Linux/Unix

Historie

50. léta
-cena HW převyšovala cena člověkohodin
-počítače bez OS = programy přímo na míru
-na konci náznaky OS (ochrana paměti, IO operace, kontrola nekonečné smyčky)

60. - 70. léta
-vyšší rychlost počítačů (čekání na IO operace)
-počítačová centra = hlavní rychlý počítač pro práci s mag. páskami, vedlejší na přepis pásků
-vznik spoolovacích systémů = běh více programů zároveň a přepínání mezi nimi (každý výrobce svůj spool systém)

1969 - **UNICS** (Assembler, maličké jádro řídící programy, hierarchický FS, vše jako soubor, pipy)
1970 - název UNIX
1975 - první komerční UNIX
1983 - projekt GNU (vyvinout OS se svobodnou licencí založený na původním UNIXu)
1991 - první LINUXové jádro, LINUX
1992 - X Window System
1997 - GNOME

Koncepce

Systémy UNIX

Jádro
=v privilegovaném režimu často jediný soubor, který využívá souvislý adresový prostor
- -řízení provádění úloh
- -správa souborového systému
- -správa paměti
- -plánování procesů
- -plánování procesů pro sdílení času CPU

-od počátku víceprocesorové víceuživatelské univerzální síťové systémy

Vrstva HAL (hardware abstraction layer)
-základní rozhraní k zařízením
-skrytí technických detailů zařízení
-načítání ovladačů
-připojování/odpojování blokových (paměťových) zařízení
-provoz abstraktního modelu HW
-dnes jej již nahrazují jiné modely jádra

Ovladače
-bloková a znaková zařízení
-síťová a virtuální zařízení
-další specializované
-souborové systémy také jako ovladače

Souborový systém
-rozhraní (často mezi ovladačem vnějšího paměťového média a vyššími vrstvami jádra)
-"všechno je soubor" (souborové systémy pro vnější paměťová média, pro přístup k informacím)

-virtuální souborové systémy = nenáleží k žádnému paměťovému médiu
-VFS (virtual file system) = abstraktní vrstva nad více FS, zajišťuje podobný přístup k různým souborovým systémům, všechny souborové systémy v jedné stromové struktuře, stejný způsob zacházení s daty kdekoliv v souborovém systému

<u>Systémy LINUX</u>
Architektura vrstvená
-**jádro** = vlastní základ OS
-**příkazový interpret (shell)** = textový příkazový režim, spouštění programu, programovací jazyk, např. GNU Bash
-**(základní) programy** = operace s daty (soubory) a úlohami (procesy), "tiché chování"

-uživatelská rozhraní jen jako další programy, OS je na něm nezávislý a tím univerzální

Charakteristika
-svobodný SW (licence GPL)
-koncepce UNIXu, ale Linux !=UNIX
-přenositelnost a škálovatelnost => nejvíce platforem
-široká HW i SW podpora
-ne nutně zpětně kompatibilní
-maskot Tux (tučňák)

Distribuce GNU/Linux
-"balení" systému pro snadnou instalaci a správu
-další administrátorské programy a specializovaný SW
-programy (i jádro) ve formě balíčků, závislosti mezi balíčky
-live varianty (Slax, Tails), pro provoz netřeba instalace
-př. Ubuntu, Debian, Fedora, SuSE, RHEL

Shell

=program, který vytváří v počítači rozhraní pro práci uživatelů se službami OS
-slupka přes kterou se dostáváme k jádru

Dělení:
-řádkové (textové) = vytvářejí příkazový řádek, zejména pro administraci
    -interaktivní režim = příkaz se provede po stisku klávesy enter
    -dávkový režim =příkazy předem zapsané v textovém souboru (shell skripty)

-grafické (vizuální) = vytvářejí grafické uživatelské rozhraní (GUI), lepší pro běžné uživatele
    -akce v GUI převedeny na pozadí do příkazů a provedena
    -Prostředí X Window System
        -X Windows manageři (Blackbox, Fluxbox)
            -plné prostředí pracovní plochy = Cinnamon, KDE, GNOME, MATE, Xfce

Souborový systém

-hierarchický systém souborů
-projekt Filesystem Hierarchy Standard

Základní strom adresářů
/ = kořen souborového systému
    /bin = základní **spustitelné soubory** pro použití **všemi uživateli**
    /boot = **jádro systému**, initrd, **soubory zavaděče** GRUB
    /dev = **jednotlivá fyzická zařízení** nebo pseudo zařízení systému
    /etc = **globální konfigurační soubory** systému
    /home = **domovské** adresáře uživatelů
    /lib = základní **sdílené knihovny**, mapování klávesnice, moduly pro kernel
    /lost+found= ztracené+opravené soubory po chybách
    /mnt = podadresáře ke kterým se **připojují další zařízení**
    /opt = **SW**, který není součástí distribuce
    /proc = **soubory nastavení** a stavu systému+procesů
    /root = domovský adresář superuživate (**root**)
    /sbin = systémové privilegované soubory (sudo)
    /sys = virtuální adresář
    /tmp = **odkládací** a pomocné soubory
    /usr = další důležité podadresáře (**knihovny, zdrojáky, konfigy**,...)
    /var = soubory, jejichž obsah se během chodu systému **mění**

Procesy

-**program** = spustitelný soubor
-**proces** = spuštěný soubor
-**multitasking** = cyklické přidělování CPU plánovačem procesů
-**PID** = číselné ID procesu
    -stromová hierarchie procesů
    -PID 1 = kořenový proces (vytvořen jádrem při startu systému)
    -systémové programy jako potomci
    -přihlašovací dialog spouští desktop a ten vytváří uživ. procesy jako potomky
    -ukončení = **dokončení běhu** nebo **násilné** (kill PID)
    -plánované spouštění =příkaz **at**, nástroj **cron**

Systémové programování

Shell

Řídící struktura (control flow statement)
=konstrukce pro zápis počítačového programu rozhodující o dalším provádění (větvení, cyklení a jiné změny běhu programu)

1. Posloupnost příkazů (sekvence)
-sled klasických příkazů, které se vykonávají postupně jeden po druhém (lineárně)

2. Větvení
-podmínka if-then (else)
    -podmíněný příkaz, podmíněná konstrukce
    -prostředek programovacího jazyka umožňující rozdílné chování programu v závislosti na vyhodnocení specifikované logické podmínky
-switch a case
    -porovnává předanou hodnotu s předem specifikovanými konstantami
    -v případě nenalezení schody možnost else, default, *)

3. Cyklus
-skládá se z posloupnosti příkazu a podmíněného skoku
    -nekonečný cyklus = za normálních okolností není nikdy ukončen
    -while-do = podmínka na začátku posloupnosti příkazů
    -do-while = podmínka na konci
    -cyklus s testem podmínky uprostřed posloupnosti
    -for = podmínka na začátku, používané pro výčet prvků, “zvláštní případ while-do”

Bash

-jeden z Unixových shellů, který interpretuje příkazový řádek
-vychází z dříve nejpoužívanějšího shellu Bourne shell (sh)
-POSIX shell (portable operating system interface)
-koncipován pro operační systémy založené na projektu GNU
-implicitní příkazový interpret v OS postavených na linuxovém jádře

Skripty

-chceme-li posloupnost daných příkazů používat opakovaně, nebo z různých míst, můžeme tuto posloupnost uložit do souboru, který necháme zpracovat interpretem
-první řádek udává cestu k interpretu (!# /bin/bash)
-bez přípony, případě. sh
-musí mít právo execute
-spuštění:
    -voláním shellu = sh jmenoSkriptu
    -aktuálním shellem = ./jmenoSkriptu
-pokud chceme aby byl skript spustitelný **všemi** uživateli, přidáme ho do adresáře z $PATH

Terminálový a grafický přístup

Terminál
-existence od počátku počítačů
-pořizování vstupů/výstupů do/z programů

-znakový (klavesnice+obrazovka)
    -řádkový = text vykreslen po řádcích bez možnosti umístění kurzoru, editor **ed**
    -celoobrazovkový = speciální kódy pro umístění kurzoru + pokročilejší editory
    -systémová konzole = základní ovládání počítače

-grafický (využití GUI)
    -ovládání výstupu na úrovni jednotlivých bodů
    -fonty = zobrazení písma
    -dnes emulátory terminálu (PuTTy, konsole)

Grafické Rozhraní (GUI)
-ovládání počítače pomocí grafických ovládacích prvků
-na monitoru různá okna = výstupy programů
-ovládání myš+klávesnice
-urychlení práce oproti CLI

-LINUX = základ CLI+GUI nádstavba
-WINDOWS = základ GUI+CLI nádstavba

X Window system

=software umožňující vytvořit GUI systému
-navržen jako nezávislý na platformě
-síťově transparentní (aplikace výstup na jiném počítači)
-několik základních komponent (X server, X protokol, knihovna Xlib)
-model klient-server
    -klient = jednotlivé aplikace
    -X-server = zajištění vstupů/výstupů
    -komunikace portem IPC nebo TCP/IP
    -knihovna Xlib = interakce s X serverem

-aplikace v podobě oken = více aplikací na obrazovce
-defaultní implementace = správce oken TWM
-desktopová prostředí (KDE, GNOME, MATE,...)

GUI

GNOME
-aplikace pro většinu potřeb klasického uživatele
-původně vyvinut pro GIMP
-distribuce = Fedora, Red Hat, Ubuntu, Debian, OpenSUSE

KDE
-založeno bez otevřené licence
-kvůli rozsáhlosti je rozdělena do balíčků (schéma) (každá dist své vlastní schéma)
-distribuce = Debian, Kubuntu, OpenSUSE, Fedora KDE

XFCE
-odlehčené prostředí psané v GTK (nízké nároky na HW)
-některé prvky převzal GNOME
-distribuce = Mint, Debian, Xubuntu, Kali

Cinnamon
-psané na GTK +3, odnož GNOME
-vyvíjen pro Mint, později i na jiných distribucích
-při vývoji byl kladen největší důraz na stálost a uživatelskou přívětivost
-plná vybavenost pro GUI uživatele

LXDE
-odlehčené prostředí psané v C na základě knihoven GTK+ (nízké nároky na HW)
-funkcionalita podobná IceWM
-nespokojenost s GTK+3 => přechod na LXQt

LXQt
-vznik fúzí LXDE a Razor-qt (nízké nároky na HW)

Distribuované systémy a systémy reálného času

OS

-programové vybavení nezbytné pro provoz počítače
-správce **fyzických** prostředků daného systému, který **zpracovává** pomocí **logických** prostředků **úlohy** zadané uživatelem

-úloh jsou zpracovávány **běžně** nebo v **reálném** čase

Systémy reálného času (RTOS)

-okamžitá odezva, resp. do "**horní časové hranice"** (max. doba reakce v nejhorším případě)
-běžné OS to neumí, nevhodné pro desktopy
-malé jádro, zbytek procesů běží jako běžné procesy

-systémy řízení letadel, telekomunikace, automobilový průmysl, řízení laboratoří nebo elektráren

Definice: RTS je systém, ve kterém je správnost výstupu závislá nejen na správnosti výsledku výpočtu, ale též na čase, v němž je výsledek spočten.

-dělíme podle přiblížení se "real time" na:
    -Hard RTOS (nedodržení požadavků real time katastrofální následky)
    -Soft RTOS (povolené drobné odchylky v reakcích)

Plánovač RTOS:
-specifickým způsobem přiděluje systémové prostředky
    -minimální latence při reakci na událost
    -minimální latence při přepínání vláken
    -minimální časových okamžiků se zákazem přerušení
    -preemptivní plánování založené na prioritách
-příklady: RMS, EDF

Pike OS
-založen na mikrojádru
-převážně ve vestavěných "embedded" systémech
-letectví, dopravní a automobilní technika, kosmonautika, zdravotnictví

QNX
-RTS na unixovém klonu
-mikrojádro
-stabilní, rychlý i na grafickém rozhraní i na slabších PC, málo aplikací
-podpora sítě, internet v případě nedostupnosti HDD
-původně komerční
RTLinux
-volně dostupný

-má RT mikrojádro, pokud není žádný RT proces, linuxové jádro zpracovává ostatní procesy

OS Windows
-v základní konfiguraci **NELZE** použít jako RTS, ale existují moduly pro podporu RTS

RTX (Real Time Extenstion)
-modul rozšiřující možnosti Win NT-Win srv2003
-jednotka času zkrácena z 5ms na 20mikro sekund
-nezávislý (na vlastním OS) plánovač vláken

VxWorks 6.x
-OS pro řízení v RT
-mikrojádro wind
-nejrozšířenější RTOS v průmyslových aplikací vestavěných systémů
-možnost použití na běžných CPU

Distribuovaný systém

=pracuje na více procesorech
-program rozdělen na vzájemně komunikující části, které mohou běžet na jiném procesoru

-**hrubá granularita**
-části systému spíše větší, samostatnější s malou vzájemnou komunikaci
-pro špatnou a pomalou komunikaci (pomale spojení, špatná kabeláž)
-jemná granularita
-části systému co nejmenší s velkou vzájemnou komunikací
-musí být dobré a spolehlivé propojení částí systému

-distribuované systémy na uživatelské úrovni
-běží na více propojených PC
-každý PC má vlastní OS
-podpora distribuce v SW vrstvě nad nedistribuovaným OS

-distribuované operační systémy
-samostatný OS běžící na síti procesorů, které nemají společnou paměť
-pro uživatele se tváří jako jednoprocesorový
-podpora distribuovaného zpracování přímo v jádru OS
-uživatel neurčuje, kde se data zpracovává nebo kde jsou uložena

-architektura založena na modelu **klient/server:**
-asymetrický = vybrané PC určeny jako servery, ostatní běh user programů
-symetrický = každý PC může být server pro jiné PC, tak klient jiných serverů (větší decentralizace, využití prostředků, rozšiřitelnost)
Výhody:
Ekonomika (cena/výkon), výkon, spolehlivost, rozšiřitelnost, sdílené, vzdálený běh procesu

Nevýhody:
Software (málo existujícího), připojení do sítě (síť může být zahlcena), bezpečnost (snadný přístup k datům)

Požadované funkce:
-efektivní řízení prostředků
-skrytí fyzického rozmístění prostředků
-autorizovaný přístup
-spolehlivá a bezpečná komunikace

Požadované vlastnosti
-transparentnost, flexibilita, spolehlivost, výkonnost, rozšiřitelnost

Transparentnost:
-strukturu či postup operací není vidět
-**přístupová** = proces přistupujeně stejně k lokálním i vzdáleným prostředkům
-lokační = proces nemusí znát fyzické umístění prostředku
-migrační = procesy libovolně přesouvány k různým částem systému
-exekuční = procesy mohou běžet na jakémkoliv procesoru
-replikační = umožňuje vytváření kopii souboru nebo jiných prostředků
-konkurenční = přístup uživatele k libovolným prostředků
-paralelizmová = OS by měl využívat všechny dostupné procesy

Flexibilita:
-přizpůsobování se změnám prostředí, každá část systému co nejvíc samostatná

Spolehlivost
-v případě výpadku některé části systém si rozdělí práci

Výkonnost
-běh aplikace nesmí být výrazně pomalejší než v běžném systému, musí být výkonnější HW

Rozšiřitelnost
-schopnost rozšíření o jakékoliv množství procesorů
-prakticky limitováno problémy při komunikaci a náročností synchronizace

Synchronizace procesů
-procesy neřeší absolutní čas, ale pořadí vzniku událostí
-transakční zpracování
-zámky, potvrzení

Distribuované systémy v AČR = ŠIS, FIS, ISSP, ISL, Zdravis