日曜研究室

技術的な観点から日常を綴ります

[xv6 #23] Chapter 2 – Traps, interrupts, and drivers – X86 protection

テキストの32〜33ページ

本文

x86は４つの保護レベルを持つ。
0番（一番権限が高い）から3番（一番権限が低い）である。
習慣では、多くのOSは２つのレベルだけ使う。
それは0と3であるが、それぞれカーネルモードとユーザーモードと呼ばれる。
x86の命令の実行に関する現在の特権レベルは、%csレジスタのCPLフィールドに保持されている。

x86では、割り込みハンドラは割り込みディスクリプタテーブル（IDT）に定義されている。
IDTは256個のエントリを持ち、そのエントリが持つ%csと%eipを対応する割り込みを制御するときに使う。

x86上でシステムコールを実行するとき、プログラムはint n命令を呼び出す。
nはIDTへのインデックスを指定する。
int命令は、次の手順を行う。

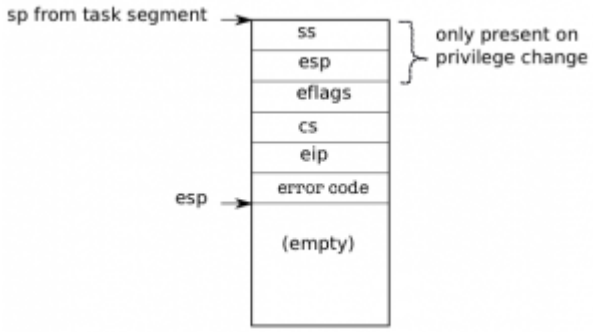
・intの引数であるnを使い、IDTからn番目のディスクリプタを検索する。
・その%csのCPLがDPL以下である事を調べる。DPLはそのディスクリプタにおける特権レベルである。
・CPU内部のレジスタ%espと%ssを保存する。ただし、対象のセグメントのセレクタのPLがCPLより小さいときだけである。
・タスクセグメントディスクリプタから%ssと%espを読み込む。
・%ssをプッシュする。
・%espをプッシュする。
・%eflagsをプッシュする。
・%csをプッシュする。

・%eipをプッシュする。
・%eflagsの一部をクリアする。
・ディスクリプタの中の値を%csと%eipにセットする。

このint命令は複合的な命令であり、それら全てのアクションが必要なのかどうか不思議かもしれない。
CPLがDPL以下である事を調べる事は、カーネルが他の特権レベルのためのシステムを禁止することを可能にする。
例えば、ユーザプログラムがint命令を上手く実行するためには、DPLは3でなければならない。
もしユーザプログラムが適切な特権を持たなければ、int命令はint 13を返す。
int 13とは一般保護違反である。
他の例では、int命令は値を保存するためにユーザスタックを使うことは出来ない。
なぜなら、ハードウェアがタスクセグメント上でスタックを使うための適切なスタックをユーザはセットアップしていないであろうからである。
それはカーネルモードでセットアップされる。

図2-1は、int命令が完了し、特権レベルの変更があった後のスタックを表している。
（ディスクリプタの特権レベルはCPLより小さい）
もし、int命令が特権レベルの変更を必要としなかったら、x86は%ssと%espを保存しないだろう。
両方の場合で、%eipはディスクリプタテーブル上で与えられたアドレスを指している。
そのアドレスの命令は、次に実行される命令とint nの為のハンドラの最初の命令である。
これが、それらのハンドラをOSが実行するための仕事であり、次からxv6がそれをどう実現するかを見ていく。

図2-1 int命令語のカーネルスタック



OSはiret命令を、int命令から戻るために使う事ができる。
その命令は、スタックからint命令の間に保存された値を取り出し、保存された%eipから実行を再開する。

感想

正直IDT周りがよく分かりません。

ただ、次の節からコードの説明が連続するので、そこで詳細が分かるんではないかと思います。

カテゴリー: 技術 | タグ: xv6 | 投稿日: 2012/2/28 火曜日 [http://peta.okechan.net/blog/archives/1364] |
