**内核链表**

written by王保明

1. 链表基础知识

|  |
| --- |
| 1. 链表回顾及api回顾   typedef struct Node {  int data;  struct Node \*next;  }SLIST;  SLIST \*SList\_Creat();  int SList\_Print(SLIST \*pHead);  int SList\_NodeInsert(SLIST \*pHead, int x , int y);  int SList\_NodeDel(SLIST \*pHead, int y);  int SList\_Destory(SLIST \*pHead); |
| 1. 传统链表PKLinux内核链表   为什么引入这样一个结构  怎么样找到一个数据？ |
|  |

2、linux内核链表api使用及分析

|  |
| --- |
| Linux内核为链表提供了一致的访问接口  void INIT\_LIST\_HEAD(struct list\_head \*list)；  void list\_add(struct list\_head \*new, struct list\_head \*head)；  void list\_add\_tail(struct list\_head \*new, struct list\_head \*head)；  void list\_del(struct list\_head \*entry);  int list\_empty(const struct list\_head \*head)； |
| struct int\_node {  int val;  struct list\_head node;  };  int main()  {  struct list\_head head, \*plist;  struct int\_node a, b;  a.val = 2;  b.val = 3;  INIT\_LIST\_HEAD(&head);  list\_add(&a.list, &head);  list\_add(&b.list, &head);  list\_for\_each(plist, &head) {  struct int\_node \*node = list\_entry(plist, struct int\_node, list);  printf("val = %d\n", node->val);  }  return 0;  } |
|  |

3、linux内核链表分析

|  |
| --- |
|  |
|  |
|  |
| 总结：  Linux内核代码博大精深，陈莉君老师曾把它形容为“覆压三百余里，隔离天日”（摘自《阿房宫赋》），可见其内容之丰富、结构之庞杂。内核里有着众多重要的数据结构，具有相关性的数据结构之间很多都是用本文介绍的链表组织在一起，看来list\_head结构虽小，作用可真不小。  Linux内核是个伟大的工程，其源代码里还有很多精妙之处，值得C/C++程序员认真去阅读，即使我们不去做内核相关的工作，阅读精彩的代码对程序员自我修养的提高也是大有裨益的。 |