Státnicový okruh 4:
Informační technologie

26. května 2015

Obsah

1

John von Neumannova a harvardská architektura počítače, princip jeho činnosti. Binární logika, logické operace a funkce, logické obvody. Reprezentace čísel a znaků v paměti počítače. Osobní počítač (PC), základní deska, chipset a sběrnice (interní, externí). Procesor (CPU), vykonávání instrukcí, podprogramy a zásobník, přerušení. Paměti počítače (RAM, cache, disk, diskové pole). Přídavné karty PC, datové mechaniky a média (CD, DVD, paměťové karty), periferie.

2

Operační systém, architektura, poskytovaná rozhraní. Správa procesoru: procesy a vlákna, plánování jejich běhu, komunikace a synchronizace. Problém uváznutí, jeho detekce a metody předcházení. Správa operační pa- měti: segmentace, stránkování, virtuální paměť. Správa diskového prostoru: oddíly, souborové systémy, zajištění konzistence dat.

2.1 Operační systém, architektura, poskytovaná rozhraní

2.1.1 Operační systém

- Základní softwarové (programové) vybavení počítače
- Rozhraní mezi hardware a ostatním software (knihovny, systémové služby, aplikace)
- Rozhraní mezi uživatelem a aplikacemi
- Běží v privilegovaném režimu (jeho funkce jsou upřednostněné)

2.1.2 Architektura OS

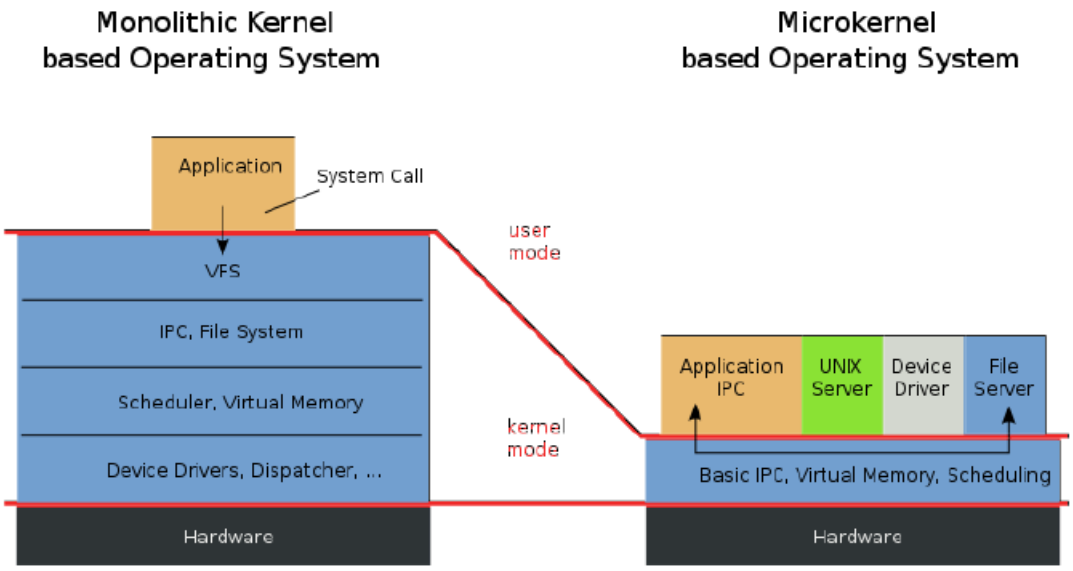
Očekáváme od něj:

- správu a sdílení procesoru (možnost spouštět více procesů "současně")
- správu paměti (procesy jsou v paměti odděleny)
- komunikaci mezi procesy (IPC - Inter-Process Communication)
- obsluhu zařízení a organizaci dat (souborový systém, síťové rozhraní, uživatelské rozhraní)
- Monolitické jádro
  - všechny služby pohromadě - lepší výkon
  - problém s chybnými ovladači
  - Linux, *BSD
- Mikrojádro
  - poskytuje správu adresního prostoru, procesů, IPC
  - oddělení serverů => bezpečnost
  - Minix, Mach
- Hybridní jádro - Windows NT, MacOS X

2.1.3 Poskytovaná rozhraní

1. Hardware - přístup pouze k OS
2. Operační systém - privilegovaný režim, přístup z vyšších vrstev pouze skrze systémové volání
3. Standardní knihovna (libc, CRT) - alokace paměti, stará se o přidělování kousků paměti (malloc)
4. Systémové nástroje - logování, ls, dir atd
5. Aplikace

- Každá vrstva může volat nižší vrstvu, může přeskočit vrstvu
- Hranice mezi vrstvami nemusí být ostré



Obrázek 1: architektura jádra

2.2 Správa procesoru: procesy a vlákna, plánování jejich běhu, komunikace a synchronizace

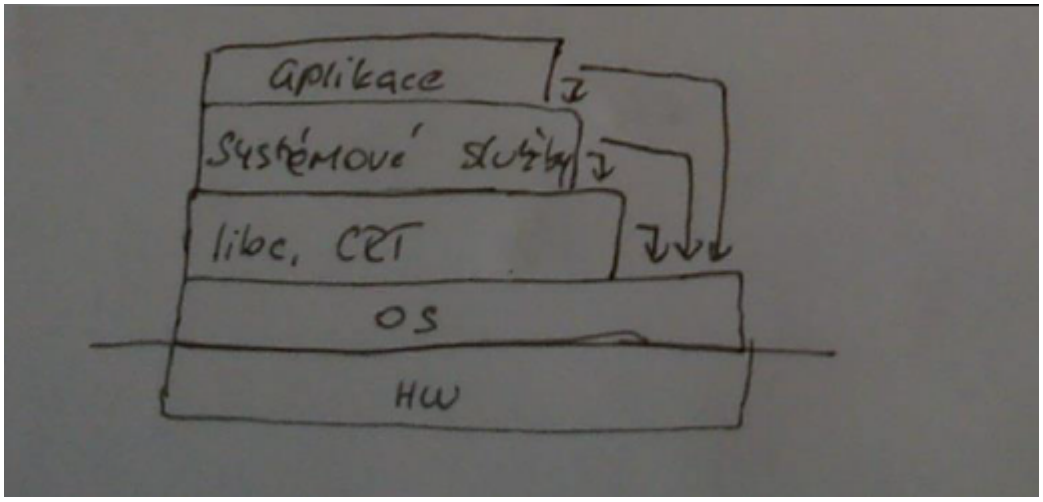
2.2.1 Procesor

OBECNÁ STRUKTURA CPU

- Je to jednotka, která zpracovává instrukce
- Aritmeticko-logická jednotka (ALU) - provádí výpočty
- Řídící jednotka - řídí chod CPU
- Registry - slouží k uchování právě zpracovávaných dat (násobně rychlejší přístup než do paměti)

INSTRUKČNÍ SADA

- Sada ovládající procesor (specifikovaná pro daný CPU/rodinu CPU)
- Instrukce a jejich operandy jsou reprezentovány jako čísla - strojový kód
- Každá instrukce má obvykle 0 až 3 operandy (může to být registr, konstantu nebo místo v paměti)
- Pro snazší porozumění, se instrukce CPU zapisují v jazyce symbolických adres (též vulgárně označován jako Assembler)
- Instrukce jsou zpracovávány (z důvodu větší efektivity) v několika krocích:
  1. načtení instrukce do CPU (Fetch)
  2. dekódování instrukce (decode)
  3. výpočet adres operandů
  4. přesun operandů do CPU
  5. provedení operace (Execute)
  6. uložení výsledku (Write-back)
- Pipelining - umožňuje zvýšit efektivitu CPU (díky rozdělení do jednotlivých kroků je možné provádět více instrukcí zároveň)



Obrázek 2: vrstvy HW/SW

- Superskalární procesor - procesor může mít víc jednotek např. pro výpočty (FPU, ALU), musíme se postarat o správnou synchronizaci
- Problém s podmíněnými skoky (branch prediction) - jakmile se má provést nějaký podmíněný, nevíme, která instrukce bude následovat -> může se stát, že se musí vyprázdnit instrukci z pipeline -> zpomalení

Instr. No.	Pipeline Stage						
1	IF	ID	EX	MEM	WB		
2		IF	ID	EX	MEM	WB	
3			IF	ID	EX	MEM	WB
4				IF	ID	EX	MEM
5					IF	ID	EX
Clock Cycle	1	2	3	4	5	6	7

Obrázek 3: pipelining

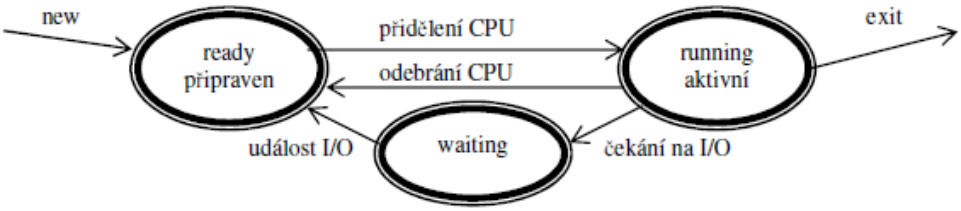
2.2.2 Procesy

- Neformálně: proces = běžící program (vykonává činnost)
- Proces charakterizuje:
  - Kód programu

  - Paměťový prostor
  - Data – statická a dynamická (halda – jsou tam uložená data, které se alokují při běhu procesu, typicky objekty alokované malloc nebo volané přes new)
  - Zásobník
  - Registry
- Oddělení jednotlivých úloh (abstrakce)
- Multiprogramování: (zdánlivě) souběžný běh více procesů
  - kooperativní - proces si sám odebere procesor
  - preemtivní - OS sám určuje, kdy odebere procesu procesor (velikost časového kvanta)
- Komunikace mezi procesy, sdílení zdrojů - synchronizace
- Informace o procesu má OS uloženy v Process Control Block (PCB)
  - identifikace procesu
  - informace o stavu
  - adresu instrukce, kterou bude program pokračovat
  - stav registrů
  - informace k plánování procesů
  - informace o přidělené paměti
  - informace o používaných I/O zařízeních, otevřených souborech, atd.
- potřeba plánování

OBECNÝ ŽIVOTNÍ CYKLUS PROCESU

- Nový (new) – proces byl vytvořen
- Připravený (ready)- proces čeká, až mu bude přidělen CPU
- Běžící (running) – CPU byl přidělen a právě provádí činnost
- Čekající (waiting/blocked) – proces čeká na vnější událost (např. na vyřízení I/O požadavku, synchronizační primitiva)
- Ukončený (terminated) – proces skončil svou činnost (dobrovolně x nedobrovolně)



Obrázek 4: stavy procesu

2.2.3 Plánování procesů

- Potřeba efektivně plánovat procesorový čas
- Časové kvantum: maximální čas přidělený procesu
- Samotné přepnutí procesu má režii (uložení kontextu, vyprázdnění cache) -> latence
- Symmetric Multi-Processing - přibývá problém, jak vybrat procesor
  - oddělené plánování pro každý procesor
  - maska affinity (definuje procesor, na kterém může proces běžet)

TYPY PLÁNOVÁNÍ:

- Dlouhodobé – rozhoduje, zda bude přijat k běhu (změna stavu z NEW na READY)
- Střednědobě – načtení/odložení procesu do sekundární paměti
- Krátkodobé – rozhoduje dostupné procesy ke spuštění na CPU

RŮZNÉ TYPY ÚLOH/SYSTÉMŮ:

- Iterativní
- Dávkové zpracování
- Pracují v reálném čase

OBECNÉ POŽADAVKY NA PLÁNOVÁNÍ PROCESŮ:

- Spravedlnost – každému procesu by v rozumné době měl být přidělen CPU
- Vyváženost – celý systém běží
- Efektivita – maximální využití CPU
- Maximalizace odvedené práce (throughput)
- Minimalizace doby odezvy
- Minimalizace doby průchodu systémem (turnaround) – od spuštění do konce procesu, aby byl co nejmenší časový úsek

ALGORITMY PRO PLÁNOVÁNÍ PROCESŮ

- FIRST-COME-FIRST-SERVED
  - První proces získává procesor
  - Jednoduchý, neefektivní
  - Nepreemptivní
- SHORTEST JOB FIRST
  - Vybere se takový proces, který poběží nejkratší dobu
  - Je potřeba znát (odhadnout) čas, který proces potřebuje
  - Nepreemptivní
- SHORTEST REMAINING TIME NEXT
  - Pokud aktivní proces potřebuje k dokončení činnosti méně času než aktuální, je spuštěn
  - Preemptivní
- ROUND ROBIN (CHODÍ PEŠEK OKOLO)

  - Každý proces má pevně stanovené kvantum
  - Připravené procesy jsou zařazeny ve frontě a postupně dostávají CPU
  - Nevýhoda: přiděluje stejný čas jak systémovým, tak uživatelským procesům
- VÍCEÚROŇOVÁ FRONTA
  - Každý proces má definovanou prioritu
  - Statické x dynamické nastavení priority (např. vyšší priorita pro I/O)
  - Systém eviduje pro každou frontu (čekající procesy)
  - Riziko vyhladovění procesů s nízkou prioritou (Rozšíření: nastavení různých velikostí kvant pro jednotlivé priority)
  - Používán Windows NT, staršími Linuxovými jádry
- SHORTEST PROCESS NEXT - Používá se odhad, podle předchozí aktivity procesu
- GUARANTEED SCHEDULING
  - Máme-li n procesů, každý proces má získat 1/n CPU
  - Volí se proces s nejmenším poměrem
  - Používají novější Linuxové jádra
- LOTTERY SCHEDULING - Proces dostane příděl "losů"
- FAIR-SHARE SCHEDULING - Plánování podle skupin procesů (např. podle uživatelů)

2.2.4 Vlákna

- OS umožňují rozdělit procesy na víc vláken (je základní entitou)
- Každé vlákno má svůj zásobník + registry (přepnutí vláken v rámci procesu je rychlejší)
- vlákna v rámci procesu sdílí paměťový prostor, data a prostředky -> nové problémy se synchronizací

IMPLEMENTACE VLÁKEN:

- Jako knihovna v uživatelském prostoru (první Linuxové systémy)
- Součást jádra operačního systému (Windows a další Linuxové systémy)
- Kombinované řešení

PROCESY A VLÁKNA V UNIXECH

- Původně základní entitou byl proces
- Procesy tvoří hierarchii rodič-potomek (na vrcholu je proces init)
- Vytvoření procesu pomocí volaní fork(), který vytvoří klon procesu
- Přepsání procesu pomocí exec
- Komunikace mezi procesy: zasílání signálů, roury, ...
- Možnost nastavit niceness procesu (prioritu od -20 do 20)
- Vlákna do Unixu přidána až později (implementace se v rámci různých OS liší)
- V Linuxu (pthreads) vnitřně implementovány stejně jako procesy (task), ale sdílí paměťový prostor

PROCESY A VLÁKNA VE WINDOWS:

- Windows NT navrženy s vlákny jako základní entitou pro běh aplikace

- Proces sdružuje vlákna, plánování se účastní vlákno
- Sofistikovaný systém priorit při plánování vláken
- Priorita vláken je odvozena od třídy priority procesu
- Proměnlivá velikost kvant
- Priority boost - dochází k dočasnému navýšení priority
  - po dokončení I/O operace
  - po dokončení čekání na semafor
  - vlákno již dlouho neběželo

2.2.5 Komunikace

- IPC – Inter-process communication
- Procesy oddělené, potřeba kooperace
- Základní kategorie
  - Sdílená paměť
    - Procesy sdílejí kousek paměti - nutná spolupráce se správou paměti
    - Čtení i zápis, náhodný přístup
    - Deklarace, že paměť je sdílená + namapování do adresního prostoru
    - SIGNÁLY - Mechanizmus podobný přerušení (asynchronní volání)
    - ROURY - Mechanizmus umožňující jednosměrnou komunikace mezi procesy
  - Zasílání zpráv
    - obecný mechanizmus komunikace mezi procesy
    - vhodný pro počítače se sdílenou paměti i pro distribuované systémy
    - lze jeho pomocí implementovat vzájemné vyloužení (mutual excluion)
  - Synchronizace
  - Vzdálené volání procedur

2.2.6 Synchronizace

- procesy a vlákna přistupují ke sdíleným zdrojům (paměť, souborový systém)
- příklad: současné zvýšení hodnoty proměnné o 1 (díky preemtivnímu plánování tohle opravdu může nastat)
  1. A: načte hodnotu proměnné X z paměti do registru (X = 1)
  2. A: zvýšení hodnotu v registru o jedna
  3. B: načte hodnotu proměnné X z paměti do registru (X = 1)
  4. A: uloží hodnotu zpět do paměti (X = 2)
  5. B: zvýší hodnotu v registru o jedna
  6. B: uloží hodnotu zpět do paměti (X = 2)
- řešení - atomické operace nebo synchronizace
- je potřeba zajistit, aby v průběhu manipulace s určitými zdroji nemohl manipulovat někdo jiný (vzájemné vyloužení)

ATOMICKÝ PŘÍSTUP DO PAMĚTI

- Obecné přístupy do paměti nemusí být atomické (záležitosti CPU, překladače)

- Lze vynutit určité chování -> klíčové slovo volatile – často záleží na překladači
- Memory barriers umožňují vynutit si synchronizaci (záležitost CPU)

ATOMICKÉ OPERACE:

- Test-and-Set (TAS): nastav proměnnou a vrať její původní hodnotu
- Compare-and-Swap (CAS): ověří, jestli se daná hodnota rovná požadované a pokud ano, přiřadí ji novou (CMPXCHG)
- Fetch-and-Add: vrátí hodnotu místa v paměti a zvýší jeho hodnotu o jedna (XADD)
- Load-link/Store-Conditional (LL/SS): načte hodnotu a pokud během čtení nebyla změněna, uloží do ní novou hodnotu

SYNCHRONIZAČNÍ PRIMITIVA

- slouží k řízení přístupu ke sdíleným zdrojům
- Petersonův algoritmus - umožňuje implementovat vzájemné vyloučení dvou procesů pomocí běžných operací

KRITICKÁ SEKCE

- Obecně třeba zajistit, aby se sdílenými zdroji pracoval jen jeden proces
- Část kódu, kdy program pracuje se sdílenými zdroji (např. pamětí)
- Pokud jeden proces je v kritické sekci, další proces nesmí vstoupit do své kritické sekce
- Každý proces před vstupem žádá o povolení vstoupit do kritické sekce
- Poadavky na kritickou sekci:
  - Vzájemné vyloučení – maximálně jeden proces je v daný okamžik v KS
  - Absence zbytečného čekání – není-li žádný proces v kritické sekci a proces do ní chce vstoupit, není mu bráněno
  - Zaručený vstup – proces snažící se vstoupit do KS, do ní v konečném čase vstoupí
- Řešení:
  - Zablokování přerušení (použitelné v rámci jádra OS); více CPU -> neefektivní
  - Aktivní čekání - Spinlocks – testujeme pořád dokola jednu proměnou

```
Př. 1) ŠPATNÉ - Vstup do kritické sekce a její uzamknutí
                není provedeno atomicky
while (lock); // čekej
lock = 1;
// kritická sekce
lock = 0;


Př. 2)
Uvažujeme následující atomickou operaci
bool test_and_set(bool * target) {
bool rv = *target;
*target = true;
return rv;
}

A kód:
```

```
while (test_and_set(&lock) == true);
// kritická sekce
lock = false;
```

```
Př. 3)
Uvažujeme následující atomickou
operaci, která prohodí dvě hodnoty
void swap(bool *a, bool *b) {
bool tmp = *a;
*a = *b;
*b = tmp;
}

A kód
key = true;
while(key == true)
swap(&lock, &key);
// kritická sekce
lock = false;
```

SEMAFOR

- Chráněná proměnná obsahující počítadlo s nezápornými celými čísly
- Operace P (proberem – zkusit): pokud je hodnota čísla nenulová sníží hodnotu o jedna, jinak čeká, až bude hodnota zvýšena (operace někdy označena i jako wait)

```
void P (Semaphore s) {
while (s <= 0) { }
s--;
}
```

- Operace V (verhogen – zvýšit): zvýší hodnotu o jedna (operace označování jako signal, post)

```
void V (Semaphore s) {
s++;
}
```

- perace P a V se provádí atomicky - operace jsou na začátku a na konci zamykané pomocí spinlocku, protože je kód relativně krátký, tak se nečeká moc dlouho na čekání
- Binární semafor – může nabývat hodnot 0, 1 (mutex, implementace kritické sekce)
- Obecný semafor – slouží k řízení přístupu ke zdrojům, kterých je končené množství
- Implementace s pomocí aktivního čekání nebo OS (pasivní čekání)

```
struct sem {
int value;
struct process * list;
};

void P (struct sem * s) {
s -> value--;
if (s -> value < 0) {
// přidej proces s->list;
block(); // uspi aktuální proces
```

```
}
}

void V(struct sem * s) {
s-> value++;
if (s -> value <= 0) {
// odeber proces P z s s-> list
wakeup(P);
}
}
```

- Operace musí být provedeny atomicky

DALŠÍ SYNCHRONIZAČNÍ NÁSTROJE:

- Bariéry - Synchronizačních metod vyžadujících, aby se proces zastavil v daném bodě, dokud všechny procesy nedosáhnout daného bodu
- Read-Write zámky
  - Vhodné pro situace, které čtou i zapisují do sdíleného prostředku
  - Čtecí a zapisovací režim zámku
  - Vhodný pokud jde rozlišit čtenáře a písaře (písařů je víc)
- Monitor
  - Modul nebo objekt
  - V jeden okamžik může kteroukoliv metodu používat pouze jeden proces/vlákno
  - Nutná podpora programovacího jazyka
  - Java (synchronized), .NET (lock)

2.3 Problém uváznutí, jeho detekce a metody předcházení

2.3.1 Deadlock

- Uváznutí – systém se dostal do stavu, kdy nemůže dál pokračovat
- U množiny procesů došlo k uváznutí (deadlocku), pokud každý proces z této množiny čeká na událost, kterou pouze proces z této množiny může vyvolat.

UŽÍVÁNÍ PROSTŘEDKŮ:

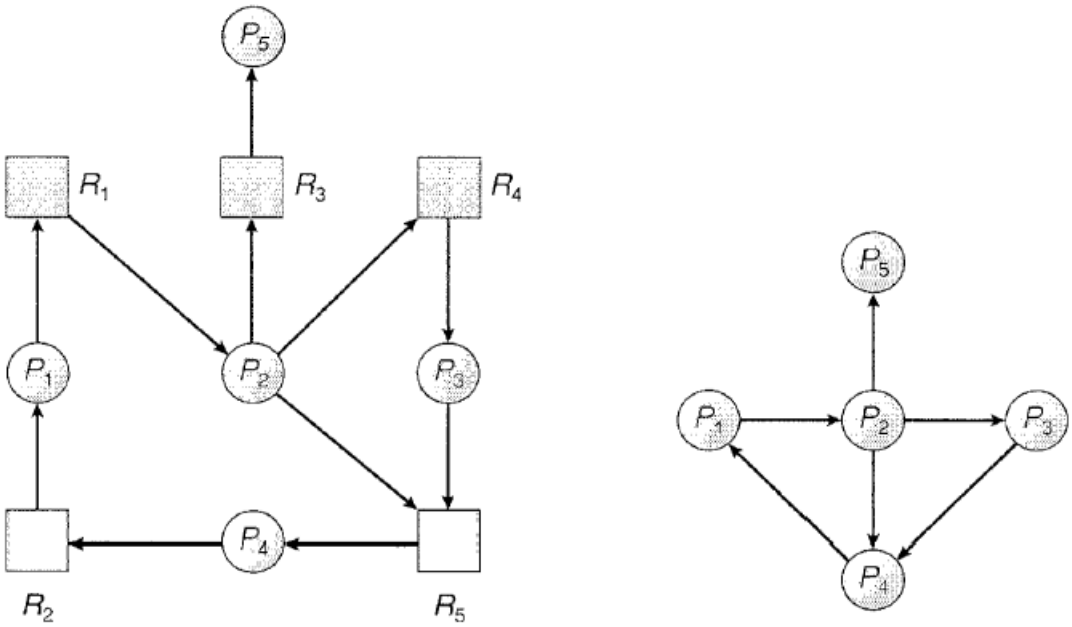
- Request – požadavek na prostředek, není-li k dispozici, proces čeká
- Use – vlákno s prostředkem pracuje
- Release – uvolnění prostředku pro další použití

UŽÍVÁNÍ PROSTŘEDKŮ

- Mutual exclusion – alespoň jeden prostředek je výlučně užíván jedním procesem
- Hold and wait – proces vlastní alespoň jeden prostředek a čeká na další
- No preemption – prostředek nelze násilně odebrat
- Circular wait – cyklické čekání (proces A vlastní prostředek 1, chce prostředek 2, který drží B a současně žádá o 1)

2.3.2 Řešení deadlocku

- IGNORACE - "neřešení", v praxi často používané tzv. pštrosí algoritmus – strčí se hlava do písku a dělá se, že žádný problém není
- DETEKCE
  - Pokud vznikne deadlock, je detekován a některý proces odstraněn
  - K detekci se používá alokační graf prostředků a graf čekání
  - Alokační graf:
    - Orientovaný graf
    - Dva typy uzlů – prostředek, proces
    - Hrana proces-prostředek – proces čeká na prostředek
    - Hrana prostředek-proces – prostředek je vlastněn procesem
  - Graf čekání vznikne vynecháním uzlů prostředků a přidáním hran Pn -> Pm pokud existovaly hrany Pn -> R a R ->Pm, kde Pn a Pm jsou procesy a R je prostředek
  - Ukázka alokačního grafu a grafu čekání:



Obrázek 5: alokacni graf a graf cekani

  - Deadlock vznikne, pokud je v grafu čekání na cyklus
- ZAMEZENÍ VZNIKU (PREVENCE)
  - Snažíme se zajistit, že některá z podmínek není splněna
  - Zamezení výlučnému vlastnění prostředků (často nelze z povahy zařízení)
  - Zamezení držení a čekání
    - Proces zažádá o všechny prostředky hned na začátku
    - Problém s odhadem
    - Plýtvání a hladovění
    - Množství prostředků nemusí být známé předem
    - Jde použít i v průběhu procesu (ale proces se musí vzdát všech prostředků)

  - Zavedení možnosti odejmout prostředek – vhodné tam, kde nelze odejmout prostředky tak, aby nešlo poznat, že byly odebrány
  - Zamezení cyklickému čekání – zavedení globálního číslování prostředků a možnost žádat prostředky jen v daném pořadí
- VYHÝBÁNÍ SE UVÁZNUTÍ
  - Procesy žádají prostředky libovolně
  - Systém se snaží vyhovět těm požadavkům, které nemohou vést k uváznutí
  - Je potřeba znát předem, kolik prostředků bude vyžádáno
  - Tomu je přizpůsobeno plánování procesů
  - Bezpečný stav – existuje pořadí procesů, ve kterém jejich požadavky budou vyřízeny bez vzniku deadlocku
  - Systém, který není v bezpečném stavu, nemusí být v deadlocku
  - Systém odmítne přidělení prostředků, pokud by to znamenalo přechod do bezpečného stavu (proces musí čekat)
  - ALGORITMUS NA BÁZI ALOKAČNÍHO GRAFU
    * Vhodný, pokud existuje jen jedna instance každého prostředku
    * Do alokačního grafu přidáme hrany (proces-prostředek) označující potenciální žádosti procesu a prostředky
    * Žádosti a prostředek se vyhoví pouze tehdy, pokud konverze hrany na hranu typu (prostředek-je vlastněn-procesem) nepovede ke vzniku cyklu
  - BANKÉŘŮV ALGORITMUS
    * Vhodný tam, kde je větší počet prostředků daného typu
    * Na začátku každý proces oznámí, kolik prostředků jakého typu bude maximálně potřebovat
    * Při žádosti o prostředky systém ověří, jestli se nedostane do nebezpečného stavu
    * Pokud nelze vyhovět, je proces pozdržen
    * Porovnávají se volné prostředky, s aktuálně přidělenými a maximálními
    * Uvažujme m prostředků a n procesů
    * Matice m x n
      · Max – počet prostředků, které bude každý proces žádat
      · Assigned – počet přiřazených prostředků jednotlivým procesům
      · Needed – počet prostředků, které bude proces ještě potřebovat (evidentně needed = max – assigned)
    * Vektory velikosti m
      · E – počet existujících prostředků
      · P – počet aktuálně držených prostředků
      · A – počet zdrojů

      1. Najdi řádek i v needed takový, že needed[i] <= A, pokud takový není, systém není v bezpečném stavu
      2. Předpokládej, že proces skončil a uvolnil své zdroje; A <- A + assigned[i] a odstraň řádný i ze všech matic
      3. Opakuj body 1 a 2 dokud nejsou odstraněny všechny procesy nebo není jasné, že systém není v bezpečném stavu

Assigned

	K	L	M	N
A	3	0	1	1
B	0	1	0	0
C	1	1	1	0
D	1	1	0	1
E	0	0	0	0

Needed

	K	L	M	N
A	1	1	0	0
B	0	1	1	2
C	3	1	0	0
D	0	0	1	0
E	2	1	1	0

E = <6, 3, 4, 2>
P = <5, 3, 2, 2>
A = <1, 0, 2, 0>

- Podmínku splňuje proces D → odebrán a A <- <2, 1, 2, 1>
- Podmínku splňuje proces A → odebrán a A <- <5, 1, 3, 2>
- Podmínku splňuje proces B → odebrán a A <- <5; 2; 3; 2>
- Podmínku splňuje proces C → odebrán a A <- <6, 3, 4, 2>
- Podmínku splňuje proces E → odebrán a A <- <6, 3, 4, 2>

Obrázek 6: bankéřův algoritmus

2.4 Správa operační paměti: segmentace, stránkování, virtuální paměť.

2.4.1 Operační paměť

- zásadní část počítače
- uložení kódu a dat běžících v OS
- přístup k zařízením (DMA) - Direct Memory Access - přístup zařízení se namapuje do operační paměti a dá se poté přistupovat k zařízení přes operační paměť
- virtualní paměť umožňuje například přístup k souborovému systému
- z HW pohledu může byt operační paměť realizovaná různými způsoby (DRAM, SRAM, (EEP)ROM, ale i HDD, SSD, flash paměti)
- přístup k CPU k paměti nemusí být přímočarý (L1, L2 a L3 cache)
- pro jednoduchost budeme HW stránku věci zanedbávat

POŽADAVKY NA OPERAČNÍ PAMĚŤ

- evidence prostoru volného a přiděleného procesům
- přidělování a uvolňování paměti procesů
- přesunutí (přiděleného prostoru) - program by neměl být závislý na místě, kde se v paměti nachází (nutně k umožňování swapování)
- ochrana (přiděleného prostoru) - jednotlivé procesy by měly být izolovány
- sdílení - pokud je to žádoucí, mělo by být možné sdílet některé části paměti mezi procesy (2x spouštěný stejný program nebo knihovny, které jsou nahrané pouze jednou a poté jsou používány několika procesy)

- logická organizace - paměť počítače (spojitý prostor, "sekvence bytů") vs. typický program skládající se z modulů (navíc některých jen pro čtení nebo ke spouštění)
- fyzická organizace - paměť může mít více částí/úrovní (RAM? disk); program jako takový se nemusí vlézt do dostupné paměti RAM

ADRESNÍ PROSTORY (LOGICKÁ ORGANIZACE)

- v současných OS existuje několik různých nezávislých adresních prostorů, způsobů číslování paměťových buněk
- každý proces by měl mít vlastní paměťový prostor (izolace)
- různá zařízení mají odlišné adresní prostory
- ve spolupráci s hardwarem dochází k mapování fyzické paměti do adresního prostoru procesu
- např. grafická paměť může být namapována od adresy 0xD0000000

2.4.2 Stránkování

- adresní (logický) prostor procesu je rozdělen na menši úseky - stránky (pages)
- fyzická paměť je rozdělena na úseky stejné délky - rámce (frame, page frames)
- provádí se mapování logický adres -> na fyzické
- procesy už nemusí být umístěny v souvislých blocích paměti
- výpočet fyzické adresy musí byt implementovaný HW (efektivita) - pokud by implementoval OS, nemohl by ho používat
- CPU si udržuje stránkovací tabulku
- logická adresa má dvě části: pd, kde p je číslo stránky a d je offset
- fyzická adresa vznikne jako fd, kde f je číslo rámce v tabulce stránek příslušného stránce p

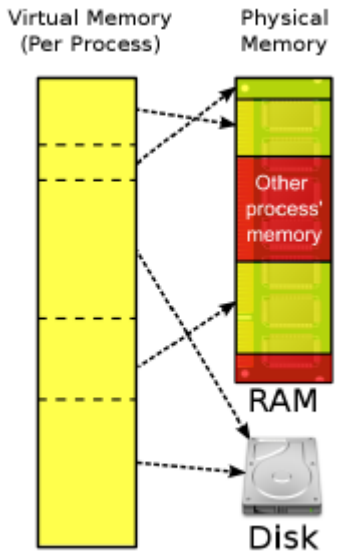
2.4.3 Segmentace

- paměť je rozdělena na několik segmentů (například kód, data, zásobník)
- speciální případ přidělování souvislých bloků paměti
- stránkování je obvykle pro programátora nerozeznatelné
- naproti tomu segmentace umožňuje rozdělit program do logických celků (kód, data)
- při použití segmentace a stránkování programy nepracují přímo s lineární adresou
- používají adresu ve tvaru segment + offset a ta se až převádí na fyzickou adresu pomocí stránkování
- ochrana paměti přes začátek a délku segmentu + oprávnění

2.4.4 Virtuální paměť

MOTIVACE

- paměť RAM je relativně drahá -> nemusí vždy dostačovat
- aktualně používaná data (například instrukce) musí byt v RAM, nepoužívaná data nemusí (velké programy -> nepoužívané funkce)
- je vhodné rozšířit primární paměť (RAM) o sekundární (např.: HDD)
- zvětšením dostupné paměti je možné zjednodušit vývoj aplikací (není potřeba se omezovat v množství použité paměti)
- sekundární paměť bývá řádově pomalejší
- k efektivní implementaci je potřeba spolupráce HW (MMU) a OS
- z pohledu aplikace musí být přístup k paměti transparentní
- virtuální paměť (VM) je součástí soudobých OS (swapování)
  - Windows NT - stránkovací soubor (pagefile.sys)
  - Linux - swap partition (ale může být i soubor)
- bezpečnost dat v sekundární paměti? (např.: po vypnuti počítače, řeší se např. kódováním)



Obrázek 7: virtuální paměť

VIRTUÁLNÍ PAMĚŤ

- způsob správy operační paměti počítače
- umožňuje předložit běžícímu procesu adresní prostor paměti, který je uspořádán jinak nebo je dokonce větší, než je fyzicky připojená operační paměť RAM
- virtuální adresy - pracují s nimi strojové instrukce, resp. běžící proces
- fyzické adresy - odkazují na konkrétní adresové buňky paměti RAM
- převod mezi virtuální a fyzickou adresou je zajišťován samotným procesorem
- virtuální paměť je implementována pomocí stránkování paměti spolu se stránkováním na disk, které rozšiřuje operační paměť o prostor na pevném disku

2.5 Správa diskového prostoru: oddíly, souborové systémy, zajištění konzistence dat

2.5.1 Souborové systémy

- způsob organizace dat ve formě souborů (a většinou i adresářů) tak, aby k nim bylo možné snadno přistupovat
- souborové systémy jsou uloženy na vhodném typu elektronické paměti, která je umístěna přímo v počítači (pevný disk nebo CD,. . .)

MOTIVACE

- potřeba uchovávat větší množství dat (primární paměť nemusí dostačovat)
- data musí být perzistentní (musí přežít ukončení procesu)
- k souborům musí být umožněn souběžný přístup
- řešení v podobě ukládání dat na vnější paměť (např.: disk)
- data ukládána do souborů tvořících souborový systém (File System/FS)
- soubor jako proud bytů (doprovázen doplňujícími informacemi)
- souborový systém jako abstrakce (odstínění od implementačních detailů)
- zajímavý problém: pojmenování objektů (souborů) a jejich organizace

OPERACE SE SOUBORY

- create - vytvoření souboru
- write/append - zápis do souboru (na konec, popř. přepis); souvislý blok vs. postupný zápis
- read - čtení ze souboru (do přichystaného bufferu)
- seek - změna pozice
- erase - odstranění souboru (uvolnění místa); link a unlink
- truncate - zkrátí daný soubor na požadovanou velikost
- open - otevře soubor, aby s nim šlo manipulovat přes popisovač (file descriptor, file handle)
- close - uzavře soubor
- "soubory"nemusí mít jméno
- jeden soubor může byt otevřen vícekrát (vice ukazatelů na pozici v souboru)

ORGANIZACE SOUBORŮ

- soubory jsou rozlišované podle názvů (často specifikované pro daný OS nebo FS)
- rozlišování velkých a malých písmen (Unix vs. MS-DOS a Windows)
- MS-DOS: požadavek na jméno souboru ve tvaru 8+3: jméno + přípona (zůstalo zachováno ve FAT)
- typicky se soubory organizuji do adresáře (složek)
- každý adresář může obsahovat běžné (popř. speciální) soubory i další adresáře -> stromová struktura
- k přístupu k souboru se používají:
  - absolutní cesty: /foo/bar/baz.dat

  - relativní cesty: foo/bar.dat -> každý proces má aktuální adresář
- operace s adresáři: Create, Delete, OpenDir, CloseDir, ReadDir, Rename
- struktura nemusí být hierarchická -> obecný graf (acyklický, cyklický)
  - hardlink - ukazatel na soubor (jeho tělo/obsah) (výhoda - transparentnos, nevýhoda - odkaz pouze v jednom FS)
  - symlink - soubor jako ukazatel na jiný soubor (specifikovaný cestou) (lze používat odkaz na jiný FS, ale nejsou plně transparentní - může vzniknout odkaz na neexistující soubor)

2.5.2 Oddíly

DĚLENÍ DISKU

- každý fyzický disk se skládá z jedné nebo více logických částí (partition, oddíl); popsané pomocí partition table daného disku
- v každém partition může existovat souborový system (označovaný jako svazek)
- v Unixech je každý svazek připojen (mounted) jako adresář (samostatný svaze pro /, /home, /usr)
- ve Windows jednotlivé svazky označeny (a:, b:, c:, . . .); ale funguje i mountovani (preferovaný jeden svazek pro vše)
- Virtual File System (VFS)
  - využití abstrakce =) umožňující kombinovat různé FS do jednoho VFS
  - možnost připojit běžný soubor jako svazek (i svazek jako soubor)
  - síťové disky (NFS, CIFS)

STRUKTURA SOUBORŮ

- často je soubor chápán jako proud bytů
- sekvenční vs. náhodný přístup
- společně s daty jsou k souboru připojena metadata (atributy)
  - vlastník souboru
  - přístupová práva
  - velikost souboru
  - příznaky (skrytý, archivace, spustitelný, systémový)

2.5.3 Souborové systémy - implementace

OČEKÁVÁME:

- (budeme předpokládat souborové systémy pro práci s disky s rotujícími částmi)
- schopnost pracovat se soubory a disky adekvátní velikosti
- efektivní práce s místem (evidence volného místa, nízká fragmentace)
- rychlý přístup k datům
- eliminace roztroušení dat na disku
- odolnost proti poškození při pádu systému (výpadku napájení) -> rychlé zotavení
- snapshoty - obraz disku v jednom okamžiku ke kterému jsme schopni se vrátit

- komprese dat
- možnost zvětšovat/zmenšovat FS za běhu
- kontrolní součty
- defragmentace za běhu
- správa oprávnění

STRUKTURA DISKU

- pro jednoduchost předpokládáme, že struktura disku je lineární
- MBR - master boot record: informace o rozdělení disku na svazky + zavaděč
- sektor disku - obvykle velikost 512 B
- pracuje se s většími bloky 1 - 32 kB (často 4 kB nebo 8 kB)
- jednotlivé svazky obsahují souborový systém (vlastní organizace dat)

FAT

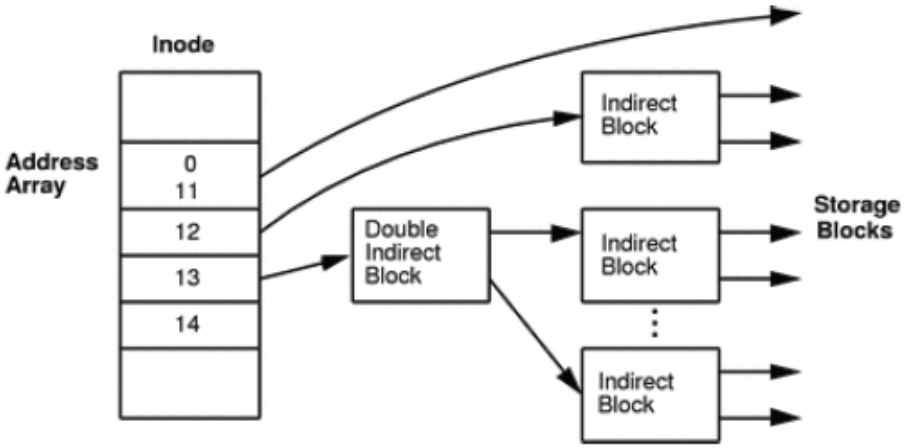
- souborový system pro MS-DOS (přežil až do Windows ME)
- jednoduchý design
- soubory se jmény ve tvaru 8+3, nepodporuje oprávnění
- nemá metody proti poškození dat
- disk rozdělený na bloky (clustery)
- soubory popsány pomocí File Allocation Table (FAT) - spojový seznam
- disk (oddíl) rozdělen na 4 části:
  - bootsector (rezervovaná oblast) + informace o svazku
  - 2x FAT
  - kořenový adresář
  - data
- adresáře jako soubory, kořenový svazek je vytvořen hned na začátku
- původní FAT nepodporoval adresáře
- FAT12, 16, 32: podle velikosti clusteru (max. kapacity - 32 MB, 2 GB, 8 TB)
- Virtual FAT
  - s Windows 95
  - podpora dlouhých jmen (LFN)
  - až 256 znaků
  - soubor má dvě jména - dlouhé a ve tvaru 8+3
- exFAT
  - určen pro flash paměti
  - podpora větších disků (512 TB, 64 ZB)
  - podpora v novějších Windows (původně Windows CE 6)

UFS: UNIX FILE SYSTEM

- v různých variantách přítomných v unixových OS - BSD, Solaris, System V, Linux (ext[2,3,4])
- disk se skládá:
  - bootblock - místo pro zavaděč OS
  - superblock - informace o souborovém systému
  - místo pro inody
  - místo pro data

INODY

- struktura popisující soubor
- informace o souboru
  - typ souboru, vlastníka (UID, GID), oprávnění (rwx)
  - časy (vytvoření, přístup)
  - počet ukazatelů, počet otevřených popisovačů
- informace o uložení dat
- struktura inody umožňuje mít řídké soubory
- adresář je soubor obsahující sekvenci dvojic (jméno souboru, číslo inody)
- k evidenci volného místa a inod se používají bitmapy



Obrázek 8: inode

FS V LINUXU

- Linux nemá jeden hlavní FS
- nejčastěji se používá: Ext2/3/4
- název souboru může mít až 256 znaků
- vychází z UFS
- ext2: maximální velikost souboru 16 GB - 2 TB, disku: 2 TB - 16 TB
- ext3: přidává žurnál (3. úrovně - journal, ordered, unordered), binárně kompatibilní s ext2
- ext4: maximální velikost souboru 16 GB - 2 TB, disku 1 EB; optimalizace alokací

NTFS

- hlavní souborový systém Windows NT
- kořeny v OS/2 a jeho HPFS (vyvíjen od roku 1993)
- velikost clusteru podle velikosti svazku (512 B - 4 KB) <- max. velikost disku 256 TB, max. velikost souboru 16 TB
- oproti FAT (souborovému systému W9x) ochrana před poškozením + práva
- žurnálování a transakce
- dlouhé názvy (255 znaků) + unicode
- podpora standartu POSIX; hardlinky, symlinky
- adresáře
  - opět technicky soubory
  - jména v B+ stromech
  - části metadat jsou součástí adresáře, část metadat je součástí souborů
- **STRUKTURA DISKU**
  - na začátku disku: boot sector
  - 12% MFT (Maste File Table), 88% data souborů
  - MFT je soubor popisující všechny soubory na FS (MFT je taky soubor)
  - MFT se skládá ze záznamů o velikosti 1 KB
  - každý soubor je popsán tímto záznamem
  - informace o souborech včetně jména, časů, atd. uloženy jako záznam v MFT jako dvojice atribut-hodnota
  - tělo souboru je taky atribut (uniformní přístup, možnost uložit malé soubory přímo do MFT
  - v případě potřeby může jeden soubor zabrat vice záznamů v MFT
  - případně lze použít místo mimo MFT (rezidentní a nerezidentní atributy)

2.5.4 Zajištění konzistence dat

TRANSAKČNÍ PAMĚŤ

- rozdělení programu na části, které jsou provedeny "atomicky"(ACI)

```
void transfer(Account a1, Account a2, int amount) {
atomic {
a1.balance += amount;
a2.balance -= amount;
}
}
```

- změny se neprovádí přímo, ale ukládají se do logu, v případě "commitu"se ověří, jestli došlo ke kolizi
- zjednodušení vývoje - na úkor režie
- omezení: transakci musí být možné provést vícekrát

```
void transfer(Account a1, Account a2, int amount) {
atomic {
a1.balance += amount;
a2.balance -= amount;
launchTheMissiles();
}
}
```

- podpora jako knihovny /rozšíření Javy/C# /C++

ŽURNÁLOVÁNÍ

- data se zapisují v transakcích (přesun FS z jednoho konzistentního stavu do druhého)
- nejdříve se transakce zapíše do žurnálu (logu)
- po zapsání do žurnálu je záznam označen speciální značkou a data se můžou zapsat na disk
- po zapsání na disk je zápis z žurnálu odstraněn
- při připojení FS se kontroluje stavu žurnálu
  - zápis záznamu do žurnálu nebyl dokončen (transakce se neprovede)
  - případně, transakce se provede podle informací ze žurnálu
- často se žurnálují jen metada - v případu výpadku můžeme ztratit data, ale není narušena bezpečnost a není potřeba obnovovat strukturu FS
- žurnál je cyklický; při zaplnění se zapíší/uvolní ty na začátku
- pro transakce je potřeba atomických zápisů na disk
- cache a buffery komplikují implementaci atomických zápisů na disk

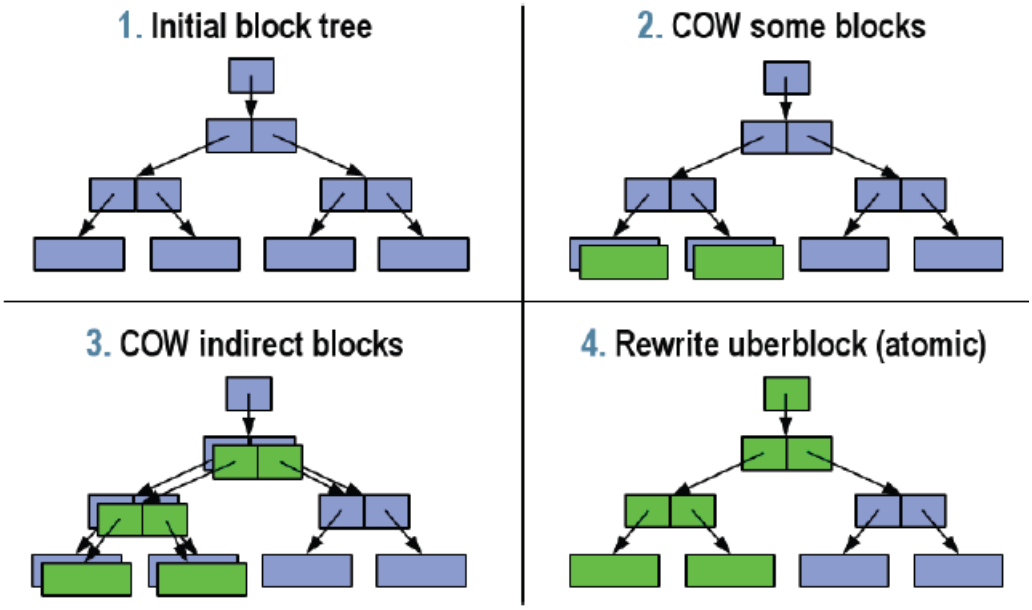
Copy-on-Write (COW)

- může být užitečné sdílet paměť (komunikace, úspora místa)
- dva stejné programy/knihovny v paměti
- stránky více procesů jsou navázány na jeden rámec
- daná stránka má nastavený příznak CoW
- dojde-li k pokusu o zápis, vznikne výjimka a procesu je vytvořena kopie rámce
- bude-li na rámec s příznakem CoW odkazovat jenom jedna stránka, příznak se odstraní
- fork() - vytvoří identickou kopii procesu; data jsou sice izolovaná, ale je možné sdílet kód
- úspora místa; úspora strojového času (není potřeba vytvářet kopie stránek/rámců); nízká penalizace pokud stránky přepíšeme
- virtuální paměť umožňuje další optimalizace (mapování souborů do paměti, atd.)

ZAJIŠTĚNÍ KONZISTENCE DAT

- používání blokového zařízení: RAID (Redundant Array of Independent Disk)
  - RAID-0 (stripping) - nepomůže konzistenci
  - RAID-1 (mirroring)
  - RAID-2 Hammingův kód - dělí data po bitech, disk pro paritu
  - RAID-3 dělí data po bytech, XOR, disk pro paritu

  - RAID-4 podobné jako RAID-3, používá paritní bloky
  - RAID-5 podobné jako RAID-4, paritní bloky jsou distribuovány
  - RAID-6 podobné jako RAID-5, Reed-Solomon kód, dva paritní bloky
- u dat jsou evidovány kontrolní součty (ochrana proti tichému poškození (chyba HW i SW)
- konzistence založena na metodě Copy-on-Write
- používaná data nikdy nejsou přepsána (nejdřív jsou zapsána data a pak jsou (automaticky) změněna metadata



1. alokujeme bloky pro data
2. pro kazda data vytvorime novy blok
3. vytvorime metadata, ktere predstavuji bloky dat
4. po zapsani provedeme prohozeni těch částí, které jsou měněné

Obrázek 9: zachování konzistence

- infrastruktura pro vytváření snapshotů/klonů souborového systému
- výhodné slučovat operace do transakcí

3

Klasifikace (LAN/MAN/WAN) a služby počítačových sítí. Síťová architektura TCP/IP a referenční model ISO OSI. Strukturovaná kabeláž, přepínaný Ethernet a WLAN. Protokol IP, adresa a maska, směrování, IP multicast. Protokoly TCP a UDP, správa spojení a řízení toku dat. Systém DNS, překlad jména (na IP adresu, reverzní), protokol. Služby WWW, elektronické pošty, přenosu souborů a vzdáleného přihlášení.

POZOR! packet = jakákoliv naformátovaná zpráva jako packet vs. datagram = packet nespolehlivého přenosu dat.

3.1 Klasifikace (LAN/MAN/WAN)

Klasifikace sítí podle různých kritérií: rozlehlost, rychlost přenosu (klasické, vysokorychlostní), forma aplikace, dělení podle postavení uzlů (perr-to-peer, klient-server), podle druhu přenášení signálů (analog vs. digital) aj.

3.1.1 Dělení podle rozlehlosti

- **PAN (Personal Area Network)**: Sítě s nejmenší rozlehlostí. Propojení mobilů, PDA, atd. Požadavky na odolnost vůči rušení, nízká spotřeba energie, snadná implementace.Ppřenosová rychlost není prioritou (zpravidla jen několik Mb/s). Dosah pouze několik metrů. Typické technologie Bluetooth, IrDA, Wifi.
- **LAN (Local Area Network)**: Sítě propojující koncové uzly typu počítač, tiskárna, server. V soukromé správě, dosah jednotky km (v rámci budovy/komplexu budov). Přenosové rychlosti 10Mb/s až 1Gb/s. Sdílené využití přenosového média. Ethernet, Wifi.
- **MAN (Metropolitan Area Network)**: Propojení několika LAN (účelem přenosové sítě, charakterem lokální). V rámci města (desítky km). Přenosové rychlosti jak vyšší (několik Gb/s), tak i nižší (<1Gb/s) ve srovnání s LAN.
- **WAN (Wide Area Network)**: Rozsáhlé sítě spojující LAN/MAN (páteřní sítě, telekomunikační - broadband). Velké vzdálenosti (prakticky neomezené). Mohou být soukromé i veřejné. Vysoké přenosové rychlosti (až stovky Gb/s). Příklad GPRS, xDSL, aj. Pronájem kapacity sítě = vyhrazené nesdílené využití přenosového média.

3.1.2 Dělení podle topologie

Topologie počítačové sítě říká, jak jsou vlastně prvky v této síti uspořádány.

Kruhová topologie (RING) Každý počítač je propojen přímo s předchozím a následujícím počítačem v kruhu. V LAN je používána velmi málo, používá se v průmyslových sítí nebo sítích MAN.
Výhody:

- lehce rozšiřitelná struktura
- malý počet spojů
- snadné vysílání, zprávy v kruhu od stanice ke stanici

Nevýhody:

- výpadek libovolné stanice zapříčiní výpadek celé sítě, úplný výpadek sítě při přerušení kabelu v libovolném místě
- poměrně veliké nebezpeční odposlechu síťové komunikace, která prochází přes spojovací počítače

Problém výpadku se řešil tzv. dvojitým kruhem, ve kterém byly stanice propojeny dvěma kruhy, každý v opačné směru.

Sběrnicová topologie (BUS) Tato topologie patří k nejstarším, všechny stanice jsou připojeny na pasivní společné médium, které sdílejí. Dnes už se tato topologie příliš nepoužívá, ale na začátku devadesátých let byla dominantní. Tím společným médiem byl koaxiální kabel, pomocí kterého se jednotlivé počítače připojily do sítě.
Výhody:

- nezávislost stanic na výpadku libovolné jiné stanice
- levné náklady takového řešení
- neexistence aktivních prvků
- snadné všesměrové vysílání

Nevýhody:

- úplný výpadek sítě při přerušení kabelu v libovolném místě
- nutnost vyřešení přístupu stanic k médiu (kdo bude vysílat)

Výhody a nevýhody jsou relativní a poplatné době. To, že se v dobách používání této topologie v sítích LAN, považovala absence aktivních prvků za výhodu, bychom v dnešní době řadili spíše k nevýhodě.

Hvězdicová topologie Tato topologie je dnes jednoznačně nejpoužívanější topologií v sítích LAN. Myšlenka spočívá v tom, že existuje centrální prvek, který spojuje všechny prvky. Dříve tím centrálním prvkem býval počítač, dnes je aktivní prvek (HUB nebo SWITCH).
Výhody:

- lehce rozšiřitelná struktura
- výpadek libovolné stanice neznamená výpadek celé sítě
- větší možnosti zabezpečení, při použití aktivních prvků typu SWITCH je většina síťové komunikace skryta před ostatními účastníky sítě

Nevýhody:

- nutnost použití hubu nebo switche
- vyžaduje veliké množství kabelů a je tak náročná na montáž

Páteřní topologie Páteřní topologií rozumíme situaci, kdy pomocí určité topologie propojujeme celé sítě LAN. Páteřní topologie může být zapojena jako sběrnice, hvězda i kruh, často se používá zapojení typu kruh. Jejím základem je vytvoření nezávislé hlavní části, která propojuje důležité celky. Na ni se naopak připojují různé subsítě nebo segmenty. V případě výpadku libovolného segmentu zůstává provoz na páteři neohrožen. Páteř může mít vyšší přenosovou rychlost.

3.1.3 Služby počítačových sítí

- připojení k síti
- vzdálený přístup, sdílení výpočetních prostředků a přenos dat (sdílené databáze, perr-to-peer sítě, sdílené soubory)
- sdílení technických prostředků (tiskárny, faxy, disky, apod.)
- adresářové služby (jednotný přístup do informačního systému a k informacím z centrální databáze, např. LDAP, Active Directory)
- elektronická pošta a výměna dokumentů (objednávky, faktury, atd.)
- online komunikace/multimedia (např. IRC, VoIP, straming, hry) - vysoké nároky na síť

- informační služby, internetové aplikace (WWW, business a desktopové aplikace)
- monitorování a vzdálená administrace sítě

Komunikace uzlů a propojovacích prvků sítě na různých úrovních:

- nižší - přenos bloků dat, většinou nespolehlivý (bez potvrzení a opakování přenosu), nespojová komunikace
  - **unicast**: dvoubodová, základní
  - **multicast**: bod-skupina, např. straming, virtuální sítě
  - **broadcast**: bod-všichni, např. konfigurace a zapojení do sítě
- vyšší - komunikace aplikací, většinou spolehlivá (s potvrzením doručení, popř. opakování přenosu), spojově orientovaná (vytvořeno spojení mezi aplikacemi)
  - **peer-to-peer**: zpravidla rovnocenná výměna dat
  - **klient-server**: hiearchická, forma požadavek-odpověď

Typy koncových uzlů:

- **pracovní stanice** (work station, klient): převážně využívá služeb sítě
  - **tenký klient**: znakový/grafický HW terminál, pouze zprostředkování vstupu a výstupu pro vzdálený server, nemůže pracovat samostatně
  - **tlustý klient**: osobní počítač - klientské části síťových služeb i lokální úlohy, může (do určité míry) fungovat samostatně
- **server**: převážně poskytuje služby v síti
  souborový (FTP, NFS, SMB), databázový/adresářový (SŘBD, LDAP), poštovní (IMAP, POP3, SMTP), terminálový (telnet, SSH), informační/WWW (HTTP), komunikační (IM, VoIP), tiskový, aj.

3.2 Síťová architektura TCP/IP a referenční model ISO OSI.

Snaha o vytvoření univerzálního konceptu sítě (topologie, formy a pravidlakomunikace, poskytování služeb atd.). Požadavky: decentralizace služeb, rozumná adresace uzlů, data zasílána v nezávislých blocích, směrování bloků, zabezpečení, kontrola a řízení přenosu aj. Dříve proprietární uzavřená řešení, následně standatdizace s koncepcí komunikace nezávislé na implementaci.
Komunikace ve vrstvách

- definované službami poskytované vyšším vrstvám a využívající služby nižších vrstev, implementace skryté okolním vrstvám
- samostatné vrstvy s funkcemi podobnými v rámci vrstvy a odlišnými v různých vrstvách, nezávislé na implementaci

Komunikace mezi vrstvami pomocí **mezivrstvových protokolů** - na každé komunikující straně zvlášť; skrze **programová rozhraní**; prostřednictvím přístupových bodů; využívající tzv. služební primitiva.

Obecná služební primitiva

- žádost o službu (request)
- oznámení poskytovatele o přijetí žádosti (indication) - nepovinné
- odezva poskytovatele (response), příp. vytvoření spojení
- potvrzení odezvy žadatelem (confirmation) - nepovinné

Komunikace ve stejných vrstvách mezi entitami (zařízeními) pomocí **vrstvových protokolů**.

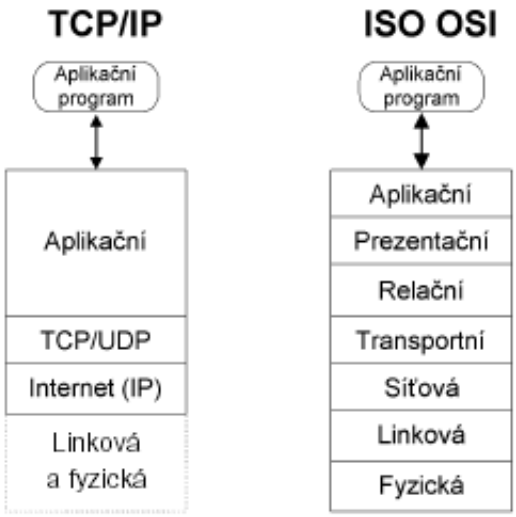
3.2.1 Protokol

= souhrn pravidel (norem a doporučení) a procedur pro komunikaci (výměnu dat). Obsahuje syntaktická a sémantická pravidla výměny protokolových datových jednotek.
Protokolové datové jednotky = režijní informace a data (rámce, pakety, segmenty). Komunikace zprostředkována sousední nižší vrstvou. Na straně odesílatele **zapouzdřování** od nejvyšší po nejnižší vrstvu. Na straně příjemce **rozbalování** dat v opačném směru.

Síťová (protokolová) architektura = definice vrstev, služeb funkcí, protokolů a forem komunikace. Normalizované (OSI, TCP/IP) a firemní proprietární (Novell NetWare, SMB, Apple Appletalk aj.)

Abstraktní referenční síťový model Abstrakce konkrétních síťových architektur - nemusí podporovat všechny funkce modelu (např. průmyslové sítě nepodporují směrování, propojení pomocí mostů a bran).

3.2.2 Referenční model ISO/OSI



Propojení otevřených systémů = zařízení podporující příslušné normy. Deinovány koncové uzly (koncová datová zařízení) a mezilehlé uzly (propojovací prvky).
Každá ze sedmi vrstev vykonává skupinu jasně definovaných funkcí potřebných pro komunikaci. Pro svou činnost využívá služeb své sousední nižší vrstvy. Své služby pak poskytuje sousední vyšší vrstvě. Podle referenčního modelu není dovoleno vynechávat vrstvy, ale některá vrstva nemusí být aktivní. Takové vrstvě se říká nulová, nebo transparentní.
Na počátku vznikne požadavek některého procesu v aplikační vrstvě. Příslušný podsystém požádá o vytvoření spojení prezentační vrstvu. V rámci aplikační vrstvy je komunikace s protějším systémem řízena aplikačním protokolem. Podsystémy v prezentační vrstvě se dorozumívají prezentačním protokolem. Takto se postupuje stále níže až k fyzické vrstvě, kde se použije pro spojení přenosové prostředí. Současně se při přechodu z vyšší vrstvy k nižší přidávají k uživatelským (aplikačním) datům záhlaví jednotlivých vrstev. Tak dochází k postupnému zapouzdřování původní informace. U příjemce se postupně zpracovávají řídící informace jednotlivých vrstev a vykonávají jejich funkce.

Fyzická vrstva Specifikuje fyzickou komunikaci. Aktivuje, udržuje a deaktivuje fyzické spoje mezi koncovými systémy. Definuje všechny elektrické a fyzikální vlastnosti zařízení (tvary konektorů; typy médií = kroucená dvojlinka, optické vlákno, mikrovlny; přenosové rychlosti). Obsahuje rozložení pinů, napěťové úrovně a specifikuje vlastnosti kabelů.
Funkce poskytované fyzickou vrstvou:

- Navazování a ukončování spojení s komunikačním médiem.
- Spolupráce na efektivním rozložení všech zdrojů mezi všechny uživatele.

- Modulace neboli konverze digitálních dat na signály používané přenosovým médiem (a zpět) (A/D, D/A převodníky).

HW zařízení: HUB, opakovač.

Linková vrstva Poskytuje spojení mezi dvěma sousedními systémy. Uspořádává data z fyzické vrstvy do logických celků = **rámce (frames)**: záhlaví s linkovou adresou příjemce a odesílatele (např. MAC) + data + zápatí s kontrolním součtem (CRC) celého rámce. Stará se o nastavení parametrů přenosu linky, oznamuje neopravitelné chyby. Formátuje fyzické rámce, opatřuje je fyzickou adresou a poskytuje synchronizaci pro fyzickou vrstvu. Příkladem je MAC u Ethernetu. Poskytuje propojení pouze mezi místně připojenými zařízeními a tak vytváří doménu na druhé vrstvě pro směrové a všesměrové vysílání.
HW zařízení: most, přepínač, síťová karta, přístupový bod.

Síťová vrstva Stará se o směrování v síti a síťové adresování. Poskytuje spojení mezi systémy, které spolu přímo nesousedí. Poskytuje funkce k zajištění přenosu dat různé délky od zdroje k příjemci skrze jednu případně několik vzájemně propojených sítí při zachování kvality služby, kterou požaduje přenosová vrstva. Síťová vrstva poskytuje směrovací funkce a také reportuje o problémech při doručování dat. Jednotkou přenosu je **síťový packet**: záhlaví se síťovou adresou příjemce a odesílatele (např. IP) + data + zápatí jen výjimečně.
Funkce poskytované síťovou vrstvou:

- abstrakce různých linkových technologií
- správa linkových spojení, multiplexování síťových spojení do linkových (více datových toků kombinováno do jednoho)
- formátování dat do packetů
- směrování packetů
- zjišťování a oprava chyb
- vytváření podsítí

HW zařízení: směrovač, brána.

Transportní vrstva Zajišťuje přenos dat mezi koncovými uzly. Jejím účelem je poskytnout takovou kvalitu přenosu, jakou požadují vyšší vrstvy. Vrstva nabízí spojově (TCP) a nespojově orientované (UDP) protokoly. Jednotkou přenosu je **transportní packet**: záhlaví s transportní adresou příjemce a odesílatele + data.
TCP: Protokol garantuje spolehlivé doručování a doručování ve správném pořadí. TCP také umožňuje rozlišovat a rozdělovat data pro více aplikací (například webový server a emailový server) běžících na stejném počítači. TCP využívá mnoho populárních aplikačních protokolů a aplikací na internetu, včetně WWW, e-mailu a SSH.
UDP: Na rozdíl od protokolu TCP nezaručuje, zda se přenášený datagram neztratí, zda se nezmění pořadí doručených datagramů, nebo zda některý datagram nebude doručen vícekrát. UDP je vhodný pro nasazení, které vyžaduje jednoduchost nebo pro aplikace pracující systémem otázka-odpověď (např. DNS, sdílení souborů v LAN).
Funkce poskytované transportní vrstvou:

- adresování (transportní na síťové)
- správa síťových spojení
- multiplexování a větvení
- rozdělení dat na datagramy, segmentace, formátování
- řízení proudu dat (správné pořadí datagramů), optimalizace služeb
- koncová detekce a oprava chyb

Umožňuje **duplexní přenos** (= přenos oběma směry).

Relační vrstva Smyslem vrstvy je organizovat a synchronizovat dialog mezi spolupracujícími relačními vrstvami obou systémů a řídit výměnu dat mezi nimi (např. sdílení síťového disku). Umožňuje vytvoření a ukončení relačního spojení, synchronizaci a obnovení spojení, oznamovaní výjimečných stavů. Jednotka přenosu je **relační packet**: poze data.
Funkce poskytované relační vrstvou:

- organizace a synchronizcae dialogu výměny dat (pomocí kontrolních bodů)
- zobrazení relačních spojení do transportních
- správa transportních spojení

Prezentační vrstva Funkcí vrstvy je transformovat data do tvaru, který používají aplikace (šifrování, konvertování, komprimace). Formát dat (datové struktury) se může lišit na obou komunikujících systémech, navíc dochází k transformaci pro účel přenosu dat nižšími vrstvami. Vrstva se zabývá jen strukturou dat, ale ne jejich významem, který je znám jen vrstvě aplikační.
Funkce poskytované prezentační vrstvou:

- transformace a výběr reprezentace dat
- formátování, komprese, zabezpečení (šifrování), integrita dat
- transparentní přenos zpráv (nezná jejich význam

Aplikační vrstva Účelem vrstvy je poskytnout aplikacím přístup ke komunikačnímu systému a umožnit tak jejich spolupráci.
Funkce poskytované aplikační vrstvou:

- zprostředkování funkcionality sítě
- přenos zpráv, určení kvality, synchronizace
- identifikace, stanovení pověření
- dohoda o ochraně, o opravách chyb

Protokoly: SMTP, SSH, Telnet, POP3, DHCP, FTP, DNS aj.

Funkce společné více vrstvám Komunikace **se spojením** má 3 fáze: navázání spojení, přenos dat, ukončení spojení. Dohoda na parametrech, použití potvrzování přijetí/nepříjetí (spolehlivost), stejné pořadí dat na vstupu i výstupu.
Komunikace **bez spojení**: při každém přenosu všechny parametry, nezávislý přenos datových jednotek, různé pořadí na vstupu a výstupu, spolehlivost i nespolehlivost.
Dále se může mezi těmito typy konvertovat. Transportní služby musí být se spojením.

3.2.3 TCP/IP

použití v síti Internet, nejpoužívanější síťová architektura. Síť tvořena směrovači, specializovanými bránami (bezpečnostní, aplikační, telekomunikační), lokálními sítěmi a koncovými zařízeními. Vlastní protokoly.

Vrstva síťového rozhraní Nejnižší vrstva umožňuje přístup k fyzickému přenosovému médiu. Je specifická pro každou síť v závislosti na její implementaci.

Síťová vrstva Vrstva zajišťuje především síťovou adresaci, směrování a předávání datagramů.

Transportní vrstva je implementována až v koncových zařízeních (počítačích) a umožňuje proto přizpůsobit chování sítě potřebám aplikace. Poskytuje transportní služby kontrolovaným spojením spolehlivým protokolem TCP nebo nekontrolovaným spojením nespolehlivým protokolem UDP.

Aplikační vrstva Vrstva aplikací. To jsou programy (procesy), které využívají přenosu dat po síti ke konkrétním službám pro uživatele.
Aplikační protokoly používají vždy jednu ze dvou základních služeb transportní vrstvy: TCP nebo UDP, případně obě dvě (např. DNS). Pro rozlišení aplikačních protokolů se používají tzv. porty, což jsou domluvená číselná označení aplikací. Každé síťové spojení aplikace je jednoznačně určeno číslem portu a transportním protokolem (a samozřejmě adresou počítače).

3.2.4 Bezpečnost

Obecné metody ochrany:

- omezování přenosu dat a přístupu k síti: blokace, filtrace
- autorizace přístupu: jméno a heslo, vícefaktorové, speciální protokoly
- zabezpečení kanálu: šifrování, výměna klíčů
- autenticita zpráv: digitální podpis, certifikáty

TCP/IP původně žádné zabezpečení, ponecháno na aplikaci.
Útoky:

- falešná adresace (spoofing)
- analýza hesla, trojský kůň
- odposlech
- odmítnutí služby (DoS, zahlcení, vyčerpání zdrojů)
- zneužití chyb aplikací
- ...

Ochrana:

- firewall (oddělení vnitřní sítě od vnější)
- překlad adres (NAT) - vlastní skrytá adresace
- aplikační brány (proxy)
- autentizace komunikujících stran
- zabezpečení komunikace (šifrování)
- opatření proti zahlcení
- ...

3.3 Strukturovaná kabeláž, přepínaný ethernet a WLAN

3.3.1 Strukturovaná kabeláž

Obecný pojem pro metalické a optické prvky, které umožňují propojení.

Telekomunikační přípojky Slouží pro připojení koncových uživatelských zařízení - např. stolní počítač, notebook, analogový nebo ISDN telefon, VoIP telefon či síťová tiskárna. Telekomunikační zásuvky bývají umístěny přímo v pracovních prostorách (např. kancelářích) každé budovy, a to buď přímo ve zdi, v parapetních žlabech, případně podlahových systémech tak, aby byly lehce dostupné.

Patch panely Na rozdíl od běžně dostupných zásuvek jsou patch panely umístěny v rozvaděčích v telekomunikační místnosti a nejsou tedy pro běžné uživatele přístupné. Patch panely slouží správci sítě k připojení jednotlivých uživatelů do aktivních zařízení jako jsou switche nebo telefonní ústředny. Pro připojení vodičů do zářezových konektorů se používá narážecí nástroj.

Horizontální kabely, Patch kabely Měděné kabely obsahující čtyři kroucené páry, které vzájemně propojují telekomunikační zásuvky a patch panely.

Koaxiální kabel se dnes už nepoužívá. Vnější (stínění) a vnitřní (jádro) vodič. BNC konektor. Maximálně 500m.

Kroucená dvojlinka je tvořena páry vodičů, které jsou po své délce pravidelným způsobem zkrouceny a následně jsou do sebe zakrouceny i samy výsledné páry. Oba vodiče jsou v rovnocenné pozici (i v tom smyslu, že žádný z nich není spojován se zemí či s kostrou), a proto kroucená dvojlinka patří mezi tzv. symetrická vedení. Maximálně 100m. Nestíněná (UTP) vs. stíněná (STP). Konektor RJ-45.

Jednou ze základních odlišností kroucené dvojlinky od koaxiálního kabelu je skutečnost, že na kroucené dvojlince není možné dělat odbočky. Kroucená dvojlinka je proto použitelná jen pro vytváření dvoubodových spojů. Nemožnost vytvářet odbočky pak ale nutně znamená, že prostřednictvím kroucené dvojlinky nelze vytvořit sběrnicovou topologii sítě.

Optická vlákna Vlákna imunní vůči elektromagnetickému rušení. Dvě vrstvy skla (obal a jádro). Vícevidové vlákno - odraz od rozhraní skel. Jednovidové vlákno - buzení laserem. Různé optické konektory (FC, LC, ST) nutné navařit. Vlákno simplexní, pro duplexní přenos dvojice vláken.

3.3.2 WLAN

= bezdrátové lokální sítě. Důvodem pro WLAN je rozšiřitelnost, nižší náklady, mobilita, snadná použitelnost, roaming (vysílače si klienta předávají). Připojení od poskytovatele i v rámci LAN. Norma IEEE 802.11.

Konfigurace

- peer-to-peer/ad-hoc: přímá komunikace mezi stanicemi (do 10ti stanic)
- AP (access point): propojení WLAN a LAN, bezpečnostní prvky (autorizace, filtrace, šifrování, atd.)
- s více AP (roaming): AP propojeny pevnou sítí, klient se připojuje k zařízení s nejsilnějším signálem
- point-to-point: připojení pomocí dvou AP

Přenosové medium jsou radiové vlny (2,4 GHz - 802.11b/g/n nebo 5GHz - 802.11a/n) Není třeba licence, vzájemné rušení.
antény: omezení výkonu normou ČTÚ na 100 mW

Bezpečnost

- obtížná ochrana proti odposlechu na fyzické vrstvě
- SSID: označení AP (název sítě), AP jej nemusí vysílat
- WEP: Autentizace stanic vůči AP (40 bitů heslo společně s MAC adresou), lze v krátkém čase prolomit
- WPA, WPA2: silná šifra AES, autentizace (heslo, EAP)

3.3.3 přepínaný Ethernet

Norma IEEE 802.1: Propojení sítí na úrovni MAC, tvorva VLAN. Propojení LAN pomocí přepínačů a mostů, propojení WAN pomocí směrovačů.

podvrstva LLC je podvrstvou linkové vrstvy. Funguje jako rozhraní mezi síťovou vrstvou a podvrstvou Media Access Control (MAC). V dnešních protokolech linkové vrstvy plní často jen funkci multiplexování. LLC hlavička říká linkové vrstvě, co má provést s paketem, když je přijat datový rámec.

Ethernet Rámce se šíří segmentem po sdíleném mediu nezávisle na sobě, stanice (síťové rozhraní) "vidí"všechny, ale přijímá jen ty adresované jí nebo všeobecné. **Promiskuitní režim** (přijímá všechny rámce). Uzly rovnocenné, jen jeden v daný čas využívá sdílené medium pro vysílání rámců (= half duplex). 10Gbit režim už bez sdíleného media (= full duplex).

Protokol CSMA/CD Kolizní přístup ke sdílenému médiu: Stanice naslouchá (Carrier) a vysílá, až nevysílá nikdo jiný. Když takto začne vysílat více stanic najednou, dojde ke kolizi. První stanice detekující kolizi vyšle **signal JAM** a všechny stanice se na náhodný čas odmlčí (čas odvozený od MAC adresy).

Přepínaný Ethernet využívá místo opakovače most. Normy IEEE 802.1d a 802.1q.
Přepínač (Switch) je multiportový most, který zpracovává příchozí rámce na svých rozhraních paralelně a vytváří souběžné virtuální linkové segmenty (dvojbodové plně duplexní spoje) propojující odesílatele s adresátem. Virtuální linkové segmenty jsou bezkolizní, kolize nastávají pouze u různých odesílatelů se shodným adresátem. Pro přepínání používá **přepínací matici**. Dokáže spojit sítě s různými rychlostmi (má vyrovnávací paměť, store-and-forward) a může hned po načtení hlavičky rámce načítat další (cut-trough).

Ethernet II předepsaný pro lokální sítě připojené přímo do internetu.

MAC adresy:

- globální (první tři bajty udávají výrobce), "zadrátované"v trvalé paměti síťové karty
- skupinová (nejnižší bit prvního bytu = 1)
- všesměrová (samé 1)

3.4 Protokol IP, adresa a maska, směrování, IP multicast

3.4.1 Protokol IP

Poskytuje nespolehlivou nespojovou službu (nevytváří spojení , nepotvrzuje příjem packetů). Spojuje lokální sítě do globální sítě Internet. Tvořen několika dílčími protokoly: vlastní protokol IP, ICMP (diagnostika a signalizace mimořádných stavů), IGMP (skupinové adresování), ARP a RARP (zjištění linkové adresy k IP adrese a opačně). Síťové rozhraní uzlu má alespoŇ jednu IP adresu.
IP protokol byl původně vyvinut pro potřeby komunikace v Internetu. IP adresa musí být v dané síti jednoznačná (jedno rozhraní může mít více IP adres, ale stejná IP adresa nemůže být na více rozhraních), avšak lze používat NAT a privátní IP adresy.

IP packet záhlaví 20B povinných položek + volitelné položky, data, maximální délka 64kB

Fragmentace Linkové rámce mají omezenou velikost (max. jednotky kB). Maximální velikost = **MTU (Maximum Transfer Unit)**. IP packet může mít až 64kB -> fragmentace. Pokud je zakázána (DF), je packet zahozen (odesílateli signalizováno pomocí ICMP). Zvyšuje nároky režie, OS se snaží vytvářet packety menší než je MTU, aby k fragmentaci nedocházelo.
Fragmentace = dělení IP packetu na fragmenty o celkové délce menší, než MTU. **Freagment** = samostatný IP packet se stejnou hlavičkou, jako původní. Skládání fragmentů do původního packetu provádí pouze příjemce. Lze fragmentovat fragmenty.

Složení hlavičky packetu: bity označující verzi (IPv4 vs IPv6), typ služby = ToS (nepoužíváno, v současnosti QoS), délka packetu, tag pomáhající k rekonstrukci packetu z více fragmentů, příznak DF (zakázání fragmentace) a MF (indikace dalších fragmentů), TTL (Time To Live - přes kolik routerů může projít, než bude zničen), bity označující protokol (ICMP, UDP, TCP, ...), kontrolní součet, zdrojová a cílová IP.

doba života TTL zamezení nekonečného "toulání"packetu. Každý směrovač snižuje alespoň o 1 (a musí tedy přepsat kontrolní součet v záhlaví). Při 0 se packet zahodí a odesílateli je to signalizováno pomocí ICMP.

3.4.2 IP adresa

Každé síťové rozhraní může mít jednu či více IP adres (jednozačných). Přidělení IP adresy staticky síťovému rozhraní pomocí ipconfig (MS Windows) nebo ifconfig/ip (UNIX, GNU/Linux).

IPv4 Číslo o délce 4B. Notace v desítkovém tvaru oddělené tečkou po každém bytu (např. 127.0.0.1).
Struktura adresy Adresa se dělí na: adresu sítě, adresu podsítě a adresu počítače. Adresu sítě pro danou koncovou síť přiděluje poskytovatel připojení (oficiálně ji přiděluje lokální registrátor, ale tím bývá právě poskytovatel). Je třeba o ni požádat prostřednictvím standardních formulářů. Strukturu lokální části adresy – zda bude rozdělena na podsítě a jaká její část bude případně věnována adrese podsítě a jaká adrese počítače – určuje správce dotyčné sítě. Ten také přiděluje adresy.
Třídy sítě Historická záležitost, třídy A-E (rozdělení podle toho jaká část IP adresy určuje síť a jaká část počítač)

třída	1. bajt	minimum	maximum	maska podstítě
A	0–127	0.0.0.0	127.255.255.255	255.0.0.0
B	128–191	128.0.0.0	191.255.255.255	255.255.0.0
C	192–223	192.0.0.0	223.255.255.255	255.255.255.0
D	224–239	224.0.0.0	239.255.255.255	255.255.255.255
E	240–255	240.0.0.0	255.255.255.255	—

Například síť třídy A je taková síť, kde první číslo čtyřčíselné IP adresy označuje síť a zbylá tři čísla označují adresu hostitele. Třída B používá první dvě pro označení síťové adresy a zbývající dvě pro hostitele a síť třídy C používá první tři čísla pro označení sítě a poslední pro označení hostitele.
Postupem času se však i toto rozdělení ukazovalo jako velice nepružné a s rostoucím nedostatkem adres se hledaly způsoby na vylepšení původního systému.

Síťová maska Maska sítě zapsaná v binárním tvaru má zleva samé jedničky až do místa, kde končí číslo sítě a na místě části pro číslo síťového rozhraní jsou samé nuly. Maska sítě je v IPv4 zapisována stejně jako IP adresa – čtyřmi desítkovými čísly oddělenými tečkami, z nichž každé odpovídá jednomu oktetu v 32bitové adrese. Někdy se udává maska sítě zkrácenou formou zápisu, kterou označujeme CIDR (Classless Inter-Domain Routing, beztřídní mezidoménové směrování), ve kterém je možno v IP adrese hranici mezi číslem sítě a číslem počítače umisťovat libovolně (mezi dva libovolné bity). Adresa síťového rozhraní je pak zapisována pomocí IP adresy a délky prefixu (místo prefixu, který určuje délku čísla sítě v bitech lze použít i desítkový zápis masky sítě) (192.168.24.0/21 nebo 192.168.24.0/255.255.248.0). Protože má maska sítě na místě čísla sítě samé jedničky, můžeme z libovolné IP adresy určit číslo sítě pomocí logického součinu IP adresy a masky sítě. Pokud známe číslo sítě a masku, můžeme spočítat rozsah IP adres, které se v této podsíti nacházejí a tím i počet IP adres, které je možné použít pro síťová rozhraní v této podsíti.
Speciální adresy:

- celá adresa samé 0: tento uzel (loopback bez přidělené adresy)
- uzel 0: adresa sítě
- síť 0: uzel v síti (nepoužívá se)
- uzel samé 1: všesměrová adresa sítě (network broadcast)

- celá adresa samé 1: všesměrová adresa lokální sítě (local broadcast), nesměřuje se
- 127.cokoliv: smyčka (loopback), většinou 127.0.0.1, odesílaný packet "ihned přijde"

IPv6 adresa má délku 128 bitů. Zpisuje se jako osm skupin po čtyřech hexadecimálních číslicích (např. 2001:0718:1c01:0016:0214:22ff:fec9:0ca5). Úvodní nuly v každé skupině lze ze zápisu vynechat. Pokud adresa obsahuje několik po sobě jdoucích nulových skupin, lze místo nich zapsat jen "::". Tato zkratka smí být v adrese pouze jedna (např. loopback je ::1).
Tři typy adres:

- Individuální (unicast) která identifikují právě jedno síťové rozhraní.
- Skupinové (multicast) označují skupinu síťových rozhraní, jejímž členům se mají data dopravit. Skupinově adresovaný datagram se doručuje všem členům skupiny.
- Výběrové (anycast) označují také skupinu síťových rozhraní, data se však doručují jen jejímu nejbližšímu členovi.

IPv6 neobsahuje všesměrové (broadcast) adresy. Byly nahrazeny obecnějším modelem skupinových adres a pro potřeby doručení dat všem zařízením připojeným k určité síti slouží speciální skupinové adresy (např. ff02::1 označuje všechny uzly na dané lince). Zavádí se také koncepce dosahu (scope) adres. Adresa je jednoznačná vždy jen v rámci svého dosahu. Nejčastější dosah je pochopitelně globální, kdy adresa je jednoznačná v celém Internetu. Kromě toho se často používá dosah linkový, definující jednoznačnou adresu v rámci jedné linky (lokální sítě, např. Ethernetu).

DHCP Používá se pro automatickou konfiguraci počítačů připojených do počítačové sítě. DHCP server přiděluje počítačům pomocí DHCP protokolu zejména IP adresu, masku sítě, implicitní bránu a adresu DNS serveru. Platnost přidělených údajů je omezená, proto je na počítači spuštěn DHCP klient, který jejich platnost prodlužuje.

Směrování Chce-li počítač vyslat IP datagram, musí nejprve zjistit, do které sítě zadaná IP adresa náleží. Tuto činnost označujeme jako směrování IP datagramů. **Směrování (routing)** = odeslání packetů na další směrovač nebo cílový uzel (next hop). **Předávání (forwarding)** = předávání packetu v rámci směrovače mezi jeho síťovými rozhraními. Packety mohou být **filtorvány** nastavením filtračních pravidel v OS, na základě IP záhlaví, TCP/UDP záhlaví nebo aplikačního protokolu
Směrování probíhá podle směrovací tabulky, ve které je na každém řádku definována jedna síť (pomocí čísla sítě a masky). Tabulku vytváří administrátor **ručně** nebo vzniká **dynamicky** pomocí směrovacích protokolů. Tabulka je setříděna podle jednotlivých masek sítí tak, že záznamy s nejdelší maskou (tj. nejkonkrétnější) jsou ve směrovací tabulce umístěny nejdříve, což umožňuje ve směrovací tabulce vytvářet výjimky (z větší sítě vyjmout jako speciální případ menší podsíť). Zjišťování příslušnosti cílové IP adresy k síti (definované v řádku směrovací tabulky) probíhá tak, že se postupně pro každý řádek ve směrovací tabulce provede logický součin cílové IP adresy s maskou definované sítě a porovná se výsledek s definovaným číslem sítě. Pokud dojde ke shodě, patří IP adresa do sítě definované na řádku, kde ke shodě došlo.

IP multicast je metoda přeposílání IP datagramů z jednoho zdroje skupině více koncových stanic. Místo odesílání jednotlivých datagramů ke každému cíli je odeslán jediný datagram. K identifikaci jednotlivých multicastových skupin se používají IP adresy třídy D. Aby cílová stanice mohla přijímat multicastová data, musí být přihlášena alespoň do jedné skupiny a samozřejmě router sítě musí multicasting podporovat, udržovat tedy v sobě tabulku skupin, které má odchytávat. Router, který přijímá informace z multicastových skupin, se od uzlů snaží zjistit, které skupiny mají být vysílány uzlům do bezprostředně připojené sítě. Tuto službu nám zajišťuje IGMP protokol, díky kterému se uzly mohou přidávat do skupin.
IGMP protokol dynamicky registruje jednotlivé hostitele, patřící do skupiny adres D. Hostitel identifikuje členství ve skupině odesláním zpráv protokolu IGMP a data zasílá vždy všem členům skupiny. Směrovače používající protokol IGMP pravidelně naslouchají zprávám protokolu IGMP a systematicky odesílají dotazy s cílem zjistit, které skupiny jsou v síti LAN aktivní.

Každá IP adresa se v síti musí překládat na MAC adresu tedy i multicastová, právě díky této adrese se přenáší multicastová data v lokální síti. Zde však nastává problém v možné nejednoznačnosti, protože 48 bitová MAC adresa musí mít pro multicast prefix 01:00:5e následovaný nulovým bitem a zbylých 23 bitů MAC adresy je vyplněno posledními 23 bity IP adresy. Více stejně končícím IP adresám je tak přiřazena stejná multicastová MAC adresa. Tento problém se někdy nazývá 32-to-1 overlapping, protože právě 32 multicastových IP adres je mapováno na jedinou multicastovou MAC adresu.

3.5 Protokoly TCP a UDP

Opakování:

- **spojová služba** - mezi aplikacemi navázáno spojení s plně duplexní výměnou dat, spolehlivá služba (ztracená a poškozená data znova vyžádána), integrita dat zajištěna kontrolním součtem, zpracovává souvislý prud/tok (uspořádaných) dat od vyšší vrstvy (stream)
- **nespojová služba** - nenavazuje spojení, nespolehlivá služba (nezaručuje pořadí doručených dat, znovuzaslání poškozených/ztracených), zpracovává nesouvislé části dat od vyšší vrstvy

Port = indentifikátor aplikace (aplikace jich mlže používat víc), transportní adresa, je to číslo o délce 2B (0 až 65535).
Rozdělení na:

- privilegované - může je používat pouze privilegovaná aplikace, rozsah 0 - 1023
- neprivilegované - může je použít kdokoliv, pokud je volný

Pro běžné služby jsou známá "standarní"čísla portů.

3.5.1 Transmission Control Protocol (TCP)

TCP je spojově orientovaný protokol pro přenos toku bajtů na transportní vrstvě se spolehlivým doručováním. TCP využívá mnoho populárních aplikačních protokolů a aplikací na internetu, včetně WWW, e-mailu a SSH.

TCP segment Aplikace posílá proud (stream) bajtů TCP protokolu k doručení sítí, TCP rozděluje proud bajtů do přiměřeně velkých segmentů. (Velikost segmentů je určena parametrem **MTU (maximum transmission unit)** linkové vrstvy sítě, ke které je počítač připojen.) TCP pak předá takto vzniklé pakety IP protokolu k přepravě internetem do TCP modulu na druhé straně TCP spojení. TCP ověří, že se pakety neztratily tím, že každému paketu přidělil **pořadové číslo**, které se také použije k ověření, že data byla přijata ve správném pořadí. TCP modul na straně příjemce posílá zpět potvrzení pro pakety které byly úspěšně přijaty. Pokud by se odesilateli potvrzení nevrátilo do rozumné doby (dané obousměrným zpožděním, anglicky round-trip time, **RTT**), vypršel by odesilatelův časovač a (pravděpodobně ztracená) data by vyslal znovu. TCP protokol ověřuje, zda přenesená data nebyla poškozena šumem tím, že před odesláním spočte **kontrolní součet**, uloží jej do odesílaného paketu a příjemce kontrolní součet vypočte znovu a ověří, že se shodují.
Příznaky:

- CWR, ECN - pro oznámení zahlcení sítě
- URG - segment nese urgentní data, která mají být zpracována přednostně
- ACK - signalizace správného pořadového čísla přijímaného bytu
- RST - odmítnutí navazovaného TCP spojení
- SYN - nová sekvence číslování odesílaných bytů, pořadové číslo odesílaného bytu je číslo 1. bytu toku dat, nastaven u 1. segmentu při navazování spojení
- FIN - ukončení odesílání dat, ukončení spojení pro daný směr přenosu dat

Délka okna je počet bytů, které je příjemce schopen přijmout - předcházení zahlcení sítě.
Protože TCP je spojovaná transportní služba, musí se před odesíláním dat navázat spojení mezi klientem a serverem. K tomu slouží trojcestný handshaking (three-way handshake). Obě strany otevřou port (pomocí socketu), klient v **aktivním režimu**, server **pasivní režim** (očekávání spojení). V průběhu navazování spojení se obě strany dohodnou na číslu sekvence a potvrzovacím čísle. Pro navázání spojení se odesílají datagramy s nastavenými příznaky SYN a ACK.

Navazování spojení:

- Klient odešle na server datagram s nastaveným příznakem SYN a náhodně vygenerovaným číslem sekvence ISN (x), potvrzovací číslo=0, maximální délka přijímaných segmentů (MSS).
- Server odešle klientovi datagram s nastavenými příznaky SYN a ACK, potvrzovací číslo=x+1, číslo sekvence ISN je náhodně vygenerované (y), navrhnutá MSS směrem od serveru.
- Klient odešle datagram s nastaveným příznakem ACK, číslo sekvence=x+1, číslo odpovědi=y+1.

Navrhované MSS je menší, než MTU, aby se zamezilo IP fragmentaci.
Obě strany si pamatují číslo sekvence své i protistrany. Používají se totiž i pro další komunikaci a určují pořadí paketů. Když úspěšně proběhne trojcestný handshaking, je spojení navázáno a zůstane tak až do ukončení spojení. To se může zneužít na SYN flood útok.

Ukončování spojení: probíhá podobně jako jeho navázání. Používá se k tomu příznaků FIN a ACK:

- Klient odešle datagram s nastaveným příznakem FIN
- Server odpoví datagramem s nastaveným příznakem ACK
- Server odešle datagram s nastaveným příznakem FIN
- Klient odpoví s nastaveným příznakem ACK

Teprve po těchto čtyřech krocích je spojení ukončeno.

Odmítnutí spojení: Pokud cílový port na straně příjemce není otevřen (např. segmenty jsou zahazovány firewallem nebo neběží aplikace serveru), klient, bez odpovědi serveru, po čase opakuje požadavek na navázání spojení do vypršení celkového času nebo počtu pokusů = časová prodleva. Odmítnutí nastává kdykoliv po zaslání segmentu s příznakem RST. Například u neúspěšného navázání šifrovaného spojení SSL/TLS.

3.5.2 Řízení toku dat

Ztráta segmentu

- Odesílatel: Má definovaný časový interval pro příjem potvrzovacího segmentu od příjemce (retransmission timeout). Při ztrátě nebo poškození segmentu po vypršení intervalu nebo příjmu tří opakovaných stejných potvrzení od příjemce **opakuje odesílání segmentu**. Hodnota intervalu se dynamicky mění podle stavu sítě na základě předpokládané doby odezvy (vypočítané z RTT).
- Příjemce: Má definovaný časový interval pro příjem následujícího segmentu v toku dat (dle pořadových čísel). Při neobdržení následujícího segmentu po vypršení časového intervalu nebo obdržení segmentu s dalšími daty mimo pořadí **opakuje potvrzení přijetí** předchozích dat. Ukládá si data obdržené mimo pořadí do bufferu a po obdržení chybějícího segmentu **potvrdí příjem všech dat**.

Zpoždění odpovědi je výhodné u interaktivních (konzolových) aplikací (Telnet, SSH, příkazový kanál FTP) vyměňujících malé segmenty. Ilustrační situace: Uživatel stiskne klávesu, klient odešle znak serveru (v segmentu IP packetu v linkovém rámci), server potvrdí příjem, zpracuje znak a odešle ho klientovi ke zobrazení, klient potvrdí příjem a zobrazí -> velká režie.
Zpoždění je tedy potvrzení příjmu dat se zpožděním a ne hned, kdy se mohou data k odeslání nahromadit. **Delayed ACK** je odesílání v intervalech (např. 200 ms (< jak 500 ms)). **Nagleův algoritmus**: odeslání dat až po obdržení dalších dat od druhé strany nebo až objem dat k odeslání překročí MSS. Lze vypnout pomocí síťového API OS příznakem TCP_NODELAY.

Posuvné okno (Sliding windows) Využití při odesílání většího množství dat, zamezení zahlcení sítě. Segmenty se posílají bez potvrzování každého až do počtu odeslaných bytů rovno délce posuvného okna. Délka okna udává, kolik bytů je příjemce schopen přijmout či zpracovat. Při navazování spojení příjemce navrhne délku posuvného okna spolu s MSS (typicky 6-8x MSS) a pak ji může v potvrzovacích segmentech měnit nebo vynulovat (zakázat odesílateli odesílat další data, když "nestíhá"). Potvrzováním příjmu dat se okno posouvá a mění velikost = řízení toku dat.

Zahlcení sítě Posuvné okno udává množství dat akceptované příjemcem. Pokud je příliš velké a síť na straně příjemce plně využita nebo pomalá, odesílatel může síť zahltit a ta začne data zahazovat. Řešením je **okno zahlcení** na straně odesílatele, které udává množství nepotvrzených dat, které je možné odeslat, aniž by došlo k zahlcení sítě (cíl je samozřejmě největší možné). Odesílatel pak odesílá data do menší z velikostí posuvného okna a okna zahlcení.
Dvě fáze určení velikosti okna zahlcení:

- **Pomalý start**: Od navázání spojení se velikost okna zahlcení s každým potvrzovacím segmentem zdvojnásobuje až do ztráty segmentu nebo dosáhnutí velikosti posuvného okna nebo parametru SSTHRESH (hranice pravděpodobnosti zahlcení sítě - typicky 64kB). Po třech stejných potvrzeních předchozího segmentu se okno zahlcení zmenší na polovinu a stejně nastaví SSTHRESH (ta však musí být minimálně 2xMSS). Po neobdržení potvrzení se okno zahlcení nastaví na MSS, SSTHRESH se nastaví na 2xMSS a začne se znovu.
- **Předcházení/vyhýbání se zahlcení**: Následuje po pomalém startu, kdy se s každým potvrzením navyšuje. Algoritmy vyhýbání se zahlcení: Reno, New Reno, BIC, CUBIC, aj. SACK = selective ACK (selektivní potvrzování) je potvrzování segmentů i mimo pořadí pomocí volitelných položek záhlaví.

Pro každé spojení uživatel udržuje velikost MSS, posuvného okna, okna zahlcení a parametru SSTHRESH. SSTRESH se ukládá do směrovací tabulky a používá se jako výchozí hodnota pro nová spojení v témže směru.
Při ztrátě segmntu:

- po třetím stejném potvrzení se nastaví SSTHRESH na polovinu aktuálního okna zahlcení (minimálně 2xMSS), zopakuje se segment, nastaví se okno o něco výšší, než SSTHRESH a při opakovaných potvrzeních se zvyšuje o MSS
- po potvrzení ztraceného segmentu se nastaví okno na původní SSTHRESH a opět probíhá pomalé zvětšování
- po neobdržení potvrzení znovu pomalý start (okno = MSS, SSTRESH = 2xMSS)

3.5.3 User Datagram Protocol (UDP)

Poskytuje nespojovou nespolehlivou službu: nezaručuje se doručení ani znovuzasílání ztracených nebo poškozených dat. Vyšší výkon a rychlost než u TCP za cenu nespolehlivosti. Využití u streamování multimediálního obsahu.
Vlastní číslování portů (nezávisle na TCP). Velikost datagramu menží, než MTU aby se zamezilo fragmentaci. Oproti TCP může mýt příjemcem skupina uzlů (všesměrová či multicast=skupinová). Záhlaví obsahuje informaci o délce dat a kontrolní součet. Protože UDP je bezestavový a odesilatel nemusí vyžadovat odpověď, zdrojový port je volitelný. Pokud není použit, zdrojový port by měl být nastaven na nulu.
Kvůli chybějícím zárukám se UDP aplikace musí smířit s nějakými ztrátami, chybami nebo duplikacemi. Některé aplikace mohou podle potřeby přidávat jednoduchý mechanismus spolehlivosti do aplikační vrstvy. Aplikace používající UDP nejčastěji opravný mechanismus nepotřebují, a dokonce jím mohou být zdržovány. Pokud aplikace vyžaduje vysoký stupeň spolehlivosti, může se místo něj použít TCP nebo opravné kódy.
Protože UDP postrádá mechanizmus předcházení a regulace zahlcení sítě, je nutné nadbytečné UDP datagramy na routerech zahazovat. Ačkoliv je celkové množství UDP provozu na typické síti jen v řádu procent, je UDP používán řadou klíčových služeb včetně DNS, SNMP, DHCP a RIP (směrovací protokol).

Rozdíl mezi TCP a UDP

- TCP je spojově orientovaný protokol což znamená, že k navázání "end-to-end"komunikace potřebuje, aby proběhl mezi klientem a serverem tzv. "handshaking". Poté, co bylo spojení navázáno, data mohou být posílána oběma směry.
  Charakteristické vlastnosti TCP:
  - spolehlivost – TCP používá potvrzování o přijetí, opětovné posílání a překročení časového limitu. Pokud se jakákoliv data ztratí po cestě, server si je opětovně vyžádá. U TCP nejsou žádná ztracená data, jen pokud několikrát po sobě vyprší časový limit, tak je celé spojení ukončeno.
  - zachování pořadí – Pokud pakety dorazí ve špatném pořadí, TCP vrstva příjemce se postará o to, aby se některá data pozdržela a finálně je předala správně seřazená.
  - vyšší režie – TCP protokol potřebuje např. tři pakety pro otevření spojení, umožňuje to však zaručit spolehlivost celého spojení.
- UDP je jednodušší protokol založený na odesílání nezávislých zpráv.
  Charakteristika:
  - bez záruky – Protokol neumožňuje ověřit, jestli data došla zamýšlenému příjemci. Datagram se může po cestě ztratit. UDP nemá žádné potvrzování, přeposílání ani časové limity. V případě potřeby musí uvedené problémy řešit vyšší vrstva.
  - nezachovává pořadí – Při odeslání dvou zpráv jednomu příjemci nelze předvídat, v jakém pořadí budou doručeny.
  - jednoduchost – Nižší režie než u TCP (není zde řazení, žádné sledování spojení atd.).

3.5.4 Bezpečnost TCP a UDP

TCP Útoky: převzetí spojení (connection hijacking, autentizováno a dále nezabezpečeno), odepření služby (Denial of Service - vyčerpání zdrojů systému pro spojení), port scanning (zjišťování otevřených portů), aj.
Řešení: šifrování, vytvoření tunelů na jiných portech, omezení počtu spojení za daný čas, sledování skenování portů, aj.

UDP na směrovačích povolený porty 53/udp a 53/tcp pro DNS, vyplnění kontrolního součtu je nepovinné, jinak lze přepočítat.

firewall = filtrace packetů a segmentů/datagramů na základě TCP/UDP záhlaví.

překlad adres (NAT) překlad IP adresy z vnitřní sítě na IP adresu hraničního směrovače vnější sítě.
maškaráda = překlad adresy a zdrojového portu spojení/přenosu na zdrojový port nového spojení/-přenosu ze směrovače. Zasahuje i do aplikační vrstvy.

3.6 Systém DNS, překlad jména, protokol

3.6.1 Domain Name System (DNS)

hlavním úkolem a příčinou vzniku jsou vzájemné převody doménových jmen a IP adres uzlů sítě. Později ale přibral další funkce (např. pro elektronickou poštu či IP telefonii) a slouží dnes de fakto jako distribuovaná databáze síťových informací. Protokol používá porty TCP/53 i UDP/53. Servery DNS jsou organizovány hierarchicky, stejně jako jsou hierarchicky tvořeny názvy domén. Systém DNS umožňuje efektivně udržovat decentralizované databáze doménových jmen a jejich překlad na IP adresy. Stejně tak zajišťuje zpětný překlad IP adresy na doménové jméno.
Protože je používání názvů pro člověka daleko příjemnější než používání číselných adres, vznikla už v dobách ARPANETu potřeba takový převod realizovat. Původně byl na všechny počítače distribuován jediný soubor (v Unixu /etc/hosts). Tato koncepce přestala velmi rychle vyhovovat potřebám a především nárokům na rychlou aktualizaci. Přesto se tento soubor dodnes používá, v závislosti na konfiguraci

systému je možné jej použít buď prioritně před dotazem na DNS nebo v případě, že DNS neodpovídá. Je možné jej také použít k uložení vlastních přezdívek pro často navštěvované servery, případně také pro blokování reklam a podobně.
Prostor doménových jmen tvoří strom s jedním kořenem. Každý uzel tohoto stromu obsahuje informace o části jména (doméně), které je mu přiděleno a odkazy na své podřízené domény. Kořenem stromu je tzv. kořenová doména, která se zapisuje jako samotná tečka. Pod ní se v hierarchii nacházejí tzv. domény nejvyšší úrovně (Top-Level Domain, TLD). Ty jsou buď tematické (com pro komerci, edu pro vzdělávací instituce atd.) nebo státní (cz pro Českou republiku, sk pro Slovensko, jo pro Jordánsko atd.).
Strom lze administrativně rozdělit do zón, které spravují jednotliví správci (organizace nebo i soukromé osoby), přičemž taková zóna obsahuje autoritativní informace o spravovaných doménách. Tyto informace jsou poskytovány autoritativním DNS serverem.
Výhoda tohoto uspořádání spočívá v možnosti zónu rozdělit a správu její části svěřit někomu dalšímu. Nově vzniklá zóna se tak stane autoritativní pro přidělený jmenný prostor. Právě možnost delegování pravomocí a distribuovaná správa tvoří klíčové vlastnosti DNS a jsou velmi podstatné pro jeho úspěch. Ve vyšších patrech doménové hierarchie platí, že zóna typicky obsahuje jednu doménu. Koncové zóny přidělené organizacím připojeným k Internetu pak někdy obsahují několik domén – například doména kdesi.cz a její poddomény vyroba.kdesi.cz, marketing.kdesi.cz a obchod.kdesi.cz mohou být obsaženy v jedné zóně a obhospodařovány stejným serverem.

Složení doménového jména Celé jméno se skládá z několika částí oddělených tečkami. Na jeho konci se nacházejí domény nejobecnější, směrem doleva se postupně konkretizuje.

- část nejvíce vpravo je doména nejvyšší úrovně, např. wikipedia.org má TLD org.
- jednotlivé části (subdomény) mohou mít až 63 znaků a skládat se mohou až do celkové délky doménového jména 255 znaků. Doména může mít až 127 úrovní.

DNS servery DNS server může hrát vůči doméně (přesněji zóně, ale ve většině případů jsou tyto pojmy zaměnitelné) jednu ze dvou rolí:

- **Autoritativní server** je ten, na němž jsou trvale uloženy záznamy k dané doméně/zóně. Autoritativních serverů je obvykle více (minimálně dva - primární a sekundární, ale běžně i více) a až na velmi speciální případy se na všech udržují totožné záznamy, tzn. každou změnu v záznamech je potřeba propagovat na všechny autoritativní servery. Autoritativní DNS servery jsou obvykle provozovány registrátorem domény nebo poskytovatelem webhostingu.
- **Rekurzivní (caching only) server** je server, na který se se svými dotazy obracejí klientská zařízení (počítač, mobil aj.). Server pro ně příslušný záznam získá rekurzivními dotazy u autoritativních DNS serverů a po stanovenou dobu (definovanou pomocí parametru TTL) je má uloženy v cache, aby mohl odpovídat klientům rychleji a šetřil zatížení serverů autoritativních. Na tomto serveru nejsou žádné zóny uloženy trvale. Informaci o DNS serverech na dané síti klient zjišťuje nejčastěji přes protokol DHCP, na IPv6 přes NDP (DHCPv6 tuto informaci neposkytuje).

typy jmenných serverů:

- **primární** - jediný hlavní autoritativní server pro doménu/zónu, poskytuje autoritativní odpověď pro autoritativní záznamy ze své zóny a neautoritativní odpověď pro záznamy z cache
- **sekundární** - autoritativní pro svoji doménu/zónu, pravidelně kopíruje záznamy zóny (zone transfer) z primárního serveru, poskytuje stejné odpovědi jako primární server
- **caching only** - neautoritativní server pro doménu nebo zónu, poskytuje pouze neautoritativní odpovědi
- **kořenový** - primární server pro kořenovou doménu/zónu, je jich víc
- **forwarder** - server provádějící překlad pro jiný server (v roli klienta)

Root servery Kořenové jmenné servery (root name servers) představují zásadní část technické infrastruktury Internetu, na které závisí spolehlivost, správnost a bezpečnost operací na internetu. Tyto servery poskytují kořenový zónový soubor (root zone file) ostatním DNS serverům. Jsou součástí DNS, celosvětově distribuované databáze, která slouží k překladu unikátních doménových jmen na ostatní identifikátory. Kořenový zónový soubor popisuje, kde se nacházejí autoritativní servery pro domény nejvyšší úrovně. Tento kořenový zónový soubor je relativně velmi malý a často se nemění – operátoři root serverů ho pouze zpřístupňují, samotný soubor je vytvářen a měněn organizací IANA. Pojem root server je všeobecně používán pro 13 kořenových jmenných serverů. Root servery se nacházejí ve 34 zemích světa, na více než 80 místech. Root servery jsou spravovány organizacemi, které vybírá IANA.

3.6.2 Překlad jména

Každý koncový počítač má ve své konfiguraci síťových parametrů obsaženu i adresu lokálního DNS serveru, na nějž se má obracet s dotazy. V operačních systémech odvozených od Unixu je obsažena v souboru /etc/resolv.conf, v MS Windows ji najdete ve vlastnostech protokolu TCP/IP (případně můžete z příkazového řádku v XP zadat textový příkaz ipconfig /all). Adresu lokálního serveru počítač typicky obdrží prostřednictvím DHCP.
Pokud počítač hledá určitou informaci v DNS (např. IP adresu k danému jménu), obrátí se s dotazem na tento lokální server. Každý DNS server má ve své konfiguraci uvedeny IP adresy kořenových serverů (autoritativních serverů pro kořenovou doménu). Obrátí se tedy s dotazem na některý z nich.
Kořenové servery mají autoritativní informace o kořenové doméně. Konkrétně znají všechny existující domény nejvyšší úrovně a jejich autoritativní servery. Dotaz je tedy následně směrován na některý z autoritativních serverů domény nejvyšší úrovně, v níž se nachází cílové jméno. Ten je opět schopen poskytnout informace o své doméně a posunout řešení o jedno patro dolů v doménovém stromě. Tímto způsobem řešení postupuje po jednotlivých patrech doménové hierarchie směrem k cíli, až se dostane k serveru autoritativnímu pro hledané jméno, který pošle definitivní odpověď.
Získávání informací z takového systému probíhá rekurzí. Resolver (program zajišťující překlad) postupuje od kořene postupně stromem směrem dolů dokud nenalezne autoritativní záznam o hledané doméně. Jednotlivé DNS servery jej postupně odkazují na autoritativní DNS pro jednotlivé části jména.
Může se ale stát, že některý z oslovených serverů má hledanou informaci ve své vyrovnávací paměti, protože odpovídající dotaz nedávno řešil. V takovém případě poskytne neautoritativní odpověď z vyrovnávací paměti a další dotazování odpadá. Ve vyrovnávací paměti mohou být i mezivýsledky - například lokální DNS server v ní skoro jistě bude mít informaci o autoritativních serverech pro doménu org, protože v ní pravděpodobně hledá každou chvíli. Lokální server by se s dotazem rovnou obrátil na některý z autoritativních serverů domény org.

Dva odlišné druhy chování Při **rekurzivním řešení dotazu** se server chopí vyřízení dotazu, najde odpověď a pošle ji tazateli. Rekurzivní přístup server zatěžuje (musí sledovat postup řešení, ukládat si mezivýsledky apod.), ale projde jím odpověď a tu si může uložit do vyrovnávací paměti. Typicky se tak chovají lokální servery, aby si plnily vyrovnávací paměti a mohly dalším tazatelům poskytovat odpovědi rovnou. Při **nerekurzivním řešení dotazu** server dotaz neřeší, pouze poskytne tazateli adresy dalších serverů, jichž se má ptát dál. Takto se chovají servery ve vyšších patrech doménové hierarchie (kořenové a autoritativní servery TLD), které by rekurzivní chování kapacitně nezvládaly.

3.6.3 Reverzní dotazy

Nejběžnějším úkolem DNS je poskytnout informace (nejčastěji IP adresu) pro zadané doménové jméno. Dovede ale i opak – sdělit jméno, pod kterým je daná IP adresa zaregistrována. Při vkládání dat pro zpětné dotazy bylo ale třeba vyřešit problém s opačným uspořádáním IP adresy a doménového jména. Zatímco IP adresa má na začátku obecné informace (adresu sítě), které se směrem doprava zpřesňují až k adrese počítače, doménové jméno má pořadí přesně opačné. Instituce připojená k Internetu typicky má přidělen začátek svých IP adres a konec svých doménových jmen.
Tento nesoulad řeší DNS tak, že při reverzních dotazech obrací pořadí bajtů v adrese. K obrácené adrese pak připojí doménu in-addr.arpa a výsledné „jméno“ pak vyhledává standardním postupem. Hledáli například jméno k IP adrese 145.97.39.155, vytvoří dotaz na 155.39.97.145.in-addr.arpa. Obrácení IP

adresy umožňuje delegovat správu reverzních domén odpovídajících sítím a podsítím správcům dotyčných sítí a podsítí. V příkladu použitou síť 145.97.0.0/16 spravuje nizozemský SURFnet a ten má také ve správě jí odpovídající doménu 97.145.in-addr.arpa. Kdykoli zavede do sítě nový počítač, může zároveň upravit data v reverzní doméně, aby odpovídala skutečné situaci.
Je dobré mít na paměti, že na data z reverzních domén nelze zcela spoléhat. Do reverzní domény se v principu dají zapsat téměř libovolná jména. Nikdo například nemůže zabránit SURFnetu, aby o počítači 145.97.1.1 prohlásil v reverzní zóně, že se jedná třeba o www.seznam.cz. Pokud na tom záleží, je záhodno si poskytnutou informaci ověřit normálním dotazem (zde nalézt IP adresu k www.seznam.cz a porovnat ji s 145.97.1.1). Jestliže odpovědí na něj bude původní IP adresa, jsou data důvěryhodná – správce klasické i reverzní domény tvrdí totéž. Pokud se liší, znamená to, že data v reverzní doméně jsou nekorektní.

DNS v praxi Téměř každý DNS server funguje zároveň jako DNS cache. Při opakovaných dotazech pak nedochází k rekurzivnímu prohledávání stromu, ale odpověď je získána lokálně. V DNS záznamech je totiž uložena i informace jak dlouho lze záznam používat (TTL) a lze také zjistit, zda byl záznam změněn. Po vypršení platnosti je záznam z DNS cache odstraněn. Problém s uložením záznamů v cache nastává v případě změny záznamu. Pokud administrátor nastaví TTL na 6 hodin, a poté provede změnu záznamu, nastane situace, že někteří uživatelé sítě dostanou informaci již novou a někteří ještě starou. Tato situace bude trvat právě oněch 6 hodin, v závislosti na nastavení ostatních serverů a také v závislosti na době která uplynula od jejich posledního dotazu. Je proto nutné zvolit správný poměr mezi rychlostí šíření změn a ušetřeným výkonem a přenosovým pásmem DNS serveru. Pokud se změny provádí často, je vhodné zvolit kratší TTL v řádu jednotek hodin, pokud se změny téměř neprovádějí, může být TTL ve dnech.
Obsah zóny (domény či několika domén) je uložen v tzv. zónovém souboru. Skládá se z jednotlivých záznamů (přesný název zní zdrojové záznamy, resource records, RR) obsahujících dílčí informace. Formát textového zápisu zónového souboru se liší v závislosti na použitém serveru, zde použijeme nejrozšířenější BIND. Záznamy v něm mají tvar 'jméno životnost třída typ parametry'.
jméno je doménové jméno, pro něž záznam vytváříte. Zpravidla patří do aktuálně definované domény, pak se píše jen samotné jméno bez tečky na konci a bude k němu doplněna aktuální doména. Pokud jméno ukončíte tečkou, nic se k němu nedoplňuje a bere se jako kompletní. Nemusí být uvedeno, pak se přebírá z předchozího záznamu.
životnost určuje dobu platnosti záznamu v sekundách. Většinou se neuvádí a záznamům se ponechává implicitní životnost. Umožňuje vám však udělat výjimku.
třída určuje rodinu protokolů, k níž se záznam vztahuje. DNS lze teoreticky používat i pro jiné síťové architektury, v praxi však třída vždy bývá IN, což znamená Internet.
typ určuje typ definovaného záznamu (viz níže).
parametry se vztahují k typu záznamu, poskytují mu potřebná data. Obsahem parametrů často bývají doménová jména. Je třeba zdůraznit, že v parametrech se smí vyskytovat jen skutečná jména, nikoli přezdívky zavedené pomocí CNAME.
Zónový soubor vždy musí začínat záznamem typu SOA.

Typy záznamů:

- **A** (address record) obsahuje IPv4 adresu přiřazenou danému jménu, například když jménu cosi.kdesi.cz náleží IP adresa 1.2.3.4, bude zónový soubor pro doménu kdesi.cz obsahovat záznam **cosi IN A 1.2.3.4**
- **AAAA** (IPv6 address record) obsahuje IPv6 adresu. Zmíněnému stroji bychom IPv6 adresu 2001:718:1c01:1:02e0:7dff:fe96:daa8 přiřadili záznamem **cosi IN AAAA 2001:718:1c01:1:02e0:7dff:fe96:daa8**
- **CNAME** (canonical name record) je alias - jiné jméno pro jméno již zavedené. Typicky se používá pro servery známých služeb, jako je například WWW. Jeho definice pomocí přezdívky umožňuje jej později snadno přestěhovat na jiný počítač. Pokud náš cosi.kdesi.cz má sloužit zároveň jako www.kdesi.cz, vložíme do zónového souboru **www IN CNAME cosi**
- **NS** (name server record) ohlašuje jméno autoritativního DNS serveru pro danou doménu. Bude-li mít doména kdesi.cz poddoménu obchod.kdesi.cz, jejímiž servery budou ns.kdesi.cz (primární) a

ns.jinde.cz (sekundární), bude zónový soubor pro kdesi.cz obsahovat **obchod IN NS ns** a **IN NS ns.jinde.cz.**

- **PTR** je speciální typ záznamu pro reverzní zóny. Obsahuje na pravé straně jméno počítače přidělené adrese na straně levé (adresa je transformována na doménu výše popsaným postupem). Držme se našeho příkladu pro záznam typu A - v souladu s ním by zónový soubor pro doménu 3.2.1.in-addr.arpa měl obsahovat (zónový soubor definuje reverzní doménu, proto je třeba psát na pravé straně kompletní jméno s tečkou, jinak by za ně připojil reverzní doménu) **4 IN PTR cosi.kdesi.cz.**
- **SOA** (start of authority record) je zahajující záznam zónového souboru. Obsahuje jméno primárního serveru, adresu elektronické pošty jejího správce (zavináč je v ní ale nahrazen tečkou) a následující údaje:
  - Serial — sériové číslo, které je třeba zvětšit s každou změnou v záznamu. Podle něj sekundární server pozná, že v doméně došlo ke změně. Pokud jej zapomenete zvětšit, rozejde se obsah sekundárních serverů s primárním, což rozhodně není dobré. Pro přehlednost často ve formátu YYYYMMDDHH.
  - Refresh — jak často se má sekundární server dotazovat na novou verzi zóny (v sekundách).
  - Retry — v jakých intervalech má sekundární server opakovat své pokusy, pokud se mu nedaří spojit s primárním.
  - Expire — čas po kterém označí sekundární servery své záznamy za neaktuální, pokud se jim nedaří kontaktovat primární server.
  - TTL — implicitní doba platnosti záznamů.

  Časové údaje jsou v sekundách. Novější implementace umožní pro vyšší pohodlí používat k číslům přípony 'M','H','D' a 'W' (minuta, hodina, den, týden) – například 8h znamená 8 hodin, čili totéž co hodnota 28800.

3.6.4 Protokol DNS

Aplikační protokol pracující na bázi dotaz-odpověď poskytující službu typu klient-server (klient pošle dotaz, server odpoví). Základní operace **DNS Query** pro získání informací z DNS databáze na serveru. Používá TCP i UDP, pro oba port 53 (stejný pro dotaz i odpověď). Pro běžné dotazy typu překlad adres používá UDP (kvůli režii TCP), odpověď případně zkrácena na 512B kvůli fragmentaci. Kompletní odpověď nebo zone transfer pomocí TCP. Protokol není zcela spolehlivý (kvůli využívání UDP). Pro různé DNS operace různé DNS packety (neobsahují kontrolní součet - přenechán na UDP datagram).

DNS Query = základní operace protokolu DNS: dotaz (klienta) a odpověď (serveru) s informacemi (záznamy) podle požadavků v dotazu nebo negativní (záznam podle požadavků neexistuje). Stejný formát DNS packetu pro žádost i odpověď. Packet obsahuje záhlaví (povinné), dotazy, odpovědi, autoritativní jmenné servery a doplňující informace.

- **záhlaví (HEADER)** v dotazu i odpovědi
  - ID - itentifikátor stejný v dotazu i odpovědi pro spárování
  - QR - 0 pro dotaz, 1 pro odpověď
  - Opcode - typ dotazu
  - AA - 1 pro autoritativní odpověď
  - TC - pro zkrácenou odpověď (na 512B)
  - RD, RA - možnosti rekurzivního překladu
  - AD - zabezpečená odpověď
  - Rcode - kód odpovědi (0 bez chyby, 1 chyba foprmátu dotazu, 2 neschopnost odpovědi, 3 negativní odpověď, 5 odmítnutí odpovědi, atd.)

- **dotazy (QUESTION)** - většinou jeden záznam, v dotazu i odpovědi (zopakovaný)
- **ANSWER** - odpověď s požadovanými záznamy
- **AUTHORITY** - seznam autoritativních serverů
- **ADDITIONAL**

Komprese DNS packetu další výskyty (části) doménového jména v packetu jsou nahrazeny odkazem na první výskyt.

Inverzní dotaz (Opcode = 1): jako reverzní, ale pro odpověď se místo záznamu typu PTR používají záznamy typu A, nemusí být servery podporován.

3.6.5 Zabezpečení DNS

DNSsec = zabezpečení záznamů na jmenných serverech a v DNS packetech, použití el. podpisu (soukromým klíčem subdomény/zóny podepsány všechny její záznamy), RRSIG pro ověření integrity DNS packetu. Ověření el. podpisů (veřejný klíč DNSKEY, podepsaný soukromým klíčem nadřízené domény). Možnost uložit certifikáty pro aplikace jako záznam CERT. **Nevýhody:** Je třeba podepsat každý packet.

TSIG (Transaction Signatures) = autorizace komunikace DNS serverů. MD5 hash přenášených záznamů a sdíleného tajemství v záznamu typu TSIG. Použití u DNS Update - může jen autorizovaný server.

Registrace domény Internet Corporation for Assigned Names and Numbers (ICANN) je organizace, která má na starost přidělování a správu doménových jmen a IP adres. Zastřešuje také další regionální organizace, které působí na jednotlivých kontinentech. Každý stát má potom určeného správce zóny, který se stará o příslušnou TLD. Správce buď může domény registrovat sám nebo prostřednictvím tzv. doménových registrátorů, kteří mají náležitá oprávnění. ICANN zveřejňuje kompletní seznam všech TLD správců. Informace o vlastnících domén jsou udržovány v online databázi, která je dostupná přes službu WHOIS. Doménové registry obsahují informace o více než 240 národních a generických doménách (ccTLD, gTLD). Např. v ČR se o doménu .cz stará CZ.NIC z.s.p.o.. **Doména musí být volná**.

3.7 Služby WWW, elektronické pošty, přenosu souborů a vzdáleného přihlášení

3.7.1 DHCP

DHCP protokol umožňuje prostřednictvím DHCP serveru nastavovat stanicím v počítačové síti sadu parametrů nutných pro komunikaci pomocí IP protokolu (tj. využívat rodinu protokolů TCP/IP). Umožňuje předávat i doplňující a uživatelsky definované parametry.[1] Významným způsobem tak zjednodušuje a centralizuje správu počítačové sítě (například při přidávání nových stanic, hromadné změně parametrů[2] nebo pro skrytí technických detailů před uživateli). DHCP servery mohou být sdruženy do skupin, aby bylo přidělování adres odolné vůči výpadkům. Pokud klient některým parametrům nerozumí, ignoruje je.
Typicky se pomocí DHCP nastavují tyto parametry:

- IP adresa
- maska sítě
- implicitní brána (default gateway)
- DNS server
- další údaje např. pro NTP, WINS,...

Princip činnosti Klienti žádají server o IP adresu, ten u každého klienta eviduje půjčenou IP adresu a čas, do kdy ji klient smí používat (doba zapůjčení, anglicky lease time). Poté co vyprší, smí server adresu přidělovat jiným klientům.
Klient komunikuje na UDP portu 68, server naslouchá na UDP portu 67. Po připojení do sítě klient vyšle broadcastem DHCPDISCOVER paket. Na ten odpoví DHCP server paketem DHCPOFFER s nabídkou IP adresy. Klient si z (teoreticky několika) nabídek vybere jednu IP adresu a o tu požádá paketem DHCPREQUEST. Server mu ji vzápětí potvrdí odpovědí DHCPACK. Jakmile klient obdrží DHCPACK, může už IP adresu a zbylá nastavení používat. Klient musí před uplynutím doby zapůjčení z DHCPACK obnovit svou IP adresu. Pokud lhůta uplyne aniž by dostal nové potvrzení, klient musí IP adresu přestat používat.
Protokol definuje roli i tzv. DHCP relay agenta. Používá se v situaci, kdy existují dvě nebo více sítí oddělené směrovačem a jen jedna síť obsahuje DHCP server. V takovém případě správce na směrovači zapne relay agenta a nastaví jej tak, aby všesměrové (broadcast) DHCP dotazy ze sítí bez DHCP serveru přeposílal DHCP serveru. Agent k přeposílanému dotazu přidá číslo sítě a masku sítě, na které klienta zaslechl, aby DHCP server poznal, ze kterého adresního rozsahu má klientovi adresu přiřadit.
IP adresa může být stanici přidělena několika způsoby:

- Ruční nastavení - nevyužívá se DHCP serveru a konfigurace se zapisuje ručně do stanic
- Statická alokace - DHCP server obsahuje seznam MAC adres a k nim příslušným IP adres. Pokud je žádající stanice v seznamu, dostane vždy přidělenu stejnou pevně definovanou IP adresu
- Dynamická alokace - Správce sítě na DHCP serveru vymezí rozsah adres, které budou přidělovány stanicím, které nejsou registrovány. Časové omezení pronájmu IP adresy dovoluje DHCP serveru již nepoužívané adresy přidělovat jiným stanicím. Registrace dříve pronajatých IP adres umožňuje DHCP serveru při příštím pronájmu přidělit stejnou IP adresu.

3.7.2 Elektornická pošta

je způsob odesílání, doručování a přijímání zpráv přes elektronické komunikační systémy. Termín e-mail se používá jak pro internetový systém elektronické pošty založený na protokolu SMTP (Simple Mail Transfer Protocol), tak i pro intranetové systémy, které dovolují posílat si vzájemně zprávy uživatelům uvnitř jedné společnosti nebo organizace (tyto systémy často používají nestandardní protokoly, mívají ovšem bránu, která jim dovoluje posílat a přijímat e-maily z internetu). K širokému rozšíření e-mailu přispěl zejména internet. Pro e-mail je definován přenos 7bitové ASCII informace. Přesto je většina e-mailových přenosů 8bitových, kde ale nelze zaručit bezproblémovost. Z toho důvodu byla elektronická pošta rozšířena o standard MIME, aby bylo umožněno kódování vkládaných HTML a binárních příloh, obrázků, zvuků a videí.

Komunikační protokoly Mezi počítači na internetu se vyměňují zprávy pomocí Simple Mail Transfer Protocol a softwaru typu MTA jako např. Sendmail.
Uživatelé mívají na svém počítači nainstalován program, který se nazývá e-mailový klient. Ten stahuje zprávy z poštovního serveru použitím protokolů POP nebo IMAP, avšak v prostředí velkých společností se stále vyskytuje použití některého komerčního protokolu jako např. Lotus Notes nebo Microsoft Exchange Server.
Je možné ukládat e-maily buď na straně serveru, nebo na straně klienta. Standardní formáty pro mailové schránky jsou např. Maildir a mbox. Několik e-mailových klientů používá vlastní formát a na konverzi mezi těmito formáty je potřebný speciální program.
Někteří uživatelé nepoužívají e-mailového klienta, ale přistupují ke zprávám umístěným na poštovním serveru přes webové rozhraní. Tento postup se často používá zejména u freemailových (bezplatných) služeb.

Doručování Při posílání pošty přes internet má být zaručen spolehlivý přenos zprávy i v případě dočasného výpadku cílového serveru. Zpráva se obvykle píše v prostředí programu typu e-mailového klienta nebo v obdobném formuláři webového rozhraní. Klient pomocí Simple Mail Transfer Protocol (SMTP) pošle zprávu programu Mail Transfer Agent (MTA), například smtp.a.org, který může běžet buď na samostatném smtp poštovním serveru, nebo i přímo na počítači odesílatele. Program MTA zjistí

z uvedených cílových adres název domény (část adresy za zavináčem) a tyto domény vyhledá v Domain Name System (DNS), aby zjistil mail exchange servery přijímající poštu pro danou doménu. DNS server domény b.org, tedy ns.b.org, odpoví MX záznamem, kde uvede mail exchange server pro danou doménu. MTA server (např. smtp.a.org) odešle zprávu na mail exchange server (např. mx.b.org) pomocí protokolu SMTP. Domény obvykle mají záložní (backup) mail exchange server, takže můžou pokračovat v přijímaní pošty, i když je právě nedostupný hlavní mail exchange server. Když není možné zprávu doručit, MTA příjemce o tom musí odeslat zpět odesílateli zprávu (bounce message), ve které ukazuje na problém. Mail exchange server zprávu doručí do schránky adresáta. Ze schránky adresáta si zprávu stáhne pomocí protokolu POP (POP3) nebo IMAP nebo ji adresátovi umožní prohlédnout poštovní klient příjemce nebo webová služba. Bývalo zvykem, že kterýkoliv MTA přijímal zprávy pro kteréhokoli uživatele na internetu a udělal, co se dalo, aby zprávu doručil. Takové MTA se nazývají open mail relays. To bylo důležité v začátcích internetu, když byla síťová spojení nespolehlivá a nepermanentní. Když MTA nemohl doručit zprávu do cíle, mohl ji alespoň poslat agentovi bližšímu k cíli. Ten by měl větší šanci ji doručit. Ukázalo se však, že tento mechanizmus byl zneužitelný lidmi posílající nevyžádanou hromadnou poštu (spam), a proto velmi málo ze současných MTA jsou open mail relays (tj. přijímají poštu pro známé uživatele resp. domény). Prakticky všechny open relays jsou rychle odhalené a zneužité spamery, kteří neustále prohledávají (skenují) IP rozsahy celého internetu.

E-mailová adresa Každý uživatel musí mít pro příjem zpráv svoji e-mailovou adresu, která identifikuje jeho elektronickou poštovní schránku. Ta je fyzicky umístěna na nějakém internetovém serveru, populární jsou zejména servery, které nabízejí e-mailovou schránku zdarma a s webovým rozhraním (např. GMail, Seznam.cz).

Hlavička e-mailové zprávy obvykle obsahuje alespoň 4 pole:

- Od (From): e-mailová adresa (popř. i jméno) odesílatele zprávy (zpravidla vyplňuje program automaticky)
- Komu (To): e-mailová adresa (popř. i jméno) příjemce zprávy, adresátů může být více současně (vyplňuje odesílatel)
- Předmět (Subject): stručný popis obsahu zprávy (vyplňuje odesílatel, nepovinně)
- Datum (Date): místní datum a čas odeslání zprávy (vyplňuje program automaticky)

Informace v hlavičce na počítači příjemce je podobná záhlaví na konvenčním dopisu – skutečná informace o tom, komu byla zpráva adresována, je odstraněná mailovým serverem potom, co je přiřazená správné mailové schránce. Pole „Od“ nemusí obsahovat adresu skutečného odesílatele. Je velmi jednoduché to zfalšovat a zpráva potom vypadá, že přišla z uvedené adresy. Je možné e-mail digitálně podepsat, aby bylo jisté, od koho zpráva pochází.
Další běžné součásti hlavičky:

- Kopie (Cc): carbon copy – kopie (carbon proto, že psací stroje používají „kopírák“ (carbon paper) k vytvoření kopií dopisů) (vyplňuje odesílatel, nepovinná položka)
- Slepá čili skrytá kopie (Bcc): blind carbon copy – neviditelná kopie (adresát bude vidět osoby uvedené v poli „Komu“ a „Cc“, ale ne adresy v „Bcc“) (vyplňuje odesílatel, nepovinná položka)
- Received: přijato – trasové informace vytvořené servery, kterými zpráva prošla (automatické zápisy serverů)
- Content-type: druh obsahu – informace o tom, jak má být zpráva zobrazena, obvykle MIME typ (automatický zápis)

MIME , plným názvem Multipurpose Internet Mail Extensions („Víceúčelová rozšíření internetové pošty“), je internetový standard, který umožňuje pomocí elektronické pošty zasílat zprávy obsahující text s diakritikou, lze k ní přiložit přílohu v nejrůznějších formátech, umožňuje funkci digitálního podpisu apod.

SMTP je internetový protokol určený pro přenos zpráv elektronické pošty (e-mailů) mezi přepravci elektronické pošty (MTA). Protokol zajišťuje doručení pošty pomocí přímého spojení mezi odesílatelem a adresátem; zpráva je doručena do tzv. poštovní schránky adresáta, ke které potom může uživatel kdykoli (off-line) přistupovat (vybírat zprávy) pomocí protokolů POP3 nebo IMAP. Jedná se o jednu z nejstarších aplikací. SMTP funguje nad protokolem TCP, používá port TCP/25.

POP3 je internetový protokol, který se používá pro stahování emailových zpráv ze vzdáleného serveru na klienta. Jedná se o aplikační protokol pracující přes TCP/IP připojení. V současné době používají téměř všichni uživatelé elektronické pošty pro stahování emailů programy využívající POP3 nebo IMAP. Ze vzdáleného serveru se stáhnou všechny zprávy, třeba i ty, které uživatel číst nechce, nebo spam (pokud ho již nefiltruje poštovní server). Většina POP3 serverů sice umožňuje stáhnout i pouze hlavičky zpráv (a následně vybrat zprávy, které se stáhnou celé), ale podpora v klientech vesměs chybí. Tuto nevýhodu může odstranit protokol IMAP, který pracuje se zprávami přímo na serveru. Pro odesílání zpráv se používá protokol SMTP, nezávisle na použitém protokolu pro příjem pošty.
Protokol POP3 má pro své účely vyhrazen TCP port 110. Komunikace probíhá na principu výměny zpráv mezi klientem a serverem. Příkaz vždy začíná na začátku řádky, v základní implementaci POP3 mají příkazy 3 nebo 4 znaky. Příkazy nerozlišují velká a malá písmena. Za příkazem můžou následovat další argumenty, oddělené mezerami. Řádky jsou oddělovány pomocí CRLF. Každá odpověď od serveru musí začínat indikací stavu operace - buď +OK, nebo -ERR. Následovat může textový řetězec s popsaným důvodem stavu. POP3 implementace jsou často poměrně komunikativní a dají se užívat i „ručně“.

IMAP je internetový protokol pro vzdálený přístup k e-mailové schránce prostřednictvím e-mailového klienta. IMAP nabízí oproti jednodušší alternativě POP3 pokročilé možnosti vzdálené správy (práce se složkami a přesouvání zpráv mezi nimi, prohledávání na straně serveru a podobně) a práci v tzv. on-line i off-line režimu. Protokol IMAP umožňuje trvalé (tzv. on-line) připojení k e-mailové schránce. Díky tomu je možné s celou poštovní schránkou plně pracovat z libovolného místa. Všechny zprávy a složky jsou uloženy na poštovním serveru a na počítač se stahují jen nezbytné informace, takže při zobrazení složky se stáhnou jen záhlaví zpráv a jejich obsah až v případě, že zprávu chce uživatel přečíst. U jednotlivých zpráv se uchovává jejich stav (nepřečtená, odpovězená, důležitá), uživatel může zprávy přesouvat mezi složkami, složky vytvářet, mazat, prohledávat na straně serveru apod. Protokol umožňuje současné připojení více klientů zároveň.
Oproti protokolu POP3 je IMAP4 velmi komplikovaný protokol. Jeho implementace je značně složitější a tedy i náchylnější k chybám než implementace POP3. Navzdory tomu IMAP používá mnoho e-mailových serverů a klientů jako jejich standardní přístupovou metodu.

Poštovní klient je program, který zajišťuje odesílání zpráv a vybírání schránek. Příkladem je např. Microsoft Outlook, Mozilla Thunderbird, Opera, Mutt, Pine a další. Je to v podstatě specializovaný editor, který umí kromě vytvoření zprávy také manipulovat se schránkami, odeslat zprávu nejbližšímu MTA a převzít zprávu ze serveru prostřednictvím POP3 nebo IMAP. Vlastním doručováním zprávy po síti až k adresátovi se klient nezabývá. Součástí klienta bývá také více či méně složitý adresář, který pomáhá uživateli udržet přehled o adresách.

Poštovní server (MTA) běží obvykle jako démon či Služba Windows a naslouchá na portu TCP/25. K tomuto portu se může připojit (navázat TCP spojení) buď poštovní klient, nebo jiný server, který předá zprávu k doručení. MTA zkontroluje, zda je zpráva určena pro systém, na kterém běží. Pokud ano, předá ji programu MDA (lokální doručení). Pokud je zpráva určena jinému počítači, naváže spojení s příslušným serverem a zprávu mu předá. Při vyhledávání vzdáleného serveru, kterému má předat zprávu, musí MTA spolupracovat se systémem DNS. Od serveru DNS si vyžádá tzv. MX záznam pro cílovou doménu, který obsahuje IP adresu počítače, který se stará o doručení pošty v této doméně. Pokud DNS tento záznam neobsahuje, pokusí se poštovní server doručit zprávu přímo na počítač uvedený v adrese za zavináčem. Poštovní server obsahuje v konfiguraci řadu parametrů, pomocí kterých můžeme mimo jiné nastavit, pro které domény MTA přijímá zprávy. Stejně tak je možné určit, od koho bude nebo nebude zprávy přijímat, což je velmi důležité z hlediska bezpečnosti a ochrany proti spamu.