Operačné systémy / Úvod

Dušan Bernát
bernat@fmph.uniba.sk

Prehľad

- Čo je OS, na čo slúži.
- Základné súčasti a funkcie.
- Štruktúra OS, Monolitické jadro, mikrokernel, moduly.
- Správa prostriedkov.
- Abstrakcia prostriedkov.
- Programátorské rozhranie (systémové volanie).
- Zavádzanie OS.
- Stručná história, prehľad.
- Používateľské rozhranie.

Ako by vyzeral počítačový systém bez OS?

Ako by vyzeral počítačový systém bez OS?

- Každý program by vždy musel obsahovať všetko.
  - Ak by mal zapísť do súboru musel by obsahovať
    - konkrétny súborový systém, teda určiť, ktoré bloky patria súboru,
    - komunikáciu s konkrétnym typom disku cez príslušné rozhranie,
    - ošetrenie súbežného prístupu iných programov.
  - Programy by boli veľké, neprenosné, ťažko udržiavateľné.
- Koľko programov by mohlo bežať naraz?
- Koľko používateľov by mohlo pracovať súčasne?
- Ako by sa delili o dostupné prostriedky?

Čo je OS?

- Základné programové vybavenie počítača. Softvérová vrstva medzi hardvérom a aplikáciami;
- Komunikuje s hardvérom (terminál, disky, sieť, zvuk, ...).
- Poskytuje jednotné rozhranie pre ostatný softvér (aplikácie).
- Riadi prístup k prostriedkom (súbežnosť, bezpečnosť).
- Efektívne rozdeľuje dostupné HW prostriedky medzi používateľov, programy a zariadenia.
  - Procesorový čas, pamäť, diskový priestor, ...
  - Cieľom je zvyčajne čo najväčšie využitie prostriedkov a priepustnosť systému a čo najmenšia doba odozvy.
  - Optimálne plánovanie pridelenia prostriedkov je zložitý problém. Používajú sa aproximácie a heuristiky.

Základné funkcie a charakteristiky

- Umožňuje používateľovi pracovať s počítačovým systémom.
  - Poskytuje mu základné rozhranie na ovládanie počítača.
- Umožňuje vykonávanie ďalších programov (aplikácií).
  - Bez OS sa spravidla môže vykonávať len jeden program.
- Poskytuje im základné prostredie v podobe štandardnej sady služieb, bez ohľadu na konkrétny HW.
  - Odstraňuje závislosť na HW.
  - Zjednodušuje prácu programátorovi (šetrí náklady).
- Zabezpečuje izoláciu procesov a používateľov.
- Umožňuje, zabezpečuje, poskytuje ... (služby procesom).
  - Vytvára infraštruktúru, sám osebe nie je užitočný.

Abstrakcia

- OS využíva abstrakcie na zakrytie nevýhod a obmedzení HW.
- Abstrakcie poskytujú rozhranie ktoré oddeľuje operácie (čo je možné so zariadením robiť) od mechanizmov (od ich konkrétnej implementácie).
- Abstrakcia periférií
  - OS vytvára vrstvu skrývajúcu detaily a rôznorodosť hardvéru.
  - Poskytuje prístup k nemu jednotným spôsobom.
    - Napríklad v Unixe sa k väčšine periférnych zariadení pristupuje rovnako ako k **súborom**, cez operácie read a write.
- Abstrakcia CPU
  - Pre každý **proces** vytvára dojem, že má procesor len pre seba.
- Abstrakcia pamäte
  - Virtuálna pamäť, logický **adresový priestor**.

Definícia OS

- Operačný systém plní viacero roličných úloh, najmä ako:
  - Rozšírený stroj
    - Proces môže okrem jednoduchých inštrukcií procesora využívať aj funkcie OS, hlavne pre rôzne V/V operácie.
  - Správca prostriedkov
    - Prideľuje procesom procesorový čas (plánovač), pamäť (správa pamäte), súborový systém, ...
- Uviesť jednotnú definíciu nie je jednoduché.
- *"An operating system is a program that controls the execution of application programs and acts as an interface between the user of a computer and the computer hardware."* [Stallings, 2005]

Systémové volania

- Operačný systém poskytuje programátorovi navyše okrem inštrukcií procesora aj sadu funkcií (rozšírených inštrukcií).
  - Proces môže vykonávať vlastný kód (inštrukcie),
  - alebo zavolá rutinu operačného systému.
- Systémové volania zahŕňajú aj tieto oblasti:
  - Procesy (fork, exec, exit, wait, getpid, getuid, sleep, ...)
  - Medziprocesová komunikácia (signály, dátovody, sokety, ...)
  - Pamäť (mmap, munmap, mprotect, mlock, brk, ...)
  - Súborový systém (open, read, write, seek, close, creat, unlink, chown, ...)
    - Súbory môžu reprezentovať periférie (konzola, tlačiareň, sieť, ...)
  - Sledovanie zdrojov, nastavovanie limitov, bezpečnosť, ...
- Systémové volania sprostredkovávajú procesu kontakt s okolím.

Správa prostriedkov

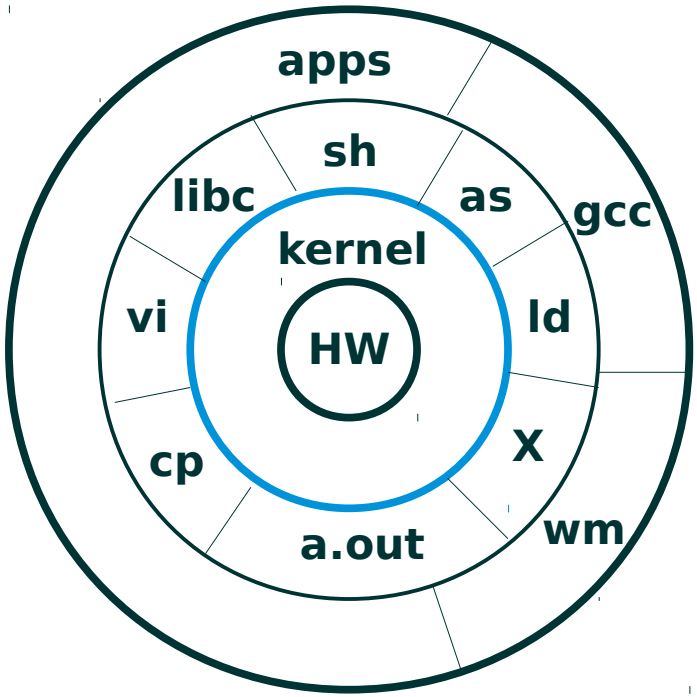
- Prostriedky sú limitované, procesy o ne súperia.
- Správa procesov
  - Plánovač, rozdeľuje procesom výpočtový čas dostupných CPU.
- Správa pamäte
  - Prideľuje procesom dostupnú pamäť.
  - Zabezpečuje virtuálnu pamäť.
  - Aktuálne nevyužívané časti odkladá na disk (*swap*).
- Súborový systém
  - Organizuje súbory v hierarchickej adresárovej štruktúre. Organizuje dáta súborov do diskových blokov.
  - Riadi prístup k súborom (vlastník, skupina, ostatní, ACL). Plánovanie diskových operácií. Správa "*cache*" v pamäti.
- Vstupno/výstupný podsystém
  - Riadi prístup k periférnym zariadeniam. Každé má svoj ovládač.
- Sieťový podystém

Jadro, prerušenia a hardvér

- Jadro (po zavedení) väčšinou samo od seba nerobí nič. Reaguje na prerušenia.
- Systémové volania sú iniciované procesmi.
  - Využívajú mechanizmus softvérového prerušenia.
- Plánovač reaguje na prerušenie od časovača.
  - Využíva ho na zabezpečenie preempcie pri prideľovaní CPU (*time sharing*).
- Správa pamäte reaguje na výnimku pri prístupe do pamäte.
  - Využíva ju na obsluhu výpadku stránky (*page fault*).
- Prerušenia od periférnych zariadení obsluhujú príslušné ovládače.

Vnútorná štruktúra OS

Štruktúra OS



- Programátorské rozhranie (API)
  - Systémé volania jadra.
  - Tie priamo využívajú len systémové programy.
  - Väčšina aplikácií však len nepriamo, cez knižnice (wrapper, napr. libc).
- Používateľské rozhranie (UI)
  - Textové (napr. shell, CLI)
  - GUI (napr. X + window manager)
- Čo z toho je operačný systém?
  - V užšom zmysle len jadro (kernel),
  - v širšom zmysle aj shell (UI), libc, ...

Niektoré architektúry OS (1)

- **Monolitické systémy** (monolitické jadro)
  - Jadro obsahuje všetko, aj to čo netreba, napr. ovládače neprítomných zariadení.
  - Bez štruktúry, každá funkcia môže volať každú inú, aj keď spolu nesúvisia (a nikdy by sa volať nemali).
  - Pôvodné systémy Unix, Linux, VMS, MS DOS, ...
- **Vrstvové systémy**
  - Každá vyššia vrstva (podsystém OS) poskytuje abstraktnejší pohľad na stroj.
  - Napr. THE (Dijkstra 1968); proces operátora > používateľské programy > V/V > komunikácia procesu s konzolou > správa pamäte.
  - Multics (1965); prechod medzi vrstvami striktne kontrolovaný, nie je možné ich obísť, každá vrstva iná privilegovaná úroveň procesora, vyžaduje mechanizmus systémového volania.
  - Dnešné OS; Jadro v podstate monolitické, vnútorne členené do vrstiev, prechod medzi vrstvami však nie je vynucovaný hardvérovo (procesorom).
  - Oproti monolitickým majú viac vnútornej štruktúry, lepšie sa vyvíjajú a udržiavajú.

Niektoré architektúry OS (2)

- V jadre by mal byť len taký kód, ktorý nevyhnutne potrebuje privilegovaný režim. Ostatné je možné nechať na procesy.
- Model **klient – server**, využíva **mikrokernel**
  - Jadro obsahuje len minimálne množstvo funkcií pre implementáciu funkčného OS; procesy a komunikáciu.
  - Všetky ostatné funkcie sú presunuté na procesy – servery. Sú súčasťou OS.
    - Ovládače zariadení, súborový systém, TCP/IP, obsluha výpadku stránky (*pager*), ...
  - Problém v jednej časti neovplyvní zvyšok systému.
  - Proces aplikácie – klient. Komunikuje prostredníctvom jadra so servermi.
  - QNX (1981), MACH (CMU 1984), Windows NT (MS 1990), ...
- Jadrá štruktúrovaných (vrstvových) monolitických systémov sa dnes zmenšujú mechanizmom zavádzateľných modulov.
- Mikrojadrá sa zväčšujú v dôsledku snahy o znižovanie réžie.

Niektoré architektúry OS (3)

- **Objektové OS**
  - Všetky prostriedky (súbory, periférie, ...) sú reprezentované objektami.
  - Jadro poskytuje mechanizmus na riadený prístup k objektom.
  - Tzv. capability = referencia na objekt + prístupové práva
  - Pokus MS Windows NT/2000
- **Virtuálne stroje**
  - Jadro (monitor, hypervisor) poskytuje vo vyššej vrstve nie len "rozšírený stroj" pre procesy, ale viacero kópií hardvéru. Na každej môže bežať samostatný OS.
  - IBM VM/CMS, v podstate aj JVM.
- Bolo by možné opísať aj ďalšie štruktúry OS.

Delenie OS (podľa vlastností)

- Jednoprocesový – v danom čase len jeden program (napr. MS DOS).
- Multiprocesový – viacero vykonávaných programov súčasne, lepšie využitie zdrojov.
- Dávkové systémy (*batch*)
  - Používateľ zadá úlohu a čaká na výsledok (bez interakcie).
  - Hlavne v minulosti, dnes sa takýto prístup používa v HPC.
- Interaktívne systémy
  - Používateľ môže komunikovať s úlohou počas vykonávania.
  - Všetky dnešné systémy.
- Systémy so zdieľaním času (*time sharing*)
  - V multiprocesových systémoch je využívané na rozdelenie času CPU medzi viaceré súbežné procesy.
- Systémy reálneho času (RT – *real time*)
  - Procesy majú striktné požiadavky na dodržanie časovania. Doba odozvy systému musí byť ohraničená.
  - Využíva sa v riadiacich a vnorených systémoch, napr. QNX, aj Linux má podporu pre soft RT.

Oddelenie jadra a procesov

Procesy a jadro (kernel)

- Ako sa jadro líši od procesov?
  - Proces môže vykonávať všetky bežné (neprivilegované) inštrukcie (napr. prístup do pamäte a nejaký výpočet).
  - Nemôže vykonávať privilegované inštrukcie (napr. vstupno/výstupné).
  - Musí o to požiadať jadro OS prostredníctvom systémového volania.
  - Jadro môže vykonávať čokoľvek (komunikácia, V/V, ...).
- Podpora od architektúry – i286/1982, i386/1985 (platné dodnes)
  - Reálny režim – každý (bežiaci) program má plnú kontrolu nad procesorom, pamäťou, perifériami, ...
  - Chránený režim (protected mode)
    - virtuálna pamäť, stránkovanie, multi-tasking, ...
    - viaceré úrovne privilégií (najmä pre vstup/výstup).

Privilegované úrovne (Intel)

- CPL (Current Privilege Level) – úroveň práve vykonávaného kódu.
- CPL, sú 4, označujú sa ring 0 až 3 (Intel).
  - Ring 0 – kernel mode (ovládače HW, služby – syscall), privilegované inštrukcie sa môžu vykonať len tu.
  - Ring 3 – user mode (procesy, žiadne priame V/V, obmedzený prístup do pamäte).
  - Ring 1, 2 – väčšinou nepoužívané
    - menej privilegovaný kód jadra (ovládače “3rd party”),
    - virtualizácia (0 – host OS kernel, 1 – guest OS kernel).
- CPL sa mení len pri zmene toku riadenia.
  - HW prerušenia od periférií, výnimky,
  - SW prerušenie od procesov (inštrukcia INT n, TRAP, ...).

Mechanizmus systémového volania

- Odovzdanie riadenia jadru
  - “skok v programe” so zmenou privilegovanej úrovne (CPL),
  - Inštrukcia softvérového prerušenia (napr. INT 80h),
  - dispečer systémových volaní obsluhuje jedno zo SW prerušení.
  - SW prerušenie má veľkú réžiu a systémové volania sú časté.
  - Špecializované inštrukcie SYSENTER/SYSEXIT (Intel), SYSCALL/SYSRET (AMD).
- Špecifikovanie služby (otvorenie/čítanie/zápis súboru, …) - číslo v registri
- Odovzdanie argumentov (meno súboru, deskriptor, …) - registre / pamäť
- Návratová hodnota (výsledok, chybový kód) - registre / pamäť
- Konkrétna implementácia závisí od architektúry a operačného systému.

Systémové volanie – MS DOS

- MS DOS pracuje v reálnom režime.
- Číslo systémového volania (služby) musí byť v AH.
- Argumenty aj návratové hodnoty sa odovzdávajú cez registre.
- OS obsluhuje prerušenie 21h.

- Napríklad (program):

```
MOV AH, 2      ; character output
MOV DL, 'Z'    ; character
INT 21h        ; system call
```

Zavádzanie OS (PC)

- Procesor po resete skočí na pevne danú adresu (FFFF:0000) v ROM.
- Adresa obsahuje skok na POST resp. BIOS (v EPROM).
- V BIOSe je nastavené, z ktorého disku sa má boot-ovať. Prečíta prvý sektor (512B) tohoto disku.
- Prvý sektor obsahuje MBR (Master Boot Record) v ktorom je prvá časť kódu bootloader-a (a “partition table” s posunutím 446B).
- Bootloader-u väčšinou nestačí 446B a tak z disku prečíta a spustí ďalšiu svoju časť (GRUB stage 1.5/2).
- Následne bootloader prečíta a spustí jadro (initrd, device tree, etc.).
- Jadro nastaví IDT, GDT, ...
  - Nastaví obsluhu výnimky prístupu do pamäte (obsluha “page fault”),
  - nastaví obsluhu systémových volaní (dispečer, napr. na 80h), ...
- Inicializuje ovládače a vytvorí prvý proces.

Stručne z histórie

Stručný prehľad histórie počítačov (1)

- Prvá generácia počítačov (1949 – 1955)
  - Elektromechanické a elektrónkové stroje.
  - Programovanie priamo v strojovom kóde (nebol ani assembler).
- Druhá generácia počítačov (1955 – 1965)
  - Elektronické stroje na báze diskrétnych polovodičov (tranzistorov). V/V magnetické pásky (a dierne štítky).
  - Programovacie jazyky (JSI/asembler, Fortran).
  - Batch systémy, znižovali časové straty pri zavádzaní úloh.
  - Prvé OS IBSYS (IBM).

Stručný prehľad histórie počítačov (2)

- Tretia generácia počítačov (1965 – 1980)
  - IBM OS/360, multiprograming, time sharing.
  - Prvý Unix pre PDP-7.
- Štvrtá generácia počítačov (1980 – 1990)
  - Vysoká integrácia polovodičov na čipe umožnila vytvorenie procesora (ako jedného IO).
  - MS DOS, IBM PC
  - V 80-tych rokoch rozvoj sietí a komunikácií.

Stručná história Unixu

- 1969 (1. verzia) pre PDP-7, Ken Thompson z Bell Labs, Dennis Ritchie.
- 1973, väčšina bola prepísaná do C. OS bol prenesený na ďalšie stroje, PDP-11.
- 1978 (7. verzia) pre PDP-11/70, neskôr prenesená na VAX. Predchodca aj dnešných systémov.
- 1983 System V Unix
- 3BSD, 4BSD Unixové systémy z Berkeley. Vznikol C-shell, editor vi, kompilátory, ...
- FreeBSD, NetBSD, OpenBSD, ...
- 1987 Minix, A. S. Tanenbaum, pre akademické účely
- 1991 Linux, Linus Torvalds, Open Source

Používateľské rozhranie

- Textové rozhranie k základným funkciám OS poskytuje interpreter príkazového riadku (CLI) – *shell*.
- Shell sám osebe umožňuje najmä vytváranie nových procesov spúšťaním programov (z binárnych súborov z disku),
  - tiež upravovať ich vstup a výstup presmerovaním z/do súborov,
  - nastavovať ich vzájomnú komunikáciu cez dátovody, ...
- Poskytuje tiež funkcie programovacieho jazyka
  - substitúcia premenných, polia,
  - podmienky, cykly, ...
- OS môže mať aj grafické používateľské rozhranie (GUI).
  - To môže byť nadstavbou (aplikáciou), napr. X Window + window manager,
  - alebo súčasťou OS, napr. Windows.

Shell

- Interpretuje príkazy riadok po riadku.
- Číta štandardný vstup (typicky terminál) alebo súbor (skript).
- Umožňuje spúšťať a spájať programy prostriedkami OS.
  - Vytvorenie procesov, dátovodov, presmerovanie, ...
- Obsahuje aj základné jazykové konštrukcie.
- Shell najskôr každý riadok skontroluje a urobí náhradu špeciálnych znakov.
  - Napr. medzera i iné biele znaky majú špeciálny význam, oddeľujú slová.
  - Prvé slovo na riadku je názov príkazu, ostatné sú jeho argumenty.
- Pri písaní programov v tomto interpretovanom jazyku (skriptov) je možné využívať externé príkazy.
  - Tieto dáta buď vytvárajú, alebo filtrujú.
  - Typická štruktúra príkazu sa označuje aj ako “filter and pipe”.
  - Napr. ls – poskytne zoznam súborov, ps – zoznam procesov, ...
  - grep – na výstup vypíše len tie riadky zo vstupu, ktoré vyhovujú danému vzoru.

To je zatiaľ všetko