当处于不同 View 中的用户访问这个域名时，将通过 BIND 解析到不同网域对应的 IP 地址，从而实现了 DNS 的智能解析功能。BIND 9. x 的最新版本可在 http://www. isc. org 中下载使用。

BIND 通过对区文件(zone file)的管理实现对 DDNS(Dynamic Domain Name Server，动态域名服务)的域名授权和查询，BIND 的组成结构如图 3-7 所示。其中，named 进程是 BIND 服务器的核心，named 启动时读取初始化文件 named. conf 并配置数据文件。当 DNS 客户端的解析器发出 DNS 解析请求时，由 named 进程将查询结果(即域名对应的 IP 地址)发送给客户端。named. conf 把所有区文件绑定在一起，以便 named 进程可以根据域名查询要求通过 named. conf 中的记录来读取区文件。作为网络应用中的关键服务，named 进程在工作过程中也会根据 BIND 的配置提供日志记录。

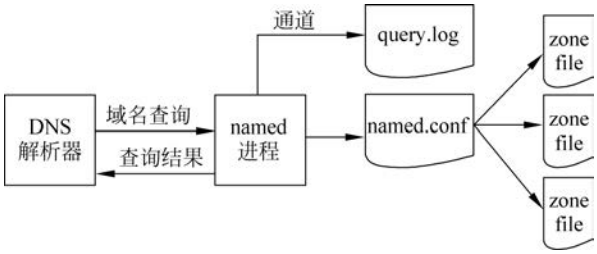

图 3-7　BIND 的组成结构

2. 一个典型的 DNS 攻击过程分析

2009 年 5 月 19 日晚，受暴风影音软件存在的设计缺陷及免费智能 DNS 软件 DNSPod 的不健壮性影响，黑客通过僵尸网络控制下的 DDoS 攻击，致使我国江苏、安徽、广西、海南、甘肃、浙江等省在内的 23 个省出现罕见的断网故障，即“5·19 断网事件”。这一事件告诫人们：在互联网中，越是由基础服务产生的安全威胁，影响力越大，范围越广。作为互联网基础服务的 DNS，每天有海量的域名解析信息产生，其个体的安全性已经直接影响着互联网的安全。与此同时，随着网络应用的不断复杂化，当潜在的条条安全暗流通过某种规则汇集在一起时，所形成的巨浪足以使正常的网络运行秩序产生混乱直至瘫痪。“5·19 断网事件”再次引起了人们对 DNS 服务及其相关安全威胁的关注。在这一事件中，僵尸网络控制下的 DDoS 攻击是问题产生的根源，免费软件的后门是问题产生的诱因，DNS 服务的脆弱性是问题产生的关键，而 DDoS 攻击、软件后门及 DNS 服务之间的内在关联是这一事件得以发生的潜在因素。

“5·19 断网事件”是多种综合因素产生的结果，其攻击过程示意图如图 3-8 所示，具体实现步骤如下。

① 攻击者(黑客)通过控制互联网上大量的“肉鸡”(被僵尸网络控制的计算机)向免费动态 DNS 服务器 DNSPod 发起 DDoS 攻击，使 DNSPod 服务器无法为正常用户提供域名解析服务，直至瘫痪。

② 因为暴风影音网站(*. baofeng. com)的域名解析使用的是 DNSPod 服务器，当

DNSPod 服务器被攻击瘫痪后，根据域名解析的递归机制，所有客户端对 *. baofeng. com 网站的解析请求将被转向 DNSPod 服务器的上一级 DNS 服务器（电信运营商 DNS 服务器）。

③ 由于暴风影音软件存在的后门（也称为“流氓化”行为），大量安装了暴风影音软件的用户计算机在联网后，不管是否启用了暴风影音软件，都会自动在后台尝试连接 *. baofeng. com 网站，并且在得不到应答的情况下，向电信运营商 DNS 服务器连续发送大量的域名解析请求报文（据称每分钟发送 100 次以上），流量达到 10GB 左右。

④ 虽然电信运营商 DNS 服务器进行了分布式部署，但大部分省份的 DNS 递归服务器无法承受如此巨大的域名解析请求报文，导致服务器 CPU、内存资源耗尽，既无法正常解析 *. baofeng. com 域名，也无法解析非 *. baofeng. com 域名，这些 DNS 服务器处于瘫痪状态。

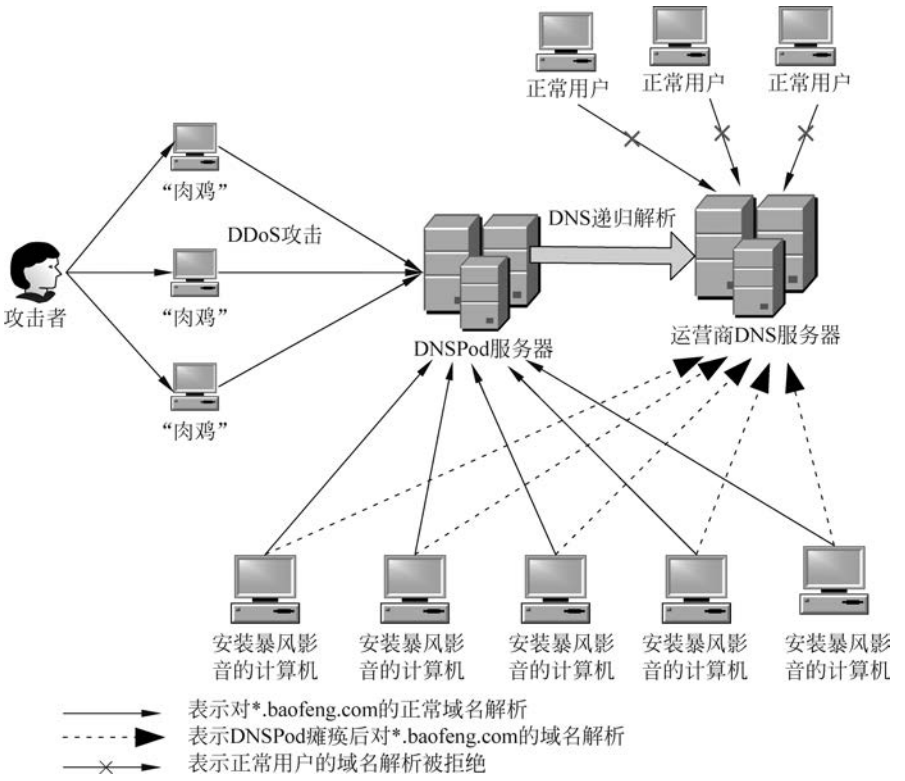

图 3-8 “5·19 断网事件”攻击过程示意图

⑤ 除直接使用 IP 地址（这种现象很少）外，绝大多数互联网用户的 DNS 域名解析请求得不到响应，从而产生断网现象。

下面对该事件进行具体分析。

（1）僵尸网络控制下的 DDoS 攻击。僵尸网络控制下的 DDoS 攻击是这一事件得以爆发的根源。僵尸网络（Botnet）是一种从传统恶意代码转化形成的新型攻击方式，它采用多种传播机制，使僵尸程序感染互联网上的大量主机，并通过一对多的命令与控制信道，控制大量僵尸主机（Bot）实现分布式拒绝服务（DDoS）攻击、信息窃取、发送垃圾邮件（Spam）等

恶意网络行为。

DDoS攻击的基本原理是黑客通过入侵并控制大量的主控端(Handler)主机(俗称“肉鸡”),并通过主控端上的代理程序(Agent)同时对被攻击者(Target)发起流量攻击。由于DDoS攻击是利用合理的服务请求来占用有限的服务资源,从而使合法用户无法得到服务的响应,因此DDoS攻击已成为目前互联网上黑客经常采用的攻击手段,也是用户很难防范的一类网络威胁。

由于DNS服务的开放性,黑客可以通过“肉鸡”上的代理向DNS服务器发起大量的DNS查询请求。黑客为了藏匿攻击者(一般为代理)的真实身份,会伪造DNS请求报文中的源地址。在“5·19断网事件”中,黑客通过控制的僵尸网络,利用互联网上数量庞大的“肉鸡”,对DNSPod服务器进行攻击,流量达到10GB左右。如此巨大的流量足以使互联网中的任何主机被淹没,造成被攻击主机无法提供网络服务,直至瘫痪。

(2) 网际间互联互通及相关问题。在“5·19断网事件”发生后,不少人将矛头指向了提供免费DNS解析的DNSPod服务器,认为是DNSPod服务器的不健壮性和私人DNS服务器提供商之间的恶意竞争直接导致了这一事件的发生。其实,持此观点者忽视了一个重要的问题:我国互联网络存在的网际间互联互通问题。

目前,国内多个运营商并存,各大运营商之间出于商业的竞争,在不同的网络之间人为设置了访问上的障碍,致使不同网络用户访问非本运营商网络内的主机时出现访问速度慢或无法访问等问题。这种“网中网”格局已严重影响了国内互联网的健康有序发展,与可持续发展的大局相违背。为了解决互联互通的问题,许多单位的无奈之举是采取了多出口方案。这种方案在国内高校校园网的应用中更加普遍,据相关资料统计,目前80%左右的高校都使用多出口接入方式,其中部分出口多达三四条。

存在互联互通这一问题,就不难理解DNSPod为什么会受到用户的普遍青睐。DNSPod是国内较早提供免费智能域名解析服务的私人DNS服务器,为各类网站提供电信、网通、教育网双线或者三线智能DNS免费解析服务,解决不同运营商网络之间访问时的互联互通问题。目前DNSPod已经管理着超过10多万用户和20多万域名,平均每天的请求量超过20亿次,其用户包括许多国内的知名网站。

为此,在分析“5·19断网事件”中的DNS查询服务时,不能只看到DNSPod的不安全性和不可靠性,应该对为什么在互联网中会出现类似DNSPod的免费智能域名解析服务器需要进行思考。其实DNSPod仅仅是事件的触发点,而事件的本因则是国内网际互联互通中游戏规则的缺失及不同运营商之间的恶性竞争。试想,如果国内电信运营商之间没有将不同的用户群进行人为的割裂,如果国内电信运营商也能提供类似的免费智能DNS解析服务,如果私人免费DNS服务商之间的竞争能够有序化,就不会发生这么严重的事件。当然,在大量网站选择DNSPod作为自己的域名解析服务器后,DNSPod已经成为网络运行的中心,DNSPod很有必要采取相应的技术手段来保障自身的安全性。对于拥有如此用户数量的DNS查询服务器,目前只有ns1.dnspod.net～ns6.dnspod.net共6台服务器,同时这6台服务器位于同一IDC(Internet Data Center,互联网数据中心)中,无法提供安全可靠的服务。

(3) 软件后门存在的安全隐患。在这次事件中,暴风影音扮演着DDoS攻击中“肉鸡”的角色。暴风影音为了获得更高的广告点击率,在后台暗藏了机关——安装了暴风影音客

户端软件的计算机在启动时会自动链接到暴风影音网站（*.baofeng.com）。而当暴风影音软件得不到 DNS 响应（DNSPod 被攻陷）时，会不断发送 DNS 解析请求报文（每分钟 100 次以上），这样在 DNS 树形结构末端的 DNS 服务器中将会汇聚大量等待应答的 DNS 请求报文。当时，暴风影音拥有数以千万计的用户数，一旦 DNS 解析出错，将会产生巨大的数据流，并对其他 DNS 服务器产生冲击。

暴风影音之所以提供以上的功能，主要出于本身的利益考虑，是由利益驱动导致的“流氓化”。当用户安装了暴风影音软件后，在操作系统中会强制随机启动一项名为 stormliv.exe 的进程。而且，即使用户在开机后没有运行暴风影音软件，也会自动运行 stormliv.exe 进程，并不断连接暴风影音网站，下载广告或在线升级。在关闭了暴风影音主程序后，stormliv.exe 进程照常驻留内存。

（4）DNS 自身的不安全因素。DNS 是互联网中的一项基本服务，早期设计上的缺陷为日后的应用埋下了安全隐患。在设计 DNS 时，由于过于强调对网络的适应性，希望在网络状况不好时照样能够使用，采用了面向非连接的 UDP，但 UDP 本身是不安全的。从体系结构来看，DNS 采用树形结构，虽然便于查询操作，但单点故障非常明显，安全威胁加剧。

除 DNS 自身易受攻击外，DNS 在应用中存在的安全缺陷和薄弱环节也逐渐显露出来。除根服务器的安全、DDoS 攻击和僵尸网络攻击外，还主要表现在以下几方面。

① 软件漏洞。不管是 DNS 服务器端软件还是客户端软件，都有可能存在安全漏洞。这些漏洞一旦被攻击者利用，就会导致域名解析服务或解析结果的错误。

② 缓存中毒。DNS 缓存中毒是指攻击者通过伪造的或错误的 DNS 记录来代替 DNS 服务器中已有的正确记录。当用户向该 DNS 服务器发送查询请求时，DNS 服务器将给出错误的应答信息或将用户的链接请求引向攻击者预设的网站。另外，一旦将缓存中毒与钓鱼网络结合起来，将会造成很大的危害性。

③ 域名劫持。域名劫持通常是指通过采用非法手段获得某一个域名管理员的账户名称和密码，或者域名管理邮箱，然后将该域名的 IP 地址指向其他的主机（该主机的 IP 地址有可能不存在）。域名被劫持后，有关该域名的记录会被改变，甚至该域名的所有权可能会落到其他人的手里。

3. DNS 安全防范方法

下面结合“5·19 断网事件”来分析 DNS 安全的防御方法。

（1）DNS 清洗服务。在“5·19 断网事件”中，当电信运营商得知 DNSPod 服务器被攻陷后，立即进行了清洗服务。根据域名解析体系中的缓存机制（逐级缓存机制），大量递归域名解析服务器中的解析记录一般至少要保存 24 小时，也就是说，即使是错误的信息也需要在 24 小时之后才能够被系统自动删除。DNS 清洗服务可以解决 DNS 缓存带来的域名错误查询问题。当发现可能引起网络故障的 DNS 错误记录时，ISP 等 DNS 服务器控制方可以采取清空缓冲区的方法，保证用户的 DNS 请求指向正确的 DNS 服务器。

在发生了“5·19 断网事件”后，电信运营商对 *.baofeng.com 域名的解析强行指向某一 IP 地址或干脆屏蔽掉其解析。不过，这只是一种应急方案，不能成为一种通用的方法。

（2）DNS 服务器的冗余备份。冗余备份是 DNS 服务器安全管理中采用的一种较为普遍的方法。即使是在局域网中，为了保障 DNS 服务的可靠性，也会采用多机备份方案。与局域网相比，互联网的结构和应用要复杂得多，对 DNS 的安全性要求会更高。在“5·19 断

网事件”中，DNSPod 服务器之所以能够被全部快速攻陷，其主要原因是 6 台服务器同时位于同一个 IDC 中。为此，对于互联网中的 DNS 服务器，建议能够创建位于不同 IDC 中的分布式系统。

(3) 根域名服务器的安全管理。DNS 是一种分布式的网络名称服务系统，其树形结构在便于扩展的同时，也带来了安全威胁，尤其是针对根域名服务器的攻击。互联网中的根域名服务器存储和控制着全球域名解析体系的主体信息，是整个域名解析递归结果的终结点。由于根服务器是整个域名系统的核心，所以根服务器的可靠性在很大程度上决定着树形结构中各级递归服务器的安全性。目前，全球有 13 台根服务器，1 台为主根服务器，放置在美国，其余 12 台均为辅助服务器，其中 9 台放置在美国，2 台在欧洲(分别位于英国和瑞典)，1 台在亚洲(位于日本)。加强根域名服务器安全的一种通用方法是在不同的国家和地区建立根域名服务器的镜像站点。在国家工信部的协调下，截至 2019 年，我国已经引入了 12 个根域名服务器的镜像服务器，在提高了国内域名解析性能的同时，加强了根域名服务器的安全性。

(4) 对软件漏洞的管理。软件漏洞主要包括操作系统的漏洞和 DNS 应用软件的漏洞。目前，互联网上 DNS 服务器的操作系统主要有 BSD、Linux、Solaris 及少量的 Windows，而 DNS 应用软件绝大多数使用的是 ISC 公司的 BIND。大量的攻击充分利用了操作系统和应用软件存在的漏洞。针对利用漏洞的攻击，最有效的防范方法是升级到最新的版本，并及时安装补丁程序。

(5) DNSSEC 的部署。DNSSEC(Domain Name System Security Extensions，域名系统安全扩展)是在原有的域名系统(DNS)上通过公钥技术，对 DNS 中的信息进行数字签名，从而提供 DNS 的安全认证和信息完整性检验。DNSSEC 被公认为是目前解决 DNS 服务安全最有效的方法，但由于 DNSSEC 需要使用数字签名系统，部署较为复杂，且扩展性较差，在大范围推广时存在一定的难度。可喜的是：目前 ICANA 已经在部分根域系统(如 .org)上部署了 DNSSEC，以提高根域服务器的安全性，此举说明 DNSSEC 已从理论探讨和区域性试验开始走上实际应用了。

(6) DDoS 攻击的防范。近年来，采用 DDoS 对 DNS 服务器的攻击不断出现。对于采用树形结构的 DNS 体系，域名节点越是靠近根，所受到的 DDoS 攻击威胁也就越严重。解决 DDoS 攻击的最有效方法有以下 4 种：一是部署 IDS(Intrusion Detection Systems，入侵检测系统)，从单一技术和设备来看，IDS 是目前防范 DDoS 攻击最有效的方法；二是对于重要的 DNS 服务器，可分别在不同的 IDC 中部署，通过冗余方式来提高 DNS 的安全；三是在防火墙上通过设置策略，对于超过某一限定值的 DNS 请求报文进行过滤；四是通过管理软件，对排名靠前的 DNS 解析请求报文进行分析，重点分析那些流量在短时间内急剧增大的报文，对可疑报文进行过滤处理。

4. 针对基于 BIND 软件的 DNS 的安全管理方法

虽然 BIND 对 DNS 提供了大量的安全防范，但是如果配置不当或没有进行必要的安全设置，其安全性仍然无法得到体现。下面结合 Linux 系统中对 BIND 软件的配置，介绍常见的安全管理方法。

(1) 正确地配置 DNS 服务器。在 Linux 系统中，DNS 服务由 named 守护进程进行控制，该进程从主文件“/etc/named.conf”中获取具体的配置信息。除此之外，还有许多与之相关的配置文件，如根域名配置服务器指向文件“/var/named/named.ca”、用户配置区正向

解析文件“/var/named/name2ip. conf”、Localhost 区正向域名解析文件“/var/named/localhost. zone”等。Linux 系统中基于 BIND 软件的 DNS 配置是由一组文件组成的，在具体配置过程中不但要清楚不同文件的功能及存放位置，而且要掌握不同文件的配置方法，同时还要熟悉不同配置文件之间的关系。一旦一个配置存在缺陷，将会留下安全漏洞。在安装 BIND 软件包时，系统自动安装了用于对 DNS 配置文件进行检查的工具，如 nslookup、dig、named-checkzone、host、named-checkconf 等，熟悉这些工具的功能及应用，对检查 DNS 配置的正确性、防止出现安全漏洞是很有帮助的。

（2）隐藏 BIND 的版本号。对目标主机的操作系统类型及版本号等信息进行搜集是网络攻击前需要完成的一项工作内容，只有掌握了目标主机的详细信息，才能从中发现可利用的漏洞。一般情况下，通用软件的设计缺陷是与特定的版本相关的，所以版本号的搜集对攻击者来说是十分关键的。攻击者可以利用 dig 命令查看 BIND 软件的版本号，进而知道该版本的 BIND 软件存在哪些漏洞。为此，隐藏 BIND 的版本号是很有必要的，具体可在配置文件“/etc/named. conf”的 option 部分添加 version 声明，将系统默认显示的版本号覆盖掉。例如，可通过以下配置，当利用 dig 查看版本号时，显示为“The platform does not provide version queries”。

```
options {
version"The platform does not provide version queries"
}
```

同时在 DNS 配置文件避免使用 HINFO 和 TXT 资源记录，可以使攻击者无法得到 DNS 服务器的相关信息。

（3）控制区域（zone）传输。DNS 区域传输（zone transfer）是指备用服务器通过主服务器的数据来更新自己的区域（zone）数据库。出于服务的可靠性考虑，一般不会仅提供一台 DNS 服务器，而是通过设置主/从（master/slave）DNS 服务器来实现安全备份功能。当设置了主、从备份服务器后，从服务器需要从主服务器中读取并更新自己的区域数据库，这便是 DNS 的区域传输操作。区域传输的主要对象是区域数据库，该数据库保存着网络架构中的主机名、主机 IP 地址列表、路由器名、路由 IP 列表，以及各主机所在位置和硬件配置等重要信息。

在 BIND 的默认配置中，区域传输是全部开放的，即 DNS 服务器允许对任何主机进行区域传输操作。如果攻击者假冒备用 DNS 服务器，向指定主 DNS 服务器（攻击主机）请求进行区域传输，就会收集到该 DNS 服务器所在网络架构中的所有配置信息。为了加强对 DNS 服务器的安全保护，需要严格限制允许区域传输的主机，一般一个主 DNS 服务器只允许它的从 DNS 服务器执行区域传输操作。对于 BIND 软件，可以通过如下的 allow-transfer 命令来控制。

```
acl"zone-transfer"{172.16.1.0;172.16.1.254}
zone"yourdomain.cn"{
type master;
file"yourdomain.cn";
allow-transfer {zero-transfer;};};
```

这样，只有 IP 地址在 17.16.1.0～172.16.1.254 的主机才能够同 DNS 服务器进行区域传输操作，限制了其他主机的操作。

(4) 限制反向解析请求。在 DNS 系统中，一个 IP 地址可以对应多个域名，即多个域名可以同时指向同一个 IP 地址。因此，由 IP 地址来查询域名，理论上是可行的，但实际上是不现实的，因为这种查询操作会遍历整个域名树，这在 Internet 上是不现实的。为了避免类似操作的发生，DNS 提供了一个被称为“反向解析域”(in-addr.arpa)的区域，由该区域负责向需要从 IP 查询域名的请求提供应答服务。例如，一个 IP 地址为 210.98.95.2 的反向解析域名表示为 2.95.98.210.in-addr.arpa，反向解析域名与 IP 地址正好相反，同时在后面加上了.in-addr.arpa。因为域名结构是自底向上(从子域到根域)的，而 IP 地址结构是自顶向下(从网络到主机)的。实质上反向域名解析是将 IP 地址表达成一个域名，以地址作为索引的域名空间。

如果任何用户都可以向 DNS 服务器发送反向解析请求报文，这无异于给 DNS 服务器实施 DoS 攻击提供帮助。所以，需要限制 DNS 服务器的反向解析服务，只允许特定 IP 地址范围内的主机使用该服务。例如，通过以下设置，只允许 IP 地址在 172.16.1.0 网段的主机使用该 DNS 服务器提供的反向地址解析服务。

```
options{
allow-query {172.16.1.0/24};
};
zone "yourdomain.cn"{
type master;
file "yourdomain.cn";
all-query{any;};
};
zone "1.16.172.in-addr.arpa" {
type master;
file "db.172.16.1";
allow-query {any;}; };
```

限制反向解析服务的范围，除能够有效保护 DNS 服务器外，还可以拒绝接收所有没有注册域名的 IP 地址发来的邮件。目前，多数垃圾邮件发送者使用动态分配或者没有注册域名的 IP 地址来发送垃圾邮件，以逃避追踪。因此，在邮件服务器上拒绝接收来自没有域名的 IP 地址发来的邮件可以大大降低垃圾邮件的数量。

3.3.4 Apache 服务器的攻防

Windows 和 Android 分别在桌面操作系统和移动智能终端领域的广泛应用，使得它们成为攻击者的主要选择目标。此现象充分说明攻击者只会选择有利用价值的目标对象，而使用越广泛的系统才潜藏着可被利用的价值。同样，在 Web 服务器应用中，Apache 的大量部署使其成为攻击者在互联网 Web 应用领域的主要研究对象和攻击目标。

1. 针对 Apache 服务器常见攻击方式

攻击者选择 Apache 服务器，主要借助 Apache 软件自身存在的安全漏洞和错误的配

置，同时还利用了传统的 DoS、缓冲区溢出等方式攻击，借助 HTTP/HTTPS 设计上的不严谨性实现攻击行为。

（1）泛洪攻击。泛洪（flood）攻击是一种中断网络服务的常见攻击方法，通常通过发起 ICMP（Internet Control Message Protocol，Internet 控制报文协议）包或 UDP（User Datagram Protocol，用户数据报协议）包实施具体的攻击行为。通过向目标主机发送泛洪数据包，使目标主机或连接主机的网络负载过重，进而无法提供正常的网络服务。要实施泛洪攻击，攻击者的网络带宽一定要大于被攻击主机所使用的网络带宽。

使用 UDP 数据包进行攻击是利用了 UDP 的工作原理。当攻击者发送了 UDP 数据包后不会有任何数据包返回到攻击者的主机，这种基于单向数据流的工作机制很适合攻击者通过向目标主机发送大量的数据包来迫使其无法提供正常的服务。使用 ICMP 数据包进行攻击是利用了该协议可以根据不同的应用需求来构造不同的数据包这一特点，攻击者通过构造有缺陷的数据包来扰乱正常的网络工作机制。泛洪攻击的本质是攻击者通过欺骗目标主机，让其相信所接收到的数据包都是正常的。

（2）硬盘攻击。不论是机械硬盘还是固态硬盘，其总体结构都是相似的。硬盘主要由处理器、缓存、Boot ROM 和主存储介质等几部分构成，对于机械硬盘还有电机驱动电路和磁头控制电路等部件。由于硬盘的电路板上已经具有了 CPU、内存和 ROM，所以可以将硬盘看作是一个小型的计算机系统，在固件的控制下独立运行。硬盘通电时，处理器执行片段内的 Loader 代码，这部分代码会加载 Boot ROM 到缓存中并执行。Boot ROM 得到控制权后，会依次初始化基本外设，初始化主存储介质，从主存储介质上加载固件主体，启动 IDE/SATA 总线接口驱动模块，并进入待命状态，此时计算机即可对硬盘进行操作。

目前，大部分硬盘都支持固件升级功能（通过下载微码命令或者厂商的私有命令实现），用户可以通过厂商提供的程序来对硬盘驱动器上的固件进行更新。这使得硬盘厂商无须召回有固件缺陷的产品，就可以在用户系统上通过软件工具升级固件来修补缺陷。由此不难看出，如果固件缺陷被利用，就会对磁盘产生破坏性的结果。通过伪造的固件更新程序来写入攻击指令，轻则硬盘中的数据泄露，重则硬盘损坏。

（3）DDoS 攻击。DDoS（Distributed Denial of Service，分布式拒绝服务）攻击是目前针对 Apache 等互联网上的 Web 服务器威胁性最大的一种攻击方式。DDoS 攻击过程中一般会隐藏攻击数据的来源，即使是被攻击者觉察后也很难追溯到数据的源头。由于 Apache 应用的广泛性，攻击者专门开发了针对 Apache 的攻击程序（如 SSL 蠕虫），然后利用 Apache 代码存在的漏洞，通过正常的网络访问将攻击程序安装在 Apache 服务器上。之后，攻击者便可以根据需要，在被感染的主机上执行恶意代码，发起对特定目标的 DDoS 攻击。

（4）分块编码远程溢出。Apache 在处理以分块（chunked）方式传输数据的 HTTP 请求报文时存在设计上的缺陷，如攻击者可能会利用此缺陷在某些 Apache 服务器上以 Web 服务器进程的权限执行任意指令或进行 DoS 攻击。

分块编码（chunked encoding）传输方式是 HTTP 1.1 中定义的 Web 用户向服务器提交数据的一种方式。当服务器收到分块编码方式的数据时会分配一个缓冲区来存放它，如果提交的数据大小未知，客户端会以一个协商好的分块大小向服务器提交数据。

Apache 服务器默认也提供了对分块编码的支持。Apache 使用了一个有符号变量保存

分块长度，同时分配了一个固定大小的堆栈缓冲区来保存分块数据。出于安全考虑，在将分块数据复制到缓冲区之前，Apache 会对分块长度进行检查，如果分块长度大于缓冲区提供的长度，Apache 将最多只复制缓冲区长度的数据，否则将根据分块长度进行数据复制。然而在进行上述检查时，没有将分块长度转换为无符号型进行比较。因此，如果攻击者将分块长度设置成一个负值，就会绕过上述安全检查，Apache 会将一个超长的分块数据复制到缓冲区中，将会造成一个缓冲区溢出。此漏洞可导致各种操作系统下运行的 Apache Web 服务器的拒绝服务。

(5) 获取远程用户权限。在安装了 Apache 软件后需要指定一个执行账户，因为有些配置文件或程序必须是 root 身份才能运行，所以 Apache 的执行账户有些需要以 root 身份运行 Apache。如果 Apache 以 root 权限运行，系统上一些存在逻辑缺陷或缓冲区溢出漏洞的程序会使攻击者很容易地获取 Linux 服务器上的 root 权限。在一些远程情况下，攻击者会利用一些以 root 身份执行的有缺陷的系统守护进程来获取 root 权限，或者利用有缺陷的服务进程漏洞来取得普通用户权限，用以远程登录 Linux 服务器，进而控制整个系统。

2. 安全防范方法

对基于 Linux 系统的 Apache Web 服务器，最有效的安全管理方法是关注 Linux 系统和 Apache 软件的缺陷，及时升级系统或安装补丁程序。同时，还可以通过以下方法来对 Apache 服务器进行安全配置。

(1) 隐藏 Apache 版本。因为软件的漏洞信息与特定的版本是相关联的，所以搜集被攻击对象的软件版本信息是实施攻击的前提。默认系统下，系统会把 Apache 版本信息通过 HTTP 应答头部显示出来，并没有提供任何的信息保护机制。隐蔽 Apache 版本信息的具体方法是修改 Apache 的配置文件“/etc/httpd. conf”，在找到 ServerSignature 和 ServerTokens 关键字后，将其设定为 ServerSignature Off 和 ServerTokens Prod，然后重启 Apache 服务器。

(2) 创建安全目录结构。Apache 服务器包括多个目录，表 3-2 列出了其主要目录的功能及安全配置建议。

表 3-2 Apache 服务器主要目录的功能及安全配置建议

目 录 名	功 能	安全配置建议
ServerRoot	保存 Apache 的配置文件、二进制文件和其他服务器配置文件	只能由 root 用户访问
DocumentRoot	保存 Web 站点的内容，包括 HTML 文件和图片等	只能由管理 Web 站点内容的用户和使用 Apache 服务器的 Apache 用户、Apache 组访问
ScripAlias	保存 CGI 脚本	只能由 CGI 开发人员和 Apache 用户访问
Customlog	保存访问日志	只能由 root 用户访问
Errorlog	保存错误日志	只能由 root 用户访问

(3) 为 Apache 分配专门的执行账户。为避免因使用 root 作为 Apache 的执行账户带来的安全问题，一般在对 Apache 配置结束后需要分配一个专用的执行账户，不再使用 root。Apache 账户权限的分配遵循“最小特权原则”，即要求该账户对系统及数据进行访问

时只拥有必需的最小权限。保证用户能够完成所操作的任务，同时也确保将非法用户或异常操作所造成的损失降到最小。

(4) Web 目录的访问控制。在 Web 服务器中，将需要发布的 Web 站点的文件保存在 Web 目录中，需要确保其安全，防止非授权访问和非法篡改。

Apache 服务器在接收到用户对一个目录的访问请求时，会查找 DirectoryIndex 指令指定的目录索引文件，默认情况下该文件为 index.html。如果该文件不存在，那么 Apache 会通过创建一个动态列表为用户显示该目录的内容。通常这样的配置会暴露 Web 站点的结构，因此需要修改配置“/etc/httpd/conf/httpd.conf”，搜索 Options Indexes FollowSymLinks，修改为 Options-Indexes FollowSymLinks 即可。其中，在 Options Indexes FollowSymLinks 的 Indexes 前面加上“－”符号表示禁止目录索引，如果是“＋”符号则表示允许目录索引，FollowSymLinks 表示允许使用符号链接。

(5) 利用.htaccess 加强对 Apache 服务器的安全管理。.htaccess 是 Apache 服务器上的一个基于文本的分布式配置文件，它提供了针对目录改变的配置方法，即将包含一些操作系统的.htaccess 文件保存在某一特定目录后，该目录及其下的所有子目录都会受到该文件的影响(index.html 文件除外)。.htaccess 通过自行修改其文件内容来实现权限控制，主要应用于为网页访问设置密码、自定义错误页面、改变首页的文件名(如 index.html)、禁止读取文件名、重定向文件等。下面通过几个实例来说明.htaccess 文件的配置和应用。

① 自定义错误页面。当用户访问某一网站时，不合理的访问或网站自身存在问题时，会出现不同的错误返回页面。攻击者可以通过该错误返回页面中的信息来了解 Apache 服务器的有关配置情况，并以此作为某种判断的依据。可以借助.htaccess 来控制错误返回页面信息的显示内容。HTTP 的错误代码被标准化定义为 400～505，但通过对.htaccess 的配置，可以使 Web 服务器处理错误时能够进行个性化的定制，而不是被协议标准化的默认页面。配置错误页面的重定向语法如下。

```
ErrorDocument[error code][url]
```

其中，“error code”为错误代码；“url”为指定保存自定义错误信息的页面所在的地址。例如，如果在当前目录下有一个保存自定义错误信息的页面文件 payattention.html，使用它作为 404 错误页面，可以写为：

```
ErrorDocument 404/payattention.html
```

404 错误页面是客户端在浏览网页时，服务器无法正常提供信息，或者服务器无法回应且不知道原因所返回的页面，而利用.htaccess 文件则可以对其进行任意的修改。具体操作时，只需要将 payattention.html 和.htaccess 两个文件同时上传到指定的目录中即可。

② 网站目录的密码保护。要使用.htaccess 进行 Web 站点所在目录的密码保护，可通过两个步骤来实现：配置.htaccess 文件和创建.htpasswd 密码文件。.htaccess 文件的相关内容如下。

```
AuthName "Section Name"
AuthType Basic
AuthUserFile /full/path/to/.htpasswd
Require valid-user
```

其中，“Section Name”将出现在用户端弹出页面的密码输入框中，可以自行定义；“/full/path/to/.htpasswd”是密码文件.htpasswd的绝对路径。密码文件.htpasswd的内容格式为“username:password”；“Require valid-user”表示.htpasswd文件中设置的任何一个合法用户都可以访问。

通过以上的设置，当用户试图访问被.htaccess文件密码保护的目录时，浏览器会弹出要求输入账户名和密码的对话框，只有当正确输入后才能够访问。

③ 限制来访主要的IP地址范围。对于只需要对特定人群（特定IP地址范围）开放的Web站点，可在.htaccess中对指定IP进行限制，有效防止其他用户访问该站点。例如：

```
Order deny, allow
deny from all
allow from 172.16
```

通过以上设置，表示只允许172.16.0.0网段的用户访问该站点，其他用户都将被拒绝。

通过上述的几个实例可以看出，使用.htaccess来保护网站更为安全和方便。因为利用.htaccess实现密码保护，可以有效地抵御字典攻击和暴力破解。

3.4 Linux用户提权方法

通过远程渗透技术，攻击者可以获得系统的远程访问权限，并能够实现远程登录。在完成了远程登录后，攻击者就转向对本地主机的攻击。本地主机攻击过程中最重要的是用户权限的提升。

3.4.1 通过获取“/etc/shadow”文件的信息来提权

Linux系统的账户分为特权账户root、普通用户账户和系统用户账户三大类，并采用VFS（Virtual File System，虚拟文件管理）来控制每个用户对文件的访问。出于安全考虑，在一些Linux发行版本中用特别分配的系统用户账户来启动和运行网络服务，只有一些频繁访问系统资源的特殊网络服务（如Samba）才直接使用root账户权限运行。

需要特别说明的是，为了养成安全使用Linux系统的习惯，建议系统管理员使用普通用户账户来登录和操作Linux系统，只有确实需要使用root权限来配置和管理系统时，再通过su、su -或sudo命令将权限提升到root用户账户。对于普通用户账户，坚持最小权限分配原则，一般禁用root账户权限。

通过获取“/etc/shadow”文件的内容来对本地用户进行权限提升，主要分为获取“/etc/shadow”文件和破解“/etc/shadow”文件以获得用户密码两个过程。

1. 获取“/etc/shadow”文件

通过远程渗透方法，攻击者如果获得了 root 账户的登录密码且系统允许 root 账户远程登录，那就可以直接登录系统进行任意的操作。但是，由于 root 账户的重要性，大部分情况下其登录密码是很难获得的，攻击者一般得到的是普通用户账户的登录权限。普通用户账户对系统的操作是受限的，一般很难完成预定的操作，这时就需要通过提权技术，将普通用户账户的权限提升到 root 权限。

在早期的 Linux 版本中，包括 root 在内的所有账户信息（包括用户名和对应的密码）全部保存在“/etc/passwd”文件中，并且普通用户也可以读取该文件的内容。当 Linux 系统引入了“the Shadow Suit”组件后，将用户账户的密码加密后单独存放在“/etc/shadow”文件中，而且只有 root 用户才能够读取该文件中的信息。

如何才能够得到 shadow 文件的内容呢？首先要能够得到 shadow 文件，然后再对 shadow 文件进行破解。因为只有具有 root 权限的用户才能够读取 shadow 文件，在无法直接获得 root 权限用户账户信息的前提下，可借助一些以 root 权限运行的服务中存在的文件任意读写漏洞来间接获得。当具有 root 权限运行的程序中存在代码任意执行安全漏洞时，可以代替攻击者主动打开具有 root 权限的 shell 命令行连接，有了该连接就可以读取“/etc/shadow”文件。攻击者在获得了 shadow 文件后再通过破解其密码以获取 root 用户的密码。

2. 破解“/etc/shadow”文件

用户密码破解是网络攻击中的一个最基本的操作，然而由于系统的复杂性和多样性，这一操作的实现却要视不同的系统和配置来确定不同的思路和方法。

Linux 系统中的“/etc/shadow”和“/etc/passwd”文件中的记录是一一对应的，每行都记录着 Linux 系统中的一个用户账户的登录信息。下面显示的是 shadow 文件中一个用户账户的登录凭证密文信息。

```
root:$1$0QPP9BPb$ZG1h9LtbwX12p.CwrWJ8..:15534:0:99999:7:::
```

它由多个字段组成，不同字段之间用“:”隔开。其中，最前面的一个字段“root”表示登录名，与“/etc/passwd”中相同行的用户名一致。“$1$0QPP9BPb$ZG1h9LtbwX12p.CwrWJ8..”存放的是加密后的用户密码，如果该字段为空或“!”，表示该账户没有设置密码；如果为“*”，则表示该用户无法从终端登录，一般应用于服务器端运行账户。除以上特殊情况下，该字段则以“$”作为分隔符，又分为使用算法编号、salt 值和加密后的密码 Hash 值。使用算法包括系统默认的 DES 算法、MD5 算法（显示为“$1”）、Blowfish 算法（显示为“$2”或“$2a”）和 SHA 算法（显示为“$5”或“$6”）。salt 值的长度范围为 2～12 字符，不同的算法长度不同。加密后的密码 Hash 值长度取决于所使用的加密算法，长度范围为 13～24 字符，且使用 Base64 进行编码。其他字段不再进行说明。

从上面关于“/etc/shadow”组成主要字段的介绍中不难发现，无论采取经典的 DES 或 MD5 算法，还是安全性更高的 SHA-256 或 SHA-512 算法，随着随机数 salt 值的加入，该加密机制使攻击者无法直接从密文反推出其明文密码，尤其是 salt 值的应用使彩虹表攻击方法无法实现。从而只有字典攻击和暴力破解两条路径可供选择。John the Ripper 是 Linux

系统上进行密码暴力破解常用的工具，该工具还提供了合成“/etc/passwd”与“/etc/shadow”后再进行破解的专门程序。

3.4.2　利用软件漏洞来提权

在无法获取“/etc/shadow”文件信息的情况下，可以利用 Linux 系统软件中存在的漏洞来完成提权操作。

1. 利用 sudo 程序的漏洞进行提权

su、su -和 sudo 是 Linux 系统提供的管理指令，是让普通用户执行一些或者全部 root 命令的一个工具。然而这些工具在设计与实现上可能存在安全漏洞，当本地提权漏洞被攻击者利用后，就可以将一个普通用户提升为一个具有 root 权限的特权用户。

2017 年 5 月，Linux 系统中的 sudo 程序被发现存在一个高危漏洞（CVE-2017-100036），该漏洞发生在 Linux 的 sudo 命令的 get_process_ttyname()函数中。攻击者可以利用这个漏洞，让普通用户在使用 sudo 命令获得临时权限时执行一些操作，将他们的权限提升到 root 级别。在运用 SELinux 机制的 Linux 系统上，sudo 用户可以使用命令行的输入提升自己的用户权限，还可以覆盖文件系统中的文件，甚至覆盖由 root 用户所拥有的文件。该 Linux 本地提权漏洞会影响从 1.8.6p7 到 1.8.20 的所有版本。

2. 利用 SUID 程序漏洞进行提权

对文件设置了 SUID 后，执行者将获得文件所有者所拥有的权限。由于一个服务和系统软件在运行过程中需要频繁地访问系统资源，而系统资源的访问需要拥有 root 权限。但是出于系统安全考虑，不会给每一个需要访问系统资源的程序都赋予 root 权限，而是仅在需要的时候才赋予，访问结束后将收回。SUID 机制实现了这一安全功能。

Linux 系统中的每一个进程在调用时都会拥有真实 UID（Real User ID）和有效 UID（Effective User ID），其中真实 UID 指的是进程执行者是谁，而有效 UID 指的是进程执行时继承的是谁的访问权限，即某一用户（真实 UID）在用另一用户（有效 UID）的权限来执行某一程序。一般情况下，普通用户在调用进程时，真实 UID 和有效 UID 是统一的。但是在某些特殊情况下，普通用户（真实 UID）会在继承了 root 用户（有效 UID）的权限后去执行某一特殊操作。这种特殊情况是通过为程序设置 SUID 特殊权限位来指定的，给某一程序设置了 SUID 位之后，普通用户在执行这一程序时，调用该进程的有效 UID 就变成了该程序拥有者的 UID（一般为 root 用户），该进程则在继承了拥有者权限（一般为 root 的权限）后执行。

下面以 Linux 系统中 passwd 程序为例来说明 SUID 特殊权限的功能及实现过程。在 Linux 系统中，任何一个普通用户都可以修改自己的密码，这一操作是通过 passwd 程序来实现的。现在的问题是：用户账户信息保存在“/etc/passwd”文件中，而用户密码则经加密处理后保存在“/etc/shadow”文件中，普通用户对“/etc/passwd”文件仅拥有读权限，只有 root 用户才拥有对“/etc/passwd”文件的写权限，而“/etc/shadow”文件只允许 root 进行读、写操作。也就是说，普通用户对“/etc/passwd”文件和“/etc/shadow”文件都没有写权限。那么，为什么普通用户能够修改自己的密码呢？这就要依靠 SUID 来实现。passwd 程序权

限位的设置类似于“-rwsr-r-x 1 root root 30768 Jul 22/2018/usr/bin/passed”（可用“ls-l/usr/bin/passwd”命令查看），即 passwd 的拥有者是 root，且拥有者权限里面本应是 x 的那一列显示的是 s，这说明 passwd 程序具有 SUID 权限。一个具有执行权限的文件在设置了 SUID 权限后，当用户在调用这个文件时将以文件所有者身份执行。也就是说，passwd 程序具有 SUID 权限，该程序的所有者为 root，当普通用户使用 passwd 命令修改自己的密码时，实际以 passwd 程序的拥有者 root 的身份作为该进程的有效 UID 在执行，自然具有对“/etc/passwd”文件和“/etc/shadow”文件的写入权限，passwd 命令执行结束后继承来的 root 权限将自动被解除。当用户在命令提示符下输入了 passwd 命令后，在输入密码前按下 Ctrl＋Z 组合键，再执行 pstree -u 命令，在图 3-9 所示的进程树中会发现 passwd 进程的权限不是 wq-js 而是 root。

图 3-9　查看进程树

需要指出的是，系统管理员可以根据需要来为普通用户在执行某些程序时调用特殊权限，具体可通过 chmod 命令来设置程序的 SUID 权限位。同时，默认情况下 Linux 系统中有一些程序本身就拥有 SUID 权限位。也就是说，Linux 系统中拥有 SUID 权限位的程序比较多，一旦其中存在安全漏洞的程序被利用于本地提权攻击，就会给攻击者提供具有 root 权限的 Shell，将攻击者使用的账户添加到 root 组中，其破坏性不言而喻。

另外，可以利用 SUID 程序的本地缓冲区溢出进行提权攻击。缓冲区溢出一般是针对设置了 SUID 权限位且用户拥有者为 root 的程序，以便在溢出之后通过向目标程序中注入经攻击者特意构造的攻击代码，并以 root 用户权限来执行命令，给出 Shell。

还有，可针对 SUID 程序的共享函数库实现本地提权攻击。Linux 系统中的共享函数库（shared libraries）是以.so 为后缀的类似于 Windows 系统动态链接库（以.dll 为后缀）的

一种函数库动态加载机制，它允许可执行文件在执行阶段从某个公共的函数库中调用一些功能代码片段。共享函数库中的函数在一个可执行程序启动时被加载，所有的程序在重新运行时都可以自动加载最新的函数库中的函数。使用共享函数库能够帮助系统程序更加有效地利用一些功能模块，并使得代码的维护更加容易。如果攻击者能够利用某些广泛使用的共享函数库中存在的安全漏洞，或者通过设置环境变量提供具有恶意功能的共享函数库，就可以攻击依赖这些共享函数库的 SUID 程序，从而获得本地 root 权限，实现提权操作。Linux 是用 C 语言编写的，glibc 是 Linux 下 GUN 的 C 函数库，glibc 除了封装 Linux 系统所提供的系统服务外，大量的 SUID 程序都依赖于 glibc。为此，一旦 C 函数库的实现中存在安全漏洞，攻击者就有可能实施本地提权攻击，其攻击手段类似于 Windows 环境中的 DLL 注入攻击。

3. 利用 Linux 内核代码漏洞进行提权

对于任何一个操作系统来说，受其运行环境的限制，能够提供的访问资源是有限的，过量或无序的访问会导致资源的耗尽或出现访问冲突。为解决这一问题，UNIX/Linux 对不同的操作赋予不同的执行等级(特权)，Intel x86 架构的 CPU 提供了 0～3 共 4 个特权级，数字越小，等级越高，Linux 操作系统中主要采用了 0 和 3 两个特权级，分别对应的是内核态和用户态。运行在内核态的进程可以执行任何操作并且在资源的使用上没有限制，而运行于用户态的进程可以执行的操作和访问的资源都会受到限制。出于资源有效利用和系统安全的考虑，很多程序开始时运行于用户态，但在执行的过程中，当需要在内核权限下才能够执行时，就涉及从用户态切换到内核态，类似的应用在前文已经有了介绍(如 SUID 的使用)。

不管是运行在内核态的代码，还是运行在用户态的应用程序，以及位于用户态的进程向内核的调用，甚至是程序在运行过程中从用户态向内核态的切换，都会存在程序漏洞或操作机制上的安全隐患。尤其是 Linux 的内核代码，因其具有开源性而成为攻击者研究的主要对象。一旦发现内核代码中存在高危提权漏洞，攻击者便可以方便地对用户进行提权操作，并实现对大量主机系统的操作，其利用价值和产生的威胁是可想而知的。

2016 年 1 月，Linux 系统被发现在内核中存在一个高危级别的本地提权 0day 漏洞(编号为：CVE-2016-0728)，该漏洞属于 Linux 平台上的 UAF(Use After Free)漏洞。其中，UAF 漏洞的产生根源是迷途指针(dangling pointer)，已分配的内存释放之后，其指针并没有因为内存释放而变为 NULL，而是继续指向已释放内存。如果是良性迷途指针，该指针不会再被使用；而如果是恶性迷途指针，则该指针还会被用来对已释放内存进行读/写操作。CVE-2016-0728 漏洞的产生，主要是由于 keyrings 组件中的引用计数问题。keyrings 的主要功能是为驱动程序在内核中保留或缓存安全数据、身份认证密钥、加密密钥及其他数据。它使用一个 32 位的无符号整数做引用计数，但是在计数器出现溢出的时候没有进行合理的处理。当对象的引用计数达到最大时会变成 NULL，因此释放对象的内存空间。而此时程序还保留对引用对象的引用，所以形成了 UAF 漏洞，可实现对本地用户的提权操作。该漏洞影响 Linux 内核 3.8 及以前版本，已影响到大量的 Linux 个人计算机、服务器及大量安卓设备(包括智能手机和平板电脑)。

3.4.3 针对本地提权攻击的安全防御方法

针对本地提权攻击，最有效的安全防范方法依然是及时更新系统的补丁程序，以便在第一时间修补存在的安全漏洞。除此之外，结合本节介绍的几类提权攻击方法，下面主要基于Linux服务器的应用，从系统管理的角度提些建议。

针对SUID特权程序，管理员首先要清楚Linux系统在默认安装时，哪些系统程序使用了SUID特权位设置，程序如果不需要就尽可能将其禁用。对于在Linux系统上运行的应用程序，管理员必须知道是否会启用SUID特权位设置，并评估可能存在的安全风险。对于安全风险大的SUID特权程序，应尽可能去除SUID特权位的设置，如果确实要使用，必须实时关注其安全状况。即使是安全风险小的SUID特权程序，管理员也要做到“清单式”管理，即对使用的SUID特权程序建立应用清单，及时安装安全补丁程序。

针对利用代码漏洞进行本地提权的问题，最根本的解决办法还是及时升级操作系统并安装补丁程序，同时辅助以必要的安全配置。例如，禁止root用户进行远程登录、对特权用户设置强口令、使用SSH对服务器进行远程管理等。

另外，针对Linux在访问控制机制中存在的本地提权漏洞，可使用SELinux安全增强模块来提高Linux抵御本地攻击的能力。早期的Linux采用自主访问控制(Discretionary Access Control，DAC)来保证系统的安全性，根据用户标识和拥有者权限来确定是否允许访问。这种机制的缺陷是忽略了用户的角色、数据的敏感性和完整性、程序的功能和可信性等安全信息，因此不能提供足够的安全性保证。而Linux在2.6内核之后集成了SELinux组件，在该组件中引入了强制访问控制(Mandatory Access Control，MAC)机制，可以有效地解决早期Linux系统中存在的一些问题。MAC根据用户操作对象(如普通文件、目录、设备、端口、被调用的进程等)所含信息的敏感性，以及用户操作(如读、写、执行等)在访问这些信息时的安全授权来限制对用户操作对象的访问。SELinux是一种通用的、灵活的、细粒度的MAC机制，为用户操作和用户操作对象定义了多种安全策略，能够最大限度地限制进程的权限，保护进程和数据的安全性、完整性和机密性，从而解决了DAC的脆弱性和传统MAC的不灵活性等问题。

习题

1. 简述Linux系统的安全机制及主要实现方法。
2. 分析PAM技术的实现过程，并简述其应用特点。
3. 分析Linux的权限分配特点及访问控制机制的实现方法。
4. Linux环境中的用户账户分为哪几类？如何获取其信息？如何进行安全防范？
5. 结合安全漏洞的概念，分析漏洞在网络远程渗透攻击过程中发挥的功能，以及如何进行安全防范？
6. 以本章介绍的“5·19断网事件”为例，详细分析DNS服务器存在的安全隐患及攻击的实现过程。在此基础上，结合Linux环境下BIND软件的配置方法，简述DNS服务器的安全防范措施。

7. 结合实际部署的 Apache Web 服务器，通过具体的操作，分析其存在的主要安全缺陷及其可能产生的结果，并简述其安全防范方法。

8. 简述分块编码远程溢出的原理及实现方法。

9. 通过实际操作，掌握利用. htaccess 对 Apache 服务器进行安全保护的方法。

10. 简述 Linux 系统中对普通用户账户进行提权的方法。

11. 通过实际操作，在掌握 SUID 特殊权限位功能及设置方法的基础上，以 Linux 系统中的 passwd 程序为例来说明 SUID 特殊权限的实现过程和应用特点。