面经预热~Day1

进程、线程和协程

进程、线程和协程是计算机中用于实现并发执行的概念。

进程（Process）是操作系统中的一个程序执行实例。它拥有独立的内存空间和系统资源，可以独立运行。每个进程都有自己的地址空间和数据栈，它们之间的通信需要通过进程间通信（IPC）机制。进程之间的切换开销较大。

线程（Thread）是进程中的一个执行流程，一个进程可以拥有多个线程。线程共享进程的地址空间和系统资源，可以并发执行。线程之间切换的开销较小，因为它们共享相同的上下文。

协程（Coroutine）是一种用户级的轻量级线程。协程通过协作式调度，可以在同一个线程中实现多个协程的并发执行。协程由程序员控制调度，可以手动挂起和恢复，因此在切换时开销更小。协程适用于需要高度并发和高效利用资源的场景。

总结来说，进程是操作系统分配资源的基本单位，线程是操作系统调度执行的基本单位，而协程是程序员控制的执行流程。它们在并发编程中有不同的特点和适用场景。

堆和栈

堆（Heap）和栈（Stack）是计算机内存中用于存储数据的两个重要区域，它们在内存管理和数据存储方面有着不同的特点。

栈是一种自动分配和释放内存的数据结构，它具有先进后出（LIFO）的特性。栈中存储的数据是以函数调用的形式组织的，每个函数的局部变量和函数参数都会被存储在栈帧中。栈的大小是有限的，通常较小，由操作系统分配和管理。栈的分配和释放速度非常快，但是空间有限，一旦栈溢出就会引发程序崩溃。

堆是一种动态分配和释放内存的数据结构，它没有特定的存储顺序。堆中存储的数据由程序员手动分配和释放，通常用于存储动态创建的对象和数据结构。堆的大小通常比栈大，并且可以动态增长。堆的分配和释放速度相对较慢，但是它提供了更大的灵活性和存储空间。

在内存中，栈和堆是相对独立的，栈位于高地址空间，向下增长，而堆位于低地址空间，向上增长。栈的内存管理由编译器和操作系统自动完成，而堆的内存管理需要由程序员显式地进行操作，包括分配和释放内存。

总结来说，栈用于存储函数调用和局部变量等固定大小的数据，具有自动分配和释放的特性。堆用于存储动态分配的对象和数据结构，具有手动分配和释放的特性，并且提供了更大的灵活性和存储空间。

TCP三次握手和四次挥手的过程，可不可以只挥手两次?

TCP（传输控制协议）是一种面向连接的可靠传输协议，它使用三次握手和四次挥手来建立和终止连接。

1. 三次握手（TCP三次握手）：
    - 第一步：客户端向服务器发送一个带有SYN（同步）标志的连接请求报文段，该报文段中包含客户端的初始序列号（ISN）。
    - 第二步：服务器接收到连接请求后，回复一个带有SYN/ACK（同步/确认）标志的报文段作为确认，并且分配资源以建立连接。
    - 第三步：客户端收到服务器的确认后，再次向服务器发送一个带有ACK（确认）标志的报文段，确认连接已建立。此时，双方都可以开始发送数据。
2. 四次挥手（TCP四次挥手）：
    - 第一步：当客户端想要关闭连接时，发送一个带有FIN（结束）标志的报文段，表示不再发送数据。
    - 第二步：服务器接收到客户端的关闭请求后，发送一个带有ACK标志的报文段作为确认，并进入到CLOSE_WAIT状态，表示服务器已经关闭发送数据的能力。

- 第三步：当服务器准备关闭连接时，发送一个带有FIN标志的报文段，表示服务器也不再发送数据。
- 第四步：客户端接收到服务器的关闭请求后，发送一个带有ACK标志的报文段作为确认，并进入到TIME_WAIT状态，等待一段时间（两倍的最大报文段生存时间），确保服务器接收到了确认报文段。

至于为什么需要四次挥手，而不是只挥手两次，原因如下： 在关闭连接时，双方还可能有未传输完的数据包，所以需要进行双向的关闭过程，确保数据完整性。另外，为了防止已经关闭连接的一方收到已失效的连接请求，需要等待一段时间（TIME_WAIT状态），以确保网络中的所有报文段都被正确处理。

如果只进行两次挥手，无法保证数据的完整性和可靠性，可能导致一方认为连接已关闭，而另一方仍然发送数据。因此，TCP采用四次挥手来确保连接的正常关闭。

讲讲http的3xx状态码

HTTP（超文本传输协议）的3xx状态码用于重定向和缓存控制。当客户端发送请求时，服务器可能会返回3xx状态码来指示客户端应该采取进一步的操作。

以下是常见的HTTP 3xx状态码及其含义：

1. 300 Multiple Choices（多种选择）： 服务器端有多个可供选择的响应，客户端可以根据需要选择一个合适的响应。
2. 301 Moved Permanently（永久重定向）： 请求的资源已永久移动到新的URL，客户端应该更新其链接并使用新的URL发起请求。
3. 302 Found（临时重定向）： 请求的资源暂时移动到了一个不同的URL，客户端应该继续使用原始URL进行请求。
4. 303 See Other（查看其他位置）： 请求的资源可以在另一个URL下找到，客户端应该使用GET方法请求新的URL获取资源。
5. 304 Not Modified（未修改）： 客户端发送了一个带有条件的GET请求，服务器返回此状态码表示资源未被修改，可以使用客户端缓存的版本。
6. 307 Temporary Redirect（临时重定向）： 类似于302状态码，但要求客户端保持请求方法不变进行重定向。

7. 308 Permanent Redirect（永久重定向）： 类似于301状态码，要求客户端保持请求方法不变进行重定向。

这些3xx状态码用于在不同的情况下指示客户端如何处理重定向和缓存。客户端在收到这些状态码后，根据具体情况采取适当的行动，例如更新URL、重新发起请求或使用缓存版本等。

http和https

HTTP（超文本传输协议）和HTTPS（安全超文本传输协议）是用于在客户端和服务器之间传输数据的协议。

HTTP是一种明文传输的协议，数据在传输过程中是以明文形式进行传输的。这意味着攻击者可以截获和查看通过HTTP传输的数据，存在安全风险。因此，HTTP适用于对安全性要求不高的场景，如普通网页浏览、无需保密的信息传输等。

HTTPS则是在HTTP的基础上增加了安全性的协议。它通过使用SSL（安全套接字层）或TLS（传输层安全）协议对数据进行加密和身份验证，确保传输过程中的数据安全。HTTPS使用公钥加密来保护数据的机密性，并使用数字证书来验证服务器的身份。这样，即使攻击者截获了HTTPS传输的数据，也无法解密和篡改其中的内容。

使用HTTPS的主要优点包括：

1. 数据安全：通过加密传输，保护数据的机密性和完整性。
2. 身份验证：使用数字证书验证服务器的身份，确保通信双方的真实性。
3. 排名优化：搜索引擎通常更倾向于将使用HTTPS的网站排名更高。

虽然HTTPS提供了更高的安全性，但也会带来额外的计算和网络传输开销。因此，HTTPS主要用于对数据安全性要求较高的场景，如电子商务、在线支付、用户登录等需要保护用户隐私和敏感信息的应用。

mysql AP复制

MySQL的AP（可用性和性能）复制是一种数据库复制模式，它强调数据的可用性和性能，而不强调强一致性。在AP复制中，主数据库的更改会异步地复制到多个从数据库，从数据库可以提供读取操作，但可能存在数据的不一致性。

在MySQL的AP复制中，主数据库接收到写操作后，会将更改记录到二进制日志（binary log）中，并异步地将这些更改传播给连接的从数据库。从数据库会根据二进制日志中的更改信息来更新自己的数据。这种异步复制的方式可以提高性能和可用性，因为主数据库处理写操作时无需等待从数据库的确认。

然而，由于异步复制的特性，从数据库的数据可能会存在一定的延迟，这意味着读取从数据库时可能读到的是稍旧的数据。此外，如果主数据库发生故障或网络中断，从数据库可能无法及时复制最新的更改，导致数据的不一致性。

AP复制适用于那些对于数据的实时性要求不高，但对于可用性和性能要求较高的应用场景。例如，社交媒体应用中的即时消息功能，用户可以实时地发送和接收消息，而数据的一致性并不是最重要的因素。

需要注意的是，AP复制模式下的数据一致性问题需要应用程序开发人员自行解决，例如通过在应用层进行数据冲突检测和解决，或者使用其他技术手段来确保数据的一致性。

Linux常用命令 查看内核版本有什么命令?

在Linux中，有一些常用的命令可以用来控制进程和线程。以下是其中一些常见的命令：

1. ps：显示当前正在运行的进程的信息。可以使用不同的选项来获取不同级别的详细信息，例如ps aux用于显示所有进程的详细信息。
2. top：实时显示系统资源使用情况和运行中的进程。可以查看CPU、内存、进程列表等信息，并且可以动态地排序和监控进程。
3. kill：发送信号给指定的进程。可以使用kill命令终止一个进程，常用的信号是SIGTERM（默认）和SIGKILL（强制终止）。
4. pkill：根据进程名或其他属性终止进程。可以使用pkill命令根据进程名或其他属性终止一个或多个进程。
5. killall：根据进程名终止进程。与pkill类似，killall命令根据进程名终止一个或多个进程。
6. nice：设置进程的优先级。nice命令可以调整进程的优先级，其中较高的值表示较低的优先级。
7. renice：修改正在运行的进程的优先级。renice命令可以修改正在运行的进程的优先级，需要提供进程ID或进程名。

8. bg：将进程放到后台运行。可以使用bg命令将一个正在前台运行的进程切换到后台运行。
9. fg：将进程切换到前台运行。可以使用fg命令将一个在后台运行的进程切换到前台运行。
10. jobs：显示当前终端会话中的作业（进程）。可以使用jobs命令列出当前终端会话中正在运行或暂停的作业。

这些命令提供了对进程和线程进行控制和管理的基本功能。通过使用它们，您可以查看正在运行的进程、终止进程、调整进程优先级等。请注意，在某些情况下，执行这些命令可能需要root或sudo权限。

要查看Linux操作系统的内核版本，可以使用以下命令之一：

1. uname -r：这个命令会显示当前正在运行的内核版本号。
2. uname -a：这个命令会显示更详细的系统信息，包括内核版本、操作系统版本、硬件架构等。
3. cat /proc/version：这个命令会显示有关内核的详细信息，包括内核版本、编译日期、编译器等。

无论您选择使用哪个命令，都会在终端输出中看到Linux内核的版本号。请注意，这些命令可能需要root或sudo权限才能正确执行。

OSI TCP/IP模型

OSI（开放系统互联）模型和TCP/IP模型都是网络通信领域中常用的参考模型，用于描述和理解计算机网络中的各个层次和协议。下面是它们的对应关系：

OSI模型的七层：

1. 物理层（Physical Layer）
2. 数据链路层（Data Link Layer）
3. 网络层（Network Layer）
4. 传输层（Transport Layer）
5. 会话层（Session Layer）
6. 表示层（Presentation Layer）
7. 应用层（Application Layer）

TCP/IP模型的四层：

1. 网络接口层（Network Interface Layer）：对应于OSI模型的物理层和数据链路层。
2. 网络层（Internet Layer）：对应于OSI模型的网络层。
3. 传输层（Transport Layer）：对应于OSI模型的传输层。
4. 应用层（Application Layer）：对应于OSI模型的会话、表示和应用层。

需要注意的是，OSI模型是一个理论上的参考模型，而TCP/IP模型则是实际应用中更为常用的模型。TCP/IP模型是根据实际的互联网协议栈发展而来的，它将原本的七层模型简化为四层，更贴近实际的网络通信架构。