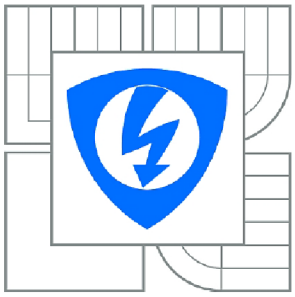
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

NÁVRH LABORATORNÍCH ÚLOH PRO SIMULAČNÍ PROSTŘEDÍ OPNET IT GURU

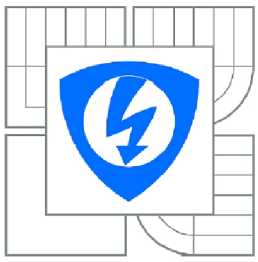
DESIGN OF PRACTICAL TUTORIALS FOR THE OPNET IT GURU SIMULATION ENVIRONMENT

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. MARTINA MATZOVÁ
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ HOŠEK, Ph.D.
SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Studentka: Bc. Martina Matzová ***ID:*** 123404
Ročník: 2 ***Akademický rok:*** 2011/2012

NÁZEV TÉMATU:

Návrh laboratorních úloh pro simulační prostředí OPNET IT Guru

POKYNY PRO VYPRACOVÁNÍ:

Cílem diplomové práce je vytvoření tří laboratorních úloh v simulačním prostředí OPNET IT Guru zaměřených na síťové technologie. První laboratorní úloha bude názorně demonstrovat rozdíly mezi transportními protokoly TCP a UDP, proto podrobně nastudujte funkce a parametry těchto transportních protokolů. V prostředí OPNET IT Guru vytvořte simulační model datové sítě, v kterém vhodně nakonfigurujte síťový provoz. Pro jeho přenos použijte na transportní vrstvě postupně protokoly TCP a UDP. Formou simulací proveďte srovnání obou zmíněných transportních protokolů. Další laboratorní úloha bude zaměřena na práci s nástrojem ACE, který je součástí prostředí IT Guru a slouží pro analýzu výkonu a problémů u uživatelských aplikací. Tato úloha bude představovat základní práci s nástrojem ACE a jeho použití při analýze aplikace typu databázový přístup. Třetí laboratorní úloha bude zaměřena na komunikační protokol ICMP a jeho použití v nástroji Ping. Cílem této úlohy bude simulace ICMP spojení a výměna řídících zpráv mezi komunikujícími uzly. Všechny vytvořené scénáře a dosažené výsledky zpracujte formou návodů k laboratorním úlohám.

DOPORUČENÁ LITERATURA:

[1] McCABE, J.: Network Analysis, Architecture, and Design. San Francisco: Morgan Kaufmann, 2007, ISBN: 978-0123704801.
[2] OLIFER, N., OLIFER, V.: Computer Networks: Principles, Technologies and Protocols for Network Design. Chichester: John Wiley & Sons, 2006, ISBN: 0470869828.

Termín zadání: 6.2.2012 ***Termín odevzdání:*** 24.5.2012

Vedoucí práce: Ing. Jiří Hošek, Ph.D.
Konzultanti diplomové práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

ANOTÁCIA

Táto diplomová práca sa zaoberá vytvorením celkovo troch laboratórnych úloh v simulačnom prostredí OPNET IT Guru. Prvá laboratórna úloha demonštruje rozdiely medzi protokolmi transportnej vrstvy RM OSI – TCP a UDP. Druhá laboratórna úloha pojednáva o protokole ICMP a jeho najznámejšej funkcii PING. Tretia laboratórna úloha sa venuje nástroju Application Characterization Environment (ACE), ktorý je súčasťou programu OPNET IT Guru. Ku všetkým spomínaným laboratórnym prácam je v diplomovej práci vypracovaná i teoretická stránka. Všetky labotatórne práce je možné využiť na vyučovaní na Fakulte elektrotechniky a komunikačních technologií VUT v predmetoch venujúcich sa sieťovým technológiám.

Kľúčové slová: laboratórna úloha, simulácia, OPNET IT Guru, TCP/UDP, ICMP, ACE

ABSTRACT

This thesis deals with the design of three labs in the simulation environment OPNET IT Guru. The first lab demonstrates the differences between the RM OSI transport layer protocols - TCP and UDP. The second lab discusses the role of ICMP protocol and its most famous feature - PING. The third lab is devoted to Application Characterization Environment (ACE) module, which is included in OPNET IT Guru. Theoretical background is provided for all of the mentioned topics. All designed labs can be used to teach at the Faculty of Electrical Engineering and Communication of BUT on subjects dealing with network technologies.

Keywords: lab, simulation, OPNET IT Guru, TCP/UDP, ICMP, ACE

BIBLIOGRAFICKÁ CITÁCIA

MATZOVÁ, M. *Návrh laboratorních úloh pro simulační prostředí OPNET IT Guru.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 96 s. Vedoucí diplomové práce Ing. Jiří Hošek, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svoji diplomovou práci na téma Návrh laboratorních úloh pro simulační prostředí OPNET IT Guru jsem vypracovala samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne ....

podpis autora

POĎAKOVANIE

Ďakujem vedúcemu mojej diplomovej práce Ing. Jiřímu Hoškovi, Ph.D. za pomoc pri vypracovávaní mojej práce, za kontinuálnu spätnú väzbu a cenné rady.

V Brne dňa ....

podpis autora

SIX research centre
sensor, information and communication systems

Faculty of Electrical Engineering
and Communication

Brno University of Technology
Purkynova 118, CZ-61200 Brno, Czechia

http://www.six.feec.vutbr.cz

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007–13
OP Výzkum a vývoj pro inovace

OBSAH

1 ÚVOD

V súčasnosti sú počítačové siete prítomné v takmer každej sfére. Vyvstáva otázka, či vzdelávanie v oblasti sietí a telekomunikácií vo všeobecnosti, má byť realizované na reálnych zariadeniach a v reálnom prostredí alebo len virtuálne, pomocou sieťových simulátorov.

Drvivá väčšina škôl žiaľ nedisponuje prostriedkami na vybudovanie laboratórií s najnovšími technológiami. V laboratórnom prostredí taktiež nie je možné vytvoriť siete väčšieho rozsahu a tiež nie je možné z prevádzkových dôvodov svojvoľne meniť nastavenia zariadení a konfiguráciu siete. Laboratórne prostredie nedokáže poskytnúť technickú rozmanitosť a je viacmenej limitované na pár použitých technológií.

Riešenie všetkých týchto nedostatkov fyzických sietí predstavujú sieťové simulátory, ktoré poskytujú širokú škálu technológií od rôznych výrobcov. Jedným z takýchto softvérov je aj program IT Guru, ktorý predstavuje akademickú verziu programu OPNET Modeler.

Cieľom mojej diplomovej práce je vytvorenie laboratórnych úloh použiteľných na vyučovaní predmetov súvisiacich so sieťovými technológiami, práve v programe IT Guru. Budem porovnávať protokoly transportnej vrstvy – TCP a UDP z hľadiska časovej odozvy vykonávaných úloh a z hľadiska spotreby sieťových prostriedkov pri vykonávaní daných úloh. Práca ďalej obsahuje vlastnosti týchto protokolov.

Ďalej sa v práci budem venovať popisu vlastností a fungovania protokolu ICMP a jeho systémových správam. Laboratórna úloha k protokolu ICMP sa bude venovať jeho najznámejšej funkcii PING.

V práci sa ešte budem venovať nástroju na diagnostiku a analýzu sieťových aplikácií implementovanom v programe OPNET IT Guru – ACE (Application Characterization Environment). Laboratórna úloha k tejto téme bude oboznamovať užívateľa so základnými funkčnosťami a nástrojmi, ktoré obsahuje ACE.

2 OPNET IT GURU

OPNET IT Guru Academic Edition predstavuje verziu programu OPNET Modeler pre akademické účely. Univerzity často nemajú možnosť praktického aplikovania vyučovaných poznatkov na reálnych sieťach vybudovaných niekde v laboratóriách, a preto je vítané použitie prostredie OPNET IT Guru, ktoré poskytuje možnosť modelovania, simulácie a následnej analýzy modelovanej sieťovej prevádzky. Hoci je IT Guru ochudobnené o určité funkcie, veľmi dobre poslúži na študijné účely. Užívateľovi dáva možnosť výberu z veľkej palety sieťových zariadení, ako napríklad smerovače, prepínače, servery a samotné pracovné stanice, a prepojiť ich spolu pomocou rôznych typov liniek dostupných v palete nástrojov. Výhodou OPNET IT Guru je možnosť „nanečisto" si vyskúšať fungovanie navrhnutej siete, poprípade zistiť dopady plánovanej zmeny na aktuálnu sieť bez nutnosti skutočného vykonania tejto zmeny. Počas času simulácie každého sieťového scenára sa zbierajú štatistiky, ktoré si užívateľ nastaví podľa aktuálnej potreby, a tie sú ďalej analyzované. [4][5]

2.1 Princíp práce s OPNET IT Guru

OPNET IT Guru umožňuje kompletný virtuálny návrh siete ľubovoľných rozmerov pri použití všetkých dostupných technológií. Každá práca s IT Guru začína vytvorením nového projektu, ktorý je ďalej členený na scenáre, alebo úpravou už vytvoreného projektu. Pri vytváraní projektu je na výber rozsah siete, ktorá bude simulovaná, poprípade ak sú tieto rozmery zanedbateľné, je možnosť zvoliť aj čisto logickú sieť. IT Guru ponúka širokú paletu zariadení od rôznych výrobcov pre rôzne sieťové technológie, preto je z týchto možné poskladať ľubovoľnú sieťovú topológiu. Paleta zariadení obsahuje aj čisto logické zariadenia, ako napríklad *Application Config* alebo *Profile Config*, ktoré slúžia na prevádzkové a konfiguračné účely a vlastne tvoria gro celej simulácie. Samotný proces simulácie ponúka tiež mnoho možností konfigurácie. Konečnou fázou simulácie je prehliadanie výsledkov, ktoré je možné vo viacerých variantách a štatistických úpravách.

3 SIEŤOVÉ MODELY OSI A TCP/IP

3.1 Referenčný model ISO/OSI

Model ISO/OSI (Open System Interconnection) reprezentuje štandard, ktorý definuje metódy vertikálneho rozloženia problému interakcie počítačov tým, že tieto úlohy sú presunuté na komunikačné protokoly rozdelené do siedmich vrstiev. Komunikačné protokoly formujú istú hierarchiu, tzv. protokolovú sadu (ďalej stack), v ktorej každá vrstva využíva vrstvu pod sebou na pohodlné vyriešenie jej úloh. [2]

3.1.1 Vrstvy modelu ISO/OSI

1) **Fyzická vrstva** – Poskytuje elektrické a mechanické spojenie so sieťou. Príkladom technológií pracujúcich na fyzickej vrstve sú napríklad optické vlákna, krútená dvojlinka resp. sieťová karta.

2) **Spojová vrstva** – Stará sa o opravu chýb v prenose a kontrolu toku dát. Vrstva je tiež známa ako vrstva pre kontrolu prístupu k médiu MAC (Media Access Control). Je tu definované MAC adresovanie.

3) **Sieťová vrstva** – Poskytuje funkciu smerovania a kombinuje segmenty alebo správy do paketov pridaním informácie o smerovaní do hlavičky. Príkladom sú napríklad protokoly IP a IPX.

4) **Transportná vrstva** – Zaoberá sa integritou správ medzi zdrojom a cieľom. Taktiež rozdeľuje a spája pakety a zabezpečuje tok dát. Príkladom sú protokoly TCP a UDP.

5) **Relačná vrstva** – Poskytuje funkcie nutné na zostavenie, manažovanie a ukončenie spojení podľa požiadaviek užívateľa, napr. NFS a SQL.

6) **Prezentačná vrstva** – Akceptuje a štrukturuje správy pre aplikácie. Prekladá správy z jedného kódu do druhého, ak je to potrebné. Je taktiež zodpovedná za kompresiu a šifrovanie dát, napr. ASCII.

7) **Aplikačná vrstva** – Interaguje s aplikačnými programami, ktoré zahŕňajú komunikačné komponenty ako napríklad internetový prehliadač alebo e-mail. Je zodpovedná za interpretáciu požiadaviek a určenie toho, aká informácie je potrebná na splnenie požiadavky. Príkladom sú protokoly HTTP, FTP alebo SMTP. [1]

V súčasnosti používané protokolové stacky sú postavené na architektúre OSI, ale aj tak ma každý svoje špecifiká, ktorými sa od OSI odlišuje. Napríklad najpopulárnejší TCP/IP stack nahrádza sedem vrstiev OSI len štyrmi. [2][3]

3.2 Sieťový model TCP/IP

Sústava protokolov TCP/IP je používaná na komunikáciu medzi zariadeniami, tzv. hostami, v lokálnych sieťach, ale aj na Internete. TCP/IP bolo vyvinuté agentúrou DARPA (Defense Advanced Research Projects Agency) ako prostriedok na vytvorenie siete počítačov vo vládnom výskume. [3]

3.2.1 Vrstvy modelu TCP/IP

1) **Vrstva sieťového rozhrania** – Definuje, ako sa host pripája do siete.
2) **Sieťová vrstva** – Definuje protokoly používané na adresovanie a smerovanie dátových paketov.
3) **Transportná vrstva** - Definuje typ vytvoreného spojenia medzi hostami a spôsob zasielania potvrdení.
4) **Aplikačná vrstva** – Definuje aplikácie, porty a sockety používané na spracovanie požiadaviek. [3]

3.3 OSI vs. TCP/IP

Zavedenie TCP/IP nie je v rozpore s OSI štandardom, pretože tieto dva protokolové stacky boli vyvíjané konkurenčne. Dalo by sa povedať, že TCP/IP prispieva k OSI a naopak. Avšak existujú určité rozdiely. Ako vidíme v Tab. 3.1, TCP/IP kombinuje aplikačnú, prezentačnú a relačnú vrstvu OSI do jedinej aplikačnej vrstvy. Taktiež kombinuje prvé dve vrstvy OSI, fyzickú a linkovú, do jednej vrstvy sieťového rozhrania. [6]

Tab. 3.1: Porovnanie modelov OSI a TCP/IP [3]

Model OSI	Model TCP/IP	TCP/IP Protokoly
Aplikačná vrstva	**Aplikačná vrstva**	Telnet, FTP, SMTP, DNS, SNMP
Prezentačná vrstva		
Relačná vrstva		
Transportná vrstva	**Transportná vrstva**	TCP, UDP
Sieťová vrstva	**Internetová vrstva**	IP, ICMP, ARP
Linková vrstva	**Vrstva sieť. rozhrania**	Ethernet, ATM, Token Ring, Frame Relay
Fyzická vrstva		

4 PROTOKOLY TRANSPORTNEJ VRSTVY

Hlavnou úlohou transportnej vrstvy je vytvoriť transportné spojenie, manažment doručovania dát medzi zdrojovými a cieľovými uzlami, v tomto prípade sú to priamo aplikačné programy, a v neposlednom rade ukončenie dátového spojenia. Podľa nárokov a požiadaviek koncových uzlov môže transportná vrstva regulovať tok dát obidvomi smermi, zaisťovať spoľahlivosť prenosu a tiež meniť nespojový charakter prenosu na spojový. Napriek tomu, že je činnosť transportnej vrstvy najčastejšie zaisťovaná protokolom TCP, nie je jediným. Ďalším protokolom na úrovni transportnej vrstvy je protokol UDP, ktorý narozdiel od TCP nezaisťuje spoľahlivosť prenosu. Je však využívaný len aplikáciami, ktoré spoľahlivosť na úrovni transportnej vrstvy vyslovene nevyžadujú. [1][2]

4.1 Protokol UDP

Ako sa uvádza v [2], protokol UDP (User Datagram Protocol) je takzvaný datagramový protokol, to znamená, že tento protokol pracuje na princípe *best effort*, čo by sa dalo voľne preložiť ako najväčšie úsilie. Protokol nepotrebuje pred prenosom vybudovať logické spojenie, ale zároveň neexistuje ani žiadne potvrdenie, že datagramy prišli do cieľovej stanice. Dátovou jednotkou UDP je užívateľský datagram. Každý UDP datagram nesie samostatnú užívateľskú správu a veľkosť nesmie presiahnuť veľkosť IP dátového poľa, ktoré je zas limitované veľkosťou rámca nižšej sieťovej technológie. Čiže keď dôjde k pretečeniu vyrovnávacej pamäte (tzv. buffer) UDP, dáta sú stratené. Protokol UDP sa používa v tzv. real-time aplikáciách ako napríklad VoIP alebo streamovanie videa. Protokoly DNS a SNMP taktiež využívajú protokol UDP na prenos informácie. [1]

4.1.1 UDP datagram

Datagram UDP je pomerne jednoduchý, skladá sa z hlavičky a dátovej časti (viď Obr. 4.1). Hlavička datagramu UDP sa skladá zo štyroch polí o veľkosti 2 bajty [2]:

- **Zdrojový port**
- **Cieľový port**
- **Dĺžka datagramu**
- **Kontrolný súčet**

Zdrojový port (16 bitov)	Cieľový port (16 bitov)
Dĺžka datagramu (16 bitov)	Kontrolný súčet (16 bitov)
Dáta (premenlivá veľkosť)	

Obr. 4.1: UDP datagram

Funkcia protokolu UDP je prakticky zredukovaná na multiplexovanie a demultiplexovanie dát medzi sieťovou a aplikačnou vrstvou. UDP využíva čísla portov, na základe ktorých vykonáva funkciu demultiplexovania. Rámce, ktoré nesú UDP datagramy, prídu na sieťové rozhranie daného zariadenia, kde sú tieto spracované rôznymi protokolmi a nakoniec sú podané smerom k UDP. UDP získa čislo portu cieľovej stanice z hlavičky datagramu a podá dáta ďalej k vhodnému portu korešpondujúcej aplikácie (demultiplexovanie). [2]

4.2 Protokol TCP

Koncept protokolu TCP (Transmission Control Protocol) je vymyslený tak, že garantuje spoľahlivé doručenie dátových paketov v správnom poradí a taktiež zahŕňa podporu pre kontrolu chýb a obnovenie stratených dát. Pred samotným prenosom je však nutné vybudovať sieťové prostredie a po dokončení prenosu toto spojenie aj ukončiť. [2]

Informácie poskytované smerom k TCP, od protokolov vyšších vrstiev, pokladá TCP za neštrukturovaný tok bajtov. Prichádzajúce data sú vložené vo vyrovnávacej pamäti TCP. Protokol potom vyberie určitý segment dát, pripojí k nemu hlavičku a posunie ho smerom k sieťovej vrstve. Protokol TCP sa využíva najmä pre dátové prenosy cez FTP a emailové protokoly ako SMTP. [2]

4.2.1 TCP segment

Hlavička TCP segmentu je zložitejšia než hlavička UDP datagramu, pretože TCP má pokročilejšie funkcie (viď Obr. 4.2). Ako sa uvádza v [2], skladá sa z nasledujúcich častí:

- **Zdrojový port** – veľkosť 2 bajty, identifikuje proces u odosielateľa
- **Cieľový port** – veľkosť 2 bajty, identifikuje proces u prijímateľa
- **Poradové číslo (Sequence Number)** – veľkosť 4 bajty, špecifikuje číslo bajtu, ktoré definuje posunutie segmentu vo vzťahu k posielanému toku dát
- **Číslo potvrdenia (Acknowledgement Number)** – veľkosť 4 bajty, obsahuje číslo najväčšieho bajtu v doručenom segmente zväčšené o 1

- **Dĺžka hlavičky (HLen)** – veľkosť 4 bajty, špecifikuje dĺžku hlavičky TCP segmentu, meraná v 32-bitových slovách, nie je pevne stanovená a závisí od parametrov nastavených v poli *Možnosti*
- **Rezervované pole** – veľkosť 6 bitov, je rezervované pre budúce použitie.
- **TCP príznaky (Code Bits)** – veľkosť 6 bitov, obsahuje prídavné informácie o type segmentu, ktoré špecifikuje jeden z nasledujúcich bitov:
    - Urgentné dáta – vyjadruje, že správa je urgentná
    - ACK – potvrdzuje prijatý segment
    - PSH – Požiadavka, že správa bude poslaná bez toho, aby sa predtým naplnila vyrovnávacia pamäť. TCP bežne čaká, kým sa vyrovnávacia pamäť naplní a až potom odošle segment. Ak je ale potrebný okamžitý prenos, aplikácia o tom musí informovať protokol parametrom push (PSH).
    - RST – žiada o obnovenie spojenia
    - SYN – používa sa na synchronizáciu počítadiel prenesených dát na začiatku spojenia
    - FIN – špecifikuje, že vysielajúca strana poslala posledný bajt vysielaných dát
- **Okno** – veľkosť 2 bajty, špecifikuje množstvo dátových bajtov začínajúce od bajtu, ktorého poradie je špecifikované v poli potvrdení ACK, ktorého príchod očakáva odosielateľ aktuálneho segmentu
- **Kontrolný súčet** – veľkosť 2 bajty, obsahuje kontrolný súčet
- **Urgentný pointer** – veľkosť 2 bajty, používa sa s kódovým bitom URG a špecifikuje koniec dát, ktoré musia byť urgentne doručené aj napriek pretečeniu vyrovnávacej pamäte
- **Možnosti (Options)** – Toto pole má premenlivú dĺžku a dokonca nemusí byť ani nutne prítomné. Maximálna dĺžka sú 3 bity. Toto pole sa používa pre riešenie prídavných úloh, napríklad na výber maximálnej dĺžky segmentu. Pole možností sa nachádza na konci hlavičky TCP a dĺžka musí byť násobkom 8 bitov.

- **Výplň (padding)** – Toto pole má tiež premenlivú dĺžku a slúži na doplnenie veľkosti hlavičky tak, aby sa veľkosť rovnala celému číslu o 32-bitových slovách.

<table>
<tr><td colspan="6">Zdrojový port (16 bitov)</td><td colspan="4">Cieľový port (16 bitov)</td></tr>
<tr><td colspan="10">Číslo sekvencie (32 bitov)</td></tr>
<tr><td colspan="10">Číslo potvrdenia (32 bitov)</td></tr>
<tr><td>Posunutie dát (4 bity)</td><td>Rezervované pole (6 bitov)</td><td>U R G</td><td>A C K</td><td>P S H</td><td>R S T</td><td>S Y N</td><td>F I N</td><td colspan="2">Okno (16 bitov)</td></tr>
<tr><td colspan="6">Kontrolný súčet (16 bitov)</td><td colspan="4">Urgentný pointer (16 bitov)</td></tr>
<tr><td colspan="6">Možnosti (premenlivá veľkosť)</td><td colspan="4">Padding (premenlivá veľkosť)</td></tr>
<tr><td colspan="10">Dáta (premenlivá veľkosť)</td></tr>
</table>

Obr. 4.2: TCP segment

4.2.2 Princíp fungovania

TCP sa uisťuje, že prenesené segmenty sa nestratia, nezduplikujú alebo prídu v nesprávnom poradí. Pri zostavovaní logického spojenia, entity TCP vyjednajú parametre pre procedúru výmeny dát. Každá strana posiela druhej strane nasledovné parametre [2]:

- Maximálna veľkosť segmentu, ktorú je pripravená prijať
- Maximálny objem dát, ktoré dovolí druhá strana vysielať jej smerom, aj keď prvá strana ešte nedostala žiadne potvrdenie o doručení predchádzajúcich časti dát – táto funkcia sa nazýva *veľkosť okna* (Window Size)
- Číslo štartovacieho bajtu, od ktorého začína počítanie dátového toku vo vnútri systému daného spojenia

Výsledkom procesu vyjednávania medzi TCP entitami obidvoch strán je definovanie parametrov spojenia. Niektoré parametre ostávajú konštantné a iné sa adaptívne menia. Napríklad veľkosť okna posielajúcej strany sa dynamicky mení v závislosti od zaťaženia vyrovnávacej pamäte prijímajúcej strany a od celkovej spoľahlivosti sieťovej prevádzky. Vytvorenie spojenia znamená, že operačný systém na každom počítači vyhradí určité systémové prostriedky pre toto spojenie od vytvorenia, až po ukončenie spojenia. V [2] sa uvádza, že logické TCP spojenie je identifikované párom socketov,

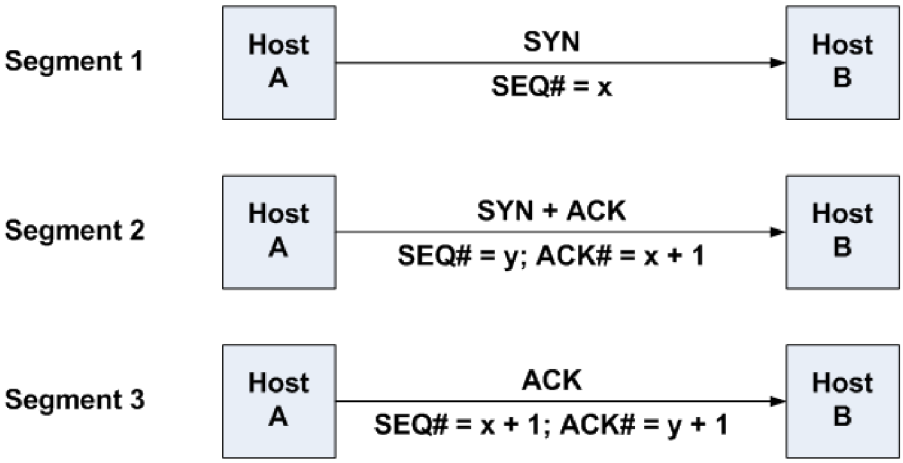
čo je vlastne kombinácia IP adresy daného počítača a portu namapovaného na danú aplikáciu. Každý socket sa môže podieľať na viacerých spojeniach. Uvažujme, že máme tri rôzne sockety troch rôznych aplikácií: (IP_1, n_1), (IP_2, n_2), (IP_3, n_3), kde IP_1, IP_2 a IP_3 predstavujú ich IP adresy a n_1, n_2, n_3 sú čísla portov. V tomto prípade by bolo možné vytvoriť nasledujúce spojenia:

- **Spojenie 1** – {(IP_2, n_2), (IP_1, n_1)}
- **Spojenie 2** – {(IP_1, n_1), (IP_3, n_3)}
- **Spojenie 3** – {(IP_2, n_2), (IP_3, n_3)}

V TCP musí byť každý správny prenos každého segmentu v rámci vytvoreného spojenia potvrdený. Potvrdenia sú tradičnou metódou zaistenia spoľahlivého prenosu. TCP používa mechanizmus so zvláštnymi potvrdeniami, tzv. *mechanizmus posuvného okna*. [2]

Mechanizmus posuvného okna implementovaný v TCP má jednu špecifickú vlastnosť. Hoci prenášanou dátovou jednotkou je segment, okno je definované ako súbor očíslovaných bajtov neštrukturovaného toku dát prichádzajúceho z vyšších vrstiev do vyrovnávacej pamäte TCP. [2]

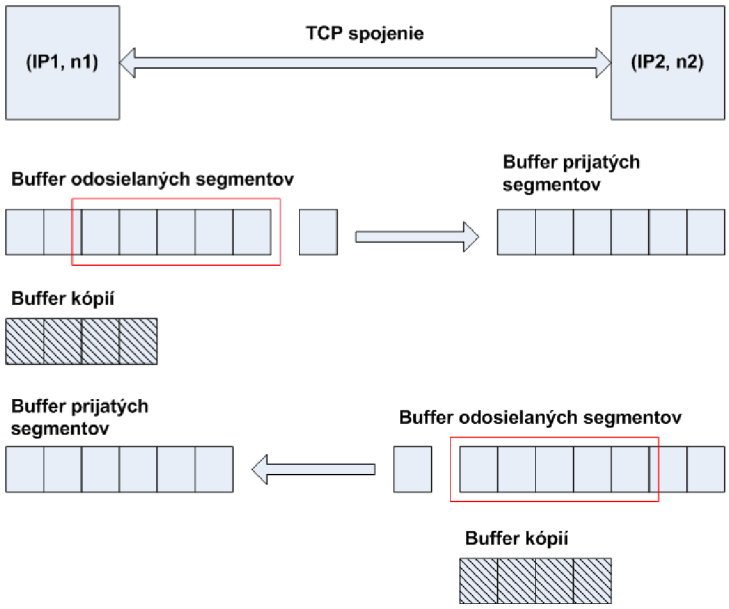
Pri zostavovaní spojenia, ktoré je znázornené na Obr. 4.3, si obidve strany dohodnú štartovacie číslo bajtu, z ktorého sa bude počítať po celú dobu spojenia. Každá strana má svoje vlastné počiatočné číslo. Identifikátor každého segmentu je číslo jeho prvého bajtu. Bajty v rámci segmentu sú číslované tak, že prvý bajt dát, ktorý nasleduje hneď za hlavičkou má najmenšie číslo a bajtom, ktoré nasledujú ďalej sa číslo zvyšuje. Prijímateľ segmentu posiela ako odpoveď potvrdenie. Toto potvrdenie je segment, do ktorého prijímateľ umiestni číslo, ktoré je o jedno väčšie ako číslo doručeného segmentu. Toto číslo sa nazýva *číslo potvrdenia* (Acknowledgement Number) a je často interpretované ako číslo nasledujúceho očakávaného bajtu. [2]



Obr. 4.3: Priebeh zostavovania TCP spojenia [3]

V TCP je potvrdenie poslané až po obdržaní korektných dát, negatívne potvrdenia sa neposielajú. Z toho vyplýva, že nedostatok potvrdení znamená, že daný segment bol stratený, prijímač prijal poškodený segment alebo sa stratilo samotné potvrdenie. V tomto protokole môže rovnaký segment obsahovať aj dáta posielané druhej strane, aj potvrdenie, že TCP entita prijala dáta. [2]

TCP je duplexný protokol, čo značí, že procedúra obojstrannej výmeny dát je regulovaná v rámci systému jedného spojenia. Každá strana je súčasne odosielateľ aj prijímateľ. Každá strana má zároveň svoj pár bufferov (viď Obr. 4.4): jeden na ukladanie prijatých segmentov a druhý je pre segmenty, ktoré čakajú na odoslanie. Okrem týchto dvoch má TCP ešte jeden buffer, ktorý slúži na skladovanie kópií odoslaných segmentov, ktorých potvrdenia o prijatí ešte neprišli. [2]



Obr. 4.4: Vyrovnávacie pamäte (buffery) TCP [2]

V priebehu zostavovania spojenia a neskôr počas obojsmerného prenosu si obidve strany navzájom pošlú tzv. *okno prijímača* (Receiver Window). Tým, že každá strana obdrží okno prijímača, súhlasia koľko bajtov je povolené poslať od momentu posledného potvrdenia. Inými slovami, poslaním okna prijímača sa každá strana snaží regulovať bajtový tok v jej smere, informujúc druhú stranu o počte bajtov (začínajúc od čísla bajtu, pre ktorý bolo poslané potvrdenie), ktoré je pripravené prijať. [2]

Prijímač môže poslať potvrdenie, ktoré potvrdzuje prijatie súčasne viacerých segmentov za predpokladu, že tieto segmenty tvoria kontinuálny bajtový tok. Keď TCP posiela segment do siete, vloží kópiu tohto segmentu do preposielajúcej vyrovnávacej pamäte a spustí časovač. Ak príde potvrdenie o prijatí tohto segmentu, kópia je vymazaná z vyrovnávacej pamäte. Ak potvrdenie nepríde pred vypršaním časovača, daný segment je preposlaný. Môže sa stať, že preposlaný segment príde až po doručení pôvodného segmentu. V takomto prípade je duplikát zahodený. Autori v [2] uvádzajú, že správne určiť, kedy časovač potvrdení vyprší, je dôležité, pretože to značne ovplyvňuje výkon TCP. Čas nesmie byť príliš krátky, ale zároveň je potrebné sa vyhnúť redundantným prenosom, ktoré ovplyvňujú celkový výkon systému. Nesmie byť však ani príliš veľký, pretože je potrebné vyhnúť sa prestojom pri čakaní na neexistujúce alebo stratené potvrdenie. Pri výbere hodnoty časovača je nevyhnutné brať do úvahy rýchlosť a spoľahlivosť fyzických liniek, ich dĺžku a množstvo ďalších faktorov. V TCP je táto hodnota definovaná podľa sofistikovaného adaptívneho algoritmu. Počas každého prenosu je meraný čas od momentu poslania segmentu až po doručenie potvrdenia

o prijatí. Tento čas sa nazýva *maximálny čas obehu* (Round Trip Time – RTT). Namerané hodnoty sú spriemerované pomocou váhových koeficientov, ktoré rastú s počtom meraní, aby boli brané do úvahy aj posledné merania. V praxi sa ukázalo, že hodnota koeficientu musí byť väčšia ako 2. V sieťach, kde sa značne líši RTT, sa pri výbere hodnoty časovača berie do úvahy aj rozptyl hodnôt RTT. [2]

Veľkosť okna prijímača závisí na dostupnosti veľkosti vyrovnávacej pamäte na prijímacej strane. Z tohto dôvodu má okno prijímača vo všeobecnosti rozdielnu veľkosť na každej strane spojenia. Napríklad, je logické očakávať, že server, ktorý má väčšiu vyrovnávaciu pamäť, pošle klientskej stanici väčšie okno než klientská stanica smerom k serveru. V závislosti od stavu siete môžu strany pravidelne meniť veľkosť okna prijímača. Zmenou veľkosti okna je možné ovplyvniť zaťaženie siete. Čím je okno väčšie, tým väčšie množstvo nepotvrdených dát je možné poslať do siete. Ak je množstvo dát väčšie než dokáže byť prijaté TCP entitou, tieto dáta budú zahodené. To bude viesť k nadmernému množstvu pokusov o preposlanie informácie a k neželanému zaťaženiu siete. Ak však bude veľkosť okna malá, môže limitovať dátový prenos na rýchlosť určenú časom potrebným pre každý segment vysielaný do siete. Aby sa vyhlo použitiu malých okien, niektoré TCP implementácie ponúkajú možnosť, že prijímač nezmení veľkosť okna, pokiaľ nebude k dispozícii 20% – 40% voľného miesta z maximálneho množstva pamäte určenej pre toto spojenie. Odosielateľ sa tiež nemusí ponáhľať s posielaním dát, pokiaľ veľkosť okna prijímacej strany nie je dostatočne veľká. Existujú dva algoritmy pre nastavenie veľkosti okna. Jeden spočíva v tom , že pri zostavení spojenia je určené veľké okno, ktoré sa postupne zmenšuje a v druhom je zas počiatočne nastavené malé okno, ktoré sa postupne zväčšuje, ak je sieť schopná zvládať toto zaťaženie. [2]

4.2.3 TCP vs. UDP

TCP je poverené komplikovanejšou a dôležitejšou úlohou zabezpečenia spoľahlivosti dátového prenosu cez prakticky nespoľahlivé siete. Na druhej strane, funkčná jednoduchosť protokolu UDP je dôvodom pre jednoduchosť algoritmu, malý objem réžijných dát a teda vysokú prevádzkovú rýchlosť. Preto niektoré aplikácie implementujú ich vlastné, dostatočne spoľahlivé mechanizmy spojovo-orientovaných správ, preferujú použitie menej spoľahlivých, ale rýchlejších nástrojov pre prenos dát v sieti. UDP je v porovnaní s TCP presne takýto nástroj. UDP dokáže byť tiež použité, ak vysoká kvalita komunikačnej linky zabezpečuje dostatočnú úroveň spoľahlivosti bez potreby zostavenia logického spojenia a posielania potvrdení o doručení. Prakticky okrem kontroly chýb pomocou kontrolného súčtu, UDP vlastne len dáta zapúzdri a pošle ďalej. Sústreďuje sa na čo najrýchlejšie odoslanie dát. Keďže UDP je sám o sebe nespoľahlivý protokol, spoľahlivosť prenosu závisí na samotných aplikáciách. Najčastejšie sa UDP využíva pre prenos informácií v DNS (Domain Name System), v SNMP (Simple Network Management Protocol) a v neposlednom rade pre prenos

hlasu VoIP (Voice over IP). To neznamená, že UDP nedokáže posielať a prijímať spoľahlivé prenosy, on ich len nedokáže garantovať. Keďže TCP je spojovo-orientované, nemôže byť použité na účely broadcastu alebo multicastu, ale používa sa na dátové prenosy, kde si nemôžeme dovoliť, aby nejaký segment chýbal a súbor by bol tým pádom nepoužiteľný, napríklad FTP (File Transfer Protocol) prenosy alebo e-mailový protokol SMTP (Simple Mail Transfer Protocol). Porovnanie protokolov je zhrnuté v tabuľke Tab. 4.1.[2][5]

Tab. 4.1: Porovnanie protokolov TCP a UDP [7]

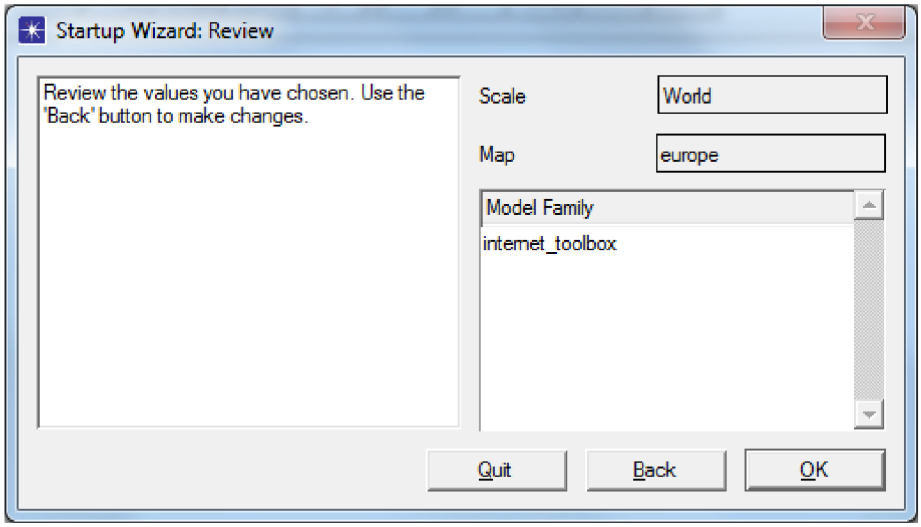
TCP	UDP
spoľahlivý	nespoľahlivý
spojovo - orientovaný	nespojovo - orientovaný
preposielanie segmentov a kontrola toku dát	opakovane nepreposiela dáta, ani nekontroluje dátový tok
pomocou veľkosti okna	
segmenty v správnom poradí	nezoradené datagramy
potvrdzovanie segmentov	bez potvrdzovania

5 LABORATÓRNA ÚLOHA: POROVNANIE TCP A UDP

Táto laboratórna úloha slúži na overenie znalostí o protokoloch transportnej vrstvy TCP a UDP. Je zameraná na porovnanie týchto dvoch protokolov z hľadiska časovej náročnosti vykonávaných úloh a taktiež aj z hľadiska náročnosti využitia sieťovej kapacity pri jednotlivých prenosoch.

5.1 Vytvorenie projektu

1. Spustíme OPNET IT Guru Academic Edition.
2. V menu zvolíme položku **File → New…**
3. Z ponuky vyberieme **Project** a potvrdíme tlačidlom **OK**. Položku **Project Name** zmeníme na **vaseVUTID_TCPvsUDP** a položku **Scenario Name** zmeníme na **TCP** a potvrdíme tlačidlom **OK**.
4. V okne **Startup Wizard: Initial Topology** zvolíme **Create Empty Scenario** a stlačíme **Next**.
5. V okne **Startup Wizard: Choose Network Scale** zvolíme **World** a stlačíme **Next**.
6. V okne **Startup Wizard: Choose Map** zvolíme **europe** a stlačíme **Next**.
7. V okne **Startup Wizard: Select Technologies** vyberieme **internet_toolbox** kliknutím na položku **Include?**, kde sa zmení stav z **No** na **Yes**. Klikneme na **Next**.
8. V okne **Startup Wizard: Review** vidíme rekapituláciu nami zvolených parametrov. Potvrdíme kliknutím na **OK**.



Obr. 5.1: Rekapitulácia zvolených parametrov

5.2 Vytvorenie a konfigurácia siete

V nasledujúcich krokoch vybudujeme systém klient – server pracujúci na sieti WAN (Wide Area Network).

9. Po potvrdení okna **Startup Wizard: Review** sa nám otvorí mapa projektu a paleta objektov **Object Palette**. V prípade, že by sa nám paleta neotvorila alebo by sme si ju omylom zatvorili, opätovne sa k nej dostaneme kliknutím na tlačidlo  . Pre náš projekt budeme potrebovať ešte jeden komponent, ktorý sa nenachádza v modelovej rodine **internet_toolbox**.

10. V palete objektov klikneme na **Configure Palette…**. V **Node Models** vyhľadáme objekt **ppp_wkstn_adv** a vyberieme ho kliknutím na **Status**. Po kliknutí sa zmení stav z **not included** na **included**. Dvomi kliknutiami na **OK** potvrdíme. Program bude vyžadovať uloženie vykonaných zmien, potvrdíme tlačidlom **OK.** V priebehu práce je dôležité ukladať si projekt, aby sme nestratili vykonané zmeny

11. Z palety objektov zvolíme objekt **ip32_cloud** a umiestnime ho na pracovnú plochu IT Guru (prostredie so štvorcami a mapou). Objekt ip32_cloud predstavuje IP cloud (oblak) reprezentujúci model, ktorý podporuje až 32 sériových liniek, cez ktoré je možné modelovať IP prevádzku.

12. Pravým kliknutím na objekt **ip32_cloud** pomocou **Set Name** zmeníme meno na **ip32 cloud**. Potvrdíme kliknutím na **OK**.

13. Z palety objektov vyberieme objekt **ppp_wkstn_adv**, ktorý sme tam predchvíľou pridali, a umiestnime ho na pracovnú plochu IT Guru. Tento objekt reprezentuje pracovnú stanicu s bežiacimi aplikáciami klient – server cez protokoly TCP a UDP.

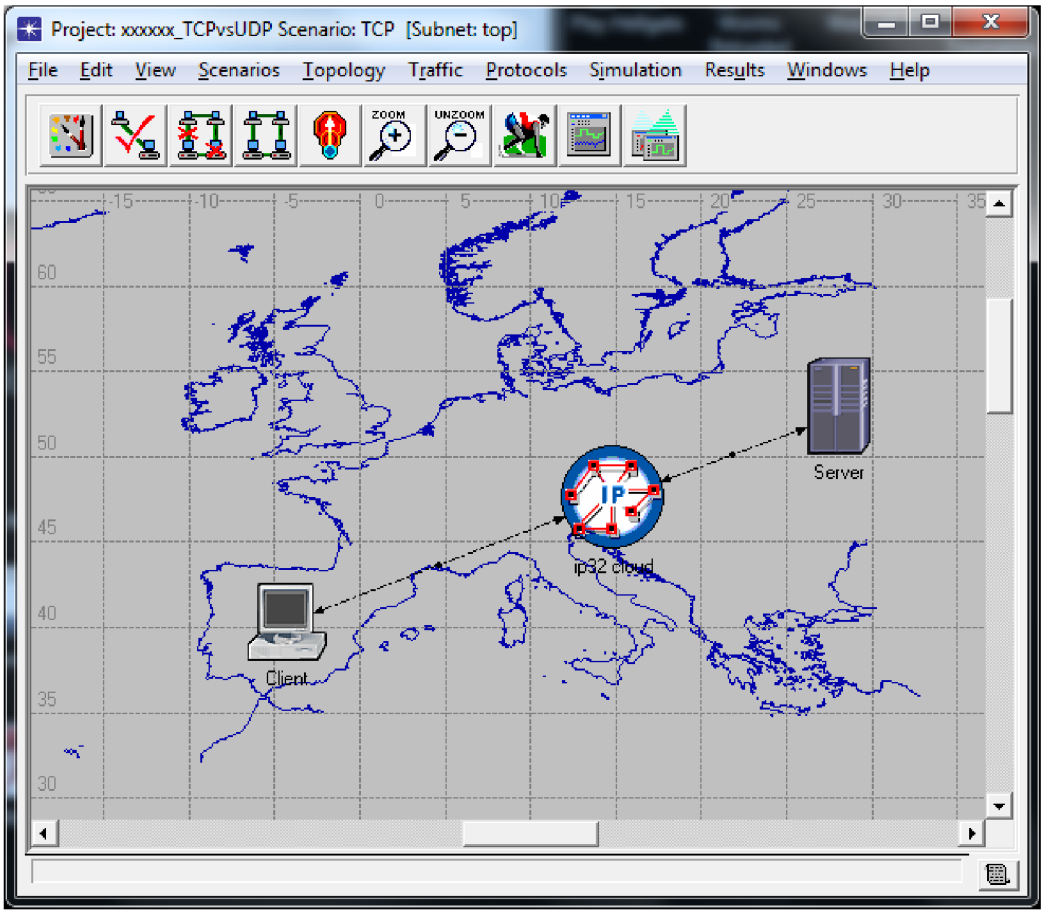
14. Pravým tlačidlom klikneme na objekt pracovnej stanice a vyberieme **Edit Attributes**. Atribút **name** zmeníme na **Client** a atribút **Client Address** tiež na **Client**. Potvrdíme tlačidlom **OK**.

15. V palete objektov ďalej vyberieme objekt **ppp_server** a umiestnime ho na pracovnú plochu IT Guru. Tento prvok reprezentuje server s aplikáciami bežiacimi cez TCP a UDP.

16. Pravým tlačidlom klikneme na objekt server a vyberieme **Edit Attributes**. Atribút **name** zmeníme na **Server** a atribút **Server Address** tiež na **Server**.

17. Zvolíme prenosové médium **PPP_DS1** a prepojíme **Client** s **ip32_cloud** a taktiež **ip32_cloud** so **Serverom**. Protokol PPP (Point-to-Point Protocol) sa používa na dlhé vzdialenosti a linka DS1, známa aj ako T1, má rýchlosť 1,544 Mb/s.

18. Klikneme pravým tlačidlom na linku medzi **Client** a **ip32_cloud** a vyberieme **Advanced edit attributes**. Rozklikneme riadok **Background utilization** a ďalej rozklikneme **row 0**. Do riadku **time (sec)** zadáme hodnotu **300** a do riadku **background utilization (%)** zadáme **60**. Zmenenú hodnotu potvrdíme tlačidlom Enter a celkové nastavenie potvrdíme kliknutím na **OK**. Rovnaký postup zopakujeme aj pre linku medzi **ip32_cloud** a **Server**.



Obr. 5.2: Vytvorená topológia

Teraz je potrebné vytvoriť aplikáciu, ktorá bude vedieť bežať aj na TCP, aj na UDP. Aplikácia bude vyžadovať, aby klient posielal jednu požiadavku (request) smerom k serveru, ktorý mu na to bude odpovedať jedinou odpoveďou (response). Na tento účel vytvoríme špeciálnu úlohu, ktorá bude definovať aplikáciu. Túto aplikáciu neskôr použijeme v profile, ktorý prepojíme s klientom a serverom.

19. Z palety objektov vyberieme **Task Config** a umiestnime ho na pracovnú plochu IT Guru.

20. Pravým tlačidlom vyberieme **Edit Attributes**. Meno zmeníme na **Task**.

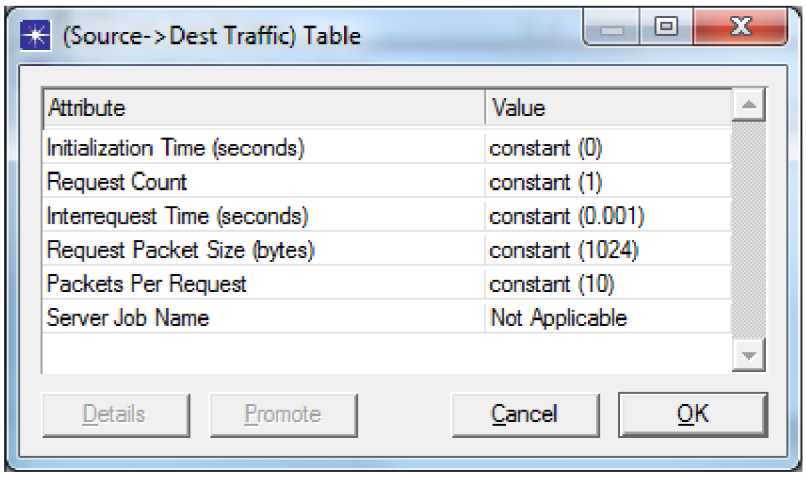
21. Rozklikneme položku **Task Specification**. Klikneme na **0** v riadku **rows** a zmeníme ju na **1**. Tým pridáme riadok.

22. Rozklikneme **row 0** a zmeníme **Task Name** na **Request Response Task**

23. Ďalej zmeníme položku **Manual Configuration**. Počet riadkov zmeníme na **1**. Položku **Phase Name** zmeníme na **Only Phase** a potvrdíme kliknutím na **OK**.

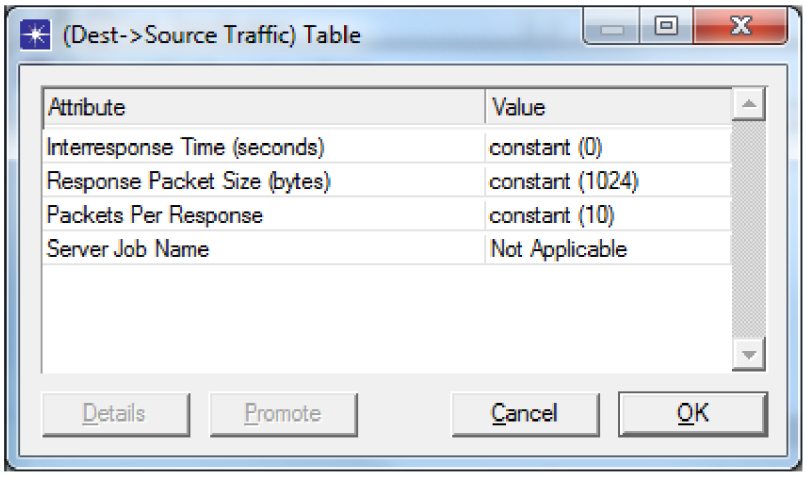
24. Položku **Source** zmeníme na **Client** a položku **Destination** na **Server**.

25. V poli **Source → Dest Traffic** nastavíme **Initialization Time** na **constant (0)**, **Request Count** na **constant (1)**, **Interrequest Time (seconds)** na **constant (0.001)** a **Packets per Request** na **constant (10)**. Potvrdíme kliknutím na **OK**.



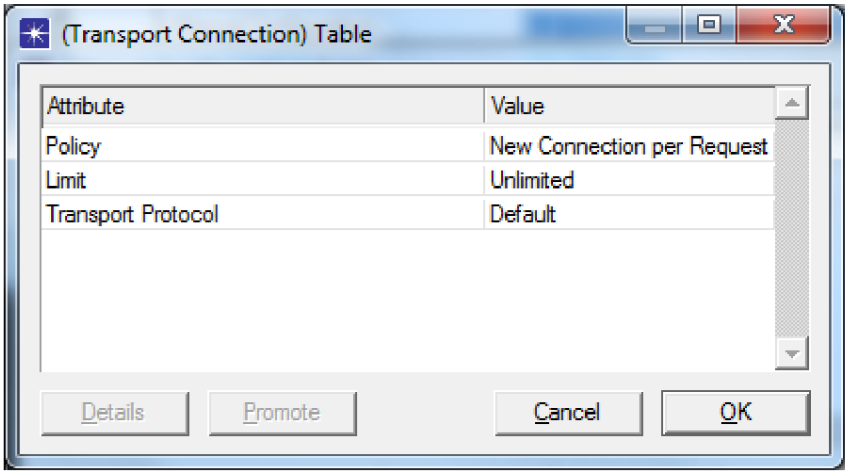
Obr. 5.3: Nastavenia Source → Dest Traffic v konfigurácii úlohy

26. V poli **Dest → Source Traffic** nastavíme **Interresponse Time** na **constant (0)** a klikneme na **OK**. To spôsobí, že server bude na požiadavku odpovedať okamžite jediným paketom.

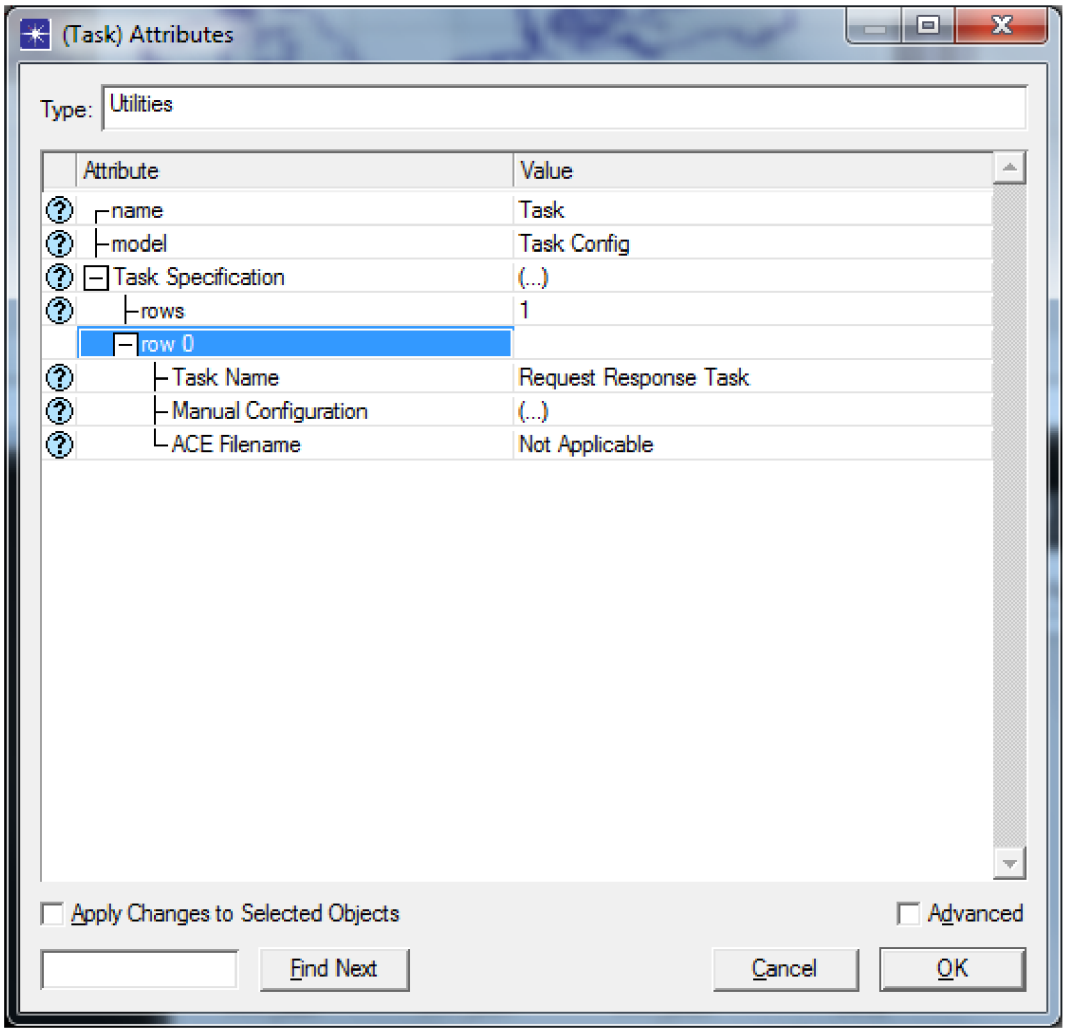


Obr. 5.4: Nastavenia Dest → Source Traffic v konfigurácii úlohy

27. V poli **Transport Connection** nastavíme **Policy** na **New Connection per Request**. Konfiguráciu trikrát potvrdíme tlačidlom **OK**.



Obr. 5.5: Nastavenie prenosových vlastností v konfigurácii úlohy



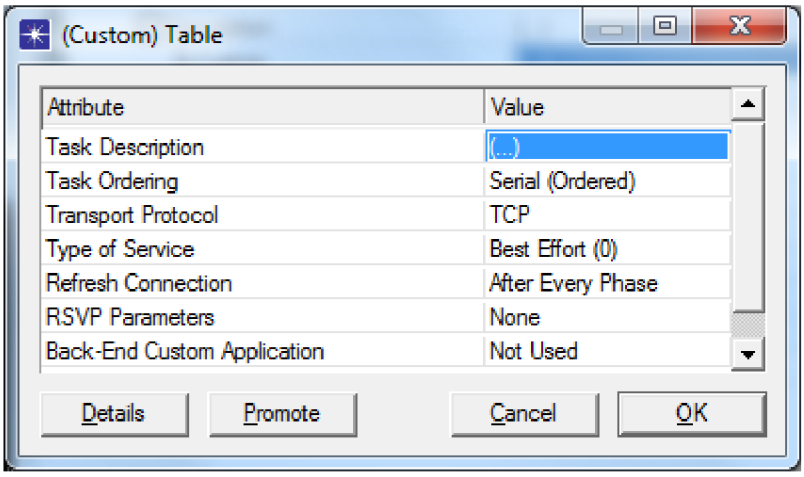
Obr. 5.6: Prehľad nastavení úlohy

28. Na pracovnej ploche sa nám po potvrdení nastavenia úlohy objavil nový objekt **Application Config**. Klikneme naň pravým tlačidlom a zvolíme **Edit Attributes**. Meno zmeníme na **Application**.

29. Rozklikneme riadok **Application Definitions** a počet riadkov zmeníme na **1**.

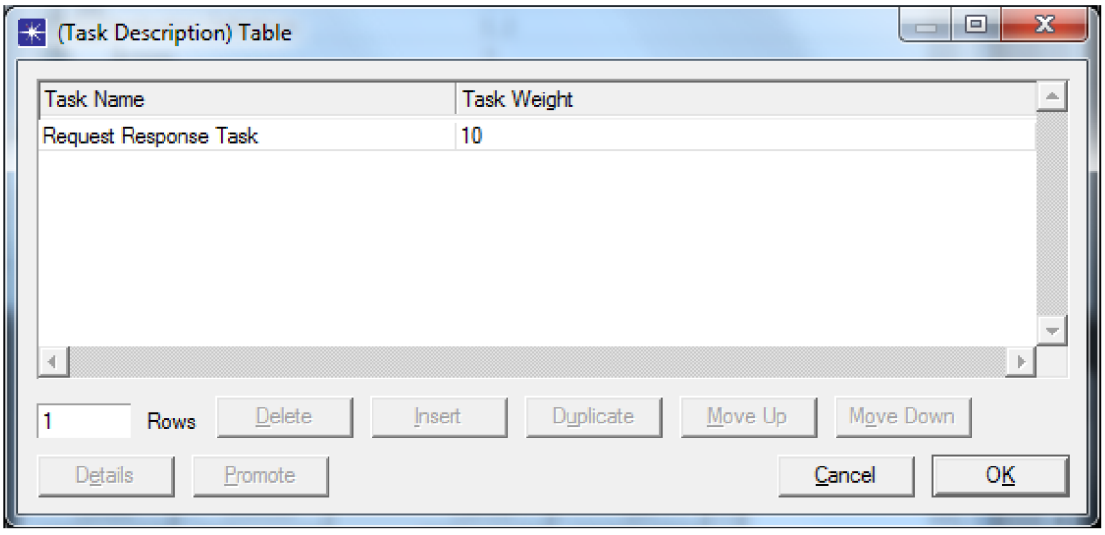
30. Rozklikneme **row 0** a zmeníme položku **Name** na **Request Response Application**.

31. Rozklikneme riadok **Description** a upravíme riadok **Custom**. Transportný protokol je nastavený na TCP, čo sme potrebovali, takže ten zatiaľ meniť nebudeme.

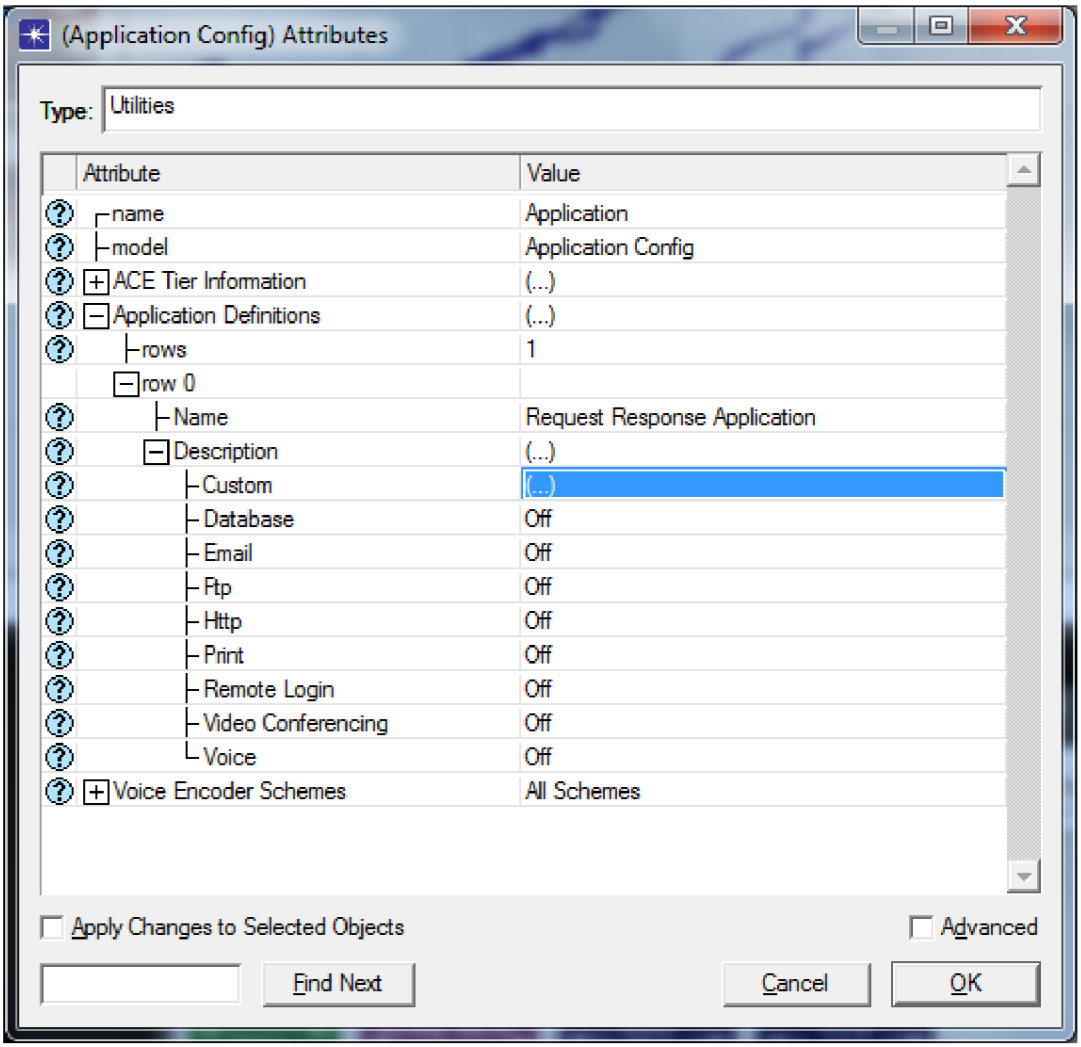


Obr. 5.7: Nastavenie protokolu TCP v konfigurácii aplikácie

32. Upravíme riadok **Task Description**, kde pridáme jeden riadok a **Task Name** upravíme na **Request Response Task**. Konfiguráciu trikrát potvrdíme tlačidlom **OK**.



Obr. 5.8: Nastavenie úlohy v konfigurácii aplikácie



Obr. 5.9: Prehľad nastavení aplikácie

33. Z palety objektov vyberieme objekt **Profile Config** a umiestnime ho na pracovnú plochu IT Guru. Klikneme naň pravým tlačidlom a zvolíme **Edit Attributes**. Meno zmeníme na **Profile**.

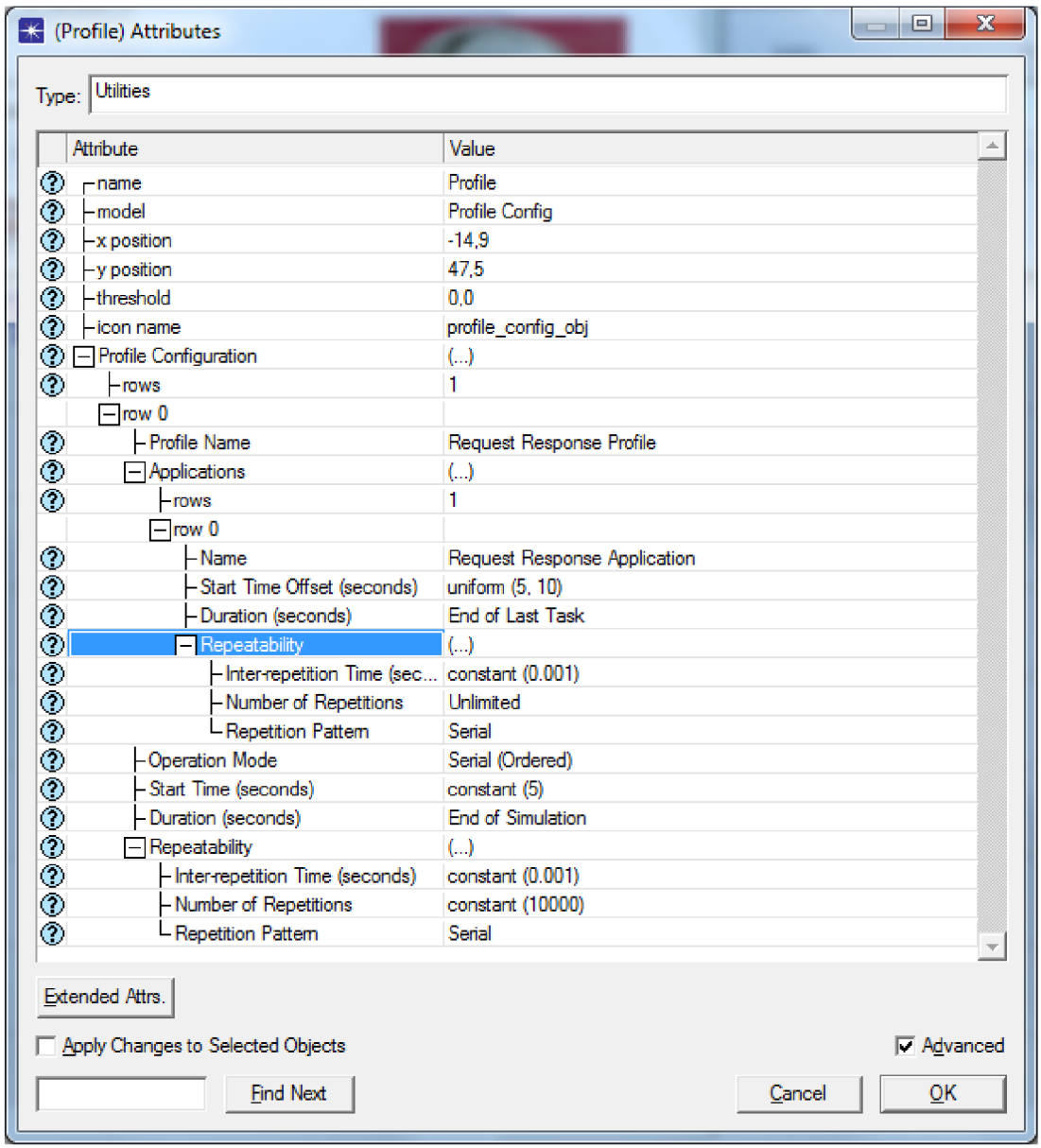
34. V riadku **Profile Configuration** pridáme riadok a meno profilu **Profile Name** zmeníme na **Request Response Profile**.

35. Rozklikneme riadok **Applications** a pridáme jeden riadok. Rozklikneme ďalej **row 0** a zmeníme meno aplikácie na **Request Response Application**.

36. Položku **Duration (seconds)** zmeníme na **End of Last Task**.

37. Rozklikneme riadok **Repeatability** súvisiaci s riadkom **row 0** a nastavíme **Inter-repetition time (seconds)** na **constant (0.001)**. Položku **Start Time (seconds)** na **constant (5)**.

38. Rozklikneme druhý **Repeatability** riadok a nastavíme **Inter-repetition Time (seconds)** na **constant (0.001)** a **Number of Repetitions** na **constant (10000)**. Potvrdíme tlačidlom **OK**.

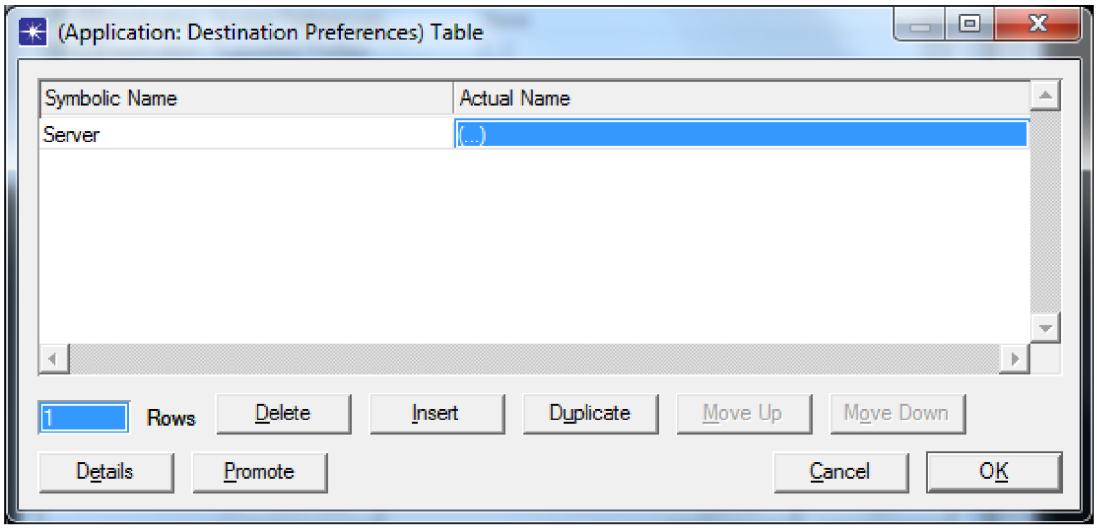


Obr. 5.10: Prehľad nastavení profilu

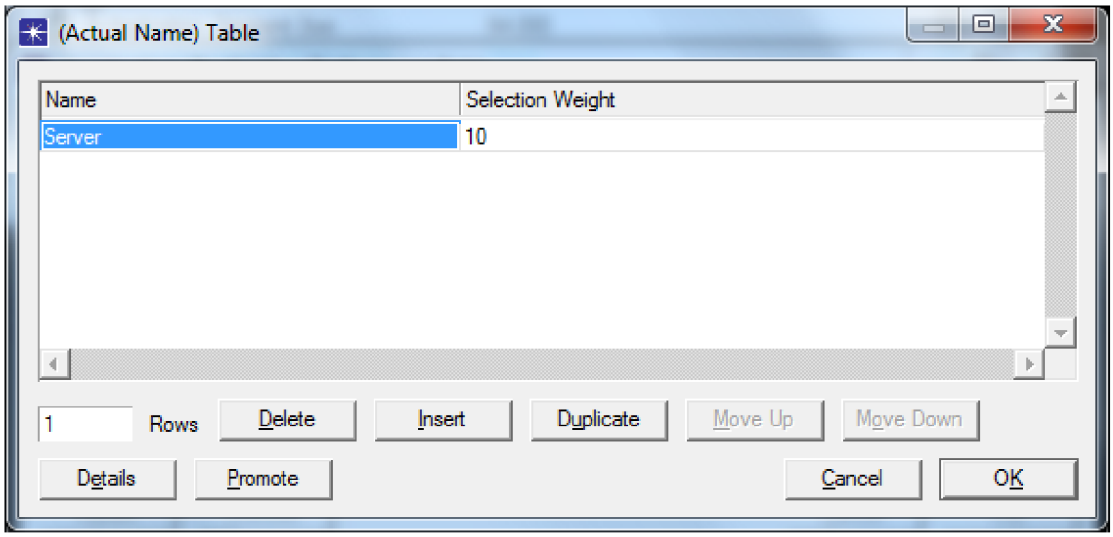
Teraz prepojíme aplikáciu s klientom a so serverom.

39. Pravým tlačidlom klikneme na objekt **Client** a zvolíme **Edit Attributes**. Budeme upravovať riadok **Application: Supported Profiles**. Pridáme riadok a rozklikneme **row 0**. Meno profilu zmeníme na **Request Response Profile**.

40. Ďalej zmeníme **Application: Destination Preferences**. Pridáme riadok a v poli **Symbolic Name** vyberieme **Server**. V poli **Actual Name** pridáme riadok a zmeníme meno na **Server**.. Dvomi klikmi na **OK** potvrdíme.

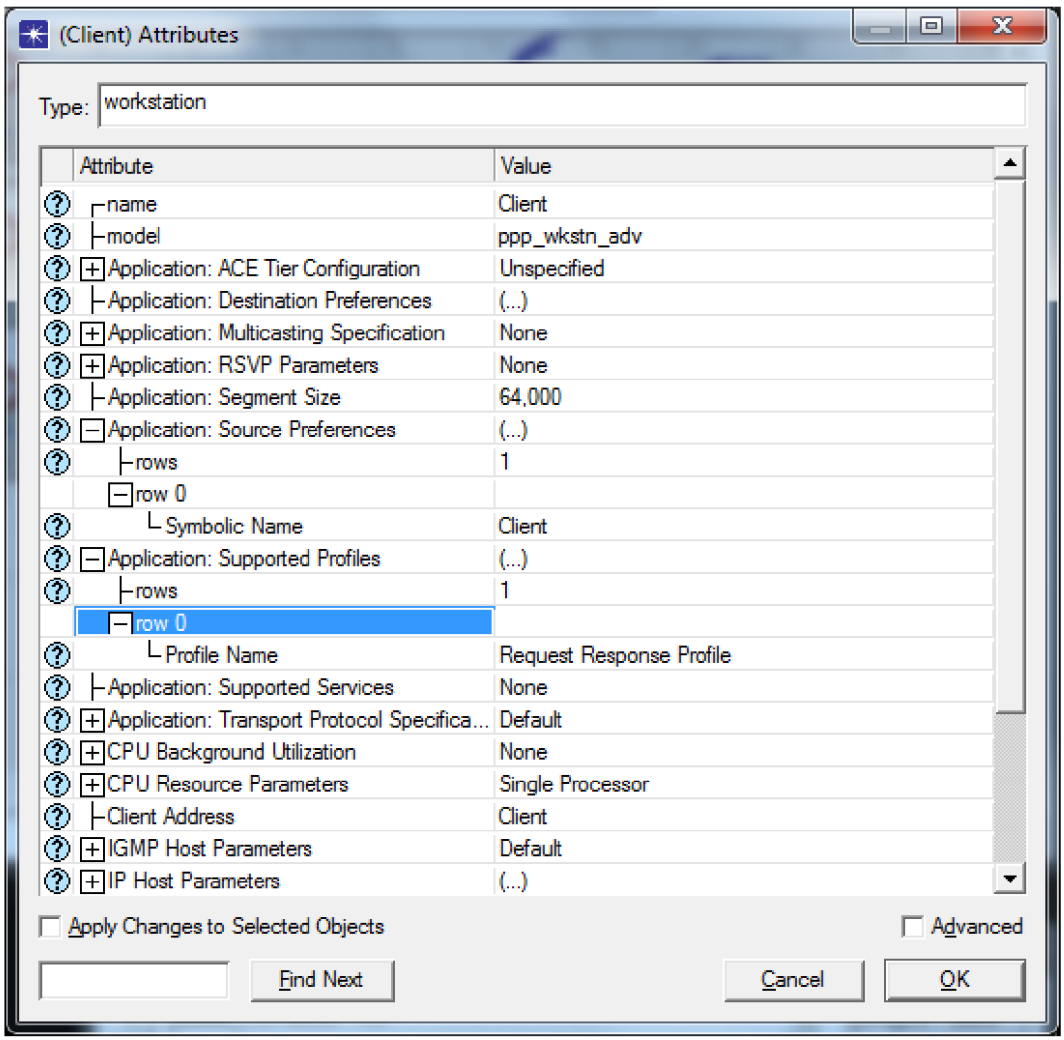


Obr. 5.11: Prehľad nastavenia cieľovej stanice v konfigurácii klienta



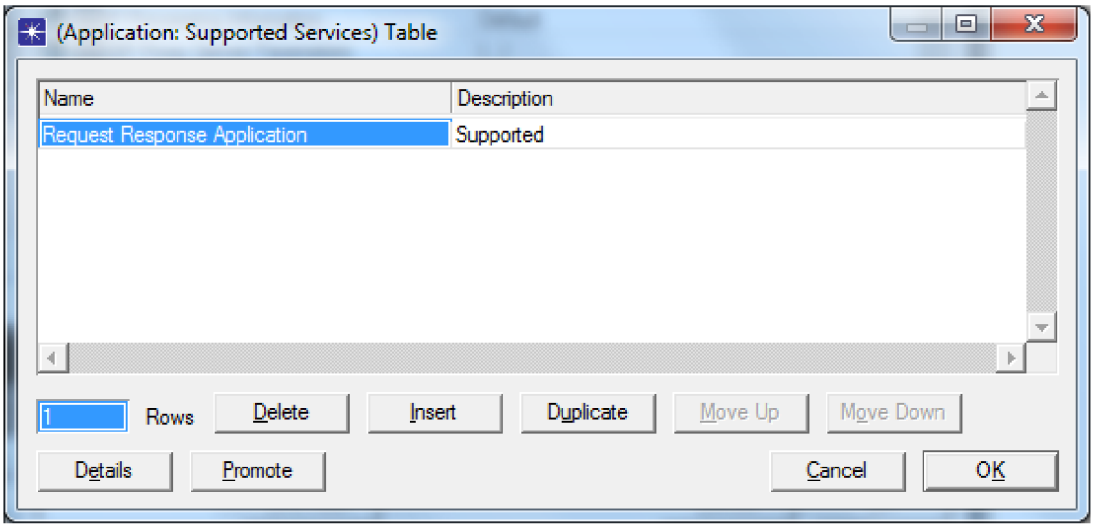
Obr. 5.12: Prehľad nastavenia cieľovej stanice v konfigurácii klienta

41. Ďalej ešte zmeníme **Application: Source Preferences**, kde pridáme riadok, rozklikneme **row 0** a zmeníme **Symbolic Name** na **Client**. Ak by sa nám z nejakého dôvodu nepodarilo zmeniť **Symbolic Name**, potvrdíme aktuálne nastavenia pre **Client** a následne znova klikneme pravým tlačidlom na objekt **Client**. Vyberieme **Edit Attributes** a opätovne sa pokúsime zmeniť **Symbolic Name** na **Client**.

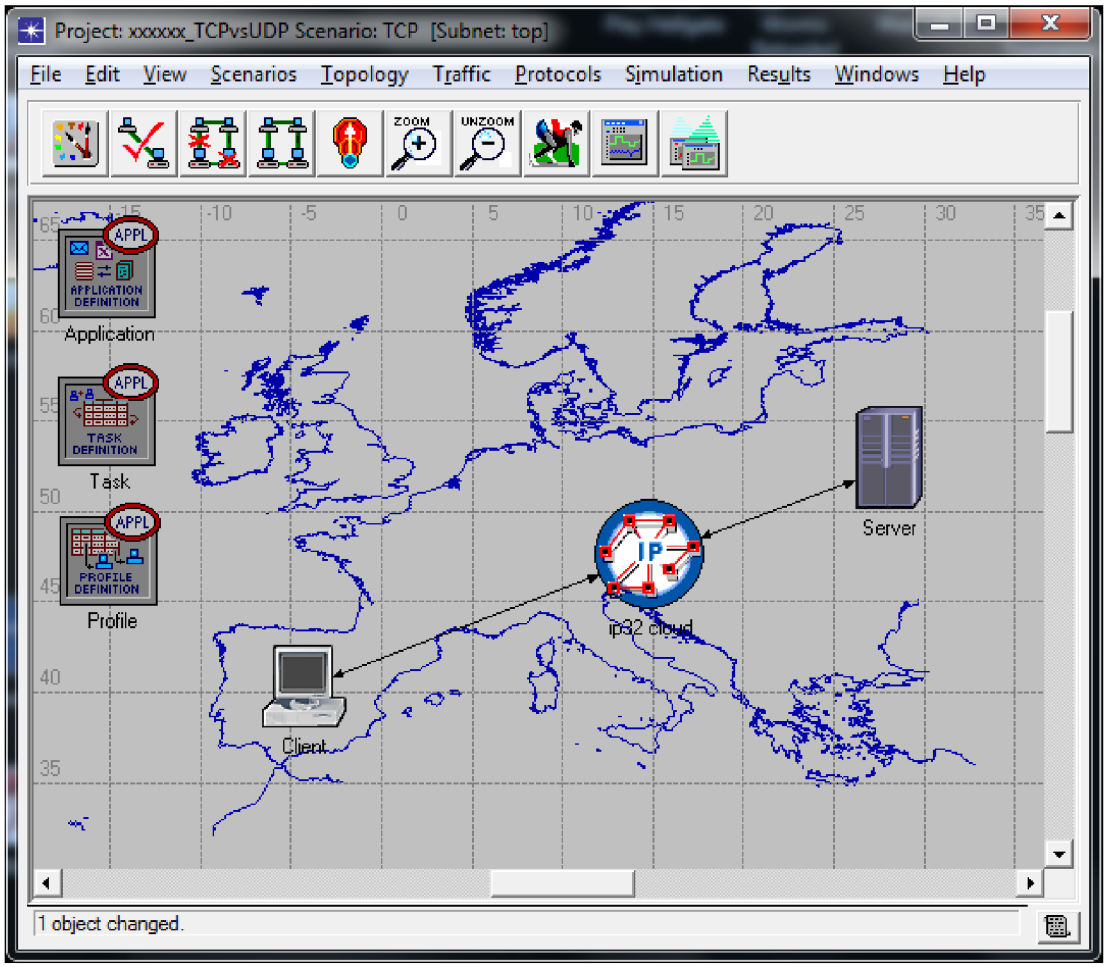


Obr. 5.13: Prehľad nastavení klienta

42. Klikneme pravým tlačidlom na objekt **Server** a zvolíme **Edit Attributes**. Zmeníme **Application: Supported Services**. Pridáme riadok a zmeníme meno na **Request Response Application**. Potvrdíme dvomi klikmi na **OK**.



Obr. 5.14: Nastavenie podporovaných služieb v konfigurácii serveru



Obr. 5.15: Výsledná topológia projektu

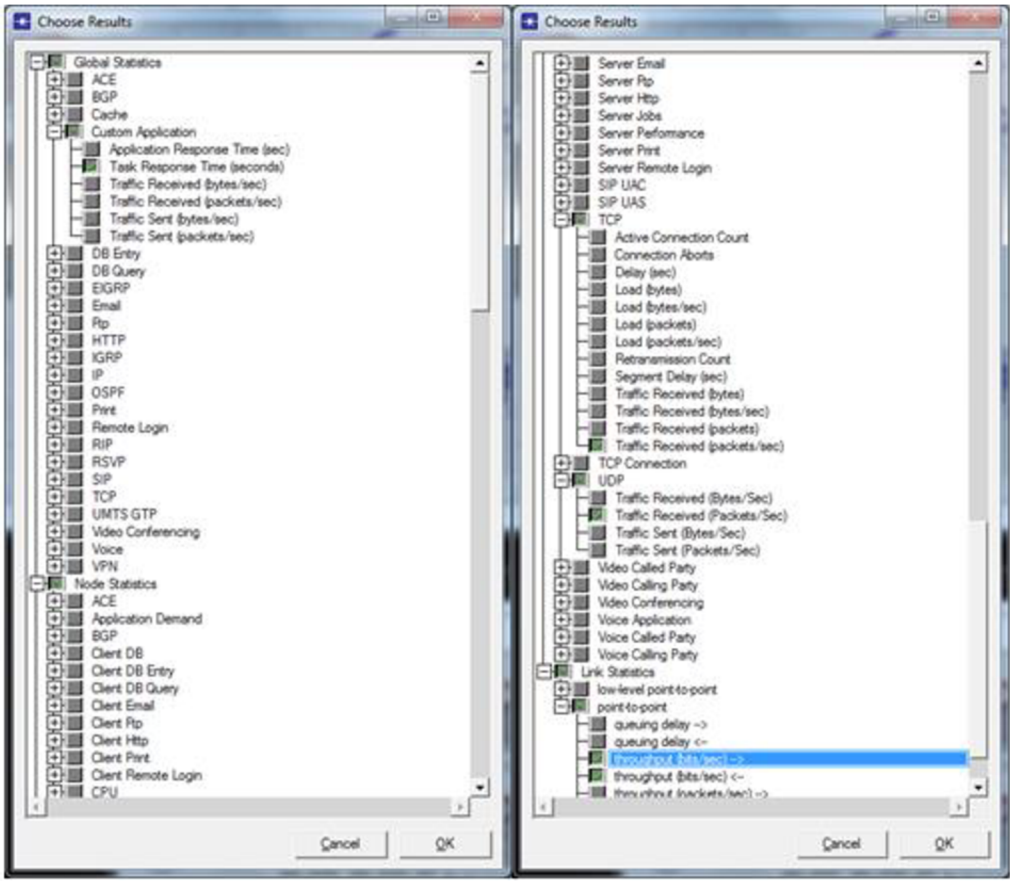
5.3 Nastavenie simulácie

43. V menu zvolíme **Simulation → Choose Individual Statistics...**.

44. Rozklikneme **Global Statistics**, kde rozklikneme **Custom Application** a zaškrtneme **Task Response Time (Seconds).**

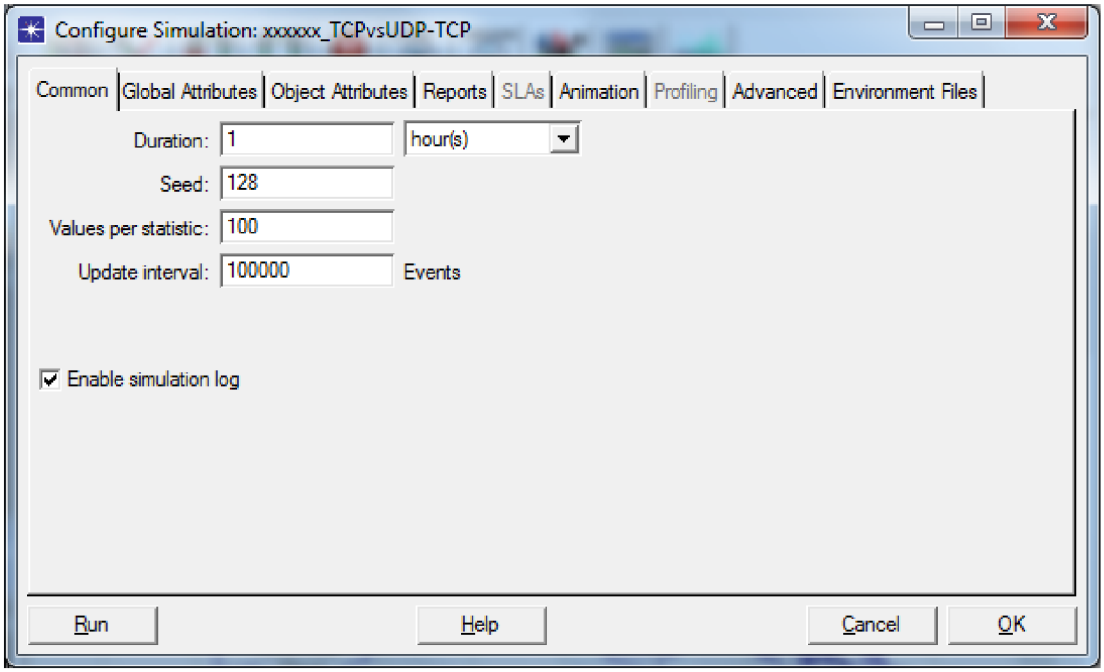
45. Rozklikneme **Node Statistics**, možnosť **IP** a zaškrtneme **Traffic Received (packets/sec)** a **Traffic Sent (packets/sec)**. V možnosti **TCP** aj v možnosti **UDP** zaškrtneme **Traffic Received (packets/sec).**

46. Rozklikneme **Link Statistics**, možnosť **point-to-point** a zaškrtneme **throughput (packet/sec) →** a **throughput (packet/sec) ←**. Kliknutím na **OK** potvrdíme nami vybrané štatistiky.



Obr. 5.16: Prehľad vybraných štatistík

47. V menu zvolíme **Simulation → Configure Discrete Event Simulation...**. V záložke **Common** nastavíme **Duration** na 1 hodinu a potvrdíme tlačidlom **OK**.



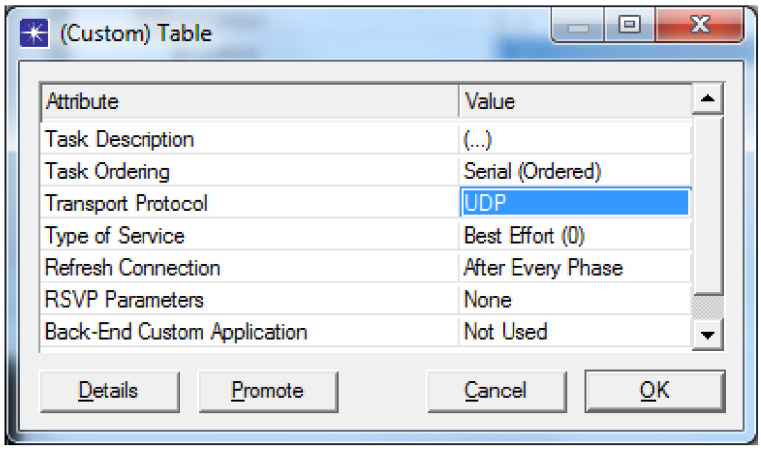
Obr. 5.17: Nastavenie simulácie

5.4 Duplikácia scenára

V tomto momente máme náš projekt nastavený pre TCP, aby sme však mohli uskutočniť porovnanie protokolov, potrebujeme si vyrobiť scenár s UDP.

48. V menu klikneme na **Scenarios → Duplicate Scenarios...**. Ďalší scenár pomenujeme **UDP** a potvrdíme tlačidlom **OK**. Medzi jednotlivými scenármi je možné sa prepínať pomocou **Scenarios → Switch To Scenario** alebo pomocou klávesových skratiek **CTRL + 1**, **CTRL + 2**, atď.

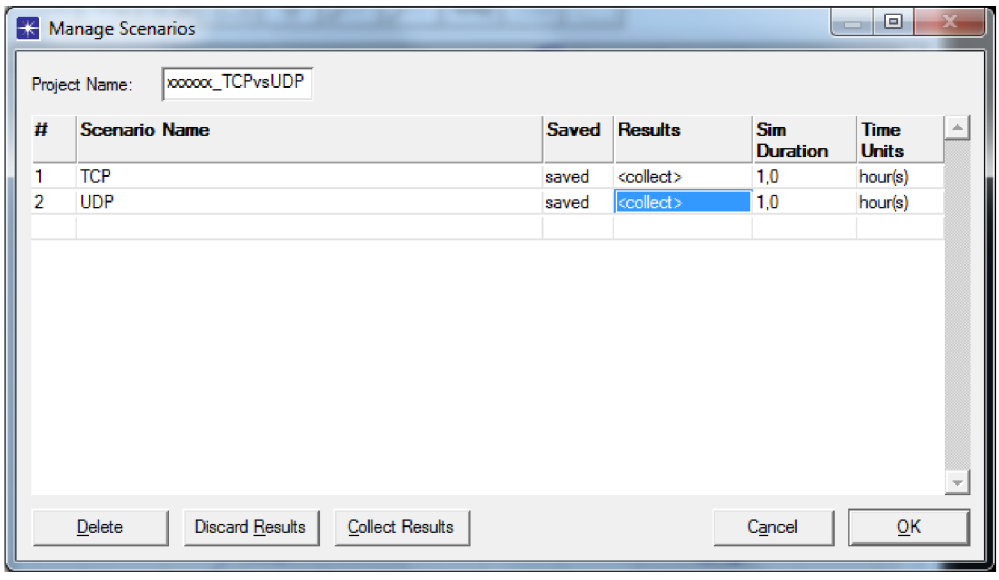
49. V nastaveniach aplikácie potrebujeme zmeniť protokol na UDP. Klikneme pravým tlačidlom na objekt **Application** a vyberieme **Edit Attributes**. Rozklikneme **Application Definitions → row 0 → Description** a v položke **Custom** zmeníme **Transport Protocol** na **UDP**. Dvomi klikmi na **OK** potvrdíme.



Obr. 5.18: Nastavenie protokolu UDP v konfigurácii aplikácie

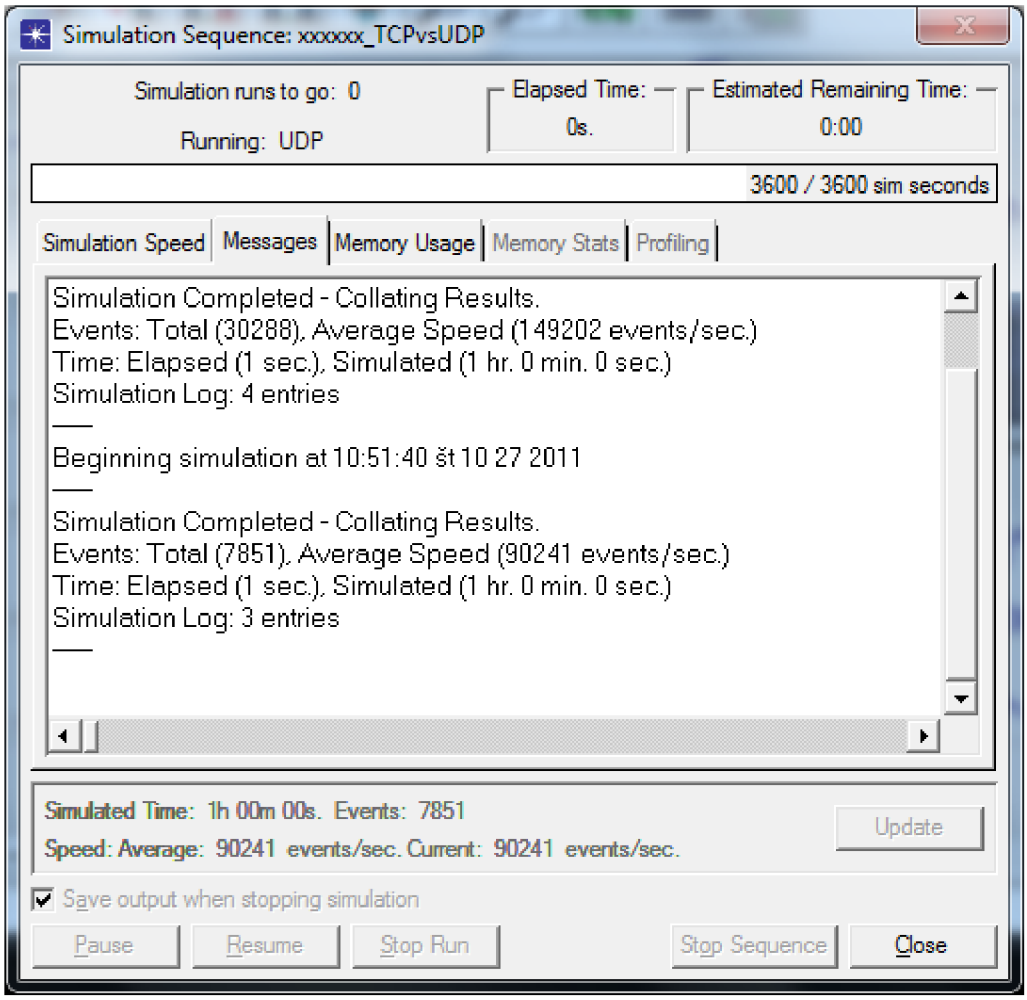
5.5 Spustenie simulácie

50. V menu klikneme na **Scenarios → Manage Scenarios...**. V poli **Results** zmeníme hodnotu na **<collect>**. Po potvrdení tlačidlom **OK** prebehne simulácia obidvoch scenárov.



Obr. 5.19: Prehľad scenárov

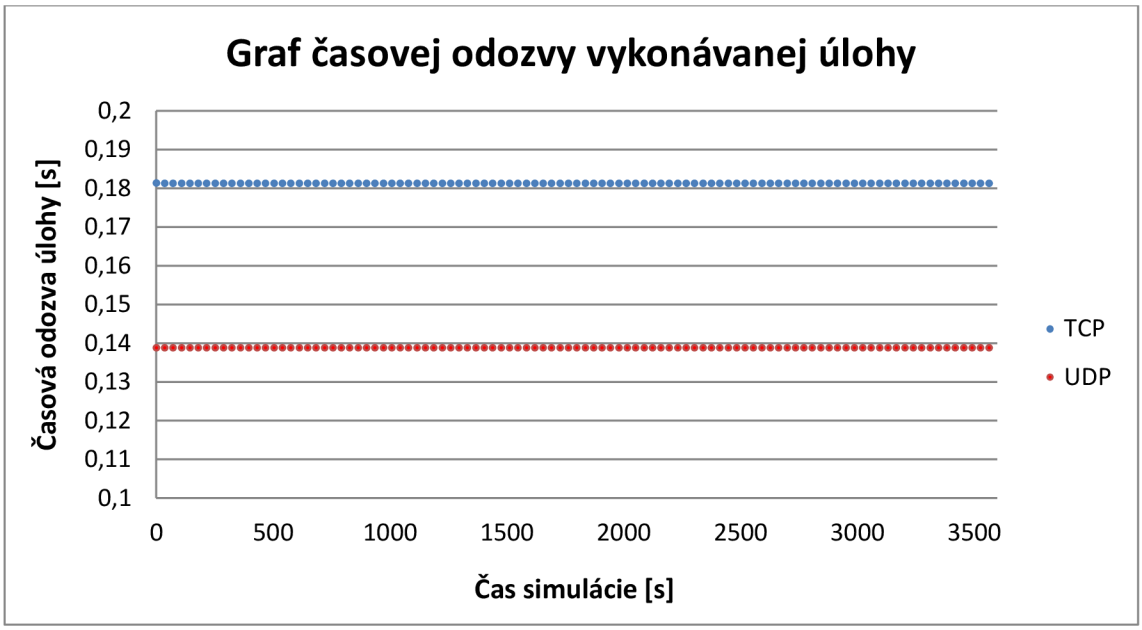
51. Keď simulácia prebehne, môžeme zavrieť okno **Simulation Sequence**.



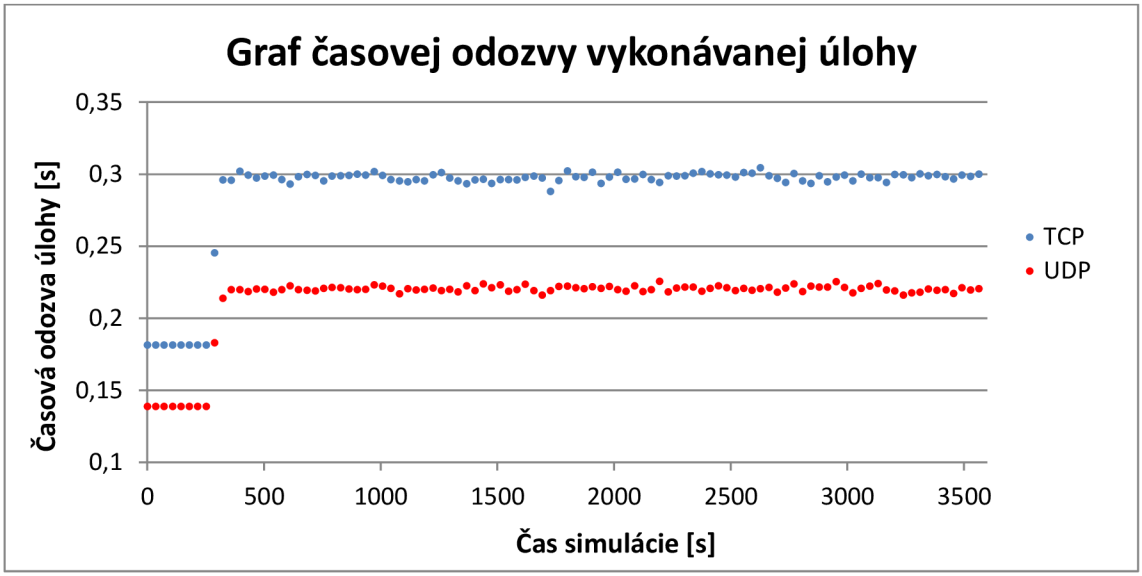
Obr. 5.20: Okno simulácie

5.6 Zobrazenie výsledkov

52. Výsledky simulácie budeme zobrazovať pomocou **Results → Compare Results**. Rozklikneme **Global Statistics → Custom Application** a zaškrtneme **Task Response Time (seconds)** a zvolíme mód **average**. Táto položka vyjadruje, ako dlho trvalo vykonanie každej z úloh. Po kliknutí na tlačidlo **Show** sa nám vykreslí graf v samostatnom okne. V grafe vidíme, že aplikácii používajúcej TCP trvalo splnenie úlohy takmer dvakrát tak dlho než aplikácii používajúcej UDP.



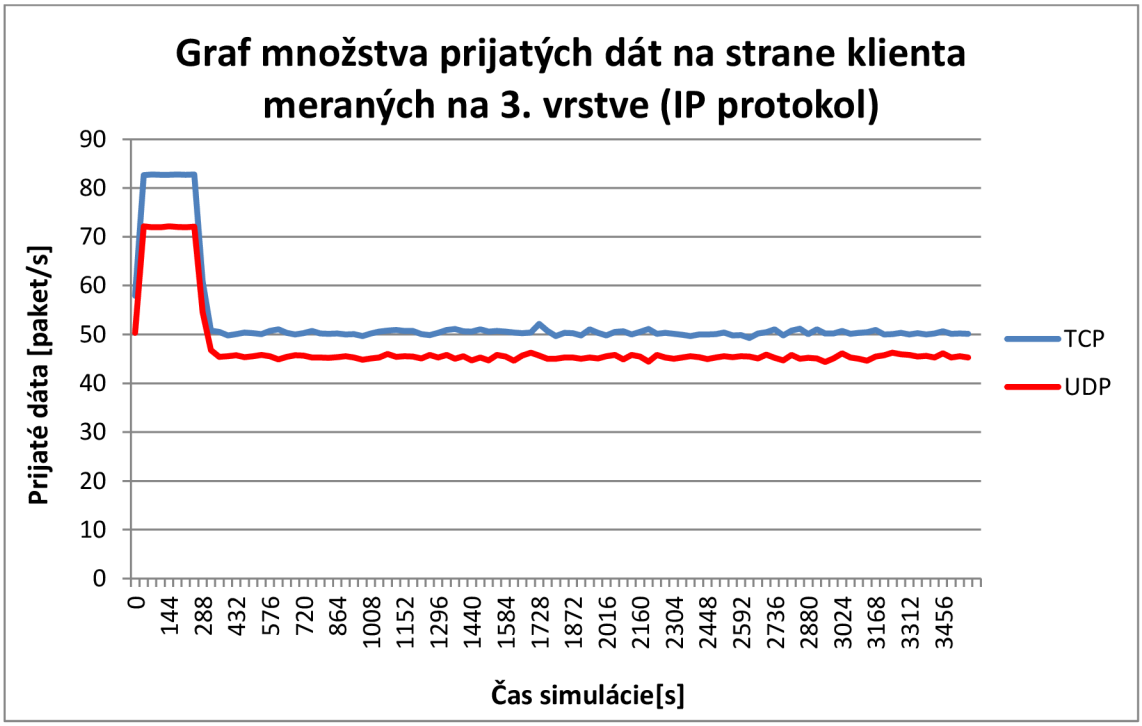
Obr. 5.21: Graf času odozvy vykonávanej úlohy bez zaťaženia linky na pozadí po exporte do MS Excel



Obr. 5.22: Graf času odozvy vykonávanej úlohy so zaťažením linky na pozadí po exporte do MS Excel

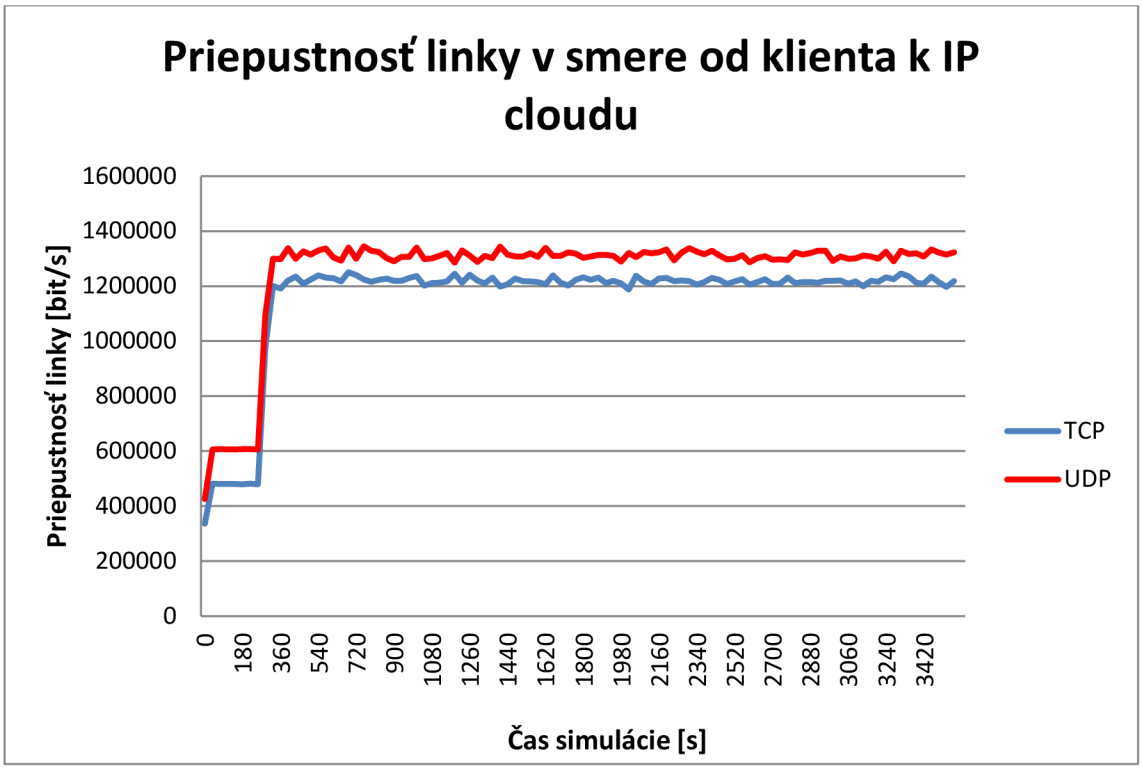
53. Rozklikneme **Object Statistics → Client → IP**. Zaškrtneme **Traffic Received (packets/sec)** a zvolíme mód **As is**. Klikneme na **Show**. V grafe vidíme, že klient prijal omnoho viac paketov za sekundu v prípade TCP, než v prípade

UDP. Dôvodom je práve potreba budovať a následne rušiť dátové spojenia u TCP.



Obr. 5.23: Graf množstva prijatých dát na strane klienta po exporte do MS Excel (spriemerované hodnoty)

54. Rozklikneme **Client** ↔ **ip32_cloud[0]** → **point-to-point** a zaškrtneme **throughput (bits/sec)** →. Nastavíme mód **As is** a klikneme na **Show**. Tento graf nám ukazuje koľko dát (správ) bolo úspešne prenesených sieťou v smere od klienta k ip32_cloud.



Obr. 5.24: Graf priepustnosti linky medzi klientom a IP cloudom po exporte do MS Excel

5.7 Otázky

1. Vysvetlite, ako prebieha vybudovanie a ukončenie spojenia v protokole TCP?

2. Zdôvodnite, ako sa zmení čas odozvy TCP a UDP, keď bude klient smerom k serveru posielať 10 paketov (request) a server mu bude ne odpovedať 100 paketmi (response).

   Zduplikujeme scenár **TCP** a pomenujeme ho popisne, napríklad **TCP_big_response**. V objekte **Task** zmeníme **Dest → Source Traffic**. Položku **Packets per Response** upravíme na **constant (100)**. Ďalej zduplikujeme scenár **UDP** a taktiež ho pomenujeme popisne, napríklad **UDP_big_response**. Zmeníme počet paketov odpovede ako pri TCP. Necháme prebehnúť simuláciu týchto dvoch scenárov a porovnáme výsledky.

   Koľkokrát je časová odozva TCP pomalšia než časová odozva UDP?

3. Doteraz sme uvažovali, že prenos v sieti bol bezchybný, v skutočnosti je však riziko straty dát vždy prítomné. Ako ovplyvní stratovosť čas odozvy protokolov TCP a UDP?

Zduplikujeme scenár **TCP** a pomenujeme ho popisne, napríklad **TCP_packets_discarded**. V objekte **ip32_cloud** zmeníme hodnotu **Packet Discard Ratio** na **5%**. To isté urobíme aj pre **UDP**. Necháme prebehnúť simuláciu a porovnáme výsledky.

Výsledky zdôvodnite.

6 HODNOTENIE DOSIAHNUTÝCH VÝSLEDKOV – TCP/IP

V simulovanej laboratórnej úlohe sme porovnávali rozdiel medzi protokolmi transportnej vrstvy – TCP a UDP. Vytvorili sme si topológiu pozostávajúcu z pracovnej stanice – klienta, servera a objektu ip32_cloud, ktorý nahrádza komplexnosť siete smerovačov a iných sieťových zariadení a to až na 32 sériových linkách. Po vytvorení topológie sme si vytvorili aplikáciu, ktorá zvláda aj prevádzku TCP aj prevádzku UDP. Definovali sme si taktiež objekt, ktorého účelom bolo vytvoriť úlohu (task), ktorá definuje danú aplikáciu. Túto aplikáciu sme prepojili s profilom. Profil využíva aplikáciu na prepojenie s klientom a serverom.

V simulácii sme sledovali časovú odozvu spojenia s použitými protokolmi TCP a UDP. Na Obr. 5.21 a Obr. 5.22 vidíme, že aplikácii používajúcej TCP trvalo splnenie danej úlohy takmer raz toľko, než aplikácii používajúcej UDP, a to z dôvodu potreby potvrdzovania doručenia TCP segmentov.

Ďalej sme sledovali množstvo prenesených dát pri jednotlivých prenosoch. Na Obr. 5.23 vidíme, že klient prijal omnoho viac paketov za sekundu v prípade TCP, než v prípade UDP, a to z dôvodu potreby budovania, rušenia spojenia a v neposlednom rade z dôvodu zasielania potvrdení o prenose.

Sledovali sme aj prevádzku v rámci siete WAN, kde sme sledovali priepustnosť linky. Na Obr. 5.24 vidíme, že UDP úspešne prenieslo viac paketov za sekundu než TCP a to z toho dôvodu, že UDP nemusí riešiť spojovaciu/ukončovaciu procedúru akú má TCP.

6.1 Odpovede na otázky z laboratórnej úlohy

1. **Vysvetlite, ako prebieha vybudovanie a ukončenie spojenia v protokole TCP?**

Budovanie spojenia v protokole TCP prebieha nasledovne:

1) Zariadenie A odošle segment SYN smerom k zariadeniu B, čím sa začína budovanie spojenia.
2) Zariadenie B potvrdí segmentom ACK, že obdržal segment SYN od zariadenia A a zároveň odošle segment SYN smerom k zariadeniu A.
3) Zariadenie A odošle segment ACK smerom k zariadeniu B, aby sa potvrdilo, že obdržal jeho SYN segment.

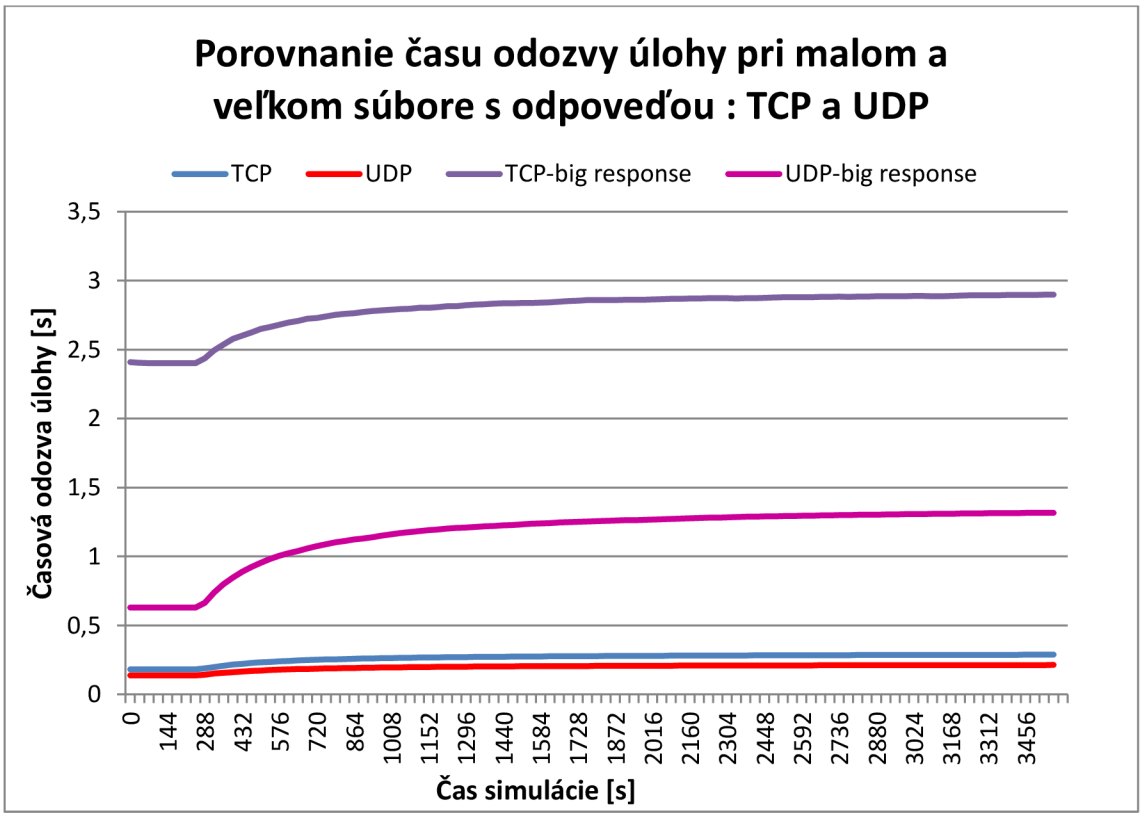
Ďalej už prebieha samotný prenos dát.

Rušenie spojenia v protokole TCP prebieha nasledovne:

1) Zariadenie A, ktoré chce ukončiť spojenie, odošle segment FIN smerom k zariadeniu B.
2) Zariadenie B pošle segment ACK, čím potvrdí, že obdržal segment FIN od zariadenia A a zároveň tiež posiela segment FIN smerom k zariadeniu A.
3) Zariadenie A odošle ACK segment smerom k zariadeniu a spojenie je týmto ukončené.

2. Zdôvodnite, ako sa zmení čas odozvy TCP a UDP, keď bude klient smerom k serveru posielať 10 paketov (request) a server mu bude na ne odpovedať 100 paketmi (response). Koľkokrát je časová odozva TCP pomalšia než časová odozva UDP?

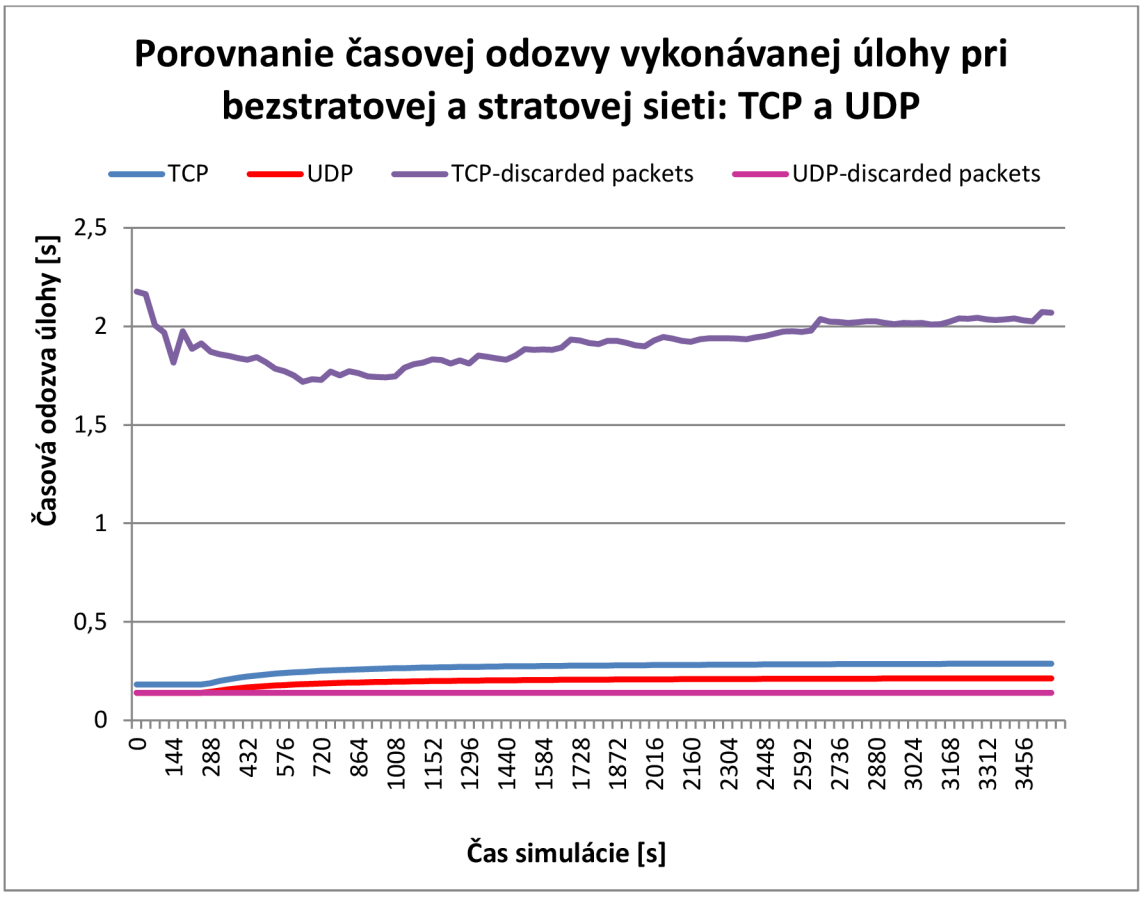
Ako vidíme na Obr. 6.1, pri odpovedi obsahujúcej 10 paketov bola časová odozva TCP takmer raz taká veľká, než pri UDP. Keď zväčšíme odpoveď, časová odozva TCP stúpne takmer paťnásobne.



Obr. 6.1: Graf porovnania časovej odozvy protokolov TCP a UDP pri desiatich paketoch odpovede (TCP a UDP) a pri odpovedi obsahujúcej 100 paketov (TCP_big_response a UDP_big_response) po exporte do MS Excel

3. **Doteraz sme uvažovali, že prenos v sieti bol bezchybný, v skutočnosti je však riziko straty dát vždy prítomné. Ako ovplyvní stratovosť čas odozvy protokolov TCP a UDP?**

Ako vidíme na Obr. 6.2, časy odozvy protokolu UDP v obidvoch prípadoch siete sú takmer identické, pretože protokol UDP posiela datagramy rovnako v sieti bez strát aj so stratami, pretože nemá, na úkor svojej rýchlosti, implementovaný systém regenerácie stratených datagramov. Čas odozvy protokolu TCP v sieti so stratami však narastá znateľne, čo je spôsobené vyjednávaním o preposielanie stratených segmentov, resp. skupín stratených segmentov.



Obr. 6.2: Graf porovnania časovej odozvy TCP pri bezchybnej sieťovej prevádzke (TCP a UDP) a pri sieťovej prevádzke s 5% stratovosťou (TCP – discarded packets a UDP – discarded packets) po exporte do MS Excel (hodnoty boli spriemerované)

7 PROTOKOL ICMP

Protokol ICMP (Internet Control Message Protocol) je súčasťou protokolovej sady TCP/IP. Považuje sa za integrálnu súčasť protokolou IP (Internet Protocol), hoci z architektonického hľadiska leží nad týmto protokolom. Úlohou ICMP je informovať o chybách, ktoré sa vyskytli v sieti, kontrolovať tok dát v sieti a v neposlednom rade aj pomáhať pri diagnostike stavu siete. Bližšie je tento protokol špecifikovaný v dokumente RFC 792. [8]

ICMP správy sú posielané v rôznych situáciách, napr. [9]:

- ak paket nie je schopný prísť do cieľovej stanice,
- ak brána nemá dostatočnú kapacitu vyrovnávacej pamäte, aby daný paket preposlala ďalej,
- ak brána môže usmerniť stanicu, aby presmerovala dátovú komunikáciu cez kratšiu cestu.

Ako uvádza [9], protokol IP nie je navrhnutý tak, aby bol absolútne spoľahlivý. Účelom kontrolných správ je poskytnúť spätnú väzbu o problémoch v komunikačnom prostredí, nie spraviť protokol IP spoľahlivejším. Neexistuje garancia toho, že paket bude doručený alebo že sa vráti kontrolná správa. Niektoré pakety tak môžu byť nedoručené bez akéhokoľvek upozornenia. Vyššie protokoly, ktoré používajú IP teda musia implementovať vlastné mechanizmy na zaručenie spoľahlivosti prenosu, ako napríklad číslovanie správ, potvrdenia doručenia paketov a prípadné preposielanie stratených paketov.

ICMP správy typicky informujú o chybách pri spracovaní paketov. Aby sa predišlo návratu nekonečného množstva správ o správach, bol tento protokol navrhnutý tak, že správy o ICMP správach sa neposielajú. [9]

Uvažujme situáciu, že protokol IP, ktorý beží na určitom smerovači v sieti detekoval, že príchodzí paket musí byť rozdelený (fragmentovaný), aby ho bolo možné ďalej odoslať. Paket však má príznak DF (Don't Fragment), ktorý značí, že takýto paket nemôže byť rozdelený. Protokol IP ale nemôže takýto paket bez predošlého rozdelenia poslať ďalej do siete, preto pošle odosielateľovi diagnostickú správu a tento paket jednoducho zahodí. [2]

Okrem diagnostiky stavu siete sa protokol ICMP veľmi často využíva na monitorovanie sietí. Jeho najpopulárnejšie funkcie sú funkcie PING (Packet Internet Groper) a TRACERT (Trace Route). Pomocou ICMP správ je možné zistiť cestu, ktorou sú dáta preposielané, vyhodnotiť použiteľnosť takejto cesty, vyžiadať hodnotu masky pre špecifické sieťové rozhranie, atď. [2]

Keďže IP paket obsahuje adresu odosielateľa, ale neobsahuje žiadne informácie o tranzitných smerovačoch, sú ICMP správy odosielané len do koncových bodov. Tam môžu byť tieto správy spracované jadrom operačného systému, transportnými alebo aplikačnými protokolmi alebo samotnými aplikáciami. Z toho teda vyplýva, že spracovanie ICMP správ nemá na starosti ani protokol IP, ani protokol ICMP. [2]

7.1 Typy ICMP správ

Všetky ICMP správy spadajú do týchto dvoch kategórií:

- Diagnostické (chybové) správy
- Informačné správy (požiadavky, odpovede)

ICMP správa je zapúzdrená do dátového poľa IP paketu. Samotná hlavička má 8 bajtov a zahŕňa tieto polia (viď Obr. 7.1):

- **Typ** – veľkosť 1 bajt, obsahuje kód, ktorý určuje typ správy
- **Kód** – veľkosť 1 bajt, bližšie špecifikuje typ chyby
- **Kontrolný súčet** – veľkosť 2 bajty, kontrolný súčet pre celú ICMP správu

Hlavička taktiež zahŕňa pole o veľkosti 4 bajty, ktorého obsah závisí od hodnoty polí *Typ* a *Kód*. V informačných správach toto pole obsahuje 2-bajtové podpolia – *Identifikátor* a *Poradové číslo sekvencie*. Čísla obsiahnuté v týchto podpoliach sú okopírované zo správy typu otázka (query/request) do správy typu odpoveď (response). [2]

Hlavička IP paketu (20 bajtov)	Typ (1 bajt)	Kód (1 bajt)	Kontrolný súčet (2 bajty)	Závislé na poli kód/typ (2 bajty)	Závislé na poli kód/typ (2 bajty)

Obr. 7.1: Formát ICMP správy a zapúzdrenie do IP paketu

Podpole *Identifikátor* umožňuje cieľovej stanici určiť, pre ktorú aplikáciu je daná odpoveď zasielaná. Podpole *Poradové číslo sekvencie* používa aplikácia na priradenie odpovede ku konkrétnej otázke (je potrebné vziať do úvahy, že jedna aplikácia môže vyprodukovať viacero otázok rovnakého typu). V chybových správach sa toto pole nepoužíva, preto je vyplnené nulami. [2]

V tabuľke Tab. 7.1, sú uvedené príklady hodnôt, ake môže obsahovať pole *Typ* ICMP správy.

Tab. 7.1: Hodnoty pre pole Typ ICMP správy [8]

Hodnota	Typ správy
0	**Echo reply** (Echo odpoveď)
3	**Destination unreachable** (Cieľová stanica nedostupná)
4	**Source quench** (Stlmenie zdroja)
5	**Redirect** (Presmerovanie)
8	**Echo request** (Echo požiadavka)
11	**Time exceeded for a datagram** (Čas pre datagram vypršal)
12	**Parameter problem on datagram** (Problém s parametrom datagramu)
13	**Timestamp request** (Požiadavka na časovú pečiatku)
14	**Timestamp response** (Odpoveď na časovú pečiatku)
17	**Address mask request** (Požiadavka na masku adresy)
18	**Address mask response** (Odpoveď na masku adresy)

Chyby každého typu môžu byť charakterizované ešte podrobnejšie pomocou kódov chýb. Tabuľka Tab. 7.2 uvádza 16 príčin, prečo môže byť cieľová stanica nedostupná (destination unreachable – type 3). To znamená, že nedostupnosť cieľovej stanice môže byť napríklad spôsobená dočasnou nefunkčnosťou hardvéru, nesprávnou cieľovou adresou, chýbajúcim protokolom aplikačnej vrstvy, atď.

Tab. 7.2: Typy kódov pre ICMP správu typu 3 – Destination Unreachable

Kód	Príčina	Popis
0	**Network unreachable** (Sieť nedostupná)	Generovaný smerovačom v prípade, že cesta do cieľovej siete je nedostupná
1	**Host unreachable** (Stanica nedostupná)	Generovaný smerovačom v prípade, že cesta k cieľovej stanici alebo do cieľovej siete je nedostupná (neodpovedá na ARP)
2	**Protocol unreachable** (Protokol nedostupný)	Generovaný v prípade, že transportný protokol určený v datagrame nie je podporovaný transportnou vrstvou cieľovej destinácie

3	**Port unreachable** (Port nedostupný)	Generovaný v prípade, že určený transportný protokol nie je schopný demultiplexovať datagram na transportnej vrstve cieľovej destinácie, ale nemá žiadne vlastné mechanizmy na informovanie odosielateľa
4	**Fragmentation needed, but DF set** (Fragmentácia potrebná, ale paket má DF príznak)	Generovaný ak smerovač potrebuje fragmentovať datagram, ale nemôže, pretože tento má nastavený DF príznak
5	**Source route failed** (Cesta k zdroju zlyhala)	Generovaný ak smerovač nemôže preposlať paket na ďalší „hop“ na ceste k zdroju
6	**Destination network unknown** (Cieľová sieť neznáma)	Tento kód by nemal byť vygenerovaný, pretože by to znamenalo, že cieľová sieť vôbec neexistuje (namiesto neho by mal byť použitý kód 0 – Network unreachable)
7	**Destination host unknown** (Cieľová stanica neznáma)	Generovaný len vtedy, keď smerovač dokáže z informácii od linkovej vrstvy určiť, že cieľová stanica skutočne neexistuje
8	**Source host isolated** (Zdrojová stanica izolovaná)	
9	**Destination network administratively prohibited** (Cieľová sieť administratívne zakázaná)	
10	**Destination host administratively prohibited** (Cieľová stanica administratívne zakázaná)	
11	**Network unreachable for type of service** (Sieť nedostupná kvôli typu služby)	Generovaný smerovačom ak nie je dostupná cesta k cieľovej stanici, ktorá spĺňa požadovaný alebo štandardný typ služby
12	**Host unreachable fo type of service** (Stanica nedostupná kvôli typu služby)	Generovaný ak smerovač nemôže preposlať paket, pretože cesta do cieľovej stanice nezodpovedá ani požadovanému, ani štandardnému typu služby

13	**Communication administratively prohibited by filtering** (Komunikácia administratívne zakázaná filtrovaním)	Generovaný ak smerovač nemôže preposlať paket kvôli administratívnemu filtrovaniu
14	**Host precedence violation** (Priorita stanice porušená)	Posielané prvým smerovačom na ceste k stanici na indikáciu, že požadovaná priorita nie je povolená pre konkrétnu kombináciu zdrojovej/cieľovej stanice alebo siete, protokolu vyššej vrstvy alebo zdrojového/cieľového portu
15	**Precedence cut off in effect** (Priorita nižšia než priorita stanovená sieťou)	Sieti bola nastavená určitá úroveň priority vyžadovaná pre sieťové operácie a datagram bol poslaný s nižšou prioritou než je priorita siete

Ako sa ďalej uvádza v [10], kód číslo 8 (Zdrojová stanica izolovaná) by nemal byť generovaný, ale ak je to vhodnejšie, mali by byť namiesto neho použité kódy 0 (Sieť nedostupná) alebo 1 (Stanica nedostupná). Taktiež existuje výnimka pre používanie kódov 9 (Cieľová sieť administratívne zakázaná) a 10 (Cieľová stanica administratívne zakázaná). Tieto boli určené na použitie v koncových šifrovacích zariadeniach používaných vojenskými agentúrami v Spojených štátoch amerických. Smerovače by teda mali používať novšie definovaný kód 13 (Komunikácia administratívne zakázaná filtrovaním), pokiaľ chcú administratívne filtrovať pakety. Smerovače musia používať kódy 1 (Stanica nedostupná) a 7 (Cieľová stanica neznáma) vždy keď iná cieľová stanica na rovnakej sieti môže byť dostupná, inak môže zdroj chybne predpokladať, že všetky stanice v danej sieti sú nedostupné. [10]

Formát dátového poľa ICMP taktiež závisí na hodnotách polí *Typ* a *Kód*. Na demonštráciu rozdielov medzi formátmi jednotlivých správ použijeme nasledujúce dva príklady:

- Echo Request/Reply
- Destination Unreachable

7.2 Formát správy Echo Request/Reply: Funkcia PING

Obrázok Obr. 7.2 ilustruje formát správy *Echo Request* a *Echo Response*. Líšia sa od seba len hodnotou v poli *Typ*. V dátovom poli požiadavky odosielateľ umiestni informáciu, ktorú následne obdrží v odpovedi od cieľovej stanice.

Typ (1 bajt)	Kód (1 bajt)	Kontrolný súčet (2 bajty)
Identifikátor požiadavky (2 bajty)		Poradové číslo sekvencie (2 bajty)
Dáta		

Obr. 7.2: Formát ICMP správy typu Echo Request/Reply

Echo Request a *Echo Reply*, ktoré sú súhrnne nazývané aj *echo protokol*, predstavujú najjednoduchší nástroj na monitorovanie siete. Počítač alebo smerovač pošle *Echo Request* cez Internet až do cieľovej stanice. Predtým je však potrebné skontrolovať, či je táto stanica dostupná. Stanica potom obdrží *Echo Request*, na ktorý odpovie pomocou *Echo Reply*, ktorú obdrží zdrojová stanica. Keďže echo správy sú v sieti posielané vo forme IP paketu, predpokladá sa, že pre ich úspešné doručenie je potrebné, aby bol funkčný transportný systém internetu. [2]

Väčšina operačných systémov používa vstavanú funkciu PING určenú na testovanie dostupnosti stanice. Táto funkcia vyšle sériu echo requestov smerom k testovanej cieľovej stanici a zároveň užívateľa informuje o štatistike prenosu, množstve stratených odpovedí a čase odozvy na požiadavky. Funkcia PING informuje užívateľa o všetkých prijatých odpovediach. Výstup vyzerá ako na Obr. 7.3.

```
C:\Users\User>ping www.vutbr.cz

Pinging piranha.ro.vutbr.cz [147.229.2.90] with 32 bytes of data:
Reply from 147.229.2.90: bytes=32 time=22ms TTL=57
Reply from 147.229.2.90: bytes=32 time=22ms TTL=57
Reply from 147.229.2.90: bytes=32 time=22ms TTL=57
Reply from 147.229.2.90: bytes=32 time=22ms TTL=57

Ping statistics for 147.229.2.90:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 22ms, Maximum = 22ms, Average = 22ms
```

Obr. 7.3: Výstup funkcie PING

Dĺžka každej odoslanej správy je 64 bajtov. Ďalšou hodnotou je hodnota RTT, ktorá vyjadruje čas za ktorý bola poslaná žiadosť a následne na to obdržaná odpoveď. V tomto prípade je to 22 ms. Zvyčajne je možné príkaz PING ešte viac špecifikovať,

napríklad určiť veľkosť dátového poľa správy, počiatočnú hodnotu TTL (Time To Live), počet pokusov o prenos paketu alebo nastavenie príznaku DF. Ak v stanovenom čase neprídu žiadne odpovede alebo ak ICMP odpovie chybovou správou, PING zobrazí zodpovedajúcu chybovú hlášku.

7.3 Formát chybovej správy: Funkcia TRACEROUTE

Obrázok Obr. 7.4 ilustruje formát ICMP chybovej správy, v tomto prípade ide o *Destination Unreachable* správu. Ostatné ICMP chybové správy majú rovnaký formát a líšia sa od seba len v hodnotách polí *Typ* a *Kód*.

Typ = 3 (1 bajt)	Kód = 0-15 (1 bajt)	Kontrolný súčet (2 bajty)
Nepoužité		
IP hlavička + prvých 8 bajtov IP datagramu, ktorý zlyhal		

Obr. 7.4: Formát ICMP chybovej správy typu Destination Unreachable

Ak smerovač nemôže vysielať alebo doručiť IP paket, vyšle o tom odosielateľovi správu *Destination Unreachable*. V tejto správe je v poli *Typ* uvedená hodnota 3 a v poli *Kód* je bližšie špecifikovaný dôvod, prečo paket nemohol byť doručený. Po tomto poli nasleduje pole kontrolný súčet a pole o veľkosti 4 bajty, ktoré je v tomto prípade nevyužité a je vyplnené nulami. [2]

Okrem príčiny chyby špecifikovanej v ICMP hlavičke, ICMP vždy vyplní svoje dátové pole IP hlavičkou a prvými 8 bajtami dátového poľa IP paketu, ktorý spôsobil chybu. Táto informácia umožní odosielateľovi bližšie určiť príčinu chyby, keďže všetky protokoly sady TCP/IP používajú na prenos svojich správ IP paket, ktorý obsahuje najdôležitejšie informácie práve v prvých 8 bajtoch TCP alebo UDP hlavičky. Hlavičky TCP a UDP obsahujú informáciu o tom, ktorá aplikácia poslala stratený paket. [2]

Ako už bolo demonštrované pri funkcii PING, ICMP správy sú veľmi účinne využívané na monitorovanie sietí. Obzvlášť chybová správa o vypršanom čase (Time Exceeded For a Datagram) je základom pre ďalšiu známu funkciu TRACEROUTE resp. TRACERT. Táto umožňuje „vystopovať“ cestu k vzdialenej stanici definovanú časom obehu, IP adresou a doménovým menom každého tranzitného smerovača (ak má tento svoje meno registrované v tzv. DNS lookup zone). Táto informácia pomáha lokalizovať, kde zlyhalo spojenie. [2]

TRACEROUTE trasuje cestu poslaním normálneho IP paketu s cieľovou adresou konečného bodu. Myšlienka stopovania spočíva v tom, že TTL prvého vysielaného

paketu je nastavené na hodnotu 1. Keď IP protokol prvého smerovača obdrží tento paket, zníži jeho TTL o hodnotu 1 podľa svojho algoritmu. TTL teda nadobudne nulovú hodnotu a zdrojovej stanici sa vracia chybová správa *Time Exceeded For a Datagram* s IP hlavičkou a prvými ôsmymi bajtmi strateného paketu. Keď ICMP doručí ICMP správu o nedoručení paketu, funkcia TRACEROUTE si zapamätá adresu prvého smerovača (získa ju z IP hlavičky v ICMP správe). Potom vypočíta RTT hodnotu pre prvý smerovač a následne TRACEROUTE pošle ďalší paket s TTL nastaveným na hodnotu 2. Tento paket už úspešne prejde prvým smerovačom, ale je zas zahodený na druhom, ktorý opäť vyšle chybovú správu *Time Exceeded For a Datagram*. TRACEROUTE si zapamätá IP adresu a TTL druhého smerovača, atď. Proces pokračuje až kým sa nedosiahne cieľová stanica. Výstup funkcie TRACEROUTE vyzerá ako na Obr. 7.5.

```
C:\Users\User>tracert www.vutbr.cz

Tracing route to piranha.ro.vutbr.cz [147.229.2.90]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  192.168.15.1
  2      1 ms     16 ms      6 ms   mpop-stone-r2-s1983-pri.mng.netbox.cz
[95.82.173.66]
  3     1 ms    <1 ms    <1 ms  brn-pop-r2-vl140.netbox.cz [83.240.3.41]
  4    23 ms    23 ms    28 ms  r98-cbix.cesnet.cz [195.113.179.201]
  5    23 ms    23 ms    23 ms  gw-ant.net.vutbr.cz [147.229.252.19]
  6    23 ms    23 ms    33 ms  hp-ant.net.vutbr.cz [147.229.253.235]
  7    23 ms    24 ms    23 ms  irf-ant.net.vutbr.cz [147.229.254.126]
  8    22 ms    22 ms    22 ms  piranha.ro.vutbr.cz [147.229.2.90]

Trace complete.
```

Obr. 7.5: Výstup funkcie TRACEROUTE

Postupnosť riadkov zodpovedá postupnosti smerovačov na ceste k cieľovej stanici. Prvé číslo každého riadku predstavuje počet skokov (hops) k vhodnému smerovaču. Funkcia TRACEROUTE testuje každý smerovač trikrát, preto sú v každom riadku tri hodnoty RTT. Tieto sú vypočítané poslaním troch paketov, ktorých TTL vypršalo na danom smerovači. Ak špecifický smerovač neodošle odpoveď do určitého času, je namiesto hodnoty RTT zobrazená hviezdička (*). Ďalej sa špecifikuje IP adresa a doménové meno smerovača (ak je dostupné). [2]

Treba podotknúť, že čas zobrazený v každom riadku nie je čas potrebný na to, aby paket prešiel medzi dvomi smerovačmi, ale je to čas, počas ktorého prejde paket od zdroja k smerovaču a zas naspäť ku zdroju.

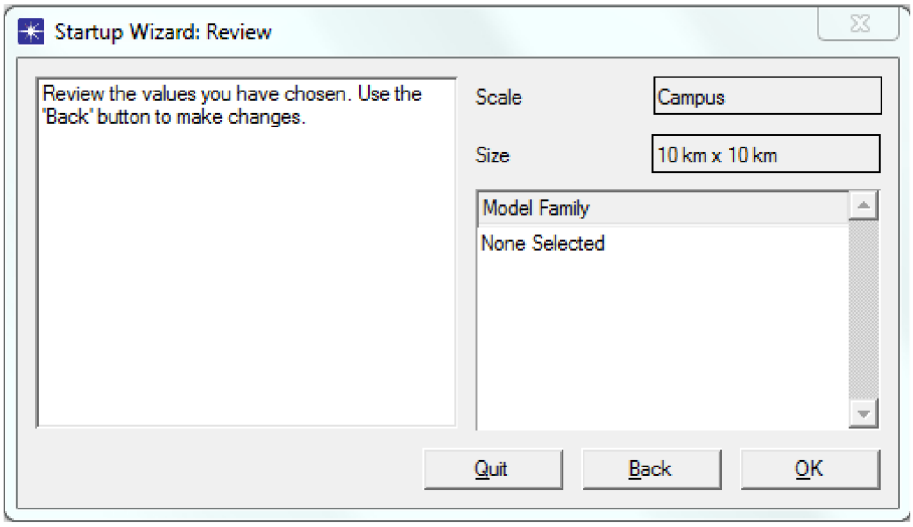
Protokol ICMP síce hrá v sieti viacmenej druhotnú úlohu, ale je veľmi používaný na kontrolu toku dát v sieti, na reportovanie chýb a na diagnostiku výkonnosti siete.

8 LABORATÓRNA ÚLOHA: ICMP – PING

V tejto laboratórnej úlohe budeme využívať funkciu PING, ktorá bude posielať ICMP správy s požiadavkou (ICMP Echo Request) smerom k cieľovej stanici. Táto bude následne posielať ICMP správy s odpoveďou (ICMP Echo reply). Trasy, ktoré prejdú ICMP správy budú zaznamenané v simulačnom logu. [11]

8.1 Vytvorenie projektu

1. Spustíme OPNET IT Guru Academic Edition.
2. V menu zvolíme položku **File → New...**.
3. Z ponuky vyberieme **Project** a potvrdíme tlačidlom **OK**. Položku **Project Name** zmeníme na **vaseVUTID_PING** a položku **Scenario Name** zmeníme na **WithoutFailure** a potvrdíme tlačidlom **OK**.
4. V okne **Startup Wizard: Initial Topology** zvolíme **Create Empty Scenario** a stlačíme **Next**.
5. V okne **Startup Wizard: Choose Network Scale** zvolíme **Campus** a stlačíme **Next**.
6. V okne **Startup Wizard: Specify Size** zvolíme veľkosť v kilometroch a rozmery **X = 10** a **Y = 10** a stlačíme **Next**.
7. V okne **Startup Wizard: Select Technologies** rovno stlačíme **Next**.
8. V okne **Startup Wizard: Review** potvrdíme nastavenia stlačením **OK**.



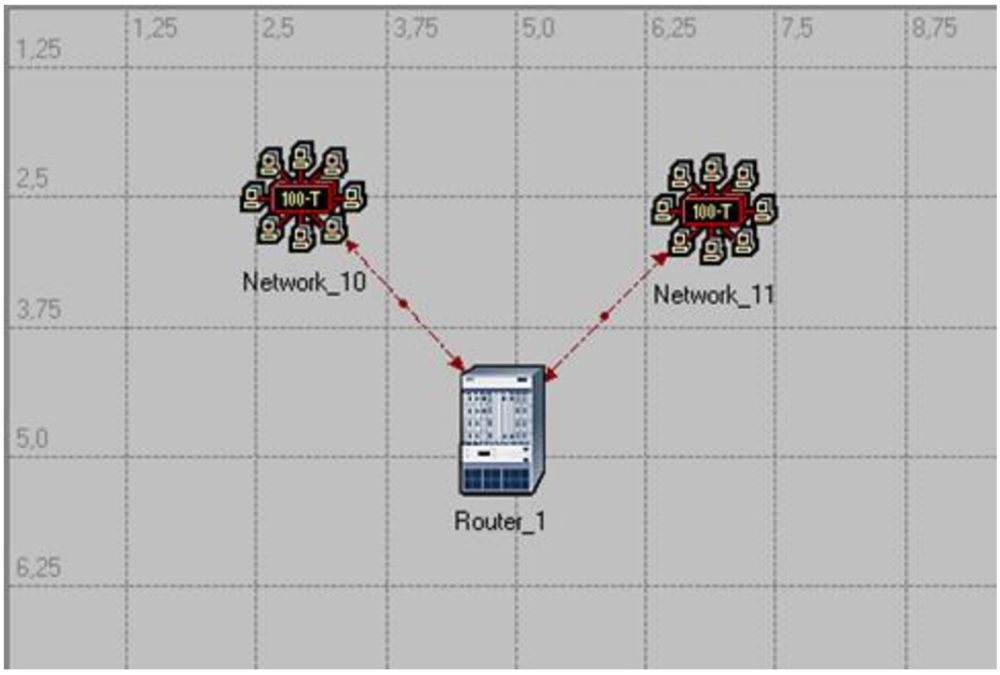
Obr. 8.1: Rekapitulácia zvolených parametrov

8.2 Vytvorenie a konfigurácia siete

9. Po potvrdení okna **Startup Wizard: Review** sa nám otvorí mapa projektu a paleta objektov **Object Palette**. V prípade, že by sa nám paleta neotvorila alebo by sme si ju omylom zatvorili, opätovne sa k nej dostaneme kliknutím na tlačidlo  . Uistíme sa, že v palete máme vybraté **internet_toolbox**.

10. Do mapy projektu si pridáme z palety nasledujúce objekty: **ethernet4_slip8_gtwy** smerovač a dva objekty **100BaseT_LAN**. Ethernet4_slip8_gtwy predstavuje IP bránu, ktorá podporuje 4 Ethernet rozhrania a 8 sériových rozhraní.

11. Pravým tlačidlom klikneme na objekt smerovača a zvolíme **Set name**. Premenujeme ho na **Router_1** a potvrdíme stlačením **OK**. K nemu príslušiace siete rovnakým spôsobom pomenujeme **Network_10** a **Network_11**. Smerovač potom prepojíme s obidvomi sieťami prenosovým médiom **100BaseT**.



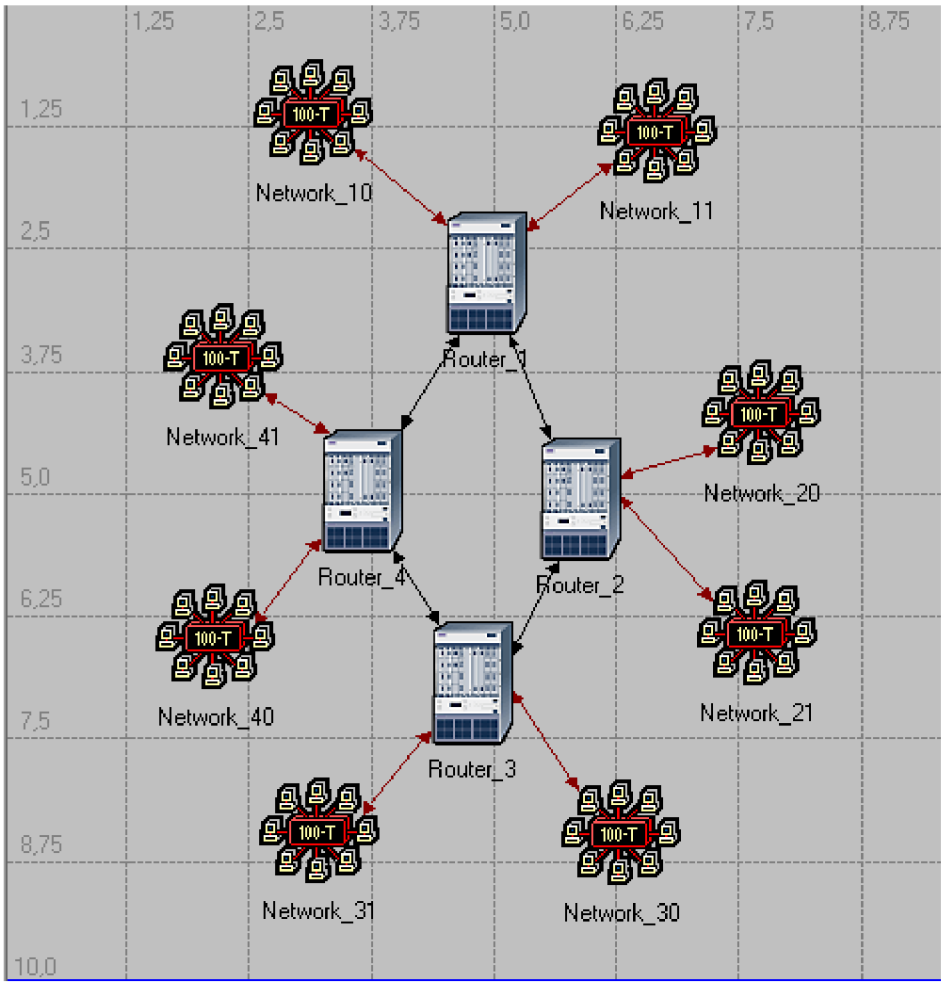
Obr. 8.2: Čiastočná topológia simulovanej siete

12. Pravým tlačidlom klikneme na objekt **Router_1** a zvolíme **Edit attributes**. Rozklikneme riadok **IP Routing Parameters** a položku **Routing Tabel Export** nastavíme na **Once at End of Simulation**. Smerovač bude týmto exportovať

smerovacie tabuľky do simulačného logu na konci simulácie. Nastavenie potvrdíme kliknutím na **OK**. Projekt si priebežne ukladáme.

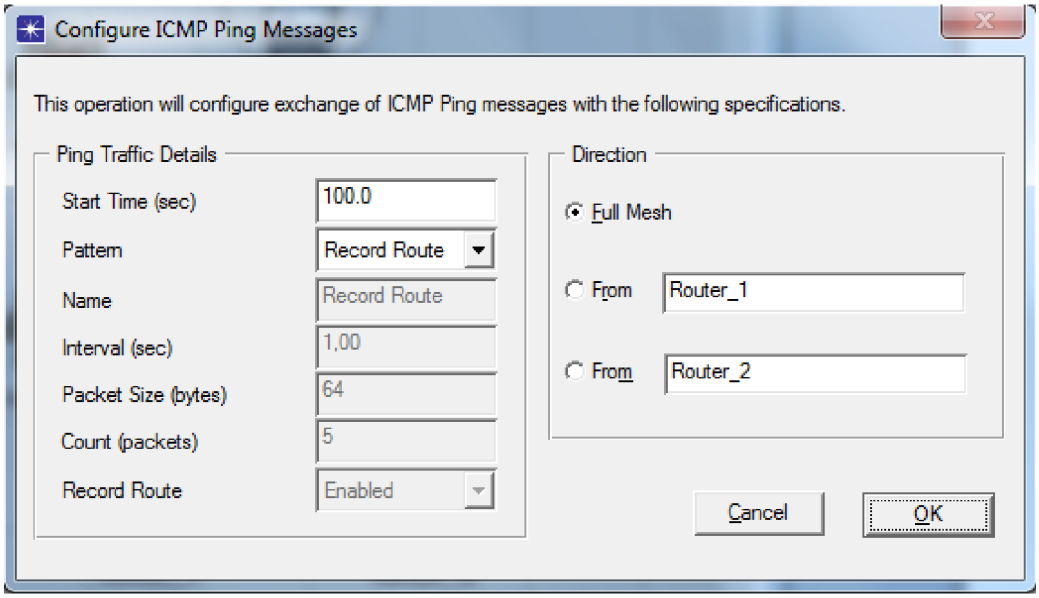
13. Podržíme **SHIFT** a ľavým tlačidlom označíme smerovač, obidve siete a aj obidve linky. Skopírujeme objekty pomocou **CTRL + C** a vložíme ich do mapy projektu pomocou **CTRL + V**. Tento proces zopakujeme ešte dvakrát, aby sme na konci mali v mape projektu 4 smerovače a 8 sietí. Všetky zariadenia premenujeme.

14. Smerovače prepojíme pomocou prenosového média **PPP_DS3**, ktoré predstavuje linku s rýchlosťou 44,736 Mbps.



Obr. 8.3: Výsledná topológia

15. Pomocou tlačidla **SHIFT** označíme **Router_1** a **Router_2.** V menu zvolíme **Protocol → IP → Demands → Configure Ping Traffic on Selected Nodes**. V okne **Configure ICMP Ping Messages** zmeníme položku **Pattern** na **Record router**. Nastavenie spôsobí, že PING bude obojstranný a Router_1 bude pingovať Router_2 a naopak.

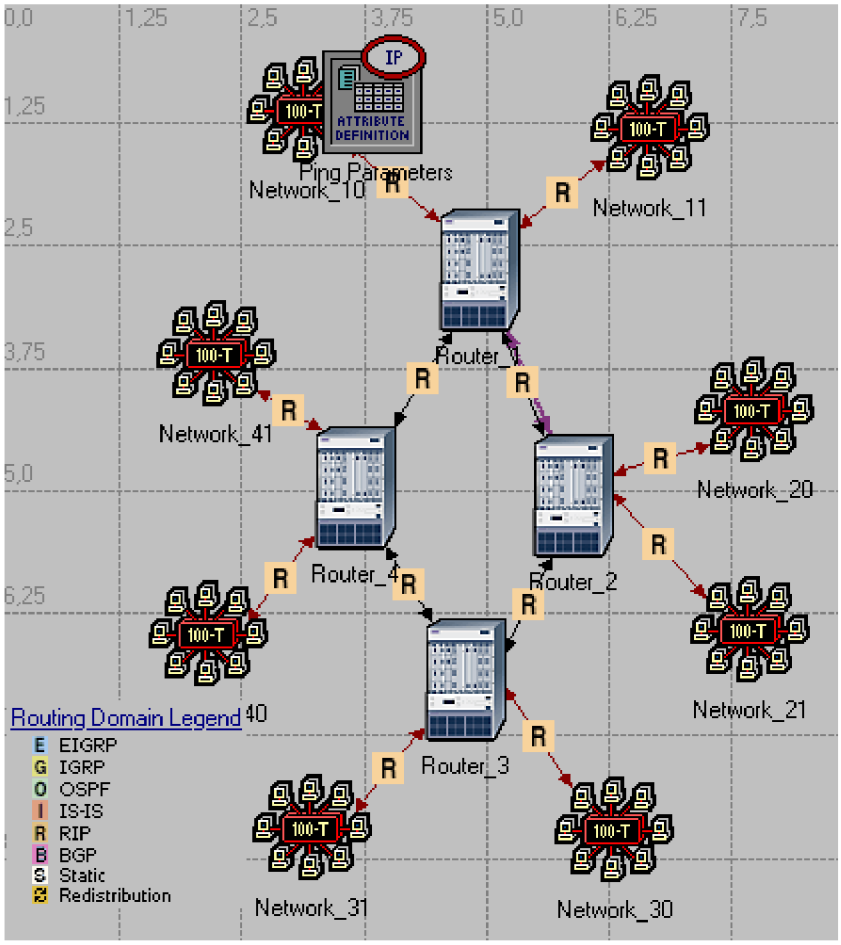


Obr. 8.4: Nastavenie ICMP PING správ

16. V menu klikneme na **Protocols → IP → Routing → Configure Routing Protocols...**. Skontrolujeme, že je označený len **RIP** a stlačíme **OK**. V mape projektu sa nám objaví nový objekt **Ping Parameters**.

17. Klikneme pravým tlačidlom na objekt **Ping Parameters** a zvolíme **Advanced Edit Attributes**. Rozklikneme riadok **IP Ping Parameters** a ďalej rozklikneme **row 1** a jeho detaily. Položku **Count** nastavíme na **10** (štandardné nastavenie je 5). Týmto budeme odosielať 10 Echo request správ. **Interval (sec)** nastavíme na **20 sekúnd** a **Timeout (Sec)** na **1 sekundu**.

18. V menu klikneme na **Protocols → RIP → Configure Start Time**. Položku **Mean Outcome** zmeníme na hodnotu **20** a stlačíme **OK**. Protokol RIP začne od tohto momentu vytvárať smerovacie tabuľky.



Obr. 8.5: Topológia s nastaveným smerovacím protokolom RIP

8.3 Nastavenie simulácie

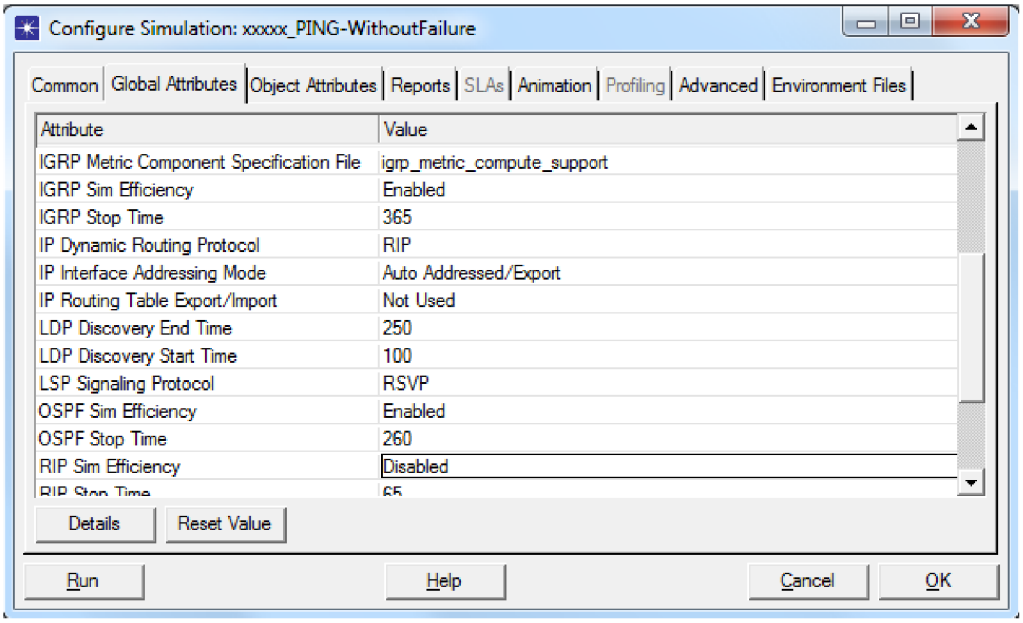
19. V menu zvolíme **Simulation → Configure Discrete Event Simulation**, resp. použijeme klávesovú skratku **CTRL + R** alebo stlačíme tlačidlo  . Nastavíme trvanie simulácie na **10 minút** a prepneme sa do záložky **Global Attributes**, kde nastavíme nasledovné:

    a) **IP Dynamic Routing Protocol** na hodnotu **RIP**

    b) **IP Interface Addressing Mode** na hodnotu **Auto Addressed/Export**, IP adresy sú počas simulácie nastavené automaticky

    c) **RIP Sim Efficiency** na hodnotu **Disabled**, toto nastavenie umožní obnovovanie smerovacej tabuľky ak v sieti nastanú nejaké zmeny

    Nastavenia potvrdíme kliknutím na **OK**. Projekt priebežne ukladáme.



Obr. 8.6: Nastavenie simulačných parametrov

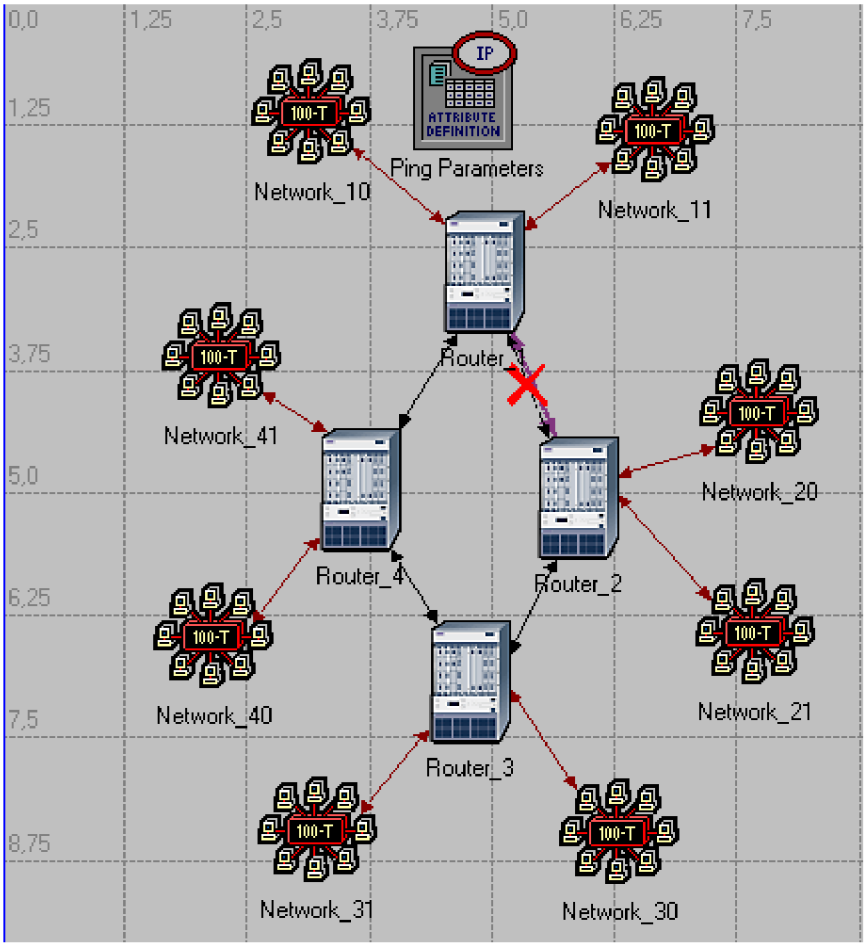
20. V menu klikneme na **Simulation → Choose Individual Statistics**. Rozklikneme **Node Statistics**. Rozklikneme **IP** a označíme **Ping Replies Received (packets)**, **Ping Requests Sent (packets)** a **Ping Response Time (sec)**. Potvrdíme stlačením **OK**.

8.4 Duplikácia scenára

Vyrobíme si scenár, kde zlyhá linka medzi smerovačmi. Následne porovnáme výsledky obidvoch scenárov.

21. V menu klikneme na **Scenarios → Duplicate Scenario...**. Scenár pomenujeme **WithFailure** a potvrdíme kliknutím na **OK**.

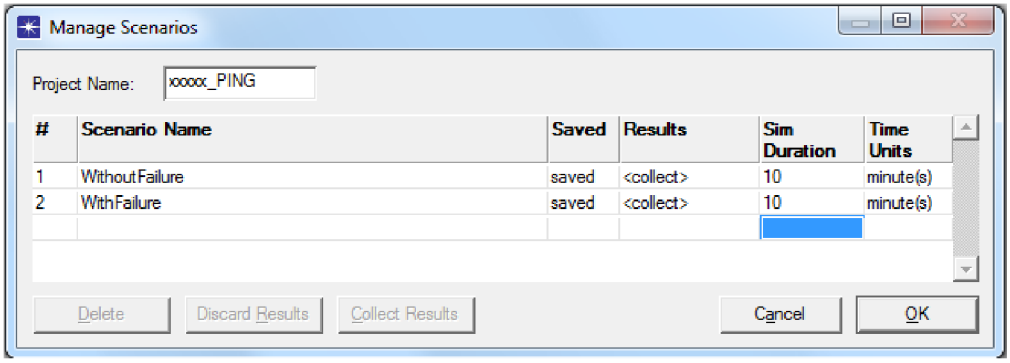
22. Označíme linku medzi smerovačmi **Router_1** a **Router_2** a klikneme na tlačidlo  . Linka bude teraz vystupovať ako prerušená. Projekt si uložíme.



Obr. 8.7: Topológia scenára so zlyhaním

8.5 Spustenie simulácie

23. V menu klikneme na **Scenarios** → **Manage Scenarios...**. V poli **Results** zmeníme hodnotu na **<collect>**. Po potvrdení tlačidlom **OK** prebehne simulácia obidvoch scenárov.



Obr. 8.8: Prehľad scenárov

24. Keď prebehne simulácia, môžeme zavrieť okno **Simulation Sequence**.

8.6 Zobrazenie výsledkov

25. V scenári **WithoutFailure** klikneme v menu na **Results → Open Simulation Log** a klikneme na pole **PING REPORT for Campus Network.Router_1**.

```
PING REPORT for "Campus Network.Router_1" (192.0.4.1)

DETAILS:
  Received  ICMP echo reply packet for a
  request packet sent to the following node:

    IP Address: 192.0.4.1
    Node Name : Campus Network.Router_1

PERFORMANCE:
  Based on the first ICMP echo request packet
  (i.e., a "ping" packet) sent to the above
  node, the following metrics were computed:

    1. Response Time: 0,00014 seconds

    2. List of traversed IP interfaces:

       IP Address       Hop Delay     Node Name
       ----------       ---------     ---------
        192.0.3.2       0,00000      Campus Network.Router_2
        192.0.4.1       0,00005      Campus Network.Router_1
        192.0.3.1       0,00002      Campus Network.Router_1
        192.0.3.2       0,00005      Campus Network.Router_2

  Note that the IP addresses shown above represent
  the address of the output interface on which the
  IP datagram was routed from the corresponding
  nodes to the next node enroute to its destination
  and back.
```

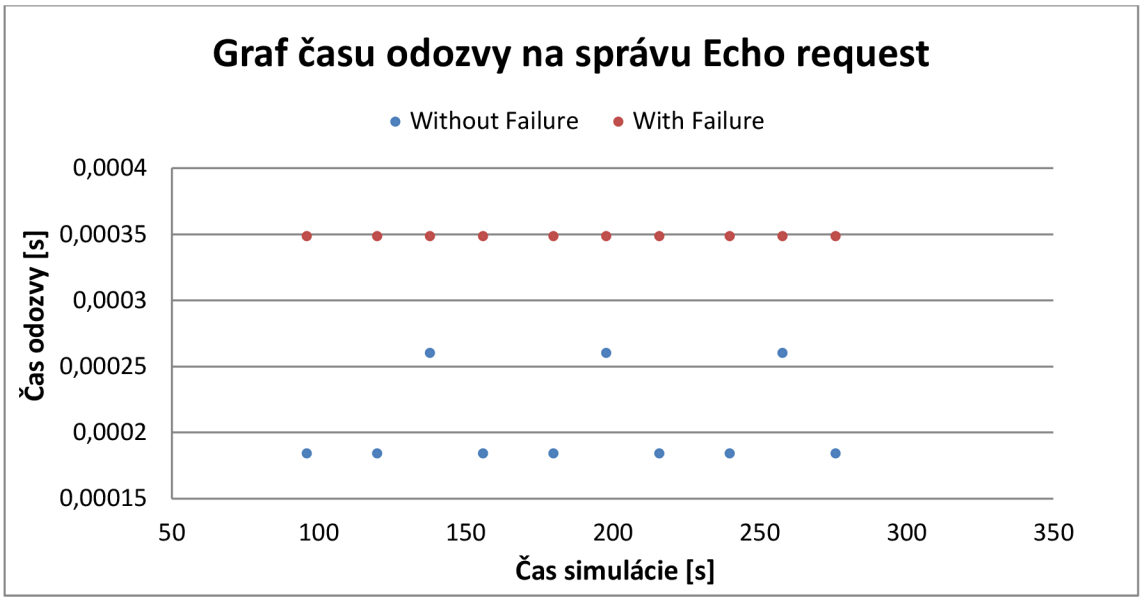
Obr. 8.9: Výstupný report pre scenár bez zlyhania

26. V scenári **WithFailure** si tiež otvoríme **PING REPORT for Campus Network.Router_1**.

```
PING REPORT for "Campus Network.Router_1" (192.0.4.1)

DETAILS:
  Received  ICMP echo reply packet for a
  request packet sent to the following node:

    IP Address: 192.0.4.1
    Node Name : Campus Network.Router_1

PERFORMANCE:
  Based on the first ICMP echo request packet
  (i.e., a "ping" packet) sent to the above
  node, the following metrics were computed:

    1. Response Time: 0,00035 seconds

    2. List of traversed IP interfaces:

       IP Address       Hop Delay     Node Name
       ----------       ---------     ---------
        192.0.7.1       0,00000      Campus Network.Router_2
        192.0.11.2      0,00005      Campus Network.Router_3
        192.0.2.2       0,00005      Campus Network.Router_4
        192.0.4.1       0,00005      Campus Network.Router_1
        192.0.2.1       0,00002      Campus Network.Router_1
        192.0.11.1      0,00005      Campus Network.Router_4
        192.0.7.2       0,00005      Campus Network.Router_3
        192.0.7.1       0,00005      Campus Network.Router_2

  Note that the IP addresses shown above represent
  the address of the output interface on which the
  IP datagram was routed from the corresponding
  nodes to the next node enroute to its destination
  and back.
```

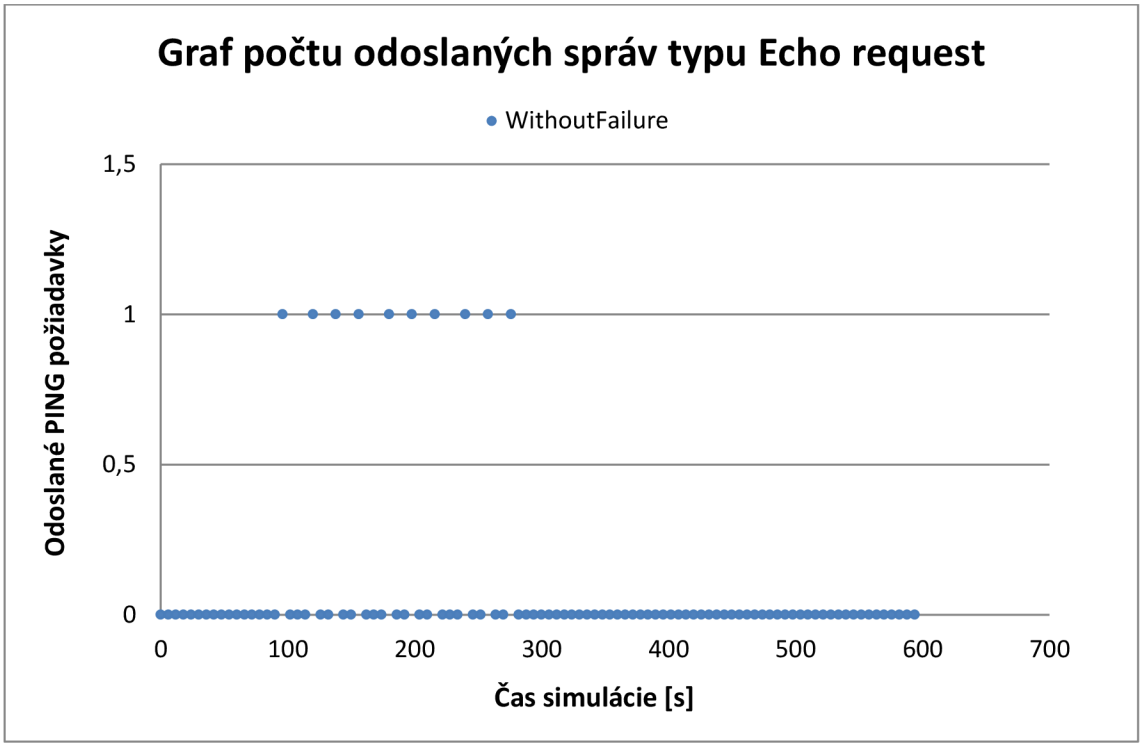
Obr. 8.10: Výstupný report pre scenár so zlyhaním

27. V menu klikneme na **Results → Compare results**. Rozklikneme **Campus Network** a **IP** štatistiky pre **Router_1**. Vyberieme **Ping Response Time (sec)**. Graf zobrazí, ako rýchlo dostane Router_1 odpoveď na ním odoslanú správu Echo Request.



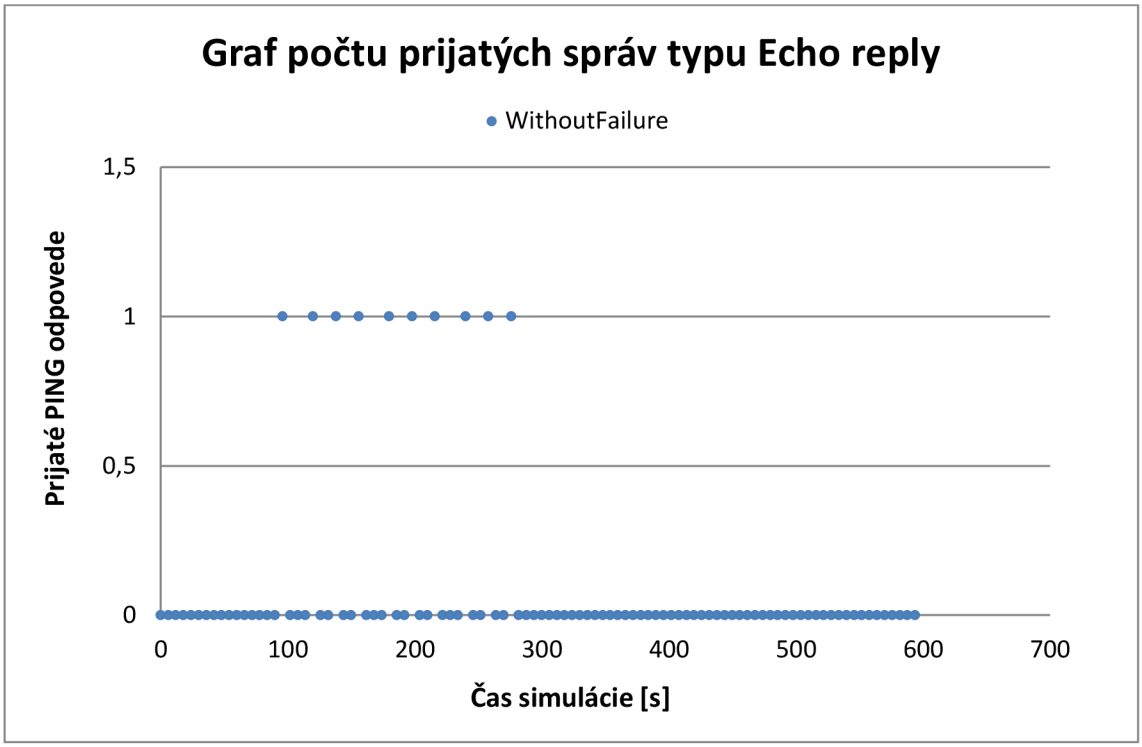
Obr. 8.11: Graf času odozvy na správu Echo Request po exporte do MS Excel

28. Ďalej vyberieme **Ping Requests Sent (packets)**, kde je zobrazené množstvo odoslaných PING požiadaviek smerom k druhému smerovaču. V našom prípade 10 správ.



Obr. 8.12: Graf počtu odoslaných správ typu Echo Request po exporte do MS Excel

29. Nakoniec vyberieme **Ping Replies Received (packets)**, kde je zobrazené množstvo odpovedí typu Echo Reply. V našom prípade 10 správ.



Obr. 8.13: Graf počtu prijatých správ typu Echo Reply po exporte do MS Excel

8.7 Otázky

1. Aký protokol zapúzdruje protokol ICMP? Na čo sa používa protokol ICMP a aké poznáte ICMP správy?

2. Ako sa zmení čas odpovede keď zmeníme veľkosť ICMP Echo Request/Reply správy na 10 000 bytov?

   Zduplikujeme scenár bez zlyhania a pomenujeme ho napr. **Big_PING**. Klikneme pravým tlačidlom na objekt **Ping Parameters** a vyberieme **Advanced Edit Attributes**. V riadku **IP Ping Parameters** zmeníme v riadku 0 aj v riadku 1 **Packet size (bytes)** na hodnotu **10 000** a potvrdíme kliknutím na **OK**. Necháme prebehnúť simuláciu a obdobne ako v predošlej úlohe si zobrazíme výsledky simulačného logu.

9 HODNOTENIE DOSIAHNUTÝCH VÝSLEDKOV – ICMP

V simulovanej laboratórnej úlohe sme porovnávali trasu ICMP Echo Request/Reply správy v korektne fungujúcej sieti s trasou ICMP Echo Request/Reply správy v sieti, kde je jedna časť siete nefunkčná. Neskôr sme porovnávali aj ako zmena veľkosti ICMP správy ovplyvní čas odozvy cieľovej stanice. Čas odozvy bol meraný v sieti s nulovou prevádzkou na pozadí, teda ide o ideálny prípad. PING medzi smerovačmi Router_1 a Router_2 sme zvolili z toho dôvodu, že pri znefunčnení linky medzi nimi, je zreteľne vidno, ako sa sieť vysporiada s prípadným presmerovaním paketov inou cestou.

9.1 Odpovede na otázky z laboratórnej úlohy

1. **Aký protokol zapúzdruje protokol ICMP? Na čo sa používa protokol ICMP a aké poznáte ICMP správy?**

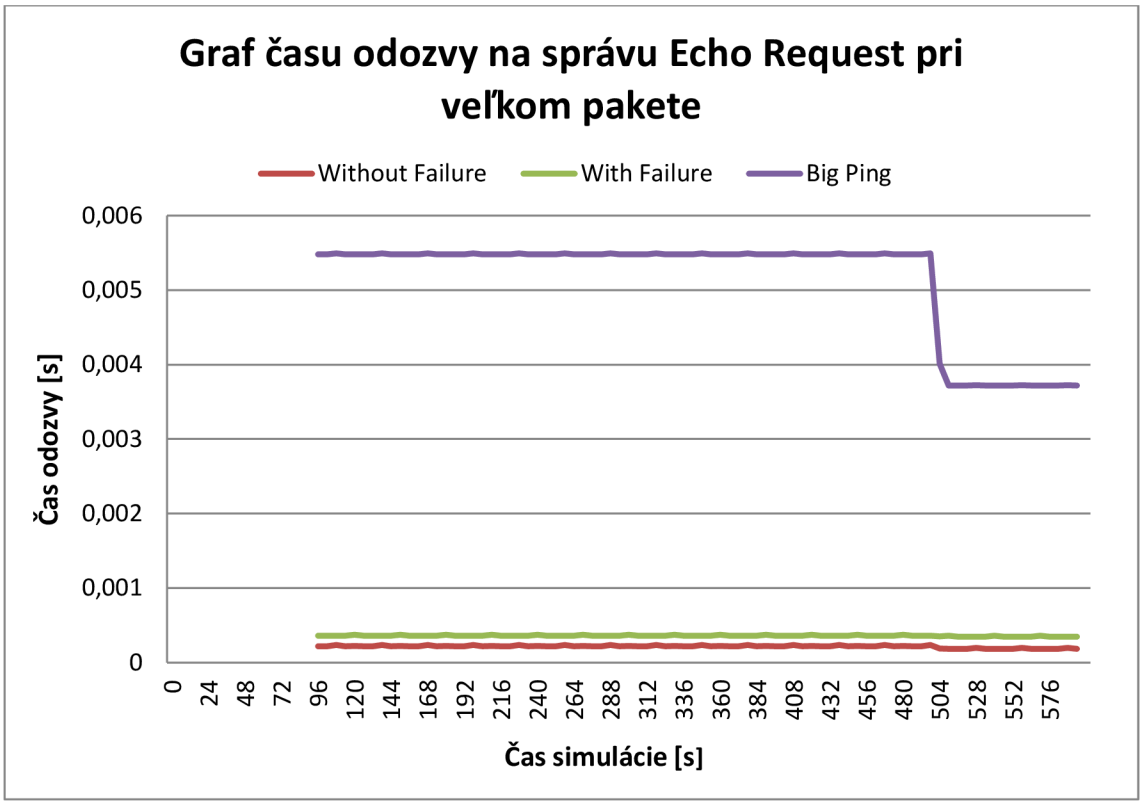
Protokol IP zapúzdruje protokol ICMP. Protokol ICMP sa používa ako nástroj pri správe sietí a hľadaní chýb pri nekorektnej funkčnosti siete. Využíva sa napríklad keď datagram nedorazí do cieľovej stanice, pretože táto nebola nájdená, správa Host Unreachable alebo správy Echo Request/Reply, ktoré slúžia na zistenie, či spojenie medzi určitými uzlami siete funguje tak, ako má.

2. **Ako sa zmení čas odpovede keď zmeníme veľkosť ICMP Echo Request/Reply správy na 10 000 bytov?**

Ako sme predpokladali, keď zmeníme veľkosť paketu (ICMP Echo Request/Reply správy) z pôvodných 64 bytov na 10 000 bytov, čas odozvy stúpne desaťnásobne (viď Obr. 9.1, Obr. 9.2).

```
PING REPORT for "Campus Network.Router_1" (192.0.4.1)

DETAILS:
  Received  ICMP echo reply packet for a
  request packet sent to the following node:

    IP Address: 192.0.4.1
    Node Name : Campus Network.Router_1

PERFORMANCE:
  Based on the first ICMP echo request packet
  (i.e., a "ping" packet) sent to the above
  node, the following metrics were computed:

    1. Response Time: 0,00372 seconds

    2. List of traversed IP interfaces:

       IP Address       Hop Delay     Node Name
       ----------       ---------     ---------
        192.0.3.2       0,00000      Campus Network.Router_2
        192.0.4.1       0,00184      Campus Network.Router_1
        192.0.3.1       0,00002      Campus Network.Router_1
        192.0.3.2       0,00184      Campus Network.Router_2

  Note that the IP addresses shown above represent
  the address of the output interface on which the
  IP datagram was routed from the corresponding
  nodes to the next node enroute to its destination
  and back.
```

Obr. 9.1: Výstupný report pre scenár s veľkou veľkosťou ICMP Echo Request/Reply správy



Obr. 9.2: Graf času odozvy na správu ICMP Echo Request pri veľkom pakete po exporte do MS Excel

10 APPLICATION CHARACTERIZATION ENVIRONMENT

Application Characterization Environment (ďalej len ACE) predstavuje voliteľný modul firmy OPNET, ktorý slúži na vizualizáciu, analýzu a hľadanie problémov v sieťových aplikáciách. Podľa [11][12] je ACE možné použiť na:

- Určenie slabých miest siete a aplikácie
- Diagnózu problémov aplikácie
- Preskúmanie navrhovaných zmien a opráv v existujúcich aplikáciách
- Predpovedanie výkonu aplikácie za rôznych podmienok pri rôznych konfiguráciách

10.1 Vlastnosti produktu ACE

Podľa [12] ACE obsahuje natívny nástroj pre zachytávanie sieťovej komunikácie pomocou, ktorú je možné zachytávať simultánne na viacerých miestach v sieti a taktiež je umožnený import zaznamenanej sieťovej komunikácie z externého programu. Pri importovaní súboru so zaznamenanou sieťovou komunikáciou je dokonca možné importovať viacero takýchto súborov naraz a tým ich spojiť do jednej jedinej aplikácie.

ACE poskytuje nástroje pre meranie šírky prenosového pásma, meranie odozvy a ostatných sieťových charakteristík. Viacstupňová korelácia umožňuje odfiltrovať nepodstatné dáta zo zachytenej komunikácie tak, aby boli viditeľné len relevantné dáta. Taktiež je možné generovať veľké toky dát, ktoré umožňujú predikovať, aký efekt bude mať veľké nasadenie, v ktorom by diskrétna simulácia tisícok transakcií trvala buď veľmi dlho alebo by vyžadovala veľké množstvo pamäte. [12]

Diskrétne simulácie v ACE umožňujú generovať sieťové modely z ACE úloh a predikovať výkon aplikácie v širokom spektre „čo ak“ scenárov. Taktiež poskytuje okamžitú spätnú väzbu na zmeny jednotlivých premenných v sieti.

V [12] sa ďalej uvádza, že ACE umožňuje navrhnúť a upravovať správanie aplikácie v grafickom prostredí a okrem iného je možné modelovať komplexné správanie aplikácie pomocou skriptov napísaných v jazyku Python.

10.2 Začiatok práce s modulom ACE

Pre vytvorenie a následnú prácu s aplikáciou v ACE je potrebne vykonať nasledovné kroky.

10.2.1 Zachytenie toku dát v aplikácii

Na zachytenie samotného toku dát v sieti je možné použiť buď nástroj na zachytávanie integrovaný v module ACE alebo nástroj tretej strany. Na modelovanie aplikácie sa používajú nespracované dáta. Aplikačné záznamové súbory, tiež známe ako záznamy sledovania paketu (Packet Trace Files), formujú tieto nespracované dáta pre ACE súbor. [12]

Integrovaný nástroj na zachytávanie sieťovej komunikácie podporuje zachytenie dát zo sieťovej prevádzky na viacerých miestach v sieti súčasne. Toto je umožnené takzvanými agentmi zachytávania, ktorých môže byť v sieti nainštalovaných ľubovoľne veľa. Títo agenti sú spravovaní manažérom zachytávania. Bohužiaľ akademická verzia IT Guru nezahŕňa zachytávanie sieťovej komunikácie, je možné pracovať len s hotovými súbormi. [12]

V publikácii [12] sa uvádza aj to, že ACE podporuje SSL šifrovanie komunikácie medzi agentmi a manažérom, medzi ktorými môžu byť dáta prenášané s použitím anonymného šifrovania v situáciách keď nie je nutné použiť autentifikáciu alebo s použitím certifikovaného šifrovania v situáciách kedy je nutné použiť autentizáciu. Agenti podporujú špeciálny mód. Kde zachytávajú čisto len informácie z hlavičky IP, TCP a UDP protokolov, ale žiaden iný protokol ani aplikačné dáta.

10.2.2 Import záznamového súboru

ACE obsahuje sprievodcu importom, kde je užívateľ v procese importu záznamového súboru vedený krok po kroku. Tento sprievodca často vyžaduje dodatočné informácie o topológii siete a jej vlastnostiach, ako napríklad šírka prenosového pásma. Keď sú záznamy importované, ACE tieto dáta analyzuje a vytvorí z nich jednu reprezentáciu výmeny. Táto reprezentácia sa nazýva úloha aplikácie (Application Task). V ACE je taktiež možné import viacerých záznamových súborov naraz, čo vedie k ich zlúčeniu do jedného presného a realistického modelu aplikácie. Pomocou ACE nástroja pre zachytávanie a nástroja pre import je možné modelovať vysokokomplexnú aplikačnú výmenu v rozsiahlych sieťach. ACE taktiež podporuje import záznamových súborov z externých programov ako napríklad Sniffer alebo TCPdump. [12]

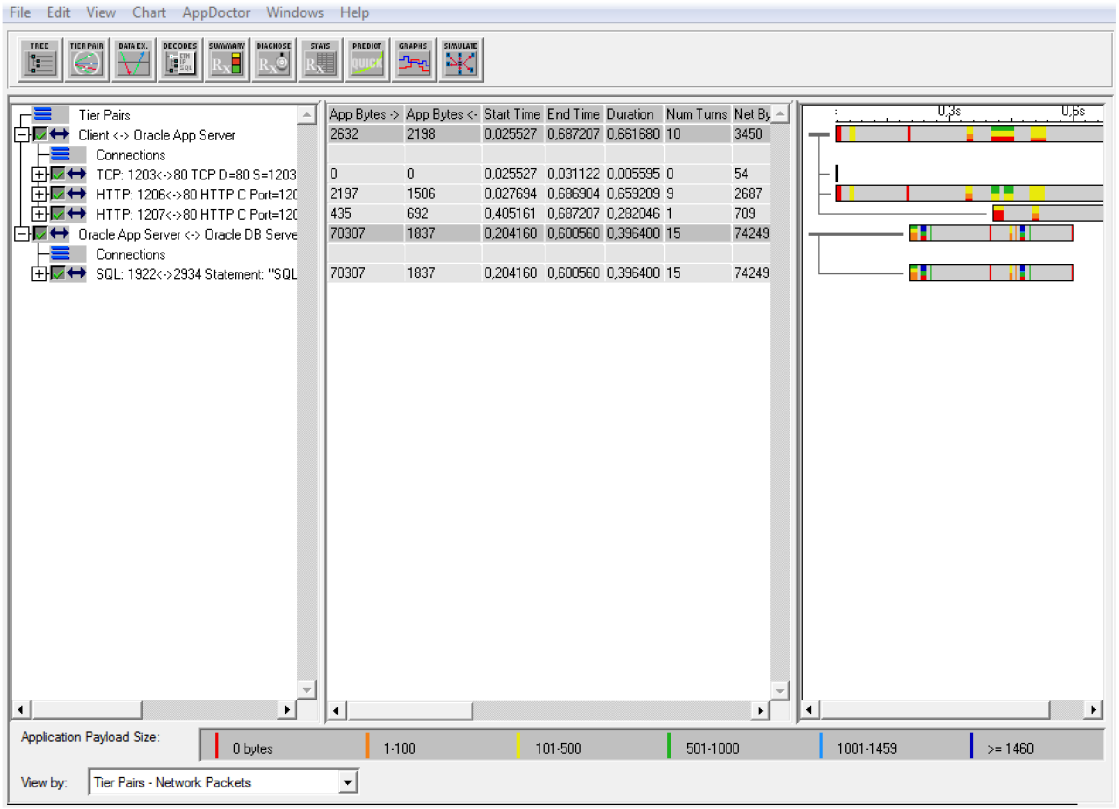
10.3 Ďalšie možnosti práce s modulom ACE

Po vytvorení ACE súboru je s týmto súborom možné vykonávať rôzne operácie vzhľadom k povahe tejto aplikácie a k žiadanému výsledku.

10.3.1 Analýza a diagnostika aplikácie

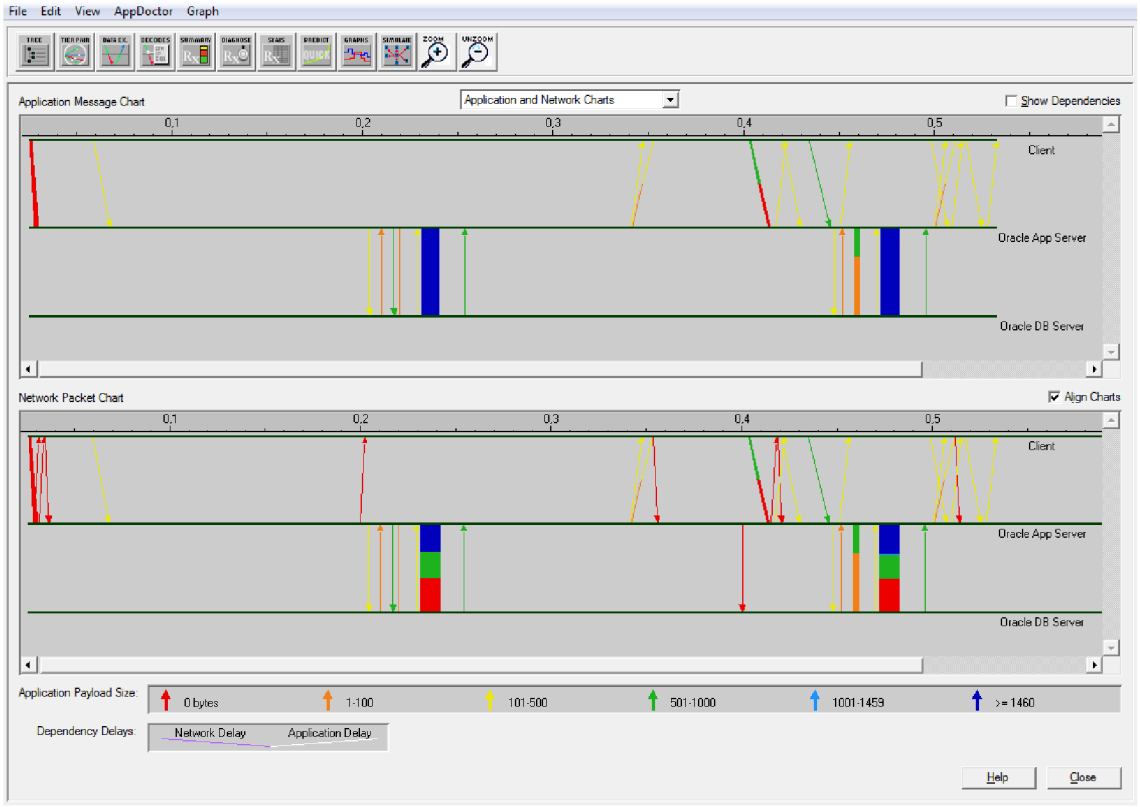
V [12] sa ďalej spomína, že na vizualizáciu ACE aplikácie sú použité tri stránky rozdelené do samostatných záložiek, ktoré je možné prehliadať naraz v jednom momente: *Treeview* okno, *Data Exchange Charts* a *Tier Pair Circle*.

Okno *Treeview* (Obr. 10.1) umožňuje prezeranie jednotlivých uzlov siete a spojení týchto uzlov siete v prehľadnej stromovej štruktúre. Toto okno zároveň sumarizuje informácie a časy jednotlivých transakcií, ktoré tvoria aplikáciu. [12]



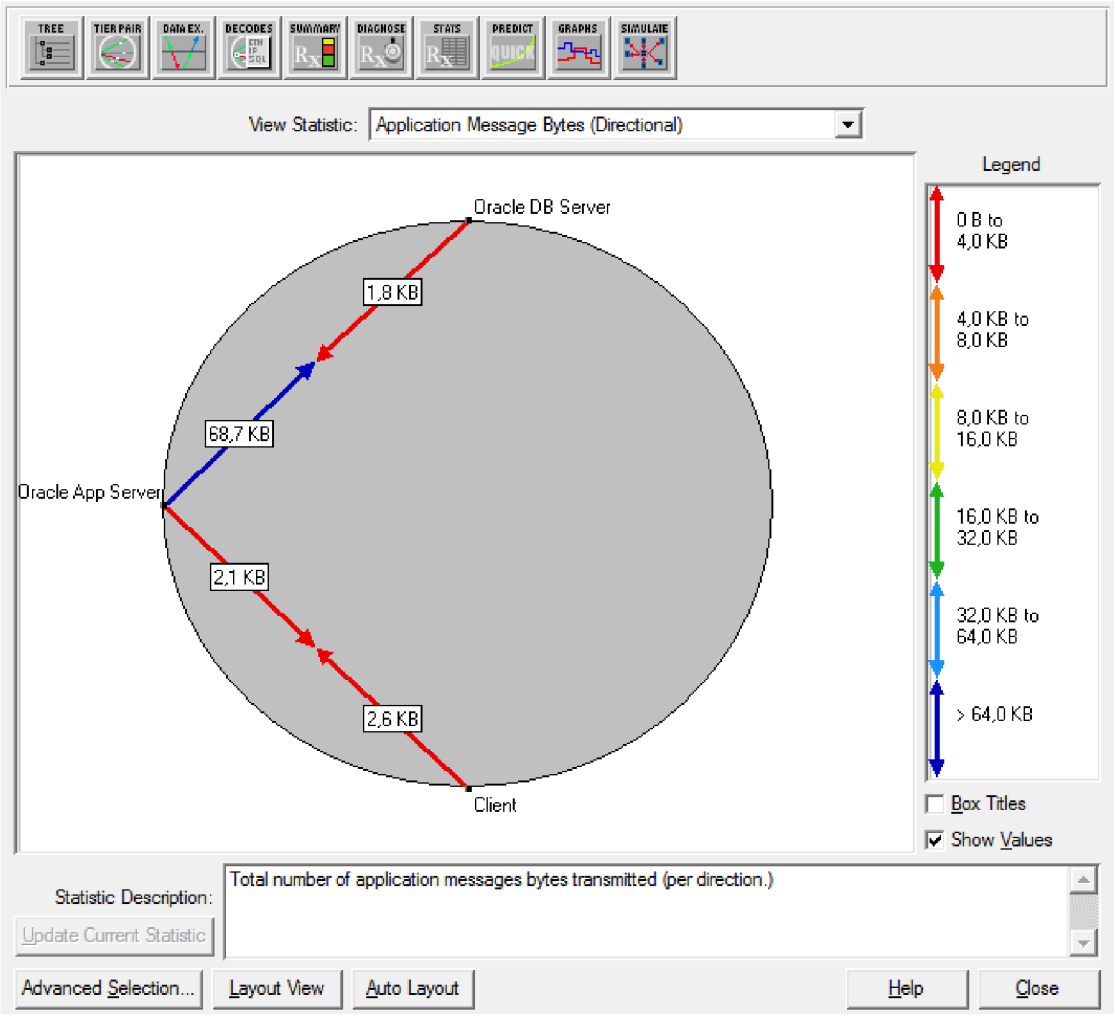
Obr. 10.1: ACE – okno Treeview

Okno *Data Exchange Charts* (Obr. 10.2) sa podľa [12] využívajú zobrazenie celkového toku dát medzi jednotlivými uzlami siete na časovej osi. Aplikáciu je tu možné sledovať ako zo sieťovej vrstvy, tak aj z aplikačnej. Čiže je možné napríklad určiť príliš neefektívne aplikácie alebo poukázať na oneskorenie v sieti alebo v aplikácii.



Obr. 10.2: ACE – okno Data Exchange Charts

Okno *Tier Pair Circle* (Obr. 10.3) sa používa na zobrazenie komunikácie medzi všetkými uzlami v ACE súbore. Umožňuje ľahko určiť, ktoré uzly práve komunikujú/nekomunikujú, zobraziť všeobecné informácie o každej tejto komunikácii a taktiež ukázať nulovú komunikáciu tam, kde je to pre nás irelevantné. [12]



Obr. 10.3: ACE – okno Tier Pair Circle

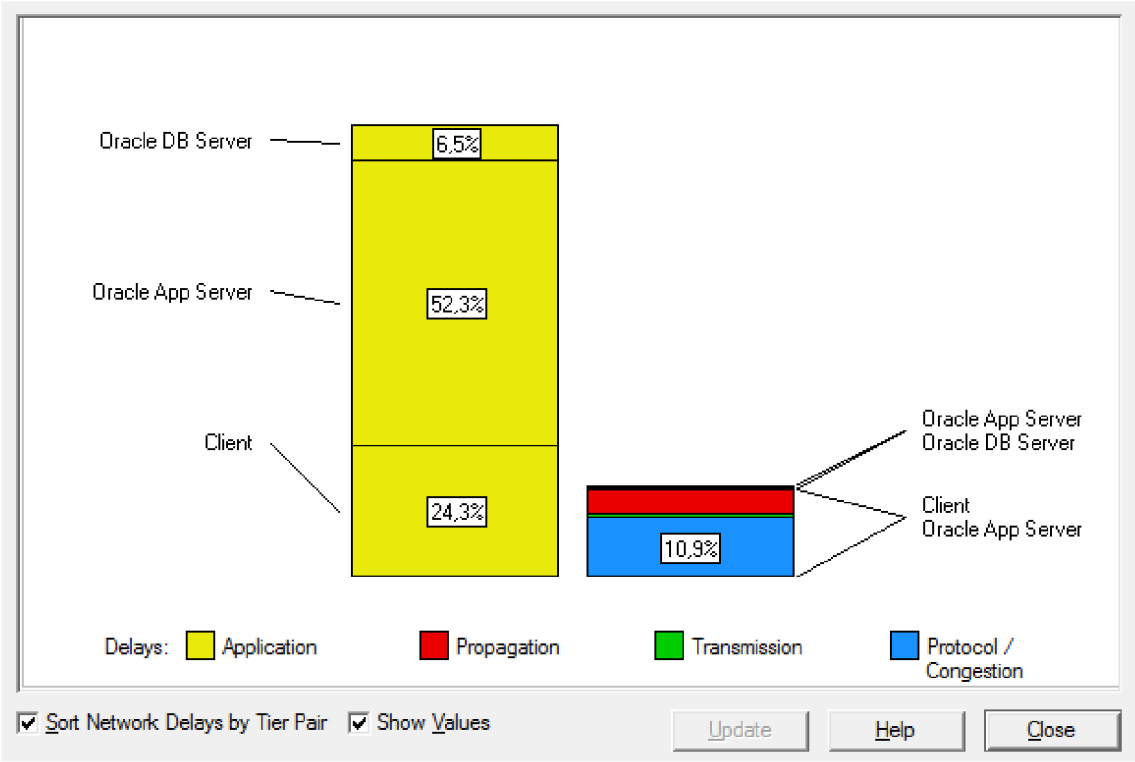
10.3.2 Analýza aplikácie a siete

Autori v [12] popisujú taktiež, že ACE poskytuje nástroje na analýzu už existujúcej aplikácie *AppDoctor*, ktorý umožňuje analýzu na vysokej úrovni a hlbšie skúma aplikačné a dáta protokolu dosiahnuté v individuálnych paketoch. *AppDoctor* má niekoľko súčastí.

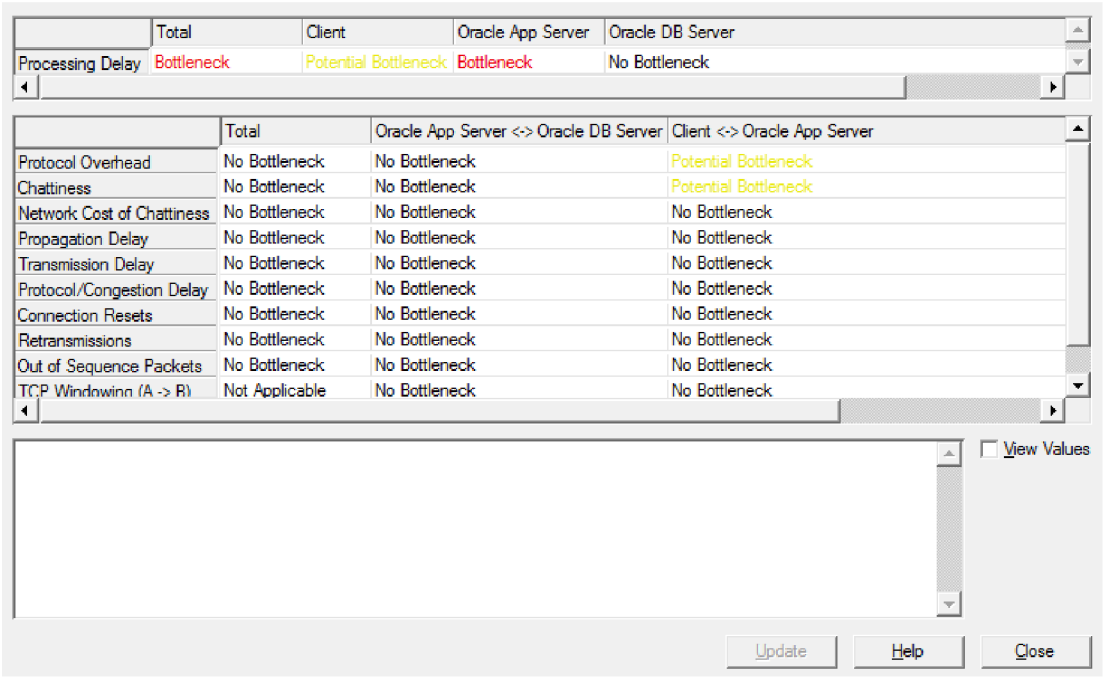
Summary of Delays (sumár oneskorení) (Obr. 10.4) rozdeľuje celkový čas odozvy aplikácie do samostatných komponent sieťového a aplikačného oneskorenia. [12]

Diagnosis (diagnóza) (Obr. 10.5) – identifikuje a diagnostikuje slabé miesta (bottlenecks) a potenciálne slabé miesta (potential bottlenecks) v aplikácii a v sieti. [12]

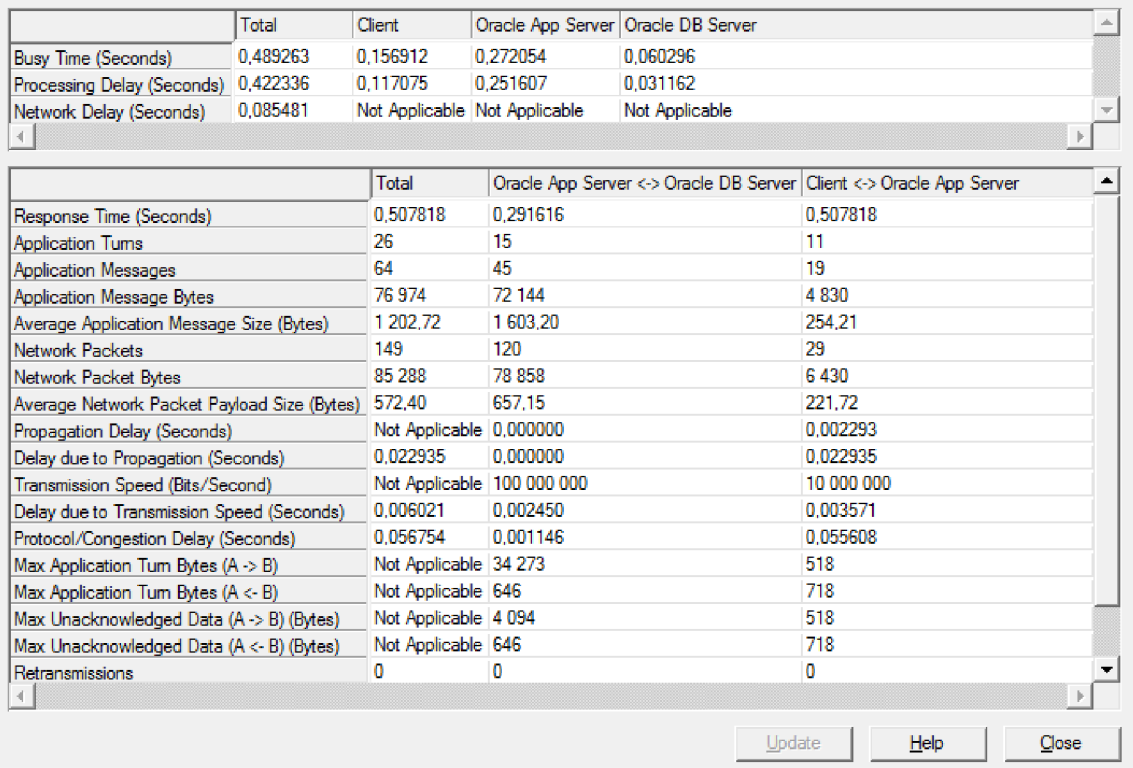
Statistics (štatistika) (Obr. 10.6) – poskytuje detailné štatistiky pre rôzne merania siete a výkonu aplikácie. [12]



Obr. 10.4: ACE – okno AppDoctor – Summary of Delays



Obr. 10.5: ACE – okno AppDoctor – Diagnosis



	Total	Client	Oracle App Server	Oracle DB Server
Busy Time (Seconds)	0,489263	0,156912	0,272054	0,060296
Processing Delay (Seconds)	0,422336	0,117075	0,251607	0,031162
Network Delay (Seconds)	0,085481	Not Applicable	Not Applicable	Not Applicable

	Total	Oracle App Server <-> Oracle DB Server	Client <-> Oracle App Server
Response Time (Seconds)	0,507818	0,291616	0,507818
Application Turns	26	15	11
Application Messages	64	45	19
Application Message Bytes	76 974	72 144	4 830
Average Application Message Size (Bytes)	1 202,72	1 603,20	254,21
Network Packets	149	120	29
Network Packet Bytes	85 288	78 858	6 430
Average Network Packet Payload Size (Bytes)	572,40	657,15	221,72
Propagation Delay (Seconds)	Not Applicable	0,000000	0,002293
Delay due to Propagation (Seconds)	0,022935	0,000000	0,022935
Transmission Speed (Bits/Second)	Not Applicable	100 000 000	10 000 000
Delay due to Transmission Speed (Seconds)	0,006021	0,002450	0,003571
Protocol/Congestion Delay (Seconds)	0,056754	0,001146	0,055608
Max Application Turn Bytes (A -> B)	Not Applicable	34 273	518
Max Application Turn Bytes (A <- B)	Not Applicable	646	718
Max Unacknowledged Data (A -> B) (Bytes)	Not Applicable	4 094	518
Max Unacknowledged Data (A <- B) (Bytes)	Not Applicable	646	718
Retransmissions	0	0	0

Obr. 10.6: ACE – okno AppDoctor – Statistics

10.3.3 Predikcia výkonu aplikácie

Ako sa uvádza v [12], ACE aktívne využíva OPNETom vyvinutú technológiu prediktívneho modelovania na určenie toho, ako zmeny v sieti alebo aplikácii ovplyvnia výkonnosť aplikácie. ACE obsahuje štyri hlavné nástroje na predikciu: *QuickPredict*, *QuickRecode*, simulácie diskrétnych udalostí a ACE toky dát.

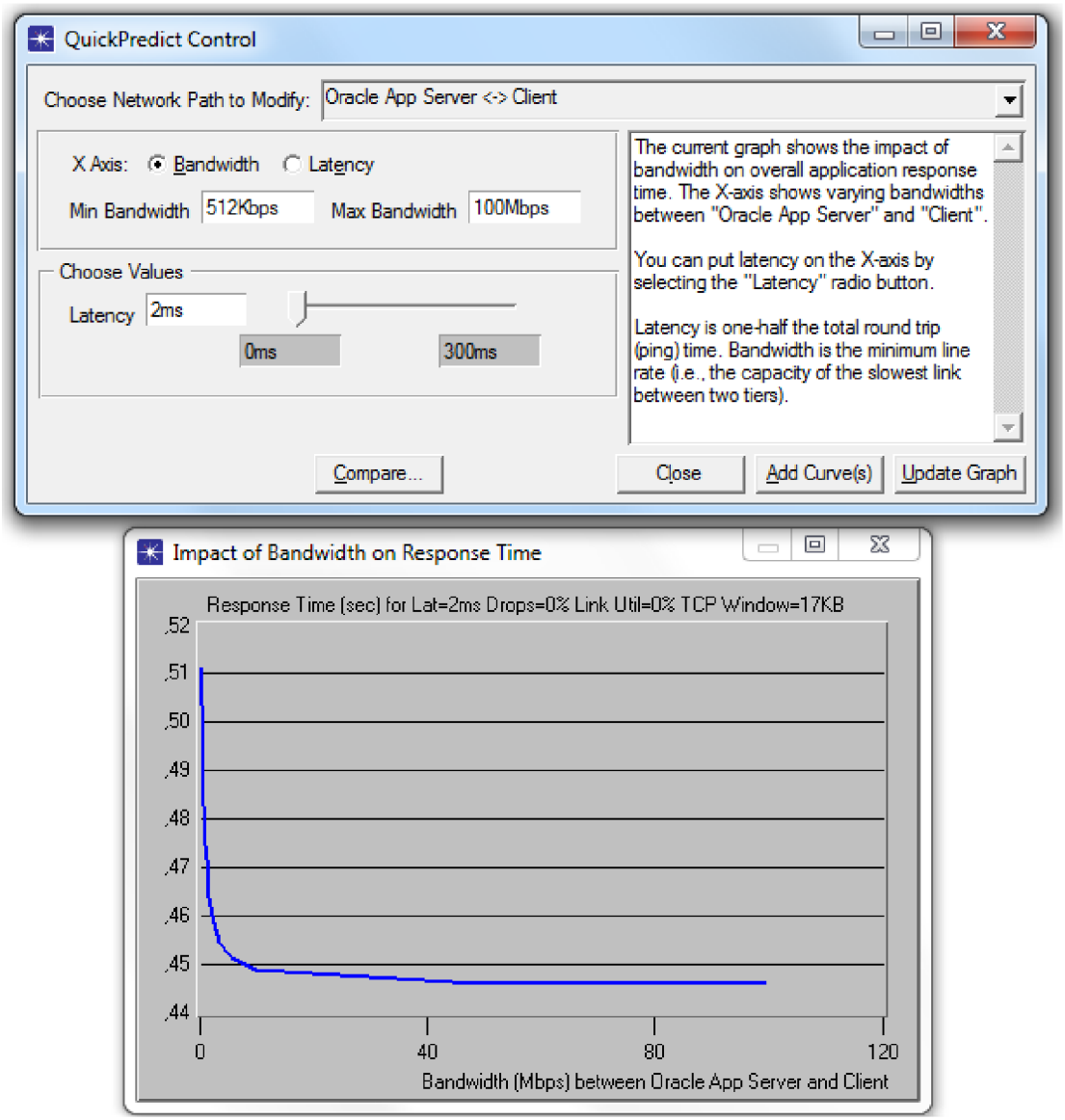
Nástroj *QuickPredict* (Obr. 10.7) poskytuje okamžitú spätnú väzbu na to, ako rôzne variácie siete ovplyvňujú výkon aplikácie. Tento nástroj umožňuje rýchlo zodpovedať otázky ako napr. „Pomôže zdvojnásobenie prenosovej rýchlosti linky medzi dvomi uzlami znížiť čas odozvy našej aplikácie?“. [12]

Nástroj *QuickRecode* umožňuje predikovať ako navrhované zmeny a opravy ovplyvnia výkon aplikácie. Tento nástroj funguje tak, že užívateľ si z pôvodnej aplikácie vytvorí ďalšiu, hypotetickú, a tú porovná s tou pôvodnou. *QuickRecode* umožňuje riešiť otázky ako napr. „Aký vplyv by malo na aplikáciu, keby databáza bola schopná prenášať záznam v 30 správach, namiesto aktuálnych 300?“. [12]

Simulácia diskrétnych udalostí ACE umožňuje naplno využiťsimulačnú technológiu OPNETu založenú na udalostiach a knižnicu modelov na testovanie v širokej škále sietí.

Napríklad je možné vytvoriť virtuálne nasadenie FTP aplikácie, aby bolo možné otestovať dôsledky viacerých súčasných sťahovaní na danej aplikácii, ale aj vplyv na celkový čas odozvy. [12]

Nástroj *Data Flow* je nesmierne užitočný pri potrebe študovania dôsledkov masívnych nasadení, kde by simulácia pomocou diskrétnych udalostí stoviek alebo tisícok transakcií trvala neúnosne dlho alebo by vyžadovala veľké množstvo pamäte. [12]



Obr. 10.7: ACE – okno QuickPredict

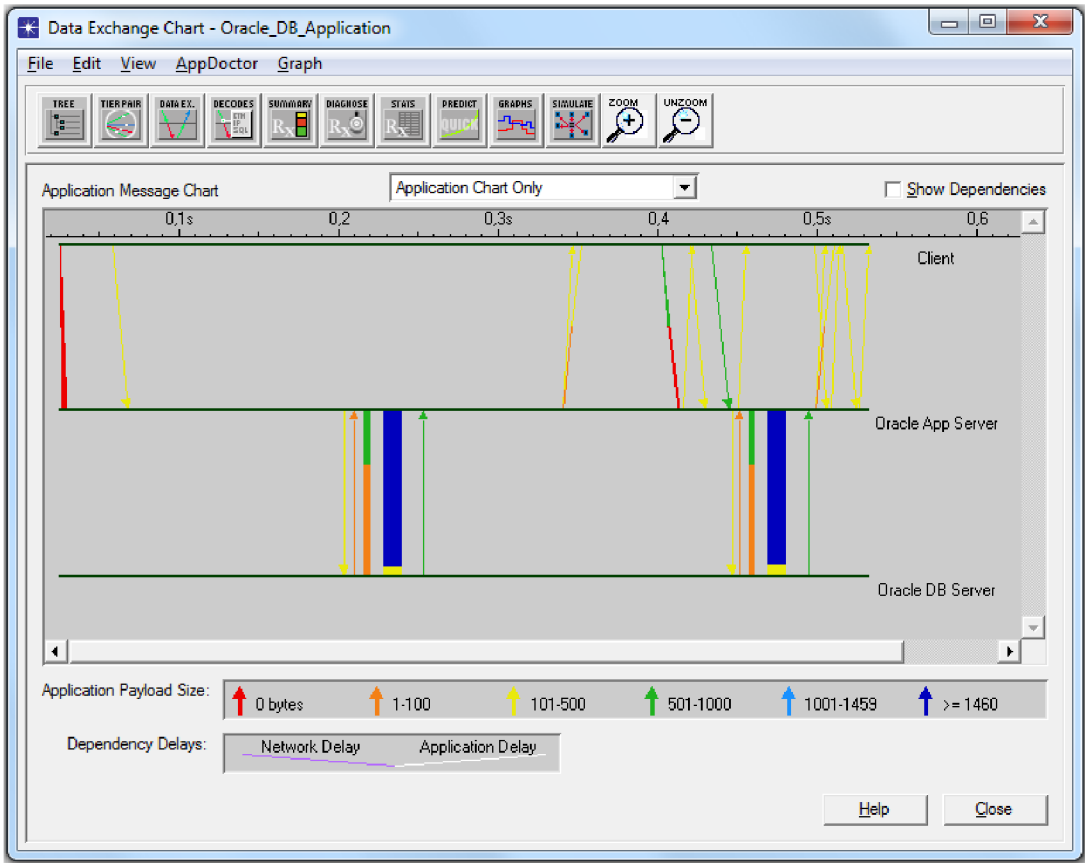
11 LABORATÓRNA ÚLOHA: ACE

V tejto laboratórnej úlohe budeme analyzovať výkon databázovej aplikácie [12]. Budeme skúmať prípadné slabé miesta aplikácie. Potrebná sieťová prevádzka už bola zachytená a je súčasťou IT Guru, keďže akademická verzia neumožňuje zachytávať sieťovú komunikáciu (OPNET Modeler to však umožňuje). IT Guru preto ponúka vzorový súbor **Oracle_DB_Application**, s ktorým budeme pracovať v našej laboratórnej úlohe.

11.1 Zoznámenie sa s aplikáciou v ACE

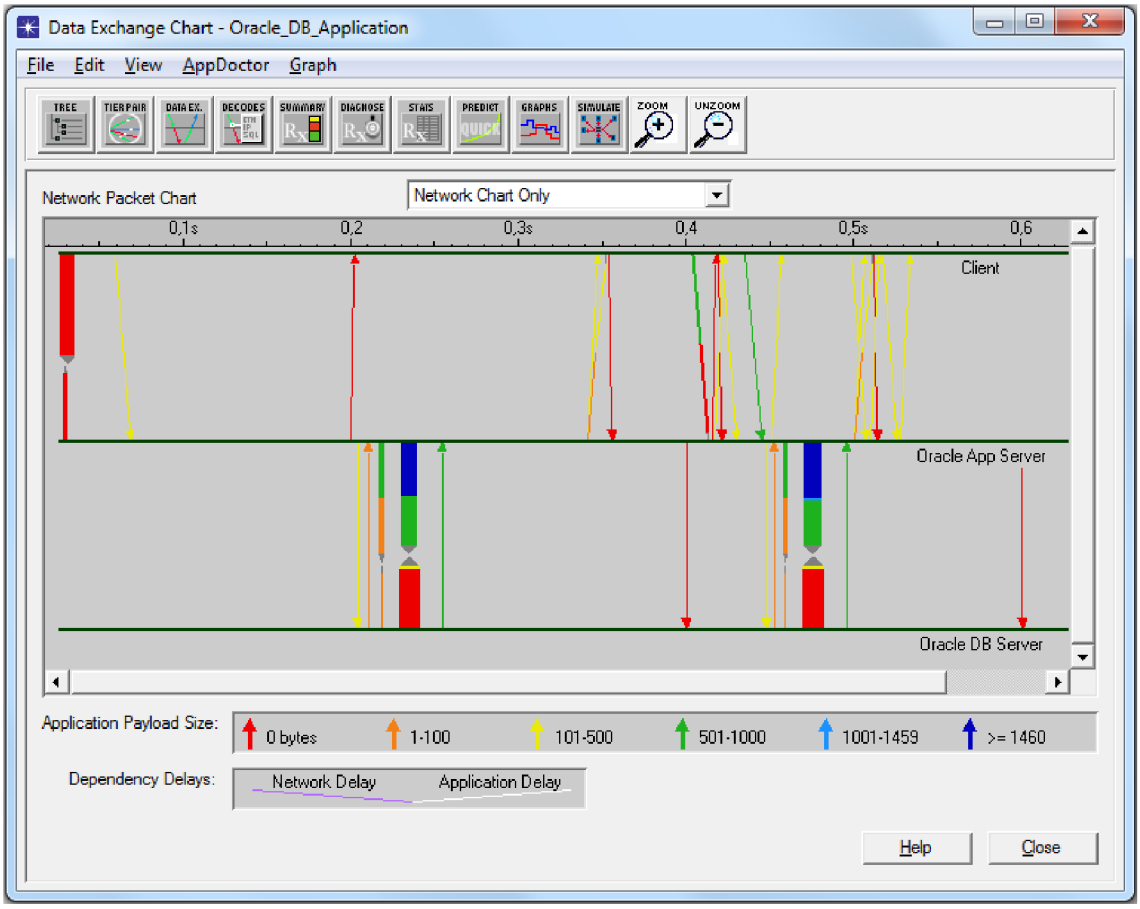
1. Spustíme OPNET IT Guru Academic Edition.
2. V menu zvolíme **File** → **Open...**, z rolovacieho menu vyberieme **Application Characterization** a vyberieme **Oracle_DB_Application**. Potvrdíme stlačením **OK**.

Po otvorení súboru sa zobrazí okno Data Exchange Chart, ktoré zobrazuje tok dát medzi uzlami siete. V našom prípade máme v sieti tri uzly: klienta, aplikačný server Oracle a databázový server Oracle. Polohu zobrazovaných uzlov v okne Data Exchange Chart je možné zmeniť jednoduchým premiestnením daného uzla v okne, napr. zhora dole.



Obr. 11.1: Okno Data Exchange Chart

3. Z rolovacieho menu v strede okna zvolíme **Network Chart Only**
4. V menu klikneme na **View → Split Groups**, čo nám prehľadnejšie zobrazí tok správ v jednotlivých smeroch. Farby indikujú veľkosť danej správy.



Obr. 11.2: Okno Data Exchange Chart s oddeleným tokom dát

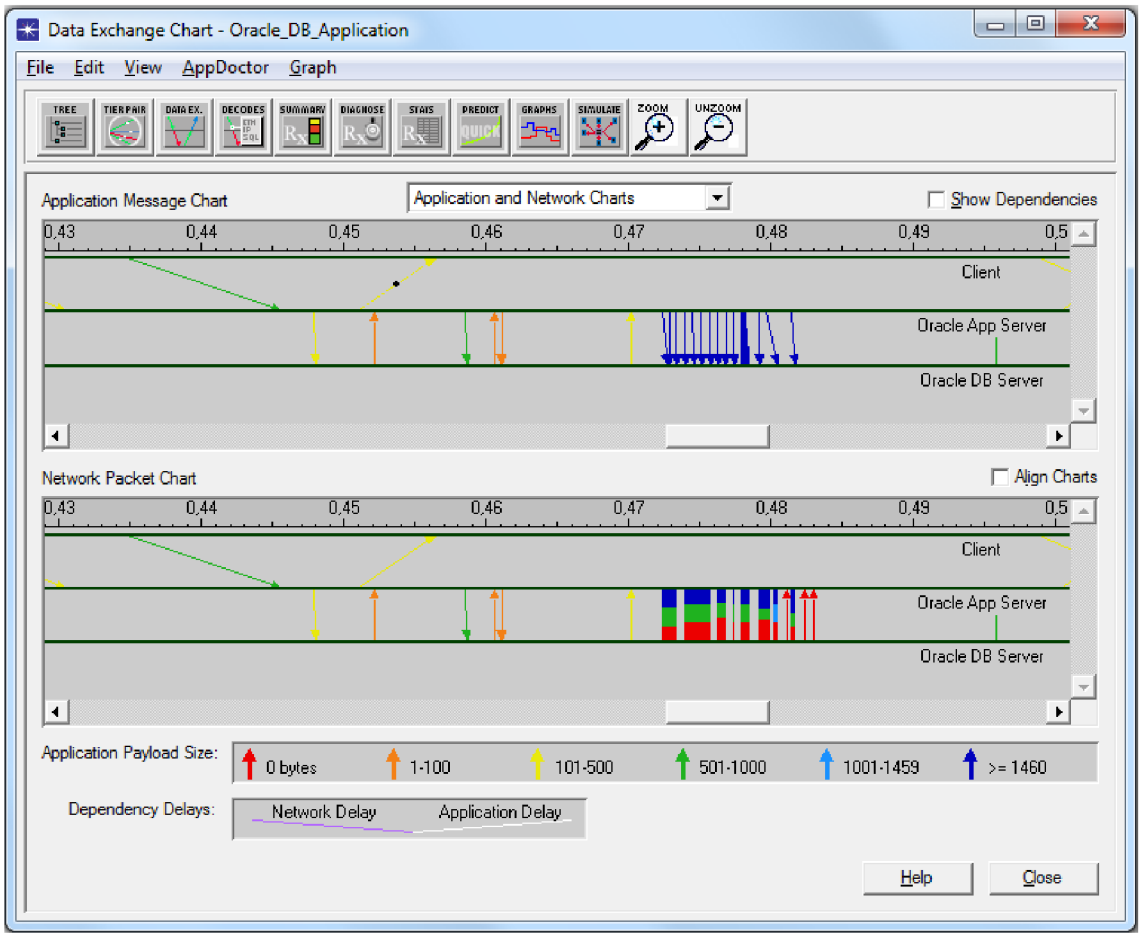
Klient v sieti posiela požiadavky na databázový aplikačný server, ktorý slúži ako medzivrstva medzi klientom a databázou a umožňuje klientovi obdržať odpoveď z databázy napríklad vo forme HTML, PDF alebo iných formátoch.

Aby sme lepšie porozumeli toku dát medzi jednotlivými uzlami, detailnejšie si zobrazíme transakcie.

5. V rolovacom menu v strede okna zvolíme **Application and Network Charts** a vo **View → Split Groups** túto možnosť odznačíme, aby dátový tok už nebol rozdelený na jednotlivé skupiny.
6. Vyberieme presnú transakciu vo **View → Set Visible Time Range**, kde nastavíme **Start Time** na **0.43** a **End Time** na **0.5**. Potvrdíme tlačidlom **OK**.

Aplikácia zobrazuje sériu správ medzi klientom a aplikačným serverom a medzi servermi. Ak klikneme na dané spojenie a podržíme nad ním myš, v tooltipe sa nám zobrazí informácia o spojení, napríklad, že aplikačný server práve posiela klientovi

HTML odpoveď cez protokol HTTP, ako je to v prípade transakcie medzi 0,45s a 0,46s medzi aplikačným serverom a klientom.



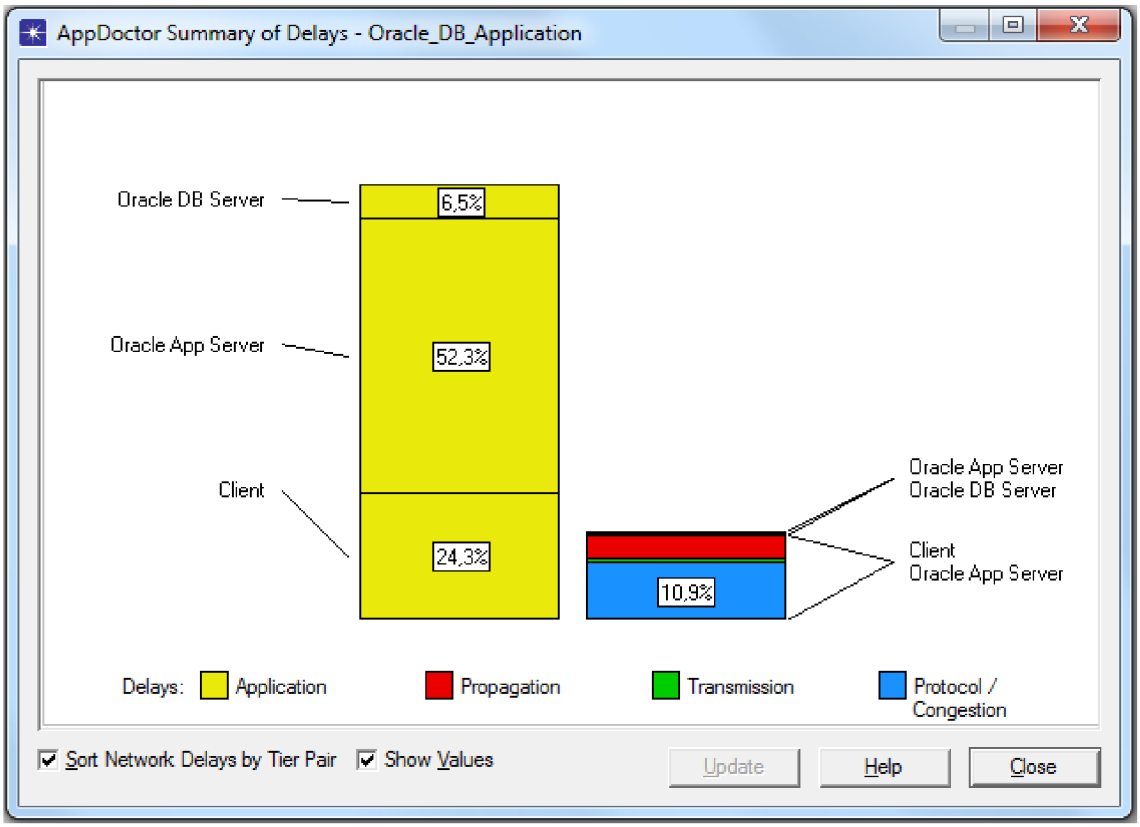
Obr. 11.3: Okno Data Exchange Chart s priblíženými transakciami

Okno Network Chart (na predchádzajúcom obrázku zobrazené dole) vyjadruje tok dát v sieti. Pri kliknutí na jednotlivé šípky uvidíme, ktorému protokolu patrí daná transakcia. Vidíme, že je prítomných aj veľa správ červenej farby, ktoré majú nulovú veľkosť payloadu, tzv. užitočných dát. V tomto prípade ide o správy protokolu TCP, ktoré potvrdzujú doručovanie segmentov.

11.1.1 Analýza pomocou nástroja AppDoctor

Nástroj AppDoctor Summary of Delays poskytuje bližší pohľad na príčinu celkového oneskorenia aplikácie.

7. V menu zvolíme **AppDoctor → Summary of Delays**, kde zaškrtneme **Show Values**.



Obr. 11.4: Okno AppDoctor Summary of Delays

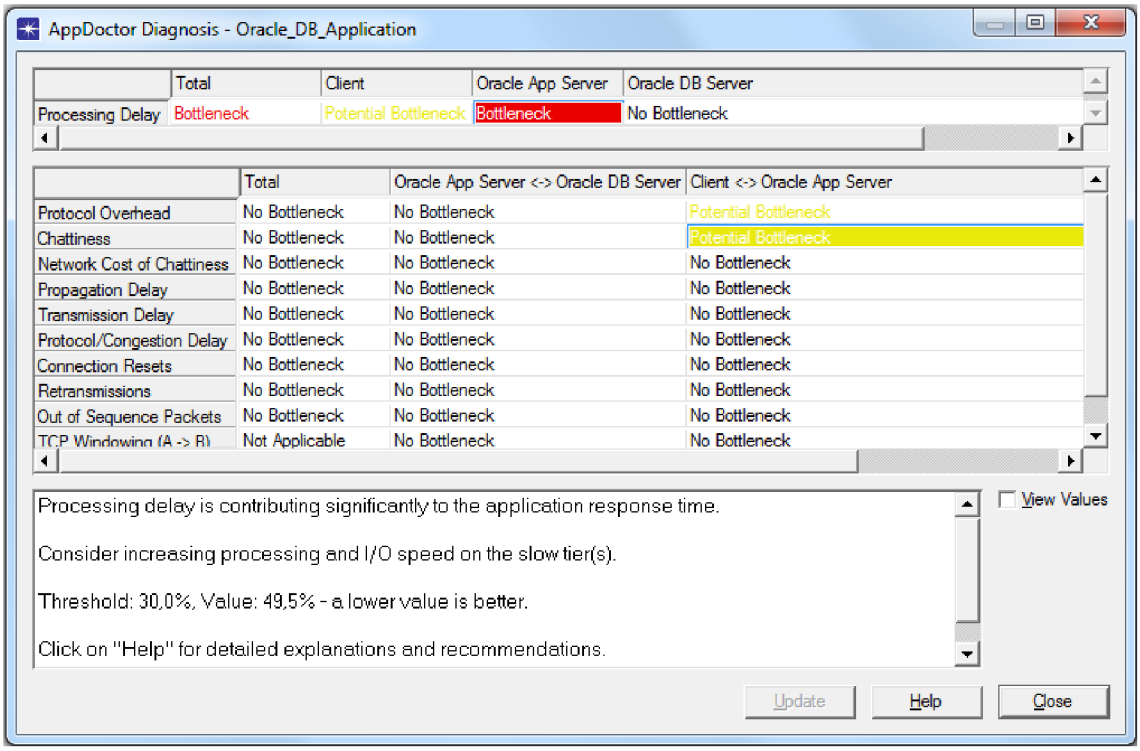
Ako si môžeme všimnúť, najväčší podiel na oneskorení má aplikačný server, pretože slúži ako rozhranie na komunikáciu medzi klientom a databázovým serverom. Nezanedbateľnú položku ešte tvorí oneskorenie spôsobené protokolom, pretože spojenia je potrebné najprv budovať a potom jednotlivé správy potvrdzovať.

8. Zavrieme okno **AppDoctor Summary of Delays**.

Nástroj AppDoctor Diagnosis poskytuje bližší pohľad na príčinu oneskorenia spôsobeného protokolom.

9. V menu zvolíme **AppDoctor → Diagnosis**. Zobrazí sa nám okno **AppDoctor Diagnosis**, kde v určitých riadkoch vidíme červený nápis **Bottleneck** alebo žltý nápis **Potential Bottleneck**.

10. Klikneme na slovo **Bottleneck** a zobrazí sa nám príčina daného bottlenecku, teda slabého miesta aplikácie, a aj prípadné odporúčanie, ako je možné tento problém vyriešiť.



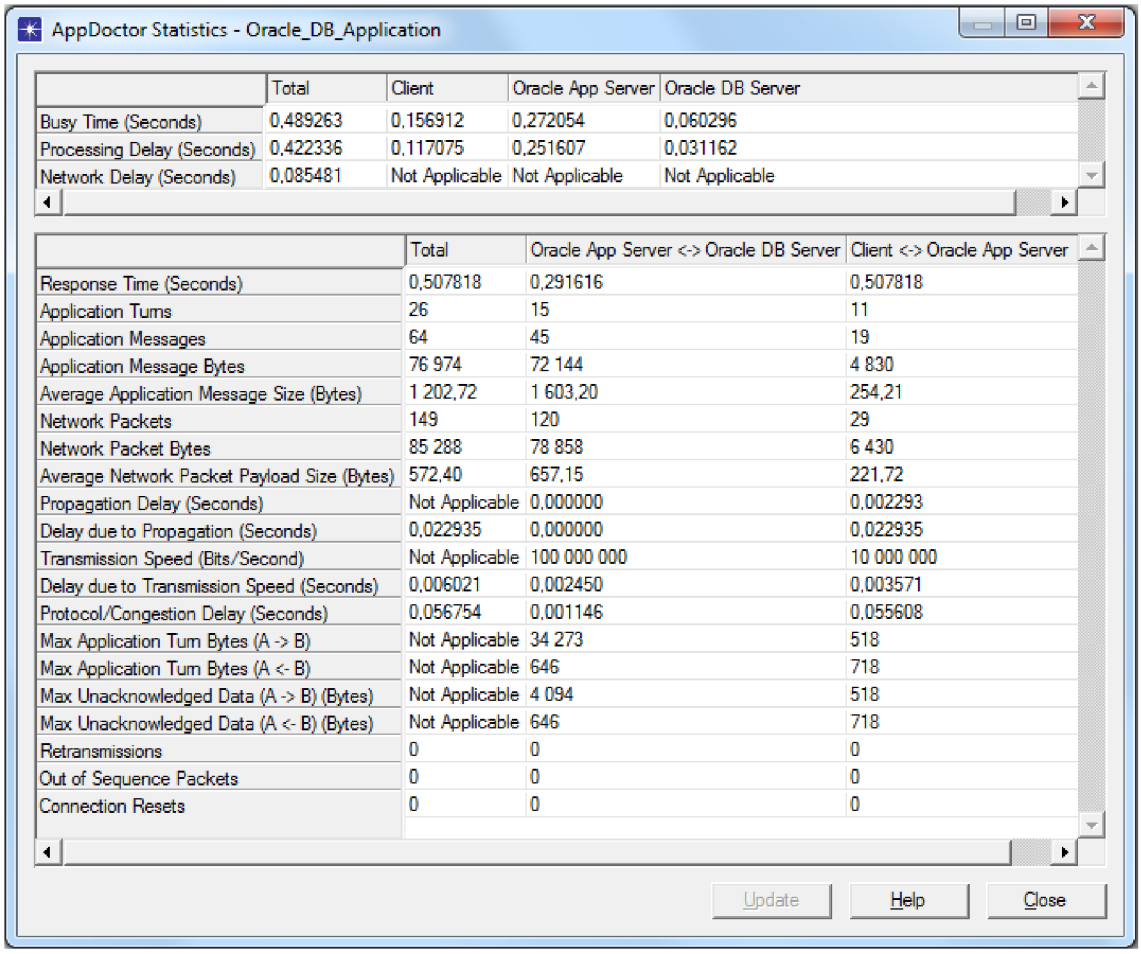
Obr. 11.5: Okno AppDoctor Diagnosis

V tomto prípade diagnostické okno ukazuje jedno slabé miesto a tri potenciálne slabé miesta. Najproblematickejším uzlom v sieti sa zdá byť aplikačný server, ktorého kapacity zrejme nepostačujú a preto spôsobuje oneskorenie v spracovaní informácií. Ďalším problémom v budúcnosti by mohla byť veľká „komunikatívnosť“ (chattiness) aplikačného servera, ktorý má veľký podiel miestami neefektívnych požiadaviek a odpovedí.

11. Zavrieme okno **AppDoctor Diagnosis**.
12. V menu zvolíme **AppDoctor → Statistics**.

Zobrazí sa nám okno so sieťovými štatistikami, kde si môžeme všimnúť, že aplikácia má veľký *Processing Delay*, čiže oneskorenie spôsobené spracovaním, ktoré spôsobuje najmä aplikačný server.

13. Zavrieme okno **AppDoctor Statistics**.

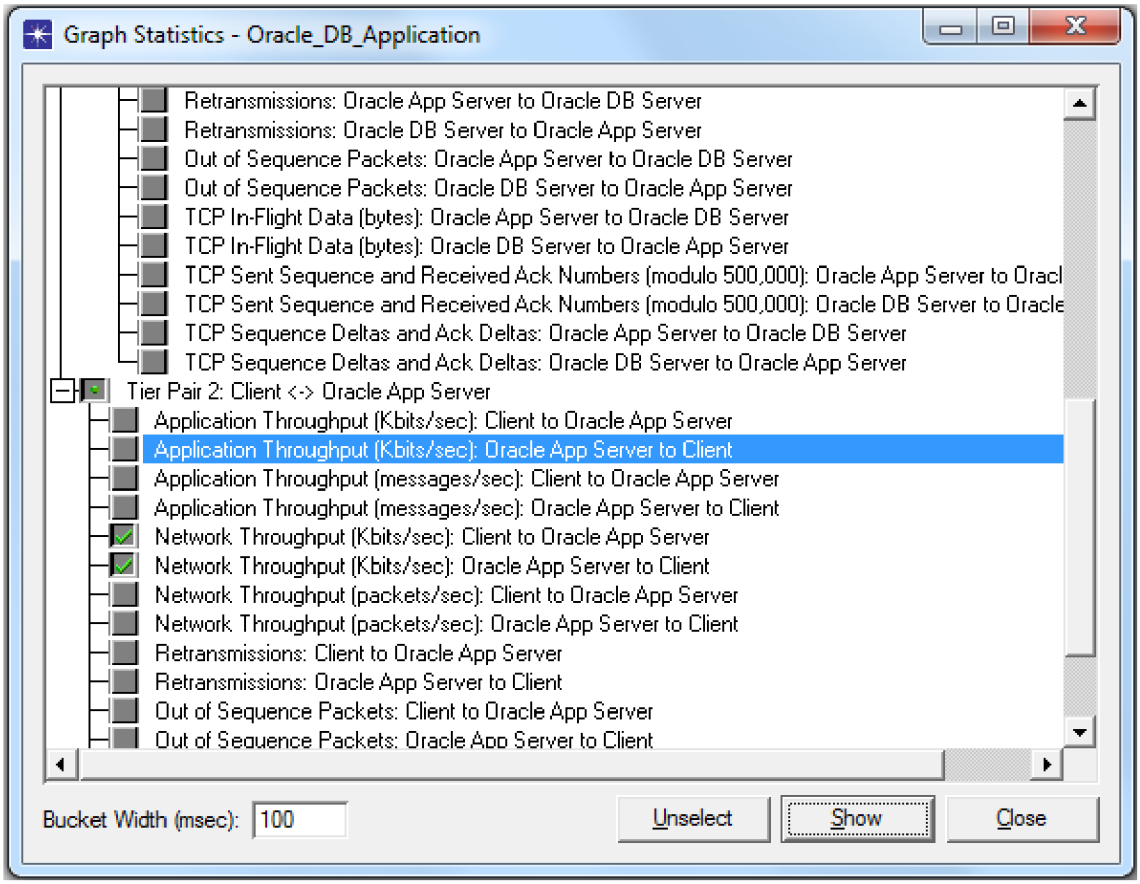


Obr. 11.6: Okno AppDoctor Statistics

11.1.2 Preskúmanie štatistík

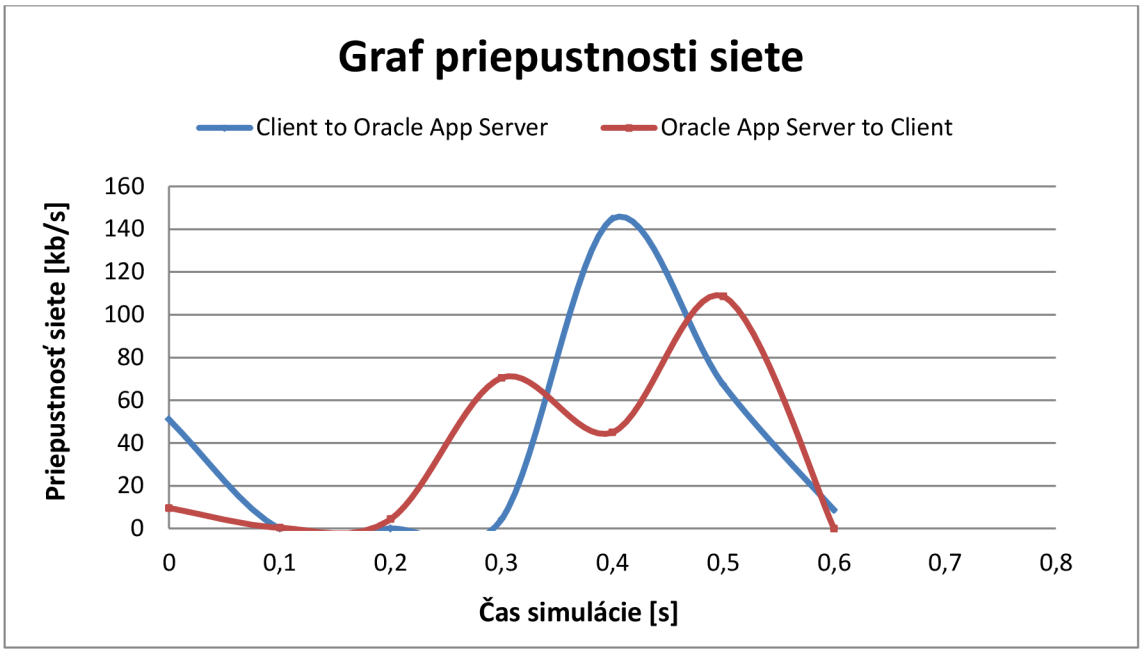
Aby sme zistili skutočnú priepustnosť siete, použijeme nástroj Graph Statistics.

14. V okne **Data Exchange Chart** zvolíme **Graph → Graph Statistics**. Zobrazí sa nám okno so všetkými štatistikami siete, ktoré si môžeme nechať zobraziť.
15. V **Tier Pair 2: Client ↔ Oracle App Server** zaškrtneme **Network Throughput (kbit/sec): Client to Oracle App server** a **Network Throughput (kbit/sec): Oracle App server to client**.



Obr. 11.7: Okno Graph Statistics

16. Zmeníme hodnotu **Bucket Width** na **100 ms** a stlačíme **Show**.



Obr. 11.8: Graf priepustnosti siete

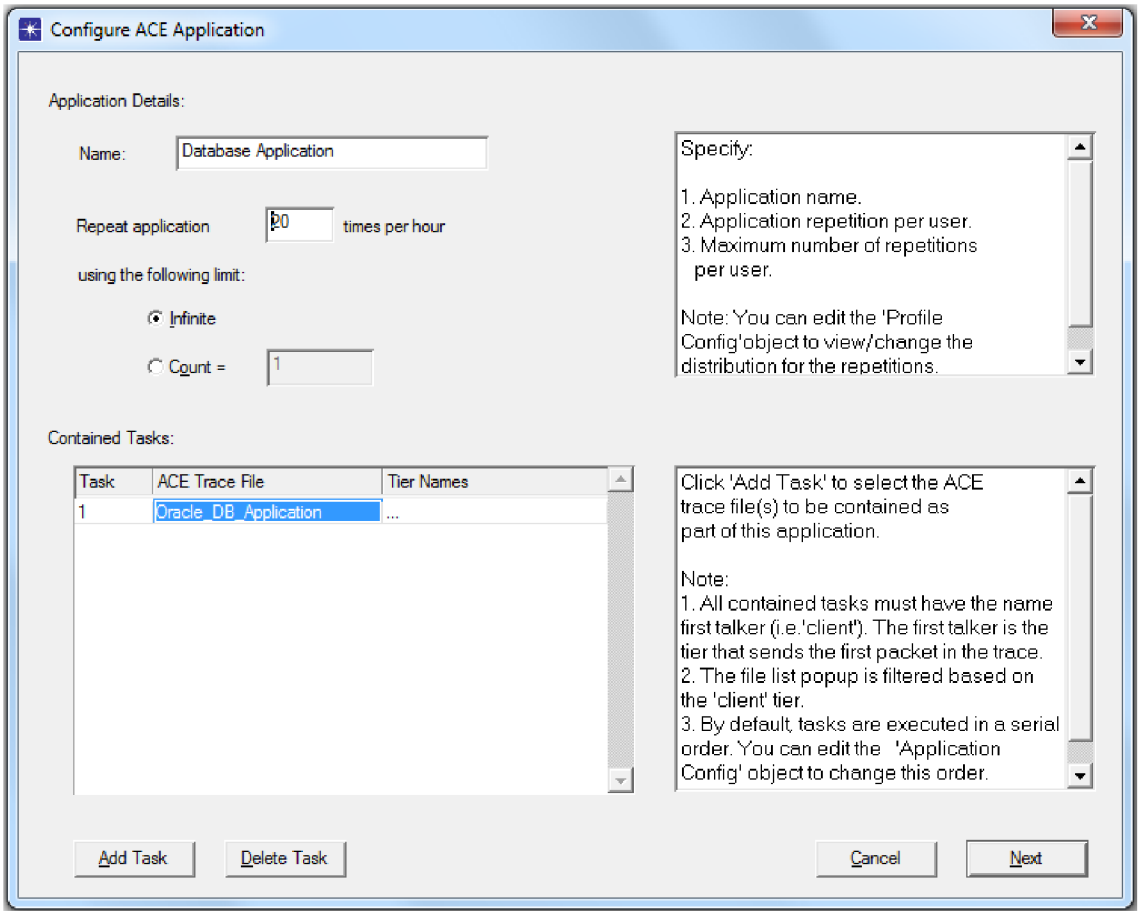
Pozn.: ACE rozdeľuje celú úlohu do tzv. časových hromád (buckets) a pomocou nich vypočíta priemernú alebo celkovú hodnotu pre daný interval. Štandardná dĺžka takejto hromady je 1 sekunda (1000 milisekúnd).

17. Zavrieme všetky otvorené grafy, aj okno **Graph Statistics**.

Vytvoríme aplikáciu, ktorá bude zodpovedať otázku, ako sa zmení časová odozva aplikácie pri rôznom zaťažení siete.

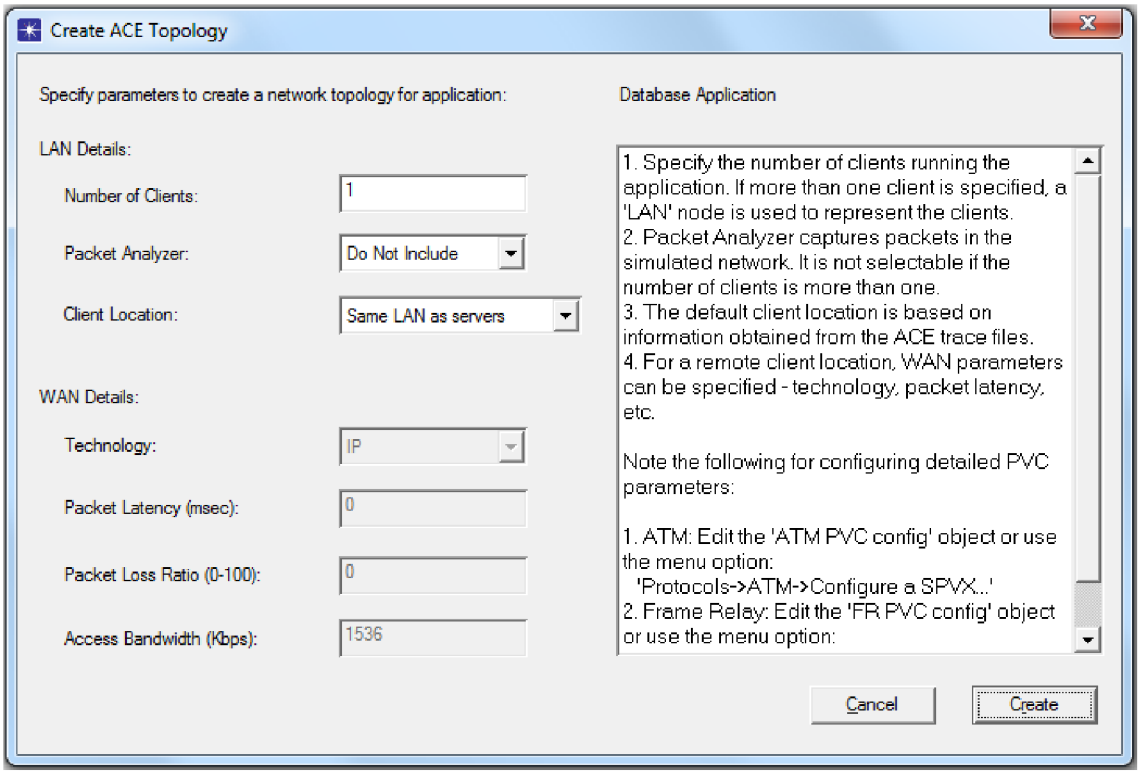
11.2 Vytvorenie projektu

18. V hlavnom okne IT Guru zvolíme **File → New...**.
19. Z ponuky vyberieme **Project** a potvrdíme stlačením **OK**. Položku **Project name** zmeníme na **vaseVUTID_DB** a **Scenario name** zmeníme na **NoBackgroundUtilization** a potvrdíme tlačidlom **OK**.
20. V okne **Startup Wizard** zvolíme **Import from ACE** a klikneme na **Next**.
21. V okne **Configure ACE Application** zmeníme name na **Database Application**. Položku **Repeat** nastavíme na **20**, čo bude vyjadrovať, koľkokrát za hodinu užívateľ spustí danú aplikáciu. Ďalej klikneme na **Add Task** a v tabuľke **Contained Tasks** klikneme na **Specify...** a vyberieme **Oracle_DB_Application**. Klikneme na **Next**.



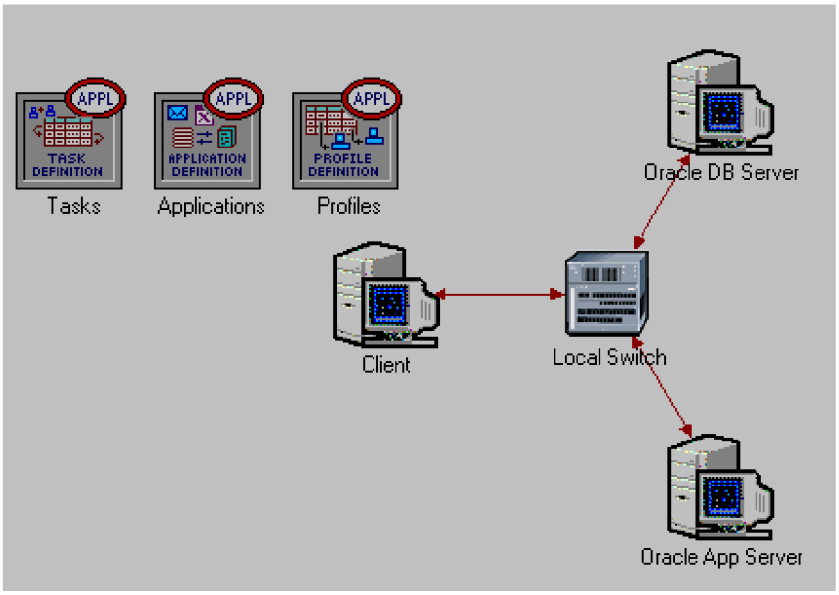
Obr. 11.9: Okno Configure ACE Application

22. V okne **Create ACE Topology** nastavíme **Number of Clients**, kde necháme **1** a klikneme na **Create**.



Obr. 11.10: Okno Create ACE Topology

23. Zobrazí sa nám okno s topológiou vytvorenej siete. Objekty **Tasks**, **Applications** a **Profiles** sú nakonfigurované podľa nastavení v sprievodcovi vytvorením ACE aplikácie.



Obr. 11.11: Topológia vytvorenej siete

11.3 Duplikácia scenára

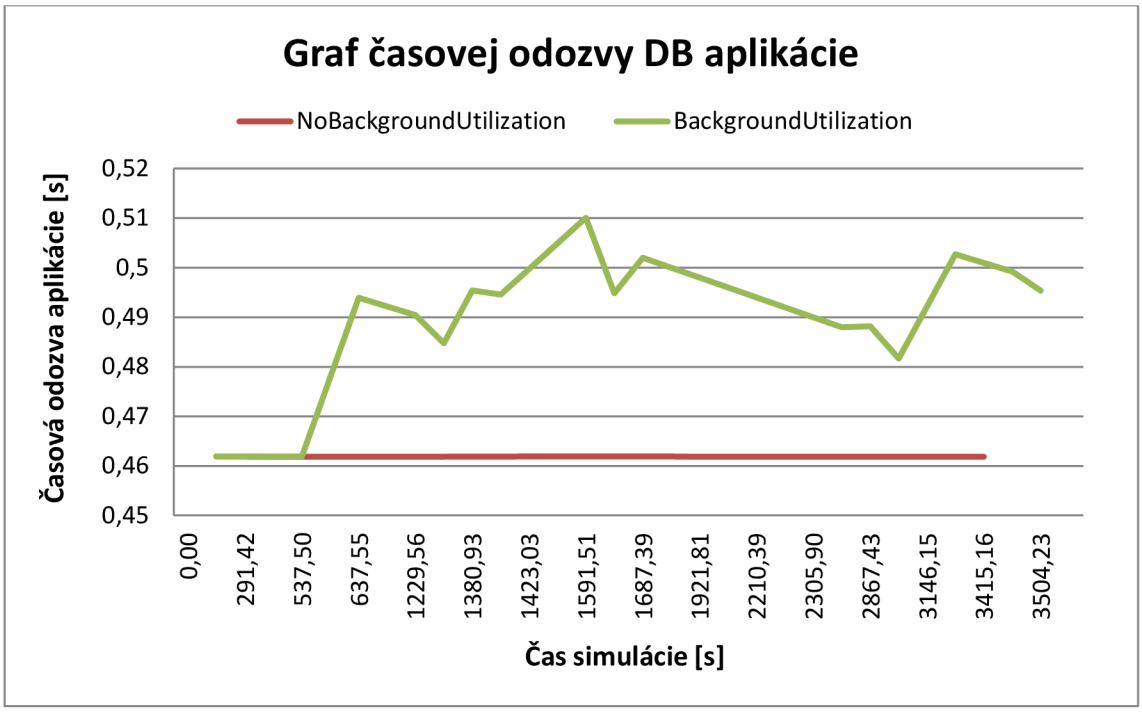
24. V menu klikneme na **Scenarios → Duplicate scenario...**. Pomenujeme ho **BackgroundUtilization**.
25. Klikneme pravým tlačidlom na linku medzi klientom a lokálnym prepínačom a vyberieme **Advanced Edit Attributes**. V časti **Background Utilization** nastavíme **time (sec)** na **600** a **background utilization (%)** na **60**. Pre linky vedúce k obidvom serverom nastavíme zaťaženie siete obdobne, ale len na 40%.

11.4 Spustenie simulácie

26. V menu klikneme na **Simulation → Configure Discrete Event Simulation...** alebo použijeme klávesovú skratku **CTRL + R**.
27. V okne **Configure Simulation** necháme vyplnené všetko tak ako je a stlačíme **OK**.
28. V menu klikneme na **Scenarios → Manage scenarios...**, kde obidva naše scenáre vyberieme tak, že v stĺpci **Results** nastavíme **<collect>** a klikneme na **OK**.
29. Keď prebehne celá simulácia, môžeme zavrieť okno **Simulation Sequence**.

11.5 Zobrazenie výsledkov

30. V menu zvolíme **Results → Compare Results...** a rozklikneme **Custom Application** a zaškrtneme štatistiku **Application Response Time (sec)** a klikneme na **Show**.



Obr. 11.12: Graf časovej odozvy DB aplikácie po exporte do MS Excel

11.6 Otázky

1. Prečo majú niektoré správy medzi Oracle databázovým serverom a Oracle aplikačným serverom veľkost 0 bytov?
2. Čo predstavujú tzv. buckets?
3. Ktorý transportný protokol je v nami vytvorenej aplikácii použitý na prenos databázových správ?

12 HODNOTENIE DOSIAHNUTÝCH VÝSLEDKOV – ACE

V laboratórnej úlohe sme sa v prvej časti úlohy prakticky oboznámili s nástrojmi, ktoré obsahuje prostredie ACE. Zisťovali sme, kde v aplikácii nám vzniká oneskorenie pomocou nástroja AppDoctor Summary of Delays. Následne pomocou nástroja AppDoctor Diagnosis sme analyzovali slabé miesta aplikácie, ktoré by mohli spôsobovať spomínané oneskorenie. V nástroji AppDoctor Statistics sme zisťovali, ktoré sieťové parametre nám spôsobujú oneskorenie. Následne sme v druhej časti vytvorili úlohu, ktorá pomáha zodpovedať hypotetickú otázku o tom, ako sa zmení časová odozva databázovej aplikácie, keď je v sieti na pozadí prevádzka. Pri jedinom klientovi sa čas odozvy aplikácie v sieti so zaťažením na pozadí, pre užívateľa citeľne nelíši od časovej odozvy v ideálnej sieti (bez zaťaženia).

12.1 Odpovede na otázky z laboratórnej úlohy

1. **Prečo majú niektoré správy medzi Oracle databázovým serverom a Oracle aplikačným serverom veľkost 0 bytov?**

Niektoré správy v smere od databázového servera k aplikačnému serveru a naopak majú nulovú veľkosť z dôvodu, že nenesú žiadne užitočné dáta (tzv. payload), pretože sú to len potvrdenia od protokolu TCP. V tomto prípade sa neberie do úvahy veľkosť hlavičky, ale len čisto veľkosť užitočných dát, preto to môže byť pre užívateľa mätúce.

2. **Čo predstavujú tzv. buckets?**

V ACE je spracovanie úlohy rozdelené do tzv. buckets, ktoré sú vlastne časové hromady, pomocou ktorých sa vypočíta priemerná alebo celková hodnota veličiny pre daný interval.

3. **Ktorý transportný protokol je v nami vytvorenej aplikácii použitý na prenos databázových správ?**

V nami vytvorenej databázovej aplikácii je použitý protokol TCP

13 ZÁVER

Program IT Guru predstavuje vhodný prostriedok pre vzdelávanie v oblasti sieťových technológií na akademickej úrovni. Dokáže simulovať širokú škálu sieťových kofigurácií a technológií, a tým podporuje lepšie pochopenie preberanej látky. Akademická verzia programu OPNET Modeler je vhodným prostriedkom pre výuku na laboratórnych cvičeniach, pretože je k dispozícii zadarmo a dokáže veľmi ilustratívne doplniť učivo preberané na prednáškach.

Táto diplomová práca popisuje fungovanie a vlastnosti protokolov transportnej vrstvy – TCP a UDP, fungovanie protokolu ICMP a v neposlednom rade sa venovala práci s nástrojom ACE. Mala za úlohu vytvoriť zadanie, ktoré bude môcť byť použiteľné na výuku predmetov súvisiacich so sieťovými technológiami.

Hlavným cieľom prvej úlohy bolo porovnanie protokolov TCP a UDP, a to z hľadiska časovej odozvy pri vykonávaných úlohách a aj z hľadiska spotreby sieťovým prostriedkov pri vykonávaní daných úloh. Práca je doplnená o prídavné otázky k zadaniu laboratórnej úlohy a zároveň aj o kapitolu, ktorá sa venuje zhodnoteniu výsledkov dosiahnutých v danej laboratórnej úlohe.

V ďalšej časti sa práca venovala popisu fungovania a vlastností protokolu ICMP, druhom riadiacich správ, ktoré daný protokol využíva a taktiež k nej bola vytvorená laboratórna úloha popisujúca jednu z hlavných funkcií protokolu ICMP – PING. Aj táto úloha bola doplnená o doplňujúce otázky o problematike a o zhodnotenie výsledkov dosiahnutých v danej laboratórnej úlohe.

V poslednej časti mojej diplomovej práce som sa venovala nástroju na analýzu a diagnostiku sietí – ACE, ktorý je súčasťou programu OPNET IT Guru, hoci len v obmedzenej verzii. K danej téme som vytvorila praktickú laboratórnu úlohu, ktorá užívateľa – študenta, oboznamuje s funkčnosťami nástroja ACE a jeho jednotlivými nástrojmi pre rýchlu diagnostiku sieťových problémov a v neposlednom rade na predikciu plánovaných zmien v sieti.

ZOZNAM POUŽITEJ LITERATÚRY

[1] JEŘÁBEK, J., *Pokročilé komunikační techniky (MPKT)*. Skripta VUT v Brně, 2009.

[2] OLIFER, Natalia; OLIFER, Victor. *Computer Networks: Principles, Technologies and Protocols for Network Design*. Chichester : John Wiley & Sons, 2006. 1095 s. ISBN 0470869828.

[3] BEASLEY, Jeffrey S. *Networking*. Second Edition. Boston: Pearson Education Inc., 2009. 693 s. ISBN 978-0-13-135838-6.

[4] KAPARTI, Ranjan *OPNET IT Guru: A Tool For Networking Education*. MSCIT Practicum Paper, Regis University

[5] RILEY, Kevan *IT Guru Lab 1: Introduction to IT Guru*. CSUMB Networking Research, 2010.

[6] ZOTTMANN, Harald. *Http://www.cellsoft.de/telecom/tcpiposi.html* [online]. 2011. 2011 [cit. 2011-11-27]. CellSoft. Dostupné z: http://www.cellsoft.de

[7] RILEY, Kevan *IT Guru Lab 2: Understanding TCP and UDP*. CSUMB Networking Research, 2010.

[8] NETWORK SORCERY, Inc., *ICMP, Internet Control Message Protocol* [online]. 2011. [cit. 2012-02-27]. Network Sorcery Inc. Dostupné z: http://www.networksorcery.com/enp/protocol/icmp.htm

[9] INTERNET ENGINEERING TASK FORCE. *Internet Engineering Task Force: RFC 792*. [online]. [cit. 2012-02-27]. Dostupné z: http://tools.ietf.org/html/rfc792

[10] INTERNET ENGINEERING TASK FORCE. *Internet Engineering Task Force*: RFC 1812. [online]. [cit. 2012-02-28]. Dostupné z: http://tools.ietf.org/html/rfc1812#section-5.2.7.1

[11] ABOELELA, Emad, Ph.D. *Network Simulation Experiments Manual.* Second Edition, University of Massachusetts, Morgan Kaufmann Publishers, 2008, ISBN 978-0-12-373974-2

[12] CISCO SYSTEMS, INC. Cisco Application Analysis Solution ACE User Guide for IT Guru, Cisco Systems, Inc, 2005, Dostupné z: http://www.cisco.com/en/US/docs/net_mgmt/application_analysis_solution/1.0/user/guide/ACE/aceugde.pdf

ABECEDNÝ PREHĽAD POUŽITÝCH SKRATIEK, VELIČÍN A SYMBOLOV

ACE – Application Characterization Environment

ACK – Acknowledgement

ARP – Address Resolution Protocol

ASCII – American Standard Code for Information Interchange

ATM – Asynchronous Transfer Mode

DF – Don't Fragment

DNS – Domain Name System

FIN – Finish

FTP – File Transfer Protocol

HTTP – Hypertext Transfer Protocol

ICMP – Internet Control Message Protocol

IP – Internet Protocol

MAC – Media Access Control

NFS – Network File System

PING – Packet Internet Groper

PSH – Push

RIP – Routing Information Protocol

RM OSI – Relational Model Open System Interconnection

RST – Reset

RTT – Round Trip Time

SMTP – Simple Mail Transfer Protocol

SNMP – Simple Network Management Protocol

SQL – Structured Query Language

SSL – Secure Sockets Layer

SYN – Synchronize

TCP – Transmission Control Protocol

TCP / IP – Transmission Control Protocol / Internet Protocol

TTL – Time To Live

UDP – User Datagram Protocol

URG – Urgent

VoIP – Voice over IP

WAN – Wide Area Network