

Simulátory a emulátory počítačových sítí

Marek Jaroš

Bakalářská práce
2025



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav počítačových a komunikačních systémů

Akademický rok: 2024/2025

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: Marek Jaroš
Osobní číslo: A22431
Studijní program: B0688A140008 Informační technologie v administrativě
Forma studia: Prezenční
Téma práce: Simulátory a emulátory počítačových sítí
Téma práce anglicky: Simulators and Emulators of Computer Networks

Zásady pro vypracování

- Provedte rešerši existujících nástrojů v oblasti simulátorů počítačových sítí.
- Porovnejte vlastnosti jednotlivých řešení.
- Na několika vybraných simulátorech vyzkoušejte a popište konfiguraci malé sítě.
- Vybrané simulátory vyhodnoťte z pohledu uživatele.

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

Seznam doporučené literatury:

1. *Computer Networks: A Systems Approach*, 2024. Online. Dostupné z: <https://book.systemsapproach.org/>. [cit. 2024-10-27]
2. ODOM, Wendell; HEALY, Rus a MEHTA, Naren. *Směrování a přepínání sítí: autorizovaný výukový průvodce. Samostudium*. Brno: Computer Press, 2009. ISBN 9788025125205.
3. HUCABY, Dave; MCQUERRY, Steve a WHITAKER, Andrew. *Cisco router configuration handbook*. 2nd ed. Indianapolis, IN: Cisco Press, c2010. ISBN 9781587141164.
4. *GNS3 Documentation*. Online. Dostupné z: <https://docs.gns3.com/docs/>. [cit. 2024-10-18]
5. *Cisco Packet Tracer*. Online. Dostupné z: <https://www.netacad.com/cisco-packet-tracer>. [cit. 2024-10-18]

Vedoucí bakalářské práce: **Ing. Jiří Korbel, Ph.D.**
Ústav počítačových a komunikačních systémů

Datum zadání bakalářské práce: **4. prosince 2024**

Termín odevzdání bakalářské práce: **9. června 2025**

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan



Ing. Miroslav Matýšek, Ph.D. v.r.
ředitel ústavu

Ve Zlíně dne 4. prosince 2024

PROHLÁŠENÍ AUTORA BAKALÁŘSKÉ PRÁCE

Beru na vědomí, že

- odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., v platném znění bez ohledu na výsledek obhajoby;
- bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému a bude dostupná k nahlédnutí;
- jedno vyhotovení bakalářské práce v listinné podobě bude ponecháno Univerzitě Tomáše Bati ve Zlíně k uložení;
- na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- podle § 60 odst. 2 a 3 mohu užít své dílo – bakalářskou práci – nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mě požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tj. k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá; neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji, že

- jsem na bakalářské práci pracoval(a) samostatně a použitou literaturu jsem řádně citoval(a); v případě publikace výsledků budu uveden(a) jako spoluautor;
- odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou obsahově totožné.
- při jazykové a stylistické úpravě textu jsem využil asistence umělé inteligence.

Ve Zlíně, dne

.....
podpis autora

ABSTRAKT

Tato bakalářská práce se zabývá řešením a porovnáním dostupných nástrojů pro simulaci počítačových sítí. Cílem je zmapovat současné možnosti v oblasti simulačních nástrojů, porovnat jejich vlastnosti a zaměřit se na jejich praktické využití. V rámci práce byly vybrány konkrétní simulátory, na kterých byla provedena konfigurace jednoduché síťové topologie. U každého z nich je popsán způsob práce, rozsah funkcionality a přívětivost uživatelského rozhraní. Následné zhodnocení nástrojů z pohledu běžného uživatele poskytuje doporučení pro výběr vhodného řešení podle konkrétních potřeb – ať už pro výuku, testování nebo vývoj síťových scénářů.

Klíčová slova: simulace počítačových sítí, emulátor, simulátor, ns-3, GNS3, Cisco Packet Tracer, OMNeT++, NetEm, síťová topologie, směrovací protokoly

ABSTRACT

This bachelor's thesis focuses on a review and comparison of available tools for computer network simulation. The main goal is to explore the current options in simulation software, compare their features, and focus on their practical use. Several specific simulators were selected, and each was used to configure a simple network topology. For each tool, the workflow, functionality, and user interface usability are described. Finally, the tools are evaluated from the perspective of an average user, and recommendations are provided for choosing the most suitable solution depending on specific needs—whether for education, testing, or network design.

Keywords: computer network simulation, emulator, simulator, ns-3, GNS3, Cisco Packet Tracer, OMNeT++, NetEm, network topology, routing protocols

Poděkování

Rád bych touto cestou poděkoval všem, kteří mě během zpracování této bakalářské práce podpořili.

Především děkuji vedoucímu práce Ing. Jiřímu Korbelovi, Ph.D. za cenné rady, vstřícnost, trpělivost a odborné vedení, které mi velmi pomohlo při orientaci v dané problematice.

Dále děkuji rodině a blízkým za jejich podporu, motivaci a pochopení v průběhu celého studia. V neposlední řadě bych rád poděkoval i vyučujícím a spolužákům, kteří mě během studia inspirovali a pomáhali mi překonávat překážky.

Tato práce by bez pomoci zmíněných osob nevznikla v podobě, jakou nyní má.

OBSAH

ÚVOD.....	13
I TEORETICKÁ ČÁST	14
1 SIMULÁTOR A EMULÁTOR.....	15
1.1 SIMULÁTORY	15
1.2 EMULÁTORY	15
2 NS-3.....	17
2.1 SIMULAČNÍ MODEL Y	17
2.2 ZÁKLADNÍ OBJEKTY	18
2.2.1 Uzel	18
2.2.2 Kanál	18
2.2.3 Aplikace	18
2.2.4 Síťové rozhraní.....	18
2.2.5 Topology helpers.....	19
2.3 JAK SE NS-3 POUŽÍVÁ	19
2.4 VÝHODY NS-3	20
2.5 NEVÝHODY NS-3	21
3 GNS3.....	22
3.1 JAK SE GNS3 POUŽÍVÁ.....	22
3.2 VÝHODY GNS3.....	24
3.3 NEVÝHODY GNS3.....	24
4 OMNET++	26
4.1 ARCHITEKTURA A HLAVNÍ KOMPONENTY OMNET++	26
4.1.1 Simulační jádro	26
4.1.2 Jazyk NED (Network Description Language)	27
4.1.3 Moduly	27
4.1.4 IDE (Integrated Development Environment).....	27
4.1.5 Rozšiřující Frameworky	27
4.2 JAK SE OMNET++ POUŽÍVÁ.....	28
4.2.1 Instalace OMNeT++.....	28
4.2.2 Vytvoření simulačního projektu.....	29
4.2.3 Definice modulů	29
4.2.4 Implementace chování (C++).....	30
4.2.5 Konfigurace a simulace (omnetpp.ini).....	30
4.2.6 Sběr a analýza výsledků	30
4.3 VÝHODY OMNET++.....	31
4.4 NEVÝHODY OMNET++	31
5 CISCO PACKET TRACER.....	32

5.1	HLAVNÍ FUNKCE A OBSAH	33
5.2	JAK SE POUŽÍVÁ.....	33
5.3	KOMPONENTY CISCO PACKET TRACER	34
5.3.1	Routery	34
5.3.2	Přepínače (Switches)	34
5.3.3	Access Pointy	34
5.3.4	Firewally.....	35
5.4	VÝHODY CISCO PACKET TRACER.....	35
5.5	NEVÝHODY CISCO PACKET TRACER.....	35
6	NETEM.....	36
6.1	FUNKCE NETEM	36
6.2	JAK SE NETEM POUŽÍVÁ	36
6.3	VÝHODY NETEM	37
6.4	NEVÝHODY NETEM.....	37
7	ROZDÍLOVÁ TABULKA.....	38
8	POPIS SMĚROVÁNÍ.....	40
8.1	STATICKE SMĚROVÁNÍ.....	40
8.2	DYNAMICKÉ SMĚROVÁNÍ.....	41
8.3	DISTANCE-VECTOR.....	41
8.3.1	RIP (Routing Information Protocol).....	41
8.4	LINK-STATE	42
8.4.1	OSPF (Open Shortest Path First).....	43
II	PRAKTICKÁ ČÁST	44
9	TESTOVÁNÍ GNS3	45
9.1	ZÁKLADNÍ SÍŤOVÁ TOPOLOGIE	45
9.2	JEDNODUCHÁ SÍŤOVÁ TOPOLOGIE	46
9.3	TOPOLOGIE S VYUŽITÍM RIP	49
9.4	TOPOLOGIE S VYUŽITÍM OSPF	51
10	TESTOVÁNÍ CISCO PACKET TRACER	54
10.1	JEDNODUCHÁ SÍŤOVÁ TOPOLOGIE	54
10.2	TOPOLOGIE S VYUŽITÍM STATICKEHO SMĚROVÁNÍ	55
10.3	TOPOLOGIE DYNAMICKÉHO SMĚROVÁNÍ S VYUŽITÍM RIP	59
10.4	TOPOLOGIE DYNAMICKÉ SMĚROVÁNÍ S VYUŽITÍM OSPF	61
11	SUBJEKTIVNÍ ZHODNOCENÍ	65
	ZÁVĚR.....	66
	SEZNAM POUŽITÉ LITERATURY	67

SEZNAM OBRÁZKŮ	70
SEZNAM TABULEK.....	71
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	72
SEZNAM PŘÍLOH.....	75

ÚVOD

V dnešní době, kdy jsou počítačové sítě stále složitější, je nezbytné, aby byly k dispozici nástroje umožňující efektivní návrh, testování a analýzu síťových infrastruktur bez potřeby fyzického vybavení. Simulátory a emulátory počítačových sítí jsou využívány k vytvoření bezpečného a flexibilního prostředí pro výuku, výzkum a praktické ověřování návrhů síťových topologií.

Tato bakalářská práce je zaměřena na rešerši existujících nástrojů pro simulaci a emulaci sítí. V teoretické části je proveden podrobný rozbor pěti konkrétních nástrojů: ns-3, GNS3, OMNeT++, Cisco Packet Tracer a NetEm. U každého nástroje je analyzována jeho architektura, způsob použití a jsou shrnuty jeho výhody a nevýhody.

V praktické části jsou vybrané nástroje otestovány a jsou na nich nakonfigurovány různé síťové topologie, přičemž je kladen důraz na směrovací protokoly. V závěru práce je poskytnuto subjektivní hodnocení práce s jednotlivými nástroji a jsou navržena konkrétní doporučení pro volbu vhodného řešení podle typu projektu a specifických požadavků.

I TEORETICKÁ ČÁST

1 SIMULÁTOR A EMULÁTOR

Emulátory i simulátory umožňují spouštět softwarové testy ve flexibilních, softwarově definovaných prostředích. Tímto způsobem umožňují spouštět testy rychleji a snadněji, než kdyby bylo nutné nastavit skutečné hardwarové zařízení. Oba jsou nezbytné pro zlepšení efektivity vývoje a rozšíření testovacího pokrytí, zejména v multiplatformních nebo více zařízeních. Proto se k provádění většiny softwarových testů obvykle používají simulátory a emulátory. Testování na reálných zařízeních se obvykle provádí až v pozdních fázích vývoje softwaru, těsně před jeho uvedením do produkčního prostředí. U emulátorů je obvykle nutné k dosažení tohoto cíle napsat emulátor pomocí assemblerového jazyka. Simulátory se však nepokoušejí emulovat skutečný hardware, který bude hostovat aplikaci v produkčním prostředí. Protože simulátory vytvářejí pouze softwarová prostředí, lze je implementovat pomocí programovacích jazyků vyšší úrovně. V jistém smyslu si tedy můžete emulátory představit jako něco, co okupuje střední cestu mezi simulátory a skutečnými zařízeními. Zatímco simulátory napodobují pouze funkce prostředí, které lze konfigurovat nebo definovat pomocí softwaru. [21][22]

1.1 Simulátory

Simulátory napodobují základní chování skutečných zařízení. Přenášejí prvky z reálného světa do virtuálního prostředí za účelem pochopení fungování daného systému. Simulují klíčové aspekty, avšak nemusí striktně dodržovat všechna pravidla reálného prostředí. Například při testování schopnosti aplikace odesílat data do jiné aplikace obvykle postačuje simulované prostředí, protože základní hardwarová konfigurace zpravidla nemá výrazný vliv na průběh datových transakcí. Stejně tak jsou simulovaná testovací prostředí vhodná pro ověření správného zobrazení uživatelského rozhraní aplikace v různých rozlišeních obrazovky. [21][22]

1.2 Emulátory

Emulátory napodobují všechny hardwarové a softwarové funkce pro produkční prostředí reálného zařízení. Emulace v podstatě znamená úplnou napodobeninu skutečné věci. Funguje to jen ve virtuálním prostředí namísto v reálném světě. Na druhou stranu jsou emulátory nejužitečnější, když je potřeba otestovat, jak software interaguje s podkladovým hardwarem nebo kombinací hardwaru a softwaru. [21][22]

Tabulka 1: Diference mezi Simulátorem a Emulátorem

(zdroj: [21])

Funkce	Emulátory	Simulátory
Reprezentace zařízení	Znovu vytvoří kompletní hardwarovou konfiguraci (procesor, paměť, senzory atd.).	Zaměřuje se na softwarové aspekty bez přesnosti na úrovni hardwaru.
Testovací schopnosti	Ideální pro testování funkcí závislých na hardwaru (např. senzory, grafika, chování specifické pro dané zařízení).	Nejlepší pro testování funkčnosti aplikace, chování UI/UX a výkonu za normálních podmínek.
Případ použití	Vhodné pro testování aplikací se složitými interakcemi zařízení nebo specifickými hardwarovými požadavky.	Vhodnější pro testování chování aplikace v konkrétním operačním systému nebo scénáři rozhraní.
Testování výkonu	Může simulovat výkon reálného světa, ale kvůli náročnosti zdrojů nemusí být tak rychlý jako skutečná zařízení.	Nesimuluje výkon zařízení v reálném světě, často se používá pro rychlé funkční testování.

2 NS-3

Ns-3 je simulátor síťového provozu založený na modelu diskrétních událostí, určený zejména pro výzkumné a vzdělávací účely. Jedná se o svobodný software dostupný pod licenci GNU GPLv2, který je volně k dispozici pro výzkum, vývoj i praktické využití. Hlavním cílem projektu ns-3 je vytvořit otevřenou a bezplatnou simulační platformu, která bude odpovídat požadavkům moderního síťového výzkumu. Projekt klade důraz na zapojení komunity, otevřené přispívání, vzájemné hodnocení a ověřitelnost kódu. Vývoj a údržbu ns-3 zajišťuje mezinárodní tým dobrovolných správců. [3]

2.1 Simulační modely

Projekt ns-3 se zaměřuje na vytvoření robustního simulačního jádra, které je důkladně zdokumentované, uživatelsky přívětivé a snadno laditelné. Cílem je pokrýt celý proces simulace – od její konfigurace až po sběr a analýzu trasovacích dat. [3]

Softwarová infrastruktura ns-3 umožňuje vývoj realistických simulačních modelů, díky nimž lze ns-3 využít jako síťový emulátor v reálném čase propojený s fyzickým světem. Zároveň podporuje opětovné použití mnoha existujících implementací reálných protokolů. [3]

Simulační jádro ns-3 podporuje výzkum jak IP sítí, tak i ne-IP sítí. Většina uživatelů se však soustředí na bezdrátové/IP simulace, které zahrnují modely pro Wi-Fi, LTE a další bezdrátové systémy na vrstvách 1 a 2. Mezi další oblíbená výzkumná témata patří výkon TCP (Transmission Control Protocol) a mobilních ad hoc směrovacích protokolů. [3]

Ns-3 obsahuje plánovač v reálném čase, který umožňuje různé scénáře „simulace in-the-loop“ pro interakci se skutečnými systémy. Uživatelé tak mohou například odesílat a přijímat pakety generované v ns-3 na reálných síťových zařízeních. Ns-3 také může sloužit jako propojovací rámec pro simulaci síťových efektů mezi virtuálními stroji. [3]

Simulátor klade důraz na opětovné využití skutečných aplikací a kódu jádra. Rámec Direct Code Execution umožňuje uživatelům spouštět aplikace napsané v C nebo C++ a také síťový zásobník linuxového jádra přímo v prostředí ns-3. [3]

2.2 Základní objekty

Před zahájením práce na ns-3 je vhodné seznámit se se základními objekty, které tento nástroj poskytuje. Tyto objekty zavádějí určitou úroveň abstrakce, což zajišťuje konfiguraci systému a zároveň spolu s větším přehledem výsledného kódu simulace. [2]

2.2.1 Uzel

V ns-3 je základní abstrakcí výpočetního zařízení tzv. uzel. V jazyce C++ je reprezentován třídou Node, která poskytuje metody pro správu a simulaci výpočetních zařízení. Uzel lze chápat jako počítač, ke kterému lze přidávat různé funkce, jako jsou aplikace, protokolové zásobníky a další síťové prvky. [2]

2.2.2 Kanál

Abstrakce komunikačního kanálu v ns-3 zajišťuje C++ třída Channel. Tato třída poskytuje metody pro správu a připojování uzlů k síti. Lze ji přirovnat k reálnému připojení počítače do lokální sítě, například prostřednictvím technologie Ethernet. Ns-3 nabízí několik specializovaných typů kanálů, mezi které patří CsmaChannel, PointToPointChannel a WifiChannel. [2]

2.2.3 Aplikace

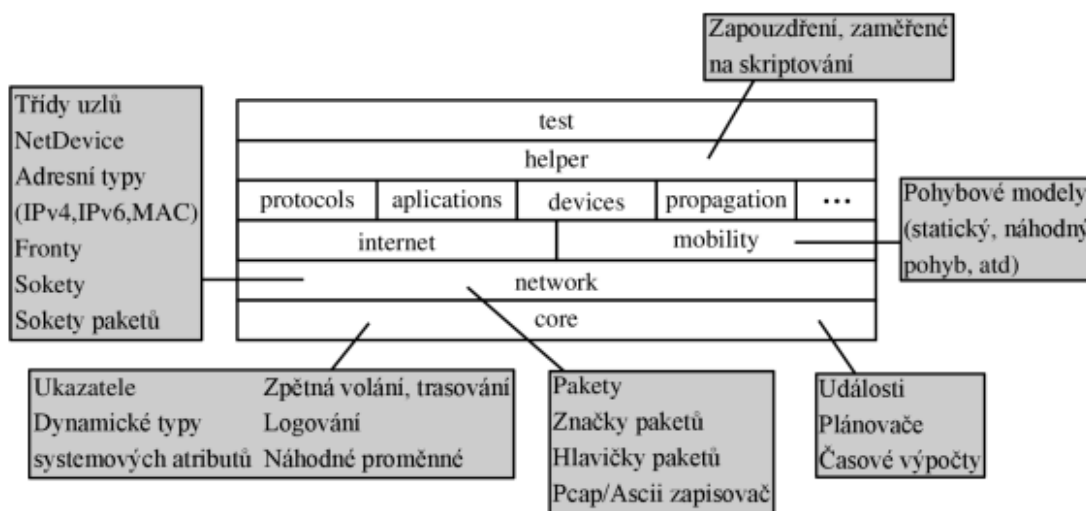
Program, který generuje aktivitu, je v ns-3 reprezentován třídou Application. Tato třída zajišťuje potřebnou funkcionalitu a poskytuje metody nezbytné pro její správu. Příkladem může být jednoduchá UDP (User Datagram Protocol) aplikace, která funguje v režimu klient-server. Konkrétně se jedná o UdpEchoClientApplication pro klientskou část a UdpEchoServerApplication pro serverovou část. [2]

2.2.4 Síťové rozhraní

V reálném světě je pro připojení počítače do sítě nezbytná síťová karta. V ns-3 tuto funkci zastřešuje třída NetDevice, která reprezentuje jak softwarový ovladač, tak simulovaný hardware. Aby mohl uzel komunikovat prostřednictvím komunikačního kanálu, je nutné do něj síťové zařízení nainstalovat. Podobně jako v běžném počítači může být jeden uzel připojen k více komunikačním kanálům pomocí několika instancí NetDevice. Ns-3 nabízí několik specializovaných typů těchto zařízení, například CsmaNetDevice, PointToPointNetDevice a WifiNetDevice, které odpovídají příslušným typům komunikačních kanálů. [2]

2.2.5 Topology helpers

V ns-3 slouží k usnadnění tvorby síťové topologie. Tyto nástroje automatizují proces připojování síťových rozhraní k uzlům a jejich následné propojení s komunikačními kanály, včetně automatického přiřazení adres, jako jsou IP a MAC adresy. Bez použití topology helperu by bylo k provedení těchto operací zapotřebí mnoho příkazů, což by vedlo ke složitějšímu a méně přehlednému kódu. Díky topology helperům je vytváření síťových modelů jednodušší a efektivnější. [2]



Obrázek 1: Struktura ns-3

(zdroj: [2])

2.3 Jak se ns-3 používá

1. Instalace – Chceme-li dostat ns-3 do počítače můžeme si jej stáhnout přes oficiální stránky ns-3 nebo přes GitHub. Instalace vyžaduje C++ kompilátor jako například (GCC/CLANG), Python, CMake a další nástroje. Doporučuje se instalovat pomocí Waf, což je build systém podporující ns-3. [1]
2. Vytvoření simulačního scriptu – Simulační scénáře se píšou v C++ nebo v Pythonu. Definuje se síťová topologie, uzly, protokoly a jejich konfigurace. Zde je příklad jednoduché simulace síťového spojení mezi dvěma uzly: [1]

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
```

```
using namespace ns3;

int main() {
    NodeContainer nodes;
    nodes.Create(2);

    PointToPointHelper pointToPoint;
    pointToPoint.SetDeviceAttribute("DataRate",
    StringValue("5Mbps"));
    pointToPoint.SetChannelAttribute("Delay", StringValue("2ms"));

    NetDeviceContainer devices = pointToPoint.Install(nodes);

    InternetStackHelper stack;
    stack.Install(nodes);

    Ipv4AddressHelper address;
    address.SetBase("192.168.1.0", "255.255.255.0");
    Ipv4InterfaceContainer interfaces = address.Assign(devices);

    Simulator::Run();
    Simulator::Destroy();

    return 0;
} [4]
```

Tento kód vytvoří dva uzly propejené pomocí Point-to-Point linky s rychlostí 5Mbps a zpožděním 2 ms.

3. Spuštění simulace – Kompilace a spuštění se provádí pomocí Waf:

```
./waf build
./waf --run "navez_skriptu" [4]
```

2.4 Výhody ns-3

- Realističnost – Podporuje skutečné protokoly a lze jej propojit s reálným hardwarem.
- Flexibilita – Modulární architektura umožňuje rozšiřování funkcionality a tvorbu vlastních modelů.
- Podpora široké škály technologií – Wi-Fi, LTE, 5G, Bluetooth, MANET, IoT a další.

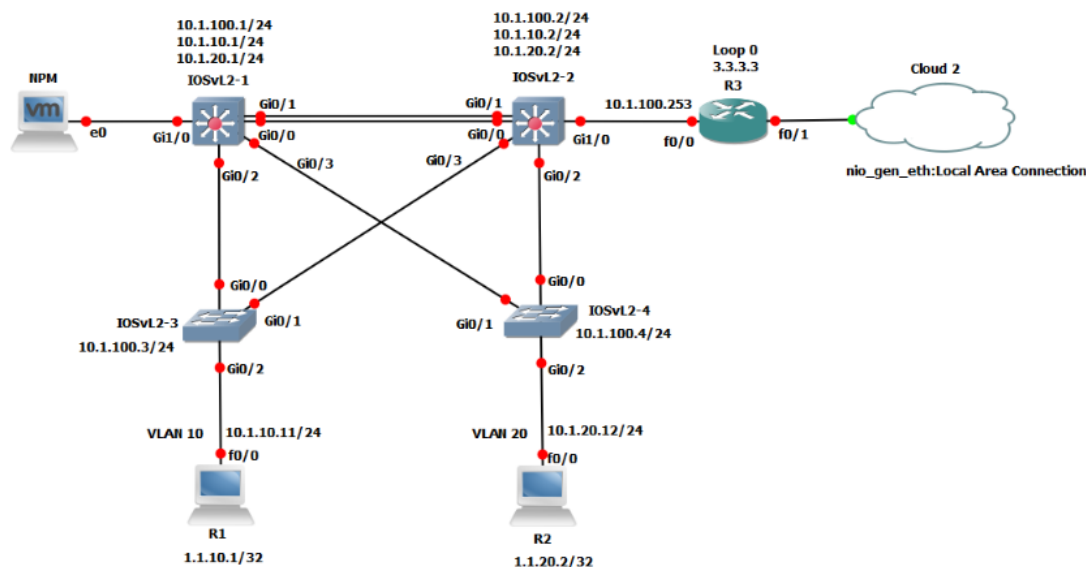
- Interoperabilita – Možnost propojení s Linux TCP/IP stackem (DCE), integrace se softwarem jako Wireshark.
- Podpora více jazyků – C++ a Python umožňují flexibilní skriptování simulací. [8]

2.5 Nevýhody ns-3

- Strmá křivka učení – Práce s ns-3 vyžaduje pokročilé znalosti C++ a síťových protokolů.
- Vyšší nároky na hardware – Simulace rozsáhlých sítí může být výpočetně náročná.
- Omezená vizualizace – Na rozdíl od některých jiných simulátorů nemá ns-3 vestavěné grafické rozhraní. [8]

3 GNS3

GNS3 (Graphical Network Simulator) je grafický síťový simulátor a emulátor, který umožňuje vytvářet složité síťové topologie pomocí reálných obrazů síťových zařízení (např. Cisco IOS (Internetwork Operating System), Juniper, MikroTik apod.). Vznikl v roce 2008 jako open-source projekt Jeana-Phillippe Boucharda a Jeremyho Grossmanna. Na rozdíl od jiných simulátorů (např. Packet Tracer), GNS3 nepracuje pouze s předdefinovanými modely – emuluje skutečné operační systémy zařízení. GNS3 lze charakterizovat jako grafickou nadstavbu nad nástrojem Dynamips, která uživatelům poskytuje přehledné a intuitivní rozhraní pro práci se síťovými topologiemi. Kromě Dynamipsu podporuje GNS3 také další emulační nástroje, jako jsou VirtualBox, QEMU nebo PEMU. Tyto nástroje slouží k emulaci různých typů zařízení – například počítačových hostů s operačními systémy Linux nebo Windows, Cisco přístupových bodů ASA (Adaptive Security Appliance), Cisco PIX firewallů či systémů Cisco IPS určených k detekci a prevenci průniků do sítě. [5][6]



Obrázek 2: Architektura GNS3

(zdroj: [5])

3.1 Jak se GNS3 používá

Prvním krokem je instalace GNS3 – Nejbezpečnější je oficiální stránka <https://www.gns3.com>, doporučuje se instalovat zároveň GNS3 VM, což je virtuální stroj (pro VirtualBox, VMware Workstation/ESXi), kde běží samotná emulace a odlehčuje hlavnímu počítači. [7][6]

Druhý krok: Přidání zařízení a obrazců – GNS3 samotný nezahrnuje síťové operační systémy kvůli licencím. Uživatel si musí obstarat a naimportovat obrazy zařízení:

- Cisco IOS – (.bin) pro routery a starší switche
- Firewall Cisco ASA – často ve formátu .qcow2
- Virtuální stroje – například Ubuntu, VyOS nebo Windows ve formátu ISO nebo QCOW2.

Po nahrání image GNS3 vytvoří „template“, který se pak používá při stavbě. [7][6]

Krok třetí: Sestavování topologie – GNS3 nabízí intuitivní prostředí pro snadné sestavení sítě metodou „drag and drop“. Zařízení se přetahují z postranního panelu do hlavního pracovního prostoru. Propojky (kabely) se vytvářejí tažením mezi rozhraními jednotlivých zařízení. V síti lze využít routery, switche, firewally, servery, klientské stanice i externí fyzická zařízení. Podporována je i kombinace různých technologií, například propojení Cisco routeru, pfSense firewallu, Linux serveru a Windows stanice v rámci jedné sítě. [7]

Krok čtvrtý: Konfigurace zařízení – Po spuštění topologie lze u zařízení zvolit možnost Console. V případě Cisco IOS se otevře konzolové okno s rozhraním příkazového řádku (CLI). Zařízení je poté možné konfigurovat stejným způsobem jako v reálném prostředí.

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet0/0
Router(config-if)# ip address 192.168.1.1 255.255.0
Router(config-if)# no shutdown [8]
```

Lze testovat běžně používané protokoly, jako například OSPF, BGP, RIP, VLAN, STP, NAT, VPN, IPv6 atd.[7]

Krok pátý: Testování a sledování síťového provozu – Je možné pomocí vestavěných nástrojů nebo propojením s aplikací Wireshark. Tímto způsobem lze sledovat paketovou komunikaci mezi zařízeními. K analýze se využívají běžné síťové příkazy.

```
ping 192.168.1.2
traceroute 8.8.8.8
show ip route
show interface status [8]
```

Lze ověřit, zda komunikace mezi zařízeními funguje podle očekávání. [7]

Krok šestý: Ukládání projektů – GNS3 umožňuje ukládat kompletní projekty včetně topologie, kabeláže a konfigurace zařízení. Projekt lze kdykoliv znovu načíst a pokračovat v práci. Praktické například při studiu na certifikace, kdy je možné připravit různé scénáře a postupně je zdokonalovat. [7][8]

Krok sedmý: Propojení s reálným světem – GNS3 podporuje připojení emulovaných zařízení k fyzickým síťovým rozhraním tvého počítače. Lze připojit virtuální router přímo do reálné domácí nebo firemní sítě. Ideální pro testování migrací, simulaci reálného nasazení, nebo propojení GNS3 s jinými virtualizačními platformami (například VMware ESXi (Elastic Sky X Integrated), KVM (Kernel-based Virtual Machine) nebo Hyper-V). [7][8]

3.2 Výhody GNS3

- Realistická simulace – Pracuje s reálnými obrazy zařízení.
- Flexibilita – Lze kombinovat různé výrobce, OS i virtualizační platformy.
- Skvělý nástroj pro učení – Umožňuje testovat konfigurace bez obav o reálný hardware.
- Komunita a dokumentace – Mnoho dostupných návodů a labů.
- Open-source – Základní funkcionality zdarma [5]

3.3 Nevýhody GNS3

- Vyžaduje výkonný počítač – Větší topologie nebo více zařízení (např. Cisco IOS-XRv) jsou náročné na RAM a CPU.
- Některé image nejsou součástí instalace – Uživatel si je musí opatřit sám (oficiálních zdrojů nebo přes Cisco Academy).
- Strmá křivka učení – Pro začátečníky v sítích nebo vizualizaci může být GNS3 složitější.
- Potřeba konfigurace GNS3 VM – Pro složitější simulace je nutné nastavit GNS3 server/VM a správně propojit s GPU. [5]

Tabulka 2: Porovnání GNS3 s ostatními SW

(zdroj: vlastní)

Funkce	GNS3	Cisco Packet Tracer	EVE-NG	Boson NetSim
Reálné obrazy	Ano	Ne (Simulované)	Ano	Ne
Podpora více výrobců	Ano	Ne (pouze Cisco)	Ano	Ne
Snadnost použití	Střední	Snadné	Pokročilé	Jednoduché
Emulace VM	Ano	Ne	Ano	Ne
Vhodné pro certifikace	Ano (CCNA/NP/IE)	Ano (CCNA)	Ano	Ano

4 OMNeT++

OMNeT++ je rozšiřitelná, modulární, komponentně založená C++ simulační knihovna a framework, primárně pro vytváření síťových simulátorů. „Síť“ je myšlena v širším smyslu, který zahrnuje kabelové a bezdrátové komunikační sítě, sítě na čipu, sítě ve frontě a tak dále. Funkčnost specifická pro doménu, jako je podpora senzorových sítí, bezdrátových ad-hoc sítí, internetových protokolů, modelování výkonu, fotonických sítí atd., je poskytována modelovými frameworky, vyvinutými jako nezávislé projekty. OMNeT++ nabízí IDE založené na Eclipse, grafické runtime prostředí a řadu dalších nástrojů. Existují rozšíření pro simulaci v reálném čase, emulaci sítě, integraci databáze, integraci SystemC a několik dalších funkcí. OMNeT++ je distribuován pod Academic Public License . Ačkoli OMNeT++ není samotný síťový simulátor, získal si širokou popularitu jako platforma pro simulaci sítí ve vědecké komunitě i v průmyslovém prostředí a vybudoval si velkou uživatelskou komunitu. OMNeT++ poskytuje komponentovou architekturu pro modely. Komponenty (*moduly*) jsou naprogramovány v C++ a následně sestaveny do větších komponent a modelů pomocí vysokoúrovňového jazyka (*NED*) . Opětovné použití modelů je zdarma. OMNeT++ má rozsáhlou podporu GUI (Graphical User Interface) a díky jeho modulární architektuře lze simulační jádro (a modely) snadno zabudovat do vašich aplikací. [9]

4.1 Architektura a hlavní komponenty OMNeT++

Architektura OMNeT++ je navržena s důrazem na modularitu, flexibilitu a opětovanou použitelnost kódu. Systém je rozdělen do několika hlavních komponent, které spolu spolupracují. [9]

4.1.1 Simulační jádro

Simulační jádro OMNeT++ je implementováno v programovacím jazyce C++. Jeho hlavním úkolem je řídit běh simulace na základě modelu diskrétní simulace událostí (anglicky discrete event simulation). Součástí jeho funkcionality je správa simulačního času, plánování jednotlivých událostí, zajištění komunikace mezi moduly a sběr statistických údajů během simulace. [8]

4.1.2 Jazyk NED (Network Description Language)

NED (Network Description Language) je deklarativní jazyk určený pro popis struktury systému v OMNeT++. Pomocí tohoto jazyka lze definovat typy modulů, a to jak jednoduchých (simple), tak složených (compound), dále jejich parametry, porty a způsob vzájemného propojení. NED soubory jsou následně automaticky překládány do jazyka C++ a integrovány do výsledné simulace. [11]

4.1.3 Moduly

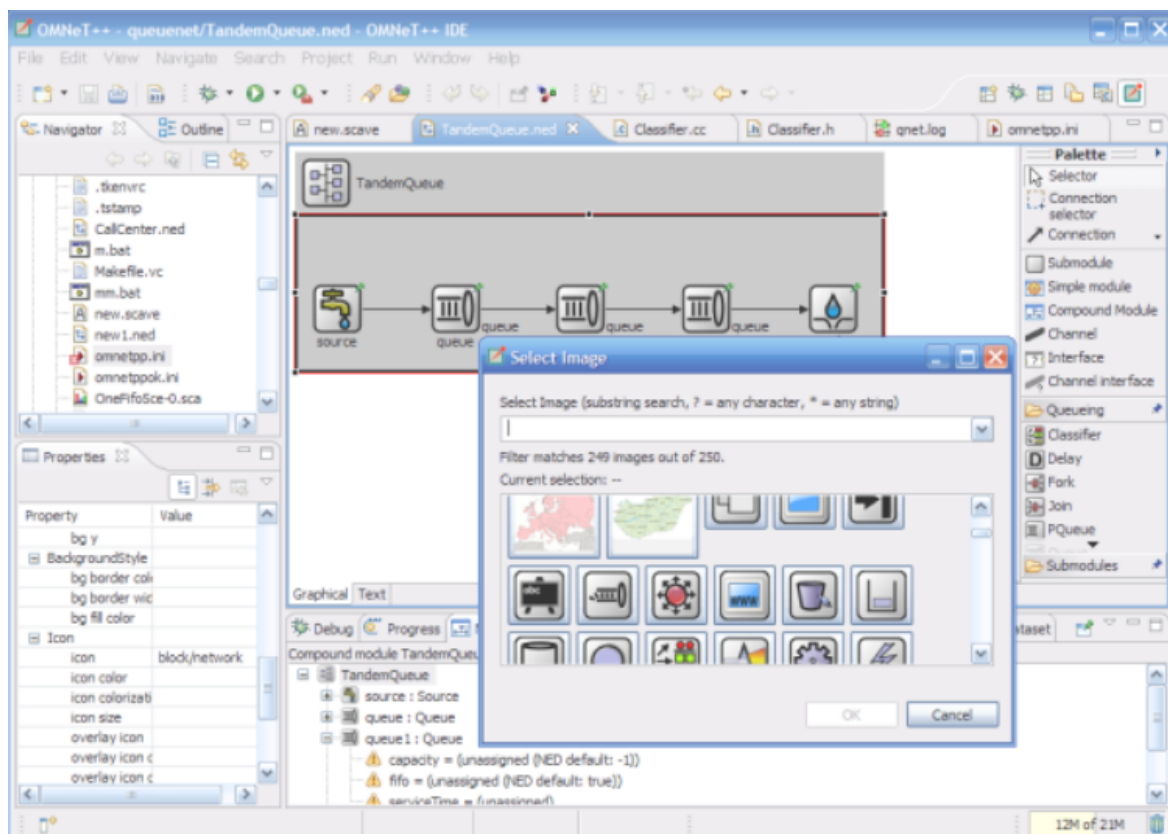
Jednoduché moduly (simple modules) jsou v OMNeT++ implementovány v jazyce C++ a obsahují logiku jednotlivých komponent systému. Složené moduly (compound modules) se skládají z několika jednoduchých modulů a definují způsob jejich vnitřního propojení. Komunikace mezi jednotlivými moduly probíhá prostřednictvím zpráv (messages), které jsou mezi nimi přenášeny přes definované komunikační porty. [10]

4.1.4 IDE (Integrated Development Environment)

OMNeT++ obsahuje vlastní integrované vývojové prostředí (IDE), které je postavené na platformě Eclipse. Toto IDE umožňuje vytvářet a upravovat soubory v jazyce C++ a NED, sestavovat projekty pomocí systému Makefile, spravovat a konfigurovat simulace a rovněž provádět ladění a vizualizaci průběhu simulací. [12]

4.1.5 Rozšiřující Frameworky

INET Framework je základní knihovnou síťových protokolů pro OMNeT++, která zahrnuje implementace technologií jako Ethernet, IP, TCP, UDP, IPv6 a dalších. Kromě toho obsahuje modely pro simulaci bezdrátových sítí WLAN, mobilních ad-hoc sítí MANET, mobilních sítí LTE a mnoha dalších síťových technologií. Veins je specializovaný framework určený pro simulaci vehikulárních sítí, přičemž je úzce integrován se simulátorem mobility SUMO, který zajišťuje realistickou simulaci pohybu vozidel. Castalia je framework zaměřený na simulaci bezdrátových senzorových sítí (Wireless Sensor Networks), přičemž klade důraz na modelování fyzické vrstvy a energetické spotřeby uzlů. SimuLTE představuje specializovaný framework pro modelování a simulaci sítí čtvrté generace (LTE), včetně pokročilých aspektů rádiového přístupu a správy mobilních uživatelů. [8][9]



Obrázek 3: OMNeT++ IDE, úprava souboru NED

(zdroj: [11])

4.2 Jak se OMNeT++ používá

OMNeT++ se používá jako simulační prostředí pro vytváření konfigurací, spuštění a vyhodnocení modelů síťových nebo distribuovaných systémů.

4.2.1 Instalace OMNeT++

OMNeT++ je dostupný ke stažení z oficiálních stránek <https://omnetpp.org/download/>. Funguje pro operační systémy Windows, Linux i macOS. Instalační balíček obsahuje integrované vývojové prostředí (IDE) založené na Eclipse, simulační jádro a potřebný C++ kompilátor (například MSYS2 pro Windows). Po rozbalení stačí spustit instalační skript `configure`, následně příkaz `make`, případně lze využít grafického instalačního průvodce. Začátečnickům se doporučuje používat předkompilovaný balík s integrovaným IDE, který umožňuje rychlé spuštění bez složité konfigurace. [13]

4.2.2 Vytvoření simulačního projektu

V prostředí OMNeT++ IDE se nový projekt vytvoří pomocí nabídky File → New → OMNeT++ Project, kde zadáme název projektu, například Demo. Projekt je strukturován do několika částí: složka `src/` slouží pro uložení C++ implementace, složka `ned/` obsahuje síťové popisy vytvořené v jazyce NED a soubor `omnetpp.ini` slouží jako konfigurační soubor pro nastavení a řízení simulací. Pokud máme v úmyslu simulovat síťové protokoly, můžeme do projektu přidat například INET Framework, který obsahuje podporu pro TCP/IP stack, WiFi, IPv6, a další běžně používané síťové komponenty. [13]

4.2.3 Definice modulů

OMNeT++ je navržen jako modulárně orientované simulační prostředí, kde každý prvek sítě, jako je například router, klient nebo server, je reprezentován jako samostatný modul. Existují dva základní typy modulů: simple modules (jednoduché moduly), které mají své chování implementováno v jazyce C++ – například `Host.cc` a jeho odpovídající popis v `Host.ned`, a compound modules (složené moduly), které se skládají z více jednoduchých modulů – například celá síť může být definována v souboru `Network.ned`. Každý modul má vstupní a výstupní brány (gates), kterými si vzájemně předávají zprávy (messages), čímž dochází ke komunikaci mezi jednotlivými částmi simulované sítě. [13]

Jednoduchý model může být definován takto:

```
simple Host {
    gates:
        input in;
        output out;
} [8]
```

Složený modul:

```
network MyNetwork {
    submodules:
        h1: Host;
        h2: Host;
    connections:
        h1.out --> h2.in;
        h2.out --> h1.in;
} [8]
```

4.2.4 Implementace chování (C++)

Vytvoříme .cc soubor (např. `Host.cc`), kde definujeme, co modul dělá např. odesílá zprávu, čeká nebo ji zpracovává. Zjednodušený příklad:

```
class Host : public cSimpleModule {
protected:
    virtual void initialize() override {
        scheduleAt(simTime() + 1, new cMessage("Ping"));
    }

    virtual void handleMessage(cMessage *msg) override {
        EV << "Zpráva přijata: " << msg->getName() << endl;
        delete msg;
    }
}; [8]
```

Simulace probíhá jako posloupnost událostí (odeslání/příjem zprávy).

4.2.5 Konfigurace a simulace (omnetpp.ini)

V souboru `omnetpp.ini` se nastavují základní parametry simulace, jako například to, který síťový model se má použít, kolik bude obsahovat uzlů, jaké budou délky spojů, latence, typy zpráv nebo například seed pro generování náhodných hodnot. Tento konfigurační soubor umožňuje řídit chování simulace bez nutnosti měnit zdrojový kód. Jednoduchý příklad může vypadat takto:

```
[General]
network = MyNetwork
sim-time-limit = 10s
tkenv-plugin-path = ../../../../src [8]
```

Dále je možné v tomto souboru definovat i více simulací s různými nastaveními (tzv. runs), což je užitečné pro statistickou analýzu a porovnání výsledků při změně konkrétních parametrů simulace. [13]

4.2.6 Sběr a analýza výsledků

Výsledky simulací v OMNeT++ se automaticky ukládají do dvou typů souborů: `.sca` a `.vec`. Soubor s příponou `.sca` obsahuje souhrnné statistiky, jako je například počet přenesených zpráv, průměrná doba přenosu nebo míra ztrát. Naproti tomu soubor `.vec`

uchovává časové hodnoty, tedy například vývoj RTT (Round Trip Time) v čase nebo změny v počtu zpráv během simulace.

K analýze těchto dat lze využít vestavěný nástroj OMNeT++ Analysis Tool, který umožňuje vizualizaci výsledků formou grafů a tabulek. Pro pokročilejší zpracování je možné výsledky exportovat do formátu CSV (Comma-Separated Values) a dále je analyzovat pomocí externích nástrojů, jako jsou Python, MATLAB nebo R. Kromě toho OMNeT++ podporuje i porovnání více simulačních běhů, což umožňuje například sledovat, jak změna parametrů, jako je šířka pásma, ovlivňuje výkonnost sítě. [11]

4.3 Výhody OMNeT++

- Flexibilita – Může simulovat nejen síť, ale jakýkoli distribuovaný systém.
- Modularita – Přehledné dělení na moduly – snadná rozšiřitelnost a údržba.
- Open source – Zdarma pro akademické použití, přístup ke zdrojovému kódu.
- Bohaté knihovny – INET Framework a další projekty poskytují hotové implementace síťových protokolů.
- Grafická podpora – Grafická simulace (animace zpráv) usnadňuje ladění a demonstraci simulací.
- Dokumentace a komunita – Dobrá oficiální dokumentace a aktivní akademická komunita. [8]

4.4 Nevýhody OMNeT++

- Strmá křivka učení – Nutnost zvládnout C++ a specifický jazyk NED.
- Náročnější konfigurace – Nastavení větších projektů může být komplikovaná.
- Méně vhodné pro reálnou emulaci – Není optimalizován pro přímou komunikaci s reálnými zařízeními jako GNS3.
- Výkonné limity – Při rozsáhlých simulacích s tisíci uzly může být simulace pomalá. [8]

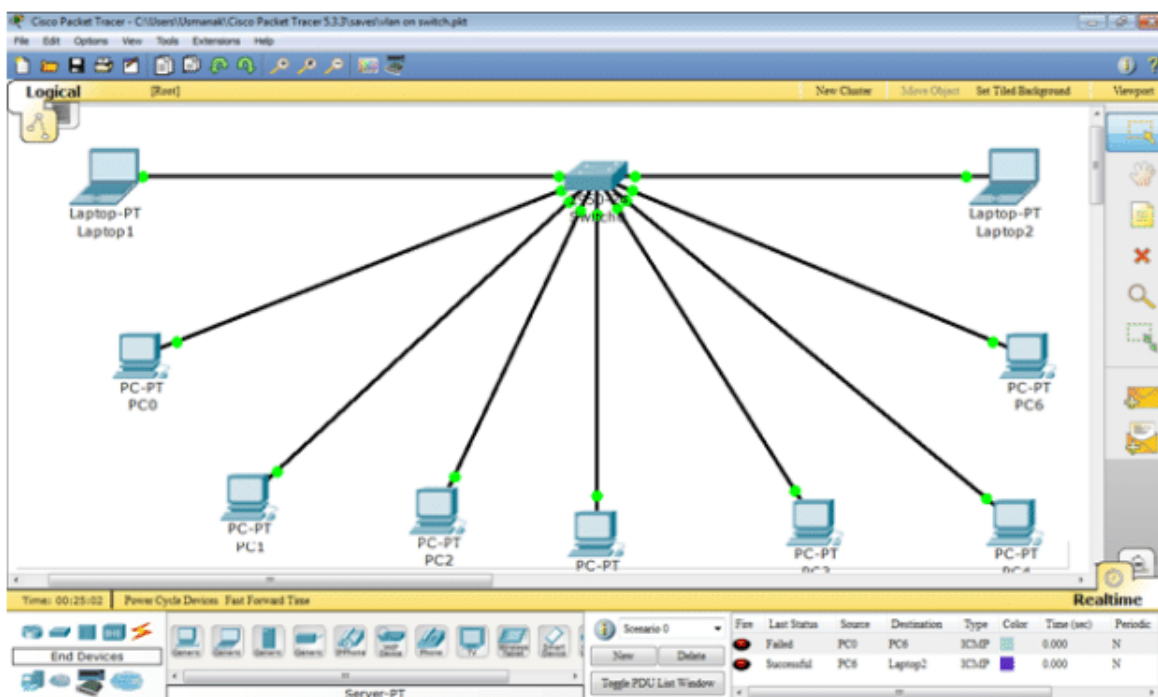
5 CISCO PACKET TRACER

Cisco Packet Tracer je výkonný síťový simulační nástroj vyvinutý společností Cisco Systems. Je určen primárně pro výukové a tréninkové účely, především v rámci Cisco Networking Academy, ale je dostupný zdarma i pro veřejnost.

Packet Tracer umožňuje uživatelům simulovat chování skutečných sítí bez nutnosti fyzického vybavení, což z něj činí ideální nástroj pro studenty, začátečníky i pokročilé techniky připravující se na certifikace jako CCNA (Cisco Certified Network Associate).

Jedná se o grafické prostředí, které kombinuje funkční emulaci síťových zařízení (routery, switche, PC, servery atd.) s nástroji pro analýzu síťové komunikace a ladění topologií.

Ačkoli má oproti profesionálním nástrojům určitá omezení (např. chybí podpora všech pokročilých protokolů), nabízí výbornou rovnováhu mezi jednoduchostí a funkcí. [15]



Obrázek 4: Příklad topologie v Cisco Packet Tracer

(zdroj: [14])

5.1 Hlavní funkce a obsah

Cisco Packet Tracer obsahuje řadu klíčových komponent a funkcí, které jej činí ideálním nástrojem pro výuku počítačových sítí. Umožňuje simulaci základních síťových zařízení, jako jsou Cisco routery, přepínače, přístupové body, osobní počítače, notebooky, servery, tablety a další síťové prvky.

Podporuje širokou škálu síťových protokolů, včetně DHCP, DNS, HTTP, FTP, ICMP, RIP, OSPF, VLAN, STP, NAT, IPv4/IPv6, EIGRP a dalších. Nabízí dva simulační režimy: Simulation Mode, který zpomaluje a vizuálně zobrazuje průběh síťové komunikace (vhodný zejména pro výukové účely a ladění), a Real-Time Mode, který simuluje chování sítě v reálném čase. Packet Tracer také podporuje programovatelná zařízení – uživatelé mohou využívat různé typy IoT (Internet of Things) zařízení, senzorů a aktuátorů a k jejich řízení používat skripty v jazyce Python.

Jednou z výrazných výhod je možnost práce s více topologiemi současně, což umožňuje simulaci komplexních sítí rozdělených do několika vrstev nebo podsítí. Vytváření topologií je velmi intuitivní díky drag-and-drop rozhraní, které usnadňuje přetahování zařízení a jejich propojení pomocí virtuálních kabelů.

Navíc Packet Tracer podporuje tzv. aktivity Cisco NetAcad, což jsou předpřipravené úlohy a scénáře vytvořené instruktory v rámci programu Cisco Networking Academy. Tyto aktivity pomáhají studentům procvičovat praktické dovednosti v reálném prostředí bez potřeby fyzického hardwaru. [15]

5.2 Jak se používá

Použití Cisco Packet Traceru probíhá ve třech základních krocích. Prvním krokem je vytvoření topologie, kdy uživatel přetáhne na pracovní plochu síťová zařízení, jako jsou routery, switche a koncové stanice, a propojí je vhodnými typy kabelů – například straight-through, crossover nebo sériovými spoji.

Druhým krokem je konfigurace zařízení. Ta může být provedena buď pomocí grafického rozhraní, nebo přímo přes CLI (Command Line Interface). Uživatel nastavuje různé síťové parametry, jako jsou IP adresy, směrovací protokoly (např. RIP nebo OSPF), VLANy, přístupová hesla, pravidla NAT a další konfigurace potřebné pro správný chod sítě.

Ve třetím kroku následuje testování a ladění. V režimu „Simulation Mode“ lze detailně sledovat tok paketů v síti – jak jsou směrovány, jak se vyhodnocují směrovací tabulky a zda

dochází k úspěšné komunikaci, nebo naopak k chybám. Tento režim je velmi vhodný pro výukové účely, protože pomáhá pochopit interní chování sítě.

Cisco Packet Tracer je velmi často využíván ve vzdělávání, především v rámci programu Cisco Networking Academy (NetAcad). Studenti si v něm mohou prakticky zkusit konkrétní scénáře, procvičovat dovednosti při řešení úloh a připravovat se na certifikační zkoušky jako například CCNA. [16]

5.3 Komponenty Cisco Packet Tracer

Cisco Packet Tracer obsahuje rozsáhlou sadu hardwarových a softwarových komponent, které simulují skutečné síťové prvky.

5.3.1 Routery

Slouží ke směrování paketů mezi různými sítěmi a tvoří páteřní prvky každé rozsáhlejší síťové infrastruktury. Cisco Packet Tracer umožňuje simulaci různých modelů routerů, jako například 2911, 1941 nebo 4321, přičemž každý z nich nabízí rozdílný počet síťových rozhraní. Routery podporují širokou škálu směrovacích a síťových protokolů, včetně RIP, OSPF, EIGRP, BGP, NAT či DHCP Relay. Zařízení je možné konfigurovat buď pomocí rozhraní příkazové řádky (CLI), nebo přes grafické uživatelské rozhraní. [17]

5.3.2 Přepínače (Switches)

Jsou zařízení určená k přepínání datových rámců na základě MAC (Media Access Control) adres a propojují koncová zařízení v rámci jedné lokální sítě (LAN). Packet Tracer nabízí například modely 2960, 3560 a multilayer přepínače, které kromě základních funkcí podporují i pokročilé vlastnosti jako jsou VLANy, trunking, Spanning Tree Protocol (STP), port security, VTP (VLAN Trunking Protocol), EtherChannel a implementace přístupových seznamů (ACL) na vrstvě L2 i L3. [18]

5.3.3 Access Pointy

Poskytují bezdrátové připojení klientským zařízením a jsou klíčové pro simulaci Wi-Fi sítí. Uživatelé mohou nakonfigurovat SSID (název bezdrátové sítě), zabezpečení pomocí protokolů WEP či WPA a integrovat AP do IoT scénářů. [19]

5.3.4 Firewally

Například Cisco ASA (Adaptive Security Appliance), představují bezpečnostní zařízení, která chrání síť před neoprávněným přístupem. I když je v Packet Traceru dostupná pouze omezená verze těchto zařízení, lze na nich simulovat základní bezpečnostní pravidla, jako je NAT, VPN, ACL (Access Control Lists) nebo Stateful Packet Inspection. [20]

5.4 Výhody Cisco Packet Tracer

- Zdarma pro uživatele Cisco NetAcad.
- Nenáročný na hardware – běží i na slabších počítačích.
- Vizuální přístup – intuitivní práce s topologií.
- Podpora CLI i GUI konfigurace.
- Přehledná simulace datových toků v režimu Simulation Mode.
- Aktivní komunita a podpora ze strany Cisco. [15]

5.5 Nevýhody Cisco Packet Tracer

- Není open-source – zdrojový kód není k dispozici.
- Nepodporuje plnou funkcionalitu IOS jako reálná zařízení nebo GNS3.
- Chybí pokročilé směrovací protokoly.
- Nemožnost nahrát vlastní IOS obrazy – vše je omezeno na to, co Cisco povolí. [8]

6 NETEM

NetEm (zkratka pro Network Emulator) je open-source nástroj integrovaný do Linuxového jádra, který umožňuje simulovat různé síťové podmínky, jako je zpoždění, ztráta paketů, duplikace, přetížení nebo jitter. Používá se primárně k testování síťových aplikací a protokolů za nestandardních nebo degradovaných síťových podmínek. NetEm není samostatný program, ale součást subsystému Traffic Control (tc) v Linuxu (balík iproute2). Funguje jako fronta (qdisc – queueing discipline) a umožňuje manipulaci s příchozím a odchozím síťovým provozem. [23]

6.1 Funkce NetEm

NetEm umožňuje simulovat různé síťové podmínky, které běžně nastávají v reálných sítích. Mezi základní funkce patří přidání zpoždění (delay), které může být pevně dané nebo náhodně proměnlivé.

Variabilita zpoždění neboli jitter simuluje kolísání v čase doručení paketů, což je typické například pro bezdrátové nebo přetížené sítě.

Funkce packet loss umožňuje emulovat ztrátu paketů, například 5 % všech odeslaných dat může být ztraceno, což je vhodné pro testování robustnosti přenosových protokolů. Duplikace paketů (packet duplication) vytváří kopie některých paketů, čímž se zvyšuje objem dat a komplikuje jejich zpracování.

Packet reordering mění pořadí přenosu paketů, což může odhalit chyby v implementaci protokolů závislých na pořadí doručení.

Packet corruption umožňuje náhodně poškodit obsah paketů, například změnou bitů, čímž se testuje schopnost detekce a opravy chyb.

Nakonec funkce rate limiting slouží k omezení přenosové rychlosti, například na 1 Mbps, a simuluje tak podmínky omezené šířky pásma. [23][8]

6.2 Jak se NetEm používá

Ovládá se pomocí nástroje tc (Traffic Control). NetEm je součástí systému queueing discipline (qdisc) v Linuxu. Působí na úrovni kernelového síťového zásobníku a je přiřazován k výstupní frontě (outgoing queue) síťového rozhraní. Zde je pár příkladů příkazů:

```
sudo tc qdisc add dev eth0 root netem delay 100ms
```

Tento příkaz přidá zpoždění 100ms na všechna odchozí data přes rozhraní `eth0`.

Příkaz na ztrátu 10 % paketů:

```
sudo tc qdisc add dev eth0 root netem loss 10%
```

Zpoždění s jitterem:

```
sudo tc qdisc add dev eth0 root netem delay 100ms 20ms distribution  
normal
```

Kombinace více efektů:

```
sudo tc qdisc add dev eth0 root netem delay 100ms loss 5% duplicate 1%
```

Odebrání pravidel:

```
sudo tc qdisc del dev eth0 root [24]
```

NetEm lze také použít s virtualizačními nástroji jako Docker, QEMU, GNS3, Mininet, kde je možné emulovat síťové podmínky mezi virtuálními uzly. [24][8]

6.3 Výhody NetEm

- Přesná emulace reálných síťových chyb (např. mobilní sítě, Wi-Fi, satelitní připojení).
- Snadné použití na jakémkoliv Linuxovém systému.
- Integrace s dalšími nástroji (Docker, Mininet, iptables, iperf, Wireshark).
- Zcela zdarma bez licencí – open-source.
- Nízká režie na výkon.
- Skriptovatelnost – vhodné pro automatizaci testů. [8]

6.4 Nevýhody NetEm

- Není dostupný pro Windows nebo macOS bez virtualizace.
- Složitější konfigurace pro pokročilé scénáře.
- Omezený GUI – převážně příkazová řádka. (CLI)
- Vyžaduje root oprávnění. [8]

7 ROZDÍLOVÁ TABULKA

Kritérium	Ns3	GNS3	OMNeT++	Cisco Packet Tracer	NetEm
Typ nástroje	Simulátor	Emulátor (s částečnou simulací)	Simulátor	Simulátor (s prvky emulace)	Emulátor síťových zpoždění (kernel-level)
Úroveň modelování	Nízká (paketová, protokolová)	Reálné zařízení (emulace operačních systémů a IOS)	Modulární simulace systémů na základě událostí	Grafický simulátor Cisco zařízení a protokolů	Emulace na úrovni kernelu (latence, jitter, packet loss)
GUI	Ne	Ano	Ano	Ano	Ne
Cílové použití	Výzkum, vývoj protokolů, akademická simulace	Trénink konfigurace síťových zařízení, zkoušky	Akademický výzkum, síťové systémy, vlastní protokoly	Výuka v Cisco akademiích, základní trénink síťování	Testování síťových podmínek (např. latence, výpadky)
Podpora IoT	Omezená	Ano	Ano	Ano	Ne
Realismus	Simulace protokolů na úrovni zdrojového kódu	Vysoce realistické (reálné IOS, QEMU, VirtualBox)	Simulace s definovanými chováním modulů	Nižší úroveň realismu, vhodné pro výuku	Vysoce realistické z hlediska zpoždění a poruchových jevů

Využitelné síťové protokoly	Vysoká podpora (TCP/IP, IPv6, WiFi, LTE, 5G atd.)	Výborné pro konfiguraci skutečných zařízení	Vysoká podpora přes INET Framework	Dobré pro základní protokoly (RIP, OSPF, DHCP, NAT...)	Nevhodné pro protokoly – simuluje jen síťové podmínky
Licence	Open Source (GNU GPLv2)	Freeware, open source komponenty	Open Source (Academic/Public)	Freeware (Cisco Networking Academy)	Součást Linuxu (GPL)

Tabulka 3: Diference mezi SW

(zdroj: [8])

8 POPIS SMĚROVÁNÍ

Směrování (anglicky routing) je proces, při kterém se v počítačových sítích určuje nejvhodnější cesta pro doručení datagramu (nebo paketu) od odesílatele k adresátovi. Tento proces neprobíhá pouze na úrovni specializovaných zařízení, jako jsou routery, ale účastní se ho také koncové stanice při odesílání dat. Směrovací tabulka (routing table) představuje klíčový prvek pro rozhodování o směru, kterým bude datový paket v síti dále odeslán. Tato tabulka obsahuje souhrn informací potřebných pro směrování a je uložena a využívána v aktivních síťových prvcích, jako jsou routery, ale také může být součástí operačních systémů koncových zařízení. Každý záznam ve směrovací tabulce je organizován jako samostatný řádek, který obsahuje údaje o cílové síti, masce podsítě, směru (next hop) a výstupním rozhraní, kterým má být paket dále odeslán. Záznamy v tabulce jsou zpravidla seřazeny podle specifity – od nejkonkrétnějších (s nejdelší maskou podsítě), které odpovídají přesně dané síti, až po nejobecnější. Na konci tabulky se obvykle nachází tzv. implicitní (výchozí) záznam, označovaný jako default gateway, který se používá v případech, kdy žádný z předchozích záznamů neodpovídá cílové IP adrese. Tento záznam odkazuje na nejbližší router, přes který je zajištěn přístup do vnějších sítí, například na internet. [27]

8.1 Statické směrování

Při statickém (neadaptivním) směrování nejsou záznamy ve směrovací tabulce během provozu zařízení automaticky aktualizovány. Tyto záznamy do tabulky obvykle vkládá správce systému, a to na základě informací poskytnutých správcem dané počítačové sítě, případně mohou být načteny při spuštění zařízení. Uchovávají se buď v konfiguračních souborech, nebo – v případě operačního systému Microsoft Windows – v registrech. Statické směrování je typické pro koncová zařízení nebo routery v menších sítích (např. LAN), kde se směrovací tabulka během provozu často nemění, a její konfigurace je jednoduchá. Příkladem konfigurace statické trasy v CISCO IOS je následující příklad:

```
ip route 192.168.2.0 255.255.255.0 10.0.0.2
```

Tento příkaz říká routeru, že síť je dosažitelná přes další router s IP adresou 10.0.0.2. Pokud chce router doručit data do této cílové sítě, odešle je na zadanou IP adresu jako výchozí bránu. [27]

8.2 Dynamické směrování

Dynamické směrování je způsob, při kterém jsou záznamy ve směrovací tabulce automaticky aktualizovány pomocí speciálních směrovacích protokolů. Tato metoda je v kontrastu ke statickému směrování, kde jsou trasy zadávány ručně. Dynamické směrování umožňuje síťovým zařízením, jako jsou routery, vzájemně si vyměňovat informace o topologii sítě a na základě těchto údajů určovat nejlepší cestu pro přenos datových paketů. V dynamickém směrování se routery automaticky přizpůsobují změnám v síti – například při výpadku spoje, přidání nové trasy nebo změně metriky určité linky. Každý router periodicky rozesílá směrovací informace do sítě, přijímá je od sousedních routerů a aktualizuje svou směrovací tabulku podle aktuální situace. Tento proces zajišťuje flexibilitu, škálovatelnost a odolnost vůči poruchám. Dynamické směrování je realizováno prostřednictvím různých směrovacích protokolů, které se liší podle způsobu fungování i použití. Tyto protokoly lze rozdělit do dvou hlavních kategorií. [28]

8.3 Distance-vector

Routery využívající distance-vector směrovací protokoly si udržují směrovací tabulku, která obsahuje informace o dosažitelnosti jednotlivých sítí a vzdálenosti k nim, přičemž vzdálenost je vyjádřena pomocí určité metriky. Nejčastěji používanou metrikou je počet skoků (hops), tedy počet mezilehlých routerů, kterými musí paket projít na cestě do cílové sítě. Každý router v pravidelných intervalech odesílá svou směrovací tabulku přímo připojeným sousedním routerům. Ti tyto informace přijmou, porovnají je se svou vlastní tabulkou a případně upraví směrovací záznamy, pokud nově získaná informace představuje výhodnější trasu. Jednoduchým příkladem distance-vector protokolu je RIP. [29]

8.3.1 RIP (Routing Information Protocol)

RIP (Routing Information Protocol) je směrovací protokol, který využívá metodu distance-vector, tedy směrování na základě vzdálenosti k cílové síti. V tomto protokolu je vzdálenost měřena pomocí počtu hopů (mezičlánků nebo směrovačů), přičemž maximální počet povolených hopů je omezen na 15. Hodnota 16 je považována za nedosažitelnou, čímž se efektivně zamezuje vzniku smyček ve směrování, které by mohly způsobit zahlcení sítě nebo nekonečné preposílání paketů. Výhody RIPu spočívají v jednoduchosti (vhodný pro použití v sítích s malým počtem uzlů), nízké spotřebě zdrojů (ideální protokol pro prostředí

s nízkým výpočetním výkonem) a kompatibility (podpora od skoro všech síťových zařízení). Příklad konfigurace RIP: [25][26]

```
router rip
version 2
network 192.168.1.0
network 10.0.0.0
```

Ověření funkčnosti:

```
show ip route rip
```

8.4 Link-state

Směrování podle stavu linky představuje druhou hlavní kategorii směrovacích protokolů. Na rozdíl od směrování založeného na vektoru vzdálenosti, kde routery využívají distribuovaný algoritmus a sdílejí své směrovací tabulky pouze se sousedy, u link-state protokolů si každý router postupně vytváří vlastní mapu celé topologie sítě. Tuto mapu získává na základě výměny speciálních zpráv (Link State Advertisements – LSA) se všemi ostatními routery v dané oblasti. Na základě takto získaných informací si každý router sám spočítá nejefektivnější cesty k cílovým sítím pomocí algoritmu pro výpočet nejkratší cesty – typický Dijkstrova algoritmu. Výsledkem je vlastní směrovací tabulka, která odpovídá aktuálnímu stavu celé sítě. Směrování podle stavu linky lze charakterizovat třemi klíčovými principy:

1. Znalost okolí:

Místo posílání celé směrovací tabulky sdílí router pouze informace o přímo připojených spojích – tedy identitu sousedních uzlů a metriky těchto spojení (např. propustnost nebo zpoždění).

2. Flooding:

Informace o stavu spojů nejsou odesílány jen sousedům, ale jsou šířeny napříč celou sítí. Každý router, který tyto informace přijme, je předává dál všem svým sousedům. Tento proces, nazývaný flooding, zajišťuje, že všechny routery získají stejný přehled o síti.

3. Reakce na změny:

Routery posílají nové informace pouze v případě, že dojde ke změně v síťové topologii – například při výpadku linky nebo připojení nového zařízení. Díky tomu je přenos směrovacích dat efektivní a nezatěžuje síť zbytečnými aktualizacemi. Typickým zástupcem této skupiny je protokol OSPF. [30]

8.4.1 OSPF (Open Shortest Path First)

Je dynamický směrovací protokol patřící do skupiny link-state protokolů. Využívá informace o stavu jednotlivých síťových spojení (linků), které jsou získávány a sdíleny mezi routery v síti, aby mohl vytvořit kompletní topologii sítě. Na základě této topologie si každý router vypočítá nejvhodnější směrovací cesty pomocí Dijkstrova algoritmu, který generuje strom nejkratších cest ke všem známým cílům. Jednou z hlavních výhod OSPF je jeho schopnost automaticky detekovat změny v síťové topologii a rychle na ně reagovat. V sítích s broadcast doménou využívá k přenosu směrovacích informací multicastové adresování, čímž snižuje zátěž na síť a zvyšuje efektivitu přenosu. OSPF je zařazen do skupiny Interior Gateway Protocols (IGP), což znamená, že je určen pro směrování uvnitř autonomního systému, typicky v rámci podnikové nebo poskytovatelské sítě. Routery používající OSPF spolupracují na určení optimální cesty pro přenos datových paketů v síti tvořené více propojenými podsítěmi. Hlavní nevýhodou OSPF oproti jednodušším protokolům, jako je RIP, je jeho vyšší komplexnost. Nastavení a správa OSPF vyžaduje hlubší znalosti síťové problematiky, avšak tato složitost je vyvážena větší flexibilitou, rychlostí konvergence a přesností směrování. [25][26]

Příklad konfigurace OSPF:

```
R1(config)# router ospf 1
R1(config-router)# network 192.168.1.0 0.0.0.255 area 0
R1(config-router)# network 10.0.0.0 0.0.0.3 area 0
```

Ověření konfigurace:

```
show ip ospf neighbor
show ip protocols
```

II PRAKTICKÁ ČÁST

9 TESTOVÁNÍ GNS3

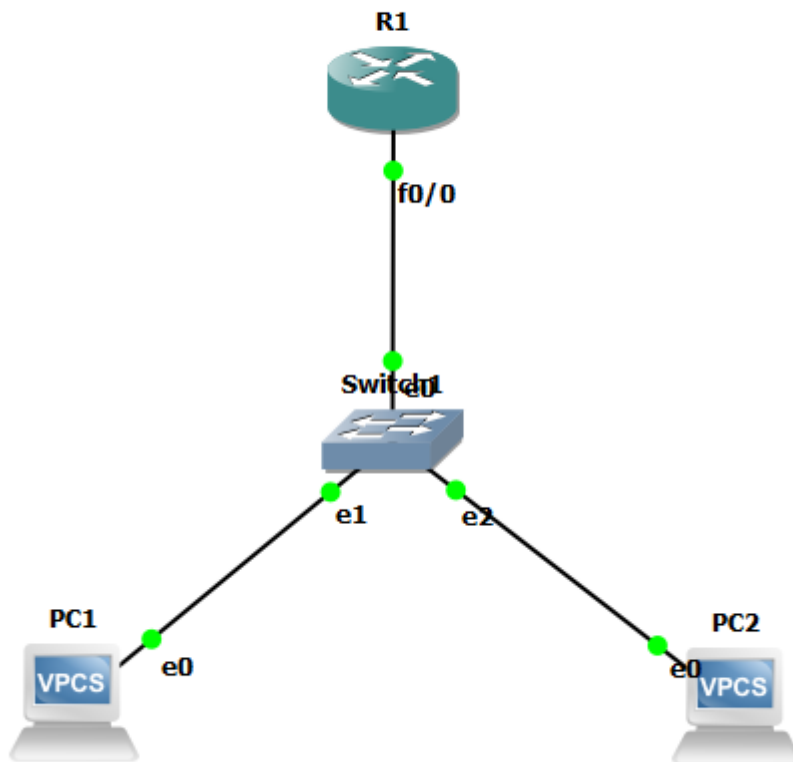
9.1 Základní síťová topologie

V rámci praktické části byla vytvořena jednoduchá síťová topologie v prostředí GNS3, složená z jednoho routeru, jednoho přepínače a dvou klientských zařízení. Cílem bylo demonstrovat základní síťové propojení v rámci jedné podsítě.

Po dokončení konfigurace všech zařízení byla základní funkčnost sítě otestována. Pomocí příkazů ping bylo ověřeno, že obě klientská zařízení (PC1 a PC2) jsou schopna vzájemné komunikace, což znamená, že přepínač (switch) úspěšně přeposílá ethernetové rámce mezi jednotlivými porty.

Dále byla otestována komunikace mezi klienty a routerem – konkrétně na IP adresu 192.168.1.1, která byla přiřazena rozhraní FastEthernet0/0 na routeru. Z obou zařízení byl router úspěšně dosažen, čímž byla potvrzena správně nastavená výchozí brána (default gateway) a skutečnost, že router aktivně přijímá a zpracovává provoz. Router zde byl použit především jako testovací výchozí brána pro demonstraci komunikace mezi zařízeními na třetí vrstvě.

Tento příklad byl použit jako stavební kámen pro složitější konfigurace, ve kterých budou dále využívány směrování, NAT či dynamické směrovací protokoly jako OSPF nebo RIP. Znalosti získané z této jednoduché laboratorní úlohy byly identifikovány jako nezbytný základ pro pochopení komplexnějších síťových návrhů, se kterými se lze setkat v praxi.



Obrázek 5: Základní síťová topologie

(zdroj: vlastní)

9.2 Jednoduchá síťová topologie

Jako 2. příklad v rámci praktické části byla navržena a nakonfigurována jednoduchá počítačová síť, která umožňuje komunikaci mezi dvěma oddělenými lokálními sítěmi.

Topologie obsahuje dva routery (R1 a R2), které slouží ke směrování dat mezi sítěmi, dále dva přepínače (Switch1 a Switch2) pro zajištění komunikace uvnitř lokálních sítí a čtyři koncové stanice (PC1 až PC4), které reprezentují uživatelská zařízení. Zařízení byla rozdělena do dvou samostatných LAN, přičemž každá LAN byla připojena k jednomu přepínači, který byl následně propojen s routerem. Routery byly mezi sebou propojeny přímým ethernetovým spojením, které sloužilo jako WAN (Wide Area Network).

Pro správnou funkčnost celé sítě bylo nezbytné nejprve stanovit IP adresaci. Sít' byla rozdělena do tří samostatných podsítí. První podsít' (192.168.1.0/24) sloužila pro levou LAN, kde se nacházely PC1 a PC2. Druhá podsít' (192.168.2.0/24) sloužila pro pravou LAN s PC3 a PC4. Třetí podsít' (10.0.0.0/30) byla určena pro spojení mezi routery. Každému zařízení v síti byla přiřazena statická IP adresa a výchozí brána směřující na příslušný router. Po správném fyzickém propojení byla provedena konfigurace jednotlivých zařízení. Na koncových stanicích (PC1–PC4) byly nastaveny IP adresy a výchozí brány. Například PC1 dostal adresu 192.168.1.2 s maskou 255.255.255.0 a bránou 192.168.1.1. Na routerech byly nakonfigurovány IP adresy na příslušných rozhraních a statické směrování, které umožnilo přenos dat mezi sítěmi. Konfigurace vypadá následovně:

```
conf t
hostname R1

interface f0/0
 ip address 192.168.1.1 255.255.255.0
 no shutdown

interface f0/1
 ip address 10.0.0.1 255.255.255.252
 no shutdown

ip route 192.168.2.0 255.255.255.0 10.0.0.2
end

conf t
hostname R2

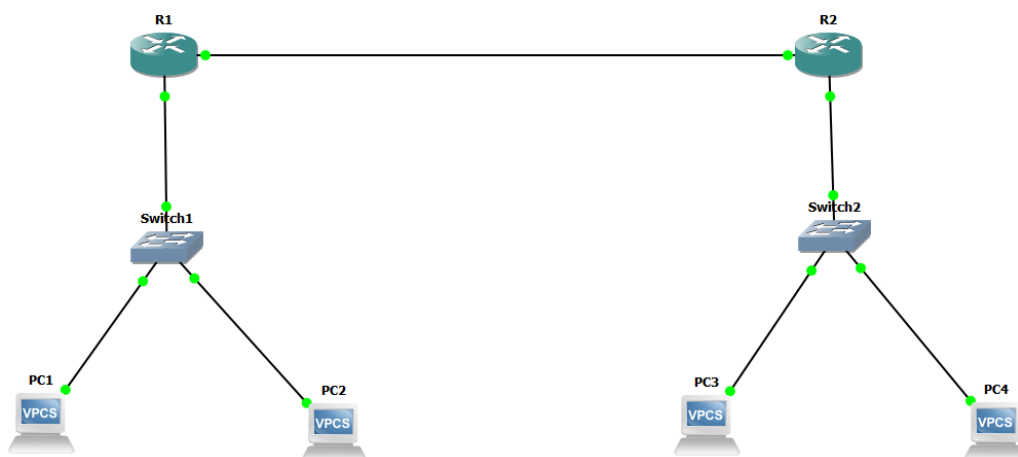
interface f0/0
 ip address 192.168.2.1 255.255.255.0
 no shutdown

interface f0/1
 ip address 10.0.0.2 255.255.255.252
 no shutdown
ip route 192.168.1.0 255.255.255.0 10.0.0.1
end
```

Po dokončení konfigurace byla provedena zkouška konektivity pomocí příkazu ping, který ověřil, že zařízení z jedné LAN mohou komunikovat se zařízeními v druhé LAN. Například při testu z PC1 na PC3 byla zaznamenána úspěšná odpověď, což potvrdilo správné nastavení IP adresace i směrování. Úspěšná odpověď vypadá takto:

```
PC1> ping 192.168.2.2
84 bytes from 192.168.2.2 icmp_seq=1 ttl=62 time=75.833 ms
84 bytes from 192.168.2.2 icmp_seq=2 ttl=62 time=60.213 ms
84 bytes from 192.168.2.2 icmp_seq=3 ttl=62 time=59.973 ms
84 bytes from 192.168.2.2 icmp_seq=4 ttl=62 time=60.127 ms
84 bytes from 192.168.2.2 icmp_seq=5 ttl=62 time=60.294 ms
```

Výsledná síť umožňuje plnou komunikaci mezi zařízeními ve dvou oddělených LAN sítích přes směrovací infrastrukturu, čímž naplňuje cíl cvičení.



Obrázek 6: Jednoduchá síťová topologie

(zdroj: vlastní)

9.3 Topologie s využitím RIP

Další příklad v prostředí GNS3 byla navržena a realizována síťová topologie složená ze tří routerů (R1, R2, R3), tří přepínačů (Switch1, Switch2, Switch3) a koncových zařízení (PC1, PC2, PC3). Každý router byl propojen se sousedním routerem, jednak s jedním přepínačem, ke kterému bylo připojeno jedno koncové zařízení. Vznikla tak jednoduchá lineární topologie, ve které bylo možné otestovat směrovací protokol RIP. Nejprve byly nastaveny IP adresy všech koncových zařízení.

PC1:

```
ip 192.168.1.2 255.255.255.0 192.168.1.1
```

PC2:

```
ip 192.168.2.2 255.255.255.0 192.168.2.1
```

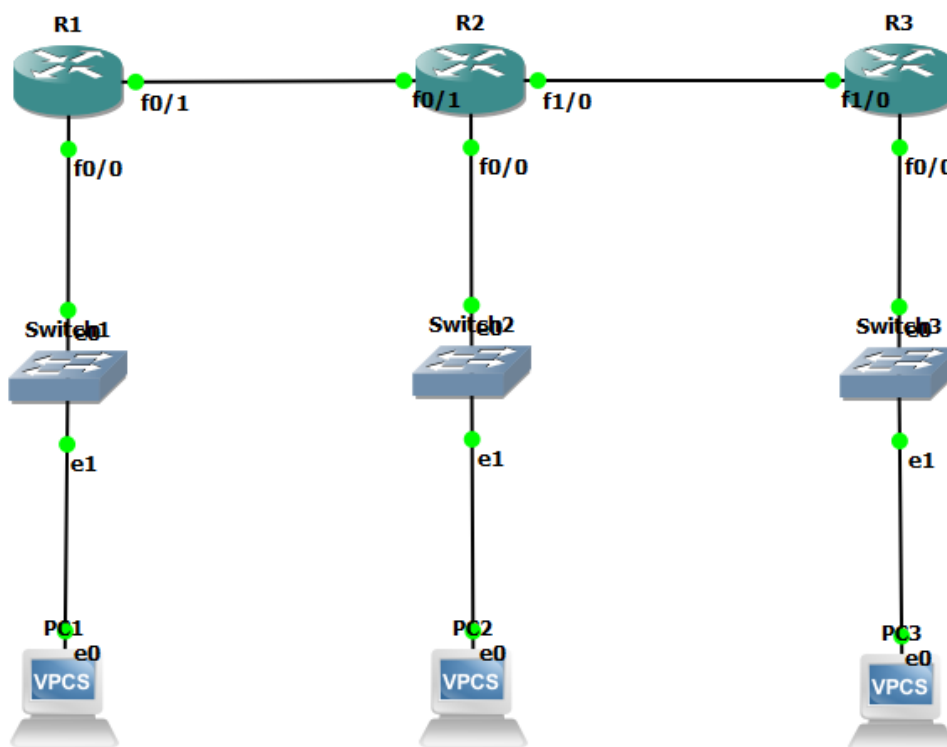
PC3:

```
ip 192.168.3.2 255.255.255.0 192.168.3.1
```

Po nastavení IP adres u jednotlivých koncových zařízení na každém směrovači byly pomocí příkazů rozhraní aktivována a nakonfigurována. Například na R1 bylo na rozhraní FastEthernet0/0 nastaveno `ip address 192.168.1.1 255.255.255.0` a na rozhraní FastEthernet0/1 `ip address 10.0.12.1 255.255.255.0`. Obě rozhraní byla zapnuta příkazem `no shutdown`. Pro zajištění směrování mezi jednotlivými sítěmi byl na všech směrovačích aktivován směrovací protokol RIP ve verzi 2. Příkazem `router rip` se vstoupilo do konfiguračního režimu RIP, kde byly zadány příslušné sítě pomocí `network`. Například na R1 se zadaly `network 192.168.1.0` a `network 10.0.12.0`. Pro zajištění správného fungování RIP verze 2 byl přidán příkaz `version 2` a `no auto-summary`, aby se zabránilo automatickému sumarizování tras. Konfigurace routerů viz. Příloha 1

Router R2 má nejdelší konfiguraci ze všech tří směrovačů, protože plní roli centrálního uzlu v topologii. Je propojen jak s R1, tak s R3, čímž zajišťuje přeposílání dat mezi dvěma dalšími směrovači a zároveň se stará o svou vlastní lokální síť (např. podsít' s PC2). Vzhledem ke svému postavení ve struktuře musí R2 konfigurovat dvě směrovací rozhraní směrem ke svým sousedům (např. Fa0/1 a Fa1/0) a jedno rozhraní pro připojení ke své lokální síti (např. Fa0/0).

Po dokončení konfigurace byla funkčnost celé sítě ověřena pomocí příkazu ping mezi jednotlivými koncovými zařízeními. Ze stanice PC1 byl otestován přístup k PC3 zadáním příkazu ping 192.168.3.1, přičemž úspěšná odpověď potvrdila správné směrování napříč všemi třemi směrovači. Pro další kontrolu byly na směrovačích zobrazeny směrovací tabulky pomocí příkazu show ip route, kde bylo vidět, že směrovače si předaly informace o sítích pomocí RIP a jsou schopny směrovat pakety do všech částí topologie.



Obrázek 7: Topologie s využitím RIP

(zdroj: vlastní)

9.4 Topologie s využitím OSPF

Pro demonstraci dynamického směrovacího protokolu OSPF byla navržena topologie, která obsahuje čtyři směrovače (R1, R2, R3, R4), čtyři přepínače (Switch1, Switch2, Switch3, Switch4) a čtyři koncová zařízení (PC1, PC2, PC3, PC4), které slouží k ověření konektivity. Pro zajištění funkce komunikace v síti a pro implementaci směrovacího protokolu OSPF byl pečlivě navržen plán IP adresace. Adresy byly rozděleny do několika skupin: point-to-point spojení mezi směrovači a sítě pro koncová zařízení. Pro konzistentnost a identifikaci směrovačů v rámci OSPF sítě byly také přiděleny Loopback adresy, které slouží jako Router ID.

Každé rozhraní bylo ručně nakonfigurováno IP adresou a aktivováno příkazem no shutdown. Všechna WAN rozhraní mezi směrovači používají síť /30 (255.255.255.252) pro úsporu IP adres a LAN rozhraní používají síť /24 pro připojené koncové zařízení.

Například:

- Spojení mezi R1 a R2 má síť 10.0.12.0/30.
- Spojení mezi R1 a R3 má síť 10.0.13.0/30.
- Spojení mezi R2 a R4 má síť 10.0.24.0/30.
- Spojení mezi R3 a R4 má síť 10.0.34.0/30.
- LAN sítě jsou 192.168.1.0/24 až 192.168.4.0/24.

Pro každý jednotlivý směrovač bylo ručně nastaveno jedinečné OSPF Router ID pomocí smyčkového rozhraní (Loopback). Každý router má vlastní loopback IP adresu, která je použita jako identifikátor v rámci OSPF protokolu. Konkrétně:

- Směrovač R1 má Router ID 1.1.1.1
- Směrovač R2 má Router ID 2.2.2.2
- Směrovač R3 má Router ID 3.3.3.3
- Směrovač R4 má Router ID 4.4.4.4

Tímto způsobem je zajištěna jednoznačná identifikace každého routeru v OSPF topologii a usnadňuje to diagnostiku i správu sítě.

Následující kroky detailně popisují konfiguraci jednotlivých zařízení pro dosažení funkční sítě a OSPF. Konfigurace routerů viz. Příloha 2.

V konfiguraci routerů příkaz `write memory` znamená, že se zajistí aktuálnost a funkčnost konfigurace, která se uloží do NVRAM a bude načtena i po restartu. Následně musely být nastaveny IP adresy všem koncovým zařízením.

Pro PC1 byla nastavena IP adresa:

```
ip 192.168.1.10 255.255.255.0 192.168.1.1
```

PC2:

```
ip 192.168.2.10 255.255.255.0 192.168.2.1
```

PC3:

```
ip 192.168.3.10 255.255.255.0 192.168.3.1
```

PC4:

```
ip 192.168.4.10 255.255.255.0 192.168.4.1
```

Po dokončení konfigurace všech zařízení je klíčové ověřit správnou funkčnost sítě a OSPF směrování. Na každém routeru se zadá příkaz `show ip ospf neighbor`. Očekávaný výstup by měl vypadat následovně:

```
R1#show ip ospf neighbor
```

Neighbor ID	Pri	State	Dead Time	Address	Interface
2.2.2.2	1	FULL/DR		00:00:31	10.0.12.2
FastEthernet0/1					
3.3.3.3	1	FULL/DR		00:00:34	10.0.13.2
FastEthernet1/0					

Nejdůležitější je ověřit, zda koncová zařízení (PC) mohou komunikovat napříč sítí. Pro ověření funkčnosti se používá příkaz `ping`. Například z PC1 do ostatních PC(2-4):

```
PC1> ping 192.168.2.10
```

```
PC1> ping 192.168.3.10
```

```
PC1> ping 192.168.4.10
```

Jestliže bylo vše nakonfigurováno správně výstup po zadání příkazu `ping` by měl vypadat následovně:

```
PC3> ping 192.168.1.10
```

```
84 bytes from 192.168.1.10 icmp_seq=1 ttl=62 time=60.045 ms
```

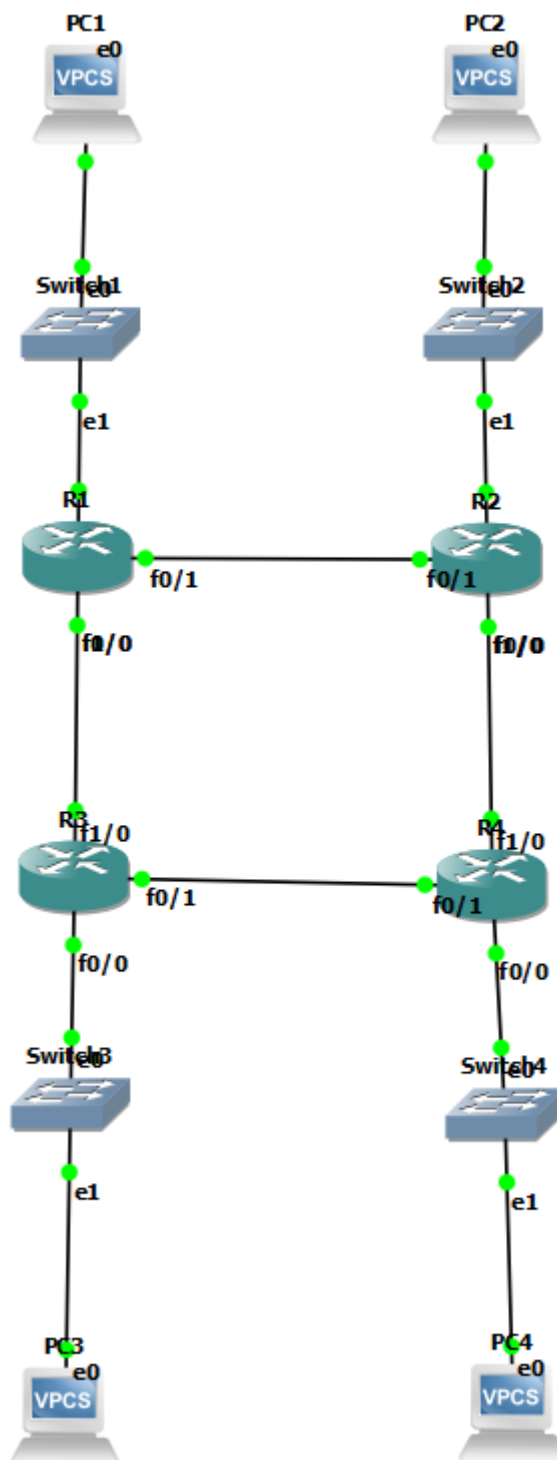
```
84 bytes from 192.168.1.10 icmp_seq=2 ttl=62 time=60.199 ms
```

```
84 bytes from 192.168.1.10 icmp_seq=3 ttl=62 time=60.094 ms
```

```
84 bytes from 192.168.1.10 icmp_seq=4 ttl=62 time=60.140 ms
```

```
84 bytes from 192.168.1.10 icmp_seq=5 ttl=62 time=60.119 ms
```

Úspěšné navázání OSPF a komunikace mezi koncovými zařízeními potvrzuje správnou implementaci a funkčnost.



S

Obrázek 8: Topologie s využitím OSPF

(zdroj: vlastní)

10 TESTOVÁNÍ CISCO PACKET TRACER

10.1 Jednoduchá síťová topologie

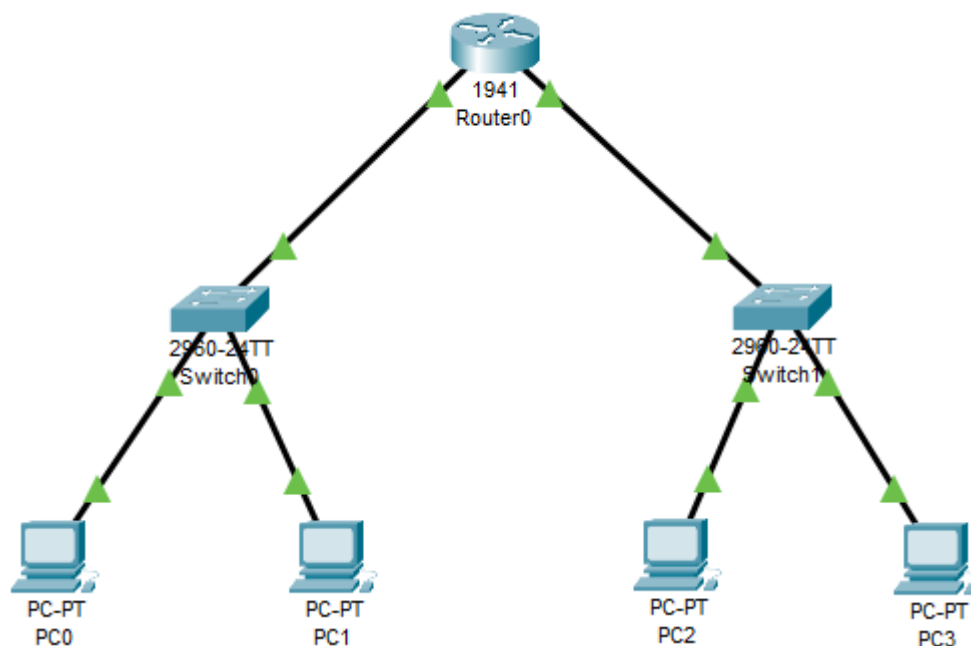
V rámci praktické části byl v prostředí Cisco Packet Tracer navržen a nakonfigurován jednoduchý síťový model. Cílem bylo vytvořit síťovou topologii s jedním centrálním směrovačem, dvěma přepínači a čtyřmi koncovými zařízeními, a následně zajistit jejich vzájemnou konektivitu. Síť slouží jako demonstrační model pro základní přepínání a směrování mezi dvěma oddělenými podsítěmi. Topologie se tedy skládá z jednoho routeru typu Cisco 1941, dvou přepínačů typu 2960-24TT a ke každému přepínači byla připojena dvě koncová zařízení (PC0-PC3). Přepínače byly připojeny ke dvěma dostupným portům směrovače, konkrétně GigabitEthernet0/0 a GigabitEthernet0/1 pomocí kabelu typu Copper Straight-Through. Poté byly nastaveny IP adresy koncových zařízení. Síť je rozdělena do dvou oddělených podsítí:

- Podsít' A (192.168.1.0/24) – připojená přes Switch0, obsahuje PC0 a PC1.
- Podsít' B (192.168.2.0/24) – připojená přes Switch1, obsahuje PC2 a PC3.

Směrovač byl nakonfigurován následovně:

```
enable
configure terminal
hostname Router0
interface GigabitEthernet0/0
 ip address 192.168.1.1 255.255.255.0
 no shutdown
interface GigabitEthernet0/1
 ip address 192.168.2.1 255.255.255.0
 no shutdown
exit
```

Správnost konfigurace byla ověřena prostřednictvím nástroje 'Add Simple PDU', který je součástí simulačního prostředí Cisco Packet Tracer. Slouží ke stejnému účelu jako příkaz ping, tedy k testování dosažitelnosti cílového zařízení.



Obrázek 9: Jednoduchá topologie v Cisco Packet Tracer

(zdroj: vlastní)

10.2 Topologie s využitím statického směrování

Druhý příklad v rámci testování Cisco Packet Traceru se zaměřil na implementaci statického směrování v rozsáhlejší topologii. Vytvořená síťová architektura sestávala ze tří směrovačů (Router0, Router1 a Router2), které byly vzájemně redundantně propojeny sériovými spoji. Ke každému směrovači byl připojen jeden přepínač (Switch0, Switch1 a Switch2), a ke každému přepínači následně dvě koncová zařízení (PC0–PC5). Tímto způsobem byly vytvořeny tři oddělené lokální podsítě (LAN).

V prvním kroku byla provedena konfigurace koncových zařízení. U každého počítače byla ručně nastavena IP adresa a výchozí brána. Pro podsít' Routeru0 (Switch0, PC0, PC1) byly použity IP adresy:

- PC0: 192.168.10.1 /24
- PC1: 192.168.10.2 /24
- Výchozí brána pro PC0 a PC1: 192.168.10.10

Podobným způsobem byly nakonfigurovány i ostatní koncová zařízení, vždy v souladu s příslušnou podsítí:

Pro podsít' Routeru1 (Switch1, PC2, PC3):

- PC2: 192.168.11.1 /24
- PC3: 192.168.11.2 /24
- Výchozí brána pro PC2 a PC3: 192.168.11.10

Pro podsít' Routeru2 (Switch2, PC4, PC5):

- PC4: 192.168.12.1 /24
- PC5: 192.168.12.2 /24
- Výchozí brána pro PC4 a PC5: 192.168.12.10

Po nastavení IP adres na všech koncových zařízeních následovala konfigurace směrovačů, aby mohla probíhat komunikace s jejich příslušnými podsítěmi i mezi jednotlivými směrovači navzájem.

U sériových rozhraní bylo na jedné straně každého spojení (typicky na straně DCE (Direct Code Execution), pokud to Packet Tracer vyžadoval) nastaveno clock rate pro synchronizaci přenosu dat.

Klíčovým krokem pro realizaci statického směrování bylo ruční zadání směrovacích záznamů do směrovacích tabulek každého routeru. Každý router musel mít definované cesty k vzdáleným sítím, aby věděl, kam předat pakety směřující do dané podsítě.

Tabulka 4: Konfigurace routerů

(zdroj: vlastní)

Router	Rozhraní	IP adresa	Maska podsítě	Popis / Účel
R0	FastEthernet0/0	192.168.10.10	255.255.255.0	LAN (192.168.10.0/24)
	Serial2/0	10.0.0.1	255.255.255.252	Propojení Router1
	Serial3/0	12.0.0.1	255.255.255.252	Propojení Router2
R1	FastEthernet0/0	192.168.11.10	255.255.255.0	LAN (192.168.11.0/24)

	Serial2/0	10.0.0.2	255.255.255.252	Propojení Router0
	Serial3/0	11.0.0.2	255.255.255.252	Propojení Router2
R2	FastEthernet0/0	192.168.12.10	255.255.255.0	LAN (192.168.12.0/24)
	Serial2/0	12.0.0.2	255.255.255.252	Propojení Router0
	Serial3/0	11.0.0.1	255.255.255.252	Propojení Router1

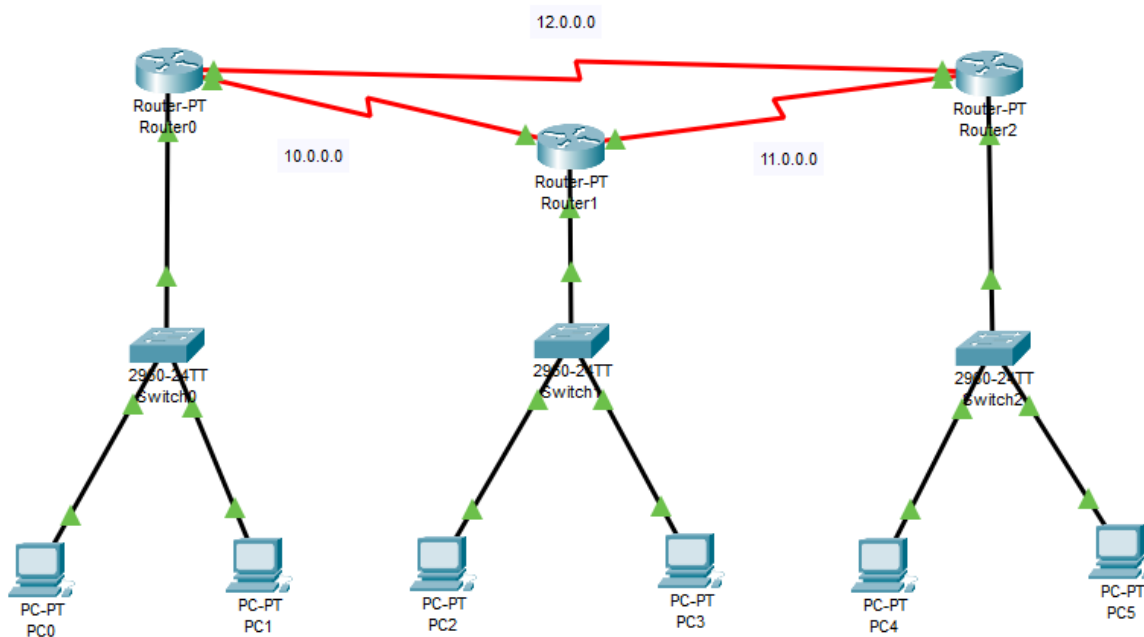
Tabulka 5: Konfigurace statického směrování

(zdroj: vlastní)

Router	Cílová síť	Maska podsítě	Následující hop	Popis/cesta
R0	192.168.11.0	255.255.255.0	10.0.0.2	Přes Router1
	192.168.12.0	255.255.255.0	12.0.0.2	Přes Router2 (Primární)
	192.168.12.0	255.255.255.0	10.0.0.2	Přes Router1 -> Router2 (záložní)
R1	192.168.10.0	255.255.255.0	10.0.0.2	Přes Router0 (primární)
	192.168.10.0	255.255.255.0	11.0.0.1	Přes Router2 -> Router0 (záložní)
	192.168.12.0	255.255.255.0	11.0.0.1	Přes Router2
R2	192.168.10.0	255.255.255.0	12.0.0.1	Přes Router0 (primární)

	192.168.10.0	255.255.255.0	11.0.0.2	Přes Router1 -> Router0 (záložní)
	192.168.11.0	255.255.255.0	11.0.0.2	Přes Router1

Po dokončení konfigurace všech IP adres a statických směrovacích záznamů byla provedena verifikace konektivity, která byla opět ověřena pomocí nástroje 'Add Simple PDU'. Ověření proběhlo způsobem poslání paketu od zdrojového zařízení např. z PC0 do cílového zařízení např. PC4. Touto metodou bylo ověřeno každé koncové zařízení s cílem zajistit správnost implementované topologie.



Obrázek 10: Cisco Static Routing

(zdroj: vlastní)

10.3 Topologie dynamického směrování s využitím RIP

Nejprve je v prostředí Cisco Packet Tracer vizuálně navržena a sestavena síťová topologie. Tato topologie je tvořena třemi routery (R0, R1, R2), které jsou vzájemně propojeny pomocí gigabitových ethernetových rozhraní. Routery jsou dále připojeny ke dvěma switchům, které slouží k propojení koncových uživatelských zařízení, jako jsou počítače (PC0-PC5). V rámci topologie jsou definovány konkrétní síťové adresy pro jednotlivé segmenty: dvě lokální sítě (LAN) s adresami 192.168.10.0/24 a 192.168.20.0/24 a dvě mezisměrovačové sítě (WAN) s adresami 10.10.10.0/30 a 20.20.20.0/30. Pro vzájemné propojení routerů jsou preferovány gigabitové ethernetové porty (např. Gig0/0 a Gig0/1).

Po fyzickém sestavení topologie je přistoupeno ke konfiguraci IP adres na všech aktivních rozhraních routerů a koncových zařízeních. Každé rozhraní routeru je individuálně konfigurováno s IP adresou a maskou podsítě odpovídající jeho připojené síti. Konfigurace IP adres na jednotlivých rozhraních routerů se provádí v grafickém uživatelském rozhraní Packet Traceru, konkrétně v záložce 'Config'. Po výběru příslušného rozhraní z levého panelu (např. GigabitEthernet0/0 nebo FastEthernet0/0) se do pole 'IPv4 Address' zadá požadovaná IP adresa a do pole 'Subnet Mask' odpovídající maska podsítě. Je také nezbytné zaškrtnout políčko 'On' vedle 'Port Status', čímž se rozhraní aktivuje (ekvivalent příkazu `no shutdown` v CLI). Například rozhraní Gig0/0 na Routeru 0, které je připojeno k síti 10.10.10.0/30, je opatřeno IP adresou 10.10.10.1 s maskou podsítě 255.255.255.252. Obdobně, rozhraní Gig0/1 na stejném routeru, připojené k LAN 192.168.10.0/24, je konfigurováno s IP adresou 192.168.10.1 a maskou podsítě 255.255.255.0. Podobným způsobem jsou konfigurována všechna rozhraní na Routeru 1 (Gig0/0: 10.10.10.2/30, Gig0/1: 20.20.20.1/30) a Routeru 2 (Gig0/0: 20.20.20.2/30, Gig0/1: 192.168.20.1/24). Zároveň je nezbytné konfigurovat IP adresy a výchozí brány na koncových zařízeních (PC), přičemž výchozí brána je nastavena na IP adresu rozhraní routeru v dané lokální síti.

Nastavení IP adres v LAN 1:

- PC0: 192.168.10.10 (Gateway: 192.168.10.1)
- PC1: 192.168.10.20 (Gateway: 192.168.10.1)
- PC2: 192.168.10.30. (Gateway: 192.168.10.1)

Nastavení IP adres v LAN 2:

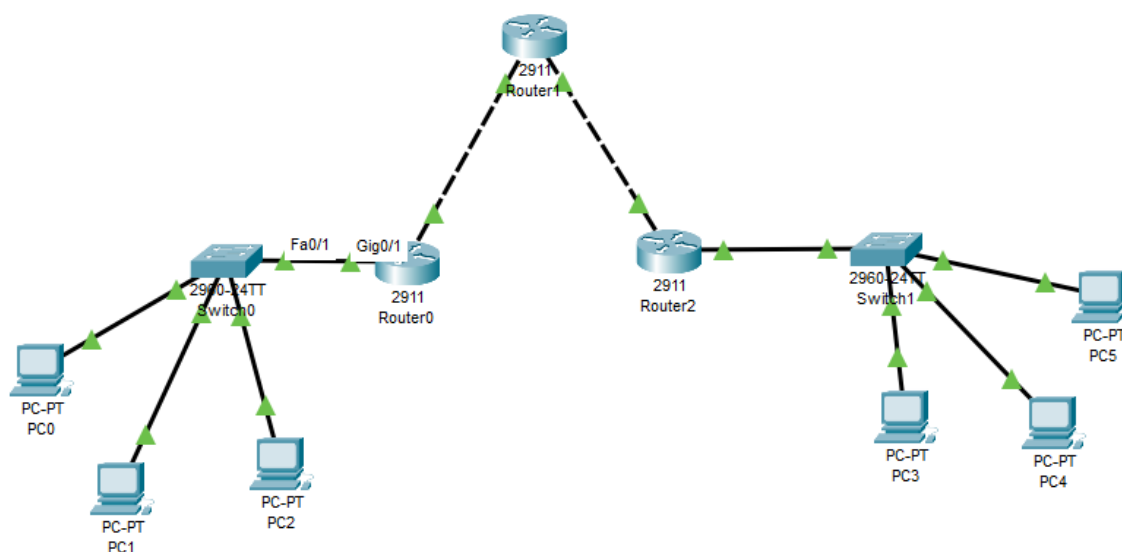
- PC3: 192.168.20.10 (Gateway: 192.168.20.1)

- PC4: 192.168.20.20 (Gateway: 192.168.20.1)
- PC5: 192.168.20.30 (Gateway: 192.168.20.1)

K zajištění dynamického směrování a výměny směrovacích informací mezi routery je na každém routeru konfigurován směrovací protokol RIP. Konfigurace routerů proběhla následovně:

```
Router>enable
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router rip
Router(config-router)#no auto-summary
Router(config-router)#version 2
Router(config-router)#network 192.168.10.0
Router(config-router)#network 10.10.10.0
Router(config-router)#exit
Router(config)#write memory
```

Ověření funkčnosti a konektivity sítě lze provést několika způsoby. V prostředí Cisco Packet Tracer je možné použít nástroj 'Add Simple PDU' pro simulaci odesílání paketů a vizuální sledování jejich cesty sítě. Alternativně lze ověřit konektivitu přímo z příkazového řádku počítačů (PC Command Prompt) pomocí příkazů ping pro testování dosažitelnosti vzdálených zařízení a tracer (nebo traceroute na routerech) pro zobrazení cesty, kterou paket urazí k cíli.



Obrázek 11: Topologie v Cisco Packet Tracer s využitím RIP

(zdroj: vlastní)

10.4 Topologie dynamické směrování s využitím OSPF

V této praktické části byla navržena, nakonfigurována a otestována komplexní síť využívající směrovací protokol OSPF (Open Shortest Path First) v simulovaném prostředí Cisco Packet Tracer. Hlavním cílem bylo ověřit správnou funkčnost OSPF, provést jeho konfiguraci a identifikovat případné problémy při implementaci s cílem zajistit plnou konektivitu v celé síti. Topologie se skládá ze čtyř směrovačů (Router0, Router1, Router2, Router3) propojené redundantními spoji (Ethernet a sériová linka) a šesti koncových zařízení (PC0-PC5) připojených k routerům přes přepínače. Rozdělení IP adres:

Tabulka 6: Propojovací síť (Router-Router)

(zdroj: vlastní)

Spojení	Rozhraní	Síť	IP (Router A)	IP (Router B)
R0 – R1	GigabitEthernet0/0	10.0.0.0/30	R0:10.0.0.1	R1:10.0.0.2
R1 – R3	GigabitEthernet1/0	10.0.0.4/30	R1:10.0.0.5	R3:10.0.0.6
R2 – R3	Serial0/0	10.0.0.8/30	R2:10.0.0.10	R3:10.0.0.9
R0 – R2	GigabitEthernet1/0	10.0.0.12/30	R0:10.0.0.14	R2:10.0.0.13
R0 – R1	FastEthernet2/0	10.0.0.16/30	R0:10.0.0.17	R1:10.0.0.21
R1 – R2	FastEthernet3/0	10.0.0.20/30	R1:10.0.0.21	R2:10.0.0.22

LAN síť (pro koncová zařízení):

- PC0 (přes R0 FastEthernet3/0): 192.168.1.0/24 (Gateway: 192.168.1.1)
- PC1 (přes R1 FastEthernet3/0): 192.168.2.0/24 (Gateway: 192.168.2.1)
- PC2 (přes R2 FastEthernet3/0): 192.168.10.0/25 (Gateway: 192.168.10.1)
- PC3 (přes R2 FastEthernet4/0): 192.168.10.128/25 (Gateway: 192.168.10.129)
- PC4 (přes R3 FastEthernet3/0): 172.16.0.0/22 (Gateway: 172.16.0.1)
- PC5 (přes R3 FastEthernet4/0): 172.16.4.0/23 (Gateway: 172.16.4.1)

Po dokončení návrhu adresního plánu byly příslušné IP adresy, masky podsítě a výchozí brány úspěšně nakonfigurovány na všech koncových zařízeních v souladu s jejich připojením ke konkrétním podsítím. Konfigurace koncových zařízení vypadala následovně:

Tabulka 7: LAN síť (koncová zařízení)

(zdroj: vlastní)

PC	IP adresa zařízení	Maska podsítě	Výchozí brána
PC0	192.168.1.10	255.255.255.0	192.168.1.1
PC1	192.168.2.10	255.255.255.0	192.168.2.1
PC2	192.168.10.10	255.255.255.128	192.168.10.1
PC3	192.168.10.130	255.255.255.128	192.168.10.129
PC4	172.16.0.10	255.255.252.0	172.16.0.1
PC5	172.16.4.10	255.255.252.0	172.16.4.1

Jakmile byly IP adresy a výchozí brány přiřazeny všem koncovým zařízením, přistoupilo se ke konfiguraci směrovačů s implementací směrovacího protokolu OSPF. Cílem bylo umožnit automatickou výměnu směrovacích informací a zajistit plnou konektivitu v rámci celé topologie. Jednotlivá konfigurace routerů viz. Příloha 3.

Po dokončení všech konfigurací byly provedeny následující kroky k ověření plné funkčnosti sítě. Nejdříve proběhlo ověření, že všechny indikační světla na propojeních svítí zeleně, což znamená aktivní spojení na fyzické a linkové vrstvě. Ověření stavu rozhraní na routerech proběhlo způsobem, že na každém routeru byl použit příkaz `show ip interface brief`. Očekávaným výsledkem bylo, že všechna nakonfigurovaná rozhraní měla status `up` a protokol `up`, což potvrzuje správnou aktivaci a přiřazení IP adres. Např. ověření Routeru2:

```
Router2#show ip interface brief
Interface IP-Address OK? Method Status Protocol
Serial0/0 10.0.0.10 YES manual up up
GigabitEthernet1/0 10.0.0.13 YES manual up up
FastEthernet2/0 10.0.0.22 YES manual up up
FastEthernet3/0 192.168.10.1 YES manual up up
FastEthernet4/0 192.168.10.129 YES manual up up
```

Ověření OSPF sousedství proběhlo pomocí příkazu `show ip ospf neighbor`, což nám ukáže, zda všechny routery navázaly sousedství se svými OSPF sousedy. Bylo také

potvrzeno, že Router ID sousedů odpovídají očekávaným hodnotám (1.1.1.1, 2.2.2.2, 3.3.3.3, 4.4.4.4). Např. ověření Routeru1:

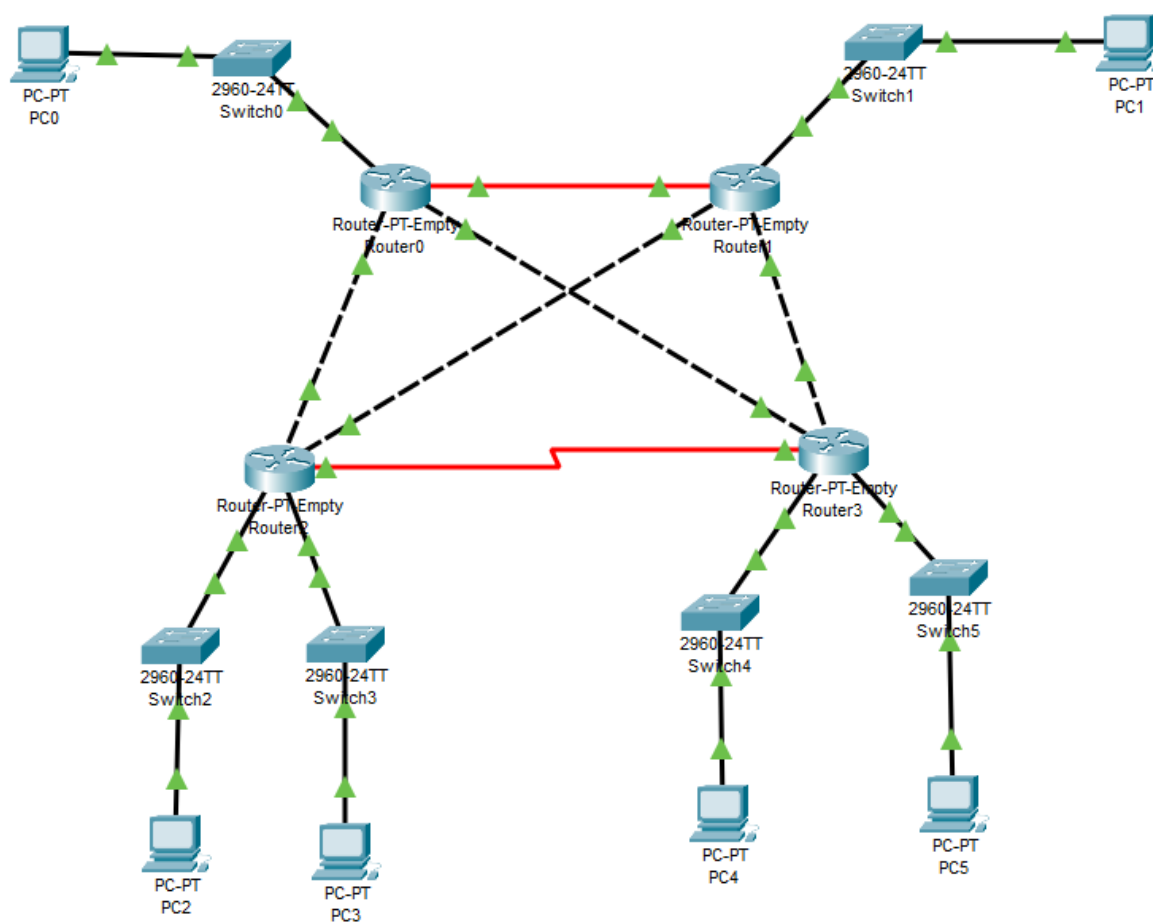
```
Router1#show ip ospf neighbor
```

```
Neighbor ID Pri State Dead Time Address Interface
1.1.1.1 1 FULL/BDR 00:00:31 10.0.0.1 GigabitEthernet0/0
3.3.3.3 1 FULL/DR 00:00:31 10.0.0.22 FastEthernet2/0
4.4.4.4 1 FULL/DR 00:00:31 10.0.0.6 GigabitEthernet1/0
```

Pomocí příkazu `show ip route` proběhlo ověření směrovacích tabulek, zda routery znají všechny sítě v topologii. Přímě spojené sítě (C), všechny vzdálené sítě naučené prostřednictvím OSPF (O). Např. ověření Routeru3:

```
Router3#show ip route
Gateway of last resort is not set
10.0.0.0/30 is subnetted, 6 subnets
O 10.0.0.0 [110/2] via 10.0.0.5, 00:09:05, GigabitEthernet1/0
C 10.0.0.4 is directly connected, GigabitEthernet1/0
C 10.0.0.8 is directly connected, Serial0/0
O 10.0.0.12 [110/3] via 10.0.0.5, 00:09:05, GigabitEthernet1/0
C 10.0.0.16 is directly connected, FastEthernet2/0
O 10.0.0.20 [110/2] via 10.0.0.5, 00:09:05, GigabitEthernet1/0
172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C 172.16.0.0/22 is directly connected, FastEthernet3/0
C 172.16.4.0/23 is directly connected, FastEthernet4/0
O 192.168.1.0/24 [110/3] via 10.0.0.5, 00:09:05, GigabitEthernet1/0
O 192.168.2.0/24 [110/2] via 10.0.0.5, 00:09:05, GigabitEthernet1/0
```

Jako poslední bylo provedeno testování konektivity mezi všemi koncovými zařízeními. Z každého počítače byl použit příkaz `ping` v CLI a bylo ověřeno spojení s ostatními zařízeními. Následně byl využit nástroj "Add Simple PDU" pro grafické ověření připojení. Veškeré testy byly úspěšně dokončeny a byla potvrzena bezproblémová komunikace mezi zařízeními.



Obrázek 12: Topologie dynamického směrování s využitím OSPF

(zdroj: vlastní)

11 SUBJEKTIVNÍ ZHODNOCENÍ

V rámci této bakalářské práce byla poprvé nanečisto vyzkoušena práce s počítačovými sítěmi a nástroji určenými k jejich simulaci a emulaci. Před zahájením práce byly k dispozici pouze základní teoretické znalosti, avšak v průběhu zpracování bylo nezbytné osvojit si praktické fungování sítí, jejich konfiguraci i vzájemné propojování jednotlivých zařízení.

Z pěti vybraných nástrojů byly nakonec důkladně využity pouze GNS3 a Cisco Packet Tracer. Nástroje ns-3 a NetEm, určené primárně pro prostředí Linux, nebyly vzhledem k technickým omezením využity, a prostředí OMNeT++ se nepodařilo úspěšně spustit. I přesto umožnila práce s těmito dvěma nástroji získat cenné poznatky o praktickém fungování sítí, identifikaci běžných problémů a rozdílech v přístupu ke konfiguraci v jednotlivých prostředích.

Cisco Packet Tracer byl vnímán jako nástroj s přehledným a uživatelsky přívětivým grafickým rozhraním. Všechna zařízení jsou zde připravena k okamžitému použití a ovládání je intuitivní, což z něj činí ideální nástroj pro začátečníky. Jeho využití je však omezeno – zejména absencí podpory pokročilejších funkcí, jako jsou komplexní směrovací protokoly nebo možnost práce s reálnými IOS obrazy. Z toho důvodu není vhodný pro složitější síťové scénáře.

Naopak GNS3 byl vnímán jako nástroj určený pro pokročilejší uživatele. Nabízí široké možnosti konfigurace a vysokou míru flexibility, zároveň však vyžaduje složitější přípravu. Před jeho použitím bylo nutné ručně stáhnout obrazy zařízení, jako jsou směrovače s IOS, a následně je do prostředí nástroje importovat a nakonfigurovat. GNS3 tedy nelze používat bez předchozí instalace a nastavení tak snadno jako Cisco Packet Tracer.

Závěrem lze uvést, že práce v Cisco Packet Traceru byla pohodlnější z hlediska přehlednosti a jednoduchosti, avšak po úspěšné konfiguraci GNS3 bylo zjištěno, že i tento nástroj je uživatelsky zvládnutelný. Přestože v GNS3 probíhá veškerá konfigurace výhradně prostřednictvím příkazové řádky, tato skutečnost přispěla k hlubšímu pochopení síťové konfigurace. Cisco Packet Tracer lze doporučit především studentům a začátečníkům pro osvojení základních principů a tvorbu jednoduchých topologií. GNS3 je naopak vhodnější pro pokročilejší testování, komplexní simulace a práci s reálnými konfiguracemi.

ZÁVĚR

Cílem této bakalářské práce bylo provedení rešerše a porovnání vybraných nástrojů pro simulaci a emulaci počítačových sítí. V teoretické části bylo analyzováno všech pět vybraných nástrojů. U každého z nich byla popsána architektura, možnosti použití, výhody i nevýhody. Pro lepší orientaci a srovnání byly popisy doplněny o tabulky a ilustrativní obrázky. U většiny analyzovaných nástrojů byly rovněž uvedeny příkazy běžně používané při konfiguraci síťových zařízení v daném prostředí.

V praktické části byla provedena konfigurace síťových topologií v prostředí GNS3 a Cisco Packet Tracer, přičemž bylo využito jak statického, tak dynamického směrování. U každého příkladu byla podrobně dokumentována konfigurace všech použitých zařízení, aby byl postup snadno srozumitelný a reprodukovatelný.

Při zpracování teoretické části se vyskytly určité obtíže, zejména při vyhledávání informací k některým analyzovaným nástrojům. V několika případech nebyla k dispozici oficiální dokumentace, což zpomalilo zpracování. V praktické části se nejčastěji vyskytovaly chyby při konfiguraci zařízení, které byly způsobeny nesprávným nastavením parametrů některých komponent. Přes uvedené komplikace však bylo dosaženo lepšího porozumění chování jednotlivých systémů a nástroje byly důkladně prakticky zvládnuty.

SEZNAM POUŽITÉ LITERATURY

- [1] *Rychlý start.* Online. Ns-3. 2019, 2025-03-22. Dostupné z: <https://www.nsnam.org/docs/tutorial/html/quick-start.html>. [cit. 2025-03-27].
- [2] *Návrh laboratorních úloh pro výuku síťových technologií a protokolů.* Diplomová práce. Brno: Vysoké učení technické v Brně, 2019.
- [3] *Co je ns-3.* Online. Ns-3. 2011. Dostupné z: <https://www.nsnam.org/about/what-is-ns-3/>. [cit. 2025-03-27].
- [4] *Jednoduchá simulace síťového spojení mezi dvěma uzly.* Online. Chatgpt. Dostupné z: <https://chatgpt.com/>. [cit. 2025-04-10].
- [5] *Co je GNS3?* Online. GNS3. 2025. Dostupné z: <https://docs.gns3.com/docs/>. [cit. 2025-04-11].
- [6] *Modelování počítačové sítě pomocí nástroje GNS3.* Bakalářská práce. Ostrava: Vysoká škola Báňská – Technická univerzita Ostrava, 2013.
- [7] *Instalace GNS3.* Online. GNS3. 2025. Dostupné z: <https://docs.gns3.com/docs/getting-started/installation/windows>. [cit. 2025-04-26].
- [8] *Informace.* Online. Chatgpt. 2025. Dostupné z: <https://chatgpt.com/>. [cit. 2025-04-26].
- [9] *Co je OMNeT++.* Online. OMNeT++. 2001. Dostupné z: <https://omnetpp.org/intro/>. [cit. 2025-05-01].
- [10] *Simulační manuál.* Online. OMNeT++. 2001. Dostupné z: <https://doc.omnetpp.org/omnetpp/manual/#sec:config-sim:rng-config>. [cit. 2025-05-01].
- [11] *Stručný přehled vývojového prostředí OMNeT++.* Online. OMNeT++. 2001. Dostupné z: <https://omnetpp.org/documentation/ide-overview/>. [cit. 2025-05-01].
- [12] *Co je IDE?* Online. Aws.amazon. 2025. Dostupné z: <https://aws.amazon.com/what-is/ide/>. [cit. 2025-05-01].
- [13] *Installation guide.* OMNeT++. 2021.

- [14] *Ultimátní průvodce sledováním paketů*. Online. ITPRC. 2019. Dostupné z: <https://www.itprc.com/packet-tracers/>. [cit. 2025-05-01].
- [15] *Co je Cisco Packet Tracer?* Online. Nwking. 2024. Dostupné z: <https://www.nwking.com/what-is-a-cisco-packet-tracer>. [cit. 2025-05-01].
- [16] *Konfigurace síťových prvků a protokolů*. Vysoká škola báňská – Technická univerzita Ostrava Fakulta strojní. 2011.
- [17] *Jak funguje router?* Online. Cisco. 2025. Dostupné z: <https://www.cisco.com/c/en/us/solutions/small-business/resource-center/networking/how-does-a-router-work.html>. [cit. 2025-05-01].
- [18] *Jak funguje switch?* Online. Cisco. 2025. Dostupné z: <https://www.cisco.com/c/en/us/solutions/small-business/resource-center/networking/network-switch-how.html>. [cit. 2025-05-01].
- [19] *Student's Performance Assessment and Learning Skill towards Wireless Network Simulation Tool – Cisco Packet Tracer*. CORE. 2019.
- [20] *Úvod do firewallů*. Online. Cisco. 2025. Dostupné z: <https://www.cisco.com/site/us/en/learn/topics/security/what-is-a-firewall.html>. [cit. 2025-05-01].
- [21] *Simulátor vs Emulátor*. Online. Saucelabs. 2024. Dostupné z: <https://saucelabs.com/resources/blog/simulators-vs-emulators-whats-the-difference-anyway>. [cit. 2025-05-01].
- [22] *Co jsou simulátory a emulátory*. Online. Perfecto. 2023. Dostupné z: <https://www.perfecto.io/blog/emulation-vs-simulation>. [cit. 2025-05-01].
- [23] *Tc-netem*. Online. Man7. 2011. Dostupné z: <https://man7.org/linux/man-pages/man8/tc-netem.8.html>. [cit. 2025-05-06].
- [24] *Použití NetEm k emulaci sítí*. Online. Srtlab.github. 2019. Dostupné z: <https://srtlab.github.io/srt-cookbook/how-to-articles/using-netem-to-emulate-networks.html>. [cit. 2025-05-06].
- [25] ODOM, Wendell; HEALY, Rus a MEHTA, Naren. *Směrování a přepínání sítí: autorizovaný výukový průvodce. Samostudium*. Brno: Computer Press, 2009. ISBN 9788025125205.
- [26] HUCABY, Dave; MCQUERRY, Steve a WHITAKER, Andrew. *Cisco router configuration handbook*. 2nd ed. Indianapolis, IN: Cisco Press, c2010. ISBN 9781587141164.

- [27] *Počítačové sítě – Statické směrování (static routing)*. Online. 2023. Dostupné z: https://sspvc.lixis.cz/psi/psi_3.rocnik/static_routing.html. [cit. 2025-05-29].
- [28] *Dynamické routování (dynamic routing)*. Online. 2023. Dostupné z: https://sspvc.lixis.cz/psi/psi_3.rocnik/dynamic_routing.html. [cit. 2025-05-29].
- [29] *Distance Vector Routing (DVR) Protocol*. Online. Geeksforgeeks. 2024. Dostupné z: <https://www.geeksforgeeks.org/distance-vector-routing-dvr-protocol/>. [cit. 2025-05-30].
- [30] *Unicast Routing – Link State Routing*. Online. Geeksforgeeks. 2025. Dostupné z: <https://www.geeksforgeeks.org/unicast-routing-link-state-routing/>. [cit. 2025-05-30].

SEZNAM OBRÁZKŮ

Obrázek 1: Struktura ns-3	19
Obrázek 2: Architektura GNS3	22
Obrázek 3: OMNeT++ IDE, úprava souboru NED	28
Obrázek 4: Příklad topologie v Cisco Packet Tracer	32
Obrázek 5: Základní síťová topologie.....	46
Obrázek 6: Jednoduchá síťová topologie.....	48
Obrázek 7: Topologie s využitím RIP.....	50
Obrázek 8: Topologie s využitím OSPF	53
Obrázek 9: Jednoduchá topologie v Cisco Packet Tracer	55
Obrázek 10: Cisco Static Routing.....	58
Obrázek 11: Topologie v Cisco Packet Tracer s využitím RIP	60
Obrázek 12: Topologie dynamického směrování s využitím OSPF	64

SEZNAM TABULEK

Tabulka 1: Diference mezi Simulátorem a Emulátorem	16
Tabulka 2: Porovnání GNS3 s ostatními SW	25
Tabulka 3: Diference mezi SW	39
Tabulka 4: Konfigurace routerů.....	56
Tabulka 5: Konfigurace statického směrování	57
Tabulka 6: Propojovací síť (Router-Router).....	61
Tabulka 7: LAN síť (koncová zařízení).....	62

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

5G	Pátá generace mobilních sítí
ACL	Access Control List
ASA	Adaptive Security Appliance
BGP	Border Gateway Protocol
CCNA	Cisco Certified Network Associate
CLI	Comand Line Interface
CSV	Comma-Separated Values
DHCP	Dynamic Host Configuration Protocol
DCE	Direct Code Execution
DNS	Domain Name Systém
EIGRP	Enhanced Interior Gateway Routing Protocol
FTP	File Transfer Protocol
GNS3	Graphical Network Simulator 3
GPL	General Public Licence
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IDE	Integrated Development Environment
IGP	Interior Gateway Protocol
INET	Síťový Framework OMNeT++
LoT	Internet of Things
IP	Internet Protocol
IPv4/IPv6	Internet Protocol version 4/6
IOS	Internetwork Operating System
KVM	Kernel-based Virtual Machine

LAN	Local Area Network
LTE	Long Term Evolution
LSA	Link State Advertisement
MAC	Media Access Control
MANET	Mobile AD-hoc Network
NAT	Network Address Translation
např.	například
NED	Network Description Language
NetAcad	Cisco Networking Academy
NetEm	Network Emulator
OMNeT++	Objektově orientovaný modulární síťový simulátor
OS	Operating System
OSPF	Open Shortest Path First
PC	Personal Computer
PEMU	PIX Emulator
QEMU	Quick Emulator
RIP	Routing Information Protocol
RTT	Round Trip Time
STP	Spanning Tree Protocol
SW	Software
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VLAN	Virtual Local Area Network
VM	Virtual Machine
VPN	Virtual Private Network
VTP	VLAN Trunking Protocol

VMware Virtual Machine Software

WiFi Wireless Fidelity

SEZNAM PŘÍLOH

Příloha P1: GNS3 konfigurace Routerů (RIP)

Příloha P2: GNS3 konfigurace routerů (OSPF)

Příloha P3: Konfigurace routerů v Cisco Packet Tracer (OSPF)

PŘÍLOHA P1: GNS3 KONFIGURACE ROUTERŮ (RIP)

Router1:

```
conf t
hostname R1
interface FastEthernet0/0
  ip address 192.168.1.1 255.255.255.0
  no shutdown
exit
interface FastEthernet0/1
  ip address 10.0.12.1 255.255.255.0
  no shutdown
exit
router rip
  version 2
  no auto-summary
  network 192.168.1.0
  network 10.0.12.0
exit
```

Router2:

```
conf t
hostname R2
interface FastEthernet0/0
  ip address 192.168.2.1 255.255.255.0
  no shutdown
exit
interface FastEthernet0/1
  ip address 10.0.12.2 255.255.255.0
  no shutdown
exit
interface FastEthernet1/0
  ip address 10.0.23.1 255.255.255.0
  no shutdown
exit
router rip
  version 2
  no auto-summary
  network 192.168.2.0
  network 10.0.12.0
```

```
network 10.0.23.0
exit
```

Router3:

```
conf t
hostname R3
interface FastEthernet0/0
  ip address 192.168.3.1 255.255.255.0
  no shutdown
exit
interface FastEthernet1/0
  ip address 10.0.23.2 255.255.255.0
  no shutdown
exit
router rip
  version 2
  no auto-summary
  network 192.168.3.0
  network 10.0.23.0
exit
```

PŘÍLOHA P2: GNS3 KONFIGURACE ROUTERŮ (OSPF)

Router1:

```
configure terminal
hostname R1
interface FastEthernet0/1
  ip address 10.0.12.1 255.255.255.252
  no shutdown
interface FastEthernet1/0
  ip address 10.0.13.1 255.255.255.252
  no shutdown
interface FastEthernet0/0
  ip address 192.168.1.1 255.255.255.0
  no shutdown
interface Loopback0
  ip address 1.1.1.1 255.255.255.255
router ospf 1
  router-id 1.1.1.1
  network 10.0.12.0 0.0.0.3 area 0
  network 10.0.13.0 0.0.0.3 area 0
  network 192.168.1.0 0.0.0.255 area 0
  network 1.1.1.1 0.0.0.0 area 0
exit
write memory
```

Router2:

```
configure terminal
hostname R2
interface FastEthernet0/1
  ip address 10.0.12.2 255.255.255.252
  no shutdown
interface FastEthernet1/0
  ip address 10.0.24.1 255.255.255.252
  no shutdown
interface FastEthernet0/0
  ip address 192.168.2.1 255.255.255.0
  no shutdown
interface Loopback0
  ip address 2.2.2.2 255.255.255.255
```

```
router ospf 1
  router-id 2.2.2.2
  network 10.0.12.0 0.0.0.3 area 0
  network 10.0.24.0 0.0.0.3 area 0
  network 192.168.2.0 0.0.0.255 area 0
  network 2.2.2.2 0.0.0.0 area 0
exit
write memory
```

Router3:

```
configure terminal
hostname R3
interface FastEthernet1/0
  ip address 10.0.13.2 255.255.255.252
  no shutdown
interface FastEthernet0/1
  ip address 10.0.34.1 255.255.255.252
  no shutdown
interface FastEthernet0/0
  ip address 192.168.3.1 255.255.255.0
  no shutdown
interface Loopback0
  ip address 3.3.3.3 255.255.255.255
router ospf 1
  router-id 3.3.3.3
  network 10.0.13.0 0.0.0.3 area 0
  network 10.0.34.0 0.0.0.3 area 0
  network 192.168.3.0 0.0.0.255 area 0
  network 3.3.3.3 0.0.0.0 area 0
exit
write memory
```

Router4:

```
configure terminal
hostname R4
interface FastEthernet1/0
  ip address 10.0.24.2 255.255.255.252
  no shutdown
interface FastEthernet0/1
```

```
ip address 10.0.34.2 255.255.255.252
no shutdown
interface FastEthernet0/0
ip address 192.168.4.1 255.255.255.0
no shutdown
interface Loopback0
ip address 4.4.4.4 255.255.255.255
router ospf 1
router-id 4.4.4.4
network 10.0.24.0 0.0.0.3 area 0
network 10.0.34.0 0.0.0.3 area 0
network 192.168.4.0 0.0.0.255 area 0
network 4.4.4.4 0.0.0.0 area 0
exit
write memory
```

PŘÍLOHA P3: KONFIGURACE ROUTERŮ V CISCO PACKET TRACER (OSPF)

Router0:

```
enable
configure terminal
hostname Router0
interface GigabitEthernet0/0
  ip address 10.0.0.1 255.255.255.252
  no shutdown
exit
interface GigabitEthernet1/0
  ip address 10.0.0.14 255.255.255.252
  no shutdown
exit
interface FastEthernet2/0
  ip address 10.0.0.17 255.255.255.252
  no shutdown
exit
interface FastEthernet3/0
  ip address 192.168.1.1 255.255.255.0
  no shutdown
exit
router ospf 1
  router-id 1.1.1.1
  network 10.0.0.0 0.0.0.3 area 0
  network 10.0.0.12 0.0.0.3 area 0
  network 10.0.0.16 0.0.0.3 area 0
  network 192.168.1.0 0.0.0.255 area 0
  passive-interface FastEthernet3/0
exit
end
write memory
clear ip ospf process
```

Router1:

```
enable
configure terminal
hostname Router1
```

```
interface GigabitEthernet0/0
  ip address 10.0.0.2 255.255.255.252
  no shutdown
exit
interface GigabitEthernet1/0
  ip address 10.0.0.5 255.255.255.252
  no shutdown
exit
interface FastEthernet2/0
  ip address 10.0.0.21 255.255.255.252
  no shutdown
exit
interface FastEthernet3/0
  ip address 192.168.2.1 255.255.255.0
  no shutdown
exit
router ospf 1
  router-id 2.2.2.2
  network 10.0.0.0 0.0.0.3 area 0
  network 10.0.0.4 0.0.0.3 area 0
  network 10.0.0.20 0.0.0.3 area 0
  network 192.168.2.0 0.0.0.255 area 0
  passive-interface FastEthernet3/0
exit
end
write memory
clear ip ospf process
```

Router2:

```
enable
configure terminal
hostname Router2
interface Serial0/0
  ip address 10.0.0.10 255.255.255.252
  clock rate 128000
  no shutdown
exit
interface GigabitEthernet1/0
  ip address 10.0.0.13 255.255.255.252
  no shutdown
```

```
exit
interface FastEthernet2/0
  ip address 10.0.0.22 255.255.255.252
  no shutdown
exit
interface FastEthernet3/0
  ip address 192.168.10.1 255.255.255.128
  no shutdown
exit
interface FastEthernet4/0
  ip address 192.168.10.129 255.255.255.128
  no shutdown
exit
router ospf 1
  router-id 3.3.3.3
  network 10.0.0.8 0.0.0.3 area 0
  network 10.0.0.12 0.0.0.3 area 0
  network 10.0.0.20 0.0.0.3 area 0
  network 192.168.10.0 0.0.0.127 area 0
  network 192.168.10.128 0.0.0.127 area 0
  passive-interface FastEthernet3/0
  passive-interface FastEthernet4/0
exit
end
write memory
clear ip ospf process
```

Router3:

```
enable
configure terminal
hostname Router3
interface Serial10/0
  ip address 10.0.0.9 255.255.255.252
  no shutdown
exit
interface GigabitEthernet1/0
  ip address 10.0.0.6 255.255.255.252
  no shutdown
exit
interface FastEthernet2/0
```

```
ip address 10.0.0.18 255.255.255.252
no shutdown
exit
interface FastEthernet3/0
ip address 172.16.0.1 255.255.252.0
no shutdown
exit
interface FastEthernet4/0
ip address 172.16.4.1 255.255.254.0
no shutdown
exit
router ospf 1
router-id 4.4.4.4
network 10.0.0.4 0.0.0.3 area 0
network 10.0.0.8 0.0.0.3 area 0
network 172.16.0.0 0.0.3.255 area 0
network 172.16.4.0 0.0.1.255 area 0
passive-interface FastEthernet3/0
passive-interface FastEthernet4/0
exit
end
write memory
clear ip ospf process
```