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

VYTVOŘENÍ VPN PŘÍSTUPOVÉHO BODU NA RASPBERRY PI
CREATING A VPN ACCESS POINT ON RASPBERRY PI

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Martin Kovařík**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jan Benedikt**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Martin Kovařík ***ID:*** 190001
Ročník: 3 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Vytvoření VPN přístupového bodu na Raspberry Pi

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s technologií VPN. Analyzujte protokoly a technologie pro vytváření virtuálních privátních sítí (IPsec, OpenVPN, WireGuard, apod.). Implementujte protokoly a technologie na jednodeskovém počítači Raspberry Pi. Otestujte funkčnost a zhodnoťte škálovatelnost implementovaného řešení. Zjistěte maximální počet účastníků současně připojených do privátní sítě.

DOPORUČENÁ LITERATURA:

[1] NEGUS, Chris. Linux bible. Ninth edition. Indianapolis, Indiana, 2015. ISBN 978-1118999875.

[2] STEWART, James Michael. Network security firewalls and VPNs. Ninth edition. Sudbury, MA, c2011. ISBN 978-0763791308.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 8.6.2020

Vedoucí práce: Ing. Jan Benedikt

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

ABSTRAKT

Tato bakalářská práce se zabývá vytvořením přístupového bodu virtuální privátní sítě (VPN) na jednodeskovém počítači Raspberry Pi. Teoretická část práce popisuje princip VPN, nejrozšířenější tunelovací protokoly a obecně počítač Raspberry Pi s jeho různými generacemi a modely. Zvolené protokoly jsou implementované na Raspberry Pi a je u nich naměřeno hardwarové zatížení, jejich odezva a propustnost. U protokolů je otestována jejich stabilita a zhodnocena spolehlivost.

KLÍČOVÁ SLOVA

iPerf, IPsec, OpenVPN, PPTP, Raspberry Pi, VPN, WireGuard

ABSTRACT

This bachelor thesis deals with creating a virtual private network (VPN) access point on a single-board Raspberry Pi computer. The theoretical part describes the principle of VPN, the most common tunneling protocols and the Raspberry Pi computer in general with its various generations and models. Selected protocols are implemented on the Raspberry Pi and their hardware load, response and throughput are measured. The stability of the protocols is tested and reliability evaluated.

KEYWORDS

iPerf, IPsec, OpenVPN, PPTP, Raspberry Pi, VPN, WireGuard

KOVAŘÍK, Martin. *Vytvoření VPN přístupového bodu na Raspberry Pi*. Brno, 2020, 74 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Jan Benedikt,

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Vytvoření VPN přístupového bodu na Raspberry Pi“ jsem vypracoval samostatně pod vedením vedoucího bakalářské 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 bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Janu Benediktovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Seznam výpisů

Úvod

Zaměstnanci firem požadují, či potřebují přístup k prostředkům firemního intranetu z jakéhokoliv místa. Obyvatelé některých zemí se snaží vyhnout cenzuře a regulování jejich připojení k Internetu, nebo uživatelé, kteří si váží svého soukromí, chtějí zvýšit svoji bezpečnost a zachovat si anonymitu na Internetu. Všechny vyjmenované problémy řeší technologie VPN (*Virtual Private Network*), která je čím dál více rozšířená. VPN zvyšuje bezpečnost prostřednictvím šifrování přenášených dat, a tím zamezení jejich odposlechu nebo jejich manipulaci třetí stranou. Zvýšenou bezpečnost také zajišťuje pomocí autentizace komunikujících stran. Výhoda bezpečnosti má však svoji cenu, a to ve zvýšení latence a náročnosti na výpočetní výkon. Zmíněné nevýhody jsou ale převáženy výhodami ve většině případů. Jaký však hardware nebo protokol využít pro implementaci VPN je důležitá otázka. Existuje mnoho VPN řešení, ale vybrat si to správné není jednoduché.

Bakalářská práce se zabývá právě protokoly VPN a zvolené z nich jsou implementovány na mini počítači Raspberry Pi. Uvedená kombinace má potenciál být levným řešením pro malé organizace vyžadující služby virtuální privátní sítě.

V kapitole 1 práce popisuje obecně technologii VPN, jejich typy a možné důvody k použití. Následuje popis nejpopulárnějších a nejrozšířenějších VPN tunelovacích protokolů. Jedná se o starší a méně používané protokoly, modernější a aktuálně velice rozšířené až po nejmladší a méně aplikované protokoly. V kapitole 2 je popsán mini počítač Raspberry Pi a jeho různé generace a modely. V předposlední kapitole 3 je popsána implementace čtyř zvolených tunelovacích protokolů na Raspberry Pi model 3 B. Ve finální kapitole 4 je otestováno hardwarové zatížení zvolených VPN protokolů a testována je jejích stabilita při připojení více klientů ve stejný čas.

1 VPN

1.1 Co je to VPN

Virtual Private Network (Virtuální privátní síť) rozšiřuje privátní síť přes veřejné sítě, a tím umožňuje uživatelům posílat a přijímat data napříč sdílenými nebo veřejnými sítěmi, jako kdyby zařízení byly přímo propojena v privátní síti. VPN vytváří šifrované spojení, které se nazývá VPN tunel, a veškerý, nebo zvolený internetový provoz a komunikace prochází tímto zabezpečeným tunelem.[1]

1.2 Typy VPN

Jsou rozlišovány 2 typy sítě VPN:

- **Remote Access** – *Remote Access* VPN umožňuje uživateli se připojit k soukromé síti a vzdáleně přistupovat ke všem jeho službám a prostředkům. Spojení mezi uživatelem a privátní sítí probíhá přes Internet a připojení je soukromé a zabezpečené. Používá se např. pro zaměstnance společnosti, který chce z domu přistoupit k souborům či jiným prostředkům v privátní síti organizace. Domácí uživatelé používají nejčastěji VPN, aby se vyhnuli regionálním omezením na Internetu a přístupu k blokovaným webovým stránkám (například obyvatelé Číny a obcházení *The Great Firewall of China*), dále z důvodu zvýšení svého soukromí a bezpečnosti.
- **Site–to–Site VPN** – V rámci *Site–to–Site* VPN můžeme ještě dále rozlišovat 2 druhy:
  - **Intranet** – Tento typ je běžně používán velkými společnostmi s pobočkami v různých geografických lokalitách, aby společně propojili sítě jednotlivých poboček.
  - **Extranet** – Vytvoření sítě vně intranetu společnosti, přístupný pouze partnerským nebo důvěryhodným organizacím.

1.3 Důvody k použití VPN

- Bezpečnost – Hlavní důvod pro využívání VPN je zajištění bezpečné komunikace. Základní bezpečnostní požadavky jsou následující:
  - **Autentizace** – ověření identity komunikujících stran, zda klient je opravdu ten, za koho se vydává. Autentizace je realizována různými metodami jako např. uživatelské jméno a heslo, sdílený klíč nebo bezpečnostní certifikát.

  - **Autorizace** – ověření, zda uživatel má oprávnění k provedení požadované operace.
  - **Důvěrnost** – zaručení, že přenášená data nemohou být získána třetí stranou. Důvěrnost je zajištěna šifrováním dat.
  - **Integrita** – schopnost zjistit, že během přenosu dat nedošlo k jejich změně. Ta může být způsobena chybou v přenosu, ale také jejich záměrnou změnu útočníkem.
- Regulovaný Internet – Některé státy na Zemi regulují pro své občany Internet a to ve formě odepření nebo omezení přístupu k jistým doménám a službám. Příklady takových států jsou Čína, Írán, Rusko a Turecko. Jednotliví ISP (*Internet Service Provider*) jsou povinni splňovat požadavky dané vládou. K často blokovaným službám patří např. Google, Youtube, Twitter, Facebook, WhatsApp anebo pornografické stránky. V těchto státech jsou dokonce virtuální privátní sítě zakázané a jejich použití je pokutováno. [3]

1.4 Tunelovací protokol

Tunelovací protokol je síťový protokol sloužící k vytvoření tunelu, tedy virtuálního *point–to–point* spojení mezi uzly. Hraniční uzly tohoto tunelu zodpovídají za zapouzdření a rozbalení přenášených paketů/rámců přes síť. Jedním nebo více takovými spojeními lze realizovat virtuální privátní síť.[2]

Nevýhoda tunelovacích protokolů je především zvýšená náročnost na výpočetní výkon, kapacitu sítě a vyšší latence způsobené šifrováním, autentizačními daty, řízením tunelu a dodatečnými hlavičkami. Hlavními faktory jsou vzdálenost, zpoždění a kvalita linky mezi klientem a VPN serverem. Se snížením kapacity linky a zpomalením se musí počítat, neboť komunikace na cílový server není směřována přímo, ale data nejdříve jsou zaslána na VPN server a až odtud jsou směrovaná k cílovému serveru. To stejné platí i pro cestu dat nazpět. Velký problém může nastat u tunelů používajících TCP protokol, ve kterém je zapouzdřen další TCP (*Transmission Control Protocol*) segment (např. L2TP). Může zde nastat problém nazývající se *TCP meltdown*.

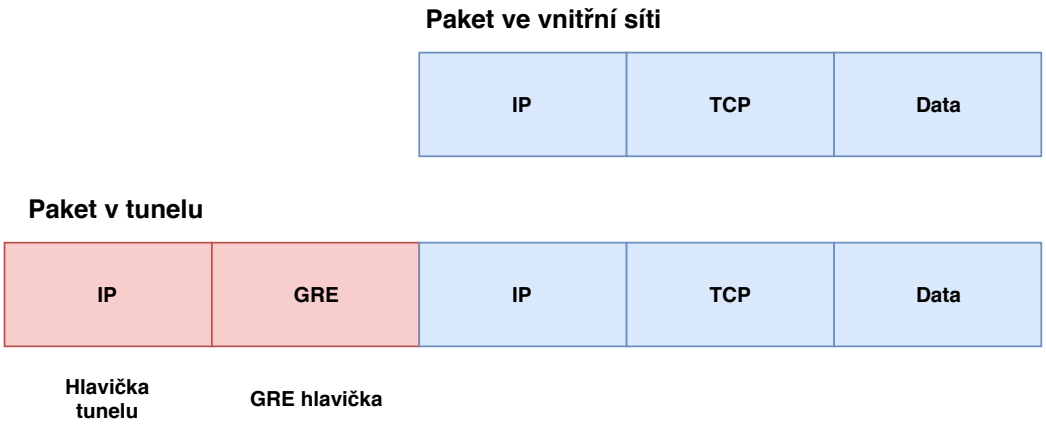
TCP Meltdown – se nazývá situace, kdy kontroly přetížení (*congestion controls*) z vnějšího protokolu TCP interferují s kontrolami z vnitřního TCP a naopak. TCP rozděluje datový tok na segmenty, které jsou zasílány jako jednotlivé datagramy internetového protokolu (IP). V každém segmentu je pořadové číslo (*sequence number*) spolu s potvrzovacím číslem (*acknowledgement number*), které udává, jaké pořadové číslo bylo naposledy přijato. TCP používá adaptivní časové limity k rozhodnutí, kdy má dojít k opětovnému odeslání. Tento design se avšak může vymstít

při vrstvení TCP, protože pomalejší připojení může způsobit, že horní vrstva dá do fronty více opakovaných přenosů rychleji, než je spodní vrstva schopna zpracovat.[4]

1.4.1 GRE

Generic Routing Encapsulation je tunelovací protokol vyvinut společností Cisco Systems, který byl definován v RFC (*Request for Comments*) 1701 a RFC 1702 v roce 1994. Zapouzdřuje pakety jednoho protokolu do protokolu druhého. Používá se přenos IPv6 (*Internet Protocol*) paketů v IPv4 síti a pro tunelování obecně. K zabaleným paketům přenášené tunelem je přidána GRE hlavička a cílová adresa. V koncovém bodě tunelu je paket rozbalen a dále postupuje podle informací v původní IP hlavičce. Strukturu GRE paketu s přidanými hlavičkami můžeme vidět na obr. 1.1.

Nevýhodou využívání GRE tunelů k vytváření VPN je škálovatelnost. Všechny tunely musí být předem ručně konfigurovány, a tedy při větším počtu tunelů roste náročnost na administrativu. GRE také nezajišťuje šifrování přenášených dat.[5]



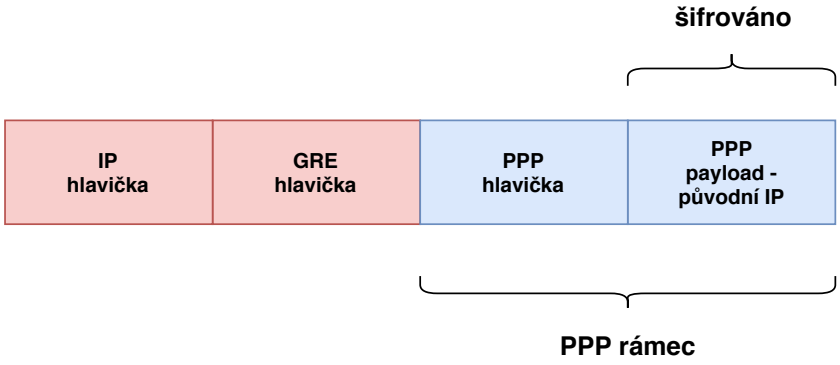
Obr. 1.1: GRE paket

1.4.2 PPTP

Point–to–Point Tunneling Protocol byl zveřejněn v roce 1999 v RFC 2637 a vyvinut společnostmi Microsoft, Ascend Communications, 3COM a ECI Telematics. Organizace IETF (*Internet Engineering Task Force*) avšak nikdy protokol PPTP nenavrhla jako standard.

Umožňuje přenos PPP (*Point-to-Point Protocol*) rámců přes síť. Je podporován většinou dnes používaných operačních systémů. Využívá TCP portu 1723 a protokolu GRE. TCP je použit pouze pro řízení tunelu – vznik, udržování a ukončení. Původní IP paket je zapouzdřen do PPP rámce a GRE rámce, následně dostane novou IP

hlavičku a je odeslán. V IP hlavičce je uvedena zdrojová a cílová adresa klienta a VPN serveru. Strukturu paketu si můžeme prohlídnout na obr. 1.2.



Obr. 1.2: Struktura paketu v PPTP

PPTP samotné neposkytuje autentizaci tunelu a autentičnost, integritu či důvěrnost přenášených dat. Bezpečnost je zcela přenechána protokolu PPP. Autentizaci uživatele PPP řeší pomocí těchto protokolů:

- PAP (*Password Authentication Protocol*)
- CHAP (*Challenge-Handshake Authentication Protocol*)
- MS–CHAP (*Microsoft Challenge–Handshake Authentication Protocol*)
- EAP–TLS (*Extensible Authentication Protocol – Transport Layer Security*)

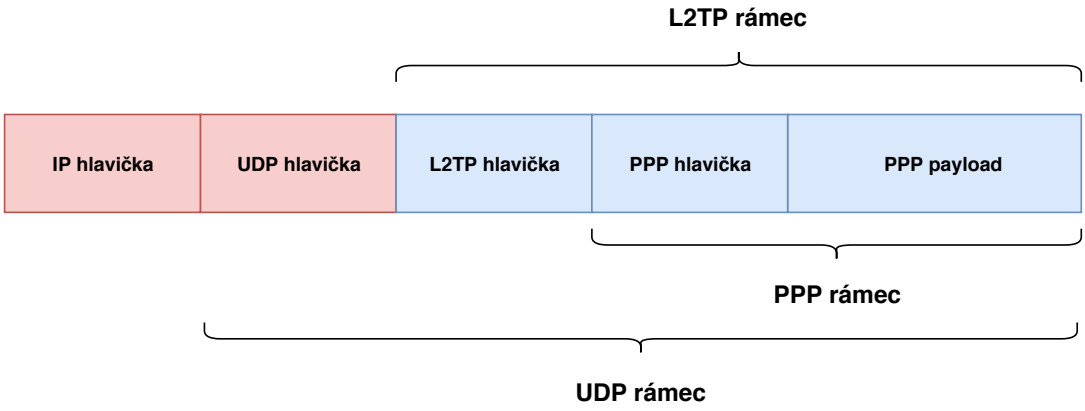
Pokud není k dispozici autentizační server pro využití protokolu EAP–TLS, musí byt použit MS–CHAPv2. Data v PPP jsou šifrovaná protokolem MPPE (*Microsoft Point–to–Point Encryption*). MPPE používá algoritmus RSA RC4. Podporuje 40bit, 56bit a 128bit klíče relace, kde nejčastěji jsou používané 128bit dlouhé klíče – MPPE128.

Protokol je lehce detekovatelný a blokovatelný, dále má známé zranitelnosti, a to v jeho implementaci autentizačního protokolu MS–CHAP a MS–CHAPv2, které jsou náchylné na slovníkové útoky. Výhodou však je vysoká rychlost a jednoduchost spojení. Z důvodu propagace společností Microsoft je protokol PPTP velice rozšířen.[6]

1.4.3 L2TP

Layer 2 Tunneling Protocol byl zveřejněný v roce 2000 a je popsán v RFC 2661. Má svoje základy ve dvou tunelovacích protokolech, a to v protokolu L2F (*Layer 2 Forwarding*) od společnosti Cisco Systems a PPTP od Microsoft. Strukturu paketu při jeho použití můžeme vidět na obr. 1.3. Využívá pouze port 1701 UDP (*User Datagram Protocol*), což jako u PPTP vede k jednoduché detekci a blokaci. Pomocí

užití UDP se vyvarovává problému *TCP Meltdown*. Stejně jako u PPTP sám o sobě L2TP nezajišťuje integritu, důvěrnost a autentičnost dat, ale vše přenechává na protokolu PPP. Protokol L2TP vytváří 2 koncové body, které se nazývají LAC (*L2TP Access Concentrator*) a LNS (*L2TP Network Server*). L2TP má 2 typy zpráv: kontrolní a datové. Kontrolním paketům protokol zajišťuje doručení, zatímco u datových tomu tak není. Oba typy zpráv sdílejí stejnou hlavičku.



Obr. 1.3: Struktura paketu v L2TP

Výhoda je jednoduchá implementace a vysoké rychlosti přenosu. L2TP je vestavěn od Windows 2000 dále a Mac OS X 10.3 a výš. V Linuxu je nutné podporu doinstalovat, kde nejpopulárnější řešení je *xl2tpd*[7]. Z důvodu používání pouze jednoho portu a to UDP 1701 je lehce detekovatelný a blokovatelný. Stejně jako u PPTP lze data v PPP zabezpečit pomocí MPPE. Autentičnost a integrita zpráv není vůbec zajištěna. Potenciální útočník sice nemůže vidět zašifrovaná data v PPP, ale může upravovat řídící a datové zprávy. Samotný L2TP se v současnosti zřídka používá z bezpečnostních hledisek. Naopak v kombinaci s IPsec je jeden z nejrozšířenějších a nejbezpečnějších tunelovacích protokolů v současnosti. [8]

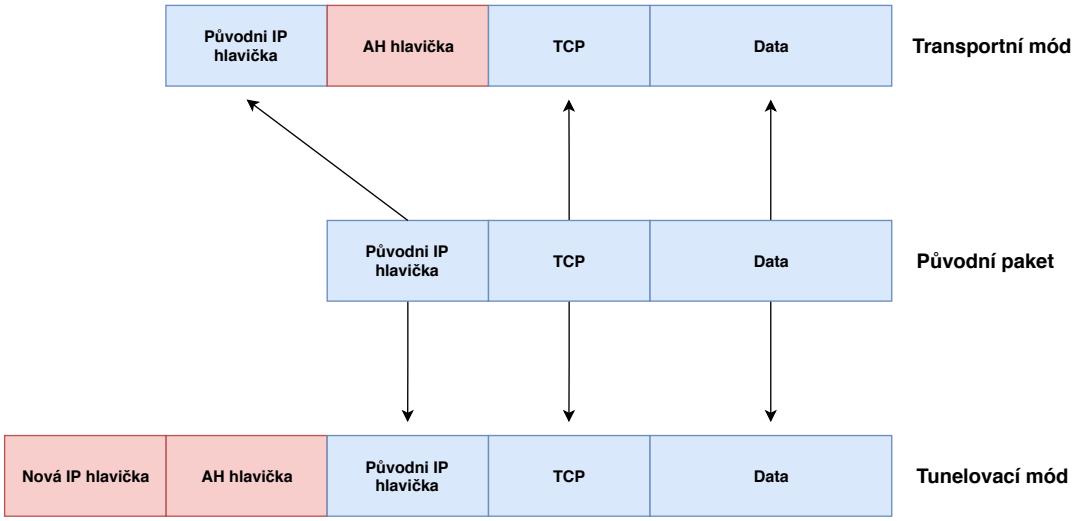
1.4.4 IPsec

IP Security je bezpečnostní rozšíření IP protokolu, umožňující autentizaci a šifrování každého přenášeného paketu. V ISO/OSI modelu se jedná o zabezpečení na síťové vrstvě. Podporuje IPv4 a v IPv6 je nativně implementovaný. IPsec zajišťuje autentičnost, integritu a důvěrnost IP komunikace. Základem jsou mechanizmy AH (*Authentication Header*) a ESP (*Encapsulation Security Payload*). Protokol AH nebo ESP se vždy používá ve spojení se *Security Associations* – SA, který poskytuje algoritmy a klíčové mechanismy pro získání parametrů, které jsou vyžadovány AH a

ESP. Autentizace se provádí prostřednictvím certifikátů, nebo pomocí předsdíleného klíče, a nelze ji zakázat.

Mechanismy AH a ESP mohou být použity samostatně nebo společně.

- **AH** – tento protokol zajišťuje autenticitu a integritu dat odesílatele, ale samotná data šifrovaná nejsou. Zaručuje také ochranu proti opakování relace (*Replay Attack*). Používá následující hashovací algoritmy: HMAC–SHA1–96, AES–128–GMAC, AES–256–GMAC, HMAC–SHA2–256, HMAC–SHA2–512 a AES–XCBC–MAC–96. Jak vypadá struktura paketu při použití AH v tunelovacím a transportním režimu můžeme vidět na obr. 1.4.



Obr. 1.4: AH v transportním a tunelovacím módu

- **ESP** - cílem tohoto protokolu je zajistit důvěryhodnost dat v IP paketu. Dále zajišťuje autentičnost odesílatele, integritu dat a šifrování. Pro šifrování používá algoritmy NULL, 3DES-CBC, AES-CBC-128, AES-CBC-192, AES-CBC-256 a AES-CCM-8, AES-GCM-16 a CHACHA20-POLY1305. Pro integritu a autentizaci pak HMAC-SHA1-96, AES-XCBC-MAC-96, AES-128-GMAC, AES-256-GMAC, HMAC-SHA2-256 a HMAC-SHA2-512. Jak vypadá struktura paketu při využití ESP v tunelovacím a transportním režimu můžeme vidět na obr. A.1.

IPsec používá 2 pracovní režimy:

- Transportní režim
- Tunelovací režim

U transportního režimu se žádný tunel nevytváří a IP pakety jsou pouze vybavené AH, ESP anebo oběma hlavičkami. Proto se tento režim používá v kombinaci s jiným tunelovacím protokolem, kde nejpopulárnějším je L2TP a vznikne L2TP/IPsec.

Tunelovací režim vytváří tunel, kde k původnímu IP paketu je přidána nová IP hlavička a hlavičky AH/ESP. Architektura IPsec se postupem času mění a umožňuje použití různých algoritmů pro integritu, autentičnost a šifrování. Přidávají se nové a povinnost či volitelnost jednotlivých algoritmů se také mění.

Aby mohli strany spolu komunikovat za použití AH a ESP, je nutné, aby se předem domluvili na protokolech, klíčích a parametrech, které budou používat. SA obsahuje následující věci:

- Zdrojovou a cílovou IP adresu
- Identifikátor bezpečnostního protokolu (AP nebo ESP)
- Security Parameter Index (SPI) - 32bitové číslo přenášené v otevřené formě v hlavičce paketu

Jednotlivé SA jsou uloženy v databázi SAD - *Security Association Database*. Většinou jsou SA a klíče ustanoveny automaticky, a to pomocí protokolu ISAKMP – *Internet Security Association and Key Management Protocol*. Protokol ISAKMP je používán pro vyjednávání, ustanovení, úpravu a odstraňování jednotlivých SA. ISAKMP používá protokol UDP a port 500. Uvnitř ISAKMP funguje IKE – *Internet Key Exchange*, který slouží k ustanovení klíčů komunikujících stran. IKE pro toto používá algoritmus Diffie-Hellman, předsdílený klič, nebo systém veřejných klíčů. SA specifikuje, jak chránit datový provoz, ale je potřeba také definovat který datový provoz je potřeba chránit. Tahle informace je uložena v SP – *Security Policy*, které jsou uloženy v databázi SPD – *Security Policy Database*. SP tedy určuje, jestli budou na paket aplikovány bezpečnostní mechanizmy protokolu IPsec, anebo jestli paket bude odmítnut.

Pro správnou funkčnost je potřeba mít povolené tyto porty a protokoly: protokol 50 - ESP, protokol 51 - AH, UDP 500 ISAKMP/IKE, UDP 4500 NAT-TRAVERSAL. Protokol IPsec je vysoce konfigurovatelný, s tím ale je také spojená kritika na náročnost zprovoznění. IPsec je snadno detekovatelný kvůli ESP a AH, které jednoznačně identifikují IPSec komunikaci. Má také problémy s přecházením přes NAT.[9]

Velmi populární je kombinace L2TP/IPsec. Při použití tohoto spojení není potřeba mít otevřený UDP port 1701, používán protokolem L2TP, protože UDP paket je zapouzdřen v ESP.

1.4.5 OpenVPN

Open-source OpenVPN byl vytvořen v roce 2001 Jamesem Yonanem a je distribuován pod licencí GNU General Public Licence. Podporuje většinu běžných operačních systémů a má velice bohatou dokumentaci. OpenVPN je vyvíjena s cílem snadné konfigurovatelnosti, pro kterou ale není obětována bezpečnost. Může využívat druhou nebo třetí ISO/OSI vrstvu. V době psaní této práce je nejaktuálnější verze 2.4.8[10]

a od verze 2.3.x podporuje IPv6. OpenVPN, na rozdíl od protokolu IPsec, pracuje v uživatelském prostředí operačního systémů a není součástí jádra. Vytváří VPN pomocí SSL/TLS - *Secure Socket Layer/Transport Layer Security*. Přesněji rozšířenou opec-source knihovnou OpenSSL. Tato knihovna poskytuje:

- Hashovací funkce – algoritmy MD5, MD4, MD2, SHA-1, SHA-2, SHA-3, MDC-2, RIPEMD-160, GOST R 34.11-94, BLAKE2, Whirpool, SM3
- Symetrické šifrování – algoritmy AES, Blowfish, Camellia, SEED, CAST-128, ChaCha20-Poly1305, DES, IDEA, RC2, RC4, RC5, TripleDES, GOST28147-89, SM4
- Asymetrická kryptografie – algoritmy RSA, DSA ,Diffie-Hellman, kryptografie nad eliptickými křivkami, X25519, ED25519, X448, ED448, GOST R34.10-2001, SM2

Komunikace je šifrovaná pomocí symetrických šifer, které jsou použity z důvodu menší výpočetní náročnosti. Typ šifry a délku klíče si lze nastavit. Je podporováno i použití HMAC (*Hash Message Authentication Code*).

Organizace IANA (*Internet Assigned Numbers Authority*) oficiálně protokolu OpenVPN přiřadila port 1194 TCP/UDP, ale výchozí port může byt změněn na libovolný port. Strukturu paketu při použití OpenVPN můžeme vidět na obr. 1.5.

Při konfiguraci OpenVPN je možnost si volit mezi dvěma virtuální síťové rozhraními:

- Rozhraní TUN – vychází ze slova *tunnel* a simuluje zařízení na 3. vrstvě ISO/OSI modelu. Lze jím přenášet jakoukoliv IP komunikaci a pro propojené více TUN rozhraní je nutné použít směrování.
- Rozhraní TAP – název vychází z *network tap* simuluje zařízení Ethernet na 2. vrstvě ISO/OSI modelu. Pracuje s Ethernet rámci a umožňuje přenos i jiných protokolů, než je IP protokol. TAP zařízení lze propojit pomocí síťových mostů.

Metody autentizace v OpenVPN:

- Autentizace na základě uživatelského jména a hesla
- Předsdílený klíč
- X.509 digitální certifikát

K autentizaci je možné také použít čipovou kartu pomocí PKI-PKCS#11 (*Public Key Infrastructure - Public Key Cryptography Standard*).

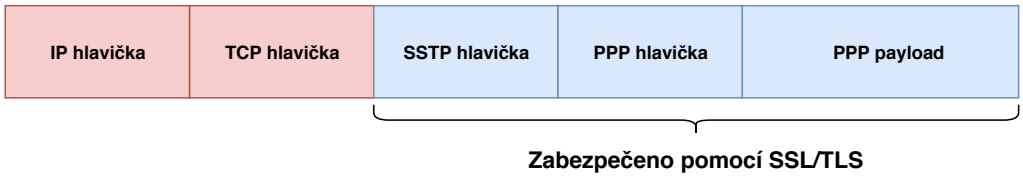
OpenVPN je jeden z nejpopulárnějších a nejbezpečnějších[11] VPN řešení. Výhoda open-source je, že zranitelnost či nedostatek je rychle objeven a opraven. Je podporován na Linuxu, Windows XP a dále, Mac OS X, FreeBSD, OpenBSD, NetBSD, Solaris[12] a pomocí OpenSSL může OpenVPN pracovat s nejmodernějšími kryptografickými metodami. OpenVPN je vysoce upravitelný (TCP/UDP, různé porty, volba zabezpečení atd.) a má snadný průchod přes NAT a PROXY.[13]

Obr. 1.5: Struktura OpenVPN paketu

1.4.6 SSTP

Protokol *Secure Socket Tunneling Protocol* byl vyvinut společností Microsoft v roce 2007. Nejnovější verze je 18.0, která byla zveřejněna na konci roku 2018[14]. Je implementovaná v operačních systémech od Windows Vista SP1 a Windows Server 2008 dále. Umožňuje přenos PPP paketů stejně jako PPTP a L2TP, ale celý SSTP protokol je zapouzdřen v SSL/TLS vrstvě. Použití protokolu PPP umožňuje podporu autentizačních metod jako je MS-CHAP a EAP-TLS. Využívá port 443 (HTTPS) a díky tomu má dobré maskování. Díky využití SSL/TLS je zajištěna integrita, autentičnost a důvěrnost. Nemá problém s průchodem přes NAT a firewall. Protokol má vysokou úroveň bezpečnosti a ochranu proti MITM (*Man-in-the-Middle*) útokům. Strukturu paketu při použití tohoto protokolu můžeme vidět na obr. 1.6.

Nevýhodou tohoto protokolu je slabší podpora v operačních systémech. Server může být pouze systém Windows 7 a novější, kde klient je také Windows systém. Pokud je jako server používán Windows Server 2008 nebo RouterOS 5.0 a novější, existuje implementace pro klienty používající Linux a Mac OS X[15]. Kvůli přenosu pouze přes port 443 TCP je náchylný na *TCP Meltdown*. Protokol SSTP má vyšší latenci oproti ostatním protokolům.



Obr. 1.6: Struktura SSTP paketu

1.4.7 WireGuard

WireGuard je open-source komunikační protokol napsaný Jasonem A. Donenfeldem a distribuovaný pod licencí GNU GPLv2. Jeho cíl je být jednoduchý na nastavení a při tom být vysoce bezpečný. WireGuard má pouze 4000 řádků kódu (pro srovnání OpenVPN s OpenSSL má zhruba 600 000 řádků a IPsec okolo 400 000 řádků),

což znamená mnohem méně útočné plochy k nalezení chyb a zranitelností. WireGuard se záměrně vyhýbá standardním algoritmům, které mají dlouhodobé slabiny ve prospěch pouze nejnovějších. Používá Noise Protocol Framework, Curve25519, ChaCha20, Poly1305, BLAKE2, SipHash24 a HKDF. Používá UDP s libovolným portem a umí zapouzdřovat IPv4 v IPv6 a naopak. Kombinací extrémně vysokorychlostních kryptografických primitivů a skutečnost, že WireGuard pracuje uvnitř Linux jádra znamená, že zabezpečená síť je velice rychlá. Nastavení WireGuardu je velice jednoduché a pro ustanovení spojení je potřeba pouze výměna IP adres a veřejných klíčů mezi klientem a serverem. V čase psaní této práce nemá WireGuard stabilní verzi 1.0. Aktuálně podporuje Windows 7 a výše, Android, iOS, macOS, BSD, Linux.[16] [17]

V srpnu roku 2018 Linus Torvalds, tvůrce Gitu a jádra Linuxu, napsal v *Linux Kernel Mailing List* o Wireguardu následující: *„Mohu ještě jednou vyjádřit svou lásku k WireGuardu a doufám, že se brzy mergne? Možná, že kód není dokonalý, ale prolítl jsem ho a ve srovnání s hrůzami, které jsou OpenVPN a IPSec, jde o umělecké dílo.“*[18] [1] To získalo pozornost veřejnosti a WireGuard se začalo více mluvit. Strukturu paketu př použití WireGuard můžeme vidět na tomto obrázku obr. 1.7.

Obr. 1.7: Struktura WireGuard paketu

[1] Can I just once again state my love for WireGuard and hope it gets merged soon? Maybe the code isn't perfect, but I've skimmed it, and compared to the horrors that are OpenVPN and IPSec, it's a work of art.

2 RaspberryPi

Raspberry Pi je jednodeskový počítač, který vznikl v roce 2012 pod britskou nadací *Raspberry Pi Foundation*. Hlavním cílem bylo vytvoření finančně dostupného zařízení, které by pomohlo studentům škol a v rozvojových zemích, rozvíjet programování a pochopit základy hardwaru. Existují různé generace a modely, ale cíl je vždy stejný, a to je být malý, kompaktní a levný počítač přístupný pro každého. Jak vypadá Raspberry Pi můžeme vidět na obr. 2.1. Jedna generace má většinou model A a model B, kde model A je zjednodušený a levnější oproti modelu B. Pak ještě existují modely plus, které nabízejí vylepšenou verzi modelu. Raspberry Pi se stal velice rychle celosvětově populární hlavně kvůli nadšencům. Na straně softwaru je mnoho možností - Raspbian, založen na Debianu a poskytuje ho *Raspberry Pi Foundation*, Arch Linux, Risc OS, Plan 9 from Bell Labs, Windows 10 IoT Core, FreeBSD, CentOS a mnoho dalších. Nejpopulárnější z těchto operačních systému je Raspbian, kvůli podpoře a dostupnosti nástrojů, ale uživatel má možnost si vybrat operační systém, který mu nejvíce vyhovuje a je mu dobře známý. Raspbian byl v květnu 2020 přejmenován na Raspberry Pi OS a operační systém je nyní k dispozici i v 64bitové verzi. Nejnovější model je Raspberry Pi 4 B, která má 4 varianty podle velikosti RAM (*Random Access Memory*), a to 1, 2, 4 nebo 8 GB. [19] [20] [21]

2.1 Raspberry Pi 1

První generace byla uvedena na trh v únoru 2012, a to prvním modelem byl Raspberry Pi 1 Model B, který stál 35 dolarů. Rozměry počítače jsou pouhých 85,60 mm na 56,5 mm a váží 45 gramů. Je vybaven procesorem BCM2835 ARM1176JZF-S 700MHz. Jako grafický čip používá VideoCore IV 250 MHz podporující OpenGL ES 2.0 a H.264. Počítač má pro video I/O HDMI (*High-Definition Multimedia Interface*), RCA jack a DSI (*Display Serial Interface*). Pro audio je dostupný 3,5mm jack. Originální model B má 256 MB operační paměti RAM sdílené s grafickým čipem, ale později byl vydán s 512 MB operační paměti. Jako externí úložiště je možné použít SD (*Secure Digital*) kartu, MMC (*MultiMediaCard*) kartu nebo SDIO (*Secure Digital Input Output*) kartu. Pro připojení k internetu je zde 10/100 Mbit/s USB (*Universal Serial Bus*) Ethernet adaptér. Má také 2 USB 2.0 porty. Později byl vydán model A za nižší cenu 25 dolarů, kterému byl odstraněn USB port, velikost RAM byla na 256 MB a odstraněno bylo Ethernet rozhraní.

V roce 2014 pak vyšly plusové modely A+ a B+. U obou byla snížena cena, oproti normálním modelům, a přidána podpora pro MicroSDHC (*Secure Digital High Capacity*) kartu. Model B+ také dostal 2 USB 2.0 porty navíc, takže nyní měl celkově 4.

2.2 Raspberry Pi 2

Druhá generace byla vydána v roce 2015 a měla pouze model B, který výbavou byl nejpodobnější modelu B+ předchozí generace. Došlo k velkému zvýšení výpočetního výkonu procesoru, který nyní měl 4 jádra Cortex-A7 900 MHz, a operační paměť byla zvýšena na 1 GB.

Později, v roce 2016, vyšla nová verze modelu B, která měla ještě výkonnější procesor s 4 jádry Cortex-A53 900MHz.

2.3 Raspberry Pi Zero

Hlavní zaměření této verze je na malou velikost a cenu, kdy model Zero má pouhých 65 mm na 30 mm, váží 9 gramů a stojí 5 dolarů. Výbavou připomíná model A+ 1. generace Raspberry Pi. Má stejný procesor, jen nataktovaný na 1 GHz, a místo HDMI portů používá Mini-HDMI. První model Zero vyšel v listopadu roku 2015 a v únoru 2017 vyšla novější verze Zero W, kde W značí *Wireless*. Tato nová verze stojí dvojnásobnou cenu, ale nabízí podporu WiFi 2,4GHz a Bluetooth 4.1 BLE (*Bluetooth Low Energy*).

2.4 Raspberry Pi 3

Třetí generace má modely A+, B a B+. Model B vyšel na začátku roku 2016 a byl opět podobný poslední iteraci předchozí generace. Takt procesoru byl zvýšen na 1,2 GHz a největší rozdíl od 2. generace je v přidání modulu WiFi 2,4 GHz a technologie Bluetooth 4.1 BLE.

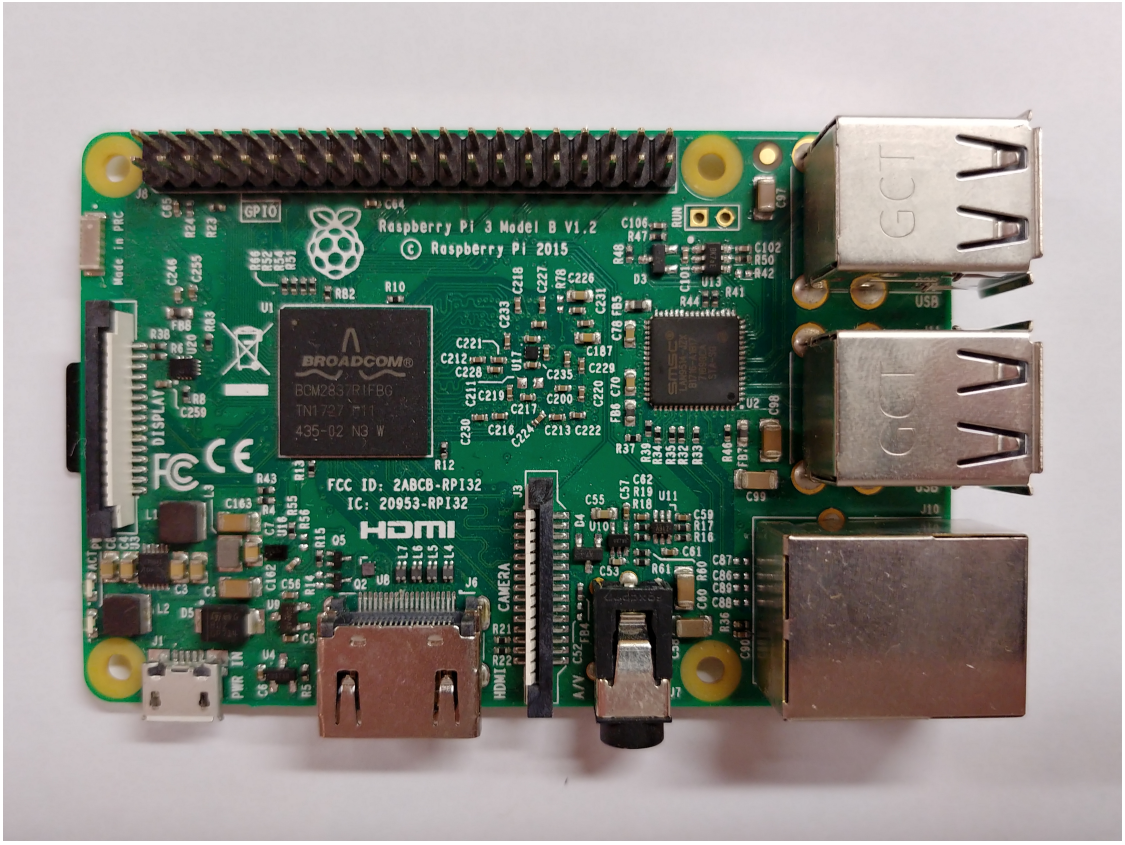
O více jak 2 roky později, v březnu 2018, vyšel model B+. Ten zvýšil dále takt procesoru o 200 MHz, přidal podporu WiFi 5 GHz a Bluetooth 4.2 BLE. Velká změna byla přidání Gigabit Ethernetu, který je limitován na 300 Mbit/s z důvodu použití USB 2.0 adaptéru.

Ke konci roku 2018 vyšel model A+, který sdílí stejný procesor o stejné rychlosti, jako model B+, a má stejné moduly WiFi a Bluetooth. Má však pouze 1 USB 2.0 port a postrádá Ethernet rozhraní.

2.5 Raspberry Pi 4

Nejnovější model 4 vyšel v červnu roku 2019 se 4 jádrovým ARM Cortex-A72 procesorem nataktovaným na 1,5 GHz, novým grafickým čipem Broadcom VideoCore VI 500 MHz, Bluetooth 5.0, nelimitovaným Gigabit Ethernetem, 2 USB 2.0 a 2 USB 3.0

porty, HDMI port byl nahrazen dvěma miniHDMI porty, čímž podporuje současně dva monitory, a to v rozlišení až 4K. Napájecí port microUSB byl nahrazen USB-C portem. Má různé varianty podle velikosti RAM - 1 GB, 2 GB a 4 GB, které stojí 35, 45 a 55 dolarů. V květnu roku 2020 byla přidána ještě 4. varianta, která má 8 GB RAM a stojí 75 dolarů[22].



Obr. 2.1: Raspberry Pi 3 B

3 Implementace zvolených VPN protokolů

V praktické části byly zvolené protokoly implementovány. Jedná se o PPTP, IPsec, OpenVPN a WireGuard. Protokoly IPsec a OpenVPN byly zvoleny z důvodu popularity a rozšíření těchto řešení. PPTP byl zvolen pro porovnání této starší technologie oproti novějším řešením. WireGuard z důvodu opačného, kdy se jedná ze zmíněných řešení o nejmladší a nejméně v praxi otestované.

Nejdříve pro účely simulace byly jako VPN servery použité čisté virtuální stroje[23] s operačním systémem Debian 10 Buster[24], z důvodu podobnosti k operačním systému Rasbian, použitého na Raspberry Pi. Po otestování správnosti konfigurace, kdy klient se úspěšně připojil přes VPN server, bylo řešení přeneseno na Raspberry Pi 3 B.

3.1 PPTP

Pro implementaci PPTP je nainstalován *pptpd*[25], což je Poptop PPTP daemon, nejpopulárnější open-source PPTP server.

Výpis 3.1: Instalace pptpd

```
# apt-get install pptpd
```

Po instalaci bude otevřen konfigurační soubor *pptpd-options*, který se nachází ve složce */etc/ppp/*. Zde se dá zvolit, které DNS (*Domain Name System*) servery budou použity. V tomto případě jsou zvoleny servery 8.8.8.8 a 8.8.4.4, což jsou DNS servery Googlu. Další populární možnost jsou OpenDNS servery 208.67.222.222 a 208.67.220.220. Budou přidána i pravidla, aby server odmítal jinou autentizaci, než je MS-CHAPV2 a šifrování dat v PPP MPPE128.

Výpis 3.2: pptpd-options

```
name pptpd
 refuse-pap
 refuse-chap
 refuse-mschap
 require-mschap-v2
 require-mppe-128
 ms-dns 8.8.8.8
 ms-dns 8.8.4.4
```

Další konfigurační soubor, který je potřeba upravit, je *pptpd.conf*, který se nachází ve složce */etc/*. Zde bude zvolena IP adresa použitého VPN serveru, v tomto

případě adresa 192.168.0.1, a rozsah IP adres pro klienty, což je nastavené na 192.168.0.100 až 192.168.0.200.

Výpis 3.3: IP adresy serveru a klientů

```
localip 192.168.0.1
remoteip 192.168.0.100-200
```

Aby se klient mohl připojit na VPN sever, musí mít v souboru *chap-secrets*, který se nachází ve složce */etc/ppp/*, svoje přihlašovací jméno a heslo. První sloupec je uživatelské jméno, druhý sloupec je název serveru, třetí je heslo uživatele a poslední sloupec je IP adresa, kde v tomto případě je znak *, což povoluje všechny IP adresy. Na server mají tedy povolený přístup 2 uživatelé, a to uživatel *user1* s heslem *userpsw* a uživatel *user2* s heslem *a8r1S54r*.

Výpis 3.4: Přidání klientů

```
user1 pptpd userpsw *
user2 pptpd a8r1S54r *
```

Aby se systém choval jako server, tak je nutné zapnout *IPv4 forwarding*. To se dá provést odkomentováním následujícího příkazu v souboru */etc/sysctl.conf*.

Výpis 3.5: IPv4 forwarding

```
net.ipv4.ip_forward = 1
```

Pro aplikování změněného souboru je potřeba restartovat systém, nebo použít tento příkaz:

Výpis 3.6: Aplikace forward pravidel

```
# sysctl -p
```

Je nutné také upravit pravidla firewallu následujícími pravidly:

Výpis 3.7: Firewall pravidla

```
iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE

iptables -A FORWARD -i enp0s3 -o ppp0 -m state \
    --state RELATED,ESTABLISHED -j ACCEPT

iptables -A FORWARD -i ppp0 -o enp0s3 -j ACCEPT
```

Tímto je PPTP server nastavený a připravený na použití. Poslední krok, který zbývá provést na serveru, je ho zapnout. To se provede prvním příkazem *systemctl*

start pptpd. V případě, že je potřeba, aby se server nastartoval vždy při spuštění systému, použije se druhý příkaz *systemctl enable pptpd.service*.

Výpis 3.8: Nastartování PPTP sevreru

```
# systemctl start pptpd

# systemctl enable pptpd.service
```

Jako klient byl použitý stroj s operačním systémem Windows 7 a jeho vestavěné funkce vytvoření VPN připojení. V menu VPN byla zadána IP adresa VPN serveru PPTP, která je 10.0.10.7. Jak toto vypadá je ukázáno na obr. A.2. Následně se vyplní jméno a heslo, jako je ukázáno na obr. A.3. Systém otestuje, jakou autentizaci může použít a následně úspěšně proběhne navázání spojení mezi VPN serverem a klientem.

3.2 IPsec

Bude využita strongSwan implementace IPsec. StrongSwan [26] je pokračování projektu FreeS/WAN a je distribuován pod licencí GNU GPL. Nejaktuálnější verze je 5.8.1 [27], která vyšla na začátku září 2019.

Nejprve je nutné na server nainstalovat strongSwan a balíčky *libcharon-extra-plugins* a *libstrongswan-extra-plugins*.

Výpis 3.9: Instalace strongSwan

```
# apt-get install strongswan strongswan-pki
# apt-get install libcharon-extra-plugins
# apt-get install libstrongswan-extra-plugins
```

Bude použitý IKEv2, a proto je nutné vytvořit X.509 certifikáty pro CA (*Certificate Authority*) - Certifikační Autoritu a server. Je potřeba vytvořit nové složky, do kterých se tyto klíče uloží. Začne se certifikátem podepsaným sám sebou. Nejprve proběhne generování 4096bitového RSA privátního klíče *ca-key.pem*, který následně bude použit na podepsání kořenové certifikátu *ca-cert.pem*, s platností na 3650 dní (10 let) a kde *dn* značí *distinguished name*, a tím bude vytvořena kořenová CA.

Výpis 3.10: Vytvoření složek, CA klíč a certifikát

```
# mkdir -p ~/pki/{cacerts,certs,private}

# ipsec pki --gen --type rsa --size 4096 \
    --outform pem > ~/pki/private/ca-key.pem

# ipsec pki --self --ca --lifetime 3650 \
    --in ~/pki/private/ca-key.pem --type rsa \
    --dn "CN=VPN root CA" \
    --outform pem > ~/pki/cacerts/ca-cert.pem
```

Nyní je možné vytvořit certifikát a klíč pro VPN server. Tento certifikát umožní klientům ověřit si autenticitu serveru. Prvně je opět nutné vytvořit 4096bitový RSA privátní klíč *server-key.pem*.

Výpis 3.11: Privátní klič serveru

```
# ipsec pki --gen --type rsa --size 4096 \
    --outform pem > ~/pki/private/server-key.pem
```

Pak bude vytvořen certifikát VPN serveru a podepsán klíčem CA. Veřejný klíč serveru bude extrahován a použit ve vytvoření certifikátu serveru *sever-cert.pem*. Tady *san* značí *Subject Alternate Name* a spolu s *dn* bude změněn na IP adresu VPN serveru, což je adresa 10.0.10.11. Certifikát má platnost 1825 dní (5 let). Část *–flag serverAuth –flag ikeIntermediate* bude použita z důvodu kompatibility, *–flag serverAuth* pro klienty používající Windows 7 a *–flag ikeIntermediate* pro klienty používající macOS 10.7.3 a starší.

Výpis 3.12: Ceritifikát serveru VPN

```
# ipsec pki --pub --in \
    /root/pki/private/server-key.pem --type rsa |
    ipsec pki --issue --lifetime 1825 \
    --cacert /root/pki/cacerts/ca-cert.pem \
    --cakey /root/pki/private/ca-key.pem  \
    --dn "CN=10.0.10.11" --san "10.0.10.11" \
    --flag serverAuth --flag ikeIntermediate \
    --outform pem > /root/pki/certs/server-cert.pem
```

Po vygenerování všech předešlých klíčů a certifikátů, budou tyto soubory přesunuty do složky */etc/ipsec.d/*.

Výpis 3.13: Přesun obsahu složek

```
# cp -r ~/pki/* /etc/ipsec.d/
```

V souboru */etc/ipsec.conf*, který lze vidět vy výpisu B.1, se nachází ukázka nastavení od strongSwan, která bude dále upravena. Na druhém řádku se upravuje logování, kde „0“ znamená základní kontrolní logy, „1“ značí obecný řídící tok s chybami, *knl* je *IPsec/Networking kernel interface* a *cfg* je *Configuration management and plugins*. Negativní možnost *uniqueids* umožňuje připojení více klientů se stejným certifikátem a/nebo heslem. V případě, že by byla tato možnost zapnutá, tak nově připojený klient by nahradil původně připojeného klienta.

Následuje sekce, která sděluje strongSwan, aby vytvořila IKEv2 VPN tunely s danou konfigurací. Další 3 řádky jsou pro čištění spojení, kde se klient nečekaně odpojí. Dále je levá a pravá strana tunelu. Levá je IPsec server, kde je specifikována IP adresa VPN serveru a certifikát serveru. Pravá strana je klientská, kde je nastaven typ autentizace, použité DNS servery a rozmezí IP adres. Možnost *eap_identity=%identity* značí, že musí klient při připojení použít svoje přihlašovací údaje. Poslední 2 řádky v konfiguračním souboru jsou pro zlepšení interoperability s operačními systémy Windows, macOS a iOS.

V souboru */etc/ipsec.secrets* je nutné sdělit strongSwan, kde najde privátní klíč serveru a seznam uživatelů, kteří se mohou k VPN serveru připojit. V tomto případě soubor obsahuje jednoho klienta, který se může připojit s přihlašovacím jménem *klient* a heslem *psw123*.

Výpis 3.14: ipsec.secrets

```
: RSA "server-key.pem"
klient : EAP "psw123"
```

Pro pracování s firewall je zde použit UFW - *Uncomplicated Firewall* a jsou povolené potřebné funkce pro normální provoz, jako např. HTTP (*Hypertext Transfer Protocol*), HTTPS (*HyperText Transfer Protocol Secure*), SSH (*Secure Shell*) a podobné. Následně byly povolené nutné porty pro fungování IPsec VPN, což jsou porty UDP 500 (ISAKMP) a 4500 (NAT-T).

Výpis 3.15: Instalace UFW a povolené porty

```
# apt-get install ufw
# ufw allow ssh
# ufw allow 500,4500/udp
```

V souboru */etc/ufw/before.rules* budou přidána nová pravidla. V sekci **nat* se upravují pravidla pro správné směrování a manipulování provozu mezi VPN klientem a Internetem. Sekce **mangle* upravuje maximálního velikost paketu, aby předešla potenciálním problémům s některými VPN klienty.

Výpis 3.16: before.rules

```
*nat
-A POSTROUTING -s 172.16.0.0/12 -o enp0s3 -m policy \
    --pol ipsec --dir out -j ACCEPT

-A POSTROUTING -s 172.16.0.0/12 -o enp0s3 -j MASQUERADE
COMMIT

*mangle
-A FORWARD --match policy --pol ipsec --dir in \
    -s 172.16.0.0/12 -o enp0s3 -p tcp -m tcp \
    --tcp-flags SYN,RST SYN -m tcpmss  \
    --mss 1361:1536 -j TCPMSS --set-mss 1360
COMMIT
```

Na konec souboru budou přidány ještě pravidla pro zacházení s ESP provozem.

Výpis 3.17: before.rules konec souboru

```
-A ufw-before-forward --match policy --pol ipsec \
    --dir in --proto esp -s 172.16.0.0/12 -j ACCEPT

-A ufw-before-forward --match policy --pol ipsec \
    --dir out --proto esp -d 172.16.0.0/12 -j ACCEPT
```

Je potřeba upravit soubor */etc/ufw/sysctl.conf*. V něm bude zapnuto *IPv4 forwarding* a vypnuto *accept_redirects* a *send_redirects*, aby se předešlo *Man-in-the-middle* útokům.

Výpis 3.18: sysctl.conf

```
net/ipv4/ip_forward = 1
net/ipv4/conf/all/accept_redirects = 0
net/ipv4/conf/all/send_redirects = 0
```

Na serveru ještě zbývá nastartování nyní nakonfigurovaného UFW a strongSwan, což bude provedeno následujícími příkazy:

Výpis 3.19: Start ufw a strongSwan

```
# ufw enable
# service ipsec restart
```

Nezbytné je překopírovat vygenerovaný CA certifikát *ca-cert.pem* na systém klienta. To lze provést například pomocí protokolu SFTP (*Secure File Transfer Protocol*), nebo pomocí USB flash disku. Jako klient je použit Windows 7 systém. Po tom, co se CA certifikát nachází na počítači klienta, se zapne *Windows Management Console*. Je nutné zvolit **File** -> **Add or Remove Snap-in**, dále možnost **Certificates** a kliknout na **Add**, jak lze vidět na obr. A.4. Dále vybrat možnost **Computer Account** a kliknout na **Next**. V dalším menu vybrat **Local Computer** a nakonec odsouhlasit tlačítkem **Finish**. V levém panelu rozkliknout položku **Trusted Root Certification Authorities** a zvolit **Certificates**. V menu zvolit **Action** -> **All Tasks** -> **Import**, kde se objeví průvodce importu certifikátů, kde se zvolí přenesený certifikát ze serveru a bude importován. Průvodce importování certifikátu lze vidět na obr. A.5 a pak již importovaný certifikát lze vidět na obr. A.6.

Nyní stačí otevřít vestavěnou funkci vytvoření VPN spojení a zadání IP adresy IPsec VPN serveru a přihlašovací jméno a heslo klienta, stejně jako bylo provedeno v implementaci PPTP 3.1. Následně se úspěšně naváže komunikace s VPN serverem.

3.3 OpenVPN

Bude nainstalován OpenVPN na server a klienta následujícím příkazem:

Výpis 3.20: Instalace OpenVPN

```
# apt-get install openvpn
```

Je nutné vygenerovat certifikát serveru a klíče. Skripty se nakopírují do jiné složky a následně ukázkový soubor *vars.example* bude duplikován do souboru *vars*, který dále bude upravován.

Výpis 3.21: Manipulace se soubory

```
# cp -r /usr/share/easy-rsa /etc/openvpn/
# cd /etc/openvpn/easy-rsa
# cp vars.example vars
```

Parametry pro certifikační autoritu v souboru *vars* budou změněny, a to pouze následující řádky:

Výpis 3.22: vars

```
export KEY_COUNTRY="CZ"
export KEY_PROVINCE=""
export KEY_CITY="Brno"
export KEY_ORG="VUT-VPN"
export KEY_EMAIL="xkovar79@stud.feec.vutbr.cz"
export KEY_OU="OpenVPN"
```

Inicializace nové PKI se provede následujícím příkazem:

Výpis 3.23: Inicializace PKI

```
# ./easyrsa init-pki
```

V případě, že není žádoucí, aby při podepisování certifikátů bylo požadováno zadávání hesla, použije se *nopass*. Při generování certifikátu se objeví dotaz na *Common Name*, kde bude zadána odpověď *server*.

Výpis 3.24: CA certifikát

```
# ./easyrsa build-ca nopass
```

Bude vygenerován klíč serveru. Při zeptání na *Common Name* bude už dříve zadaný název přednastaven, takže stačí pouze odsouhlasit.

Výpis 3.25: Klíč serveru

```
# ./easyrsa gen-req server nopass
```

Následně je potřeba podepsat certifikát, kde znovu jenom bude odsouhlaseno, že název je správný.

Výpis 3.26: Podepsání certifikátu

```
# ./easyrsa sign-req server server
```

Další bude vygenerován ještě *Diffie-Hellman* klíč a následně *HMAC* podpis.

Výpis 3.27: Diffie–Hellman klíč a HMAC podpis

```
# ./easyrsa gen-dh
# openvpn --genkey --secret ta.key
```

Co se týče klíčů serverů, zbývá je ještě nakopírovat nebo přesunout do správné složky.

Výpis 3.28: Kopírování souborů

```
# cp pki/issued/server.crt /etc/openvpn/
# cp pki/private/server.key /etc/openvpn/
# cp pki/dh.pem /etc/openvpn/
# cp pki/ca.crt /etc/openvpn/
# cp ta.key /etc/openvpn/
```

Nyní se vygenerují potřebné klíče pro klienta a jeho certifikát. Postup bude podobný jako při generování těchto věcí pro server. Při dotazu na *Common Name* se vloží název *client*.

Výpis 3.29: Vytvoření klientského certifikátu a jeho podepsání

```
# ./easyrsa gen-req client nopass
# ./easyrsa sign-req client client
```

A opět všechny tyto položky budou nakopírované do vyžadované složky.

Výpis 3.30: Kopírování souborů

```
# cp pki/ca.crt /etc/openvpn/client/
# cp pki/issued/client.crt /etc/openvpn/client/
# cp pki/private/client.key /etc/openvpn/client/
```

Následně tyto 3 soubory a ještě soubor *ta.key* se nakopírují na systém klienta, například protokolem SFTP, do složky */etc/openvpn/*.

Nyní nastane čas na konfigurování VPN serveru. Vytvoří se soubor */etc/openvpn/server.conf*, který lze vidět ve výpisu B.2, a do něho bude vloženo nastavení. Zde se nastaví, který port bude používán, protokol UDP, rozhraní TUN, kde lze nalézt certifikát CA a certifikát a klíče serveru. Nastavuje se také IP rozsah serveru, v tomto případě to je 10.8.0.0 s maskou 255.255.255.0, DNS servery které budou použity, což v tomto případě se jedná o OpenDNS servery. Položku *redirect-gateway def1 bypass-dhcp*, aby klient přesměroval svojí komunikaci skrze VPN server. Dále také specifikováno způsob a lokace logování.

Na serveru bude zapnut *IPv4 Forwarding* v souboru */etc/sysctl.conf*. Následně se provede restart serveru, aby se změna projevila.

Výpis 3.31: IPv4 Forwarding

```
net.ipv4.ip_forward = 1
```

Je nutné také upravit pravidla firewall, kde bylo znovu použito UFW. V souboru */etc/default/ufw* bude změněna položka *DEFAULT_FORWARD_POLICY="DROP"* na *DEFAULT_FORWARD_POLICY="ACCEPT"*. Do souboru */etc/ufw/before.rules* budou přidány nálsedující pravidla:

Výpis 3.32: before.rules

```
*nat
:POSTROUTING ACCEPT [0:0]

-A POSTROUTING -s 10.8.0.0/16 -o enp0s3 -j MASQUERADE

COMMIT
```

V UFW bude povolen port 1194 UDP a následně bude vypnut a zapnut firewall.

Výpis 3.33: UFW port 1194 UDP a restart

```
# ufw allow 1194/udp

# ufw disable
# ufw enable
```

Na klientovi bude vytvořen konfigurační soubor */etc/openvpn/client.conf*. V něm se nastaví například, který režim bude použit, použitý protokol UDP, IP adresa a port VPN serveru, lokaci klíčů a certifikátů atd.

Výpis 3.34: client.conf

```
client
dev tun
proto udp
remote 10.0.10.8 1194
resolv-retry infinite
nobind
user nobody
group nogroup
persist-key
persist-tun
ca ca.crt
cert client.crt
key client.key
remote-cert-tls server
tls-auth ta.key 1
cipher AES-256-CBC
verb 3
```

Nastartuje se nastavený VPN server a klient. Nyní spojení bude funkční. [28]

Výpis 3.35: Nastartování OpenVPN

```
# systemctl start openvpn@server
# systemctl start openvpn@client
```

3.4 WireGuard

WireGuard slibuje jednoduchou instalaci a konfiguraci VPN [16]. Následujícím příkazem bude WireGuard nainstalován na server a klienta.

Výpis 3.36: Instalace WireGuard

```
 # apt-get install wireguard
```

Pro generování klíčů na serveru bude použit následující příkaz:

Výpis 3.37: Generování klíčů serveru

```
# wg genkey |
tee wg-private.key |
wg pubkey > wg-public.key
```

Vygenerovány budou také klíče na klientovi.

Výpis 3.38: Generování klíčů klienta

```
# wg genkey |
tee client_private_key |
wg pubkey > client_public_key
```

Pokud je ve firewall zablokovaný zvolený port pro WireGuard, v tomto případě bude zvolen port 51820, je nutné ho povolit. Musí být také správně nastaven *IPv4 Forwarding* v souboru */etc/sysctl.conf*. Dále bude restartován server, aby se projevila změna, nebo lze použít tyto příkazy bez nutnosti restartování.

Výpis 3.39: Ipv4 Forwarding

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
# sysctl -p
```

Bude vytvořen konfigurační soubor pro server, a to v */etc/wireguard/*, a bude pojmenován *wg0.conf*. Zde se nastaví IP adresa VPN serveru a jeho privátní klíč, IP adresy povolených klientů a jejich veřejné klíče, na kterém portu bude server poslouchat a pravidla pro firewall. Pro tuhle konfiguraci je nutné přenést veřejné klíče serveru na klienta a naopak. K tomu lze například použít opět SFTP.

Výpis 3.40: wg0.conf

```
[Interface]
Address = 10.0.10.7/24
SaveConfig = true
PrivateKey = 8Heex3WIRBC8M4kL5/eXG/R3fpxQdnhW/6AddzHclUY=
ListenPort = 51820
PostUp = iptables -A FORWARD -i %i -j ACCEPT;
iptables -A FORWARD -o %i -j ACCEPT;
iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE

PostDown = iptables -D FORWARD -i %i -j ACCEPT;
iptables -D FORWARD -o %i -j ACCEPT;
iptables -t nat -D POSTROUTING -o enp0s3 -j MASQUERADE

[Peer]
PublicKey = tkN/ZGaa4Sasxbek1l1P0jnzXhgWHq3yuR4A/lZJ32I=
AllowedIPs = 10.0.10.6/32
```

Bude vytvořen konfigurační soubor pro klienta a to v */etc/wireguard/* a bude pojmenován *wg0-client.conf*. Zde se nastaví IP adresa klienta a jeho privátní klíč, IP adresa serveru, jeho port a veřejný klíč serveru. *PersistentKeepalive* značí, že jednou za x sekund se pošle paket, který umožňuje jednodušší přechod přes NAT a firewall. *AllowedIPs* nastavený na 0.0.0.0/0 nám udává, že klient může posílat svůj provoz na jakoukoliv IPv4 adresu.

Výpis 3.41: wg0-client.conf

```
[Interface]
Address = 10.0.10.6/32
PrivateKey = wD/6MAkSjkTcj7mNMD5iyhIZPERbVpQQS9V8xPioVVo=

[Peer]
PublicKey = WuJxT1nJlielXR46ipjrRKy78pzircCsmaJWAbmk+WY=
Endpoint = 10.0.10.7:51820
AllowedIPs = 0.0.0.0/0
PersistentKeepalive = 21
```

VPN server bude spuštěn a druhým příkazem se nastaví, aby se automaticky zapnul při nastartování systému.

Výpis 3.42: Start WireGuard

```
# wg-quick up wg0
# systemctl enable wg-quick@wg0.service
```

Následně bude spuštěn na klientovi WireGuard.

Výpis 3.43: Spuštění klienta

```
# wg-quick up wg0-client
```

Nyní bude fungující připojení přes VPN server.

4 Testování protokolů

Hlavním cílem této části bylo zjistit, zda je vhodné použít Raspberry Pi jako VPN server a jaký VPN protokol by byl nejvhodnější pro jeho realizaci. Pro zodpovězení těchto otázek byla změřena odezva od VPN serveru každého protokolu a propustnost vytvořené VPN sítě. Dále bylo naměřeno zatížení na hardware a stabilita jednotlivých protokolů při připojení více klientů.

Po nakonfigurování VPN protokolů na Raspberry Pi model 3 B, byla jednotlivá VPN řešení testována. Pomocí nástroje ping byla naměřena odezva od VPN serveru a Google DNS.

Dále byl pro testování použit open-source nástroj *iPerf*[29] verze 2.0.13., který je využíván pro měření výkonnosti sítě. Má stranu serveru a klienta, mezi kterými nástroj generuje datové proudy TCP nebo UDP. Nástroj *iPerf* byl spuštěný na Raspberry Pi s přepínačem *-s*, označující tento stroj za server očekávající připojení klienta a na klientech byl použitý přepínač *-c* s parametrem IP adresy VPN serveru, které udávají, kam se klient připojuje. Na obou stranách také přepínač *-t*, udávající na straně serveru, jak dlouho bude čekat na příchozí komunikaci, a na straně klienta, jak dlouho si bude data posílat se serverem. Při měření propustnosti sítě byl na klientovi přidán přepínač *-d* označující *dualtest* neboli oboustrannou komunikaci. Pro použití protokolu UDP místo TCP, musí být na obou stranách přidán argument *-u*, kde na straně klienta je nutné přidat ještě přepínač *-b*, označující *bandwidth*, a parametr, udávající limit šířky pásma. Jako poslední byl použitý na obou stranách přepínač *-i* a parametr „1“, díky kterému jsou hodnoty zaznamenané každou sekundu.

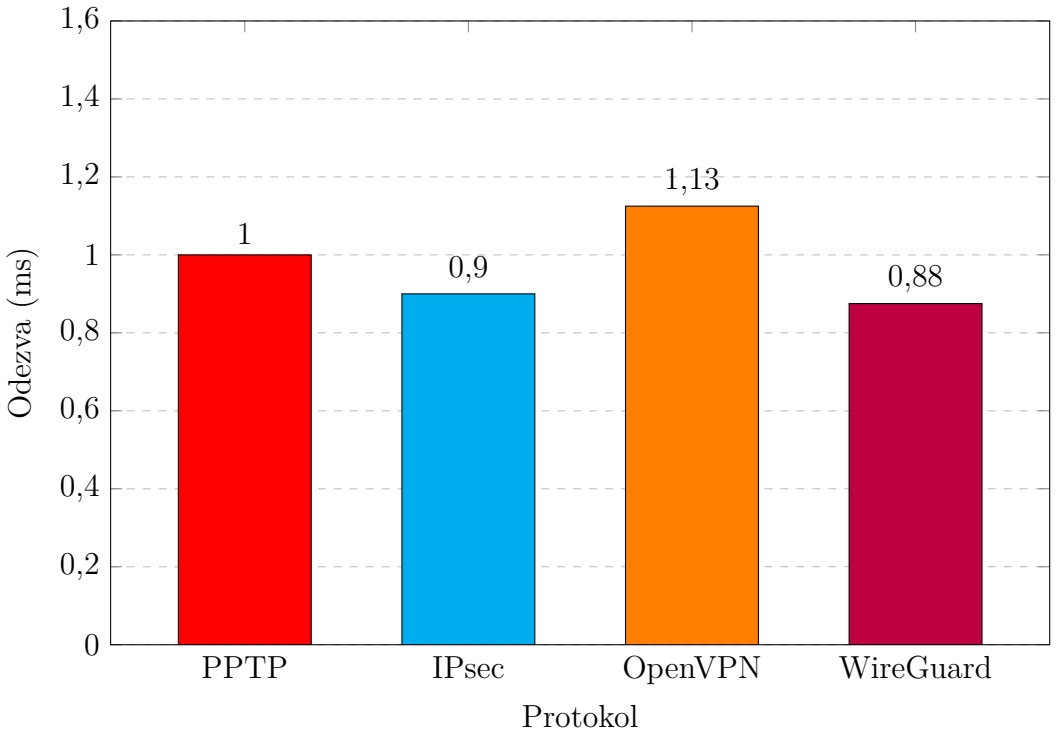
Pro testování zátěže na hardware byl použitý balíček *sysstat*[30], obsahující nástroje pro monitorování výkonu systému. Konkrétně byl z balíčku použitý nástroj *sar* (*System Activity Report*), který získává různé informace ze souborového systému */proc*. U nástroje byl na VPN serveru použitý přepínač *-u*, pro zjištění využití CPU, dále přepínač *-r*, pro zjištění využití RAM a parametr „1“ pro zaznamenání hodnot každou sekundu, dokud nebude proces zastaven.

4.1 Odezva

Nejdříve byla z klienta pomocí nástroje *ping* zaznamenaná odezva VPN serveru a Google DNS. Odezva byla měřena po dobu 10 sekund a z výsledných hodnot byl vypočten medián, který byl zaznačen do grafu. Výsledky jednotlivých protokolů v odezvě VPN serveru lze vidět v grafu 4.1.

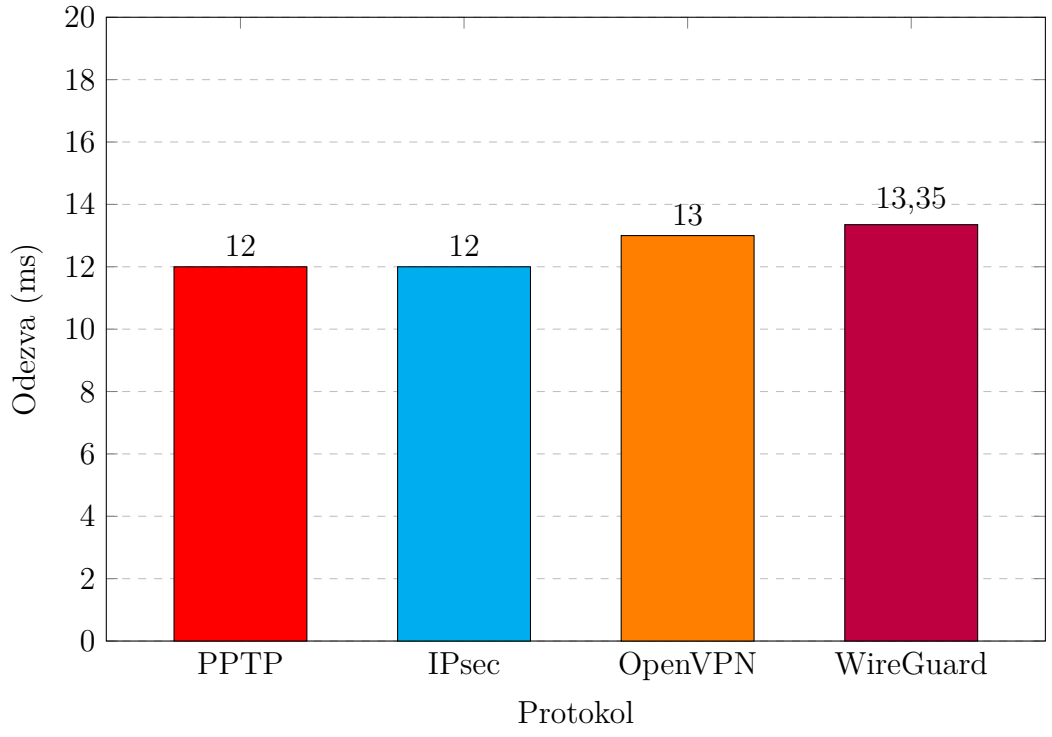
Mezi protokoly není značný rozdíl v odezvě, kde u VPN serveru rozdíl nepřesahuje 0,6 ms.

Obr. 4.1: Medián odezvy VPN



Hodnoty odezvy od Google DNS na IP adrese 8.8.8.8 jsou zaznačené na grafu 4.2 a zde maximální rozdíl činí 1,35 ms.

Obr. 4.2: Medián odezvy Google DNS



4.2 Hardwarová zátěž

Potřebné nástroje *sar* a *iPerf* byly nainstalované na Raspberry Pi pomocí následujících příkazů:

Výpis 4.1: Instalace iPerf a sysstat

```
# apt-get install sysstat
# apt-get install iperf
```

Na serveru byly oba tyto nástroje spuštěné pomocí těchto příkazů:

Výpis 4.2: sar a iPerf příkazy na Raspberry Pi

```
# sar -u 1
# iperf -s -t 100
```

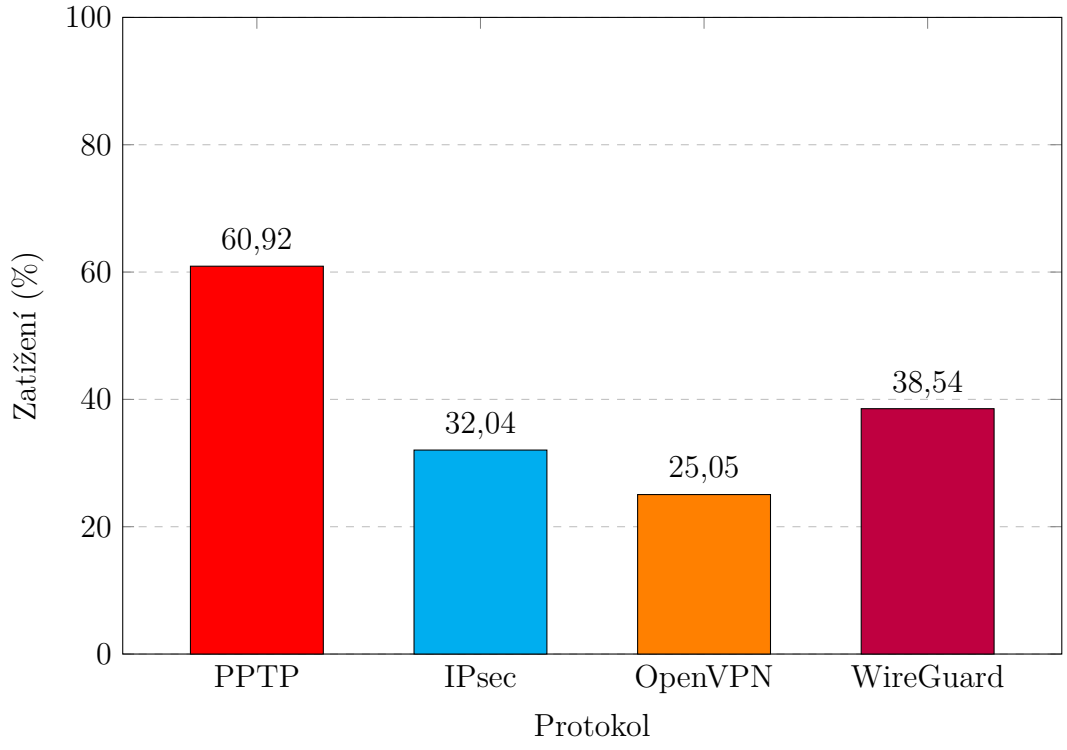
Na klientech byl spuštěný nástroj *iPerf* pomocí následujícího příkazu:

Výpis 4.3: iPerf příkaz na klientech

```
# iperf -c <IP adresa serveru> -d -t 60
```

Následující graf 4.3 ukazuje maximální využití CPU na serveru. Hodnoty byly naměřené pomocí nástroje *sar* při připojených 5 klientech, generující obousměrnou TCP komunikaci po dobu 60 sekund.

Obr. 4.3: Maximální zatížení CPU



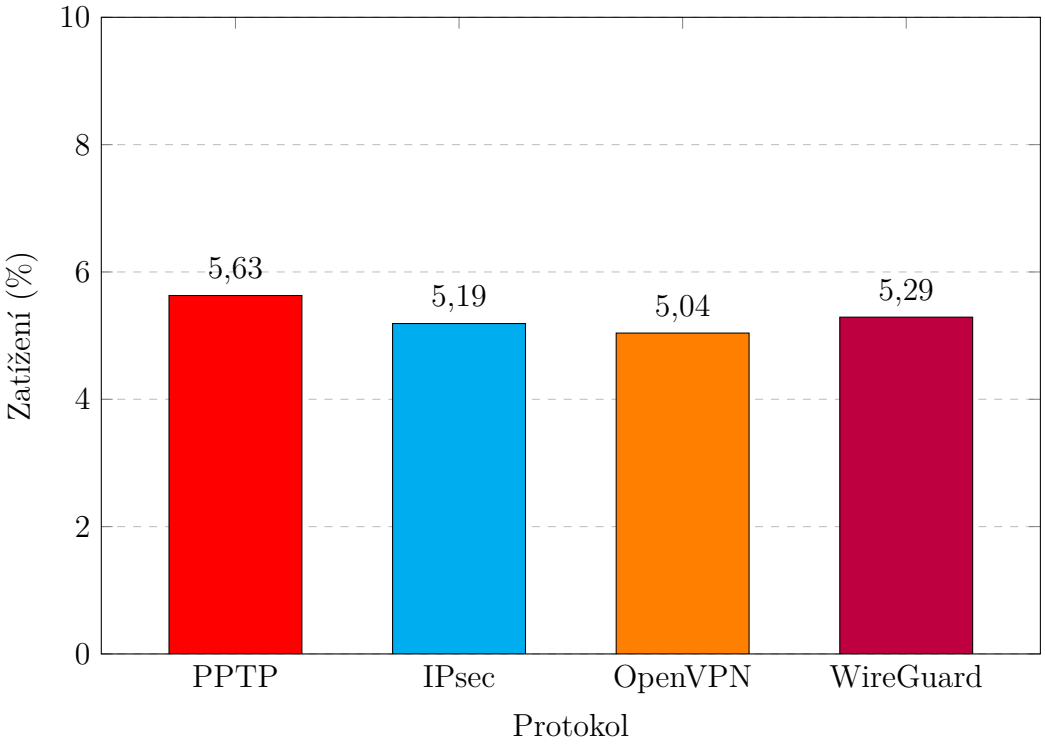
Překvapivý byl výsledek protokolu PPTP, který podle teorie v sekci 1.4.2, má být rychlý, jednoduchý a nenáročný, avšak při tomto specifickém testu bylo maximální zatížení CPU na zařízení Raspberry Pi 60,92 %. Druhý nejnáročnější protokol se stal WireGuard s 38,54 %, třetí nejnáročnější protokol byl IPsec s 32,04 % a nejmenší zatížení ukázal protokol OpenVPN s pouhými 25,05 %. WireGuard i přes výrazně jednodušší a kratší zdrojový kód oproti IPsec a OpenVPN, se projevil vyšším zatížením.

Za stejných podmínek bylo otestováno zatížení RAM na VPN serveru a maximální zatížení lze vidět na grafu 4.4. Na serveru byl použitý následující příkaz pro zaznamenání využití RAM:

Výpis 4.4: sar využití RAM na Raspberry Pi

```
# sar -r 1
```

Obr. 4.4: Maximální zatížení RAM

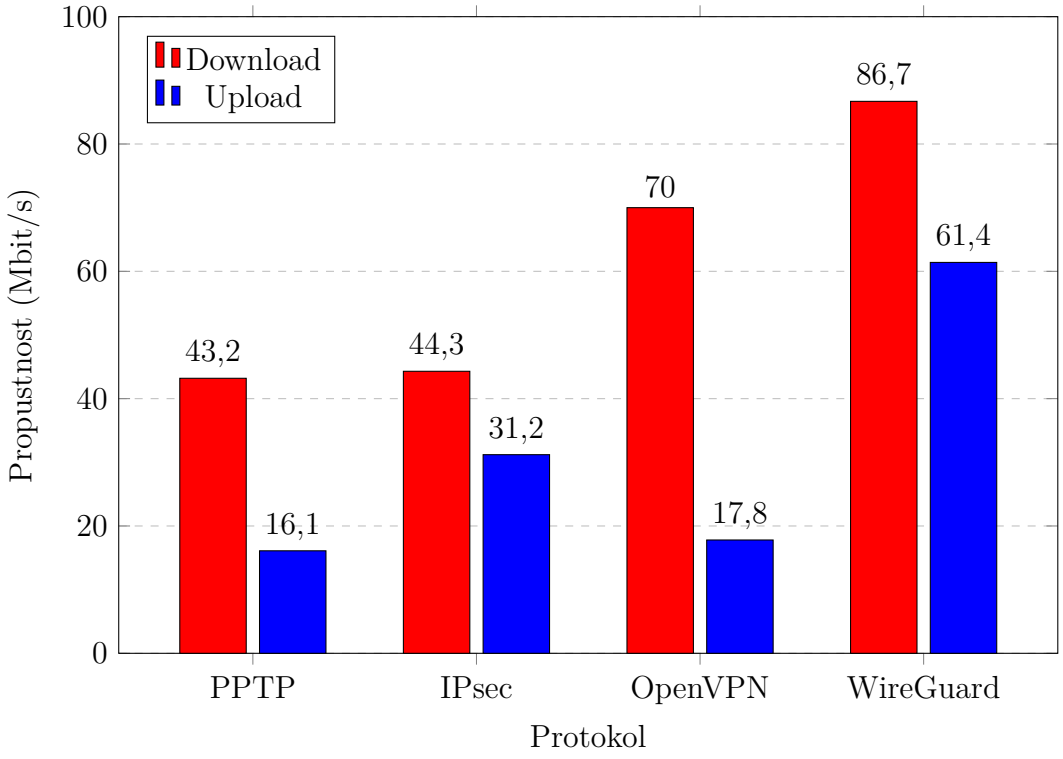


Zatíženi RAM nebylo vysoké a rozdíl nejnáročnějšího protokolu oproti nejméně náročnému je pouhých 0,59 %. Protokol PPTP měl nejhorší výsledek s 5,63 % a nejlepší výsledek měl protokol OpenVPN s 5,04 %.

4.3 Propustnost

Na grafu 4.5 lze vidět propustnost jednotlivých protokolů. Propustnost byla měřena pomocí nástroje *iPerf* a obousměrným spojením serveru a 1 klienta. Jedná se tedy o stejné příkazy nástroje *iPerf*, jako při měření hardwarové zátěže v předchozí sekci 4.2, akorát při připojení pouze 1 klienta.

Obr. 4.5: Propustnost



Nejhorší výsledek zde má protokol PPTP s nejnižší hodnotou download a upload. Protokol IPsec má podobnou rychlost download, ale má vyšší rychlost upload než protokoly PPTP a OpenVPN. Následně je velký skok při rychlosti download u protokolu OpenVPN, ale má podobně nízkou rychlost upload jako protokol PPTP. Zcela nejlepší výsledek ze všech má nejnovější protokol WireGuard, který má 2x vyšší rychlost download a 3.8x vyšší rychlost upload než protokol PPTP.

4.4 Stabilita VPN řešení

V tomto kroku se testovala stabilita VPN řešení pomocí nástroje *iPerf*, kdy byla vytvořena jednostranná UDP komunikace při různých hranicích šířky pásma. U každé hranice bylo vytvořeno spojení s 5 klienty, kde byl zaznamenán jitter, počet ztracených paketů a rychlost připojení. Průběh jitteru při hranici šířky pásma na 5 Mbit/s

je zaznačen do grafů u každého protokolu. Pro tato měření byly použity níže uvedené příkazy, kde důležitou částí je použití přepínačů *-u* a *-b* pro použití protokolu UDP a stanovení limitu šířky pásma.

Výpis 4.5: iPerf příkaz na serveru

```
# iperf -s -u -t 100 -i 1
```

Výpis 4.6: iPerf příkaz na klientech

```
# iperf -c <IP adresa serveru> -u -b <limit> -t 60 -i 1
```

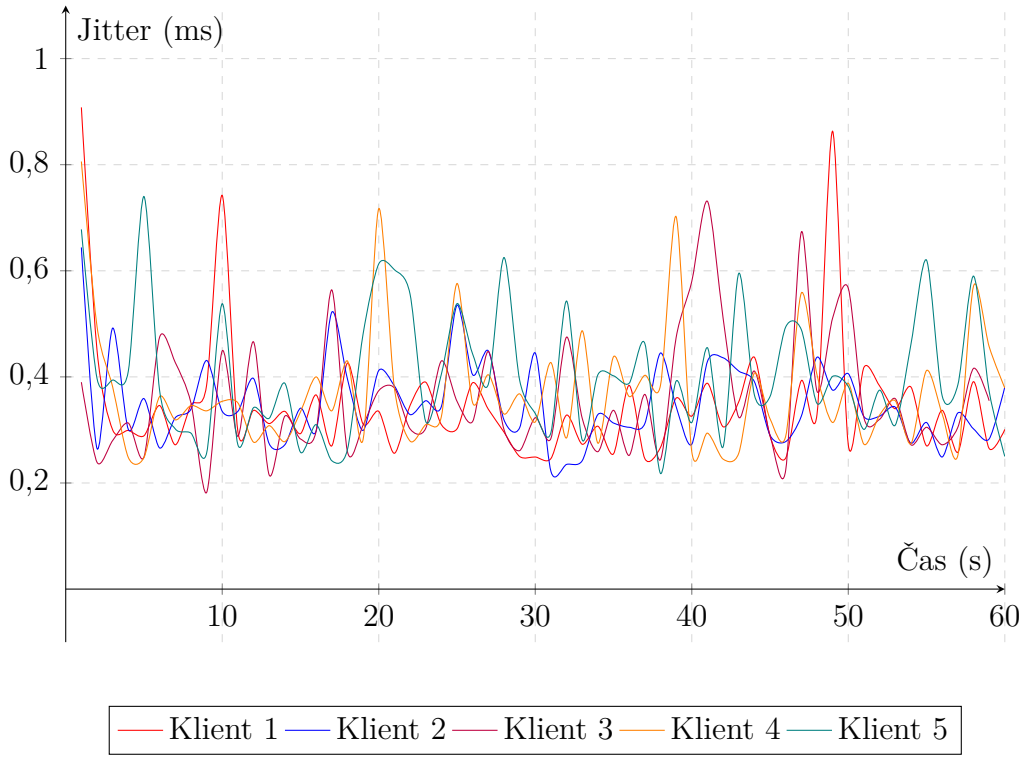
Limity byly nastavené na 5, 10, 15, 20 a 25 Mbit/s. Pro porovnání, sledování videa na platformě YouTube v dnes běžném rozlišení 1080p s 60 snímky za sekundu spotřebuje 6,8 Mbit/s a rozlišení 2160p s 30 snímky za sekundu spotřebuje 23,5 Mbit/s [31]. V tabulce 4.1 je zaznačený medián jitteru a maximální hodnota jitteru u každého testovaného protokolu, avšak pouze u hranic, u kterých nedošlo k selhání spojení neboli intervalu, ve kterém bylo ztraceno 100 % paketů. Hranice 25 Mbit/s je zcela z tabulky vynechána. Je to z důvodu, že žádný z měřených VPN protokolů nebyl schopný při této hranici mít připojení bez selhání.

Tab. 4.1: Jitter u jednotlivých protokolů

		Jitter (ms)							
		5 Mbit/s		10 Mbit/s		15 Mbit/s		20 Mbit/s	
		Maximální	Medián	Maximální	Medián	Maximální	Medián	Maximální	Medián
Protokol	PPTP	0,908	0,339	1,172	0,371	24,465	15,43	24,761	1,845
	IPsec	0,758	0,335	1,093	0,440	-	-	-	-
	OpenVPN	0,776	0,222	0,581	0,215	0,528	0,203	-	-
	WireGuard	0,572	0,134	0,320	0,204	0,278	0,142	-	-

4.4.1 PPTP

V grafu 4.6 je zaznačený průběh hodnoty jitter během UDP komunikace při nastaveném limitu na 5 Mbit/s. Ze všech testovaných protokolů má protokol PPTP nejvyšší medián a nejvyšší hodnotu maxima jitteru. Při tomto limitu také dorazilo 47 paketů, což je 0,04 % všech odeslaných paketů, mimo pořadí a 7 paketů (0,005 %) nedorazilo vůbec.

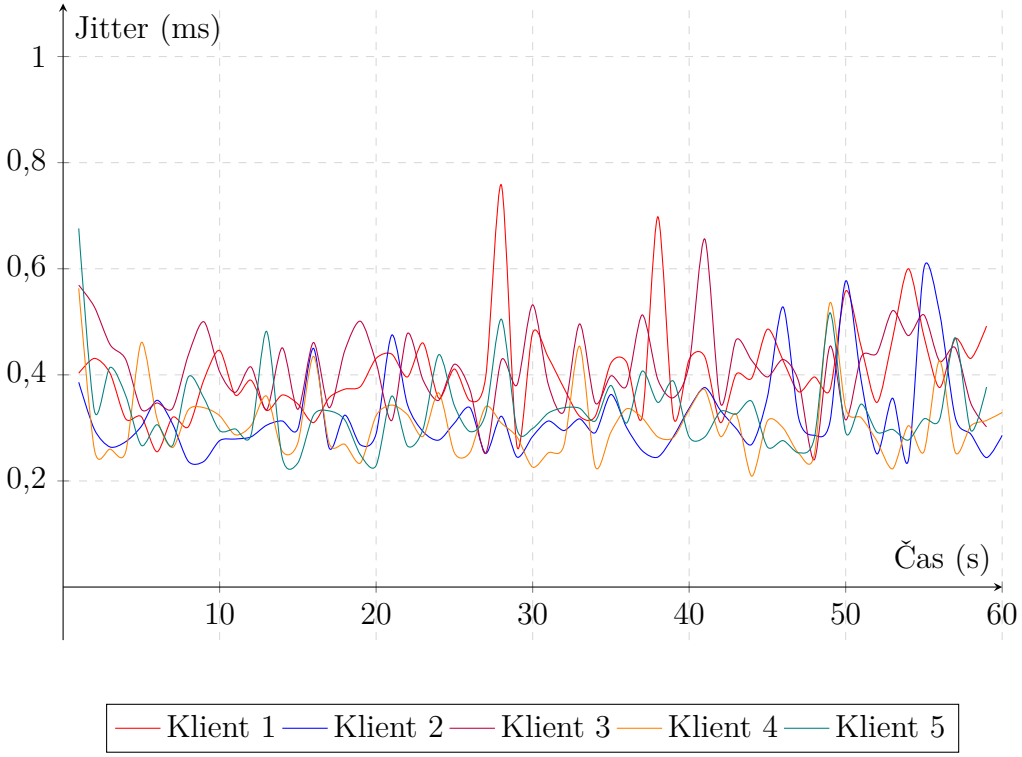


Obr. 4.6: PPTP průběh jitteru při 5 Mbit/s

Při hranici 10 Mbit/s nedorazilo ve správném pořadí 409 paketů (0,16 %) a jitter se mírně zvýšil. Na hranici 15 Mbit/s ale již začíná docházet k větším problémům, kdy rychlost přenosu všech klientů se není schopná udržet na daném limitu a v nejhorším případě rychlost klesla na 6,71 Mbit/s. Zároveň došlo ve špatném pořadí 15788 paketů (4,13 %), vůbec nedorazilo 55557 paketů (14,52 %) a jitter vyskočil extrémně vysoko, kde maximální hodnota činí 24,465 ms a medián 15,43 ms. Při nastavené hranici na 20 Mbit/s se k této rychlosti nedostal žádný klient a ve špatném pořadí došlo 10657 paketů (2,09 %) a vůbec nedošlo 112472 paketů (22,04 %). Maximální jitter byl pouze o 296 ms vyšší než při minulém limitu, ale překvapivě medián hluboce klesl na pouhých 1,845 ms. U jako jediného měřeného protokolu nedošlo při této hranici ke zkolabování připojení. Při poslední měřené hranici 25 Mbit/s již komunikace selhává a objevují se intervaly, kdy nedojdou do cíle žádné pakety.

4.4.2 IPsec

U protokolu IPsec můžeme v grafu 4.7 vidět při limitu 5 Mbit/s porovnatelné hodnoty jitter s předešlým protokolem PPTP, u kterého je mediánová hodnota o 0,004 ms a maximální jitter o 0,150 ms vyšší. Při komunikaci byly ztraceny 3 pakety, a to na úplném začátku spojení.

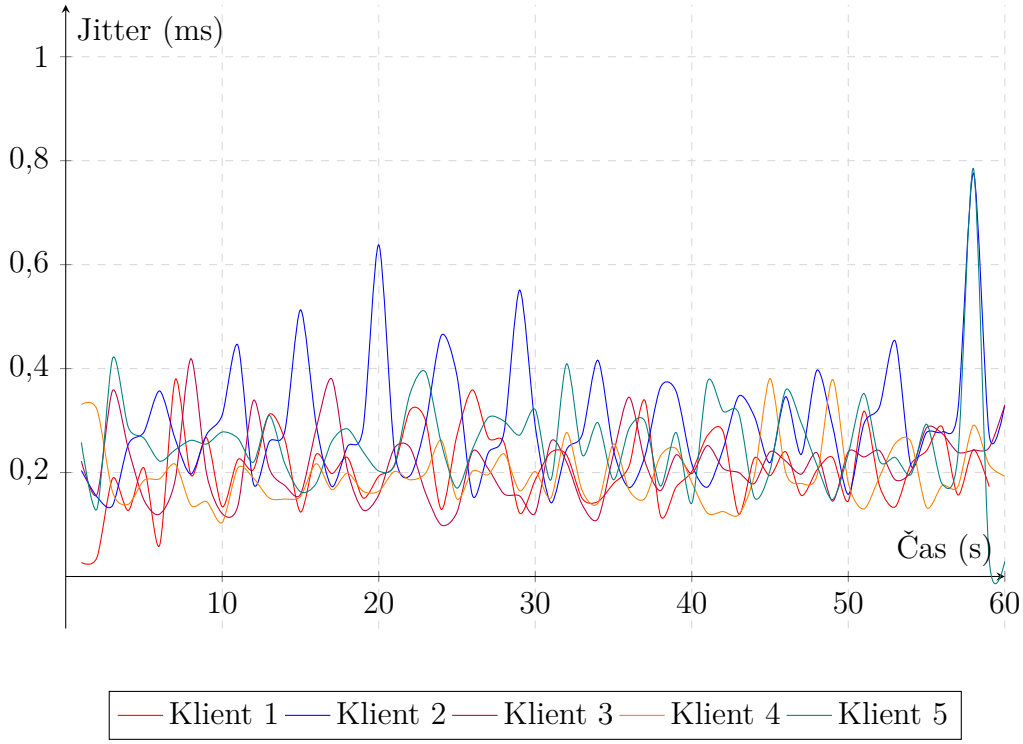


Obr. 4.7: IPsec průběh jitteru při 5 Mbit/s

Při zvýšení hranice na 10 Mbit/s se opět nedoručilo 5 paketů na začátku spojení, maximální jitter vzrostl na 1,093 ms a medián byl o 0,69 ms vyšší než u protokolu PPTP. Když se změnil limit na 15 Mbit/s, tak došlo po 1 sekundě ke zkolabování komunikace, kdy předchozí limit nenaznačoval, že by mělo dojít k této situaci. Protokolu IPsec, jako jedinému ze všech testovaných protokolů, selhalo připojení při této hranici. Vyšší hranice 20 Mbit/s a 25 Mbit/s mají stejné chování, kdy nastává zhroucení spojení a žádné pakety nedojdou do svého cíle.

4.4.3 OpenVPN

V grafu 4.8 lze vidět průběh jitteru při limitu 5 Mbit/s, který průměrně nabývá nižších hodnot než u VPN protokolu PPTP a IPsec. Maximální jitter má hodnotu 0,776 ms, která je pouze o 0,008 ms vyšší než u protokolu IPsec, a medián je nižší než u předešlých protokolů, nabývající hodnoty 0,222 ms. Komunikace při této hranici nemá žádné komplikace a došly všechny zaslané pakety.

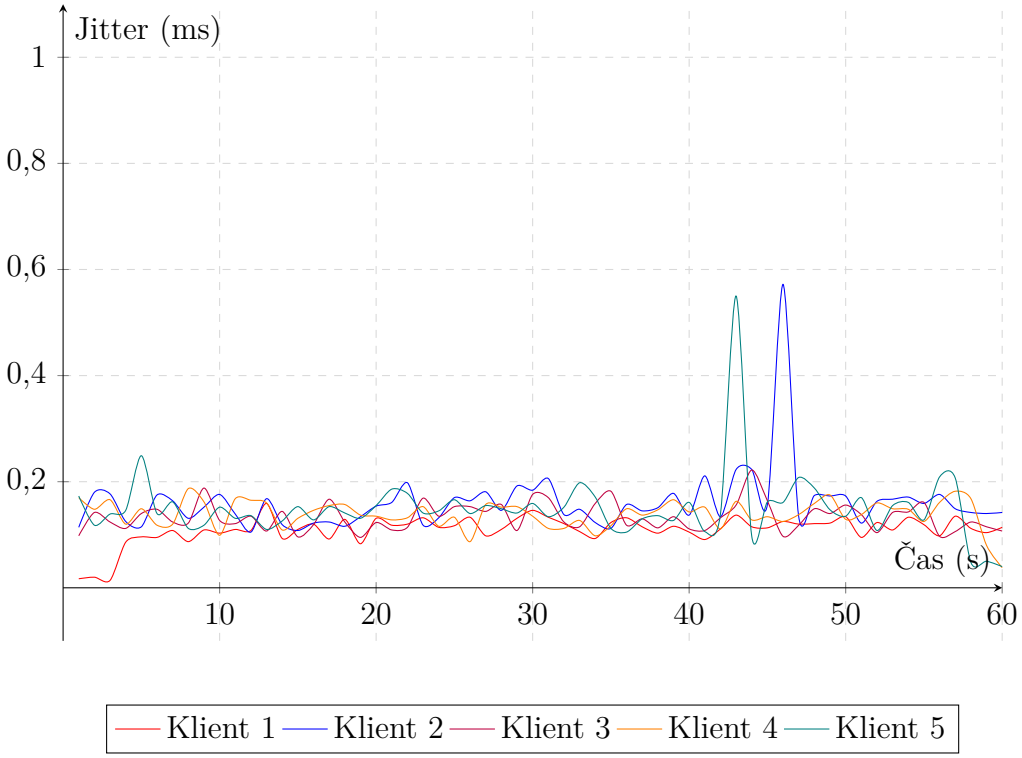


Obr. 4.8: OpenVPN průběh jitteru při 5 Mbit/s

Při zvýšení limitu na 10 Mbit/s dochází ke ztrátě 238 paketů (0,09 %), ale jitter zůstává na nízké úrovni, kde medián se snížil na 0,215 ms a maximální jitter se snížil na 0,581 ms. Když se navýší hranice na 15 Mbit/s, tak dochází k větší ztrátě paketů, kterých je 431 (0,11 %), ale hodnoty mediánu i maximálního jitteru se opět o něco snížily oproti předešlé hranici. Po zatím nejlepším výkonu na hranici 15 Mbit/s se situace u limitu 20 Mbit/s náhle změnila a objevil se interval kdy komunikace selhala a nedošlo k doručení žádných paketů. Při poslední měřené hranici na 25 Mbit/s se situace pouze zhoršila a intervaly zkolabovaného spojení se prodloužily a rozmnožily.

4.4.4 WireGuard

V grafu 4.9 lze vidět průběh jitteru při dané hranici 5 Mbit/s. Protokol WireGuard má ze všech testovaných protokolů nejnižší maximální hodnotu jitteru a medián je pouhých 0,134 ms. Také nedošlo ke žádné ztrátě paketů.



Obr. 4.9: WireGuard průběh jitteru při 5 Mbit/s

Při navýšení hranice na 10 Mbit/s nedochází pořád ke žádné ztrátě paketů, medián jitteru se zvýšil na 0,204 ms, což je porovnatelné s protokolem OpenVPN, ale maximální jitter se snížil na 0,320 ms. WireGuard jako jediný protokol nezaznamenal u hranice 15 Mbit/s jakoukoliv ztrátu paketů a má ze všech protokolů nejlepší výsledek, kde maximální jitter má hodnotu pouhých 0,278 ms a medián dosáhl hodnoty 0,142 ms. U hranice 20 Mbit/s ale došlo ke selhání spojení a stejný stav nastává i při hranici 25 Mbit/s.

4.5 Hodnocení

Testované VPN protokoly měly porovnatelné odezvy jak od VPN serveru, tak od Google DNS a velké rozdíly se neobjevily ani u zatížení RAM na Raspberry Pi. Diference se ale objevila v zatížení CPU, kde byl nečekaný výsledek u protokolu PPTP a WireGuard. U obou protokolů bylo očekávané nižší zatížení oproti protokolu IPsec, přitom protokol WireGuard měl mírně vyšší využití CPU a protokol PPTP dokonce mnohem vyšší. Při testu propustnosti měl protokol PPTP nejhorší výsledek a WireGuard naopak nejlepší. Během měření stability VPN řešení bylo zjištěno, že protokol PPTP jako jediný neselhal při daném UDP limitu na 20 Mbit/s, ale měl u všech testovaných hranic nejhorší jitter a ztrátovost paketů. Protokol IPsec měl lepší hodnoty jitteru a minimální ztrátovost paketů, avšak u jako jediného protokolu zkolabovala komunikace již při hranici 15 Mbit/s. Protokol OpenVPN měl nižší jitter než předešlé protokoly a nízkou ztrátovost paketů, ale při hranici 20 Mbit/s dochází ke selhání komunikace. Poslední protokol WireGuard měl lepší výsledky než protokol OpenVPN a nedocházelo ke žádné ztrátě paketů, až při hranici 20 Mbit/s, stejně jako u protokolu OpenVPN, dochází ke zkolabování spojení.

Nejspolehlivější VPN řešení na Raspberry Pi je tedy protokol OpenVPN a WireGuard, kde oba mají vysoké zabezpečení, dobrý výkon a relativně jednoduchou konfiguraci. Použití Raspberry Pi jako VPN serveru pro malou firmu ale není vhodné řešení, z důvodu nízké spolehlivosti spojení při připojení více klientů.

Závěr

V rámci práce se student seznámil s fungováním virtuálních privátních sítí, jejich využitím, typy, problémy, výhody a nevýhody. Seznámil se s různými tunelovacími protokoly a provedl jejich analýzu, konkrétně se jednalo o protokoly GRE, PPTP, L2TP, IPsec, OpenVPN, SSTP a WireGuard. Z teoretické části práce vyplývá, že nejstarší protokoly PPTP a L2TP je lepší nepoužívat, z důvodu nízké úrovně zabezpečení. Lepší alternativy jsou modernější a bezpečnější protokoly IPsec a OpenVPN, které nabízejí vysokou konfigurovatelnost. Protokol SSTP má výhodu obtížné detekce díky maskování jako HTTPS komunikace, ale má nevýhodu nízké podpory operačních systémů. U protokolů VPN používající protokol TCP nastává problém nazvaný *TCP meltdown* způsobující snížení výkonu tunelu. Nejmladší protokol WireGuard má vysoký potenciál, z důvodu vysoké rychlosti, bezpečnosti a jednoduchosti konfigurace, ale stále nemá stabilní verzi 1.0. Student se také seznámil s jednodeskovým počítačem Raspberry Pi a jeho všemi generacemi a modely. Popsal jeho vývoj od první generace až po poslední 4. generaci.

V praktické části student nejdříve zvolené tunelovací protokoly PPTP, IPsec, OpenVPN a WireGuard implementoval ve virtuálním prostředí. VPN server běžel na Debian 10 Buster, z důvodu podobnosti s operačním systémem Rasbian a klient používal operační systém Debian 10, nebo Windows 7. Konfigurace serveru i klienta byla detailně popsána. Byl konfigurován VPN server používající implementaci protokolu PPTP s názvem pptpd a jako klient byl použit stroj s operačním systémem Windows 7. Pro IPsec byla použita implementace strongSwan, a klient stejný jako u PPTP. Pro protokoly OpenVPN a WireGuard byl klientem systém Debian 10.

Následně popisované řešení bylo aplikováno v reálném prostředí na Raspberry Pi 3 B a jeho funkčnost byla ověřena. U zvolených protokolů byla změřena propustnost a odezva od VPN serveru a Google DNS. Prostřednictvím nástroje *sar* bylo monitorováno hardwarové zatížení při připojení 5 klientů ve stejný čas. Byla otestována stabilita protokolů pomocí nástroje *iPerf* a stanovením limitu šířky pásma na 5 různých hranicích. U každé hranice byl zaznamenán počet nedoručených paketů a jitter. Jedinému protokolu PPTP se nezhroutilo spojení při limitu 20 Mbit/s, ale při ostatních hranicích měl nejhorší ztrátovost a jitter ze všech protokolů. WireGuard měl nejlepší výsledky a nulovou ztrátovost paketů, dokud se nezvýšila hranice na 20 Mbit/s, kdy spojení selhalo.

Student došel k závěru, že mini počítač Raspberry Pi není vhodný pro použití jako VPN server, kde bude připojeno větší počet klientů zároveň. Je to z důvodu snížené spolehlivosti připojení, kdy Raspberry Pi měl problém obsloužit 5 klientů ve stejný čas.

Literatura

[1] DAVIES, Jon. *Chapter 14 - Virtual Private Networking*. In: Microsoft Docs [online]. 27. 12. 2005 [cit. 7. 12. 2019]. Dostupné z: <https://docs.microsoft.com/en-us/previous-versions//bb727019(v=technet.10)>

[2] *Virtual Private Networking: An Overview*. In: Microsoft Docs [online]. 12. 9. 2009 [cit. 7. 12. 2019]. Dostupné z: <https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/bb742566(v=technet.10)>

[3] WILLIAMS, Martyn. *China tightens Internet control by blocking VPN services*. In: Networks Asia [online]. 23. 1. 2016 [cit. 7. 12. 2019]. Dostupné z: <https://search.proquest.com/docview/1648173791/fulltext/25627FC549B441F2PQ/>

[4] TITZ, Olaf. *Why TCP Over TCP Is A Bad Idea*. In: INKA [online]. 23. 4. 2001 [cit. 7. 12. 2019]. Dostupné z: <http://sites.inka.de/~W1011/devel/tcp-tcp.html>

[5] FARINACCI, Dino, Tony LI, Stan HANKS, David MEYER a Paul TRAINA. *RFC 2784: Generic Routing Encapsulation (GRE)*. In: Internet Engineering Task Force [online]. Březen 2000 [cit. 7. 12. 2019]. Dostupné z: <https://tools.ietf.org/html/rfc2784>

[6] HAMZEH, Kory, William VERTHEIN, Jeff TAARUD, W. Andrew LITTLE a Glen TOWNSLEY.*RFC 2637: Point-to-Point Tunneling Protocol (PPTP)*. In: Internet Engineering Task Force [online]. Červenec 1999 [cit. 7. 12. 2019]. Dostupné z: <https://tools.ietf.org/html/rfc2637>

[7] *Xl2tpd*. In: Github [online]. 13 .10. 2019 [cit. 17. 12. 2019]. Dostupné z: <https://github.com/xelerance/xl2tpd/releases/tag/v1.3.15>

[8] PALTER, Bill, Allan RUBENS, Andrew J. VALENCIA, Glen ZORN a W. Mark TOWNSLEY. *RFC 2661: Layer Two Tunneling Protocol "L2TP"*. In: Internet Engineering Task Force [online]. Říjen 1999 [cit. 7. 12. 2019]. Dostupné z: <https://tools.ietf.org/html/rfc2661>

[9] KENT, Stephen a ATKINSON, Randall. *RFC 2401: Security Architecture for the Internet Protocol*. In: Internet Engineering Task Force [online]. Listopad 1998 [cit. 7. 12. 2019]. Dostupné z: <https://tools.ietf.org/html/rfc2401>

[10] *Community Downloads*. In: OpenVPN [online]. [cit. 7. 12. 2019]. Dostupné z: <https://openvpn.net/community-downloads/>

[11] SCHUSTER, Bea. *What Is OpenVPN & Is It Safe Enough To Use In 2019?*. In: VpnMentor [online]. 19. 11. 2019 [cit. 17. 12. 2019]. Dostupné z: <https://www.vpnmentor.com/blog/what-is-openvpn-is-it-safe-enough-to-use/>

[12] *OpenVPN Compatibility*. In: OpenVPN [online]. [cit. 7. 12. 2019]. Dostupné z: <https://openvpn.net/faq/openvpn-compatibility/>

[13] *OpenVPN*. In: OpenVPN [online]. [cit. 7. 12. 2019]. Dostupné z: <https://openvpn.net/>

[14] *[MS-SSTP]: Secure Socket Tunneling Protocol (SSTP): 18.0*. In: Microsoft [online]. Září 2018 [cit. 7. 12. 2019]. Dostupné z: <https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-sstp/c50ed240-56f3-4309-8e0c-1644898f0ea8>

[15] *Sstp-client*. In: SourceForge [online]. 17. 11. 2019 [cit. 17. 12. 2019]. Dostupné z: <https://sourceforge.net/projects/sstp-client/>

[16] *WireGuard*. In: WireGuard [online]. [cit. 7. 12. 2019]. Dostupné z: <https://www.wireguard.com/>

[17] SALTER, Jim. *WireGuard VPN review: A new type of VPN offers serious advantages*. In: Ars Technica [online]. 26. 8. 2018 [cit. 7. 12. 2019]. Dostupné z: <https://bit.ly/2S0xAIS>

[18] TORVALDS, Linus. *Re: [GIT] Networking*. In: Linux Kernel Mailing List [online]. 1. 8. 2018 [cit. 7. 12. 2019]. Dostupné z: <http://lkml.iu.edu/hypermail/linux/kernel/1808.0/02472.html>

[19] *Raspberry Pi Documentation: FAQs*. In: Raspberry Pi - Teach, Learn, and Make with Raspberry Pi [online]. [cit. 16. 12. 2019]. Dostupné z: <https://www.raspberrypi.org/documentation/faqs/>

[20] *Raspberry Pi*. In: Raspberry Pi - Teach, Learn, and Make with Raspberry Pi [online]. [cit. 7. 12. 2019]. Dostupné z: <https://www.raspberrypi.org/>

[21] *Raspberry Pi hardware*. In: Raspberry Pi - Teach, Learn, and Make with Raspberry Pi [online]. [cit. 7. 12. 2019]. Dostupné z: <https://www.raspberrypi.org/documentation/hardware/raspberrypi/>

[22] *8GB Raspberry Pi 4 on sale now at $75*. In: Raspberry Pi - Teach, Learn, and Make with Raspberry Pi [online]. [cit. 28. 5. 2020]. Dostupné z: <https://www.raspberrypi.org/blog/8gb-raspberry-pi-4-on-sale-now-at-75/>

[23] *Debian images*. In: OSBoxes [online]. 23. 10. 2013 [cit. 7. 12. 2019]. Dostupné z: <https://www.osboxes.org/debian/>

[24] *Debian*. In: Debian [online]. 2013-10-23 [cit. 7. 12. 2019]. Dostupné z: <https://www.debian.org/>

[25] *Poptop*. In: Poptop - Open Source Server [online]. 2013-10-23 [cit. 7. 12. 2019]. Dostupné z: <http://poptop.sourceforge.net/>

[26] *StrongSwan*. In: StrongSwan - IPsec VPN [online]. 2. 9. 2019 [cit. 7. 12. 2019]. Dostupné z: <https://www.strongswan.org/>

[27] *StrongSwan: Download*. In: StrongSwan - IPsec VPN [online]. 2. 9. 2019 [cit. 7. 12. 2019]. Dostupné z: <https://www.strongswan.org/download.html>

[28] *OpenVPN: How To Guide*. In: OpenVPN [online]. [cit. 7. 12. 2019]. Dostupné z: <https://openvpn.net/community-resources/how-to/>

[29] *IPerf*. In: IPerf: The TCP, UDP and SCTP network bandwidth measurement tool [online]. [cit. 18. 5. 2020]. Dostupné z: <https://iperf.fr>

[30] *SYSSTAT*. In: SYSSTAT [online]. [cit. 18. 5. 2020]. Dostupné z: <http://sebastien.godard.pagesperso-orange.fr>

[31] STEGNER, Ben. *How Much Data Does YouTube Use?*. In: MakeUseOf: Technology, Simplified [online]. 19. 2. 2020 [cit. 18. 5. 2020]. Dostupné z: <https://www.makeuseof.com/tag/how-much-data-does-youtube-use/>

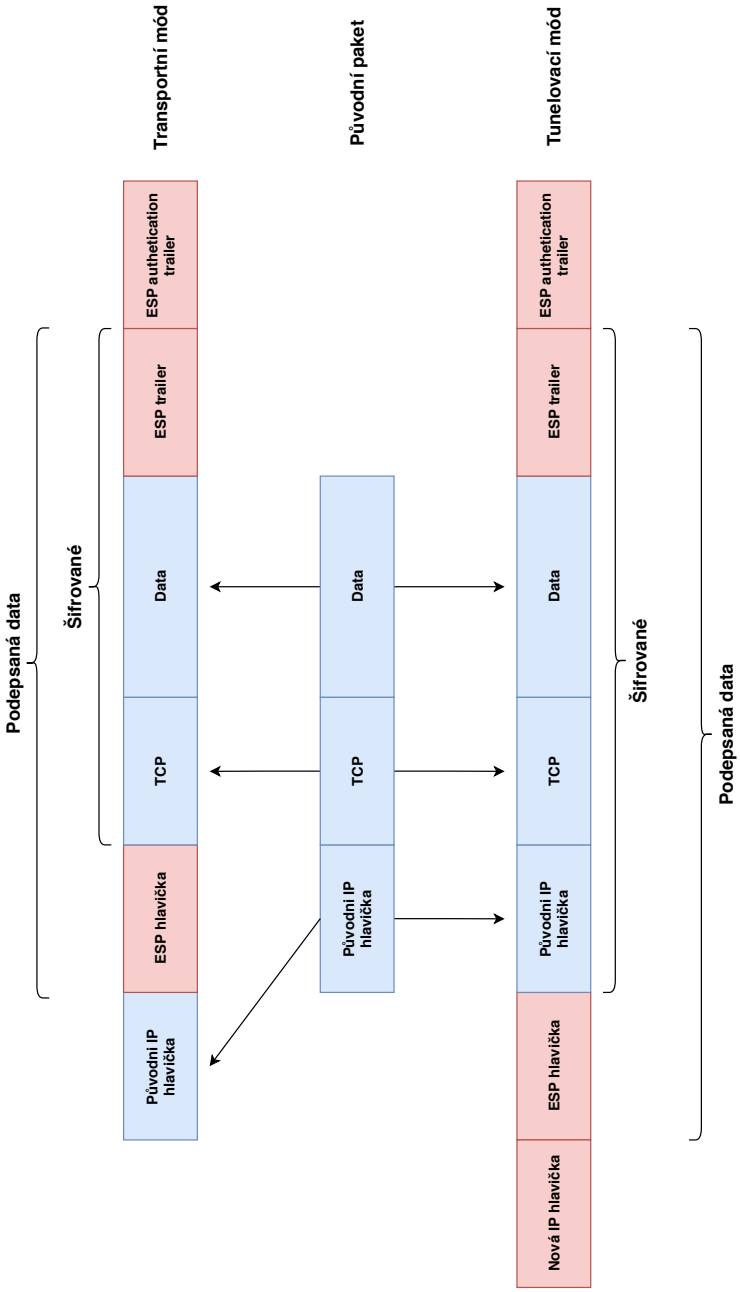
Seznam symbolů, veličin a zkratek

AH	Authentication Header
BLE	Bluetooth Low Energy
CA	Certificate Authority
CHAP	Challenge-Handshake Authentication Protocol
CPU	Central Processing Unit
DN	Distinguished Name
DNS	Domain Name System
DSI	Display Serial Interface
EAP	Extensible Authentication Protocol
ESP	Encapsulating Security Payload
GNU GPL	GNU General Public License
GRE	Generic Routing Encapsulation
HDMI	High-Definition Multimedia Interface
HMAC	Hash-based Message Authentication Code
HTTP	Hypertext Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IANA	Internet Assigned Numbers Authority
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
IP	Internet Protocol
IPSEC	Internet Protocol Security
ISAKMP	Internet Security Association and Key Management Protocol
ISP	Internet Service Provider
L2F	Layer 2 Forwarding
L2TP	Layer 2 Tunneling Protocol
LAC	L2TP Access Concentrator
LNS	L2TP Network Server
MAC	Message Authentication Code
MITM	Man-In-The-Middle
MMC	MultiMediaCard
MPPE	Microsoft Point-to-Point Encryption
NAT	Network Address Translation
NAT-T	Network Address Translation Traversal
PAP	Password Authentication Protocol rozvinutí zkratky
PKI	Public Key Infrastructure
PKCS	Public Key Cryptography Standard
PPP	Point-to-Point Protocol

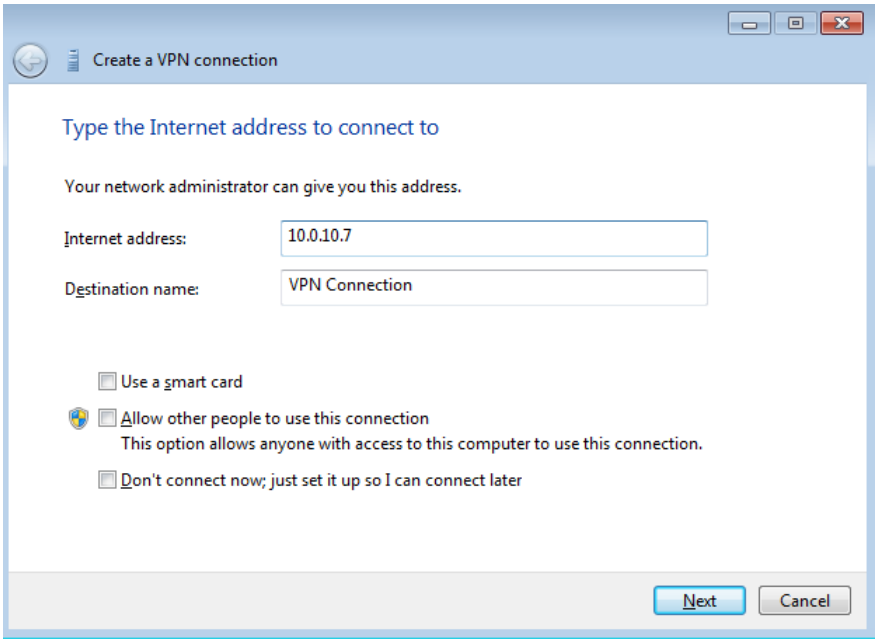
PPTP	Point-to-Point Tunneling Protocol
RAM	Random Access Memory
RFC	Request for Comments
SA	Security Associations
SAD	Security Association Database
SAN	Subject Alternate Name
SAR	System Activity Report
SD	Secure Digital
SDHC	Secure Digital High Capacity
SDIO	Secure Digital Input Output
SFTP	Secure File Transfer Protocol
SP	Security Policy
SPD	Security Policy Database
SPI	Security Parameter Index
SSH	Secure Shell
SSL	Secure Sockets Layer
SSTP	Secure Socket Tunneling Protocol
SYSTAT	System Statistics
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
UFW	Uncomplicated Firewall
USB	Universal Serial Bus
VPN	Virtual Private Network

Seznam příloh

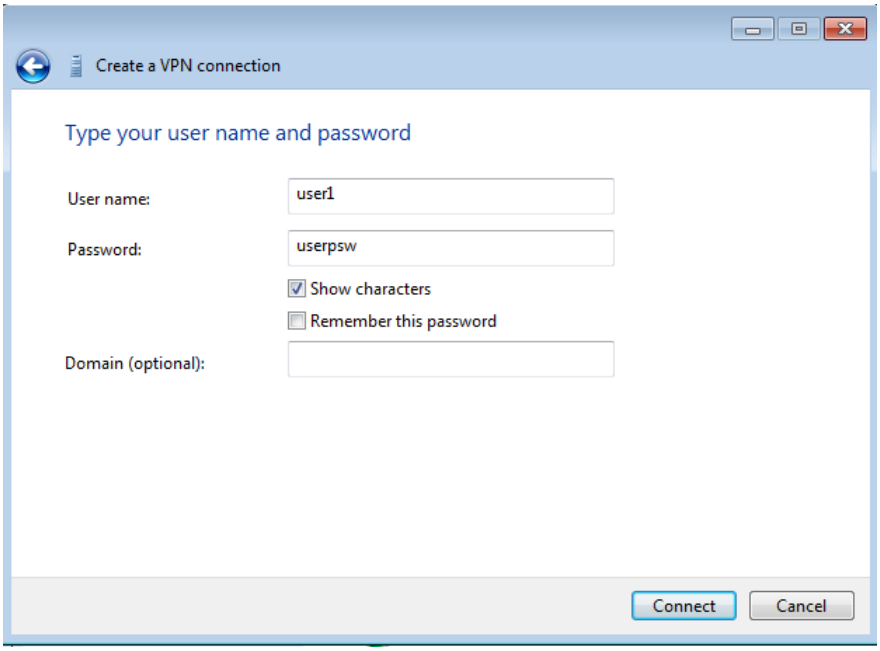
A Obrázky



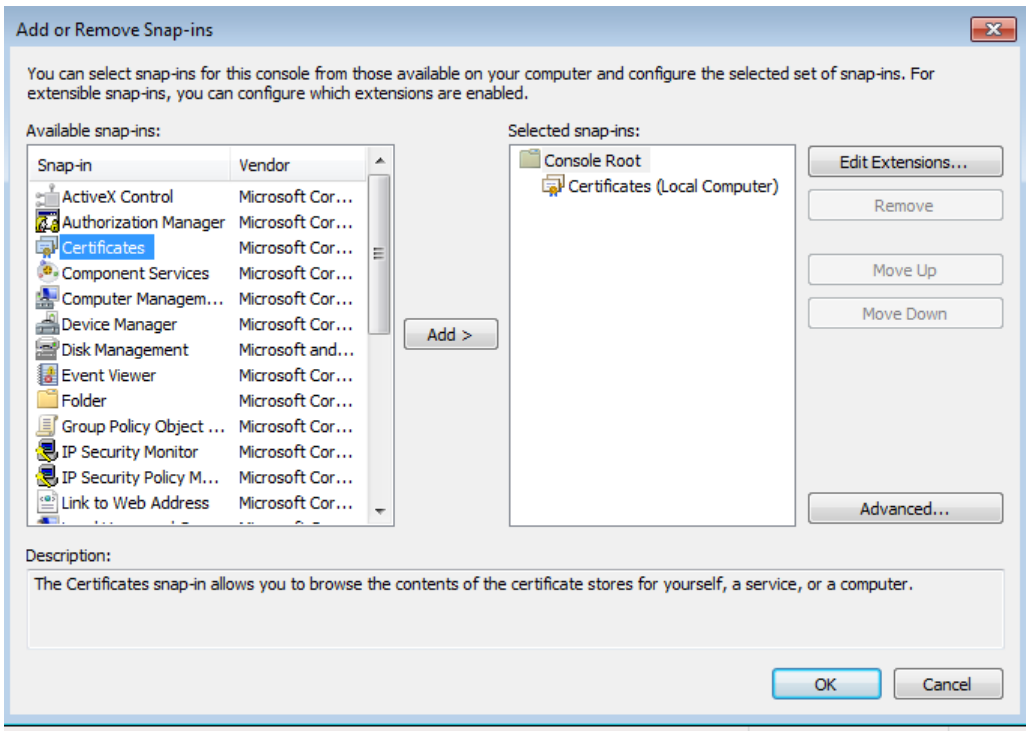
Obr. A.1: ESP v transportním a tunelovacím módu



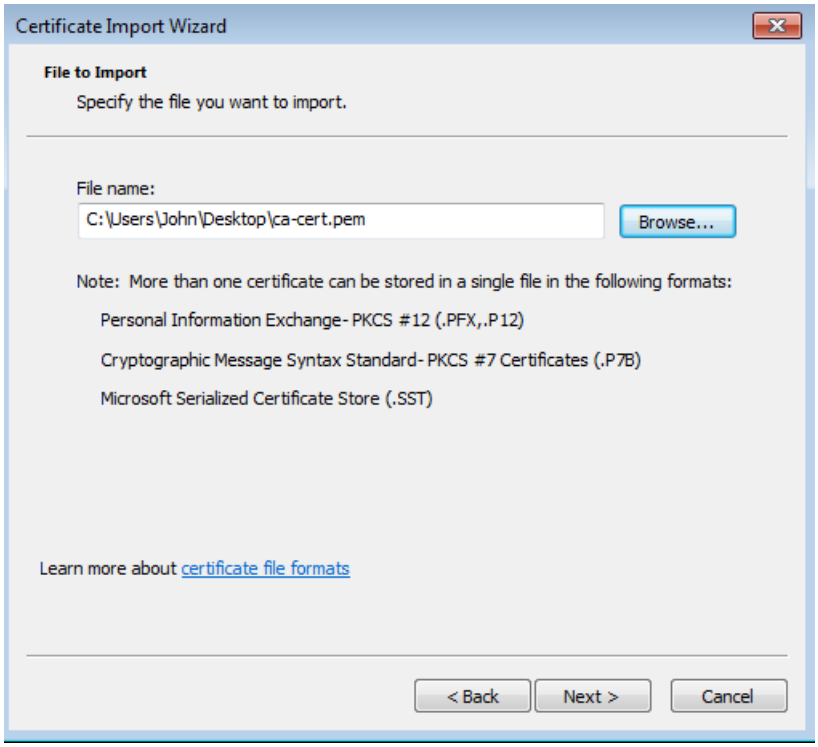
Obr. A.2: Zadání IP adresy VPN serveru



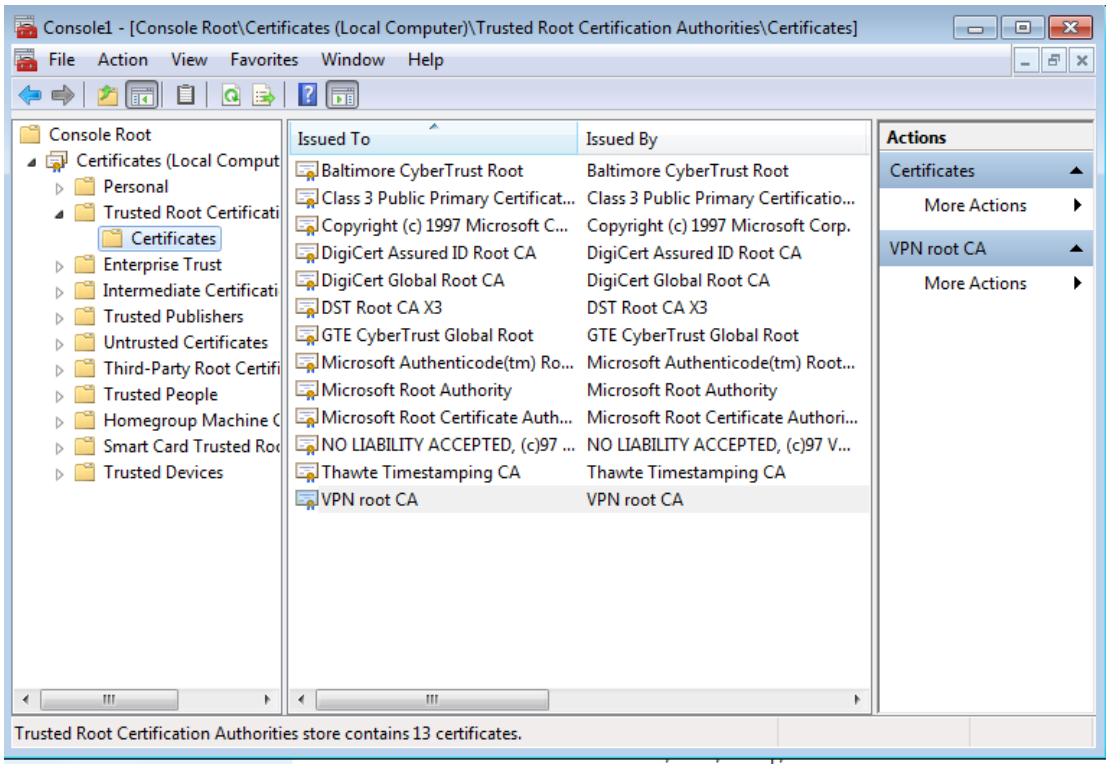
Obr. A.3: Přihlašovací údaje



Obr. A.4: Vložení certifikátu



Obr. A.5: Zvolení certifikátu



Obr. A.6: Importovaný certifikát

B Výpisy

Výpis B.1: ipsec.conf

```
config setup
    charondebug="ike 1, knl 1, cfg 0"
    uniqueids=no
conn ikev2-vpn
    auto=add
    compress=no
    type=tunnel
    keyexchange=ikev2
    fragmentation=yes
    forceencaps=yes
    dpdaction=clear
    dpddelay=300s
    rekey=no
    left=%any
    leftid=10.0.10.11
    leftcert=server-cert.pem
    leftsendcert=always
    leftsubnet=0.0.0.0/0
    right=%any
    rightid=%any
    rightauth=eap-mschapv2
    rightsourceip=172.16.0.0/12
    rightdns=8.8.8.8,8.8.4.4
    rightsendcert=never
    eap_identity=%identity
    ike=aes256-sha1-modp1024,aes128-sha1-modp1024,\
        3des-sha1-modp1024!
    esp=aes256-sha256,aes256-sha1,3des-sha1!
```

Výpis B.2: server.conf

```
port 1194
proto udp
dev tun
ca ca.crt
cert server.crt
key server.key
dh dh.pem
server 10.8.0.0 255.255.255.0
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 208.67.222.222"
push "dhcp-option DNS 208.67.220.220"
keepalive 10 120
tls-auth ta.key 0
cipher AES-256-CBC
user nobody
group nogroup
persist-key
persist-tun
status /var/log/openvpn/openvpn-status.log
log         /var/log/openvpn/openvpn.log
log-append  /var/log/openvpn/openvpn.log
verb 3
explicit-exit-notify 1
```