Univerzita Pardubice

Fakulta elektrotechniky a informatiky

# Tvorba materiálů pro wiki projekt základů počítačových sítí

Aneta Malířová

Bakalářská práce
2014

**Univerzita Pardubice**
**Fakulta elektrotechniky a informatiky**
Akademický rok: **2013/2014**

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

| | |
|---|---|
| Jméno a příjmení: | **Aneta Malířová** |
| Osobní číslo: | **I11124** |
| Studijní program: | **B2646 Informační technologie** |
| Studijní obor: | **Informační technologie** |
| Název tématu: | **Tvorba materiálů pro wiki projekt základů počítačových sítí** |
| Zadávající katedra: | **Katedra informačních technologií** |

Zásady pro vypracování:

Cílem práce je vytvořit materiál pro wiki projekt počítačových sítí na upce.cz. Autor práce připraví přehledný materiál, který bude integrován do projektu wiki základů počítačových sítí na upce.cz. Materiál bude zaměřen na výklad principů datových sítí založených na architektuře TCP/IP a principů přepínaných sítí. Práce bude doplněna o informace z analyzátoru wireshark a praktické řešení laboratorní úlohy.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

**SANDERS, Chris. Analýza sítí a řešení problémů v programu Wireshark. 1. vyd. Brno: Computer Press, 2012, 288 s. ISBN 978-80-251-3718-5.**
**LAMMLE, Todd. CCNA: výukový průvodce přípravou na zkoušku 640-802. Vyd. 1. Brno: Computer Press, 2010, 928 s. ISBN 978-802-5123-591.**
**EMPSON, Scott. CCNA kompletní přehled příkazů: autorizovaný výukový průvodce. Vyd. 1. Brno: Computer Press, 2009, 336 s. ISBN 978-80-251-2286-0.**
**ODOM, Wendell, Rus HEALY a Naren MEHTA. Směrování a přepínání sítí: autorizovaný výukový průvodce. Vyd. 1. Brno: Computer Press, 2009, 879 s. ISBN 978-80-251-2520-5.**

Vedoucí bakalářské práce: **Mgr. Josef Horálek**
Katedra softwarových technologií

Datum zadání bakalářské práce: **20. prosince 2013**
Termín odevzdání bakalářské práce: **9. května 2014**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2014

**Prohlášení autora**

Prohlašuji, že jsem tuto práci vypracovala samostatně. Veškeré literární prameny a informace, které jsem v práci využila, jsou uvedeny v seznamu použité literatury.

Byla jsem seznámena s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 9. 5. 2014 Aneta Malířová

**Poděkování**

Mé poděkování patří vedoucímu práce panu Mgr. Josefu Horálkovi, Ph.D. za odbornou pomoc, cenné rady, trpělivost, věnovaný čas a poskytnuté materiály nejen při zpracování bakalářské této práce, ale v průběhu celého studia. Dále bych chtěla poděkovat své rodině a přátelům za jejich psychickou podporu.

**Anotace**

Tato bakalářská práce je vytvořena jako materiál pro wiki projekt počítačových sítí na upce.cz, do kterého bude integrována. Materiál je zaměřen na výklad principů datových sítí založených na architektuře TCP/IP a principů přepínaných sítí. Práce je doplněna o informace z analyzátoru Wireshark a praktické řešení laboratorní úlohy.

**Klíčová slova**

síť, komunikace, model, protokol, datová jednotka, přepínání

**Title**

Creation of material for the wiki project about networking basics.

**Annotation**

This bachelor's thesis is created as material for the wiki project about networking basics on the website upce.cz to which will be integrated. The material is focused on the interpretation of the principles of data networks based on the architecture of TCP/IP and principles switched networks. The work is supplemented by information from the analyzer Wireshark and practical solutions for laboratory tasks.

**Keywords**

network, communication, model, protocol, data unit, switching

# Obsah

# Seznam zkratek

| | |
|---|---|
| PDA | Personal Digital Assistant |
| VoIP | Voice over Internet Protocol |
| QoS | Quality of Service |
| ISO | International Organization for Standardization |
| IEEE | Institute of Electrical Enginners |
| IETF | Internet Engineering Task Force |
| RFC | Request For Comments |
| LAN | Local Area Network |
| WLAN | Wireless Local Area Network |
| WAN | Wide Area Network |
| MAN | Metropolitan Area Network |
| PAN | Personal Area Network |
| SAN | Storage Area Network |
| PDU | Protocol Data Unit |
| ISO/OSI | International Organization for Standardization/Open Systém Interconnection |
| TCP/IP | Transmission Control Protocol/Internet Protocol |
| TCP | Transmission Control Protocol |
| HTTP | Hypertext Transfer Protocol |
| URL | Uniform Resource Locator |
| HTML | Hypertext Markup Language |
| HTTPS | Hypertext Transfer Protocol Secure |
| SSL | Secure Socket Layer |
| SMTP | Simple Mail Trasnfer Protocol |
| POP | Post Office Protocol |
| IMAP | Internet Message Protocol |
| MUA | Mail User Agent |
| MTA | Mail Transfer Agent |
| DNS | Domain Name System |
| MDA | Mail Delivery Agent |
| IP | Internet Protocol |
| TLD | Top Level Domain |
| SLD | Second Level Domain |
| IANA | Internet Assigned Numbers Authority |
| ICANN | Internet Corporation for Assigned Names and Numbers |
| ccTLD | Country Code Top Level Domain |
| gTLD | Generic Top Level Domain |
| sTLD | Sponsored Top Level Domain |
| BIND | Berkely Internet Name Domain |
| Telnet | Teletype Network Service |
| SSH | Secure Shell |
| DHCP | Dynamic Host Configuration Protocol |

| | |
|---|---|
| UDP | User Datagram Protocol |
| MAC | Media Access Control |
| FTP | File Transfer Protocol |
| TFTP | Trivial File Transfer Protocol |
| IPv4 | Internet Protocol version 4 |
| IPv6 | Internet Protocol version 6 |
| MTU | Maximum Transfer Unit |
| IHL | Internet Header Length |
| TTL | Time-To-Live |
| RIR | Regional Internet Registry |
| LIR | Local Internet Registry |
| ISP | Internet Service Provider |
| NAT | Network Address Translator |
| CIDR | Classless Inter-Domain Routing |
| ARP | Address Resolution Protocol |
| ICMP | Internet Control Message Protocol |
| ICMPv4 | Internet Control Message Protocol version 4 |
| ICMPv6 | Internet Control Message Protocol version 6 |
| SLAAC | Stateless Address Autoconfiguration |
| EUI-64 | Extended Unique Identifier-64 |
| RIP | Routing Information Protocol |
| RIPv2 | Routing Information Protocol version 2 |
| EIGRP | Enhanced Interior Gateway Routing Protocol |
| OSPF | Open Shortest Path First |
| BGP | Border Gateway Protocol |
| IS-IS | Intermediate System to Intermediate System |
| OUI | Organizationally Unique Identifier |
| VLSM | Variable-Length Subnet Masking |
| LLC | Logical Link Control |
| CSMA/CD | Carrier Multiple Access with Collision Detection |
| CSMA/CA | Carrier Multiple Access with Collision Avoidance |
| ITU | International Telecommunication Union |
| ANSI | American National Standards Institute |
| VLAN | Virtual Local Area Network |
| PPP | Point-to-Point Protocol |
| NCP | Network Control Protocol |
| ROM | Read Only Memory |
| ACL | Access Control listopad |
| VTP | Virtual Trunking Protocol |
| NVRAM | Non-Volatile Random Access Memory |
| STP | Spanning Tree Protocol |
| STA | Spanning Tree Algorithm |
| BID | Bridge ID |
| BPDU | Bridge Protocol Data Units |

# Seznam obrázků

# Seznam tabulek

# Úvod

Dnešní svět je doslova obklopen informačními technologiemi, které zasahují téměř do všech sfér každodenního života. Absence počítačů, chytrých telefonů a Internetu je tedy pro většinu lidí nepředstavitelná. S rozvojem informačních technologií roste i zájem lidí o tuto problematiku. Zejména se jedná o mladé zvídavé studenty. Z tohoto důvodu je třeba, aby studenti měli dostatečné množství aktuálních, dostupných a věrohodných materiálů, ze kterých mohou čerpat nové vědomosti. Výše uvedené potřeby vedly ke vzniku této práce, jejímž cílem je studentům, a nejen jim, přiblížit problematiku moderních technologií, konkrétně z oblasti počítačových sítí. Tato práce bude postupně integrována do projektu wiki na stránkách upce.cz, který bude sloužit jako materiál pro budoucí „síťaře“.

Tyto materiály jsou zaměřeny na výklad základních principů datových sítí, zejména vrstvových modelů, a také principů přepínaných sítí. Celá práce je rozdělena do tří základních částí, které shrnují informace nutné pro absolvování kurzu CCNA1 a přiblížení principů kurzu CCNA3.

V první části je zaveden pojem počítačová síť, dále jsou představeny její komponenty a základní dělení.

Druhou, velice rozsáhlou částí, jsou vrstvové modely, ve které jsou konkrétně představeny a porovnány dva základní modely – ISO/OSI a TCP/IP. Důležitým prvkem je však architektura TCP/IP, ze které vychází moderní počítačové sítě. Je tedy představena funkčnost jednotlivých vrstev této architektury a konkrétní protokoly pracující na jejích vrstvách. Z důvodu významného rozvoje protokolu IPv6 je převážná část věnována problematice internetových protokolů síťové vrstvy, kde je vysvětlen důvod vzniku IPv6 a změny oproti IPv4. Nedílnou součástí jsou také kapitoly týkající se metod vytváření podsítí a převodů mezi číselnými soustavami, které jsou obohaceny o praktické příklady. Kromě síťových protokolů je také část věnována rozdílům ve strukturách jednotlivých ethernetových rámců. Nechybí ani odchycení některých těchto rámců analyzátorem Wireshark.

Třetí, zároveň poslední částí, jsou kapitoly týkající se problematiky kurzu CCNA3, tedy přepínaných sítí. V této části jsou představeny základní pojmy, zejména switch a VLAN, dále je popsána struktura třívrstvého hierarchického modelu, důležité protokoly a je vysvětlen princip komunikace v rámci různých VLAN. V jednotlivých kapitolách jsou uvedeny důležité

příkazy, jež jsou spojeny s tématem. Na závěr je uvedena topologie, na které je vysvětlen postup konfigurace vlastní přepínané sítě.

# 1 Seznamujeme se s počítačovou sítí

Současný svět je obklopen moderními technologiemi, jimiž jsou lidé doprovázeni téměř na každém kroku. Rychlá komunikace se v této hektické době stala doslova nutností. Díky Internetu mají lidé přístup k nespočetnému množství informací a služeb. Těžko si představit fungování, nejen v pracovní sféře, bez emailových zpráv, telefonních hovorů a dalších forem komunikace.

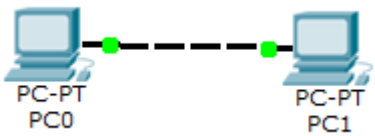
Pro běžného uživatele znamená poslání emailové zprávy pouze napsání textu a kliknutí na jedno tlačítko. Vše však funguje díky počítačové síti, jejíž existenci si ani odesílatel nemusí uvědomovat, případně má jen částečné představy o tom, jak daný mechanismus funguje.

Tato kapitola je proto věnována základním informacím o počítačové síti. Po přečtení této kapitoly by měl být čtenář schopen definovat tento pojem, mít představu o základních komponentách a dělení sítí.

## 1.1 Definice počítačové sítě

Je zřejmé, že se představy o počítačové síti budou u různých lidí lišit. Z pohledu běžného uživatele byla již představena, ale jak přesně tento pojem definovat?

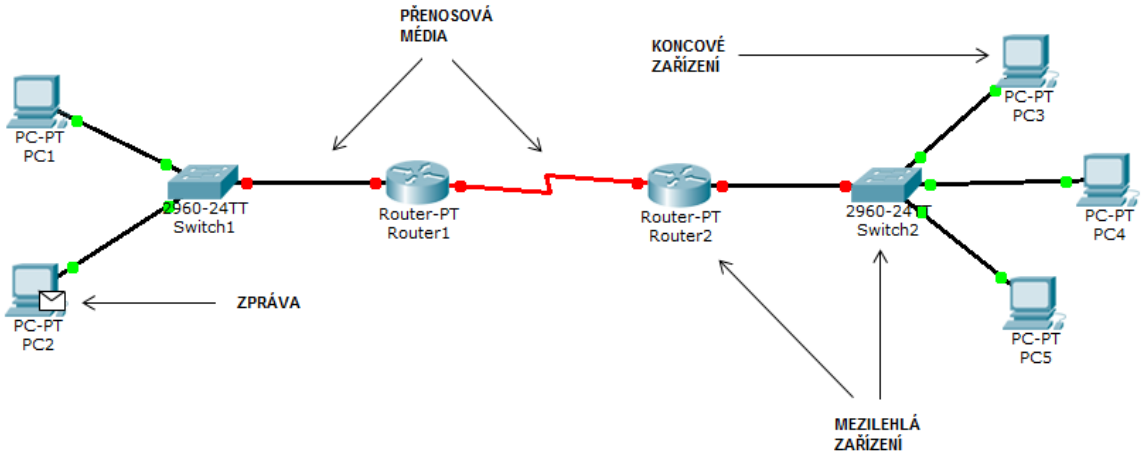
**Počítačová síť** vzniká propojením několika zařízení, čímž je umožněna jejich vzájemná komunikace. Jedná se o výpočetní zařízení, která jsou spojena pomocí specifických médií. Těchto zařízení může být v síti nespočetné množství, minimální počet je však dvě. Nejmenší počítačovou síť si proto můžeme představit jako propojení dvou počítačů (Obrázek 1). [23]




**Obrázek 1 – Příklad nejmenší počítačové sítě**

## 1.2 Základní komponenty

Pojem počítačová síť byl již přiblížen. Na základě procesů neboli funkcí jsou uživatelům poskytovány různé služby. Nyní budou představeny její jednotlivé součásti.




**Obrázek 2 – Komponenty sítě**

### 1.2.1 Zprávy

Veškerá komunikace po síti probíhá formou **zpráv**, které jsou posílány z jednoho zařízení do druhého. Tyto zprávy si lze představit jako posloupnost binárních čísel neboli nul a jedniček. Skládají se z dat přenášených sítí a dále doplňujících informací potřebných pro průchod touto sítí. Mezi uvedené informace patří například zdrojová a cílová adresa komunikujících zařízení nebo číslo, které udává pořadí zprávy. Struktura zprávy se během průchodu sítí mění. Jsou postupně přidávány potřebné informace ve formě záhlaví, případně zápatí. Tato problematika však bude podrobněji probrána v kapitole týkající se vrstvových protokolů.

### 1.2.2 Zařízení

Aby tato data mohla být vůbec sítí přeposílána, je třeba, aby existovali nějací iniciátoři komunikace, kteří jsou nazýváni jako **koncová zařízení** neboli **hosté.** Jedná se o zařízení, která jsou zdrojem nebo příjemcem dané zprávy, což znamená, že se přímo podílí na komunikaci. Patří mezi ně nejen osobní počítače, notebooky, servery, tablety, chytré telefony, tiskárny, scannery, PDA, ale také VoIP telefony, scannery čárových kódů, čtečky karet a mnoho dalších.

Tato zařízení tvoří rozhraní mezi uživatelem a počítačovou sítí. To znamená, že veškeré procesy uvnitř sítě jsou pro uživatele skryty. Jak již bylo řečeno, odesílatele emailu nezajímá, jakým způsobem jeho zpráva doputuje k adresátovi, zajímá ho jen, zda byla doručena.

Každý host má v rámci sítě nějakou roli, která je určena integrovaným softwarem. Mezi tyto role patří:

- server,
- klient,
- klient-server.

**Server** slouží pro klienty jako zdroj informací, poskytuje jim určité služby. Na základě požadavku klienta odesílá požadovaná data.

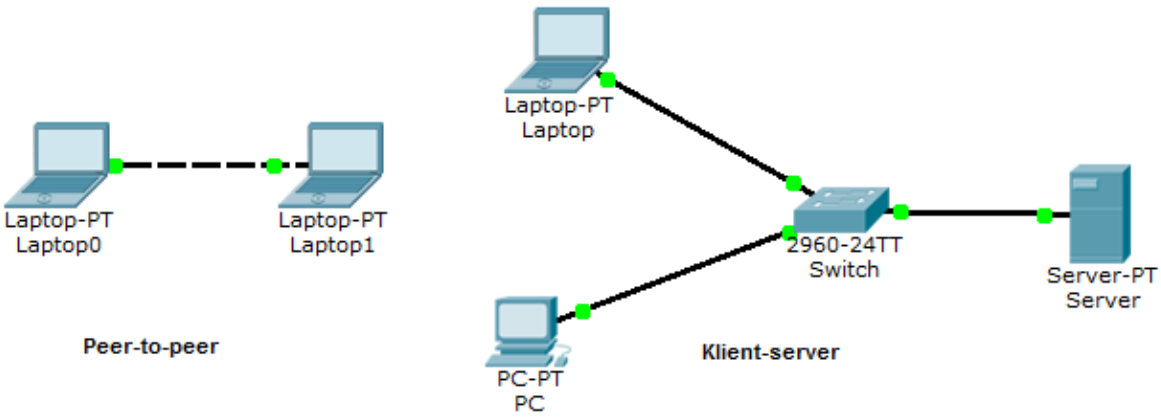
Existují různé typy serverů. Mezi nejčastěji využívané patří webový, emailový či souborový server, díky nimž je možné si prohlížet webové stránky, číst emailové zprávy nebo spravovat soubory.

Není pravidlem, že jeden server může poskytovat v danou chvíli služby pouze jednomu klientovi, stejně tak, že jeden klient nemůže být připojen v danou chvíli k více serverům. Právě naopak. Uživatel běžně současně prohlíží více webových stránek, v ten samý okamžik čte emaily a například tiskne nějaké dokumenty. To znamená, že v danou chvíli využívá služeb více serverů. Podobně server může vyřizovat více žádostí najednou od několika klientů.

Jak již bylo uvedeno, **klient** využívá služeb serveru za účelem získání požadovaných informací. Příkladem klienta může být webový prohlížeč, který na základě informací získaných od serveru umožní zobrazit požadovanou webovou stránku.

Další zmíněnou rolí je **klient-server**. Jedná se zařízení, která se mohou chovat nejen jako server, ale i klient. Nejčastěji se vyskytují v síťové architektuře typu **peer-to-peer**, která je tvořena nejméně dvěma zařízeními typu klient-server, aniž by byla připojena k nějakému centrálnímu serveru. Tato zařízení poté mohou sama poskytovat služby ostatním zařízením v síti a zároveň jejich služby využívat. Pokud daný host v jednu chvíli funguje jako klient i server zároveň, může to způsobit snížení výkonu.

Opakem architektury typu peer-to-peer je architektura **klient-server**, která je tvořena centrálním serverem a hosty typu klient, kteří jsou k tomuto centrálnímu serveru připojeni a posílají mu požadavky na zaslání daných informací. Pozor neplést si architekturu klient-server a roli zařízení klient-server. [23], [26]




**Obrázek 3 – Peer-to-peer vs. Klient-server**

Kromě hostů jsou součástí sítě i **mezilehlá zařízení**. Opět jde o ty části sítě, díky kterým je umožněn průchod dat, ale liší se tím, že nejsou iniciátory komunikace. Jejich úkolem je řídit, jakým způsobem data sítí poputují.

Na základě cílové adresy uvedené ve zprávě vybírají cestu, kterou budou data poslána, případně pokud daná cesta selže, hledají alternativní cestu. Během posílání zprávy může dojít k jejímu poškození. V takovém případě bývá zpráva těmito zařízeními zahozena a poslána nová. V okamžiku výskytu jakékoliv chyby je jejich úkolem informovat ostatní zařízení v síti o této skutečnosti. Jelikož sítí neprochází v danou chvíli pouze jedna zpráva, je třeba nějakým způsobem řídit tok dat a tím zabránit zahlcení sítě. K tomuto účelu se využívá QoS (Quality of Service), kdy jsou zprávy zpracovávány na základě priorit. Mezilehlá zařízení poskytují také bezpečnostní mechanismy, které umožňují zakazovat nebo povolovat tok dat. [10], [23]

Mezilehlá zařízení lze rozdělit dle jejich účelu do jednotlivých kategorií [10]:

- **Přístupová zařízení**, která umožňují uživatelům přístup k síti. Patří mezi ně například switch nebo hub.
- **Zařízení propojující sítě.** Jak plyne z názvu, jde o zařízení, která propojují jednotlivé sítě navzájem. Příkladem je router, modem nebo komunikační server. Propojením daných sítí vzniká „internetwork".
- **Bezpečnostní zařízení** udávající pravidla, na základě kterých je povolován nebo zakazován tok dat. Patří sem například firewall.

### 1.2.3 Přenosová média

Jako součásti sítě byly již uvedeny zprávy, koncová a mezilehlá zařízení. Neméně důležitá jsou i přenosová média, která slouží k propojení již zmíněných zařízení.

Tato média lze rozdělit na základě použitého typu signálu [26]:

- **Metalická** – data přenášena pomocí elektrických pulsů.
- **Optická** – přenos dat pomocí světelných pulsů.
- **Bezdrátová** – přenos dat pomocí elektromagnetických signálů.

Každý typ média má své výhody a nevýhody. Při volbě vhodného média jsou zohledňovány některé aspekty. Pro představu lze zmínit například vzdálenost, na kterou budou data posílána, stejně tak i prostředí, kterým daná síť povede. Neméně důležitým kritériem je množství dat, které má být sítí přeposíláno a také požadovaná rychlost. Při zvažování těchto požadavků je důležité také nezapomenout na cenu daného média. Je třeba zvážit, zda se za dané finanční prostředky zvolené médium vůbec vyplatí a zda by nebylo výhodnější zvolit například pomalejší variantu, ale za nižší cenu. [10]

### 1.2.4 Protokoly

Počítačová síť se fyzicky skládá z již uvedených komponent. Pro úspěšnou komunikaci to však není dostačující. Další důležitou částí sítě jsou totiž protokoly, případně sada protokolů (vzájemně kooperující protokoly).

Jedná se o standardizovaná pravidla, dle kterých probíhá komunikace. Jelikož jednotlivé složky sítě mohou být založeny na různých platformách, nemusely by si bez těchto protokolů jednotlivé komponenty sítě „rozumět" a tím by selhala komunikace.

Je to podobné jako v dopravě. Pokud by pro každého měla dopravní značka jiný význam, vznikal by na silnici chaos. Dalším příkladem může být mezilidská komunikace. Každý jazyk má definované určité normy, kterými se řídí. V případě, že by se němý Jarda, jenž je schopen komunikovat pouze pomocí znakové řeči, setkal s Pepou, který znakovou řeč neumí, nastal by problém. Řešením by bylo, aby se Pepa znakovou řeč doučil nebo měl k dispozici někoho, kdo je schopen ji přeložit do jemu známého jazyka. Vždy je potřebné, aby si oba komunikující systémy, v tomto případě Jarda s Pepou, rozuměli, respektive znali stejná pravidla komunikace. Proto je existence protokolů tak důležitá.

V předchozím textu byly představeny zprávy jako prostředky pro přenos dat sítí, jejichž struktura se postupně mění. Jak daná zpráva vypadá a jak se mění, je dáno právě protokolem. Stejně tak byla představena zařízení, která požadované zprávy přeposílala z jednoho zařízení na druhé, než dosáhly cíle. To, jakým způsobem spolu daná zařízení zahajují komunikaci, jak si předávají informace o chybových stavech či změnách v síti, je opět dáno určitým protokolem.

Tyto standardy jsou vždy spravovány konkrétními institucemi, mezi které patří zejména International Organization for Standardization (ISO), dále Institute of Electrical and Electronics Engineers (IEEE) a Internet Engineering Task Force (IETF). Standardizace mnoha protokolů je uvedena v RFC (Request for Comments) dokumentech vydaných IETF. [10], [26]

# 2 Není síť jako síť

Z předchozího textu je zřejmé, že se od sebe sítě liší. Každá síť může zabírat různě velkou oblast od budov až po celé státy a poskytovat různé služby různým uživatelům. Jednodušeji řečeno, každá slouží k nějakému účelu. V této kapitole se zaměříme na dělení sítí dle jejich rozlehlosti a dále na základě uspořádání jejich komponent.

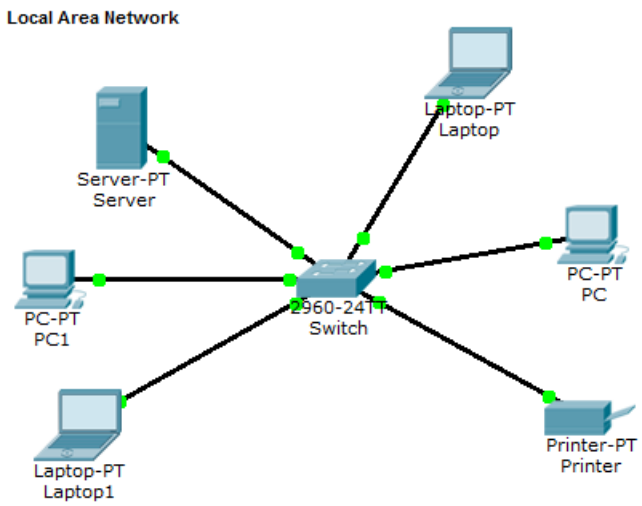
## 2.1 Každá síť je různě velká

Sítě se mnohou dělit do různých kategorií na základě různých kritérií. Jedním z nich je oblast, kterou zahrnují.

### 2.1.1 LAN (Local Area Network)

Local Area Network je síťová infrastruktura v rámci malých geografických oblastí. Jednotlivá zařízení nejsou příliš vzdálena, většinou se jedná o domácnosti, školy či nějaké firemní kancelářské budovy. Používá se ke sdílení prostředků, jako jsou například soubory a tiskárny. Daná síť je většinou zřízena a spravována samostatnou organizací nebo individuální osobou. Díky vysoké přenosové kapacitě (množství přenesených dat za jednotku času) tyto sítě umožňují vysokorychlostní přenos. [8]

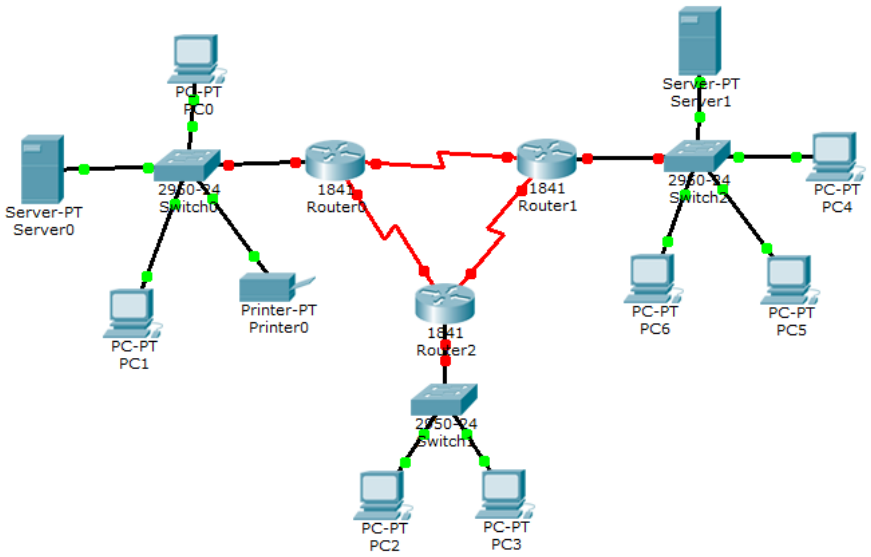
Zvláštním případem je Wireless LAN (WLAN), která používá k propojení bezdrátovou technologii.




**Obrázek 4 – Příklad LAN**

### 2.1.2 WAN (Wide Area Network)

Wide Area Network je opět síťová infrastruktura, tentokrát však zahrnující rozsáhlé geografické oblasti, což umožňuje komunikaci na velké vzdálenosti. Jedná se o propojení jednotlivých LAN, MAN a jiných menších sítí do větších a složitějších celků. Na rozdíl od LAN, které dosahovaly do vzdálenosti pouze pár kilometrů, mohou být WAN rozprostřeny na území měst, států, ale i kontinentů. Pokud se jedná o soukromé WAN sítě, správa je v rukou dané organizace. V případě pronajatých linek je v rukou poskytovatele služeb. WAN sítě, které slouží pro připojení dané organizace k Internetu, využívají služeb poskytovatelů internetových služeb. Oproti sítím LAN je přenosová kapacita u WAN sítí menší, což znamená nižší množství přenesených dat za jednotku času, přenosy jsou tedy pomalejší. [8]




**Obrázek 5 – Příklad WAN**

Mezi nejznámější WAN síť proto patří Internet. Internet je často nazýván jako „síť sítí“ z toho důvodu, že se jedná o nejrozsáhlejší existující síť. Jak již bylo řečeno, Internet je WAN síť, kterou tvoří velké množství různých vzájemně propojených sítí, ať už veřejných, nebo soukromých. Tyto sítě slouží jako prostředek pro komunikaci se zdroji mimo lokální sítě.

Pozor! Internet s velkým „I“ není to samé jako internet s malým „i“.

- **Internet** – všeobecně známy globální systém vzájemně propojených počítačových sítí.
- **internet** – jakýkoliv systém vzájemně propojených sítí, termín vznikl z anglického slova „internetworking“, což znamená „propojování sítí“.

### 2.1.3 Ostatní sítě dle rozlehlosti

Jako další síťovou infrastrukturu lze zmínit **Metropolitan Area Network (MAN)**, která bývá větší než LAN, ale menší než WAN. Opět se jedná o spojení několika LAN, tentokrát ve velikosti měst. Provozovatelem MAN bývá nějaká samostatná skupina či organizace.

V případě přenosu dat mezi zařízeními vzdálenými maximálně na pár metrů mluvíme o **Personal Area Network (PAN)**. Příkladem této sítě je technologie bluetooth, která umožňuje uživatelům posílat soubory mezi zařízeními, jako jsou notebooky, mobilní telefony, PDA a podobně. Tato síť je spravována jedním individuálním člověkem.

Jako poslední síť bude zmíněna **Storage Area Network (SAN)**, se kterou je možné se setkat zejména ve větších firmách z důvodu nákladnosti jejího zřízení. Tato síťová infrastruktura slouží připojení k souborovým serverům a poskytování úložiště dat za účelem jejich zálohování. [8], [28]

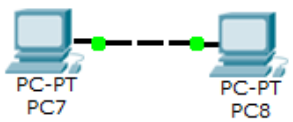
## 2.2 Topologie sítí

V předchozí části bylo uvedeno rozdělení sítí dle oblasti, kterou zahrnují. Nyní budou představeny sítě z hlediska uspořádání a zapojení jejich komponent. Na základě toho kritéria rozlišujeme různé topologie sítí. Než však budou představeny, je třeba zmínit dva základní pojmy. Jedním z nich je **fyzická topologie**, která popisuje fyzické uspořádání komponent v síti včetně jejich kabelového propojení. Druhým pojmem je **logická topologie**, jež je nezávislá na fyzickém propojení. Popisuje, jakým způsobem prochází data sítí. [26]

Dle fyzického uspořádání jsou rozlišovány tyto topologie [8], [23]:

### 2.2.1 Point-to-point

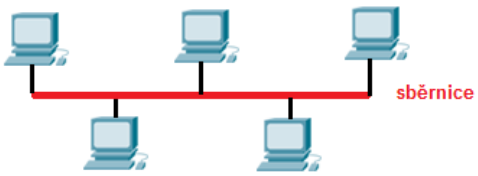
V případě této topologie se jedná o dvoubodové spojení, tedy o spojení pouze mezi dvěma zařízeními. Jde o nejjednodušší možnou topologii. Příkladem je telefonní spojení.




**Obrázek 6 – Topologie point-to-point**

### 2.2.2 Sběrnicová topologie

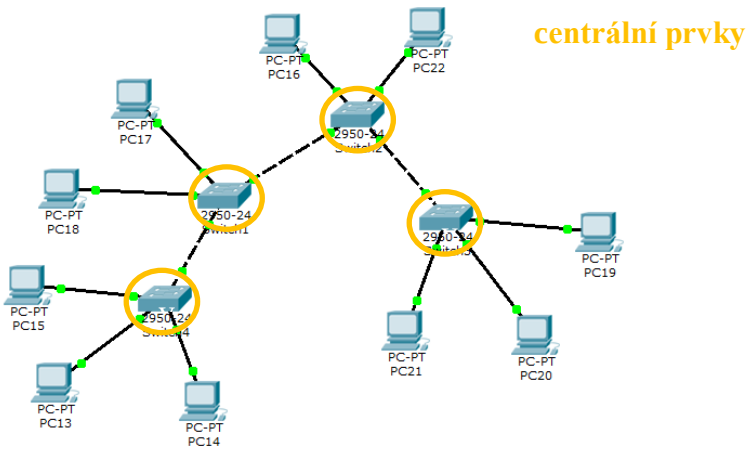
Tato topologie využívá jedno společné přenosové médium, které se nazývá sběrnice. Odtud pojmenování této topologie. Toto médium umožňuje obousměrný tok dat. Sběrnicová topologie se lehce zapojuje, je snadno rozšiřitelná a její pořizovací náklady nejsou vysoké. Nevýhodou je častý vznik kolizí v případě, že se o přístup ke sběrnici pokusí více zařízení najednou. Z tohoto důvodu je vhodnější tuto topologii využívat spíše u menších sítí. Největším problémem je právě sdílené médium, pokud u něj dojde k poškození, celá síť se stane nefunkční. Při velkém počtu připojených zařízení může dojít ke snížení výkonu.




**Obrázek 7 – Sběrnicová topologie**

### 2.2.3 Stromová topologie

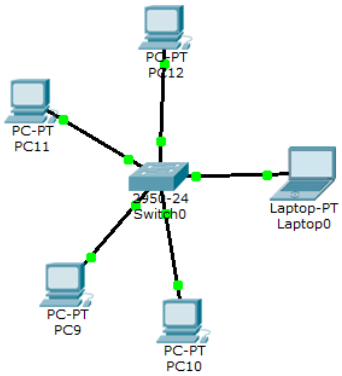
Tato topologie je hierarchickým zapojením uzlů sítě. Použití je výhodné zejména u složitějších sítí, která se skládá z mnoha uzlů. Uzel na vrcholu nové větve je vždy prvkem centrálním pro danou podřízenou část sítě. Výpadek jednoho zařízení neznamená pád celé sítě, ale pouze nefunkčnost té části sítě, která se skládá z daného zařízení a všech prvků na nižší hierarchické úrovni, které jsou k němu připojeny.




**Obrázek 8 – Stromová topologie**

### 2.2.4 Hvězdicová topologie

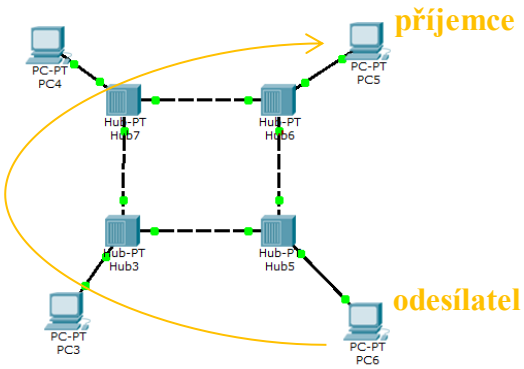
Uvedená topologie se skládá z jednoho centrálního prvku, například switche, ke kterému jsou připojena další zařízení, která přes tento centrální prvek komunikují. Pokud dojde k výpadku vedlejšího zařízení, síť funguje nadále, pokud by však došlo k výpadku centrálního uzlu, celá síť by se stala nefunkční.




**Obrázek 9 – Hvězdicová topologie**

### 2.2.5 Kruhová topologie

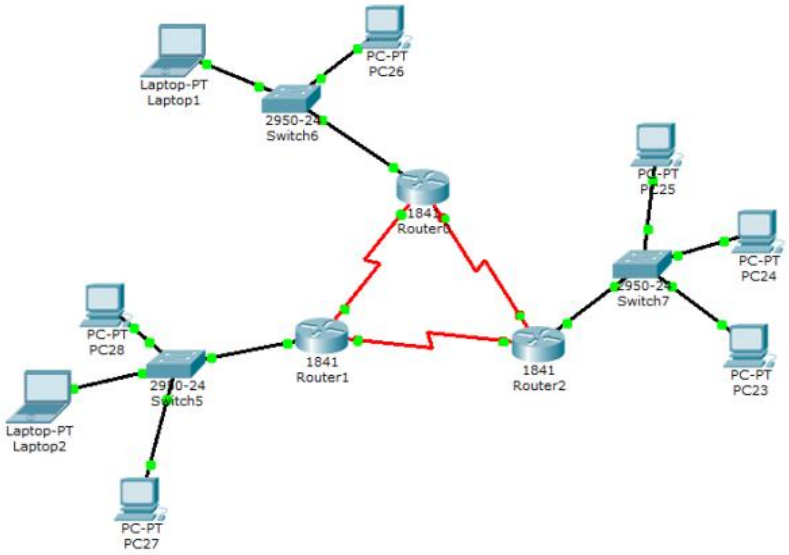
Topologie typu kruh je tvořena, jak z názvu plyne, zařízeními propojenými do tvaru kruhu. To znamená, že ke každému zařízení jsou připojena dvě další. Data v této topologii putují pouze jedním směrem. Vždy má právo vysílat pouze jedno zařízení, přičemž toto právo je dáno přiděleným tokenem. Centrální prvek v topologii typu kruh neexistuje. Data tedy musí cestou do cíle projít přes všechna zařízení, která se vyskytují mezi odesílatelem a příjemcem. S neexistencí centrálního prvku také souvisí hlavní nevýhoda tohoto uspořádání prvků sítě, což je nefunkčnost celé sítě v případě, že vypadne jedno ze zapojených zařízení.




**Obrázek 10 – Kruhová topologie**

### 2.2.6 **Smíšená topologie**

Jedná se o takovou topologii, kde jsou různá zařízení propojena s mnoha libovolnými dalšími. Žádný prvek není centrální. Tímto vznikají redundantní[1] spoje, což díky alternativním cestám mezi zařízeními snižuje výskyt výpadků sítě. Ne vždy však musí tato alternativní cesta existovat. Tato topologie je snadno rozšiřitelná, ale z důvodu mnoha spojů a žádného centrálního prvku je třeba nějakým způsobem řídit tok dat, jinak by mohlo docházet k nekonečnému putování dat sítí, aniž by se dostala do cíle. Z toho důvodu je nutné zavést směrování. Zvláštním případem této topologie je topologie, kde je každé zařízení propojeno se všemi ostatními uzly v síti, což je ale v praxi většinou těžko realizovatelné, zejména ve složitých sítích.




**Obrázek 11 – Smíšená topologie**

[1] Redundance = nadbytečnost, v této souvislosti znamená existenci náhradních (nadbytečných) cest mezi zařízeními v síti

# 3 Vrstvové modely, aneb bez pravidel to nejde

Jak již z názvu kapitoly vyplývá, tato část práce bude věnována protokolům uspořádaných do jednotlivých vrstev určitého modelu. Konkrétně se jedná o standardy ISO/OSI a TCP/IP.

Proč právě vrstvy? Tímto členěním je daná komunikace mezi dvěma systémy rozdělena na jednotlivé části, kde každá z nich má na starosti určité funkce, které jsou definovány konkrétním protokolem. Dané vrstvy mají jistou hierarchii a jsou seřazeny v určitém pořadí. Nižší vrstva vždy poskytuje své služby nejbližší vyšší vrstvě, aniž by vyšší vrstva měla informace o přesné realizaci služby, kterou využívá. To umožňuje náhradu daného protokolu na určité vrstvě jiným stejně pracujícím protokolem, aniž by to nějak ovlivnilo sousední vrstvy.

Každá vrstva se skládá z **entit**, které vykonávají funkce umožňující poskytování požadovaných služeb. Entity mezi sebou komunikují. Tato komunikace může být **vertikální** (mezi nižší a vyšší sousední vrstvou daného systému) nebo **horizontální** (entity komunikují v rámci odlišných systémů neboli zařízení, ale prostřednictvím stejné vrstvy).

Důležitým pojmem spojeným s touto problematikou je **datová jednotka** neboli PDU (Protocol Data Unit). Jedná se o data, která putují sítí a případně další doplňkové informace. Na každé vrstvě modelu má tato jednotka odlišnou podobu, která je definovaná příslušným protokolem dané vrstvy.

Změny ve struktuře PDU jsou způsobeny **zapouzdřováním (encapsulation)**. Při každém přechodu z jedné vrstvy do druhé je totiž přidáno v prvním komunikujícím systému záhlaví specifikované konkrétním protokolem. Záhlaví obsahuje řídící informace k tomu, aby uživatelská data mohla být postupně přeposílána sítí. Tato přidaná záhlaví jsou poté v druhém systému postupně odebírána, což opět umožňuje průchod mezi vrstvami. Tomuto procesu se říká **odpouzdřování (decapsulation).** Záhlaví může být rozbaleno pouze tím samým protokolem, kterým bylo zapouzdřeno. V tomto případě se jedná o již zmíněnou horizontální komunikaci. [10], [23], [27]

## 3.1 Referenční model ISO/OSI vs. model TCP/IP

Prvním zmíněný model ISO/OSI byl standardizován organizací ISO roku 1984. Skládá se ze sedmi hierarchicky uspořádaných vrstev, kde každá z nich plní jistou funkci. Model ISO/OSI je referenčním modelem[2], to znamená, že slouží pouze jako neformální popis funkčnosti jednotlivých vrstev, tedy popisuje, co jaká vrstva dělá. Nejedná se však o přesnou specifikaci implementace a definici konkrétních protokolů. Jeho účelem je názorný popis funkčnosti a také jednotlivých procesů, které na daných vrstvách probíhají. Struktura je uvedena v tabulce níže (Tabulka 1).

**Tabulka 1 – Referenční model ISO/OSI**

| název datové jednotky | vrstva | název vrstvy |
|---|---|---|
| Data | 7 | Aplikační |
| | 6 | Prezentační |
| | 5 | Relační |
| Segmenty | 4 | Transportní |
| Pakety | 3 | Síťová |
| Rámce | 2 | Linková |
| Bity | 1 | Fyzická |

V levém sloupci tabulky jsou uvedeny názvy, které se používají pro jednotlivé formáty datových jednotek (PDU) na příslušných vrstvách, v druhém sloupci jsou očíslovány jednotlivé vrstvy a ve třetím sloupci jejich názvy.

Další zmíněný model TCP/IP je v podstatě podobný ISO/OSI modelu, ale liší se tím, že popisuje konkrétní architekturu sady protokolů TCP/IP. To znamená, že je přesně standardizována nejen funkčnost jednotlivých vrstev, ale i konkrétní protokoly pracující na těchto vrstvách. Díky tomu je tento model, který původně vznikl pouze pro vojenskou sféru, v praxi běžně využíván. Výjimkou je poslední vrstva síťového rozhraní, pro kterou není jednotná standardizace, vše je závislé na konkrétní použité přenosové technologii.

[2] Vedle referenčního modelu existuje také protokolový model, který přesně odpovídá struktuře konkrétní sady protokolů a popisuje funkce jednotlivých vrstev této sady protokolů. Příkladem je TCP/IP.

Struktura TCP/IP se od ISO/OSI liší počtem vrstev. Jejich počet je totiž omezen na číslo čtyři. Jak již bylo řečeno, TCP/IP v podstatě vychází z modelu ISO/OSI, proto si jednotlivé vrstvy těchto modelů odpovídají. Jakým způsobem si odpovídají, je uvedeno v tabule níže (Tabulka 2).

**Tabulka 2 – ISO/OSI vs. TCP/IP**

| vrstva | název vrstvy | | vrstva | název vrstvy |
|---|---|---|---|---|
| 7 | Aplikační | | 4 | Aplikační |
| 6 | Prezentační | | | |
| 5 | Relační | | | |
| 4 | Transportní | | 3 | Transportní |
| 3 | Síťová | | 2 | Internetová |
| 2 | Linková | | 1 | Vrstva síťového rozhraní |
| 1 | Fyzická | | | |

Z výše uvedené tabulky (Tabulka 2) je patrné, že vrstvy 7 až 5 OSI modelu jsou v TCP/IP spojeny do jedné, transportní si vzájemně odpovídají, dále místo síťové vrstvy je v TCP/IP vrstva internetová a vrstvy 2 až 1 jsou sloučeny do vrstvy síťového rozhraní. [10], [27]

## 3.2 Základní charakteristika modelu ISO/OSI

Základní rozdíly mezi ISO/OSI a TCP/IP byly již uvedeny. Nyní budou představeny základní funkce jednotlivých vrstev OSI modelu.

Jak již bylo řečeno, OSI model se skládá ze sedmi vrstev. Vrstvy 7 až 3 jsou vždy realizovány softwarově, vrstvy 2 až 1 mohou být realizovány jak softwarově tak i hardwarově. Z hlediska využití lze rozdělit vrstvy na 7 až 5, které pracují na úrovni uživatele a mají za úkol zajistit uživateli přístup do sítě, převést data do správného formátu a navázat spojení mezi koncovými systémy. Dá se říci, že připravují vše potřebné pro přenos dat. Vrstvy 4 až 1 se zabývají samotným přenosem dat. [10]

**Aplikační vrstva** je nejvyšší vrstvou. Poskytuje rozhraní mezi aplikacemi, jež uživatelům umožňují komunikaci, a konkrétní sítí, přes kterou dochází k přenosu požadovaných dat. Na této vrstvě probíhá ověřování komunikujících systémů, jejich synchronizace a je definováno, jak má vypadat práce s daty. Je tedy určeno, jakým způsobem budou přeposílána,

jak má daná zpráva vypadat, použitá syntaxe, jakým způsobem budou zprávy zabezpečeny, kdo je zodpovědný za chyby apod. [10], [23]

**Prezentační vrstva** překládá data pocházející z jednoho systému do srozumitelného formátu pro aplikace na druhém systému. U každého komunikujícího systému se totiž může lišit syntaxe dat, což znamená, že by u druhého komunikujícího zařízení nemusela být data čitelná. Na této vrstvě mohou být dále data komprimována či šifrována a naopak. Tím může dojít ke změně původních dat, ne pouze k přidání záhlaví. Díky dešifrování se data poté k cílovému systému dostanou opět nezměněná. Dalším důvodem, proč tato vrstva existuje, je posílání požadavků na zahájení relace čí jejího zastavení. [23], [26]

**Relační vrstva** slouží k vytváření a ukončování relace mez danými aplikacemi, kterou udržuje na potřebnou dobu, případně se stará o její obnovení. Posílá žádosti na tvorbu spojení transportní vrstvou. Opět je jejím úkolem synchronizovat komunikaci a starat se o řízení výměny dat. Komunikace může probíhat obousměrně, kdy dané systémy komunikují najednou nebo se střídají, případně jednosměrně. [10], [26]

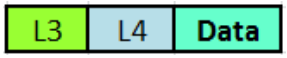

**Obrázek 12 – PDU L7 až L5**

**Transportní vrstva** tvoří jakýsi předěl mezi uživatelskými aplikacemi a danou sítí, jelikož díky této vrstvě je implementace samotné sítě skryta před vrstvami na vyšší úrovni. Stará se o end-to-end komunikaci neboli přenos dat mezi koncovými systémy. Mezi danými systémy nemusí existovat pouze jedno spojení. Jejím úkolem je tato data spolehlivě doručit do cíle a v případě zjištění chyby najít její řešení. Do této chvíle se stále mluvilo o datové jednotce PDU jako o datech. Na této vrstvě jsou však data rozdělena do jednotlivých částí, kterým je přidána hlavička s informacemi pro jejich další průchod sítí, tyto části s přidanou hlavičkou jsou nazývány jako **segmenty**. [10]


**Obrázek 13 – PDU L4 (segment)**

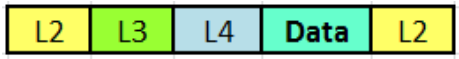
**Síťová vrstva** slouží pro přenos dat mezi nesousedními zařízeními. Jejím hlavním úkolem je najít vhodnou cestu do cíle pomocí logických adres. To znamená, že má tato vrstva přehled o tom, jak jsou zařízení v této síti uspořádána a tím i o možných cestách do jednotlivých

koncových zařízení. Datovou jednotkou po zapouzdření na této vrstvě již není segment, ale **paket**. [10]



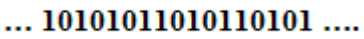

**Obrázek 14 – PDU L3 (paket)**

**Linková (spojová) vrstva** se stará o řízení přenosu dat mezi sousedními systémy neboli mezi těmi, co jsou propojeny stejnou linkou. Pakety jsou na této vrstvě zapouzdřeny do **rámce** přidáním hlavičky a patičky a poslány další vrstvě. Na rozdíl od síťové vrstvy, kde byla využívána logická adresace, je na této vrstvě použita adresace fyzická. Jejím úkolem je utvářet spojení, které je nutné poté udržet na požadovanou dobu a poté ho ukončit. Během spojení se mohou vyskytovat chyby, které by měla být spojová vrstva schopná detekovat (například pomocí kontrolního součtu), případně je napravit, a pokud by se vyskytla chyba bez možnosti opravy, musí tuto skutečnost oznámit. [10], [23]




**Obrázek 15 – PDU L2 (rámec)**

**Fyzická vrstva** je vrstva, která se stará o komunikaci pomocí signálu ve formě **bitů**, aniž by jim přiřazovala nějaký význam. Z toho plyne, že ve spojitosti s fyzickou vrstvou již nemluvíme o rámcích, ale bitech. Jejím úkolem je přenést tento signál přes dané fyzické médium. Na této vrstvě je definována kabeláž, konektory, charakter signálu apod. [10], [26]




**Obrázek 16 – PDU L1 (bity)**

## 3.3 Základní charakteristika modelu TCP/IP

Jak již bylo řečeno, vrstvy modelu TCP/IP mohou být mapovány na vrstvy v modelu ISO/OSI, což bylo znázorněno v tabulce s označením Tabulka 2. Proto není nutné rozepisovat funkčnost jednotlivých vrstev TCP/IP, jelikož je velmi podobná.

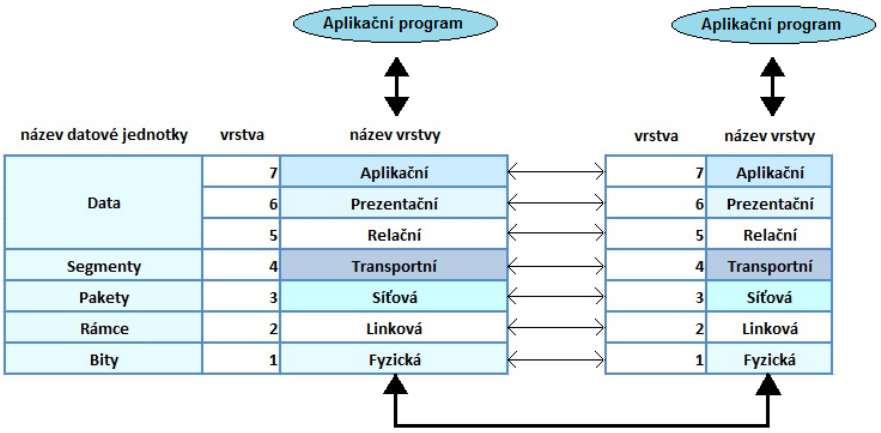
## 3.4 Průchod dat sítí

Než se začneme věnovat konkrétním protokolům využitých u těchto modelů, bylo by dobré si říci, jakým způsobem data putují z jednoho zařízení do druhého. Pro srozumitelnost je uveden názorný příklad.

Nejprve musí existovat iniciátor komunikace, který chce odeslat data na jiné zařízení. Tohoto odesílatele si je možné si jako babičku píšící dopis své kamarádce do Prahy. Jelikož šlo dědečkovi psaní textů vždy o něco lépe, předá mu napsaný dopis ke kontrole a finálním úpravám, podobně jako aplikační vrstva prezentační vrstvě. Vnučka poté nadepíše adresu a vloží dopis do obálky, což odpovídá činnostem na vrstvě relační. Poté je třeba donést dopis na poštu, o což se postará druhá vnučka, která má cestu kolem. Tato část odpovídá činnostem na transportní vrstvě. Pracovnice přepážky si přečte adresu uvedenou na obálce, a na základě ní zvolí, do jaké přihrádky zásilku uloží, stejně tak dochází k volbě vhodné cesty do cíle na síťové vrstvě. Pošta dále zaměstnává pracovníky, kteří zabalí všechny zásilky do různých balíků na základě jejich místa určení, stejně jako jsou zapouzdřeny pakety do rámce. Poté přijede řidič, který má za úkol dané balíky s dopisy doručit, balíky naloží a odveze na místo určení. Tento mechanismus je analogií pro přenos dat sítí prostřednictvím přenosových médií na fyzické vrstvě. Jakmile dorazí balík s dopisy na poštu, je úkolem pracovníka balík rozbalit, což odpovídá procesu rozbalování rámců na pakety na linkové vrstvě. Dopisy z balíků je třeba dále roztřídit na hromádky dle uvedených adres. Podobným způsobem je volena vhodná cesta pro pakety na síťové vrstvě. Dopisy z hromádek budou postupně doručovány poštovní doručovatelkou. Poté, co poštovní doručovatelka, doručí dopis do schránky, manžel kamarádky cestou domů zásilku vyzvedne a informuje Lenku o příchodu dopisu. Jakmile se vrátí Lenka z práce, dopis si přečte.

Důležité je si z předchozího příkladu uvědomit to, že komunikace mezi dvěma systémy neprobíhá pouze díky jednomu průchodu modelem. Každý bit přijatý na druhém zařízení musí být opět průchodem přes jednotlivé vrstvy zpracován do původní podoby, aby byl čitelný pro příjemce. Pokud byla data na určité vrstvě prvního komunikujícího systému zašifrována, jsou na stejné vrstvě u druhého komunikujícího systému opět dešifrována.

Na obrázku níže (Obrázek 17) je znázorněn průchod dat sítí.




**Obrázek 17 – Průchod dat sítí**

V předchozích částech byla představena funkčnost jednotlivých vrstev ISO/OSI modelu, která je do jisté míry stejná jako u TCP/IP, nyní budou představeny jednotlivé protokoly pracující na těchto vrstvách.

# 4 Aplikační vrstva

Tato část bude věnována prvním třem vrstvám modelu ISO/OSI, resp. první vrstvě modelu TCP/IP, tedy vrstvě aplikační.

Jak již bylo uvedeno, aplikační vrstva slouží jako rozhraní pro aplikace využívané uživateli a danou síť, přes kterou dochází k přenosu dat. Protokoly aplikační vrstvy se používají k výměně dat mezi programy běžícími na obou koncových zařízeních.

Prezentační vrstva převádí data do formátu srozumitelného pro koncové zařízení. Dochází ke kompresi a šifrování dat.

Relační vrstva vytváří, udržuje a obnovuje relace mezi koncovými zařízeními.

U modelu TCP/IP jsou tyto tři vrstvy spojeny do jedné.

**Tabulka 3 – Aplikační vrstva**

| ISO/OSI | | TCP/IP | |
|---|---|---|---|
| **vrstva** | **název vrstvy** | **vrstva** | **název vrstvy** |
| 7 | Aplikační | 4 | Aplikační |
| 6 | Prezentační | | |
| 5 | Relační | | |
| 4 | Transportní | 3 | Transportní |
| 3 | Síťová | 2 | Internetová |
| 2 | Linková | 1 | Vrstva síťového rozhraní |
| 1 | Fyzická | | |

Nyní budou představeny jednotlivé protokoly pracující na výše uvedených vrstvách.

## 4.1 HTTP (Hypertext Transfer Protocol)

Jedním z nejpoužívanějších protokolů je **HTTP**, který uživateli poskytuje webové služby. Protokol je definovaný v RFC 2616, respektive jeho verze 1.1, jež je nyní běžně používána. Využívá protokol TCP transportní vrstvy a port s číslem 80. Obojí bude popsáno v kapitole týkající se transportní vrstvy. Protokol HTTP je schopen přenášet soubory jakéhokoliv typu. Díky němu je po zadání platné URL adresy možné prohlížet webové stránky.

URL neboli Uniform Resource Locator jednoznačně identifikuje jednotlivé informační zdroje vyskytující se na Internetu. Adresa URL má určitou strukturu, pro představu je uvedena adresa http://www.pokus.cz/stranky/index.html.

- **http** slouží k identifikaci použitého protokolu
- **www.pokus.cz** je doménové jméno serveru, které se skládá z domény prvního řádu (cz), druhého řádu (pokus) a třetího (www)
- **stranky** slouží k identifikaci adresáře, ve kterém se nachází soubor s názvem index.html
- **index.html** je název požadovaného souboru

Jak ale přesně tento protokol funguje? Jedná se o komunikační model typu klient-server, kdy server odpovídá na požadavky klienta. Protokol http je bezstavový, což znamená, že není schopen udržovat informace o stavu komunikace. Případné související dotazy jsou pro tento protokol na sobě nezávislé.

Jeho mechanismus může být představen na procesu zobrazování obsahu webové stránky uloženém v souboru index.html. Jako rozhraní pro uživatele v tomto případě slouží webový prohlížeč, do kterého má možnost zadat URL adresu požadované stránky. Po zadání této URL adresy pošle prohlížeč (klient) GET žádost na server o zobrazení požadovaného souboru (index.html). Jelikož je protokol HTTP nešifrovaný, jsou veškeré žádosti posílány formou obyčejného textu. Server přijme žádost a odpoví. Jednou z možných odpovědí je status HTTP/1.1 200 OK jako informace o tom, že soubor existuje, společně s nalezeným obsahem. Druhou možnou odpovědí je chybové hlášení. V případě nalezení požadované stránky zašle server webovému prohlížeči HTML kód, díky kterému je daný prohlížeč schopen požadovanou stránku uživateli zobrazit.

Kromě žádostí GET existují další žádosti například PUT a POST, které slouží pro nahrání dat uživatele na server, dále HEAD, DELETE, TRACE, OPTIONS a CONNECT.

Jak již bylo zmíněno, protokol HTTP je nešifrovaný, což není příliš vhodné, jelikož daná komunikace by mohla být zachycena nežádoucím uživatelem. Proto existuje zabezpečená varianta a tou je protokol HTTPS, který využívá port 443. Díky Secure Socket Layer (SSL) jsou data před přenosem zašifrována, což značně zvyšuje bezpečnost. Nevýhodou je však o něco vyšší zátěž a tím i částečné zpomalení. [9], [25]

Další důležitou službou jsou emailové služby. Emailové zprávy jsou uloženy v databázi na poštovních serverech, které patří danému poskytovateli internetových služeb. Veškeré čtení a posílání emailových zpráv probíhá právě přes tyto servery.

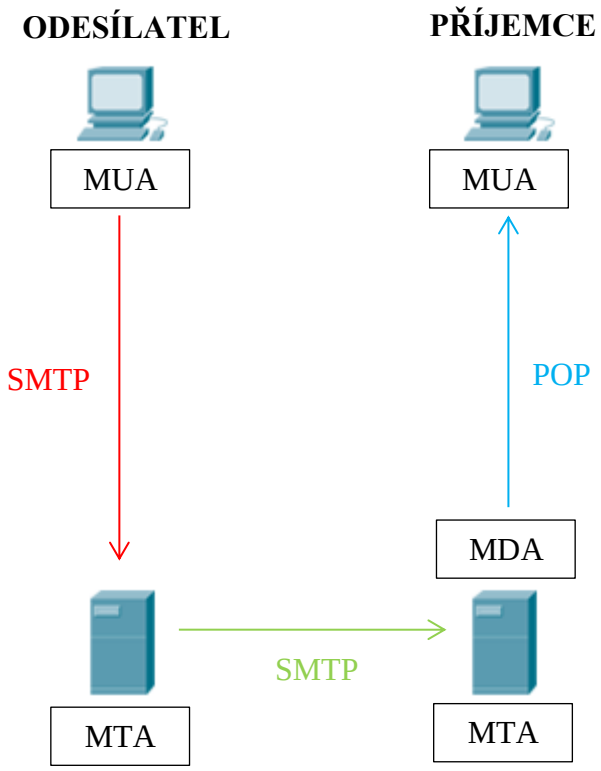
Pro poskytování emailových služeb existují tři různé protokoly SMTP (Simple Mail Transfer Protocol), POP3 (Post Office Protocol a IMAP (Internet Message Protocol). Pro posílání je používán SMTP. Pro zpřístupnění zpráv je používán POP3 nebo IMAP.

## 4.2 SMTP (Simple Mail Transfer Protocol)

**SMTP** je protokol definovaný v RFC 2821 a u nešifrovaného spojení naslouchá na portu TCP/25, u šifrovaného SSL[3] spojení na portu 465. Jeho komunikace probíhá na základě modelu klient-server. Jako klient zde funguje **poštovní klient MUA (Mail User Agent)**. Jedná se o program, který slouží jako rozhraní uživatelům pracujícím s elektronickou poštovní schránkou, ale nestará se o doručování zpráv. Příkladem je MS Outlook. **Poštovní server MTA (Mail Transfer Agent)** opět běží jako program v počítači. Jeho úkolem je zjistit, zda příjemce spadá do místní domény (doména je uvedena za @ v emailové adrese). V případě, že je zpráva určena příjemci na jiném systému, přepošle zprávu na příslušný server v příslušné doméně na základě adresy přijaté z DNS serveru. Cílový server může být zaneprázdněn nebo off-line, v tom případě se zprávy řadí do fronty. MTA se nestará o samotné doručení zprávy do schránky, to je úkolem **poštovního doručovacího agenta MDA (Mail Delivery Agent)**. MDA je počítačový program, který běží na stejném zařízení jako MTA. Stará se nejen o doručení zpráv do schránek, ale také o filtraci nežádoucí pošty. V případě, že nedojde, k doručení zprávy před vypršením expiračního času, je zpráva vrácena odesílateli jako nedoručitelná. [9], [23]

[3] Secure Sockets Layer (SSL) - protokol umožňující šifrovanou komunikaci mezi zařízeními. Používá dvě formy klíčů (veřejné a soukromé).

Na obrázku níže (Obrázek 18) je uveden mechanismus přeposílání poštovních zpráv.




**Obrázek 18 – Mechanismus přeposílání poštovních zpráv**

Uživatel prostřednictvím poštovního klienta (MUA) napíše zprávu a odešle pomocí tlačítka „odeslat“ na zadanou emailovou adresu. Prostřednictvím protokolu SMTP je zpráva přeposlána na poštovní server (MTA), kde je rozhodováno, zda je zpráva určena pro místní doménu, k čemuž využívá DNS server. V případě na obrázku doména neodpovídá a proto je nutné přeposlat pomocí SMTP protokolu zprávu na příslušný další poštovní server (MTA) ve správné doméně. Zde již poštovní server zjišťuje, že zpráva je určena pro uživatele v místní doméně a proto ji předává doručovacímu agentovi (MDA) k doručení do schránky. Uživatel si poté prostřednictvím svého poštovního klienta může zprávu vyzvednout. Toto vyzvednutí zprávy probíhá konkrétně pomocí protokolu POP, který bude představen v následující části. [23]

## 4.3 POP (Post Office Protocol)

Jak již bylo uvedeno, POP protokol umožňuje vyzvedávání elektronické pošty ze serveru. Protokol POP prošel několika verzemi, v současnosti je používán POP3 (dále jen POP). Tento protokol je standardizován v RFC 1939 a v případě nešifrovaného spojení využívá port 110,

u šifrovaného SSL spojení 995. Standardně je protokol POP nešifrovaný, ale je zde i možnost využít například hash funkcí[4] pro zabezpečení hesel. Pro stažení zpráv je nutné vytvořit TCP spojení klienta se serverem. Nevýhodou tohoto protokolu je stažení veškeré pošty uložené na serveru, i když uživatel požaduje pouze některé. Z tohoto důvodu je výhodnější použití protokolu POP u uživatelů, kteří spravují svou elektronickou poštu pouze na jednom počítači, kam jsou dané zprávy staženy. Jakmile je emailová zpráva ze serveru stažena, je tato zpráva na serveru smazána, což zamezuje zahlcení serveru zprávami. Výhodou je však použití pro uživatele, kteří mají problém s internetovým připojením. Připojení je nutné pouze pro stažení zpráv, jejich prohlížení je poté možné i bez aktivního připojení. K dané schránce může být připojen vždy jen jeden uživatel. [9], [23]

## 4.4 IMAP (Internet Message Access Protocol)

IMAP, stejně jako POP, umožňuje vyzvedávání elektronické pošty. Existuje mnoho verzí, v současnosti využívaná je verze IMAP4, jež je standardizována v RFC 3501. Nešifrovaná verze naslouchá na portu TCP/143, šifrované SSL spojení na portu 993. Výhoda tohoto protokolu je ta, že na rozdíl od POP, kde byly stahovány všechny zprávy, jsou prostřednictvím IMAP ze serveru stahovány pouze hlavičky zpráv. Díky tomu je umožněno vyhledávat požadované zprávy, aniž by byly staženy. Obsah zprávy je stažen, pouze pokud uživatel vyžaduje její přečtení. Díky tomuto mechanismu je možné, aby s jednou schránkou pracovalo více uživatelů odkudkoliv. Případné změny stavů zpráv (přečteno/nepřečteno apod.) provedené jedním uživatelem jsou viditelné u všech právě připojených uživatelů. Nevýhodou práce přímo na serveru je ta, že je nutné neustálé připojení k Internetu. Další nevýhodou je větší náchylnost k výskytu chyb, jelikož se jedná o značně komplikovanou implementaci. [9], [23]

Kromě poštovních klientů jako je MS Outlook existují i webové aplikace, které také umožňují přístup k poštovním schránkám. Mezi ně patří například gmail.com, seznam.cz apod.

Webové aplikace jako je například gmail.com je možné prostřednictvím POP nebo IMAP synchronizovat s poštovním klientem jako je MS Outlook.

[4] Hash funkce jsou matematické funkce, které převádí data na vstupu do speciálního výstupu za účelem rychlejšího prohledávání či bezpečnosti

## 4.5 DNS (Domain Name System)

Tento protokol je definovaný v RFC 1035 a naslouchá na portu TCP/53 i UDP/53. Slouží pro překlad doménových jmen na IP adresy. Doménová jména existují pro jednodušší zapamatování. Pamatují se lépe než samotné IP adresy. V případě, že dojde ke změně IP adresy, doménové jméno zůstává stále stejné a uživatel se nemusí učit novou IP adresu.

**Plně kvalifikované doménové jméno** je složeno z jednotlivých domén oddělených tečkou, jež jsou hierarchicky uspořádány. Příkladem může být doménové jméno www.gmail.com. Pořadí domén je určováno zprava doleva. Doména **nejvyššího řádu (TLD)** je ta nejvíce vpravo, v tomto případě „com“, další doménou je **doména druhého řádu (SLD)**, konkrétně „gmail“, poslední doménou, je doména třetího řádu, ve výše uvedeném příkladu se jedná o „www“.

Důvodem hierarchického uspořádání byl omezený prostor pro doménová jména. Pro danou doménu existuje vždy několik domén nižšího řádu, které jsou pro tuto nadřízenou doménu unikátní. Díky tomuto může pro doménu „.com“, existovat subdoména „google.com“ a pro ni další subdomény jako jsou „maps.google.com“, „mail.google.com“ a například „docs.google.com“.

Domény prvního řádu přiděluje organizace IANA[5]. Jsou rozděleny do tří hlavních kategorií. První kategorií jsou **národní domény (ccTLD)**, které jsou tvořeny zejména ISO kódy daných zemí a skládají se ze dvou písmen. Příkladem národních domén jsou „.cz“, „.uk“, „.eu“. Další kategorií jsou všeobecné, **generické domény (gTLD)** složené zejména ze tří písmen. Příkladem může být „.org“, „.com“, „.net“. Do této kategorie spadají také domény, které jsou složeny z více znaků, sem spadá například „.bike“, „.mobi“. Jedná se o domény **sponzorované (sTLD)**. Další kategorií jsou **infrastrukturní domény**, kam spadá doména „.arpa“. Stejně jako domény, jsou jednotlivé servery na základě informací, které mají o daných doménách, uspořádány do hierarchické struktury, konkrétně stromové.

Veškerá komunikace mezi klientem a serverem, ať už se jedná o klientské žádosti, odpovědi ze serverů, chybové zprávy nebo zprávy mezi servery navzájem, probíhá pomocí zpráv, jejichž formát je definovaný BIND (Berkely Internet Name Domain).

[5] Internet Assigned Numbers Authority (IANA) je organizace zodpovědná za přidělování doménových jmen, IP adres, čísel portů a správu různých internetových protokolů. V současnosti spadá pod organizaci ICANN (Internet Corporation for Assigned Names and Numbers).

Zpráva se skládá z pěti částí, její struktura je zobrazena v tabulce (Tabulka 4) pod tímto odstavcem. První částí je samotná hlavička DNS zprávy, dále je obsažena žádost klienta na server, pod ní následuje blok pro odpověď serveru na žádost klienta, další část poukazuje na autority a v poslední kolonce jsou uvedeny případné doplňující informace. [9], [13], [23]

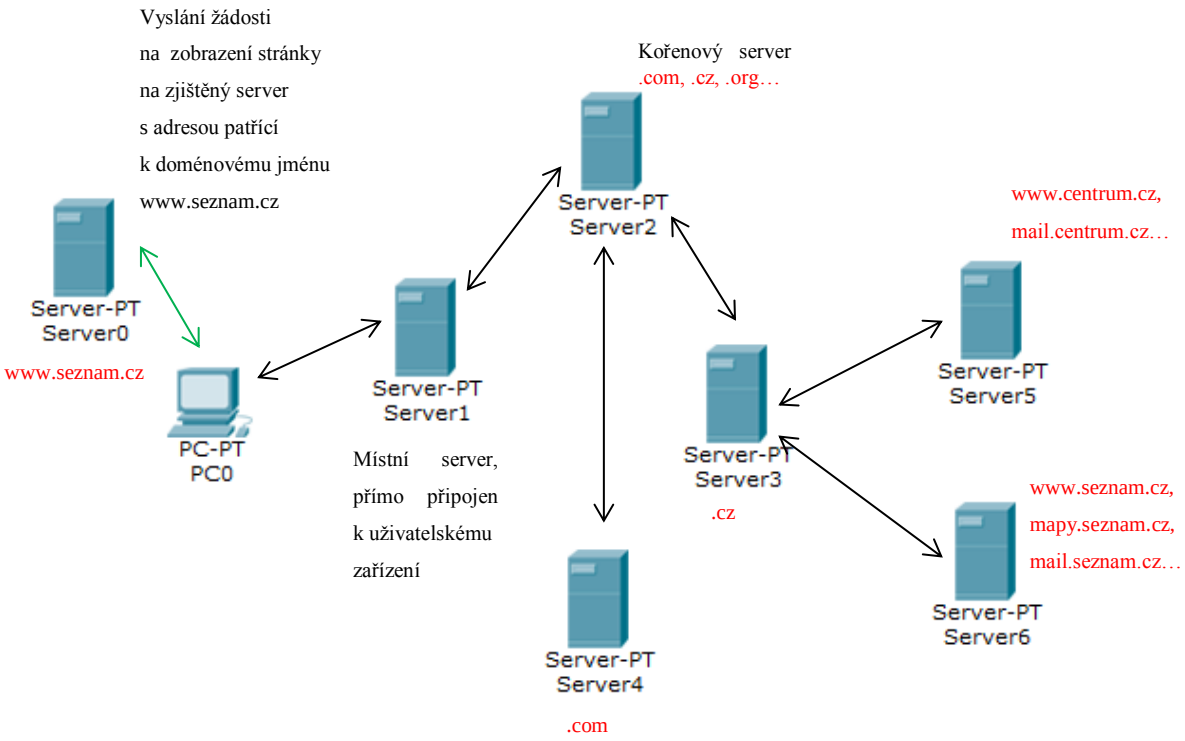
**Tabulka 4 – Formát DNS zprávy**

| HLAVIČKA |
| --- |
| ŽÁDOST |
| ODPOVĚĎ |
| AUTORITA |
| DOPLŇUJÍCÍ INFORMACE |

Jak ale probíhá vlastní komunikace mezi klientem a serverem?

1) Uživatel zadá do webového prohlížeče doménové jméno. Například www.seznam.cz. Tím je klientem poslána žádost na místní jmenný server, kde jsou uloženy informace o IP adresách patřícím k příslušným doménovým jménům.
2) V případě, že server najde záznam doménového jména ve své databázi, zašle klientovi ihned odpověď, která obsahuje IP adresu odpovídající danému doménovému jménu.
3) V případě, že záznam nenalezl, zašle klientskou žádost na další jmenné servery.
4) Pokud žádný z těchto serverů opět nezná doménové jméno www.seznam.cz, ale má například informace od doméně „.cz“, sdělí tuto informaci místnímu serveru (na níže uvedeném obrázku (Obrázek 19) tuto informaci sděluje kořenový server Server2).
5) Místní server tentokrát pošle žádost pouze na ty servery, které doménu „.cz“ obsahují.
6) Tyto servery buď posílají zpět IP adresu patřící k doménovému jménu, pokud toto jméno znají, nebo posílají informace o serverech, které ví o doméně „seznam.cz“ (v tomto případě Server3).
7) Místní server opět posílá žádost o překlad doménového jména www.seznam.cz pouze na servery znající doménu „seznam.cz“.
8) Ten, který má o daném doménovém jménu záznam (jedná se o Server6), zašle místnímu serveru odpovídající IP adresu (jedná se o adresu Server0).
9) Místní server danou IP adresu ji přepošle na klienta, který ji poté může využít pro žádost o zobrazení příslušné webové stránky u Serveru0. [13]

Celý mechanismus je znázorněn na tomto obrázku (Obrázek 19):




**Obrázek 19 – DNS mechanismus**

Na DNS serveru jsou v databázích uloženy různé typy záznamů. Mezi ně patří zejména:

- A – adresa koncového zařízení, která je přiřazena právě danému doménovému jménu.
- NS – autoritativní jmenný server pro danou doménu.
- CNAME – jedná se o kanonické jméno, které využívá IP adresu jiné domény (například pro www.seznam.cz a ftp.seznam.cz existuje pouze jedna IP adresa).

Pro posílání dotazů na DNS server je možné využít příkazový řádek. K tomuto účelu slouží příkaz ***nslookup***. Existují dva režimy zadávání. První je interaktivní, který se zadává bez parametrů. Po zadání *nslookup* bez parametrů se spustí interaktivní režim, ve kterém se zadávají pouze parametry. Parametry, které je možné zadat lze vypsat příkazem *help* zadaným v tomto režimu. Další variantou je neinteraktivní režim, kde je *nslookup* zadáván přímo s požadovanými parametry. Rozdíly v režimech jsou uvedeny v následujících výpisech. Jako příklad je uvedena žádost na získání IP adresy domény „seznam.cz“. [7]

*Ukázkové výpisy:*

**INTERAKTIVNÍ REŽIM**

```
C:\>nslookup
Vychozi server:   ns2.erkor.cz
Address:  62.240.183.6
Aliases:  6.183.240.62.in-addr.arpa

> seznam.cz
Server:  ns2.erkor.cz
Address:  62.240.183.6
Aliases:  6.183.240.62.in-addr.arpa

Neautorizovana odpoved:
Nazev:   seznam.cz
Addresses:  2a02:598:2::3
77.75.76.3
>
```

**NEITERAKTIVNÍ REŽIM**

```
C:\>nslookup seznam.cz
Server:  ns2.erkor.cz
Address:  62.240.183.6
Aliases:  6.183.240.62.in-addr.arpa

Neautorizovana odpoved:
Nazev:   seznam.cz
Addresses:  2a02:598:2::3
          77.75.76.3
C:\>
```

Z výpisu příkazového řádku je možné zjistit, že IPv4 adresa domény „seznam.cz“ je 77.75.76.3 a IPv4 je 2a02:598:1::3.

## 4.6 Telnet (TELetype NETwork service)

Telnet je nešifrovaný protokol používaný pro připojení ke vzdálenému síťovému zařízení. Je definován v RFC 854 a naslouchá na portu TCP/23.

Pro připojení ke vzdálenému zařízení (serveru) je využíván klient Telnetu, který běží na místním počítači. Tento klient bývá součástí jak MS Windows, tak i Linuxu, ale existuje i možnost použití speciálních aplikací, mezi které patří zejména Putty nebo HyperTerminal. Tito klienti jsou schopni emulovat terminál vzdáleného zařízení. Díky vytvořenému spojení je tedy možné realizovat úkony se zařízením, které nemá uživatel fyzicky dostupné jako by fyzicky dostupné bylo. Dané spojení je utvořeno protokolem TCP transportní vrstvy.

Aby server mohl vůbec odpovídat, je nutné, aby na něm běžel démon[6] Telnetu, který reaguje na požadavky klienta. Jedná se tedy o komunikaci typu klient-server.

Příkazy pro spuštění vzdáleného připojení (MS Windows):

- implicitní port (23): *telnet <host>*
- jiný než výchozí port: *telnet <host> <port>*

Během běžícího spojení je možné zadávat různé příkazy, které dovolují s tímto spojením pracovat. Mezi tyto příkazy patří například *status* pro vypsání aktuálního stavu (klient připojen/nepřipojen), *close* se stará o ukončování daných relací Telnetu, *quit* také ukončuje, ale celý program Telnet, *display* vypisuje parametry programu Telnet, pro opuštění relace se do příkazového řádku klienta využívá kombinace kláves Ctrl+] na klávesnici a pro návrat do relace ENTER Veškeré příkazy je možné vypsat pomocí nápovědy *?/help*.

V úvodu bylo zmíněno, že Telnet je protokolem nešifrovaným, to znamená, že může dojít k odposlechnutí přenášených dat nežádoucí osobou, včetně hesel zadaných při autentizaci (přihlášení). Z tohoto důvodu je využití protokolu Telnet minimální a spíše se využívá protokol SSH, který je šifrovaný. [9], [10], [23]

## 4.7 SSH (Secure Shell)

Jedná se o opět o protokol používaný pro připojení ke vzdálenému síťovému zařízení. SSH je protokol definovaný v RFC 4253, který naslouchá na portu TCP/22. Na rozdíl od Telnetu jde o šifrovaný protokol, čímž je poskytnuto zabezpečení dat a udržení jejich integrity.

Bezpečnost dat je zvýšena také díky možnosti autentizace uživatelů. Existuje několik variant tohoto zabezpečení. Jako první bude zmíněno klasické heslo, které však musí být v šifrované podobě, aby nedošlo k odposlechnutí. Hesla nezůstávají stále stejná, uživatel má možnost je měnit.

Další variantou autentizace je veřejný klíč. Server má databázi veřejných klíčů patřících k jednotlivým uživatelům. V případě, že klient zašle veřejný klíč na server, server si tento klíč ověří se svou databází a pokud klíče odpovídají, posílá server klientovi zprávu šifrovanou tímto klíčem, jinak spojení zamítá. Pokud je schopen klient tuto zprávu rozšifrovat, zašle ji zpět na server, ten vše opět zkontroluje a pokud vše souhlasí, je klient autentizován. Existuje

[6] Program běžící na síťovém zařízení (v tomto případě serveru), který slouží k obsluze událostí (žádostí od klienta).

ještě pár variant autentizace uživatelů. Jelikož se však v této kapitole jedná pouze o přiblížení tohoto protokolu, není nutné je více rozebírat. [9], [23]

## 4.8 DHCP (Dynamic Host Configuration Protocol)

DHCP protokol slouží pro přiřazování IP adres, masek sítí, výchozích bran, adres DNS serverů a dalších parametrů jednotlivým zařízením v síti. Jak z názvu plyne, jedná se o dynamické přiřazování, ne o statické. Protokol je definován v RFC 2131, pro IPv6 v RFC 3315. Využívá port UDP/68 u klienta a UDP/67 u serveru. Opět se jedná o architekturu typu klient-server.

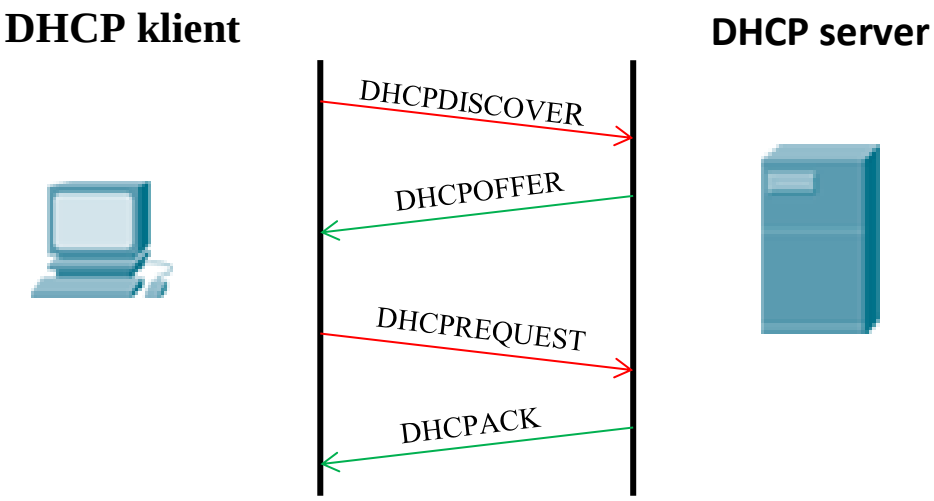
Jaký je rozdíl mezi dynamickou a statickou konfigurací? U statické musí být IP adresy a veškeré parametry zadávány ručně uživatelem, u dynamické jsou získány automaticky ihned po připojení k síti. Tyto adresy jsou přiřazovány serverem z dostupného rozsahu přiděleným poskytovatelem internetových služeb, jsou vždy unikátní a přiřazeny na omezenou dobu. Jakmile daný časový interval vyprší nebo je zařízení ze sítě odpojeno, jsou tyto adresy opět k dispozici pro jiná zařízení. V sítích se většinou vyskytuje jak dynamické, tak i statické adresování.

V domácnostech jsou IP adresy získávány prostřednictvím směrovače, který je připojen k poskytovateli internetových služeb. Jak ale probíhá samotné přiřazování těchto adres?

Po připojení zařízení (DHCP klient) do sítě je tímto zařízením nejprve vyslána broadcastová[7] zpráva (DHCPDISCOVER) na zjištění dostupných serverů. Dostupné servery odpovídají zprávou (DHCPOFFER) obsahující nabízené parametry, daná nabídka má však pouze omezené trvání. DHCP klient si mezi těmito nabídkami musí vybrat a zvolenému zařízení zašle žádost (DHCPREQUEST) o přiřazení parametrů. Pokud do oné chvíle nevypršela platnost nabídky, odpovídá DHCP server potvrzovací zprávou (DHCPACK), čímž jsou klientovi přiřazeny dané parametry. V případě, že by došlo k vypršení časového limitu nabídky nebo by byla nabídka přijata dříve jiným klientem, odpovídá klient zamítající zprávou (DHCPNAK). V tomto případě je nutné celý mechanismus opakovat již od zjištění dostupných serverů. Jak bylo uvedeno v úvodu, parametry jsou zapůjčovány na omezenou dobu, po uplynutí tohoto intervalu jsou opět nabízeny k přidělení jiným klientům. Proto je nutné, aby klient, pokud stále požaduje dynamické přidělení adres, požádal o obnovení

[7] Broadcast – zprávy jsou zaslány na všechna dostupná zařízení v síti mimo toho, odkud jsou zprávy odesílány

výpůjčky (DHCPPREQUEST). Jakmile by nedošlo k žádosti o obnovení tohoto nájmu, dané parametry by neměl více k dispozici. [10], [23], [26]




**Obrázek 20 – DHCP komunikace**

Existují dvě varianty zapůjčování adres. První je dynamické, které přiděluje adresy náhodně z přiděleného rozsahu, opakem je statické, kdy je danému zařízení na základě MAC[8] adresy zapůjčena vždy ta samá adresa.

Pro zobrazení přiřazené adresy, doby vypůjčení adresy apod. je možné využít příkaz *ipconfig /all*.

```
C:\>ipconfig /all
…
Adaptér bezdrátové sítě LAN Bezdrátové připojení k síti:
   Přípona DNS podle připojení . . . :
   Popis . . . . . . . . . . . . . . : Broadcom 802.11n Network Adapter
   Fyzická Adresa. . . . . . . . . . : 68-94-23-FD-A7-39
   Protokol DHCP povolen . . . . . . : Ano
   Automatická konfigurace povolena  : Ano
   Místní IPv6 adresa v rámci propojení . . . :
fe80::f570:b1c7:68a8:9f36%13(Preferované)
   Adresa IPv4 . . . . . . . . . . . : 192.168.0.10(Preferované)
   Maska podsítě . . . . . . . . . . : 255.255.255.0
   Zapůjčeno . . . . . . . . . . . . : 21. dubna 2014 13:29:20
   Zápůjčka vyprší . . . . . . . . . : 22. dubna 2014 13:29:21
   Výchozí brána . . . . . . . . . . : 192.168.0.1
   Server DHCP . . . . . . . . . . . : 192.168.0.1
```

[8] MAC adresa je fyzická adresa vrstvy síťového rozhraní modelu TCP/IP, jež jednoznačně identifikuje zařízení v síti.

## 4.9 FTP (File Transfer Protocol)

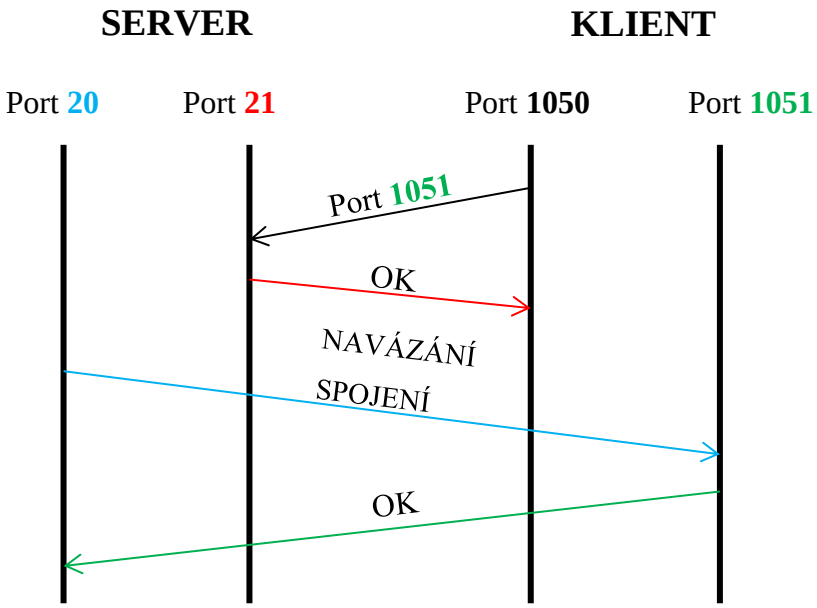
FTP je nešifrovaný protokol umožňující přenos souborů mezi zařízeními, jenž je definován v RFC 959 a bezpečnostní rozšíření v RFC 2228. Využívá protokol TCP transportní vrstvy a porty 20 a 21.

Dva porty využívá z důvodu nutnosti dvou spojení. Nejprve se utváří spojení prostřednictvím **portu 21** pro **řízení komunikace** a teprve poté přichází na řadu samotný **přenos souborů** pomocí **portu 20**. Přenos je realizován ze serveru na klienta (stažení) nebo z klienta na server (nahrání). Opět jde o komunikaci typu klient-server, kde FTP klientem je aplikace běžící na daném počítači a na serveru běží FTP démon. Jako FTP klienta lze v MS Windows využít příkazový řádek, na Internetu je však k dispozici i celá řada FTP klientů. Lze uvést například WInSCP nebo TotalCommander.

### 4.9.1 Přenosové režimy FTP

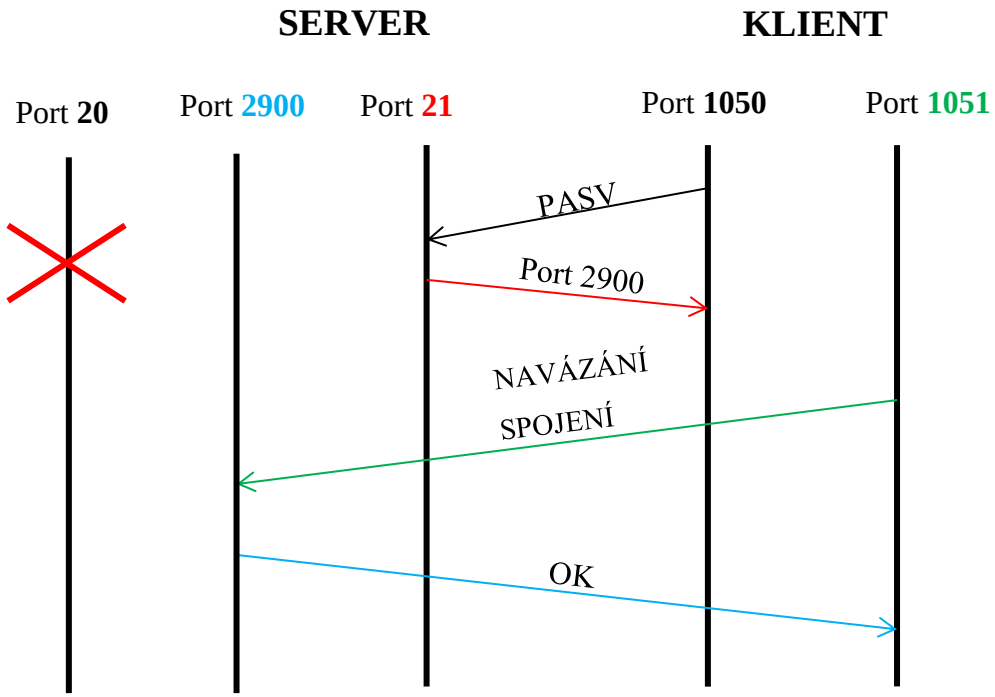
Existují dva přenosové režimy. První režim je **aktivní.** Nejprve je utvořeno řídící spojení, kde klient z portu větším než 1023 serveru sdělí číslo portu, které využije pro přenos dat (jedná se o číslo portu o 1 vyšší než port využitý pro řídící spojení). Server naslouchající na portu 21 poté potvrdí přijetí dané informace. Po řídícím spojení může být iniciováno spojení **ze strany serveru**. U aktivního režimu využívá server již zmíněný klasický port 20 a klient využívá port domluvený v řídícím spojení. Po navázání spojení může dojít k samotnému přenosu dat. U tohoto spojení se však mohou vyskytnout problémy v podobě firewallu[9], kdy klient není schopen přijmout příchozí spojení, stejně tak při použití privátní adresy u klienta nastane problém, jelikož klient není pro server viditelný. [11]

[9] Firewall je zařízení v síti, jehož úkolem je filtrovat nežádoucí komunikaci.




**Obrázek 21 – Aktivní režim FTP**

Druhou variantou je **pasivní** režim. U tohoto režimu je opět nejprve navázáno řídící spojení využitím portu 21 na straně serveru a portu větším než 1023 na straně klienta. V řídícím spojení zašle klient požadavek *PASV* na server, ve kterém žádá sdělení portu serveru, které využije pro přenos dat v pasivním módu. Hlavní rozdíl oproti aktivnímu režimu je ten, že je spojení pro přenos dat iniciováno **ze strany klienta**. Klient využije port o jedničku vyšší než port využitý v řídícím spojení a server naslouchá na portu, který byl v tomto řídícím spojení domluven. Poté, co server odpoví, že je připraven pro přenos dat, může přenos začít. Díky iniciaci spojení ze strany klienta je odstraněn problém s firewallem. [11]




**Obrázek 22 – Pasivní režim FTP**

Pro připojení k FTP serveru se používá příkaz *open* parametrem udávajícím adresu FTP serveru. Při nutnosti přihlášení je třeba zadat uživatelské jméno a heslo. Veškeré příkazy lze vypsat prostřednictvím nápovědy *help*. V následujících několika řádcích je uveden příklad připojení (*open*) ke školnímu FTP serveru z příkazového řádku MS Windows včetně ukončení spojení (*close*) a ukončení klienta FTP (*quit*). [7]

```
C:\>ftp

ftp> open st36008.fei-hosting.upceucebny.cz
Systém je připojen k st36008.fei-hosting.upceucebny.cz.
220---------- Welcome to Pure-FTPd [privsep] [TLS] ----------
220-You are user number 1 of 50 allowed.
220-Local time is now 17:03. Server port: 21.
220-This is a private system - No anonymous login
220-IPv6 connections are also welcome on this server.
220 You will be disconnected after 15 minutes of inactivity.
Uživatel (st36008.fei-hosting.upceucebny.cz:(none)): st36008
331 User st36008 OK. Password required
Heslo: není viditelné
230 OK. Current restricted directory is /
ftp> close
221-Goodbye. You uploaded 0 and downloaded 0 kbytes.
221 Logout.
ftp> quit
C:\>
```

## 4.10 TFTP (Trivial File Transfer Protocol)

TFTP je nešifrovaný protokol umožňující, stejně jako FTP, přenos souborů mezi zařízeními, je však mnohem jednodušší. Je definován v RFC 1350 a jeho rozšíření v RFC 2347. Využívá nespojový protokol UDP transportní vrstvy a port 69.

Jelikož je protokol oproti FTP zjednodušený, neumožňuje pracovat se soubory stejným způsobem jako u FTP. Nepodporuje autentizaci a je k dispozici jen pár příkazů. Při přenosu souborů je umožněno přenášet v danou chvíli vždy pouze jeden paket, jehož přijetí musí být potvrzeno. Tímto mechanismem je realizováno řízení toku dat, je tím však snížena efektivita této komunikace. [23]

## 4.11 Shrnutí protokolů aplikační vrstvy:

**Tabulka 5 – Shrnutí protokolů aplikační vrstvy**

| Název protokolu | Zkratka | Účel | RFC dokument | Protokol transportní vrstvy | Číslo portu |
|---|---|---|---|---|---|
| **Hypertext Markup Language** | **HTTP** | *webové služby* | 2616 | TCP | 80 |
| **Simple Mail Transfer Protocol** | **SMTP** | *emailové služby* | 2821 | TCP | 25 |
| **Post Office Protocol** | **POP3** | *emailové služby* | 1939 | TCP | 110 |
| **Internet Message Access Protocol** | **IMAP4** | *emailové služby* | 3501 | TCP | 143 |
| **Domain Name Systém** | **DNS** | *překlad doménových jmen na IP adresy* | 1035 | TCP i UDP | 53 |
| **TELetype NETwork service** | **Telnet** | *připojení ke vzdálenému zařízení* | 854 | TCP | 23 |
| **Secure Shell** | **SSH** | *připojení ke vzdálenému zařízení* | 4253 | TCP | 22 |
| **Dynamic Host Configuration Protocol** | **DHCP** | *přiřazování adres* | 2131, 3315 | UDP | 67 |
| **File Transfer Protocol** | **FTP** | *přenos souborů* | 959, 2228 | TCP | 20,21 |
| **Trivial File Transfer Protocol** | **TFTP** | *přenos souborů* | 1350, 2347 | UDP | 69 |

# 5 Transportní vrstva

O transportní vrstvě bylo již dříve uvedeno, že se jedná o jakýsi předěl mezi uživatelskými aplikacemi a danou sítí, jelikož skrývá implementaci samotné sítě před vrstvami na vyšší úrovni a dále, že se stará přenos dat mezi koncovými systémy. Transportní vrstva také nabízí možnost několika paralelně běžících komunikací, jež je nazývána jako **multiplexing**. Je tedy možné, aby pracovalo více aplikací najednou, to znamená, že uživatel může například prohlížet webové stránky, číst emaily a zároveň prohlížet videa. [10]

Jelikož některá data určená pro přenos sítí mohou být příliš velká, dochází na transportní vrstvě u zdrojového systému k dělení dat na jednotlivé datové jednotky transportní vrstvy, které jsou nazývány **segmenty**. Tyto segmenty jsou na cílovém systému opět složeny do původní podoby (dat). Procesu znovusložení se jinak říká také **reassembling.** Každý segment má své pořadové číslo a díky rozdělení původních dat na menší části není nutné v případě výskytu chyby přeposílat celá data, ale pouze onu poškozenou část. Každý segment je určen pro zpracování konkrétní cílovou aplikací. Aby však transportní vrstva cílového systému věděla, pro jakou aplikaci je daný segment určen, je nutné mu přidat nějaké označení. Danému označení se říká **číslo portu.** Tento pojem byl zmíněn již v předchozí kapitole týkající se aplikačních protokolů, nyní bude přiblížen podrobněji. [10], [23]

Číslo portu, nejen cílové, ale i zdrojové, je přidáno k danému segmentu v hlavičce během zapouzdření. Jak již bylo zmíněno, čísla portů jsou využívána v protokolech transportní vrstvy k identifikaci aplikačních služeb na daném počítači. Existuje možnost nastavit vlastní čísla portů pro dané služby, byl však sepsán seznam organizací IANA (ICANN), který pro určité aplikační služby stanovuje obecně známé hodnoty portů, jež jsou běžně používány. Porty nabývají hodnot od 0 do 65535, jedná se tedy o 16bitová čísla, jelikož $2^{16}$ = 65536 hodnot. Jsou rozděleny do tří kategorií:

- **Dobře známé porty** - hodnoty od 0 do 1023. Přiřazeny IANA (ICANN) pro často využívané služby. Slouží jako označení cílového portu.
- **Registrované porty** – hodnoty od 1024 do 49151. Přidělovány dynamicky jak pro cílové, tak i zdrojové porty. Použití nutno registrovat u IANA (ICANN).
- **Soukromé/dynamické porty** - hodnoty od 49152 do 65535. Dynamicky přidělená čísla zdrojových portů pro soukromé účely.

Jakmile zdrojový systém žádá o komunikaci, je dané komunikující aplikaci přiděleno číslo portu. Toto číslo je vybráno zcela náhodně, hodnota však musí překročit číslo 1023 (do 1023 označení pro dobře známé porty patřící cílovým portům). Cílový port serveru je určen použitou aplikací. Většinou na základě přidělení dobře známých portů organizace IANA, případně je použito vlastní číslo portu, o kterém však musí zdrojový systém vědět. Zdrojové číslo portu se při zpětné komunikaci stává cílovým a cílové naopak zdrojovým.

Pouze čísla portů jako označení pro komunikující aplikace by však nestačila pro jednoznačnou identifikaci specifické komunikace dvou zařízení. Pro tyto účely byly zavedeny tzv. **sockety**, které se skládají nejen z čísla portu dané aplikace, ale také IP adresy[10] daného zařízení, čímž je jednoznačně určeno, na kterém zařízení konkrétně aplikace běží. Danou komunikaci tedy identifikuje dvojice socketů, jeden patřící zdrojovému systému a druhý patřící koncovému systému. [10], [23], [26]

*Příklad socketu:*

**172.16.13.1:21**

IP adresa Číslo portu

V příkazovém řádku existuje možnost výpisu aktivních připojení, tedy aktivních socketů příkazem ***netstat***. Samotný ***netstat*** vypíše aktivní spojení, pro výpis všech aktivních spojení plus aktivních portů existuje příkaz ***netstat –a***. Všechny možné parametry spojené s příkazem ***netstat*** lze vypsat zadáním ***netstat /?*** do příkazového řádku. [7]

Na níže uvedených řádcích je uvedena část obou variant výpisu z příkazového řádku. V prvním sloupci je vypsán použitý protokol, tedy TCP nebo UDP. V dalším socket zdrojového zařízení, ve třetím sloupci cílový socket a poslední sloupec obsahuje stav daného spojení. Místo IP adresy se může ve výpisu vyskytnout také doménové jméno.

[10] IP adresa = logická adresa, jednoznačný identifikátor síťového uzlu

```
C:\>netstat
Aktivní připojení
  Proto  Místní adresa          Cizí adresa            Stav
  TCP    127.0.0.1:5905         User-PC:49165          NAVÁZÁNO
  TCP    127.0.0.1:5905         User-PC:49166          NAVÁZÁNO
  TCP    127.0.0.1:49165        User-PC:5905           NAVÁZÁNO
  TCP    127.0.0.1:49166        User-PC:5905           NAVÁZÁNO
  TCP    127.0.0.1:59278        User-PC:47986          SYN_SENT
  TCP    192.168.0.103:57355    channelproxy-shv-06-ash2:https  NAVÁZÁNO
  TCP    192.168.0.103:59069    fa-in-f94:https        NAVÁZÁNO
  TCP    192.168.0.103:59181    edge-star-shv-12-frc1:https  NAVÁZÁNO

C:\>netstat -a
Aktivní připojení
 Proto  Místní adresa          Cizí adresa            Stav
 TCP    0.0.0.0:49155          User-PC:0              NASLOUCHÁNÍ
 TCP    127.0.0.1:2559         User-PC:0              NASLOUCHÁNÍ
 TCP    127.0.0.1:5905         User-PC:49165          NAVÁZÁNO
 TCP    127.0.0.1:8307         User-PC:0              NASLOUCHÁNÍ
 TCP    127.0.0.1:49165        User-PC:5905           NAVÁZÁNO
 UDP    0.0.0.0:5355           *:*
 UDP    0.0.0.0:51337          *:*
```

Spojení se může nacházet v několika stavech:

- NASLOUCHÁNÍ – místní zařízení je připraveno na spojení a čeká na žádost od cizího zařízení.
- NAVÁZÁNO – stav, kdy dochází k přenosu dat, spojení je již utvořeno.
- TIME_WAIT – místní zařízení požádalo o ukončení spojení, v tomto stavu zůstává určitý časový interval (běžně 30 až 120 sekund), poté je spojení ukončeno.
- CLOSE_WAIT – byly vyslány požadavky na ukončení, vzdálené zařízení spojení uzavřelo, ale nedostalo potvrzení od místního zařízení.
- SYN_SENT – místní zařízení poslalo požadavek na spojení a čeká na potvrzení od vzdáleného zařízení. V tomto stavu by se socket neměl nacházet příliš dlouho.
- SYN_RECEIVED – místní zařízení přijalo požadavek na spojení.
- FIN_WAIT_1 – čekání na potvrzení přijetí žádosti o ukončení serverem.
- FIN_WAIT_2 – čekání na potvrzení žádosti o ukončení klientem.
- CLOSED – spojení ukončeno.

Na transportní vrstvě existují dvě varianty přenosu dat. Spolehlivé a nespolehlivé. Tyto varianty jsou dány použitým protokolem. Spolehlivé zaručí, že veškerá data dojdou do cíle nepoškozená a ve správném pořadí. Nespolehlivé nezaručuje v podstatě nic. V tomto ohledu se však modely ISO/OSI a TCP/IP poněkud rozcházejí. U ISO/OSI se pracuje pouze

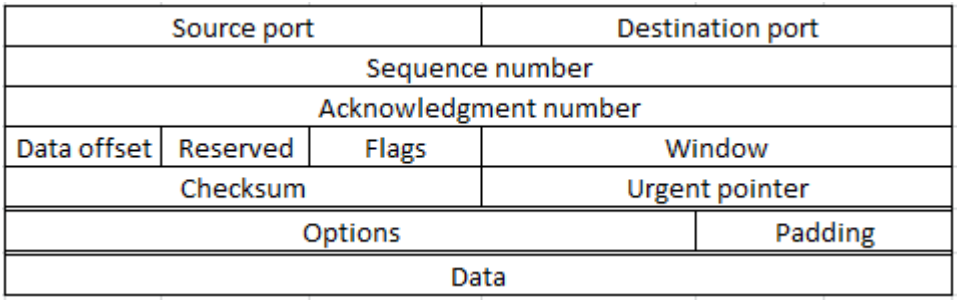
se spojovým protokolem, u TCP/IP naopak s oběma protokoly spolehlivým i nespolehlivým. Volba vhodného protokolu je pak dána požadavky, jaké má konkrétní aplikace. [24]

## 5.1 TCP (Transmission Control Protocol)

Prvním protokolem transportní vrstvy je spojově-orientovaný protokol TCP, jenž je definován v RFC 793. Spojově-orientovaný znamená, že ještě před samotným přenosem dat mezi klientem a serverem **je utvořeno spojení, konkrétně třícestné**. Díky tomuto spojení má odesílatel jistotu toho, že je dostupný příjemce dat. Dané spojení je udržováno na dobu celého trvání komunikace tak, aby mohla být spolehlivě doručena všechna požadovaná data. Protokol TCP umožňuje současný běh několika spojení (multiplexing) a u každého spojení kontroluje jeho stav, jedná se tedy o protokol **stavový**. Kontroluje, která data byla nebo nebyla přijata apod. O tom, zda byl daný segment doručen v pořádku, je odesílatel vždy příjemcem informován, **potvrdí** tedy jeho přijetí. Pokud by nedošlo k doručení daného segmentu nebo by byl porušen, odesílatel ho znovu přepošle. Segmenty nejsou doručeny vždy v tom pořadí, v jakém byly odeslány. K tomu, aby však zpráva byla čitelná, je třeba, aby byly sestaveny ve správném pořadí. Proto jsou jednotlivé segmenty, jak již bylo zmíněno v úvodu, označeny pořadovým číslem, dle kterého jsou poté řazeny a následně složeny do původní podoby (reassembling). Další důležitou vlastností protokolu TCP je **řízení toku dat**, jež může znamenat snížení ztráty segmentů. Pokud by totiž byla snaha přeposlat sítí větší množství dat najednou, než je schopna zvládnout, mohlo by dojít k jejich ztrátě, což by značně zvyšovalo potřebu znovu odeslat ztracené segmenty. Tím by se síť opět zbytečně zahlcovala. Z tohoto důvodu je protokol TCP schopen řídit množství dat procházející sítí. Možná se nyní TCP protokol zdá jako nejlepší možná volba pro jakékoliv aplikace, ve skutečnosti tomu tak však není. Díky mechanismům, které umožňují řízení toku dat, potvrzování přijetí apod., se zvyšuje režie[11], čímž je snížena i přenosová rychlost. Proto je výhodné tento protokol využít u aplikací, které tolerují zpomalení, ale rozhodně ne ztrátu dat. U protokolu TCP segmentu zabírá samotná režie 20 bajtů. [23], [24]

[11] Režie = zdroje potřebné k realizaci určitého cíle. Mezi tyto zdroje patří například potřebná paměť či časová náročnost.

### 5.1.1 TCP segment




**Obrázek 23 – TCP segment**

Na výše uvedeném obrázku (Obrázek 23) je uvedena struktura TCP segmentu.

*Význam polí segmentu:*

**Zdrojový port** (Source port) – 16bitové číslo, které uvádí zdrojový port.

**Cílový port** (Destination port) – 16bitové číslo, které uvádí cílový port pro identifikaci aplikace, které je segment určen.

**Sekvenční číslo** (Sequence number) – 32bitové pořadové číslo prvního bajtu segmentu. Číslování začíná od náhodně vygenerovaného čísla z daného rozsahu. Jelikož tento blok má 32 bitů, jde o číslo v rozmezí 0 až $2^{32}$-1. Při překročení horní hodnoty pokračuje od 0. Slouží k sestavení segmentů ve správném pořadí.

**Potvrzovací číslo** (Acknowledgment number) – 32bitové číslo, které potvrzuje přijatá data, a říká také, jaké číslo segmentu očekává pro další přijetí, jedná se tedy o dopředné potvrzování.

**Délka hlavičky** (Data offset) – 4bitové číslo. Vyjadřuje násobky 32 bitů. Indikuje, kde začínají data.

**Rezervovaný blok** (Reserved) – 6bitový blok, který je rezervován pro další použití.

K**ontrolní bity** neboli příznaky (Flags) – celkem 6 bitů, které zahrnují příznaky říkající, k jakému účelu segment slouží a jak s ním zacházet. Patří sem URG pro označení důležitosti dat, ACK potvrzuje přijatá data, RST pro resetování spojení, SYN pro utvoření spojení, FIN pro ukončení spojení, PSH říká, že segment obsahuje nějaká aplikační data.

V**elikost okna** (Window) – 16bitové číslo určující maximální počet bajtů, které smí cílový systém přijmout před potvrzením přijetí. Při překročení je segment zahozen.

**Kontrolní součet** (Checksum) – 16bitové číslo, které slouží ke kontrole toho, zda nedošlo k poškození segmentu, tedy obsažených dat i hlavičky. Pokud nesouhlasí součet, u odesílatele a příjemce, je segment také zahozen.

**Ukazatel naléhavosti** (Urgent pointer) – 16bitové pole. Je platný, pokud je nastaven kontrolní bit URG, dává přednost určitým datům. Ukazuje na první bajt dat, která jsou označena jako urgentní.

Výše uvedená pole tvoří režii nutnou pro spolehlivost protokolu TCP.

**Volitelný blok** (Options) – obsahuje volitelné položky. Toto pole obsahuje násobky 8 bitů. Patří sem například maximální délka segmentu.

**Padding –** slouží k dorovnání hlavičky na velikost dělitelnou 32 bity beze zbytku.

**Data** – data přijatá z vyšších vrstev. [23]

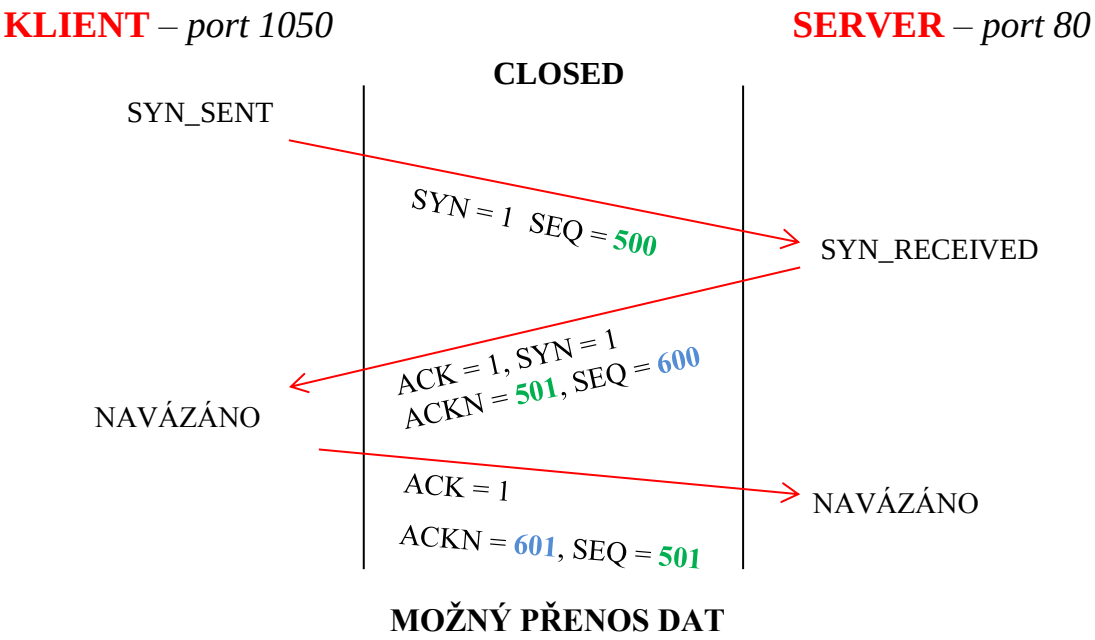
### 5.1.2 Three-way handshake

Před uvedením dalšího protokol transportní vrstvy bude znázorněn průběh navázání **třícestného spojení** mezi koncovými systémy, tedy klientem a serverem. Toto spojení se skládá ze tří následujících kroků (Obrázek 24):

1. Neexistuje žádné spojení, je tedy CLOSED. Klient požaduje zobrazení webové stránky, a proto vyžaduje spojení se serverem poskytujícím webové služby, který naslouchá na portu 80. Klientovi je zdrojový port vygenerován na hodnotu vyšší než 1023, konkrétně 1505. Klient, který zasílá žádost o vytvoření spojení, musí mít v části hlavičky segmentu, jež obsahuje kontrolní bity, nastaven příznak SYN (na hodnotu SYN = 1). Tento segment má náhodně vygenerované sekvenční číslo SEQ (v tomto případě SEQ = 500), které zašle na cílové zařízení (server). Tímto se klient dostává do stavu SYN_SENT. Server byl do této chvíle ve stavu NASLOUCHÁNÍ. Jakmile přijme SYN signál od klienta, dostává se do stavu SYN_RECEIVED.
2. Server má nyní za úkol zaslat potvrzovací segment, jenž v kontrolních bitech hlavičky musí mít nastaven příznak ACK (na hodnotu ACK = 1) a také příkaz SYN (také

na hodnotu SYN = 1). Daný segment má opět vygenerované sekvenční číslo (v níže uvedeném případě SEQ = 600). V hlavičce segmentu je také nastaveno potvrzovací číslo ACKN. Toto číslo je nastaveno na hodnotu sekvenčního čísla přijatého od klienta zvýšeného o jedničku, tedy na hodnotu ACKN = 501. Tím server říká, že přijal segment se sekvenčním číslem SEQ = 500 a dalším očekávaným je segment s číslem SEQ = 501, proto se jedná o již zmíněné dopředné potvrzování.

3. Klient přijal od serveru potvrzení přijetí žádosti o utvoření spojení. Tímto považuje spojení za NAVÁZANÉ a posílá potvrzovací segment serveru. Opět musí být v segmentu nastaven příznak ACK , tedy na hodnotu ACK = 1, jako potvrzovací číslo je použito sekvenční číslo segmentu přijatého od serveru zvýšené o jedničku, tedy ACKN = 601. Sekvenční číslo toho potvrzovací segmentu je nastaveno na předchozí hodnotu sekvenčního čísla u klienta zvýšeného o jedničku, tedy na SEQ = 501. Server přijme potvrzení, čímž považuje spojení za NAVÁZANÉ a může tedy dojít k samotnému přenosu dat. [17]

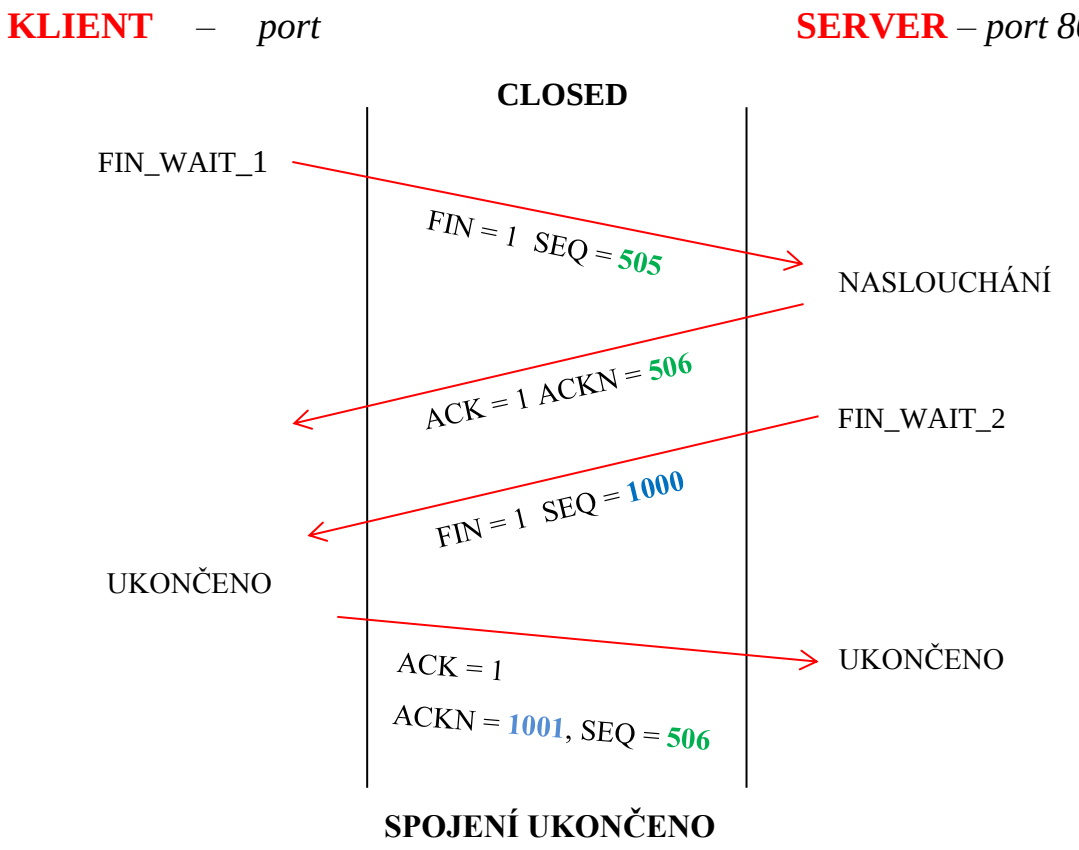



**Obrázek 24 – Navázání spojení protokolu TCP**

### 5.1.3 **Four-way handshake**

Pokud byla přenesena všechna požadovaná data a není nadále třeba udržovat spojení, dojde k jeho ukončení. Toto ukončení spojení je **čtyřcestné**, skládá se tedy z následujících čtyř kroků (Obrázek 25):

1. Klient chce ukončit spojení, z tohoto důvodu musí poslat žádost na server. K tomuto účelu musí mít v segmentu nastaven příkaz FIN na hodnotu FIN = 1. Daný segment má opět nějaké náhodně vygenerované pořadové číslo SEQ (v tomto případě SEQ = 505). Klient se z NAVÁZANÉHO stavu dostává do stavu FIN_WAIT_1.
2. Úkolem serveru je následně přijetí tohoto segmentu potvrdit. Pro tento účel musí mít nastaven příkaz ACK v hlavičce potvrzovacího segmentu. Potvrzovací číslo ACKN je v tomto segmentu nastaveno na sekvenční číslo přijaté od klienta zvýšené o jedničku, tedy na ACKN = 506. Klient již data posílat nemůže, jelikož požádal o uzavření spojení, server však může v posílání dat pokračovat do té doby, než zašle také žádost o ukončení spojení. V této fázi se jedná o tzv. polouzavřené spojení.
3. V dalším kroku server odesílá žádost o ukončení, má tedy nastaven příkaz FIN na FIN = 1. Sekvenční číslo SEQ je náhodně vygenerováno, tedy SEQ = 1000. Server se nyní nachází ve stavu FIN_WAIT_2.
4. Klient přijímá žádost o ukončení. Jeho úkolem je zaslat potvrzení o přijetí tohoto ukončovacího segmentu. Klient tedy zasílá potvrzovací segment s nastaveným příznakem ACK na hodnotu ACK = 1, jehož potvrzovací číslo má hodnotu ACKN = 1001, tedy sekvenční číslo přijaté od serveru zvýšené o jedničku. Sekvenční číslo posledního potvrzovacího segmentu je předchozí sekvenční číslo použité u klienta inkrementované o jedničku, tedy SEQ = 506. Jakmile server přijme tento potvrzovací segment, je spojení definitivně ukončeno. [23]




**Obrázek 25 – Ukončení spojení TCP**

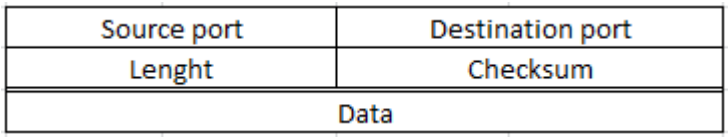
Jak již bylo zmíněno, protokol TCP umožňuje kontrolu toku dat. K tomuto účelu se používá mechanismu **posuvného okna**, který využívá velikosti okna v hlavičce segmentu. Velikost okna říká, jaké množství dat může být přeneseno mezi danými aplikacemi koncových zařízení, než bude vyžadováno potvrzení přijetí daných dat. Protokol TCP je na základě úspěšných nebo neúspěšných přijetí dat schopen regulovat velikost okna tak, aby docházelo ke ztrátám dat co nejméně. Pokud je příjemce zahlcen a segmenty nebývají doručeny správně, dojde ke snížení velikosti okna a v opačném případě dochází k jejímu zvýšení. [23]

Již v kapitole týkající se aplikační vrstvy byly uvedeny protokoly využívající služeb protokolů transportní vrstvy, mezi protokoly používající spojový protokol patří například HTTP, FTP, SSH, Telnet, SMTP a POP3.

## 5.2 UDP (User Datagram Protocol)

V případě protokolu UDP, který je definován v RFC 768, se dá říci, že se v podstatě jedná o pravý opak protokolu TCP. Je to nespojový protokol, což znamená, že před samotným přenosem dat není utvořeno žádné spojení. Z tohoto důvodu odesílatel nemá jistotu, že je dostupný příjemce, což může vést k tomu, že nejsou segmenty doručeny. U protokolu UDP neexistují žádné potvrzovací mechanismy, takže se odesílatel o této skutečnosti nedozví. Stejně tak i v případě výskytu chyby či chybném kontrolním součtu. Segmenty tedy nemohou být znovu přeposlány. Dalšími odlišnostmi od protokolu TCP jsou ty, že u UDP protokolu nedochází k řazení segmentů do původního pořadí a také nenabízí mechanismus posuvného okna, tok dat tedy není žádným způsobem řízen. Protokol UDP je opravdu jednoduchý a nenabízí příliš mnoho funkcí, což je ale jeho hlavní výhodou. Díky absenci výše uvedených mechanismů není režie tak vysoká jako u protokolu TCP. Nedochází tedy k příliš velkému zpomalení. Z toho plyne, že je tento protokol vhodnější použít u aplikací tolerujících ztrátu části dat, ale vyžadujících vysokou rychlost přenosu. Příkladem může být internetová televize či streaming[12] videa. Režie u protokolu UDP je pouze 8 B.

### 5.2.1 UDP segment




**Obrázek 26 – UDP segment**

*Význam polí segmentu:*

**Zdrojový port** (Source port) – 16bitové číslo udávající zdrojový port. Náhodně generovaný (nad 1023). Pokud není využit, je nastaven na nulu.

**Cílový port** (Destination port) – 16bitové číslo udávající cílový port. Ten je dán konkrétní aplikací, která běží na serveru. Při komunikaci ze serveru na klienta je to přesně naopak (zdrojový port odpovídá dobře známému portu přiřazenému konkrétní aplikaci a cílový port je ten náhodně vygenerovaný).

**Délka segmentu** (Length) – 16bitové číslo udávající délku segmentu v bajtech, tedy hlavičky včetně dat. Minimum je 8 bajtů (délka hlavičky, data nulová).

[12] Streaming - možnost prohlížet videa či poslouchat audio nahrávky bez nutnosti uložení do vlastního počítače, veškerá data vysílána přímo ze serveru

**Kontrolní součet** (Checksum) – 16bitové pole pro kontrolu, zda nebyl segment poškozen. **Data** – data přijatá z vyšších vrstev.

Protokol UDP je využíván některými protokoly, které byly zmíněny na transportní vrstvě. Mezi ně patří například DNS, DHCP a TFTP. [10], [23]

# 6 Síťová vrstva

Jak již bylo zmíněno, tato vrstva slouží pro přenos dat mezi nesousedními uzly v síti, jejím hlavním úkolem je najít vhodnou cestu k cílovému zařízení. Tomuto hledání cesty se jinak říká směrování, jelikož hledání cesty je úkolem routeru neboli směrovače. Každé koncové zařízení je identifikováno pomocí logické IP adresy. Zdrojová a cílová IP adresa je přidána k přijatému segmentu z transportní vrstvy ve formě hlavičky během zapouzdřování do **paketu**, což je název pro datovou jednotku na síťové vrstvě. Data zůstávají nezměněná. Každé zařízení, které má přidělenou adresu, se nazývá **host.** V hlavičce nejsou uvedeny jen IP adresy, součástí jsou další doplňkové informace, které budou uvedeny v části týkající se jednotlivých IP protokolů. Paket během své cesty do cíle projde skrz řadu mezilehlých zařízení, mezi ně patří i již zmíněné směrovače. Jakmile paket dojde správně do cíle, je třeba odebrat hlavičku, aby mohl být nyní už segment předán příslušné službě transportní vrstvy. Tomuto procesu se říká odpouzdřování. [10], [14], [23]

Na síťové vrstvě, stejně jako na jiných vrstvách, pracují nějaké protokoly. Konkrétně se jedná o protokoly IP, respektive IPv4 (Internet Protocol version 4) a IPv6 (Internet Protokol version 6), případně ne příliš využívané IPX (Novell Internetwork Packet Exchange) nebo AppleTalk.

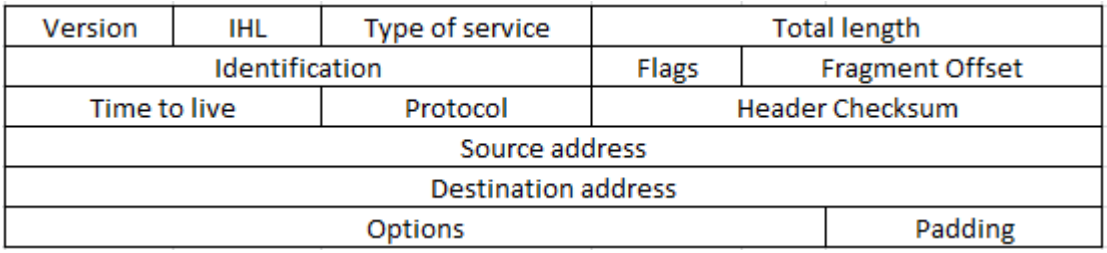
# 7 IPv4 (Internet Protocol version 4)

Tento protokol je definován v RFC 791 a patří do sady protokolů TCP/IP. Jedná se o jednoduchý protokol, který má za úkol doručit paket do cíle a nic víc. Jedná se nespojový protokol, což znamená, že před samotným přenosem dat není utvořeno spojení. Z tohoto důvodu může dojít k tomu, že paket nebude doručen, protože odesílatel nemá jistotu toho, že je koncový host dostupný. Stejně tak příjemce neví, kdy daný paket očekávat a není jisté, že bude schopen daný paket přečíst. Protokol IP je nepotvrzovaný, takže odesílatel nemá jistotu, že paket dorazil do cíle a nevyužívá žádné mechanismy pro řízení toku dat. V případě výskytu chyby tedy nedochází k opakovanému odeslání poškozeného paketu. Díky absenci těchto mechanismů se jedná o protokol s nízkou režií. Pokud by však bylo požadováno spolehlivé doručení dat, je třeba, aby se o to postaraly vrstvy jiné, konkrétně transportní vrstva využitím spojového protokolu TCP. Další vlastností tohoto protokolu je nezávislost na médiu, přes které jsou přenášena data. Nezáleží tedy na tom, zda jde o bezdrátový přenos či optické signály. Jediným omezením týkajícím se typu média je maximální velikost datové

jednotky, která může být tímto médiem přenesena. Tato velikost je charakterizována **MTU** (Maximum Transfer Unit). Pokud by byl paket přenášený sítí pro dané médium příliš veliký, neboli by byla překročena velikost MTU, je třeba ho rozdělit na více menších paketů, které je dané médium schopno přenést. Tomuto procesu dělení paketů na menší se říká **fragmentace.** [10], [23]

## 7.1 Hlavička IPv4

Nyní se zaměříme na hlavičku protokolu IPv4, její struktura je uvedena na obrázku níže (Obrázek 27).




**Obrázek 27 – Hlavička IPv4**

**Version** (4 bity) určuje typ použitého protokolu IP, pro IPv4 se konkrétně jedná o bity 0100. **Internet Header Length** (IHL) je 4bitové pole, které určuje velikost hlavičky, jež je uváděna ve velikosti 32bitových slov, minimum je 5 a maximum 15. **Differentiated Services** (dříve Type of Service) je 8bitové pole sloužící k určení priority paketů. Pole je složené z **Differentiated Services Code Point** (6 bitů) a **Explicit Congestion Notification** (2 bity). Prvních šest bitů je hodnota používána QoS a poslední dva bity slouží k zabránění zahození paketu v případě zahlcení sítě. **Total length** (16 bitů) udává velikost celého paketu v bytech, tedy hlavičky včetně dat. Minimum je 20 bytů (pouze hlavička), maximum 65535 bytů. V případě nutnosti fragmentace paketu, je třeba zajistit, že všechny fragmenty budou následně správně složeny do původního paketu. K tomuto slouží pole **Identification** (16 bitů), což je unikátní identifikátor fragmentu původního paketu. **Flags** (3 bity) určuje typ fragmentace. **Fragment Offset** (13 bitů) identifikuje pořadí, ve kterém má být fragment umístěn do původního paketu během složení do původní podoby. **Time-To-Live** (TTL) je 8bitové pole, které určuje životnost paketu. Určuje maximální počet přeskoků paketu (výběr cesty a následný přenos paketu na další směrovač na cestě do cíle), než bude směrovačem zahozen. Životnost má vždy nějakou výchozí hodnotu a při každém přeskoku je tato hodnota snížena

o jedničku. Jakmile dosáhne nuly, je tento paket zahozen a je odesílateli zaslána ICMP[13] (Internet Control Message Protocol) zpráva o překročení životnosti. Pokud by neexistovalo omezení životnosti paketu, mohlo by dojít zacyklení paketu v síti, tedy k jeho nekonečnému putování. **Protocol** (8 bitů) určuje, kterým protokolem vyšší (transportní vrstvy), bude paket po odpouzdření na segment zpracován. **Header checksum** (16 bitů) slouží ke kontrole neporušenosti hlavičky paketu. Po přijetí paketu koncovým zařízením je kontrolní součet přepočítán a v případě, že neodpovídá hodnotě kontrolního součtu v hlavičce, je paket zahozen. Dalšími dvěma bloky jsou 32bitové adresy, jedna je **zdrojová IP adresa** (Source IP Adress) a druhá je 32bitová **cílová adresa** (Destination Adress) paketu. Tyto adresy slouží k určení odesílatele a příjemce paketu. Dalšími dvěma nepovinnými bloky jsou **Options**, které slouží k rozšíření informací pro přenos paketu. Příkladem mohou být bezpečnostní omezení, možnost sledování cesty paketu. **Padding** slouží k zarovnání velikosti hlavičky na velikost celých 32bitových slov.

Paket IPv4 se však neskládá jen z hlavičky, zbytek paketu je složen z původních dat a hlavičky přidané na transportní vrstvě, to znamená, že je tvořen daným segmentem.

Jak již bylo zmíněno, IP adresování je důležitou složkou pro hledání vhodné cesty paketů do cíle. Adresy protokolu IPv4 jsou 32bitové. Je tedy poskytnut pouze omezený adresní prostor ($2^{32} = 4{,}2 \times 10^9$ adres). Tyto adresy bývají reprezentovány buď v binárním tvaru, tedy ve tvaru nul a jedniček, který je srozumitelný pro počítače, nebo v decimálním tvaru po jednotlivých bytech oddělených tečkou. Decimální tvar je využívanější, jelikož je lépe zapamatovatelný. Decimální tvar je možné poté převést na binární a naopak. [23]

[13] ICMP je protokol, který umožňuje posílat zprávy o výskytu chyb v síti

## 7.2 Převody mezi soustavami

Příkladem adresy v binárním tvaru může být například **00001010 00001010 00000001 00000011**. Nyní bude uveden způsob, jakým se adresy mezi sebou převádí.

oktet

### 7.2.1 Binární do decimálního tvaru

Adresu v binárním tvaru máme rozdělenou na jednotlivé byty (oktety), tedy cifry po osmi bitech (32 bitů / 8 = 4 byty)[14]. Binární soustava má jako základ číslo dvě. Každý bit zprava reprezentuje zvyšující se mocniny čísla dvě a jednotlivé nuly a jedničky říkají, kolikrát se daná mocnina v daném čísle vyskytuje. Součet těchto hodnot dává dohromady číslo 255, což je maximální hodnota v daném oktetu, která odpovídá binárnímu tvaru 11111111.

- $2^7 = 128$
- $2^6 = 64$
- $2^5 = 32$
- $2^4 = 16$
- $2^3 = 8$
- $2^2 = 4$
- $2^1 = 2$
- $2^0 = 1$

1) První oktet, tedy binární číslo 00001010 do decimálního tvaru bude převedeno takto:
   $0 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 0 + 0 + 0 + 0 + 8 + 0 + 2 + 0 = 10$
2) Druhý oktet 00001010 je shodný jako předchozí, tedy odpovídá číslu 10 v decimálním tvaru.
3) Třetím oktetem je 00000001. Ten bude převeden takto:
   $0 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 = 1$
4) Poslední čtvrtý oktet 00000011 je převede takto:
5) $0 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 0 + 0 + 0 + 0 + 0 + 0 + 2 + 1 = 3$
6) Výsledná adresa v decimálním tvaru vznikne oddělením těchto čísel tečkami, tedy **10.10.1.3**.

[14] 1 byte odpovídá 8 bitům

### 7.2.2 **Decimální do binárního tvaru**

Nyní bude ukázán opačný převod z decimálního tvaru **10.10.1.3** na binární. Pracuje se opět s těmito mocninami.

- $2^7 = 128$
- $2^6 = 64$
- $2^5 = 32$
- $2^4 = 16$
- $2^3 = 8$
- $2^2 = 4$
- $2^1 = 2$
- $2^0 = 1$

Začíná se odleva. Prvním krokem je porovnat první decimální číslo s nejvyšší mocninou, tedy 128. Pokud by byla hodnota větší nebo rovna 128, na tuto pozici v binárním čísle by byla zapsána 1 a od decimálního čísla odečtena mocnina, tedy 128. Pokud je decimální číslo menší, je zapsána 0. Stejným způsobem se pokračuje s mocninou nižší, tedy 64, poté s 32, 16, 8, 4, 2 a 1.

Prvním převáděným číslem je číslo 10.

1) 10 >= 128 ? ne -> 0
2) 10 >= 64 ? ne -> 0
3) 10 >= 32 ? ne -> 0
4) 10 >= 16 ? ne -> 0
5) 10 >= 8 ? ano -> 1……je tedy odečteno 10 – 8 = 2
6) 2 >= 4 ? ne -> 0
7) 2 >= 2 ? ano -> 1……....odečteno 2 – 2 = 0
8) 0 >= 1 ? ne -> 0
9) Výsledné binární číslo odpovídající číslu 10 v decimálním tvaru je 00001010.

Dalším číslem v decimálním tvaru adresy je opět číslo 10. Binární tvar je proto stejný jako v předchozím kroku 00001010.

Následuje však číslo 1. Jeho převod do binárního tvaru vypadá takto:

1) 1 >= 128 ? ne -> 0
2) 1 >= 64 ? ne -> 0
3) 1 >= 32 ? ne -> 0
4) 1 >= 16 ? ne -> 0
5) 1 >= 8 ? ne -> 0
6) 1 >= 4 ? ne -> 0
7) 1 >= 2 ? ne -> 0
8) 1 >= 1 ? ano -> 1……...odečteno 1 – 1 = 0
9) Výsledkem je číslo 000000001.

Nyní bude převedeno číslo 3.

1) 3 >= 128 ? ne -> 0
2) 3 >= 64 ? ne -> 0
3) 3 >= 32 ? ne -> 0
4) 3 >= 16 ? ne -> 0
5) 3 >= 8 ? ne -> 0
6) 3 >= 4 ? ne -> 0
7) 3 >= 2 ? ano -> 1……...odečteno 3 – 2 = 1
8) 1 >= 1 ? ano -> 1……...odečteno 1 – 1 = 0
9) Výsledkem je číslo 000000011.

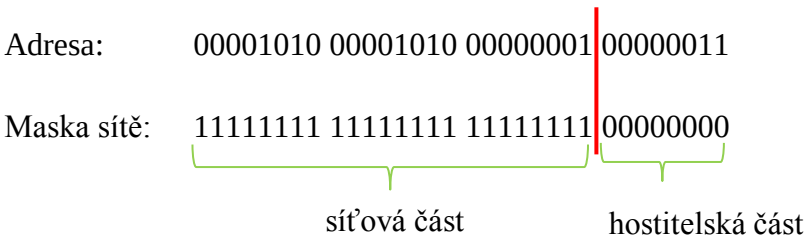
Po převodu do binárního tvaru má tedy adresa tvar **00001010 00001010 00000001 00000011**, lze si všimnout, že binární tvar odpovídá tvaru na začátku kapitoly týkající se převodu adres.

## 7.3 Tvar IPv4 adresy

IP adresy mají hierarchickou strukturu. Adresa IPv4 se skládá ze dvou částí. První je **část síťová** a druhá je **hostitelská část**. Síťová část je společná pro všechna zařízení v rámci jedné sítě a hostitelská je pro každé zařízení v dané síti unikátní. Pro rozlišení, která část je síťová a která hostitelská, je využívána **maska sítě**. Maska sítě je stejně jako IPv4 adresa 32bitové číslo, které může být uváděno jak v binárním, tak i decimálním tvaru. Pro určení síťové a hostitelské části je však třeba IP adresu i masku převést do binárního tvaru a porovnat je

zleva bit po bitu. Bity s hodnotou 1 v masce sítě určují síťovou část a bity s hodnotou 0 část hostitelskou. [10]

K dispozici je adresa 10.10.1.3 s maskou 255.255.255.0. Maska je někdy také udávána ve tvaru délky prefixu za lomítkem jako decimální číslo, které odpovídá počtu 1 v masce, tedy 10.10.1.3/24. Určení síťové a hostitelské části adresy je provedeno takto.




Maska sítě slouží nejen k určení síťové a hostitelské části, ale také říká, jaký je maximální počet hostů v dané síti. Tento počet je vždy určen jako číslo dvě umocněné na počet nul v masce sítě. Jelikož je první adresa adresou sítě a poslední adresa je přiřazena pro broadcast[15], je tento počet snížen o číslo dvě. [4], [10]

[15] Broadcast je adresa všesměrového vysílání, po vyslání zprávy na broadcast ji přijmou všechna zařízení v dané síti.

## 7.4 Příklady na výpočet počtu hostitelských adres

Konkrétně pro adresu **10.10.1.3** s prefixem **/24** je počet hostů spočítán takto:

1) Maska se skládá z 32 bitů, z toho 24 je nastaveno na jedničku, zbývá tedy osm nul.
2) Maska: 11111111 11111111 11111111 00000000
   Adresa: 00001010 00001010 00000001 00000011
3) $2^8 = 256$ adres

První adresa je vždy adresa sítě, v binárním tvaru hostitelské části má pouze samé nuly. Poslední adresou je broadcast, ten naopak má v binárním tvaru v hostitelské části samé jedničky.

4) Adresa sítě: 00001010 00001010 00000001 00000000
   Adresa sítě: 10.10.1.0
5) Adresa broadcastu: 00001010 00001010 00000001 11111111
   Broadcast: 10.10.1.255
6) $2^8 - 2 = 256 - 2 = 254$
   Po odečtení těchto adres zbývá **254 adres** pro hosty.
   První adresa dostupná pro hosty je 10.10.1.1.
   Poslední adresa dostupná pro hosty je 10.10.1.254.

Dalším příkladem je například hostitelská adresa **10.10.1.3** s prefixem **/27**.

1) Maska se skládá z 32 bitů, z toho 27 je nastaveno na jedničku, zbývá tedy pět nul.
2) Maska: 11111111 11111111 11111111 11100000
   Adresa: 00001010 00001010 00000001 00000011
3) $2^5 = 32$ adres
4) Adresa sítě: 00001010 00001010 00000001 00000000
   Adresa sítě: 10.10.1.0
5) Adresa broadcastu: 00001010 00001010 00000001 00011111
   Broadcast: 10.10.1.31
6) $32 - 2 = 30$
   Po odečtení těchto adres zbývá **30 adres** pro hosty.
   První adresa dostupná pro hosty je 10.10.1.1.
   Poslední adresa dostupná pro hosty je 10.10.1.30.

V předchozích příkladech bylo uvedeno, že adresa sítě má v hostitelské části samé nuly. **Adresu sítě** lze také zjistit pomocí logického bitové **operace AND** mezi maskou sítě a adresou daného hosta.

- 1 AND 1 = 1
- 0 AND 0 = 0
- 1 AND 0 = 0
- 0 AND 1 = 0

Pro adresu hosta 10.10.1.3/27 je adresa sítě spočítána takto:

1) Maska: 11111111 11111111 11111111 11100000 AND
   Adresa: 00001010 00001010 00000001 00000011
   00001010 00001010 00000001 00000000
   hostitelská část
2) Po převedení binárního tvaru do decimálního je výsledkem adresa sítě **10.10.1.0**.

Díky operaci bitového logického součinu je možné určit, zda zařízení, mezi kterými má být doručen paket, spadají do stejné sítě.

Podobně jako adresa sítě se dá spočítat i **adresa broadcastu**. Tato adresa v hostitelské části samé jedničky. Nevyužívá se však operace bitového logického součinu, ale operace logického bitového součtu neboli bitové **operace OR**. Rozdíl je ten, že se nejedná o operaci OR mezi adresou hosta a maskou sítě, ale jedná se o operaci OR mezi adresou hosta a tzv. **inverzní maskou sítě**. Inverzní maska sítě má převrácené hodnoty bitů oproti původní masce, to znamená, že na místě 1 je 0 a naopak. [4]

- 1 OR 1 = 1
- 0 OR 0 = 0
- 1 OR 0 = 1
- 0 OR 1 = 1

Pro adresu hosta 10.10.1.3/27 je adresa broadcastu spočítána takto:

1) Maska: 11111111 11111111 11111111 11100000
   Inverzní maska: 00000000 00000000 00000000 00011111
2) Inverzní maska: 00000000 00000000 00000000 00011111 OR
   Adresa: 00001010 00001010 00000001 00000011
   00001010 00001010 00000001 00011111
   hostitelská část

3) Po převedení binárního tvaru do decimálního je výsledkem adresa broadcastu **10.10.1.31**.

## 7.5 Statické a dynamické adresování

Adresování v síti se dělí na dvě kategorie, konkrétně statické a dynamické adresování. U statického adresování je IP adresa, maska sítě a výchozí brána nastavena ručně administrátorem z přiděleného rozsahu adres. U dynamického adresování jsou tyto informace naopak přiřazeny automaticky z přiřazeného adresního prostoru prostřednictvím protokolu DHCP. Tomuto přiřazenému rozsahu se jinak říká bazén adres. Oba způsoby mají své výhody i nevýhody.

Statické adresování se využívá spíše u zařízení, která často nemění umístění a u kterých by měnící se adresa činila problémy, zejména u serverů. Administrátor mi musí uchovávat informace o přiřazených informacích, aby byl schopen se v dané síti orientovat a pokud je vyžadována změna, vše opět ručně nastavit. Pokud je síť rozsáhlá, je statická adresace časově náročná a může odcházet k výskytu chyb. Nikdo totiž není neomylný a čísla se snadno spletou. Výhodou však je možnost nastavení povolení přístupu k síti na základě přiřazených adres.

Dynamické adresování se doporučuje zvolit zejména u rozsáhlých sítí z důvodu omezení výskytu chyb a menší časové náročnosti oproti předchozí variantě. Jelikož je adresa přiřazena vždy na určitou dobu a poté je vrácena do bazénu adres, je dynamické adresování výhodnější použít zejména u zařízení, která se do sítě připojují na omezenou dobu a často mění své umístění. Mezi ně patří mobilní zařízení jako notebooky nebo chytré telefony. V případě nutnosti si o přiřazení adresy mohou znovu zažádat, tato problematika byla však již probrána v kapitole týkající se aplikačního protokolu DHCP.

Rozsah dostupných adres si administrátor neurčí sám. Pro tento účel existují konkrétní organizace, které se o přidělování adres starají. Nejvyšší organizací je již zmíněná organizace IANA, nyní ICANN. Ta rozděluje adresní prostor mezi regionální registrátory (RIR). Pro každý kontinent existuje vlastní regionální registrátor, který má na starost rozdělit adresní prostor přidělený od ICANN mezi lokální registrátory (LIR). Mezi ně patří i například poskytovatelé internetových služeb (ISP). Tito poskytovatelé poté rozdělují adresní prostor mezi své zákazníky, tedy jednotlivé firmy či další uživatele. [9], [10], [23], [26]

## 7.6 Konfigurace adres IPv4 na rozhraní směrovače

V globálním konfiguračním režimu je nutné nejprve zadat rozhraní, kterému bude přiřazena adresa, v tomto případě se jedná o rozhraní s označením fa0/0. Příkaz je ve tvaru *interface <typ rozhraní> <označení>*.

```
R1(config)#interface fastethernet 0/0
```

Po přepnutí na konfiguraci rozhraní je možno zadat požadovanou IP adresu příkazem ve tvaru *ip address <adresa sítě> <maska sítě>*.

```
R1(config-if)#ip address 192.168.10.1 255.255.255.0
```

Aby bylo rozhraní v provozu, je nutné ho zapnout příkazem *no shutdown*.

```
R1(config-if)#no shutdown
```

Po dokončení konfigurace rozhraní je možné se navrátit do globálního konfiguračního režimu příkazem *exit*.

```
R1(config-if)#exit
```

Pro výpis konfigurace rozhraní lze v privilegovaném uživatelském režimu zadat příkaz *show ip interface brief*. Ve výpisu je název rozhraní, jeho stav a přiřazená adresa.

```
R1#show ip interface brief
```

[7]

## 7.7 Rozdělení IPv4 adres

Již dříve bylo zmíněno, že zdrojová i cílová IP adresa je zapouzdřena v hlavičce paketu. Cílové adresy jsou na základě počtu hostů, pro které je paket určen, rozděleny do tří kategorií a na základě nich se odlišují i typy komunikace.

Prvním typem komunikace je **unicastová**. Tento pojem je používán v případě, že se jedná o klasickou komunikaci pouze mezi dvěma hosty. Zdrojová adresa v hlavičce paketu je adresa odesílatele a jako cílová adresa je adresa pouze jednoho konkrétního zařízení, ne více.

Dalším typem je **broadcastová** komunikace neboli všesměrové vysílání. Tento název je použit pro situaci, kdy jeden host zasílá paket všem hostům v síti. Pro tento účel využívá jako

cílovou adresu již zmíněnou adresu broadcastu a zdrojová adresa je adresa odesílatele. Jsou rozlišovány dva typy, směrovatelný a omezený broadcast. Směrovatelný broadcast se využívá v případě, že host chce zaslat paket i mimo lokální síť. Adresa broadcastu má v hostitelské části samé jedničky a je zapouzdřena v hlavičce paketu jako cílová adresa. Příkladem je pro síť s adresou 192.168.10.0/25 adresa broadcastu 192.168.10.127. V tomto případě je paket zpracován i routerem tvořícím hranici mezi jednotlivými sítěmi a případně zaslány mimo lokální síť. Omezený broadcast je použit v případě zaslání paketu pouze v rámci lokální sítě. Pro tento účel je využita adresa broadcastu 255.255.255.255, která je opět uvedena jako cílová adresa v hlavičce paketu. Paket zaslaný na tuto adresu není směrovačem posílán pryč z lokální sítě, směrovač tím tedy tvoří hranice tzv. **broadcastové domény**, což není nic jiného než daná lokální síť. Příkladem využití tohoto způsobu komunikace je například u dynamického adresování protokolem DHCP. Na broadcastovou adresu jsou totiž zasílány žádosti klienta o přiřazení IP adresy.

Třetím a posledním typem je **multicastová** komunikace. V tomto případě je paket zaslán z hosta vybrané skupině hostů v síti, tzv. multicastové skupině. Zdrojová adresa je adresa odesílatele a cílová adresa je multicastová adresa určující skupinu příjemců. Tato varianta může být využita například při žádosti určité skupiny hostů o připojení k nějaké hře. Pro multicastové adresy je rezervován specifický rozsah od 224.0.0.0 do 239.255.255.255, který se dále dělí na podskupiny.

- **Rezervované** spadají do rozsahu 224.0.0.0 až 224.0.0255. Jsou využívány pouze v rámci lokální sítě, směrovač tedy pakety určené pro tyto adresy nezašle mimo lokální síť. Jsou využívány zejména zmíněnými směrovacími protokoly pro výměnu důležitých informací.
- **Veřejné** mají rozsah 224.0.1.0 až 238.255.255.255. Slouží pro multicastovou komunikaci i mimo lokální síť, tedy i prostřednictvím Internetu.
- **Administrativní** jsou z rozsahu 239.0.0.0 až 239.255.255.255. Tento rozsah je použit pro soukromé využití.

Je důležité si uvědomit, že zdrojová adresa zapouzdřená v hlavičce paketu je vždy adresa jednoho konkrétního hosta, který paket odesílá. Kdežto cílová adresa může být jak adresa jednoho konkrétního hosta, tak i určité skupiny či všech hostů v síti.

IP adresy nejsou rozděleny jen na multicastové, unicastové a broadcastová. Existuje několik typů adres.

První dva typy jsou rozděleny na základě toho, zda jsou či nejsou, určeny pro adresaci v rámci Internetu, tedy veřejné globální sítě.

**Veřejné adresy** jsou přiřazovány hostům, kteří mají být dostupní v rámci sítě Internet. Naopak privátní adresy jsou využívány pouze v rámci soukromé sítě. Díky tomuto je možné, aby v rámci různých soukromých sítí byly přiřazeny hostům stejné adresy, které ale v rámci dané konkrétní sítě musí být unikátní. Privátní adresy z tohoto důvodu značně šetří adresní prostor. Pokud by se host z privátní sítě chtěl připojit k síti Internet, bylo by třeba přeložit jeho privátní adresu na unikátní veřejnou adresu, jež je určena pro komunikaci v rámci Internetu. K tomu účelu je využívána IP maškaráda prostřednictvím protokolu NAT (Network Adress Translator), který nejen překládá privátní adresy na unikátní veřejné, ale i naopak. Pro **privátní adresy** byl vyhrazen specifický adresní prostor.

- 10.0.0.0 až 10.255.255.255, což odpovídá adrese 10.0.0.0/8
- 172.16.0.0 až 172.31.255.255, což odpovídá adrese 172.16.0.0/12
- 192.168.0.0 až 192.168.255.255, což odpovídá adrese 192.168.0.0/16

Dalším speciálním typem IP adresy jsou **loopback** adresy, které slouží hostům k posílání paketů sobě samotným. Pro tento účel by vyhrazen adresní prostor od 127.0.0.0 do 127.255.255.255. Nejčastěji je však využívána adresa 127.0.0.1.

Pokud nebyla hostovi přiřazena adresa, tedy staticky ani dynamicky, je možnost využít tzv. automatické konfigurace, což znamená, že operační systém danému hostovi sám automaticky přiřadí IP adresu z rozsahu 169.254.0.0 až 169.254.255.255. Opět se ale jedná pouze o komunikaci pouze v rámci lokální sítě.

Rozsah adres 192.0.2.0 až 192.0.2.255 je určen pro vzdělávací účely a rozsah 240.0.0.0 až 255.255.255.254 je určen pro další použití, konkrétně pro experimentální účely, není použitelný v IPv4 síti.

Dále jsou adresy děleny na základě tříd, které jsou uvedeny v tabulce (Tabulka 6, Tabulka 7). Jedná se o **classfull** (plnotřídní) **adresování.**

**Tabulka 6 – Třídy IPv4 adres**

| Třída | První bity | Prefix | Maska desítkově | Počet bitů síťové části | Počet bitů hostitelské části | Počet sítí | Počet hostů v síti |
|---|---|---|---|---|---|---|---|
| A | 0 | /8 | 255.0.0.0 | 8 - 1 = 7 | 24 | $2^7 = 128$ | $2^{24} - 2 = = 16777214$ |
| B | 10 | /6 | 255.255.0.0 | 16 - 2 = = 14 | 16 | $2^{14} = 16384$ | $2^{16} - 2 = 65534$ |
| C | 110 | /24 | 255.255.255.0 | 24 - 3 = = 21 | 8 | $2^{21} = 2097152$ | $2^8 - 2 = 254$ |
| D | 1110 | multicastové adresy | | | | | |
| E | 1111 | adresy pro experimentální účely a další speciální využití | | | | | |

**Tabulka 7 – Rozsahy adres jednotlivých tříd**

| Třída | První bity | Rozsah prvního bytu | První adresa | Poslední adresa | Prefix | Maska desítkově |
|---|---|---|---|---|---|---|
| A | 0 | 0 - 127 | 0.0.0.0 | 127.255.255.255 | /8 | 255.0.0.0 |
| B | 10 | 128 - 191 | 128.0.0.0 | 191.255.255.255 | /6 | 255.255.0.0 |
| C | 110 | 192 - 223 | 192.0.0.0 | 223.255.255.255 | /24 | 255.255.255.0 |
| D | 1110 | 224 - 239 | 224.0.0.0 | 239.255.255.255 | - | - |
| E | 1111 | 240 - 255 | 240.0.0.0 | 255.255.255.255 | - | - |

Tento způsob dělení adres je však neefektivní a zastaralý, jelikož je plýtváno adresním prostorem. Z tohoto důvodu bylo zavedeno **classless** (neplnotřídní) **adresování** neboli CIDR (Classless Inter-Domain Routing). Které umožňuje pracovat s libovolnou maskou sítě, ne pouze s těmi třídními. Jako názorný příklad by mohla posloužit síť, kde má být naadresováno 300 hostů. V tomto případě by rozsah třídy C s prefixem /24 nestačil, jelikož v jedné síti je v dispozici pouze 254 adres. Z tohoto důvodu je třeba využít adresy třídy B s prefixem /16, kde je k dispozici v každé síti 65534 adres. Je však zřejmé, že dostupný rozsah adres je zbytečně veliký, jelikož bylo potřeba pouze 300 adres. Classless adresace umožňuje využít rozsahu s prefixem /23, kde je k dispozici $2^9$ - 2= 510 adres pro hosty, což značně snižuje plýtvání s adresami. [9], [10], [21], [23]

# 8 IPv6 (Internet Protocol version 6)

V předchozím textu bylo již uvedeno, že protokol IPv4 poskytuje 32bitové adresy, což ale znamená, že je k dispozici pouze $2^{32} = 4{,}29 \times 10^9$ adres. Tento počet je však pouze teoretický, některé adresy jsou totiž přiřazeny pro specifické účely a proto jich je pro hosty k dispozici mnohem méně. Počet zařízení, která vyžadují přiřazení IP adresy, stále roste, a proto tento počet není dostačující. Pro zajímavost, poslední dva volné adresní prostory s prefixem /8 byly alokovány společností IANA (Internet Assigned Numbers Authority) v roce 2011 pro společnost RIR[16] (Regional Internet Registry). Z tohoto důvodu byl zaveden protokol IPv6, který značně zvyšuje počet dostupných adres. Jedná se totiž o protokol, který poskytuje **128bitové adresy**, je tedy k dispozici $2^{128} = 3{,}4 \times 10^{38}$ adres. Pro představu tento počet prý odpovídá přibližně počtu zrnek písku na Zemi.

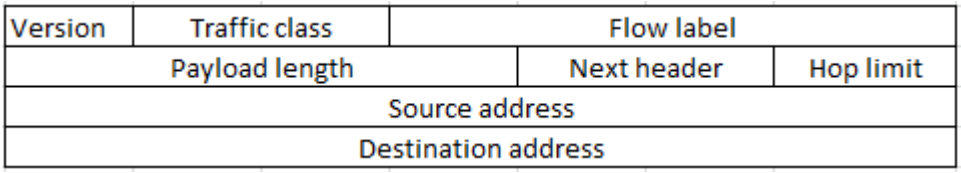
K tomu, aby mohl být protokol IPv6 zaveden a koexistovat s protokolem IPv4, bylo třeba vytvořit určité nástroje, které umožní přechod mezi těmito dvěma protokoly. Prvním nástrojem je **Dual Stack**, který umožní, aby host používal zároveň IPv4 i IPv6 adresu. **Tunelování** umožňuje průchod paketu s IPv6 hlavičkou skrz IPv4 síť tím, že zapouzdří IPv6 paket do IPv4 paketu. Dalším nástrojem je **NAT64** (Network Address Translation 64), který překládá IPv6 pakety na IPv4 a obráceně.

IPv6 má oproti IPv4 řadu výhod. Pakety IPv6 mají jednodušší hlavičku, což usnadňuje manipulaci s nimi. Dále se liší od IPv4 tím, že není nutné využití IP maškarády prostřednictvím protokolu NAT (Network Adress Translator), tedy překládat privátní[17] adresy na unikátní veřejné a naopak. IPv6 totiž poskytuje dostatečný počet adres k tomu, aby všem mohly být přiřazeny veřejné adresy. Protokol IPv4 neposkytoval žádné mechanismy ověřování či ochrany osobních údajů, protokol IPv6 je poskytuje. [10], [12], [23]

[16] RIR se stará o přidělování adres na jednotlivých kontinentech. Poskytuje adresy poskytovatelům internetových služeb či organizacím.
[17] Privátní IP adresy slouží pro účely interních sítí, na Internetu se nevyskytují. Rozsahy budou uvedeny později.

## 8.1 Hlavička IPv6

Následující část bude zaměřena na IPv6 hlavičku. Jak již bylo zmíněno, je mnohem jednodušší než hlavička IPv4. Díky tomu je pro směrovač zpracování paketu IPv6 mnohem jednodušší a rychlejší.



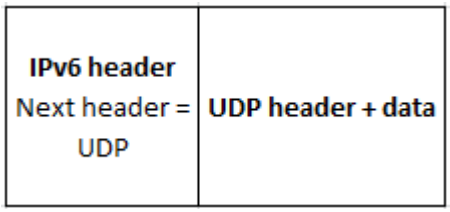

**Obrázek 28 – Hlavička IPv6**

Prvním polem v hlavičce IPv6 je **Version** (4 bity), které říká, jaká je verze paketu. V případě IPv6 je nastaveno vždy na 0110. **Traffic Class** (8 bitů) odpovídá poli **Differentiated Services** v IPv4 hlavičce. Prvních šest bitů tedy třídí pakety a další dva bity slouží k jejich řízení v případě přetížení. Následujícím polem v hlavičce IPv6 je **Flow Label** (20 bitů), které slouží k zachování stejné cesty pro pakety, které tvoří jeden logický celek tak, aby nedošlo k porušení jejich pořadí. Dalším polem je **Payload Length** (16 bitů), které je ekvivalentní k Total Length u IPv4, udává velikost IPv6 paketu, tedy dat včetně hlavičky. **Next Header** (8 bitů) je ekvivalentem k poli Protocol u IPv4. Určuje tedy, kterým protokolem transportní vrstvy má být přijatý segment zpracován. Případně mohou být využity nějaké rozšiřující hlavičky, jsou uvedeny právě v tomto poli. **Hop Limit** (8 bitů) odpovídá TTL poli v hlavičce IPv4 a slouží k označení životnosti paketu, aby nedocházelo k nekonečnému putování paketu sítí. Původní nastavená hodnota je při každém průchodu směrovačem snížena o jedničku, při dosažení nuly je paket zahozen a je poslána ICMPv6 zpráva odesílateli o tom, že paket nebyl doručen. Předposledním polem hlavičky IPv6 je **Source Adress** (128 bitů), kde je uvedena IPv6 adresa odesílatele, naopak **Destination Adress** (128 bitů) určuje IPv6 adresu příjemce. [6], [23]

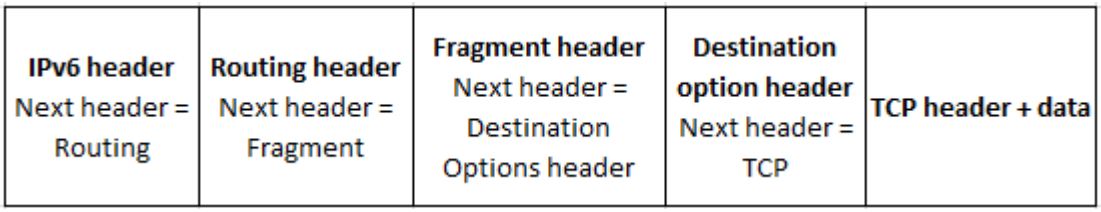
Případně mohou v IPv6 paketu existovat i **rozšiřující hlavičky**, které slouží k zabezpečení či fragmentaci. Ty jsou vždy přidávány za samotnou hlavičku IPv6, tedy mezi hlavičku IPv6 a hlavičku protokolu vyšší (transportní) vrstvy. Tyto rozšiřující hlavičky jsou identifikovány hodnotami v poli Next Header, které slouží k jejich propojení. Pořadí rozšiřujících hlaviček je striktně dáno. Nejprve jsou uvedeny hlavičky, které jsou zpracovávány směrovači a jako poslední jsou hlavičky pro zpracování koncovým hostem.

- **IPv6 header** – základní hlavička protokolu IPv6.
- **Hop-by-Hop Options Header** – volby určené pro všechny uzly na cestě do cíle.
- **Destination Options Header** – volby, pro první cílovou adresu a další adresy uvedené ve směrovací hlavičce.
- **Routing Header –** mezilehlá zařízení, která mají být navštívena na cestě do cíle.
- **Fragment Header –** umožňuje fragmentaci paketů (pouze u odesílatele).
- **Authentication Header –** umožňuje autentizaci, pro zachování integrity dat.
- **Encapsulating Security Payload header –** bezpečnostní služby.
- **Destination Options Header** – volby pouze pro koncového hosta.
- **Upper-layer Header** – hlavička protokolu transportní vrstvy.

Zřetězení hlaviček je uvedeno na obrázcích níže (Obrázek 29, Obrázek 30) [6]:




**Obrázek 29 – Hlavička IPv6 a UDP segment**



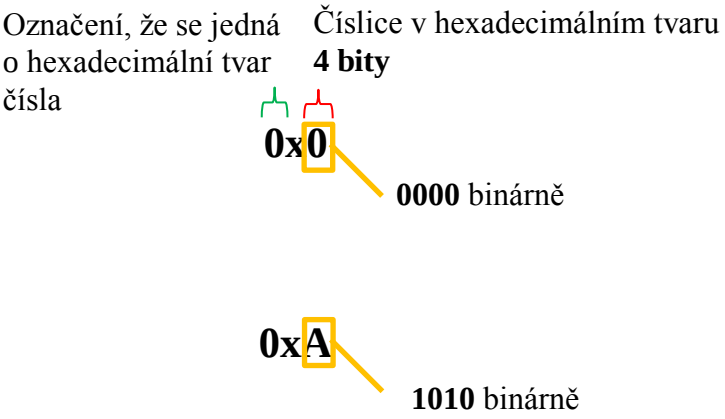

**Obrázek 30 – Zřetězení rozšiřujících hlaviček IPv6**

## 8.2 Tvar IPv6 adresy

IPv4 adresy byly ve dvou tvarech, v decimálním nebo binárním. IPv6 adresy jsou reprezentovány v hexadecimálním tvaru, tedy v šestnáctkové soustavě.

Šestnáctková soustava využívá čísla od 0 do 9, která odpovídají číslům v desítkové soustavě, a dále písmen A až F, která odpovídají číslům 10 až 15 také v desítkové soustavě. Tato čísla navíc odpovídají 4bitovým hodnotám od 0000 do 1111. Každé čtyři bity tedy odpovídají jednomu hexadecimálnímu číslu, což znamená, že jeden byte (8 bitů) je reprezentován dvěma hexadecimálními čísly. Pro označení, že se jedná o šestnáctkové číslo, se používá značka 0x. Například binární číslo 0100 1111 o velikosti 1 byte je tedy reprezentováno hexadecimálním číslem 0x4F. V tabulce níže jsou zobrazeny cifry ze šestnáctkové soustavy a jim odpovídající hodnoty v soustavě desítkové a dvojkové.

Příklad:




**Tabulka 8 – Číselné soustavy**

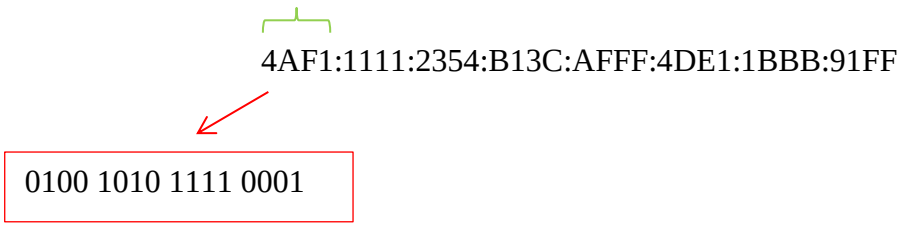
| Decimální | Hexadecimální | Binární |
|---|---|---|
| 0 | 0x0 | 0000 |
| 1 | 0x1 | 0001 |
| 2 | 0x2 | 0010 |
| 3 | 0x3 | 0011 |
| 4 | 0x4 | 0100 |
| 5 | 0x5 | 0101 |
| 6 | 0x6 | 0110 |

| Decimální | Hexadecimální | Binární |
|---|---|---|
| 7 | 0x7 | 0111 |
| 8 | 0x8 | 1000 |
| 9 | 0x9 | 1001 |
| 10 | 0xA | 1010 |
| 11 | 0xB | 1011 |
| 12 | 0xC | 1100 |
| 13 | 0xD | 1101 |
| 14 | 0xE | 1110 |
| 15 | 0xF | 1111 |

IPv6 adresa je 128bitová. Pro přehlednost je rozdělena na části po 16 bitech **(hextetech)** oddělených dvojtečkou. Každá 16bitová část odpovídá čtyřem hexadecimálním číslicím. V IPv6 adrese se označení 0x, že se jedná o číslice v šestnáctkové soustavě, nepoužívá. [6], [12]

Příkladem může být tato adresa:

16 bitů = 4 hexadecimální číslice = 1 hextet




Jak je možno vidět na příkladu výše, adresy IPv6 mohou být velice složité. Jejich zápis se však dá částečně zjednodušit a to v případě, že se v něm vyskytují nuly.

První možností je vynechání všech počátečních nul v každém 16bitovém bloku.

Příklad:

1) Původní tvar: 4AF1:0011:0300:B13C:000F:4DE1:0000:91F0
   Upravený tvar: 4AF1: 11: 300:B13C: F:4DE1: 0:91F0

Druhou možností je nahrazení jednoho nebo více 16bitových bloků samých nul dvojitými dvojtečkami. V adrese se tato dvojitá dvojtečka smí vyskytnout pouze jednou. Tato varianta se dá kombinovat s předchozí.

Příklad:

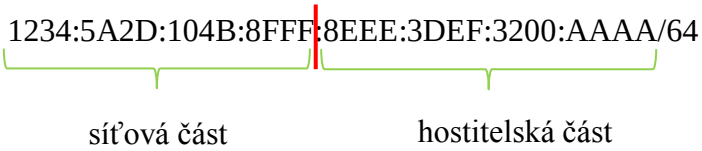
1) Původní tvar: 4AF1:0011:0000:0000:000F:4DE1:0000:91F0
   Upravený tvar: 4AF1: 11 :: F:4DE1: 0:91F0

2) Původní tvar: 4AF1:0011:0000:0000:000F:0000:0000:91F0
   Upravený tvar: 4AF1: 11 :: F: 0: 0:91F0

   nemůže být podruhé zapsáno pomocí **::**

3) Původní tvar: 0000:0000:0000:0000:0000:0000:0000:0000
   Upravený tvar: ::
4) Původní tvar: 0000:0000:0000:0000:0000:0000:0002:0000
   Upravený tvar: ::2: 0

Stejně jako adresy IPv4, skládají se i adresy IPv6 z hostitelské a síťové části. Tyto části jsou odděleny **délkou prefixu**, který má stejný tvar jako u IPv4. U IPv6 však není používána maska sítě ve tvaru s tečkami. Prefix má typicky hodnotu /64, ale může nabývat různých hodnot od /0 do /128. [10], [12]



## 8.3 Rozdělení IPv6 adres

Podobně jako se dělily IPv4 adresy, dělí se i IPv6 adresy podle různých kritérií.

Na základě toho, pro koho je IPv6 paket určen jsou IPv6 adresy děleny na unicastové, multicastové a anycastové. Zdrojové adresy jsou vždy unicastové. **Unicastové** adresy slouží, stejně jako u IPv4, k identifikaci komunikace mezi dvěma konkrétními hosty. Paket je zaslán odesílatelem jednomu konkrétnímu hostovi. **Multicastové** adresy se opět jako u IPv4 používají k poslání paketu z jednoho hosta na skupinu hostů. Slouží pouze jako cílové adresy. U IPv4 existovaly broadcastové adresy, ty u IPv6 nejsou, ale dají se nahradit multicastovými. Některé multicastové adresy jsou rezervovány pro speciální účely, my si uvedeme pouze některé.

Některé rezervované multicastové adresy:

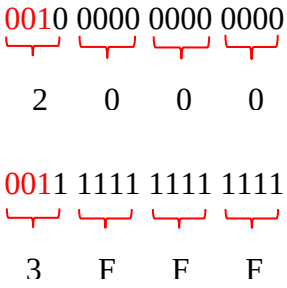
- **FF02::1** = adresa pro všechny uzly v síti (all nodes). Nahrazuje broadcast.
- **FF02:2** = adresa pro všechny IPv6 směrovače (all routers), které jsou zapnuty příkazem *ipv6 unicast-routing*. Paket dorazí na všechny IPv6 směrovače v síti.
- Prefix **FF02:0:0:0:0:1:FF00::/104** je využíván pro zjištění MAC adresy hosta v síti z jeho IPv6 adresy, dalších 24 bitů je tedy tvořeno posledními 24 bity IPv6 adresy hosta. Jedná se o obdobu protokolu ARP u IPv4, který slouží ke stejnému účelu. Zjištění MAC adresy je důležité pro další průchod paketu sítí, respektive jeho zpracování vrstvou síťového rozhraní.

Posledním typem adres jsou anycastové. **Anycastová** adresa je unicastová adresa, která je přiřazena více rozhraním najednou (převážně na různých zařízeních), paket je však doručen pouze na nejbližší z nich.

Rozdělení IPv6 unicastových adres [6], [10], [23]:

- **Lokální linkové** – slouží pouze pro adresaci v rámci lokální sítě, pakety s touto adresou směrovač nepustí ven z lokální sítě. Jedná se o adresy z rozsahu **FE80::/10**. Každé rozhraní musí mít tuto adresu přiřazenou. Pokud zařízení nemá na rozhraní přiřazenou lokální linkovou ani globální adresu, přiřadí si samo lokální linkovou adresu, aby mohlo komunikovat s ostatními zařízeními v síti. Jelikož každé rozhraní spadá do jiné sítě, je možné, aby každé rozhraní na jednou zařízení mělo přiřazenou stejnou lokální linkovou adresu, jelikož musí být unikátní pouze v rámci místní sítě.

- **Globální** – obdoba veřejných adres u IPv4. Jedná se o adresy používané pro adresaci mimo lokální síť, tedy v Internetu. Tato adresa se skládá ze tří částí. První částí je globální prefix, který odpovídá adrese sítě a je přiřazován poskytovatelem internetových služeb. Jedná se o prefix /48. Další částí je ID podsítě, které je přiřazeno danou organizací k identifikaci podsítě v rámci dané sítě. Poslední částí je ID rozhraní, které odpovídá hostitelské části adresy. Každý host může mít více rozhraní. První tři bity v prvním hextetu u globálních adres jsou 001, což odpovídá adrese 2000::/3. V šestnáctkové soustavě je tedy rozsah prvního hextetu od 2000 do 3FFF.

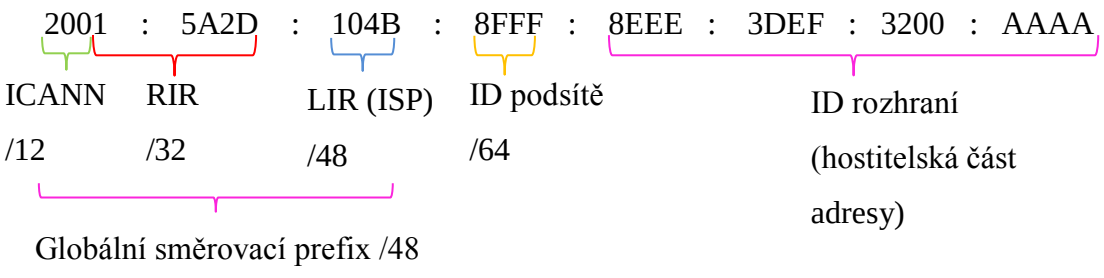



- **Loopback** – adresa, která slouží hostům k poslání paketu sobě samým. Jedná se o adresu **::1/128**.
- **Neurčená** – jedná se o adresu složenou ze samých nul, tedy **::/128**. Používá se pouze jako zdrojová adresa pro zařízení, které ještě nemá přiřazenou IPv6 adresu.
- **Unikátní lokální** – jedná se o obdobu privátních adres u IPv4. Rozsah adres je **FC00::/7 až FDFF::/7**. Slouží pro adresaci v rámci lokálních síti, ale nikoliv v Internetu.
- **IPv4 vestavěné** – jedná se o adresy, které ulehčují přechod mezi IPv4 a IPv6.

## 8.4 Přidělování IPv6 adres

Stejně jako u IPv4 má na starosti přidělování adres organizace **ICANN**. Tato organizace rozděluje adresy s prefixem **/12** mezi **regionální registrátory (RIR),** ti poté rozdělují přiřazený adresní prostor mezi poskytovatele internetových služeb (ISP) a jiné **lokální registrátory (LIR)** o velikosti prefixu **/32.** Lokální registrátoři (ISP) poté poskytují adresní prostor svým zákazníkům o velikosti prefixu **/48**, kterému se říká **globální směrovací prefix**. Mezi tyto zákazníky patří například konkrétní společnosti, které poté vytváří jednotlivé podsítě (například pro jednotlivá oddělení či pobočky) patřící pod tuto společnost.

Dalších 16 bitů slouží k identifikaci podsítě v rámci dané sítě. Jedná se o prefix **/16**, který je označen jako **ID podsítě**. Zbývajících 64 bitů slouží k identifikaci konkrétního rozhraní konkrétního zařízení. Jedná se o hostitelskou část adresy a je označována jako **ID rozhraní**.




[10]

## 8.5 Konfigurace IPv6 na rozhraní směrovače

Konfigurace je velmi podobná jako u IPv4. V globálním konfiguračním režimu je nutné nejprve zadat rozhraní, kterému bude přiřazena adresa, v tomto případě se jedná o rozhraní s označením gi0/0. Příkaz je ve tvaru *interface <typ rozhraní> <označení>*.

```
R1(config)#interface gigabitethernet 0/0
```

Po přepnutí na konfiguraci rozhraní je možno zadat požadovanou globální IPv6 adresu příkazem ve tvaru *ipv6 address <adresa sítě>/prefix.*

```
R1(config-if)#ipv6 address 2003:ee12:aabe:2::2/64
```

Pokud by se jednalo o lokální linkovou adresu, je třeba přidat do příkazu další parametr a /prefix je vynecháno. Parametr *link-local* říká, že se jedná o lokální linkovou adresu, tedy adresu platnou pouze v rámci místní sítě.

```
R1(config-if)#ipv6 address fe80::1 link-local
```

Aby bylo rozhraní v provozu, je nutné ho zapnout příkazem *no shutdown*.

```
R1(config-if)#no shutdown
```

Po dokončení konfigurace rozhraní je možné se navrátit do globálního konfiguračního režimu příkazem *exit*.

```
R1(config-if)#exit

R1(config)#
```

Pro výpis konfigurace rozhraní lze v privilegovaném uživatelském režimu zadat příkaz *show ipv6 interface brief*. Ve výpisu je název rozhraní, jeho stav a přiřazené adresy.

```
R1#show ipv6 interface brief
```

[7]

## 8.6 Konfigurace IPv6 u hosta

Konfigurace může být statická či dynamická. Statická konfigurace znamená, že je adresa, prefix i výchozí brána přiřazena ručně administrátorem. U dynamické konfigurace existují dvě cesty, jak získat tyto potřebné síťové parametry.

První je **bezstavová autokofigurace SLAAC** (Stateless Address Autoconfiguration), u kterého je adresa sítě, délka prefixu a výchozí brána přiřazována směrovačem IPv6. Směrovač IPv6 je takový směrovač, který přeposílá pakety IPv6, má staticky či dynamicky nakonfigurované IPv6 cesty sítí a posílá ICMPv6 zprávy obsahující parametry síťové vrstvy pro zařízení v síti využívající IPv6. Pro přepnutí směrovače z IPv4 na IPv6 směrovač je třeba zadat příkaz *ipv6 unicast-routing* v globálním konfiguračním režimu směrovače.

```
R1(config)#ipv6 unicast-routing
```

Host po připojení do sítě zasílá žádost o přiřazení síťových parametrů formou multicastové žádosti na všechny IPv6 směrovače v síti (adresa FF02::2), tato žádost je jinak nazývána „router solicitation“. Směrovač poté v odpovědi zasílá požadované parametry, tato odpověď je nazývána „ICMPv6 router advertisement“. Hostitelská část adresy, tedy identifikátor rozhraní, je určen metodou EUI-64.

Pokud by dané informace nebyly dostatečné, je možnost zaslat žádost na DHCPv6 server o přiřazení například adresy DNS serveru. V případě nutnosti i zažádat o adresu sítě, prefix a výchozí bránu. Protokol **DHCPv6** je stavový a pracuje jako DHCP u IPv4. [7], [12], [23]

## 8.7 EUI-64 (Extended Unique Identifier)

Identifikátor rozhraní tvoří posledních 64 bitů adresy IPv6. Jeden ze způsobů, jak tento identifikátor určit, je právě metoda EUI-64 definovaná IEEE. Pro určení identifikátoru rozhraní využívá 48bitovou MAC adresu síťové karty. Prvních 24 bitů MAC adresy tvoří jednoznačný **identifikátor výrobce** (**OUI** – Organizationally Unique Identifier) přiřazený IEEE. Dalších 24 bitů jednoznačně identifikuje dané zařízení.

***Příklad:***

MAC adresa hexadecimálně:

FC:99:47:55:12:BE

MAC adresa binárně:

1111 1100 : 1001 1001 : 0100 0111 : 0101 0101 : 00001 0010 : 1011 1110

Prvním krokem je převrátit hodnotu 7. bitu v adrese zleva (0 = globálně jednoznačný identifikátor, 1 = lokálně jednoznačný identifikátor).

1111 1110 : 1001 1001 : 0100 0111 : 0101 0101 : 00001 0010 : 1011 1110

Nová MAC adresa hexadecimálně:

FE:99:47:55:12:BE

V dalším kroku je mezi OUI a identifikátor zařízení vložena 16bitová hodnota FFFE.

FE:99:47:FF:FE:55:12:BE

Dále je pouze změněn zápis po 2 bytech, ne po jednom. Výsledkem je konečný identifikátor rozhraní.

FE99:47FF:FE55:12BE

Tato metoda má výhodu v tom, že z IPv6 adresy lze lehce určit MAC adresu zařízení, což je zároveň i značnou nevýhodou, protože by mohly být pakety sledovány až k fyzickému

počítači, což znamená, že by mohlo dojít k porušení ochrany osobních údajů. Z tohoto důvodu jsou spíše využívány náhodně generované identifikátory rozhraní. U MS Windows jsou tyto identifikátory generovány od verze Windows Vista.

Pro určení identifikátoru rozhraní metodou EUI-64 lze na směrovači zadat příkaz *ipv6 address <adresa sítě>/prefix eui-64*:

```
Router(config)#interface fastEthernet 0/0
Router(config-if)#ipv6 address abcd::/64 eui-64
Router(config-if)#exit
```

Jakmile je utvořen identifikátor rozhraní, je možné určit globální i lokální linkovou adresu. Globální adresa jsou ta adresa, která je unikátní i mimo lokální síť. Je tvořena zadaným prefixem a dopočítaným identifikátorem. Lokální linkové adresy jsou unikátní pouze v rámci místní sítě. Jejich síťová část začíná na FE80::/10, typicky je doplněna nulami na FE80::/64. Hostitelské části obou adres jsou poté tvořeny určeným identifikátorem rozhraní. [7], [12]

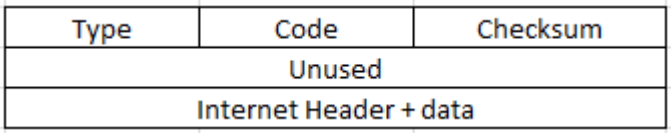
Pro zjištění vytvořených adres existuje příkaz:

```
Router(config)#do show ipv6 interface fastEthernet 0/0

FastEthernet0/0 is administratively down, line protocol is down
  IPv6 is enabled, link-local address is FE80::260:2FFF:FE5D:8C01 [TEN]
  No Virtual link-local address(es):
  Global unicast address(es):
    ABCD::260:2FFF:FE5D:8C01, subnet is ABCD::/64 [EUI/TEN]
  Joined group address(es):
    FF02::1
    FF02::1:FF5D:8C01
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ICMP unreachables are sent
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  ND advertised default router preference is Medium
  Hosts use stateless autoconfig for addresses.
```

# 9 ICMP (Internet Message Control Protocol)

Tento protokol je zapouzdřen v hlavičce IP paketu a existuje ve dvou verzích ICMPv4 (RFC 792) a ICMPv6 (RFC 4443), obě verze poskytují stejné funkce, ICMPv6 jich má však pár navíc. Úkolem protokolu ICMP je zasílat zprávy v případě výskytu jisté události.

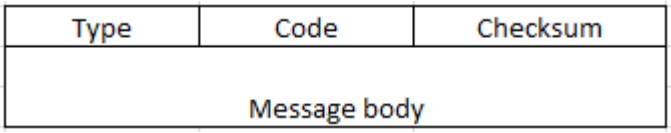
Obecný formát zprávy ICMPv4 [22]:




**Obrázek 31 – ICMPv4 zpráva**

Obecný formát zprávy ICMPv6 [5]:

Tento formát zprávy je pouze obecný, u jednotlivých typů zpráv se pole s tělem zprávy liší.




**Obrázek 32 – ICMPv6 zpráva**

Každá zpráva má určitý **typ**, který určuje typ dané zprávy a **kód**, který případně upřesňuje důvod jejího výskytu. Obojí je zapouzdřeno v hlavičce ICMP protokolu. Stejně tak je v hlavičce ICMP zpráv obraženo pole **kontrolního součtu**, které slouží ke kontrole úplnosti paketu, a nakonec následuje tělo samotné zprávy, které se u jednotlivých typů zprávy může odlišovat. [23]

Některé typy ICMPv4 zpráv [22]:

- 0 – Echo Reply (odpověď na žádost)
- 3 – Destination Unreachable (nedostupnost cíle)
- 8 – Echo Request (požadavek na odpověď)
- 11 – Time Exceeded (čas překročen)
    - kód 0 - překročena životnost paketu (TTL exceeded)
    - kód 1 – překročen čas pro znovusložení fragmentů

Některé typy ICMPv6 zpráv [5]:

Zprávy se dělí do dvou hlavních kategorií. První jsou chybová hlášení, která mají kódy v rozsahu 0 až 127 a druhou kategorií jsou informační zprávy, které mají kódy od 128 do 255.

- 1 – Destination Unreachable (nedostupnost cíle)
- 2 – Packet Too Big (příliš velký paket)
- 3 – Time Exceeded (čas překročen)
- 128 – Echo Request (požadavek na odpověď)
- 129 – Echo Reply (odpověď na žádost)

Základní funkcí protokolu ICMP je zjištění, **zda je daný host k dispozici.** K tomuto účelu je zaslán z jednoho hostitele na druhého požadavek (**Echo Request**), pokud je cílový hostitel dostupný, zašle odpověď (**Echo Reply**). Dalším úkolem tohoto protokolu je informovat odesílatele paketu v případě, že **je cíl nedostupný**. Pokud tato situace nastane, je zaslána u ICMPv4 zpráva typu 3, u ICMPv6 zpráva typu 1 společně s kódem, který určuje, proč je cíl nedostupný. Jako cílová adresa tohoto typu zprávy slouží zdrojová adresa paketu. Jednotlivé kódy se u ICMPv4 a ICMPv6 lehce liší. [10], [23]

Zpráva ICMPv4 typu 3 [22]:

| Type | Code | Checksum |
|---|---|---|
| Unused | | |
| Internet Header + data | | |

**Obrázek 33 – ICMPv4 zpráva typu 3**

Kódy ICMPv4 zpráv typu 3 (Destination Unreachable):

- 0 – síť je nedostupná
- 1 – host je nedostupný
- 2 – protokol je nedostupný
- 3 – port je nedostupný
- 4 – je třeba fragmentace
- 5 – zdrojová cesta selhala

Zprávy s kódy 0, 1, 4 a 5 jsou odesílány z výchozí brány. Jedná se například o situaci, kdy výchozí brána, tedy směrovač, na základě obsahu své směrovací tabulky zjistí, že daná síť je

nedostupná, v tomto případě je zaslána zpráva o nedostupnosti cíle s kódem 0, v některých sítích je směrovač schopen zjistit i nedostupnost hosta, v tomto případě zaslána zpráva s kódem 1. Pokud paket přesahuje velikost MTU (maximální velikost přenášené jednotky), je nutná fragmentace a je bránou zaslána ICMP zpráva typu 3 s kódem 4. Pokud je paket doručen k cílovému hostu, ale vyskytne se problém s předáním rozbaleného paketu na segment protokolu, uvedeném v hlavičce segmentu, je zaslána zpráva o nedostupnosti protokolu s kódem 2, v případě, že není dostupný port, je zaslána zpráva s kódem 3. [10], [22]

Některé kódy ICMPv6 zprávy typu 1 (Destination Unreachable) [5]:

- 0 – nenalezena cesta k cíli
- 1 – komunikace s cílem, který byl administrativně zakázán
- 2 – mimo rozsah zdrojové adresy
- 3 – cílová adresa nedostupná
- 4 – port nedostupný

Další případem, kdy je zaslána ICMP zpráva je **překročení životnosti paketu.** V případě, že dojde ke snížení životnosti paketu na nulu, je tento paket zahozen a odesílatel je o této skutečnosti informován právě prostřednictvím ICMP zprávy. Další ICMP zprávou je zpráva o **přesměrování trasy**. Slouží k informování hostů o lepší cestě do cíle. [10], [23]

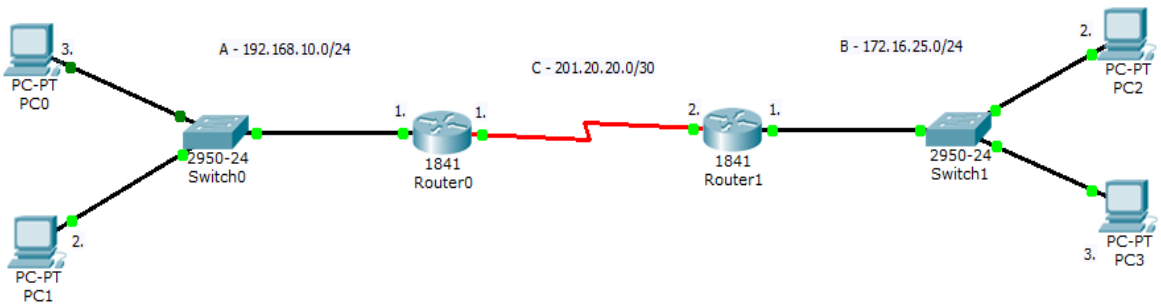
Protokol ICMPv6 poskytuje oproti ICMPv4 pár funkčností navíc. První jsou již zmíněné „router solicitation“ a „router advertisement“ zprávy sloužící pro bezstavovou autokonfiguraci pomocí SLAAC v IPv6. Další funkcí navíc je již zmíněný překlad IPv6 adresy na MAC adresu zasláním žádosti na speciální multicastovou adresu s prefixem FF02:0:0:0:0:1:FF00::/104. Tato žádost je nazývána „neighbor solicitation“ a obsahuje IPv6 adresu hosta, jehož MAC adresu chceme zjistit. Daný host poté odpoví zprávou, která obsahuje požadovanou MAC adresu, tato zpráva je nazývána „neighbor advertisement“. Třetí funkcí navíc je zjištění duplicity adres. Toto zjišťování probíhá tak, že daný host zašle „neighbor solicitation“ zprávu na svou vlastní adresu, pokud mu přijde odpověď „neighbor advertisement“, znamená to, že danou adresu má přiřazenou i jiné zařízení, pokud žádná odpověď nedorazí, je jeho adresa unikátní. Tento mechanismus je použitelný jak pro lokální linkové adresy (unikátní v rámci místní sítě), tak i pro globální (unikátní i mimo místní síť). [5], [9], [10], [23]

# 10 Testování dostupnosti hostů

Pro ověření, zda je daný host pro jiného dostupný, existuje příkaz ***ping***. Funguje na již zmíněném principu žádost/odpověď u ICMP zpráv. Host pošle požadavek na druhého a čeká na odpověď, toto čekání má však omezenou dobu. Pokud nepřijde odpověď včas, znamená to, že z nějakého důvodu jsou zprávy blokovány. Odesílající host přijímá statistiku úspěšnosti těchto žádostí.

Příkaz ping sice informuje o dostupnosti cíle, ale někdy je třeba vědět, ze kterého rozhraní není daný paket dále přeposílán. K tomuto účelu slouží příkaz ***traceroute*** (Cisco)**,** zkráceně ***tracert*** (MS Windows). *Traceroute* vypisuje všechny navštívených uzly na cestě do cíle. V případě výskytu chyby je ve výpisu *. *Traceroute* pracuje tak, že nejprve zašle zprávu s životností 1. Po přeskoku na první směrovač je tato hodnota snížena na nulu a zaslána ICMP zpráva o překročení životnosti, tím získá host adresu prvního přeskoku. V dalším kroku je životnost o jedničku vyšší, tedy má hodnotu 2, čímž se dostane na druhý směrovač a z ICMP zprávy opět zjistí jeho adresu. Takhle se pokračuje do té doby, dokud není dosaženo cíle. Po dosažení cíle je totiž místo ICMP zprávy o překročení životnosti zaslána zpráva s odpovědí nebo informace o nedostupnosti hosta. [7], [10], [23]

Na níže uvedené topologii (Obrázek 34) jsou uvedeny příklady výpisu po zadání příkazu *ping* a *tracert*.




**Obrázek 34 – Topologie tracert a ping**

***Příklad ping 1:***

Z počítače s označením PC2 je testována dostupnost hosta s adresou 192.168.10.2, tedy PC1. Host je dostupný, jelikož nebyl ztracen žádný paket.

```
PC>ping 192.168.10.2
Pinging 192.168.10.2 with 32 bytes of data:

Reply from 192.168.10.2: bytes=32 time=1ms TTL=126
Reply from 192.168.10.2: bytes=32 time=6ms TTL=126
Reply from 192.168.10.2: bytes=32 time=1ms TTL=126
Reply from 192.168.10.2: bytes=32 time=16ms TTL=126

Ping statistics for 192.168.10.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 16ms, Average = 6ms
```

***Příklad ping 2:***

Z počítače s označením PC2 je testována dostupnost hosta s adresou 192.168.10.5, žádný takový host v dané topologii však není. Host tedy není dostupný, proto jsou všechny pakety zahozeny.

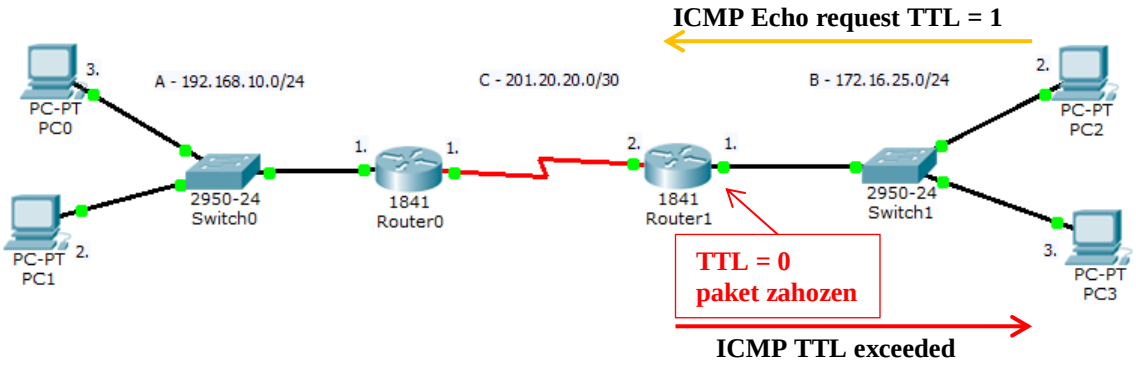
```
PC>ping 192.168.10.5
Pinging 192.168.10.5 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.10.5:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),…
```

***Příklad tracert 1:***

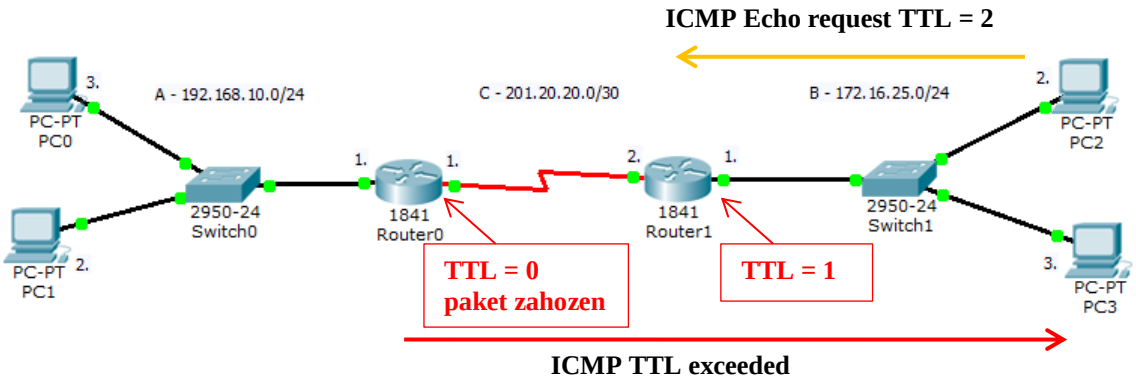
V tomto případě je uveden výpis příkazu *tracert*. Je zadán příkaz *tracert* na sledování cesty paketu z PC2 do PC0 s adresou 192.168.10.3. Ve výpisu jsou vypsány všechny přeskoky na cestě do cíle včetně adresy cílového hostitele.

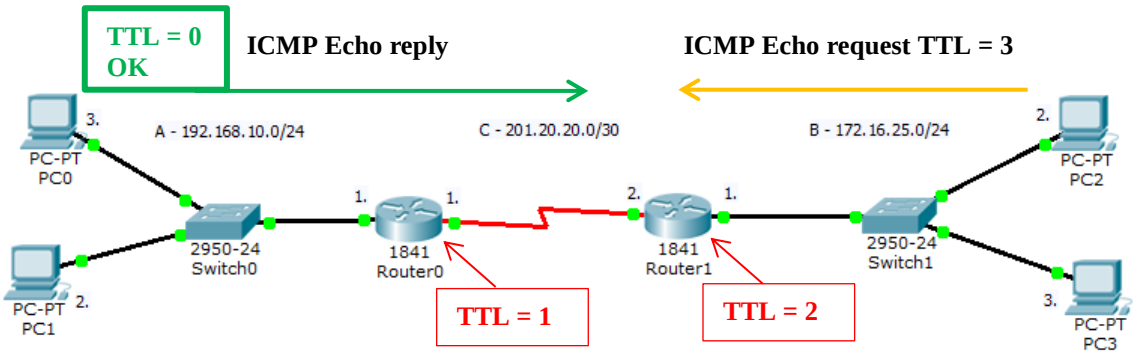
Nejprve je zaslán ICMP paket s žádostí o odpověď s životností TTL = 1. Ten je na prvním přeskoku Router1 s adresou 172.16.25.1 zahozen a je zaslána ICMP zpráva o překročení životnosti. Poté je životnost nastavena na TTL = 2. Paket je tentokrát zahozen až na druhém přeskoku Router0 s adresou 201.20.20.1. Po zvýšení životnosti na TTL = 3 již paket dojde do cíle s adresou 192.168.10.3, odkud dojde ICMP zpráva s odpovědí na žádost.

**C:>tracert 192.168.10.3**




**Obrázek 35 – Tracert**




**Obrázek 36 – Tracert**

```
PC>tracert 192.168.10.3
Tracing route to 192.168.10.3 over a maximum of 30 hops:

  1    0 ms      1 ms      0 ms      172.16.25.1
  2    1 ms      1 ms      0 ms      201.20.20.1
  3    0 ms      1 ms      1 ms      192.168.10.3

Trace complete.
```

***Příklad tracert 2:***

V tomto případě je zadán příkaz *tracert* na sledování cesty paketu z PC2 do zařízení s adresou 192.168.10.5. Toto zařízení však neexistuje, ale patří do sítě 192.168.10.0/24, proto jsou ve výpisu viditelné přeskoky na cestě do této sítě. Hvězdičky ve výpisu ukazují pokusy o dosažení hosta s adresou 192.168.10.5, které jsou však neúspěšné. Celé sledování trasy by běželo do doby, než by dosáhlo hodnoty maxima 30 přeskoků.

```
PC>tracert 192.168.10.5

Tracing route to 192.168.10.5 over a maximum of 30 hops:

  1    1 ms      0 ms      1 ms      172.16.25.1
  2    0 ms      1 ms      0 ms      201.20.20.1
  3    *         *         *         Request timed out.
  4    *         *         *         Request timed out.
       …
 30    *         *         *         Request timed out.
```

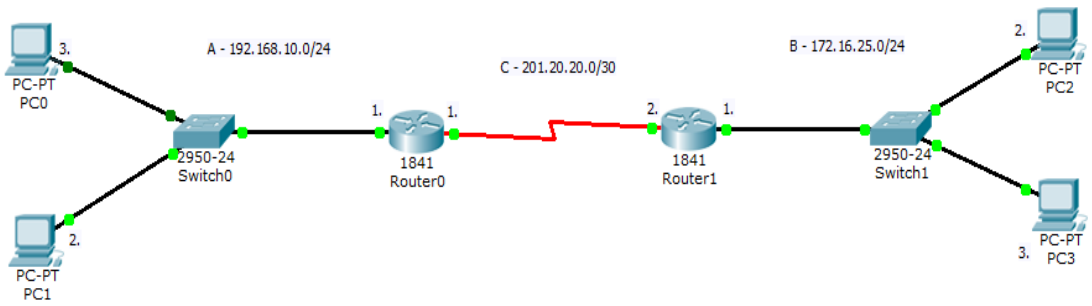
# 11 Směrování

Jak již bylo zmíněno, úkolem síťové vrstvy je nalezení vhodné cesty tak, aby paket byl správně doručen do cíle. Většinou se nejedná pouze o přeposílání paketů v rámci jedné malé lokální sítě, ale spíše o přeposílání do jiných lokálních sítí či Internetu.

Jednotlivé sítě jsou propojeny mezilehlými zařízeními, konkrétně **směrovači (routery),** které mají za úkol ono hledání vhodné cesty mezi různými sítěmi. Tomuto procesu hledání nejlepší cesty do cíle se jinak říká **směrování**. Router, respektive jeho vstupní rozhraní, připojený k lokální síti slouží jako **výchozí brána** (default gateway), přes kterou jsou pakety přeposílány do vzdálených sítí. Při posílání paketů v rámci jedné sítě směrovač potřeba není.

Směrovač totiž obsahuje informace o možných cestách mimo lokální síť, což jiná zařízení v této lokální síti nemají. Tyto informace má směrovač uloženy ve své **směrovací tabulce.** V tabulce jsou nejen informace o jiných přímo připojených sítích, ale také o směrech do dalších vzdálených sítí. Nemá však informace o přesné cestě do vzdálené sítě. Přímo připojené sítě jsou do směrovací tabulky přidány automaticky, jakmile je nakonfigurováno rozhraní směrovače. Ostatní jsou přidány administrátorem ručně (statické směrování) nebo prostřednictvím směrovacích protokolů (dynamické směrování). [14], [15], [23]

Pro názornou představu o směrovací tabulce je uvedena následující topologie. Zařízení jsou nakonfigurována dle obrázku (Obrázek 37).




**Obrázek 37 – Ukázková topologie**

V případě, že PC1 bude chtít zaslat paket PC0, není třeba využití směrovače s označením Router0, jelikož jsou oba počítače ve stejné síti, o přeposlání v rámci sítě se postará přepínač. Jakmile by však chtěl PC1 zaslat paket na PC2, bylo by už využití směrovače nezbytné. Pro výpis směrovací tabulky počítače PC1 lze použít příkaz ***netstat –r*** nebo ***route print*** v příkazovém řádku [7]:

```
PC>netstat -r

Route Table
===========================================================================
Interface List
0x1 ........................... PT TCP Loopback interface
0x2 ...00 16 6f 0d 88 ec ...... PT Ethernet interface
===========================================================================
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      192.168.10.1    192.168.10.2       1
Default Gateway:      192.168.10.1
===========================================================================
…
```

Směrovací tabulka počítače PC1 obsahuje pět sloupců. V prvním sloupci je uvedena adresa cílové sítě, ve druhém sloupci je uvedena maska sítě, dále je uvedena výchozí brána, dále odchozí rozhraní, přes které je zaslán paket na výchozí bránu, a metrika.

Uvedená tabulka obsahuje jediný řádek, který obsahuje **implicitní cestu**. Tato cesta je využita vždy v případě, že v tabulce není záznam o cestě do požadované sítě. Jelikož se jedná o jediný záznam v této tabulce, veškeré pakety, které budou odeslány z PC1, budou poslány z odchozího rozhraní s adresou 192.168.10.2. Paket z PC1 tedy bude poslán přes přepínač na výchozí bránu s adresou 192.168.10.1 směrovače s označením Router0. Směrovač odebere paketu hlavičku, sníží TTL o 1 a na základě uvedené cílové adresy prohledává směrovací tabulku, toto je provedeno na každém směrovači. V případě, že by došlo ke snížení životnosti TTL na nulu, paket by byl zahozen a zaslána ICMP zpráva o vypršení životnosti paketu. [15]

Obsah směrovací tabulky směrovače lze vypsat příkazem ***show ip route*** v privilegovaném[18] režimu směrovače:

```
Router#show ip route

Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter
area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is 0.0.0.0 to network 0.0.0.0

     172.16.0.0/24 is subnetted, 1 subnets
S       172.16.25.0 [1/0] via 201.20.20.2
C    192.168.10.0/24 is directly connected, FastEthernet0/0
     201.20.20.0/30 is subnetted, 1 subnets
C       201.20.20.0 is directly connected, Serial0/0/0
S*   0.0.0.0/0 is directly connected, Serial0/0/0
```

V prvním sloupci tabulky je uvedeno, jakým způsobem byl konkrétní řádek tabulky přidán. Zda staticky (S), dynamicky (zkratka záleží na konkrétním protokolu) nebo automaticky přímo připojené sítě (C). V druhém sloupci je uvedena cílová adresa sítě a za lomítkem je uvedena maska sítě. Dále je uvedena administrativní vzdálenost[19]/metrika[20]. Pokud existují dvě cesty do cíle, je využita ta s menší metrikou. Dalším záznamem v tabulce může být další přeskok (next hop), což je adresa vstupního rozhraní na následujícím směrovači, slouží tedy jako brána do vzdálené sítě. Dále je uvedeno odchozí rozhraní, přes které paket dojde na tuto bránu. Pokud by na daném řádku nebylo odchozí rozhraní uvedeno, ale pouze adresa dalšího přeskoku, je třeba toto odchozí rozhraní rekurzivně dohledat ve směrovací tabulce. Odchozí rozhraní se vyhledá na základě adresy sítě, do které patří adresa dalšího přeskoku. V posledním řádku tabulky je uvedena implicitní cesta (default route), která slouží jako brána poslední záchrany (výchozí brána) v případě, že na předchozích řádcích nebyla nalezena cesta do požadované sítě. Tato implicitní cesta je nastavena ručně, má vždy adresu sítě 0.0.0.0, stejně tak i masku 0.0.0.0 a odchozí rozhraní záleží na konkrétní topologii. [7], [15]

Paket je tedy posílán z PC1 s adresou 192.168.10.2 na PC2 s adresou 172.16.25.2. Směrovač najde záznam o dalším přeskoku 201.20.20.2. Jelikož směrovač zná pouze adresu dalšího

[18] Privilegovaný uživatelský režim umožňuje vypsat informace o směrovači, ale není umožněna jeho konfigurace.
[19] Administrativní vzdálenost (AD) určuje, jak je použitý směrovací protokol důvěryhodný. Čím nižší číslo, tím lepší. Přímo připojené sítě mají AD rovnu nule a staticky přidané cesty AD rovnu jedné.
[20] Metrika vyjadřuje cenu cesty, opět čím nižší číslo, tím lepší.

přeskoku, je třeba, aby rekurzivně dohledal odchozí rozhraní. Toto odchozí rozhraní leží ve stejné síti jako další přeskok s adresou 201.20.20.2, tedy v síti 201.20.20.0/30. Pro tuto síť je uvedeno jako odchozí rozhraní Serial0/0/0. Paket, který byl rozbalen je opět zapouzdřen a zaslán z tohoto rozhraní směrem ke vstupnímu rozhraní dalšího směrovače Router1 s adresou 201.20.20.2. Tento směrovač má již záznam o síti, v níž je PC2, jako o přímo připojené síti. Proto může paket zaslat přímo na PC2 s adresou 172.16.25.2 uvedeným odchozím rozhraním FastEthernet0/0 s adresou 172.16.25.1. Po přijetí paketu počítačem PC2 je paketu odstraněna hlavička a daný segment je předán vrstvám vyšším ke zpracování.

```
Router#show ip route
…

Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 1 subnets
C       172.16.25.0 is directly connected, FastEthernet0/0
S    192.168.10.0/24 is directly connected, Serial0/0/0
     201.20.20.0/30 is subnetted, 1 subnets
C       201.20.20.0 is directly connected, Serial0/0/0
```

Tabulka směrovače obsahuje přímo připojené sítě, které jsou směrovačem přidány automaticky při nakonfigurování daného rozhraní, ostatní sítě mohou být přidány dvěma způsoby. První způsob je **statické směrování**, kde cesty nastaví ručně administrátor, druhý způsob je za pomoci směrovacích protokolů, které samy reagují na změny v síti a upravují obsah směrovací tabulky. Statické směrování se doporučuje používat u méně rozlehlých sítí, u větších je možné, že administrátor snadno udělá chybu, protože topologie může být velice složitá. Tento způsob směrování nemá na rozdíl od dynamického žádnou režii, nevýhodou však jsou nutné zásahy administrátora v případě výskytu jakékoliv změny v síti. **Dynamické směrování** prostřednictvím směrovacích protokolů vyžaduje neustálou výměnu informací mezi směrovači, což znamená režii navíc, jelikož je využíván procesorový čas. Hlavní výhodou dynamického směrování je však to, že nejsou příliš nutné zásahy administrátora. Jeho úkolem je nakonfigurovat vhodný směrovací protokol a ten se poté sám stará o vytváření řádek směrovací tabulky. [15], [23]

Existuje několik směrovacích protokolů, mezi které patří například třídní RIP nebo beztřídní RIPv2, EIGRP, OSPF nebo BGP a IS-IS. Tyto protokoly jsou však mimo náplň těchto studijních materiálů. [15]

# 12 Vytváření podsítí

Tato kapitola je věnována vytváření podsítí neboli podsíťování (subnetting). Pro začátek je uvedeno, co podsíťování znamená a jaké jsou důvody k jeho využití. Tento pojem znamená rozdělení jedné větší sítě do menších sítí (podsítí) využitím delšího prefixu neboli masky podsítě než je třídní prefix.

U velmi malých sítí o pár zařízeních není vytváření podsítí třeba, ale v případě větších sítí je to v podstatě nutnost. Jakmile je síť větší a několik hostů pošle zprávu na broadcastovou adresu, dochází ke značnému zahlcení a zpomalení sítě, protože jsou tyto zprávy zpracovávány všemi zařízeními v síti. Z tohoto důvodu je výhodnější rozdělit tuto síť na menší podsítě na základě například geografického umístění či jiného kritéria pro zvýšení výkonu v síti. Tyto zprávy jsou poté zaslány pouze na omezený počet zařízení, která danou zprávu opravdu potřebují. Se vzniklými podsítěmi se pracuje následně jako s oddělenými sítěmi, navenek však působí stále jako jedna síť. Pro přeposílání zpráv mezi jednotlivými podsítěmi je již třeba využití směrovače, jehož vstupní rozhraní tvoří výchozí bránu do další podsítě.

Druhým a zároveň hlavním důvodem vzniku podsíťování bylo značné plýtvání adresním prostorem. Dříve totiž byly přiřazovány jednotlivým sítím adresy s prefixem /8, /16 nebo /24, tedy třídy A, B nebo C. Což znamenalo, že síť, která se skládala pouze z deseti hostů, měla k dispozici adresní prostor o velikosti 256 adres.

Později však bylo zavedeno beztřídní adresování neboli **CIDR (Classless Inter-Domain Routing)**, které umožňuje použití i jiných prefixů než těch třídních. Je tedy možné použít například prefix /12, /18 nebo /30. S CIDR velmi úzce souvisí již zmíněné podsíťování. Kromě vytváření podsítí umožňuje také existenci tzv. **supersítí** (supernet).

K vytvoření **podsítí se využívá prodloužení prefixu** adresy sítě, tedy využívá bity hostitelské části. Naopak **supersítě** slouží k agregování několika sítí do jedné **s kratším prefixem**, než je třídní prefix, jsou tedy využívány bity ze síťové části.

Pokud by se síť skládala například z 2000 hostů, může jí být přiřazena adresa sítě třídy B, která však poskytuje příliš velký adresní prostor (65536 adres). Z tohoto důvodu by bylo spíše přiřazeno několik síťových adres třídy C (po 256 adresách), tedy minimálně osm (8 * 256 = 2048 adres), což by však znamenalo, že by pro každou tuto síť vznikl záznam

ve směrovací tabulce[21] směrovače, což není příliš vhodné. V případě, že je směrovací tabulka příliš rozsáhlá, dochází totiž ke zpomalení procesu hledání nejvhodnější cesty do cíle. Existuje ale možnost všech osm sítí agregovat do jedné supersítě, což vede k tomu, že by místo několika záznamů ve směrovací tabulce, které vedou do stejného místa, byla v tabulce pouze adresa této supersítě. [4], [10], [23]

## 12.1 Metody vytváření podsítí v IPv4

Pro vytváření podsítí jsou používány dvě metody, obě využívají CIDR. CIDR je náhrada třídní adresace, která umožňuje využívat různé délky prefixů. První variantou je rozdělení sítě na **stejně velké podsítě**, tedy prefixy jsou u všech podsítí totožné, a druhou variantou je rozdělení sítě na **různě velké podsítě** s různou délkou prefixů.

Jak již bylo zmíněno, k podsíťování se využívá maska podsítě neboli prefix. Maska podsítě vždy určuje, která část adresy je hostitelská a která síťová. Pro připomenutí 192.168.10.0/24 znamená, že prvních 24 bitů tvoří síťovou část adresy a zbylých osm z 32 bitů tvoří síťovou část adresy. Pro vytvoření podsítě je třeba si vypůjčit bity z hostitelské části, které se stanou síťovými.

### 12.1.1 První metoda – stejně velké podsítě

Jelikož jsou u této metody všechny podsítě stejně velké, je možné z počtu vypůjčených bitů spočítat počet podsítí.

Pro představu poslouží adresa 192.168.10.0/24. Z této adresy bude vypůjčen jeden bit na vytvoření podsítí, prefix bude tedy prodloužen z /24 na /25. Počet podsítí se spočítá jako $2^{\text{počet vypůjčených bitů}} = 2^1 = 2$. Zbývající bity v hostitelské části určují počet dostupných adres v každé podsíti, tedy $2^7 = 128$. Čím více bitů z hostitelské části je využito na podsíťování, tím méně je dostupných hostitelských adres v rámci dané podsítě. Podsítě jsou číslovány od nuly. [4], [10]

[21] Směrovací tabulka je tabulka, kterou si uchovává každý směrovač a obsahuje možné cesty do různých uzlů v síti.

*Příklad 1:*

Adresa sítě s prefixem: 192.168.10.0/24

Maska sítě binárně: 11111111.11111111.11111111.00000000

Adresa sítě binárně: 11000000.10101000.00001010.00000000

8 bitů hostitelská část

Na podsíťování je využit jeden bit hostitelské části.

Maska podsítě binárně: 11111111.11111111.11111111.10000000

7 bitů hostitelská část

Počet podsítí: $2^{\text{počet vypůjčených bitů}} = 2^1 = 2$

Počet adres v podsíti: $2^7 = 128$

V každé síti je první adresa určena pro adresu sítě a poslední jako broadcastová adresa. Pro hosty tedy zbývá o dvě adresy méně, tedy $2^7 - 2 = 126$.

Pro zajímavost je vždy **adresa sítě sudá a broadcastu lichá**.

Adresa 0. podsítě s prefixem: 192.168.10.0/25

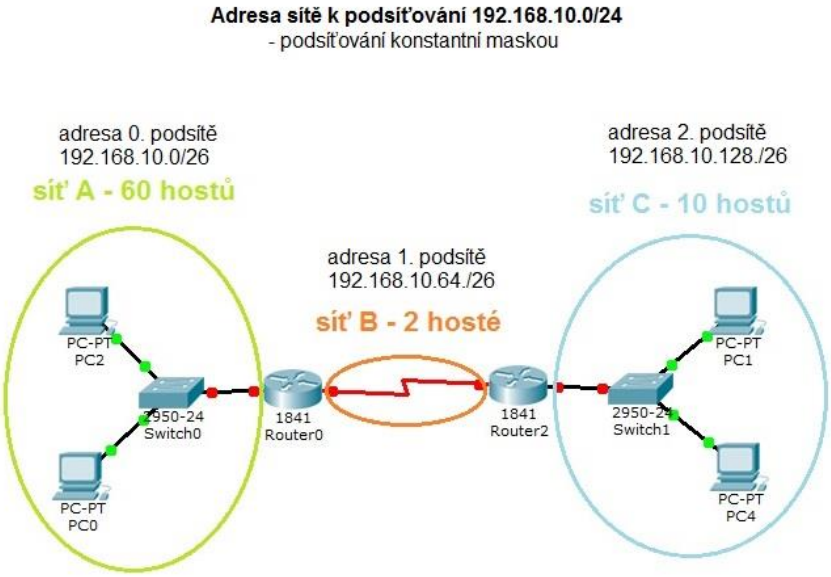
- Adresa prvního hosta 0. podsítě: 192.168.10.1
- Adresa posledního hosta 0. podsítě: 192.168.10.126
- Adresa broadcastu 0. podsítě: 192.168.10.127

Adresa 1. podsítě s prefixem: 192.168.10.128/25

- Adresa prvního hosta 1. podsítě: 192.168.10.129
- Adresa posledního hosta 1. podsítě: 192.168.10.254
- Adresa broadcastu 1. podsítě: 192.168.10.255

Ve výše uvedeném příkladu byly vytvořeny dvě podsítě, ale je možné jich vytvořit i více. Pro vytvoření tří podsítí je však třeba využít větší počet bitů než jeden.

*Příklad 2:*




**Obrázek 38 – Topologie vytváření podsítí s konstantní maskou**

Adresa pro vytváření podsítí: **192.168.10.0/24**

Maska sítě binárně: 11111111.11111111.11111111.00000000

Adresa sítě binárně: 11000000.10101000.00001010.00000000

8 bitů hostitelská část

K dispozici je adresa sítě 192.168.10.0/24. Pro vytvoření tří podsítí je třeba využít dva bity hostitelské části ($2^{\text{počet využitých bitů}} = 2^2 = 4$ podsítě). Jelikož chceme pouze tři podsítě, jednu podsíť nevyužijeme. V každé podsíti bude k dispozici $2^6 = 64$ adres, z toho 62 pro hosty.

dva vypůjčené bity

Maska podsítě binárně: 11111111.11111111.11111111.11000000

6 bitů hostitelská část

Počet podsítí: $2^{\text{počet vypůjčených bitů}} = 2^2 = 4$

Počet adres v podsíti: $2^6 = 64$

V každé síti je první adresa určena pro adresu sítě a poslední jako broadcastová adresa. Pro hosty tedy zbývá o dvě adresy méně, tedy $2^6 - 2 = 62$.

**Adresa 0. podsítě** - síť A (60 hostů): 192.168.10.0/26

- Vyplýtváno: 2 adresy
- Adresa prvního hosta 0. podsítě: 192.168.10.1
- Adresa posledního hosta 0. Podsítě: 192.168.10.62
- Adresa broadcastu 0. podsítě: 192.168.10.63

**Adresa 1. podsítě** – síť B (2 hosté): 192.168.10.64/26

- Vyplýtváno: 60 adres
- Adresa prvního hosta 1. podsítě: 192.168.10.65
- Adresa posledního hosta 1. Podsítě: 192.168.10.126
- Adresa broadcastu 1. podsítě: 192.168.10.127

**Adresa 2. podsítě** – síť C (10 hostů): 192.168.10.128/26

- Vyplýtváno: 52 adres
- Adresa prvního hosta 2. podsítě: 192.168.10.129
- Adresa posledního hosta 2. podsítě: 192.168.10.190
- Adresa broadcastu 2. podsítě: 192.168.10.191

**Adresa 3. podsítě** - nevyužita: 192.168.10.192/26

- Vyplýtváno: 62 adres
- Adresa prvního hosta 3. podsítě: 192.168.10.193
- Adresa posledního hosta 3. Podsítě: 192.168.10.254
- Adresa broadcastu 3. podsítě: 192.168.10.255

Je zřejmé, že pro uvedené sítě B a C je počet adres v každé podsíti zbytečně velký. Proto je zavedený další způsob, který tuto problematiku do jisté míry řeší.

### 12.1.2 **Druhá metoda – různě velké sítě**

Předchozí metoda se stejně velkými podsítěmi sice šetří adresní prostor oproti třídnímu adresování, ale ne natolik, jako tato metoda. Tato metoda, jinak nazývaná jako **VLSM (Variable-Length Subnet Masking)**, totiž umožňuje vytvořit různě velké podsítě, které se liší délkou prefixu, tedy maskou podsítě. Díky různým délkám prefixům je umožněno vytvořit sítě s kapacitou hostů, která buď přesně odpovídá požadovanému počtu, nebo ho jen lehce převyšuje. Tímto dochází k výraznému vylepšení předchozí metody.

U vytváření různě velkých podsítí se začíná s adresováním **podsítě s největším počtem hostů** a postupuje se směrem k nejmenší.

Pro názornost použijeme stejnou topologii jako v předchozím případě, kde byly vytvořeny čtyři podsítě o 62 dostupných adresách pro hosty. Nejvíce hostů je vyžadováno v síti A. Jedná se o 60 hostů, k tomu je třeba ještě dvou adres na adresu sítě a broadcast, tedy celkem 62 adres. Při hledání vhodného prefixu, se vždy hledá **mocnina dvou**, která je **rovna** požadovanému počtu adres (v tomto případě 62) nebo **nejmenší mocnina**, která je **větší** než tento počet. V tomto případě bude použit prefix /26 ($2^6 = 64$). [4], [10], [23]

Jako pomůcka při hledání vhodného prefixů může posloužit následující tabulka.

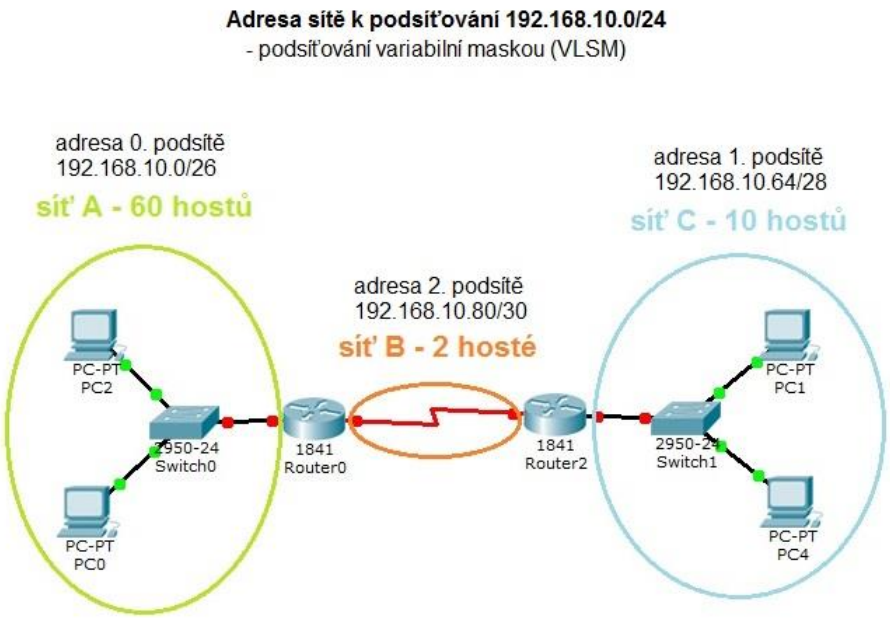
**Tabulka 9 – Prefixy**

| Počet adres | Počet bitů hostitelské části | Prefix | Maska |
|---|---|---|---|
| 1 | 0 | /32 | 255.255.255.255 |
| 2 | 1 | /31 | 255.255.255.254 |
| 4 | 2 | /30 | 255.255.255.252 |
| 8 | 3 | /29 | 255.255.255.248 |
| 16 | 4 | /28 | 255.255.255.240 |
| 32 | 5 | /27 | 255.255.255.224 |
| **64** | **6** | **/26** | **255.255.255.192** |
| 128 | 7 | /25 | 255.255.255.128 |
| 256 | 8 | /24 | 255.255.255.0 |
| 512 | 9 | /23 | 255.255.254.0 |
| 1024 | 10 | /22 | 255.255.252.0 |
| 2048 | 11 | /21 | 255.255.248.0 |
| 4096 | 12 | /20 | 255.255.240.0 |
| 8192 | 13 | /19 | 255.255.224.0 |
| 16384 | 14 | /18 | 255.255.192.0 |
| 32768 | 15 | /17 | 255.255.128.0 |
| 65536 | 16 | /16 | 255.255.0.0 |
| 131072 | 17 | /15 | 255.254.0.0 |
| 262144 | 18 | /14 | 255.252.0.0 |
| 524288 | 19 | /13 | 255.248.0.0 |
| 1048576 | 20 | /12 | 255.240.0.0 |
| 2097152 | 21 | /11 | 255.224.0.0 |
| 4194304 | 22 | /10 | 255.192.0.0 |
| 8388608 | 23 | /9 | 255.128.0.0 |
| 16777216 | 24 | /8 | 255.0.0.0 |
| 33554432 | 25 | /7 | 254.0.0.0 |
| 67108864 | 26 | /6 | 252.0.0.0 |
| 134217728 | 27 | /5 | 248.0.0.0 |
| 268435456 | 28 | /4 | 240.0.0.0 |
| 536870912 | 29 | /3 | 224.0.0.0 |
| 1073741824 | 30 | /2 | 192.0.0.0 |

Příliš málo adres

Nejmenší síť se skládá ze dvou hostů + adresa sítě a broadcastu, nejdelší použitelný prefix je tedy /30.

*Příklad:*




**Obrázek 39 – Topologie vytváření podsítí s variabilní maskou**

Adresa sítě pro vytváření podsítí: **192.168.10.0/24**

**Adresa 0. podsítě** - síť A (60 hostů): 192.168.10.0/26

- Vyplýtváno: 2 adresy
- Adresa prvního hosta 0. podsítě: 192.168.10.1
- Adresa posledního hosta 0. Podsítě: 192.168.10.62
- Adresa broadcastu 0. podsítě: 192.168.10.63

Značný rozdíl je znát už u sítě C. Jelikož do této podsítě patří pouze 10 hostů (+2 adresy), postačí prefix /28, který dává k dispozici 16 adres. Plýtvá se tedy pouze čtyřmi adresami. U stejně velkých podsítí se plýtvalo 52 adresami.

**Adresa 1. podsítě** – síť C (10 hostů): 192.168.10.64/28

- Vyplýtváno: 4 adresy
- Adresa prvního hosta 1. podsítě: 192.168.10.65
- Adresa posledního hosta 1. Podsítě: 192.168.10.78
- Adresa broadcastu 1. podsítě: 192.168.10.79

Nejvíce se plýtvalo u sítě s označením B, konkrétně 60 adresami. V tomto případě je zvolen prefix /30, který dává k dispozici 4 adresy na podsíť (2 hosté + 2). Díky tomu k žádnému plýtvání nedošlo.

**Adresa 2. podsítě** – síť B (2 hosté): 192.168.10.80/30

- Vyplýtváno: 0 adres
- Adresa prvního hosta 2. podsítě: 192.168.10.81
- Adresa posledního hosta 2. podsítě: 192.168.10.82
- Adresa broadcastu 2. podsítě: 192.168.10.83

U každého vytváření podsítí je také třeba dávat pozor na počet dostupných adres v každé podsíti. Nejdelší možný prefix je /30. Jelikož nejmenší síť je vždy složena ze dvou zařízení a k nim je třeba síťová a broadcastová adresa. Nejmenší nutný počet dostupných adres je tedy čtyři. Proto by nebylo například pro síť 192.168.10.0/24 možné vytvořit podsítě, které by měly k dispozici 300 adres pro hosty. K dispozici je totiž maximálně 6 bitů z hostitelské části, což dává maximální počet $2^6 = 64$ podsítí o 4 dostupných adresách a z toho pouze 2 pro hosty. Pro vytvoření většího počtu podsítí než je 64 by bylo třeba mít k dispozici větší adresní prostor, například 192.168.10.0/16. To samé platí, i kdybychom požadovali vytvoření podsítí sítě 192.168.10.0/24 o požadovaném počtu hostitelských adres přesahujícím 126 (+ adresa sítě a adresa broadcastu).

## 12.2 Vytváření podsítí v IPv6

U IPv4 byl hlavním důvodem vytváření podsítí omezený adresní prostor, u IPv6 je důvod jiný. IPv6 poskytuje neskutečně veliký adresní prostor a proto je třeba v něm uchovat jisté logické uspořádání, hierarchii. Tato hierarchie je postavena na počtu směrovačů a sítí, do kterých patří.

Jak bylo zmíněno v předchozí části, poskytovatelé internetových služeb přidělují svým zákazníkům globální směrovací prefix /48 a zákazníci si poté dle vlastních potřeb mohou vytvářet podsítě. K tomuto účelu slouží dalších 16 bitů, které slouží jako ID podsítě. Díky těmto 16 bitům je možnost utvořit až $2^{16} = 65536$ podsítí, kde každá z nich má prostor pro $2^{64} = 1{,}8 \times 10^{19}$ adres. [12]

Podsítě jsou poté vytvářeny tak, že v ID podsítě se začne od nejnižší hodnoty, tedy 0000 v hexadecimální soustavě a pro každou podsíť je tato hodnota zvýšena o 1 až do hodnoty FFFF, která znamená poslední 65536. podsíť.

Přiřazený globální směrovací prefix:

2001:5A2D:104B::/48

Adresa první podsítě:

2001 : 5A2D : 104B : 0000 :: /64

Globální směrovací prefix ID podsítě

**Další podsítě:**

2001:5A2D:104B:0001::/64

2001:5A2D:104B:0002::/64

2001:5A2D:104B:0003::/64

2001:5A2D:104B:0004::/64

…

2001:5A2D:104B:000A::/64

2001:5A2D:104B:000B::/64

2001:5A2D:104B:000C::/64

2001:5A2D:104B:000D::/64

….

2001:5A2D:104B:FFFC::/64

2001:5A2D:104B:FFFD::/64

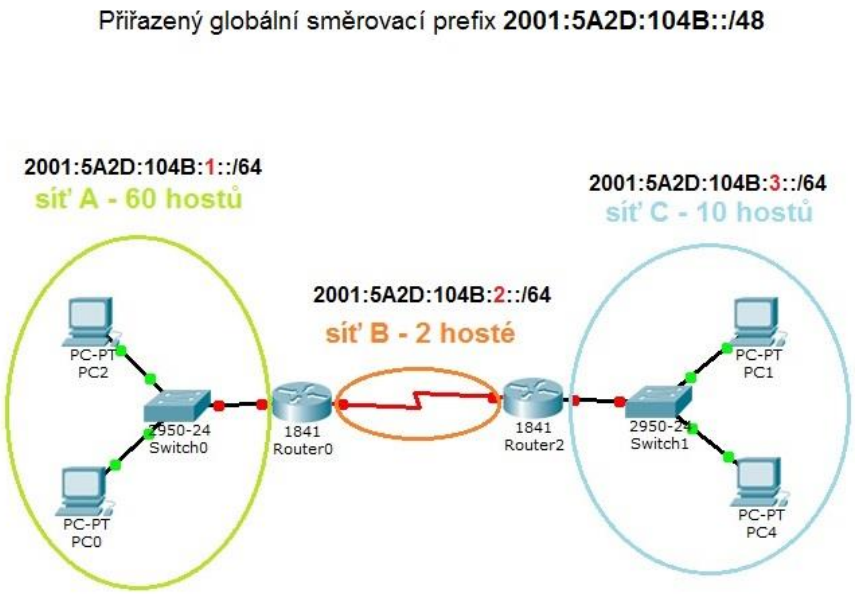
2001:5A2D:104B:FFFE::/64

Poslední podsíť:

2001:5A2D:104B:FFFF::/64

*Příklad:*

U následující topologie je třeba vytvořit tři podsítě A, B, C. Podsíť A má být tvořena 60 hosty, podsíť B dvěma hosty a podsíť C deseti hosty. U IPv6 není třeba brát ohledy na plýtvání adresami jako u IPv4, proto bude každé podsíti přiřazena adresa s adresním prostorem pro $2^{64} = 1{,}8 \times 10^{19}$ adres.




**Obrázek 40 – Vytváření podsítí IPv6**

Je k dispozici globální směrovací prefix **2001:5A2D:104B::/48**. Podsítě jsou vytvořeny jako v předchozí části, pouze jsou vynechány zbytečné nuly.

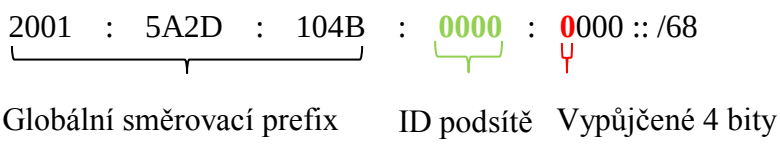
| | |
|---|---|
| Podsíť A: | **2001:5A2D:104B:1::/64** |
| Podsíť B: | **2001:5A2D:104B:2::/64** |
| Podsíť C: | **2001:5A2D:104B:3::/64** |

Existuje i možnost využití bitů z hostitelské části adresy IPv6, tedy části, která je nazývána jako ID rozhraní. Výsledkem je více podsítí a méně hostitelských adres v podsíti. Tento způsob však není nutné používat, jelikož IPv6 poskytuje dostatečné množství adres a předchozí varianta byla dostačující. Důvodem, proč může být tento způsob použit, jsou například bezpečnostní důvody.

Tato varianta je realizována tak, že jsou z ID rozhraní vypůjčovány vždy celé hexadecimální cifry, tedy po 4 bitech. Délky prefixů potom tedy mohou být kromě /64 i /68, /72, /76 apod.

U předchozí adresy by vypůjčení jedné hexadecimální cifry vypadalo takto:

Pro jeden identifikátor podsítě je vytvořeno více podsítí než v předchozím případě. A délka prefixu již není /64, ale /68.




2001:5A2D:104B:0000:1000::/68

2001:5A2D:104B:0000:2000::/68

2001:5A2D:104B:0000:3000::/68

…

2001:5A2D:104B:0000:D000::/68

2001:5A2D:104B:0000:E000::/68

2001:5A2D:104B:0000:F000::/68

…

2001:5A2D:104B:EEEE:1000::/68

2001:5A2D:104B:EEEE:1000::/68

…

2001:5A2D:104B:FFFF:E000::/68

2001:5A2D:104B:FFFF:F000::/68

# 13 Vrstva síťového rozhraní

Tato vrstva je poslední vrstvou modelu TCP/IP. Jejím úkolem je řídit přístup k přenosovému médiu. Její funkčnost odpovídá logické vrstvě ISO/OSI modelu.

**Tabulka 10 – Poslední vrstvy vrstvových modelů**

| ISO/OSI | | TCP/IP | |
|---|---|---|---|
| **vrstva** | **název vrstvy** | **vrstva** | **název vrstvy** |
| 7 | Aplikační | 4 | Aplikační |
| 6 | Prezentační | | |
| 5 | Relační | | |
| 4 | Transportní | 3 | Transportní |
| 3 | Síťová | 2 | Internetová |
| 2 | Linková | 1 | Vrstva síťového rozhraní |
| 1 | Fyzická | | |

Vrstva síťového rozhraní je zodpovědná za řízení přístupu k fyzickému médiu, aby poté mohly pakety přijaté z vyšší vrstvy být přes toto médium ve formě bitů přeneseny. Zařízení, která jsou **přímo připojena** k přenosovému médiu, se nazývají **uzly**, mezi nimi poté prostřednictvím tohoto média dochází k samotnému přenosu. Díky vrstvě síťového rozhraní jsou vyšší vrstvy nezávislé na použitém přenosovém médiu. Tato nezávislost je umožněna díky zapouzdřování paketů do **rámců** přidáním příslušné hlavičky s řídícími informacemi a patičky s kontrolními mechanismy.

**Tabulka 11 – Rámec**

| Hlavička rámce | **PAKET** | Patička rámce |
|---|---|---|

Rámec obsahuje důležité informace zajišťující přístup k danému médiu. Na každém směrovači je po přijetí tento rámce rozbalen, znovu zapouzdřen do příslušného rámce, který je uzpůsoben typu přenosového média, a poté je přeposlán dál. Daný rámec je přenosovým médiem přenášen ve formě nul a jedniček, je však nutné rozlišit, které bity označují konec a začátek rámce a které slouží k jinému účelu. Z tohoto důvodu obecný rámec obsahuje

několik polí, kde každé z nich má svůj účel. Tento tvar je však pouze obecný, u konkrétních protokolů vrstvy síťového rozhraní se tato pole mohou lišit.

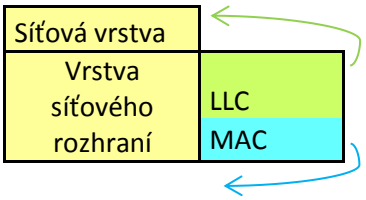
**Tabulka 12 – Obecná pole rámce**

| Začátek rámce | Adresace | Typ | Kontrolní pole | Data | Detekce chyb | Konec rámce |
|---|---|---|---|---|---|---|

- **Začátek rámce** – slouží k identifikaci začátku rámce
- **Adresace** – slouží k identifikaci zdrojových a koncových uzlů v síti
- **Typ** – určuje použitý protokol síťové vrstvy
- **Kontrolní pole** – speciální služby pro řízení přenosu
- **Data** – samotný paket
- **Detekce chyb** – pole sloužící k detekci chyb, kontrolní součet
- **Konec rámce** – identifikuje konec rámce

Nezávislost síťové vrstvy na typu přenosového média je také umožněna díky rozdělení vrstvy síťového rozhraní na dvě podvrstvy. Vyšší softwarově realizovaná podvrstva **Logical Link Control (LLC)** poskytuje služby síťové vrstvě a naopak nižší hardwarově realizovaná podvrstva **Media Access Control (MAC)** slouží k zpřístupnění daného přenosového média fyzické vrstvy.

**Tabulka 13 – Podvrstvy vrstvy síťového rozhraní**



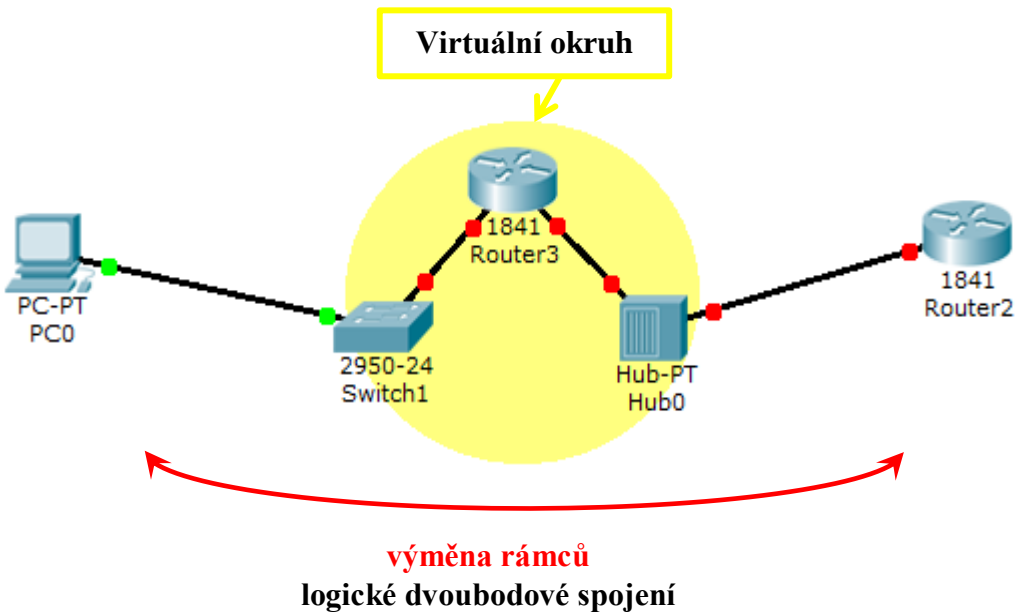

Jak již bylo zmíněno, LLC je implementovaná softwarově, nejčastěji se jedná o ovladač síťové karty NIC (Network Interface Controller) v počítači. Ovladač je obecně počítačový program, který umožňuje komunikaci operačního systému s hardwarem. Zde se jedná konkrétně o komunikaci se síťovou kartou počítače, za účelem přenesení dat mezi podvrstvou MAC a fyzickým médiem. Podvrstva LLC se stará o již zmíněné zapouzdřování paketů do rámců a určuje, pro který protokol síťové vrstvy je rámec určen. Dalším úkolem této podvrstvy je zabránit chybám a v případě výskytu chyby o ní informovat a pokusit se ji opravit. MAC podvrstva je realizována hardwarově ve formě integrovaných obvodů síťové karty, která může být přímo integrovaná na základní desce počítače nebo v podobě zásuvné

karty, která je zapojována do příslušného slotu na základní desce. MAC podvrstva identifikuje začátek a konec rámce na základě příslušných polí v rámci. Dále kontroluje, zda je rámec určen pro dané zařízení a provádí adresaci fyzickou adresou. Jejím úkolem je také převést data do podoby nul a jedniček, aby mohla být předána danému fyzickému médiu, řídí přístup k médiu a stará se o volbu vhodného protokolu vrstvy síťového rozhraní, který bude použit. [10], [23]

## 13.1 Přístup k médiu

To, jakým způsobem je řízen přístup k médiu, je dáno použitou topologií, konkrétně logickou, a jakým způsobem je médium mezi uzly v síti sdíleno. Pro připomenutí, logická topologie je nezávislá na fyzickém uspořádání zařízení v síti, je charakterizována tím, jak data (rámce) putují sítí.

Nejběžnější logickou topologií je **dvoubodové spojení** (point-to-point). Fyzicky se může se jednat o dva uzly v síti propojené prostřednictvím několika mezilehlých zařízení, které se navenek tváří jako dvoubodové spojení. Je vytvořen jakýsi **virtuální okruh**. Příklad point-to-point topologie s virtuálním okruhem je uveden na obrázku níže (Obrázek 41).



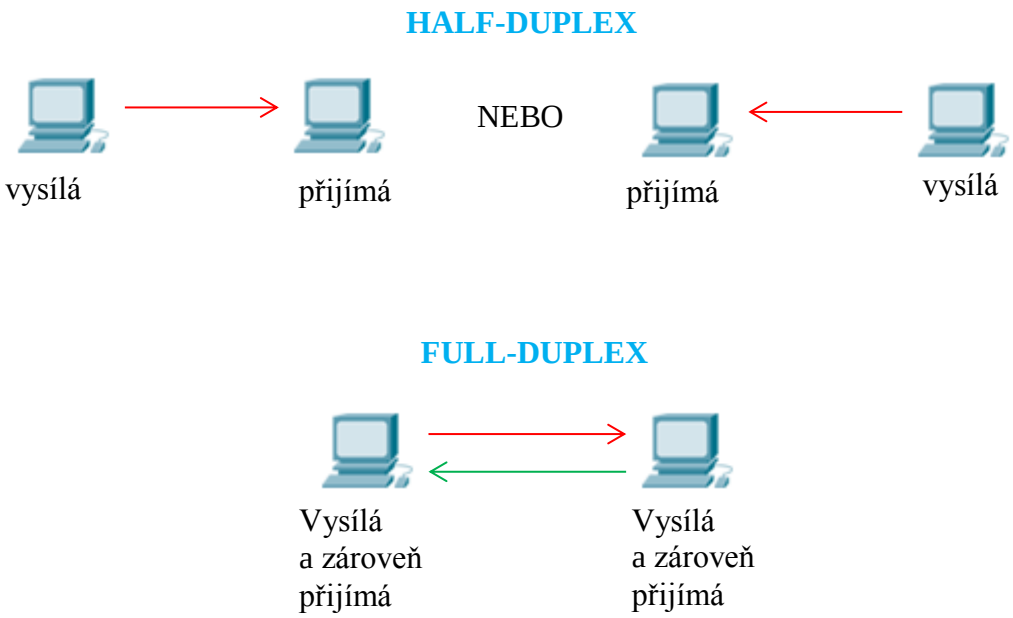

**Obrázek 41 – Virtuální okruh**

U dvoubodového spojení jsou rozlišovány dva typy komunikace – plno a polo-duplexní mód. Na obou zařízeních musí být mód stejný.

**Full-duplex** – v jednu a tu samou chvíli mohou vysílat obě koncová zařízení, není nutné žádným způsobem řídit přístup k médiu.

**Half-duplex** – v jednu a tu samou chvíli smí vysílat pouze jedno koncové zařízení, z tohoto důvodu je třeba zavést mechanismy pro řízení přístupu k médiu. Konkrétně se jedná o metodu CSMA/CD, která bude uvedena později.

Na obrázku níže (Obrázek 42) je uveden rozdíl mezi half-duplex a full-duplex módem.




**Obrázek 42 – Rozdíl half-duplex a full-duplex**

U topologie typu **sběrnice** je jedno přenosové médium sdíleno několika uzly, což znamená, že každý rámec, který je poslán přes toto médium, přijmou všechny uzly k danému médiu připojené. U dvoubodového spojení byl rozlišován přístup half-duplex a full-duplex, v případě, že jedno médiu sdílí více zařízení, je třeba řídit přístup jiným způsobem. Konkrétně existují dvě varianty řízení přístupu k médiu. Jedná se o deterministickou a nedeterministickou variantu. [14], [23]

**Deterministický přístup** (řízený přístup) – jednotlivá zařízení se mohou pokusit o přístup k médiu pouze ve chvíli, kdy přijdou na řadu. Zařízení, které je na řadě, má jakýsi **token**. Jakmile dané zařízení již nepotřebuje přistupovat k médiu, je token uvolněn pro umožnění přístupu dalšímu zařízení. V jednu a tu samou chvíli smí vždy k médiu přistupovat pouze jedno zařízení, díky čemuž nedochází ke kolizím, tedy nedojde ke střetu signálů dvou či více

rámců. Tato metoda má však díky řídícím mechanismům vysokou režii a díky častému čekání zařízení na povolení přístupu k médiu je i neefektivní. Mezi technologie využívající tento způsob přístupu patří Token Ring (IEEE[22] 802.5) a Fiber Distributed Data Interface (FDDI) neboli standard IEEE 802.4.

**Nedeterministický** neboli **stochastický přístup** (soutěžení o přístup) – zařízení se může pokusit o přístup k médiu kdykoliv, když je připraveno. Může se však stát, že v jednu a tu samou chvíli se pokusí o přístup k médiu více zařízení a dojde ke kolizi. Čím více zařízení sdílí médium, tím jsou kolize častější, avšak díky absenci řídících mechanismů má tento přístup oproti předchozímu menší režii. U nedeterministického přístupu existují dvě metody, které určují, jakým způsobem zařízení o médium soupeří a co se stane v případě přístupu více zařízení k médiu v jednu a tu samou chvíli. Jedná se o metody CSMA/CA a CSMA/CD.

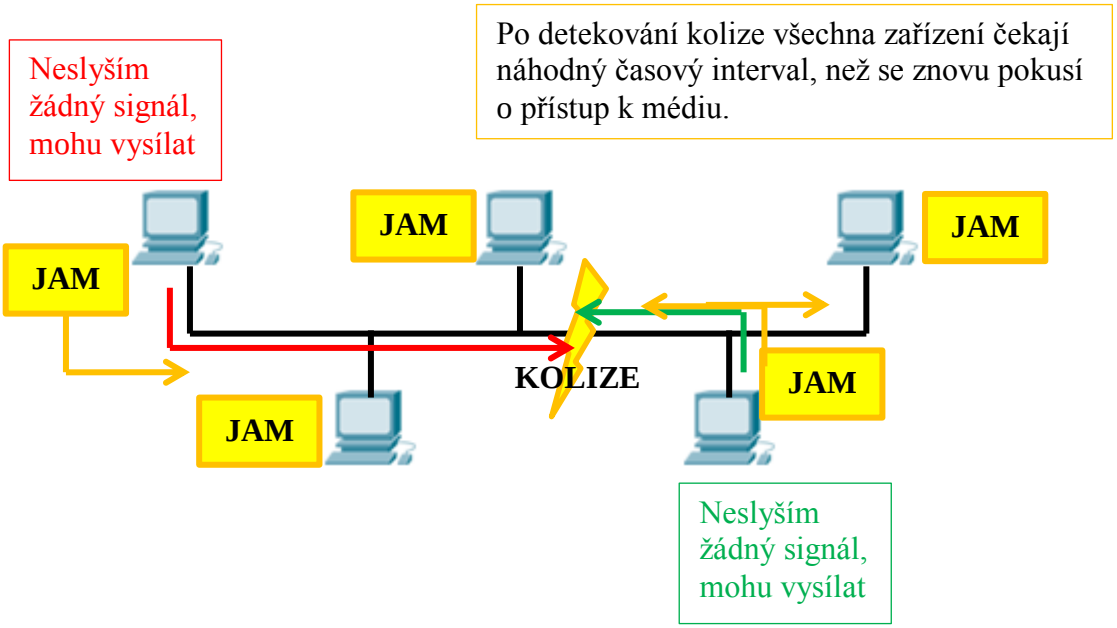
Obě metody mají společnou část **CSMA**, která znamená, že v obou případech nejprve zařízení, které se chce pokusit o přístup k médiu, zkoumá, zda je z média slyšitelný nějaký signál. Pokud ano, znamená to, že v danou chvíli vysílá jiné zařízení a je třeba počkat jistý časový okamžik, než bude možné se znovu pokusit o přístup k médiu. Délka čekání se u každého zařízení liší. Toto zkoumání média je jinak nazýváno jako **listen-before-transmit**. Jakmile zařízení zjistí, že žádný signál nepřichází, je médium volné a může se pokusit o přístup k médiu. V následujícím kroku se dané metody odlišují.

**CSMA/CA** (Carrier Multiple Access with Collision Avoidance) – jedná se o metodu, která se snaží kolizím zabránit. Zařízení, které se chce pokusit o přístup k volnému médiu, nejprve zašle oznámení prostřednictvím média o tom, že hodlá dané médium využít. Jakmile obdrží povolení, začne posílat data. Tato metoda je využívána především bezdrátovou technologií IEEE 802.11, která bude uvedena později.

**CSMA/CD** (Carrier Multiple Access with Collision Detection) – jedná se o metodu, která detekuje chyby v okamžiku jejich výskytu, nesnaží se jim však příliš předejít. Jakmile zařízení zjistí, že je médium volné, nezasílá oznámení a úmyslu dané médiu využít a místo toho začne rovnou vysílat. O tom, že je médium k dispozici, se však může v danou chvíli dozvědět více zařízení, která se poté ve stejnou chvíli pokusí dané médium využít pro přenos dat. Za této situace je **detekována kolize**. Vysílající zařízení vyšlou všem zařízením **kolizní**

[22] IEEE (Institute of Electrical and Electronics Engineers) – organizace vydávající standardy pro různé technologie, nejznámější jsou standardy IEEE 802 LAN/WAN.

**rámec**, tedy informaci o vzniklé kolizi (tzv. **jam signál**), a přestanou vysílat. Stejným mechanismem se poté pokusí o přístup k médiu znovu. Každé zařízení však čeká náhodný časový interval, než dojde k dalšímu pokusu. Tato metoda je využívána u technologie typu Ethernet, která bude uvedena později. [10], [23]




**Obrázek 43 – Kolize CSMA/CD**

Zařízení uvedená na obrázku (Obrázek 43) tvoří **kolizní doménu**, což je část sítě, ve které dochází ke kolizi, jakmile se pokusí vysílat více zařízení najednou. Tato část sítě se skládá ze sdíleného přenosového média a k němu připojených uzlů. Větší kolizní doména znamená větší výskyt kolizí, proto je třeba mít kolizní domény co nejmenší. Jak toho ale docílit?

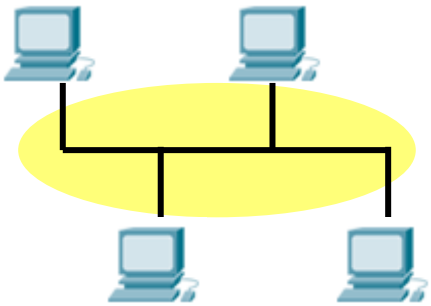
Je možné použít mezilehlé zařízení, například **router (směrovač)**, který síť rozdělí více menších podsítí, tedy i menších kolizních domén, vhodnější je však použití **switche (přepínače)**. Přepínač pracuje na vrstvě síťového rozhraní, jedná se o chytré zařízení, které zasílá rámce pouze těm zařízením, kterým jsou určeny. Kromě toho umí i rozdělit kolizní doménu na více menších kolizních domén, které přísluší jednotlivým portům přepínače. Je zapojen jako centrální prvek, čímž vznikne fyzicky topologie typu hvězda, logicky však tuto kolizní doménu rozdělí na menší části, které jsou méně náchylné na kolize, konkrétně vzniknou dvoubodová spojení.

Stejně tak existuje i mezilehlé zařízení, které kolizní doménu nedělí, tímto zařízením je **hub**. Jedná se o „hloupé“ zařízení, které zašle rámec na všechna zařízení mimo toho, od kterého rámec obdržel. Hub je zařízení, které slouží k propojení jednotlivých uzlů, ale kolizní doménu rozdělit neumí. [10], [16], [26]

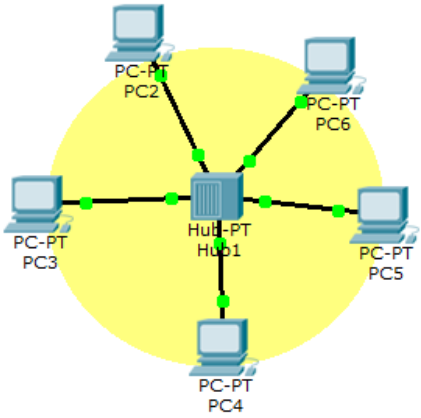
Na níže uvedených obrázcích (Obrázek 44, Obrázek 45) jsou uvedeny kolizní domény při použití různých mezilehlých zařízení. Jednotlivé kolizní domény jsou znázorněny žlutou barvou.

**Sdílené přenosové médium (sběrnice)**

**Jedna kolizní doména**



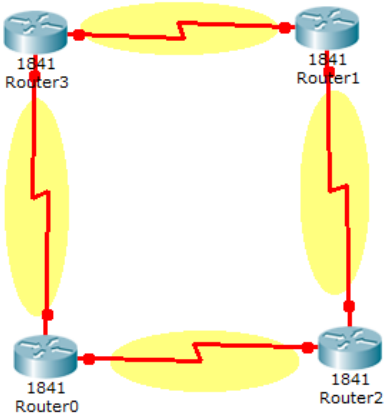
**Použití hubu**
**Opět sdílené přenosové médium (fyzicky hvězda, logicky sběrnice)**
**Jedna kolizní doména**



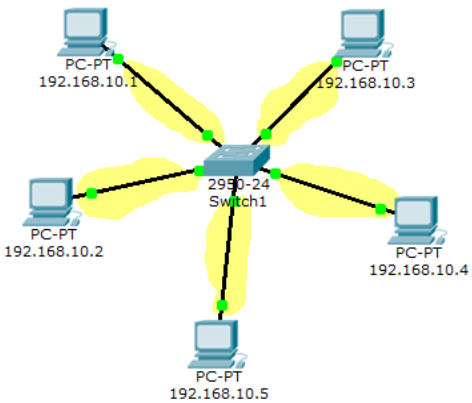

**Obrázek 44 – Kolizní doména u sběrnice a hubu**

**Použití směrovače (router)**

**Čtyři kolizní domény**

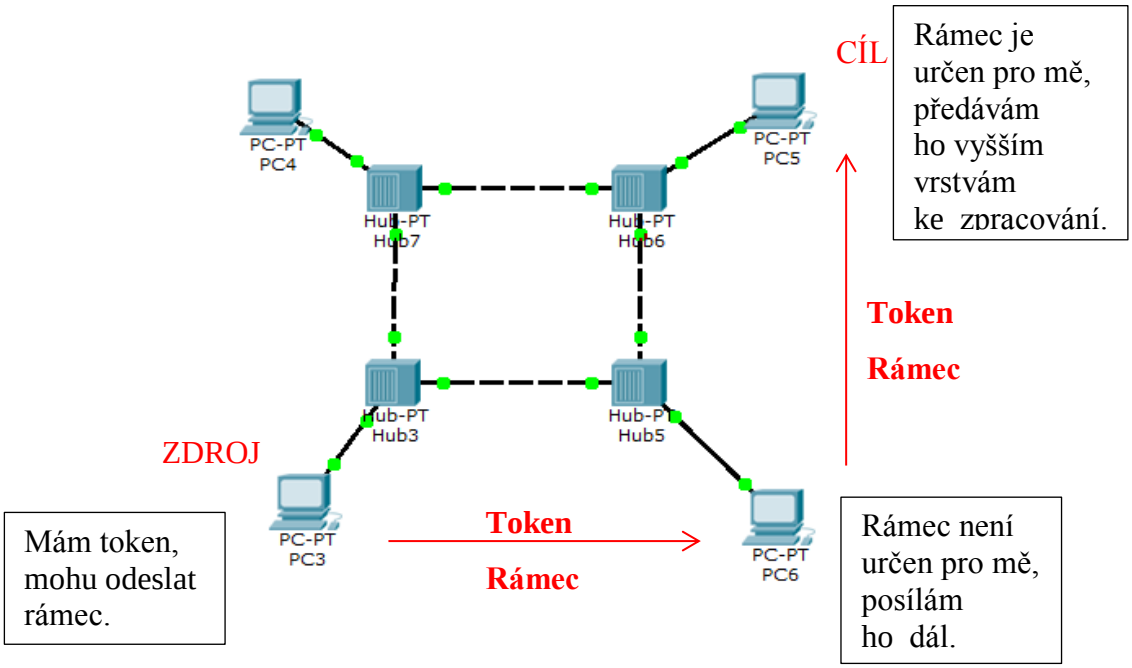



**Použití přepínače (switch)**
**(fyzicky hvězda, logicky dvoubodová spojení)**
**Pět kolizních domén**




**Obrázek 45 – Kolizní doména u routeru a switche**

U logické **kruhové topologie** je rámec přijat zařízením, které je na řadě (má token), zde je zjištěno na základě fyzické adresy v rámci, zda je určen pro dané zařízení, pokud ne, je předán token vedlejšímu zařízení, které přijme i daný rámec. Rámec je přeposílán do té doby, než dojde do cíle, kde může být předán ke zpracování vyšším vrstvám. [23]




**Obrázek 46 – Rámce v kruhové topologii**

# 14 Protokoly vrstvy síťového rozhraní

Jak již bylo zmíněno v kapitole týkající se úvodu do TCP/IP, poslední vrstva tohoto modelu, tedy vrstva síťového rozhraní, nemá jednotnou standardizaci funkcí a služeb definovaných na této vrstvě. Protokoly tedy obecně nejsou definovány v RFC dokumentech vydávaných IETF. Existuje však několik standardů, které jsou vydávány různými organizacemi v závislosti na konkrétních přenosových technologiích. Mezi tyto organizace patří například IEEE (Institute of Electrical and Electronisc Engineers), ITU (International Telecommunication Union), ISO (International Organization for Standardization) a ANSI (American National Standards Institute). Nejdůležitější standardy jsou uvedeny v tabulce (Tabulka 14) níže.

**Tabulka 14 – Standardy**

| Organizace | Standard |
|---|---|
| IEEE | 802.2: Logical Link Control |
| | 802.3: Ethernet |
| | 802.11: Wireless LAN |
| | 802.15 Bluetooth |
| ITU | ADSL |
| | Frame Relay |
| ISO | HDLC |
| ANSI | Fiber Distributed Data Interface (FDDI) |

Který protokol bude použit, je závislé na logické topologii a konkrétním přenosové technologii. Použitá přenosová technologie je závislá na počtu zařízení v dané topologii, geografické oblasti, do které daná topologie spadá, a na poskytovaných službách. Například LAN mají díky omezené geografické rozlehlosti typicky vyšší přenosovou kapacitu[23] a naopak u sítí WAN je tato přenosová kapacita nižší z důvodu vysokých nákladů na vybudování linek pro přenos na větší vzdálenosti. Z tohoto důvodu jsou využity jiné protokoly pro LAN sítě a jiné pro sítě WAN. [9], [10], [23]

[23] Přenosová kapacita = maximální množství přenesených dat za jednotku času (uváděno typicky v bitech za sekundu)

## 14.1 Není Ethernet jako Ethernet

Ethernet je nejpoužívanější technologie v sítích LAN. Má několik forem, které se liší v přenosových rychlostech a použitých přenosových médiích. Ve všech případech má příslušný rámec, definovaný tímto standardem, stejnou podobu, jednotlivé formy Ethernetu se odlišují až způsobem, jakým umísťují data na konkrétní přenosové médium. Ethernet je využíván pro logické topologie typu **sběrnice**. Jak již bylo uvedeno v kapitole týkající se přenosových metod, u Ethernetu je využívána nedeterministická, tedy soutěžní, přístupová metoda **CSMA/CD**, která detekuje kolize v okamžiku jejich výskytu. V dnešní době však většina mezilehlých zařízení umí pracovat ve full-duplex módu, což značně snižuje výskyt těchto kolizí.

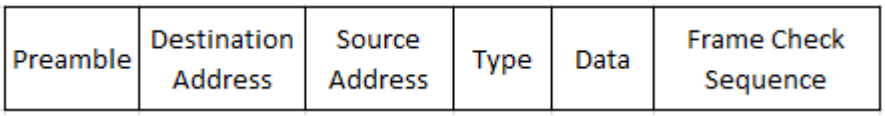
Jak již bylo uvedeno, Ethernet pracuje nad logickou topologií typu sběrnice, což znamená, že veškeré rámce zaslané přes dané přenosové médium, jsou přijaty všemi připojenými zařízeními. Rámec však nemusí být určen pro všechna tato zařízení, z tohoto důvodu je tedy třeba na úrovni vrstvy síťového rozhraní zavést adresaci. Na síťové vrstvě byly využívány logické IP adresy, na vrstvě síťového rozhraní se pracuje s adresami fyzickými. Ethernet k identifikaci koncových zařízení využívá **48bitovou MAC adresu**. Zdrojové a cílové MAC adresy jsou zapouzdřeny v hlavičce rámce. [10], [23]

V úvodu o Ethernetu bylo uvedeno, že existuje několik jeho forem, které jsou dány různými přenosovými rychlostmi a použitým přenosovým médiem. Ethernet prošel vývojem od roku 1973 (firma Xerox) až po současnost, ve kterých došlo ke změnám v použitých přenosových technologiích od obyčejných koaxiálních kabelů po optické linky a poskytovaných přenosových rychlostech, které se od původních přibližně 3 Mb/s posunuly až k rychlostem 100 Gb/s.

Existují dva důležité standardy **IEEE 802.3** a **Ethernet II**. Pro pochopení jejich rozdílu je však dobré si částečně přiblížit jejich vývoj. Za vznikem Ethernetu stály tři firmy – Digital, Intel, Xerox (dále jen DIX), které utvořily jeho základní podobu, a v roce 1980 vydaly první specifikace, které byly veřejně dostupné. Ethernet bylo však třeba standardizovat, o což se postarala organizace IEEE vydáním standardu IEEE 802.3. Tento standard měl však jisté nedostatky, které bylo třeba poupravit. Proto společnosti DIX vydaly nové specifikace tohoto protokolu, které byly v roce 1982 vydány pod názvem Ethernet II. V závislosti na těchto nových specifikacích poté vydala IEEE v roce 1985 nový standard IEEE 802.3 Carrier Sense

Multiple Access with Collision Detection Access Method and Physical Layer Specifications. Oba tyto standardy se od sebe odlišují strukturou rámců. Ethernet II se od dob svého vydání nezměnil a veškeré změny jsou vydávány organizací IEEE ve standardech IEEE 802.3, do kterých bylo začleněno i využívání rámců typu Ethernet II, jelikož byla vyráběna zařízení používající oba typy rámců a rámce typu Ethernet II jsou stále velmi využívány. Obecně je standard 802.3 nazýván jako Ethernet, ale formálně se tento název využívat nemohl, jelikož práva na používání tohoto názvu měla pouze firma Xerox. Od roku 2012 jsou však již standardy IEEE 802.3 nazývány jako „Standards for Ethernet“. V TCP/IP sítích je stále běžně využívána struktura rámců typu Ethernet II. [19]

### 14.1.1 **Rámec Ethernet II**




**Obrázek 47 – Rámec Ethernet II**

**Preamble** (8 bajtů) je pole, které slouží k synchronizaci uzlů, mezi kterými má dojít k přeposlání rámce. Slouží jako upozornění pro cílový uzel, aby se připravil na přijetí rámce.

**Destination Address** (6 bajtů) je pole obsahující MAC adresu cílového uzlu nebo cílových uzlů.

**Source Address** (6 bajtů) obsahuje MAC adresu zdrojového uzlu.

**Type** (2 bajty) určuje, pro jaký protokol síťové vrstvy je rámec po rozbalení na paket určen.

**Data** (46 až 1500 bajtů) obsahuje posílaná data (paket přijatý ze síťové vrstvy).

**Frame Check Sequence** (4 bajty) je pole, které slouží k detekci chyb. Obsahuje **kontrolní součet CRC** (Cyclic Redundancy Check). Kontrolní součet je spočítán odesílajícím uzlem a zapouzdřen v rámci. Cílový uzel po přijetí rámce také spočítá kontrolní součet, pokud neodpovídá údaji v patičce rámce, je tento rámec zahozen. [23], [26]

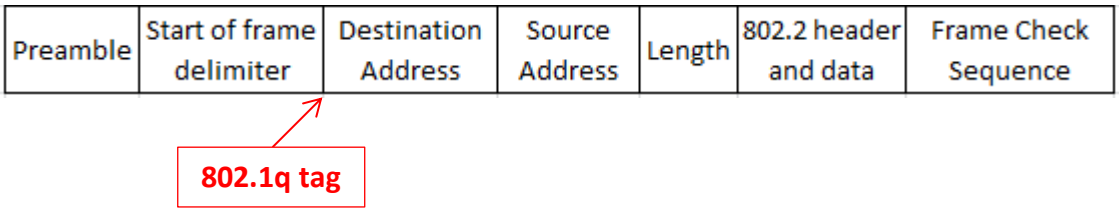
Na následujícím obrázku (Obrázek 48) je uveden rámec Ethernet II, který byl odchycen v programu Wireshark.

```
⊞ Frame 6: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
⊟ Ethernet II, Src: HonHaiPr_fd:a7:39 (68:94:23:fd:a7:39), Dst: TendaTec_05:5d:b0 (c8:3a:35:05:5d:b0)
  ⊟ Destination: TendaTec_05:5d:b0 (c8:3a:35:05:5d:b0)
      Address: TendaTec_05:5d:b0 (c8:3a:35:05:5d:b0)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  ⊟ Source: HonHaiPr_fd:a7:39 (68:94:23:fd:a7:39)
      Address: HonHaiPr_fd:a7:39 (68:94:23:fd:a7:39)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IP (0x0800)
⊞ Internet Protocol Version 4, Src: 192.168.0.103 (192.168.0.103), Dst: 157.55.56.148 (157.55.56.148)
⊞ Transmission Control Protocol, Src Port: 50462 (50462), Dst Port: 40030 (40030), Seq: 984, Ack: 5, Len: 0
```

**Obrázek 48 – Ethernet II Wireshark**

Ve výpisu nejsou zobrazena pole jako „preamble“ a „frame check sequence“. Je uvedena zdrojová MAC adresa, konkrétně se jedná o adresu 68-94-23-FD-A7-39. Jako cílová MAC adresa je uvedena adresa C8-3A-35-05-5D-B0. Kromě těchto adres je uvedeno i pole „Type“, jehož hodnota 0x0800 určuje, že má být paket po rozbalení rámce zpracován protokolem IPv4.

### 14.1.2 **Rámec Ethernet 802.3**




**Obrázek 49 – Rámec Ethernet 802.3**

```
⊞ Frame 65: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
⊟ IEEE 802.3 Ethernet
  ⊟ Destination: Spanning-tree-(for-bridges)_00 (01:80:c2:00:00:00)
      Address: Spanning-tree-(for-bridges)_00 (01:80:c2:00:00:00)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...1 .... .... .... .... = IG bit: Group address (multicast/broadcast)
  ⊟ Source: Cisco_6b:c0:81 (00:24:f7:6b:c0:81)
      Address: Cisco_6b:c0:81 (00:24:f7:6b:c0:81)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Length: 38
    Padding: 0000000000000000
⊟ Logical-Link Control
    DSAP: Spanning Tree BPDU (0x42)
    IG Bit: Individual
    SSAP: Spanning Tree BPDU (0x42)
    CR Bit: Command
  ⊟ Control field: U, func=UI (0x03)
      000. 00.. = Command: Unnumbered Information (0x00)
      .... ..11 = Frame type: Unnumbered frame (0x03)
⊞ Spanning Tree Protocol
```

**Obrázek 50 – Rámec 802.3 Wireshark**

**Preamble** (7 bajtů) a **Start of Frame Delimiter** (1 bajt) jsou pole odpovídající poli Preamble u Ethernetu II. Opět slouží k synchronizaci uzlů, mezi kterými má dojít k přeposlání rámce. Slouží jako upozornění pro cílový uzel, aby se připravil na přijetí rámce.

**Destination Address** (6 bajtů) a **Source Address** (6 bajtů) jsou opět pole obsahující MAC adresu cílového a zdrojového uzlu.

Konkrétně ve výstupu z Wiresharku (Obrázek 50) je zdrojovou adresou adresa 00-24-f7-6b-c0-81, cílovou adresou je 01-80-c2-00-00-00.

**Length** (2 bajty) má více účelů, které jsou závislé na hodnotě, kterou obsahuje. Pokud je hodnota menší nebo rovna 1500 (0x05DC), jedná se o rámec 802.3 a toto pole určuje velikost (délku) dat v rámci v bajtech. V případě, že je hodnota větší nebo rovna 1536 (0x0600), jedná se o rámec typu Ethernet II a určuje, jakým protokolem třetí vrstvy má být rozbalený rámec zpracován. Hodnota 0x0800 je použita pro IPv4 a hodnota 0x86DD pro IPv6.

Ve výstupu z Wiresharku (Obrázek 50) je možné vidět, že se jedná o hodnotu 38, tedy se jedná o rámec typu 802.3 a toto pole určuje velikost obsažených dat (38 bajtů). Minimální velikost dat je však 46 bajtů a celého rámce 64 bajtů (6 bajtů cílová adresa + 6 bajtů zdrojová adresa + 2 bajty délka + 4 bajty FCS + minimum 46 bajtů dat). Z tohoto důvodu existuje pole padding, které tuto velikost doplní.

**802.2 Header** a **Data** (46 až 1500 bajtů) opět obsahuje posílaná data (paket přijatý ze síťové vrstvy). Toto pole obsahuje jakýsi vnitřní rámec, konkrétně rámec **IEEE 802.2**, což je rámec horní podvrstvy LLC vrstvy síťového rozhraní, který obsahuje informace o tom, kterým protokolem síťové vrstvy byl paket zapouzdřen a zároveň jakým protokolem síťové vrstvy má být rozbalený rámec zpracován. Důvodem existence tohoto vnitřního paketu je sjednocení podoby Ethernetových rámců. Rámce se poté liší u jednotlivých typů protokolů způsobem, jakým jsou umísťovány na konkrétní přenosové médium.

Hlavička IEEE 802.2 byla odchycena v programu Wireshark (Obrázek 50). Skládá se z těchto polí:

- **DSAP** (Destination Service Access Point) – 8bitové pole, které identifikuje protokol, kterým má být rámec zpracován, konkrétně Spanning Tree Protocol.
- **SSAP** (Source Service Access Point) – 8bitové pole, identifikuje protokol, kterým byl rámec vytvořen, konkrétně Spanning Tree Protocol.

- **Control field** – 8bitové pole, případně 16bitové. Identifikuje formát paketu. Jedná se o Typ 1 (nespojový), Typ 2 (spojový, sekvenční číslo pro doručení ve správném pořadí) nebo Typ 3 (nespojový, pro dvoubodová spojení). Dále slouží také například pro kontrolu toku dat.

**Frame Check Sequence** (4 bajty) má stejný význam jako pole u Ethernetu II, slouží k detekci chyb a obsahuje **kontrolní součet CRC**.

Rámec IEEE 802.3 může navíc obsahovat 4bajtový **802.1Q** VLAN (Virtual Local Area Network) **tag**. Jeho struktura a dále použití VLAN bude představeno v části týkající se přepínaných sítí. [20], [23], [26]

## 14.2 Point-to-Point Protocol (PPP)

Tento protokol je využíván u dvoubodových spojení. Jedná o jeden z mála protokolů definovaných v RFC dokumentech, jedná se konkrétně o RFC 1661. Tento protokol je na rozdíl od Ethernetu využíván v sériových[24] WAN linkách. Poskytuje možnosti autentizace, komprese[25] dat, detekce chyb a kontroly kvality linky. Protokol PPP je složen ze dvou vrstev. První vrstvou je LLC (Link Control Protocol), která se stará o utvoření spojení a připravení linky pro přenos a detekci chyb. Druhou vrstvou je NCP (Network Control Protocol), která díky zapouzdřování umožňuje, aby na jedné lince pracovalo několik různých protokolů síťové vrstvy. Pro jednotlivé protokoly síťové vrstvy existují různé varianty protokolu NCP. [14], [23]

## 14.3 Wi-Fi (IEEE 802.11)

Jedná se o standard, u kterého není použito fyzické spojení, jedná se o bezdrátovou technologii. Tato technologie využívá nedeterministickou metodu přístupu k médiu CSMA/CA. Díky bezdrátovému přenosu může být přenos dat rušen okolními vlivy, například rádiovými signály, z tohoto důvodu jsou využívány kontrolní mechanismy, konkrétně potvrzovací rámce, které potvrzují doručení příslušného rámce do cíle. Dále je umožněna autentizace a šifrování dat. [10]

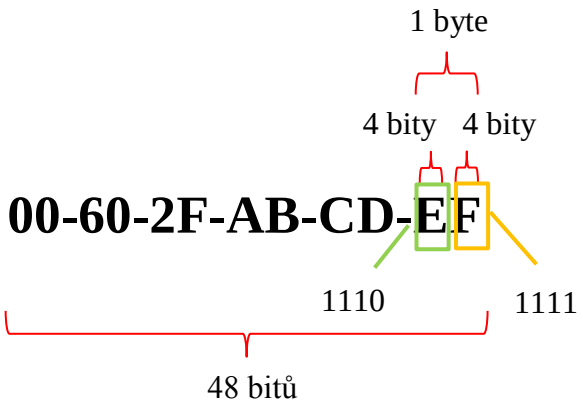
[24] Sériové linky – přenos bit po bitu za sebou
[25] Komprese dat = proces zmenšení velikosti dat, z typu komprese je odvozeno množství dat, které může nebo nemůže být ztraceno

# 15 MAC adresa

MAC adresa slouží k identifikaci uzlů na úrovni vrstvy síťového rozhraní, musí být pro každé zařízení unikátní. Tato adresa je hardwarově neměnná, každé zařízení ji má vypálenou v čipu paměti ROM (Read Only Memory), který je umístěn na síťovém adaptéru. Po spuštění počítače se však adresa nahrává do paměti RAM (Random Access Memory), což umožňuje její případnou změnu prostřednictvím softwaru.

Jedná se o 48bitovou adresu, která je uváděna v hexadecimálním tvaru. Jak je již známo z kapitoly týkající se protokolu síťové vrstvy IPv6, jedná hexadecimální číslice je tvořena 4 bity. MAC adresa je tedy reprezentována 12 hexadecimálními číslicemi, které mohou být zapisovány různými způsoby. [15], [23]

Nejčastěji:

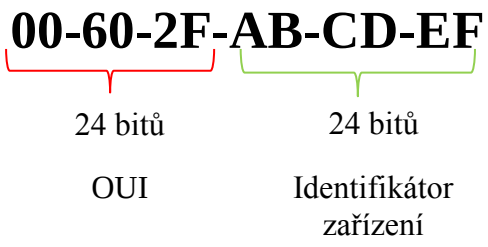



Další způsoby zápisu:

00:60:2F:AB:CD:EF

0060.2FAB.CDEF

Struktura MAC adresu je dány IEEE standardy. Prvních 24 bitů, tedy 6 hexadecimálních číslic, tvoří identifikátor **OUI (Organizationally Unique Identifier)**, který slouží k identifikaci prodejce, a dalších 24 bitů tvoří **unikátní identifikátor daného zařízení**.



MAC adresa počítače se dá zjistit prostřednictvím příkazového řádku. Konkrétně příkazem ***ipconfig /all***. [7]

```
C:\>ipconfig /all

Adaptér bezdrátové sítě LAN Bezdrátové připojení k síti:

   Přípona DNS podle připojení . . . :
   Popis . . . . . . . . . . . . . . : Broadcom 802.11n Network Adapter
   Fyzická Adresa. . . . . . . . . . : 68-94-23-FD-A7-39
   Protokol DHCP povolen . . . . . . : Ano
   Automatická konfigurace povolena  : Ano
   Místní IPv6 adresa v rámci propojení . . . :
fe80::f570:b1c7:68a8:9f36%13(Preferované)
   Adresa IPv4 . . . . . . . . . . . : 92.168.0.6(Preferované)
   Maska podsítě . . . . . . . . . . : 255.255.255.0
```

Jak probíhá samotná práce s MAC adresou při posílání rámců?

Vrstva síťového rozhraní, konkrétně podvrstva MAC, zařízení, které odesílá rámec, zapouzdří svou MAC adresu uvedenou ve své paměti RAM do rámce jako zdrojovou adresu a také uvede MAC adresu koncového uzlu nebo koncových uzlů. Poté je rámec odeslán prostřednictvím příslušného přenosového média. Daný rámec je přijat všemi zařízeními připojenými k danému přenosovému médiu. Každé toto zařízení prostřednictvím své podvrstvy MAC zjistí cílovou adresu uvedenou v hlavičce rámce a porovná ji se svou MAC adresou uvedenou v paměti RAM, pokud odpovídá, rámec je předán síťové vrstvě k dalšímu zpracování, pokud ne, je rámec zahozen.

Stejně jako logické IP adresy byly děleny na několik skupin, i MAC adresy jsou děleny do skupin dle cílových uzlů. K jednoznačné identifikaci hosta nebo hostů v síti je třeba, aby tito hosté měli přiřazenou logickou i fyzickou adresu. Logická adresa se mění v závislosti na tom, do jaké sítě je dané zařízení připojeno, fyzická adresa je vypálena v ROM síťové karty, je tedy neměnná. Z důvodu jednoznačné identifikace hosta nebo hostů je tedy třeba do rámce zapouzdřit MAC adresu, která koresponduje s IP adresou zapouzdřenou v paketu a umožní přenos rámce do cíle.

**Unicastové MAC adresy** jsou unikátní a slouží k posílání rámců z jednoho cílového uzlu do jednoho **cílového** uzlu.

**Broadcastová MAC adresa** je adresa **FF-FF-FF-FF-FF-FF**. Slouží k zaslání rámců na všechna zařízení v síti.

**Multicastové MAC adresy** slouží k zaslání rámců na skupinu několika cílových uzlů. Má speciální tvar. Prvních 24 bitů tvoří **01-00-5E** a dalších 24 bitů je tvořeno posledními 23 bity z IP adresy převedenými do hexadecimálního tvaru. [10], [15], [23]

*Příklad:*

Multicastové adresy jsou v rozsahu 224.0.0.0 až 239.255.255.255. V tomto případě bude použita adresa 230.0.5.100.

IP adresa decimálně: 230.230.5.100

IP adresa binárně: 11100110 11100110 00000101 01100100

0x6 0x6 0x0 0x5 0x6 0x4

Výsledná MAC adresa: 01-00-5E- 64-66-05-64

# 16 ARP (Address Resolution Protocol)

Address Resolution Protocol je protokol definovaný v RFC 826, který se stará o **mapování logických IP adres na fyzické MAC adresy**. Před zapouzdřením paketu do rámce je nejprve třeba zjistit zdrojovou a cílovou MAC adresu, která bude v daném rámci zapouzdřena. Zdrojová adresa je zjištěna z paměti RAM odesílajícího zařízení, konkrétně z tabulky ARP, někdy také nazývaná jako **ARP cache**. Tato tabulka obsahuje řádky s IP adresami a jejich příslušnými MAC adresami zařízení v rámci lokální sítě. Cílová MAC adresa je také zjištěna z této tabulky, pokud požadovaný záznam neexistuje, je třeba ho získat.

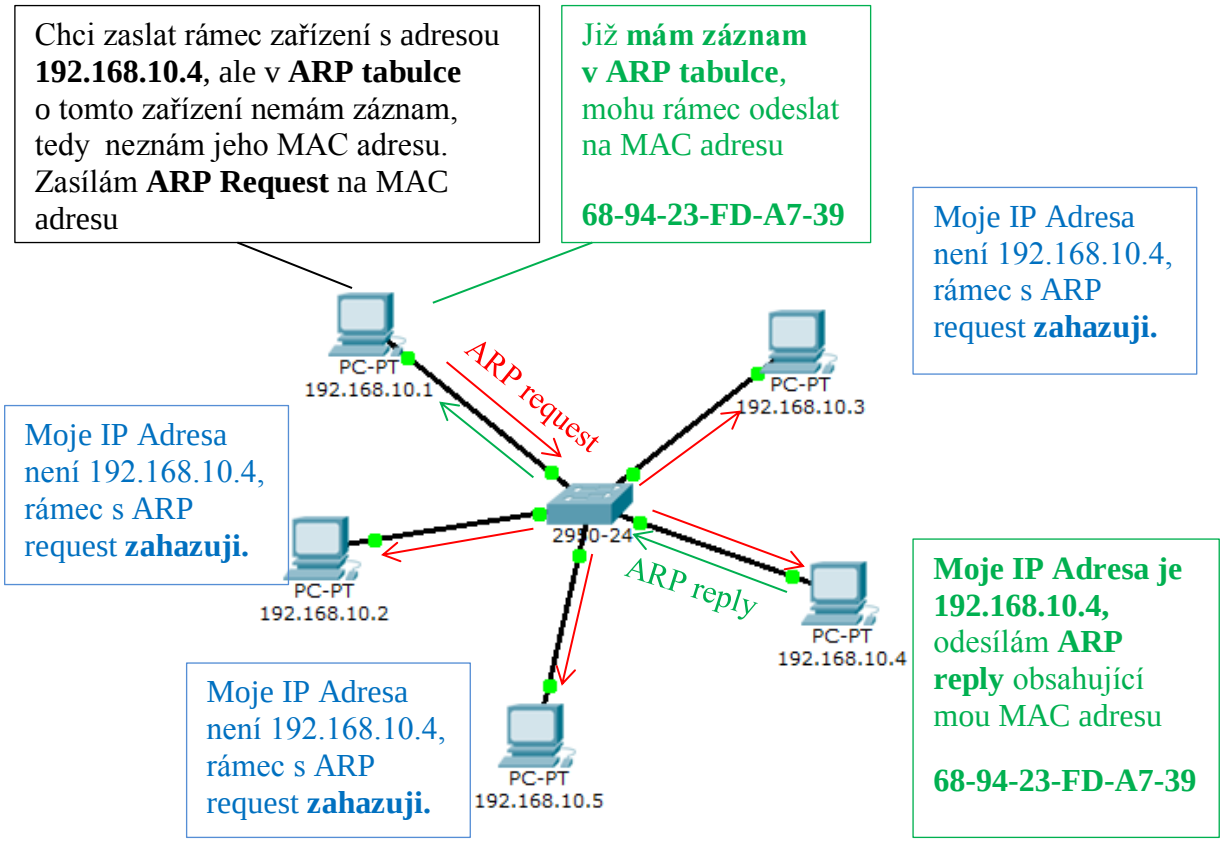
Jakým způsobem tabulka naplněna? Existují dvě možné cesty – **statická a dynamická.** Statická není příliš využívána, jedná se o případ, kdy jsou záznamy do tabulky přidávány ručně, o což se stará administrátor. Staticky přidané záznamy jsou v tabulce uvedeny do té doby, než jsou administrátorem opět smazány.

Druhá, tedy dynamická, varianta je využívána zcela běžně. Takto získané záznamy v tabulce však mají omezenou platnost (například 2 minuty), a pokud ze zařízení, které je uvedeno v tabulce, nedojde delší dobu k přijetí rámce, konkrétně do vypršení této platnosti, je tento záznam smazán.

U dynamické cesty jsou rozlišovány dva způsoby, jak záznamy získat. V případě, že dané zařízení není odesílajícím zařízením, může získat IP adresy a jim příslušné MAC adresy ze záznamů v přijatých rámcích. V případě, že chce dané zařízení odeslat rámec prostřednictvím přenosového média a neobsahuje v tabulce záznam o MAC adrese patřící k cílové IP adrese, je třeba pro zjištění této MAC adresy zaslat žádost na všechna zařízení v síti. [9], [23]

## 16.1 Princip ARP request/response

Dané zařízení zašle ARP žádost (**ARP request**) na broadcastovou MAC adresu **FF-FF-FF-FF-FF-FF**. Tato žádost obsahuje cílovou IP adresu zařízení, jehož MAC adresu je třeba zjistit. Žádost přijmou všechna zařízení v síti a porovnají IP adresu uvedenou v žádosti se svou IP adresou. V případě, že adresy neodpovídají, je daný rámec s žádostí zahozen, v opačném případě je zaslána unicastová ARP odpověď (**ARP reply**), která obsahuje MAC adresu daného zařízení, jehož IP adresa odpovídá IP adrese v ARP žádosti. Paket může být po zjištění cílové adresy zapouzdřen s konkrétní cílovou MAC adresou a přeposlán. Pokud by nepřišla žádná ARP odpověď s příslušnou MAC adresou, nemůže být paket do rámce zapouzdřen a je tedy zahozen a je generována ICMP zpráva o výskytu chyby.




**Obrázek 51 – Princip ARP request a ARP response**

Dynamické zjišťování IP adres prostřednictvím ARP žádostí však může mít vliv na výkon linky. V případě, že síť obsahuje velké množství zařízení, která v tu samou chvíli budou zasílat ARP žádosti pro naplnění svých ARP tabulek, bude linka zahlcena a dojde k jejímu zpomalení. Dalším problémem může být tzv. **ARP spoofing**. Jedná se o nežádoucí útok, kdy

je útočníkem zaslána falešná ARP odpověď, jež obsahuje MAC adresu, která ve skutečnosti neodpovídá IP adrese obsažené v ARP žádosti. V ARP tabulce poté vznikne záznam s nesprávnou MAC adresou, a veškeré rámce, určené odpovídající IP adrese, jsou zasílány na MAC adresu „podstrčenou“ útočníkem. Pokud dojde k takovémuto podstrčení u dvou uzlů v síti, které mezi sebou komunikují, a útočník bude dále oběma přeposílat přijaté rámce, daná zařízení se o tomto typu útoku nemusí dozvědět, komunikace mezi nimi tedy může být odposlouchávána. Proti tomuto typu útoku je možné se bránit statickým přidáváním záznamů do ARP tabulky. [10], [15], [26]

Obsah ARP tabulky lze u směrovače zjistit zadáním příkazu *show ip arp* v privilegovaném módu. [7]

```
Router#sh ip arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  192.168.10.1            -   0060.7066.B601  ARPA   FastEthernet0/0
```

Pokud se jedná o počítač s MS Windows, je možné obsah ARP tabulky zjistit zadáním příkazu *arp –a* do příkazového řádku.

```
C:\ >arp -a

Rozhraní: 192.168.0.103 --- 0xd
internetová adresa      fyzická adresa         typ
192.168.0.1             c8-3a-35-05-5d-b0     dynamická
192.168.0.104           78-e4-00-8c-2d-ab     dynamická
192.168.0.255           ff-ff-ff-ff-ff-ff     statická
224.0.0.22              01-00-5e-00-00-16     statická
224.0.0.252             01-00-5e-00-00-fc     statická
239.255.255.250         01-00-5e-7f-ff-fa     statická
255.255.255.255         ff-ff-ff-ff-ff-ff     statická

Rozhraní: 192.168.56.1 --- 0x1b
internetová adresa      fyzická adresa         typ
192.168.56.255          ff-ff-ff-ff-ff-ff     statická
224.0.0.22              01-00-5e-00-00-16     statická
224.0.0.252             01-00-5e-00-00-fc     statická
239.255.255.250         01-00-5e-7f-ff-fa     statická
```

# 17 Přepínané sítě

**Switch (přepínač)** je zařízení pracující na vrstvě síťovhé rozhraní (případně síťové) modelu. Jeho úkolem je **přepínání rámců** neboli jejich přeposílání v lokálních sítích. Zařízení, kterému má být rámec doručen, je identifikováno cílovou MAC adresou. Switch si podobně jako počítač uchovává **tabulku MAC adres**, na základě které může rámce posílat. Tato tabulka obsahuje jednotlivé porty přepínače a MAC adresy uzlů k daným portům připojených.

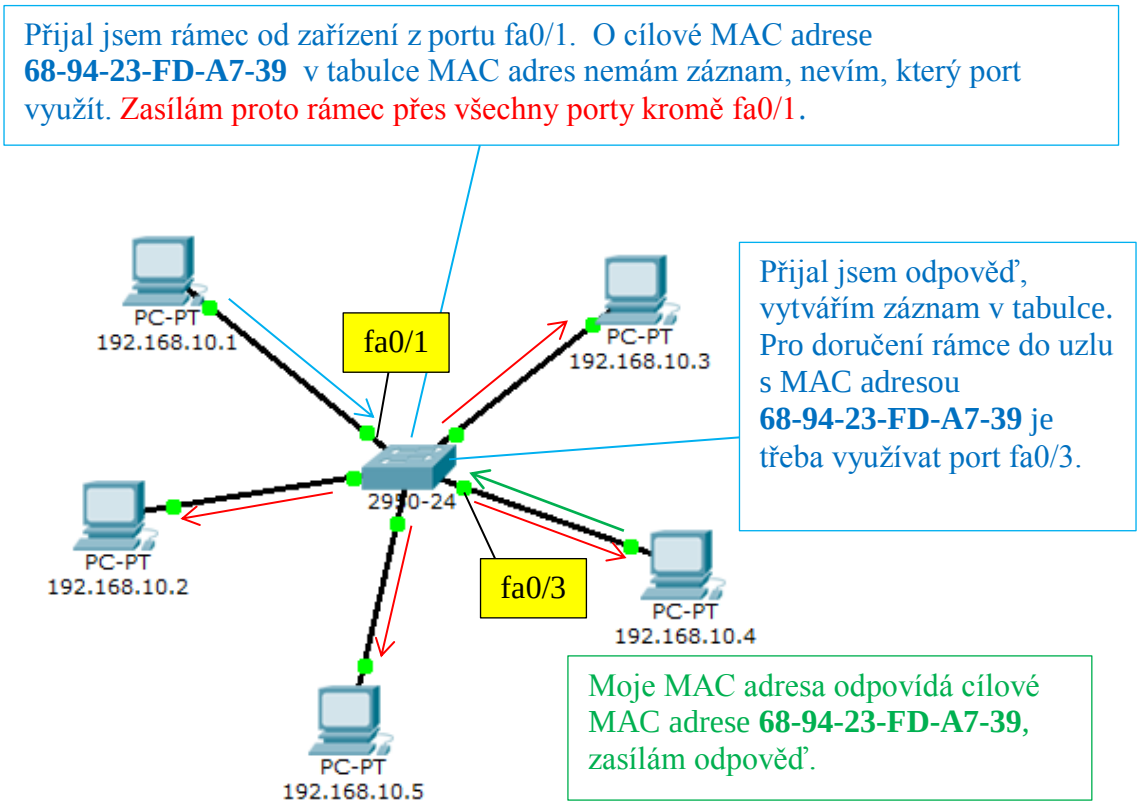
Obsah tabulky je možné vypsat příkazem [7]:

```
Switch#show mac-address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----

   1    0050.0f16.1362    DYNAMIC     Fa0/2
   1    0060.7066.b601    DYNAMIC     Fa0/3
```

Tabulka obsahuje číslo VLAN (pojem bude představen později), do které zařízení spadá, dále MAC adresu tohoto zařízení, zda byla adresa do tabulky přidána staticky či dynamicky a port, přes který má switch rámec určený pro dané zařízení poslat.

Switch je chytré zařízení, které se umí **„učit“**. Jakmile přijme rámec ze zařízení s MAC adresou, o které nemá přepínač záznam ve své tabulce MAC adres, ihned si daný záznam vytvoří. V daném záznamu je obsažena MAC adresa odesílatele rámce a port, na kterém byl rámec přijat. V případě, že přepínač nemá záznam o MAC adrese cílového uzlu, zašle rámec na všechny porty mimo toho, ze kterého rámec přijal. Jakmile přijde odpověď, vytvoří si záznam v tabulce obsahující MAC adresu uzlu společně s číslem portu, přes který je tento uzel k přepínači připojen. Tabulka může být opět naplněna nejen dynamicky, ale i staticky administrátorem sítě. Statické záznamy jsou uchovávány do té doby, než jsou administrátorem smazány, dynamické mají omezenou platnost. [14], [15]

Pro představu je na obrázku níže (Obrázek 52) uveden mechanismus vytváření záznamů v tabulce.




**Obrázek 52 – Vytváření MAC tabulky přepínače**

Switch je zařízení, které **dělí kolizní domény** na více menších kolizních domén, přičemž vznikají **logická dvoubodová spojení**. Výchozím nastavením Cisco switche je half-duplex mód, lze ho však nastavit na mód full-duplex či automatický režim, u kterého je switch schopen pracovat ve stejném módu jako druhé připojené zařízení. Níže uvedené příkazy uvádí jednotlivé možnosti konfigurace na rozhraní switche [7]:

```
Switch(config-if)#duplex half
Switch(config-if)#duplex full
Switch(config-if)#duplex auto
```

K tomu, aby mezi sebou zařízení mohla komunikovat, musí být oba uzly ve stejném módu a musí být použit správný kabel. Konkrétně se rozlišuje mezi přímým (pro zařízení na různých vrstvách ISO/OSI modelu) a kříženým (pro zařízení na stejné vrstvě). U některých Cisco přepínačů existuje možnost konfigurace rozhraní switche pro automatickou úpravu

jejich konfigurace v závislosti na správném typu kabelu. Tato konfigurace je zadána příkazem ***mdix auto*** v konfiguraci rozhraní přepínače [7].

```
Switch(config-if)#mdix auto
```

V takovém případě není nutné řešit zapojení správného typu kabelu. U novějších verzí přepínačů je tato funkce zapnutá automaticky, u starších je třeba tuto funkci výše uvedeným příkazem zapnout.

Existují dva způsoby, jakými switch zpracovává rámce, jinak řečeno **přepíná**. První metodou je store-and-forward, druhou je cut-through.

## 17.1 Metody přepínání rámců

**Store-and-forward** znamená, že switch nejprve přijme kompletní rámec, který ukládá do vyrovnávací paměti neboli **bufferu**. V případě, že byl před daným rámcem přijat do vyrovnávací paměti jiný rámec a ten ještě nebyl odeslán, řadí se nově přijatý rámec do fronty. Vyrovnávací paměť může být společná pro všechny porty nebo ji má každý port oddělenou. Po přijetí celého rámce se počítá kontrolní součet. V případě, že neodpovídá, je rámec zahozen, pokud je kontrolní součet v pořádku, je rámec na základě záznamu z tabulky MAC adres odeslán cílovému zařízení. Z důvodu kontrolních mechanismů v podobě kontrolního součtu je však tento typ o něco pomalejší.

**Cut-through** je metoda, kdy přepínač nevyužívá buffer. Přijatá data jsou rovnou přeposílána i v případě, že daný rámec nebyl ještě celý přijat. Před zahájením přeposílání je však nutné, aby switch přijal část rámce s cílovou MAC adresou, která je uvedena jeho začátku. V tomto případě nemá počítání kontrolního součtu význam, což sice zvyšuje rychlost přepínání, ale je také vyšší riziko výskytu chyb. Tento typ přepínání je dále dělen na fast-forward a fragment-free.

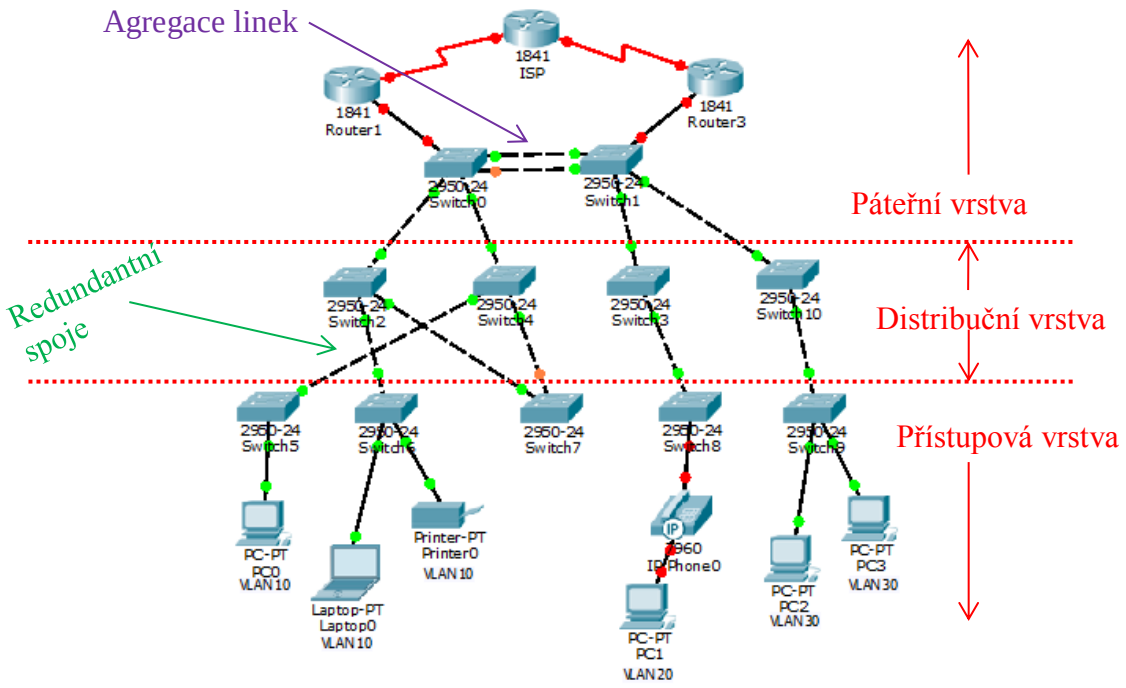
- **Fast-forward** je běžně používaný způsob přepínání, je velice rychlý. Jakmile přepínač zjistí cílovou MAC adresu, ihned začne odesílat data, která přijal. Může však nastat situace, kdy je rámec doručen poškozený, což je zjištěno po přepočítání kontrolního součtu, který je uveden v hlavičce paketu. V takovém případě je paket zahozen.
- **Fragment-free** je metoda, kdy switch přijímá prvních 64 bajtů rámce a až poté ho začne přeposílat. Tímto částečně dochází ke kontrole možného výskytu chyby, jelikož část obsahující prvních 64 bajtů je k chybám nejnáchylnější.

Kromě rozdělení způsobu, jakým přepínač přijímá a odesílá rámce, je dále rozlišováno **asymetrické a symetrické přepínání**. Tyto dva typy jsou odlišovány na základě přenosových rychlostí na portech přepínače. U symetrického přepínání je tato rychlost u všech portů totožná, u asymetrického se některé porty svojí rychlostí od ostatních liší.

Jak již bylo zmíněno v úvodu, switche mohou pracovat i na síťové vrstvě (L3 switch). V takovém případě se jedná o jakési směrování bez přítomnosti směrovače. Rámce nejsou přepínány na základě MAC adres, ale IP adres. Tyto IP adresy musí být společně s MAC adresami a čísly portů uvedeny v tabulce, kterou přepínač k přepínání využívá. [15], [23]

## 17.2 Hierarchický model

Switch slouží pro přepínání rámců v sítích LAN. U těchto sítí se běžně využívá hierarchický, konkrétně **trojvrstvý model**. Existence vrstev má své důvody. Každá vrstva definuje funkce pro zařízení, která do ní spadají. [10]




**Obrázek 53 – Hierarchický model**

Vrchní vrstvou je **páteřní vrstva** (core layer). Na této vrstvě dochází k propojení jednotlivých podsítí, je tedy nutné směrování, které je zajištěno prostřednictvím směrovačů, případně L3 switchů. Umožňuje také připojení sítě k Internetu. Ne této vrstvě je třeba vysokorychlostního přenosu. Z tohoto důvodu jsou zaváděny redundantní spoje, které zajišťují ochranu proti při

výpadku určité cesty. V případě propojení dvou switchů více rovnocennými linkami dochází k **agregaci** těchto linek (logicky „se tváří“ jako jedna linka), čímž je umožněna vyšší rychlost přenosu.

Prostřední vrstvou je **distribuční vrstva** (distribution layer), která propojuje vrchní páteřní vrstvu se spodní přístupovou vrstvou. Přístupová vrstva může prostřednictvím distribuční vrstvy předat data vrstvě páteřní, které se postará o doručení do cíle. Distribuční vrstva poskytuje bezpečnostní mechanismy. Konkrétně se jedná o **ACL** (Access Control List), na základě kterého je určováno, zda data od daného zařízení budou přijata nebo ne. Opět obsahuje redundantní spoje, čímž snižuje výskyt problému v případě výpadku některé linky. Pro zvýšení rychlosti přenosu dat mezi přístupovou a distribuční vrstvou je opět možné využít agregace linek utvořením více rovnocenných spojení mezi přepínači těchto dvou vrstev. Jednotlivé logicky související celky je možné rozdělit do virtuálních podsítí, které jsou nazývány jako **VLAN** (Virtual Local Area Network). Rozdělení může být zvoleno například dle jednotlivých oddělení ve firemní síti. Pro komunikaci mezi těmito virtuálními podsítěmi je třeba využití L3 switchů, případně routeru.

Nejspodnější vrstvou je **přístupová vrstva** (access layer). Na této vrstvě jsou připojena koncová zařízení, která mezi sebou mohou komunikovat prostřednictvím vyšších vrstev. Jsou zde implementovány bezpečnostní mechanismy, jako je povolení nebo zakázání připojení konkrétního zařízení k dané síti na základě MAC adresy. Tento mechanismus je nazýván jako **port-security**. Dále je možné nastavit, kolik zařízení může být maximálně k danému portu připojeno. Na přístupové vrstvě redundantní spoje utvářeny nejsou. Kromě koncových zařízení jako jsou stolní počítače, notebooky či tiskárny zde mohou být zapojena i mezilehlá zařízení, tedy switche, routery, hub apod. Přístupová vrstva také umožňuje rozdělení portů do logických skupin (VLAN). [10]

U port-security existuje několik možností konfigurace [3], [7], [10]. Nejprve je nutné port-security na daném portu zapnout:

```
S1(config)#interface <rozhraní>
S1(config-if)#switchport port-security
```

Vypnutí port-security:

```
S1(config-if)#no switchport port-security
```

Maximální počet povolených adres pro daný port:

```
S1(config-if)#switchport port-security maximum <číslo>
```

Povolení dynamického „sticky“ ukládání MAC adres (do MAC tabulky a běžící konfigurace). Statické adresy jsou ukládány do běžící konfigurace, dynamické do tabulky adres, sticky jsou uloženy na obou místech.

```
S1(config-if)#switchport port-security mac-address sticky
```

Zakázání dynamického „sticky“ ukládání MAC adres:

```
S1(config-if)#no switchport port-security mac-address sticky
```

Smazání dynamicky uložené adresy:

```
S1(config-if)#no switchport port-security sticky mac-address
<macadresa>
```

Staticky přidaná adresa:

```
S1(config-if)#switchport port-security mac-address <macadresa>
```

Smazání MAC adresy z tabulky:

```
S1(config-if)#no switchport port-security <macadresa>
```

Reakce portu na příchozí data z nepovolené MAC adresy:

```
S1(config-if)#switchport port-security violation <typ reakce>
```

Typy reakcí:

- Protect – data z MAC adresy, která není povolena, jsou zahozena, z ostatních MAC adres jsou zpracovány běžným způsobem.
- Restrict – uložena informace o porušení port-security.
- Shutdown – port je ve stavu error-disabled, není možné přes něj komunikovat. V případě, že má být port z tohoto stavu znovu zapnut, musí být nejprve vypnut. Tato reakce je nastavena jako výchozí.

Výpis bezpečných adres:

```
S1#show port-security address
```

Hierarchický model je využíván zejména díky řadě výhod, které poskytuje. Pár výhod bylo již zmíněno, těmi jsou bezpečnostní mechanismy na úrovni přístupové vrstvy, případně i vrstev vyšších, a neméně důležitá existence záložních cest pro případ výpadku některé linky. Hierarchický model je snadno rozšiřitelný, není tedy problém připojit jakékoliv nové zařízení, a také poskytuje vysoký výkon. Díky rozdělení do vrstev se daná síť poměrně snadno udržuje.

Typicky nejsou sítí přenášena pouze „klasická“ data, ale může se jednat i o audio či video. V případě, že jsou všechny tyto typy dat přenášeny přes jedno přenosové médium, jedná se o **konvergovanou síť**. V takovém případě je však třeba řídit důležitost přednosti zpracování dat, k čemuž slouží **QoS**, které řídí zpracovávání dat na základě jejich priorit. [10]

## 17.3 VLAN

Jak již bylo zmíněno, VLAN slouží k rozdělení LAN sítě do jednotlivých logicky souvisejících skupin, které na sobě nejsou závislé. Může se jednat o logické oddělení zařízení ve firmě na základě oddělení, do kterého spadají, nebo například o rozdělení zařízení určených pro studenty a učitele. Všechna zařízení v dané skupině musí patřit do stejné podsítě. Každá VLAN má své označení (**VLAN ID**), jednotlivým portům přepínače je poté přiřazena příslušnost k dané VLAN právě na základě tohoto VLAN ID. Tato označení jsou na základě jejich hodnot rozdělena do skupin. [2]

**Standardní VLAN ID** mají rozsah od 1 do 1005.

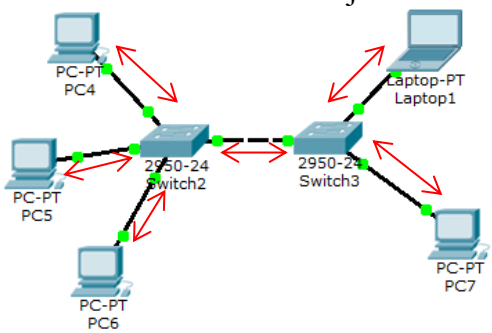
- VLAN ID 1002 až 1005 slouží pro FDDI VLAN a Token Ring. Obojí je mimo náplň této práce.
- VLAN ID 1 je označení pro **výchozí VLAN**, kam spadají všechna zařízení po připojení do sítě, což umožňuje jejich komunikaci. Je nastavena automaticky a nejde smazat, pouze lze změnit příslušnost portu k jiné VLAN.
- VLAN ID 1002, 1003, 1004 a 1005 se také utváří automaticky pro speciální účely a není možné je smazat.

**Rozšířená VLAN ID** jsou v rozsahu 1006 až 4094. Mají omezené funkce a jsou využívána poskytovateli internetových služeb.

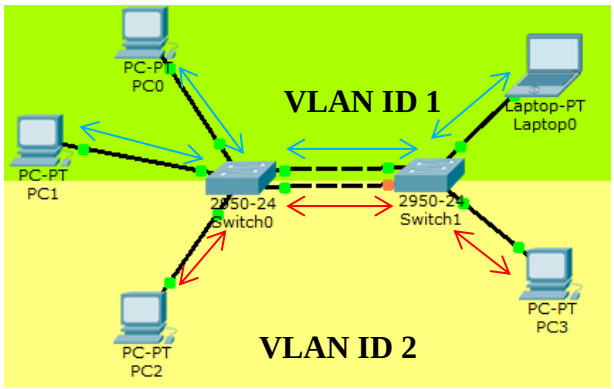
Kromě výchozí VLAN existuje také **nativní VLAN**, která slouží pro přenos rámců, které nepochází z žádné konkrétní VLAN, a dále **management VLAN** pro vzdálený přístup k zařízení. Management VLAN je třeba přiřadit IP adresu a masku podsítě. Typicky bývají management a nativní VLAN totožné. [10]

Proč se VLAN vytváří? Hlavním důvodem je snížení zátěže v síti, zvýšení výkonu a bezpečnost. V případě, že je v síti více propojených přepínačů, a jeden z nich zašle rámec na broadcastovou adresu, je tato zpráva postupně rozesílána všemi přepínači do celé sítě i v případě, že to není potřeba. Switche spojují tzv. **broadcastové domény**, tedy části sítě, kde jsou rozesílány broadcastové rámce. Pokud by takovýto rámec zaslalo více zařízení, síť by byla zahlcena. Broadcastová doména může být ohraničena. První možnost je směrovačem, který fyzicky odděluje jednotlivé lokální sítě, nebo pomocí rozdělení sítě do několika VLAN, které danou lokální síť rozdělí na logické podsítě. V takovém případě je broadcastový rámec rozeslán pouze v rámci dané VLAN. Rozdělení portů do jednotlivých VLAN také usnadňuje konfiguraci nově připojených zařízení, pouze stačí nastavit příslušnost k dané VLAN a tím je přiřazena úloha daného zařízení v rámci sítě. [2], [23]

**Jedna brodacastová doména.** Zařízení nejsou rozdělena do více VLAN.




**Dvě broadcastové domény.** Zařízení jsou rozdělena do dvou VLAN.

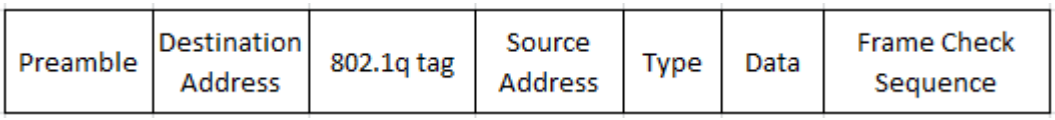



**Obrázek 54 – Broadcastové domény**

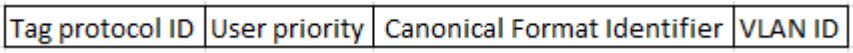
### 17.3.1 Komunikace mezi VLAN - trunky

V kapitole týkající se standardu IEEE 802.3 bylo uvedeno, že jeho hlavička může případně obsahovat 802.1q tag.

**IEEE 802.1q** je protokol, který umožňuje logické rozdělení sítě do VLAN a komunikaci v rámci těchto VLAN. Tato komunikace je umožněna díky označení, které je přidáno do hlavičky rámce během zapouzdření, konkrétně se jedná o 802.1q tag.




**Obrázek 55 – 802.3 Ethernet hlavička a 802.1q tagem**




**Obrázek 56 – IEEE 802.1q tag**

*Význam polí:*

**Tag Protocol ID** – 16bitové označení, že se jedná o 802.1q tag. Konkrétně obsahuje hodnotu 0x8100.

**User Priority** – 3bitové pole, které určuje prioritu rámce. Hodnoty 0 až 7 (nejvyšší priorita).

**Canonical Format Identifier** – 1bitové pole, které určuje formát MAC adresy.

**VLAN ID** – 12bitové pole, které určuje VLAN, do které daný rámec spadá.

Po přidání tohoto tagu je třeba znovu přepočítat kontrolní součet, který je uveden v patičce rámce. Rámce, které jsou označeny tagem, mohou být poté rozeslány pouze na zařízení spadající do VLAN s ID v něm uvedeném. K označení dochází na tzv. **trunk portu**. Díky zapouzdření je poté možné poslat rámce spadající do různých VLAN přes jednu a tu samou linku. Tato linka, jež je tvořena spojem mezi trunkovými porty, je nazývána jako **trunk**. Na druhém konci je poté rámec rozbalen a předán do uvedené VLAN.

Na trunk portu může být nastavena nativní VLAN, která slouží pro přenos rámců, které nejsou označeny žádným tagem, v takovém případě se jedná o rámce nepocházející z žádné VLAN.

Pozor nativní VLAN ID musí být na obou koncích trunkového spoje stejné!

Kromě trunk portů existují i porty s označením access, tento mód je pro všechny porty výchozí. Běžně jsou přes tyto porty připojeny koncové uzly. Slouží pro komunikaci pouze v rámci jedné VLAN, buď v té, kterou se nastavíme, nebo ve výchozí VLAN 1. [10], [15]

Každý port může být označen pouze v jednom módu trunk nebo access, ne obojí!

### 17.3.2 **Konfigurace VLAN**

Vytvoření VLAN s daným VLAN ID probíhá následovně:

```
S1(config)#vlan <VLAN ID>
```

Případně je možné si VLAN pojmenovat pro lepší orientaci v konfiguraci:

```
S1(config-vlan)#name <název>
```

Smazání VLAN:

```
S1(config)#no vlan <VLAN ID>
```

V případě management VLAN je třeba přiřadit IP adresu, masku podsítě a výchozí bránu pro vzdálený přístup. Výchozí brána slouží pro přenos dat do vzdálených sítí.

```
S1(config)#interface vlan <management VLAN ID>
S1(config-if)#ip address <IP adresa> <maska>
S1(config-if)#exit
S1(config)#ip default-gateway <adresa brány>
```

Přepnutí na konkrétní rozhraní a nastavení na mód **access**:

```
S1(config)#interface fastEthernet 0/1
S1(config-if)#switchport mode access
```

Přiřazení access portu do konkrétní VLAN:

```
S1(config-if)#switchport access vlan <VLAN ID>
```

Přepnutí na několik rozhraní z rozsahu, nastavení módu access a přiřazení do VLAN:

```
S1(config)#interface range fa0/1 - 6
S1(config-if)#switchport mode access
```

```
S1(config-if)#switchport access vlan <VLAN ID>
```

Odstranění portů z konkrétní VLAN:

```
S1(config-if)#no switchport access vlan <VLAN ID>
```

Označení portu jako **trunk**:

```
S1(config)#interface fastEthernet 0/2
S1(config-if)#switchport mode trunk
```

Přiřazení portu do konkrétní native VLAN:

```
S1(config-if)#switchport trunk native vlan <VLAN ID>
```

Odstranění všech VLAN z daného rozhraní:

```
S1(config)#interface fastEthernet 0/2
S1(config-if)#no switchport trunk allowed vlan
```

Pro výpis jednotlivých VLAN a k nim přiřazených portů slouží příkaz:

```
S1#show vlan brief
```

Podrobné informace o VLAN lze zjistit příkazy:

```
S1#show vlan id <VLAN ID>
S1#show vlan name <název VLAN>
S1#show interfaces vlan <VLAN ID>
```

Informace o rozhraní a jeho příslušnosti k VLAN:

```
S1#show interfaces <označení rozhraní> switchport
```

Odstranění nastavení všech VLAN a restart pro projevení změn:

```
S1#delete flash:vlan.dat
S1#reload
```

[3], [7], [15]

## 17.4 VTP (Virtual Trunking Protocol)

Jedná se o proprietární protokol vytvořený společností Cisco. Jeho úkolem je zjednodušit práci s VLAN. Princip spočívá v rozdělení switchů do jednotlivých rolí. Jedná se o módy server, klient a transparentní. Výchozím nastavením je mód server. Nastavení probíhá těmito způsoby [7]:

```
S1(config)#vtp mode server
S1(config)#vtp mode client
S1(config)#vtp mode transparent
```

**Server** má na starosti informování ostatních přepínačů o změnách ve VLAN formou zpráv. Jeho úkolem je vytvořit VLAN, smazat ji a případně měnit její název. Veškerou konfiguraci si uchovává v NVRAM paměti (nezávislá na napájení), po restartu přepínače tedy informace zůstávají zachovány. Informace o změnách jsou přeposílány pouze v rámci stejné **domény**, tedy konkrétních vybraných switchů. Mód server je výchozím nastavením u všech Cisco switchů, případná změna módu musí být nastavena ručně. Zejména u větších topologií se doporučuje mít nastaveny alespoň dva přepínače v módu server. V případě, že jeden „vypadne“, je díky existenci dalšího serveru stále možné provádět změny ve VLAN.

**Klient** také rozesílá informace o změnách, ke kterým v dané doméně došlo. Nemá však možnost mazat VLAN, vytvářet j či měnit název. Konfiguraci si uchovává v RAM (závislá na napájení), po restartu je konfigurace smazána.

**Transparent** slouží pouze k přeposílání přijatých informací o změnách ve VLAN. Sám od sebe žádné informace nerozesílá. Uchovává si svou konfiguraci v NVRAM a okolní změny ho „nezajímají“. [10]

Jeden switch spadá vždy do jedné domény. Příslušnost do dané domény je nastavena takto:

```
S1(config)#vtp domain <název domény>
```

Pozor! Je třeba, aby byl u všech přepínačů, které mají spadat do stejné domény, název stejný i včetně velikosti písmen! V případě, že má server nakonfigurovanou doménu, okamžitě je její jméno propagováno na ta zařízení, která do žádné domény přiřazena ještě nebyla.

Kromě domény se také nastavuje u jednotlivých switchů heslo, které musí být opět u všech zařízení stejné. V případě, že hesla neodpovídají, není zpráva přijata.

```
S1(config)#vtp password <heslo>
```

Dalším parametrem, který je třeba nastavit, je verze. U VTP existují verze 1, 2 a 3. U všech switchů v jedné doméně musí být číslo verze stejné.

```
S1(config)#vtp version <číslo verze>
```

Konfiguraci je možné zkontrolovat příkazem ***show vtp status***, který vypíše informace o verzi protokolu, revizi konfigurace, počtu VLAN apod.

```
S1#show vtp status
```

Případně pro výpis trunkových portů a příslušných VLAN:

```
S1#show interfaces trunk
```

[7], [15]

Veškeré zprávy o změnách (**vtp advertisements**) jsou zasílány na multicastovou MAC adresu **01-00-0C-CC-CC-CC** a jsou odesílány pouze trunk porty. Zpráva obsahuje název domény, verzi protokolu VTP a také o kolikátou revizi konfigurace se jedná. Dále jsou uvedeny informace o změnách v jednotlivých VLAN, formát rámce a další doplňující informace. Číslo revize konfigurace je číslováno od nuly a při každé změně ve VLAN je zvýšeno o jedničku. Toto číslo je uchováváno, aby každý switch poznal, zda má uchovány aktuální informace. Existují tři typy zpráv:

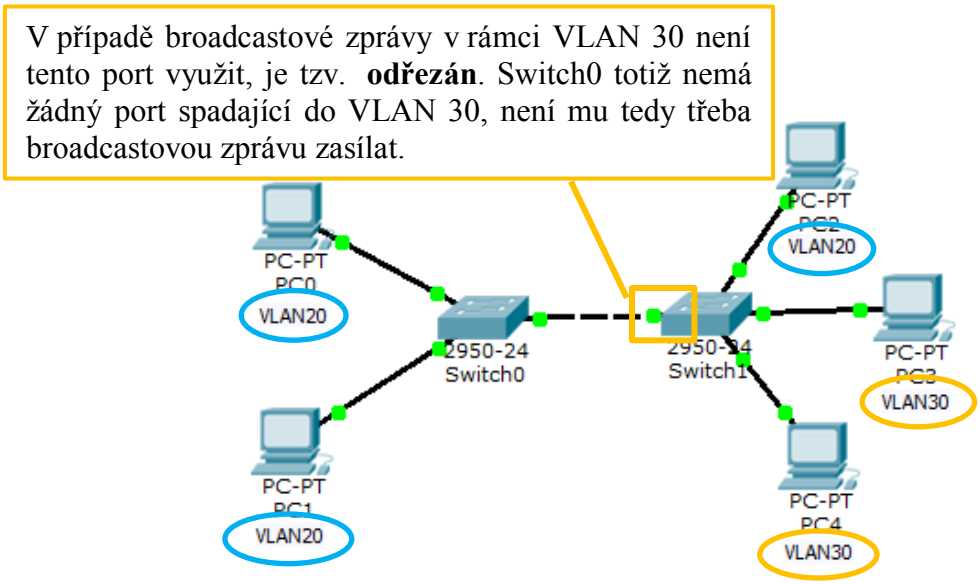
**Souhrnné zprávy** jsou zasílány pravidelně každých 5 minut nebo při výskytu změny, jejich úkolem je rozeslat informace o aktuálním číslu revize.

**Částečné aktualizace** mají v sobě zahrnuty informace o změnách ve VLAN.

**Žádosti** slouží k zaslání požadavku na server o zaslání souhrnné zprávy a následně i částečných aktualizací.

Na serveru existuje možnost konfigurace tzv. **VTP pruning** (odřezávání). Ostatní swtiche tuto konfiguraci obdrží během rozesílání změn ve VLAN. VTP pruning slouží k omezení rozesílání broadcastových rámců pouze na ty switche, které mají nějaké porty spadající do VLAN, ve které byl broadcastový rámec odeslán. [10], [15]

```
Switch1(config)#vtp pruning
```




**Obrázek 57 – VTP Pruning**

## 17.5 STP (Spanning Tree Protocol)

V části týkající se hierarchického modelu bylo uvedeno, že jsou často využívány **redundantní spoje** pro vytvoření záložních cest v případě výpadku linky. Díky těmto záložním cestám však mohou v topologii vzniknout **smyčky**, což je situace, kdy rámec neustále putuje sítí, aniž by se dostal do cíle (v hlavičce rámce není pole s životností jako u paketu). Díky tomu může dojít k zahlcení sítě. Z tohoto důvodu vznikl STP. Jeho úkolem je vypínat nepotřebné linky, v případě výpadku některé linky potřebnou náhradu naopak zapnout. [1], [23]

### 17.5.1 STA (Spanning Tree Algorithm)

Spanning Tree Algorithm je algoritmus, který určuje, jaká z linek bude nebo nebude vypnuta. Je tvořen několika kroky.

Prvním krokem je volba „**root bridge**“. Což je switch s nejnižším **Bridge ID (BID)**. Toto BID je určeno z priority (číslo od 1 do 65536) a MAC adresy switche.

Pro komunikaci switchů jsou u STP protokolu využívány **BPDU zprávy**, které jsou zasílány na multicastovou adresu **01-80-C2-00-00-00**. Formou těchto zpráv si mezi sebou přepínače vymění hodnoty svých BID a na základě toho je zvolen root bridge.

Nejprve si každý switch myslí, že je root bridge právě on a nastaví si BID jako root BID. V případě, že přijme BPDU od jiného přepínače, jehož hodnota BID je nižší, zjistí, že root bridge není a jeho BID již nemá označení root. Takto je postupně zvoleno nejnižší BID a společně s ním i root bridge.

Doporučuje se však ruční nastavení root bridge. Konkrétně by se mělo jedna o přepínač, který je přibližně uprostřed dané sítě.

Pro ruční nastavení root bridge lze zadat příkaz:

```
S1(config)#spanning-tree vlan <VLAN ID> root primary
```

Případně náhradní, pokud by původní switch „vypadl“:

```
S1(config)#spanning-tree vlan <VLAN ID> root secondary
```

Existuje i varianta ručního zadání priority přepínače, tato priorita je násobkem čísla 4096:

```
S1(config)#spanning-tree vlan <VLAN ID> priority <číslo>
```

[1], [7]

Dalším krokem je určení rolí jednotlivým portům přepínačů.

- Root
- Designated
- Non-designated

Nejprve jsou určeny **root porty**. Jedná se o porty, které nejsou součástí switche zvoleného jako root bridge. Jsou určovány na ostatních přepínačích. Jako root port je zvolen ten port, od něhož je cesta k root bridge nejkratší.

V případě, že na daném přepínači existují dva porty se stejnou délkou cesty k root bridge, je jako root port zvolen port s nižší hodnotou BID. Pokud by i BID bylo stejné, je jako root port zvolen port s nižší hodnotou označení (fa0/1 má přednost před fa0/2). Druhý se stává non-designated.

Pozor, na každém přepínači je zvolen vždy pouze jeden root port! [1], [15]

**Délku cesty** určuje suma ohodnocení portů, které je nutné navštívit na cestě k root bridge. Ohodnocení portů jsou dána na základě rychlosti daných linek [1]:

**Tabulka 15 – Tabulka ohodnocení portů**

| rychlost | ohodnocení |
|---|---|
| 10 Gbps | 2 |
| 2 Gbps | 3 |
| 1 Gbps | 4 |
| 100 Mbps | 19 |
| 10 Mbps | 100 |

Ohodnocení portu může být změněno příkazem:

```
S2(config-if)#spanning-tree cost 25
```

Obnova původní hodnoty:

```
S2(config-if)#no spanning-tree cost
```

Hodnoty portů a případně další informace lze zjistit příkazem:

```
S1#show spanning-tree
S1#show spanning-tree detail
```

[7]

V dalším kroku jsou určeny **designated porty**, což jsou všechny porty na root bridge a také některé porty ostatních zařízení. Designated porty jsou aktivní, přeposílají tedy zprávy. Na každém spoji mezi dvěma přepínači může být však pouze jeden designated port. Druhý port musí mít roli jinou, tedy root port nebo non-designated port. Jako non-designated je zvolen port s vyšší délkou cesty nebo případně s vyšší hodnotou BID.

**Non-designated (Alternate)** port je port, který byl zablokován z důvodu odstranění redundantního spoje. Pouze přeposílá BPDU zprávy.

Topologie s určenými rolemi portů je znázorněna v kapitole s názvem „Konfigurace přepínané sítě“.

Z důvodu existence redundantních spojů není možné, aby byly porty ihned po aktivaci schopny přenášet rámce, je třeba nejprve projít několika stavy, jinak by mohla vzniknout

smyčka. V průběhu změn stavů je rozhodnuto, jaký stav bude pro daný port při dané topologii konečný. Jedná se o tyto stavy:

- **Blocking** (blokující) – jedná se o port, který by způsobil smyčku, pokud by byl aktivní. Jedná se o již zmíněné non-designated porty, které jsou schopné přijímat BPDU, ale nepřeposílají ani nepřijímají žádné uživatelské rámce. Vše kromě BPDU zpráv je zahazováno. Port může být v blokujícím stavu maximálně 20 sekund, což je interval nazývaný „max age“.
- **Listening** (naslouchající) – přijímá a přeposílá BPDU zprávy, ale nic jiného zatím schopný přeposílat není. Ve stavu listening přepínač zůstává 15 sekund, tento interval je nazýván jako „forward delay“.
- **Learning** (učící se) – jedná se o port, který přijímá a přeposílá BPDU zprávy a navíc je schopen přijímat uživatelské rámce, ze kterých se učí MAC adresy a vytváří si záznamy ve své tabulce MAC adres, rámce dále však neodesílá. Opět v tomto stavu zůstává po dobu intervalu forward delay, který bývá nastaven na 15 sekund.
- **Forwarding** (přeposílající) – jedná se o normálně fungující port, který nejen přijímá a přeposílá BPDU, ale je také schopen přijímat a odesílat uživatelské rámce.
- **Disabled** (vypnutý) – administrátorem ručně vypnutý port, případně port, u kterého se vyskytla porucha.

Kromě předchozích uvedených časovačů existuje i „hello time“, který udává, jak často jsou zasílány BPDU, typicky každé 2 sekundy.

Po rozhodnutí o stavech portů zůstávají porty ve stavech forwarding, blocking, případně disabled. Jakmile dojde ke změně v topologii, přechází zpět do stavu listening a je opět rozhodnuto o konečném stavu.

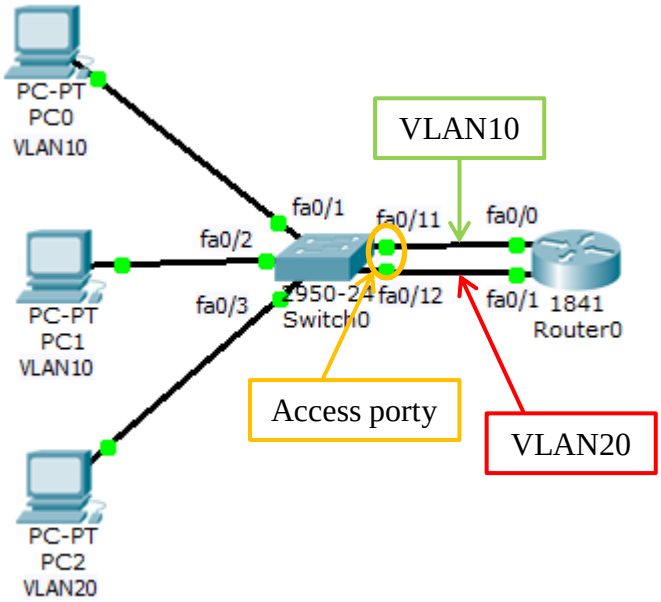
Proces volby root bridge a rolí portů, včetně jejich koncových stavů, je nazýván jako **konvergence**. [1], [15]

## 17.6 Inter-VLAN Routing

Slouží pro směrování mezi VLAN. Existují tři varianty.

### 17.6.1 **Klasický směrovač**

První variantou je použití klasického routeru, který má pro každou VLAN utvořeno jedno fyzické spojení přes rozhraní do portů switche. Porty na přepínači jsou v módu access (každý spadá do jedné VLAN). Výhodou je oproti další metodě rychlost, jelikož pro každou VLAN existuje oddělená linka.




**Obrázek 58 – Klasický směrovač**

Přiřazení adres rozhraním routeru:

```
Router0(config)#interface fa0/0
Router0(config-if)#ip address 192.168.10.1 255.255.255.0
Router0(config-if)#no shutdown
Router0(config)#interface fa0/1
Router0(config-if)#ip address 192.168.20.1 255.255.255.0
Router0(config-if)#no shutdown
```

Vytvoření VLAN:

```
Switch0(config)#vlan 10
Switch0(config-vlan)#vlan 20
Switch0(config-vlan)#exit
```

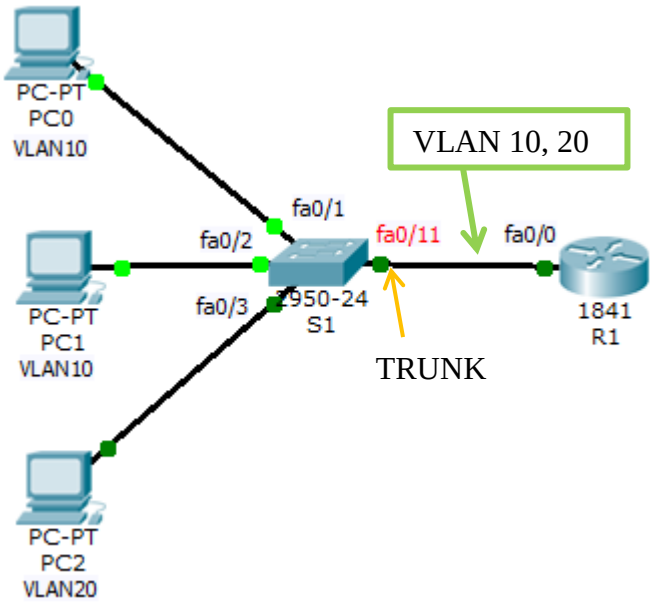
Přiřazení portů přepínače do jednotlivých VLAN:

```
Switch0(config)#interface fa0/1
Switch0(config-if)#switchport mode access
Switch0(config-if)#switchport access vlan 10
Switch0(config-if)#interface fa0/11
Switch0(config-if)#switchport mode access
Switch0(config-if)#switchport access vlan 10
Switch0(config-if)#interface fa0/2
Switch0(config-if)#switchport mode access
Switch0(config-if)#switchport access vlan 10
Switch0(config-if)#interface fa0/12
Switch0(config-if)#switchport mode access
Switch0(config-if)#switchport access vlan 20
Switch0(config-if)#interface fa0/3
Switch0(config-if)#switchport mode access
Switch0(config-if)#switchport access vlan 20
```

Na jednotlivých PC je třeba zadat IP adresu, která spadá do rozsahu adres dané VLAN (zařízení ve stejné VLAN musí být ve stejné podsíti) a jako výchozí bránu nastavit rozhraní routeru v této VLAN. Konkrétně pro VLAN 10 je použita podsíť s adresou 192.168.10.0/24. Pro PC0 bude tedy IP adresa například 192.168.10.2, maska 255.255.255.0 a výchozí brána je 192.168.10.1 (fa0/0 na routeru).

### 17.6.2 **Router-on-a-stick**

Druhou variantou je router-on-a-stick. Zde je utvořeno pouze jedno spojení z jednoho portu na směrovači do jednoho portu na přepínači. Díky tomu je tato varianta méně nákladná. Rozhraní routeru je rozděleno do softwarově vytvořených rozhraní tzv. **subinterfaces**. Každé toto rozhraní spadá do jiné VLAN a má přiřazenou vlastní IP adresu. Hardwarově se stále jedná pouze o jeden port. Jelikož port na přepínači spadá do více VLAN, je v módu trunk. Toto zapojení je však oproti předchozímu pomalejší, jelikož je jednou linkou přenášeno více rámců mezi různými VLAN. Konfigurace je také o něco složitější.




**Obrázek 59 – Router-on-a-stick**

Vytvoření subinterface:

```
R1(config)#interface fa0/0.10
```

Tento příkaz nakonfiguruje linku jako trunk (zapouzdření IEEE 802.1q) a přiřadí jí VLAN 10.

```
R1(config-subif)#encapsulation dot1q 10
R1(config-subif)#ip address 192.168.10.1 255.255.255.0
R1(config)#interface fa0/0.20
```

Přiřazení také VLAN 20.

```
R1(config-subif)#encapsulation dot1q 20
R1(config-subif)#ip address 192.168.20.1 255.255.255.0
R1(config-subif)#interface f0/0
R1(config-if)#no shutdown
```

Vytvoření VLAN a přiřazení access portů do těchto VLAN:

```
S1(config)#vlan 10
S1(config-vlan)#vlan 20
S1(config-vlan)#exit
S1(config)#interface fa0/1
S1(config-if)#switchport mode access
S1(config-if)#switchport access vlan 10
S1(config-if)#switchport mode access
S1(config-if)#interface fa0/2
S1(config-if)#switchport mode access
```

```
S1(config-if)#switchport access vlan 10
S1(config-if)#interface fa0/3
S1(config-if)#switchport mode access
S1(config-if)#switchport access vlan 20
```

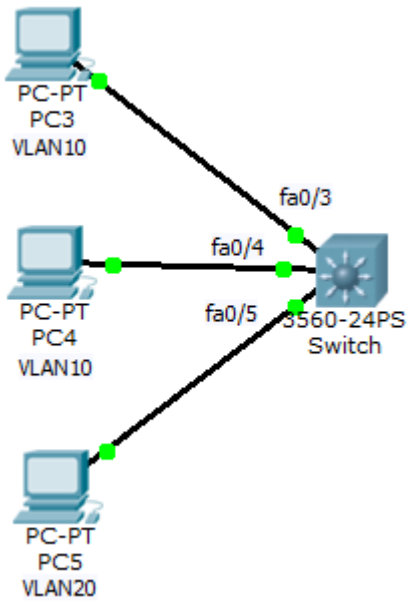
Port na switchi, přes který je připojen router, je nastaven jako trunk (pro přenos rámců pocházejících z různých VLAN).

```
S1(config)#interface fa0/11
S1(config-if)#switchport mode trunk
```

Konfigurace počítačů se řídí stejnými pravidly jako v předchozím případě. Jako výchozí brány jsou však nastaveny adresy nastavené na subinterface v dané VLAN.

### 17.6.3 L3 switch

Třetí variantou je použití L3 switche. Kde je směrování realizováno přes trunk s L3 switchem. Jednotlivé porty v různých VLAN.




**Obrázek 60 – L3 switch**

```
Switch(config)#vlan 10
Switch(config-vlan)#vlan 20
Switch(config-vlan)#int vlan 10
Switch(config-if)#ip add 192.168.10.1 255.255.255.0
Switch(config-if)#int vlan 20
Switch(config-if)#ip add 192.168.20.1 255.255.255.0
Switch(config-if)#int range fa0/3-4
Switch(config-if-range)#switchport mode access
```

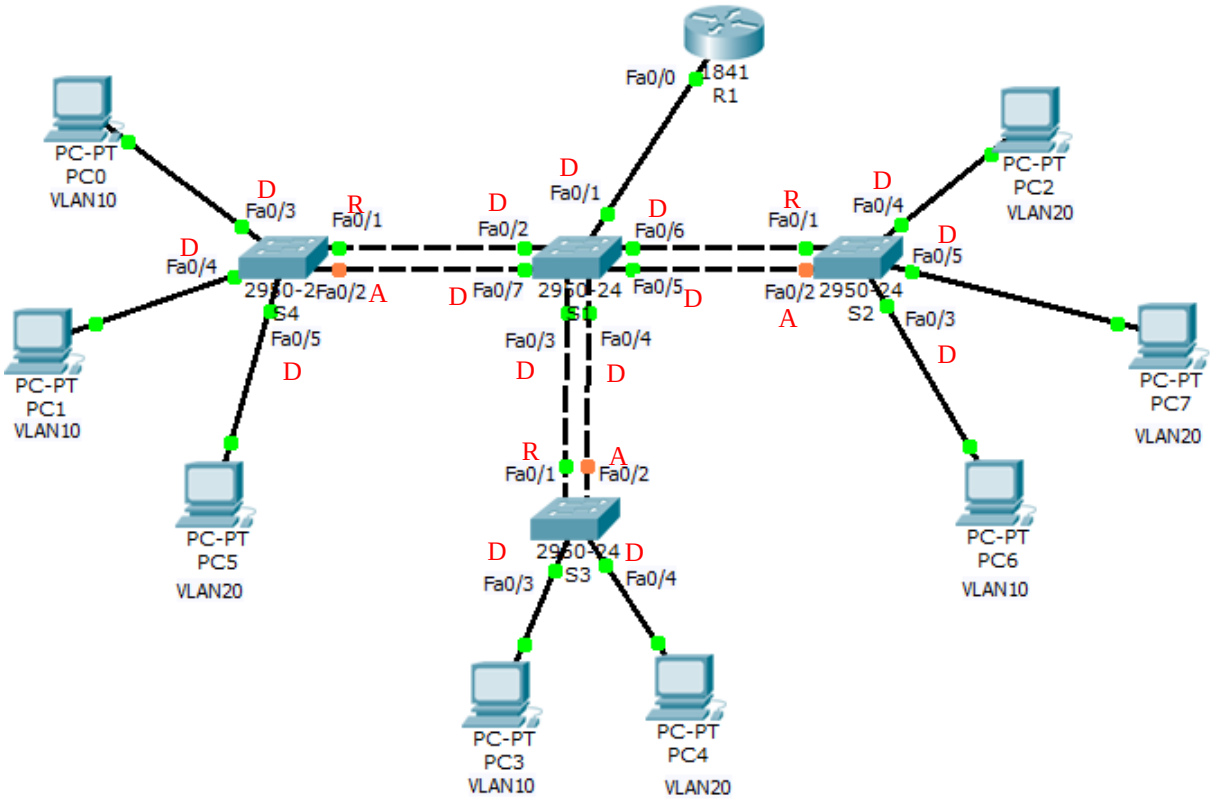
```
Switch(config-if-range)#switchport access vlan 10
Switch(config-if-range)#int fa0/5
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 20
```

Povolení směrování:

```
Switch(config)#ip routing
```

Počítače jsou nastaveny jako v předchozích případech, výchozí brány jsou adresy nastavené u jednotlivých VLAN, tedy například pro PC3 je výchozí brána 192.168.10.1. [7],[10], [15]

## 17.7 Konfigurace přepínané sítě




**Obrázek 61 – Topologie přepínání**

**Tabulka 16 – Adresace topologie**

| VLAN | Zařízení | Rozhraní | IP adresa | Maska podsítě | Výchozí brána |
|---|---|---|---|---|---|
| 10 | PC0 | NIC | 192.168.10.10 | 255.255.255.0 | 192.168.10.1 |
| 10 | PC1 | NIC | 192.168.10.11 | 255.255.255.0 | 192.168.10.1 |
| 20 | PC2 | NIC | 192.168.20.10 | 255.255.255.0 | 192.168.20.1 |
| 10 | PC3 | NIC | 192.168.10.12 | 255.255.255.0 | 192.168.10.1 |
| 20 | PC4 | NIC | 192.168.20.11 | 255.255.255.0 | 192.168.20.1 |
| 20 | PC5 | NIC | 192.168.20.12 | 255.255.255.0 | 192.168.20.1 |
| 10 | PC6 | NIC | 192.168.10.13 | 255.255.255.0 | 192.168.10.1 |
| 20 | PC7 | NIC | 192.168.20.13 | 255.255.255.0 | 192.168.20.1 |
| - | R1 | Fa0/0.10 | 192.168.10.1 | 255.255.255.0 | n/a |
| | | Fa0/0.20 | 192.168.20.1 | 255.255.255.0 | n/a |
| | | Fa0/0.99 | 192.168.99.1 | 255.255.255.0 | n/a |
| - | S1 | VLAN99 | 192.168.99.10 | 255.255.255.0 | 192.168.99.1 |
| - | S2 | VLAN99 | 192.168.99.11 | 255.255.255.0 | 192.168.99.1 |
| - | S3 | VLAN99 | 192.168.99.12 | 255.255.255.0 | 192.168.99.1 |
| - | S4 | VLAN99 | 192.168.99.13 | 255.255.255.0 | 192.168.99.1 |

**Tabulka 17 – VLAN**

| VLAN | Adresa sítě | Název VLAN |
|---|---|---|
| 10 | 192.168.10.0/24 | podrizeni |
| 20 | 192.168.20.0/24 | vedeni |
| 99 | 192.168.99.0/24 | management |

**Tabulka 18 – Trunk porty**

| Přepínač | Trunk port | VLAN |
|---|---|---|
| S2 | Fa0/1-2 | 99 |
| S3 | Fa0/1-2 | 99 |
| S1 | Fa0/1-6 | 99 |
| S4 | Fa0/1-2 | 99 |

**Tabulka 19 – Access porty**

| Přepínač | Access port | VLAN |
|---|---|---|
| S2 | fa0/3 | 10 |
| | fa0/4-5 | 20 |
| S3 | fa0/3 | 10 |
| | fa0/4 | 20 |
| S4 | fa0/3-4 | 10 |
| | fa0/5 | 20 |

V prvním kroku je v konfiguračním režimu switche změněn jeho název. Poté je vytvořena management VLAN s ID 99 pro vzdálený přístup, která bude zároveň nativní. Pro tuto VLAN musí být nastavena IP adresa s maskou podsítě a také výchozí brána. Příkazem no shutdown je zapnuto rozhraní VLAN 99. Tato konfigurace je u všech přepínačů stejná, liší se pouze uvedenými IP adresami. Jako výchozí brána je nastavena adresa rozhraní směrovače, brána je využívána pro směrování do jiných VLAN.

**S1:**

```
Switch>enable
Switch#configure terminal
Switch(config)#hostname S1
S1(config)#vlan 99
S1(config-vlan)#name management
S1(config-vlan)#interface vlan 99
S1(config-if)#ip address 192.168.99.10 255.255.255.0
S1(config-if)#no shutdown
S1(config-if)#exit
S1(config)#ip default-gateway 192.168.99.1
```

**S2:**

```
Switch>enable
Switch#configure terminal
Switch(config)#hostname S2
S2(config)#vlan 99
S2(config-vlan)#name management
S2(config-vlan)#interface vlan 99
S2(config-if)#ip address 192.168.99.11 255.255.255.0
S2(config-if)#no shutdown
S2(config-if)#exit
S2(config)#ip default-gateway 192.168.99.1
```

**S3:**

```
Switch>enable
Switch#configure terminal
Switch(config)#hostname S3
S3(config)#vlan 99
S3(config-vlan)#name management
S3(config-vlan)#interface vlan 99
S3(config-if)#ip address 192.168.99.12 255.255.255.0
S3(config-if)#no shutdown
S3(config-if)#exit
S3(config)#ip default-gateway 192.168.99.1
```

**S4:**

```
Switch>enable
Switch#configure terminal
Switch(config)#hostname S4
S4(config)#vlan 99
S4(config-vlan)#name management
S4(config-vlan)#interface vlan 99
S4(config-if)#ip address 192.168.99.13 255.255.255.0
S4(config-if)#no shutdown
S4(config-if)#exit
S4(config)#ip default-gateway 192.168.99.1
```

V následujícím kroku je příkazem ***show vlan brief*** ověřeno vytvoření VLAN s ID 99. Je uveden výpis pouze pro S4, jelikož je u ostatních přepínačů totožný.

```
S4#show vlan brief

VLAN Name                              Status    Ports
---- --------------------------------- --------- -------------------------------
----
1    default                           active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                 Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                 Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                                 Fa0/13, Fa0/14, Fa0/15, Fa0/16
                                                 Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                 Fa0/21, Fa0/22, Fa0/23, Fa0/24
99   management                        active
1002 fddi-default                      active
1003 token-ring-default                active
1004 fddinet-default                   active
1005 trnet-default                     active
```

Dále je třeba nastavit trunk porty:

```
S1(config)#interface range fa0/1-7
S1(config-if-range)#switchport mode trunk
S1(config-if-range)#switchport trunk native vlan 99
S1(config-if-range)#no shutdown
S1(config-if-range)#exit

S2(config)#interface range fa0/1-2
S2(config-if-range)#switchport mode trunk
S2(config-if-range)#switchport trunk native vlan 99
S2(config-if-range)#no shutdown
S2(config-if-range)#exit

S3(config)#interface range fa0/1-2
S3(config-if-range)#switchport mode trunk
S3(config-if-range)#switchport trunk native vlan 99
S3(config-if-range)#no shutdown
S3(config-if-range)#exit

S4(config)#interface range fa0/1-2
S4(config-if-range)#switchport mode trunk
S4(config-if-range)#switchport trunk native vlan 99
S4(config-if-range)#no shutdown
S4(config-if-range)#exit
```

Pro rychlejší konfiguraci je vytvořena doména VTP s názvem „domena“ a heslem „heslo“, kde S1 pracuje jako server, S2, S3 a S4 jsou nastaveni jako klienti. Poté stačí vytvořit dané

VLAN u serveru, na klienty se tato konfigurace přepošle. Pozor při zadávání názvu domény a hesla, u všech přepínačů si musí odpovídat! Jinak by komunikace selhala.

```
S1#conf terminal
S1(config)#vtp mode server
S1(config)#vtp domain domena
S1(config)#vtp password heslo

S2(config)#vtp mode client
S2(config)#vtp domain domena
S2(config)#vtp password heslo

S3(config)#vtp mode client
S3(config)#vtp domain domena
S3(config)#vtp password heslo

S4(config)#vtp mode client
S4(config)#vtp domain domena
S4(config)#vtp password heslo
```

Kontrola konfigurace VTP, opět uveden pouze výpis pro S4, kromě módu na S1 jsou všechny výpisy totožné.

```
S4#show vtp status
VTP Version                     : 2
Configuration Revision          : 4
Maximum VLANs supported locally : 255
Number of existing VLANs        : 8
VTP Operating Mode              : Client
VTP Domain Name                 : domena
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0xA6 0xE9 0xF6 0xCD 0xDD 0x70 0xFD 0x5F
Configuration last modified by 0.0.0.0 at 3-2-93 23:02:23
```

Vytvoření VLAN 10 s názvem „podrizeni“ a VLAN 20 s názvem „vedeni“:

```
S1(config)#vlan 10
S1(config-vlan)#name podrizeni
S1(config-vlan)#vlan 20
S1(config-vlan)#name vedeni
```

Kontrola vytvoření VLAN a zároveň funkčnosti VTP. Opět na S4:

```
S4#show vlan brief

VLAN Name                              Status    Ports
---- --------------------------------- --------- -------------------------------
---
1    default                           active    Fa0/3, Fa0/4, Fa0/5, Fa0/6
                                                 Fa0/7, Fa0/8, Fa0/9, Fa0/10
                                                 Fa0/11, Fa0/12, Fa0/13, Fa0/14
                                                 Fa0/15, Fa0/16, Fa0/17, Fa0/18
                                                 Fa0/19, Fa0/20, Fa0/21, Fa0/22
                                                 Fa0/23, Fa0/24
10   podrizeni                         active
20   vedeni                            active
99   management                        active
1002 fddi-default                      active
1003 token-ring-default                active
1004 fddinet-default                   active
1005 trnet-default                     active
```

Dále je třeba nastavit access porty na jednotlivých přepínačích:

```
S2(config)#interface fa0/3
S2(config-if)#switchport mode access
S2(config-if)#switchport access vlan 10
S2(config-if)#no shutdown
S2(config-if)#exit

S2(config)#interface range fa0/4-5
S2(config-if-range)#switchport mode access
S2(config-if-range)#switchport access vlan 20
S2(config-if-range)#no shutdown

S3(config)#interface fa0/3
S3(config-if)#switchport mode access
S3(config-if)#switchport access vlan 10
S3(config-if)#no shutdown
S3(config-if)#exit
S3(config)#interface fa0/4
S3(config-if)#switchport mode access
S3(config-if)#switchport access vlan 20
S3(config-if)#no shutdown
S3(config-if)#exit

S4(config)#interface range fa0/3-4
S4(config-if-range)#switchport mode access
S4(config-if-range)#switchport access vlan 10
S4(config-if-range)#no shutdown
S4(config-if-range)#exit
S4(config)#interface fa0/5
```

```
S4(config-if)#switchport mode access
S4(config-if)#switchport access vlan 20
S4(config-if)#no shutdown
```

Pro komunikaci mezi jednotlivými VLAN je třeba nakonfigurovat inter-vlan routing, konkrétně s použitím router-on-a-stick.

```
Router(config)#hostname R1
R1(config)#interface fa0/0
R1(config-if)#no shutdown
R1(config-if)#exit
```

Vytvoření subinterface:

```
R1(config)#interface fa0/0.10
R1(config-subif)#encapsulation dot1Q 10
R1(config-subif)#ip address 192.168.10.1 255.255.255.0
R1(config-subif)#exit
R1(config)#interface fa0/0.20
R1(config-subif)#encapsulation dot1Q 20
R1(config-subif)#ip address 192.168.20.1 255.255.255.0
R1(config-subif)#exit
R1(config)#interface fa0/0.99
R1(config-subif)#encapsulation dot1Q 99
R1(config-subif)#ip address 192.168.99.1 255.255.255.0
R1(config-subif)#exit
```

Z důvodu redundantních spojů mezi přepínači je nutná existence STP protokolu. V tomto kroku je ručně nastaven root bridge pro VLAN 10, 20 i nativní 99, konkrétně se jedná o prostřední přepínač S1, jako sekundární root bridge je nastaven přepínač S3.

```
S1(config)#spanning-tree vlan 10,20,99 root primary
S3(config)#spanning-tree vlan 10,20,99 root secondary
```

Ověření nastavení, ve výpisu je možné si všimnout hodnot dříve zmíněných časovačů a uvedených rolí jednotlivých portů:

```
S1#show spanning-tree
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    32769
              Address     0001.9786.1B14
              This bridge is the root
              Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32769  (priority 32768 sys-id-ext 1)
              Address     0001.9786.1B14
              Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  20

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
--
Fa0/3            Desg FWD 19        128.3    P2p
Fa0/4            Desg FWD 19        128.4    P2p
Fa0/5            Desg FWD 19        128.5    P2p
Fa0/6            Desg FWD 19        128.6    P2p
Fa0/7            Desg FWD 19        128.7    P2p
Fa0/1            Desg FWD 19        128.1    P2p
Fa0/2            Desg FWD 19        128.2    P2p

VLAN0010
  …
VLAN0020
  …
VLAN0099
  …
```

Role portů, pokud je root bridge S1, jsou znázorněny v návrhu topologie.

- A = non-designated (alternate) port
- D = designated port
- R = root port

Pro procvičení je dalším krokem je nastavení port-security. V tomto případě bude nastavena port-security na portu fa0/3 u přepínače S3.

Maximum povolených adres je 3. Dále je povoleno dynamické ukládání MAC adres a v případě porušení port-security přechází port do stavu error-disabled.

```
S3(config)#interface fa0/3
S3(config-if)#switchport port-security
S3(config-if)#switchport port-security maximum 3
S3(config-if)#switchport port-security mac-address sticky
S3(config-if)#switchport port-security violation shutdown
```

Posledním krokem je ověření funkčnosti této topologie, konkrétně inter-vlan routingu. Toto ověření bude realizováno pokusem o ***ping*** mezi PC0 s adresou 192.168.10.10 z VLAN 10 a PC8 s adresou 192.168.20.13 z VLAN 20.

```
PC>ping 192.168.20.13

Pinging 192.168.20.13 with 32 bytes of data:

Reply from 192.168.20.13: bytes=32 time=2ms TTL=127
Reply from 192.168.20.13: bytes=32 time=0ms TTL=127
Reply from 192.168.20.13: bytes=32 time=0ms TTL=127
Reply from 192.168.20.13: bytes=32 time=0ms TTL=127

Ping statistics for 192.168.20.13:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms
```

Ping byl úspěšný, inter-vlan routing funguje. Z důvodu příliš dlouhých výpisů však nebude zobrazen výpis ostatních pokusů mezi jinými zařízeními.

[7], [10], [15]

# Závěr

Tato práce vznikla jako podklad pro wiki projekt počítačových sítí, který měl za úkol představit základní principy datových sítí se zaměřením na architekturu TCP/IP a dále vysvětlit problematiku přepínaných sítí.

V první části bakalářské práce byl vymezen pojem počítačová síť, včetně uvedení jednotlivých komponent, společně s popisem jejich funkčnosti. Dále byly uvedeny typy sítí, které jsou děleny na základě geografické oblasti. Zejména byl představen rozdíl mezi LAN a WAN sítěmi. Tato část také obsahuje popis jednotlivých fyzických topologií, který mimo jiné uvádí výhody, případně nevýhody, jednotlivých topologií a také jejich použití.

Nejrozsáhlejší částí této práce je část druhá, ve které byl zaveden pojem vrstvových modelů. Tato část byla věnována zejména protokolům jednotlivých vrstev architektury TCP/IP, na které byl také přiblížen mechanismus průchodu dat sítí, včetně forem dat (datových jednotek) na jednotlivých vrstvách této architektury, což úzce souvisí s pojmem zapouzdřování.

Kromě protokolů aplikační a transportní vrstvy byly představeny zejména protokoly vrstvy síťové, konkrétně IPv4 a IPv6. IPv6 slouží jako náhrada za IPv4, bylo tedy třeba vystvětlit důvod této náhrady a základní rozdíly mezi těmito protokoly a také jejich principy. Na praktických příkladech byly vysvětleny převody mezi jednotlivými soustavami, ve kterých mohou být reprezentovány IP adresy. Jedná se tedy o převody mezi desítkovou, binární a šestnáctkovou soustavou. Po kapitole týkající se převodů soustav byly uvedeny principy tvorby podsítí u IPv4 i IPv6, včetně konkrétních příkladů. Bylo představeno třídní adresování, CIDR , VLSM a postup přidělování adresních rozsahů internetovým poskytovatelům a jejich zákazníkům. Kromě výše uvedených vrstev byla také představena vrstva síťového rozhraní, která byla zaměřena zejména na popis přístupových metod k fyzickému médiu a protokolů této vrstvy. Dále byl vymezen pojem rámec. Důležitými protokoly jsou jednotlivé formy Ethernetu, konkrétně Ethernet II a IEEE 802.3 a IEEE 802.2. Byly popsány struktury jejich rámců a rozdíly mezi nimi. U Ethernetu II a IEEE 802.3 byly zobrazeny rámce odchycené analyzátorem Wireshark, včetně vysvětlení konkrétních hodnot. Dále byl vymezen protokol ARP, včetně názorného zobrazení jeho funkčnosti, a pojem MAC adresa, u které byla popsána její struktura.

Třetí část této práce byla věnována problematice přepínaných sítí. Byl představen přepínač a jeho mechanismus „učení“. Dále byla popsána struktura třívrstvého hierarchického modelu, byly vysvětleny důležité pojmy a příkazy spadající do přepínaných sítí. Zejména se jednalo o VLAN, trunk, inter-VLAN routing, také byly popsány protokoly, které jsou důležité pro funkčnost a usnadnění práce s přepínanou sítí, konkrétně STP a VTP. Na závěr práce byla uvedena konfigurace vlastní topologie přepínané sítě, včetně ověření její funkčnosti. Celá konfigurace byla realizována v programu PacketTracer.

Výstupy z programu Wireshark a nakonfigurovaná topologie přepínané sítě v programu PacketTracer jsou uloženy na přiloženém CD.

## Literatura


[1] BOUŠKA, Petr. Cisco IOS 9 - Spanning Tree Protocol. *www.samuraj-cz.com* [online]. 2007, 03. 05. 2009 [cit. 2014-04-28]. Dostupné z:http://www.samuraj-cz.com/clanek/cisco-ios-9-spanning-tree-protocol/

[2] BOUŠKA, Petr. VLAN - Virtual Local Area Network. *www.samuraj-cz.com* [online]. 2007 [cit. 2014-04-23]. Dostupné z: http://www.samuraj-cz.com/clanek/vlan-virtual-local-area-network/

[3] BOUŠKA, Petr. Cisco IOS 3 - nastavení interface/portu - access, trunk, port security. *www.samuraj-cz.com* [online]. 2007, 18. 5. 2009 [cit. 2014-04-23]. Dostupné z:http://www.samuraj-cz.com/clanek/cisco-ios-3-nastaveni-interfaceportu-access-trunk-port-security/

[4] BOUŠKA, Petr. TCP/IP - adresy, masky, subnety a výpočty. *www.samuraj-cz.com* [online]. 2007, 11. 08. 2008 [cit. 2014-04-28]. Dostupné z:http://www.samuraj-cz.com/clanek/tcpip-adresy-masky-subnety-a-vypocty/

[5] CONTA, A., S. DEERING a M. GUPTA. Internet Control Message Protocol (ICMPv6). *Request for Comments* [online]. March 2006 [cit. 2014-04-23]. ISSN 2070-1721. Dostupné z: http://tools.ietf.org/html/rfc4443

[6] DEERING, S. a R. HINDEN. Internet Protocol, Version 6 (IPv6): Specification. *Request for Comments* [online]. December 1998 [cit. 2014-04-23]. ISSN 2070-1721. Dostupné z: https://tools.ietf.org/html/rfc2460

[7] EMPSON, Scott. *CCNA kompletní přehled příkazů: autorizovaný výukový průvodce*. Vyd. 1. Brno: Computer Press, 2009, 336 s. ISBN 978-80-251-2286-0.

[8] HORÁLEK, Josef. Krátký popis LAN a WAN sítí. [online]. [cit. 2014-04-28]. Dostupné z: http://www.horalek.org/clanky/lanwan.pdf

[9] KABELOVÁ, Alena a Libor DOSTÁLEK. *Velký průvodce protokoly TCP/IP a systémem DNS*. 5., aktualiz. vyd. Brno: Computer Press, 2008, 488 s. ISBN 978-80-251-2236-5.

[10] LAMMLE, Todd. *CCNA: výukový průvodce přípravou na zkoušku 640-802*. Vyd. 1. Brno: Computer Press, 2010, 928 s. ISBN 978-802-5123-591.

[11] M., Thomas. One Byte at a Time: Is Your FTP Active or Passive?. *Cisco* [online]. 1999 [cit. 2014-04-28]. Dostupné z:http://www.cisco.com/web/about/ac123/ac147/ac174/ac199/about_cisco_ipj_archive_article09186a00800c85a7.html

[12] MCGEHEE, Brian. IPv6 Addressing. *IPv6 Summit, Inc.* [online]. 2003 [cit. 2014-04-23]. Dostupné z: http://www.usipv6.com/ppt/IPv6Addressing-BrianMcGehee.pdf

[13] O DOMÉNÁCH A DNS. c*z.nic: správce domény cz* [online]. 2014 [cit. 2014-04-28]. Dostupné z: https://www.nic.cz/page/312/o-domenach-a-dns/

[14] ODOM, Wendell. *Počítačové sítě bez předchozích znalostí*. Vyd. 1. Brno: CP Books, 2005, 383 s. ISBN 80-251-0538-5.

[15] ODOM, Wendell, Rus HEALY a Naren MEHTA. *Směrování a přepínání sítí: autorizovaný výukový průvodce*. Vyd. 1. Brno: Computer Press, 2009, 879 s. ISBN 978-80-251-2520-5.

[16] ODVÁRKA, Petr. Ethernet. *Svět sítí* [online]. 2000 [cit. 2014-04-28]. Dostupné z: http://www.svetsiti.cz/clanek.asp?cid=Ethernet-1992000

[17] ODVÁRKA, Petr. TCP handshake krok za krokem. *Svět sítí* [online]. 2000 [cit. 2014-04-28]. Dostupné z: http://www.svetsiti.cz/clanek.asp?cid=TCP-handshake-krok-za-krokem-3122000#tab-2-big

[18] PETERKA, Jiří. Referenční model ISO/OSI - sedm vrstev. *eArchiv.cz* [online]. 1992 [cit. 2014-04-28]. Dostupné z:http://www.earchiv.cz/a92/a213c110.php3

[19] PETERKA, Jiří. Ethernet II vs. IEEE 802.3. e*Archiv.cz* [online]. 1999 [cit. 2014-04-23]. Dostupné z: http://www.earchiv.cz/anovinky/ai2058.php3

[20] PETERKA, Jiří. Formáty Ethernetových rámců. e*Archiv.cz* [online]. 1997 [cit. 2014-04-23]. Dostupné z: http://www.earchiv.cz/a97/a729k150.php3

[21] PETERKA, Jiří. Subnetting, supernetting a CIDR. *eArchiv.cz* [online]. 1999 [cit. 2014-04-23]. Dostupné z: http://www.earchiv.cz/anovinky/ai1681.php3

[22] POSTEL, J. INTERNET CONTROL MESSAGE PROTOCOL. *Request for Comments* [online]. September 1981 [cit. 2014-04-23]. ISSN 2070-1721. Dostupné z: http://tools.ietf.org/html/rfc792

[23] PUŽMANOVÁ, Rita. *TCP/IP v kostce*. 2. upr. a rozš. vyd. České Budějovice: Kopp, 2009, 619 s. Mistrovství (Computer Press). ISBN 978-80-7232-388-3.

[24] RUSEK, Ondřej. Transportní vrstva. [online]. [cit. 2014-04-28]. Dostupné z: http://www.gybon.cz/~rusek/vyuka/site/site007.html

[25] SANDERS, Chris. *Analýza sítí a řešení problémů v programu Wireshark*. 1. vyd. Brno: Computer Press, 2012, 288 s. ISBN 978-80-251-3718-5.

[26] SOSINSKY, Barrie A. *Mistrovství – počítačové sítě*. Vyd. 1. Brno: Computer Press, 2010, 840 s. Mistrovství (Computer Press). ISBN 978-80-251-3363-7.

[27] WENYUAN, Xu, Dave HOLLINGER a Badri NATH. OSI Models & Data link layer. *University of South Carolina* [online]. 2007 [cit. 2014-04-28]. Dostupné z: http://www.cse.sc.edu/~wyxu/515Fall08/slides/OSI-Link.pdf

[28] ZANDBERGEN, Paul. Types of Networks: LAN, WAN, WLAN, MAN, SAN, PAN, EPN & VPN. *Education Portal* [online]. [cit. 2014-04-28]. Dostupné z: http://education-portal.com/academy/lesson/types-of-networks-lan-wan-wlan-man-san-pan-epn-vpn.html#lesson