Linux系统内核接收以太帧的处理程序 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/145/2021_2022_Linux_E7_B3_BB_E7_BB_c103_145333.htm 1. 前言 以太头中除了6字节目的MAC地址、6字节源MAC地址外，还有两字节的以太帧类型值，如IPv4为0x0800，ARP为0x0806等，网卡驱动收到以太帧后通过接口函数netif_receive_skb()(netif_rx实际最后也是调用netif_receive_skb)交到上层，而这个接口函数就完成对以太帧类型的区分，交到不同的协议处理程序。如果想自己编写某一以太类型帧的处理程序，需要自己添加相应的代码。以下为Linux内核2.6代码。 2. 数据结构 每种协议都要定义一个packet_type结构，引导进入相关的协议数据处理函数，所有节点组成一个链表(HASH链表)。 /* include/linux/netdevice.h */struct packet_type {__be16 type. /* This is really htons(ether_type). */struct net_device *dev. /* NULL is wildcarded here */int (*func) (struct sk_buff *,struct net_device *,struct packet_type *,struct net_device *).void *af_packet_priv.struct list_head list. }.参数说明： type：以太帧类型，16位。 dev：所附着的网卡设备，如果为NULL则匹配全部网卡。 func：协议入口接收处理函数。 af_packet_priv：协议私有数据。 list：链表扣。 一般各协议的packet_type结构都是静态存在，初始化时只提供type和func两个参数就可以了，每个协议在初始化时都要将此结构加入到系统类型链表中。 3. 处理函数 3.1 添加节点 /* net/core/dev.c *//*** dev_add_pack - add packet handler* @pt: packet type declaration** Add a protocol handler to the networking stack. The passed amp.ptype_lock).// 如果类型是全部

以太类型，则节点链接到ptype_all链if (pt->type == htons(ETH_P_ALL)) {netdev_nit .list_add_rcu(amp.ptype_all).} else {// 根据协议类型取个HASH，共15个HASH链表hash = ntohs(pt->type) amp.pt->list, amp.ptype_lock). } 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com