non-definitive-guide-ASSO

ASSO

Wykład 1

Zignorowałęm slajdy 15 - 17, nie czuje ich treści.

Sieciowy system operacyjny

system operacyjny ze zintegrowanym w jądrze stosem sieciowym. Zawiera sterowniki obsługujące interfejsy sieciowe (karty), wbudowane oprogramowanie obsługi co najmniej warstwy łącza danych, sieciowej i transportowej, funkcje systemowe/biblioteczne wspierające: gniazda sieciowe Berkeley, interfejs TLI i/lub gniazda Windows. Przykłady: Unix/Linux, Windows NT, Windows 7,8,2012 server

---

Sterowniki urządzeń I/O

Sterownik - standardowy interfejs, część jądra SO odpowiedzialna za obsługę urządzenia I/O.

Klasyfikacja sterowników:

1. Urządzenia znakowe - char devices
2. Urządzenia blokowe - block devices (mają bufor)
3. Interfejsy sieciowe - network interfaces

Metody dołączania sterowników do jądra:

- statycznie - w trakcie kompilacji jądra
- dynamicznie - moduły

Urządzenia znakowe (char devices):

1. Można używać jako pliki specjalne
2. nazwy zdefiniowane ścieżką
    - np. `/dev/tty0`, `/dev/null`
3. Funkcja systemowa / komenda `mknod`:

- `mknod /dev/urz major minor`
    - liczba glowna (major) identyfikuje sterownik
    - liczba poboczna (minor) przekazywana do sterownika

Podstawowe funkcje interfejsu znakowego:

- Struktura `file_operations`:
    - open,
    - release,
    - read,
    - write,
    - ioctl (input output control)

Urządzenia blokowe (block devices):

1. operacje `read`/`write` sterownika wywoływane są przez system buforowania
2. operacje `open`/`release` wywoływane są przez funkcje `mount` i `umount`
3. ścieżki, nr głowny i poboczny jak w urz. znakowych (odrębna numeracja główna sterowników)

READ MORE

Podstawowe funkcje interfejsu blokowego:

- Struktura `file_operations`:
    - `open`
    - `release`
    - `block_read` / `block_write`
    - `check_media_change`
    - `revalidate`
- `request_fn`
    - `READ / WRITE <sektor bufora> <liczba>`

Moduły

są to fragmenty jądra dołączane opcjonalnie, moga byc dynamicznie ładowane i usuwane z jądra, wykorzystuja funkcje wewnetrzne jadra a nie funkcje systemowe przestrzeni uzytkowej. Pozwalaja tez ograniczyc wielkosc jadra bez rekompilacji. Ładowanie modułu przez `insmod`: alokacja pamięci w przestrzeni jądra, łączenie symboli globalnych/zewn z jadrem, rejestracja przez wykonanie init_module (`register_chrdev`, `register_blkdev`), śledzenie aktywnosci modułu (usage count) `/proc/modules`, przed usunieciem modulu `cleanup_module`. Moduł usuwa się przez `rmmod`.

Zarządzanie modułami:

1. Ręczne ładowanie i usuwanie.
2. Konfiguracja modułu:
    - `inmod` zm=”my_dev” nr=81
    - konfigurowalne dowolne zmienne globalne/statyczne modułu

---

3. wykorzystanie kerneld
    - jadro musi byc skompilowane z wsparciem kerneld
    - daemon `kerneld` musi byc stale uruchomiony
    - program modrobe
    - konfiguracja modulow w /etc/modules.conf (options)

Interfejsy sieciowe:

- Nie ma do nich dostepu przez sciezke - zamiast tego sterownik rejestruje swoja nazwe np. `eth0`
- Sterowniki ukrywają zależność od konkretnego protokolu
- Operuja na pojedynczych pakietach - w odroznienia od urzadzen blokowych operacje nie zawsze sa inicjowane przez wywołanie operacji na sterowniku
- Rejestrowany przez `register_netdev` - `struct device`, zawiera wskaznik na funkcje init sterownika, struktura zawiera nazwę dla urzadzenia (lub null)

<Wykład 2>