



# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

## FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

### ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

## VÝKONOVÁ A BEZPEČNOSTNÍ ANALÝZA TECHNOLOGIE WIREGUARD

PERFORMANCE AND SECURITY ANALYSIS OF WIREGUARD TECHNOLOGY

### BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

### AUTOR PRÁCE

AUTHOR

Oliver Varga

### VEDOUCÍ PRÁCE

SUPERVISOR

Ing. Ondřej Krajsa, Ph.D.

BRNO 2023

# Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

**Student:** Oliver Varga

**ID:** 223335

**Ročník:** 3

**Akademický rok:** 2022/23

**NÁZEV TÉMATU:**

## Výkonová a bezpečnostní analýza technologie Wireguard

### POKYNY PRO VYPRACOVÁNÍ:

V práci proveďte podrobnou analýzu technologie Wireguard, popište jednotlivé použité protokoly a šifrovací algoritmy. Proveďte porovnání s technologiemi IPsec a OpenVPN. Porovnání zaměřte na bezpečnost, dosažitelné přenosové rychlosti, zpoždění a jitter při přenosu.

### DOPORUČENÁ LITERATURA:

[1] WireGuard: Next Generation Kernel Network Tunnel [online]. 2020 [cit. 2022-11-18]. Dostupné z: <https://www.wireguard.com/papers/wireguard.pdf>

[2] Standards for Efficient Cryptography SEC 1. [online]. 2009 [cit. 2022-12-10]. Dostupné z: <http://www.secg.org/sec1-v2.pdf>

**Termín zadání:** 6.2.2023

**Termín odevzdání:** 26.5.2023

**Vedoucí práce:** Ing. Ondřej Krajsa, Ph.D.

**doc. Ing. Jan Hajný, Ph.D.**  
předseda rady studijního programu

### UPOZORNĚNÍ:

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

## **Abstrakt**

Táto práca sa zaoberá popisom technológie virtuálnych privátnych sietí WireGuard, jej porovnaním s technológiami IPsec a OpenVPN na základe nimi používaných šifrovacích algoritmov zabezpečujúcich bezpečnosť a taktiež porovnáva ich technické parametre ako sú oneskorenie, jitter a prenosové rýchlosti. Práca zahŕňa popis technológie WireGuard, porovnanie spomínaných technológií a nakoniec výsledky meraní technických parametrov. Popísaná je taktiež nami zvolená metodika testovania parametrov, použitý hardvér k získaniu výsledkov a topológia siete. Nakoniec sú získané výsledky z testovaní porovnané medzi sebou.

## **Klíčov<sup>á</sup> slova**

WireGuard, IPsec, OpenVPN, site-to-site, iperf3, Mikrotik, jitter, oneskorenie, prenosová rýchlosť.

## **Abstract**

This thesis deals with the description of WireGuard virtual private networks technology, its comparison with the technologies IPsec and OpenVPN based on the encryption algorithms they use to ensure security and also compares their technical parameters such as latency, jitter and transfer speeds. The work includes a description of the WireGuard technology, a comparison of the mentioned technologies and finally the results of the measurements of their technical parameters. The methodology chosen by us for testing the parameters, the hardware used to obtain the results and the network topology are also described. Finally, the results obtained from the tests are compared.

## **Keywords**

WireGuard, IPsec, OpenVPN, site-to-site, iperf3, Mikrotik, jitter, latency, transfer speed.

## **Bibliografická citace**

VARGA, Oliver. *Výkonová a bezpečnostní analýza technologie Wireguard* [online]. Brno, 2023 [cit. 2023-05-23]. Dostupné z: <https://www.vut.cz/studenti/zav-prace/detail/151235>. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce Ondřej Krajsa.

# Prohlášení autora o původnosti díla

**Jméno a příjmení studenta:** *Oliver Varga*

**VUT ID studenta:** *223335*

**Typ práce:** *Bakalářská práce*

**Akademický rok:** *2022/23*

**Téma závěrečné práce:** *Výkonová a bezpečnostní analýza  
technologie Wireguard*

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: 20. května 2023

-----  
podpis autora

## **Poděkování**

Ďakujem vedúcemu bakalárskej práce Ing. Ondřejovi Krajsovi, Ph. D. za účinnú metodickú, pedagogickú a odbornú pomoc a ďalšie cenné rady pri spracovaní mojej bakalárskej práce.

V Brně dne: 20. května 2023

-----  
podpis autora

# Obsah

<b>1. TECHNOLÓGIA WIREGUARD.....</b>	<b>11</b>
1.1 SMEROVANIE A SMEROVACIA TABUĽKA.....	12
1.2 ROAMING DIZAJN A KONCOVÉ BODY .....	13
1.3 TOKY ODOSIELANIA A PRIJÍMANIA .....	15
1.4 KRYPTOGRAFIA A POUŽITÉ PROTOKOLY .....	16
1.4.1 Voliteľný predom zdieľaný kľúč .....	17
1.4.2 Princíp cookies.....	18
1.5 PRINCÍP SPRÁV .....	20
1.5.1 Správa iniciátor – príjemca .....	21
1.5.2 Správa príjemca naspäť iniciátorovi.....	22
1.5.3 Cookie funkcie MAC .....	23
1.5.4 Cookie reply správa .....	23
1.5.5 Transportné dáta – odvodenie kľúča a správy o transportných údajoch.....	24
<b>2. POROVNANIE WIREGUARD S IPSEC A OPENVPN .....</b>	<b>26</b>
<b>3. TESTOVANIE.....</b>	<b>28</b>
3.1 METODIKA TESTOVANIA .....	28
3.2 TOPOLOGIA SIETE – SITE-TO-SITE.....	29
3.3 POUŽITÉ KOMPONENTY .....	29
<b>4. VÝSLEDKY TESTOVANÍ TECHNICKÝCH PARAMETROV VPN.....</b>	<b>31</b>
4.1 TESTOVANIE BEZ POUŽITIA VIRTUÁLNEJ PRIVÁTNEJ SIETE – STATICKÉ SMEROVANIE MEDZI LOKÁLNymi SIETAMI (RB-493G).....	31
4.2 TESTOVANIE S POUŽITÍM WIREGUARD VIRTUÁLNEJ PRIVÁTNEJ SIETE (RB-493G) .....	33
4.3 TESTOVANIE S POUŽITÍM IPSEC VIRTUÁLNEJ PRIVÁTNEJ SIETE (RB-493G).....	35
4.4 TESTOVANIE S POUŽITÍM OPENVPN VIRTUÁLNEJ PRIVÁTNEJ SIETE (RB-493G) .....	37
4.5 TESTOVANIE BEZ POUŽITIA VIRTUÁLNEJ PRIVÁTNEJ SIETE – STATICKÉ SMEROVANIE MEDZI LOKÁLNymi SIETAMI (RB941-2ND) .....	39
4.6 TESTOVANIE S POUŽITÍM WIREGUARD VIRTUÁLNEJ PRIVÁTNEJ SIETE (RB941-2ND).....	41
4.7 TESTOVANIE S POUŽITÍM IPSEC VIRTUÁLNEJ PRIVÁTNEJ SIETE (RB941-2ND) .....	42
4.8 TESTOVANIE S POUŽITÍM OPENVPN VIRTUÁLNEJ PRIVÁTNEJ SIETE (RB941-2ND).....	45
<b>5. POROVNANIE NAMERANÝCH HODNÔT .....</b>	<b>47</b>
<b>6. ZÁVER.....</b>	<b>48</b>

## SEZNAM OBRÁZKŮ

4.1	Zaťaženie procesora Mikrotik RB-493G, statické smerovanie .....	31
4.2	Iperf3 ukážka výstupu testu, statické smerovanie .....	32
4.3	Zaťaženie procesora Mikrotik RB-493G, WireGuard .....	33
4.4	Iperf3 výstup, WireGuard.....	34
4.5	Zaťaženie procesora Mikrotik RB-493G, IPsec .....	35
4.6	Iperf3 výstup, IPsec.....	36
4.7	Bottleneck procesora Mikrotik, IPsec odosielajúci .....	36
4.8	Zaťaženie procesora Mikrotik RB-493G, OpenVPN .....	37
4.9	Iperf3 výstup, OpenVPN.....	38
4.10	Zaťaženie procesora Mikrotik RB941-2nD, statické smerovanie .....	39
4.11	Iperf3 výstup, statické smerovanie RB941-2nD.....	40
4.12	Zaťaženie procesora Mikrotik RB941-2nD, WireGuard.....	41
4.13	Iperf3 výstup, WireGuard, RB941-2nD .....	42
4.14	Zaťaženie procesora Mikrotik RB941-2nD, IPsec .....	43
4.15	Iperf3 výstup, IPsec, RB941-2nD .....	44
4.16	Zaťaženie procesora Mikrotik RB941-2nD, OpenVPN .....	45
4.17	Iperf3 výstup, OpenVPN, RB941-2nD .....	46



## SEZNAM TABULEK

1.1	Konfigurácia 1a smerovacej tabuľky .....	12
1.2	Konfigurácia 2a smerovacej tabuľky .....	13
1.3	Konfigurácia 2b smerovacej tabuľky .....	14
1.4	Konfigurácia 1b smerovacej tabuľky .....	14
1.5	Správa iniciátora príjemcovi .....	21
1.6	Správa od príjemcu naspäť iniciátorovi.....	22
1.7	Cookie Reply Správa.....	24
5.1	Tabuľka priemerných hodnôt – RB-493G.....	47
5.2	Tabuľka priemerných hodnôt – RB941-2nD.....	47

# ÚVOD

V dnešnej dobe, kedy sa prakticky väčšina procesov spojených s našimi každodennými požiadavkami a povinnosťami odhráva online, stúpajú nároky na samotnú bezpečnosť týkajúcu sa či už osobných, ale aj finančných informácií. Jednou z možností, ako tieto informácie chrániť ponúkajú takzvané virtuálne privátne siete, ktoré vytvárajú zabezpečené spojenie na menej zabezpečených sieťach. Medzi služby, ktoré VPN poskytujú, patrí napríklad skrytie reálnej IP adresy a lokácie. Využívané sú častejšie veľkými spoločnosťami, ktoré nimi zabezpečujú svojim zamestnancom bezpečný prístup k podnikovým aplikáciám počas práce z domu, ale aj mnoho iných výhod. Jednou z týchto virtuálnych sietí je aj technológia WireGuard, ktorou sa táto práca zaoberá.

Cieľom práce bolo vykonať analýzu technológie WireGuard, popísať jej jednotlivé algoritmy a použité šifrovacie algoritmy. Ďalej bolo potrebné porovnať ju s technológiami IPsec a OpenVPN z hľadiska bezpečnosti. Poslednou požiadavkou bolo pomocou testovacích scenárov a vhodným testovacím nástrojom odmerať technické parametre spomínaných VPN technológií, ako sú jitter, prenosová rýchlosť či oneskorenie.

V prvej kapitole a jej podkapitolách práca popisuje technológiu WireGuard a jej fungovanie. V ďalšej časti, teda druhej kapitole, je vykonané porovnanie zadaných technológií. V ďalšej časti práce je popísaná metodika merania jednotlivých parametrov, topológia siete a taktiež použitý hardvér a testovacie nástroje. V predposlednej časti, teda v štvrtej kapitole, sú nami namerané údaje týkajúce sa technických parametrov odprezentované a v poslednej časti sa práca venuje vzájomnému porovnaniu získaných výsledkov testov.

# 1. TECHNOLÓGIA WIREGUARD

WireGuard je bezpečný VPN tunel, ktorý funguje na tretej vrstve referenčného modelu ISO/OSI. Spočiatku implementovaný ako jadrové virtuálne sieťové rozhranie pre Linux, požadovaným cieľom jeho vzniku bolo nahradenie technológií IPsec a OpenVPN vo väčšine prípadov použitia, pričom by, pri zachovaní optimálneho výkonu, stále zostal rýchlejší, bezpečnejší a jednoduchší na používanie ako konkurencia.

Rozhranie vychádza z fundamentálnych princípov zabezpečených tunelov, a teda samotnou podstatou je asociácia medzi IP adresou zdroja tunela a verejným kľúčom rovesníka (peer public key). Výmena kľúčov a ustanovenie zašifrovaných kanálov medzi klientami sú zabezpečené NoiseIK protokolom. Všetka tvorba relácií je spracovávaná transparentne pre používateľa. Vzájomná autentizácia prebieha v štýle podobnom OpenSSH, teda používajú sa krátke vopred zdieľané statické kľúče, ktoré sú vo svojej podstate bodmi na eliptickej krivke Curve25519 [2].

Protokol zabezpečuje mimo iné aj vysokú úroveň skrývania identity a vyznačuje sa taktiež vysokou mierou doprednej bezpečnosti (perfect forward secrecy). Používaním kryptografického algoritmu ChaCha20Poly1305 [8] nedochádza k významnému obmedzeniu prenosových rýchlostí. Tento algoritmus zohráva úlohu pri zapúzdrení paketov v UDP pomocou overeného šifrovania. Docielené je taktiež aj zmiernenie útokov odmietnutia služby (denial of service), a to za pomoci vylepšeného prístupu k súborom cookie viazaných na IP. Výrazne zlepšenie je patrné aj pri mechanizmoch súborov cookie IKEv2 [3] a DTLS [4], ktoré majú za úlohu šifrovanie a autentifikáciu. Výsledný dizajn umožňuje zaobísť sa bez pridelenia zdrojov v reakcii na prijaté pakety. Jednoduchosť protokolu WireGuard nakoniec spočíva v samotnej implementácii, kedy pre operačné systémy Linux stačí menej ako 4 000 riadkov kódu, čo v konečnom dôsledku znamená značné zjednodušenie potrebného manažmentu alebo prípadnej údržby.

WireGuard sa vyhýba zložitému vrstveniu ako je tomu napríklad pri protokole IPsec, kde je využívaná takzvaná transformačná vrstva xfrm. Pri jej používaní používateľ vyplní jadrovú štruktúru tak, že zadá akú šifrovú súpravu alebo aké kľúče, poprípade môže tiež špecifikovať možnosti transformácií, ako je napríklad kompresia, bude používať ktorý selektor paketov prechádzajúcich preddefinovaným operačným prostredím. Vo všeobecnosti je potom za aktualizáciu týchto dátových štruktúr zodpovedný používateľský priestorový daemon, a to na základe výsledkov výmeny kľúčov, kedy vo väčšine prípadov je toto docielené a manažované protokolom IKEv2 [3], čo je sám o sebe zložitý protokol, a preto je náročnosť a obsiahlosť samotného kódu značná. I keď toto spomínané vrstvenie IPsecu môže byť zo sémantického hľadiska v poriadku, ako aj riešenie oddelenia transformačnej vrstvy od vrstvy rozhrania môže byť múdre z pohľadu sieťovania, spôsobuje problémy vychádzajúce z jeho náročnosti a zvyšuje aj nároky na správnu implementáciu a nasadenie. Namiesto toho WireGuard ponúka riešenie vo virtuálnom rozhraní, ktoré pri konfigurácii môže byť pomenované

ľubovoľne, napríklad `wireguard1`. Toto rozhranie môže byť spravované pomocou nástrojov ako `ip` a `ifconfig`. Konfigurácia rozhrania pre správne fungovanie tunela nie je zložitá a spočíva v nastavení privátneho kľúča a rozličných verejných kľúčov rovesníkov, s ktorými bude prebiehať zabezpečená komunikácia. Voliteľne je možné konfigurovať aj predzdieľaný symetrický kľúč. Konkrétne operácie, ako napríklad výmena kľúčov, spojenie, prerušenie spojenia, obnovenie spojenia, zisťovanie možných spojení a podobne, sú uskutočňované mimo zraku používateľa (správcu), pričom zostávajú transparentné a spoľahlivé, a vďaka tomu z pohľadu spravovania WireGuard rozhranie pôsobí bezstavovo. Zaručené je, že pakety prichádzajúce z rozhrania WireGuard budú zašifrované a overené aj pri bežnom nastavení pravidiel brány firewall. V konečnom dôsledku vďaka jednoduchosti a priamočiarosti protokolu sú šance na zlyhanie alebo nesprávnu konfiguráciu oveľa menšie ako pri IPsec.

Nakoniec, WireGuard ako VPN technológia používa špecifické kryptografické nástroje a protokoly. Je nadizajnovaný tak, aby bol odolný voči zraniteľnostiam a útokom pomocou fixnej sady kryptografických primitív. Jedná sa o Noise [5] protokol zabezpečujúci výmenu kľúčov, Curve25519 [2] pre ECDH, HKDF [6] pre rozširovanie kľúčov, ChaCha20 [7] verzia podľa RFC7539 [8] a Poly1305 [8] pre overené šifrovanie a nakoniec protokol BLAKE2s [9] na používaný pri procese hashovania.

## 1.1 Smerovanie a smerovacia tabuľka

Základným princípom každej bezpečnej VPN je asociácia medzi rovesníkmi a ich pridelenými IP adresami použitými ako zdrojové IP adresy. V technológii WireGuard sa každý rovesník vyznačuje verejným kľúčom dlhým 32 bajtov. To znamená, že vo WireGuard existuje jednoduché mapovanie asociácií medzi verejnými kľúčmi a množinou autorizovaných IP adries [1].

Tabuľka 1.1 Konfigurácia 1a smerovacej tabuľky

<i>Verejné kľúče rozhraní</i>	<i>Súkromné kľúče rozhrania</i>	<i>Načúvajúci UDP Port</i>
<i>dec1...QnbV</i>	<i>aiKm...sR9j</i>	<i>41414</i>
<i>Verejné kľúče rovesníkov</i>	<i>Povolené zdrojové adresy</i>	
<i>jan0...4xEf</i>	<i>10.192.122.7/32, 10.192.124.0/24</i>	
<i>XBnR...1qIn</i>	<i>10.192.122.8/32</i>	
<i>N7op...iJpO</i>	<i>10.10.10.198/32</i>	

Samotné rozhranie má svoj vlastný verejný kľúč a priradený UDP port, na ktorom načúva. Nasleduje list rovesníkov, kde každý rovesník je charakterizovaný vlastným verejným kľúčom a zoznamom autorizovaných zdrojových IP adries.

Pri prenose odchádzajúceho paketu cez rozhranie WireGuard, `wireguard1`, Tabuľka 1.1 je použitá ako zdroj informácií pre určenie, ktorý verejný kľúč bude použitý

na šifrovanie. Napríklad, paket určený cieľovej IP adrese 10.192.122.8 bude zašifrovaný pomocou bezpečnej relácie odvodenej od verejného kľúča príslušného rovesníka, teda *XBnR...Iqln*. Ak rozhranie wireguard1 naopak zašifrovaný paket prijíma, po dešifrovaní a verifikovaní bude prijatý iba v prípade, že sa jeho zdrojová IP adresa zhoduje s adresou pridelenou verejnému kľúču použitému v zabezpečenej relácii na jeho dešifrovanie. Napríklad, ak je paket dešifrovaný z *jan0...4xEf*, nebude zahodený len v prípade, ak má zdrojovú IP adresu zodpovedajúcu 10.192.122.7 alebo bližšie špecifikovanému rozsahu, napríklad od 10.192.124.0 do 10.192.124.255. [1]

Tento jednoduchý prístup autorov WireGuardu k smerovaniu umožňuje spoľahnúť sa aj na základné nastavenia firewallu. Keďže WireGuard je tunel striktné založený na fungovaní iba na tretej vrstve, vo všeobecnosti, každý paket prechádzajúci jeho rozhraním má zaručenú nielen autentickosť zdrojovej IP adresy, ale aj garantovanú doprednú bezpečnosť prenosu. Takéto fungovanie, teda identifikácia rovesníkov už na tretej vrstve, umožňuje oproti iným bežným zabezpečeným tunelom, ako napríklad IPsec, omnoho prehľadnejší a jednoduchší sieťový návrh.

V situácii, kedy sa WireGuard rovesník rozhodne smerovať všetku komunikáciu cez iného WireGuard rovesníka, napríklad prípad, kedy používateľ oprávňuje *dec1...QnbV* poslať pakety rozhraním wireguard1 s ľubovoľnou IP adresou zdroja, kedy všetky odchádzajúce pakety z wireguard1 budú zašifrované zabezpečenou reláciou vychádzajúcou z tohto verejného kľúča a ďalej odoslané na jeho koncový bod(endpoint), by mohla tabuľka smerovania vyzeráť napríklad takto [1]:

Tabuľka 1.2 Konfigurácia 2a smerovacej tabuľky

<i>Verejné kľúče rozhraní</i>	<i>Súkromné kľúče rozhrania</i>	<i>Načúvajúci UDP Port</i>
<i>kv7Q...jLMx</i>	<i>15uT...CnhE</i>	21841
<i>Verejné kľúče rovesníkov</i>	<i>Povolené zdrojové adresy</i>	
<i>dec1...QnbV</i>	<i>0.0.0.0/0</i>	

## 1.2 Roaming dizajn a koncové body

Pre WireGuard rovesníkov je dôležité, aby bolo navzájom medzi nimi možné uskutočňovať výmenu šifrovaných paketov UDP na špecifické koncové body internetu. Každý z rovesníkov vyskytujúcich sa v smerovacej tabuľke môže vopred voliteľne upresniť vonkajšiu IP adresu a UDP port svojho koncového bodu. Toto upresnenie adresy a portu je voliteľné, pretože aj v prípade, kedy nie sú špecifikované stačí, aby WireGuard obdržal správne overený rovesníkov paket a na určenie koncového bodu použije okrajovú vonkajšiu IP adresu zdroja.

Verejné kľúče slúžia ako identifikátory konkrétnych rovesníkov. Na druhú stranu, okrajová vonkajšia IP adresa zdroja zašifrovaného paketu umožňuje rozpoznať vzdialený koncový bod rovesníka. Toto dáva rovesníkom možnosť voľného pohybu medzi rozličnými vonkajšími IP adresami, napríklad ako je to možné medzi mobilnými sieťami pomocou implementácie SSH známej ako Mosh [10]. Môžu teda svojvoľne meniť svoju IP adresu bez toho, aby stratili možnosť komunikovať s ostatnými v sieti. Toto je možné, pretože v konfigurácii je uvedená len inicializačná adresa, pomocou ktorej príde k prvotnému odoslaniu dát. Po prijatí prvej odpovede sa WireGuard zaoberá už len IP adresami, z ktorých mu prichádzajú len najnovšie overené dáta. Zmenu adresy si ostatní rovesníci všimnú a podľa nej potom riadia následné smerovanie. Po uplatnení týchto rozšírení by teda uvedená tabuľka smerovania mohla vyzeráť nasledovne [1]:

Tabuľka 1.3 Konfigurácia 2b smerovacej tabuľky

	<b>Súkromné kľúče</b>	
<b>Verejné kľúče rozhraní</b> <i>kv7Q...jLMx</i>	<b>rozhrania</b> <i>15uT...CnhE</i>	<b>Načúvajúci UDP Port</b> 21841
<b>Verejné kľúče rovesníkov</b> <i>dec1...QnbV</i>	<b>Povolené adresy</b> <i>0.0.0.0/0</i>	<b>Koncový bod</b> <i>192.158.1.38:41414</i>

V prípade, kedy *dec1...QnbV* na adrese 192.158.1.38:41414 obdrží zašifrovaný paket od hostiteľa *kv7Q...jLMx*, aktualizuje sa jeho tabuľka s novou informáciou o koncovom bode, na ktorý bude potom odosielať pakety s odpoveďami. Tento koncový bod môže mať adresu napríklad 192.95.5.64:21841 [1]:

Tabuľka 1.4 Konfigurácia 1b smerovacej tabuľky

	<b>Súkromné kľúče</b>	
<b>Verejné kľúče rozhraní</b> <i>dec1...QnbV</i>	<b>rozhrania</b> <i>aiKm...sR9j</i>	<b>Načúvajúci UDP Port</b> 41414
<b>Verejné kľúče rovesníkov</b> <i>jan0...4xEf</i> <i>XBnR...1qln</i> <i>N7op...iJpO</i>	<b>Povolené zdrojové adresy</b> <i>10.192.122.7/32, 10.192.124.0/24</i> <i>10.192.122.8/32</i> <i>10.10.10.198/32</i>	<b>Koncový bod</b> <i>192.158.1.33:21841</i>

Je potrebné uviesť si, že port, na ktorom načúvajú rovesníci, je vždy rovnaký ako port zdroja odosielaných paketov. Tým je zaručená nielen výrazná jednoduchosť, ale

aj dôveryhodné prechádzanie cez NAT. Vďaka tejto roamingovej vlastnosti majú rovesníci vždy najaktuálnejšie informácie o vonkajších IP adresách a UDP portoch.

Takéto fungovanie zabezpečuje nielen minimálnu potrebnú konfiguráciu, ale aj značné pohodlie. Zatiaľ čo z pohľadu man-in-the-middle útočníka je možné pozmeniť neoverené vonkajšie zdrojové IP adresy, nebude takýto útočník schopný dešifrovať alebo úmyselne upraviť payload. Samozrejme, útok odmietnutia služby by mohol byť pre takéhoto útočníka jednoduchý, stačilo by, aby zahadzoval pôvodné pakety, avšak hostitelia, ktorí nie sú schopní dešifrovať a následne potom odpovedať na pakety sú rýchlo zabudnutí.

### 1.3 Toky odosielania a prijímania

Keď spojíme roamingový dizajn a dizajn smerovania WireGuardu, ktoré sú vysvetlené v predchádzajúcich podkapitolách 1.1 a 1.2, dostaneme sa k samotnému fungovaniu prijímania a odosielania paketov prostredníctvom wireguard1 rozhrania použitím tabuliek z konfigurácie 1a.

Prípad, kedy sa lokálne vygenerovaný (prípadne preposlaný) paket bude prenášať cez odchádzajúce rozhranie wireguard1 [1]:

1. Najprv na rozhranie wireguard1 dorazí plaintextový paket.
2. Je skontrolovaná cieľová adresa tohto paketu, teda 192.168.87.21. Táto adresa zodpovedá rovesníkovi TrMv...WXX0 v tabuľke. V prípade, že adresa nezodpovedá žiadnemu z rovesníkov, paket je zahodený a odosielateľ je o tom informovaný ICMP odpoveďou „no route to host“ a zároveň je vrátená aj hodnota `-ENOKEY`.
3. Paket je zašifrovaný pomocou ChaCha20Poly1305 [8]. Pri šifrovaní je použitý symetrický šifrovací kľúč spolu s nonce počítačom zabezpečenej relácie, ktoré sú asociované s identifikovaným rovesníkom.
4. K paketu sa po zašifrovaní pridá hlavička obsahujúca rozličné polia.
5. Nakoniec sú paket s priradenou hlavičkou odoslané vo forme UDP paketu do internetového UDP/IP koncového bodu asociovaného s rovesníkom. Ak nie je rozpoznávaný koncový bod, paket je zahodený a informuje o tom ICMP paket odpoveďou a vrátená je tiež aj hodnota `-EHOSTUNREACH`.

Paket dorazí na UDP port 41414, čo je port, na ktorom načúva rozhranie wireguard1 [1]:

1. Na vyhovujúcom porte je prijatý paket spolu s priradenou hlavičkou a zašifrovaným payloadom.
2. Na základe hlavičky WireGuard rozpozná asociáciu s rovesníkovou reláciou a skontroluje správnosť počítača správ. Následne sa ju pokúsi overiť a dešifrovať pomocou symetrického kľúča relácie. Nerozpoznanie rovesníka alebo neúspešné overenie končia zahodením paketu.

3. Paket je overený, zdrojová IP adresa vonkajšieho UDP/IP paketu aktualizuje koncový bod konkrétneho rovesníka.
4. Po dešifrovaní payloadu má rozhranie k dispozícii plaintextový paket. Ak sa nejedná o IP paket, je zahodený. Kontroluje sa či IP adresa zdroja vnútorného paketu korešponduje s údajmi uvedenými v smerovacej tabuľke. Ak je teda zdrojová IP adresa dešifrovaného paketu napríklad 192.168.31.28, paket zodpovedá smerovaniu rovesníka. V prípade, že smerovaniu nezodpovedá, je zahodený.
5. Ak v celom procese nebol paket zahodený, dostáva sa ďalej do fronty prijímania na rozhraní wireguardl.

## 1.4 Kryptografia a použité protokoly

Aby bolo možné zahájiť posielanie paketov, je potrebné, aby pred tým došlo k výmene kľúčov. Táto výmena je realizovaná TLS 1.3 1-RTT (one round trip time) handshake protokolom, kedy iniciátor komunikácie posielajú správu odpovedajúcemu a ten naspäť odosiela správu iniciátorovi. Po uskutočnení tejto výmeny môže iniciátor odosielať príjemcovi zašifrované správy pomocou zdieľaného páru symetrických kľúčov. Jeden z tejto dvojice kľúčov slúži k odosielaniu, druhý k prijímaniu správ. Po prijatí prvej zašifrovanej správy môže príjemca ďalej odosielať vlastné zašifrované správy iniciátorovi. Bližšie fungovanie tohto procesu výmeny je popísané pre KEA+C protokol [11]. Je dôležité poznamenať, že táto správa o podaní rúk je uskutočňovaná asynchrónne s prenosom dátových správ. Aby sa docielilo zmiernenie útokov na odmietnutie služby, tieto správy sú riadané podľa „IK“ predlohy Noise protokolu [5] spolu s novou cookie konštrukciou. Výsledkom takto zostaveného protokolu je nakoniec robustný systém, ktorý je zabezpečený a navyše spĺňa požiadavky zabezpečenia autentifikovanej výmeny kľúčov (AKE) podľa [11]. Ďalšími kľúčovými vlastnosťami sú vysoká miera doprednej bezpečnosti, utajovanie statických verejných kľúčov, odolnosť proti útokom odmietnutia služby a v neposlednom rade je prítomná schopnosť vyhnúť sa odcudzeniu identity na základe zisteného kľúča.

Wireguard je navrhnutý s cieľom vyhnúť sa ukladaniu akéhokoľvek stavu pred samotnou autentifikáciou a neodpovedať na neoverené pakety. Takýmto návrhom je zabezpečené to, že WireGuard zostáva pre neautentifikovaných rovesníkov a rôzne sieťové skenery neviditeľný, vďaka čomu je možné vyhnúť sa hneď niekoľkým typom útokov. WireGuard je na druhú stranu možné implementovať aj bez použitia dynamickej alokácie pamäte, a to ani pre overené pakety. Tento predstavený dizajn avšak vyžaduje, aby bola vždy prvá správa doručená odpovedajúcemu účastníkovi použitá na overenie iniciátora komunikácie, s čím prichádza možnosť zneužitia útočníkom, kedy je mu umožnené prevedenie útoku prehraním (replay attack), kde môže zopakovať prvotnú správu o podaní rúk. Takto oklame obeť, ktorá nevedome obnoví svoj efemérny



(ephemeral) kľúč, čím príde k zneplatneniu relácie iniciátora. Správa avšak aj napriek takémuto útoku zostane utajená a nenaruší sa ani jej autentickosť. Aby sa vyšlo útokom prehraním, je vo WireGuard použitá 12-bytová časová značka TAI64N [12], ktorá sa pridáva do prvej správy, kde je následne zašifrovaná a autentifikovaná. Odpovedajúci následne uchováva informáciu iba o najväčšej časovej značke prijatej od rovesníka a ďalej zahadzuje pakety s rovnakou alebo menšou hodnotou. Pri možnom narušení tohto stavu nedochádza k žiadnemu problému ani možnosti rozvrátiť aktuálne prebiehajúcu reláciu medzi iniciátorom a odpovedajúcim rovesníkom pomocou útoku prehraním, pretože odpovedajúci práve znovu začal a aktuálne nemá žiadnu zabezpečenú reláciu k narušeniu. Akonáhle iniciátor znovu naviaže zabezpečenú reláciu s odpovedajúcim rovesníkom, bude použitá časová značka s vyššou hodnotou, čím sa znehodnotí značka predošlá. Výhodou tejto časovej značky je jej samotná implementácia, keďže sa jedná o big-endian. Je možné teda porovnávať dve dvanásť bytové časové značky pomocou štandardnej funkcie programovacieho jazyka C, `memcmp()`. Aby zostala zachovaná miera popierateľnosti, prvotné správy vo WireGuard používajú výsledok Diffie-Hellmanovho výstupu zo statických kľúčov oboch rovesníkov pre autentifikáciu. Pri kompromitácii jedného zo statických kľúčov môže útočník síce zostaviť iniciačnú správu, nebude však schopný dokončiť proces podania rúk (handshake). Na druhej strane tým ale získa možnosť zabrániť všetkým budúcim spojeniam, pretože vytvorená iniciačná správa obsahuje maximálnu hodnotu časovej značky. Takýto scenár sa spočiatku môže podobáť na zvyčajné zraniteľnosti vychádzajúce z odcudzenia identity na základe kompromisu medzi kľúčmi (KCI útok) – na tie nie je WireGuard zraniteľný – v skutočnosti ide avšak o niečo iné. V tomto prípade samotný útočník pomôže odhaliť a odvrátiť ním využitú zraniteľnosť, pretože kompromitovaný kľúč už nemôže byť ďalej používaný rovesníkmi. V prípade, že by presnosť TIA64N časovej značky predstavovala bezpečnostné riziko v podobe nežiadúceho úniku informácií, implementácia dovoľuje skrátiť 24 bitov z jej nanosekundovej časti.

#### **1.4.1 Voliteľný predom zdieľaný kľúč**

WireGuard zakladá na nutnosti výmeny statických verejných kľúčov rovesníkov pred zahájením akejkoľvek komunikácie. Dôvernosť všetkých prenášaných informácií ďalej závisí od zabezpečenia Curve25519 ECDH funkcie. Pri navrhovaní zabezpečenia autori nezabudli ani na kvantovú výpočtovú techniku a s cieľom zabrániť jej možným budúcim pokrokom, technológia WireGuard podporuje mód, v ktorom si ktorékoľvek páry rovesníkov môžu navzájom dodatočne vopred zdieľať jeden 256-bitový symetrický šifrovací kľúč. Podstatou takéhoto vopred zdieľaného kľúča je pridať ďalšiu vrstvu symetrického šifrovania, čím sa zvýši úroveň ochrany. Myslená hrozba spočíva v scenári, kedy útočníci odchyťávajú zašifrovanú komunikáciu s nádejou, že jedného dňa bude možné prelomiť Curve25519 a im tak bude umožnené dešifrovať predom nazbierané

dáta. Z hľadiska správy kľúčov sú v dnešnej dobe dopredu zdieľané symetrické šifrovacie kľúče problematické. Pri takomto zdieľaní kľúča môže prísť k jeho kompromitácii, avšak myšlienkou navrhnutého zabezpečenia je, že v čase, kedy budeme pomocou kvantových výpočtov schopní prelomiť Curve25519, bude takto predom zdieľaný symetrický kľúč dávno zabudnutý. Najdôležitejšie avšak je, že ak by aj nakoniec ku kompromitácii predzdieľaného kľúča došlo, samotné kľúče eliptickej krivky Curve25519 stále poskytujú dostatočnú ochranu. Nakoniec výsledkom takéhoto voliteľného spojenia predom zdieľaných symetrických kľúčov a kryptografickej eliptickej krivky je dosiahnutý prijateľný kompromis aj pre tých najviac podozrievavých používateľov.

#### 1.4.2 Princíp cookies

Aby bolo možné pri zostavovaní komunikácie overiť autentickosť handshake správy, je potrebné, aby sa na krivke Curve25519 vynásobili jej body. Tento proces násobenia aj napriek tomu, že samotná krivka Curve25519 je na väčšine procesorov rýchla, pôsobí značne zaťažujúco pre procesor. Dochádza počas neho k vystaveniu protokolu potenciálnej hrozbe v podobe útoku odmietnutia služby. Aby sa bol teda odpovedajúci rovesník – príjemca správy – schopný vyhnúť takémuto útoku, kedy je pod určitým zaťažením spôsobeným procesom násobenia, má možnosť odmietnuť spracovanie handshake správy (iniciačnej alebo správy slúžiacej ako odpoveď) a namiesto toho odpovedať pomocou cookie reply správy obsahujúcej cookie. Iniciátor takto získané cookie potom môže použiť k opätovnému odoslaniu správy, kde požaduje jej následné prijatie príjemcom. Keď je teda, napríklad server, pod určitým zaťažením, nemôže odpovedať kým neobdrží platný paket. Preto je vyžadované, aby počas zaťaženia všetky správy obsahovali MAC funkciu, ktorá kombinuje verejný kľúč prijímajúceho a voliteľne aj predom zdieľaný kľúč ako MAC kľúč.

Spomínané cookie je v podstate výsledkom výpočtu MAC funkcie iniciátorovej zdrojovej IP adresy pomocou tajnej náhodnej hodnoty meniacej sa každé dve minúty, ktorú si uchováva odpovedajúci rovesník. Táto meniacia sa hodnota pri výpočte slúži ako MAC kľúč. V situácii, kedy iniciátor opätovne odosiela správu, posíla ju spolu s MAC funkciou jeho správy a použije toto cookie ako MAC kľúč. Príjemca správy sa následne (pri zaťažení) na základe platnosti, teda prípad, kedy ide o správnu MAC funkciu používajúcu cookie ako kľúč, tejto správy rozhodne či ju prijať a spracovať. Vďaka tomuto mechanizmu je zaručený aj dôkaz o vlastníctve IP, pretože odosielané správy sú spjané s iniciátorovou IP adresou, čo ďalej umožňuje obmedzovanie rýchlosti pomocou klasických algoritmov obmedzujúcich rýchlosť IP.

Takáto schéma avšak predstavuje tri hlavné problémy. Prvý z problémov sa spája s problematikou opísanou na začiatku kapitoly 1.4, a teda so samotným návrhom, kedy nedochádza k posielaniu odpovedí na neautentifikované správy. Odosielanie cookie reply správy by túto vlasnosť porušovalo. Ďalší problém predstavuje odosielanie cookie, ktoré

by v žiadnom prípade nemalo byť odosielané v podobe čistého textu, pretože by sa tým otvorila možnosť pre útok človekom medzi, ktorý by ho mohol použiť na odosielanie podvodných správ. Nakoniec tretím problémom je fakt, že samotný iniciátor by sa mohol ocitnúť pod útokom odmietnutia služby, kedy sú mu odosielané podvodné cookie, ktoré by mu ďalej znemožňovali správne vypočítať MAC funkciu jeho správy. Odpoveďou autorov WireGuardu na tieto problémy je použitie dvoch MAC funkcií (`msg.mac1` a `msg.mac2`). Výpočet týchto funkcií je vysvetlený neskôr v kapitole 1.5.3.

Aby v situácii popísanej ako prvý problém zostal odpovedajúci rovesník počas zaťaženia mlčať, je v každej správe prítomná `msg.mac1` funkcia používajúca jeho verejný kľúč. Takéto fungovanie prinajmenšom zabezpečí, že pre vyvolanie akejkoľvek odpovede, rovesník posielajúci správu musí poznať partnera, s ktorým komunikuje, čo je možné na základe poznania jeho verejného kľúča. Je vhodné poznamenať, že platná `msg.mac1` musí byť prítomná v každej situácii, nielen v situácii počas zaťaženia. I keď verejný kľúč rovesníka sám o sebe nie je tajný, je stále dostačujúci v rámci tohto modelu útoku, ktorého podstatou je zabezpečiť neviditeľnosť určitých procesov a služieb, a teda poznatok o tomto kľúči slúži ako dostatočný dôkaz o znalosti jeho existencie. Aj napriek tomu však prvá MAC funkcia `msg.mac1` dáva možnosť pasívnemu útočníkovi pokúšať sa o uhádnutie, pre ktorý verejný kľúč je daný paket počas komunikácie zamýšľaný, čím sú do určitej miery oslabené schopnosti skrývania identity. Neznamená to však, že správnym odhadom by prišlo ku kryptografickému dôkazu, pretože pri procese tvorenia MAC nie je použitý žiadny súkromný materiál.

Druhý problém, teda odosielanie MAC funkcií v podobe čistého textu, je riešený pomocou autentifikovaného šifrovania s pripojenými dátami (AEAD). Pri prenose sú AEAD spolu s rozšírenou náhodnou hodnotou nonce aplikované na súbor cookie, pričom ako symetrický šifrovací kľúč slúži verejný kľúč odpovedajúceho rovesníka.

AEAD je použité aj pri riešení posledného spomínaného problému, tentoraz sa však pri šifrovaní cookie aplikuje AEAD pole s dodatočnými údajmi, z anglického (additional data). Dôvodom je pridanie ďalšej formy autentifikácie prvej MAC funkcie iniciačnej správy, ktorá vyvolala cookie reply správu. Zmyslom takéhoto riešenia je znemožniť útočníkovi posielat' prúdy neplatných cookie odpovedí iniciátorovi s cieľom zabrániť jeho autentifikácii pomocou platného cookie.

Po vyriešení všetkých spomínaných problémov prichádza na rad použitie druhej MAC funkcie (`msg.mac2`). Tá je pridaná za použitia preneseného cookie ako MAC kľúča. V čase, kedy je odpovedajúci rovesník pod vplyvom zaťaženia, bude prijímať iba správy navyše obsahujúce túto druhú MAC funkciu.

Teda po zhrnutí, rovesník v roli odpovedajúceho, po spočítaní MAC funkcií a ich následnom porovnaní s tými prijatými v správach, musí vždy odmietnuť správy obsahujúce neplatné prvé MAC funkcie (`msg.mac1`). Dodatočne, ak sa práve nachádza v procese, kedy je pod vplyvom zaťaženia, je možné, aby odmietol aj neplatné `msg.mac2`. Ak v tejto situácii (pri zaťažení) obdrží správu s platnou `msg.mac1`, ale neplatnou

`msg.mac2`, má možnosť odpovedať pomocou cookie reply správy. Schéma cookie reply správy je vysvetlená v 1.5.4. Konečným výsledkom takéhoto fungovania je zlepšenie cookie mechanizmov prítomných v DTLS [4] a IKEv2 [3].

## 1.5 Princíp správ

Pre WireGuard sú typické štyri druhy správ, pričom každá z nich má predponu jednobajtového indetifikátora typu `msg.type`. Ide o nasledovné správy:

- Iniciačná správa handshake – začína proces vytvárania bezpečného spojenia
- Odpoveď na iniciačnú správu handshake (handshake response) – ukončuje podanie rúk, po nej nasleduje možnosť vytvoriť bezpečné spojenie
- Odpoveď na jednu z uvedených správ v predošlých bodoch – posiela zašifrované cookie
- Zapúzdrený a zašifrovaný IP paket - prenášaný zabezpečeným spojením vyjednaným podaním rúk

Ďalej v rámci kapitoly budú popísané schémy jednotlivých správ a budú použité rozličné symboly a operátory – binárny operátor  $\parallel$  reprezentuje zreťazenie operandov, ďalej binárny operátor  $:=$  predstavuje priradenie pravého operandu k operandu ľavému. Iniciátor podania rúk je označený ako dolný index  $i$ , naopak označenie odpovedajúceho rovesníka je dolný index  $r$ . Ktorýkoľvek z nich je ďalej označený pomocou dolného indexu  $*$ . Pre správy, ktoré môže vytvárať ako iniciátor, tak aj odpovedajúci, platí, že  $(m, m') = (i, r)$ , ak správu vytvára rovesník v roli iniciátora a naopak, ak správu vytvára odpovedajúci rovesník, platí, že  $(m, m') = (r, i)$ . Oba z rovesníkov si lokálne spravujú aj niekoľko premenných, ktoré následne pri zostavovaní správ používajú, ide o [1]:

- $I_*$  32-bitový index, lokálne reprezentuje druhého rovesníka
- $S_*^{pub}, S_*^{priv}$  hodnoty pre statický verejný a statický súkromný kľúč
- $E_*^{pub}, E_*^{priv}$  hodnoty pre efemérny verejný a efemérny súkromný kľúč
- $Q$  hodnota nepovinného vopred zdieľaného symetrického kľúča; ak nie je prítomný mód vopred zdieľaných kľúčov, hodnota je nastavená na  $0^{32}$
- $H_*, C_*$  hodnota výsledku hashu, hodnota kľúča reťazenia

Ďalej je použitá anotácia  $\hat{n}$ , ktorá vracia hodnotu  $(n+16)$  – Poly1305 [8] autentifikačná značka pridaná k  $n$ . Symbolom  $\epsilon$  je vyjadrený prázdny bitový reťazec nulovej dĺžky,  $0^n$  predstavuje celý nulový ( $0 \times 0$ ) bitový reťazec dĺžky  $n$  bajtov a  $\rho^n$  reprezentuje náhodný bitový reťazec dĺžky  $n$  bajtov. Nakoniec nech je  $\tau$  dočasná premenná a  $\kappa$  dočasný šifrovací kľúč. Všetky hodnoty celých čísel sú typu little-endian. Dodatočne sú ešte použité nasledujúce funkcie a konštanty [1]:

- **DH(PRIVATE KEY, PUBLIC KEY)** – vracia 32-bitový výstup Curve25519 násobenia bodov súkromného a verejného kľúča
- **DH-GENERATE()** – generuje náhodný Curve25519 súkromný kľúč a k nemu odvodí verejný, vráti pár 32 bajtových hodnôt (PRIVATE, PUBLIC)
- **AEAD(KEY, COUNTER, PLAIN TEXT, AUTH TEXT)** – ChaCha20Poly1305 AEAD podľa RFC7539 [8] s nonce skladajúcej sa z 32 bitov núl nasledované 64-bitovou hodnotou COUNTER
- **XAEAD(KEY, NONCE, PLAIN TEXT, AUTH TEXT)** – XChaCha20Poly1305 AEAD s 24 bajtovou náhodnou NONCE, vytvorené pomocou HChaCha20 [23] a ChaCha20Poly1305 [8]
- **HASH(INPUT)** – BLAKE2S(INPUT, 32), vráti 32 bajtový výstup
- **MAC(KEY, INPUT)** – KEYED-BLAKE2S(KEY, INPUT, 16), kľúčovaná MAC verzia BLAKE2S funkcie, vracia 16 bajtový výstup
- **HMAC(KEY, INPUT)** – HMAC-BLAKE2S(KEY, INPUT, 32), obyčajná BLAKE2S hash funkcia, vracia 32 bajtový výstup
- **KDF<sub>n</sub>(KEY, INPUT)** – nastaví  $\tau_0 := \text{HMAC}(\text{KEY}, \text{INPUT})$ ,  $\tau_1 := \text{HMAC}(\tau_0, 0 \times 1)$ ,  $\tau_i := \text{HMAC}(\tau_0, \tau_{i-1} \parallel i)$  a vráti  $n$ -ticu 32 bajtových hodnôt  $(\tau_1, \dots, \tau_n)$ . Toto je HKDF [6] funkcia
- **TIMESTAMP()** – vracia TAI64N časovú značku [12] aktuálneho času (12 bajtový výstup), prvých 8 bajtov je big-endian celé číslo počtu sekúnd od 1970 TAI a posledné 4 bajty predstavujú big-endian celé číslo počtu nanosekúnd od začiatku tejto sekundy
- **CONSTRUCTION** – „Noise\_IKpsk2\_25519\_ChaChaPoly\_BLAKE2s“ UTF-8 reťazec, 37 bajtový výstup
- **IDENTIFIER** – „WireGuard v1 zx2c4 Jason@zx2c4.com“ UTF-8 reťazec, 34 bajtový výstup
- **LABEL-MAC1** – „mac1----“ UTF-8 reťazec, 8 bajtový výstup
- **LABEL-COOKIE** – „cookie--“ UTF- reťazec, 8 bajtový výstup

### 1.5.1 Správa iniciátor – príjemca

V tomto scenári sa snaží iniciátor naviazať spojenie pomocou správy msg [1]:

Tabuľka 1.5 Správa iniciátora príjemcovi

type:=0x1 (1 bajt)	reserved:=0 <sup>3</sup> (3 bajty)
sender:=I <sub>i</sub> (4 bajty)	
ephemeral (32 bajty)	
static (32 bajtov)	
timestamp (12 bajtov)	
mac1 (16 bajtov)	mac2 (16 bajtov)

Funkcie `mac1` a `mac2` budú vysvetlené neskôr v podkapitole 1.5.1.  $I_i$  je generované náhodne ( $\rho^4$ ) v čase odoslania správy a slúži k spojeniu nasledujúcich odpovedí s reláciou, ktorá začala touto správou. Zostávajúce funkcie sú vypočítané nasledovne [5]:

$$\begin{aligned}
 C_i &:= \text{HASH}(\text{CONSTRUCTION}) \\
 H_i &:= \text{HASH}(C_i \parallel \text{IDENTIFIER}) \\
 H_i &:= \text{HASH}(H_i \parallel S_r^{\text{pub}}) \\
 (E_i^{\text{priv}}, E_i^{\text{pub}}) &:= \text{DH-GENERATE}() \\
 C_i &:= \text{KDF}_1(C_i, E_i^{\text{pub}}) \\
 \text{msg.ephemeral} &:= E_i^{\text{pub}} \\
 H_i &:= \text{HASH}(H_i \parallel \text{msg.ephemeral}) \\
 (C_i, \kappa) &:= \text{KDF}_2(C_i, \text{DH}(E_i^{\text{priv}}, S_r^{\text{pub}})) \\
 \text{msg.static} &:= \text{AEAD}(\kappa, 0, S_i^{\text{pub}}, H_i) \\
 H_i &:= \text{HASH}(H_i \parallel \text{msg.static}) \\
 (C_i, \kappa) &:= \text{KDF}_2(C_i, \text{DH}(S_i^{\text{priv}}, S_r^{\text{pub}})) \\
 \text{msg.timestamp} &:= \text{AEAD}(\kappa, 0, \text{TIMESTAMP}(), H_i) \\
 H_i &:= \text{HASH}(H_i \parallel \text{msg.timestamp})
 \end{aligned}$$

Po prijatí takejto správy rovesník vykoná rovnaké operácie tak, aby boli jeho premenné konečného stavu totožné. Preto pri výpočtoch zamení operandov.

### 1.5.2 Správa príjemca naspäť iniciátorovi

Rovesník v roli odpovedajúceho posiela správu po spracovaní prvej, predom popísanej v 1.5.1. Aplikuje rovnaké operácie výpočtov, aby sa dostal do rovnakého stavu.  $I_r$  je generované náhodne ( $\rho^4$ ) v čase odoslania správy a slúži k spojeniu nasledujúcich odpovedí s reláciou ako tomu bolo v prvom prípade. Posiela takúto správu `msg` [1]:

Tabuľka 1.6 Správa od príjemcu naspäť iniciátorovi

<code>type:=0x2 (1 bajt)</code>	<code>reserved:=0<sup>3</sup> (3 bajty)</code>
<code>sender:=I<sub>r</sub> (4 bajty)</code>	<code>receiver:=I<sub>i</sub> (4 bajty)</code>
<code>ephemeral (32 bajtov)</code>	
<code>empty (0 bajtov)</code>	
<code>mac1 (16 bajtov)</code>	<code>mac2 (16 bajtov)</code>

Kde funkcie `mac1` a `mac2` sú vysvetlené v neskoršej časti kapitoly (1.5.3) a ostatné výpočty sú nasledovné [5]:

$$\begin{aligned}
(E_r^{priv}, E_r^{pub}) &:= \text{DH-GENERATE}() \\
C_r &:= \text{KDF}_1(C_r, E_r^{pub}) \\
\text{msg.ephemeral} &:= E_r^{pub} \\
H_r &:= \text{HASH}(H_r \parallel \text{msg.ephemeral}) \\
C_r &:= \text{KDF}_1(C_r, \text{DH}(E_r^{priv}, E_i^{pub})) \\
C_r &:= \text{KDF}_1(C_r, \text{DH}(E_r^{priv}, S_i^{pub})) \\
(C_r, \tau, \kappa) &:= \text{KDF}_3(C_r, Q) \\
H_r &:= \text{HASH}(H_r \parallel \tau) \\
\text{msg.empty} &:= \text{AEAD}(\kappa, 0, \epsilon, H_r) \\
H_r &:= \text{HASH}(H_r \parallel \text{msg.empty})
\end{aligned}$$

Znova, ako v predošlom prípade, iniciátor po obdržaní tejto správy uskutoční rovnaké operácie tak, aby boli jeho premenné konečného stavu totožné, čo dosiahne pozmenením operandov DH funkcie. Táto správa je veľkosťou menšia ako správa prvá.

### 1.5.3 Cookie funkcie MAC

V podkapitolách vysvetľujúcich správy, 1.5.1 a 1.5.2, sú prítomné dve funkcie,  $\text{msg.mac1}$  a  $\text{msg.mac2}$ . Pre konkrétnu handshake správu –  $\text{msg}_\alpha$  predstavuje všetky bajty správy  $\text{msg}$  pred  $\text{msg.mac1}$  a  $\text{msg}_\beta$  reprezentuje všetky bajty správy  $\text{msg}$  pred  $\text{msg.mac2}$ . Najposlednejšie obdržané cookie pred sekundami  $\tilde{L}_*$  je reprezentované  $L_*$ . Nakoniec, hodnota  $\text{HASH}(\text{LABEL-MAC1} \parallel S_{m'}^{pub})$  funkcie môže byť vopred vypočítaná. Vyplnenie týchto mac funkcií je nasledovné [1]:

$$\begin{aligned}
\text{msg.mac1} &:= \text{MAC}(\text{HASH}(\text{LABEL-MAC1} \parallel S_{m'}^{pub}), \text{msg}_\alpha) \\
\text{ak } L_m = \epsilon \text{ or } \tilde{L}_m \geq 120: \\
\text{msg.mac2} &:= 0^{16} \\
\text{inak:} \\
\text{msg.mac2} &:= \text{MAC}(L_m, \text{msg}_\beta)
\end{aligned}$$

### 1.5.4 Cookie reply správa

V prípadoch, kedy rovesník obdrží platnú prvú mac funkciu a následne neplatnú druhú mac funkciu a zároveň sa nachádza pod zaťažením, má možnosť poslať cookie reply správu. V takom prípade je  $L_{m'}$  určené podľa poľa  $\text{msg.sender}$  zo správy, ktorá túto cookie reply správu vyžiadala. Názorné zobrazenie [1]:

Tabuľka 1.7 Cookie Reply Správa

type:=0x3 (1 bajt)	reserved:=0 <sup>3</sup> (3 bajty)
receiver:=I <sub>m</sub> ' (4 bajty)	
nonce:=ρ <sup>24</sup> (24 bajtov)	
cookie (16 bajtov)	

Tajná náhodná premenná,  $R_m$ , mení svoju hodnotu každé dve minúty,  $A_m$  označuje zreťazenie vonkajšej IP zdrojovej adresy rovesníka označeného dolným indexom a UDP zdrojového portu.  $M$  je hodnota `msg.mac1` správy, na ktorú odpovedá. Zbytok cookie poľa sa vyplní takto[1]:

$$\tau := \text{MAC}(R_m, A_{m'})$$

$$\text{msg.cookie} := \text{XAEAD}(\text{HASH}(\text{LABEL-COOKIE} \parallel S_m^{\text{pub}}), \text{msg.nonce}, \tau, M)$$

Výsledkom je, že cookie reply správa je zviazaná s relevantnou správou, a tým je znemožnený akýkoľvek útok, kedy by boli rovesníkovi posielané podvodné cookie reply správy. V situácii, kedy je takáto správa platná, po jej prijatí si príjemca uloží cookie spolu s časom prijatia.

### 1.5.5 Transportné dáta – odvodenie kľúča a správy o transportných údajoch

Po ukončení výmeny spomínaných správ medzi iniciátorom a príjemcom dochádza k výpočtu kľúčov iniciátora a jeho odpovedajúceho pre odosielanie a prijímanie správ transportných dát.

WireGuard rovesníci si počas fázy podávania rúk vymenia medzi sebou Diffie-Hellman verejné kľúče a následne ešte uskutočnia postupnosť Diffie-Hellman operácií, kedy zahashujú výsledky do zdieľaného tajného kľúča. Po ukončení fázy podávania rúk je takto vytvorený tajný kľúč ďalej používaný pri odosielaní zašifrovaných transportných správ [5].

WireGuard používa koncept transportných kľúčov na zabezpečenie komunikácie medzi rovesníkmi. Tvorba týchto kľúčov spočíva v derivácii zo zdieľaných statických verejných kľúčov a efemérnych (dočasných) kľúčov relácie generovaných počas procesu podávania rúk. Takáto derivácia sa skladá z niekoľkých krokov [5]:

- Každá relácia má svoju konkrétnu krátkodobú dvojicu kľúčov pre každé podanie rúk
- Lokálny statický verejný kľúč, vzdialený statický verejný kľúč a dočasné verejné kľúče relácie sú skombinované čím vytvoria zdieľané tajomstvo



- Toto zdieľané tajomstvo je použité ako vstup do funkcie derivovania kľúčov (KDF), ktorej výstupom sú transportné kľúče použité pre šifrovanie a overovanie správnosti dát. KDF zabezpečuje, že čo i len minimálna zmena vstupnej hodnoty do výraznej miery pozmení hodnotu výstupu
- Nakoniec sú teda takto odvodené transportné kľúče použité k šifrovaniu a overovaniu prenášaných dát medzi rovesníkmi

## 2. POROVNANIE WIREGUARD S IPSEC A OPENVPN

Porovnávanie virtuálnych privátnych sietí WireGuard, IPsec a OpenVPN je v nasledujúcej časti vykonané na základe nimi používaných šifrovacích algoritmov a kryptografických primitív.

WireGuard je VPN protokol dosahujúci vysoké rýchlosti, pričom si stále zachováva svoju jednoduchosť a zároveň poskytuje efektívne a zabezpečené pripojenie. Aj napriek týmto kladným vlastnostiam je to protokol, ktorý má veľmi malú réžiu. Používa nástroje najmodernejšej kryptografie. Ide o relatívne novú technológiu, a preto nebola zatiaľ dôkladne preverená, ako napríklad OpenVPN. Za zmienku stojí aj fakt, že WireGuard je implementovaný do samotného linuxového jadra od verzie 5.6. V čase písania tejto práce nemá WireGuard žiadne známe závažnejšie zraniteľnosti.

Základným kameňom protokolu ju symetrická kryptografia postavená na ChaCha20 [8], ďalej je pre proces dohody o kľúčoch dôležitá eliptická krivka Diffie-Hellman (ECDH) (Curve25519 [2]). Hashovanie je v protokole zaobstarávané kryptografickou hashovacou funkciou BLAKE2s [9]. V neposlednom rade je využívaná funkcia SipHash24 [13], ktorá poskytuje kľúče pre hashovaciu tabuľku. Ďalej je použité podanie rúk (handshake) založené na UDP a samotná výmena kľúčov je zabezpečená doprednou bezpečnosťou. Nakoniec, odvodzovanie kľúčov je zaručené pomocou jednoduchej funkcie HKDF [6].

IPsec predstavuje sadu niekoľkých protokolov a v dnešnej dobe je považovaný za štandard v oblasti zabezpečenia internetovej komunikácie. Zabezpečuje dôvernosť a integritu autentifikácie. Rozsahovo sa jedná o zďaleka robustnejšiu kódovú základňu ako je tomu pri WireGuard, čo značne sťažuje jeho potrebný manažment a správu. Narozdiel od technológie WireGuard, IPsec mal časom objavené dokonca aj vyriešené niektoré chyby zabezpečenia. Patrila medzi ne napríklad zraniteľnosť spôsobujúca odmietnutie služby CVE-2018-5389 alebo zraniteľnosť umožňujúca útočníkom znížiť silu zabezpečenia pozmenením šifrovej súpravy, tiež známa ako FREAK útok.

IPsec ponúka rozsiahle množstvo použiteľných kryptografických a šifrovacích algoritmov, čo prináša aj riziko, že pri nedostatočne znalej implementácii môže prísť k nesprávnemu definovaniu algoritmu, ako je známe napríklad pre RSA [14] algoritmus (jeho 1024-bitové kľúče už tiež nie sú považované za bezpečné).

Protokol IPsec IKEv2 [3] ponúka širokú škálu možností implementácie kryptografických protokolov. Z nich ide napríklad o AES [15] spolu s ChaCha20-Poly1305 taktiež pre zaručenie dôvernosti. Naopak ako algoritmus výmeny kľúčov je v RFC3526 [16] špecifikovaný Diffie-Hellman [17] alebo v RFC4753 [18] algoritmus ECDH [19]. Nakoniec pre algoritmy autentifikácie je možné použiť, napríklad ECDSA [20] alebo PSK [21].

OpenVPN je open-source VPN protokol, ktorý nie je špecifikovaný žiadnymi štandardmi (RFC štandardy) a chýba mu dokonca aj kompatibilita s IPsec, no aj napriek

tomu je v dnešnej dobe značne rozšírený. Pre zabezpečenie komunikácie používa vlastnú sadu protokolov SSL/TLS pre výmenu kľúčov. Poskytuje úplnú diskretnosť, autentifikáciu a integritu. Štandardne prebieha komunikácia pomocou UDP, no je možné použiť aj TCP. Na šifrovanie sa používajú protokoly z knižnice OpenSSL [22]. Patria medzi ne napríklad AES [15], IKEv2 [3] s 256 bitovými kľúčmi.

System OpenVPN je tiež stabilný a rýchly najmä cez bezdrátové a mobilné siete alebo akékoľvek iné siete, ktorých spoľahlivosť nemusí byť vždy 100% a môže dochádzať ku stratám paketov. Pre takéto nespoľahlivé siete má OpenVPN implementovaný mód s použitím TCP namiesto UDP, čím je ale obetovaná časť jeho výkonu, čo je spôsobované neefektívnosťou zapúzdrovania TCP v rámci TCP.

## 3. TESTOVANIE

### 3.1 Metodika testovania

Predmetom testovania je vyhodnotenie výkonových parametrov technológií WireGuard, IPsec a OpenVPN a ich následné porovnanie. Testovanými parametrami teda budú – jitter, oneskorenie, dosahované rýchlosti a vyťaženie procesora

- **Jitter** – predstavuje zmeny veľkosti oneskorenia paketov v dôsledku zmien smerovania alebo chovania routeru
- **Oneskorenie** (latency) – časový rozdiel medzi pokynom k začatiu prenosu a samotným začatím prenosu

Testovanie bolo vykonané podľa nami zvolenej topológie siete (podkapitola 3.2), pričom ako nástroj použitý na meranie parametrov priepustnosti siete bol `iperf3`[24], do testovania bolo taktiež zakomponované aj sledovanie vyťaženie procesorov nami zvolených sieťových prvkov (podkapitola 3.3), pričom tento parameter bol meraný proprietárnym nástrojom spoločnosti Mikrotik, takzvaný *Profile*.

Nástroj `iperf3` podporuje ladenie parametrov týkajúcich sa časovania, vyrovnávacej pamäte a protokolov, ktoré nám dovoľia prispôsobiť test našim sieťovým prvkom a vlastnostiam siete. Z protokolov je možné použiť ako UDP, tak TCP, čo je pre naše testovanie postačujúce. Funguje na princípe zostavenia spojenia server-klient, kde zo strany klienta sú na stranu servera odosielané dáta a následne meria rozličné parametre, ktoré si môže sám testujúci špecifikovať podľa potreby. Samotné testy v našom prípade mali v priemere 10 minút.

Po zostavení testovacích scenárov, teda implementovaní konfigurácií virtuálnych privátnych sietí zodpovedajúcich nami zvolenej topológie siete (podkapitola 3.2) a následnom vykonaní konkrétnych testov na všetkých troch VPN protokoloch, budú získané výsledky medzi sebou porovnané. Porovnaním dostaneme lepšiu predstavu o výkonnosti meraných VPN technológií. Je avšak nutné poznamenať, že výsledky testov môžu byť ovplyvnené rozličnými faktormi. Jeden z faktorov predstavujú aj použité šifrovacie algoritmy, kedy v prípade protokolu IPsec máme na výber z viacerých možností. Nakoniec bude každý test vykonaný viac krát, aby nami získané výsledky mali čo najvyššiu mieru spoľahlivosti a výpovednej hodnoty.

Ako prvú sme otestovali virtuálnu privátnu sieť WireGuard, pretože práca sa z najväčšej časti venuje práve jej, kedy sme potom na základe výsledkov a pozorovaní vývoja testu v reálnom čase odvodili testovacie kritériá, ktorými sme sa riadili vo zvyšných testoch ostatných technológií.

Prvotne stanovené kritérium sa týkalo strátovosti paketov počas prenosu, kedy sme si ustanovili približnú hranicu 5-10 %. Od tohoto kritéria sa následným testovaním ďalej odvodilo kritérium veľkosti šírky pásma, kedy nedochádzalo k neprípustnému

zahadzovaniu paketov – takúto veľkosť predstavovala hodnota 60 Mbit/s pre oba použité sieťové prvky podľa podkapitoly 3.3.

## 3.2 Topológia siete – Site-to-site

Topológia siete použitá pri testovaní nami zadaných parametrov predstavovala situáciu tiež známu ako Site-to-site virtuálna privátna sieť. Takéto označenie predstavuje spojenie medzi viacerými sieťami, kedy by sa mohlo napríklad jednať o sieť spoločnosti, v ktorej spolupracuje viacero fyzicky vzdialených pobočiek, ktoré potrebujú medzi sebou navzájom komunikovať prostredníctvom siete Internet, pričom je potrebné, aby takto naviazané spojenie bolo bezpečné a dochádzalo k šifrovaniu takto vymieňaných dát.

Site-to-site virtuálne privátne siete sú taktiež bežne používané pre účely replikácie údajov a zálohovania dátových centier. Vytvorením bezpečných spojení medzi primárnymi a sekundárnymi dátovými centrami, je zaručené, že replikácia a zálohovanie kritických dát prebieha v reálnom čase, čo značne znižuje redundanciu dát a poskytuje možnosti obnovy po incidente.

Cloudové pripojenie: Pri pripájaní sa k poskytovateľom cloudových služieb môže sieť VPN takéhoto typu vytvoriť bezpečné a súkromné prepojenie medzi lokálnou a cloudovou infraštruktúrou.

Sieťová konektivita internetu vecí (IoT): Site-to-site v takomto prípade poskytuje zabezpečené pripojenie a následnú správu zariadení internetu vecí medzi rozličnými lokáciami. Ďalej je takto zabezpečená centralizovaná kontrola a monitorovanie týchto zariadení, pričom je dodržané utajenie a integrita prenášaných dát medzi zariadeniami aj samotným centrálnym riadiacim systémom.

V prípade nášho testovania teda išlo o scenár, kedy spájame dve lokálne siete s cieľom zabezpečiť ich vzájomnú komunikáciu naprieč Internetom, ktorá je bezpečná a šifrovaná.

## 3.3 Použité komponenty

Počas testovania technických parametrov spomínaných virtuálnych privátnych sietí boli použité sieťové prvky spoločnosti Mikrotik. Konkrétne sa jednalo o sieťové karty RB-493G a sieťové karty RB941-2nD. Na oboch komunikujúcich stranách sa vždy vyskytovali rovnaké sieťové prvky, teda išlo o site-to-site testovanie medzi RB-493G – RB493G a následne rovnako pre RB941-2nD.

Hardvérové špecifikácie nami použitých prvkov boli nasledovné:

- MikroTik RB941-2nD – router s dostatočným hardvérovým vybavením pre naše testy, ktorý disponuje štyrmi 10/100 Mbit/s ethernetovými portami, výkonným 650 MHz procesorom a 32MB RAM

- MikroTik RB-493G – prenosové rýchlosti 10/100/1000 Mbit/s, 1 jadrový procesor s frekvenciou 680 MHz a 256 MB pamäťou RAM

Testovanie bolo uskutočnené na vytvorenej virtuálnej privátnej sieti medzi dvoma lokálnymi sieťami, kedy v jednej z nich bol používateľ reprezentovaný počítačom s operačným systémom Windows 10, pričom v druhej lokálnej sieti išlo o používateľa s operačným systémom Debian Linux.

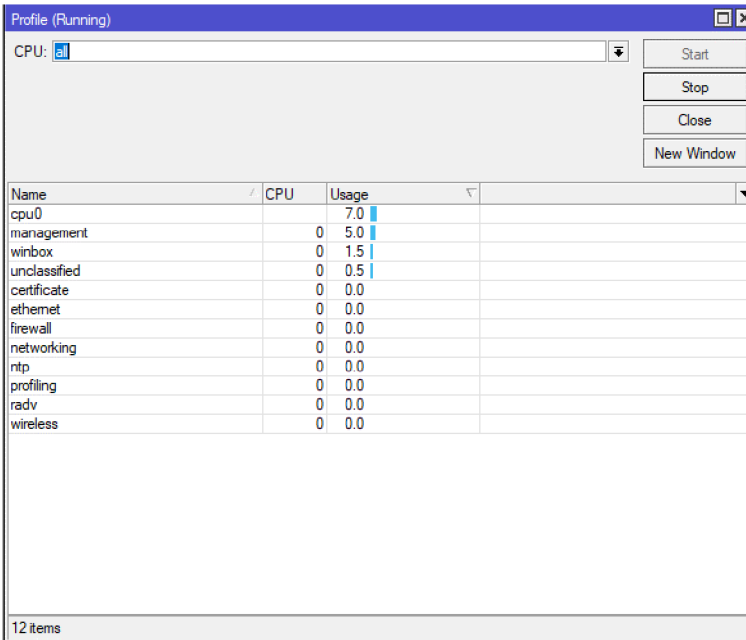
MikroTik RB-493G sa nachádzal počas testovania vo verzii 7.9, pričom bola použitá aj takzvaná Fasttrack funkcia [25], ktorej význam spočíva v znížení zaťaženia procesora zariadenia MikroTik, pričom sa teoreticky zlepšuje využitie dostupnej šírky pásma. Na druhej strane, zariadenia MikroTik RB941-2nD, boli použité vo verzii 7.5, taktiež s nakonfigurovanou funkciou Fasttrack.

## 4. VÝSLEDKY TESTOVANÍ TECHNICKÝCH PARAMETROV VPN

Najprv budú v rámci kapitoly predstavené výsledky testovania s použitím Mikrotik zariadení RB-493G pre konkrétne virtuálne privátne siete. Uvedené budú taktiež aj výsledky testovania, kedy pre komunikáciu medzi lokálnymi sieťami v našej topológii nebola použitá žiadna virtuálna privátna sieť a komunikácia bola sprostredkovaná len pomocou statického smerovania medzi našimi sieťovými prvkami.

Ďalej v rámci kapitoly budú predstavené výsledky týkajúce sa sieťových prvkov Mikrotik RB-941-2nD s rovnakými kritériami (popis ich stanovenia v podkapitole 3.1) ako tomu bolo pri RB-493G.

### 4.1 Testovanie bez použitia virtuálnej privátnej siete – statické smerovanie medzi lokálnymi sieťami (RB-493G)



Name	CPU	Usage
cpu0		7.0
management	0	5.0
winbox	0	1.5
unclassified	0	0.5
certificate	0	0.0
ethernet	0	0.0
firewall	0	0.0
networking	0	0.0
ntp	0	0.0
profiling	0	0.0
radv	0	0.0
wireless	0	0.0

Obrázok 4.1 Zaťaženie procesora Mikrotik RB-493G, statické smerovanie

Testovanie vlastností siete a taktiež hardvérového zaťaženia pri použití nástroja `iperf3` pri použití statického smerovania dovoľujúce komunikáciu medzi lokálnymi sieťami v našej testovacej topológii. Na obrázku 4.1 môžeme vidieť, že pri konfigurácii so statickým smerovaním dochádza len k minimálnemu zaťaženiu procesora nami použitého

sieťového prvku Mikrotik RB-493G. Toto zaťaženie sa počas celého testu pohybovalo len v rozmedzí do 10 % s občasnými výstupmi až na 80 % celkového zaťaženia na strane príjemcu komunikácie, iperf3 servera. Na strane odosielajúceho, teda takzvaného iperf3 klienta, dochádzalo len k minimálnemu navýšeniu zaťaženia procesora oproti strane prijímajúceho, kedy občasné výkyvy predstavovali až 85 %.

```

[ 5] 49.00-50.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.097 ms 0/5137 (0%)
[ 5] 50.00-51.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.104 ms 0/5137 (0%)
[ 5] 51.00-52.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.098 ms 0/5137 (0%)
[ 5] 52.00-53.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.111 ms 0/5137 (0%)
[ 5] 53.00-54.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.092 ms 0/5137 (0%)
[ 5] 54.00-55.00 sec 7.14 Mbytes 59.9 Mbits/sec 0.225 ms 0/5127 (0%)
[ 5] 55.00-56.00 sec 7.17 Mbytes 60.1 Mbits/sec 0.117 ms 0/5147 (0%)
[ 5] 56.00-57.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.094 ms 0/5137 (0%)
[ 5] 57.00-58.00 sec 6.91 Mbytes 57.9 Mbits/sec 1.090 ms 0/4965 (0%)
[ 5] 58.00-59.00 sec 7.39 Mbytes 62.1 Mbits/sec 0.096 ms 0/5309 (0%)
[ 5] 59.00-60.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.096 ms 0/5137 (0%)
[ 5] 60.00-61.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.127 ms 0/5137 (0%)
[ 5] 61.00-62.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.096 ms 0/5137 (0%)
[ 5] 62.00-63.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.098 ms 0/5137 (0%)
[ 5] 63.00-64.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.104 ms 0/5135 (0%)
[ 5] 64.00-65.00 sec 7.16 Mbytes 60.0 Mbits/sec 0.092 ms 0/5139 (0%)
[ 5] 65.00-66.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.103 ms 0/5138 (0%)
[ 5] 66.00-67.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.086 ms 0/5136 (0%)
[ 5] 67.00-68.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.093 ms 0/5137 (0%)
[ 5] 68.00-69.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.091 ms 0/5137 (0%)
[ 5] 69.00-70.00 sec 7.13 Mbytes 59.8 Mbits/sec 0.097 ms 13/5137 (0.35%)
[ 5] 70.00-71.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.100 ms 0/5137 (0%)
[ 5] 71.00-72.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.101 ms 0/5137 (0%)
[ 5] 72.00-73.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.090 ms 0/5137 (0%)
[ 5] 73.00-74.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.090 ms 0/5137 (0%)
[ 5] 74.00-75.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.093 ms 0/5137 (0%)
[ 5] 75.00-76.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.093 ms 0/5137 (0%)
[ 5] 76.00-77.00 sec 7.13 Mbytes 59.8 Mbits/sec 0.104 ms 13/5137 (0.25%)
[ 5] 77.00-78.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.108 ms 0/5136 (0%)
[ 5] 78.00-79.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.152 ms 0/5138 (0%)
[ 5] 79.00-80.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.098 ms 0/5138 (0%)
[ 5] 80.00-81.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.101 ms 0/5136 (0%)
[ 5] 81.00-82.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.091 ms 0/5137 (0%)
[ 5] 82.00-83.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.098 ms 0/5137 (0%)
[ 5] 83.00-84.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.111 ms 0/5136 (0%)
[ 5] 84.00-85.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.095 ms 0/5138 (0%)
[ 5] 85.00-86.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.107 ms 0/5137 (0%)
[ 5] 86.00-87.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.135 ms 0/5137 (0%)
[ 5] 87.00-88.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.100 ms 0/5137 (0%)
[ 5] 88.00-89.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.100 ms 0/5137 (0%)
[ 5] 89.00-90.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.091 ms 0/5137 (0%)
[ 5] 90.00-91.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.102 ms 0/5135 (0%)
[ 5] 91.00-92.00 sec 7.14 Mbytes 59.7 Mbits/sec 0.356 ms 0/5128 (0%)
[ 5] 92.00-93.00 sec 7.17 Mbytes 60.3 Mbits/sec 0.096 ms 0/5148 (0%)
[ 5] 93.00-94.00 sec 7.15 Mbytes 60.0 Mbits/sec 0.094 ms 0/5137 (0%)
[ 5] 94.00-94.45 sec 3.23 Mbytes 59.9 Mbits/sec 0.098 ms 0/2323 (0%)
-----
[ ID] Interval      Transfer      Bitrate      Jitter      Lost/Total Datagrams
[ 5] 0.00-94.45 sec 675 Mbytes 60.0 Mbits/sec 0.098 ms 93/485194 (0.019%) receiver
iperf3: interrupt - the server has terminated

```

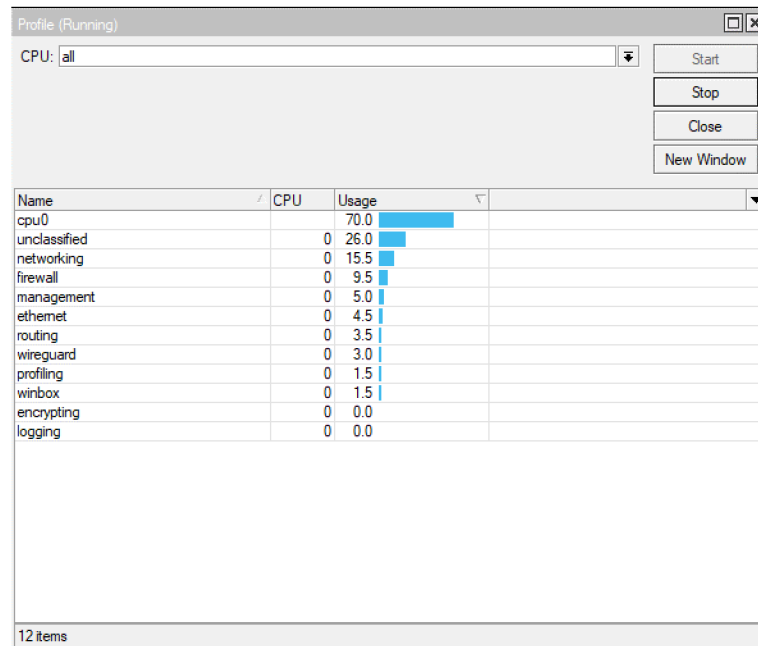
Obrázok 4.2 Iperf3 ukážka výstupu testu, statické smerovanie

V prípade statického smerovania a jeho testovania pomocou nástroja iperf3, kedy maximálna šírka pásma predstavovala 60 Mbit/s podľa nami stanovených kritérií (podkapitola 3.1), sme obdržali výsledky vypovedajúce o takmer 100% využití dostupnej šírky pásma.

Iperf3 test bol v takomto testovacom scenári v priemere schopný dosahovať až 59,6 Mbit/s prenosové rýchlosti. V prípade parametra jitter sa jednalo o priemerné hodnoty 0,096 milisekúnd. Nakoniec priemerné hodnoty parametra oneskorenia predstavovali 2,156 milisekúnd.



## 4.2 Testovanie s použitím WireGuard virtuálnej privátnej siete (RB-493G)



Obrázok 4.3 Zaťaženie procesora Mikrotik RB-493G, WireGuard

Ďalšie z testov boli uskutočnené na implementácii virtuálnej privátnej siete WireGuard medzi dvoma lokálnymi sieťami. Počas testovania dochádzalo k prípustným hodnotám zahadzovania datagramov `iperf3` testu (vysvetlené v podkapitole 3.1) pri špecifikovaní šírky pásma na 60 Mbit/s.

V takomto scenári sme sa stretli s vyťažovaním procesora na strane prijímajúceho v hodnotách okolo 70 %, kde proces *unclassified* a *wireguard* (na oboch stranách komunikácie) z toho dokopy tvorili okolo 30 %, počas celého trvania testovania. Na strane odosielajúceho (klient) potom išlo až o 100% vyťaženie procesora Mikrotiku.

```

Príkazový riadok - iperf3 -s
[ 5] 566.00-567.00 sec 6.46 MBytes 53.9 Mbits/sec 0.240 ms 333/5243 (6.4%)
[ 5] 567.00-568.01 sec 6.83 MBytes 57.0 Mbits/sec 1.371 ms 169/5356 (3.2%)
[ 5] 568.01-569.00 sec 7.18 MBytes 60.8 Mbits/sec 0.222 ms 244/5701 (4.3%)
[ 5] 569.00-570.00 sec 6.69 MBytes 56.1 Mbits/sec 0.213 ms 352/5435 (6.5%)
[ 5] 570.00-571.00 sec 6.96 MBytes 58.4 Mbits/sec 0.274 ms 146/5432 (2.7%)
[ 5] 571.00-572.00 sec 6.74 MBytes 56.5 Mbits/sec 0.234 ms 317/5439 (5.8%)
[ 5] 572.00-573.00 sec 6.40 MBytes 53.8 Mbits/sec 0.231 ms 570/5433 (10%)
[ 5] 573.00-574.00 sec 6.01 MBytes 50.2 Mbits/sec 0.556 ms 753/5317 (14%)
[ 5] 574.00-575.00 sec 7.07 MBytes 59.6 Mbits/sec 0.337 ms 104/5478 (1.9%)
[ 5] 575.00-576.00 sec 6.89 MBytes 57.6 Mbits/sec 0.423 ms 271/5506 (4.9%)
[ 5] 576.00-577.00 sec 6.95 MBytes 58.5 Mbits/sec 0.186 ms 165/5445 (3%)
[ 5] 577.00-578.00 sec 6.81 MBytes 57.1 Mbits/sec 0.206 ms 262/5435 (4.8%)
[ 5] 578.00-579.00 sec 6.79 MBytes 57.0 Mbits/sec 0.295 ms 271/5429 (5%)
[ 5] 579.00-580.00 sec 6.43 MBytes 54.0 Mbits/sec 0.196 ms 545/5434 (10%)
[ 5] 580.00-581.01 sec 6.55 MBytes 54.6 Mbits/sec 0.206 ms 161/5141 (3.1%)
[ 5] 581.01-582.00 sec 6.76 MBytes 57.1 Mbits/sec 0.250 ms 597/5733 (10%)
[ 5] 582.00-583.00 sec 6.34 MBytes 53.1 Mbits/sec 0.223 ms 623/5437 (11%)
[ 5] 583.00-584.00 sec 6.72 MBytes 56.3 Mbits/sec 0.246 ms 330/5433 (6.1%)
[ 5] 584.00-585.00 sec 6.35 MBytes 53.2 Mbits/sec 0.179 ms 550/5372 (10%)
[ 5] 585.00-586.00 sec 6.59 MBytes 55.3 Mbits/sec 0.274 ms 477/5486 (8.7%)
[ 5] 586.00-587.00 sec 6.82 MBytes 57.2 Mbits/sec 0.231 ms 236/5415 (4.4%)
[ 5] 587.00-588.00 sec 6.40 MBytes 53.6 Mbits/sec 0.253 ms 186/5047 (3.7%)
[ 5] 588.00-589.00 sec 6.51 MBytes 54.7 Mbits/sec 0.210 ms 668/5612 (12%)
[ 5] 589.00-590.00 sec 6.46 MBytes 54.2 Mbits/sec 0.240 ms 752/5663 (13%)
[ 5] 590.00-591.00 sec 6.84 MBytes 57.4 Mbits/sec 0.205 ms 209/5410 (3.9%)
[ 5] 591.00-592.00 sec 6.99 MBytes 58.6 Mbits/sec 0.192 ms 95/5410 (1.8%)
[ 5] 592.00-593.00 sec 6.77 MBytes 56.8 Mbits/sec 0.246 ms 324/5467 (5.9%)
[ 5] 593.00-594.01 sec 6.63 MBytes 55.3 Mbits/sec 0.848 ms 312/5351 (5.8%)
[ 5] 594.01-595.01 sec 7.00 MBytes 58.3 Mbits/sec 1.952 ms 51/5372 (0.95%)
[ 5] 595.01-596.00 sec 6.93 MBytes 58.9 Mbits/sec 0.324 ms 267/5529 (4.8%)
[ 5] 596.00-597.00 sec 6.55 MBytes 55.0 Mbits/sec 0.251 ms 466/5445 (8.6%)
[ 5] 597.00-598.00 sec 6.74 MBytes 56.6 Mbits/sec 0.514 ms 262/5385 (4.9%)
[ 5] 598.00-599.00 sec 6.76 MBytes 56.7 Mbits/sec 0.230 ms 380/5518 (6.9%)
[ 5] 599.00-600.00 sec 6.41 MBytes 53.7 Mbits/sec 0.223 ms 544/5412 (10%)
[ 5] 600.00-600.01 sec 99.7 KBytes 111 Mbits/sec 0.179 ms 0/74 (0%)
-----
[ ID] Interval          Transfer      Bitrate      Jitter      Lost/Total Datagrams
[SUM] 0.0-600.0 sec 22 datagrams received out-of-order
[ 5] 0.00-600.01 sec 3.94 GBytes 56.4 Mbits/sec 0.179 ms 195809/3260865 (6%) receiver

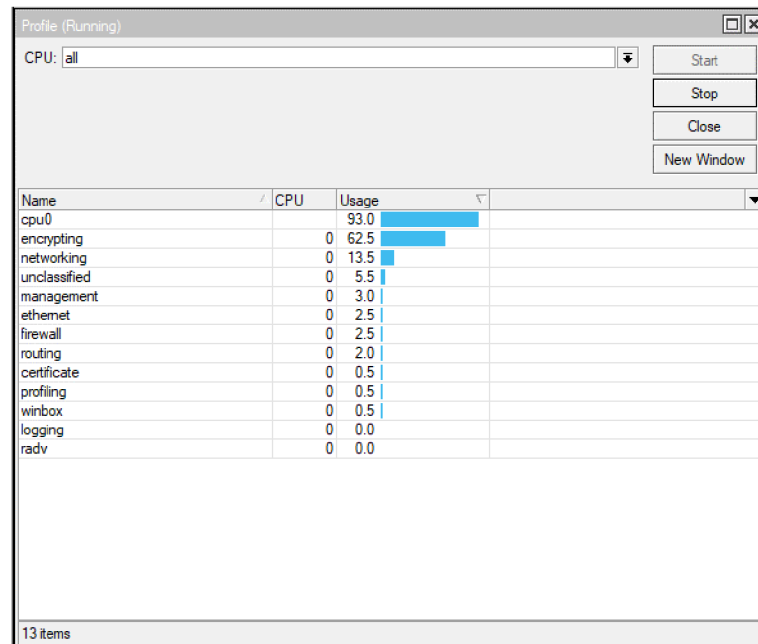
```

Obrázok 4.4 Iperf3 výstup, WireGuard

Po skončení testovania technológie WireGuard sme obdržali výsledky, kde prijímajúca strana oznamuje priemerné hodnoty prenosovej rýchlosti vo výške až 56,4 Mbit/s, pričom priemerná hodnota na strane odosielajúceho predstavovala 59 Mbit/s.

Priemerná hodnota parametra jitter nakoniec bola 0,179 milisekúnd a v prípade meraného oneskorenia sa jednalo o priemernú hodnotu 2,917 milisekúnd.

### 4.3 Testovanie s použitím IPsec virtuálnej privátnej siete (RB-493G)



Obrázok 4.5 Zaťaženie procesora Mikrotik RB-493G, IPsec

Šifrovací algoritmus technológie IPsec použitý počas nášho testovania bol AES-256-cbc [15], ako hashovaciu funkciu sme zvolili sha2-256 bit [27]. Diffie-Hellman grupa 14 [26] a IKEv2 [3] návrh boli zvolené ako zvyšné parametre. Bolo tiež zvolené overovanie pomocou digitálneho podpisu s využitím certifikátov pre obe strany.

Pri takejto konfigurácii našej virtuálnej privátnej siete IPsec dochádzalo k zaťaženiu procesora Mikrotiku na strane prijímajúceho v medziach 90-100 %, kedy značnú časť z toho predstavoval proces šifrovania *encrypting* (obrázok 4.5). Na strane odosielajúceho sme potom mohli sledovať vyťaženie procesora neustále na 100 %. Ako tomu bolo aj v prípade prijímajúceho, najväčší podiel na zaťažení mal proces šifrovania.

```

Príkazový riadok - iperf3 -s
[ 5] 558.00-559.00 sec 2.20 MBytes 18.5 Mbits/sec 0.874 ms 3682/5334 (69%)
[ 5] 559.00-560.00 sec 2.27 MBytes 19.1 Mbits/sec 1.350 ms 3601/5306 (68%)
[ 5] 560.00-561.00 sec 2.23 MBytes 18.8 Mbits/sec 0.642 ms 3766/5442 (69%)
[ 5] 561.00-562.00 sec 2.19 MBytes 18.4 Mbits/sec 0.886 ms 3722/5367 (69%)
[ 5] 562.00-563.00 sec 2.21 MBytes 18.5 Mbits/sec 0.764 ms 3706/5361 (69%)
[ 5] 563.00-564.01 sec 2.21 MBytes 18.4 Mbits/sec 1.240 ms 3712/5367 (69%)
[ 5] 564.01-565.00 sec 2.20 MBytes 18.5 Mbits/sec 0.941 ms 3539/5188 (68%)
[ 5] 565.00-566.01 sec 2.20 MBytes 18.3 Mbits/sec 3.155 ms 3648/5295 (69%)
[ 5] 566.01-567.00 sec 2.29 MBytes 19.3 Mbits/sec 1.164 ms 3901/5622 (69%)
[ 5] 567.00-568.01 sec 2.19 MBytes 18.3 Mbits/sec 1.712 ms 3692/5335 (69%)
[ 5] 568.01-569.00 sec 2.23 MBytes 18.7 Mbits/sec 1.946 ms 3782/5452 (69%)
[ 5] 569.00-570.00 sec 2.21 MBytes 18.5 Mbits/sec 1.847 ms 3671/5327 (69%)
[ 5] 570.00-571.00 sec 2.21 MBytes 18.6 Mbits/sec 1.957 ms 3502/5161 (68%)
[ 5] 571.00-572.00 sec 2.27 MBytes 19.0 Mbits/sec 0.826 ms 3815/5517 (69%)
[ 5] 572.00-573.00 sec 2.19 MBytes 18.4 Mbits/sec 0.885 ms 3729/5372 (69%)
[ 5] 573.00-574.00 sec 2.19 MBytes 18.4 Mbits/sec 1.247 ms 3730/5376 (69%)
[ 5] 574.00-575.00 sec 2.19 MBytes 18.4 Mbits/sec 0.684 ms 3675/5321 (69%)
[ 5] 575.00-576.00 sec 2.20 MBytes 18.5 Mbits/sec 0.765 ms 3721/5373 (69%)
[ 5] 576.00-577.00 sec 2.21 MBytes 18.5 Mbits/sec 2.183 ms 3593/5247 (68%)
[ 5] 577.00-578.01 sec 2.23 MBytes 18.6 Mbits/sec 3.039 ms 3704/5374 (69%)
[ 5] 578.01-579.00 sec 2.26 MBytes 19.1 Mbits/sec 1.326 ms 3827/5525 (69%)
[ 5] 579.00-580.00 sec 2.22 MBytes 18.7 Mbits/sec 0.895 ms 3690/5358 (69%)
[ 5] 580.00-581.00 sec 2.21 MBytes 18.5 Mbits/sec 0.831 ms 3705/5359 (69%)
[ 5] 581.00-582.00 sec 2.20 MBytes 18.5 Mbits/sec 0.873 ms 3664/5316 (69%)
[ 5] 582.00-583.01 sec 2.24 MBytes 18.7 Mbits/sec 0.948 ms 3712/5395 (69%)
[ 5] 583.01-584.01 sec 2.21 MBytes 18.5 Mbits/sec 0.775 ms 3704/5364 (69%)
[ 5] 584.01-585.01 sec 2.18 MBytes 18.4 Mbits/sec 1.372 ms 3753/5391 (70%)
[ 5] 585.01-586.00 sec 2.22 MBytes 18.7 Mbits/sec 0.687 ms 3677/5343 (69%)
[ 5] 586.00-587.01 sec 2.19 MBytes 18.2 Mbits/sec 1.088 ms 3733/5375 (69%)
[ 5] 587.01-588.00 sec 2.26 MBytes 19.0 Mbits/sec 0.827 ms 3577/5275 (68%)
[ 5] 588.00-589.00 sec 2.17 MBytes 18.2 Mbits/sec 1.311 ms 3659/5283 (69%)
[ 5] 589.00-590.00 sec 2.18 MBytes 18.2 Mbits/sec 1.524 ms 3707/5339 (69%)
[ 5] 590.00-591.02 sec 1.91 MBytes 15.8 Mbits/sec 1.072 ms 4043/5479 (74%)
[ 5] 591.02-592.00 sec 2.22 MBytes 18.9 Mbits/sec 1.447 ms 3715/5380 (69%)
[ 5] 592.00-593.00 sec 2.24 MBytes 18.8 Mbits/sec 0.915 ms 3579/5256 (68%)
[ 5] 593.00-594.00 sec 2.28 MBytes 19.1 Mbits/sec 1.182 ms 3840/5548 (69%)
[ 5] 594.00-595.00 sec 2.22 MBytes 18.6 Mbits/sec 0.791 ms 3699/5362 (69%)
[ 5] 595.00-596.00 sec 2.20 MBytes 18.5 Mbits/sec 0.699 ms 3709/5360 (69%)
[ 5] 596.00-597.00 sec 2.19 MBytes 18.3 Mbits/sec 1.098 ms 3713/5354 (69%)
[ 5] 597.00-598.00 sec 2.22 MBytes 18.7 Mbits/sec 0.663 ms 3515/5182 (68%)
[ 5] 598.00-599.00 sec 2.29 MBytes 19.2 Mbits/sec 0.723 ms 3850/5564 (69%)
[ 5] 599.00-600.00 sec 2.21 MBytes 18.5 Mbits/sec 0.798 ms 3707/5364 (69%)
[ 5] 600.00-600.06 sec 152 KBytes 19.7 Mbits/sec 0.789 ms 262/373 (70%)

ID Interval Transfer Bitrate Jitter Lost/Total Datagrams
SUM 0.0-600.1 sec 2 datagrams received out-of-order
[ 5] 0.00-600.06 sec 1.29 GBytes 18.5 Mbits/sec 0.789 ms 2225414/3218879 (69%) receiver

Server listening on 5201 (test #3)

```

Obrázok 4.6 Iperf3 výstup, IPsec

Konečné výsledky testovania takejto konfigurácie VPN IPsec predstavovali v priemere 18,5 Mbit/s prenosové rýchlosti na strane prijímajúceho, pričom strana odosielajúceho oznamovala priemer 59 Mbit/s.

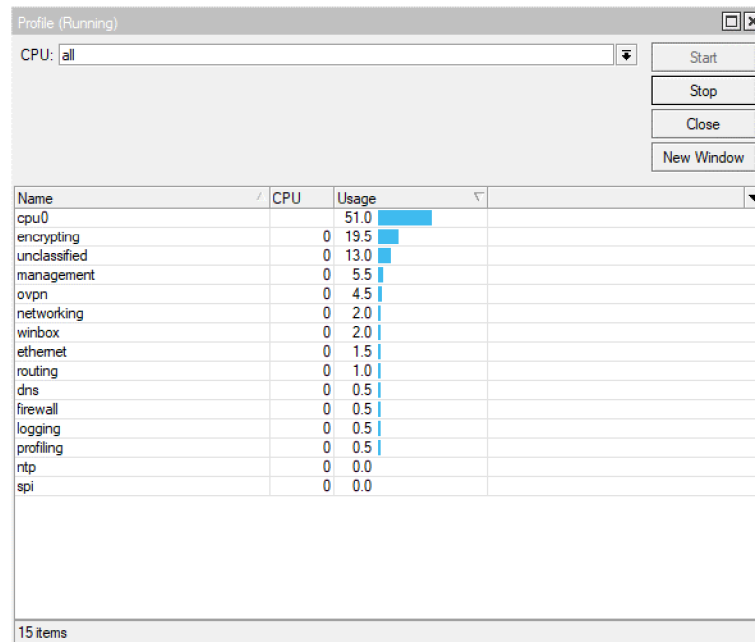
Interface List												
Interface	Interface List	Ethernet	EoIP Tunnel	IP Tunnel	GRE Tunnel	VLAN	VXLAN	VRRP	VETH	MACsec	Bonding	LTE
R	ether1	Ethernet					20.6 Mbps			0 bps	1 698	0
R	ether2	Ethernet				130.9 kbps				62.4 Mbps	15	5 411

Obrázok 4.7 Bottleneck procesora Mikrotik, IPsec odosielajúci

V tomto prípade sme sa teda stretli s bottleneckom na procesore Mikrotika (obrázok 4.7) zo strany odosielajúceho, kedy nebol schopný podobných 59 Mbit/s šifrovať a zároveň prenášať na druhú stranu. Toto sa odzrkadlilo aj na výslednej priemernej hodnote obdržaných datagramov prijímajúcim, kedy išlo o hodnotu 31 %.

Oznámená priemerná hodnota parametra jitter na konci testu – 0,789 milisekúnd.  
Nameraná priemerná hodnota oneskorenia v takomto scenári predstavovala hodnotu 3,492 milisekúnd.

#### 4.4 Testovanie s použitím OpenVPN virtuálnej privátnej siete (RB-493G)



Obrázok 4.8 Zaťaženie procesora Mikrotik RB-493G, OpenVPN

Algoritmy systému OpenVPN použité počas testovania boli AES-256 [15] pre šifrovanie a ako protokol overovania sme zvolili sha2-256 bit [27].

Rovnako ako v prípade testovania technológie IPsec, aj pri tomto testovaní sa opäť stretávame s bottleneckom zo strany procesora Mikrotiku odosielajúceho. Zatiaľ čo na strane prijímajúceho sa vytťaženie pohybuje v priemere okolo 50 %, strana odosielajúceho zaznamenáva neustále vytťaženie 100 %. Teda tento procesor nie je schopný zároveň šifrovať a ďalej posielat' dostupnou maximálnou rýchlosťou.

```

Príkazový riadok - iperf3 -s
5] 557.01-558.00 sec 777 KBytes 6.38 Mbits/sec 1.890 ms 4486/5031 (89%)
5] 558.00-559.00 sec 759 KBytes 6.22 Mbits/sec 1.831 ms 4753/5285 (90%)
5] 559.00-560.00 sec 777 KBytes 6.36 Mbits/sec 1.231 ms 4323/4868 (89%)
5] 560.00-561.00 sec 753 KBytes 6.17 Mbits/sec 4.594 ms 4940/5468 (90%)
5] 561.00-562.00 sec 811 KBytes 6.67 Mbits/sec 1.193 ms 4292/4861 (88%)
5] 562.00-563.00 sec 821 KBytes 6.71 Mbits/sec 0.886 ms 4264/4840 (88%)
5] 563.00-564.00 sec 827 KBytes 6.77 Mbits/sec 1.697 ms 4971/5551 (90%)
5] 564.00-565.01 sec 776 KBytes 6.30 Mbits/sec 1.042 ms 4486/5030 (89%)
5] 565.01-566.01 sec 767 KBytes 6.32 Mbits/sec 1.582 ms 4820/5358 (90%)
5] 566.01-567.00 sec 741 KBytes 6.09 Mbits/sec 2.302 ms 4611/5131 (90%)
5] 567.00-568.02 sec 790 KBytes 6.40 Mbits/sec 1.013 ms 4444/4998 (89%)
5] 568.02-569.01 sec 773 KBytes 6.37 Mbits/sec 1.378 ms 4548/5090 (89%)
5] 569.01-570.01 sec 733 KBytes 6.00 Mbits/sec 1.880 ms 4807/5321 (90%)
5] 570.01-571.00 sec 855 KBytes 7.04 Mbits/sec 1.592 ms 4559/5159 (88%)
5] 571.00-572.01 sec 761 KBytes 6.20 Mbits/sec 1.113 ms 4111/4645 (89%)
5] 572.01-573.00 sec 840 KBytes 6.94 Mbits/sec 1.083 ms 4851/5440 (89%)
5] 573.00-574.01 sec 810 KBytes 6.60 Mbits/sec 0.825 ms 4672/5240 (89%)
5] 574.01-575.00 sec 756 KBytes 6.21 Mbits/sec 0.788 ms 4400/4930 (89%)
5] 575.00-576.01 sec 840 KBytes 6.84 Mbits/sec 0.730 ms 4723/5312 (89%)
5] 576.01-577.01 sec 820 KBytes 6.74 Mbits/sec 0.760 ms 4483/5058 (89%)
5] 577.01-578.01 sec 766 KBytes 6.27 Mbits/sec 2.509 ms 4716/5253 (90%)
5] 578.01-579.00 sec 817 KBytes 6.73 Mbits/sec 0.820 ms 4357/4930 (88%)
5] 579.00-580.00 sec 757 KBytes 6.20 Mbits/sec 2.777 ms 4822/5353 (90%)
5] 580.00-581.01 sec 808 KBytes 6.54 Mbits/sec 2.438 ms 4221/4788 (88%)
5] 581.01-582.00 sec 820 KBytes 6.79 Mbits/sec 1.369 ms 4807/5382 (89%)
5] 582.00-583.01 sec 798 KBytes 6.51 Mbits/sec 0.696 ms 4360/4920 (89%)
5] 583.01-584.00 sec 777 KBytes 6.40 Mbits/sec 1.913 ms 4813/5358 (90%)
5] 584.00-585.02 sec 827 KBytes 6.67 Mbits/sec 1.640 ms 4559/5139 (89%)
5] 585.02-586.00 sec 808 KBytes 6.72 Mbits/sec 1.579 ms 4751/5318 (89%)
5] 586.00-587.01 sec 781 KBytes 6.35 Mbits/sec 1.024 ms 4410/4958 (89%)
5] 587.01-588.01 sec 808 KBytes 6.64 Mbits/sec 0.798 ms 4565/5132 (89%)
5] 588.01-589.00 sec 808 KBytes 6.68 Mbits/sec 0.640 ms 4601/5168 (89%)
5] 589.00-590.01 sec 808 KBytes 6.59 Mbits/sec 1.244 ms 4425/4992 (89%)
5] 590.01-591.01 sec 817 KBytes 6.65 Mbits/sec 1.825 ms 4581/5154 (89%)
5] 591.01-592.00 sec 817 KBytes 6.77 Mbits/sec 0.816 ms 4637/5210 (89%)
5] 592.00-593.01 sec 816 KBytes 6.04 Mbits/sec 0.826 ms 4566/5138 (89%)
5] 593.01-594.02 sec 753 KBytes 6.13 Mbits/sec 1.321 ms 4594/5122 (90%)
5] 594.02-595.00 sec 840 KBytes 6.96 Mbits/sec 1.339 ms 4725/5314 (89%)
5] 595.00-596.01 sec 749 KBytes 6.10 Mbits/sec 2.291 ms 4420/4945 (89%)
5] 596.01-597.01 sec 771 KBytes 6.30 Mbits/sec 1.454 ms 4747/5288 (90%)
5] 597.01-598.00 sec 761 KBytes 6.30 Mbits/sec 1.057 ms 4459/4993 (89%)
5] 598.00-599.01 sec 801 KBytes 6.53 Mbits/sec 1.624 ms 4678/5240 (89%)
5] 599.01-600.01 sec 704 KBytes 5.74 Mbits/sec 1.461 ms 4165/4659 (89%)
5] 600.01-600.22 sec 92.7 KBytes 3.67 Mbits/sec 0.731 ms 530/595 (89%)
-----
ID Interval Transfer Bitrate Jitter Lost/Total Datagrams
5] 0.00-600.22 sec 454 MBytes 6.35 Mbits/sec 0.731 ms 2755414/3081616 (89%) receiver

```

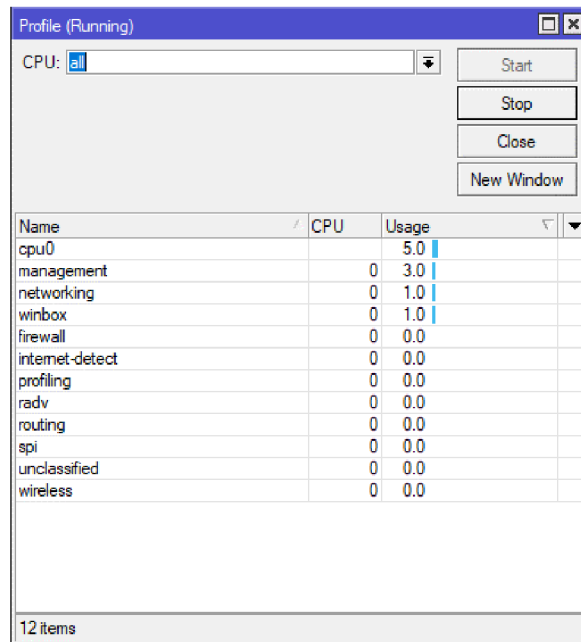
Obrázok 4.9 Iperf3 výstup, OpenVPN

Pri nastavenej šírke pásma 60 Mbit/s dochádza počas testu k značnému zahadzovaniu datagramov, strane prijímajúceho prijme asi 11 % z celkového počtu odoslaných. Toto je spôsobené spomínaným bottleneckom procesora.

Výsledná priemerná hodnota prenosovej rýchlosti nakoniec predstavuje 6,35 Mbit/s.

Oznámená priemerná hodnota parametra jitter je 0,731 milisekúnd a nakoniec priemerné oneskorenie bolo 4,359 milisekúnd.

## 4.5 Testovanie bez použitia virtuálnej privátnej siete – statické smerovanie medzi lokálnymi sieťami (RB941-2nD)



The screenshot shows the 'Profile (Running)' window in Mikrotik WinBox. At the top, there is a dropdown menu for 'CPU' set to 'all', and four buttons: 'Start', 'Stop', 'Close', and 'New Window'. Below this is a table with columns 'Name', 'CPU', and 'Usage'. The table lists 12 items with their respective CPU usage percentages. The 'cpu0' process shows the highest usage at 5.0%.

Name	CPU	Usage
cpu0		5.0
management	0	3.0
networking	0	1.0
winbox	0	1.0
firewall	0	0.0
internet-detect	0	0.0
profiling	0	0.0
radv	0	0.0
routing	0	0.0
spi	0	0.0
unclassified	0	0.0
wireless	0	0.0

12 items

Obrázok 4.10 Zaťaženie procesora Mikrotik RB941-2nD, statické smerovanie

Podobne ako v prípade testovania statického smerovania s použitím sieťových prvkov RB-493G (podkapitola 4.1), aj tu zaznamenávame len minimálne vyťaženie procesorov na oboch stranách komunikácie, pričom významnú časť testu išlo len o hodnoty do 10 % s občasnými výkyvmi do 30 %.

```

Prikazový riadok - iperf3 -s
[ 5] 558.00-559.00 sec 7.05 MBytes 59.1 Mbits/sec 0.179 ms 76/5137 (1.5%)
[ 5] 559.00-560.00 sec 7.15 MBytes 60.0 Mbits/sec 0.189 ms 0/5137 (0%)
[ 5] 560.00-561.00 sec 7.15 MBytes 60.0 Mbits/sec 0.193 ms 0/5137 (0%)
[ 5] 561.00-562.00 sec 7.14 MBytes 59.9 Mbits/sec 0.189 ms 11/5137 (0.21%)
[ 5] 562.00-563.01 sec 7.15 MBytes 59.3 Mbits/sec 0.864 ms 0/5133 (0%)
[ 5] 563.01-564.00 sec 7.16 MBytes 60.7 Mbits/sec 0.177 ms 0/5141 (0%)
[ 5] 564.00-565.00 sec 7.15 MBytes 60.0 Mbits/sec 0.172 ms 0/5137 (0%)
[ 5] 565.00-566.00 sec 7.15 MBytes 60.0 Mbits/sec 0.174 ms 0/5137 (0%)
[ 5] 566.00-567.00 sec 7.15 MBytes 60.0 Mbits/sec 0.177 ms 0/5137 (0%)
[ 5] 567.00-568.00 sec 7.15 MBytes 60.0 Mbits/sec 0.178 ms 0/5136 (0%)
[ 5] 568.00-569.00 sec 7.15 MBytes 60.0 Mbits/sec 0.193 ms 0/5138 (0%)
[ 5] 569.00-570.00 sec 7.15 MBytes 60.0 Mbits/sec 0.189 ms 0/5137 (0%)
[ 5] 570.00-571.00 sec 7.15 MBytes 60.0 Mbits/sec 0.190 ms 0/5137 (0%)
[ 5] 571.00-572.00 sec 7.15 MBytes 60.0 Mbits/sec 0.176 ms 0/5137 (0%)
[ 5] 572.00-573.00 sec 7.15 MBytes 60.0 Mbits/sec 0.195 ms 0/5137 (0%)
[ 5] 573.00-574.00 sec 6.85 MBytes 57.5 Mbits/sec 0.213 ms 216/5135 (4.2%)
[ 5] 574.00-575.00 sec 7.11 MBytes 59.6 Mbits/sec 0.217 ms 5/5110 (0.098%)
[ 5] 575.00-576.00 sec 6.77 MBytes 56.8 Mbits/sec 0.189 ms 301/5166 (5.8%)
[ 5] 576.00-577.00 sec 7.15 MBytes 60.0 Mbits/sec 0.179 ms 0/5137 (0%)
[ 5] 577.00-578.00 sec 7.15 MBytes 60.0 Mbits/sec 0.198 ms 0/5137 (0%)
[ 5] 578.00-579.00 sec 7.15 MBytes 60.0 Mbits/sec 0.201 ms 0/5137 (0%)
[ 5] 579.00-580.00 sec 7.15 MBytes 60.0 Mbits/sec 0.193 ms 0/5137 (0%)
[ 5] 580.00-581.00 sec 7.05 MBytes 59.1 Mbits/sec 0.227 ms 0/5062 (0%)
[ 5] 581.00-582.00 sec 7.21 MBytes 60.5 Mbits/sec 0.193 ms 31/5212 (0.59%)
[ 5] 582.00-583.00 sec 6.95 MBytes 58.3 Mbits/sec 0.191 ms 144/5137 (2.8%)
[ 5] 583.00-584.00 sec 7.15 MBytes 60.0 Mbits/sec 0.199 ms 0/5137 (0%)
[ 5] 584.00-585.00 sec 7.14 MBytes 59.9 Mbits/sec 0.198 ms 6/5137 (0.12%)
[ 5] 585.00-586.01 sec 6.97 MBytes 58.1 Mbits/sec 1.328 ms 0/5003 (0%)
[ 5] 586.01-587.00 sec 7.13 MBytes 60.2 Mbits/sec 0.189 ms 147/5271 (2.8%)
[ 5] 587.00-588.00 sec 7.15 MBytes 60.0 Mbits/sec 0.166 ms 0/5136 (0%)
[ 5] 588.00-589.00 sec 7.15 MBytes 60.0 Mbits/sec 0.192 ms 0/5138 (0%)
[ 5] 589.00-590.00 sec 7.14 MBytes 59.9 Mbits/sec 0.172 ms 0/5129 (0%)
[ 5] 590.00-591.00 sec 7.16 MBytes 60.1 Mbits/sec 0.183 ms 0/5144 (0%)
[ 5] 591.00-592.01 sec 7.10 MBytes 59.1 Mbits/sec 0.907 ms 0/5097 (0%)
[ 5] 592.01-593.00 sec 7.10 MBytes 59.9 Mbits/sec 0.341 ms 41/5138 (0.8%)
[ 5] 593.00-594.00 sec 7.21 MBytes 60.5 Mbits/sec 0.192 ms 0/5177 (0%)
[ 5] 594.00-595.00 sec 7.15 MBytes 60.0 Mbits/sec 0.193 ms 0/5136 (0%)
[ 5] 595.00-596.00 sec 7.15 MBytes 60.0 Mbits/sec 0.191 ms 0/5138 (0%)
[ 5] 596.00-597.00 sec 7.15 MBytes 60.0 Mbits/sec 0.182 ms 0/5136 (0%)
[ 5] 597.00-598.00 sec 7.15 MBytes 60.0 Mbits/sec 0.178 ms 0/5138 (0%)
[ 5] 598.00-599.00 sec 7.15 MBytes 60.0 Mbits/sec 0.192 ms 0/5137 (0%)
[ 5] 599.00-600.00 sec 7.15 MBytes 60.0 Mbits/sec 0.169 ms 0/5135 (0%)
[ 5] 600.00-600.00 sec 1.43 KBytes 7.46 Mbits/sec 0.248 ms 0/1 (0%)
[ ID] Interval          Transfer          Bitrate          Jitter          Lost/Total Datagrams
[SUM] 0.0-600.0 sec 51 datagrams received out-of-order
[ 5] 0.00-600.00 sec 4.17 GBytes 59.7 Mbits/sec 0.248 ms 13350/3082183 (0.43%) receiver

```

Obrázok 4.11 Iperf3 výstup, statické smerovanie RB941-2nD

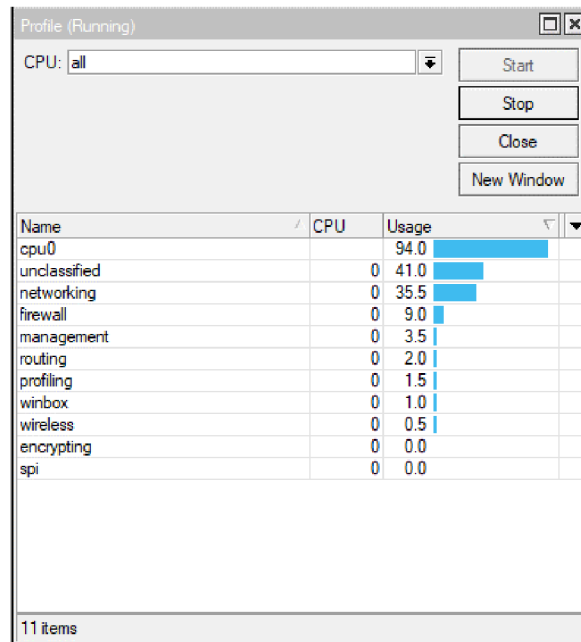
Priemerné prenosové rýchlosti 59,7 Mbit/s, takmer 100% zúžitkovanie ponúknutej šírky pásma. Stratovosť veľmi nízka, prijímajúci obdrží takmer všetky odoslané datagramy.

Priemerná hodnota parametra jitter nakoniec 0,248 milisekúnd.

Nameraná priemerná hodnota oneskorenia pri takomto scenári predstavovala 2,214 milisekúnd.



## 4.6 Testovanie s použitím WireGuard virtuálnej privátnej siete (RB941-2nD)



Obrázok 4.12 Zaťaženie procesora Mikrotik RB941-2nD, WireGuard

Pri tomto testovaní sa už stretávame so značným zaťažením procesorov Mikrotiku na oboch stranách. Narozdiel od testovania technológie WireGuard na RB-493G (podkapitola 4.2) dochádzalo k väčšiemu zaťaženiu procesora na strane prijímajúceho, a to v medziach od 90 do 100 % počas celého trvania testu. Značný podiel na takomto zaťažení zohrával proces *unclassified* (50-60 %), pod ktorým sa môže nachádzať wireguard, no router ho v našom prípade nevie klasifikovať. Procesor na strane odosielajúceho sa nachádzal v neustálom vyťažení 100 %.

```

Príkazový riadok - iperf3 -s
[ 5] 558.00-559.00 sec 6.87 MBytes 57.8 Mbits/sec 0.754 ms 98/5315 (1.8%)
[ 5] 559.00-560.00 sec 7.08 MBytes 59.3 Mbits/sec 0.278 ms 170/5548 (3.1%)
[ 5] 560.00-561.00 sec 7.14 MBytes 60.1 Mbits/sec 0.259 ms 27/5449 (0.5%)
[ 5] 561.00-562.00 sec 7.04 MBytes 59.1 Mbits/sec 0.272 ms 90/5442 (1.7%)
[ 5] 562.00-563.00 sec 6.80 MBytes 57.0 Mbits/sec 0.801 ms 0/5168 (0%)
[ 5] 563.00-564.01 sec 7.04 MBytes 58.3 Mbits/sec 1.081 ms 289/5639 (5.1%)
[ 5] 564.01-565.00 sec 7.24 MBytes 61.6 Mbits/sec 0.258 ms 12/5513 (0.22%)
[ 5] 565.00-566.00 sec 6.91 MBytes 58.0 Mbits/sec 0.270 ms 123/5376 (2.3%)
[ 5] 566.00-567.00 sec 6.78 MBytes 56.7 Mbits/sec 0.871 ms 100/5253 (1.9%)
[ 5] 567.00-568.00 sec 7.00 MBytes 58.9 Mbits/sec 0.296 ms 295/5616 (5.3%)
[ 5] 568.00-569.00 sec 7.14 MBytes 59.9 Mbits/sec 0.265 ms 71/5494 (1.3%)
[ 5] 569.00-570.00 sec 7.01 MBytes 58.8 Mbits/sec 0.185 ms 58/5383 (1.1%)
[ 5] 570.00-571.00 sec 7.05 MBytes 59.2 Mbits/sec 0.241 ms 113/5472 (2.1%)
[ 5] 571.00-572.00 sec 7.03 MBytes 58.9 Mbits/sec 0.201 ms 37/5375 (0.69%)
[ 5] 572.00-573.00 sec 7.16 MBytes 60.1 Mbits/sec 0.296 ms 4/5444 (0.073%)
[ 5] 573.00-574.01 sec 7.18 MBytes 59.5 Mbits/sec 1.122 ms 35/5494 (0.64%)
[ 5] 574.01-575.00 sec 7.13 MBytes 60.6 Mbits/sec 0.296 ms 22/5439 (0.4%)
[ 5] 575.00-576.00 sec 7.09 MBytes 59.5 Mbits/sec 0.335 ms 35/5421 (0.65%)
[ 5] 576.00-577.00 sec 7.05 MBytes 59.2 Mbits/sec 0.256 ms 9/5368 (0.17%)
[ 5] 577.00-578.00 sec 6.90 MBytes 57.9 Mbits/sec 0.251 ms 182/5426 (3.4%)
[ 5] 578.00-579.00 sec 7.25 MBytes 60.8 Mbits/sec 0.239 ms 0/5507 (0%)
[ 5] 579.00-580.00 sec 6.86 MBytes 57.6 Mbits/sec 0.547 ms 26/5242 (0.5%)
[ 5] 580.00-581.00 sec 7.30 MBytes 61.0 Mbits/sec 0.689 ms 33/5583 (0.59%)
[ 5] 581.00-582.00 sec 7.21 MBytes 60.6 Mbits/sec 0.376 ms 11/5493 (0.2%)
[ 5] 582.00-583.00 sec 6.67 MBytes 56.1 Mbits/sec 0.255 ms 342/5408 (6.3%)
[ 5] 583.00-584.00 sec 6.84 MBytes 57.4 Mbits/sec 0.261 ms 260/5458 (4.8%)
[ 5] 584.00-585.00 sec 6.95 MBytes 58.3 Mbits/sec 0.292 ms 160/5443 (2.9%)
[ 5] 585.00-586.00 sec 6.97 MBytes 58.5 Mbits/sec 0.244 ms 97/5393 (1.8%)
[ 5] 586.00-587.00 sec 7.02 MBytes 58.8 Mbits/sec 0.520 ms 97/5430 (1.8%)
[ 5] 587.00-588.00 sec 7.06 MBytes 59.3 Mbits/sec 0.239 ms 35/5403 (0.65%)
[ 5] 588.00-589.00 sec 7.11 MBytes 59.5 Mbits/sec 0.235 ms 34/5435 (0.63%)
[ 5] 589.00-590.00 sec 7.04 MBytes 59.1 Mbits/sec 0.297 ms 166/5516 (3%)
[ 5] 590.00-591.00 sec 7.04 MBytes 59.0 Mbits/sec 0.302 ms 86/5434 (1.6%)
[ 5] 591.00-592.00 sec 6.78 MBytes 56.9 Mbits/sec 0.218 ms 137/5288 (2.6%)
[ 5] 592.00-593.00 sec 7.20 MBytes 60.4 Mbits/sec 0.283 ms 0/5473 (0%)
[ 5] 593.00-594.00 sec 7.11 MBytes 59.6 Mbits/sec 0.276 ms 129/5531 (2.3%)
[ 5] 594.00-595.00 sec 6.97 MBytes 58.4 Mbits/sec 0.257 ms 40/5336 (0.75%)
[ 5] 595.00-596.00 sec 7.21 MBytes 60.5 Mbits/sec 0.211 ms 61/5538 (1.1%)
[ 5] 596.00-597.00 sec 7.09 MBytes 59.5 Mbits/sec 0.281 ms 45/5433 (0.83%)
[ 5] 597.00-598.00 sec 6.96 MBytes 58.4 Mbits/sec 0.456 ms 72/5363 (1.3%)
[ 5] 598.00-599.00 sec 7.22 MBytes 60.5 Mbits/sec 0.284 ms 24/5508 (0.44%)
[ 5] 599.00-600.00 sec 7.16 MBytes 60.1 Mbits/sec 0.322 ms 0/5442 (0%)
[ 5] 600.00-600.00 sec 25.6 KBytes 67.3 Mbits/sec 0.272 ms 0/19 (0%)
-----
[ ID] Interval      Transfer      Bitrate      Jitter      Lost/Total Datagrams
SUM] 0.0-600.0 sec 11 datagrams received out-of-order
[ 5] 0.00-600.00 sec 4.01 GBytes 57.4 Mbits/sec 0.272 ms 142122/3260864 (4.4%) receiver

```

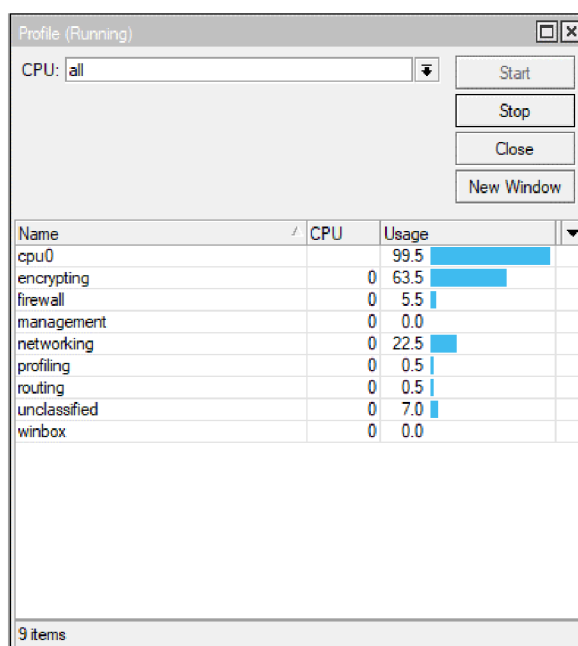
Obrázok 4.13 Iperf3 výstup, WireGuard, RB941-2nD

Výsledná priemerná hodnota prenosovej rýchlosti v takomto testovacom scenári predstavovala 57,4 Mbit/s. Aj napriek vysokým hodnotám zaťažovania procesora na strane odosielajúceho nedochádzalo k neprípustným hodnotám zahadzovania testovacích datagramov ako tomu bolo v prípadoch 4.3 a 4.4.

Priemerná hodnota pre jitter nakoniec 0,272 milisekúnd.

Priemerná hodnota pre sledovaný parameter oneskorenia – 3,062 milisekúnd.

## 4.7 Testovanie s použitím IPsec virtuálnej privátnej siete (RB941-2nD)



Obrázok 4.14 Zaťaženie procesora Mikrotik RB941-2nD, IPsec

Šifrovacie algoritmy a návrh použité v tomto scenári boli rovnaké ako v prípade 4.3, s jedinou zmenou týkajúcou sa použitého spôsobu overovania. Tu sa jedná o metódu predzdieľaného kľúča.

V tomto scenári dochádzalo k takmer 100% vyťaženiu procesorov na oboch stranách počas celého trvania testu. Významnú časť (priemerne 65 %) z toho tvoril proces *encrypting*.

Na strane odosielajúceho sme sa začali potýkať s obrovskými nestabilitami a celkovým bottleneckom procesora Mikrotiku. Počas testovania dochádzalo k častým výpadkom a degradáciám rýchlostí, ktoré potom len málokedy boli schopné vrátiť sa k maximálnym hodnotám. Odsielajúca strana pod takýmto zaťažením dokonca zhadzovala aj grafické užívateľské rozhranie Mikrotiku a následnými pokusmi o vrátenie ho späť tak do určitej miery ovplyňovala aj samotné výstupy testov.

```

Príkazový riadok - iperf3 -s
[ 5] 557.00-558.00 sec 2.27 MBytes 19.1 Mbits/sec 0.835 ms 3673/5376 (68%)
[ 5] 558.00-559.00 sec 2.27 MBytes 19.0 Mbits/sec 1.034 ms 3678/5380 (68%)
[ 5] 559.00-560.00 sec 2.27 MBytes 19.0 Mbits/sec 1.547 ms 3692/5393 (68%)
[ 5] 560.00-561.00 sec 2.27 MBytes 19.0 Mbits/sec 1.966 ms 3674/5376 (68%)
[ 5] 561.00-562.00 sec 2.27 MBytes 19.0 Mbits/sec 2.563 ms 3674/5377 (68%)
[ 5] 562.00-563.00 sec 2.28 MBytes 19.1 Mbits/sec 0.612 ms 3485/5112 (67%)
[ 5] 563.00-564.00 sec 2.27 MBytes 19.1 Mbits/sec 0.572 ms 3674/5378 (68%)
[ 5] 564.00-565.00 sec 2.27 MBytes 19.0 Mbits/sec 0.564 ms 3684/5383 (68%)
[ 5] 565.00-566.00 sec 2.27 MBytes 19.0 Mbits/sec 0.605 ms 3686/5389 (68%)
[ 5] 566.00-567.00 sec 2.27 MBytes 19.1 Mbits/sec 0.646 ms 3715/5418 (69%)
[ 5] 567.00-568.00 sec 2.25 MBytes 18.9 Mbits/sec 1.619 ms 3771/5461 (69%)
[ 5] 568.00-569.00 sec 2.27 MBytes 19.0 Mbits/sec 1.362 ms 3667/5369 (68%)
[ 5] 569.00-570.00 sec 2.26 MBytes 19.0 Mbits/sec 0.973 ms 3651/5349 (68%)
[ 5] 570.00-571.00 sec 2.27 MBytes 19.0 Mbits/sec 1.141 ms 3676/5375 (68%)
[ 5] 571.00-572.00 sec 2.26 MBytes 19.0 Mbits/sec 0.482 ms 3488/5105 (67%)
[ 5] 572.00-573.00 sec 2.26 MBytes 18.9 Mbits/sec 0.534 ms 3739/5432 (69%)
[ 5] 573.00-574.00 sec 2.26 MBytes 18.9 Mbits/sec 0.466 ms 3676/5368 (68%)
[ 5] 574.00-575.00 sec 2.25 MBytes 18.9 Mbits/sec 0.561 ms 3726/5413 (69%)
[ 5] 575.00-576.00 sec 2.25 MBytes 18.9 Mbits/sec 0.647 ms 3726/5412 (69%)
[ 5] 576.00-577.00 sec 2.25 MBytes 18.9 Mbits/sec 0.766 ms 3746/5437 (69%)
[ 5] 577.00-578.00 sec 2.27 MBytes 19.0 Mbits/sec 1.065 ms 3688/5388 (68%)
[ 5] 578.00-579.00 sec 2.27 MBytes 19.0 Mbits/sec 2.592 ms 3788/5407 (69%)
[ 5] 579.00-580.00 sec 2.27 MBytes 19.0 Mbits/sec 0.588 ms 3412/5111 (67%)
[ 5] 580.00-581.00 sec 2.27 MBytes 19.0 Mbits/sec 0.613 ms 3718/5417 (69%)
[ 5] 581.00-582.00 sec 2.27 MBytes 19.0 Mbits/sec 0.644 ms 3712/5413 (69%)
[ 5] 582.00-583.00 sec 2.27 MBytes 19.1 Mbits/sec 0.613 ms 3690/5395 (68%)
[ 5] 583.00-584.00 sec 2.27 MBytes 19.1 Mbits/sec 0.533 ms 3676/5380 (68%)
[ 5] 584.00-585.00 sec 2.27 MBytes 19.0 Mbits/sec 0.549 ms 3586/5285 (68%)
[ 5] 585.00-586.00 sec 2.27 MBytes 19.0 Mbits/sec 1.030 ms 3809/5511 (69%)
[ 5] 586.00-587.00 sec 2.27 MBytes 19.1 Mbits/sec 1.365 ms 3663/5369 (68%)
[ 5] 587.00-588.00 sec 2.27 MBytes 19.1 Mbits/sec 1.296 ms 3665/5370 (68%)
[ 5] 588.00-589.01 sec 2.25 MBytes 18.7 Mbits/sec 0.576 ms 3393/5083 (67%)
[ 5] 589.01-590.00 sec 1.19 MBytes 10.0 Mbits/sec 0.715 ms 4035/4928 (82%)
[ 5] 590.00-591.00 sec 2.13 MBytes 18.0 Mbits/sec 0.687 ms 4386/5907 (73%)
[ 5] 591.00-592.00 sec 2.28 MBytes 19.1 Mbits/sec 0.620 ms 3679/5390 (68%)
[ 5] 592.00-593.00 sec 2.27 MBytes 19.0 Mbits/sec 0.588 ms 3692/5393 (68%)
[ 5] 593.00-594.00 sec 2.26 MBytes 19.0 Mbits/sec 0.581 ms 3728/5425 (69%)
[ 5] 594.00-595.00 sec 2.26 MBytes 18.9 Mbits/sec 0.829 ms 3692/5386 (69%)
[ 5] 595.00-596.00 sec 2.25 MBytes 18.9 Mbits/sec 0.764 ms 3690/5380 (69%)
[ 5] 596.00-597.00 sec 2.26 MBytes 19.0 Mbits/sec 1.292 ms 3695/5390 (69%)
[ 5] 597.00-598.00 sec 2.26 MBytes 19.0 Mbits/sec 2.661 ms 3704/5402 (69%)
[ 5] 598.00-599.00 sec 2.27 MBytes 19.0 Mbits/sec 0.580 ms 3431/5133 (67%)
[ 5] 599.00-600.00 sec 2.27 MBytes 19.0 Mbits/sec 0.477 ms 3648/5348 (68%)
[ 5] 600.00-600.21 sec 35.5 KBytes 1.38 Mbits/sec 0.982 ms 81/107 (76%)

ID Interval Transfer Bitrate Jitter Lost/Total Datagrams
[ 5] 0.00-600.21 sec 1.14 GBytes 16.4 Mbits/sec 0.982 ms 2340312/3218692 (73%) receiver

```

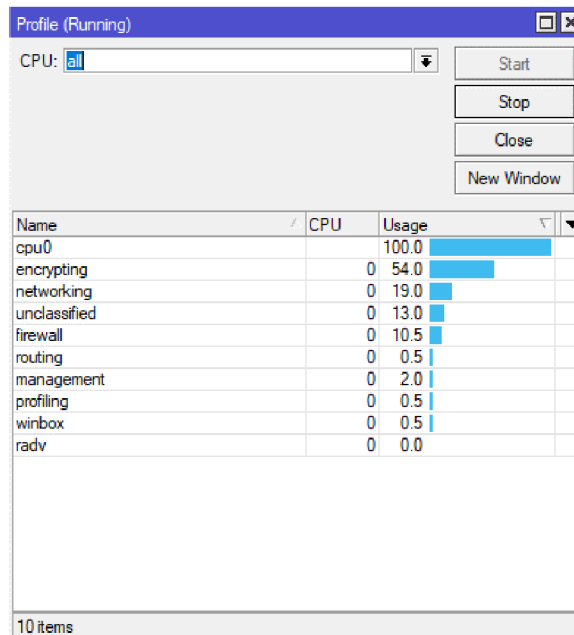
Obrázok 4.15 Iperf3 výstup, IPsec, RB941-2nD

Výsledná priemerná hodnota prenosových rýchlostí 16,4 Mbit/s. Viditeľná aj obrovská strátovosť, kedy prijímajúci obdržal len 27 % odosielaných datagramov.

Priemerná hodnota meraného parametra jitter nakoniec 0,982 milisekúnd.

Pre hodnotu oneskorenia išlo v priemere o 3,782 milisekúnd.

## 4.8 Testovanie s použitím OpenVPN virtuálnej privátnej siete (RB941-2nD)



Obrázok 4.16 Zaťaženie procesora Mikrotik RB941-2nD, OpenVPN

Počas celého trvania testu na strane odosielajúceho neustále maximálne vyťaženie procesora, pričom až 65 % z toho tvoril proces šifrovania *encrypting*. Strana prijímajúceho taktiež zaznamenávala neustále maximálne hodnoty využitia procesora. Narozdiel od merania s použitím konfigurácie IPsec (podkapitola 4.7) sme sa však nestretli s nestabilitou Mikrotiku ani častými pádmi grafického užívateľského rozhrania.

```

Príkazový riadok - iperf3 -s
[ 5] 558.00-559.00 sec 2.05 MBytes 17.2 Mbits/sec 1.031 ms 3670/5142 (71%)
[ 5] 559.00-560.00 sec 2.07 MBytes 17.3 Mbits/sec 1.059 ms 3686/5170 (71%)
[ 5] 560.00-561.00 sec 2.08 MBytes 17.4 Mbits/sec 1.041 ms 3651/5145 (71%)
[ 5] 561.00-562.00 sec 2.03 MBytes 17.0 Mbits/sec 1.004 ms 3660/5117 (72%)
[ 5] 562.00-563.00 sec 2.03 MBytes 17.0 Mbits/sec 1.201 ms 3620/5078 (71%)
[ 5] 563.00-564.00 sec 2.09 MBytes 17.5 Mbits/sec 1.094 ms 3612/5115 (71%)
[ 5] 564.00-565.00 sec 2.17 MBytes 18.2 Mbits/sec 1.114 ms 3640/5196 (70%)
[ 5] 565.00-566.00 sec 2.11 MBytes 17.8 Mbits/sec 1.252 ms 3655/5173 (71%)
[ 5] 566.00-567.00 sec 2.07 MBytes 17.4 Mbits/sec 0.835 ms 3660/5147 (71%)
[ 5] 567.00-568.00 sec 2.03 MBytes 17.1 Mbits/sec 1.061 ms 3650/5111 (71%)
[ 5] 568.00-569.00 sec 2.04 MBytes 17.1 Mbits/sec 0.605 ms 3701/5166 (72%)
[ 5] 569.00-570.00 sec 2.01 MBytes 16.9 Mbits/sec 1.286 ms 3663/5107 (72%)
[ 5] 570.00-571.00 sec 2.06 MBytes 17.3 Mbits/sec 0.769 ms 3640/5121 (71%)
[ 5] 571.00-572.00 sec 2.03 MBytes 17.0 Mbits/sec 0.866 ms 3726/5184 (72%)
[ 5] 572.00-573.00 sec 1.98 MBytes 16.6 Mbits/sec 1.170 ms 3632/5057 (72%)
[ 5] 573.00-574.00 sec 2.05 MBytes 17.2 Mbits/sec 0.699 ms 3727/5198 (72%)
[ 5] 574.00-575.00 sec 2.03 MBytes 17.0 Mbits/sec 1.123 ms 3668/5123 (72%)
[ 5] 575.00-576.00 sec 2.01 MBytes 16.9 Mbits/sec 1.307 ms 3631/5076 (72%)
[ 5] 576.00-577.00 sec 2.06 MBytes 17.2 Mbits/sec 0.873 ms 3671/5149 (71%)
[ 5] 577.00-578.01 sec 1.99 MBytes 16.6 Mbits/sec 1.671 ms 3635/5065 (72%)
[ 5] 578.01-579.01 sec 1.77 MBytes 14.8 Mbits/sec 1.133 ms 3875/5143 (75%)
[ 5] 579.01-580.02 sec 778 KBytes 6.32 Mbits/sec 3.788 ms 2148/2094 (80%)
[ 5] 580.02-581.00 sec 2.02 MBytes 17.1 Mbits/sec 1.681 ms 6241/7091 (81%)
[ 5] 581.00-582.01 sec 1.77 MBytes 14.8 Mbits/sec 0.305 ms 3854/5125 (75%)
[ 5] 582.01-583.01 sec 2.07 MBytes 17.4 Mbits/sec 1.879 ms 3689/5177 (71%)
[ 5] 583.01-584.00 sec 2.07 MBytes 17.4 Mbits/sec 0.953 ms 3634/5119 (71%)
[ 5] 584.00-585.00 sec 2.04 MBytes 17.1 Mbits/sec 1.204 ms 3656/5123 (71%)
[ 5] 585.00-586.00 sec 2.07 MBytes 17.4 Mbits/sec 1.125 ms 3673/5161 (71%)
[ 5] 586.00-587.00 sec 2.07 MBytes 17.4 Mbits/sec 0.967 ms 3635/5122 (71%)
[ 5] 587.00-588.00 sec 2.07 MBytes 17.4 Mbits/sec 1.361 ms 3653/5142 (71%)
[ 5] 588.00-589.01 sec 2.06 MBytes 17.1 Mbits/sec 1.526 ms 3634/5114 (71%)
[ 5] 589.01-590.00 sec 2.06 MBytes 17.4 Mbits/sec 1.145 ms 3661/5137 (71%)
[ 5] 590.00-591.00 sec 2.05 MBytes 17.2 Mbits/sec 0.775 ms 3667/5139 (71%)
[ 5] 591.00-592.00 sec 2.00 MBytes 16.7 Mbits/sec 0.686 ms 3662/5097 (72%)
[ 5] 592.00-593.00 sec 1.68 MBytes 14.1 Mbits/sec 1.000 ms 3000/5119 (76%)
[ 5] 593.00-594.00 sec 1.29 MBytes 10.8 Mbits/sec 0.953 ms 4268/5195 (82%)
[ 5] 594.00-595.00 sec 2.03 MBytes 17.0 Mbits/sec 0.763 ms 3729/5184 (72%)
[ 5] 595.00-596.00 sec 2.02 MBytes 16.9 Mbits/sec 1.045 ms 3677/5126 (72%)
[ 5] 596.00-597.00 sec 2.04 MBytes 17.2 Mbits/sec 0.978 ms 3689/5155 (72%)
[ 5] 597.00-598.00 sec 2.03 MBytes 17.0 Mbits/sec 1.290 ms 3575/5031 (71%)
[ 5] 598.00-599.01 sec 2.09 MBytes 17.3 Mbits/sec 1.022 ms 3676/5174 (71%)
[ 5] 599.01-600.00 sec 2.07 MBytes 17.5 Mbits/sec 0.569 ms 3707/5191 (71%)
[ 5] 600.00-600.22 sec 38.5 KBytes 1.45 Mbits/sec 1.471 ms 93/120 (78%)

-- -- --
[ ID] Interval      Transfer      Bitrate      Jitter      Lost/Total Datagrams
[SUM] 0.0-600.2 sec 5 datagrams received out-of-order
[ 5] 0.00-600.22 sec 1.19 GBytes 17.1 Mbits/sec 1.471 ms 2204007/3082188 (72%) receiver

```

Obrázok 4.17 Iperf3 výstup, OpenVPN, RB941-2nD

Na konci testovania sme obdržali výslednú priemernú hodnotu prenosových rýchlostí 17,1 Mbit/s na úkor až 72% strátovosti.

Meraný parameter jitter na konci testu zaznamenáva priemernú hodnotu 1,471 milisekúnd.

Výsledná priemerná hodnota pre oneskorenie – 4,613 milisekúnd.

## 5. POROVNANIE NAMERANÝCH HODNÔT

V rámci tejto kapitoly sú nami namerané priemerné hodnoty sledovaných parametrov reprezentované v tabuľkách a na ich základe je uskutočnené záverečné porovnanie technológií, ktorými sa práca zaoberala.

Tabuľka 5.1 Tabuľka priemerných hodnôt – RB-493G

	jitter [ms]	oneskorenie [ms]	prenosová rýchlosť [Mbit/s]
<b>Statické smerovanie</b>	0,096	2,156	59,6
<b>WireGuard</b>	0,179	2,917	56,4
<b>IPsec</b>	0,789	3,492	18,5
<b>OpenVPN</b>	0,731	4,359	6,35

Tabuľka 5.2 Tabuľka priemerných hodnôt – RB941-2nD

	jitter [ms]	oneskorenie [ms]	prenosová rýchlosť [Mbit/s]
<b>Statické smerovanie</b>	0,248	2,214	59,7
<b>WireGuard</b>	0,272	3,062	57,4
<b>IPsec</b>	0,982	3,782	16,4
<b>OpenVPN</b>	1,471	4,613	17,1

Na základe nami nameraných výsledkov pre oba scenáre môžeme usúdiť, že technológia WireGuard naozaj lepšie využíva poskytnuté parametre siete, ale aj hardvérové vybavenie sieťových prvkov narozdiel od IPsec a OpenVPN. Potvrďuje sa tým predpoklad, ktorý mohol vyplývať z teoretickej časti práce aj samotného zámeru autorov WireGuard.

Na druhej strane sme testovaním zistili, že technológia OpenVPN nepredstavuje takmer žiadnu konkurenciu pre WireGuard.

Síce v testoch s použitím RB941-2nD môžeme vidieť len rozdiely v rádoch desiatok Mbit/s medzi prenosovými rýchlosťami OpenVPN a IPsec, pripisujeme takéto výsledky bottlenecku a neustálym problémom s prvkom Mikrotik.

## 6. ZÁVER

Cieľom bakalárskej práce bolo analyzovať technológiu WireGuard, ďalej ju porovnať s technológiami OpenVPN a IPsec a nakoniec uskutočniť testy, pomocou ktorých by boli otestované spomínané technológie a ich parametre (jitter, oneskorenie, prenosové rýchlosti).

V prvej časti práce sa nachádza opis virtuálnej privátnej siete WireGuard a jeho fungovanie.

V ďalšej časti je WireGuard porovnaný s technológiami IPsec a OpenVPN na základe nimi používaných šifrovacích algoritmov.

V nasledujúcej kapitole práca popisuje metodiku a kritériá testovania výkonu jednotlivých technológií s použitím konkrétneho hardvéru a odpovedajúcej konfigurácie topológie siete a taktiež predstavuje namerané výsledky pre konkrétne virtuálne privátne siete, prípadne scenár, kedy virtuálna privátna sieť nie je implementovaná.

Predposledná kapitola predstavuje namerané výsledky uskutočnených testov pre každú z virtuálnych privátnych sietí.

Posledná kapitola sa venuje porovnaniu nami nameraných výsledkov týkajúcich sa technických parametrov virtuálnych privátnych sietí, ktorými sú jitter, oneskorenie pri prenose a dosiahnuteľné prenosové rýchlosti, ktoré sú relevantné pre nami vopred zvolené parametre siete a kritériá testovania a taktiež použitý testovací hardvér a softvér.



## LITERATURA

- [1] *WireGuard: Next Generation Kernel Network Tunnel* [online]. 2020 [cit. 2022-11-18]. Dostupné z: <https://www.wireguard.com/papers/wireguard.pdf>
- [2] *Curve25519: new Diffie-Hellman speed records* [online]. 2006 [cit. 2020-11-18]. Dostupné z: <https://cr.yp.to/ecdh/curve25519-20060209.pdf>
- [3] *Internet Key Exchange Protocol Version 2 (IKEv2)*. 2010 [cit. 2022-11-18]. Dostupné z: <https://www.rfc-editor.org/rfc/rfc5996.txt>
- [4] *Datagram Transport Layer Security Version 1.2*. [online]. 2012 [cit. 2022-11-18]. Dostupné z: <https://www.rfc-editor.org/rfc/rfc6347.txt>
- [5] *The Noise Protocol Framework*. [online]. 2018 [cit. 2022-11-18]. Dostupné z: <http://noiseprotocol.org/noise.pdf>
- [6] *Cryptographic Extraction and Key Derivation: The HKDF Scheme*. [online]. 2010 [cit. 2022-11-18]. Dostupné z: <https://eprint.iacr.org/2010/264.pdf>
- [7] *ChaCha, a variant of Salsa20*. [online]. 2008 [cit. 2022-11-20]. Dostupné z: <https://cr.yp.to/chacha/chacha-20080128.pdf>
- [8] *ChaCha20 and Poly1305 for IETF Protocols*. [online]. 2015 [cit. 2022-11-20]. Dostupné z: <https://www.rfc-editor.org/rfc/rfc7539.txt>
- [9] *BLAKE2: simpler, smaller, fast as MD5*. [online]. 2013 [cit. 2022-11-20]. Dostupné z: <https://www.blake2.net/blake2.pdf>
- [10] *Mosh: An interactive Remote Shell for Mobile Clients*. [online]. 2012 [cit. 2022-11-23]. Dostupné z: <https://mosh.org/mosh-paper.pdf>
- [11] LAUTER, Kristin a Anton MITYAGIN. Security Analysis of KEA Authenticated Key Exchange Protocol. In: YUNG, Moti, Yevgeniy DODIS, Aggelos KIAYIAS a Tal MALKIN, ed. *Public Key Cryptography - PKC 2006* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, 2006, s. 378-394 [cit. 2022-11-27]. Lecture Notes in Computer Science. ISBN 978-3-540-33851-2. Dostupné z: doi:10.1007/11745853\_25
- [12] *TAI64, TAI64N and TAI64NA*. [online]. [cit. 2022-11-27]. Dostupné z: <https://cr.yp.to/libtai/tai64.html>
- [13] DOBRAUNIG, Christoph, Florian MENDEL a Martin SCHLÄFFER. Differential Cryptanalysis of SipHash. In: JOUX, Antoine a Amr YOUSSEF, ed. *Selected Areas in Cryptography -- SAC 2014* [online]. Cham: Springer International Publishing, 2014, 2014-11-29, s. 165-182 [cit. 2022-12-4]. Lecture Notes in Computer Science. ISBN 978-3-319-13050-7. Dostupné z: doi:10.1007/978-3-319-13051-4\_10
- [14] RIVEST, R. L., A. SHAMIR a L. ADLEMAN. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM* [online]. 1978, **21**(2), 120-126 [cit. 2022-12-04]. ISSN 0001-0782. Dostupné z: doi:10.1145/359340.359342

- [15] *Biclique Cryptanalysis of the Full AES*. [online]. [cit. 2022-12-04]. Dostupné z: <https://web.archive.org/web/20160306104007/http://research.microsoft.com/en-us/projects/cryptanalysis/aesbc.pdf>
- [16] *More Modular Exponential Diffie-Hellman groups for Internet Key Exchange*. [online]. 2003 [cit. 2022-12-08]. Dostupné z: <https://www.ietf.org/rfc/rfc3526.txt>
- [17] MERKLE, Ralph C. Secure communications over insecure channels. *Communications of the ACM* [online]. 1978, **21**(4), 294-299 [cit. 2022-12-12]. ISSN 0001-0782. Dostupné z: doi:10.1145/359460.359473
- [18] *ECP Groups for IKE and IKEv2*. [online]. 2007 [cit. 2022-12-09]. Dostupné z: <https://datatracker.ietf.org/doc/html/rfc4753>
- [19] *Standards for Efficient Cryptography*. [online]. 2009 [cit. 2022-12-10]. Dostupné z: <http://www.secg.org/sec1-v2.pdf>
- [20] *Digital Signature Standard (DSS)*. [online]. 2013 [cit. 2022-12-10]. Dostupné z: <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>
- [21] *Authenticating remote peers and clients*. [online]. [cit. 2022-12-11]. Dostupné z: <https://docs.fortinet.com/document/fortigate/6.0.0/handbook/530590/authenticating-remote-peers-and-clients>
- [22] *OpenSSL*. [online]. 2021 [cit. 2022-12-11]. Dostupné z: <https://www.openssl.org>
- [23] *Extending the Salsa20 nonce*. [online]. 2011 [cit. 2022-12-10]. Dostupné z: <https://cr.yp.to/snuffle/xsalsa-20110204.pdf>
- [24] [online]. [cit. 2023-05-22]. Dostupné z: <https://iperf.fr>
- [25] <https://wiki.mikrotik.com/wiki/Manual:IP/Fasttrack> [online]. [cit. 2023-05-20]. Dostupné z: <https://wiki.mikrotik.com/wiki/Manual:IP/Fasttrack>
- [26] KIVINEN, T. a M. KOJO. *More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)* [online]. [cit. 2023-05-22]. Dostupné z: doi:DOI 10.17487/RFC3526
- [27] PENARD, Wouter a Tim van WERKHOVEN. *On the Secure Hash Algorithm family* [online]. [cit. 2023-05-23]. Dostupné z: [https://web.archive.org/web/20160330153520/https://www.staff.science.uu.nl/~warkh108/docs/study/Y5\\_07\\_08/infocry/project/Cryp08.pdf](https://web.archive.org/web/20160330153520/https://www.staff.science.uu.nl/~warkh108/docs/study/Y5_07_08/infocry/project/Cryp08.pdf)

# SEZNAM SYMBOLŮ A ZKRATEK

Zkratky:

FEKT	Fakulta elektrotechniky a komunikačních technologií
VUT	Vysoké učení technické v Brně
AEAD	Authenticated Encryption with Associated Data
AKE	Authenticated Key Exchange
AES	Advanced Encryption Standard
DH	Diffie-Hellman
DTLS	Datagram Transport Layer Security
ECDH	Elliptic-curve Diffie-Hellman
ECDSA	Elliptic-curve Digital Signature Algorithm
HKDF	Hash-based Key Derivation Function
ICMP	Internet Control Message Protocol
IKEv2	Internet Key Exchange version 2
IP	Internet Protocol
KCI	Key Compromise Impersonation
KEA+C	Key Exchange Algorithm with key confirmation
MAC	Message Authentication Code
MB	Megabyte
MHz	Megahertz
NAT	Network Address Translation
PSK	Pre-shared Key
RAM	Random Access Memory
RSA	Rivest-Shamir-Adleman cryptosystem
RTT	Round Trip Time
SSH	Secure Shell
SSL	Secure Socket Layer
TAI	International Atomic Time
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
UTF-8	Unicode Transformation Format
VPN	Virtual Private Network