1、Binder的用途

2、原理，Binder通信机制流程，涉及的四个重要角色

3、 Binder 与AIDL的关系，IPC

4、使用场景：在系统中的应用

service的使用

第三方登录为什么要用到service （Demo）

总结出怎么使用。

1. Binder是android中一种特有的跨进程通信方式，它模糊了进程的边界，淡化了进程通信的过程。 整个进程过程就像运行在一个OOP面向对象的程序当中。

Binder就像黏贴、粘接的胶水一样。

Binder英文翻译为胶水。

你调的使用binder，就能拿到别的进程的服务。

应用场景：媒体播放、音视频捕获、智能传感器（加速度、方位、温度、亮度等），

甚至是startActivity、startService都用到了binder。

存在一个Server： 提供服务，并负责一些管理。

而APP应用程序作为一个client客户端：主要与这些server建立连接，就可以轻松的使用服务。



1. 为什么会陌生呢？

因为没有涉及到多进程的通信。

即使涉及到了，也是使用AIDL(接口定义语言)，专门为binder设计的，是一种框架。

AIDL是一种架构，能够为我们使用binder的时候，更加的轻便，简单。

进行了很好的封装。

startActivity()内部就是使用了AIDL。

1. 为什么要学习Binder？

以为你之前是从事应用层开发。

如果你要从事硬件交互开发，如果你要从事系统层开发，会发现binder无处不在。

1. binder可以看成是提供特定服务的接入点

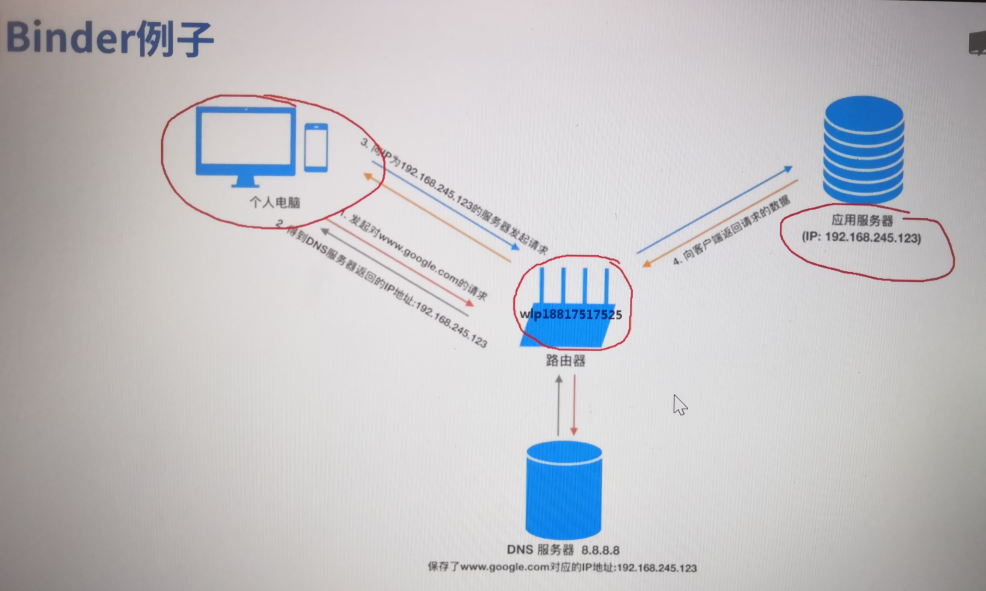
client通过binder向服务器发起请求，我要使用你的这个功能。

pc相当于我们的客户端

应用服务器是我们的server

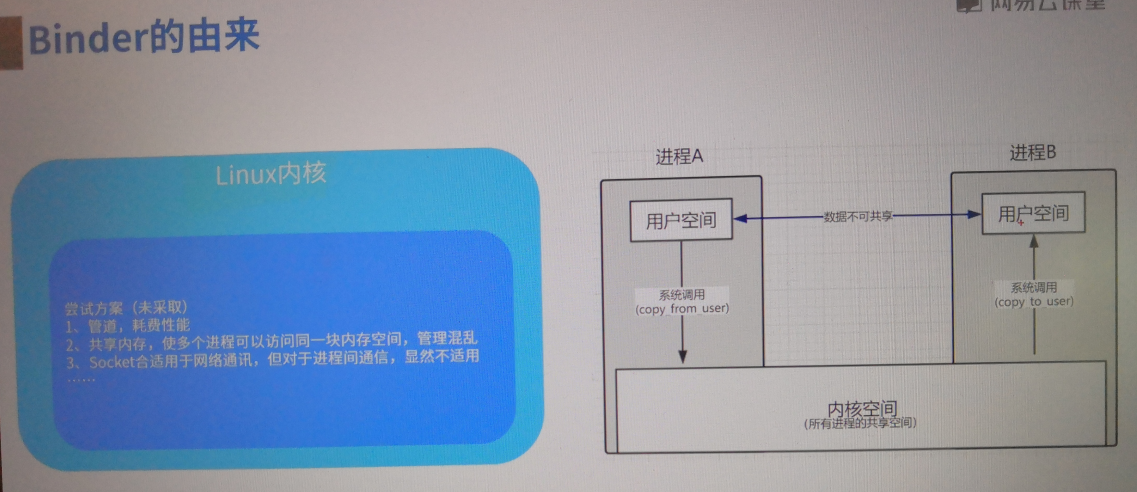
binder就像是路由器

ServiceManager就像是DNS服务器



5、Linux尝试的进程通信：

管道、socket、内存共享等。



管道，半双工管道：非常消耗性能。

共享内存：管理混乱。

socket：杀鸡用牛刀，适用于网络通信。

管道中的一种是使用传统的IPC。如图右边示意。没有安全性的校验，只要知道管道的接入点，就可能被进行无节操的篡改。

IPC 进程间通信(Inter Process Communication)

而，android9.0中，startActivity() 使用Binder机制，会验证UID(用户APPID )、PID(目标进程ID)。

所以：提出需求--> 高效率、安全高的进程间通信。

**就是Binder了，是C/S 模型，只需要一次的内存copy。**

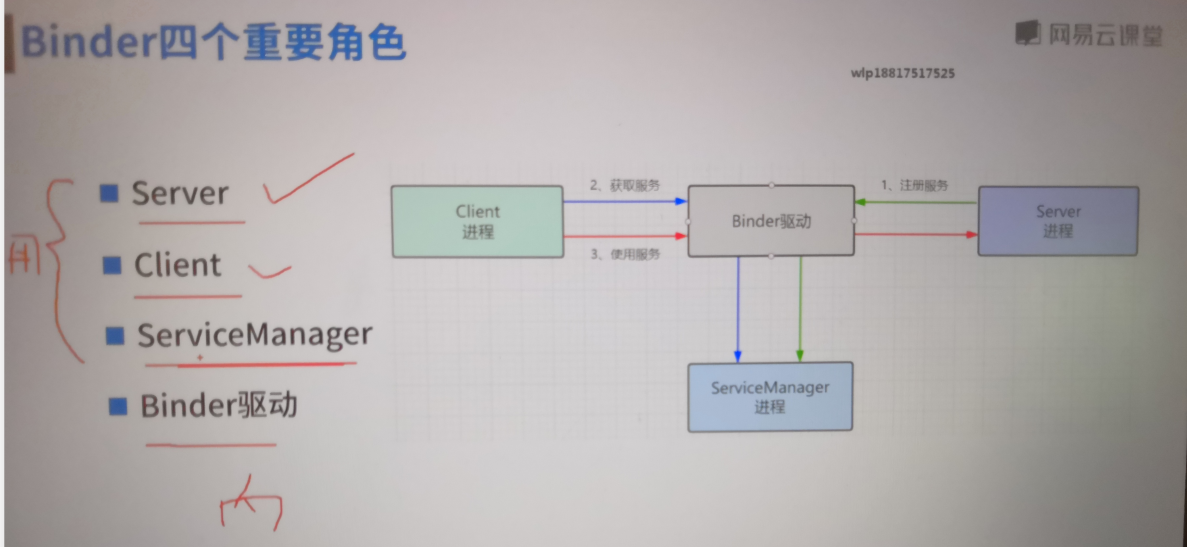
**性能很高，仅次于内存共享，并且有PID、UID的校验。**

1. 用户空间：server、client、ServiceManager

内核空间：Binder驱动

server要给别人提供服务，就需要在ServiceManager进行注册。

Binder驱动非常重要。



1. **AIDL android接口定义语言**

**C/S之间进行通信，需要定义统一的语言。**

7、

IBinder 代表了跨进程通信的能力，接口

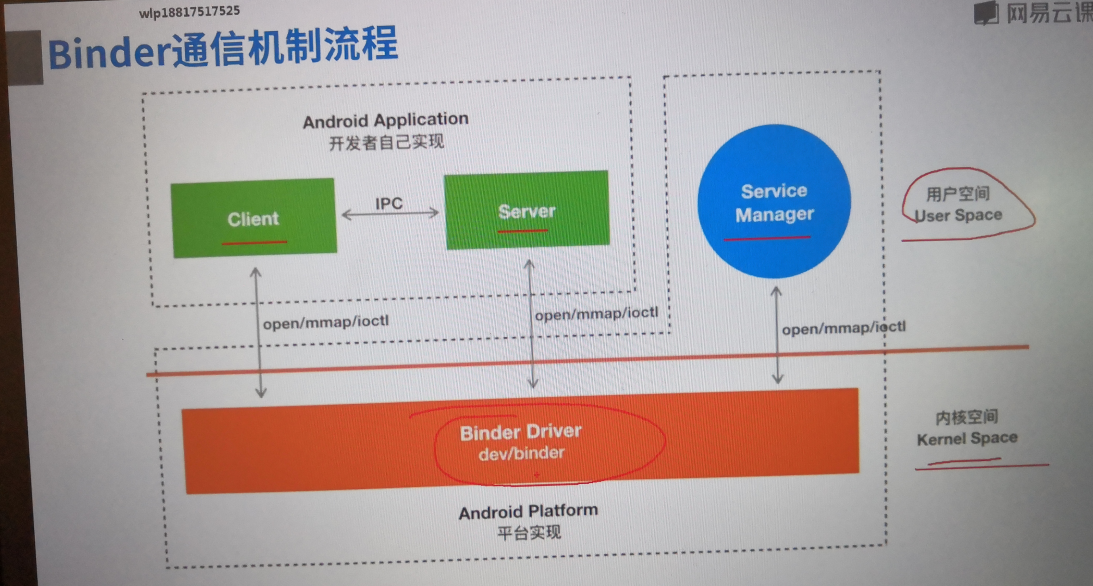
IInterface 代表server提供的服务

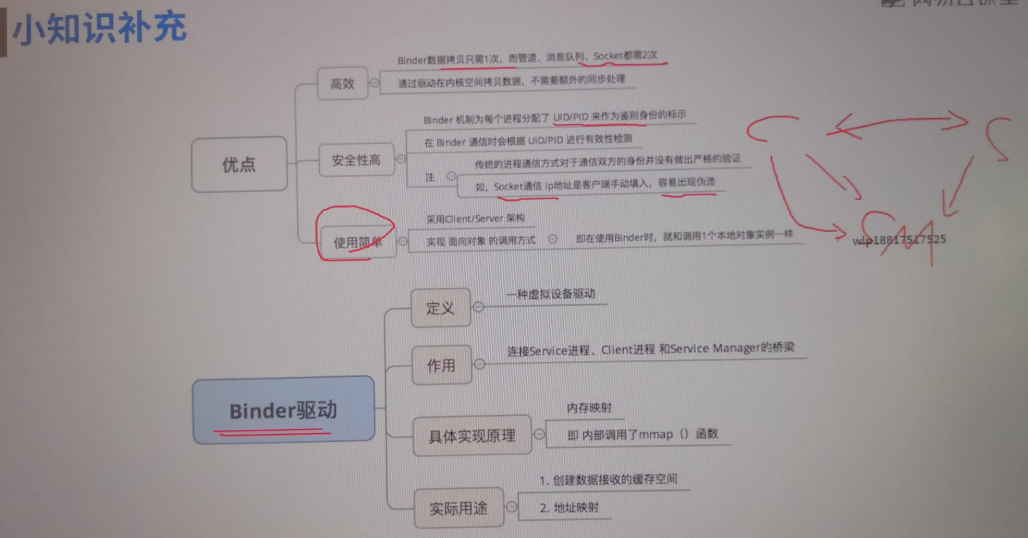
Binder java层的binder代理类 BinderProxy

stub 编译生成的静态内部类，继承与Binder,又实现了IInterface

8、Binder 通信机制流程



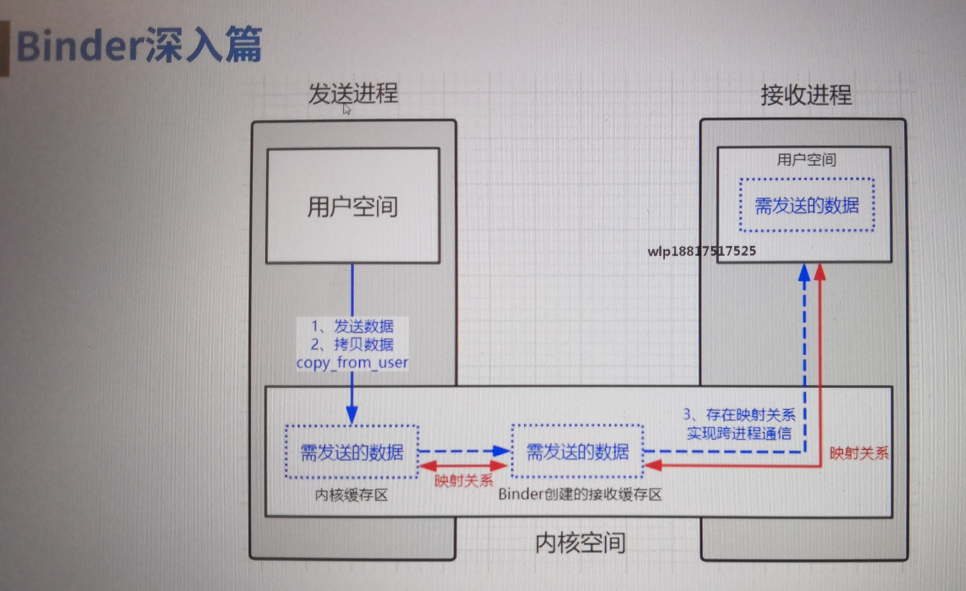




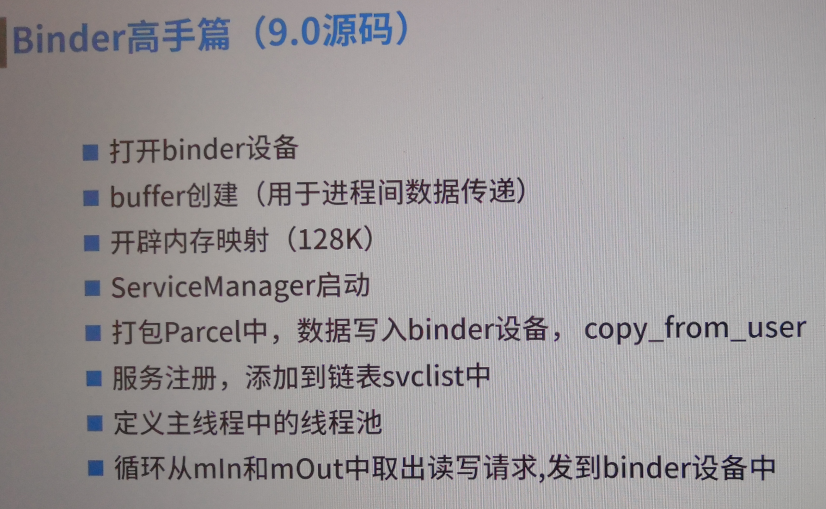
1. startActivity()就是用到了Binder，用到了AIDL。

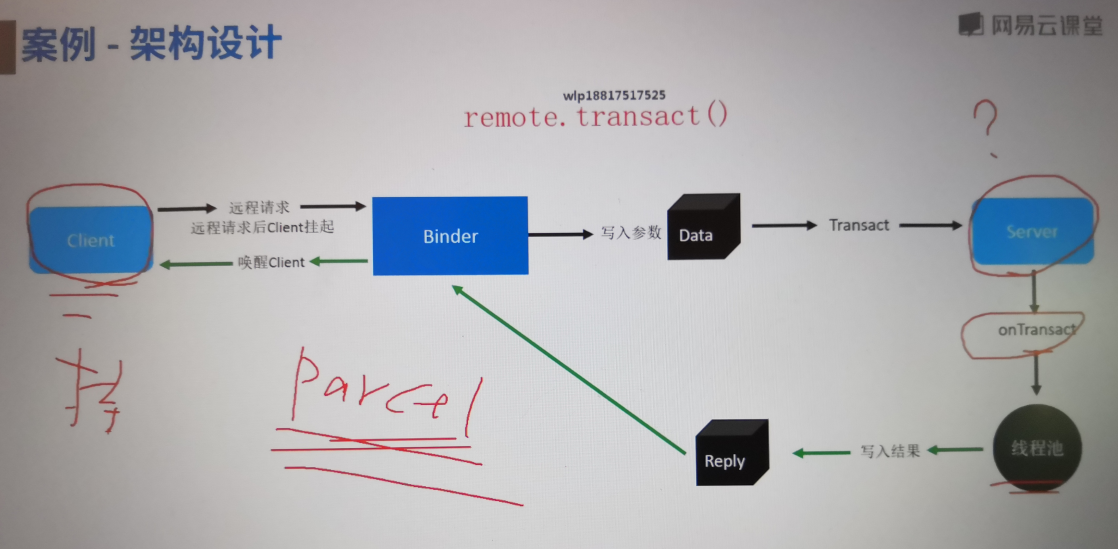
10、android内核源码

<https://www.androidos.net.cn/android/9.0.0_r8/xref>









1. 第三方登录就是一种跨进程通信
2. service中使用到Ibinder 的demo

