



VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS
ELEKTRONIKOS FAKULTETAS
TELEKOMUNIKACIJŲ INŽINERIJOS KATEDRA

Gediminas Uža

**REALIOJO LAIKO SRAUTO, NAUDOJANT DAUGIATINKLIŠKUMO
TECHNOLOGIJAS, PERDAVIMO TYRIMAS**
**CASE STUDIES OF REAL TIME TRAFFIC TRANSMISSION OVER
MULTIHOMED NETWORKS**

Baigiamasis magistro darbas

Telekomunikacijų inžinerijos studijų programa, valstybinis kodas 621H64003

Telekomunikacijų technologijų specializacija

Elektronikos ir elektros inžinerijos studijų kryptis

Vilnius, 2016

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS
ELEKTRONIKOS FAKULTETAS
TELEKOMUNIKACIJŲ INŽINERIJOS KATEDRA

TVIRTINU

Katedros vedėjas

(Parašas)

dr. doc. Darius Guršnys

(Vardas, pavardė)

(Data)

Gediminas Uža

**REALIOJO LAIKO SRAUTO, NAUDOJANT DAUGIATINKLIŠKUMO
TECHNOLOGIJAS, PERDAVIMO TYRIMAS**
**CASE STUDIES OF REAL TIME TRAFFIC TRANSMISSION OVER
MULTIHOMED NETWORKS**

Telekomunikacijų inžinerijos studijų programa, valstybinis kodas 621H64003

Telekomunikacijų technologijų specializacija

Elektronikos ir elektros inžinerijos studijų kryptis

Vadovas dr. doc. Arūnas Šaltis

(Parašas)

(Data)

Lietuvių kalbos konsultantė dr. Vaida Buivydienė

(Parašas)

(Data)

Vilnius, 2016

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS
ELEKTRONIKOS FAKULTETAS
TELEKOMUNIKACIJŲ INŽINERIJOS KATEDRA

Elektronikos ir elektros inžinerijos studijų kryptis

Telekomunikacijų inžinerijos studijų programa, valstybinis kodas 612H64002

Telekomunikacijų technologijos specializacija

TVIRTINU
Katedros vedėjas

(Parašas)

Darius Guršnys
(Vardas, pavardė)

(Data)

**BAIGIAMOJO MAGISTRO DARBO
UŽDUOTIS**

.....Nr.

Vilnius

Studentui (ei) Gediminui Užai

Baigiamojo darbo tema: Realiojo laiko srauto, naudojant daugiatainkliškumo technologijas, perdavimo tyrimas (Case Studies of Real Time Traffic Transmission over Multihomed Networks)

patvirtinta 201....m. d. dekanų potvarkiu Nr.

Baigiamojo darbo (projekto) užbaigimo terminas 201... m. d.

BAIGIAMOJO DARBO UŽDUOTIS:

Tikslas: Ištirti MPTCP daugiatainkliškumo technologijos galimybes transportuoti realaus laiko srautą. Ištirti tokio perdavimo teikiamas kokybines charakteristikas.

Duomenys: Apžvelgti daugiatainkliškumo protokolus. Sudaryti realistišką tinklo modelį naudojant mobilios ryšio 4G, 3G, 2G tinklus bei VPN technologiją. Atlikti tyrimo analizę, apdoroti ir pateikti tyrimo metu gautus duomenis ir rezultatus. Pateikti darbo išvadą.

Aiškinamasis raštas:

1. Įvadas. Užduoties analizė
2. Daugiatainkliškumo protokolų analitinė apžvalga
3. MPTCP daugiatainkliškumo technologijos tyrimo analizė
4. Tyrimo metu gautų duomenų ir rezultatų apdorojimas, analizė ir apžvalga
5. Darbo išvados
6. Literatūros ir informacinių šaltinių sąrašas

Vadovas
(Parašas)

dr. doc. Arūnas Šaltis
(Pedag. vardas, vardas, pavardė)

Užduotį gavau

.....
(Parašas)

Gediminas Uža

.....
(Data)

Vilniaus Gedimino technikos universitetas

Elektronikos fakultetas

Telekomunikacijų inžinerijos katedra

ISBN ISSN

Egz. sk. 4

Data 2016-05-24

Telekomunikacijų inžinerijos studijų programos baigiamasis magistro darbas

Pavadinimas **Realiojo laiko srauto, naudojant daugiatainkliškumo technologijas, perdavimo tyrimas**

Autorius **Gediminas Uža**

Vadovas dr. doc. **Arūnas Šaltis**

Kalba

☒ X lietuvių

☐ anglų

Anotacija

Magistro baigiamajame darbe tiriamos MPTCP protokolo kokybinės charakteristikos perduodant realiojo laiko srautą mobiliojo korinio ryšio tinklais. Tyrimas atliktas naudojant *Openvpn* ir *vtun* programas VPN tuneliui tarp serverio ir kliento sukurti. Tinklo kokybiniams parametrams nustatyti naudojamos *Iperf* ir *D-ITG* duomenų srauto generavimo programos. Darbe analizuojami įvairūs daugiatainkliškumo sprendimai, jų skirtumai, privalumai ir trūkumai. Tyrimo metu nustatyta duomenų perdavimo sparta siunčiant TCP tipo srautą, taip pat, vėlinimas, paketų praradimas ir vėlinimo švytavimas siunčiant UDP tipo srautą 4G, 3G, 2G technologijomis. Nustatyta, kad naudojant MPTCP protokolą galima nesudėtingai sujungti skirtingas mobiliojo korinio ryšio technologijas į vieną srautą. Tokiu būdu padidinama bendra perdavimo sparta bei tinklo patikimumas.

Darbą sudaro 7 dalys: įvadas ir užduoties analizė, analogiškų koncepcijų apžvalga, MPTCP protokolo apžvalga, eksperimentų struktūrinės schemos kūrimas, MPTCP protokolo veikimo tyrimas, rezultatų apibendrinimas, literatūra.

Darbo apimtis – 62 p. teksto be priedų, 58 iliustr., 5 lent., 29 bibliografiniai šaltiniai.

Atskirai pridedami darbo priedai.

Prasminiai žodžiai: daugiatainkliškumas, MPTCP, UDP, TCP, realiojo laiko srautas, VPN.

Vilnius Gediminas Technical University
Faculty of Electronics
Department of Telecommunication Engineering

ISBN ISSN
Copies No. 4
Date 2016-05-24

Telecommunications Engineering study programme master thesis.

Title: **Case Studies of Real Time Traffic Transmission over Multihomed Networks**

Author **Gediminas Uža**

Academic supervisor associate professor **Arūnas Šaltis**

Thesis language

☒ Lithuanian
☐ English

Annotation

The aim of this thesis is to investigate MPTCP protocol for real-time data streaming over multi-homed mobile cellular networks. The data for research were collected from live network benchmarking using *Openvpn* and *vtun* VPN tunnels. Network KPI's were collected using *Iperf* and *D-ITG* data stream generators. The paper provides an overview of various multi-homing solutions, their pros and cons. On the basis of the results of this research, it can be concluded that MPTCP protocol suits very well for TCP data transmission, giving aggregated data throughputs and improved reliability over 4G, 3G, 2G networks Using VPN over MPTCP protocol improves real-time data reliability, and also aggregates total connection data throughput.

Structure: introduction and task analysis, review of similar solutions, MPTCP protocol analysis, creating experiment network scheme, MPTCP protocol usage experiment, evaluation of results, conclusions, references.

Thesis consist of: 62 p. text without appendixes, 58 pictures, 5 tables, 29 bibliographical entries.

Appendixes included.

Keywords: Multi-homing, MPTCP, UDP, TCP, real-time data, VPN

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

Gediminas Uža, studento pažymėjimo Nr. 1045897

(Studento vardas ir pavardė, studento pažymėjimo Nr.)

Elektronikos fakultetas

(Fakultetas)

Telekomunikacijų inžinerija, TETfm – 14

(Studijų programa, akademinė grupė)

BAIGIAMOJO MAGISTRO DARBO SAŽININGUMO DEKLARACIJA

201... m. _____ d.

(Data)

Patvirtinu, kad mano baigiamasis magistro darbas tema _____
Realiojo laiko srauto, naudojant daugiatinkliškumo technologijas, perdavimo tyrimas
(Case studies of real time traffic transmission over multihomed networks)

yra savarankiškai parašytas. Šiame baigiamajame magistro darbe pateikta medžiaga nėra plagijuota.
Tiesiogiai ar netiesiogiai panaudotos kitų šaltinių citatos pažymėtos literatūros nuorodose.

Baigiamąjį magistro darbą tikrino: _____

Mano darbo vadovas dr. doc. A. Šaltis

Kitų asmenų indėlio į baigiamąjį magistro darbą nėra. Jokių įstatymų nenumatytų piniginių
sumų už šį darbą niekam nesu mokėjęs.

(Parašas)

Gediminas Uža
(Vardas ir pavardė)

TURINYS

TURINYS	6
PAVEIKSLŲ SĄRAŠAS	8
LENTELIŲ SĄRAŠAS	10
SANTRUMPŲ SĄRAŠAS	11
1. ĮVADAS. UŽDUOTIES ANALIZĖ	13
2. ANALOGIŠKŲ KONCEPCIJŲ APŽVALGA.....	16
2.1. DAUGIATINKLIŠKUMAS.....	16
2.1.1. Priklausančios OSI lygmenims savybės	17
2.1.2. Nepriklausančios OSI lygmenims savybės.....	20
2.2. DAŽNIAUSIAI NAUDOJAMI SPRENDIMAI	23
2.3. APIBENDRINIMAS	24
3. MPTCP PROTOKOLAS	25
3.1. SUJUNGIMO SUDARYMAS	26
3.2. SUBKANALO PRIJUNGIMAS	27
3.3. DUOMENŲ PERDAVIMAS	29
3.3.1. Duomenų sekos žymėjimas	30
3.3.2. Duomenų patvirtinimai.....	31
3.3.3. Sujungimo nutraukimas.....	31
3.3.4. Gavėjo veikimas	32
3.3.5. Siuntėjo veikimas	32
3.3.6. Patikimumas ir persiuntimas	33
3.3.7. Grūsties kontrolė	33
3.3.8. Subkanalų strategija.....	33
3.4. ADRESŲ APSIKEITIMAS (KANALŲ VALDYMAS)	34
3.4.1. Adresų siūlymas	35
3.4.2. Adreso šalinimas	36
3.5. GREITAS NUTRAUKIMAS	36
3.6. SUBKANALŲ PERKROVIMAS.....	37
3.7. ATSITRAUKIMAS.....	38
3.8. APIBENDRINIMAS	39
4. EKSPERIMENTŲ STRUKTŪRINĖS SCHEMOS KŪRIMAS.....	40
4.1. REALAUS TINKLO STRUKTŪRINĖ SCHEMA.....	40
4.2. VIRTUALAUS TINKLO STRUKTŪRINĖ SCHEMA	41

4.3.	NAUDOJAMOS PROGRAMOS	41
4.3.1.	VPN technologija	41
4.3.2.	Kitos programos	43
4.4.	APIBENDRINIMAS	44
5.	MPTCP PROTOKOLO VEIKIMO TYRIMAS.....	45
5.1.	VIRTUALAUS TINKLO TYRIMAS	45
5.1.1.	TCP srauto perdavimas.....	46
5.1.2.	UDP srauto perdavimas	48
5.2.	REALAUS TINKLO TYRIMAS	54
5.2.1.	TCP srauto perdavimas.....	54
5.2.2.	UDP srauto perdavimas	57
5.3.	APIBENDRINIMAS	59
	REZULTATŲ APIBENDRINIMAS.....	60
	LITERATŪRA.....	61
	PRIEDAI	63
	PRIEDAS A. PAŽYMA APIE DALYVAVIMĄ XIX-TOJOJE JAUNŲJŲ MOKSLININKŲ KONFERENCIJOJE....	63

PAVEIKSLŲ SĄRAŠAS

1 pav. Mobiliųjų įrenginių augimas [19].....	13
2 pav. Duomenų perdavimo mobiliaisiais tinklais augimas [19]	13
3 pav. Duomenų kiekio, perduodamo mobiliais tinklais, augimas pagal duomenų tipą [22]	14
4 pav. Daugiatinkliškumas	16
5 pav. OSI modelis [1]	17
6 pav. Programų lygmens protokolai daugiatinkliškumui įgyvendinti [2].....	17
7 pav. Transporto lygmens protokolai daugiatinkliškumui įgyvendinti [2]	18
8 pav. Tinklo lygmens protokolai daugiatinkliškumui įgyvendinti [2]	19
9 pav. Kanalinio lygmens protokolai daugiatinkliškumui įgyvendinti [2].....	20
10 pav. Tinklo palaikymo ir komunikavimo modeliai [2]	22
11 pav. SCTP protokolo sujungimo schema [29].....	24
12 pav. TCP ir MPTCP dėklų skirtumai [8].....	25
13 pav. MPTCP subkanalų struktūra [26]	25
14 pav. MPTCP antraštės struktūra [8]	26
15 pav. MTCP sesijos užmezgimas [4]	26
16 pav. MP_CAPABLE antraštės struktūra [8].....	27
17 pav. Papildomos MTCP sesijos užmezgimas [4]	28
18 pav. MP_JOIN antraštės struktūra (kai siunčiamas SYN) [8].....	28
19 pav. MP_JOIN antraštės struktūra (kai siunčiamas SYN/ACK) [8]	29
20 pav. MP_JOIN antraštės struktūra (kai siunčiamas ACK) [8]	29
21 pav. DSS antraštės struktūra [8]	30
22 pav. MP_PRIO antraštės struktūra [8]	34
23 pav. Daugiatinkliškumo pavyzdys	35
24 pav. ADD_ADDR2 antraštės struktūra [8]	35
25 pav. REMOVE_ADDR antraštės struktūra [8]	36
26 pav. MP_FASTCLOSE antraštės struktūra [8]	36
27 pav. MP_TCP_RST antraštės struktūra [8].....	37
28 pav. MP_FAIL antraštės struktūra [8]	38
29 pav. Struktūrinė realaus tinklo schema.....	40
30 pav. „Huawei“ E5186 maršruto parinktuvas [9]	40
31 pav. Struktūrinė virtualaus tinklo schema	41
32 pav. VPN technologija	42
33 pav. <i>Openvpn</i> raktų bei sertifikatų veikimo schema [17].....	42

34 pav. Virtualaus tinklo schema ir parametrai	45
35 pav. TCP srauto pasiskirstymas naudojant sąsajų sujungimo metodą.....	46
36 pav. TCP srauto pasiskirstymas naudojant MPTCP protokolą.....	47
37 pav. UDP srauto pasiskirstymas naudojant sąsajų sujungimo metodą.....	48
38 pav. Vidutinis UDP srauto vėlinimas naudojant sąsajų sujungimo metodą	48
39 pav. Vidutinis UDP srauto vėlinimo švytavimas naudojant sąsajų sujungimo metodą.....	49
40 pav. UDP srauto pasiskirstymas naudojant MPTCP protokolą su <i>Openvpn</i>	50
41 pav. UDP srauto perdavimo sparta ir paketų praradimai naudojant MPTCP protokolą su <i>Openvpn</i>	50
42 pav. UDP srauto vėlinimas į vieną pusę naudojant MPTCP protokolą su <i>Openvpn</i>	50
43 pav. UDP srauto pasiskirstymas naudojant MPTCP protokolą su <i>vtun</i>	51
44 pav. UDP srauto perdavimo sparta ir paketų praradimai naudojant MPTCP protokolą su <i>vtun</i> , eth0 vėlinimas 10 ms	51
45 pav. UDP srauto perdavimo sparta ir paketų praradimai naudojant MPTCP protokolą su <i>vtun</i> , eth0 vėlinimas 50 ms	52
46 pav. UDP srauto perdavimas naudojant MPTCP protokolą su <i>vtun</i>	52
47 pav. UDP srauto paketų praradimas naudojant MPTCP protokolą su <i>vtun</i>	53
48 pav. UDP srauto vėlinimas į vieną pusę naudojant MPTCP protokolą su <i>vtun</i>	53
49 pav. 4G technologijų sujungimas naudojant MPTCP protokolą	54
50 pav. 4G technologijų sujungimas naudojant MPTCP protokolą	55
51 pav. 4G/3G technologijų sujungimas naudojant MPTCP protokolą	55
52 pav. 4G/3G technologijų sujungimas naudojant MPTCP protokolą	55
53 pav. 4G/2G technologijų sujungimas naudojant MPTCP protokolą	56
54 pav. 4G/2G technologijų sujungimas naudojant MPTCP protokolą	56
55 pav. 3G technologijų sujungimas naudojant MPTCP protokolą	56
56 pav. 3G technologijų sujungimas naudojant MPTCP protokolą	57
57 pav. UDP srauto perdavimas 4G/4G technologijomis realiame tinkle.....	58
58 pav. MPTCP paprasto transportavimo režimo antraštės struktūra	58

LENTELIŲ SĄRAŠAS

1 lentelė. Tyrimo scenarijai ir naudojamos technologijos.....	45
2 lentelė. TCP srauto pasiskirstymas naudojant sąsajų sujungimo metodą.	46
3 lentelė. TCP srauto pasiskirstymas naudojant MPTCP protokolą.	47
4 lentelė. UDP srauto rezultatai 4G/4G tinkluose naudojant MPTCP protokolą su <i>Openvpn</i>	57
5 lentelė. UDP srauto rezultatai 4G/4G tinkluose naudojant MPTCP protokolą su <i>vtun</i>	57

SANTRUMPŲ SĄRAŠAS

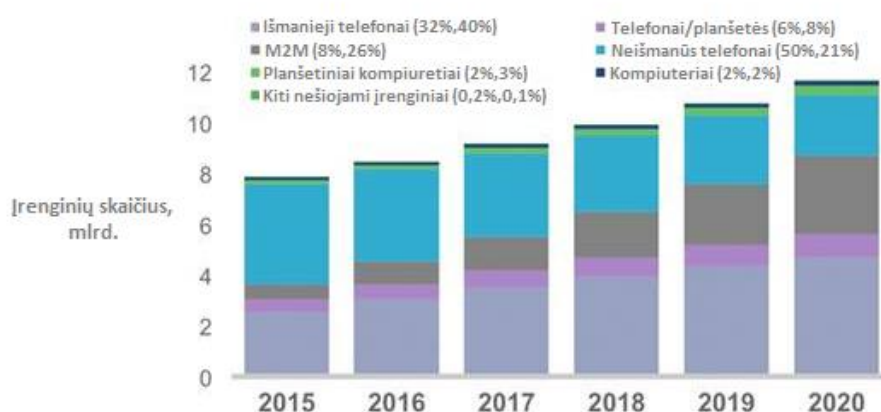
2G	–	Antrosios kartos tinklas (angl. <i>Second Generation</i>)
3G	–	Trečiosios kartos tinklas (angl. <i>Third Generation</i>)
4G	–	Ketvirtosios kartos tinklas (angl. <i>Third Generation</i>)
DSL	–	Skaitmeninė vartotojo linija (angl. <i>Digital Subscriber Line</i>)
DSS	–	Duomenų sekos signalas (angl. <i>Data Sequence Signal</i>)
GPRS	–	Bendras paketinis radijo ryšys (angl. <i>General Packet Radio Service</i>)
IANA	–	Interneto paskirtųjų numerių asociacija (angl. <i>Internet Assigned Numbers Authority</i>)
ID	–	Identifikatorius (angl. <i>Identifier</i>),
IETF	–	Interneto inžinerijos grupė (angl. <i>Internet Engineering Task Force</i>)
IP	–	Interneto protokolas (angl. <i>Internet Protocol</i>)
IPv4	–	Interneto protokolo 4 versija (angl. <i>Internet Protocol Version 4</i>)
IPv6	–	Interneto protokolo 6 versija (angl. <i>Internet Protocol Version 6</i>)
ISP	–	Interneto paslaugų tiekėjas (angl. <i>Internet Service Provider</i>)
LTE	–	Mobilios telefonijos tinklo standartas (angl. <i>Long Term Evolution</i>)
M2M	–	Įrenginys su įrenginiu (angl. <i>Machine To Machine</i>)
MAC	–	Terpės prieigos valdymo protokolas (angl. <i>MAC – Media Access Control</i>)
MMTP	–	Multimedijos sujungimo perdavimo protokolas (angl. <i>Multimedia Multiplexing Transport Protocol</i>)
MPTCP	–	Daugiakelis perdavimo kontrolės protokolas (angl. <i>Multipath Transmission Control Protocol</i>)
MUP	–	Radijo apjungimo protokolas (angl. <i>Multi-radio Unification Protocol</i>)
NAT	–	Tinklo adreso keitimas (angl. <i>Network Address Translation</i>)
OSI	–	Atvirų sistemų tarpusavio sujungimo modelis (angl. <i>Open Systems Interconnection Reference Model</i>)
PKI	–	Viešojo rakto struktūra (angl. <i>Public Key Infrastructure</i>)
QoS	–	Paslaugų kokybė (angl. <i>Quality of Service</i>)
RCP	–	Tempo kontrolės protokolas (angl. <i>Rate Control Protocol</i>)
RTO	–	Pakartotinio perdavimo laikas (angl. <i>RTO – Retransmission Timeout Timer</i>)
RTT	–	Pirmyn ir atgal laikas (angl. <i>RTT – Round Trip Time</i>)
SCTP	–	Srautų kontrolės perdavimo protokolas (angl. <i>Stream Control Transmission Protocol</i>)
TCP	–	Perdavimo kontrolės protokolas (angl. <i>Transmission Control Protocol</i>)

TUN	–	Tinklo tunelis (angl. <i>Network Tunnel</i>)
UDP	–	Vartotojo datagramų protokolas (angl. <i>User Datagram Protocol</i>)
UMTS	–	Mobilios telefonijos tinklo standartas (angl. <i>Universal Mobile Telecommunications System</i>)
VPN	–	Virtualusis privatus tinklas (angl. <i>Virtual Private Network</i>)
WiFi	–	Bevielio ryšio standartas (angl. <i>Wireless Fidelity</i>)
WiMAX	–	Bevielio ryšio technologija (angl. <i>Worldwide Interoperability for Microwave Access</i>)

1. ĮVADAS. UŽDUOTIES ANALIZĖ

Temos aktualumas. Šiandien turėti prieigą prie interneto didžiajai daliai žmonių nėra naujiena. Sparčiai besivystančios technologijos suteikia galimybę naudotis internetu namie, darbe, pakeliui į darbą savo išmaniajame telefone, planšetėje ar net skrendant lėktuvu. Interneto greitis taip pat sparčiai auga dėl naujų technologijų, pavyzdžiui, LTE (angl. *Long Term Evolution*) ar WiMAX (angl. *Worldwide Interoperability for Microwave Access*).

Žmonės vis daugiau laiko praleidžia internete, naudojami įvairiais išmaniaisiais įrenginiais. Skaičiuojama, kad 2014 metais visame pasaulyje buvo 4,3 milijardo žmonių, naudojančių mobiliuosius tinklus, o 2019 metais jų kiekis padidės iki 5,2 milijardo [19]. Mobiliųjų įrenginių skaičius augs nuo 7,9 milijardo 2015 metais iki 11,6 milijardo 2020 metais (1 pav.) [19].



1 pav. Mobiliųjų įrenginių augimas [19]

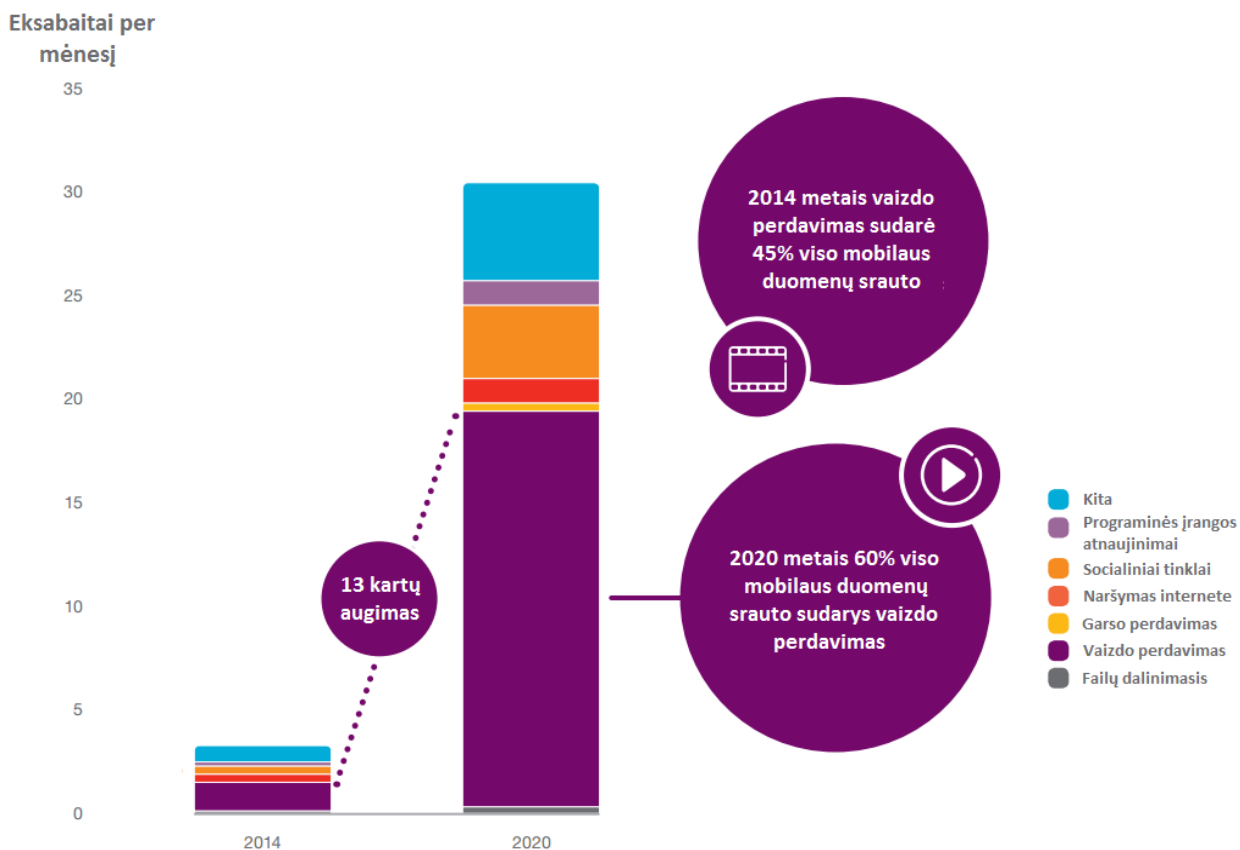
Didžiausias įrenginių augimas prognozuojamas M2M (angl. *Machine To Machine*). Lyginant su 2015 metais, prognozuojama, kad 2020 metais jų skaičius padidės nuo 8 % iki 26 %. Išmaniųjų telefonų skaičius augs nuo 32 % 2015 metais iki 40 % 2020 metais, o ne išmaniųjų telefonų skaičius sumažės nuo 50 % iki 21 % [19]. Visi šie skaičiai rodo, kad tikėtinas perduodamų duomenų kiekis mobiliaisiais tinklais ir toliau sparčiai augs (2 pav.).



2 pav. Duomenų perdavimo mobiliaisiais tinklais augimas [19]

Prognozuojama, kad perduodamų duomenų kiekis mobiliaisiais tinklais 2020 metais pasieks 30,6 eksabaitų per mėnesį, o tai yra beveik 10 kartų daugiau nei 2015 metais [19].

Pagal „Ericsson“ kompanijos skaičiavimus ir prognozę, didžiąją dalį duomenų, perduodamų mobiliaisiais tinklais, sudaro vaizdo perdavimas. 2014 metais vaizdo perdavimas sudarė 45 % viso mobilios duomenų srauto [22]. Prognozuojama, kad šis skaičius 2020 metais išaugs iki 60 %, o perduodamas vaizdo duomenų kiekis padidės 13 kartų (3 pav.) [22].



3 pav. Duomenų kiekio, perduodamo mobiliais tinklais, augimas pagal duomenų tipą [22]

Nors didžiąją dalį viso vaizdo duomenų srauto sudaro tokių paslaugų kaip „Youtube“ ar „Netflix“ duomenys, kurie iš esmės nėra realaus laiko srautas, savo dalį užima ir realaus laiko vaizdo transliacijos, tokios kaip „UStream“ [22]. 2016 metais socialinis tinklas „Facebook“ vartotojams pasiūlė galimybę skelbti tiesiogines vaizdo transliacijas savo socialinio tinklo paskyroje, tad noras perduoti realaus laiko duomenis ir toliau auga, atsiranda naujos programos ir paslaugos [23].

Negana to, dabar galima turėti net kelias prieigas prie interneto viename įrenginyje, taip padidinant prieigos patikimumą, saugumą ir spartą. Tiesa, paprastiems vartotojams tokios galimybės dažniausiai nėra reikalingos, nes ir su viena interneto prieiga galima laisvai naršyti, siųsti ar gauti informaciją. Tačiau didesnėms įmonėms, duomenų centrams ar itin didelį patikimumo lygį užtikrinantiems tinklams daugiatinkliškumas ypač reikalinga technologija [24]. Kadangi daugiatinkliškumui įgyvendinti netinka dažniausiai internete naudojami TCP (angl. *Transmission*

Control Protocol) ir UDP (angl. *User Datagram Protocol*) protokolai, reikėjo sugalvoti ką nors naujo, leisiančio turėti kelias interneto prieigas vienu metu, bet tuo pačiu sugebėtų veikti jau egzistuojančiame tinkle.

Daugiatinkliškumas taip pat suteikia galimybę turėti didesnę prieigą prie interneto patikimumą. Gaunant paslaugas iš dviejų skirtingų interneto paslaugų tiekėjų rizika netekti ryšio sumažėja iki minimumo. Tokias atvejais, kai realaus laiko srautą reikia perduoti iš atokiau nuo miestų esančių vietų, kaimo gyvenviečių, kur interneto ryšys yra daug prastesnis nei mieste ir fiksuoto ryšio operatoriai neteikia paslaugų, mobilus internetas tampa vienintele išeitimi. Tiesa, mobilus ryšys kaimo vietovėse gali būti daug silpnesnis nei mieste, todėl daugiatinkliškumas, apjungiant kelis mobiliojo ryšio operatorius, galėtų pagerinti prieigą prie interneto patikimumą ir perdavimo spartą.

Šiam tikslui įgyvendinti buvo sugalvota nemažai įvairių sprendimų. Visi jie turi savo privalumų ir trūkumų, vieni yra naudojami, kiti nepritaupo dėl didelės konkurencijos. Šiame darbe apžvelgsime pagrindinius daugiatinkliškumo sprendimo variantus, išanalizuosime MPTCP (angl. *Multipath Transmission Control Protocol*) protokolo veikimą, ištirsime jo galimybes perduoti duomenis mobiliais tinklais bei apžvelgsime realiojo laiko srauto perdavimo kokybines charakteristikas [5].

Problema. Daugiatinkliškumo realizavimas naudojant įvairias prieigas prie interneto technologijas, realiojo laiko srautui perduoti.

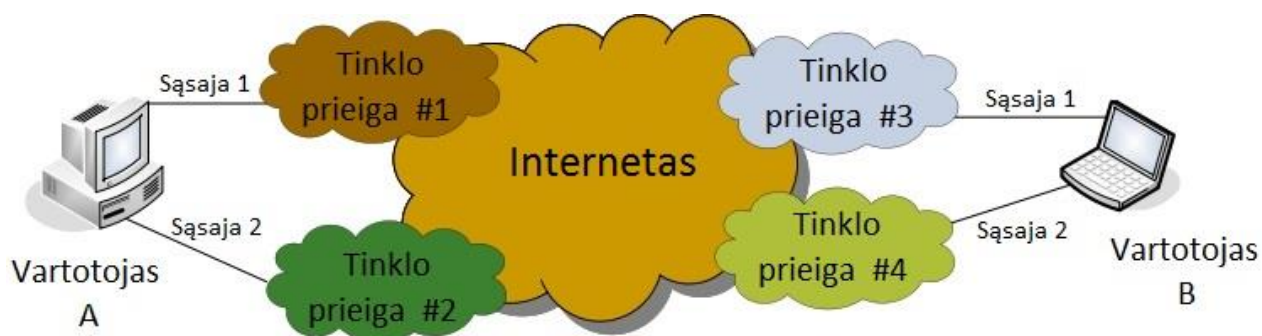
Tikslas. Ištirti MPTCP daugiatinkliškumo technologijos galimybes perduoti realiojo laiko srautą ir tokio perdavimo teikiamas kokybines charakteristikas.

Uždaviniai. Apžvelgti daugiatinkliškumo protokolus. Sudaryti realistišką tinklo modelį naudojant mobilaus ryšio 4G, 3G, 2G tinklus bei VPN technologiją. Atlikti tyrimo analizę, apdoroti ir pateikti tyrimo metu gautus duomenis ir rezultatus. Pateikti darbo išvadas.

2. ANALOGIŠKŲ KONCEPCIJŲ APŽVALGA

2.1. Daugiatinkliškumas

Daugiatinkliškumas (angl. *Multi-homing*) – tai įrenginio galimybė vienu metu būti prisijungus prie interneto dvejomis skirtingomis sąsajomis ir jomis siųsti vienos sesijos duomenis (4 pav.). Ši galimybė nėra naujiena, poreikis naudoti kelias prieigas prie interneto technologijas atsirado seniai, su pirmosiomis didelėmis įmonėmis ir duomenų centrais, kuriems reikėjo užtikrinti nenutrūkstamą interneto prieigą. Šiandien šis būdas po truputį skinasi kelią ir į paprastų vartotojų gyvenimus, ypač naudojant bevielės interneto prieigas. Bevielės ryšys yra nuolat kintanti terpė, kurioje dažnai gali atsirasti nenumatytų trukdžių. Todėl įrenginiams, naudojantiems tokias terpes, daugiatinkliškumas yra didelis privalumas. Pirmoji šios idėjos link žengė kompanija „Apple“, savo operacinėje sistemoje „iOS-7“ įdiegusi galimybę naudotis keliomis duomenų perdavimo technologijomis vienu metu [28]. Tiesa, ši funkcija gali būti naudojama tik sujungimui su „Apple“ kompanijos serveriais, likusiai interneto daliai pasiekti naudojama viena prieiga [6].



Pagrindinės problemos sprendžiamos daugiatinkliškumui įgyvendinti yra dvi: a) skirtingų prieigų sujungimo būdas; b) sprendimo integracija į egzistuojantį interneto tinklą. Ieškant pirmosios problemos sprendimų reikia atsižvelgti į skirtingų prieigų charakteristikų matavimą, duomenų perdavimo skirtingoms prieigoms algoritmą. Antrojoje problemoje reikia atsižvelgti į dabar egzistuojančią interneto struktūrą, kadangi dauguma internete veikiančių įrenginių yra pritaikyti naudoti TCP/IP protokolus, o šie gali veikti tik vieno IP adreso gavėjo įrenginiui principu. Egzistuoja begalė įvairių sprendimų šiam tikslui pasiekti, bet ne visi jie iki galo tinkami arba turi tam tikrų apribojimų. Galima išskirti dvi pagrindines sprendimų savybes:

- Priklausančios OSI (angl. *Open Systems Interconnection Reference Model*) lygmenims.
- Nepriklausančios OSI lygmenims.

Toliau apžvelgsime abi daugiatinkliškumo savybių grupes.

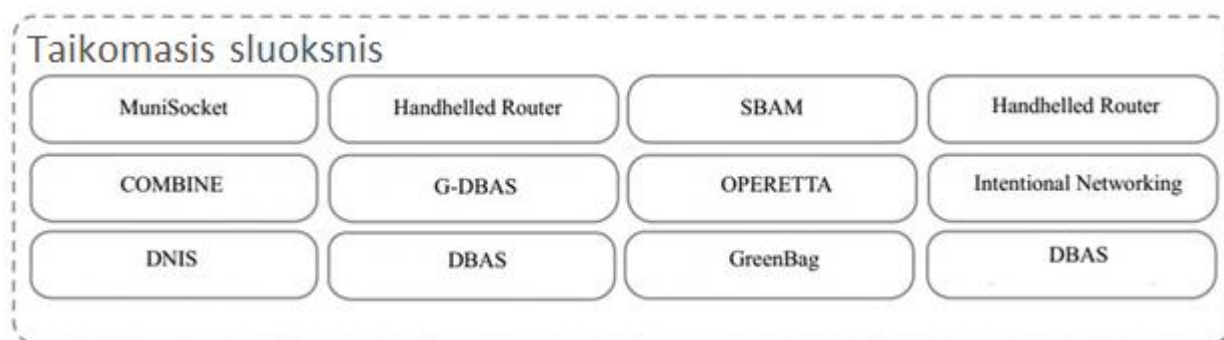
2.1.1. Priklausančios OSI lygmenims savybės

OSI modelis – tai abstraktus ryšio protokolų, naudojamų ryšio ir kompiuteriniuose tinkluose aprašymas. OSI modelis aprašo 7 sluoksnius – funkcijų, naudojamų protokoluose, grupes. Kiekvienas sluoksnis naudojami žemesnio sluoksnio paslaugomis ir teikia paslaugas aukščiau esančiam sluoksniui. Bendravimas vyksta tik tarp šalia esančių sluoksnių, taip supaprastinant sudėtingų sistemų kūrybą.



5 pav. OSI modelis [1]

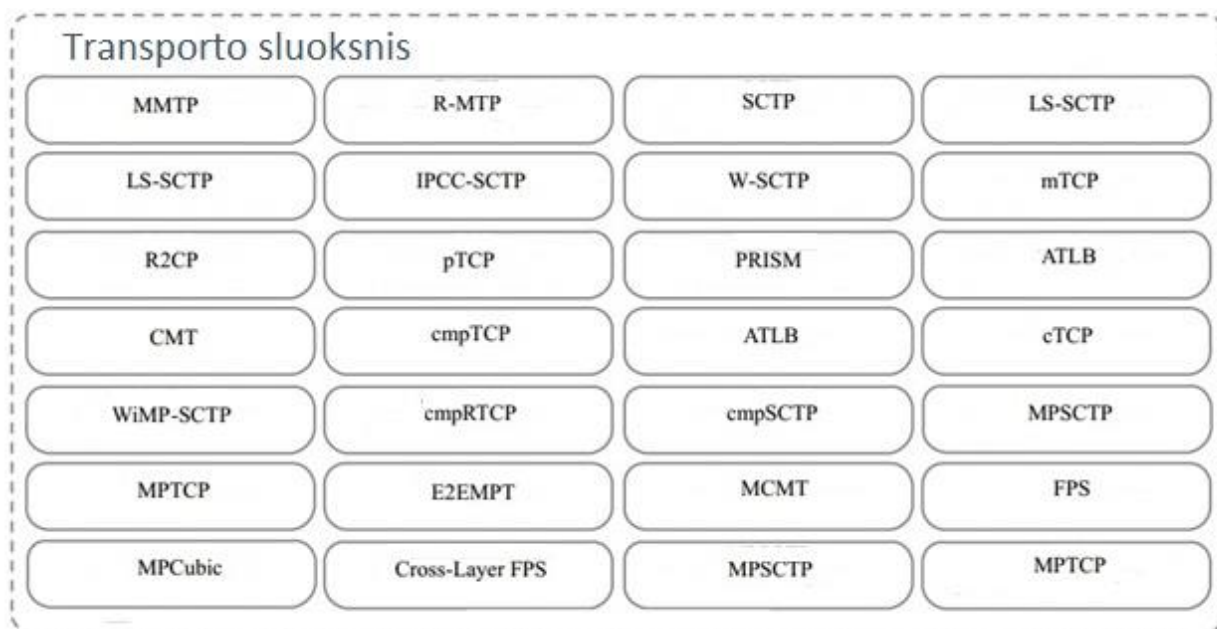
Sprendimai daugiatainkliškumui įgyvendinti kuriami atsižvelgiant į OSI modelio sluoksnių struktūrą (5 pav.). Pagal tai, kuriam lygiui priklauso sprendimas, juos galima suskirstyti į atitinkamas grupes.



6 pav. Programų lygmens protokolai daugiatainkliškumui įgyvendinti [2]

Programų lygmens sprendimams priskiriamos programos, kurios turimas prieigas naudoja tik savo funkcijoms atlikti, bei programos, kurios tarsi veikia tarp programų ir transporto lygmenų (6 pav.). Pastarosios dar skirstomos į dvi kategorijas:

- a) Paslėptosios tarpinės (angl. *Hidden Middleware*) – programos veikia tarsi nematomos aukščiau jų veikiančioms programoms ir suteikia galimybę naudoti kelias prieigas vienu metu. Pavyzdžiui DNIS, DBAS, OPERETTA programos.
- b) Matomosios-tarpinės (angl. *Non-transparent Middleware*) – šios programos yra matomos aukščiau jų veikiančioms programoms ir reikalauja tam tikrų modifikacijų jų taisyklių rinkiniuose. Pavyzdžiui MuniSocket programa.



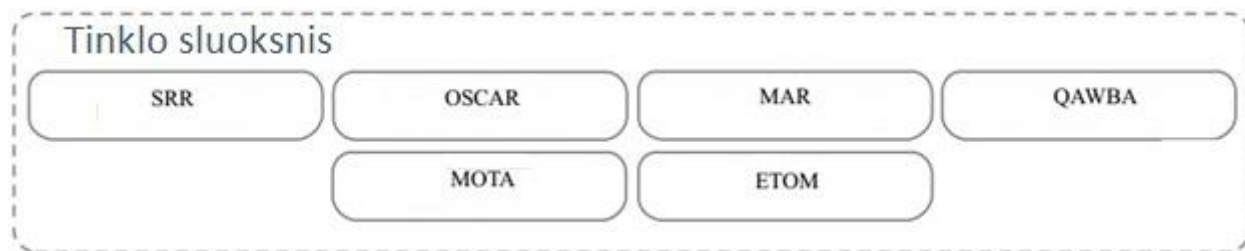
7 pav. Transporto lygmens protokolai daigiatinkliškumui įgyvendinti [2]

Transporto lygmens sprendimai užima didžiausią dalį iš visų sprendimų ir skirstomi į 3 kategorijas (7 pav.):

- a) Patobulinti plačiai naudojami protokolai – protokolas, kurį bandoma dažniausiai modifikuoti yra TCP. Kadangi TCP protokolas yra dažniausiai naudojamas, perduodant interneto duomenis, egzistuoja daugybė šio protokolo modifikacijų bei papildymų. Visos modifikacijos sprendžia pagrindinius TCP protokolo trūkumus – paketų priklausančių skirtingiems subkanalams pertvarkymas, jų paskirstymas skirtingoms prieigoms, mazgų atpažinimas, grūsties kontrolės mechanizmas. Kadangi TCP leidžia turėti tik vieną siuntėjo ir gavėjo IP adresą, visos modifikacijos naudoja kelių IP adresų vartotojui režimą. Šiam tipui priklauso MPTCP, MPCubic, E2EMPT protokolai.
- b) Egzistuojančių protokolų pritaikymas daigiatinkliškumui – SCTP (angl. *Stream Control Transmission Protocol*) protokolas vienas labiausiai nagrinėjamų protokolų, kadangi jis atitinka daigiatinkliškumo principus. SCTP protokolas gali turėti prieigas prie interneto, tačiau negali jų naudoti vienu metu. Todėl yra kuriamos įvairios patobulintos šio protokolo versijos, leidžiančios išnaudoti visas daigiatinkliškumo galimybes.

- c) Nauji protokolai – nors sukurti protokolai laikomi naujais, jie dažniausiai paremti tokiu pat veikimo principu, kaip TCP protokolas, dėl didelio jo paplitimo. Tokių protokolų pavyzdžiai: R-MTP, RCP (angl. *Rate Control Protocol*), MMTP (angl. *Multimedia Multiplexing Transport Protocol*).

Tinklo lygmens sprendimai – skirti išlaikyti jau naudojamus transporto lygmens protokolus pritaikius jų veikimą skirtingoms sąsajoms ir atliekant pakeitimus tinklo sluoksnyje (8 pav.).



8 pav. Tinklo lygmens protokolai daugiatainkliškumui įgyvendinti [2]

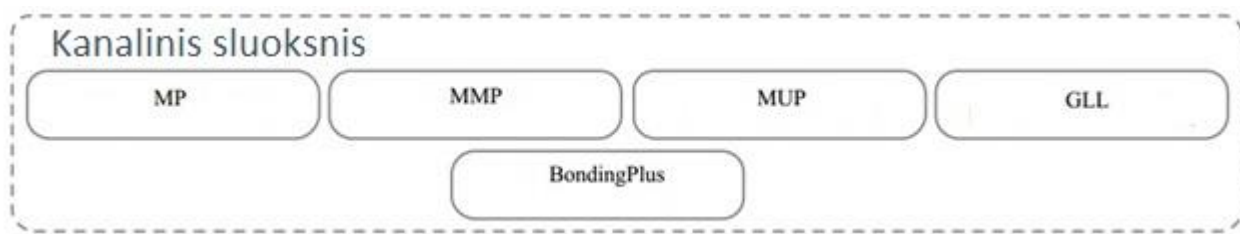
Dažniausiai tokiuose sprendimuose transporto protokolu dėl savo paplitimo ir populiarumo naudojamas TCP protokolas. Šiems sprendimams reikia įgyvendinti tris pagrindinius iššūkius:

- Apeiti TCP sesijos algoritmą – kadangi TCP nepritaikytas veikti su keliais IP adresais turinčiomis sąsajomis vienu metu, šie sprendimai turi tai „nuslėpti“. Informacija tvarkoma taip, kad TCP protokolas nepastebėtų, jog iš tikro naudojama ne viena, o kelios sąsajos su joms skirtais IP adresais. Pavyzdžiui galima naudoti „IP-in-IP“ inkapsuliaciją, kai iš TCP sluoksnio atėję paketai su IP adresu yra inkapsuliuojami į dar vieną IP paketą su jam skirtos sąsajos IP adresu. Pasiekus gavėją tinklo lygmenyje inkapsuliacija yra panaikinama ir gavėjo TCP protokolas gauna originalų paketą. Taip pat gali būti naudojami įgaliojtieji serveriai (angl. *Proxy Servers*).
- Perpildymo kontrolės klaidinimas – naudojant skirtingas technologijas internetui pasiekti neišvengiama skirtingo paketų vėlinimo. TCP protokolui išsibarsčius, skirtingu laiku atkeliavę paketai gali sudaryti sąlygas nutraukti sesiją. Todėl turi būti naudojama papildoma paketų numeracija tinklo lygmenyje. Iš TCP atkeliavę paketai yra papildomai numeruojami ir pasiekus adresatą jie yra kaupiami buferyje, kol susirinks trūkstanti, pavėlininti paketai. Surinkus reikalingus paketus jie perduodami į TCP sluoksnį tolesniam apdorojimui. Reikia atsižvelgti, kad apie dingusius paketus bus sužinoma tik kai siuntėjas praneš apie nutrūkusį sujungimą dėl nežinomų priežasčių (angl. *Time-out*). Šiai problemai spręsti tinklo lygmenyje yra integruojamos papildomos funkcijos ir laikmačiai, sekantys naujai sunumeruotus paketus.

Pirmyn ir atgal laiko (angl. RTT – *Round Trip Time*) matavimas – duomenims keliaujant skirtingais keliais skiriasi ir jų RTT trukmė. Šiai problemai spręsti gali būti pasitelkiami matematinio

skaičiavimo modeliai, vengiantys klaidų atliekant RTT skaičiavimą. Tokiu būdu parenkama ir tikslesnė pakartotinio perdavimo (angl. RTO – *Retransmission Timeout Timer*) vertė.

Kanalinio lygmens sprendimai – kitaip dar vadinami terpės prieigos valdymo lygmens (angl. MAC – *Media Access Control*) sprendimai. Juose dažnai naudojamas kanalų sujungimas tarp dviejų tiesiogiai sujungtų įrenginių (9 pav.).



9 pav. Kanalinio lygmens protokolai daugiatinkliškumui įgyvendinti [2]

Tokio sprendimo pavyzdys galėtų būti MUP protokolas. Šiuo protokolu galima sujungti skirtingų radijo dažnių kanaluose veikiančius įrenginius. Taip pat egzistuoja ir daugiau įvairių šio lygmens protokolų.

2.1.2. Nepriklausančios OSI lygmenims savybės

Nors įvairius daugiatinkliškumo sprendimus galima priskirti tam tikriems OSI lygiams, dalis sprendimų savybių nepriklauso nuo lygio, kuriame jis veikia. Pagrindines savybes suskirstykime į 4 grupes: sąsajos charakteristikų matavimas, programos charakteristikų matavimas, planavimas, tinklo suderinamumo ir komunikavimo modelis.

Sąsajos charakteristikų matavimas – viena svarbiausių savybių sujungiant skirtingas prieigas. Ši funkcija atsakinga už skirtingų tinklų pralaidumo, apkrovos, talpumo ir klaidų kiekio matavimus. Išskiriamos penkios pagrindinės charakteristikos:

- a) Pralaidumo matavimas – labai svarbi charakteristika, į kurią atsižvelgiama agreguojant skirtingas sistemas. Nuo jo priklauso, kokie algoritmai gali būti naudojami duomenų paskirstymui sąsajose. Pralaidumui apskaičiuoti naudojama nemažai skirtingų būdų. Vienas populiariausių sprendimų yra paketų poravimas. Paketų poravimo metodu siuntėjas išsiunčia du šalia esančius paketus vienas po kito su tam tikro laiko pauze. Gavėjas, gavęs abu paketus, išsiunčia du ACK patvirtinimus siuntėjui su tokio paties ilgio pauze tarp paketų. Iš intervalo tarp gautų paketų siuntėjas nustato, kokio pralaidumo yra kanalas. Norint nustatyti tikslesnį pralaidumą reikia naudoti ne dviejų paketų porą, o kelių paketų grandis ir matuoti laiko intervalus tarp kiekvienos šalia esančių paketų poros.
- b) Vėlinimas – charakteristika, nustatanti paketų pametimą kanale bei organizuojanti paketų planavimą. Dažni sprendimai matuoja RTT trukmę skirtingose sąsajose, o tada juos lygina

ir gauna bendrą vėlinimą. Taip pat galima matuoti laiko intervalus tarp ACK žinučių, panašiai kaip ir matuojant pralaidumą.

- c) Energijos suvartojimas – siekiant maksimaliai išnaudoti pralaidumą su minimaliomis energijos sąnaudomis yra labai svarbi kiekvieno sprendimo savybė. Dažniausiai šis parametras sužinomas pagal tinklo plokštės parametrus, kurie laikomi tam tikroje duomenų bazėje. Taip pat yra galimybė apskaičiuoti šį rodiklį iš perdavimo trukmės ir naudojamo pralaidumo bei pasirinktai technologijai priklausančių faktorių.

Programos charakteristikų matavimas – tam tikri sprendimai naudoja gautą informaciją pagerinti planavimo algoritmams. Į šią charakteristiką patenka:

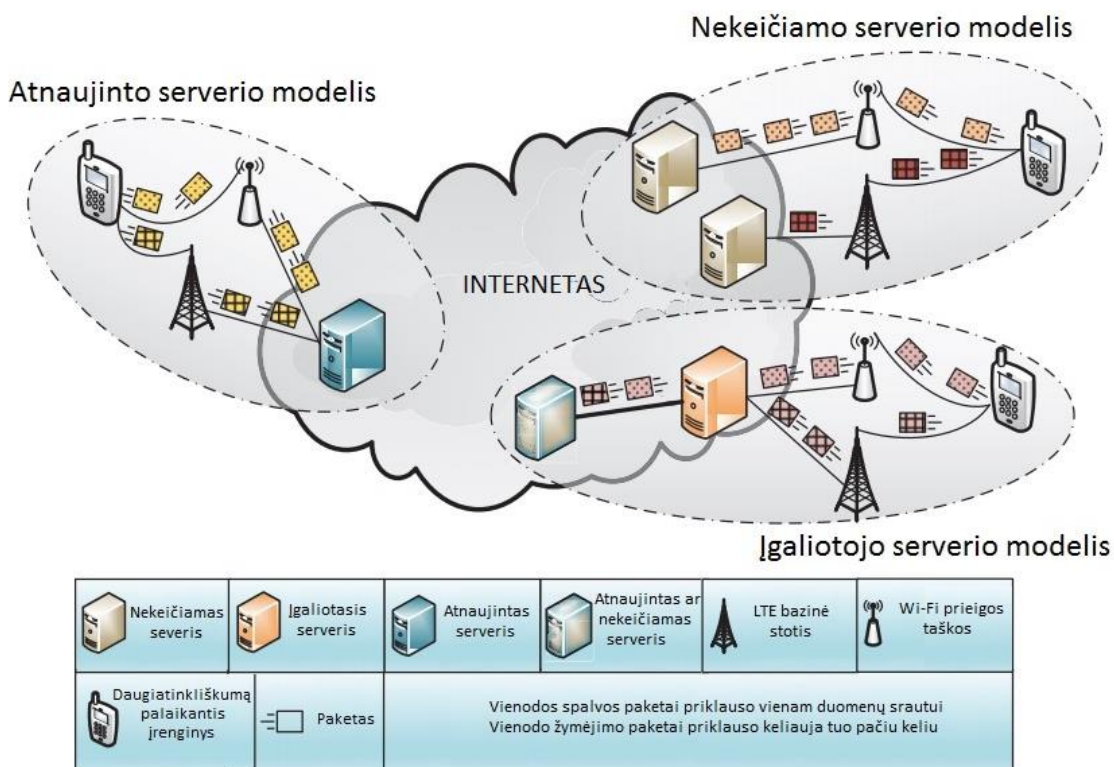
- a) Kokybinis indėlis – sistema iš programos gauna tam tikrus patarimus, kaip pagerinti paketų planavimo algoritmą.
- b) Kiekybinis indėlis – sistema paprašo programos pranešti, koks bus reikalingas pralaidumas.
- c) Matavimas – kiti sprendimai vietoj to, kad prašytų programos parametrų, stengiasi juos nusistatyti patys pagal jau programos naudojimo istoriją ar tipą. Taip padidėja galimybė sprendimus suderinti su įvairiomis programomis.

Planavimas – bene svarbiausia kiekvieno sprendimo užduotis yra duomenų planavimas kelių sąsajų sistemose. Aptarsime du pagrindinius planavimo aspektus:

- a) Planavimo detalumas – tai duomenų kiekis, kurį galima priskirti tinklo sąsajai. Yra dvi pagrindinės planavimo detalumo kategorijos. Pirmoji – planavimas paketų lygmenyje, antroji – planavimas sujungimo lygmenyje. Pirmasis atvejis naudojamas dažniausiai, kadangi paketai, priklausantys tai pačiai sesijai, gali būti paskirstyti po skirtingas sąsajas. Šiuo atveju abi sujungimo pusės privalo leisti naudoti tokį metodą, arba įgaliotuosius serverius. Šis metodas dažniausiai padidina duomenų perdavimo greitį. Antruoju atveju skirtingų sesijų paketai gali būti siunčiami tik jiems skirtomis sąsajomis. Pagrindinis šio metodo privalumas yra suderinamumas su įvairaus tipo serveriais. Tačiau reikia atsižvelgti į sujungimo ilgį, nes šis metodas gali ne tik pagreitinti veikimą bet ir jį pabloginti.
- b) Planavimo metodai – tai būdai paskirstyti paketus skirtinguose daugiatinkliškumo sprendimuose. Egzistuoja nemažai įvairių metodų, todėl pabandysime trumpai apžvelgti tik pačius perspektyviausius ir populiariausius:
 - 1. „Round-Robin“ – šio metodo principas skirstyti paketu ratu, paketai skiriami visoms sąsajoms iš eilės jei tik ji turi vietos neatsižvelgiant į sąsajos talpą ar programos poreikius.

2. Kai kurios sistemos naudoja maksimalaus pralaidumo metodą, kai kiekviena nauja duomenų sesija priskiriama skirtingoms sąsajoms. Taip stengiamasi išnaudoti kiekvienos sąsajos spartą atskirai.
3. Pralaidumo vėlinimo metodas – planuoja paketų siuntimą pagal kiekvienos sąsajos vėlinimo parametrus.
4. Perpildymo kontrolės metodas – atsižvelgia į paketų perpildymo kontrolės mechanizmą. Paketai skirstomi pagal perpildymo lango talpą kiekvienai sąsajai atskirai. Naudojant šį metodą, perdavimo sparta padidėja beveik iki maksimalaus lygio kiekvienoje sąsajoje.
5. „Rate-based“ – įvertina visų sąsajų pralaidumą ir pagal jų dalį bendrame pralaidumo kiekyje atitinkamai paskirsto paketus.
6. Energijos ir kainos metodas – šiame metode paketai gali būti skirstomi pagal energijos suvartojimo kiekį kiekvienai sesijai, arba maksimizuojant perdavimo spartą atsižvelgiant į suvartojamą energijos kiekį (rodiklį nurodyti turi vartotojas).
7. Paslaugos kokybės (QoS – angl. *Quality of Service*) metodas – įvairūs metodai taikomi atsižvelgiant į kokybinius parametrus.

Tinklo palaikymo ir komunikavimo modelis – skirtingose sistemose taikomi skirtingi kliento ir serverio komunikavimo modeliai, todėl reikalingas skirtingas tinklo suderinamumas. Šiame modelyje stengiamasi maksimizuoti sistemos veikimą už mažiausią sprendimo pritaikymo kainą. Galima išskirti tris pagrindinius modelius (10 pav.):



10 pav. Tinklo palaikymo ir komunikavimo modeliai [2]

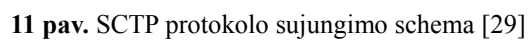
- a) Atnaujinto serverio komunikacijos modelis (angl. *Updated-Server-Based*). Šiam modeliui priklauso sistemos, kurias galima įgyvendinti nekeičiant centrinio tinklo dalių. Pakeitimai reikalingi tik kliento ir serverio įrengioje.
- b) Įgaliotojo serverio komunikavimo modelis (angl. *Proxy Based*). Pasitelkus šį modelį siekiama išvengti arba bent jau sumažinti pakeitimų kiekį serveryje. Įgaliotieji serveriai sujungia du, skirtingais keliais perduodamus, duomenų srautus ir serveriui pateikia kaip vieną, todėl serveryje reikalingi minimalūs kaštai sprendimui įgyvendinti.
- c) Nekeičiamo serverio komunikavimo metodas (angl. *Legacy-Server-Based*) – šio metodo tikslas visiškai nekeisti serverio parametrų, todėl galima įgyvendinti sprendimus reikalaujančius mažai resursų.

2.2. Dažniausiai naudojami sprendimai

Nepaisant daugybės egzistuojančių protokolų ir sprendimų daugiatainkliškumui įgyvendinti, dažniausiai naudojami tik keli. Galima išskirti du plačiausiai paplitusius protokolus, tai MPTCP ir SCTP protokolai. Dauguma kitų daugiatainkliškumo sprendimų yra šių protokolų patobulinimai ar pritaikymai skirtingoms problemoms spręsti.

MPTCP protokolas yra TCP protokolo papildymas, kuris leidžia naudoti daugiau nei vieną sujungimą tarp kliento ir serverio. Pagrindinis MPTCP protokolo tikslas yra pagerinti kanalų resursų išnaudojimą ir sumažinti nutrūkimų riziką. Kadangi šis protokolas paremtas TCP protokolo veikimu, jis yra visiškai suderinamas su tarpiniais tinklo įrenginiais, kurie naudoja TCP protokolą. Pagrindiniai MPTCP protokolo skirtumai, lyginant jį su kitais transporto lygmens protokolais, yra TCP jungties panaudojimas (lemia visišką suderinamumą su TCP protokolą naudojančiomis programomis) ir visų egzistuojančių sujungimų veikimas vienu metu (pagerina bendrą sujungimų spartą, resursų išnaudojimą) [7].

SCTP – tai transporto lygmens protokolas, kuris veikia panašiai kaip TCP ar UDP protokolai. SCTP protokolas pagrįstas žinučių perdavimu (panašiai kaip UDP), bei garantuoja žinučių pristatymą teisinga seka naudojant grūsties kontrolę (panašiai kaip TCP). Pagrindinis SCTP protokolo skirtumas – daugiatainkliškumo galimybė. Sujungimo sudarymo metu gavėjas ir siuntėjas apsikeičia adresų sąrašu, kuris nurodo kokiais adresais jie gali būti pasiekiami, pavyzdžiui IP (angl. *Internet Protocol*) adresais ir SCTP prievadais (11 pav.) [29].

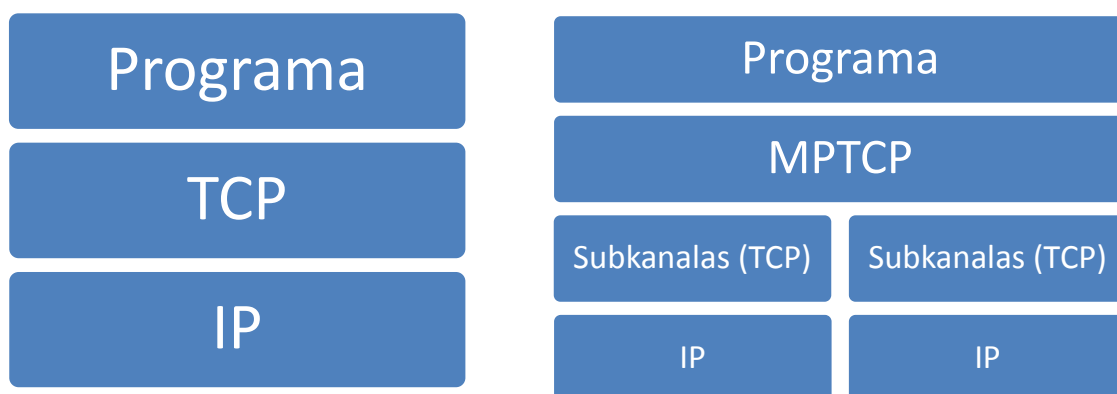


Pagrindinis STCP protokolo trūkumas yra programų stoka. Nors *Linux* tipo operacinėse sistemose šis protokolas įdiegtas pagal nutylėjimą, o *Windows* operacinėse sistemose taip pat galima jį įdiegti, dauguma interneto prieigą naudojančių programų naudoja TCP arba UDP transporto lygmens protokolus.

Išanalizavus šaltinius galima teigti, kad daugiatainkliškumas – tai galimybė įrenginiams turėti daugiau nei vieną prieigą prie interneto ir naudoti jas vienu metu. Egzistuoja daugybė įvairių būdų daugiatainkliškumui įgyvendinti, kuriuos galima suskirstyti pagal OSI lygmenis, taip pat pagal veikimo principą ir naudojamus algoritmus. Didžioji dalis sprendimų yra jau egzistuojančių protokolų modifikacijos ir patobulinimai. Dažniausiai šaltiniuose mini ir nagrinėjami yra MPTCP ir SCTP transporto lygmens protokolai. Būtent dėl gero suderinamumo su jau egzistuojančiu tinklu ir programomis nuspręsta detaliau nagrinėti ir tyrimą atlikti su MPTCP protokolu.

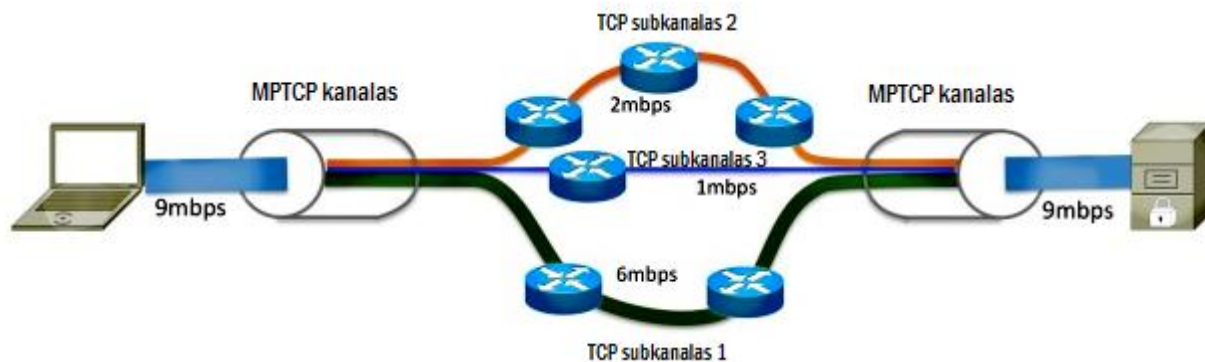
3. MPTCP PROTOKOLAS

MPTCP – tai papildymų rinkinys TCP protokolui, kuris leidžia sukurti TCP daugiatainkliškumo paslaugą, naudojant kelis TCP sujungimus vienu metu [13]. Pagal OSI modelį MPTCP veikia transporto lygmenyje ir yra nepastebimas tiek aukštesniems, tiek žemesniems lygmenims (12 pav.). Tai papildomų taisyklių rinkinys, veikiantis kartu su TCP protokolu. Pagrindinis uždavinys realizuojant MPTCP veikimą yra protokolo pritaikymas jau egzistuojančiame tinkle, neatliekant jokių pakeitimų [7].



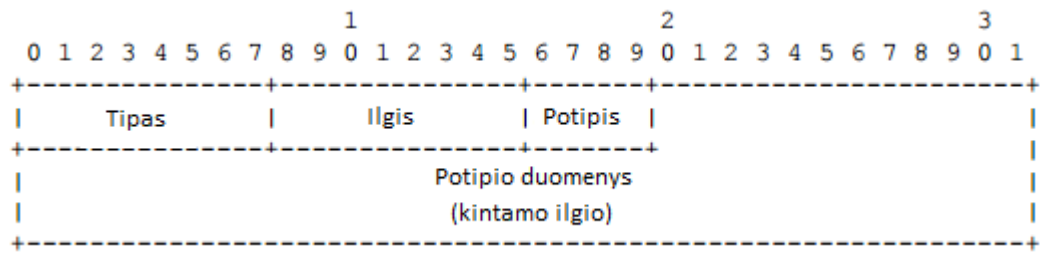
12 pav. TCP ir MPTCP dėklų skirtumai [8]

MPTCP protkolo sujungimo schema, naudojant daugiau nei vieną subkanalą, pavaizduota 13 paveikslėlyje.



13 pav. MPTCP subkanalų struktūra [26]

Visa informacija apie MPTCP protokolą yra perduodama naudojant papildomus TCP protokolo antraštės laukus. IANA (angl. *Internet Assigned Numbers Authority*) suteikė MPTCP protokolui kodą papildomoje TCP antraštės dalyje, kuriuo leidžiama naudoti MPTCP [12]. Kita reikalinga informacija nurodoma papildomuose laukuose, kurie taip pat įtraukti į IANA registrą. MPTCP antraštės formatas pavaizduotas 14 pav.

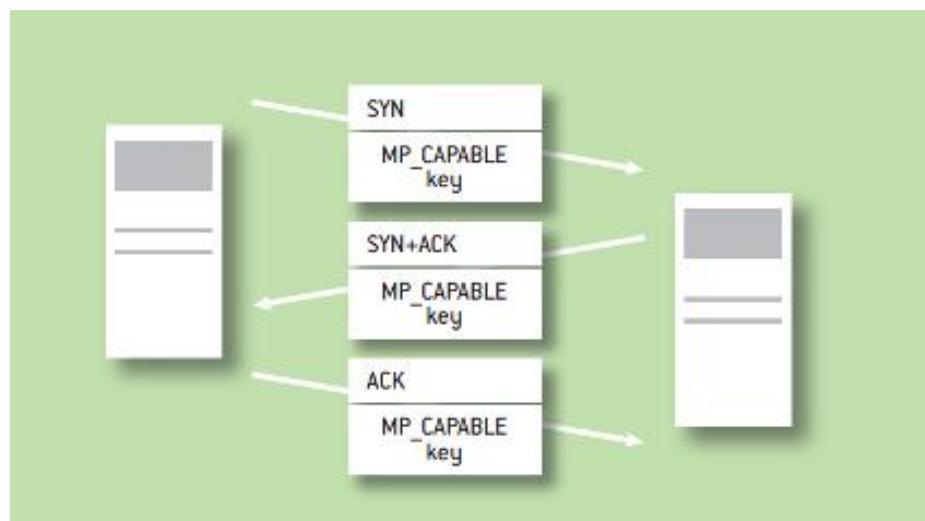


14 pav. MPTCP antraštės struktūra [8]

Pirmieji 8 bitai nurodo tipą, kad pasirinkimų antraštė bus naudojama MPTCP protokolui aprašyti. Sekantys 8 bitai nurodo antraštės ilgį, o 4 potipio bitai MPTCP žinutės tipą. Toliau eina kintamo ilgio papildoma informacija pagal pasirinktą žinutės tipą.

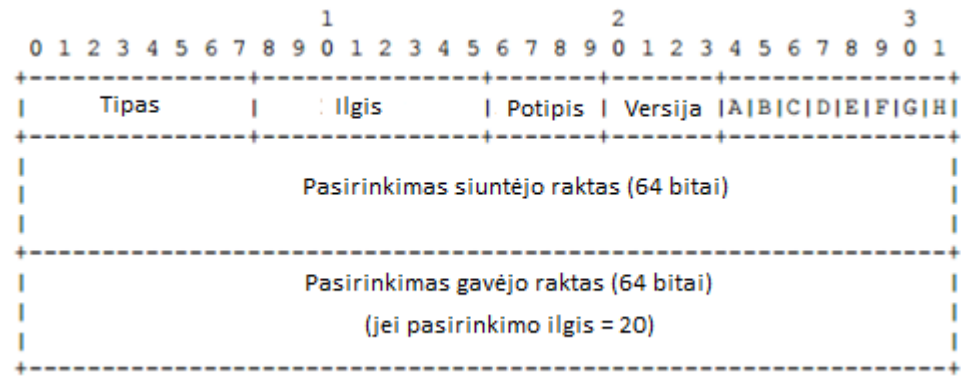
3.1. Sujungimo sudarymas

Sujungimo sudarymas prasideda *SYN*, *SYN/ACK*, *ACK* žinučių apsikeitimu tarp serverio ir kliento pagrindiniu keliu. Kiekviename pakete yra *MP_CAPABLE* parinktis, kuri nurodo jog siuntėjas geba naudoti ir nori naudoti MPTCP protokolą. Šios operacijos metu perduodami 64 bitų ilgio raktai, kuriuos siuntėjas sugeneruoja MPTCP sujungimui sukurti. Raktai naudojami sekantiems subkanalams sudaryti ir atskirti. Sujungimo procedūra pavaizduota 15 paveikslėlyje [7].



15 pav. MPTCP sesijos užmezgimas [4]

MP_CAPABLE parinktis perduodama *SYN*, *SYN/ACK* ir *ACK* paketais, kurie pradeda pirmąjį MPTCP sujungimą. Siunčiant *SYN* paketą, siuntėjas nurodo atsitiktinai sugeneruotą skaičių seką (raktą), o gavėjas jam atsako atsiųsdamas *SYN/ACK* paketą su savo unikaliu raktu šiam sujungimui. Tada siuntėjas išsiunčia patvirtinimą *ACK* ir dar kartą nurodo savo raktą bei gavėjo raktą. *MP_CAPABLE* antraštės struktūra pavaizduota 16 pav. [8].



16 pav. MP_CAPABLE antraštės struktūra [8]

Sujungimo sudarymo potipio lauke nurodoma, kad siunčiamas *MP_CAPABLE* tipo pranešimas. Versijos laukas nurodo MPTCP protokolo versiją, o likę 8 bitai aprašo naudojamą vėliavėlę (angl. *Flags*):

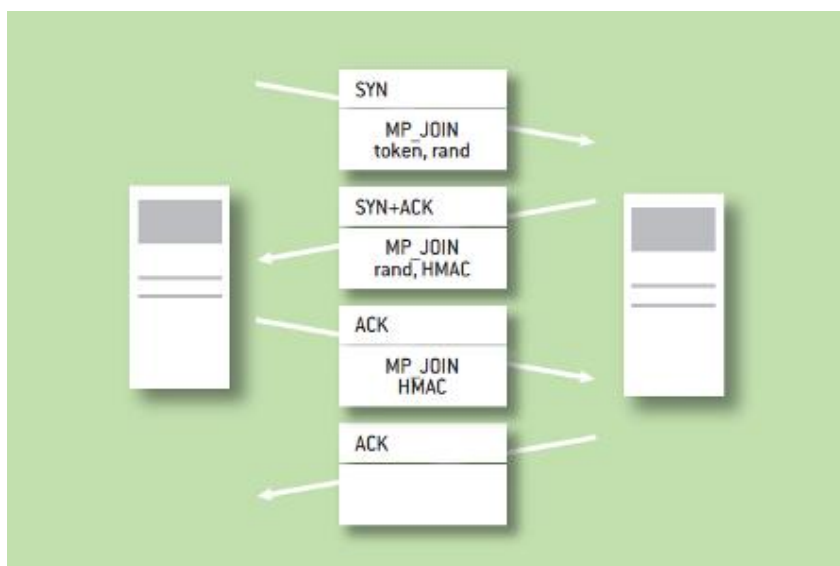
- A* vėliavėlė nurodo ar reikalinga kontrolinė bitų suma. Kai $A = 1$ kontrolinė bitų suma yra naudojama. Jei bent vienas iš sujungimo dalyvių nurodo, kad kontrolinė bitų suma turi būti naudojama, kitas taip pat bus priverstas ją naudoti.
- B* vėliavėlė naudojama papildomoms funkcijoms, kurios bus apibrėžtos ateityje. Šiandienos specifikacija nurodo naudoti $B = 0$. Jei gaunamas paketas su verte $B = 1$, jis yra ignoruojamas.
- C – H* vėliavėlės naudojamos kodavimo algoritmo derinimui. Šiandien naudojama tik *H* vėliavėlė, kuri nurodo naudoti kol kas vienintelį palaikomą algoritmą HMAC-SHA1 (angl. *Hash Message Authentication Code - Secure Hash Algorithm*). Likusios vėliavėlės nuo *C* iki *G* turi būti lygios 0, kitu atveju toks paketas bus ignoruojamas ir pereinama prie tradicinio TCP protokolo naudojimo.

3.2. Subkanalo prijungimas

Kai sukuriamas pirmasis MPTCP sujungimas, atsiranda galimybė sukurti papildomus subkanalus prie jau egzistuojančio sujungimo. Siuntėjas ir gavėjas žino papildomus savo adresus, o kito įrenginio papildomą adresą gali sužinoti apsikeičiant signalizacijos žinutėmis. Kai turimas papildomas mazgo adresas, galima pradėti papildomo subkanalo sudarymą jau sudarytu sujungimu. Papildomą sujungimą leidžiama sudaryti abiem mazgams, tačiau priimta, kad sujungimą pradeda siuntėjas [8].

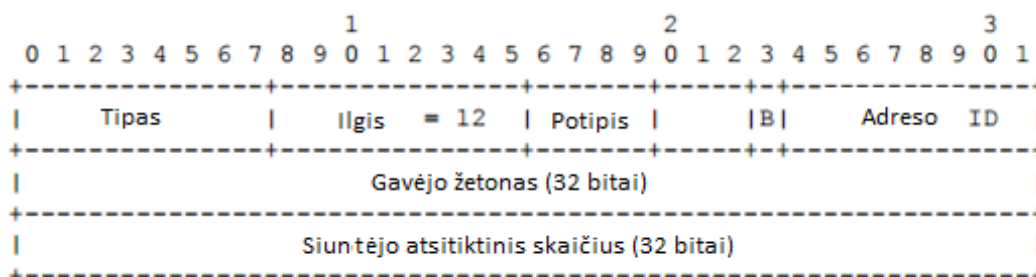
Naujas sujungimas pradedamas kaip paprastas TCP sujungimas, apsikeičiant *SYN/ACK* žinutėmis, o kad tai papildomas sujungimas nurodo *MP_JOIN* parinktis (17 pav.). Papildomam

sujungimui sudaryti naudojami prieš tai sugeneruoti gavėjo ir siuntėjo raktai, pagal kuriuos sudaromas HMAC-SHA1 algoritmas.



17 pav. Papildomos MTCP sesijos užmezgimas [4]

Pirmuoju *SYN* paketu siuntėjas nurodo žetoną (angl. *Token*), atsitiktinį skaičių ir adresą ID (18 pav.). Žetonas – tai užkoduotas gavėjo raktas, naudojamas atskirti jau egzistuojančiam MPTCP sujungimui. Atsitiktinis skaičius skirtas išvengti pakartojimo atakų. Adreso ID nurodo siuntėjo adresą, taip apeinama NAT (angl. *Network Address Translation*) sistema, kuri gali pakeisti siuntėjo IP adresą TCP antraštėje. Taip pat adresą ID užkerta kelią sukurti dubliuotus sujungimus, jei *MP_JOIN* ir *ADD_ADDR2* pasirinkimai yra siunčiami vienu metu [8].

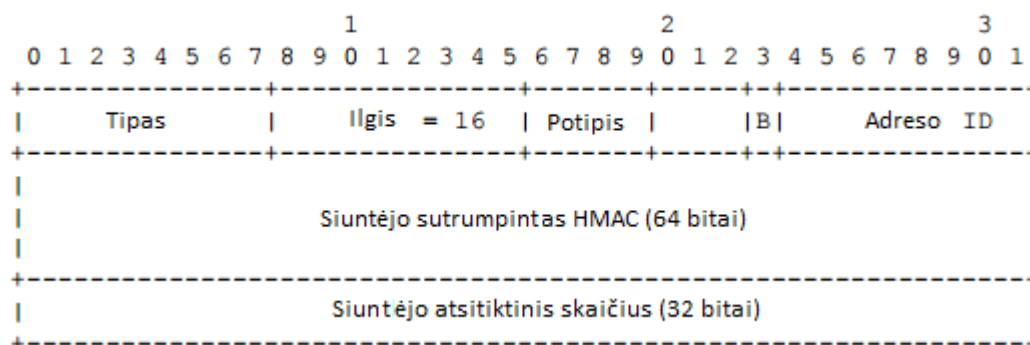


18 pav. *MP_JOIN* antraštės struktūra (kai siunčiamas *SYN*) [8]

MP_JOIN SYN paketuose 4 bitai yra skirti vėliavėlėms. Pirmieji 3 bitai yra rezervuoti ateičiai ir privalo būti lygūs 0. Paskutinis bitas *B* nurodo ar siuntėjas naująjį subkanalą nori naudoti kaip atsarginį kitų subkanalų nutrūkimo atveju ($B = 1$) ar nori naudoti šį subkanalą kaip papildomą sujungimą ($B = 0$).

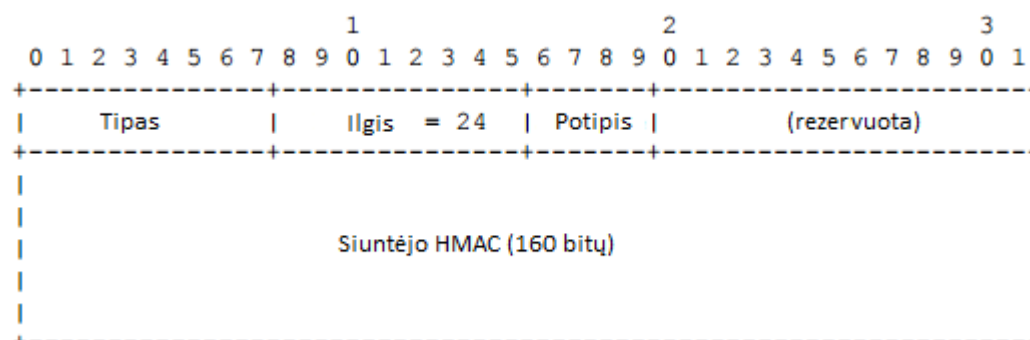
Jei informacija *SYN* pakete nurodyta teisingai, į jį gavėjas atsako siųsdamas *MP_JOIN SYN/ACK* paketą (19 pav.). Pakete nurodomas siuntėjo sutrumpintas HMAC kodas ir atsitiktinis siuntėjo sugeneruotas skaičius. Jei gavėjas neturi žetono arba nenori sudaryti papildomo subkanalo,

jis siunčia perkrovimo (angl. *Reset*) signalą (analogišką kai nežinomas TCP prievadas) su *MP_TCPRST* parinktimi, nurodant atmetimo priežasties kodą.



19 pav. MP_JOIN antraštės struktūra (kai siunčiamas SYN/ACK) [8]

Siuntėjas į *MP_JOIN SYN/ACK* atsako siųsdamas *ACK* paketą su siuntėjo HMAC kodu (20 pav.). Kadangi HMAC kodai yra siunčiami tik vieną kartą, jų siuntimas privalo būti patikimas. Gavėjui negavus *ACK* jis kartoja *SYN/ACK* paketo siuntimą.



20 pav. MP_JOIN antraštės struktūra (kai siunčiamas ACK) [8]

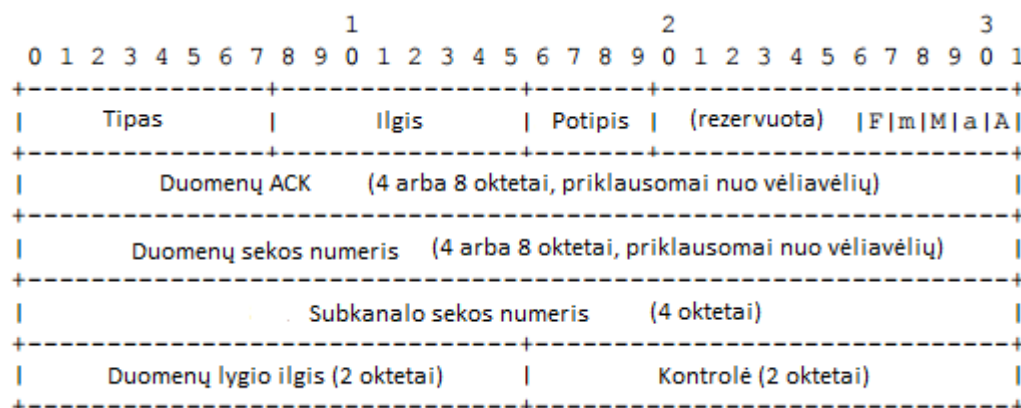
Jei siuntėjas ar gavėjas gauna neatpažįstamą žetoną ar HMAC kodą jis privalo atsakyti *RST* žinute ir uždaryti TCP sesiją. Taip padidinama apsauga nuo įsibrovėlių, kadangi atspėti HMAC jie turi tik vieną galimybę. Nepataikius – TCP sesija yra uždaroma. Iš naujo kuriant subkanalą naudojami jau naujai sudaryti atsitiktiniai skaičiai.

3.3. Duomenų perdavimas

Perduodant duomenis MPTCP protokolu, iš programos yra paimamas vienas duomenų srautas, suskaidomas į vieną ar daugiau srautų su papildoma informacija, kurios pagalba galima patikimai perduoti srautus atskirais subkanalais ir vėl sudėti duomenis į vientisą srautą, prieš perduodant gavėjo programai.

Duomenų sekos žymėjimas ir duomenų *ACK* yra nurodomas DSS (angl. *Data Sequence Signal*) papildomoje antraštėje (21 pav.). Juos abu galima perduoti tiek vienoje, tiek atskirose DSS

antraštėse priklausomai nuo vėliavėlių žymėjimo. Duomenų sekos žymėjimas nurodo kaip sekos erdvė veikia subkanalo sujungimo lygmenyje bei kaip veikia *ACK* patvirtinimai sujungimo lygmenyje.



21 pav. DSS antraštės struktūra [8]

Vėliavėlių reikšmė antraštėje:

- A* - nurodytas *ACK*.
- a* - duomenų *ACK* užima 8 oktetus (pagal nutylėjimą *ACK* užima 4 oktetus).
- M* - duomenų sekos numeris (angl. *Data Sequence Number*), subkanalo duomenų antraštės ilgis, kontrolinė suma.
- m* - duomenų sekos numeris užima 8 oktetus (pagal nutylėjimą užima 4 oktetus).
- F* - duomenų pabaiga. Jei $F = 1$, siunčiami paskutiniai duomenys.

Vėliavėlės *a* ir *m* turi prasmę tik tada, kai *A* ir *M* vėliavėlės lygios 1, kitu atveju jos bus ignoruojamos. Maksimalus antraštės ilgis 28 oktetai (224 bitai). Sujungimas neuždaromas, kol neatsiunčiamas DSS su $F = 1$ arba nepasibaigia nustatytas laikas. Rezervuoti bitai privalo būti lygūs 0.

Kontrolinė suma naudojama tik tada, jei atliekant *MP_CAPABLE* sujungimą, buvo numatyta naudoti kontrolinę sumą. Jei buvo sutarta nenaudoti kontrolinės sumos, o DSS ji yra nurodyta, kontrolinė suma ignoruojama. Jei buvo sutarta naudoti kontrolinę sumą, o ji nenurodyta, subkanalas privalo būti uždarytas, kadangi laikoma, kad jis klaidingas. Sujungimas uždaromas siunčiant *MP_TCPRST* žinutę.

3.3.1. Duomenų sekos žymėjimas

Pagal DSS – perduodamų duomenų srautas yra surenkamas iš atskirų subkanalų srautų. DSS pagalba, viso srauto duomenų seka susiejama su subkanalo sekos žymėjimu. Taip gavėjas sugeba atkurti perduotų duomenų eilės tvarką ir pateikia duomenis programai. Subkanalo sekos numeris turi prasmę tik tame subkanale.

Žymėjimas yra fiksuotas ir subkanalo sekos numeris yra surištas su duomenų sekos numeriu. Siuntėjas negali jų keisti prasidėjus duomenų siuntimui, tačiau tas pats duomenų sekos numeris gali būti susietas su kito subkanalo sekos numeriu kai atliekamas duomenų persiuntimas. Tokiu būdu galima siųsti tuos pačius duomenis skirtingais subkanalais siekiant užtikrinti patikimą duomenų perdavimą. Gavus vienodus duomenis skirtingais kanalais naudojami pirmiausiai gauti duomenys, pasikartojantys – ignoruojami [8].

Kontrolinė suma reikalinga tam, kad būtų galima patikrinti ar nebuvo atlikta pakeitimų su duomenimis. Dažniausiai tokie pakeitimai gali būti atliekami MPTCP protokolo nesuprantančiuose mazguose. Tokiu atveju subkanalas turi būti uždarytas arba pereinama prie paprasto TCP sujungimo, kadangi tampa nebeaišku kokia yra subkanalo seka. Kaip ir TCP protokolo atveju, sekos numeris turėtų prasidėti ne nuo 0, o nuo atsitiktinio skaičiaus, kad būtų sunkiau užgrobti sesiją įsilaužėliams.

Duomenų sekos žymėjimas nebūtinai turi būti nurodytas kiekviename MPTCP pakete, jei subkanalo seka imtuve yra žinoma. Tokiu būdu galima sumažinti perteklinę informaciją. Tai įmanoma tada kai veikia tik vienas MPTCP kanalas arba segmentai suskaidyti į didelius paketus.

3.3.2. Duomenų patvirtinimai

Siekiant užtikrinti pilną duomenų perdavimo patikimumą, MPTCP naudoja patvirtinimus sujungimo lygmenyje. Kiek vienas subkanalas turi savo patvirtinimo sistemą. Patvirtinimai nurodomi duomenų *ACK* laukelyje (21 pav.). Duomenų patvirtinimas veikia kaip ir standartiniame TCP protokole, nurodant koks duomenų kiekis buvo sėkmingai priimtas (be tarpų). Taigi subkanaluose *ACK* veikia analogiškai TCP *SACK*, atsižvelgiant, kad gali būti likusių tarpų, kuriuose trūksta duomenų. Duomenų *ACK* nurodo sekantį sekos numerį, iš kurio tikisi gauti duomenis.

Duomenų *ACK* gali būti 32 arba 64 bitų dydžio. Jei duomenys gauti su 64 bitų dydžio DSN reikia atsakyti 64 bitų dydžio *ACK*. Jei duomenys gauti su 32 bitų dydžio DSN, galima rinktis kokio dydžio *ACK* siųsti.

Duomenų patvirtinimas parodo, kad duomenys buvo sėkmingai priimti. Patvirtinimų pagalba galima nustatyti kairį gavimo lango kraštą. Gavimo langas yra bendras visiems subkanalams. Todėl siunčiant paketus negalima naudoti *RCVWND* laukelio, jei pakete nenurodomas DSS su duomenų *ACK* laukeliu. Dėka skirtingų paketo patvirtinimų subkanaluose atsiranda galimybė juos valdyti skirtingai.

3.3.3. Sujungimo nutraukimas

Paprastame TCP protokole, *FIN* paketas praneša imtuvui, kad siuntėjas nebeturi daugiau siunčiamų duomenų. Tam, kad subkanalai galėtų veikti nepriklausomai vienas nuo kito, *FIN* paketas

veikia tik tą kanalą, kuriuo buvo pasiųstas. Tokiu būdu mazgams leidžiama pasirinkti kiek ir kokių kanalų naudoti. *FIN* pranešimo struktūra išlieka tokia pati kaip TCP protokole.

Jei programa nori uždaryti jungtį – tai reiškia, kad ji nebeturi daugiau duomenų, kuriuos nori siųsti. Paprastame TCP sujungime tai reikštų, kad galima uždaryti TCP sesiją. MPTCP protokole reikalingas atskiras mechanizmas vadinamas *DATA_FIN*.

DATA_FIN perduodamas nustatant $F = 1$ DSS parinktyje. Kadangi sujungimo nutraukimas veikia MPTCP sesijos lygmenyje, visi duomenys siųsti visais subkanalais prieš tai turi būti patvirtinti. Kai gaunamas *ACK* dėl sesijos nutraukimo, visi likę subkanalai gali būti nutraukiami. Subkanalai, kuriuose dar yra likę neišsiųstų duomenų (kitais kanalais pakartota informacija), gali būti uždaromi naudojant *MP_TCPRST* funkciją.

Sujungimas laikomas uždarytu kai abu mazgai patvirtina, kad gavo *DATA_FIN* atsakydami *DATA_ACK* [8].

3.3.4. Gavėjo veikimas

Paprastame TCP sujungime priėmimo lango dydis yra pranešamas siuntėjui, kad siuntėjas žinotų kokį kiekį duomenų gali siųsti. Priėmimo langas naudojamas siuntimo srauto kontrolei, sulėtinant daug duomenų perduodančius mazgus, kai gavėjas nebespėja apdoroti duomenų.

MPTCP taip pat naudoja unikalų priėmimo langą, kuris yra bendras visiems subkanalams. Tokiu būdu leidžiama bet kuriam subkanalui siųsti duomenis tol, kol imtuvas sugeba juos priimti. Skirtingi priėmimo langai sulėtintų kai kuriuos kanalus, o kitus išnaudotų nepakankamai.

Kai sprendžiama, ar priimti paketą subkanalo lygmenyje, paprastas TCP palygina sekos numerį pakete su leidžiamu priėmimo lango dydžiu. Kai naudojamas MPTCP protokolas, toks palyginimas vyksta sujungimo lygmenyje. Taip pat paketai yra skirstomi į iš eilės atėjusių paketų eilę ir ne iš eilės atėjusių paketų eilę.

3.3.5. Siuntėjo veikimas

Siuntėjas privalo atsižvelgti į gavėjo priėmimo lango dydį. Savo priėmimo langą siuntėjas turėtų keisti tik kai padidėja didžiausias leistinas sekos numeris gavus *DATA_ACK*. Tai svarbu naudojant kelius su skirtingais RTT laikais, kadangi jie turi skirtingą atsako trukmę.

MPTCP naudoja vieną priėmimo langą visiems subkanalams, tad jei priėmimo langas bus nekeičiamas, galima naudoti paskutinį žinomą priėmimo lango dydį. Tačiau kai kurie tarpiniai tinklo įrenginiai gali keisti priėmimo lango dydį TCP lygmenyje, o tai sumažina MPTCP siūlomo priėmimo lango dydį. Tai trunka neilgai, kadangi tarpinis įrenginys vis tiek gauną didelį duomenų kiekį ir privalės padidinti priėmimo lango dydį. Taigi jei priėmimo lango dydis bus skirtingas keliuose subkanaluose, siunčiant duomenis reikia naudoti didžiausią iš paskutinių priėmimo langų.

Siuntėjas taip pat privalo prisiminti kiekvieno subkanalo leistiną priėmimo lango dydį.

Iš abiejų teiginių tampa aišku, kad siuntėjas gali siųsti duomenų segmentus su duomenų sekos numeriu nuo *DATA_ACK* iki *DATA_ACK* + priėmimo lango dydis.

Siuntimo buferis turi būti bent tokio pat dydžio kaip priėmimo buferis. Kitaip bus neįmanoma pasiekti maksimalią spartą.

3.3.6. Patikimumas ir persiuntimas

Duomenų sekos žymėjimas suteikia galimybę pakartoti tuos pačius duomenis kitu subkanalu, tačiau siuntėjas privalo pakartoti duomenis ir pradiniu subkanalu. Gavėjas pasikartojančius duomenis tiesiog ignoruos.

Siuntėjas privalo laikyti duomenis siuntimo buferyje tol, kol duomenys bus patvirtinti sujungimo lygmenyje bei subkanaluose, kuriais jie buvo siunčiami. Tokiu būdu atsiranda galimybė pakartoti duomenis tiek tuo pačiu subkanalu, tiek kitu. Gali pasitaikyti situacija kai vienas iš subkanalų nutrūksta. Tokiu atveju, pasibaigus tam tikram laikui, duomenys bus kartojami kitu subkanalu, tačiau bus bandoma siųsti ir tuo kanalu kuris nutrūko. Tik po tam tikro bandymų skaičiaus, bus laikoma, kad kanalas nutrūkęs.

Didelis duomenų persiuntimų skaičius rodo, kad subkanalas veikia nepatikimai ir gali privesti prie kanalo perkrovimo naudojant *RST*.

3.3.7. Grūsties kontrolė

Skirtingi MPTCP subkanalai turi skirtingą grūsties langą. Kad būtų pasiekta subkanalų lygybė „butelio kakliuko“ linijose ir resursų naudojime, būtina sujungti grūsties langus kiekviename subkanale. Taip galima valdyti duomenų srautą ir didesnę duomenų kiekį siųsti mažiau apkrautais kanalais. Galima naudoti įvairius grūsties kontrolės algoritmus, svarbu, kad algoritmai būtų suderinami su jau egzistuojančiu tinklu.

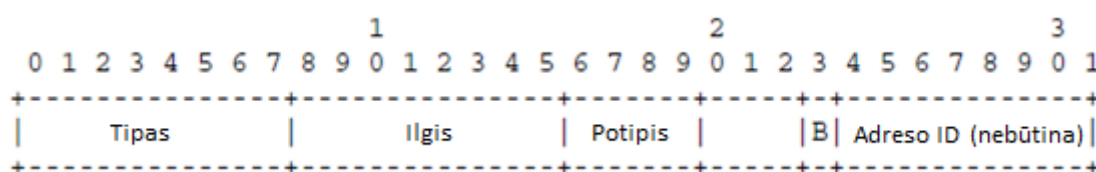
MPTCP galima naudoti skirtingus grūsties lango algoritmus. Kiekvienas jų turi skirtingas resursų skirstymo, stabilumo ar patikimumo savybes. Nepaisant skirtingų grūsties kontrolės algoritmų MPTCP protokolas pateikia pakankamai informacijos grūsties kontrolei, kad būtų priimami teisingi sprendimai. Pateikiamoje informacijoje nurodoma kada ir kiek paketų buvo prarasta kiekviename subkanale.

3.3.8. Subkanalų strategija

Lokaliame MPTCP tinkle vartotojas gali rinktis bet kurią subkanalų panaudojimo strategiją: pilno išnaudojimo, atsarginio subkanalo, perpildymo. Pagrindinis MPTCP tikslas – pasiekti maksimalią duomenų perdavimo spartą, todėl vienu metu naudojami visi įmanomi kanalai (pilnas

išnaudojimas). Taip pat yra numatyta, kad vartotojai gali siekti ir kitokių tikslų. Pavyzdžiui, naudoti papildomus kanalus tik tada, kai pirmasis kanalas nutrūksta (atsarginio subkanalo), arba padidinti kanalų skaičių, jei nepakanka duomenų perdavimo spartos jau naudojamais kanalais (perpildymo).

Tokie taikymai dažniausiai naudojami kai norima atsižvelgti į skirtingų kanalų savybes. Skirtingi keliai turi skirtingą vėlinimą bei vėlinimo švytavimą, taip pat gali būti taikomas skirtingas apmokestinimas. Būtent todėl *MP_JOIN* pranešime yra atsarginio kanalo vėliavėlė *B*, kuri nurodo ar kanalas skirtas tik atsargai. Jei situacija pasikeičia ir norima pakeisti kanalo prioritetą, siunčiama žinutė *MP_PRIO* (22 pav.) [8].



22 pav. *MP_PRIO* antraštės struktūra [8]

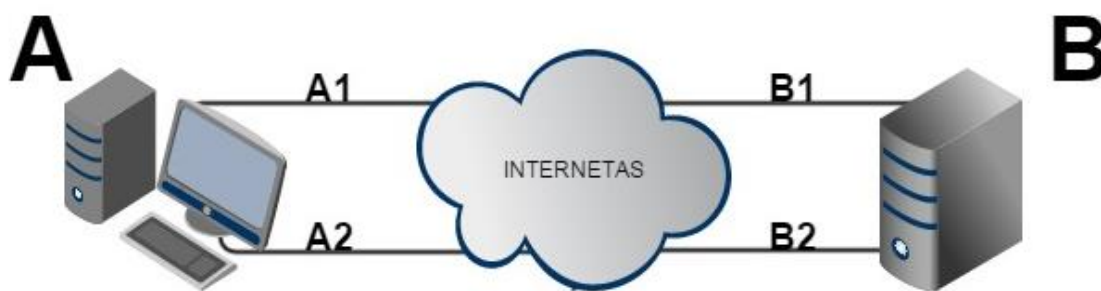
Reikia paminėti, kad *MP_PRIO* gali siųsti tik gavėjas, taip pat ši žinutė veikia tik viena kryptimi. Jei siuntėjas pasikeitė kanalo tipą į „atsarginį“, gavėjas vis tiek gali juo ir toliau siųsti duomenis. Su *MP_PRIO* žinute galima pakeisti ir kito subkanalo prioritetą nurodžius to kanalo adreso ID.

3.4. Adresų apsikeitimas (kanalų valdymas)

Kanalų valdymas nusako papildomų kanalų adresų apsikeitimą tarp mazgų. MPTCP architektūroje galimi du būdai adresų apsikeitimui. Pirmasis būdas jau paminėtas 2.2 poskyryje, kai siuntėjas iš anksto žino kitus gavėjo adresus. Antrasis būdas yra sudėtingesnis, kadangi adresai turi būti perduodami jau egzistuojančiais sujungimais. Naudojant abu šiuos būdus, adresais galima keistis pilnai veikiančioje sistemoje.

Apžvelgsime dažniausiai naudojamus sujungimo sudarymo būdus (23 pav.):

- Pirmasis sujungimas jau sudarytas tarp *A1* ir *B1* mazgų. Jei mazgas turi daugiatinkliškumo galimybę, jis gali sukurti antrąjį sujungimą tarp *A2* ir *B1*, siųsdamas *SYN* paketą su *MP_JOIN* pranešimu. Taip pat, jei mazgas *B* turi daugiatinkliškumo galimybę, jis gali sudaryti antrąjį sujungimą tarp *B2* ir *A1*.
- Po kanalo sudarymo arba po kurio laiko galima siųsti *ADD_ADDR2* pranešimą, kuriuo gavėjas yra informuojamas apie kitus siuntėjo adresus. Gavėjas, pasinaudojus gauta informacija, gali sudaryti sujungimą su nurodytais adresais.

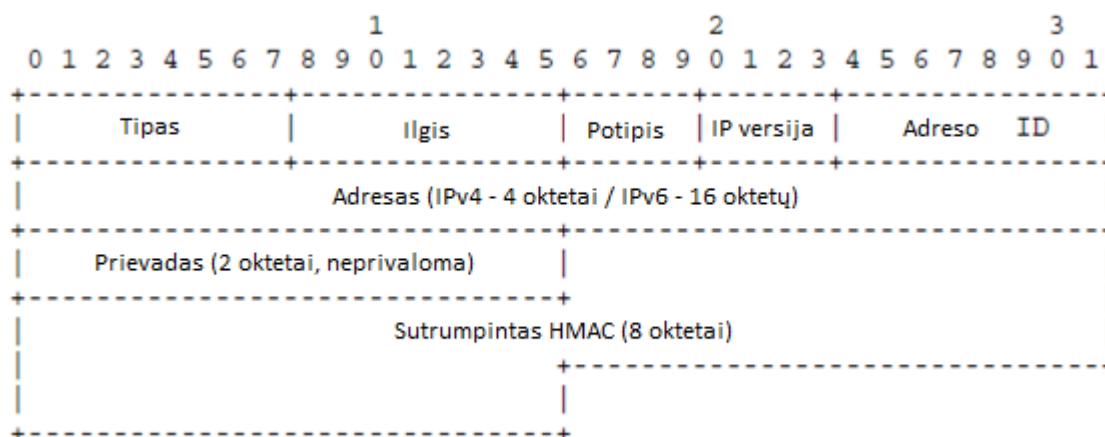


23 pav. Daugiatinkliškumo pavyzdys

3.4.1. Adresų siūlymas

Adresų pridėjimo funkcija *ADD_ADDR2*, perduoda žinutės gavėjui adresus (jei reikia ir prievadus), kuriais galima pasiekti siuntėją. Šis pranešimas gali būti siunčiamas bet kuriuo sujungimo metu, priklausomai nuo to, kada siuntėjas nori pranešti apie savo kitus adresus.

Kiekvienas adresas turi jam priskirtą adreso ID (angl. *Identifier*), kuris turi būti unikalus. Visi adresai, apie kuriuos sužinoma iš *MP_JOIN* ir *ADD_ADDR2* pranešimų, privalo būti saugomi gavėjo duomenų bazėje, kad bet kada prireikus būtų galima juos panaudoti ar ištrinti. Detalesnė *ADD_ADDR2* antraštė pateikta 24 paveikslėlyje. Antraštės laukelyje *IPver* nurodoma IP protokolo versija, kurią naudoja *Address* laukelyje nurodytas IPv4 (angl. *Internet Protocol Version 4*) arba IPv6 (angl. *Internet Protocol Version 6*) mazgo adresas. Prievado laukas *Port* (jei naudojamas) nurodo prievado numerį. Jei prievado numeris nepateiktas, sujungimą bandoma atlikti naudojant tokį pat kanalo prievadą, kuriuo atėjo *ADD_ADDR2* pranešimas. Antraštės gale nurodomas sutrumpintas HMAC kodas, kuris yra sugeneruojamas kaip ir *MP_JOIN* pranešimo atveju.



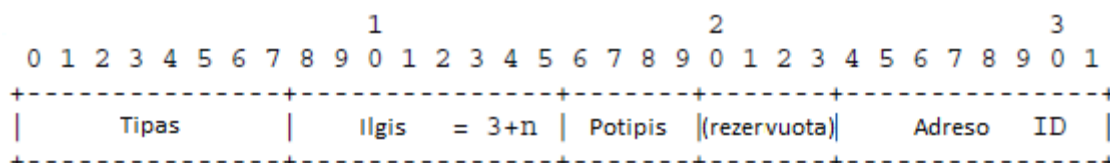
24 pav. *ADD_ADDR2* antraštės struktūra [8]

Perduodant *ADD_ADDR2* pranešimus rekomenduojama juos siųsti visais turimais kanalais, siekiant sumažinti paketo pradingimo tikimybę. Taip pat galima siųsti pranešimą su jau egzistuojančiu adreso ID, tačiau tuomet mazgo adresas turi būti toks pat kaip ir prieš tai nurodytame adreso ID, o prievadas privalo būti skirtingas. Jei šios dvi sąlygos netenkinamos, gavėjas tokią žinutę

ignoruoja. Pridėti identišką adreso ID į duomenų bazę galima tik prieš tai nutrynus jau egzistuojantį adreso ID.

3.4.2. Adreso šalinimas

Jei MPTCP sujungimo metu viena iš siuntėjo sąsajų tampa nebeaktyvi, apie tai turėtų būti pranešama gavėjui, kad gavėjas galėtų panaikinti visus sujungimus su nebeegzistuojančiu subkanalu. Apie tokį pasikeitimą pranešama *REMOVE_ADDR* pranešimu (25 pav.).



25 pav. REMOVE_ADDR antraštės struktūra [8]

Dėl saugumo sumetimų, kai mazgas gauna *REMOVE_ADDR* pranešimą jis privalo patikrinti ar tikrai tas adresas nebepasiekiamas. Patikrinimas vykdomas siunčiant TCP palaikymo (angl. *Keepalive*) pranešimą į planuojamą ištrinti adresą. Jei atsakymas gaunamas, adresas nėra šalinamas. Jei atsakymas negaunamas, toks adresas yra ištrinamas iš adresų duomenų bazės ir uždaromi visi su juo susiję sujungimai, siunčiant *RST* pranešimą. Veikiantys subkanalai turėtų būti uždaromi naudojant reguliarius TCP *FIN* pranešimus.

3.5. Greitas nutraukimas

Paprasti TCP sujungimai yra uždaromi siunčiant perkrovos pranešimą *RST*. Naudojant MPTCP, *RST* pranešimas uždaro tik vieną subkanalą, kiti kanalai lieka nepalieti. Norint uždaryti visą MPTCP sujungimą, naudojamas *MP_FASTCLOSE* pranešimas (26 pav.).

Šis pranešimas reiškia, kad sujungimas bus nutrauktas ir daugiau duomenų nebebus priimama. Sąlygos, kada siunčiamas toks pranešimas, priklauso nuo MPTCP protokolo taikymo konkrečiu atveju.



26 pav. MP_FASTCLOSE antraštės struktūra [8]

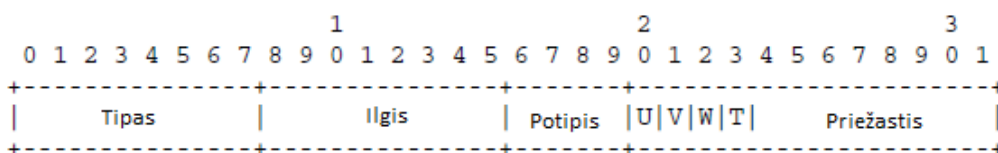
Jei norima visiškai nutraukti MPTCP sujungimą, nutraukimas vykdomas tokia seka:

- a) Mazgas *A* siunčia *ACK* pranešimą su *MP_FASTCLOSE* funkcija vienu iš subkanalų. Funkcijoje nurodomas gavėjo raktas, kurį gavo pirmojo sujungimo metu. Visais kitais subkanalais siuntėjas siunčia paprastą *TCP RST* pranešimą.
- b) Gavęs *MP_FASTCLOSE* pranešimą, mazgas *B* atsako siųsdamas *TCP RST* pranešimus visais subkanalais. Nutraukęs visus sujungimus su mazgu *A*, mazgas *B* uždaro visą *MPTCP* sujungimą.
- c) Kai tik mazgas *A* gauna *TCP RST* pranešimą likusiais subkanalais, jis gali uždaryti visą *MPTCP* sujungimą. Jei mazgas *A* gauna *MP_FASTCLOSE* žinutę, reiškia, kad abu mazgai nutraukė *MPTCP* sujungimą vienu metu. Į *MP_FASTCLOSE* žinutę mazgas *A* atsako *TCP RST* pranešimu.
- d) Jei mazgas *A* negauna jokio atsakymo į *MP_FASTCLOSE* pranešimą jis yra kartojamas po vieno *RTO* (angl. *Retransmission Timeout*) (kanalu, kuriuo buvo siųstas pirmasis pranešimas). Pakartotinių siuntimų skaičius turėtų būti ribotas, rekomenduojamas kartojimų skaičius yra 3.

3.6. Subkanalų perkrovimas

Kaip jau minėta anksčiau, *TCP RST* pranešimas gali nutraukti sujungimą su vienu iš subkanalų, kuriuo jis yra siunčiamas. *TCP RST* naudojami pasirinkimai nurodo, kodėl norima uždaryti sujungimą, tačiau tik kanalo lygmenyje.

Naudojant *MPTCP*, gali būti daug priežasčių, kodėl atmetamas subkanalo sujungimas, bet jų negalima nurodyti naudojant *TCP RST*. Siekiant geriau perteikti informaciją mazgams, kodėl neleista sudaryti sujungimo ir kada jį bus galima sudaryti, naudojama *MP_TCPRST* funkcija (27 pav.). Ji perduodama kartu su *TCP RST* pranešimu.



27 pav. *MP_TCPRST* antraštės struktūra [8]

MP_TCPRST funkcija leidžia nurodyti atmetimo priežasties kodą, kuris pateikia detalesnę informaciją, kodėl įvyko atmetimas. 4 bitai yra naudojami vėliavėlėms nurodyti (šiai dienai iš jų naudojama tik *T*), o likę 8 bitai nurodo priežasties kodą. Vėliavėlė *T* nurodo ar atmetimas galioja trumpam ($T = 1$), ar visam laikui ($T = 0$). Jei atmetimas galioja ribotą laiką, mazgas po kurio laiko gali vėl bandyti sudaryti sujungimą. Kitos vėliavėlės yra rezervuotos ateičiai ir šiuo metu nenaudojamos.

Galimos atmetimo priežastys:

- a) Neapibrėžta klaida (angl. *Unspecified Error*) – subkanalas nebegalimas.

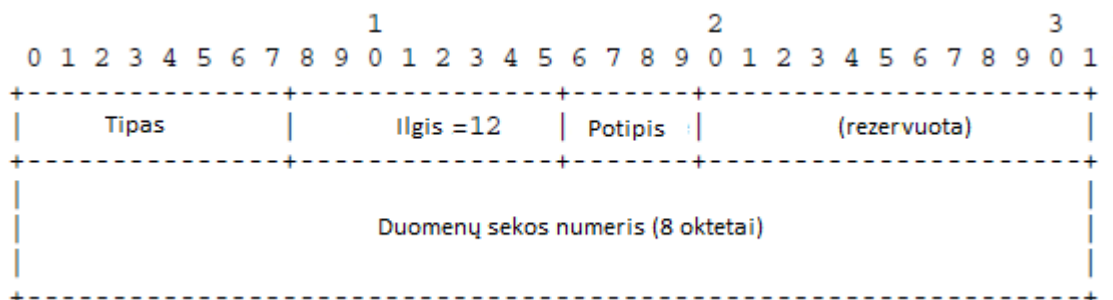
- b) MPTCP klaida (angl. *MPTCP Specific Error*) – klaida MPTCP pasirinkimų lauke.
- c) Trūksta resursų (angl. *Lack of Resources*) – mazgui neužtenka resursų sukurti kanalą.
- d) Uždrausta administratoriaus (angl. *Administratively Prohibited*).
- e) Stovinčių duomenų perteklius (angl. *Too Much Outstanding Data*) – jei kurį laiką kanalas buvo nepasiekiamas, jame kaupėsi neišsiųsti duomenys. Kartais geriau jį tiesiog uždaryti, kadangi duomenys buvo persiųsti kitais subkanalais.
- f) Netenkinamas veikimas (angl. *Unacceptable Performance*) – subkanalo veikimo statistika netenkina lyginant su kitais subkanalais.
- g) Tarpinių mazgų trukdžiai (angl. *Middlebox interference*) – tarpiniuose taškuose esančių įrenginių veikimo sukelti pakitimai gadina MPTCP signalizaciją.

3.7. Atsitraukimas

Nors MPTCP protokolas yra suderinamas su beveik visais tinkle esančiais tarpiniais įrenginiais, kartais pasitaiko įrenginių, kurie yra nesuderinami su MPTCP protokolu. Su MPTCP nesuderinamais laikomi įrenginiai, kurie panaikina MPTCP antraštę arba atlieka pakeitimus svarbiuose duomenyse. Tokiu atveju, kai kanale yra toks įrenginys reikia naudoti atsitraukimą (angl. *Fallback*). Atsitraukimas tai perėjimas iš MPTCP į paprastą TCP.

Sudarant pirminį MPTCP sujungimą svarbu, kad visi kanalo įrenginiai būtų suderinami su MPTCP protokolu, kitu atveju nepavyks užmegzti MPTCP sesijos. Jei pagrindiniu keli MPTCP protokolas veikia, o kitame subkanalas ne, jis yra naikinamas arba pereinama prie paprasto TCP sujungimo. Kartais gali pasitaikyti atvejų, tarpiniai įrenginiai leidžia perduoti sinchronizacijos paketus, tačiau siunčiant duomenis atsiranda trukdžiai. Tokiu atveju taip pat reikia naudoti paprastą TCP protokolą.

Kai naudojami keli subkanalai, ir viename jų su perduodamais duomenimis keliauja pašaliniai duomenys, gali pasitaikyti situacija, kai atliekami pakeitimai to kanalo segmentuose. Tokiu atveju neįmanoma nustatyti, kokie pakeitimai buvo atlikti duomenims ir kanalas privalo būti nutrauktas siunčiant *RST* pranešimą su *MP_FAIL* funkcija (28 pav.). Ši funkcija nurodo duomenų sekos numerį, kuriame įvyko klaidų kontrolės klaida.



28 pav. MP_FAIL antraštės struktūra [8]

Gavęs tokį pranešimą imtuvas privalo išmesti visus duomenis nuo nurodyto duomenų sekos numerio. Sugadinti duomenys negali būti patvirtinami atsiunčiant *ACK*, kadangi jie bus pakartojami kitais subkanalais.

3.8. Apibendrinimas

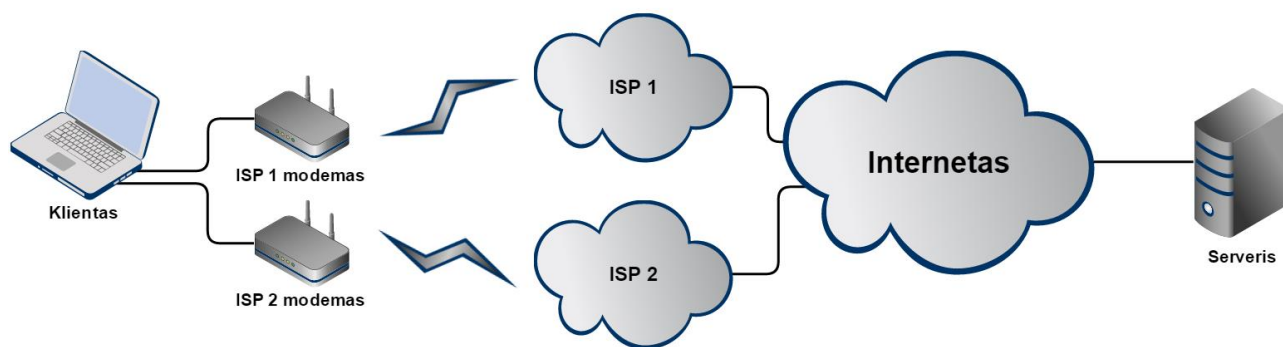
Siekiant geriau suprasti MPTCP protokolo veikimą aptartos pagrindinės jo funkcijos: sujungimo sudarymas, subkanalų sudarymas, subkanalų prijungimas, duomenų perdavimas, adresų apsikeitimas, greitas nutraukimas, subkanalų perkrovimas, atsitraukimas. Aptarti MPTCP protokolo sisteminiai pranešimai ir galimos jų reikšmės. Iš žinučių antraščių matyti kaip MPTCP protokolas veikia su TCP protokolu, kaip nustatomi veikimo parametrai. Taip pat vertėtų atkreipti dėmesį, kad nemažai vietos yra rezervuota naujiems žinučių tipams. Tai reiškia, kad MPTCP protokolas ir toliau gali būti tobulinimas, pritaikomas naujiems sprendimams.

4. EKSPERIMENTŲ STRUKTŪRINĖS SCHEMOS KŪRIMAS

Siekiant išsiaiškinti MPTCP protokolo naudą, matavimus nuspręsta atlikti virtualiame bei realiame tinkle. Šiam tikslui įgyvendinti buvo sudarytos dvi tinklo schemos.

4.1. Realios tinklo struktūrinė schema

Norint atlikti MPTCP protokolo vėlinimo tyrimą bevieliais tinklais buvo sudaryta struktūrinė sistemos schema (29 pav.) ir parinkti tinklo elementai.



29 pav. Struktūrinė realios tinklo schema

Tinklo schemos elementai:

- Klientas – nešiojamas kompiuteris palaikantis MPTCP protokolą.
- ISP (angl. *Internet Service Provider*) modemai - įrenginiai skirti prieigai prie interneto, naudojant du skirtingus interneto paslaugų tiekėjus.
- ISP 1 / ISP 2 – interneto paslaugų tiekėjų tinklai.
- Serveris – tai įrenginys palaikantis MPTCP protokolą.

Prieigai prie interneto paslaugų tiekėjų tinklų pasirinkta naudoti bevielius maršruto parinktuvus „Huawei“ E5186 (30 pav.).

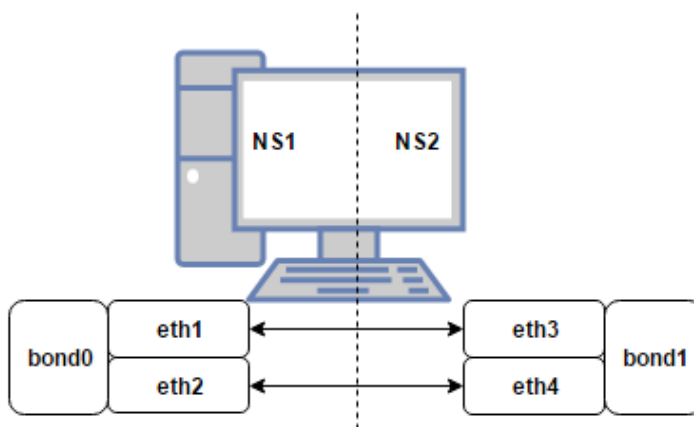


30 pav. „Huawei“ E5186 maršruto parinktuvas [9]

Šie maršruto parinktuvai naudojami dėl plataus mobiliojo korinio ryšio technologijų suderinamumo.

4.2. Virtualaus tinklo struktūrinė schema

Perduodant realiojo laiko srautą labai svarbus tinklo vėlinimas. Vėlinimui į vieną pusę išmatuoti naudojamas vidinis įrenginio laikrodis. Skirtingų įrenginių vidiniai laikrožiai nėra tarpusavyje sinchronizuoti, todėl siekiant kuo tiksliau išsiaiškinti, kokią įtaką MPTCP protokolo veikimui daro vėlinimas, buvo sukurta virtualaus tinklo struktūrinė schema. Siekiant išsiaiškinti MPTCP protokolo naudą taip pat nuspręsta naudoti sąsajų sujungimo metodą (angl. *Bonding*). Šis metodas suteikia galimybę palyginti dviejų skirtingų metodų veikimą tai pačiai užduočiai atlikti [3].



31 pav. Struktūrinė virtualaus tinklo schema

Šioje schemoje naudojamas vienas kompiuteris, kurio tinklas padalintas į du atskirus, vienas nuo kito nepriklausomus tinklus.

Tinklo schemos elementai:

- NS1 / NS2 – vidiniai įrenginio tinklai.
- bond0 / bond1 – virtuali sąsaja, skirta sujungti realioms sąsajoms.
- eth1 / eth2 / eth3 / eth4 – kompiuterio sąsajos.

Kompiuterio sąsajos sujungtos UTP tipo kabeliu.

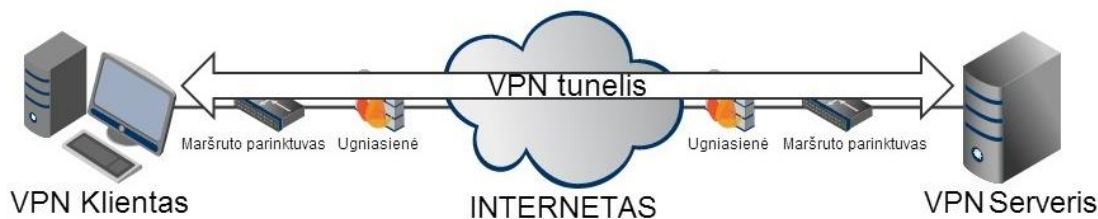
4.3. Naudojamos programos

Tyrimui atlikti naudojami įvairūs protokolai ir programos, kurias naudojant sukuriamas tinklas, atliekami matavimai, keičiami parametrai.

4.3.1. VPN technologija

VPN (angl. *Virtual Private Network*) technologija gali būti įgyvendinta naudojant VPN programą. VPN – virtualusis privatus tinklas skirtas sujungti internete nutolusius taškus į virtualų lokalų tinklą. Duomenys tarp įrenginių siunčiami sukurtu tuneliu (32 pav.). Įrenginiai sujungiami tarsi

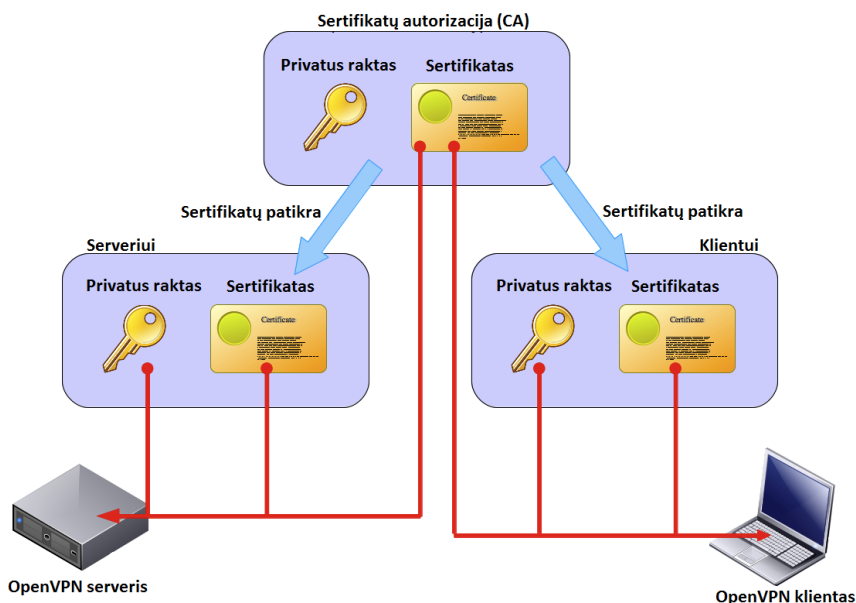
lokaliame tinkle, išlaikant funkcionalumą, valdymą bei saugumą. Tunelis sukuriamas naudojant taškas-taškas (angl. point-to-point) sujungimą [17].



32 pav. VPN technologija

VPN gali veikti vienu iš dviejų režimų: TAP – naudojamas kai perduodami duomenys nesusiję su IP technologija arba norima naudoti IPv6; TUN (angl. *Network Tunnel*) – kai naudojamas maršrutizavimas ir perduodami IP paketais. Duomenys VPN programoje gali būti siunčiami naudojant tiek TCP, tiek UDP protokolą per 1194 prievadą. TCP naudojamas kai svarbu užtikrinti duomenų perdavimą, taip pat TCP retai blokuojamas ugniasienėmis, tačiau sujungimas yra lėtas. UDP naudojamas kai reikalinga kuo didesnė duomenų perdavimo sparta. Kadangi tyrimo metu bus perduodamas realiojo laiko srautas per MPTCP protokolą, galima naudoti tik TCP režimą [16]. Tyrimui atlikti naudojamos *vtun* ir *Openvpn* programos.

Openvpn tunelio sukūrimas pagrįstas simetriniu duomenų kodavimu. Abu tunelio galuose esantys įrenginiai naudoja užkodavimo ir atkodavimo raktus duomenų srautui perduoti. Norint sukurti VPN tunelį reikia įgyvendinti PKI (angl. *Public Key Infrastructure*) struktūrą (33 pav.).



33 pav. *Openvpn* raktų bei sertifikatų veikimo schema [17]

PKI sudaro:

- a) Sertifikatas (viešasis raktas) ir privatusis raktas. Serveris ir kiekvienas klientas privalo turėti tik jiems skirtus raktus.
- b) Sertifikatų autorizacijos sertifikatas (angl. *Certificate Authority Certificate*) ir raktas. Šį sertifikatą privalo turėti serveris ir kiekvienas klientas, o raktą – tik serveris.

Openvpn naudoja abipusę autentifikaciją, paremtą sertifikatais. Tai reiškia, kad klientas prieš siųsdamas informaciją turi būti autentifikuotas serverio ir atvirkščiai. Tik patvirtinus, kad sertifikatai teisingi, sudaromas VPN sujungimas. Toks sertifikatų/raktų modelis turi privalumų:

- a) Serveriui užtenka turėti tik savo sertifikatą/raktą, jam nereikia turėti visų klientų sertifikatų.
- b) Serveris priima tik tuos klientus, kurie buvo patikrinti sertifikatų autorizacijos.
- c) Pats svarbiausias modelio raktas – sertifikatų autorizacijos raktas. Kadangi jis panaudojamas tik vieną kartą, jį galima laikyti visiškai atskiroje sistemoje be jokios tinklo prieigos. Taip užtikrinama, kad niekada nebus pažeistas sertifikatų sistemos saugumas.
- d) Jei kažkuris sistemos privatusis raktas būna pažeistas, galima jį uždrausti nekuriant visos PKI sistemos iš naujo.
- e) Kiekvienam klientui galima suteikti skirtingas prieigų teises.

4.3.2. Kitos programos

Sistemos tyrimams atlikti, perduodant TCP tipo srautą, bus naudojama *Iperf* programinė įranga. Ši programa skirta IP sujungimų testavimui. Pagrindinis parametras, kurį siekiama nustatyti naudojant *Iperf*, yra maksimalus tinklo pralaidumas [14].

UDP srauto perdavimui pasirinkta naudoti *D-ITG* srauto generavimo programą. Ji bus generuoja pastovaus dydžio UDP tipo srautą. Ši programa suteikia galimybę nustatyti norimo srauto intensyvumą, paketų dydį, taip pat gauti informaciją apie perduoto srauto vėlinimą, paketų praradimus, vėlinimo švytavimą ir kitus parametrus [21].

Sąsajų sujungimo metodui įgyvendinti naudojama *Bonding* tinklo sąsajų sujungimo sistema. Sąsajų sujungimas leidžia sujungti dvi ir daugiau tinklo plokščių į vieną sąsają. Taip pat ši programa suteikia galimybę padidinti tinklo patikimumą, pralaidumą, pagerinti apkrovos tinklo plokštės paskirstymą [3].

Fizinėms sąsajoms viename įrenginyje paskirstyti į nepriklausomus vidinius tinklus buvo naudojama *netns* funkcija. Tokiu būdu sąsajos buvo paskirstytos į du skirtingus tinklus (angl. *Namespace*) [20].

4.4. Apibendrinimas

Šiame skyriuje sudarytos ir aptartos sistemos struktūrinės schemos, trumpai aptarti schemų elementai. Taip pat apžvelgta naudojama įranga tyrimui atlikti, pagrįstas įrangos pasirinkimas. Aptarta VPN technologija, kuri bus naudojama tyrimo metu sukurti UDP protokolo sujungimą. Parinkta sistemos parametrų matavimo programa, su kuria bus matuojama perdavimo sparta, vėlinimas, paketų praradimai ir vėlinimo švytavimas, naudojant MPTCP protokolą bevieliuose tinkluose bei virtualiame tinkle.

5. MPTCP PROTOKOLO VEIKIMO TYRIMAS

Sistemos veikimo tyrimas atliekamas parenkant kelis skirtingus scenarijus. Tyrimo metu matuojami šie kokybiniai tinklo parametrai:

- a) perdavimo sparta (angl. *Throughput*);
- b) vėlinimas (angl. *Delay*);
- c) paketų praradimai (angl. *Packet Loss*);
- d) vėlinimo švytavimas (angl. *Jitter*).

Šie parametrai tyrinėjami naudojant skirtingas prieigos prie interneto technologijas bei perdavimo protokolus.

Realioje tinkle tiriamos skirtingų interneto paslaugų tiekėjų technologijų kombinacijos [10]:

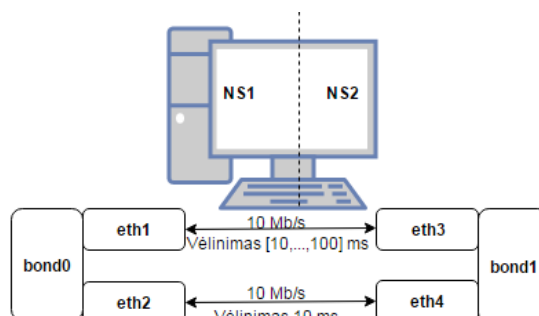
1 lentelė. Tyrimo scenarijai ir naudojamos technologijos.

Bandymo nr.	ISP 1	ISP 2
1	GPRS	GPRS
2	GPRS	UMTS
3	GPRS	LTE
4	UMTS	UMTS
5	UMTS	LTE
6	LTE	LTE

Parametrus tirti naudojami skirtingi transporto lygmens protokolai. TCP protokolas naudojamas tirti perdavimo spartą, o UDP protokolas – vėlinimą, paketų praradimus, vėlinimo švytavimas. UDP protokolui perduoti per MPTCP protokolą sukuriama VPN tunelis tarp serverio ir kliento. VPN tunelio sujungimas sudaromas naudojant TCP protokolą [18].

5.1. Virtualaus tinklo tyrimas

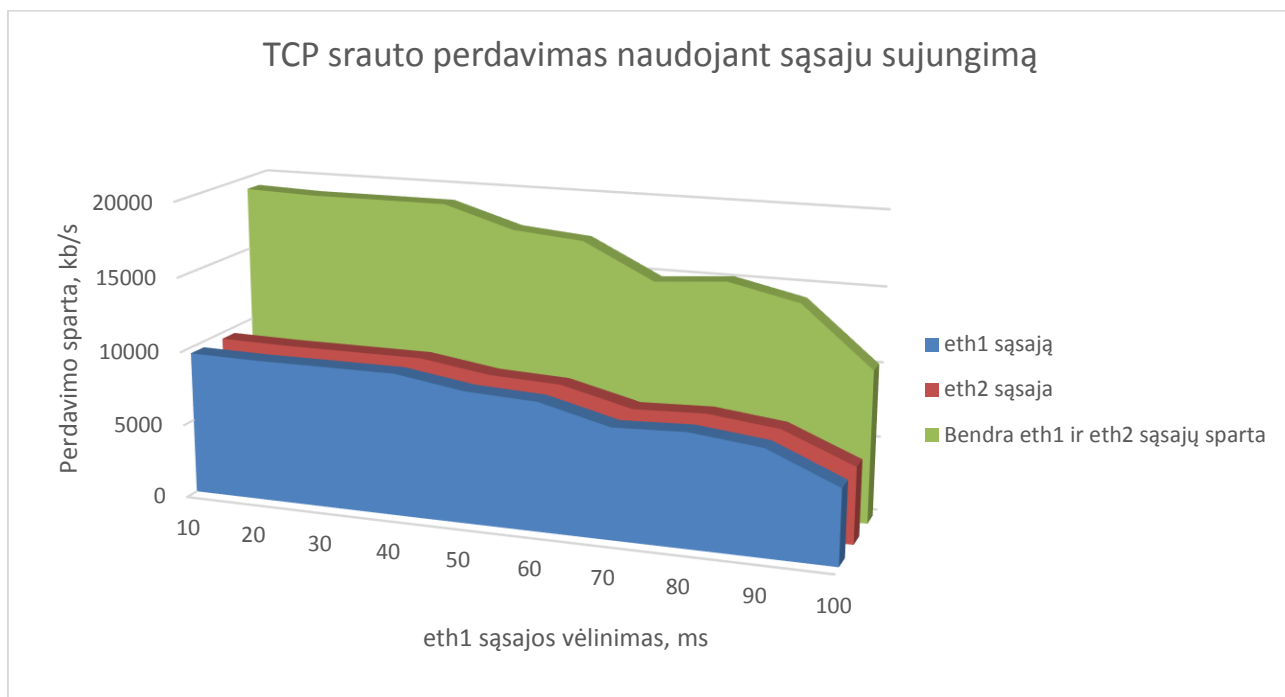
Atliekant matavimus virtualiame tinkle, siekiama tiksliai nustatyti koks yra sistemos vėlinimas į vieną pusę kintant tinklo parametrus. Matavimų metu keičiamas vieno sujungimo vėlinimas ir stebima įtaka visos sistemos veikimui. Matavimai atliekami naudojant MPTCP protokolą ir sąsajų sujungimo metodą. Sujungimo schema ir tinklo parametrai pavaizduoti 34 paveikslėlyje.



34 pav. Virtualaus tinklo schema ir parametrai

5.1.1. TCP srauto perdavimas

Perduodant TCP srautą svarbiausias parametras yra perdavimo sparta. Testas buvo atliekamas naudojant *Iperf* [15]. Perduodant TCP srautą sąsajų sujungimo metodu abu sujungimai išnaudojami vienodai, kadangi duomenys realioms sąsajoms skirstomos vienodo skirstymo (angl. *Round Robin*) būdu. Kintant vieno iš sujungimų vėlinimui pastebimas pralaidumo sumažėjimas (35 pav.)



35 pav. TCP srauto pasiskirstymas naudojant sąsajų sujungimo metodą

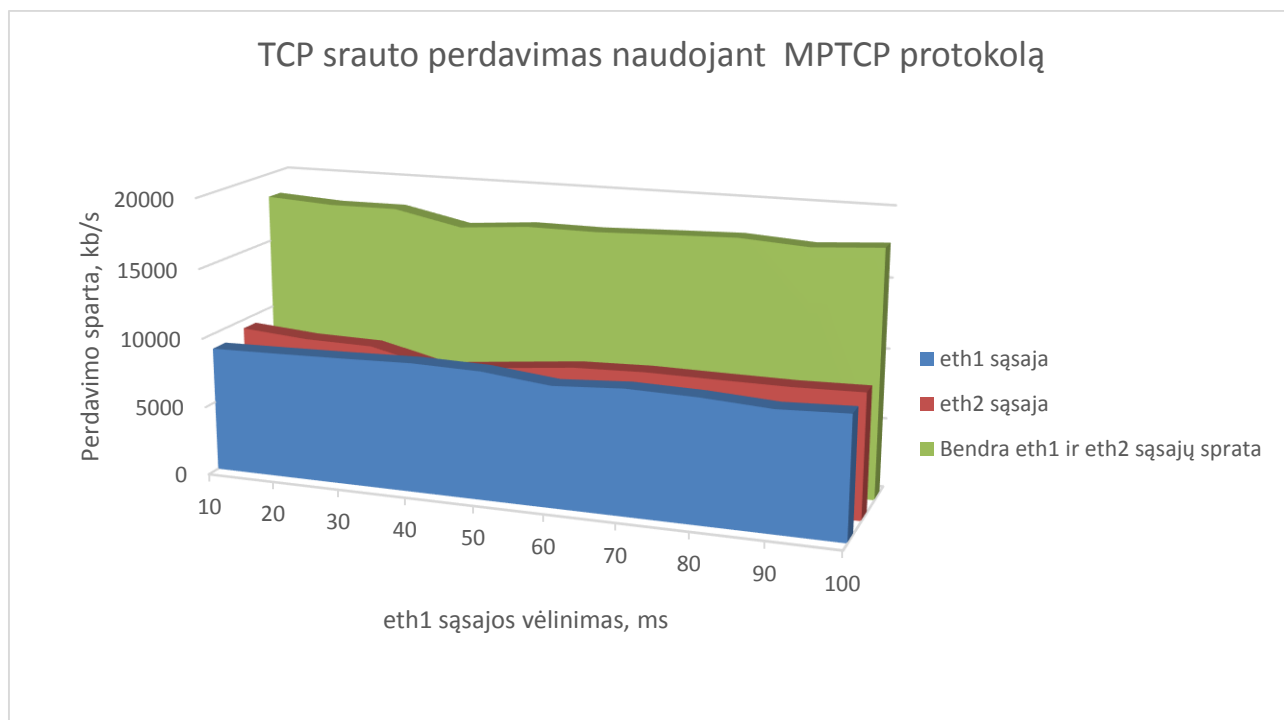
Didėjant vėlinimui viename sujungime, atitinkamai mažėja perdavimo sparta abiejuose sujungimuose. Taip yra todėl, kad TCP protokolas yra labai jautrus vėlinimui. Kad išvengtų paketų praradimų, protokolas mažina perduodamų duomenų spartą abejomis sąsajomis vienodai.

2 lentelė. TCP srauto pasiskirstymas naudojant sąsajų sujungimo metodą.

Vėlinimas eth1, ms	Sparta eth1 sąsaja, kb/s	Sparta eth2 sąsaja, kb/s	Bendra srauto sparta, kb/s
10	9602,23	9576,88	19179,11
20	9524,97	9497,26	19022,23
30	9534,22	9494,88	19029,10
40	9535,69	9490,02	19025,71
50	8821,36	8790,57	17611,94
60	8603,81	8592,56	17196,37
70	7417,76	7432,90	14850,66
80	7611,44	7604,84	15216,28
90	7103,21	7074,09	14177,30
100	5087,45	5117,26	10204,70

Iš 2 lentelės matome, kad vėlinimui viename sujungime pasiekus 100 ms, bendra perdavimo sparta krinta beveik per pusę.

Analogiškas matavimas atliekamas naudojant MPTCP protokolą (36 pav.).



36 pav. TCP srauto pasiskirstymas naudojant MPTCP protokolą

Šiuo atveju vėlinimo didėjimas viename iš sujungimų didelės įtakos perdavimo spartai neturi. Apkrova vienodai padalinama tarp sąsajų ir perdavimo sparta nukenčia daug mažiau. Taip yra todėl, kad MPTCP protokolą sugeba geriau suvaldyti skirtingų vėlinimų sujungimus, išlaikant aukštą perdavimo spartą.

3 lentelė. TCP srauto pasiskirstymas naudojant MPTCP protokolą.

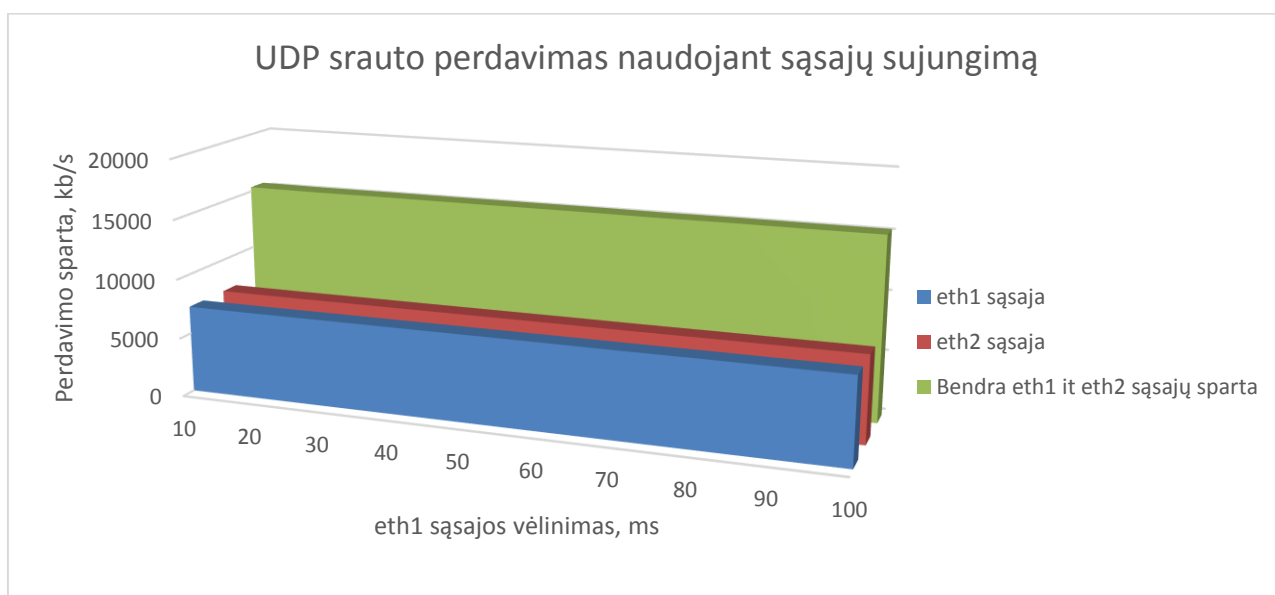
Vėlinimas eth1, ms	Sparta eth1 sąsaja, kb/s	Sparta eth2 sąsaja, kb/s	Bendra srauto sparta kb/s
10	8,947	9,363	18,309
20	9,032	8,985	18,017
30	9,138	8,897	18,034
40	9,273	7,750	17,023
50	9,165	8,243	17,408
60	8,642	8,731	17,373
70	8,942	8,882	17,824
80	8,835	8,837	17,672
90	8,550	8,819	17,369
100	8,760	8,937	17,696

Palyginus gautus rezultatus abiem atvejais aiškiai matomas MPTCP protokolo pranašumas prieš sąsajų sujungimo metodą, kadangi didėjantis vėlinimas vienoje iš sąsajų geriau suvaldomas būtent MPTCP protokolu.

5.1.2. UDP srauto perdavimas

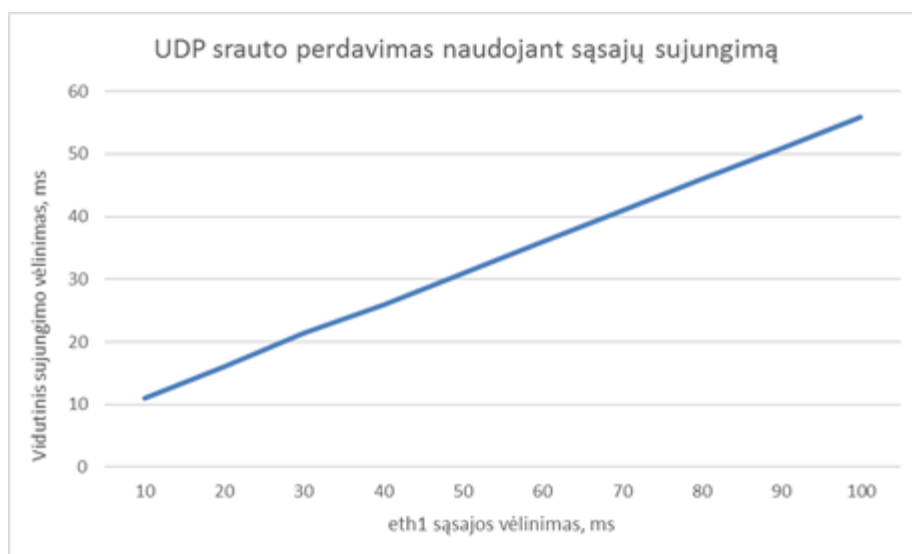
Perduodant realiojo laiko srautą vien aukštos perdavimo spartos neužtenka. Taip pat labai svarbus vėlinimas, vėlinimo švytavimas, paketų praradimai. Atliekant matavimą buvo perduodamas pastovus 15 Mb/s UDP srautas.

Sąsajų sujungimo metodu srautas tarp sujungimų pasiskirsto vienodai, nepriklausomai nuo vėlinimo (37 pav.).



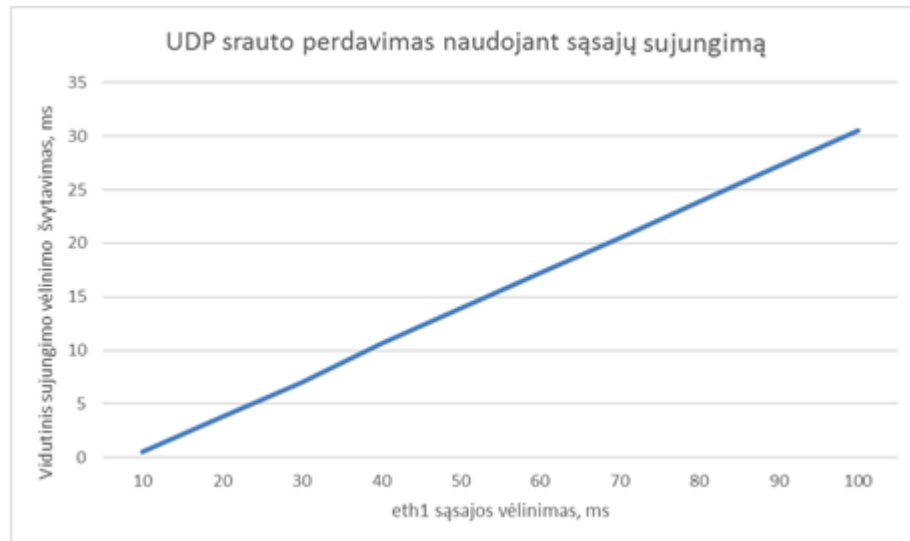
37 pav. UDP srauto pasiskirstymas naudojant sąsajų sujungimo metodą

Vidutinis sistemos vėlinimas, kintant vienos sąsajos vėlinimui, auga tiesiškai (38 pav.).



38 pav. Vidutinis UDP srauto vėlinimas naudojant sąsajų sujungimo metodą

Vėlinimo švytavimas perduodant UDP srautą taip pat auga tiesiškai (39 pav.).



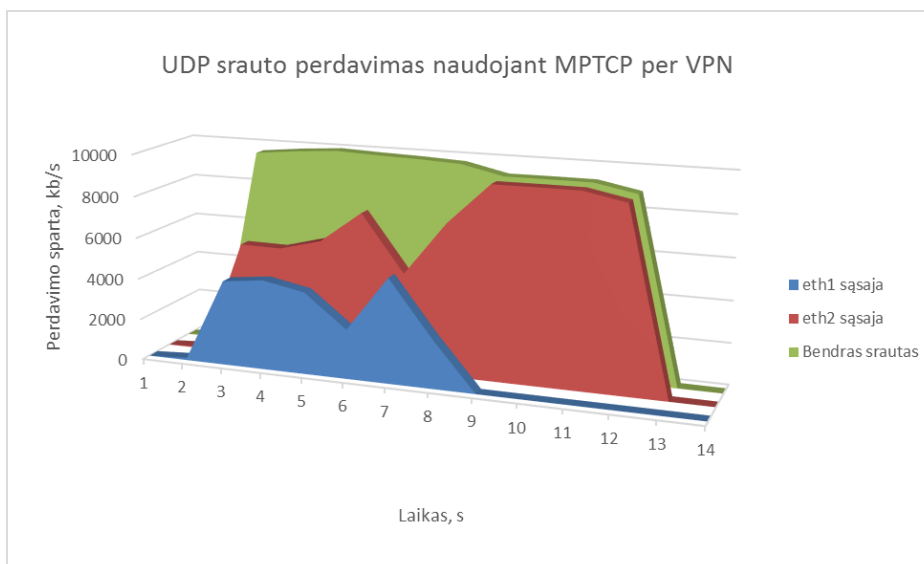
39 pav. Vidutinis UDP srauto vėlinimo švytavimas naudojant sąsajų sujungimo metodą

Šiuo atveju vidutinis vėlinimo švytavimas yra gana didelis dėl srauto skirstymo į sąsajas. Kaip jau minėta anksčiau, sąsajų sujungimo metodas veikia vienodo padalijimo tarp sąsajų režimu.

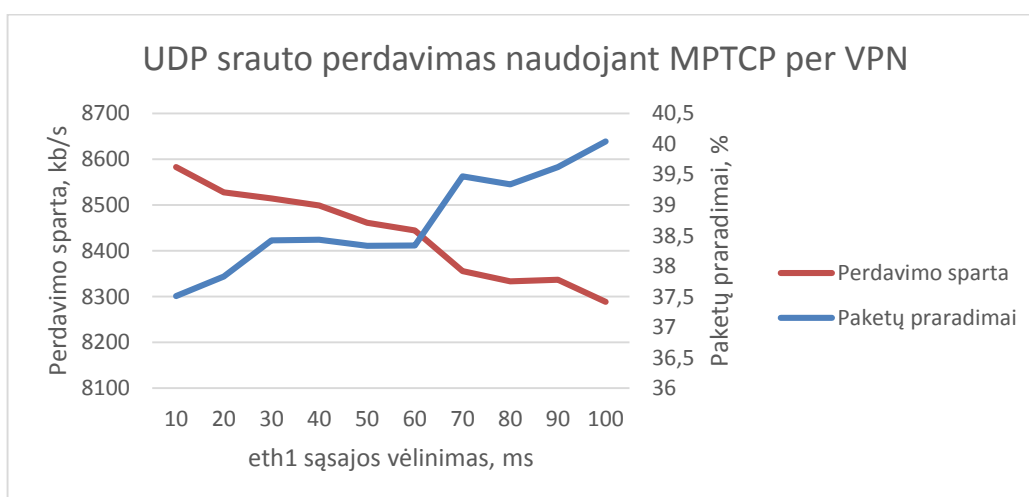
Taip pat dėl šios priežasties atsiranda didžiulis paketų praradimas, kadangi pusė paketų gaunami pavėluotai ir nebėra tinkami naudoti. Paketų praradimai atsiranda jei abiejose sujungimuose vėlinimas yra nevienodas. Esant nevienodam sujungimų vėlinimui prarandama net 50% paketų.

Kitokia situacija UDP srautą perduodant MPTCP protokolu. Norint siųsti UDP srautą visų pirmą buvo sukurtas VPN tunelis tarp serverio ir kliento. VPN tunelis veikė TCP protokolo pagrindu, todėl atsiranda galimybė išnaudoti MPTCP protokolą realiojo laiko duomenims perduoti. Realiojo laiko UDP srautas inkapsuliuojamas į TCP paketus, todėl gali būti siunčiamas MPTCP protokolu.

Perduodant duomenis MPTCP protokolu ir *Openvpn* programos pagalba sukurtu VPN tuneliu, srautas tarp sąsajų pasiskirsto nevienodai (40 pav.). Pastebėta, kad siuntimo pradžioje duomenys siunčiami per abi sąsajas, o po kurio laiko visas srautas siunčiamas tik per vieną sąsają, kurios vėlinimas yra mažiausias. Analogiška situacija buvo stebima ir [25] tyrimo metu. Jei vėlinimas abejomis sąsajomis yra vienodas srautas vis tiek siunčiamas tik per vieną sąsają. Taip pat pastebėta, kad perduodant pastovų 15 Mb/s srautą, nepriklausomai kiek sąsajų naudojama, perdavimo sparta neviršija vienos iš sąsajų maksimalios perdavimo spartos – šiuo atveju 10 Mb/s. Dėl to perduodant didesnį nei 10 Mb/s srautą neišvengiama paketų praradimų. Perduodant 15 Mb/s srautą praradimai siekė net 40% (41 pav.).

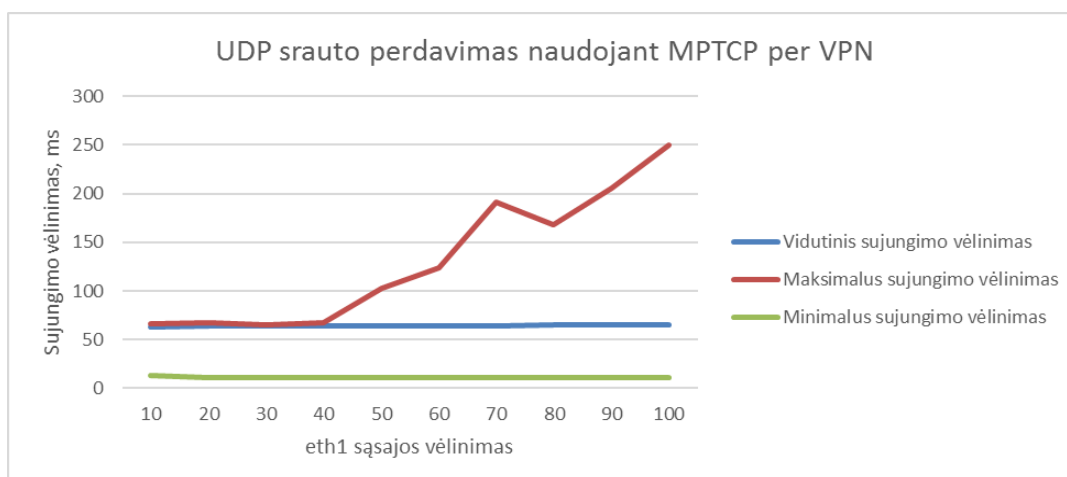


40 pav. UDP srauto pasiskirstymas naudojant MPTCP protokolą su *Openvpn*



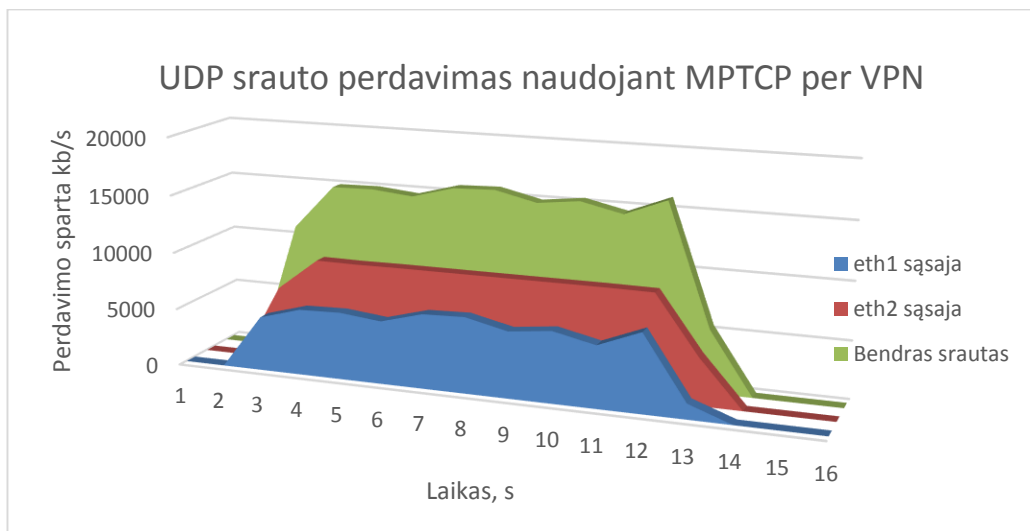
41 pav. UDP srauto perdavimo sparta ir paketų praradimai naudojant MPTCP protokolą su *Openvpn*

Taip pat buvo stebimas ir vėlinimas (42 pav.). Nors vidutinis sistemos vėlinimas buvo gana pastovus, didėjant vienos iš sąsajų vėlinimui, didėja ir maksimalus sistemos vėlinimas.



42 pav. UDP srauto vėlinimas į vieną pusę naudojant MPTCP protokolą su *Openvpn*

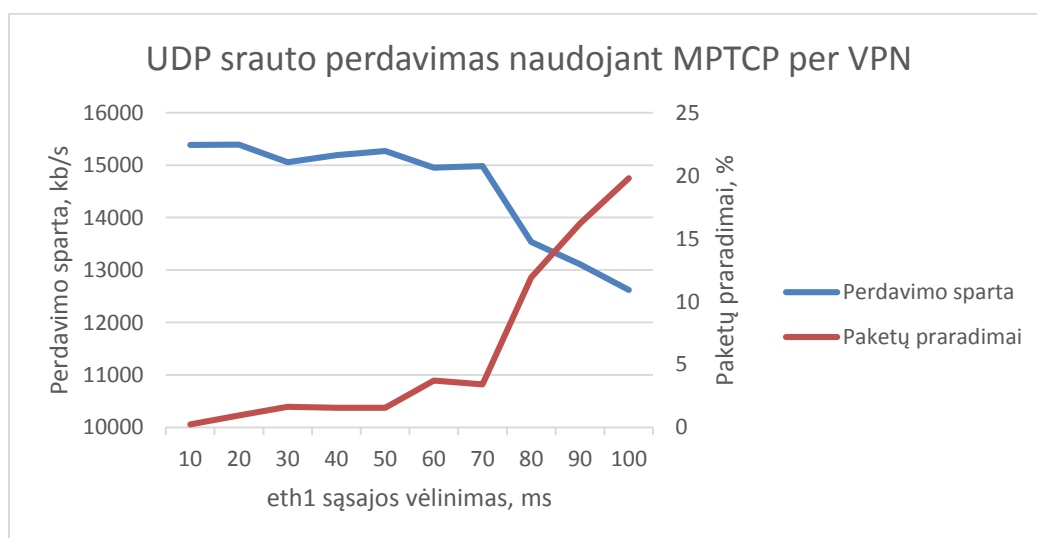
Siekiant išsiaiškinti, ar visais atvejais UDP srautas perduodamas tik per vieną sąsają, pabandyta analogišką eksperimentą atlikti su kita VPN programa. Pasirinkta naudoti *vtun* programą. Kaip ir prieš tai, VPN tuneliu, buvo perduodamas pastovus 15 Mb/s UDP tipo srautas. Atlikus tyrimą paaiškėjo, kad su *vtun* programa vienu metu išnaudojamos abi sąajos, o bendra perdavimo sparta yra didesnė nei kurios nors iš sąsajų atskirai (43 pav.). Šiuo atveju abiejų sąsajų pralaidumas 10 Mb/s, vėlinimas 10 ms.



43 pav. UDP srauto pasiskirstymas naudojant MPTCP protokolą su *vtun*

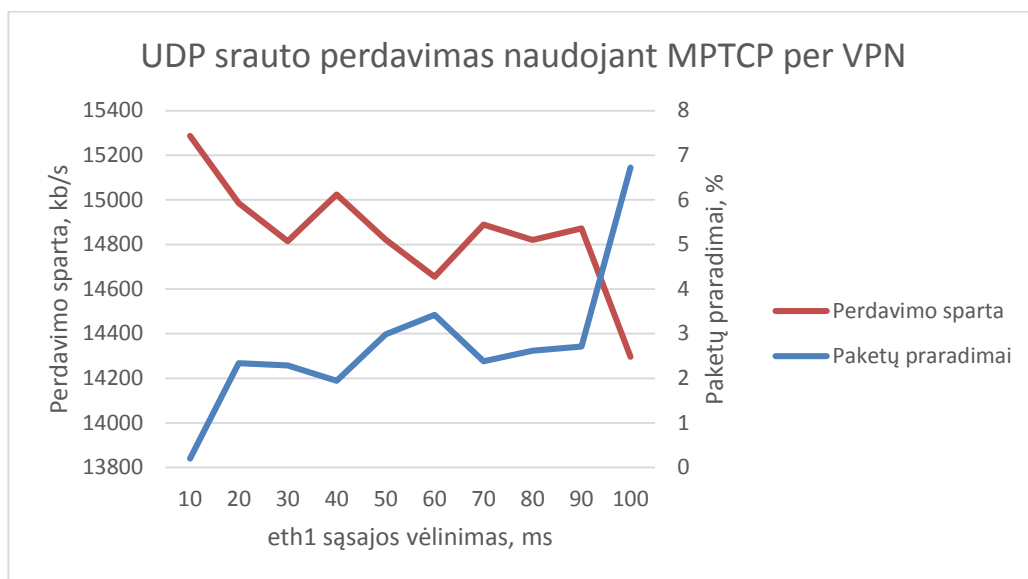
Iš 43 pav. matyti, kad viena sąsaja (*eth2*) yra išnaudojama pilnai, o kita sąsaja perduodami duomenys, kurie viršija *eth2* sąsajos pralaidumą.

Naudojant *vtun* programą, ženkliai sumažėjo paketų praradimas, tačiau didėjant vienos iš sąsajų vėlinimui, praradimų procentas auga, o bendra perdavimo sparta taip pat mažėja (44 pav.). Kuo didesnis vėlinimo skirtumas tarp naudojamų sąsajų, tuo didesni praradimai bei perdavimo sparta. 44 pav. pavaizduoti rezultatai kai *eth0* sąsajos vėlinimas 10 ms.



44 pav. UDP srauto perdavimo sparta ir paketų praradimai naudojant MPTCP protokolą su *vtun*, *eth0* vėlinimas 10 ms

Kai *eth0* sąsajos vėlinimas yra 50 ms, vėlinimo skirtumas tarp sąsajų nebe toks didelis, todėl praradimai mažesni (45 pav.).



45 pav. UDP srauto perdavimo sparta ir paketų praradimai naudojant MPTCP protokolą su *vtun*, *eth0* vėlinimas 50 ms

Kadangi paketų praradimas siunčiant realiojo laiko srautą yra labai svarbus, pabandyta išsiaiškinti kokios praradimų priežastys. Pastebėta, kad paketų praradimas siunčiant realiojo laiko duomenis yra nepastovus ir paketai prarandami tik tam tikrais laiko momentais, o ne viso duomenų perdavimo metu (46 pav.).

Interval	Transfer	Bandwidth	Jitter	Lost/Total Datagrams
0.00-1.00	sec 1.39 MBytes	11.7 Mbits/sec	9.337 ms	0/178 (0%)
1.00-2.00	sec 1.74 MBytes	14.6 Mbits/sec	9.074 ms	0/223 (0%)
2.00-3.00	sec 1.75 MBytes	14.7 Mbits/sec	9.445 ms	0/224 (0%)
3.00-4.00	sec 1.41 MBytes	11.9 Mbits/sec	12.942 ms	0/181 (0%)
4.00-5.00	sec 1.65 MBytes	13.8 Mbits/sec	11.249 ms	0/211 (0%)
5.00-6.00	sec 1.65 MBytes	13.8 Mbits/sec	3.594 ms	0/211 (0%)
6.00-7.00	sec 1.62 MBytes	13.6 Mbits/sec	8.691 ms	50/258 (19%)
7.00-8.00	sec 2.12 MBytes	17.8 Mbits/sec	8.598 ms	0/272 (0%)
8.00-9.00	sec 1.43 MBytes	12.0 Mbits/sec	9.590 ms	0/183 (0%)
9.00-10.00	sec 2.12 MBytes	17.8 Mbits/sec	8.437 ms	0/272 (0%)
10.00-11.00	sec 1.46 MBytes	12.3 Mbits/sec	9.424 ms	0/187 (0%)
11.00-12.00	sec 2.06 MBytes	17.3 Mbits/sec	11.339 ms	0/264 (0%)
12.00-13.00	sec 1.87 MBytes	15.7 Mbits/sec	8.930 ms	0/239 (0%)
13.00-14.00	sec 1.57 MBytes	13.2 Mbits/sec	10.394 ms	0/201 (0%)
14.00-15.00	sec 2.08 MBytes	17.4 Mbits/sec	9.118 ms	0/266 (0%)
15.00-16.00	sec 1.42 MBytes	11.9 Mbits/sec	13.064 ms	0/182 (0%)
16.00-17.00	sec 2.02 MBytes	16.9 Mbits/sec	10.803 ms	0/258 (0%)
17.00-18.00	sec 1.64 MBytes	13.8 Mbits/sec	9.597 ms	0/210 (0%)
18.00-19.00	sec 2.00 MBytes	16.8 Mbits/sec	9.131 ms	0/256 (0%)
19.00-20.00	sec 1.60 MBytes	13.4 Mbits/sec	9.296 ms	0/205 (0%)
20.00-21.00	sec 1.91 MBytes	16.0 Mbits/sec	11.234 ms	0/244 (0%)
21.00-22.00	sec 1.93 MBytes	16.2 Mbits/sec	11.560 ms	0/247 (0%)
22.00-23.00	sec 1.65 MBytes	13.8 Mbits/sec	7.590 ms	0/211 (0%)
23.00-24.00	sec 1.92 MBytes	16.1 Mbits/sec	11.859 ms	0/246 (0%)
24.00-25.00	sec 1.61 MBytes	13.5 Mbits/sec	11.845 ms	0/206 (0%)
25.00-26.00	sec 1.90 MBytes	15.9 Mbits/sec	9.313 ms	0/243 (0%)
26.00-27.00	sec 1.94 MBytes	16.3 Mbits/sec	8.669 ms	0/248 (0%)
27.00-28.00	sec 1.64 MBytes	13.8 Mbits/sec	4.415 ms	0/210 (0%)
28.00-29.00	sec 1.94 MBytes	16.3 Mbits/sec	8.701 ms	0/248 (0%)
29.00-30.00	sec 1.59 MBytes	13.4 Mbits/sec	7.968 ms	0/204 (0%)

Interval	Transfer	Bandwidth	Jitter	Lost/Total Datagrams
0.00-30.00	sec 53.6 MBytes	15.0 Mbits/sec	3.494 ms	50/6867 (0.73%)
Sent 6867 datagrams				

46 pav. UDP srauto perdavimas naudojant MPTCP protokolą su *vtun*

Iš 46 pav. pateiktų rezultatų matyti, kad siunčiant pastovų 15 Mb/s UDP srautą 19 % praradimų atsiranda tik 6-7 perdavimo sekundėmis. Visais kitais laiko momentais duomenys siunčiami be praradimų.

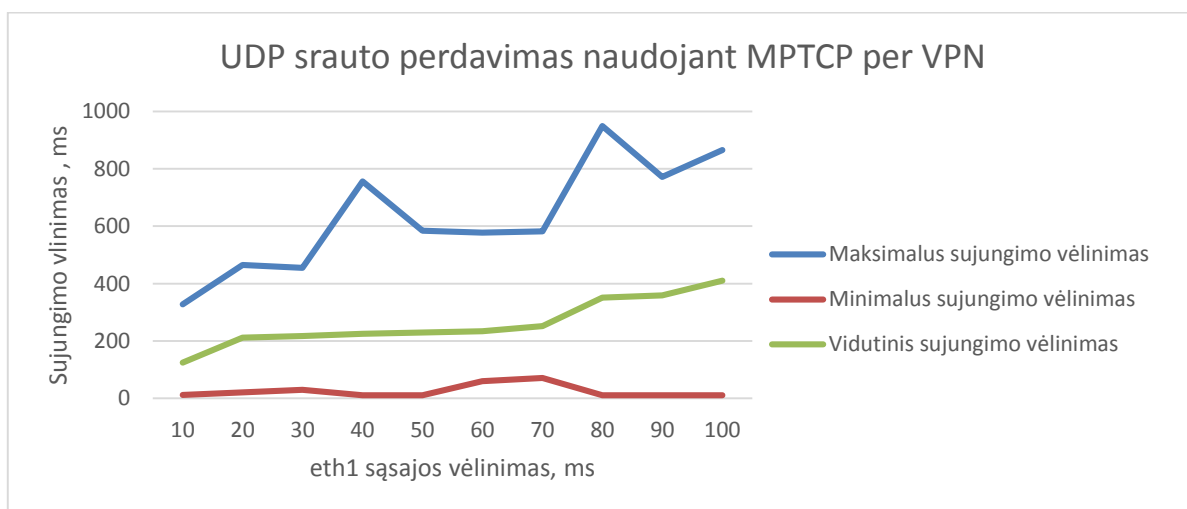
Taip pat pastebėta, kad paketų praradimai atsiranda tik siuntėjo pusėje, VPN tunelio sąsajoje (47 pav.). Todėl galima daryti išvadą, kad tam tikrai laiko momentais, kai nebeužtenka sujungimo pralaidumo duomenims siųsti, duomenys kaupiasi VPN sąsajos buferyje, o užsipildžius buferiui yra išmetami.

```
tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1300
    inet 10.10.1.1 netmask 255.255.255.255 destination 10.10.1.4
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500 (UNSPEC)
    RX packets 3059267 bytes 1989723549 (1.8 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 724595 bytes 867264605 (827.0 MiB)
    TX errors 0 dropped 4278 overruns 0 carrier 0 collisions 0
```

47 pav. UDP srauto paketų praradimas naudojant MPTCP protokolą su *vtun*

Pabandžius perduoti mažesnę nei 15 Mb/s duomenų srautą praradimų nebelieka. Atitinkamai perduodant didesnę nei duomenų srautą – praradimų procentas didėja. Jei 15 Mb/s siunčiamas 100 Mb/s pralaidumo sąsajomis praradimų taip pat nėra.

Dar vienas labai svarus sujungimo parametras yra vėlinimas. Iš 48 pav. matyti, kad didėjant vienos iš sąsajų vėlinimui, bendras sujungimo vėlinimas į vieną pusę taip pat didėja.



48 pav. UDP srauto vėlinimas į vieną pusę naudojant MPTCP protokolą su *vtun*

Apžvelgus realiojo laiko srauto perdavimą abiem būdais galima daryti išvadą, kad sąsajų sujungimo metodas yra tinkamas tik tada, kai visų sąsajų vėlinimas yra identiškas, o tai pasiekti labai sudėtinga. Perduodant duomenis MPTCP protokolu per VPN kai naudojama *Openvpn* programa, maksimali sujungimo sparta išnaudojama neefektyviai, o siunčiant didesnę duomenų srautą nei maksimali sujungimo sparta, atsiranda dideli paketų praradimai. Naudojant *vtun* programą VPN tuneliui sudaryti, perdavimo sparta išnaudojama efektyviai, esant mažam vėlinimui praradimų nėra.

arba jie labai maži. Didėjant vėlinimui perdavimo sparta mažėja ir atsiranda didesni paketų praradimai. Paketai prarandami dėl kanalo pralaidumo trūkumo, todėl paketai kaupiasi VPN sąsajų buferiuose, o jiems persipildžius – yra išmetami.

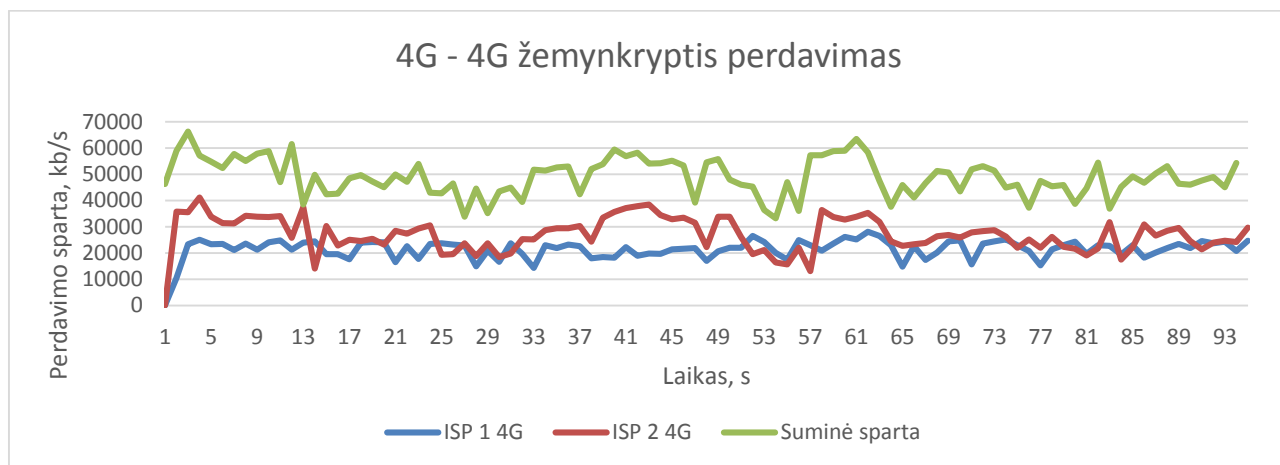
Iš gautų rezultatų matyti, kad norit perduoti realiojo laiko srautą labai svarbu pasirinkti tinkamą daugiatainkliškumo sprendimą, išsiaiškinti visų naudojamų programų veikimo principus ir įtaką bendram sistemos veikimui.

5.2. Realaus tinklo tyrimas

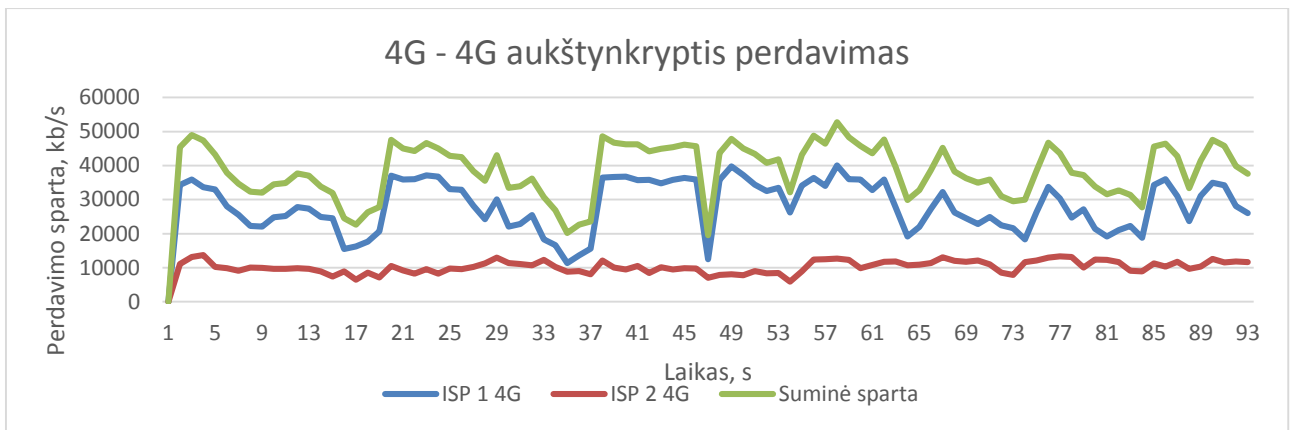
Taip pat šiame darbe atliktas MPTCP protokolo tyrimas realiame tinkle, naudojant prieigą prie interneto, bevielio korinio mobiliojo ryšio tinklais. Tyrimui atlikti pasirinkti du skirtingi mobiliojo ryšio operatoriai, taip siekiant išvengti įrenginių prisijungimo prie tos pačios bazinės stoties atvejų. Jei būtų naudojami vieno operatoriaus tinklai, atsirastų galimybė nepilnai išnaudoti maksimalią perdavimo spartą, kadangi įrenginiai gali prisijungti prie tos pačios stoties, o tokiu atveju tektų dalintis bazinės stoties resursus.

5.2.1. TCP srauto perdavimas

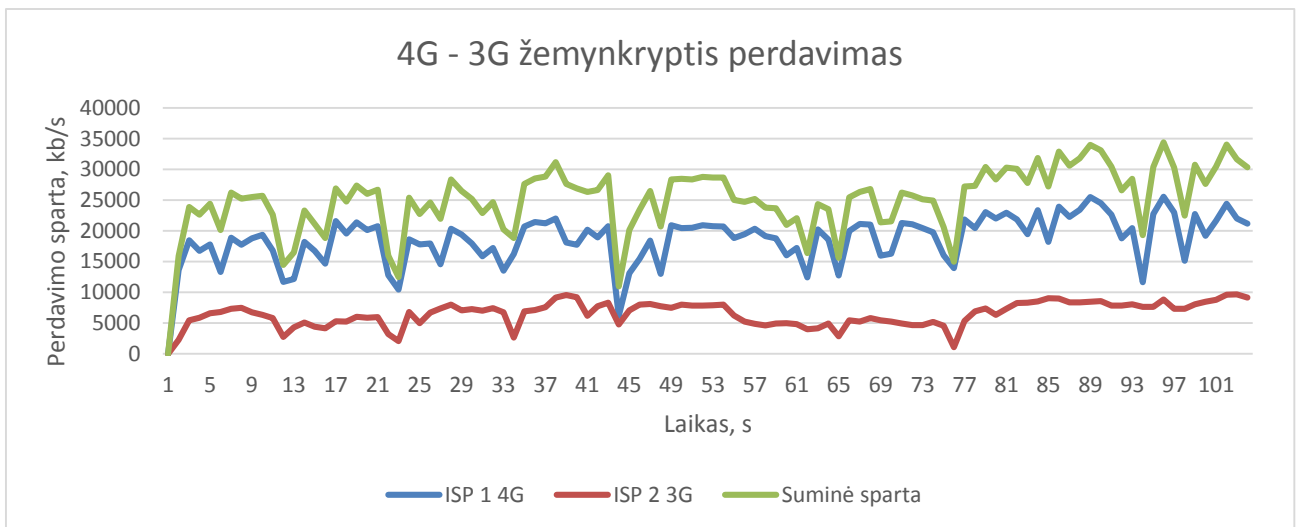
Perduodant TCP srautą buvo sujungiamos 4G, 3G, 2G technologijos. Iš 49-56 paveikslėlių matyti, kad MPTCP protokolas puikiai išnaudoja abi prieigas prie interneto ir leidžia pasiekti didesnę spartą nei naudojant kurią nors technologiją atskirai.



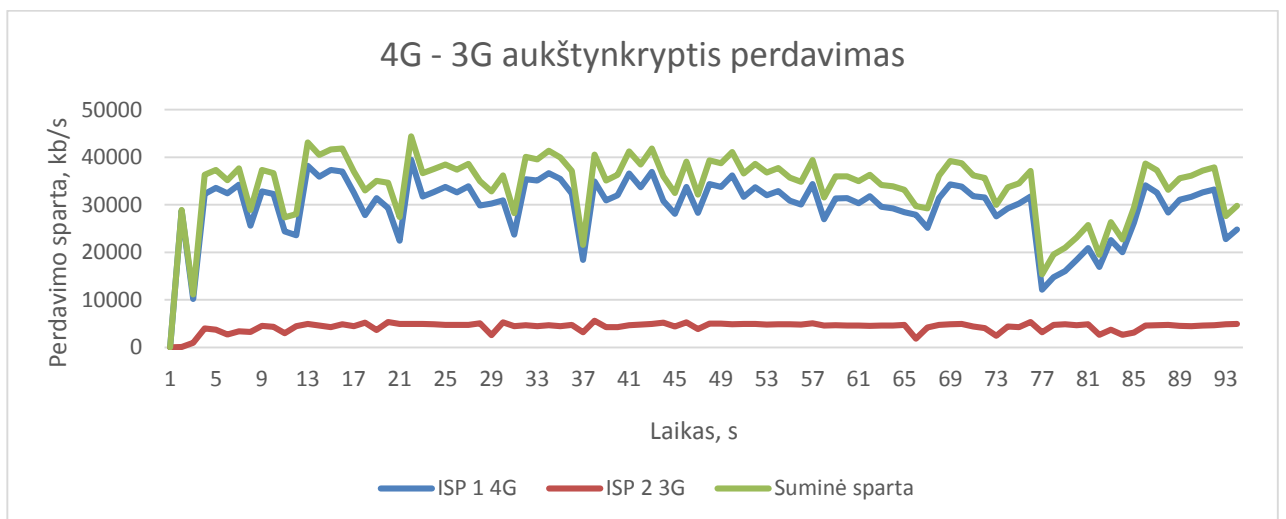
49 pav. 4G technologijų sujungimas naudojant MPTCP protokolą



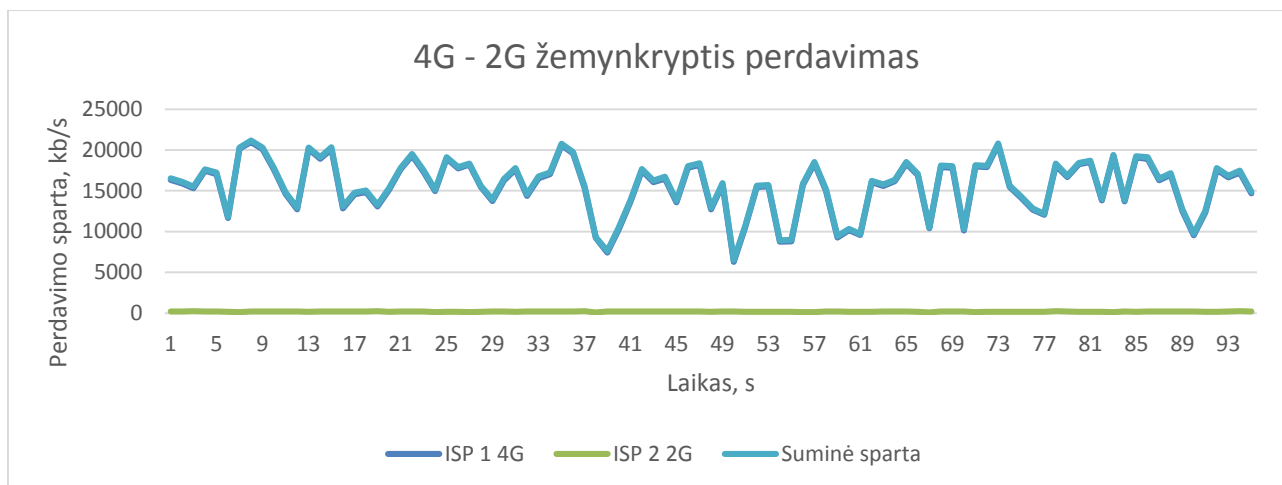
50 pav. 4G technologijų sujungimas naudojant MPTCP protokolą



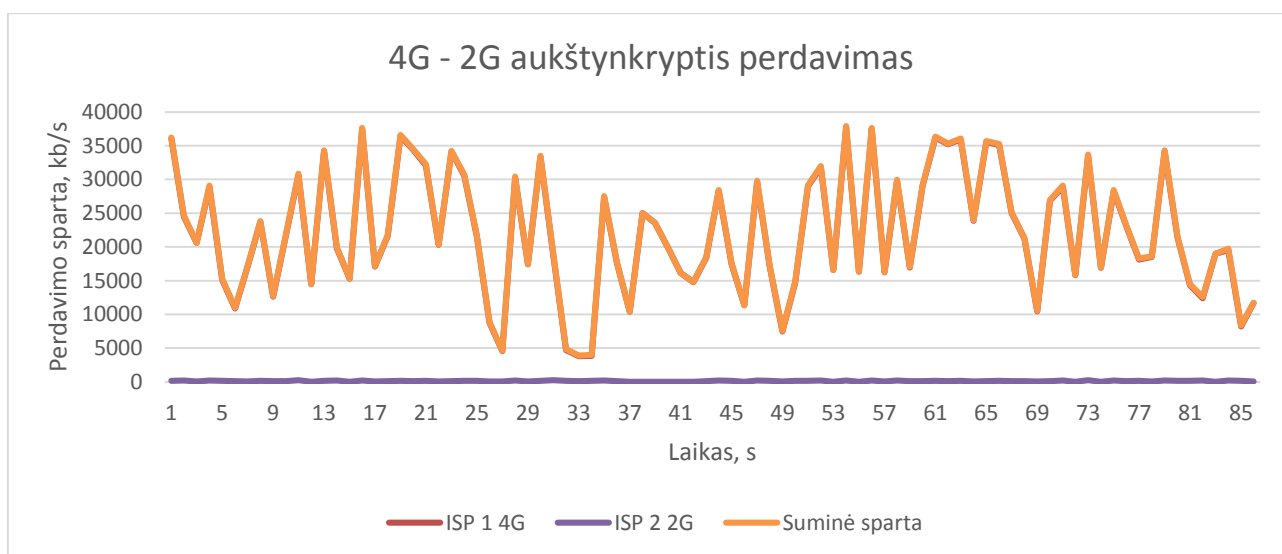
51 pav. 4G/3G technologijų sujungimas naudojant MPTCP protokolą



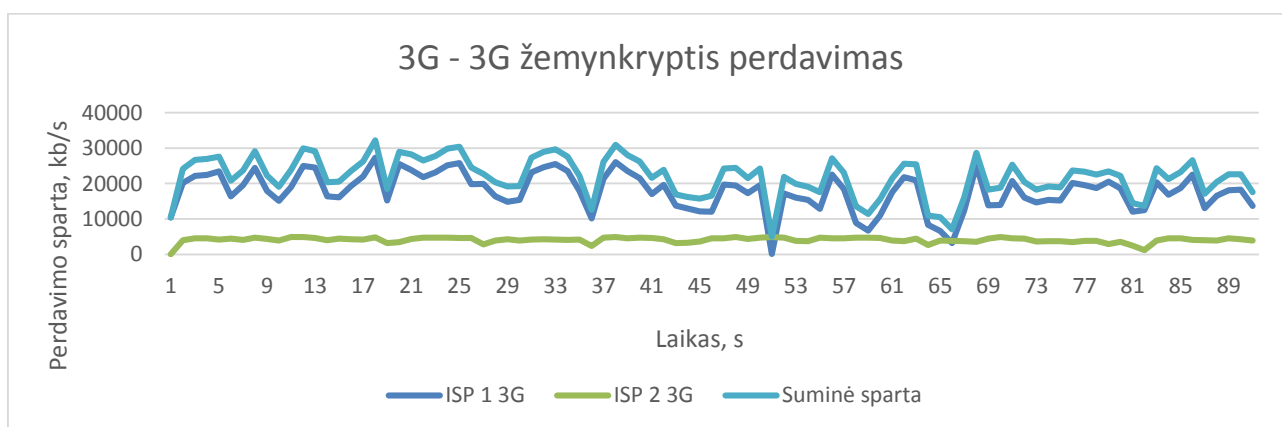
52 pav. 4G/3G technologijų sujungimas naudojant MPTCP protokolą



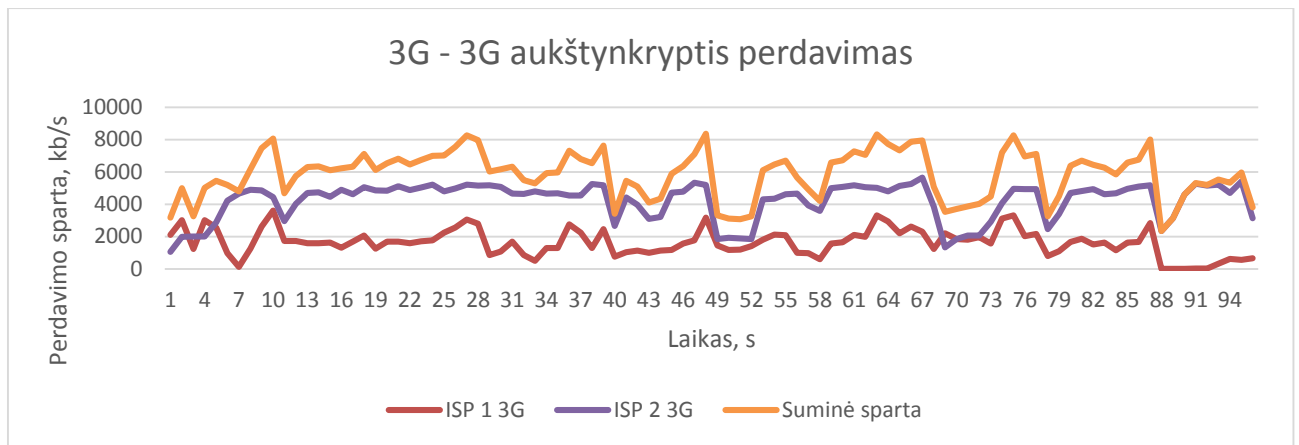
53 pav. 4G/2G technologijų sujungimas naudojant MPTCP protokolą



54 pav. 4G/2G technologijų sujungimas naudojant MPTCP protokolą



55 pav. 3G technologijų sujungimas naudojant MPTCP protokolą



56 pav. 3G technologijų sujungimas naudojant MPTCP protokolą

Iš 54 pav. matyti, kad naudojant skirtingo pralaidumo technologijas, kurių vėlinimas labai skiriasi, maksimali perdavimo sparta smarkiai kinta. Šiuo atveju reikėtų pasvarstyti ar 4G ir 2G technologijų sujungimas duoda daugiau naudos nei naudojant tik 4G technologiją duomenis perduoti.

5.2.2. UDP srauto perdavimas

Realiam tinkle buvo siunčiamas įvairaus dydžio pastovus UDP srautas. MPTCP protokolą įgyvendintas, panaudojant TCP protokolo pagrindą veikiantį VPN tunelį. Taip pat buvo naudojamos įvairios 4G, 3G, 2G technologijų kombinacijos.

4 lentelė. UDP srauto rezultatai 4G/4G tinkluose naudojant MPTCP protokolą su *Openvpn*.

Perduodamų duomenų srautas	Vėlinimo švytavimas	Paketų praradimas
512 Kbits/sec	5.589 ms	0/ 1308 (0%)
1.03 Mbits/sec	5.448 ms	0/ 2614 (0%)
2.05 Mbits/sec	4.888 ms	0/ 5225 (0%)
5.13 Mbits/sec	2.857 ms	0/13068 (0%)
8.52 Mbits/sec	2.349 ms	4324/26133 (17%)

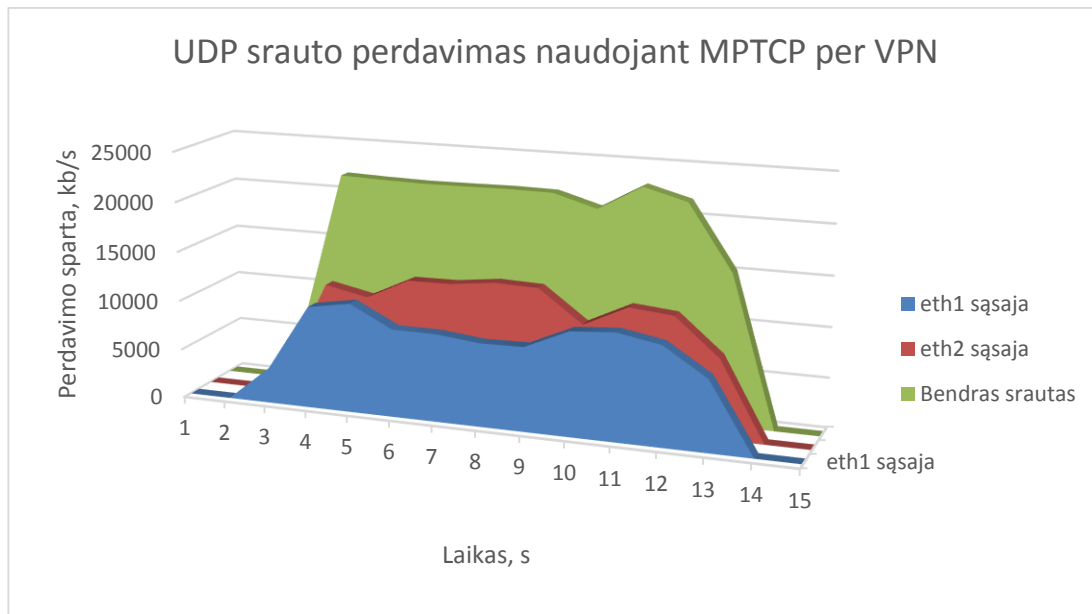
Iš 4 lentelės matyti, kad siunčiant didesnę duomenų srautą, atsiranda paketų praradimai. Tai patvirtina ir virtualaus tinklo tyrime su *Openvpn* programa gauti rezultatai.

5 lentelė. UDP srauto rezultatai 4G/4G tinkluose naudojant MPTCP protokolą su *vtun*.

Perduodamų duomenų srautas	Vėlinimo švytavimas	Paketų praradimas
1.03 Mbits/sec	8.788 ms	0%
5.13 Mbits/sec	4.822 ms	0%
10.2 Mbits/sec	4.677 ms	0%
20.5 Mbits/sec	9.701 ms	0.58%

Atlikus tokius pat matavimus naudojant *vtun* programą pastebėta, kad maksimali perdavimo sparta yra didesnė, o paketų praradimas esant dideliui srautui yra nedidelis (5 lentelė). Taip pat

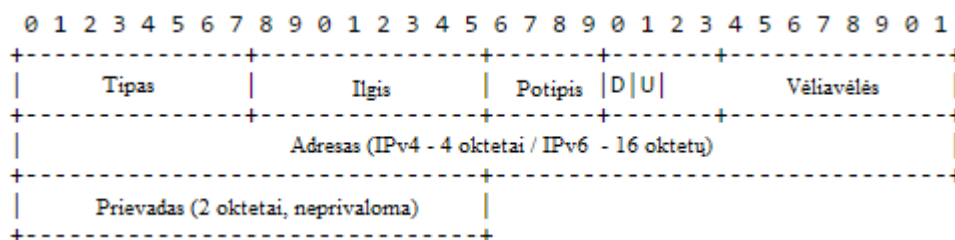
pastebėta, kad kanalų pralaidumas išnaudojamas daug efektyviau, kadangi duomenys perduodami abiem kanais vienu metu. Pavyzdžiui 57 pav. pavaizduotas 20 Mb/s UDP srautas, perduodamas 4G/4G tinklais.



57 pav. UDP srauto perdavimas 4G/4G technologijomis realiame tinkle

Iš realiame tinkle atlikto tyrimo matyti, kad MPTCP protokolas puikiai tinka TCP srautui perduoti, agreguojant 4G,3G,2G technologijas. Perduodant realiojo laiko srautą, reikia atkreipti dėmesį į naudojamos VPN programos galimybes veikti kartu su MPTCP protokolu. Siunčiant didesnio kiekio srautą atsiranda paketų praradimai, todėl reikia įvertinti naudojamų kanalų pralaidumą ir perduodamų duomenų kiekį. Dar vienas privalumas perduodant duomenis MPTCP protokolu yra patikimumas. Jei staiga sutriktų vieno iš paslaugų tiekėjų ryšys, duomenys būtų perduodami kito paslaugų tiekėjo tinklu.

2015 metų gruodžio mėnesio IETF (angl. *Internet Engineering Task Force*) MPTCP specifikacijos priede užsimenama apie naują protokolo režimą ir antraštės potipį (58 pav.) [27]. Tai paprasto transportavimo režimas, leidžiantis geriau suderinti visiškai skirtingų technologijų sujungimą, pavyzdžiui 4G ir DSL (angl. *Digital Subscriber Line*).



58 pav. MPTCP paprasto transportavimo režimo antraštės struktūra

Labai svarbi šio režimo funkcija – UDP srauto perdavimas. U – laukelis nurodo, kad gali būti perduodamas UDP srautas. Taip išvengiama inkapsuliacijos sukeltų problemų, kai UDP srautas inkapsuliuojamas po TCP protokolu. Tiesa, tokiam režimui įgyvendinti reikalingi papildomi MPTCP įgaliojimai serveriai (angl. *Proxy*) kliento ir serverio dalyse [27]. Jie atliktą UDP srauto konvertavimą į TCP srautą ir paskirstytą po kelis subkanalus. Šis MPTCP protokolo papildymas dar kartą patvirtina, kad realiojo laiko duomenų perdavimas daugiatinkliško technologijomis turi paklausą.

5.3. Apibendrinimas

Atlikus TCP ir UDP srauto matavimus dviem skirtingais metodais virtualiame tinkle paaiškėjo, kad rezultatai yra panašūs. Perduodant TCP srautą kanalų sujungimo metodu, vėlinimas turi didesnę įtaką perdavimo spartai, nei naudojant MPTCP protokolą. Taip yra todėl, kad kanalų sujungimo metodas naudojo vienodą paketų dalinimo režimą, o MPTCP protokolas veikė prisitaikančio paketų dalinimo režimu. Perduodant realiojo laiko srautą virtualiame tinkle geresnį rezultatą gavome naudojant MPTCP protokolą. Reikia pabrėžti, kad tyrimo metu naudotos VPN programos veikia skirtingai. *Vtun* programa daug efektyviau išnaudoja daugiatinkliškumą lyginant su *Openvpn* programa. Taip pat didelę įtaką daro perduodamo srauto intensyvumas ir naudojamų kanalų pralaidumas. Siunčiant didesnę kiekį duomenų, naudojant MPTCP protokolą, atsiranda paketų praradimas dėl per mažo kanalo pralaidumo.

Analogiški rezultatai gauti naudojant MPTCP protokolą ir realiame tinkle. Naudojant skirtingas prieigas prie interneto technologijas, perduodant TCP srautą, perdavimo sparta agreguojama. Bendra dviejų technologijų perdavimo sparta buvo didesnė, nei naudojant kurią nors vieną technologiją atskirai. Realiojo laiko duomenys nebuvo prarandami, kol srauto intensyvumas, neviršijo kanalų pralaidumo. Viršijus šią ribą, atsiranda paketų praradimai. Iš gautų rezultatų galima teigti, kad MPTCP protokolas, perduodant realiojo laiko duomenis, padidina bendrą sujungimo pralaidumą bei sujungimo patikimumą.

REZULTATŲ APIBENDRINIMAS

- Darbe atlikta daugiatainkliškumo sprendimų, veikiančių kanaliniame, tinklo, transporto ir taikomajame lygmenyse, analizė. Plačiau apžvelgti dažniausiai naudojami SCTP ir MPTCP protokoliai.
- Tyrimui atlikti naudotas MPTCP protokolas. Apžvelgus mokslinę literatūrą ir protokolo specifikaciją, išanalizuotas MPTCP protokolo veikimas. Detaliai pateiktos protokolo tarnybinės žinutės, jų sandara, galimos reikšmės ir funkcijos.
- Pagal iškeltus uždavinius sudaryta sistemos struktūrinė schema. Sudarytos virtualaus ir realaus tinklo schemas tyrimams atlikti. Pasirinkti protokoliai duomenų perdavimui: TCP protokolas ir UDP protokolas realiojo laiko duomenims perduoti. Taip pat parinktos programos sujungimo įgyvendinimui: *Iperf*, *D-ITG*, *Openvpn*, *vtun*. Parinkta matavimams reikalinga įranga, sudarytas matavimų planas.
- Atlikti matavimai virtualiame tinkle naudojant du skirtingus daugiatainkliškumo sprendimus: kanalų sujungimo metodas, MPTCP protokolas su VPN technologija. Sprendimai tarpusavyje palyginti pagal gautus rezultatus. Iš gautų rezultatų matyti, kad patikimesnis yra MPTCP sprendimas, kadangi juo perduodami duomenys gali pasiekti didesnę perdavimo spartą, o realiojo laiko duomenys patiria mažiau praradimų.
- Realiojo laiko srauto perdavimui didelę įtaką daro VPN programinė įranga. Perduodant UDP srautą ir naudojant *Openvpn*, duomenų kanalai lygiagrečiai nebuvo išnaudojami. Tuo tarpu nesudėtingas VPN sprendimas – *vtun*, išnaudoja kelis kanalus vienu metu.
- MPTCP protokolas idealiai veikia, kaip aukštą patikimumą užtikrinantis sprendimas. Nutrūkus vienam iš subkanalų duomenys neprarandami, o visas srautas paskirstomas tarp likusių subkanalų. Protokolo patikimumas užtikrinamas siunčiant paprastą TCP srautą, taip pat ir UDP tipo srautą per VPN tunelį.
- Realiam tinkle atlikti matavimai rodo, kad kuo didesnis skirtumas tarp naudojamų technologijų, pavyzdžiui 4G ir 2G, tuo mažesnė naudo bendrai perdavimo spartai ir sujungimo patikimumui.
- Perduodant realiojo laiko srautą VPN tuneliu, paketų praradimai atsiranda, ne dėl linijos klaidų, o dėl programinės įrangos buferio persipildymo, todėl praktiniai tokio sprendimo realizacijai reiktų modernizuoti vieną iš atviro kodo VPN sprendimų, pritaikant jį realaus srauto perdavimui.

LITERATŪRA

1. *OSI model* [interaktyvus]. [žiūrėta 2015 m. sausio 11 d.]. Prieiga per internetą: <<http://af.wikipedia.org/wiki/OSI-model/>>.
2. *Bandwidth Aggregation Techniques in Heterogeneous Multi-homed Devices* [interaktyvus]. [žiūrėta 2015 m. sausio 11 d.]. Prieiga per internetą: <<http://arxiv.org/pdf/1309.0542.pdf>>.
3. *Interface bonding Ubuntu* [interaktyvus]. [žiūrėta 2015 m. sausio 11 d.]. Prieiga per internetą: <<https://help.ubuntu.com/community/UbuntuBonding>>.
4. *MTCP* [interaktyvus]. [žiūrėta 2015 m. sausio 15 d.]. Prieiga per internetą: <<http://queue.acm.org/detail.cfm?id=2591369/>>.
5. *Multihoming media streaming* [Interaktyvus]. [Žiūrėta 2016-05-08]. Prieiga per internetą: <<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1460621>>.
6. *Multi-homing* [interaktyvus]. [žiūrėta 2015 m. sausio 20 d.]. Prieiga per internetą: <<http://cwi.unik.no/images/1/16/Multi-homing.pdf>>.
7. C. Paasch, et al., 2015 Multipath TCP in the Linux Kernel. [interaktyvus]. [žiūrėta 2015 m. balandžio 28 d.]. Prieiga per internetą: <<http://www.multipath-tcp.org>>.
8. A. Ford, et al., 2015, TCP Extensions for Multipath Operation with Multiple Addresses. [interaktyvus]. [žiūrėta 2015 m. gegužės 31 d.]. Prieiga per internetą: <<https://tools.ietf.org/pdf/draft-ietf-mptcp-rfc6824bis-04.pdf>>.
9. *Huawei E5186*. [interaktyvus]. [žiūrėta 2016 m. sausio 20 d.]. Prieiga per internetą: <<http://support.huawei.com/ecomunity/bbs/10201313.html>>.
10. *LTE-Advanced* [interaktyvus]. [žiūrėta 2016 m. sausio 20 d.]. Prieiga per internetą: <<http://www.3gpp.org/technologies/keywords-acronyms/97-lte-advanced>>.
11. *VPN* [interaktyvus]. [žiūrėta 2016 m. sausio 20 d.]. Prieiga per internetą: <<http://www.cisco.com/c/en/us/support/docs/security/vpn/ipsec-negotiation-ike-protocols/14106-how-vpn-works.html>>.
12. *TCP Option Kind Numbers* [interaktyvus]. [žiūrėta 2016 m. sausio 20 d.]. Prieiga per internetą: <<http://www.iana.org/assignments/tcp-parameters/tcp-parameters.xhtml>>.
13. *Towards Multipath TCP Adoption: Challenges and opportunities* [interaktyvus]. [žiūrėta 2016 m. sausio 20 d.]. Prieiga per internetą: <<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5534465>>.
14. *IPerf* [interaktyvus]. [žiūrėta 2016 m. sausio 21 d.]. Prieiga per internetą: <<https://iperf.fr/>>.
15. *IPerf* [interaktyvus]. [žiūrėta 2016 m. sausio 21 d.]. Prieiga per internetą: <<https://code.google.com/p/iperf/>>.

16. *OpenVPN* [interaktyvus]. [Žiūrėta 2016 m. sausio 20 d.]. Prieiga per internetą: <<http://openmaniak.com/openvpn.php/>>.
17. *OpenVPN* [interaktyvus]. [Žiūrėta 2016 m. sausio 20 d.]. Prieiga per internetą: <<https://openvpn.net/index.php/open-source/documentation/howto.html#pki/>>.
18. *Experimental performance comparison between TCP vs UDP tunnel using OpenVPN* [interaktyvus]. [Žiūrėta 2016 m. sausio 20 d.]. Prieiga per internetą: <<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7374133>>.
19. *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2014–2019* [Interaktyvus]. [Žiūrėta 2016-05-08]. Prieiga per internetą: <<http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html>>.
20. *Namespace Ubuntu* [Interaktyvus]. [Žiūrėta 2016-05-08]. Prieiga per internetą: <<http://blog.scottlowe.org/2013/09/04/introducing-linux-network-namespaces/>>.
21. *D-ITG* [Interaktyvus]. [Žiūrėta 2016-05-08]. Prieiga per internetą: <<http://traffic.comics.unina.it/software/ITG/>>.
22. *Ericsson mobility report* [Interaktyvus]. [Žiūrėta 2016-05-08]. Prieiga per internetą: <<https://www.ericsson.com/res/docs/2015/ericsson-mobility-report-june-2015.pdf>>.
23. *Facebook live broadcast* [Interaktyvus]. [Žiūrėta 2016-05-08]. Prieiga per internetą: <<https://code.facebook.com/posts/1653074404941839/under-the-hood-broadcasting-live-video-to-millions/>>.
24. *MPTCP incast in data center network* [Interaktyvus]. [Žiūrėta 2016-05-08]. Prieiga per internetą: <<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6838660>>.
25. *Preliminary experiments with Multipath-TCP VPNs* [Interaktyvus]. [Žiūrėta 2016-05-08]. Prieiga per internetą: <<http://battlemesh.org/BattleMeshV8/Agenda?action=AttachFile&do=get&target=mptcpvpns.pdf>>.
26. *MPTCP Overview* [Interaktyvus]. [Žiūrėta 2016-05-11]. Prieiga per internetą: <<http://www.cisco.com/c/en/us/support/docs/ip/transmission-control-protocol-tcp/116519-technote-mptcp-00.html>>.
27. *An MPTCP Option for Network-Assisted MPTCP Deployments: Plain Transport Mode* [Interaktyvus]. [Žiūrėta 2016-05-11]. Prieiga per internetą: <<https://tools.ietf.org/html/draft-boucadair-mptcp-plain-mode-06#section-5>>.
28. *Commercial usage of Multipath TCP* [Interaktyvus]. [Žiūrėta 2016-05-11]. Prieiga per internetą: <http://blog.multipath-tcp.org/blog/html/2015/12/25/commercial_usage_of_multipath_tcp.html>.
29. *Stream Control Transmission Protocol* [Interaktyvus]. [Žiūrėta 2016-05-16]. Prieiga per internetą: <<https://tools.ietf.org/html/rfc4960>>.

PRIEDAI

Priedas A. Pažyma apie dalyvavimą XIX-tojoje jaunųjų mokslininkų konferencijoje



VILNIAUS GEDIMINO
TECHNIKOS UNIVERSITETAS
ELEKTRONIKOS FAKULTETAS

XIX-toji Lietuvos jaunųjų mokslininkų konferencija
„MOKSLAS – LIETUVOS ATEITIS“
ELEKTRONIKA IR ELEKTROTECHNIKA

PAŽYMA

Gediminas Uža

*Telekomunikacijų inžinerijos (T180) sekcijoje perskaitė
pranešimą*

**REALAUS LAIKO SRAUTO PERDAVIMO,
NAUDOJANT DAUGIATINKLIŠKUMO
TECHNOLOGIJAS TYRIMAS**

Mokslo komiteto pirmininkas

prof. habil. dr. Romanas Martavičius

Sekcijos pirmininkas

doc. dr. Aurimas Anskaitis

2016 m. kovo 18 d.
Vilnius