表 1　実装環境

OS	Debian 7.11
カーネル	Linux カーネル 3.15.0

資料タイトル

2018/4/10
吉田 修太郎

1　はじめに

本手順書では，Linux カーネルに対して新たにシステムコールを追加するための手順について述べる．以降では，実装環境，実装したシステムコールの概要，実装の手順，動作テストについて，順に章立ててそれぞれの詳細を述べる．

2　実装環境

本手順書における実装環境を下表に示す．

3　実装したシステムコールの概要

形式　asmlinkage int prt_to_rbuf(char *s)
引数　char *s: 出力する文字列のポインタ
戻り値　カーネルバッファに書き込んだ文字数

4　実装の手順

4.1　ソースコードの作成

4.2　プロトタイプ宣言

ここでは，プロトタイプ宣言の手順について述べる．システムコール関数のプロトタイプ宣言が，まとめて書かれているヘッダファイルを探し，編集する．本手順書では，以下のファイルを編集する．

- /home/git/linux-stable/include/linux/syscalls.h

本手順書では，このヘッダファイルの末尾に以下の行を追加する．

- /home/git/asmlinkage int prt_to_rbuf(char *s);

4.3 システムコール番号の定義

ここでは，システムコール番号を定義する手順について述べる．システムコール関数と，システムコール番号との対応づけが書かれているファイルを探し，編集する．本手順書では，以下のファイルを編集する．

- /home/git/linux-stable/arch/x86/syscalls/syscall_64.tbl

このファイルの内容の一部を抜粋し，以下に示す．

上記のファイル内容の先頭付近に，このファイルのフォーマットは< number >< abi >< name >< entry point >であるという旨が記述されている．それぞれの要素についての簡単な説明を以下に示す．

number　システムコール番号
abi　Applicatioin Binary Interface
name　関数名
entry point　関数が

このフォーマットに従い，実装したいシステムコールをこのファイルに追記する．ただし，システムコール番号は，システムコール呼出しの際に関数の特定に使用されるため，他の関数と重複してはならない．本資料では，以下のように設定する．

number　317
abi　common
name　sys_prt_to_rbuf
entry point　sys_prt_to_rbuf

上記の場合におけるファイルへの記入例を以下に示す．

4.4 Makefile 編集

ここでは，Makefile の編集について述べる．今回編集する Makefile を以下に示す．

- /home/git/linux-stable/kernel/Makefile

make コマンドは，このファイルの内容に基づいて実行されるため，今回追加したシステムコール関数をコンパイルするためには，ここにその処理を追記する必要がある．具体的には，各システムコール関数のオブジェクトファイルが代入される obj-y という変数に対して，新たに作成したシステムコールのオブジェクトファイルも代入されるように追記する．本手順書における，この Makefile の編集内容を以下に示す．

変更前

変更後

4.5 カーネルの再構築

5 動作テスト

5.1 概要

本章では，実装したシステムコールの動作テストについて述べる．

5.2 動作テスト用プログラムの準備

prt_to_rbuf の動作テストに用いるプログラムを作成する．本手順書における動作テスト用プログラムを以下に示す．

5.3 動作テストの手順

動作テストの手順を以下に示す．

(1) 動作テスト用プログラムの実行
(2) dmesg コマンドの実行

5.4 参考文献の挿入例

参考文献を記載する際は bibtex を利用する．mybibdate.bib に参考文献の情報を記載する．たとえば，乃村先生の論文 [?] を参考文献として記載する．

6 おわりに

本資料では