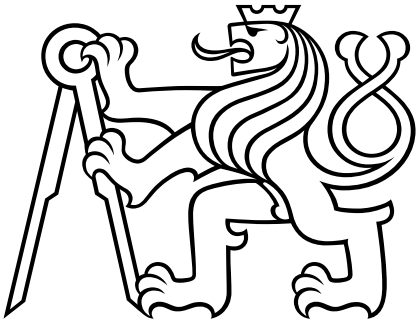
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA ELEKTROTECHNICKÁ

KATEDRA TELEKOMUNIKAČNÍ TECHNIKY



Protokoly pro virtuální sítě

Diplomová práce

květen 2020	**Diplomant:**	**Dmitriy Doroshenko**
	Vedoucí práce:	**Ing. Tomáš Vaněk, Ph.D.**
	Magisterský studijní program:	**Elektronika a komunikace**
	Studijní obor:	**Komunikační sítě a Internet**

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Doroshenko** Jméno: **Dmitriy** Osobní číslo: **484276**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra telekomunikační techniky**

Studijní program: **Elektronika a komunikace**

Specializace: **Komunikační sítě a internet**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Protokoly pro virtuální sítě

Název diplomové práce anglicky:

New Protocols for Virtual Private Networks

Pokyny pro vypracování:

Seznamte se protokoly a technologiemi pro virtuální privátní sítě. Porovnejte klasická řešení (IPsec, OpenVPN,SSL VPN) s nově nastupujícími technologiemi (SSTP, Wireguard, SoftEther, TINC, FreeLAN ,...). Proveďte praktická měření vybraných technologií s cílem identifikovat vhodného nástupce tradičních technologií v běžných VPN scénářích (lan-to-lan VPN, remote access VPN). Zhodnoťte také obtížnost nasazení vícefaktorové autentizace u dostupných implementací jednotlivých technologií. Na základě provedených experimentů navrhněte laboratorní úlohu vhodně demonstrují přínosy vybrané technologie pro realizaci virtuální privátní sítě.

Seznam doporučené literatury:

[1] IKEv2 - https://tools.ietf.org/html/rfc7296 [on-line]
[2] Wireguard - https://www.wireguard.com/ [on-line]
[3]SSTP - https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-sstp/70adc1dfc4fe-4b02-8872-f1d8b9ad806a [on-line]
[4] SoftEther - https://www.softether.org/4-docs/1-manual [on-line]

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Tomáš Vaněk, Ph.D., katedra telekomunikační techniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **08.01.2020** Termín odevzdání diplomové práce: _____

Platnost zadání diplomové práce: **30.09.2021**

Ing. Tomáš Vaněk, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

. _____ Datum převzetí zadání

_____ Podpis studenta

Poděkování

V první řadě bych chtěl poděkovat vedoucímu mé diplomové práce Ing. Tomáši Vaňkovi, Ph.D. za jeho vysokou profesionalitu a trpělivost.

Rád bych také poděkoval své rodině a přátelům za jejich víru a podporu.

Dmitriy Doroshenko

Prohlášení

Prohlašuji, že jsem danou diplomovou práci vypracoval samostatně za odborného vedení vedoucího diplomové práce a s použitím uvedených zdrojů v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 22. května 2020

Dmitriy Doroshenko

Abstrakt

Diplomová práce je věnována protokolům virtuálních privátních sítí. V teoretické části jsou probrány podstatné principy technologie VPN a její nejčastěji používané protokoly. Praktická část obsahuje měření základních parametrů přenosu dat přes VPN spojení s cílem porovnat dosavadní VPN protokoly s vybranými novými protokoly a návrh laboratorní úlohy, jejímž cílem je praktické seznámení s konfigurací VPN serveru a klienta a s realizací vícefaktorové autentizace pro zabezpečený přístup.

Klíčová slova: VPN, virtuální privátní sítě, 2FA, vícefaktorová autentizace, IPSec, OpenVPN, Wireguard

Abstract

The diploma thesis is devoted to the protocols used in Virtual Private Networks. The theoretical part discusses the essential principles of VPN technology and its most commonly used protocols. The practical part contains the measurement of basic parameters of data transfer via a VPN connection in order to compare existing VPN protocols with selected new protocols and the design of a laboratory task, which aim is a practical acquaintance with the configuration of the VPN server and client and with the implementation of multifactor authentication for secure access.

Keywords: VPN, Virtual Private Network, 2FA, two-factor authentication, IPSec, OpenVPN, Wireguard

Obsah

1. Úvod

Současný vývoj technologií přenosu dat, rozšiřování pokrytí přenosových sítí, dostupnost a rostoucí výkonnost uživatelských zařízení zvyšuje mobilitu uživatelů a zároveň vyvolává celou řadu úkolů. Především zajištění bezpečné výměny informací přes veřejné sítě.

Jedním z možných řešení tohoto problému je použití technologie virtuálních privátních sítí. Tato technologie pomáhá zajistit komunikace mezi geograficky distribuovanými jednotkami státních a komerčních struktur s minimálními náklady. Může být také používána uživateli k zajištění bezpečného přístupu k soukromým úložištím dat prostřednictvím veřejně dostupných komunikačních kanálů. Virtuální privátní síť je vybudována na základě logického spojení mezi koncovými body prostřednictvím veřejné sítě s přepojováním paketů (například Internet). Toto spojení je na logické úrovni izolované od ostatních uživatelů této sítě.

Diplomová práce se věnuje problematice technologie virtuálních privátních sítí (VPN) a protokolů, které jsou v této technologii v současné době aktivně používané nebo je v blízké budoucnosti nahradí.

Praktickým cílem práce je provedení měření vybraných protokolů a jejich srovnání. Pro tyto účely je opodstatněně použit neplacený software s otevřeným zdrojovým kódem. Na základě výsledků měření následuje vytvoření laboratorní úlohy. Záměrem této úlohy je seznámit studenty se základními principy konfigurace a nastavením parametrů VPN spojení a dalšími opatřeními na ochranu přístupů, jako je dvoufaktorová autentizace.

Práci můžeme rozdělit na teoretickou a praktickou část.

V teoretické části jsou shrnuty základní body uvedené technologie.

Kapitola 2 obsahuje obecný přehled koncepce VPN a její klíčové pojmy.

Kapitola 3 popisuje základní principy nejčastěji používaných protokolů VPN a jejich fungování.

Kapitola 4 představuje základní metody vícefaktorové autentizace.

Praktická část se zabývá odhadem parametrů přenosu dat a návrhem laboratorní úlohy.

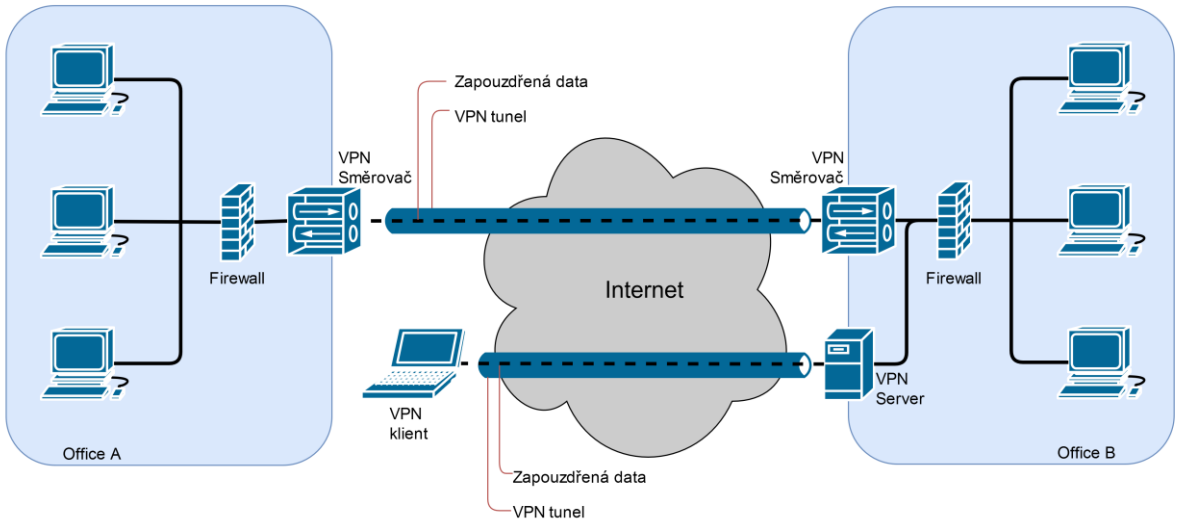
Kapitola 5 má v obsahu metodiku a výsledky měření doby přenosu a rychlosti přenosu datových bloků různých velikostí, jakož i analýzu výsledků měření.

Kapitola 6 je připraveným návodem laboratorní úlohy, jejímž úkolem je praktické seznámení studentů s jedním z moderních VPN protokolů a dvoufaktorovou autentizací.

2. Technologie virtuálních privátních sítí

Technologie virtuálních privátních sítí (VPN, anglicky Virtual Private Network) - obecný název pro technologie, které umožňují jedno nebo více síťových propojení (logická síť) přes jinou síť, nejčastěji přes Internet. Ačkoliv je komunikace prováděna v sítích s nižší nebo neznámou důvěryhodností, například ve veřejných sítích, díky použití kryptografické ochrany míra důvěryhodnosti vytvořené logické sítě nezáleží na úrovni důvěryhodnosti v základních sítích.

V závislosti na použitých protokolech a cílech může VPN poskytovat tři typy připojení: bod-bod, bod-síť a síť-síť.



Obr. 2.1. Virtuální privátní síť

2.1. Účely

Technologie VPN dává možnost uživatelům se bezpečným způsobem připojit ke vzdálenému serveru pomocí infrastruktury poskytované veřejnou sítí. Pro ten případ VPN připojení z pohledu uživatele je spojením typu bod-bod mezi počítačem uživatele a VPN serverem. Technologie VPN také umožňuje zabezpečené spojení ústředí se svými pobočkami nebo s jinými společnostmi prostřednictvím veřejné sítě. Zde VPN spojení přes Internet jedná jako spojení v rozlehlé síti mezi jejími uzly [1].

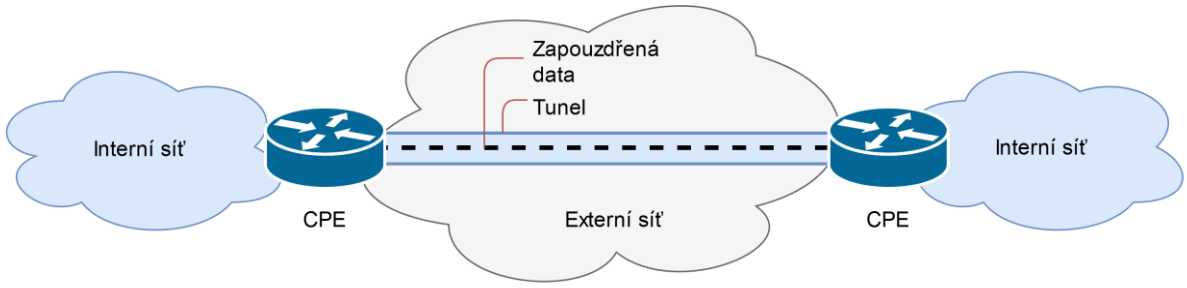
V obou případech bude uživatelům v rámci veřejné sítě poskytováno zabezpečené spojení jako ve vlastní lokální síti, podstata mezilehlé sítě je pro uživatele irelevantní, protože se výměna dat koná prostřednictvím vyhrazeného privátního spojení.

Technologie VPN je navržena tak, aby řešila současné záležitosti související s trendem, který předpokládá široce distribuovanou činnost, kde by se pracovníci měli možnost připojit ke korporativní síti a bezpečně v ní spolu komunikovat.

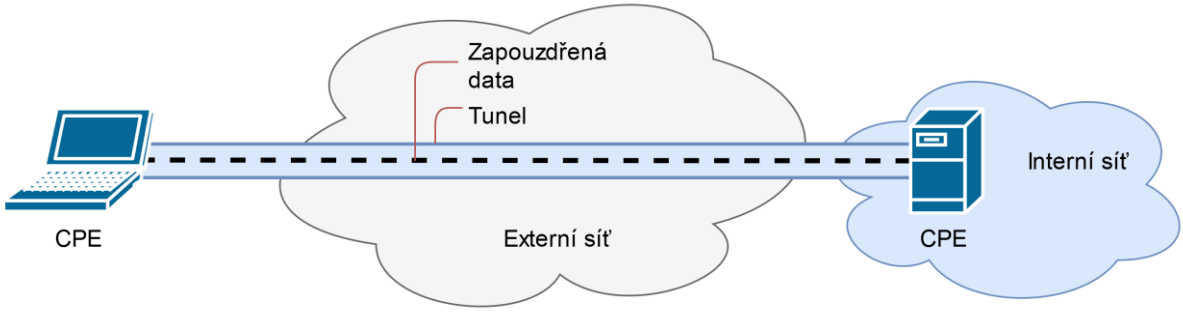
2.2. Struktura virtuálních privátních sítí

Virtuální privátní síť můžeme rozdělit do několika základních částí [2] (Obr. 2.2):

- Klientská zařízení (CPE, Customer Premises Equipment). CPE mohou být směrovače nebo hosty.
- Tunel – logické spojení dvou klientských zařízení, které vzniká zapouzdřením přenášeného protokolu do nosného protokolu.
- Interní síť, jichž může být více, než jedna.
- Externí (veřejná) síť, jejímž prostřednictvím prochází zapouzdřená data.



a)

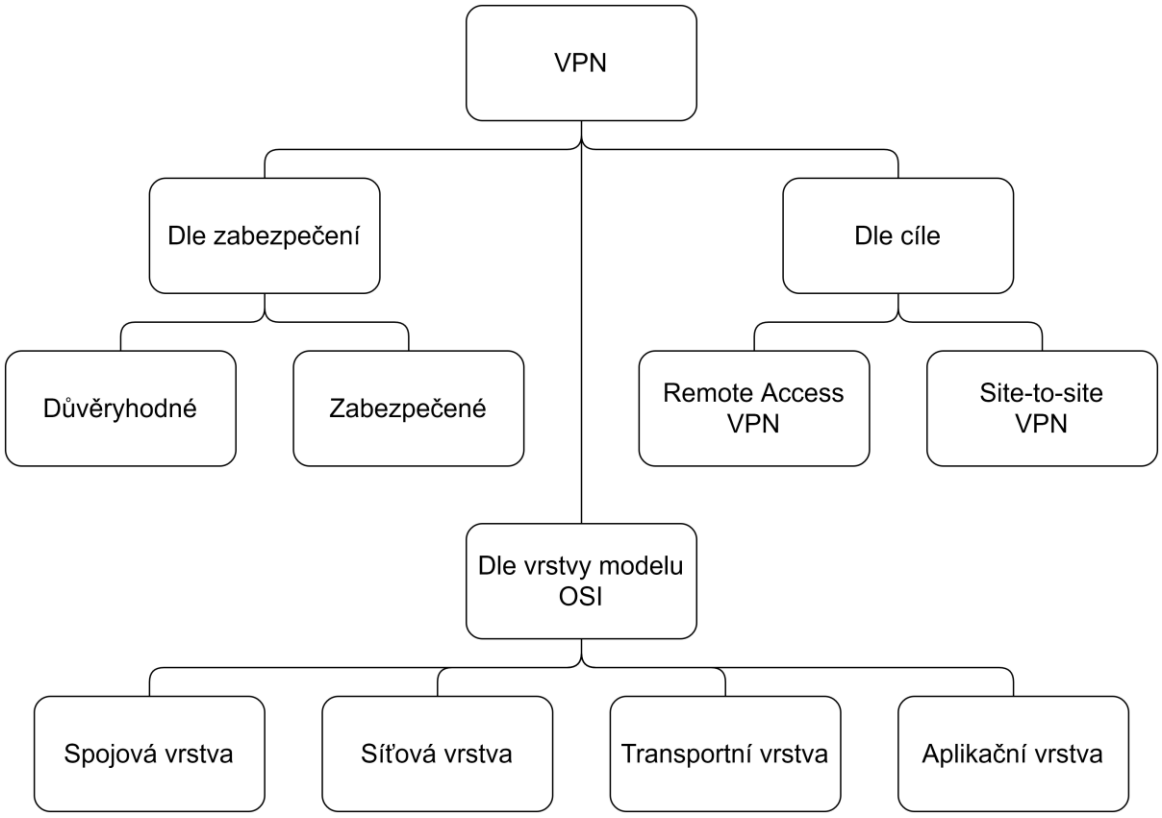


b)

Obr. 2.2. Zobecněná struktura VPN sítě: a) CPE jsou směrovače; b) CPE jsou vzdálený počítač a VPN server

2.3. Klasifikace VPN

Virtuální privátní sítě lze klasifikovat podle několika základních parametrů (Obr. 2.3):



Obr. 2.3. Klasifikace virtuálních privátních sítí

Dle použitého prostředí:

- **Důvěryhodné.** V těchto případech jsou média považována za důvěryhodná a jde jen o vytvoření virtuální sítě v rámci větší sítě. Příkladem jsou MPLS (Multi-protocol label switching) sítě nebo použití L2TP (Layer 2 Tunneling Protocol). Měli bychom si však uvědomit, že v případě L2TP se zabezpečení přesouvá na jiný protokol (IPsec).
- **Zabezpečené.** Nejběžnější varianta virtuálních privátních sítí, s jejíž pomocí je možné vytvořit spolehlivou a bezpečnou síť založenou na nedůvěryhodné veřejné síti. Zabezpečení se realizuje pomocí zabezpečovacích protokolů.

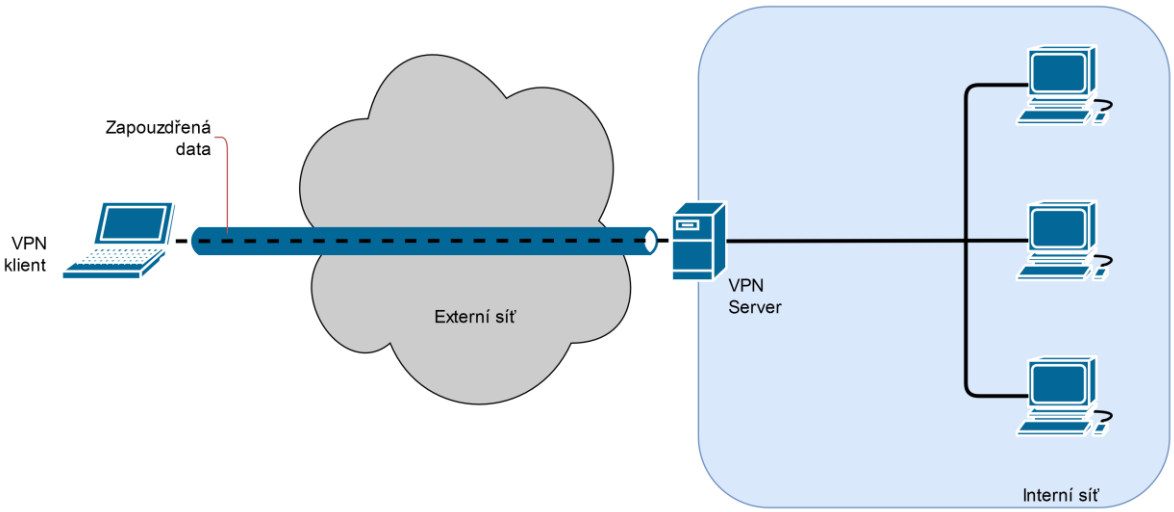
Dle vrstvy modelu OSI:

- **Spojová vrstva.** Spojuje sítě na L2 vrstvě modelu OSI. Příklady jsou L2VPN na MPLS sítích, PPTP či L2TP.
- **Síťová vrstva.** Poskytuje komunikaci mezi sítěmi na L3 vrstvě modelu OSI. Příklady jsou IPsec, IPIP (IP-in-IP), GRE (Generic Routing Encapsulation).

- **Transportní vrstva.** Zajišťuje komunikaci mezi sítěmi na vrstvě L4 modelu OSI. Příklady jsou OpenVPN.
- **Aplikační vrstva.** Zaručuje zabezpečený tunel mezi protistranami na L7 vrstvě modelu OSI. Příklady jsou SSH tunnel nebo Clientless SSL.

Dle cíle:

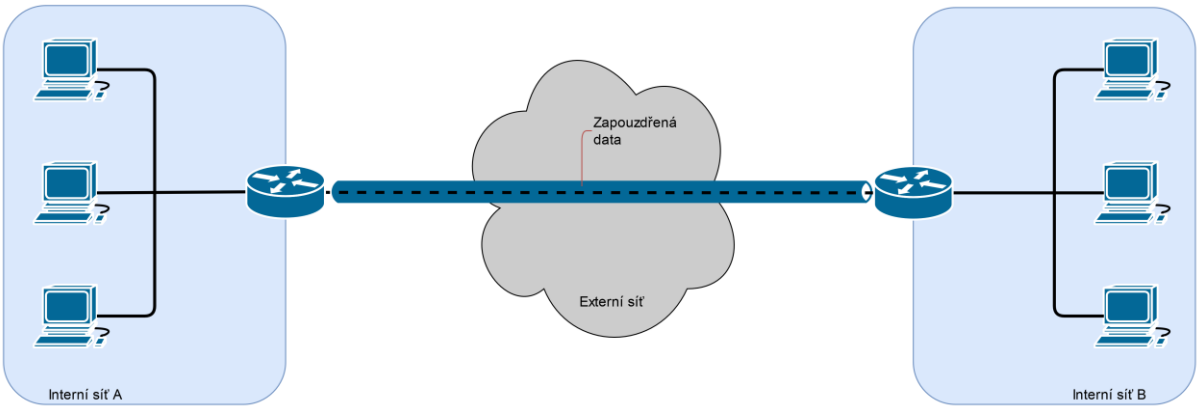
- **Vzdálený přístup** (Remote Access VPN) (Obr. 2.4) umožňuje připojení jednotlivého počítače k interní síti. Vzdálený uživatel je připojen k VPN prostřednictvím přístupového serveru, který je připojen k interní i externí (veřejné) síti. Při připojení vzdáleného uživatele (nebo při navazování spojení s jinou zabezpečenou sítí) vyžaduje přístupový server proces identifikace a poté proces autentizace. Po úspěšném dokončení obou procesů má vzdálený uživatel oprávnění pro práci v interní síti, tj. dochází k procesu autorizace.



Obr. 2.4. VPN sítě se vzdáleným přístupem

- **Site-to-site VPN.** Pro vzájemné propojení několika od sebe geograficky vzdálených sítí mohou být použity buď vyhrazené WAN linky, což výrazně zvyšuje náklady na vybudování a správu korporativní sítě, nebo site-to-site VPN síť, která umožňuje sloučení distribuovaných poboček do jediné zabezpečené sítě, na bázi otevřených komunikačních kanálů (Obr 2.5).
  V některých případech site-to-site VPN umožňuje připojení k interní síti vnějších uživatelů z jiných sítí, např. zákazníky nebo dodavatele. Kvůli tomu, že úroveň důvěry k externím uživatelům je mnohem nižší, než k

interním, pro zabránění nebo omezení přístupu těchto osob k cenným a důvěrným informacím je potřeba zařídit speciální ochranná opatření.



Obr. 2.5. Site-to-site VPN

2.4. Ohrožení bezpečností

Vzhledem ke struktuře a principům vytvoření VPN můžeme definovat základní body ohrožující bezpečnost:

- **Útoky na použitý kryptografický algoritmus.** Vyžadují obrovský výpočetní výkon pro prolomení algoritmů, které se v současně době používají pro šifrování, proto je můžeme zatím považovat za neprolomitelné.
- **Útoky na kryptografické klíče.** Protože data jsou obvykle přenášena přes nezabezpečenou síť, je nutné implementovat výměnu šifrovacích klíčů mezi stranami VPN. Táto fáze může být použita pro zachycení klíčů.
- **Útoky na protokoly VPN.** V současné době se k realizaci VPN používá řada protokolů, včetně IPSec, PPTP, L2TP. Tyto protokoly samy o sobě data nešifrují, pouze určují, jaké se použijí šifrovací algoritmy a řadu dalších podmínek nezbytných pro vytvoření VPN (včetně monitorování integrity, autentizace apod.). Nesprávná nastavení a implementace sítě se mohou stát ohrožením bezpečnosti.
- **Útoky na autentifikační protokoly.** Navázání spojení mezi uživateli vyžaduje jejich autentifikaci. Jako protokoly používají RADIUS, TACACS , včetně TACACS+, a také certifikáty. Základním principem takových útoků je pokušení útočníka vydat se za jinou stranu procesu autentifikace.
- **Chyby realizace a lidský faktor.** Špatná realizace předávání nosičů s klíči a certifikáty uživatelům, ztráta takových nosičů, slabá hesla a nedostatečně silný hlavní klíč se mohou stát slabými místy bezpečnosti.

3. Protokoly VPN

3.1. Současné protokoly VPN

3.1.1. IPsec

IPsec (Internet Protocol Security) je standard poskytující bezpečnostní služby v IP vrstvě tím, že umožňuje zvolit požadovaný bezpečnostní protokol, stanoví algoritmy, které budou použité pro služby, a zajišťuje výměnu klíčů nezbytných pro zabezpečení požadovaných služeb [3].

Pokud se jedná o IPsec VPN, je třeba si povšimnout zásadních bodů tohoto standardu:

Bezpečnostní asociace SA s určitými metodami a parametry, které se budou používat pro zabezpečení komunikačního kanálu. SA zajišťuje další funkce:

1) důvěrnost pomocí šifrování;
2) autentizace, a v případě použití certifikátů taky nepopiratelnost;
3) integritu.

IPsec tunel potřebuje dvě SA, jednu pro každý směr. Každá z nich je identifikována vlastním SPI (Security Parameter Index), tento index umožňuje rozdělit toky s různými pravidly pro autentizace, šifrování a použitý bezpečnostní protokol (AH nebo ESP). Jinými slovy SA seskupuje následující komponenty pro bezpečnostní spojení:

1) bezpečnostní algoritmy a klíče;
2) režim protokolu, buď transportní, nebo tunelovací;
3) metoda výměny klíčů;
4) životnost SA.

Řízení klíčů a jejich distribuce je nejdůležitější složka standardu. Je možné ho rozdělit do dvou typu: ruční a automatické.

Ruční řízení je jednoduchou formou, kde pro každý prvek VPN sítě bude klíč zadán ručně. Tento způsob řízení je vhodný pro malé sítě.

Automatické řízení se používá ve velkých a rozšiřujících se sítích. IPsec podporuje automatickou generaci a výměnu klíčů pomocí protokolu IKE (Internet Key Exchange).

Bezpečnostní algoritmy IPsec. IPsec používá dva algoritmy pro zabezpečení v síťové vrstvě:

Autentizační záhlaví AH (Authentication Header) - protokol pro autentizaci zdrojů IP paketů a ověřování integrity jejich obsahu. Možnost ověření autentičnosti a integrity paketů je realizována pomocí kontrolního součtu na

základě HMAC (Hash Message Authentication Code) s použitím tajného klíče nebo pomocí hashovací funkce MD5 (Message Digest 5) nebo SHA (Secure Hash Algorithm) s různou délkou výstupního otisku [4].

ESP (Encapsulating Security Payload) - protokol pro šifrování IP paketu a autentizaci jeho obsahu. Jako nástroj pro autentizaci a kontrolu integrity ESP, jako AH, používá MD5 nebo SHA algoritmy. Pro šifrování může být zvolen jeden ze šifrovacích algoritmů [5]:

1) 3DES (Triple DES) - šifrování je realizováno ve třech rundach s použitím 168 bit dlouhého klíče. 3DES sice poskytuje dostatečnou ochranu, ale není doporučen pro šifrování dat s vyšším stupněm tajnosti.
2) AES (Advanced Encryption Standard) - symetrický blokový šifrovací algoritmus s klíčem délky 128/192/256 bit

Sestavení tunelu pro IKEv1 se skládá ze dvou fází:

První fáze v hlavním režimu se skládá ze třech oboustranných výměn mezi Iniciátorem a Responderem (Obr. 3.1) [6], [7]:

1) Iniciátor odesílá Responderovi výzvu ①, která obsahuje nabídku s algoritmy pro autentizaci, šifrování, kontrolu integrity a DH grupu. Responder odesílá zpátky odpověď ②, ve které souhlasí s nabídkou.
2) Iniciátor odesílá výzvu ③, která obsahuje data pro vygenerování tajného klíče Responderem v rámci DH a náhodné číslo (nonce), které může být použito jen jednou, čímž zabrání replay útokům. Responder zpátky odešle odpověď ④ s daty pro vygenerování tajného klíče Iniciátorem v rámci DH a svoje nonce.
3) Vzájemná identifikace a autentizace Iniciátorem ⑤ a Responderem ⑥.

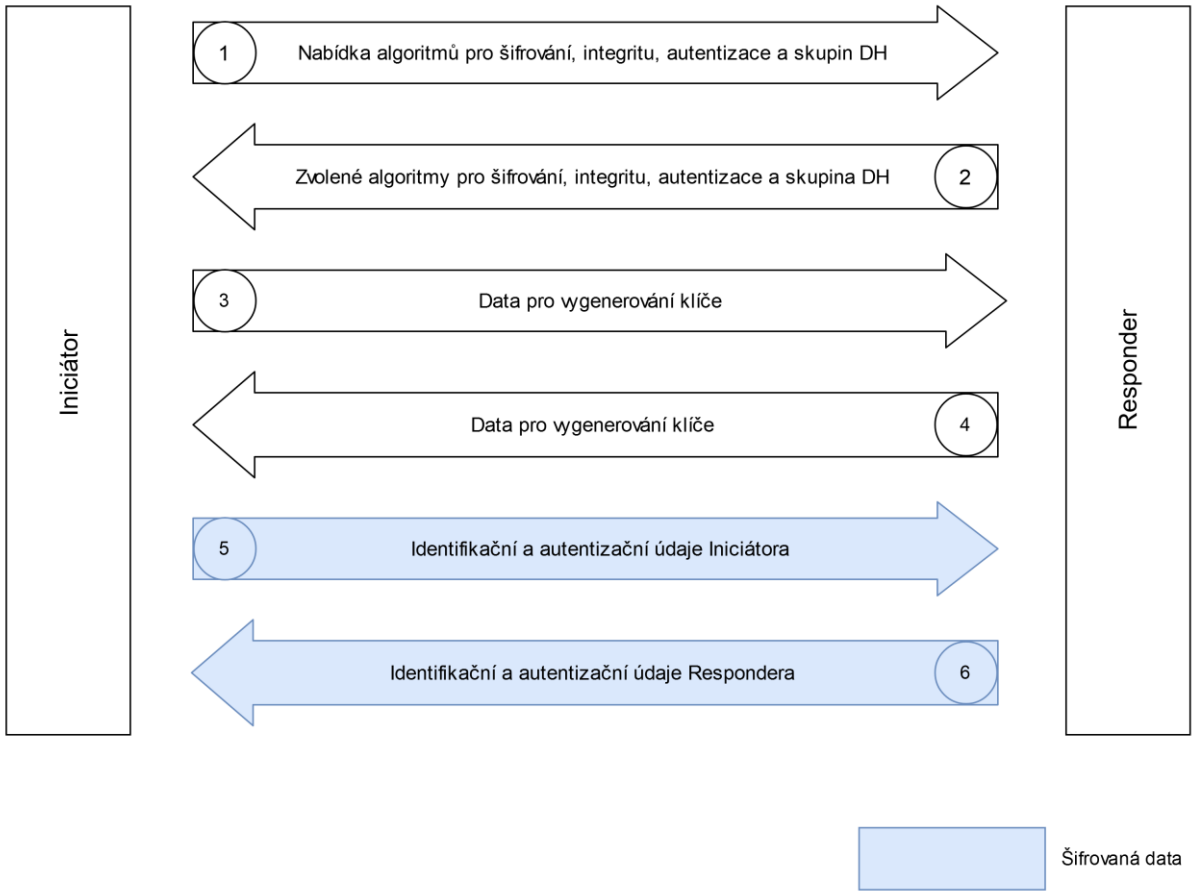
Po úspěšném ukončení první fáze vzniká Phase 1 SA neboli IKE SA. Daný tunel neslouží k přenosu uživatelských dat, používá se pouze pro výměnu informací ve Phase 2.

Ve druhé fázi, tzv. rychlém režimu, jsou všechny výměny šifrovány tajným klíčem dohodnutým v první fázi (Obr. 3.2):

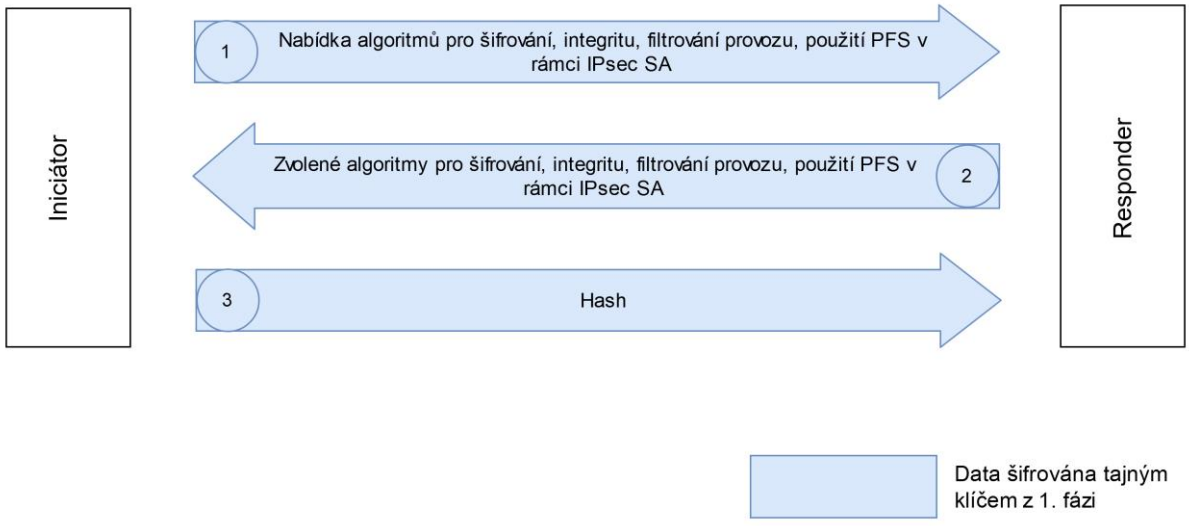
1) Iniciátor odešle výzvu ① s nabídkou algoritmů pro šifrování, kontrolu integrity, filtrování provozu a použití PFS (Perfect Forward Secrecy) pro určité spojení.
2) Responder odpoví ② souhlasem s nabídkou.
3) Iniciátor odešle ③ s hashem odpovědi ②.

Po úspěšném ukončení druhé fáze vzniká Phase 2 SA neboli IPsec SA.

Dvoufázová realizace umožňuje použití několika IPsec SA s různými stupni zabezpečení pro různé druhy trafiku v rámci jedné IKE SA. Vyjednávání o ustanovení nových SA nebo o obnovení existujících SA prochází přes tunel sestavený ve fázi 1, který má delší životnost oproti životnosti SA ze 2. fáze.

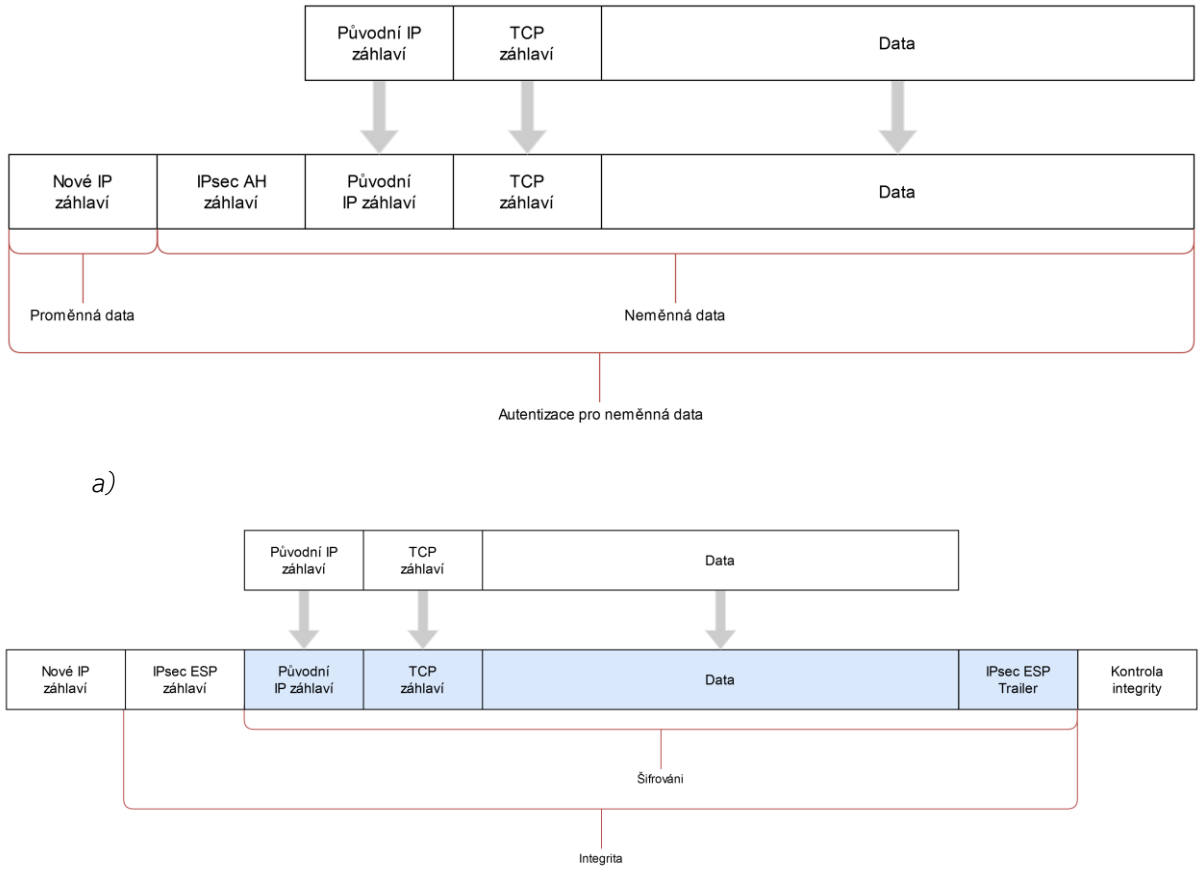


Obr. 3.1. Vyjednávání parametrů spojení v první fázi IKEv1 v hlavním režimu



Obr. 3.2. Vyjednávání parametrů spojení ve druhé fázi IKEv1

Potom se data zapouzdřená do IPsec paketů (Obr. 3.3) posílají přes IPsec tunel.



Obr. 3.3. Zapouzdření IP paketu v a) IPsec AH paket v tunelovacím režimu [8]; b) IPsec ESP paket v tunelovacím režimu [9]

Avšak během využití bylo zjištěno, že některé vlastnosti verze IKEv1, která byla vypracována v roce 1998, mohou být považovány za závady, které byly opraveny ve verzi IKEv2 [10]:

- velký počet zpráv pro navázání spojení v první fázi (9 v hlavním režimu nebo 6 v agresivním režimu);
- velký počet autentizačních metod, přičemž obě komunikující strany musejí mít stejnou metodu;
- IPsec SA umožňuje jedinou kombinaci rozsahů cílových a zdrojových IP adres a taky cílových a zdrojových portů;
- životnost SA je stanovená předem a SA nemůže být zrušena, i když mezi komunikujícími stranami není žádný datový tok;

- NAT-T je definován jako doplnění k protokolu;
- detekce žijící SA je definována jako doplnění k protokolu;
- není podporován vzdálený přístup, pouze ve formě implementace jednotlivými výrobci;
- neexistuje podpora mobilních uživatelů.

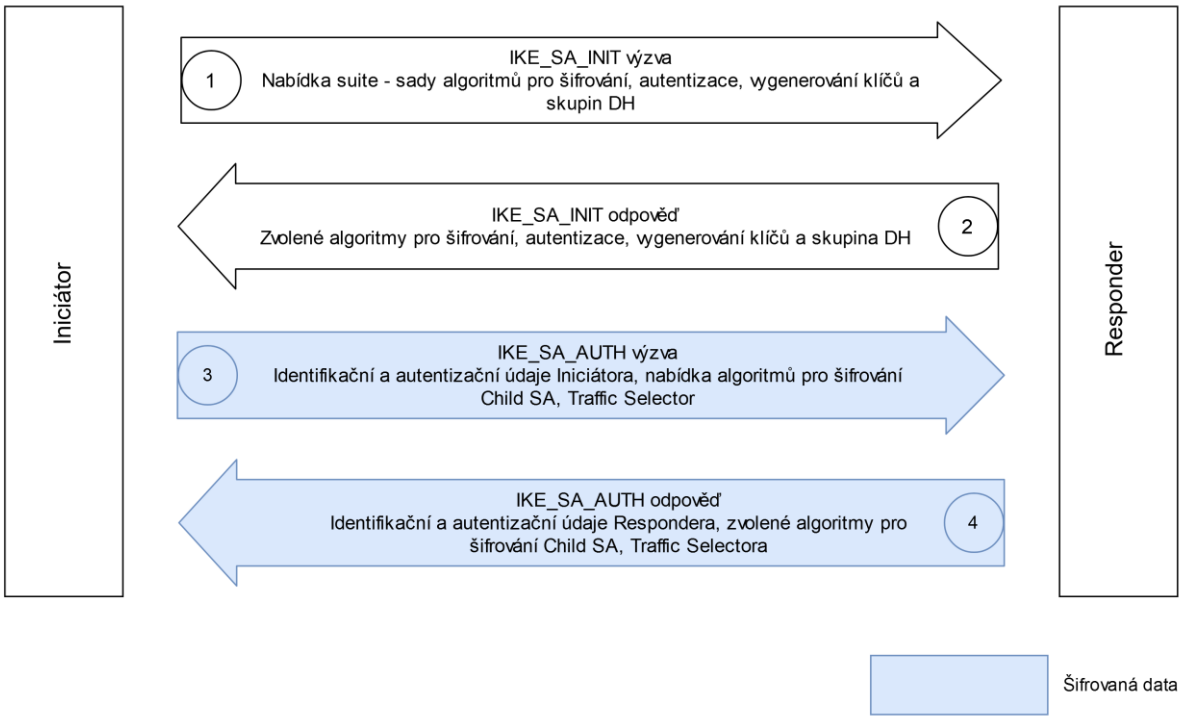
3.1.2. IKEv2/IPsec

Za účelem opravy chyb první verze IKE, její zjednodušení, zesílení bezpečnosti a adaptace k současným trendům (v první řadě k rostoucí mobilitě uživatelů), v roce 2005 vznikla nová verze IKEv2:

- Do IKEv2 byla zahrnuta možnost EAP (Extensible Authentication Protocol) autentizace a podpora MOBIKE (Mobility and Multihoming Protocol), což umožňuje použití IPsec v mobilních platformách s mnoha uživatelskými nastaveními.
- Je realizovaná kontrola stavu spojení, které může být obnoveno v případě zrušení, nebo ukončeno, pokud jedna ze stran "nežije" (Dead Peer Detection).
- Byl přidán komponent Control Plane, který zvyšuje ochranu před DoS útoky tím, že Responder v odpovědí na výzvu odešle tzv. cookie a čeká v další výzvě hodnotu této cookie. Pokud tu hodnotu nedostane, bude proces spojení ukončen.
- Podpora NAT-T je součástí protokolu. Zapouzdření IKE a ESP paketu do UDP na portu 4500 umožňuje průchod přes NAT. Vyšší spolehlivost je implementována pomocí striktního mechanismu výzva-odpověď, kde strana bude posílat zprávy, dokud nedostane odpověď nebo nezjistí, že protistrana "nežije".
- Zjednodušený jediný mechanismus dojednání spojení proti osmi možným různým druhům u IKEv1.

Na rozdíl od IKEv1, které obsahuje dvě fáze a výměnu šesti zpráv, probíhá u IKEv2 dojednání všech parametrů spojení v jediné fázi se dvěma výměnami, z nichž každá obsahuje pár výzva-odpověď (Obr. 3.4) [11]:

1) Iniciátor odesílá výzvu ① IKE_SA_INIT, která obsahuje tzv. kryptografický suite – sadu algoritmů pro zabezpečení SA. Responder odpoví ②, kde oznámí jaké z nabízených algoritmů byly zvoleny.
2) Během výměny výzvou ③ a odpovědí ④ vzniká první Child SA s domluvenými parametry pro zabezpečení. Pro vznik dalších Child SA se tento krok opakuje.



Obr. 3.4. Vyjednávání parametrů spojení IKEv2

3.1.3. L2TP/IPsec

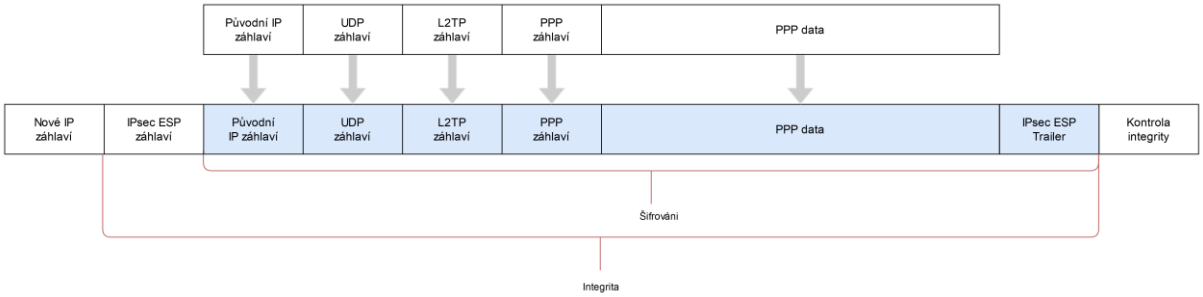
Tunelovací protokol 2. vrstvy L2TP je kombinací PPTP (Point-to-Point Tunneling Protocol) od společnosti Microsoft a L2F (Layer 2 Forwarding Protocol) od společností Cisco Systems Inc., který zahrnuje jejich nejlepší vlastnosti. L2TP přenáší zapouzdřené rámce PPP (Point-to-Point Protocol) přes sítě s přepojováním paketů, dnes nejčastěji přes IP sítě.

L2TP umožňuje založit tunely s požadovanými prioritami přístupu za účelem realizace VPN sítě, avšak kvůli nedostatku šifrovacích a autentizačních mechanismů se používá ve spojení s bezpečnostním protokolem IPsec (IP Security). Celé navázání spojení L2TP over IPsec pak probíhá následujícím způsobem [12]:

1) Vyjednávání o navázání bezpečné asociace (SA, Security Association) obvykle prochází přes UDP port 500 pomocí protokolu IKE (Internet Key Exchange) za použití buď předsdíleného klíče (pre-shared keys), nebo veřejných klíčů, nebo X.509 certifikátů.
2) Navázání ESP (Encapsulating Security Payload) komunikace v transportním režimu. V této fázi vzniká zabezpečený kanál, ale nedochází k tunelování.

3) Vyjednávání a realizace L2TP tunelu mezi koncovými body SA. V dané etapě vyjednávání parametrů probíhá přes zabezpečený kanál mezi SA v rámci šifrování IPsec na portu 1701 UDP.

Po ukončení procesu jsou L2TP pakety zapouzdřené v IPsec pakety (Obr. 3.5), původní zdrojová i cílová IP adresa jsou šifrované.



Obr. 3.5. Zapouzdření L2TP rámce v IPsec ESP paket v tunelovacím režimu [9]

Vzhledem k vlastnostem IPsec použití překladu síťových adres NAT (Network Address Translation) během realizace VPN spojení může vyvolávat řadu problémů.

Zařízení NAT překládá zdrojovou adresu odchozího paketu na vnější adresu. Zdrojovou adresu a nový zdrojový port zařízení NAT ukládá do tabulky, která bude použita pro zpětný překlad při příjmu odpovědí a jejich směrování k původnímu hostu. Avšak kvůli použití ESP v tunelovacím režimu jsou TCP/UDP záhlaví neviditelná a nelze je použít k překladu externích adres na interní adresy a naopak, což vede k tomu, že pakety budou vyřazené. Kromě toho se při překladu cílové adresy změní hodnota ICV (Integrity Check Value), která bude vypočítaná na druhé straně, a proto integrita paketu bude považována za porušenou.

Řešením je průchod skrz NAT neboli NAT-T (NAT traversal) protokol, který přidává IPSec ESP paketům UDP záhlaví a zapouzdřuje je do UDP paketů. Zařízení, které implementuje NAT-T, posílá nové UDP pakety přes externí síť bez ohledu na jejich obsah. Poté, co paket dorazí do cíle, bude hlavička UDP odstraněna a dále se pracuje s IPSec ESP paketem.

3.1.4. OpenVPN

OpenVPN je svobodný software s otevřeným zdrojovým kódem pro realizaci zabezpečených kanálů typu bod-bod nebo klient-server mezi počítači. Distribuuje se buď v podobě zdrojového kódu, nebo pro instalaci připravených balíčků zkompilovaných pro různé operační systémy.

Ve výchozím nastavení OpenVPN funguje na základě UDP na portu 1194, ale například v restriktivních sítích může být také použit TCP a jiný port, např. 443 [13].

Obr. 3.6. Zobecněná struktura OpenVPN paketu

Pro šifrování OpenVPN používá knihovnu OpenSSL, zejména RSA šifrování s klíčem délky 1024 až 4096 bitů, nejčastěji 2048. Pro zabezpečení informací s vyšším stupněm tajnosti mohou být aplikovány klíče větší délky nebo šifrovací protokoly AES, 3DES, Blowfish atd.

Autentizace může být realizována následujícími způsoby:

1) Pomocí certifikátů. Na základě knihovny OpenSSL může být snadno vytvořena infrastruktura privátních klíčů PKI (Private Key Infrastructure), která vydává uživatelské certifikáty. Certifikační autorita zajišťuje řízení přístupu uživatelů k OpenVPN serveru. Po uplynutí doby platnosti certifikátu nebo jeho revokace, bude-li certifikát kompromitován, přístup pro uživatele bude blokován a certifikát bude zapsán do revokačního seznamu certifikátů CRL (Certificate Revocation List).
2) Privátní bod-bod klíče. Varianta pro jedinečná připojení jednoho uživatele k interní síti. Jeden z komunikujících hostů vygeneruje pro klienta a server společný klíč, který platí po dobu spojení.
3) Pomocí jména a hesla. Toto schéma umožňuje připojení klienta na základě již existujícího jména a hesla pro přístup např. k doméně. Certifikát serveru bude pořád zapotřebí.

Posílení zabezpečení je zajištěné přidáním HMAC podpisu ke každému SSL/TLS handshake paketu. Jakýkoliv UDP paket beze správného HMAC podpisu bude vyhozen [14].

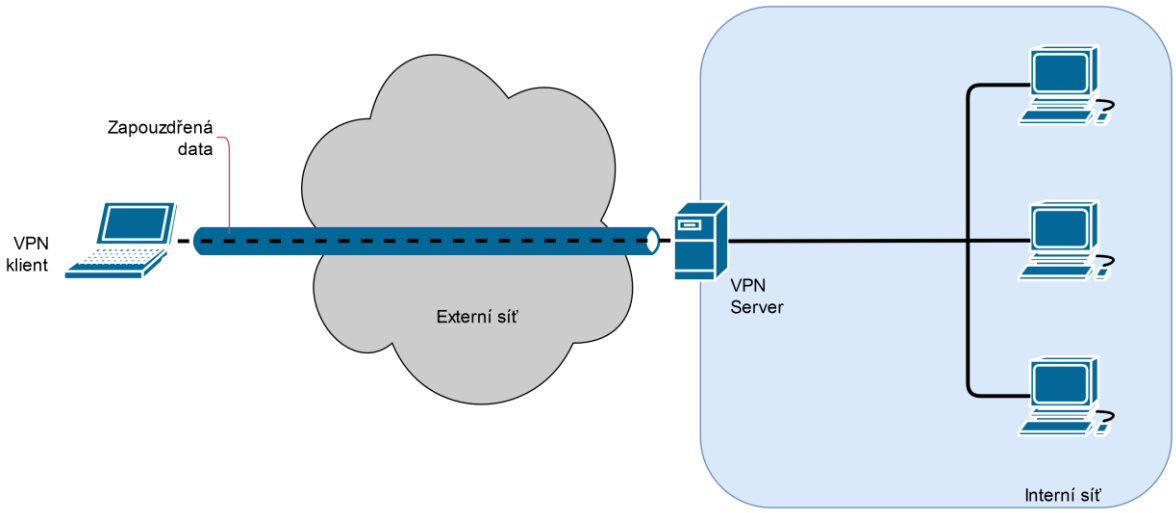
3.1.5. SSL VPN

Virtuální privátní síť ve vrstvě bezpečných socketů Secure Socket Layer VPN, dává možnost vzdáleným uživatelům se bezpečně připojit k Web a klient-server aplikacím, síťovým službám a adresářům interní sítě přes veřejnou nezabezpečenou síť bez nutnosti instalovat speciální klientský software.

SSL, nebo v současné době jeho příjemce TLS (Transport Layer Security), zabezpečuje důvěrnost end-to-end šifrováním (E2EE), integritu, autentizaci klienta a serveru [15]. Pro asymetrické šifrování, autentizaci protistran a vyjednávání symetrického klíče relace tento druh VPN využívá X.509 certifikáty.

Existují tři typy SSL VPN:

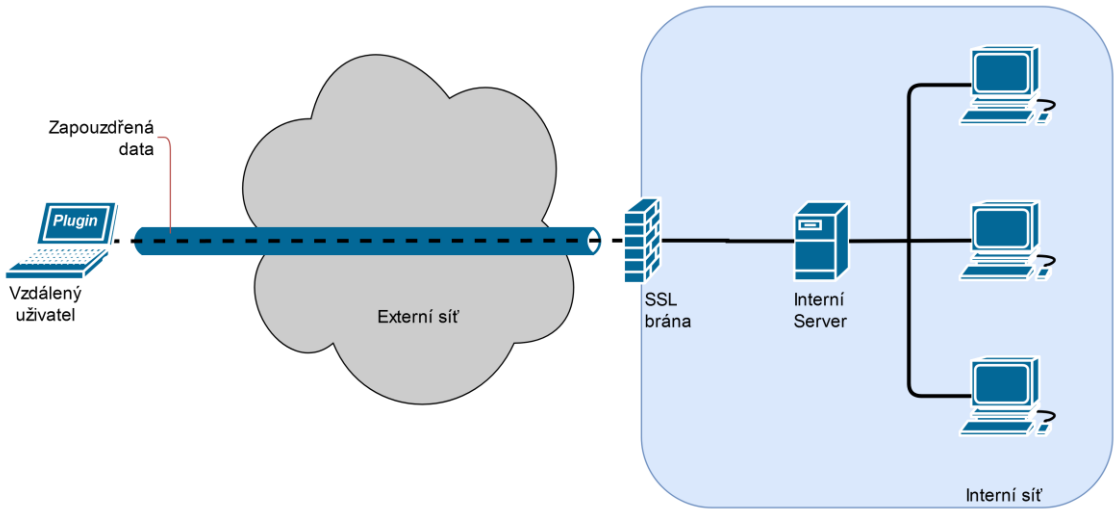
1) Proxy SSL (Obr. 3.7.). Jednoduché řešení z hlediska architektury sítě. SSL připojení je realizováno mezi vzdáleným uživatelem a SSL bránou. SSL brána v tomto případě je Application Level Gateway (ALG), která je určena pro jednotlivý typ aplikačního protokolu a je pro něho transparentní. Pro každý nový typ aplikačního protokolu je vyžadován vlastní ALG.
   SSL brána emuluje protistrany pro klienta i server. Dešifruje příchozí požadavky a pak je odesílá serveru, nebo, v opačném směru, šifruje odpovědi serveru a posílá je vzdálenému uživateli v zašifrované podobě.
   Pro klienta není nutné instalovat speciální software, pro připojení může být použit obecný web prohlížeč, který umožňuje obsluhu webových aplikací pomocí HTTPS.



Obr. 3.7. Proxy SSL

2) Předávání portu (Port Forwarding) - metoda, která funguje stejně jako u firewallů. Vzdálený uživatel dostává od ALG plugin Java-ActiveX, který na vzdáleném počítači jedná jako VPN SSL Proxy. Prohlížeč jako klientská aplikace komunikuje přímo s pluginem, jako kdyby komunikoval s lokálním proxy serverem. Plugin provádí předávání portu, tj. změní cílovou adresu v závislosti na cílovém TCP portu v paketu

zaslaném vzdáleným klientem. SSL relace je tedy navázána přímo mezi pluginem vzdáleného klienta a cílovou SSL bránou (Obr. 3.8).



Obr. 3.8. SSL port forwarding

3) Rozšíření sítě. Výše uvedené typy mají omezení v tom, že pracují pouze s webovými aplikacemi, tj. přes HTTP protokol s použitím SSL.
Rozšíření sítě dává možnost vyřešit tento problém, ale vyžaduje instalaci specializovaného softwaru u vzdáleného uživatele, který vytváří tunel mezi SSL bránou a klientem a převezme síťové připojení. Šifrování bude provedeno v aplikační vrstvě pomocí SSL.
Potom klientská aplikace šifruje všechna data a posílá je přes sestavenou TCP/SSL relaci na SSL bránu. Na SSL bráně budou data dešifrována a odeslána příjemci.

3.2. Nastupující protokoly VPN

3.2.1. WireGuard

Protokol WireGuard byl oficiálně vydán v roce 2018, je vyvinut Jasonem Donefeldem, zakladatelem společnosti Edge Security LLC [16], jako jednodušší a bezpečnější alternativa vůči stávajícím protokolům IPsec a OpenVPN [17], [18].

WireGuard je L3 tunelovací protokol s využitím UDP v transportní vrstvě. V Linuxových systémech navíc existuje možnost vytvořit UDP tunel přes TCP [19] nebo použít řešení třetích stran [20]. Jsou podporovány protokoly IPv4 a IPv6 a také enkapsulace v4-in-v6 a v6-in-v4. Umí překonat NAT a firewally, za předpokladu, že UDP není blokován. Podporuje změnu IP adresy VPN serveru bez přerušení spojení s automatickým přeladěním klienta.

Pro šifrování se používá proudová šifra ChaCha20 [21] a pro autentizaci a zjištění integrity zpráv MAC (Message Authentication Code) realizovaný algoritmem Poly1305. Vygenerování společného klíče se provádí pomocí Diffie-Hellmanova protokolu nad eliptickými křivkami s využitím Bernstainovy křivky Curve25519 [22]. Pro hashování jsou použity kryptografické hashovací funkce BLAKE2s [23] a SipHash-24 [24]. Funkce pro odvození klíče je realizována pomocí HKDF (Hashed Key Derivation Function) [25]. Časová značka TAI64N umožňuje zabezpečení ochrany před reply útoky.

Tento protokol je postaven na základě konceptu zvaného Cryptokey Routing:

- WireGuard vytváří rozhraní s názvem wg, kterému se přiřadí IP adresa, port na kterém bude poslouchat, veřejný a privátní klíč. Každé wg rozhraní v jeho konfiguračním souboru má informace o peerech – jejich veřejných klíčích a IP adresách (AllowedIPs). Tato informace hraje roli Cryptokey Routing Table. Během komunikace se do této informace přidávají údaje o fyzické IP adrese peeru (Endpoint), pokud nebyly zadány předem.
- Když se odchozí paket dostane na rozhraní wg, na základě IP adresy z tabulky bude zvolen peer, z jehož veřejného klíče bude odvozen klíč pro šifrování odchozího paketu.
- Když se příchozí paket dostane na rozhraní wg, bude dešifrován privátním klíčem příjemce, identifikován na základě veřejného klíče peera (pokud je takový zapsán v tabulce) a zdrojová IP adresa bude srovnána se záznamem v tabulce, jestli tomu záznamu odpovídá.
- V případě jakéhokoliv nesouladu bude paket vyhozen. Pro realizaci tunelu musí server dostat alespoň jeden správně šifrovaný paket.

3.2.2. SoftEther VPN

SoftEther (Software Ethernet) VPN je svobodný, multiplatformní, multiprotokolový software s otevřeným zdrojovým kódem, který umožňuje založení tunelu mezi SoftEther serverem a SoftEther klientem. Byl vyvinut na Univerzitě v Tsukubě a oficiálně představen v roce 2014 [26].

Software podporuje známé protokoly L2TP/IPsec, OpenVPN, MS-SSTP a také vlastní protokol na základě SSL VPN tunelování přes HTTPS, což dává možnost obejít NAT a Firewally. V restriktivních sítích dokonce existuje možnost VPN spojení na bázi ICMP nebo DNS protokolů.

SoftEther umožňuje spojení typu klient-server (L3) nebo site-to-site (L2) s použitím Ethernet přemosťování. Vzhledem k tomu, že SoftEther má vestavěné

DDNS (Dynamic DNS) a NAT-T, statická nebo fixní IP adresa u klienta není zapotřebí. Kromě toho je podporován IPv4/IPv6 dual-stacking.

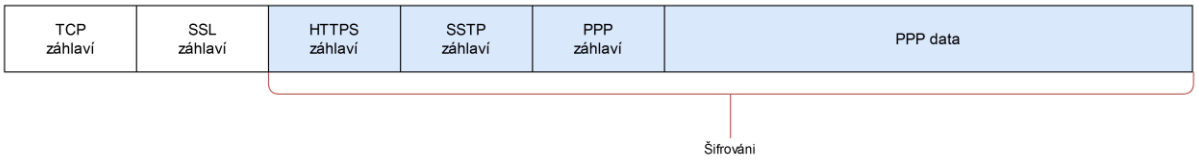
Šifrování je realizováno na základě AES-256 nebo RSA-4096, autentizace – pomocí RSA certifikátů.

V podstatě SoftEther je server se sbírkou VPN serverů pro L2TP, OpenVPN a SSTP protokoly s jediným grafickým rozhraním nebo CLI pro nastavení parametrů, a "vlastní" VPN protokol je obecný SSL VPN, který běží na portu 443 TCP. Kromě toho, dá se předpokládat, že realizace VPN přes ICMP nebo DNS může být zablokována vlastníkem Firewallu (nebo jiného omezujícího trafik zařízení) na základě DPI analýzy kvůli netypickému chování protokolu, když bude použit pro přenos uživatelských dat, anebo na základě sondy připojení (connection probe)[1].

3.2.3. MS-SSTP

Microsoft Secure Socket Tunneling Protocol je protokolem pro přenos rámců linkové vrstvy přes HTTPS spojení. Protokol momentálně podporuje zapouzdření pouze PPP rámce. Spojení probíhá na portu 443 TCP. Většinou se používá spojení bod-bod (klient-server) [27].

Šifrování dat je zajištěno pomocí šifrovacího protokolu, který bude zvolen během handshaku, nejčastěji pomocí AES. Pakety SSTP a PPP jsou přenášeny pouze ve šifrovaném tvaru (Obr. 3.9).



Obr. 3.9. Zobecněná struktura MS-SSTP paketu

Autentizace je prováděna všemi třemi protokoly SSL (autentizace SSTP serveru na základě certifikátu), SSTP a PPP (autentizace klienta serverem pomocí MS-CHAPv2, EAP-TLS, PEAP-MSCHAPv2, PEAP-TLS a dodatečná autentizace serveru).

Navázání spojení obsahuje následující kroky:

1) Klient otevírá TCP spojení na portu 443.

[1] Sonda připojení – metoda, která spočívá v tom, že provoz omezující zařízení na základě inspekce záhlaví, zejména cílového portu, zanalyzuje druh provozu/služby a pošle příslušný požadavek na tento port. Pak podle odpovědi protistrany bude detekováno, zda druhý bod opravdu poskytuje deklarovanou službu. V případě nesrovnalosti bude provoz zablokován.

2) Sestavení SSL/TLS spojení nad TCP. Autentizace serveru klientem na základě SSL certifikátu.
3) Prochází HTTPS handshake, po ukončení kterého se sestaví SSTP spojení. Klient posílá serveru dotaz na sestavení spojení Call Connect Request message. Všechny SSTP pakety jsou přenášeny přes HTTPS spojení.
4) Server odpoví na dotaz zprávou Call Connect Acknowledge message, která obsahuje 32bitové Nonce, které klient musí odeslat zpátky v dalším dotazu, a seznam hashovacích funkcí (SHA1 nebo SHA256) pro podpis tohoto dotazu.
5) PPP autentizace. Všechny PPP rámce jsou zapouzdřené do SSTP paketů, které jsou šifrovány SSL.
6) Klient odesílá zprávu Call Connected message, který obsahuje obdržené Nonce od serveru ve 3. kroku a hash certifikátu serveru. Tento dotaz se podepisuje HMAC na základě hashovací funkce zvolené z výše uvedených a klíčem získaným v etapě PPP autorizace.
7) Server kontroluje Call Connected message. SSTP spojení je sestaveno.
8) Nastavení PPP parametrů.

Pokud v tunelu během 60 vteřin nebyl přenášen žádný paket, protistrany navzájem posílají Echo Request pakety. Když v následujících 60 vteřinách protistrana nedostane odpověď, spojení bude zrušeno.

3.2.4. Další VPN protokoly

TINC. Svobodný, s otevřeným zdrojovým kódem protokol pro smíšenou topologii. Na rozdíl od běžných VPN protokolů nepotřebuje server - pokud je to možné, zařízení s nainstalovaným démonem komunikuje přímo s jiným zařízením bez meziskoků (peer-to-peer).

Šifrování je zajištěno pomocí knihoven OpenSSL nebo Libre SSL. Podporuje NAT-T a IPv6 a kompresi zlib nebo LZO [28].

FreeLAN je software s otevřeným zdrojovým kódem, který implementuje full mesh VPN pro realizace klient-server, peer-to-peer nebo hybridního spojení.

Šifrování je realizováno na základě knihovny OpenSSL. Autentizace je realizována pomocí předsdíleného klíče, jména a hesla, nebo certifikátů [29].

3.3. Závěr

Zásadní vlastnosti popsaných protokolů shrneme do následující tabulky

Tab. 3.1.

Zásadní vlastností některých VPN protokolů

	Transportní vrstva (protokol/port)	Remote Access	Site-to-site	Podpora IPv6	NAT-T	Šifrování	Autentizace
L2TP/IPsec	UDP, port 1701	Ano	Ano	Ano	Ano	AES	předsdílený klíč; X.509 certifikáty
IPsec	TCP UDP	Ano	Ano	Ano	Ano, jako volba UDP, port 4500	3DES; AES	ručně zadaný předsdílený klíč; automaticky odvozený klíč
IKEv2/IPsec	TCP UDP	Ano	Ano	Ano	Ano, povinná součást protokolu UDP, port 4500	AES	předsdílený klíč; automaticky odvozený klíč
OpenVPN	TCP/UDP, port 1194	Ano	Ano	Ano	Ano	Blowfish; RSA; AES; 3DES	jednorázový bod-bod klíč; X.509 certifikáty; jméno/heslo
SSL VPN	TCP UDP	Ano	Ne	Ano	Ano	RC4; IDEA; 3DES; SEED; Camellia; AES	autentizace serveru pomocí certifikátu (povinná); autentizace uživatele pomocí certifikátu (volba)
WireGuard	UDP, port 51820	Ano	Ano	Ano	Ano	ChaCha20	veřejné klíče peerů
SoftEther VPN	TCP UDP ICMP DNS	Ano	záleží na typu použitého VPN protokolu	záleží na typu použitého VPN protokolu	záleží na typu použitého VPN protokolu	záleží na typu použitého VPN protokolu	záleží na typu použitého VPN protokolu
SSTP	TCP, port 443	Ano	Ne	Ano	Ano	AES	autentizace serveru pomocí certifikátu (povinná); autentizace uživatele pomocí certifikátu (volba)

4. Vícefaktorová autentizace

Vícefaktorová autentizace MFA (Multi-Factor Authentication) – metoda kontroly přístupu uživatele, kde systém pro přístup k prostředkům vyžaduje od uživatele více než jeden důkaz autentizace (Obr. 4.1).

Tyto důkazy jsou založené na následujících faktorech:

- Znalost – něco, co uživatel zná. Utajená informace, kterou musí vlastnit pouze autorizovaný subjekt. Touto informací může být heslo, kódová kombinace, PIN.
- Vlastnictví – něco, co uživatel má. Nějaký unikátní předmět, který má v držení autorizovaný subjekt. Tímto předmětem může být token, který vygeneruje číslo, nebo mobilní/chytrý telefon. V tomto případě jednorázové heslo uživatel dostane přes SMS nebo ho vygeneruje speciální aplikace.
- Osobnost – čím uživatel je. Fyzická vlastnost subjektu, biometrie.

Obr. 4.1. Schematické zobrazení vícefaktorové autentizace

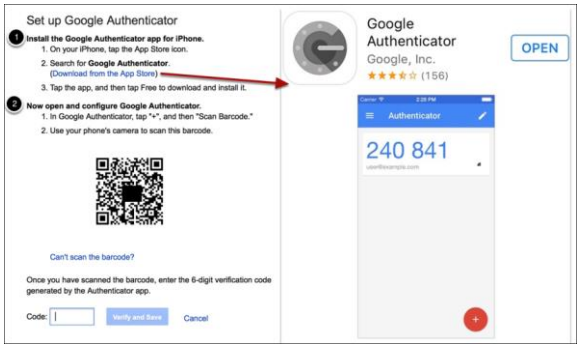
Ve většině případů se používají dva ze tří výše uvedených principů, pak se tento způsob autentizace označuje jako dvoufázová verifikace (Two Step Verification, 2SV) nebo dvoufaktorová autentizace (Two Factor Authentication, 2FA).

Dvoufázová verifikace probíhá ve dvou postupných fázích. V první fázi se provede kontrola jména a hesla, potom bude na odpovídající emailovou adresu nebo telefonní číslo odeslán kód. V dnešní době je 2SV přes SMS považována za nebezpečnou kvůli zranitelnosti SS7 (Signaling System No 7) [30] nebo možnosti v některých případech snadno naklonovat SIM kartu a tím získat přístup ke zprávám. Kromě toho zpráva může být zachycena útokem MITM (man-in-the-middle).

Při dvoufaktorové autentizaci budou její výsledky až po předložení dvou faktorů. 2FA předpokládá, že vygenerování jednorázového hesla OTP (One Time

Password) bude provedeno zařízením, které vlastní uživatel – token (Obr. 4.2) nebo chytrý telefon s nainstalovanou speciální aplikací (Obr. 4.3).

Obr. 4.2. Ukázkový příklad tokenů [31]



Obr. 4.3. Ukázkové zobrazení použití aplikace Google Authenticator [32]

Existují dva typy jednorázových hesel – na základě událostí (Event Based OTP) nebo na základě času (Time Based OTP).

Jednorázové heslo na základě události HOTP (HMAC based OTP) se vypočítá na bázi dvou hodnot [33]:

1) tajný klíč, který mají klient a server, doporučené délky 160 bitů. Každý generátor HOTP musí mít různý i unikátní klíč;
2) 8 bajtová hodnota čítače, událost, která se zúčastní v generování hesla (moving factor). Čítač mají klient a server. Hodnota čítače klienta bude inkrementovaná po každém vygenerování hesla, čítač serveru bude inkrementován po každé úspěšné validaci hesla. Hodnoty čítačů klienta a serveru se mohou lišit o nějakou veličinu, ale po překročení musejí být synchronizované.

Z těchto hodnot bude pomocí HMAC-SHA1 vygenerován hash délky 20 bajtů, z nichž určitým způsobem budou vytaženy 4 bajty, pak převedeny do desítkového čísla. Toto číslo modulo 10^n (n – počet číslic v hesle) je vlastně jednorázovým heslem.

Jednorázové heslo na základě času TOTP má stejný princip jako HOTP, ale místo hodnoty čítače se jako moving factor používá časový interval (time step), ve kterém platí heslo. Hodnota intervalu se vypočítá dělením Unixového času délkou časového intervalu [34].

Díky tomu, že se v případě TOTP heslo mění po každém časovém intervalu (obvykle 30 vteřin), i když nebylo použito, uplatnění TOTP může být považované za bezpečnější.

5. Způsob měření doby přenosů a přenosové rychlosti VPN protokolů

Pro porovnávání různých VPN protokolů je nutné zvolit parametry, na jejichž základě existuje možnost provést odhad. Zřejmé je, že z hlediska uživatelů jsou takovými parametry doba přenosu určitého objemu dat a rychlost přenosu.

Zhodnotit uvedené parametry můžeme pomocí následujících utilit pro Linux systémy:

- dd – utilita umožňující přenos (kopírování) bloků dat určité velikosti;
- pv (pipeviewer) – utilita, která zobrazuje ukazatel průběhu (progress bar);
- nc (netcat) – utilita umožňující sestavení TCP nebo UDP spojení s přenosem dat přes toto spojení.

Tak celý příkaz pro posílání dat bude mít podobu:

`dd if=/dev/urandom bs=1000 count=1000 | pv | nc [cílová IP adresa] [cílový port]`, kde:

`dd if=/dev/urandom` – vygenerování bloků náhodných dat;

`bs=1000` – velikost bloku, [bajt];

`count=1000` – počet bloků;

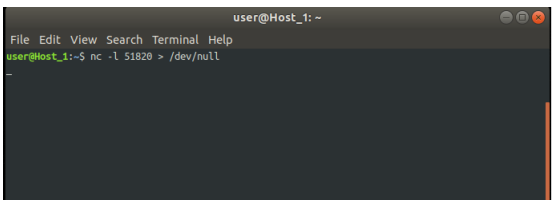
`pv` – zobrazení ukazatele průběhu;

`nc [cílová IP adresa] [cílový port]` – sestavení TCP spojení s cílovou IP adresou a portem.

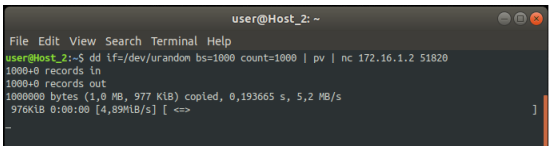
Příkaz pro příjem dat zapíšeme jako:

```
nc -l 51820 > /dev/null
```

Výsledky použití uvedených příkazů jsou zobrazené na Obr. 5.1 a 5.2.



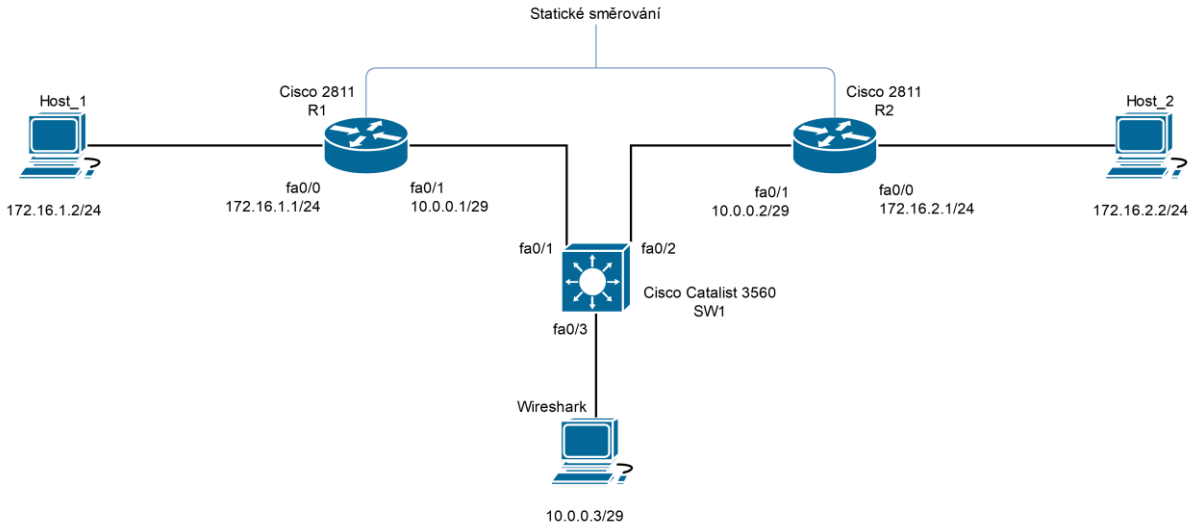
Obr. 5.1. Nastavení na Host_1



Obr. 5.2. Nastavení na Host__2

Schéma zapojení uvedené na Obr. 5.3 poskytuje možnost provést odhad parametrů pro dva základní typy topologií:

- site-to-site, s použitím IPSec protokolu;
- remote access, s použitím protokolů OpenVPN a WireGuard.



Obr. 5.3. Schéma pracoviště

Host__1 a Host__2 – jsou počítače, mezi nimiž budou přenášena data;

R1 a R2 – jsou směrovače Cisco 2811, které slouží jako jednoduchá implementace externí sítě.

SW1 a Wireshark – jsou přepínač Cisco Catalyst 3560, který funguje v režimu SPAN (Switched Port Analyzer), a počítač s aplikací Wireshark pro monitorování provozu.

Měření budeme provádět pro bloky dat velikosti 100, 1 000, 10 000 a 100 000 bajtů, každý blok dat budeme posílat 100x, 1 000x, 10 000x a 100 000x. Každý krok měření budeme opakovat 10krát.

Výsledky měření zapíšeme ve tvaru:

$$\mu = \bar{x} \pm \Delta x \qquad [5.1]$$

kde:

$\bar{x}$ – střední hodnota, která se vypočte jako

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \qquad [5.2]$$

Δx – standardní chyba, která se vypočte dle vzorce

$$\Delta x = t \cdot \sqrt{\frac{\sum_{i=1}^{n} \left(\bar{x} - x_i\right)^2}{n(n-1)}} \qquad [5.3]$$

kde

n – počet opakování, $n = 10$;

t – Studentův koeficient, $t = 2{,}262$ pro $n = 10$ a konfidenční interval $P = 0.95$

Výsledky měření zapíšeme do tabulky:

Tab. 5.1

Vzorová tabulka pro zápis výsledků

Objem přenesených dat

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	10 kB	100 kB	1 MB	10 MB
	1 000	100 kB	1 MB	10 MB	100 MB
	10 000	1 MB	10 MB	100 MB	1 GB
	100 000	10 MB	100 MB	1 GB	10 GB

Doba přenosu [s]

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**				
	1 000				
	10 000				
	100 000				

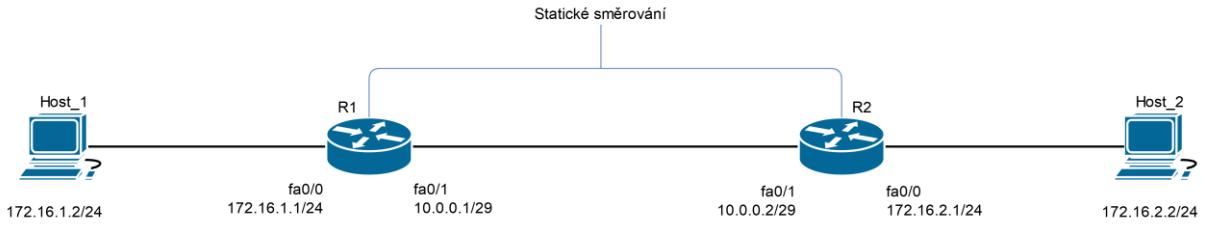
Přenosová rychlost [MBps]

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**				
	1 000				
	10 000				
	100 000				

5.1. Odhad doby přenosů a přenosové rychlosti bez VPN protokolů

Pro referenci provedeme měření bez VPN protokolů.

Schéma zapojení je uvedené na Obr. 5.4. Konfigurace směrovačů a přepínače jsou uvedené v příloze A v tabulkách A.1 až A.3.



Obr. 5.4. Schéma zapojení pro měření bez VPN protokolů (zde a dále přepínač SW1 a host Wireshark nebudou zobrazené)

Příkazový řádek pro Host_1:

```
nc -l 51820 > /dev/null
```

Pro Host_2:

```
dd if=/dev/urandom bs=[délka bloku dat] count=[počet bloků] | pv | nc 172.16.1.2 51820
```

Parametry `bs` a `count` musejí být nastavené podle uvedených výše hodnot.

Výsledky měření jsou uvedené v Tab. 5.2.

Tab. 5.2

Výsledky měření odhadů doby přenosů a přenosové rychlosti bez VPN protokolů

Objem přenesených dat

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	10 kB	100 kB	1 MB	10 MB
	1 000	100 kB	1 MB	10 MB	100 MB
	10 000	1 MB	10 MB	100 MB	1 GB
	100 000	10 MB	100 MB	1 GB	10 GB

Doba přenosu [s]

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	0.003 ± 0.003	0.02 ± 0.01	0.09 ± 0.01	0.835 ± 0.007
	1 000	0.008 ± 0.008	0.06 ± 0.01	0.716 ± 0.009	8.293 ± 0.006
	10 000	0.069 ± 0.005	0.706 ± 0.009	8.298 ± 0.008	84.775 ± 0.009
	100 000	0.712 ± 0.006	8.30 ± 0.01	84.77 ± 0.01	849.56 ± 0.03

Tab. 5.2 (dokončení)

		Přenosová rychlost [MBps]			
		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	7 ± 1	11 ± 2	11.6 ± 0.9	11.97 ± 0.05
	1 000	22 ± 4	16 ± 1	13.98 ± 0.08	12.07 ± 0.02
	10 000	14.7 ± 0.5	14.18 ± 0.09	12.04 ± 0.02	11.8 ± 0
	100 000	14.04 ± 0.06	12.06 ± 0.02	11.8 ± 0	11.8 ± 0

5.2. Odhad doby přenosů a přenosové rychlosti pro IPSec

Schéma pracoviště pro měření IPSec protokolu je uvedeno na Obr. 5.5.

Obr. 5.5. Schéma zapojení pro měření parametrů IPSec protokolu

Pro dané měření doplníme konfigurace směrovačů příkazy uvedenými v tabulkách A.4 a A.5.

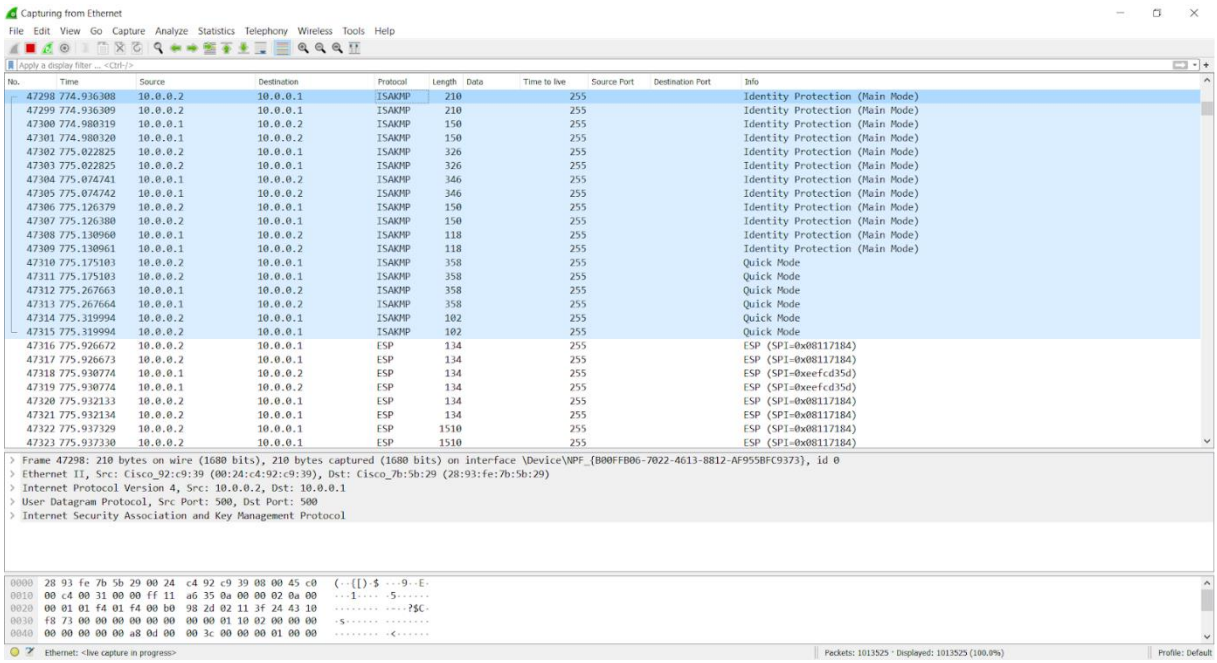
Příkazový řádek pro Host_1:

```
nc -l 51820 > /dev/null
```

Pro Host_2:

```
dd if=/dev/urandom bs=<délka bloku dat> count=<počet bloků> | pv
| nc 172.16.1.2 51820
```

Na Obr. 5.6. je vidět první, druhou fázi zahájení IPSec spojení a část provozu.



Obr. 5.6. Fragment záchytu provozu IPSec

Výsledky měření jsou uvedené v Tab. 5.3.

Tab. 5.3

Výsledky měření odhadů doby přenosů a přenosové rychlosti pro IPSec protokol

Objem přenesených dat

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	10 kB	100 kB	1 MB	10 MB
	1 000	100 kB	1 MB	10 MB	100 MB
	10 000	1 MB	10 MB	100 MB	1 GB
	100 000	10 MB	100 MB	1 GB	10 GB

Doba přenosu [s]

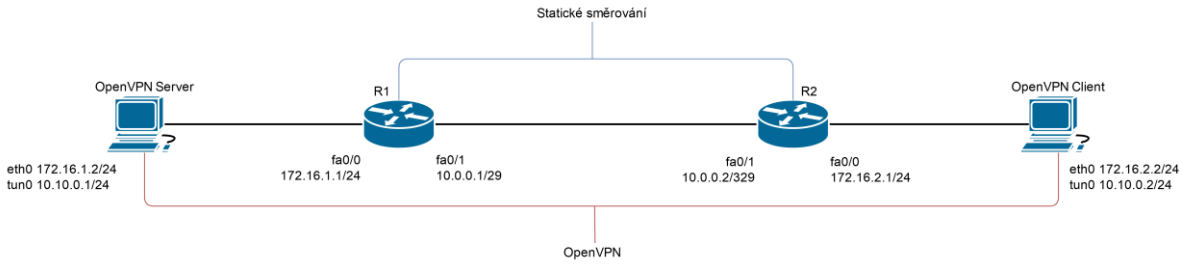
		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	0.002 ± 0.001	0.0053 ± 0.0007	0.14 ± 0.02	1.93 ± 0.03
	1 000	0.0043 ± 0.0005	0.15 ± 0.01	1.94 ± 0.03	20.6 ± 0.1
	10 000	0.12 ± 0.02	1.95 ± 0.03	24.0 ± 7.0	209.0 ± 2.0
	100 000	1.94 ± 0.02	20.78 ± 0.18	209.0 ± 2.0	2084.0 ± 10.0

Přenosová rychlost [MBps]

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	8.0 ± 2.0	20.0 ± 3.0	7.0 ± 1.0	5.18 ± 0.07
	1 000	24.0 ± 3.0	6.7 ± 0.8	5.16 ± 0.09	4.86 ± 0.04
	10 000	9.0 ± 2.0	5.12 ± 0.07	4.85 ± 0.05	4.77 ± 0.03
	100 000	5.13 ± 0.07	4.81 ± 0.05	4.78 ± 0.03	4.79 ± 0.02

5.3. Odhad doby přenosů a přenosové rychlosti pro OpenVPN

Schéma pracoviště pro měření OpenVPN protokolu je uvedeno na Obr. 5.7.



Obr. 5.7. Schéma zapojení pro měření parametrů OpenVPN protokolu

Pro realizaci OpenVPN tunelu vytvoříme pomocí utility Easy-RSA vlastní certifikační autoritu (CA) a infrastrukturu veřejných klíčů.

Vygenerujeme privátní a veřejné klíče pro certifikační autoritu (`ca.crt` a `ca.key`), požadavky na vydání certifikátů pro OpenVPN server a OpenVPN klienta.

Vydáme certifikáty pro OpenVPN server (`server.crt` a `server.key`) a OpenVPN klienta (`client.crt` a `client.key`) a podepíšeme je privátním klíčem CA.

Na serveru OpenVPN vytvoříme soubor s klíčem Diffie-Hellman `dh.pem` pro realizaci bezpečného kanálu, ve kterém bude probíhat výměna společného tajného klíče, a statický klíč HMAC `ta.key`.

Obsah konfiguračního souboru `server.conf` pro OpenVPN server je uveden v příloze A v tabulce A.6, obsah konfiguračního souboru `client.conf` pro OpenVPN klienta je uveden v tabulce A.7.

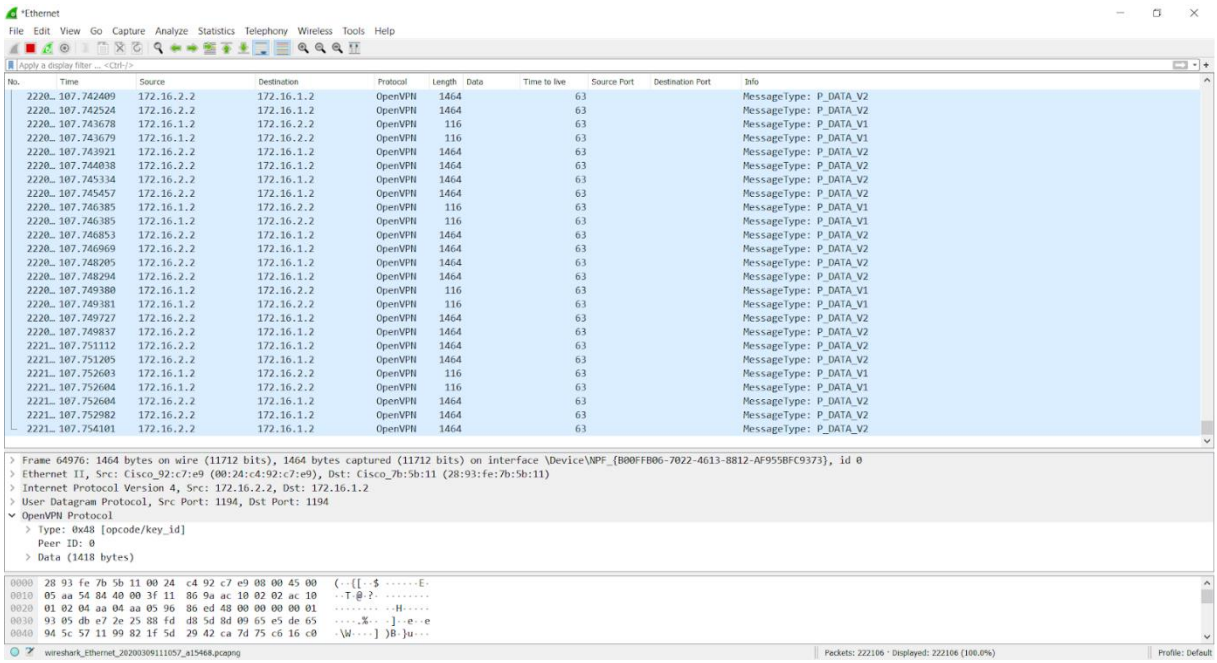
Příkazový řádek pro OpenVPN server:

```
nc -l 1194 > /dev/null
```

Pro OpenVPN klienta:

```
dd if=/dev/urandom bs=<délka bloku dat> count=<počet bloků> | pv
| nc 10.10.0.1 1194
```

Fragment zachyceného OpenVPN provozu je vidět na Obr. 5.8.



Obr. 5.8. Fragment záchytu provozu OpenVPN

Výsledky měření jsou uvedené v Tab. 5.4.

Tab. 5.4

Výsledky měření odhadů doby přenosů a přenosové rychlosti pro OpenVPN protokol

Objem přenesených dat

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	10 kB	100 kB	1 MB	10 MB
	1 000	100 kB	1 MB	10 MB	100 MB
	10 000	1 MB	10 MB	100 MB	1 GB
	100 000	10 MB	100 MB	1 GB	10 GB

Doba přenosu [s]

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	0.005 ± 0.002	0.013 ± 0.004	0.86 ± 0.03	10.9 ± 0.4
	1 000	0.015 ± 0.007	0.85 ± 0.03	10.9 ± 0.4	112.0 ± 4.0
	10 000	0.83 ± 0.04	10.8 ± 0.4	112.0 ± 4.0	1135.0 ± 49.0
	100 000	10.9 ± 0.5	113.0 ± 5.0	1138.0 ± 46.0	11823.0 ± 367.0

Přenosová rychlost [MBps]

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	2.0 ± 1.0	9.0 ± 2.0	1.18 ± 0.03	0.91 ± 0.03
	1 000	8.0 ± 1.0	1.18 ± 0.03	0.92 ± 0.03	0.89 ± 0.03
	10 000	1.22 ± 0.06	0.92 ± 0.03	0.90 ± 0.03	0.88 ± 0.04
	100 000	0.91 ± 0.04	0.89 ± 0.04	0.88 ± 0.03	0.85 ± 0.03

5.4. Odhad doby přenosů a přenosové rychlosti pro WireGuard

Schéma pracoviště pro měření WireGuard protokolu je uvedeno na Obr. 5.9.

Obr. 5.9. Schéma zapojení pro měření parametrů WireGuard protokolu

Pro realizaci WireGuard spojení pomocí nástroje `wg genkey` vygenerujeme privátní a veřejné klíče pro server (`server_private_key` a `server_public_key`) a klienta (`client_private_key` a `client_public_key`). Klíče budou uložené jako soubory s odpovídajícím názvem.

Obsah konfiguračního souboru virtuálního rozhraní `wg0.conf` pro WireGuard server a obsah konfiguračního souboru virtuálního rozhraní `wg0.conf` pro WireGuard klienta jsou uvedené v příloze A v tabulkách A.8 a A.9.

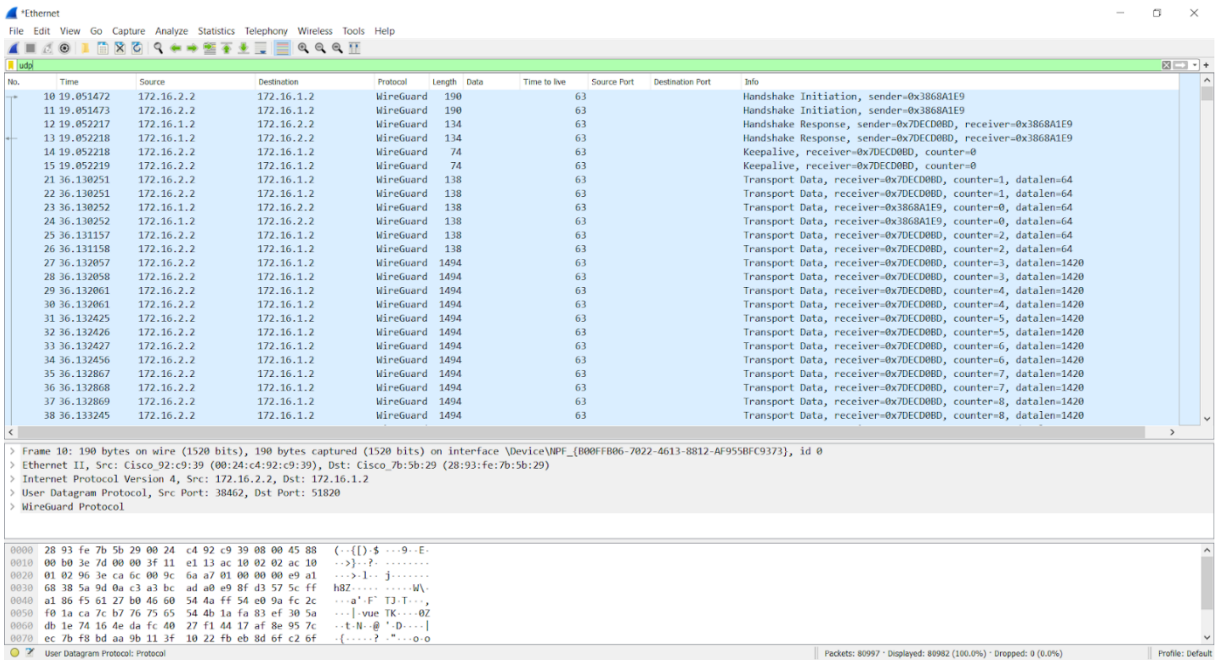
Příkazový řádek pro WireGuard server:

```
nc -l 51820 > /dev/null
```

Pro WireGuard klienta:

```
dd if=/dev/urandom bs=[délka bloku dat] count=[počet bloků] | pv
| nc 172.16.1.2 51820
```

Na Obr. 5.10 je vidět zahájení WireGuard spojení a další provoz.



Obr. 5.10. Fragment záchytu provozu WireGuard

Výsledky měření jsou uvedené v Tab. 5.5.

Tab. 5.5

Výsledky měření odhadů doby přenosů a přenosové rychlosti pro WireGuard protokol

Objem přenesených dat

		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	10 kB	100 kB	1 MB	10 MB
	1 000	100 kB	1 MB	10 MB	100 MB
	10 000	1 MB	10 MB	100 MB	1 GB
	100 000	10 MB	100 MB	1 GB	10 GB

Doba přenosu [s]

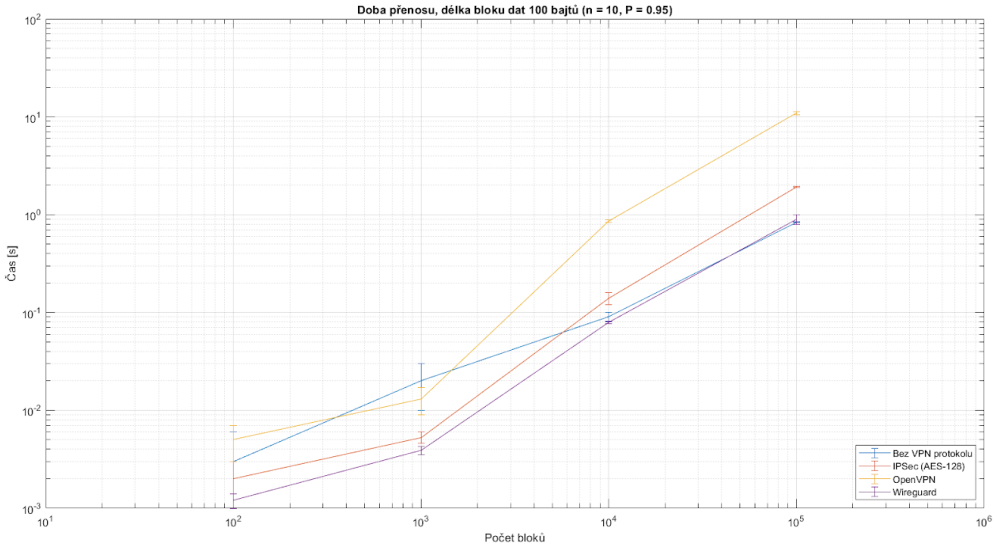
		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	0.0012 ± 0.0002	0.0039 ± 0.0004	0.079 ± 0.002	0.9 ± 0.1
	1 000	0.0033 ± 0.0003	0.082 ± 0.009	0.799 ± 0.006	8.664 ± 0.004
	10 000	0.074 ± 0.001	0.801 ± 0.008	8.662 ± 0.006	88.562 ± 0.007
	100 000	0.802 ± 0.007	8.661 ± 0.003	88.563 ± 0.009	887.55 ± 0.02

Přenosová rychlost [MBps]

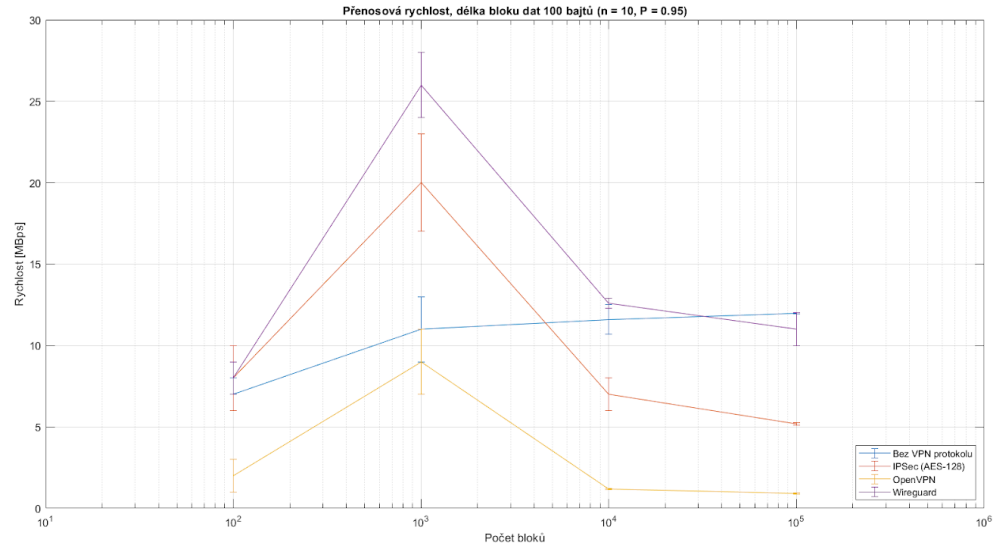
		Počet bloků			
		100	**1 000**	**10 000**	**100 000**
Délka bloku [Bajt]	**100**	8.0 ± 1.0	26.0 ± 2.0	12.6 ± 0.3	11 ± 1.0
	1 000	31.0 ± 3.0	12.0 ± 1.0	12.52 ± 0.09	11.51 ± 0.02
	10 000	13.4 ± 0.2	12.5 ± 0.1	11.52 ± 0.03	11.3 ± 0
	100 000	12.5 ± 0.1	11.53 ± 0.03	11.3 ± 0	11.3 ± 0

5.5. Závěr

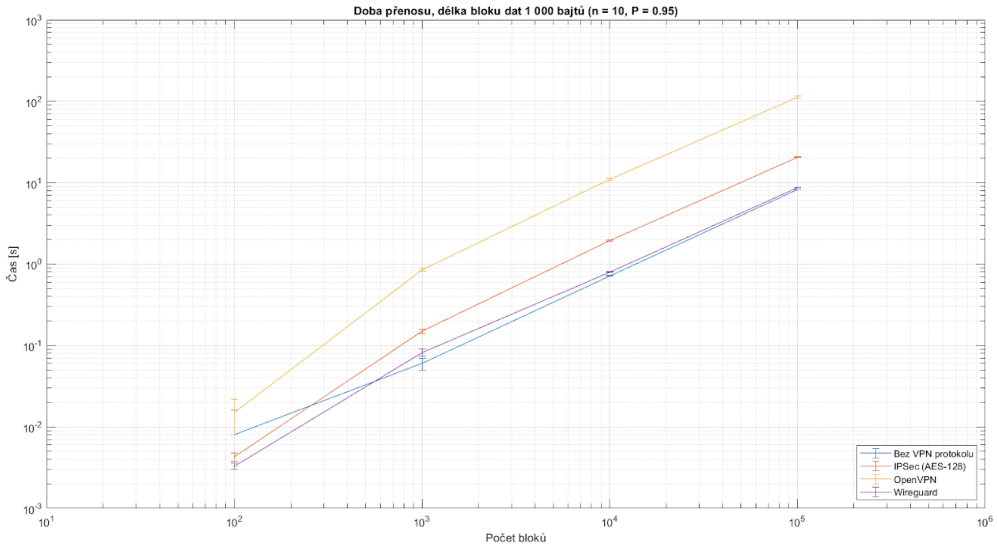
Pro lepší názornost budou výsledky měření pro každou velikost datového bloku zobrazeny na jednom grafu (Obr. 5.11 až 5.18)



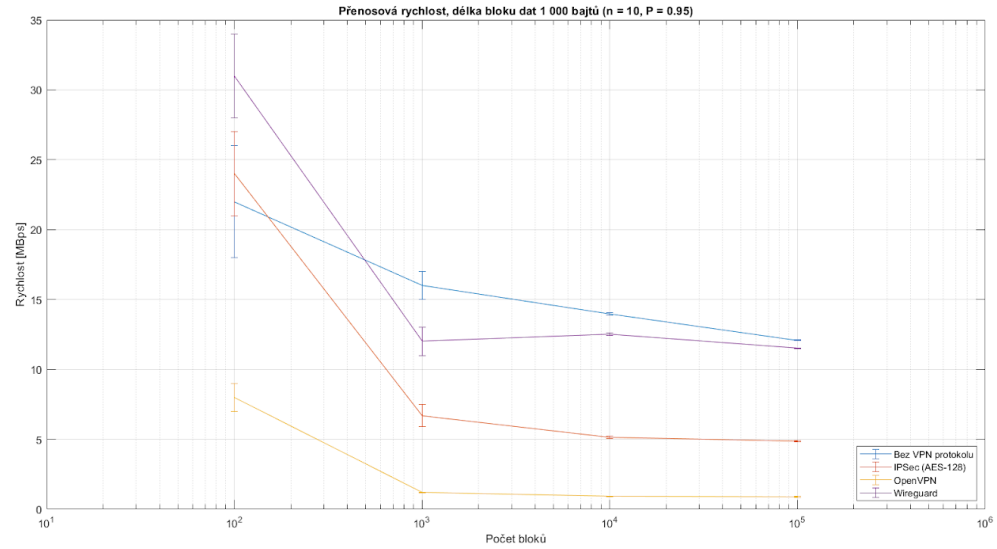
Obr. 5.11. Doba přenosu pro bloky dat velikosti 100 bajtů (menší je lepší)



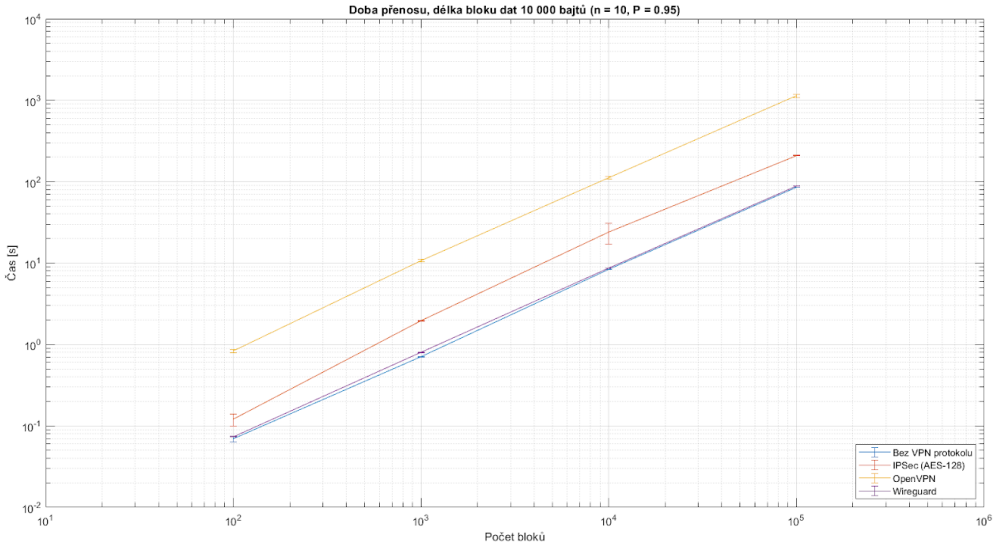
Obr. 5.12. Přenosová rychlost pro bloky dat velikosti 100 bajtů (větší je lepší)



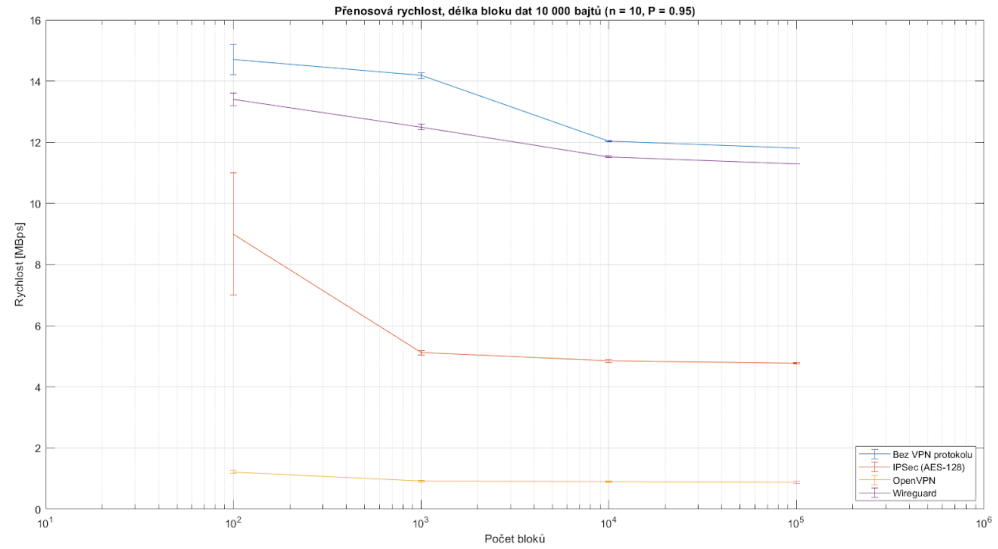
Obr. 5.13. Doba přenosu pro bloky dat velikosti 1 000 bajtů (menší je lepší)



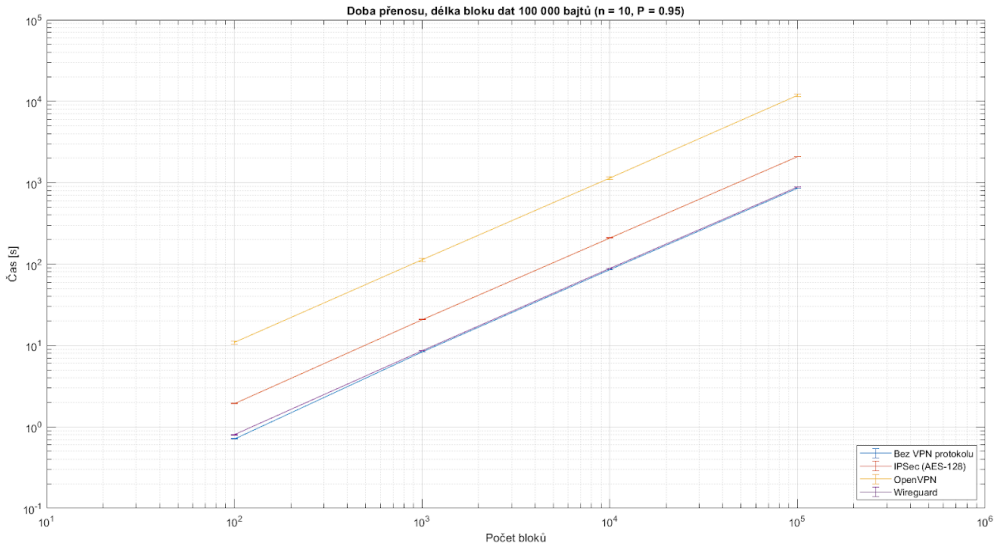
Obr. 5.14. Přenosová rychlost pro bloky dat velikosti 1 000 bajtů (větší je lepší)



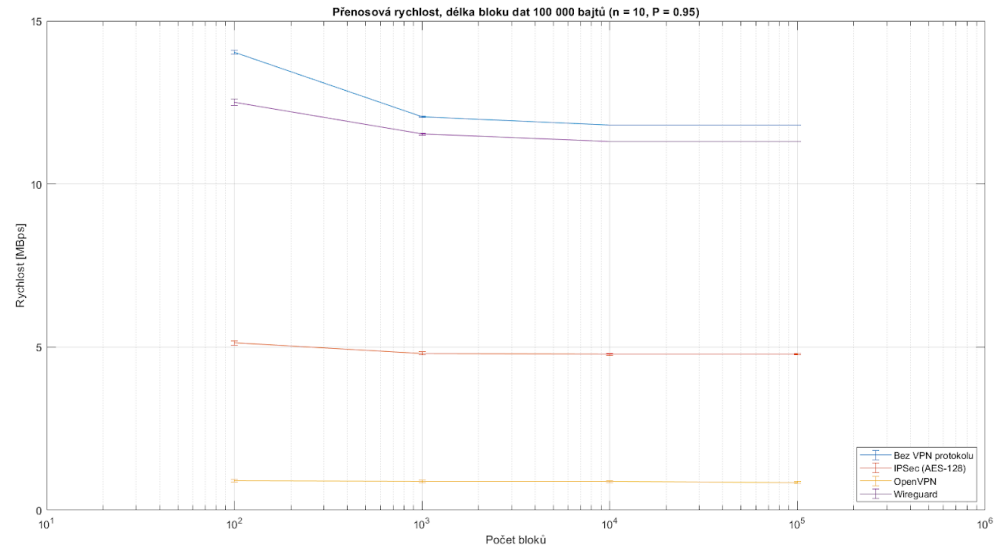
Obr. 5.15. Doba přenosu pro bloky dat velikosti 10 000 bajtů (menší je lepší)



Obr. 5.16. Přenosová rychlost pro bloky dat velikosti 10 000 bajtů (větší je lepší)



Obr. 5.17. Doba přenosu pro bloky dat velikosti 100 000 bajtů (menší je lepší)



Obr. 5.18. Přenosová rychlost pro bloky dat velikosti 100 000 bajtů (větší je lepší)

Zhodnocení naměřených výsledků:

1. Vzhledem k tomu, že pro malé objemy přenesených dat (až 100 kB), dosahuje relativní chyba veličin až 100 %, výsledky těchto kroků brát v úvahu nemůžeme. Dá se předpokládat, že tak velká chyba je vyvolána špatně zvoleným krokem měření.

2. Vzhledem k době přenosu můžeme rozmístit otestované protokoly v následujícím pořadí:
    1) WireGuard (menší doba přenosu, lepší hodnota);
    2) IPSec (AES-128);
    3) OpenVPN (větší doba přenosu, horší hodnota).
3. Vzhledem k přenosové rychlosti můžeme rozmístit otestované protokoly v následujícím pořadí:
    1) WireGuard (větší přenosová rychlost, lepší hodnota);
    2) IPSec (AES-128);
    3) OpenVPN (menší přenosová rychlost, horší hodnota).
4. Dosažené výsledky kvantitativně odpovídají výsledkům dosaženým v [36].
5. Hodnoty doby přenosu a přenosové rychlosti za použití WireGuard protokolu jsou velmi blízké k obdobným hodnotám provozu bez použití VPN protokolů. Tak výrazný rozdíl mezi obdobnými parametry testovaných protokolů můžeme odůvodnit následovně:
    1) Použité algoritmy šifrování. IPSec a OpenVPN ve výchozím nastavení [37] využívají AES, zatímco WireGuard šifruje data pomocí ChaCha20, který je méně časově a výpočetně náročný [38], [39].
    2) WireGuard funguje v jádru operačního systému (zde jde o Linux systémy) na rozdíl od OpenVPN, který běží v prostoru uživatele, což zvětšuje čas komunikace mezi procesy.
    3) Implementace virtuálního rozhraní. OpenVPN rozhraní je realizováno na bázi TUN/TAP, které také funguje v uživatelském prostoru. WireGuard rozhraní je implementováno ve RTNL vrstvě (Routing Netlink) jádra operačního systému.
6. Nastavení klientské a serverové části pro WireGuard je mnohem jednodušší, než pro IPSec a OpenVPN, což snižuje dobu nastavení, pravděpodobnost výskytu chyby v konfiguracích a usnadňuje debugování. Po drobné úpravě konfiguračního souboru serveru se dá použít WireGuard i pro site-to-site topologie [40].

Tím pádem můžeme shrnout, že WireGuard je náležitou alternativou dosavadním VPN protokolům a jejich možnou náhradou v budoucnosti, až bude nějakým způsobem standardizován. Nyní by se však měla při návrhu a realizaci konkrétního projektu brát v úvahu existující infrastruktura a požadavky na zabezpečení informací.

6. Laboratorní úloha

6.1. Cíle měření

Cílem laboratorní úlohy je seznámit se s jedním z moderních protokolů pro virtuální privátní sítě WireGuard, provést základní konfigurace serveru a klienta s využitím vícefaktorové autentizace pro přístup.

6.2. Teoretický úvod [2]

Technologie virtuálních privátních sítí (VPN, anglicky Virtual Private Network) - obecný název pro technologie, které umožňují realizovat jedno nebo více síťových propojení (logická síť) přes jinou síť, nejčastěji přes internet. Ačkoliv je komunikace prováděna v sítích s nižší nebo neznámou důvěryhodností, například ve veřejných sítích, díky použití kryptografické ochrany nezáleží míra důvěryhodnosti vytvořené logické sítě na úrovni důvěryhodnosti v základních sítích.

VPN dává možnost uživatelům se bezpečným způsobem připojit ke vzdálenému serveru pomocí infrastruktury poskytované veřejnou sítí. Pro ten případ je VPN připojení z pohledu uživatele spojením typu bod-bod mezi počítačem uživatele a VPN serverem. Technologie VPN také umožňuje zabezpečené spojení ústředí se svými pobočkami nebo s jinými společnostmi prostřednictvím veřejné sítě. Zde VPN spojení přes internet jedná jako spojení v rozlehlé síti mezi jejími uzly.

V obou případech bude uživatelům v rámcích veřejné sítě poskytováno zabezpečené spojení jako ve vlastní lokální síti, podstata mezilehlé sítě je pro uživatele irelevantní, protože se výměna dat koná prostřednictvím vyhrazeného privátního spojení.

Koncept VPN dále umožňuje efektivně řešit současné trendy v oblasti práce jako je možnost bezpečného vzdáleného připojení do korporátní sítě.

Protokol WireGuard, který byl oficiálně zveřejněn v roce 2018, je navržen Jasonem Donefeldem, zakladatelem společnosti Edge Security LLC, jako jednodušší a bezpečnější alternativa vůči stávajícím protokolům IPsec či OpenVPN. Od konce března 2020 je WireGuard oficiálně součástí jádra Linux (verze 5.6 či novější) [41].

[2] Tato podkapitola je kompilace z kapitol 2 a 3 dané diplomové práce.

WireGuard je L3 tunelovací protokol s využitím UDP v transportní vrstvě. Jsou podporovány protokoly IPv4 a IPv6. Při změně IP adresy VPN serveru bude nastavení klienta automaticky upraveno bez přerušení spojení.

Pro šifrování se používá proudová šifra ChaCha20 [21] a pro autentizaci a zjištění integrity zpráv MAC (Message Authentication Code) realizovaný algoritmem Poly1305. Vygenerování společného klíče se provádí pomocí Diffie-Hellmanova protokolu nad eliptickými křivkami s využitím Bernstainovy křivky Curve25519 [22]. Pro hashování jsou použity kryptografické hashovací funkce BLAKE2s [23] a SipHash-24 [24]. Funkce pro odvození klíče je realizována pomocí HKDF (Hashed Key Derivation Function) [25]. Časová značka TAI64N umožňuje zabezpečení ochrany před reply útoky.

Tento protokol je postaven na základě konceptu zvaného Cryptokey Routing:

- WireGuard vytváří rozhraní s názvem wg, kterému se přiřadí IP adresa, port na kterém bude poslouchat, veřejný a privátní klíč. Každé wg rozhraní v jeho konfiguračním souboru má informace o peerech – jejich veřejných klíčích a IP adresách (AllowedIPs). Tato informace hraje roli Cryptokey Routing Table. Během komunikace se do této informace přidávají údaje o fyzické IP adrese peeru (Endpoint), pokud nebyly zadány předem.
- Když se odchozí paket dostane na rozhraní wg, na základě IP adresy z tabulky bude zvolen peer, z jehož veřejného klíče bude odvozen klíč pro šifrování odchozího paketu.
- Když se příchozí paket dostane na rozhraní wg, bude dešifrován privátním klíčem příjemce, identifikován na základě veřejného klíče peera (pokud je takový zapsán v tabulce) a zdrojová IP adresa bude srovnána se záznamem v tabulce, jestli tomu záznamu odpovídá.
- V případě jakéhokoliv nesouladu bude paket vyhozen. Pro realizaci tunelu musí server dostat alespoň jeden správně šifrovaný paket.

V roce 2018 byla Ludvigem Strigeusem realizována VPN služba TunSafe [42], která poskytuje uživatelům jiných operačních systémů, než Linux, možnost využití WireGuardu. Momentálně existují TunSafe aplikace pro Windows 7, 8, 10, Linux, MacOS, FreeBSD a Android. Ve verzi 1.5-rc2 pro Windows je dodána možnost vícefaktorové autentizace pomocí jednorázového hesla.

Na rozdíl od "čistého" WireGuardu pro implementaci virtuálního rozhraní TunSafe, jako OpenVPN, používá TUN/TAP adaptéry, což omezuje využití výhod WireGuard protokolu v plné míře.

6.3. Základní konfigurace a nastavení

6.3.1. Vygenerování privátního a veřejného klíče pro server a klienta.

Vygenerovat klíče můžeme dvěma způsoby:

Pomocí utility `wg` a příkazu `genkey` vytváříme privátní klíč:

```
$ wg genkey > [název souboru s privátním klíčem]
```

Výsledkem bude textový soubor v domácí složce (`/home/user`).

Potom pomocí příkazu `pubkey` z privátního klíče odvodíme veřejný klíč:

```
$ wg pubkey < [název souboru s privátním klíčem] > [název souboru s veřejným klíčem]
```

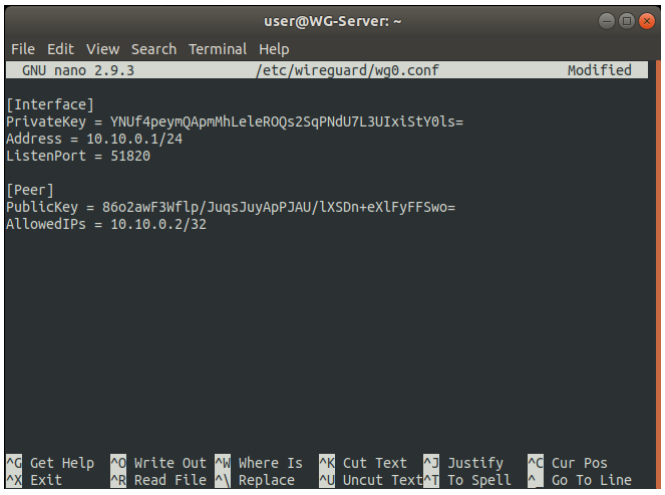
Výsledkem opět bude textový soubor v domácí složce.

V jednom příkazovém řádku

```
$ wg genkey | tee [název souboru s privátním klíčem] | wg pubkey > [název souboru s veřejným klíčem]
```

6.3.2. Konfigurační soubor pro server.

Konfigurační soubor pro server (Obr. 6.1) musí obsahovat informaci o virtuálním rozhraní VPN serveru:



Obr. 6.1. Ukázkový obsah konfiguračního souboru pro server

```
[Interface]
PrivateKey = [privátní klíč serveru z odpovídajícího souboru]
Address = [IP adresa serveru ve virtuální privátní síti/maska]
ListenPort = [číslo portu pro server]
```

A informace o peerech (klientech):

```
[Peer]
PublicKey = [veřejný klíč klienta z odpovídajícího souboru]
AllowedIPs = [IP adresa klienta ve virtuální privátní síti/maska]
```

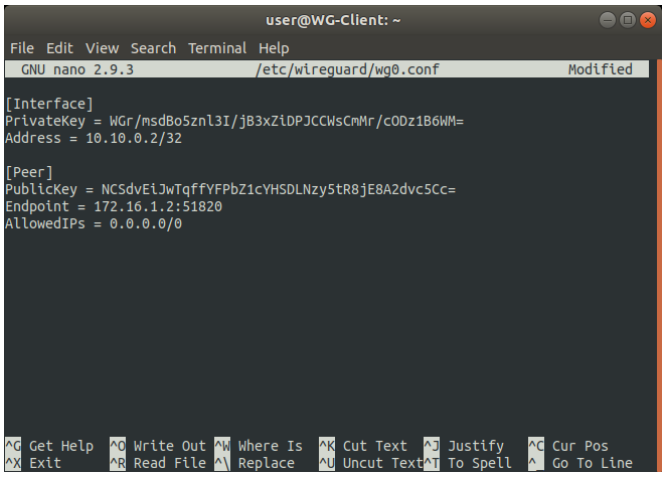
6.3.3. Konfigurační soubor pro klienta.

Konfigurační soubor pro klienta (Obr. 6.2) má obdobnou strukturu – nastavení pro virtuální rozhraní:

```
[Interface]
PrivateKey = [privátní klíč klienta z odpovídajícího souboru]
Address = [IP adresa klienta ve virtuální privátní síti/maska]
```

Informace o peeru (serveru)

```
[Peer]
PublicKey = [veřejný klíč serveru z odpovídajícího souboru]
Endpoint = [IP adresa serveru:číslo portu]
AllowedIPs = [Cílové IP adresy , které budou dosazené přes tunel mezi klientem a serverem/maska]
```



Obr. 6.2. Ukázkový obsah konfiguračního souboru pro klienta

6.3.4. Spuštění WireGuard serveru a klienta.

Konfigurační soubor musí být uložen ve složce `/etc/WireGuard`.

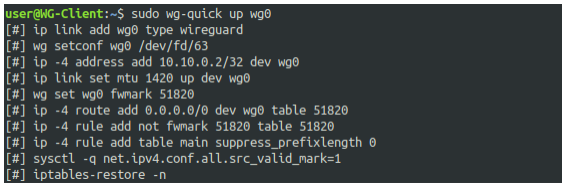
Spustit WireGuard server a klient můžeme na příslušných počítačích příkazem

`$ sudo wg-quick up wg0` , kde `wg0` je název odpovídajícího konfiguračního souboru.

V terminálovém okně uvidíme jako výsledek pro server

Obr. 6.3. Nastartování WireGuard serveru

a pro klienta

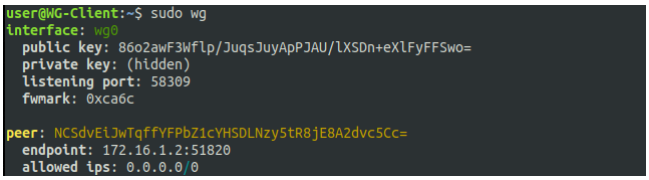


Obr. 6.4. Nastartování WireGuard klienta

Příkazem `$ sudo wg` máme možnost ověřit stav virtuálního WireGuard rozhraní serveru

Obr. 6.5. Stav serverového WireGuard rozhraní

a klienta



Obr. 6.6. Stav klientského WireGuard rozhraní

6.3.5. Konfigurační soubory pro TunSafe server a klienta.

TunSafe používá stejnou syntaxi jako WireGuard, a proto můžeme využít již připravené konfigurační soubory.

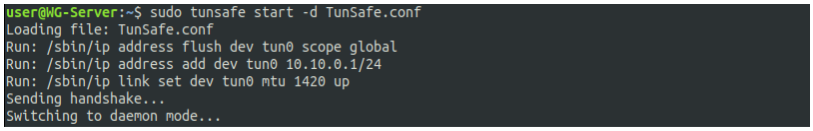
6.3.6. Spuštění TunSafe serveru.

Otevřeme terminál ve složce, kde je uložen konfigurační soubor. TunSafe server jako démon spustíme příkazem

```
$ sudo tunsafe start -d TunSafe.conf
```

kde `TunSafe.conf` je konfigurační soubor pro server.

V terminálovém okně uvidíme jako výsledek



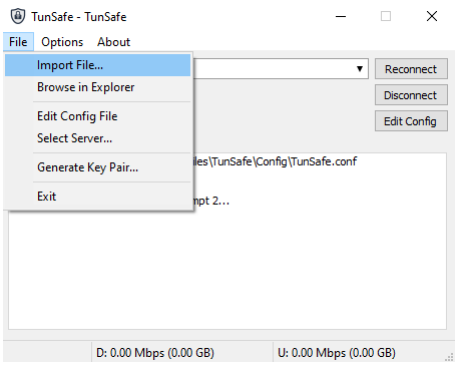
Obr. 6.7. Nastartování TunSafe serveru

Příkazem `tunsafe show` se ujistíme, že server běží.

Obr. 6.8. Stav serverového TunSafe rozhraní

6.3.7. Importování konfiguračního souboru do TunSafe klienta ve Windowsu.

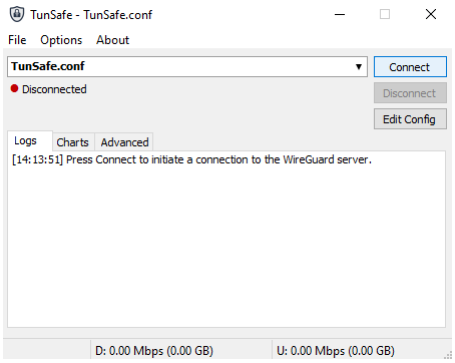
Spustíme aplikaci TunSafe. V nabídce **File** zvolíme položku **Import File** a vybereme konfigurační soubor, který jsme připravili v předchozím kroku.



Obr. 6.9. Importování konfiguračního souboru do TunSafe klienta ve Windowsu

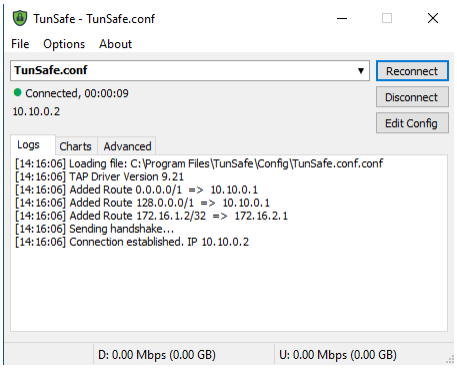
6.3.8. Připojení TunSafe klienta k serveru

Po importu klientské konfigurace stiskneme tlačítko **Connect**



Obr. 6.10. Připojení TunSafe klienta k serveru

Máme-li všechna nastavení správná, změní se stav spojení na Connected



Obr. 6.11. Stav Connected (připojen) TunSafe klienta

6.3.9. Realizace vícefaktorové autentizace s jednorázovým heslem na základě času (TOTP)

Pro realizaci TOTP je nutné v konfiguraci TunSafe serveru přidat do nastavení peeru následující řádek:

`RequireToken = totp-sha1:[32 symbolová alfanumerická posloupnost],digits=[počet číslic v hesle],period=[s jakou periodou se bude heslo měnit],precision=[kolik vteřin bude heslo platné po jeho změně]`

Nainstalovat na chytrý telefon aplikaci FreeOTP a zvolit ikonku **Přidat klíč**.

V okénku zadáme název pro jednorázové heslo a nějaký komentář.

V položce Secret napíšeme stejnou posloupnost jako v nastavení peeru z předchozího kroku.

Zvolíme typ jednorázového hesla. Pro nás TOTP.

Zvolíme počet číslic v jednorázovém hesle, který musí být stejný s počtem v parametru `digits` v předchozím kroku.

Vybereme typ hashovacího algoritmu. Pro náš případ SHA-1.

Zadáme časový interval, po kterém se bude heslo měnit, stejný jako hodnota parametru `period` v předchozím kroku.

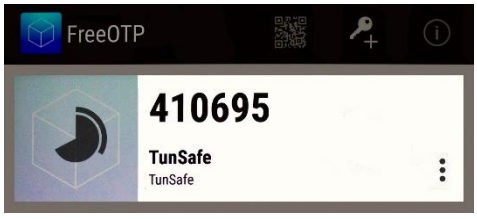
Stisknutím tlačítka **Add** uložíme generátor hesel.

Obr. 6.12. Přidání generátoru jednorázového hesla v aplikaci FreeOTP

Restartujeme server a v klientské aplikaci znovu stiskneme tlačítko **Connect**.

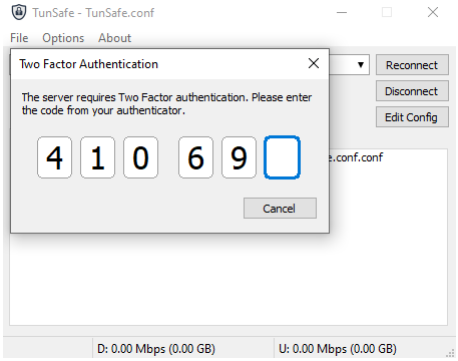
Objeví se okénko s prázdnými položkami pro jednorázové heslo.

Spustíme dříve nakonfigurovaný generátor hesel v aplikaci FreeOTP. Vygeneruje se šestimístné heslo,



Obr. 6.13. Vygenerované jednorázové heslo v aplikaci FreeOTP

které zadáme do okénka v klientské aplikaci.

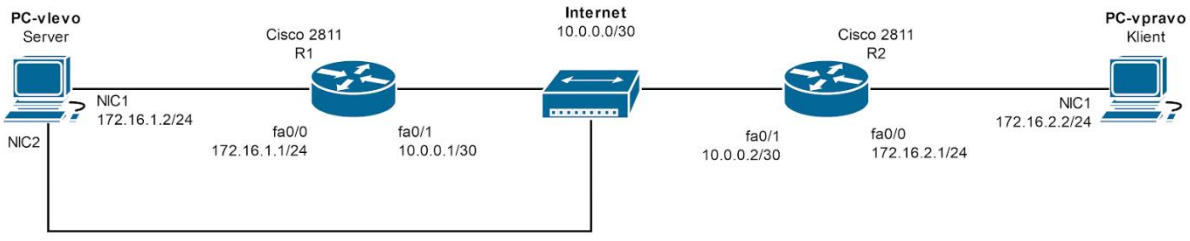


Obr. 6.14. Zadání jednorázového hesla v klientské aplikaci TunSafe

Máme-li všechna nastavení správná, dojde k dokončení spojení a jeho stav se změní na Connected.

6.4. Schéma topologie úlohy

VPN server a klient jsou implementované na virtuálních počítačích, které mají připojení k síti v režimu síťového mostu (Bridged adapter).



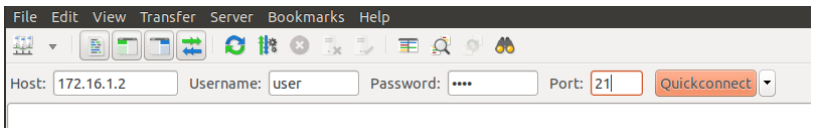
Obr. 6.15. Schéma laboratorní úlohy

6.5. Postup

6.5.1. Vyzkoušení WireGuard protokolu

1. Zapojte pracoviště podle schématu na Obr. 6.15. Mezi směrovače R1 a R2 zapojte rozbočovač (hub).
2. Na PC-vlevo spusťte virtuální stroj WG-WinClient. Přihlaste se jako uživatel `user` s heslem `user`.
3. Na uvedeném virtuálním počítači spusťte program PUTTY a připojte se na rozhraní Console směrovačů.
4. Nastavte IP adresy rozhraní a statické směrování mezi směrovače.

5. Na PC-vlevo na virtuálním stroji WG-WinClient spusťte program Wireshark, a zvolte příslušné rozhraní pro záchyt provozu.
6. Na PC-vlevo ve VirtualBoxu spusťte VPN server (WG-Server). Na PC-vpravo ve VirtualBoxu spusťte VPN klienta (WG-Client). Přihlašte se jako uživatel `user` s heslem `user`.
7. Příkazem `ping` se ujistěte, že jsou virtuální stroje navzájem přístupné a konfigurace topologie je správná.
8. Na virtuálním stroji WG-Client příkazem `$ filezilla` spusťte FTP klienta. V příslušných položkách zadejte IP adresu, jméno uživatele, heslo a číslo portu pro přístup k FTP serveru, který běží na WG-Server.



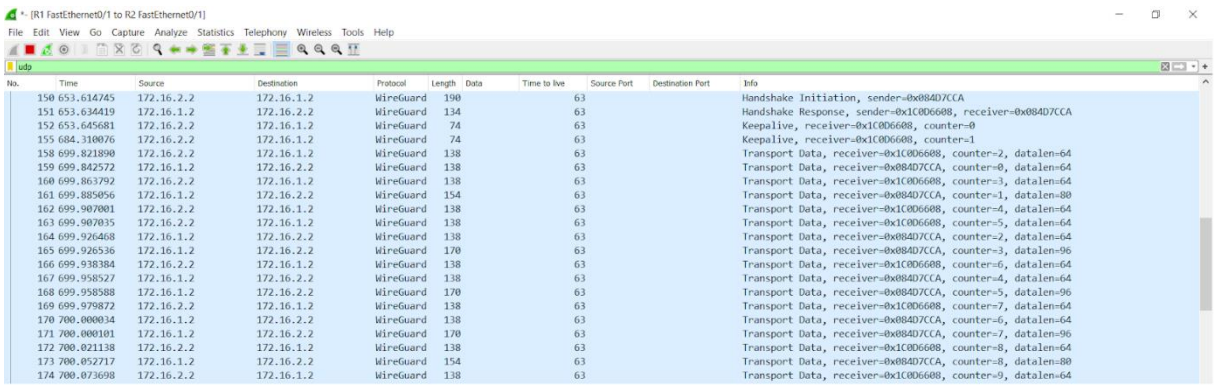
Obr. 6.16. Zadání parametrů FTP klienta

9. Stisknutím tlačítka **Quickconnect** se pokuste připojit k FTP serveru.
10. V aplikaci Wireshark klikněte pravým tlačítkem myši na FTP paket a zvolte položku **Follow** pak **TCP Stream**. V okně, které se otevře, je vidět komunikaci klienta a serveru a hlavně – jméno uživatele a heslo v otevřené podobě (Obr. 6.17).
11. Dle návodu vygenerujte veřejné a privátní klíče pro WireGuard server a klienta. Vytvořte konfigurační soubory, které umístěte do složky `/etc/WireGuard`.
12. Spusťte WireGuard server a klienta. Ověřte stav WireGuard rozhraní na obou virtuálních strojích.
13. Opakujte kroky 8 až 10 s ohledem na protokol, který WireGuard využívá v transportní vrstvě. Poznamenejte, jak se změnil zachycený provoz (Obr. 6.18) a jeho obsah (Obr. 6.19). Určete fázi navázání spojení.

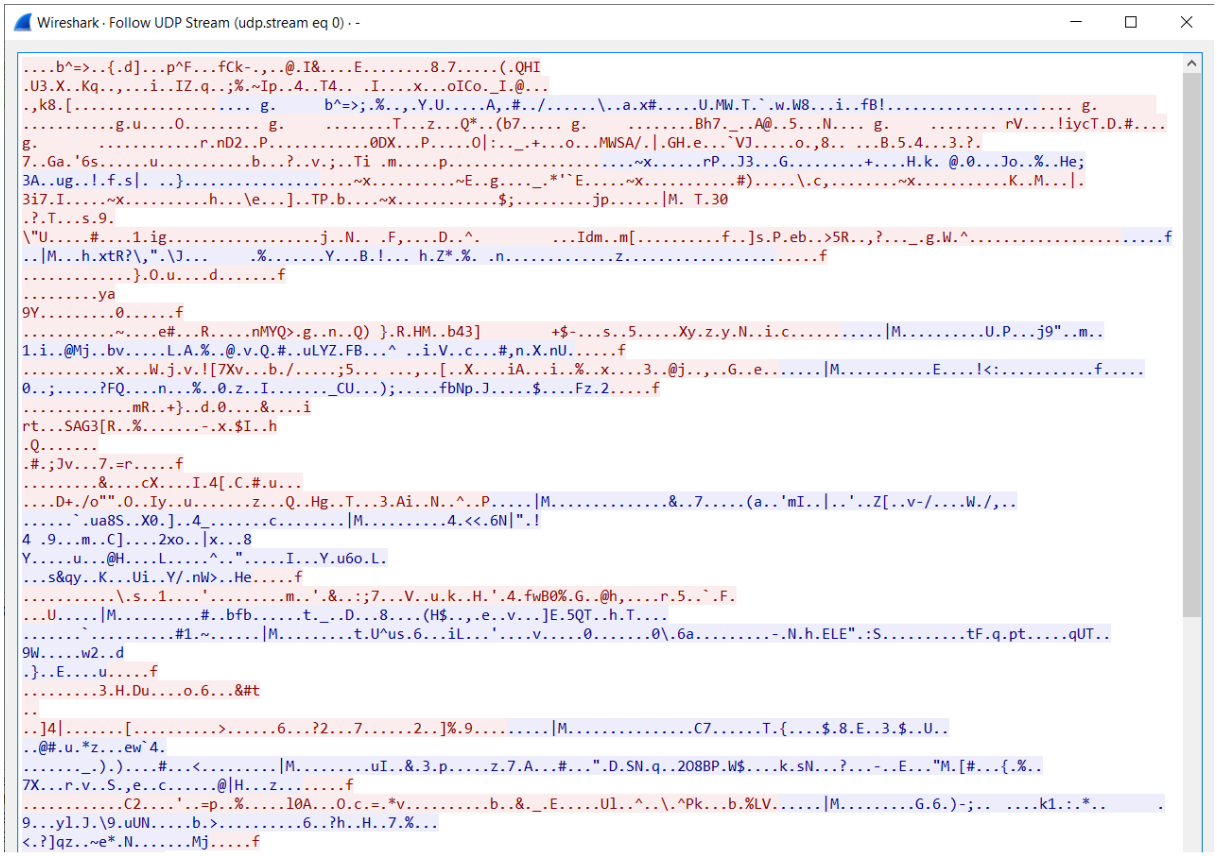
Wireshark · Follow TCP Stream (tcp.stream eq 0) · -

```
220 Welcome to FTP server!
AUTH TLS
530 Please login with USER and PASS.
AUTH SSL
530 Please login with USER and PASS.
USER user
331 Please specify the password.
PASS user
230 Login successful.
SYST
215 UNIX Type: L8
FEAT
211-Features:
 EPRT
 EPSV
 MDTM
 PASV
 REST STREAM
 SIZE
 TVFS
211 End
PWD
257 "/home/user" is the current directory
TYPE I
200 Switching to Binary mode.
PASV
227 Entering Passive Mode (172,16,1,2,51,179).
LIST
150 Here comes the directory listing.
226 Directory send OK.
MDTM TunSafe.conf
213 20200413221939
```

Obr. 6.17. FTP komunikace v otevřené podobě



Obr. 6.18. Fragment WireGuard provozu



Obr. 6.19. Obsah WireGuard provozu

6.5.2. Vyzkoušení vícefaktorové autentizace

14. Na PC-vpravo můžete vypnout WG-Client a nastartujte virtuální stroj WG-WinClient. Na WG-Client zastavte WireGuard server.
15. Dle teoretického návodu podle potřeby modifikujte konfigurační soubory pro TunSafe server a klienta. Konfigurace pro TunSafe server by měla obsahovat nastavení pro dvoufaktorovou autentizaci.
16. Na virtuálním počítači WG-WinClient spusťte klientskou aplikaci TunSafe a importujte do ní konfigurační soubor.
17. V aplikaci FreeOTP v chytrém telefonu nakonfigurujte generátor jednorázového hesla.
18. Na WG-Server spusťte TunSafe server a ověřte stav jeho rozhraní.

```
user@WG-Server:~$ sudo tunsafe show
interface: tun0
  public key: NCSdvEiJwTqffYFPbZ1cYHSDLNzy5tR8jE8A2dvc5Cc=
  private key: (hidden)
  listening port: 51820
  address: 10.10.0.1/24

peer: 86o2awF3Wflp/JuqsJuyApPJAU/lXSDn+eXlFyFFSwo=
  allowed ips: 10.10.0.2/24
```

Obr. 6.20. Stav rozhraní TunSafe serveru po spuštění

19. Zkuste připojit TunSafe klient k serveru. Ujistěte se, že ověření uživatele probíhá pomocí vícefaktorové autentizace. Poznamenejte ve Wiresharku fázi navázání spojení.
20. Znovu ověřte stav TunSafe serveru. Co se změnilo?

Obr. 6.21. Stav rozhraní TunSafe serveru po připojení klienta

21. Smažte soubory s klíči a konfigurační soubory. Pracoviště uveďte do původního stavu.

6.6. Použitá zařízení a nástroje:

- Virtualizační nástroj VirtualBox
- Ubuntu 18.04
- Windows 10
- WireGuard
- TunSafe
- Wireshark
- TunSafe Client for Windows 1.5-rc2
- FileZilla Client 3.47.2.1
- FreeOTP

6.7. Seznam použitých příkazů

Tab. 6.1

Seznam použitých příkazů

`wg genkey > [název souboru s privátním klíčem]`	Vygeneruje privátní klíč
`wg pubkey < [název souboru s privátním klíčem] > [název souboru s veřejným klíčem]`	Odvodí veřejný klíč z privátního
`wg-quick up [název konfiguračního souboru]`	Zapne virtuální WireGuard rozhraní s názvem konfiguračního souboru

Tab. 6.1 (dokončení)

`wg-quick down wg0`	Vypne virtuální WireGuard rozhraní s názvem konfiguračního souboru
`Wg`	Ukáže stav virtuálního WireGuard rozhraní
`tunsafe start -d [název konfiguračního souboru.conf]`	Zapne virtuální TunSafe rozhraní
`tunsafe stop [název virtuálního rozhraní]`	Vypne virtuální TunSafe rozhraní
`tunsafe show`	Ukáže stav virtuálního TunSafe rozhraní
`filezilla`	Spustí FTP klienta

6.8. Zdroje pro domácí přípravu

- Oficiální stránky WireGuard https://www.WireGuard.com/
- Oficiální stránky TunSafe https://tunsafe.com/
- Stránka FreeOTP https://freeotp.github.io/
- RFC 6238. TOTP: Time-Based One-Time Password Algorithm ttps://tools.ietf.org/html/rfc6238

7. Závěr

Cílem diplomové práce byla analýza vybraných protokolů pro virtuální privátní sítě, praktické měření jejich základních charakteristik, srovnání výsledků a návrh laboratorní úlohy pro studenty. Jako primární kritéria hodnocení byly zvoleny doba přenosu vybraného objemu dat a přenosová rychlost, které jsou důležité z pohledu uživatelů. Za tímto účelem byly zvolené následující protokoly:

1. IPSec
2. OpenVPN
3. WireGuard

Výběr protokolů IPSec a OpenVPN byl způsoben jejich častým využitím v dnešní době a dále také možnosti provést měření ve stávajících laboratorních podmínkách s minimálními náklady.

Měření bylo provedeno v laboratoři Katedry telekomunikační techniky. Topologie byly implementovány pomocí zařízení Cisco a virtuálních strojů na bázi VirtualBox. Jako software byly použity volně dostupné programy s otevřeným zdrojovým kódem a operační systém Ubuntu 18.04 LTS.

Z provedených praktických měření vyplývá, že WireGuard protokol zajišťuje lepší dobu přenosu a přenosovou rychlost, srovnatelnou s obdobnými parametry bez použití VPN protokolu. Zároveň nastavení klientské a serverové části pro Wireguard je mnohem jednodušší, než pro IPSec a OpenVPN. Z teoretického průzkumu plyne, že WireGuard pro svou funkčnost využívá algoritmy méně časově a výpočetně náročné v porovnání s IPSec a OpenVPN.

Na základě teoretické analýzy a praktických měření byl zvolen protokol WireGuard jako nejlepší kandidát pro realizaci výukové laboratorní úlohy, která seznámí studenty se základními principy konfigurace a nastavením parametrů VPN spojení a dalšími opatřeními na ochranu přístupů, jako je dvoufaktorová autentizace.

Seznam zdrojů

[1] Microsoft Docs. *Virtual Private Networking: An Overview* [online]. Microsoft, 2019. [cit. říjen 2019]. Dostupné z: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/bb742566(v=technet.10)?redirectedfrom=MSDN

[2] RFC 2764. *A Framework for IP Based Virtual Private Networks* [online]. Únor 2000. [cit. říjen 2019]. Dostupné z: https://www.ietf.org/rfc/rfc2764.txt

[3] 3.1. What IPsec Does. In: RFC 2401. *Security Architecture for the Internet Protocol* [online]. Listopad 1998. [cit. říjen 2019]. Dostupné z: https://tools.ietf.org/html/rfc2401#section-3.1

[4] RFC 4302. *IP Authentication Header* [online]. Prosinec 2005. [cit. říjen 2019]. Dostupné z: https://tools.ietf.org/html/rfc4302

[5] RFC 8221. *Cryptographic Algorithm Implementation Requirements and Usage Guidance* [online]. Říjen 2017. [cit. říjen 2019]. Dostupné z: https://tools.ietf.org/html/rfc8221

[6] RFC 2409. *The Internet Key Exchange (IKE)* [online]. Listopad 1998. [cit. říjen 2019]. Dostupné z: https://tools.ietf.org/html/rfc2409

[7] RFC 4109. *Algorithms for Internet Key Exchange version 1 (IKEv1)* [online]. Květen 2005. [cit. říjen 2019]. Dostupné z: https://tools.ietf.org/html/rfc4109

[8] 3.1.2. Tunnel Mode. In: RFC 4302. *IP Authentication Header* [online]. Prosinec 2005. [cit. říjen 2019]. Dostupné z: https://tools.ietf.org/html/rfc4302#section-3.1.2

[9] 3.1.2. Tunnel Mode Processing. In: RFC 4303. *IP Encapsulating Security Payload (ESP)* [online]. Prosinec 2005. [cit. říjen 2019]. Dostupné z: https://www.ietf.org/rfc/rfc4303.txt

[10] *IPsec/IKEv2-based VPN software for Linux. What are differences between IKEv1 and IKEv2? (IKEv1 vs. IKEv2)*. In: Rockhopper VPN. [online]. T. HANADA, 2011. [cit. říjen 2019]. Dostupné z: http://rockhoppervpn.sourceforge.net/techdoc_ikev1vsikev2.html

[11] RFC 7296. *Internet Key Exchange Protocol Version 2 (IKEv2)* [online]. Říjen 2014. [cit. říjen 2019]. Dostupné z: https://tools.ietf.org/html/rfc7296

[12] RFC 3193. *Securing L2TP using IPSec* [online]. Listopad 2001. [cit. říjen 2019]. Dostupné z: https://tools.ietf.org/html/rfc3193

[13] *Why does OpenVPN use UDP and TCP?* In: Oficiální stránky OpenVPN [online]. OPENVPN INC, 2019. [cit. listopad 2019]. Dostupné z: https://openvpn.net/faq/why-does-openvpn-use-udp-and-tcp/

[14] *Hardening OpenVPN Security.* In: Oficiální stránky OpenVPN [online]. OPENVPN INC, 2019. [cit. listopad 2019]. Dostupné z: https://openvpn.net/community-resources/hardening-openvpn-security/

[15] NIST Special Publication 800-113. *Guide to SSL VPNs* [online]. Červenec 2008. [cit. listopad 2019]. Dostupné z: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-113.pdf

[16] Oficiální stránky WireGuard [online]. Jason A. Donenfeld, 2019. [cit. listopad 2019]. Dostupné z: https://www.wireguard.com/

[17] Sven Taylor. *WireGuard: What You Need to Know* [online]. Restore Privacy, LLC, 28. června 2019. [cit. listopad 2019]. Dostupné z: https://restoreprivacy.com/wireguard/

[18] Lucian Constantin. *What is WireGuard? Secure, simple VPN now part of Linux* [online]. IDG Communications, Inc., 2. dubna 2020. [cit. duben 2020]. Dostupné z: https://www.csoonline.com/article/3434788/what-is-wireguard-secure-simple-vpn-still-in-development.html

[19] Ubuntu Manuals. *udptunnel - Tunnel UDP packets over a TCP connection* [online]. Canonical Ltd, 2019. [cit. listopad 2019]. Dostupné z: http://manpages.ubuntu.com/manpages/precise/en/man1/udptunnel.1.html#name

[20] strigeus. *WireGuard over TCP.* In: GitHub [online]. 16. prosince 2018. [cit. listopad 2019]. Dostupné z: https://github.com/TunSafe/TunSafe/blob/master/docs/wireguard%20TCP.txt

[21] RFC 7539. *ChaCha20 and Poly1305 for IETF Protocols* [online]. Květen 2015. [cit. listopad 2019]. Dostupné z: https://tools.ietf.org/html/rfc7539

[22] 6.1. Curve25519 In: RFC 7748. *Elliptic Curves for Security.* [online]. Leden 2016. [cit. listopad 2019]. Dostupné z: https://tools.ietf.org/html/rfc7748#section-6.1

[23] RFC 7693. *The BLAKE2 Cryptographic Hash and Message Authentication Code (MAC)* [online]. Listopad 2015. [cit. listopad 2019]. Dostupné z: https://tools.ietf.org/html/rfc7693

[24] Jean-Philippe Aumasson and Daniel J. Bernstein. *SipHash: a fast short-input PRF.* In: DIAC - Directions in Authenticated Ciphers Workshop. July 05-06, 2012, Stockholm, Sweden [online]. [cit. listopad 2019]. Dostupné z: https://131002.net/siphash/siphash.pdf

[25] RFC 5869. *HMAC-based Extract-and-Expand Key Derivation Function (HKDF)* [online]. Květen 2010. [cit. listopad 2019]. Dostupné z: https://tools.ietf.org/html/rfc5869

[26] Oficiální stránky SoftEther VPN Project [online]. SoftEther Project, 2019. [cit. listopad 2019]. Dostupné z: https://www.softether.org/

[27] Microsoft Docs. *[MS-SSTP]: Secure Socket Tunneling Protocol (SSTP)* [online]. Microsoft, 2019. [cit. listopad 2019]. Dostupné z: https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-sstp/c50ed240-56f3-4309-8e0c-1644898f0ea8

[28] Oficiální stránky tinc [online]. 26. srpna 2019. [cit. listopad 2019]. Dostupné z: https://www.tinc-vpn.org/

[29] Oficiální stránky Freelan [online]. Prosinec 2017. [cit. listopad 2019]. Dostupné z: https://www.freelan.org/

[30] Marvin the Robot. *Cybercriminals intercept codes used for banking to empty your accounts* [online]. AO Kaspersky Lab, 1. února 2019. [cit. listopad 2019]. Dostupné z: https://www.kaspersky.com/blog/ss7-hacked/25529/

[31] Obrázek. In: *How This Single Little Token Allows You Perform Transactions Across All Your Accounts in Nigerian Banks* [online]. 19. září 2018. [cit. listopad 2019]. Dostupné z: https://medium.com/parkway-projects/how-this-single-little-token-allows-you-perform-transactions-across-all-your-accounts-in-nigerian-52fefced30d9

[32] Obrázek. In: Christopher Cuttriss. *How Do I: Set up 2-Factor Authentication* [online]. 14. března 2020. [cit. duben 2020]. Dostupné z: https://cait.calarts.edu/hc/en-us/articles/217057228-How-Do-I-Set-up-2-Factor-Authentication

[33] RFC 4226. *HOTP: An HMAC-Based One-Time Password Algorithm* [online]. Prosinec 2005. [cit. listopad 2019]. Dostupné z: https://tools.ietf.org/html/rfc4226

[34] RFC 6238. *TOTP: Time-Based One-Time Password Algorithm* [online]. Květen 2011. [cit. listopad 2019]. Dostupné z: https://tools.ietf.org/html/rfc6238

[35] NIST Special Publication 800-63. *Digital Identity Guidelines* [online]. 22. června 2017. [cit. listopad 2019]. Dostupné z: https://pages.nist.gov/800-63-3/

[36] Benchmarking. In: Oficiální stránky WireGuard [online]. Jason A. Donenfeld, 2020. [cit. březen 2020]. Dostupné z: https://www.wireguard.com/performance/

[37] Change encryption cipher in Access Server. In: Oficiální stránky OpenVPN [online]. OPENVPN INC, 2020. [cit. březen 2020]. Dostupné z: https://openvpn.net/vpn-server-resources/change-encryption-cipher-in-access-server/

[38] *TLS Symmetric Crypto.* In: Adam Langley's Weblog [online]. 27. února 2014. [cit. březen 2020]. Dostupné z: https://www.imperialviolet.org/2014/02/27/tlssymmetriccrypto.html

[39] Nick Sullivan. *Do the ChaCha: better mobile performance with cryptography.* In: The Cloudflare Blog [online]. 23. února 201ř. [cit. březen 2020]. Dostupné z: https://blog.cloudflare.com/do-the-chacha-better-mobile-performance-with-cryptography/

[40] mjtechguy. *Wireguard site-to-site (network-to-network) VPN Configuration examples.* In: GitHub [online]. 2020. [cit. březen 2020]. Dostupné z: https://github.com/mjtechguy/wireguard-site-to-site

[41] Jim Salter. *WireGuard VPN makes it to 1.0.0-and into the next Linux kernel.* In: Ars Technica [online]. Condé Nast, 30. března 2020 [cit. duben 2020]. Dostupné z: https://arstechnica.com/gadgets/2020/03/wireguard-vpn-makes-it-to-1-0-0-and-into-the-next-linux-kernel/

[42] Oficiální stránky TunSafe [online]. TunSafe AB, 2019 [cit. duben 2020]. Dostupné z: https://tunsafe.com/

Příloha A

Tab. A.1

Konfigurace statického směrování pro směrovač R1

Příkaz	Komentář
`Router(config)#hostname R1`	Zadáme jméno směrovače

`R1(config)#line console 0`
`R1(config-line)#logging synchronous`
`R1(config-line)#exit`
	Zabrání výpisu hlášek do psaného textu v CLI

`R1(config)#int fa0/0`
`R1(config-if)#ip add 172.16.1.1 255.255.255.0`
`R1(config-if)#speed 100`
`R1(config-if)#no shut`
	Nastavení IP adresy, rychlosti a zapnutí rozhraní směrovače FastEthernet 0/0

`R1(config-if)#int fa0/1`
`R1(config-if)#ip add 10.0.0.1 255.255.255.248`
`R1(config-if)#speed 100`
`R1(config-if)#no shut`
`R1(config-if)#exit`
	Nastavení IP adresy, rychlosti a zapnutí rozhraní směrovače FastEthernet 0/1

`R1(config)#ip route 0.0.0.0 0.0.0.0 10.0.0.2` `R1(config)#end`	Nastavení statické cesty

Tab. A.2

Konfigurace statického směrování pro směrovač R2

Příkaz	Komentář
`Router(config)#hostname R2`	Zadáme jméno směrovače

`R2(config)#line console 0`
`R2(config-line)#logging synchronous`
`R2(config-line)#exit`
	Zabrání výpisu hlášek do psaného textu v CLI

`R2(config)#int fa0/0`
`R2(config-if)#ip add 172.16.2.1 255.255.255.0`
`R2(config-if)#speed 100`
`R2(config-if)#no shut`
	Nastavení IP adresy, rychlosti a zapnutí rozhraní směrovače FastEthernet 0/0

`R2(config-if)#int fa0/1`
`R2(config-if)#ip add 10.0.0.2 255.255.255.248`
`R2(config-if)#speed 100`
`R2(config-if)#no shut`
`R2(config-if)#exit`
	Nastavení IP adresy, rychlosti a zapnutí rozhraní směrovače FastEthernet 0/1

`R2(config)#ip route 0.0.0.0 0.0.0.0 10.0.0.1` `R2(config)#end`	Nastavení statické cesty

Tab. A.3

Konfigurace SPAN pro přepínač SW1

Příkaz	Komentář

`Switch(config)#line console 0`
`Switch(config-line)#logging synchronous`
`Switch(config-line)#exit`
	Zabrání výpisu hlášek do psaného textu v CLI

`Switch(config)#int range fa0/1 - 3`
`Switch(config-if-range)#speed 100`
`Switch(config-if-range)#no shut`
	Nastavení rychlosti a zapnutí rozhraní přepínače FastEthernet 0/1, 0/2, 0/3

`Switch(config)#monitor session 1 source int fa0/1 - 2`
`Switch(config)#monitor session 1 destination int fa0/3`
`Switch(config)#end`
	Zadáme zdrojová a cílová rozhraní pro monitorování provozu

Tab. A.4

Konfigurace IPSec pro směrovač R1

Příkaz	Komentář
`R1(config)#crypto isakmp policy 1`	Konfigurace politiky ISAKMP č. 1.
`R1(config-isakmp)#authentication pre-share`	Metoda autentizace je předsdílený klíč.
`R1(config-isakmp)#encryption aes 128`	Pro fázi 1 bude použit šifrovací algoritmus AES-128.
`R1(config-isakmp)#hash sha`	Bude použit algoritmus SHA.
`R1(config-isakmp)#group 2`	Použitá skupina Diffie-Hellman je skupina 2.
`R1(config-isakmp)#lifetime 86400`	Životnost fáze 1 je 86 400 sekund.
`R1(config-isakmp)#exit`	
`R1(config)#crypto isakmp key KkEeYy address 10.0.0.2`	Zadání předsdíleného klíče `KkEeYy` a IP adresy protistrany.
`R1(config)#crypto ipsec transform-set ESP-AES esp-aes 128 esp-sha-hmac` `R1(cfg-crypto-trans)#exit`	Zadání transformační sady s názvem ESP-AES s algoritmem šifrování provozu AES-128 a protokolem ESP, hashovacím algoritmem SHA
`R1(config)#ip access-list extended LIST-1`	Zadání rozšířeného přístupového seznamu s názvem LIST-1.
`R1(config-ext-nacl)#permit ip 172.16.1.0 0.0.0.255 172.16.2.0 0.0.0.255`	Data mezi sítěmi `172.16.1.0` a `172.16.2.0` budou přenesena přes IPSec tunel.

Tab. A.4 (dokončení)

`R1(config-ext-nacl)#exit`	
`R1(config)#crypto map IPSEC-MAP 1 ipsec-isakmp`	Zadání kryptomapy s názvem IPSEC-MAP a pořadovým číslem 1.
`R1(config-crypto-map)#match address LIST-1`	Kryptomapa platí pro provoz ze seznamu LIST-1.
`R1(config-crypto-map)#set peer 10.0.0.2 default`	Zadání IP adresy protistrany.
`R1(config-crypto-map)#set transform-set ESP-AES`	Pro šifrování v rámci kryptomapy bude použita transformační sada ESP-AES.
`R1(config-crypto-map)#set pfs group2`	V rámci kryptomapy bude použitá metoda Perfect Forward Secrecy
`R1(config-crypto-map)#set security-association lifetime seconds 86400`	Životnost spojení v rámci kryptomapy je 86 400 sekund.
`R1(config-crypto-map)#exit`	

`R1(config)#int fa0/1`
`R1(config-if)#crypto map IPSEC-MAP`
`R1(config-if)#exit`
`R1(config)#end`
	Aplikace kryptomapy IPSEC-MAP na výchozí rozhraní FastEthernet 0/1

Tab. A.5

Konfigurace IPSec pro směrovač R2

Příkaz	Komentář
`R2(config)#crypto isakmp policy 1`	Konfigurace politiky ISAKMP č. 1.
`R2(config-isakmp)#authentication pre-share`	Metoda autentizace je předsdílený klíč.
`R2(config-isakmp)#encryption aes 128`	Pro fázi 1 bude použit šifrovací algoritmus AES-128.
`R2(config-isakmp)#hash sha`	Bude použit algoritmus SHA.
`R2(config-isakmp)#group 2`	Použitá skupina Diffie-Hellman je skupina 2.
`R2(config-isakmp)#lifetime 86400`	Životnost fáze 1 je 86 400 sekund.
`R2(config-isakmp)#exit`	
`R2(config)#crypto isakmp key KkEeYy address 10.0.0.1`	Zadání předsdíleného klíče `KkEeYy` a IP adresy protistrany.
`R2(config)#crypto ipsec transform-set ESP-AES esp-aes 128 esp-sha-hmac` `R2(cfg-crypto-trans)#exit`	Zadání transformační sady s názvem ESP-AES s algoritmem šifrování provozu AES-128 a protokolem ESP, hashovacím algoritmem SHA
`R2(config)#ip access-list extended LIST-1`	Zadání rozšířeného přístupového seznamu s názvem LIST-1.

Tab. A.5 (dokončení)

`R2(config-ext-nacl)#permit ip 172.16.2.0 0.0.0.255 172.16.1.0 0.0.0.255`	Data mezi sítěmi `172.16.1.0` a `172.16.2.0` budou přenesena přes IPSec tunel.
`R2(config-ext-nacl)#exit`	
`R2(config)#crypto map IPSEC-MAP 1 ipsec-isakmp`	Zadání kryptomapy s názvem IPSEC-MAP a pořadovým číslem 1.
`R2(config-crypto-map)#match address LIST-1`	Kryptomapa platí pro provoz ze seznamu LIST-1.
`R2(config-crypto-map)#set peer 10.0.0.1 default`	Zadání IP adresy protistrany.
`R2(config-crypto-map)#set transform-set ESP-AES`	Pro šifrování v rámci kryptomapy bude použita transformační sada ESP-AES.
`R2(config-crypto-map)#set pfs group2`	V rámci kryptomapy bude použitá metoda Perfect Forward Secrecy
`R2(config-crypto-map)#set security-association lifetime seconds 86400`	Životnost spojení v rámci kryptomapy je 86 400 sekund.
`R2(config-crypto-map)#exit`	

`R2(config)#int fa0/1`
`R2(config-if)#crypto map IPSEC-MAP`
`R2(config-if)#exit`
`R2(config)#end`
	Aplikace kryptomapy IPSEC-MAP na výchozí rozhraní FastEthernet 0/1

Tab. A.6

Obsah konfiguračního souboru `server.conf` pro OpenVPN server

`port 1194`	Číslo portu OpenVPN službu.
`proto udp`	Zadáme protokol, který bude použit na transportní vrstvě.
`dev tun`	Virtuální síťový adaptér, který bude vytvořen pro OpenVPN spojení.
`persist-key`	Nepřepočítávat klíče při opakovaném spojení kvůli výpadkům tunelu.
`persist-tun`	Neměnit virtuální síťový adaptér po restartování serveru.
`tls-server`	Server využívá TLS.
`tls-timeout 120`	Timeout pro opakované posílání nepotvrzených paketů.
`dh dh.pem`	Ukážeme klíč Diffie-Hellman.
`ca ca.crt`	Ukážeme certifikát CA.
`cert server.crt`	Ukážeme certifikát serveru.
`key server.key`	Ukážeme privátní klíč serveru.
`crl-verify crl.pem`	Ukážeme revokační seznam certifikátů.
`tls-auth ta.key 0`	Ukážeme statický klíč HMAC.
`server 10.10.0.0 255.255.255.0`	Zadáme diapazonu IP adres v privátní síti.
`comp-lzo`	Povolit kompresi provozu.

Tab. A.6 (dokončení)

`keepalive 10 120`	Každých 10 vteřin posílat ping klientovi. Pokud během 120 vteřin nebude dostána odpověď, restartovat tunel.
`status /var/log/openvpn/openvpn-status.log 1`	Cesta do záznamu statusů.
`log-append /var/log/openvpn/openvpn-server.log`	Cesta do záznamového souboru.
`verb 3`	Stupeň logování.
`mute 20`	Omezení událostí současně odeslaných do záznamového souboru.

Tab. A.7

Obsah konfiguračního souboru `client.conf` pro OpenVPN klienta

`client`	Označíme, že host je OpenVPN klientem.
`proto udp`	Zadáme protokol, který bude použit na transportní vrstvě.
`dev tun`	Virtuální síťový adaptér, který bude vytvořen pro OpenVPN spojení.
`remote 172.16.1.2 1194`	Zadáme IP adresu a číslo portu OpenVPN serveru
`resolv-retry infinite`	Pořád se snažit připojit k serveru, pokud předchozí pokus selhal.
`persist-key`	Nepřepočítávat klíče při opakovaném spojení kvůli výpadkům tunelu.
`persist-tun`	Neměnit virtuální síťový adaptér po restartování serveru.
`comp-lzo`	Povolit kompresi provozu.
`ca ca.crt`	Ukážeme certifikát CA.
`cert client.crt`	Ukážeme certifikát klienta.
`key client.key`	Ukážeme privátní klíč klienta.
`tls-client`	Klient využívá TLS.
`tls-auth ta.key 1`	Ukážeme statický klíč HMAC.
`dh dh.pem`	Ukážeme klíč Diffie-Hellman.
`status /var/log/openvpn/openvpn-status.log 1`	Cesta do záznamu statusů.
`log-append /var/log/openvpn/openvpn-client.log`	Cesta do záznamového souboru.
`verb 3`	Stupeň logování.

Tab. A.8

Obsah konfiguračního souboru virtuálního rozhraní `wg0.conf` pro WireGuard server

`[Interface]`	Nastavení pro WireGuard rozhraní
`Address = 10.10.0.1/24`	IP adresa serveru v privátní síti.
`SaveConfig = true`	Automaticky aktualizovat konfigurační soubor za provozu.
`PrivateKey = YNUf4peymQApmMhLeleROQs2SqPNdU7L3UIxiStY0ls=`	Zadáme privátní klíč serveru z odpovídajícího souboru.
`ListenPort = 51820`	Číslo portu pro rozhraní
`[Peer]`	Nastavení pro WireGuard peer (klienta).
`PublicKey = 86o2awF3Wflp/JuqsJuyApPJAU/lXSDn+eXlFyFFSwo=`	Zadáme veřejný klíč klienta z odpovídajícího souboru
`AllowedIPs = 10.10.0.2/32`	Přiřadíme klientovi IP adresu v privátní síti.

Tab. A.9

Obsah konfiguračního souboru virtuálního rozhraní `wg0.conf` pro WireGuard klienta

`[Interface]`	Nastavení pro WireGuard rozhraní
`Address = 10.10.0.2/32`	IP adresa klienta v privátní síti.
`PrivateKey = WGr/msdBo5znl3I/jB3xZiDPJCCWsCmMr/cODz1B6WM=`	Zadáme privátní klíč klienta z odpovídajícího souboru.
`[Peer]`	Nastavení pro WireGuard peer (server).
`PublicKey = NCSdvEiJwTqffYFPbZ1cYHSDLNzy5tR8jE8A2dvc5Cc=`	Zadáme veřejný klíč serveru z odpovídajícího souboru
`Endpoint = 172.16.1.2:51820`	Zadáme IP adresu a číslo portu peeru (zde serveru).
`AllowedIPs = 0.0.0.0/0`	Seznam pro řízení přístupu (Access-control List, ACL). `0.0.0.0/0` - veškerý provoz bude procházet přes WireGuard tunel.
`PersistentKeepalive = 21`	Odesílání keep alive paketů každých 21 vteřin