**1.操作系统面临的安全威胁有哪些？简要说明。**

**要点：（1）病毒修改其他程序或把自身植入其中，具备自我复制功能；蠕虫同样有传染性和复制能力，但不需要寄生，复制时是自身完整拷贝；木马伪装自身身份，隐藏攻击行为；（2）逻辑炸弹是指特定条件满足时实施破坏的计算机程序；（3）隐蔽通道是指系统中不受安全策略控制的、违反安全策略的信息泄露路径；（4）后门嵌入操作系统的非法代码，能够绕过系统安全策略，获取非法权限。**

**参考答案：**

操作系统是一种系统软件，不可避免地存在缺陷或漏洞，因此容易遭到各种入侵攻

击的威胁。威胁操作系统安全的因素很多，主要包括以下几个方面：

(1)病毒、蠕虫和木马。

病毒是一种能够自我复制的计算机程序，通过修改其他程序将自身或其演化体插入其中，从而感染它们。病毒可以破坏计算机功能，损毁计算机上保存的数据和文件，从而达到攻击计算机的目的。

蠕虫类似于病毒，都具有传染性和复制功能，因此，其带来的破坏与病毒一样严重。但与病毒不同，蠕虫是一个单独的程序，可以独立存在，无须寄生在某个程序之中，其复制时是自身的完整拷贝。同时，不同于病毒通过被感染文件运行来感染目标文件，蠕虫是利用系统自身存在的漏洞来传染。

木马是一段计算机程序，表面上在执行合法的功能，实际上却完成了用户不曾料到的功能，如控制一台计算机。木马程序与一般的病毒不同，它不会自我复制，也不“刻意”地去感染其他文件，它通过伪装自身，吸引用户下载执行，向实施者打开被感染主机的门户，使实施者可以任意毁坏、窃取被感染主机的文件，甚至远程操控被感染主机。

(2)逻辑炸弹。

逻辑炸弹是指在特定逻辑条件满足时，实施破坏的计算机程序。该程序触发后造成计算机数据丢失，更改视频显示，计算机不能从硬盘或者软盘引导，甚至会使整个系统瘫痪，并出现物理损坏的假象。与病毒不同，它不能进行自我复制，不具有传染性，它针对的是特定对象，但破坏力巨大。

(3) 隐蔽通道。

隐蔽通道是指系统中不受安全策略控制的、违反安全策略的信息泄露路径。按信息传递的方式区分，隐蔽通道分为隐蔽存储通道和隐蔽定时通道。隐蔽存储通道在系统中通过两个进程利用不受安全策略控制的存储单元传递信息。前一个进程通过改变存储单元的内容发送信息，后一个进程通过观察存储单元的变化来接收信息。隐蔽定时通道在系统中通过两个进程利用一个不受安全策略控制的广义存储单元传递信息。前一个进程通过改变广义存储单元的内容发送信息，后一个进程通过观察广义单元的变化接收信息，并用如时钟这样的坐标进行测量。广义存储单元只能在短时间内保留前一个进程发送的信息，后一个进程必须迅速地接收广义存储单元的信息，否则信息将消失。判别一个隐蔽通道是否是隐蔽定时通道，关键是看它有没有一个实时时钟、间隔定时器或其他计时装置，不需要时钟或定时器的隐蔽通道是隐蔽存储通道。

(4) 天窗

天窗又称后门，是嵌在操作系统里的一段非法代码，它能绕过系统安全策略控制而获取对程序或系统的访问权，一般很难发现。天窗由专门的命令激活，渗透者利用该代码提供的方法侵入操作系统而不受检查。通常天窗在操作系统内部，而不在应用程序中。天窗只能利用操作系统的缺陷或混入系统开发队伍进行安装。一个有效、可靠的操作系统不得有天窗，必须提供相应的保护措施，消除或限制病毒、蠕虫、木马、逻辑炸弹和隐蔽通道等对系统构成的安全隐患。

**2.为操作系统提供的硬件安全机制有哪些？简要说明。**

**要点：（1）利用虚地址空间和占用内存时的上下文切换进行存储保护；（2）设定分层的运行域，隔离操作系统程序和用户程序；（3）依靠操作系统安全机制和硬件的结合，对读写分别进行安全保护。  
参考答案：**

1. 存储保护

存储保护是指保护用户在存储器中的数据，它是操作系统安全最基本的要求。存储保护的精度取决于保护单元的大小，保护单元越小，存储保护的精度越高。保护单元可以是字、字块、页面或段，它是存储器中最小的数据存储范围。对于在内存中一次只能运行一个进程的操作系统，存储保护机制应能够防止用户程序对操作系统的影响。而对于在内存中可以一次运行多个进程的多任务操作系统，还需要进一步要求存储保护机制对各进程的存储空间进行隔离。存储保护与存储管理是密切联系在一起的，存储保护机制保证系统中各进程之间互不干扰，而存储管理则是为了更有效地利用存储空间。

（1）虚地址空间。在多任务操作系统中，每个进程的运行需要一个独立的存储空间，进程的程序和数据都存储在该空间中，这个空间不包括该进程通过 I/O 指令访问的辅助存储空间（如磁盘、硬盘等），即虚地址空间。虚地址空间由内核空间（Kernel Space）和用户模式空间（User Mode Space）两部分组成，两者是静态隔离的。虚拟地址会通过页表（Page Table）映射到物理内存，页表由操作系统维护并被处理器引用，每个进程都有自己的页表。内核空间是持续存在的，在所有进程中都映射到同样的物理内存，并且在页表中拥有较高特权级给操作系统使用，因此用户态程序试图访问这些页时会导致一个页故障（page fault）。而用户模式空间的映射随进程切换的发生而不断变化。

（2）内存管理。程序在系统模式下运行时，允许对所有的虚地址空间进行读写操作，无论是内核空间还是用户模式空间，而在用户模式下运行时，则禁止非特权进程直接向内核空间进行写操作，但用户代码可以通过调用系统服务来间接地访问内核空间中的数据或间接执行内核空间中的代码。当调用系统服务时，调用线程会从用户模式切换到内核模式，调用结束后再返回用户代码，这就是所谓的模式切换，也被称为上下文切换。

2. 运行保护

操作系统安全很重要的一点是分层设计，而运行域正是这样一种基于保护环的等级域结构。运行域是进程运行的区域，在最内层具有最小环号的环具有最高特权，而在最外层具有最大环号的环是最低的特权环，一般系统不少于 3~4 个环。运行保护是隔离操作系统程序与用户程序，保证进程在运行时免受同等级运行域内其他进程的破坏。

3.I/O 保护

一个进行 I/O操作的进程必须受到对设备读、写两种访问控制。这就意味着设备到 I/O 介质间的路径可以不受什么约束，而处理器到设备间的路径则需要实施一定的读写访问控制。但是若对系统中的信息提供足够的保护，防止被未授权用户的滥用或毁坏，只靠硬件是不能提供充分的保护手段的，必须由操作系统的安全机制与适当的硬件相结合才能提供强有力的保护。

**3. 操作系统安全的主要目标是什么？实现操作系统安全目标需要建立哪些安全措施？**

**参考答案:**

操作系统安全的主要目标是：对系统中的用户进行身份认证，依据系统安全策略对用户的操作进行访问控制，防止用户对计算机资源的非法访问，监督系统运行的安全，保证系统自身的安全性和完整性。实现操作系统安全目标需要建立相应的安全机制，包括：存储保护，用户标识与认证，访问控制，最小权限管理，可信路径和安全审计等。

**4.什么是最小特权管理？为什么在操作系统中，需要采用最小特权管理？**

**要点：（1）系统不分配给用户超过执行任务所需特权以外的特权，或仅给用户赋予必不可少的特权。（2）为了防止赋予权限过大时，执行不符合系统安全策略的操作、影响系统运行的操作和误操作。**

**参考答案：**

最小特权管理的思想是系统不应给用户超过执行任务所需特权以外的特权，或仅给用户赋予必不可少的特权。最小特权原则一方面赋予主体“必不可少”的特权，以保证用户能完成承担的任务或操作；另一方面它仅给用户“必不可少”的特权，从而限制用户所能进行的操作。同时，为了保证系统的安全性，不应对某个用户赋予一个以上的职责，而一般系统中的超级用户通常肩负系统管理、审计等多项职责，因此，应将超级用户的特权划分为一组细粒度的特权，分别授予不同的系统操作员或管理员，使各种系统操作员或管理员只具有完成其任务所需的特权，从而减少由于特权用户口令丢失或错误软件、恶意软件、误操作所引起的损失。

在操作系统中，为了维护系统正常运行及其安全策略的实施，往往需要为某些特殊用户赋予一定的特权以执行一些受限的操作或进行违反安全控制策略的操作，例如执行软件安装，维护用户账号等。所谓特权是指可违反系统安全策略的一种操作能力，它与访问控制相结合，提供系统的灵活性。在现有的一般多用户操作系统中，都存在一个超级用户，它拥有所有特权，可以执行任意操作，如Windows操作系统中的超级用户Administrator，UNIX 和 Linux 操作系统中的超级用户 ROOT。普通用户不具有任何特权。这种特权管理方式便于系统的维护和配置，却不利于系统的安全性。一旦超级用户的口令丢失或超级用户被冒充，将会对系统造成极大的损失。另外，超级用户的误操作也是系统极大的潜在安全隐患。因此，TCSEC 标准要求 B2 级以上的安全操作系统必须提供最小特权管理机制，以确保系统安全。

**5.Windows 操作系统的身份认证方法？**

**要点：（1）需验证用户、对象和服务的身份；（2）本地认证根据本地计算机账户确认身份，网络认证根据用户试图访问的网络服务确认用户身份。**

**参考答案：**

Windows 系统中，身份认证除了用户登录操作系统验证身份外，还需要验证对象和服务的身份。当你对某个对象进行身份验证时，是在验证该对象是否为正版。当你对某个服务或用户进行身份验证时，在于验证出示的凭据是否可信。按照登录的方式不同，Windows 操作系统提供了两种基本的身份认证类型，即本地认证和网络认证。其中，本地认证是根据用户的本地计算机账户确认用户的身份，而网络登录是根据用户试图访问的网络服务确认用户的身份。在 Window 7 及以上的版本中，采用凭据提供（Credential Provider）登录模块取代传统的 GINA（Graphical Identification and Authentication）登录模块，其登录验证如图 6.5所示。在登录验证中，涉及 Winlogon 模块、LogonUI 模块、CredentialUI 模块、LSA 模块、SAM 模块和第三方开发的凭据提供程序。本地用户登录的身份认证过程如下：

（1）用户首先按下 Ctrl＋Alt＋Delete 组合键。

（2）Winlogon 模 块（Winlogon. exe 和 Secure32. dll）检 测 到 用 户 按 下 SAS（Secure Attention Sequence）热键，就调用 CredentialUI 模块（CredUI.dll），并向其发送凭据请求。

（3）CredentialUI 模块收到凭据请求后，通过凭据程序提供接口向凭据提供程序发送凭据请求，并从凭据提供程序获得凭据信息。

（4）CredentialUI 模块收到凭据信息后，显示登录 UI界面（LogonUI模块）。

（5）用户选择类型并输入自己掌握的凭据（用户名和口令）到登录界面。

（6）logonUI模块收到用户输入的凭据后，将其发送给 CredentialUI 模块。

（7）CredentialUI 模块通过凭据程序提供接口将处理后的用户输入请求发送给凭据提供程序。然后凭据提供程序将凭据通过凭据程序提供接口返回给 CredentialUI 模块。

（8）CredentialUI 模块将凭据发送给 Winlogon 模块。

（9）Winlogon 模块将收到的凭据发送给 LSA 模块进行验证。

（10）LSA 模块收到用户登录的凭据后，将调用 Msc1\_0 模块（Msc1\_0.dll是验证程序包），将用户凭据生成 Hash 值，发送给 SAM 模块。

（11）SAM 模块将收到的 Hash 值与存储的 HSAH 值进行比对。如果比对后，发现用户身份合法，SAM 将用户安全标识 SID、用户所属用户组的 SID 和其他一些相关信息发送给 LSA 模块。

（12）LSA 将收到的 SID 信息创建安全访问令牌，然后将安全访问令牌和登录信息发送给 Winlogon 模块。Winlogon 模块对用户登录稍作处理后，完成整个本地身份认证过程。

所有用户登录的凭据信息都保存在 SAM 数据库中，而 SAM 数据库一般保存在“C:\Windows\system32\config”文件夹下的 SAM 文件中，该文件不能删除，如果删除将直接导致系统无法登录。SAM 文件记录的数据包括：所有组和账户的信息，口令 HASH，账户的 SID 等。该文件受到操作系统保护，不能被直接打开。

**6.简述 Linux 系统的安全机制。**

**要点：(1)** **组和用户的口令与其他信息分离,采用shadow文件机制，并提供文件一致验证方法；（2）根据文件的3种访问权限和3种用户类型，利用加密文件系统保证文件系统安全；（3）跟踪所有进程的活动，以及它们对系统资源的使用情况以保证进程的活性和系统资源不被浪费；（4）审计和监测系统日志；（5）SELinux对 root 账号采用强制访问控制机制，同时限制用户程序和系统服务器完成任务的最低权限.**

**参考答案：**

（1）用户和组文件安全。

组账号文件和用户账号文件，为了加强口令的安全性，采用组和用户的口令与其他信息分离的安全机制。

Linux 采用“shadow 文件”机制，将加密口令转移到/etc/shadow 文件中，只有 root 超级用户可读。而在/etc/passwd文件中，口令字段处只存放一个“x”或“\*”，从而最大限度地减少了密文被泄露的机会。加密文件系统。同时Linux提供了pwck和grpck两个命令分别验证用户和组文件，以保证两个文件的一致性和正确性。

（2）文件系统安全。

Linux规范了文件和目录的访问权限分为只读、只写和可执行 3 种。有 3 种不同类型的用户可对文件或目录进行访问：文件所有者，同组用户，其他用户。所有者一般是文件的创建者，它可以允许同组用户有权访问该文件，还可以将文件的访问权限赋予系统的其他用户。在这种情况下，系统中的每一位用户都有可能访问该用户拥有的文件或目录。同时采用CFS、TCFS、AFS、eCryptFS、ReiserFS 等加密文件系统保护用户的数据。

（3）进程的安全

Linux 要跟踪所有进程的活动，以及它们对系统资源的使用情况，从而实施对进程和资源的管理。可以说，Linux 的安全与进程是息息相关的。在这里，不能把进程和程序混为一谈，虽然在操作系统中运行的任何程序都可以叫作进程，但程序是静态的，进程是动态的，而且多个进程可以并发地调用一个程序。

根据进程的特点和属性，Linux 系统中的进程可以分为 3 类：交互进程，批处理进程和守护进程（即在后台执行的系统服务）。每个进程有 3 种基本状态：运行态（R），就绪态（W）和封锁态（S）（或称挂起态）。在进程的启动、执行、挂起和恢复过程中，通过进程和进程资源的安全管理，防止进程僵死，避免进程消耗过多的系统资源。

（4）日志管理安全。日志子系统起到审计和监测的功能，记录系统每天发生的各类事情，包括哪些用户曾经或正在使用系统，可以通过日志检查错误发生的原因。一旦系统受到攻击后，日志可以记录攻击者留下的痕迹，通过查看这些痕迹，系统管理员可以发现攻击者攻击的某些手段和特点，为抵御下一次攻击做好准备。

（6）安全增强

为了解决root权限过大带来的安全问题，用户可以采用 SELinux（Security Enhanced Linux）。SELinux 控制 root 权限，对 root 账号采用强制访问控制机制，同时限制用户程序和系统服务器完成任务的最低权限。

**7.简述 Android 和 Linux 操作系统安全机制之间的异同。**

**要点：（1）Android继承扩展了Linux 内核安全模型的用户与权限机制。但通过应用程序沙盒隔离应用，避免操作系统内存损坏;（2）Android 通过 AndroidManifest.xml 文件可以赋予应用额外的、细粒度的权限；（3）与Linux 的 CORBA 类似，Android 系统也采用一个基于抽象接口的分布式组件架构——Binder，但与前者不同的是，Binder 机制更加灵活，可靠性更高。它运行在单个设备上，不支持 RPC 网络远程调用；（4）Android 沿用了 Linux 安全增强技术 SELinux。**

**参考答案：**

Android操作系统是建立在Linux内核基础之上的，其体系结构如图 6.11所示。Android的内核除了提供 Linux 标准版的硬件驱动、网络、文件系统访问控制和进程管理等功能外，还增加了一些新的特征，如：低内存管理机制，唤醒锁机制（集成进了 Linux 内核的唤醒机制中），匿名共享内存（ashmen），闹钟，网络访问控制机制和 Binder 机制等。其中 Binder 机制和网络访问控制机制是 Android 操作系统两个非常重要的新增机制，Binder 机制实现了 IPC（进程间通信）及其相关安全机制，网络访问控制机制（paranoid networking）严格限制了特定权限的应用对网络套接字的访问能力。

Android 平台的基础是 Linux 内核，因此 Android 的安全机制也是建立在 Linux 内核的安全基础之上的，它为 Android 系统提供了 4 个方面的安全支持：基于用户权限的安全模型，进程隔离，提供安全 IPC 的扩展接口，移除内核中没有必要或有安全隐患的部分。Android 系统在继承 Linux 安全机制的同时，也要考虑其移动平台的特殊性，结合应用需求，Android 系统主要在以下几个方面提高系统的安全性。

(1) 应用程序沙盒

Android 继承和扩展了 Linux 内核安全模型的用户与权限机制。在每个应用程序安装阶段，赋予一个独立的 UID，通常称为 app ID。应用执行时，就在特定进程内以该UID 运行。同时，无论该进程是在原生还是在虚拟机内执行，每个应用都有一个只有它才有读写权限的专用数据目录，由此建立了内核级的应用沙盒。

在 Linux 内核之上的程序库、虚拟机、系统服务和应用都可以运行在沙盒内。在一些操作系统中内存损坏是严重的安全隐患，但在 Android 系统中由于应用的隔离，内存的损坏只影响其对应的程序，其他程序不受影响。

(2) 访问权限

由于 Android 应用是沙盒隔离的，它们只能访问自己的文件和一些设备上可全局访问的资源，但这将限制 Android 的应用，为此，Android 通过 AndroidManifest.xml 文件可以赋予应用额外的、细粒度的权限，从而控制硬件设备、网络连接、数据或系统服务的访问。

(3) IPC

Android 使用内核驱动和用户空间层的组合来实现 IPC 机制，称为 Binder 机制。与Linux 类似，Android 进程的地址空间是独立的，一个进程不能直接访问另一个进程的内存空间，即进程隔离。这种机制给系统带来稳定性和安全性的同时，也阻隔了一个进程为另一个进程提供服务，因此，Android 系统需要 IPC 机制帮助进程能够发现为其提供的服务并与之交互的进程。与 Windows 的 COM 和 Linux 的 CORBA 类似，Android 系统也采用一个基于抽象接口的分布式组件架构——Binder，但与前者不同的是，Binder 机制更加灵活，可靠性更高。它运行在单个设备上，不支持 RPC 网络远程调用。

(4) SeLinux

传统的 Android 安全模型严重依赖于应用的 PID，虽然这是由内核提供的功能，并且每个应用的文件都是私有的，但无法完全禁止应用自身将它的文件改成全局可访问的文件。同理，也无法禁止恶意应用程序利用系统文件或原生套接字过于宽松的访问标志位进行攻击。为此，Android 沿用了 Linux 安全增强技术 SELinux。

**8.POSIX是IEEE为Linux定制的可移植操作系统接口，请简述POSIX的权能机制。**

**要点：（1）权能的含义；（2）POSIX权能机制的目的；（3）权能机制的含义**

**参考答案**：

权能是一种用于实现恰当特权的能力令牌。POSIX权能机制是一种基于权能的最小特权控制机制，其目的是实现比超级用户模式更细粒度的授权控制。它将特权划分成了一个权能集合。

POSIX权能机制提供了更为便利的权能管理和控制：一是提供了为系统进程指派一个权能去调用或执行受限系统服务的便捷方法；二是提供了一种使进程只能调用其执行特定任务必须权能的限制方法，支持最小特权安全策略的实现。它通过进程和程序文件权能状态(许可集、可继承集、有效权能集)，明确定义了进程如何获取和改变权能的语义，从而提供了一种基于进程所运行上下文控制进程权能的方法。