秘密

Linux 内核研究说明书

（Statement of Demand

and Design）

2020年1月

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **作者** | **版本** | **用时（小时）** | **备注** |
| 2020/1/18 | 石文斌 | 0.1 | 2 | 初建，建立文档目录 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

变更历史

目录

[1 文档说明 1](#_Toc915847728)

[2 需求分析 2](#_Toc1719202834)

[2.1 数据结构运用 2](#_Toc76661569)

[2.2 内核算法 2](#_Toc20561436)

[2.3 源码级理解 2](#_Toc565570426)

[3 数据结构 4](#_Toc3439170)

[4 功能分析 5](#_Toc379246023)

[4.1 进程管理 5](#_Toc1420881998)

[4.2 进程调度 5](#_Toc398739013)

[4.3 中断处理 5](#_Toc1561875862)

[4.4 内核同步 5](#_Toc1799095458)

[4.5 定时器与时间管理 5](#_Toc1473725767)

[4.6 物理内存管理 5](#_Toc1532141754)

[4.7 进程地址空间 5](#_Toc1152987983)

[4.8 页高速缓存 6](#_Toc1406612824)

[4.9 文件系统 7](#_Toc1589826416)

[4.10 IO处理 7](#_Toc1779139746)

[4.11 设备与模块 7](#_Toc312796382)

[参考文献 8](#_Toc436653001)

# 文档说明

本说明书用来记录linux内核源码研究的成果，有助于增强对linux内核源码的理解。

本文件为苏芯秘密，未经总经理签字允许，严禁以任何形式对公布本文件所含任何内容。

# 需求分析

## 数据结构运用

## 内核算法

## 源码级理解

# 数据结构

# 功能分析

## 进程管理

## 进程调度

## 中断处理

## 内核同步

## 定时器与时间管理

## 物理内存管理

## 进程地址空间

Linux操作系统采用虚拟内存技术，系统中所有进程以虚拟方式共享内存。每个进程都有一个唯一的内存描述符。

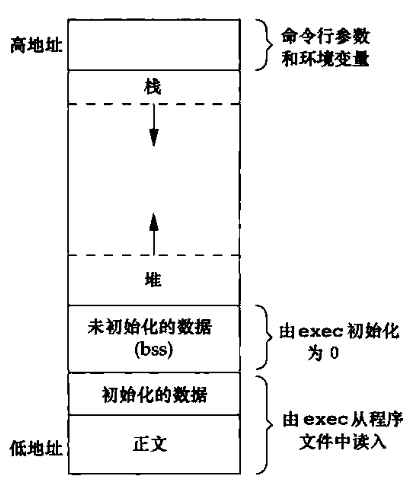


图 4-1. 进程地址空间内存布局

如果父进程与子进程共享地址空间，这样的进程称为线程。共享内存代码段不一样怎么办？

内核线程没有进程地址空间，就没有相关的内存描述符。内核线程运行时，直接使用前一个进程的内存描述符。当一个内核线程被调度时，发现它的mm域为NULL，就会保留前一个进程的地址空间，随后内核更新内核线程对应的进程描述符中的active\_mm域，使其指向前一个进程的内存描述符。因为内核线程不访问用户空间的内存，所以它们仅仅使用地址空间中和内核内存相关的信息，这些信息的含义和普通进程完全相同。

## 页高速缓存

（1）页高速缓存是linux内核实现磁盘缓存，主要用来减少对磁盘的I/O操作。具体地讲，是通过把磁盘中的数据缓存到物理内存中，把对磁盘的访问变为对物理内存的访问。

（2）临时局部原理：如果在第一次访问数据时缓存它，那就极有可能在短时间内再此被高速缓存命中。

（3）当内核开始一个读操作（比如，进程发起一个read()系统调用），它首先会检查需要的数据是否在页高速缓存中，如果在，则放弃访问磁盘，而直接从内存中读取。这个行为称作缓存命中，如果数据不在缓存中，称为缓存未命中，那么内核必须调度块I/O操作从磁盘去读取数据。然后内核将读来的数据放入页缓存中，于是任何后续相同的数据读取都可命中缓存了。

（4）页回写：程序执行写操作直接写到缓存中，后端存储不会立刻直接更新，而是将页高速缓存中被写入的页面标记成“脏”，并且被加入到脏页链表中。然后由回写进程周期性将脏页链表中的页写回到磁盘，从而将磁盘中的数据和内存中最终一致，最后清理“脏”页标识。

（5）缓存回收：通过选择干净页进行简单替换，如果缓存中没有足够的干净页面，内核将强制地进行回写操作，以腾出更多的干净可用页。缓存回收策略，维护两个链表，活跃链表和非活跃链表。处于活跃链表上的页面被认为是“热”的而且不会被换出，而在非活跃链表上的页面则是可以被换出的。在活跃链表中的页面在其被访问是就处于非活跃链表中。两个链表都被LRU规则维护：页面从尾部加入，从头部移除，如同队列。

## 文件系统

## IO处理

## 设备与模块

# 参考文献

【1】Linux内核设计与实现，Robert Love.