



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV MIKROELEKTRONIKY

DEPARTMENT OF MICROELECTRONICS

PŘENOSNÉ ZÁZNAMOVÉ ZAŘÍZENÍ S MULTI GIGABIT ETHERNET ROZHRANÍM

PORTABLE RECORDING DEVICE WITH MULTI GIGABIT ETHERNET INTERFACE

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

Filip Brázda

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. Edita Hejátková

BRNO 2025



Bakalářská práce

bakalářský studijní program **Mikroelektronika a technologie**

Ústav mikroelektroniky

Student: Filip Brázda

ID: 246792

Ročník: 3

Akademický rok: 2024/25

NÁZEV TÉMATU:

Přenosné záznamové zařízení s multi gigabit Ethernet rozhraním

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte a realizujte přenosné záznamové zařízení s multi gigabit Ethernet rozhraním, které umožní rychlý a efektivní záznam velkých objemů dat. Provedte rešerši současných záznamových zařízení a technologií, které se používají pro záznam multi gigabit Ethernet rozhraní a zhodnoťte jejich výhody a nevýhody z hlediska výkonu, rozměrů, složitosti, ceny a mobility.

Navrhněte architekturu přenosného záznamového zařízení, které bude schopné zpracovat a uložit data z multi gigabit Ethernet rozhraní na paměťové médium. Vyberte vhodnou technologii a komponenty, které budou vyhovovat požadavkům na přenosovou rychlost, mobilitu, rozšiřitelnost. Berte v úvahu cenu a dostupnost komponent. Při návrhu pracujte s obecně podporovanými souborovými systémy, kdy se zaznamenávaná data na disk ukládají ve formě souborů ve formátu, který je podporován nástrojem Wireshark (např. pcap).

Vytvořte podklady pro sestavení funkčního vzoru zařízení ve smyslu výběru HW komponent a jejich propojení. Specifikujte požadavky na SW vybavení (knihovny, vlastní kód apod.). Naceňte navržené řešení a definujte rizika a omezení spojená s navrženým řešením.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce.

Termín zadání: 10.2.2025

Termín odevzdání: 5.6.2025

Vedoucí práce: Ing. Edita Hejátková

doc. Ing. Pavel Šteffan, Ph.D.

předseda rady studijního programu

UPOZORNĚNÍ:

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

ABSTRAKT

Tato bakalářská práce se zabývá návrhem a realizací přenosného záznamového zařízení s podporou více-gigabitového Ethernetového rozhraní. Cílem je vytvořit kompaktní a výkonné zařízení schopné spolehlivě zaznamenávat velké objemy síťového provozu na paměťové médium ve formátu kompatibilním s běžnými nástroji pro analýzu, jako je Wireshark. V rámci práce je provedena rešerše dostupných komerčních řešení a použitých technologií, které jsou porovnány z hlediska výkonu, rozměrů, ceny, složitosti a mobility. Na základě této analýzy je navržena architektura vlastního zařízení včetně výběru vhodných hardwarových a softwarových komponent. Součástí návrhu je také specifikace způsobu ukládání dat, softwarového vybavení a vytvoření podkladů pro sestavení funkčního vzoru zařízení. Výsledné řešení je zhodnoceno z hlediska nákladů, technické proveditelnosti a identifikovaných rizik.

KLÍČOVÁ SLOVA

síťový provoz, záznamové zařízení, multi-gigabit Ethernet, analýza paketů, Wireshark, PCAP, návrh systému, přenosné zařízení

ABSTRACT

This bachelor thesis deals with the design and implementation of a portable recording device with support for a multi-gigabit Ethernet interface. The aim is to create a compact and powerful device capable of reliably recording large volumes of network traffic to a storage medium in a format compatible with common analysis tools such as Wireshark. The thesis includes a review of available commercial solutions and technologies, which are compared in terms of performance, size, price, complexity, and mobility. Based on this analysis, the architecture of the device is designed, including the selection of suitable hardware and software components. The design also includes specifications for data storage, software equipment, and the creation of documentation for assembling a functional prototype of the device. The resulting solution is evaluated in terms of cost, technical feasibility, and identified risks.

KEYWORDS

network traffic, capture device, multi-gigabit Ethernet, packet analysis, Wireshark, PCAP, system design, portable device

BRÁZDA, Filip. *Přenosné záznamové zařízení s multi gigabit Ethernet rozhraním*. Bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav mikroelektroniky, 2025. Vedoucí práce: Ing. Edita Hejátková

Prohlášení autora o původnosti díla

Jméno a příjmení autora: Filip Brázda
VUT ID autora: 246792
Typ práce: Bakalářská práce
Akademický rok: 2024/25
Téma závěrečné práce: Přenosné záznamové zařízení s multi gigabit Ethernet rozhraním

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

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno

.....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucí bakalářské práce paní Ing. Editě Hejátkové za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Úvod	12
1 Požadavky na záznamové zařízení s multi-gigabit Ethernet rozhraním	13
1.1 Síťové a systémové vrstvy	13
1.2 OSI model	13
1.2.1 Ethernet rozhraní	14
1.2.2 Jumbo rámec	15
1.2.3 Síťová vrstva a IP	16
1.2.4 Transportní vrstva a protokol UDP	17
1.3 Formáty záznamu a softwarová podpora	18
1.3.1 PCAP	18
1.3.2 PCAP Next Generation	19
1.3.3 Extensible Record Format	20
1.3.4 Wireshark	21
1.3.5 Scapy	21
1.4 Operační systémy	22
1.4.1 Windows	22
1.4.2 Linux	22
2 Architektury a komponenty vhodné pro záznam síťového provozu	24
2.1 Přehled architektonických možností	24
2.1.1 Architektura s využitím notebooku a externí síťové karty	24
2.1.2 Řešení záznamu datového toku pomocí kombinace FPGA a CPU	25
2.1.3 Modulární řešení s embedded počítačem	26
2.2 Přehled komerčně dostupných zařízení	26
2.2.1 PacketRAID 100G Portable	26
2.2.2 PROFISHARK 10G	27
2.2.3 n2disk by ntop	28
2.2.4 PacketExpert™ 100G	28
2.2.5 FMADIO PORTABLE	29
2.3 Srovnání technických parametrů a cenové dostupnosti	30
2.4 Odhad vnitřní architektury těchto zařízení	31
2.5 Síťové adaptéry a rozhraní	31
2.5.1 Síťové karty	31
2.5.2 RJ45	32

2.5.3	SFP+	32
2.5.4	Thunderbolt	32
2.6	Úložiště a paměťové subsystémy	33
2.6.1	SSD - Solid State Drive	33
2.6.2	NVMe SSD - Non-Volatile Memory Express SSD	34
2.6.3	RAID - Redundant Array of Independent Disks	34
3	Návrh a realizace vlastního řešení	37
3.1	Výběr hardwarové platformy	37
3.1.1	Notebook jako základ hardwarového řešení	37
3.1.2	Výběr a připojení síťové karty	37
3.1.3	Výběr a připojení datového disku	38
3.2	Výběr operačního systému a jeho úprava	38
3.2.1	AlmaLinux	38
3.2.2	Úprava jádra a síťových parametrů	39
3.2.3	Nastavení externí síťové karty	41
3.3	Vývoj vlastní aplikace	42
3.3.1	Optimalizace tcpdump pomocí Gulp a RX_RING	42
3.3.2	Použité technologie	43
3.3.3	Struktura a funkce aplikace	44
3.3.4	Rozbor zdrojového kódu	45
3.3.5	Přiložený zdrojový kód	48
4	Testování a ladění výkonu	49
4.1	Generování datového provozu	49
4.1.1	Python skript	49
4.1.2	Scapy	50
4.1.3	Mikrotik RB5009	51
4.1.4	Další možnosti	54
4.2	Měření propustnosti při zátěži	54
4.2.1	Výsledky měření a porovnání v ostatních nástrojích	54
4.2.2	Optimalizace výkonu	58
4.2.3	Vyhodnocení výkonu po optimalizaci aplikace	61
4.3	Zatížení vyšším počtem paketů menší velikosti	62
5	Návrh alternativního řešení	65
5.1	Shrnutí zkušeností s notebookem	65
5.2	Řešení s embedded počítačem	65
5.2.1	Advantech MIO-4370	68
5.2.2	Advantech MIO-2364	68

5.2.3	Advantech MIO-5379	69
5.2.4	Maxtang MTN-FP750	69
5.2.5	Specializovaná zařízení pro vysokorychlostní síťový provoz . .	70
5.3	Výběr PC a komponent	70
5.3.1	NVMe disk	70
5.3.2	Síťová karta 10 <i>GbE</i>	71
5.3.3	Paměť RAM	71
5.4	Shrnutí	71
6	Postup pro sestavení funkčního prototypu	73
	Závěr	75
	Literatura	78
	Seznam symbolů a zkratk	85
A	ZDROJOVÝ KÓD APLIKACE (ELEKTRONICKÁ PŘÍLOHA)	87

Seznam obrázků

1.1	Struktura PCAPNG formátu	20
2.1	PacketRAID 100G	27
2.2	ProfiShark 10G	28
2.3	PacketExpert 100G	29
2.4	FMADIO PORTABLE	30
2.5	Srovnání paramaterů	30
2.6	SFP+ Transceiver	33
2.7	RAID 10 diagram	36
3.1	Nastavené hodnoty systémových bufferů	40
4.1	Mikrotik RB5009	52
4.2	Nastavení hodnoty MTU pro 2.5 Gb port	52
4.3	Nastavení traffic generatoru	53
4.4	První test s 500 Mbps	55
4.5	Druhý test s 1 000 Mbps	55
4.6	Reálná rychlost při nastavení 1 000 Mbps	56
4.7	Třetí test s 1 500 Mbps	56
4.8	Reálná rychlost při nastavení 1 500 Mbps	56
4.9	Čtvrtý test s 2 000 Mbps	57
4.10	Reálná rychlost při nastavení 2 000 Mbps	57
4.11	Test záznamu 2 500 Mbps pomocí tcpdump	58
4.12	Pátý test s 2 500 Mbps	58
4.13	Reálná rychlost při nastavení 2 500 Mbps	59
4.14	Původní hodnoty nastavení bufferů	59
4.15	Změněné hodnoty nastavení bufferů	59
4.16	Uspávání programu pomocí funkce <i>usleep()</i>	59
4.17	Replikování výsledků testu aplikace při 2 500 Mbps	61
4.18	Replikování výsledků testu tcpdump při 2 500 Mbps před přidáním parametru -B	62
4.19	Nastavení rychlejšího toku s menšími pakety	63
4.20	Měření propustnosti aplikace pro zátěž vyšším počtem paketů menší velikosti	64
4.21	Měření propustnosti nástroje Wireshark pro zátěž vyšším počtem pa- ketů menší velikosti	64
5.1	Zatížení zařízení v klidovém stavu	67
5.2	Zatížení zařízení při spuštěné aplikaci	67
5.3	Srovnání procesorů testovacího zařízení a MIO-4370	69
6.1	Umístění slotu pro RAM paměť na desce embedded počítače MIO-4370	73

6.2	Umístění slotů pro připojení NVMe disků nebo síťové karty 10 <i>GbE</i> .	74
6.3	Ukázka připojení dvou NVMe disků k MIO-4370	74

Úvod

S rostoucími nároky na přenosovou kapacitu počítačových sítí a stále větším objemem přenášených dat roste i potřeba efektivního záznamu síťového provozu při vysokých rychlostech. V oblasti síťové analýzy, vývoje komunikačních technologií, kybernetické bezpečnosti nebo forenzní analýzy dat je nezbytné mít k dispozici zařízení schopná zachytit síťový provoz v reálném čase bez ztráty paketů a s možností jeho dalšího zpracování.

Komerčně dostupná zařízení určená pro záznam vícegigabitového Ethernetového provozu jsou často finančně nákladná, fyzicky rozměrná a málo flexibilní, což omezuje jejich použití v mobilních nebo provozně náročných podmínkách. V praxi však stále více vzniká potřeba kompaktních, přenosných a výkonných řešení, která umožní záznam síťového provozu mimo standardní laboratorní nebo datová prostředí.

Tato bakalářská práce se zabývá návrhem a realizací přenosného záznamového zařízení s podporou vícegigabitového Ethernetového rozhraní. Cílem je vytvořit zařízení, které bude schopno spolehlivě zaznamenávat velké objemy síťových dat na paměťové médium ve formátu podporovaném nástroji pro analýzu provozu, jako je například Wireshark (např. formát PCAP). Práce zahrnuje rešerši současně dostupných řešení a použitých technologií, porovnání jejich výhod a nevýhod z hlediska výkonu, rozměrů, ceny, složitosti a mobility.

Na základě této analýzy je navržena vlastní architektura záznamového zařízení včetně výběru vhodných hardwarových a softwarových komponent s ohledem na přenosovou rychlost, mobilitu, rozšiřitelnost, cenu a dostupnost. Součástí návrhu je také specifikace požadavků na softwarové vybavení, návrh způsobu ukládání dat a vytvoření podkladů pro sestavení funkčního vzoru zařízení. Výsledkem je návrh realizovatelného řešení včetně odhadu nákladů a identifikace možných rizik a omezení.

1 Požadavky na záznamové zařízení s multi-gigabit Ethernet rozhraním

Návrh záznamového zařízení pro vysokorychlostní síťovou komunikaci, například multicastový přenos přes UDP ve formátu jumbo paketů, vyžaduje pečlivé sladění hardwarových a softwarových komponent. Po hardwarové stránce je důležité zajistit dostatečnou propustnost mezi síťovým rozhraním, procesorem, pamětí a úložištěm, aby bylo možné zachytit a uložit data bez ztrát. Klíčovou roli hraje výběr síťové karty s podporou vyšších přenosových rychlostí a rámců větší velikosti, dostatečná operační paměť pro buffering a rychlé úložiště pro plynulý zápis dat.

Z pohledu softwaru je nutné využít efektivní způsoby přístupu k síťovým datům, například přímé mapování paměti nebo nízkoúrovňové sockety, a optimalizovat nastavení systému tak, aby nedocházelo ke ztrátám kvůli přetížení. Zároveň je třeba přizpůsobit aplikaci charakteru přenášeného provozu – například správně nastavit příjem multicastového toku, zpracovávat velké pakety a zajistit co nejmenší zpoždění. Návrh takového zařízení tedy vyžaduje znalosti z oblasti počítačových sítí, systémového programování i práce s výkonným embedded nebo serverovým hardwarem.

1.1 Síťové a systémové vrstvy

Pro návrh zařízení určeného k záznamu síťového provozu je nezbytné porozumět principům síťové komunikace a způsobu, jakým jsou data přenášena a zpracovávána v jednotlivých vrstvách. Tato část práce se věnuje popisu základních síťových a systémových vrstev s důrazem na ty, které mají přímý vliv na přenos, strukturu a analýzu paketů. Úvodem je představen referenční model ISO/OSI, který poskytuje rámec pro pochopení funkce jednotlivých vrstev a jejich významu při záznamu dat na úrovni Ethernetu.

1.2 OSI model

OSI (Open Systems Interconnection) model je koncepční rámec standardizující funkce síťového systému do 7 kategorií - vrstev. Byl vydán mezinárodní organizací pro standardizaci ISO a má zaručovat správnou komunikaci mezi zařízeními připojenými do sítě jasným definováním, co má jaká vrstva sítě na starost.[1]

1. Fyzická vrstva: První a nejnižší vrstva se zabývá fyzickým aspektem přenosu dat. Zahrnuje kabely pro přenos, elektrické signály a hardwarové zařízení. Přeměňuje data do binárního formátu pro přenos přes různá média.

2. Linková vrstva: Tato vrstva zajišťuje přenos dat z jednoho uzlu do druhého a také detekci vzniklých chyb při přenosu, případně jejich korekci. Zabaluje data do rámců a přes vybraný protokol, jako je například Ethernet nebo Wi-Fi, předává rámce fyzickému médiu pro přenos. Na této vrstvě operují síťová zařízení jako přepínače (switche), které rozhodují o směrování rámců na základě MAC adres, a opakovací (repeatery), které slouží k zesílení a obnově signálu při jeho degradaci během delšího přenosu.
3. Síťová vrstva: Síťová vrstva vybírá nejlepší cestu, kterou budou muset odesílané pakety projít tak, aby dorazily ke správnému adresátovi. Na této vrstvě pracují routery a protokol IP.
4. Transportní vrstva: Přenáší data použitím protokolů jako jsou TCP nebo UDP. Odpovídá za rozdělení větších datových bloků do menších segmentů, které lze efektivně přenášet přes síť, a za jejich opětovné složení na straně příjemce. Dále se stará o řízení toku dat tak, aby nedošlo k zahlcení přijímající strany, a zajišťuje správné pořadí doručených segmentů. Každé spojení mezi aplikacemi identifikuje pomocí tzv. portů, čímž umožňuje více současných datových toků mezi stejnými zařízeními.
5. Relační vrstva: Relační vrstva je zodpovědná za řízení, udržování a ukončování relací mezi aplikacemi na různých zařízeních. Udržuje komunikaci konzistentní a bez přerušení.
6. Prezentační vrstva: Prezentační vrstva slouží jako překladatel dat mezi vrstvou aplikační a relační. Zajišťuje správné formátování a interpretování dat na obou stranách komunikace. Mezi její funkce patří také kódování a dekodování dat, komprese, dekomprese a šifrování či dešifrování přenášených informací. Díky tomu může zajistit, že data odeslaná jedním systémem budou správně pochopena jiným systémem, i když mají odlišné vnitřní formáty nebo znakovou sadu (např. ASCII vs. EBCDIC).
7. Aplikační vrstva: Aplikační vrstva poskytuje rozhraní mezi uživatelem a sítí, tak aby aplikace mohla komunikovat s ostatními aplikacemi přes síť. Na této vrstvě se využívá protokolů HTTP/HTTPS, FTP, DNS nebo SNMP.[2, 3]

1.2.1 Ethernet rozhraní

Ethernetové rozhraní je široce používaná technologie, která definuje metody a standardy pro přenos dat na lokální síti. Zahrnuje náležitosti posílání dat, jako je struktura odesílaných rámců, principy a způsoby odesílání a různé typy ethernetových technologií. Ethernet protokol pracuje na druhé vrstvě OSI modelu.

Rámec je standardizované uspořádání a zapouzdření dat pro přenos skrz Ethernetovou síť. Používá se pro spolehlivost a rychlost přenosu. V základu ethernetového

rámce jsou odesílaná data, ke kterým se přidávají tzv. metadata obsahující informace, jako je MAC adresa odesílatele a příjemce, nebo bitové sekvence pro kontrolu. Konkrétní části ethernetového rámce pak jsou:

- Úvodní data: Sedm bajtů zajišťující synchronizaci příjemcova hodinového signálu s přijímanými daty.
- SFD (Start Frame Delimiter): Ukazuje na konec úvodních dat a na začátek samotného rámce. V ethernetových sítích má velikost jednoho bajtu.
- MAC adresa příjemce: Každé zařízení má svoji specifickou MAC (Media Access Control) adresu o velikosti šesti bajtů (48 bitů).
- MAC adresa odesílatele
- Typ protokolu nebo velikost datového balíku: Dva bajty informující o typu použitého protokolu nebo o délce dat. Konkrétní význam těchto bajtů závisí na typu rámce – v případě rámce Ethernet II tato položka určuje typ protokolu vyšší vrstvy (např. IPv4, ARP), zatímco u rámce dle standardu IEEE 802.3 označuje délku datového pole.
- Užitečná data (payload): Samotná přenášená data. Může dosahovat velikostí od 46 až po 1500 bajtů. Pokud by odesílaný datový balík nedosahoval ani na spodní hranici 46 bajtů, doplní se balík obvykle nulovými bity tak, aby minimální velikosti splňoval. Minimální délka rámce umožňuje detekci kolizí v síti při polo-duplexním provozu.
- FCS (Frame Check Sequence): Čtyři bajty umožňující detekci přenosových chyb pomocí CRC (cyklický redundantní součet). FCS pomáhá redukovat chyby tím, že umožňuje přijímajícímu zařízení ověřit integritu rámce – pokud výpočet CRC neodpovídá přijaté hodnotě, je rámec považován za chybný a je zahozen. Tím se zajišťuje, že do vyšších vrstev nebudou předána poškozená data.[4]

1.2.2 Jumbo rámec

Jumbo rámec je název pro speciální typ datového přenosu, kdy dochází k překročení standardní velikosti Ethernetového rámce, která činí 1500 bajtů. V kontextu Ethernetu hovoříme o jumbo rámcích, jejichž velikost obvykle dosahuje až 9000 bajtů. Na úrovni IP protokolu je teoretická maximální velikost paketu až 65 535 bajtů, avšak v praxi je tato hodnota limitována vlastnostmi použitých síťových technologií.

Jumbo rámce se využívají především pro zvýšení přenosové efektivity – díky většímu poměru užitečných dat ku režii hlaviček, které zůstávají stejné jako u běžných rámců, je možné snížit zatížení procesoru a zvýšit propustnost sítě. Jsou běžně používány ve vysokorychlostních sítích nebo tam, kde je třeba efektivně přenášet velké objemy dat.

Pokud má síť zpracovávat jumbo rámce, musí být všechna zařízení, kterými tyto rámce procházejí, na tuto velikost správně nastavena. V opačném případě může docházet k jejich zahazování. Použití jumbo rámců může v některých případech způsobit zvýšenou latenci, zejména pokud zpracování větších rámců není na všech prvcích sítě dostatečně optimalizované.[5, 6]

1.2.3 Síťová vrstva a IP

Internetový protokol (IP) určuje, jak mají IP adresy vypadat a jak se používají pro směrování dat v síti, ale samotné přiřazování IP adres jednotlivým zařízením zajišťují jiné mechanismy, jako je DHCP (Dynamic Host Configuration Protocol) nebo statická konfigurace. Každé zařízení připojené do sítě tak získá jedinečnou IP adresu, která slouží k jeho identifikaci v síťové komunikaci.

IP protokol zajišťuje rozdělení dat na menší bloky zvané pakety, které obsahují zdrojovou a cílovou IP adresu, kontrolní informace a vlastní užitečná data. Směrovače (routery) pak rozhodují o optimální cestě, kterou tyto pakety putují k cílovému zařízení. Díky nezávislosti na typu fyzické sítě (např. Ethernet, Wi-Fi, optická síť) je IP protokol široce použitelný napříč různými technologiemi.

Jedná se však o nespolehlivý protokol – nezaručuje doručení paketů bez ztrát ani jejich správné pořadí. Tyto vlastnosti zajišťují až protokoly vyšších vrstev, jako je například TCP. IP protokol také podporuje čtyři základní režimy přenosu:

- Unicast - Data jsou přenášena od zdroje pouze k jednomu příjemci. Paket vytvořený odesílatelem obsahuje jeho IP adresu a adresu příjemce. Využívá se pro komunikaci pouze mezi dvěma zařízeními, například při zadávání URL adres do webového prohlížeče. Jelikož jsou data určena pouze pro jednoho příjemce snižuje se pravděpodobnost nežádoucího přístupu nim. Pokud je však požadavků na přenos hodně, může dojít k přetížení sítě.
- Multicast - Pakety jsou odesílány z jednoho zdroje k příjemcům, kteří jsou součástí určité multicastové skupiny. Ta je identifikována pomocí specifické multicastové IP adresy. Tento způsob je využíván po televizní vysílání přes internet bez nutnosti individuálního unicastového přenosu. Dále je možné použít při streamování multimedií.
- Broadcast – Pakety putují od zdroje ke všem zařízením připojeným v dané síti. Rozsah broadcastu je omezen na tzv. broadcastovou doménu, která je určena specifickou IP adresou (např. 192.168.1.255 pro síť 192.168.1.0/24). Routery broadcastové pakety zpravidla nepřeposílají do dalších segmentů, proto se broadcast využívá převážně v lokálních sítích (LAN). Tento režim umožňuje efektivní distribuci dat všem zařízením v síti bez nutnosti znát jejich adresy, například při síťových oznámeních nebo při žádosti o IP adresu od DHCP

serveru. Broadcast je podporován pouze v protokolu IPv4. V IPv6 byl tento režim odstraněn a nahrazen multicastem na speciální adresu, která reprezentuje všechna zařízení v síti. To pomáhá snížit zahlcení sítě, které by jinak mohlo vzniknout při častém používání broadcastu.

- Anycast - Anycast je způsob přenosu paketů od zdroje k jednomu příjemci, který je nejbližší ke zdroji. Paket je směrován na zařízení, které je z hlediska topologie nejbližší ke zdroji. Anycast je ideální pro služby, které musí být dostupné z více míst současně. Cílové zařízení mají stejnou IP adresu, ale data se doručí pouze na nejbližší z nich. Výhodou je zajištění nejrychlejšího přenosu, kontinuita funkce pokud je jeden z uzlů nedostupný a rozložení zatížení sítě mezi více serverů. Využívá se například pro DNS servery (Domain Name System) pro poskytování rychlých odpovědí z geograficky nejbližšího serveru a pokud je ten mimo provoz, routery automaticky směřují dotazy na další nejbližší.[7, 8]

1.2.4 Transportní vrstva a protokol UDP

Transportní vrstva je čtvrtá vrstva OSI modelu a zajišťuje přímou komunikační cestu mezi zdrojem a příjemcem. V rámci této vrstvy může být implementována kontrola toku (flow control), která reguluje rychlost přenosu dat s cílem předejít zahlcení příjemce. Tento mechanismus je však používán pouze u protokolu TCP, který poskytuje spolehlivý přenos a zpětnou vazbu od příjemce. Naopak UDP žádnou kontrolu toku neprovádí, protože je nespojovaný a neobsahuje mechanismy pro potvrzování nebo zpětnou vazbu.[9]

Protokol UDP (User Datagram Protocol) je společně s TCP protokolem (Transmission Control Protocol) jedním z hlavních protokolů transportní vrstvy v sadě internetových protokolů (IP). Je znám svou jednoduchostí a efektivitou. Oproti TCP protokolu na počátku komunikace nezačíná ověřením spojení mezi příjemcem a odesílatelem před začátkem odesílání dat. Ta jsou odesílána ve formě datagramů, které obsahují veškeré informace potřebné pro cestu od zdroje k cíli. Hlavička protokolu UDP má velikost pouze 8 bajtů, obsahující pouze informace o zdrojovém portu, cílovém portu, délce celého datagramu a bezpečnostní součet. Tato struktura a malá velikost umožňuje větší rychlosti přenosu, avšak za cenu menší spolehlivosti. Protokol UDP na rozdíl od TCP nevyžaduje potvrzení o přijetí paketů a ani potvrzení přijetí paketů ve stejném pořadí, v jakém byly odeslány. Proto se UDP používá pouze v případech, kde není potřeba zajištění velké spolehlivosti, jako je například přenos videa.[10, 11]

1.3 Formáty záznamu a softwarová podpora

Pro účely záznamu síťového provozu se využívá několik zavedených formátů, které slouží k uložení zachycených paketů pro pozdější analýzu. Nejrozšířenějším z nich je formát PCAP (Packet Capture), případně jeho rozšířená verze PCAPng (PCAP Next Generation). Tyto formáty jsou široce podporovány nástroji pro analýzu síťového provozu, jako je například Wireshark, a umožňují efektivní ukládání metadat, časových značek i samotných datových rámců.

1.3.1 PCAP

Formát PCAP (Packet CAPture), vynalezen v 90. letech minulého století, je nejrozšířenějším formátem určeným pro záznam síťových paketů nebo rámců. Ukládá všechny bajty přesně tak, jak je zachytil při průtoku sítí. Zachycené pakety/rámce ukládá ve strukturované formě tak, aby je bylo později možné jednoduše analyzovat. Pro záznam paketů je možné využít tcpdump, libPCAP, WinPCAP, Wireshark nebo jiný program umožňující jejich analýzu a ukládání.

Soubory s koncovkou PCAP mají nejdříve svoji vlastní globální hlavičku o velikosti 24 bajtů, která je v obecném formátu a platí pro celý soubor:

- Magic number (4 bajty) - Magic number neboli Magické číslo identifikuje, že jde o PCAP soubor, určuje endianitu dat (MSB nebo LSB na začátku) a pomáhá zajistit správnou interpretaci dat softwarem pro analýzu.
- Version Major (2 bajty) - Určuje hlavní verzi formátu PCAP. Pro většinu současných PCAP souborů je standardní hlavní verze "2".
- Version Minor (2 bajty) - Určuje vedlejší verzi nebo menší aktualizace formátu PCAP. Pro většinu současných PCAP souborů je standardní vedlejší verze "4".
- Timezone (4 bajty) - Představuje časový posun od UTC (Coordinated Universal Time) v sekundách. Například pro středoevropský čas by hodnota byla "+7200". V praxi se však toto pole nikdy nepoužívá – ve formátu PCAP je vždy nastaveno na 0 a všechny časové údaje v záznamu jsou uváděny přímo v UTC. Pole je zachováno pouze z historických důvodů pro zpětnou kompatibilitu.
- Snap Length (4 bajty) - Udává maximální počet bajtů, které chce zachytávací nástroj uložit z každého paketu. Pro ethernetový rámec je maximální délka "65535".
- Link-Layer Type (4 bajty) - Udává typ linkové vrstvy, která je pro ethernet "1".

Paket uložený v souboru s takovouto hlavičkou má na svém začátku další hlavičku, která již neobsahuje obecné informace, ale informace spojené přímo se zachyceným paketem.

- Timestamp Seconds (4 bajty) - Časová značka zachyceného paketu v sekundách od 1.1.1970, jinak označované jako Unix čas.
- Timestamp Microseconds/Nanoseconds (4 bajty) - Stejně jako předchozí časová značka, tentokrát ale v mikrosekundách nebo v nanosekundách podle magického čísla nastaveného v hlavičce souboru.
- Délka zachycených dat (4 bajty) - Ukazuje délku zachycených dat přesně tak, jak je program zachytil a zapsal.
- Původní délka (4 bajty) - Vyjadřuje délku dat jak původně putovaly po síti. Například pokud by bylo v zachycovacím programu zadáno zapisovat pouze prvních 128 bajtů, ale původní délka paketu by byla delší, pak se do délky zachycených dat uloží 128, ale do původní délky se zapíše celá původní délka.

Formát PCAP však není vhodný pro všechny aplikace. Při větších velikostech souborů PCAP může docházet ke snížení rychlosti operací v analyzačních aplikacích, jako je Wireshark, zvláště pokud dochází zároveň k filtraci. [12, 13]

1.3.2 PCAP Next Generation

Formát PCAP Next Generation (PCAPNG) byl navržen jako moderní náhrada původního formátu PCAP, a to s cílem odstranit jeho omezení a rozšířit možnosti záznamu síťového provozu. Na rozdíl od původního PCAP, který umožňuje pouze omezené uložení informací o typu zachyceného provozu, poskytuje PCAPNG podstatně větší flexibilitu.

Tento pokročilejší formát využívá tzv. Interface Description Blocks (IDB), které obsahují popis konkrétního síťového rozhraní, přes které byly jednotlivé pakety zachyceny. Díky tomu je možné přesně určit původ každého paketu i v případě, že byl provoz snímán současně na více rozhraních s odlišnými parametry. PCAPNG tak představuje robustní a rozšiřitelný formát, který lépe odpovídá nárokům moderních síťových analýz.

Dále je v novějších formátech záznamu, jako je PCAPNG, umožněno vyšší rozlišení časových značek (timestampů). Zatímco původní formát PCAP používá dvě 32bitová pole – jedno pro sekundy a druhé pro mikrosekundy – toto rozlišení může být pro moderní vysokorychlostní sítě nedostatečné z hlediska analytické přesnosti, nikoliv však samotné schopnosti zachytit data.

Formát PCAPNG často využívá buď 64bitové timestampy, nebo dvě 32bitová pole, přičemž skutečné rozlišení (např. mikrosekundy, nanosekundy) je definováno pomocí příznaků v bloku Interface Description Block (IDB). Tímto způsobem lze dosáhnout přesnějšího časování zachycených paketů v prostředích s velmi vysokou přenosovou rychlostí.

Navíc také ukládá formát PCAPNG další metadata o zachyceném paketu,

a to konkrétně jeho zdroj nebo jak bylo zachycení provedeno. Do tohoto formátu je také možné přidávat komentáře pro každý paket zvlášť a ne pouze do hlavičky souboru.

Při vývoji nového formátu PCAPNG byl brán ohled na budoucí rozšiřitelnost, a proto byla zavedena plně bloková struktura, která umožňuje přidávání nových typů bloků bez narušení stávajícího formátu. Tím se usnadňuje podpora nových funkcí, jako jsou vyšší časová přesnost, více rozhraní v jednom souboru nebo ukládání metadat.

Na rozdíl od staršího formátu PCAP, který obsahuje pevnou globální hlavičku následovanou sekvencí záznamů o jednotlivých paketech, je struktura PCAPNG zcela bloková. Každá část (např. popis rozhraní, záznam paketu, metadata) je uložena jako samostatný blok. Přehled jednotlivých bloků je znázorněn na obrázku 1.1.[14]

Blok	Popis	Sekce	Velikost v bajtech
Header Block	Obsahuje globální metadata o souboru	Magic Number	4
		Major Version	2
		Minor Version	2
		Section Length	8
Interface Description Block (IDB)	Definuje vlastnosti síťového rozhraní	Block Type	4
		Block Total Length	4
		LinkType	2
		Reserved	2
		SnapLen	4
		Options	Variabilní
Enhanced Packet Block (EPB)	Obsahuje zachycené síťové pakety	Block Type	4
		Block Total Length	4
		Interface ID	4
		Timestamp	8
		Captured Packet Length	4
		Original Packet Length	4
		Packet Data	Variabilní
		Options	Variabilní
		Block Total Length (pro kontrolu)	4

Obr. 1.1: Struktura PCAPNG formátu

1.3.3 Extensible Record Format

Extensible Record Format (rozšiřitelný formát záznamu) je datová struktura, která umožňuje přidávání nových polí informací nebo typů dat bez narušení kompatibility s již existujícími záznamy. Tento formát byl navržen tak, aby se dal kdykoliv rozšířit, byl flexibilní a přizpůsobitelný. Klíčovými vlastnostmi rozšiřitelného formátu jsou:

1. Modularita - Data jsou strukturována do jednotlivých polí nebo bloků, které mohou být snadno přidány, změněny nebo odebrány.

2. Kompatibilita - Nové záznamy mohou být přidány tak, aby byly stále kompatibilní se staršími verzemi čteček nebo analyzátorů dat.
3. Flexibilita - Umožňuje práci s různorodými typy dat, aniž by bylo nutné redefinovat celý formát.
4. Metadata - Obvykle obsahuje popis jednotlivých polí nebo bloků, což usnadňuje interpretaci dat.

Na tomto formátu je založen formát PCAPNG. Dále se využívá v NoSQL databázích nebo souborových formátech JSON, Avro a Protobuf. [15]

1.3.4 Wireshark

Wireshark je výkonný analyzátor síťových paketů/rámců, umožňující zachytávání, ukládání a analýzu paketů z počítačové sítě. Pomocí tohoto programu je možné nahlédnout hluboko do datových paketů, díky čemuž je možné hledat chyby v síti, vyvíjet software, nové komunikační protokoly nebo názorně ukazovat síťový tok při výuce. Wireshark je velice rozšířený díky Open Source licenci, která umožňuje používání bez jakýchkoli poplatků a také díky schopnosti rozeznat všechny dnes běžně používané protokoly, jako jsou UDP, TCP, ICMP, Ethernet... Aplikace dokáže zachytávat pakety z USB, Wireless LAN, Bluetooth a Ethernet. Je také možné zapnout filtrování na základě protokolových hlaviček nebo generování statistik proteklých dat.

Je ale nutné dávat si pozor na přehlcení. Při vyšších průtocích nebo požadavcích na filtraci může docházet ke snížení výkonu aplikace.[16]

1.3.5 Scapy

Aplikace Scapy vyvinutá Philippem Biondim v roce 2003 umožňuje manipulaci se síťovými pakety. Mezi její hlavní funkce patří:

- Posílání na míru vytvořených paketů - umožňuje vytvoření paketů od základu nebo modifikování již existujících. Díky možnosti využívat všechny základní protokoly je Scapy velice univerzální a schopný nástroj.
- Vyhledávání a ukládání paketů.
- Rozkládání paketů na části pro jejich následnou analýzu.
- Skenování sítě.
- Simulace útoků na síť pro bezpečnostní účely.

Díky jejímu založení na programovacím jazyku Python je práce ve Scapy přívětivá a jednodušší než v ostatních programovacích jazycích. [17, 18]

1.4 Operační systémy

Při výběru vhodného operačního systému pro záznam multi-gigabitového síťového provozu bylo nutné zohlednit několik klíčových oblastí, které ovlivňují spolehlivost a výkon celého systému. Jedním z důležitých parametrů byla spotřeba paměti samotného operačního systému ve výchozím stavu, tedy při nečinnosti, neboť nadměrné zatížení paměti může negativně ovlivnit dostupné prostředky pro samotný záznamový proces.

Dalším důležitým aspektem je otevřenost systému vůči úpravám. Vzhledem k tomu, že většina běžných operačních systémů není od základu optimalizována pro práci s multi-gigabitovým provozem, je často nutné provádět úpravy jádra či měnit různá nízkoúrovňová systémová nastavení. Tyto zásahy však nemusí být u všech systémů možné nebo snadno proveditelné.

V neposlední řadě se jednotlivé operační systémy liší ve své připravenosti na práci s vysokorychlostní síťovou komunikací. Některé distribuce mohou již v základu obsahovat optimalizace či vhodnější výchozí konfiguraci, což následně snižuje množství potřebných úprav a usnadňuje vývoj i samotné nasazení záznamové aplikace.

Z hlediska celkových nákladů je rovněž výhodné, pokud je operační systém i samotná aplikace co nejméně náročná na výpočetní a paměťové zdroje. Díky tomu lze systém provozovat i na cenově dostupnějším embedded nebo kompaktním hardwaru, což výrazně snižuje náklady na nasazení a zvyšuje dostupnost řešení v praxi.

1.4.1 Windows

Operační systém od společnosti Microsoft patří mezi nejrozšířenější platformy a je běžně využíván širokou veřejností, zejména pro svou uživatelskou přívětivost. I když na tomto systému existují nástroje, které by teoreticky umožnily zachytávání síťového provozu s vyšší přenosovou rychlostí, samotný systém je značně náročný na paměť i procesorové prostředky, a to i ve výchozím stavu.

Navíc platforma Windows neposkytuje jednoduchý a přímý přístup k nízkoúrovňovým úpravám jádra, které jsou pro efektivní záznam multi-gigabitového provozu často nezbytné. Tento nedostatek flexibility je mimo jiné dán i tím, že se jedná o placený operační systém s uzavřeným kódem a silně graficky orientovaným prostředím, které je optimalizováno spíše pro uživatelský komfort než pro maximální výkon v oblasti síťového zpracování. [61]

1.4.2 Linux

Operační systém Linux patří mezi nejrozšířenější operační systémy na světě. Ačkoliv není tak běžně nasazován na domácích počítačích jako některé komerční alterna-

tivy, je široce využíván napříč servery, embedded zařízeními, síťovou infrastrukturou, vývojovým prostředím i desktopovými systémy. Linux je oblíbený především mezi vývojáři, správci systémů a technicky zaměřenými uživateli, ale díky moderním uživatelským prostředím je stále přístupnější i pro běžné uživatele.

Základem každého linuxového systému je jádro (kernel) – nízkoúrovňová softwarová vrstva, která přímo komunikuje s hardwarem a poskytuje rozhraní pro běh aplikací a správu systémových prostředků. Nad tímto jádrem je postavena konkrétní distribuce – tedy kompletní operační systém, který zahrnuje nástroje, systém pro správu balíčků, ovladače, služby a případně grafické prostředí. Existují stovky distribucí zaměřených na různé účely – od uživatelsky přívětivých desktopových (např. Ubuntu) přes odlehčené verze pro embedded zařízení až po specializované serverové distribuce s důrazem na stabilitu a výkon (např. Debian, CentOS, AlmaLinux).

Linux vyniká svou modularitou, upravitelností a otevřeným kódem (open-source). Díky tomu je možné přizpůsobit nejen systémové prostředí, ale i samotné jádro konkrétním požadavkům aplikace. To je výhodné zejména v případech, kdy je potřeba nízkoúrovňový přístup k síťovému hardwaru, řízení paměti nebo optimalizace pro konkrétní architekturu.

Další výhodou Linuxu je jeho nízká hardwarová náročnost – většina distribucí spotřebuje méně paměti a procesorového času než komerční alternativy, což umožňuje více systémových prostředků vyhradit pro samotnou aplikaci. Díky tomu lze systém efektivně nasadit i na levnější nebo méně výkonný hardware, aniž by tím byla ohrožena funkčnost např. záznamového zařízení. [62, 63]

2 Architektury a komponenty vhodné pro záznam síťového provozu

Volba vhodné architektury a hardwarových komponent představuje klíčový krok při návrhu zařízení pro záznam síťového provozu s vysokou přenosovou rychlostí. S ohledem na požadavky na propustnost, latenci, spolehlivost záznamu a mobilitu zařízení je nutné zvážit různé varianty uspořádání systému i jednotlivých komponent. V této části jsou popsány možné architektonické přístupy k řešení úlohy záznamu datového toku, včetně jejich výhod a omezení. Zároveň je věnována pozornost klíčovým prvkům, jako jsou síťové adaptéry, typy rozhraní nebo úložné systémy, které hrají zásadní roli v celkovém výkonu navrženého řešení.

2.1 Přehled architektonických možností

Záznamová zařízení mohou mít různou architekturu v závislosti na požadavcích kladených na jejich funkčnost, modularitu, pořizovací náklady, přenosnost, rozšiřitelnost, jednoduchost použití a výkonnost. Na základě těchto kritérií lze navrhovaná řešení rozdělit do několika typových skupin, přičemž každá z nich nabízí odlišný poměr mezi flexibilitou, technickou složitostí a cenovou dostupností.

2.1.1 Architektura s využitím notebooku a externí síťové karty

Architektura založená na notebooku a externí síťové kartě představuje jedno z nejjednodušších a cenově nejdostupnějších řešení záznamu síťového provozu, zároveň však přináší určitá výkonnostní omezení. Princip spočívá v připojení externí síťové karty do vhodného portu notebooku, přičemž klíčovým parametrem je propustnost daného rozhraní.

Externí síťové karty jsou dostupné například s konektivitou přes USB-C. Pro připojení 2,5 *GbE* síťové karty postačuje port ve verzi USB-C 3.2 Gen 1, avšak pro vyšší rychlosti (např. 5 *GbE* nebo 10 *GbE*) je nutné využít notebook s portem USB-C s podporou Thunderbolt, který poskytuje dostatečnou šířku pásma. Výhodou Thunderbolt rozhraní je navíc možnost využití redukci na PCIe Gen. 4, čímž lze připojit síťové karty s konektorem M.2 M-key, běžně používané v desktopových počítačích.

Přes Thunderbolt lze připojit také pokročilejší síťové adaptéry – například převodníky z optického rozhraní na metalické pomocí konektoru SFP+, čímž lze dále rozšířit možnosti připojení a dosáhnout vyšších přenosových rychlostí až do 10 *GbE* a více.

Pro stabilní provoz této architektury je dále nezbytné zajistit dostatečnou kapacitu operační paměti, výpočetní výkon procesoru a odpovídající datové úložiště s vysokou přenosovou rychlostí.

2.1.2 Řešení záznamu datového toku pomocí kombinace FPGA a CPU

Jednou z nejvýkonnějších variant pro záznam síťového provozu ve vysokých přenosových rychlostech je použití specializovaných zařízení kombinujících FPGA (Field Programmable Gate Array) a klasický procesor (CPU). Tato zařízení bývají zpravidla integrována přímo do pokročilých síťových karet (SmartNIC) a umožňují zachytávání, filtrování a zpracování paketů v reálném čase bez ztráty dat, a to i při plném vytížení sítě na úrovni 10, 40 nebo 100 Gb/s. Díky tomu jsou tato řešení hojně využívána v oblasti bezpečnosti, monitoringu i testování síťové infrastruktury.

Zásadní výhodou těchto platform je rozdělení úloh mezi dvě části: FPGA se stará o časově citlivé operace přímo na příchozím datovém proudu – například o zachytávání při linkové rychlosti, předzpracování paketů, klasifikaci nebo filtrování, zatímco CPU zajišťuje správu systému, záznam dat na úložiště nebo komunikaci s vyššími vrstvami softwaru. Mezi FPGA a procesorem probíhá přenos dat přes rozhraní PCI Express (PCIe), obvykle ve verzi Gen. 3 nebo vyšší, aby byla zaručena dostatečná propustnost.

Moderní karty tohoto typu umožňují plně programovatelné filtrování provozu, včetně výběru konkrétních toků či protokolů. Často disponují vlastní operační pamětí, která slouží jako buffer při krátkodobých špičkách zatížení (tzv. microbursts). Díky tomu dokážou zachovat provoz zcela bez ztrát i v extrémních podmínkách.

V oblasti softwarové podpory se tyto karty stále více přibližují běžným síťovým zařízením – výrobci jako Silicom, Napatech nebo Accolade Technology poskytují jednotná softwarová rozhraní (např. knihovny typu fbCAPTURE nebo Link™ Capture Software), která umožňují snadnou integraci s běžnými nástroji, jako jsou Wireshark, Suricata, Snort nebo TRex. To zjednodušuje nasazení i v komplexních prostředích.

Vedle komerčních produktů existují také open-source řešení, jako například platforma NetFPGA, která je vhodná zejména pro výukové a výzkumné účely. Ta poskytuje otevřenou základnu pro vývoj vlastních FPGA aplikací a podporuje standardní formáty, jako je PCAP, čímž usnadňuje napojení na běžné analytické nástroje.

Celkově lze říci, že FPGA+CPU zařízení představují technologicky nejpokročilejší, ale zároveň finančně a implementačně náročnější variantu, která je vhodná pro profesionální nasazení v prostředích, kde je vyžadována nulová ztrátovost, přesné časování paketů a schopnost pracovat s extrémními objemy dat. [52, 53]

2.1.3 Modulární řešení s embedded počítačem

Další variantou je modulární řešení, které je možné složit z jednotlivých komponent a následně jej dále upravovat či rozšiřovat podle potřeby. Základem takového zařízení je embedded počítač, který musí disponovat dostatečným množstvím rozhraní pro připojení všech potřebných periférií.

K embedded systému je následně připojena síťová karta, datové úložiště a případně další zařízení určená pro ovládání a interakci – typicky klávesnice, myš a monitor. Podle konkrétní aplikace může být integrováno také další specializované vybavení, například pro zpracování dat v reálném čase nebo jejich vizualizaci.

Embedded počítače tohoto typu se zpravidla vyznačují kompaktními rozměry, nízkou spotřebou a širokou dostupností pro různá průmyslová i vývojová prostředí. Díky jejich flexibilitě a možnosti přizpůsobit konfiguraci dle požadavků aplikace představují atraktivní variantu pro stavbu vlastního záznamového zařízení s vysokou mírou kontroly nad hardwarem i softwarem. [64]

2.2 Přehled komerčně dostupných zařízení

Na trhu existuje řada specializovaných zařízení určených pro záznam síťového provozu, která pokrývají široké spektrum požadavků z hlediska přenosové rychlosti, přesnosti timestampů, možnosti filtrování či formátů ukládaných dat. Přehled těchto komerčně dostupných řešení poskytuje důležitý referenční rámec pro návrh vlastního zařízení, umožňuje porovnání výkonových parametrů a zároveň napomáhá identifikovat běžně používané architektury a technologie. Tato část uvádí konkrétní příklady profesionálních záznamových zařízení, včetně jejich základních vlastností.

2.2.1 PacketRAID 100G Portable

PacketRAID 100G Portable výrobce The Packet Company dokáže zaznamenávat tok paketů a následně tento tok i analyzovat. Pakety umí zaznamenávat v rychlostech 10 Gbps, 25 Gbps, 40 Gbps a 100 Gbps. Zařízení se vyrábí ve dvou variantách. První je určeno pro použití v serverech v uzavřených prostorech a neobsahuje žádné další prvky, kromě záznamového zařízení jako takového a úložiště. Druhá varianta je přenosné provedení určené k záznamu a analýze dat v poli, k čemuž také potřebuje, mimo základní záznamové jádro, ještě klávesnici, touchpad a obrazovku pro analýzu nasbíraných dat.

Pro přístup k uloženým datům je na obou variantách předinstalovaný systém Linux. Pakety jsou ukládány do úložiště ve standardním formátu PCAP, který je kompatibilní s většinou aplikací používaných pro analýzu síťových paketů, takže je

možné je analyzovat pomocí intuitivního webového GUI nebo pomocí jakéhokoliv konvenčního softwaru, jako je například program Wireshark. Pro záznam dat je možné si navolit 2 až 8 QFSP28 portů. Po zpracování jsou data ukládána na úložiště SSD RAID array o velikostech 8 *Tb* až 240 *Tb*. Od výrobce je poskytnuta záruka na 1 rok až 5 let. [33]



Obr. 2.1: PacketRAID 100G Portable[33]

2.2.2 PROFISHARK 10G

Profishark 10G je malé zařízení od společnosti PROFITAP, které je určeno pro zachytávání paketů tak, že jej postavíme do cesty jejich toku. Tento produkt odposlouchává tok dat na připojeném kabelu a posílá jej skrz USB 3.0 port buď do připojeného počítače pro okamžitou analýzu a ukládání, nebo je možné jej připojit na externí úložiště, jako je například Synology NAS. Pakety jsou ukládány ve standardním formátu PCAP a je možné je přímo na zařízení filtrovat. Pokud je připojen přes USB 3.0 na počítač nebo úložiště, které je schopné dodávat přes USB port napájení, není potřeba Profishark 10G externě napájet. V jiných případech je potřeba zařízení připojit ke zdroji 5 *V*/1,5 *A*. [34]



Obr. 2.2: ProfiShark 10G[34]

2.2.3 n2disk by ntop

n2disk od společnosti ntop není zařízení, ale pouze aplikace, která se stáhne na jakýkoliv přenosný počítač a pro záznam využívá hardwaru tohoto počítače. Na dostatečně dobrém počítači zvládá rychlosti přes 10 *Gbps*. Hlavními požadavky na takový počítač tak budou velikost paměti RAM, velikost a typ úložiště, rychlost procesoru a port, přes který budeme rámce nahrávat. Nasbírané rámce ukládá v klasickém formátu PCAP. Program má v sobě zabudované možnosti pro přidání filtrace paketů, ale funguje pouze s operačním systémem Ubuntu nebo jiným druhem Linuxu. [35]

2.2.4 PacketExpert™ 100G

PacketExpert 100G je sofistikovaný nástroj, designovaný pro testování spolehlivosti a složité evaluace velmi rychlých ethernetových sítí. Podle typu portu, přes který vstupují data do zařízení, je možné ukládat pakety do paměti rychlostí od 1 *Gbps* až po 100 *Gbps*. Vyrábí se jak pro použití ve vnitřních prostorech, tak jako přenosné zařízení s připojenou klávesnicí a monitorem pro analýzu dat přímo na místě měření. K zařízení je i uživatelsky přívětivé webové rozhraní podporující ovládání více uživateli zároveň. Nevýhodou je potřeba externího napájení, jelikož zařízení neobsahuje interní baterii.[36]



Obr. 2.3: PacketExpert 100G Portable[36]

2.2.5 FMADIO PORTABLE

Firma FMADIO Packet Capture nabízí celou škálu zařízení zaznamenávajících pakety. Hlavní rozdíly mezi nabízenými přístroji jsou velikost paměti, počet a typ portů pro zaznamenávání, rychlost zaznamenávání, velikost paměti RAM, váha zařízení a odebíraný výkon. Velikost paměti je možné zvolit od 1 *TB* po 300 *TB*, typy portů jsou RJ45, SFP+, QSFP a QSFP28. Podle typu portu je rychlost zaznamenávání od 100 *Mbps* až po 200 *Gbps*. Paměť RAM lze zvolit v rozmezí 32-128 *GB*. Váha se pohybuje od 5 do 13 *kg* a příkon 200-250 *W*.

V nabídce jsou i přenosné verze tohoto zařízení. Dělí se na dvě skupiny, kde první poskytuje rychlost až 200 *Gbps* a druhá 20 *Gbps*. Dále se pak dělí hlavně podle zvolené velikosti paměti. Verze s rychlostí 20 *Gbps* má nejmenší možnou paměť 4 *TB* a začíná na 9800 \$. Nevýhoda všech přenosných zařízení od tohoto výrobce je potřeba externího napájení. [37]



Obr. 2.4: FMADIO PORTABLE[37]

2.3 Srovnání technických parametrů a cenové dostupnosti

Z obrázku 2.5 je vidět, že jako nejlevnější možnost se jeví n2disk od společnosti ntop. Jedná se však pouze o program, ke kterému je nutné zakoupit navíc hardware. Druhý nejlevnější je PacketExpert 100G. Za cenu 4 200 \$ je možné pořídit zařízení zvládající 10 Gbps rychlost, ale v nabídce je i vyšší rychlost, a to až 100 Gbps. Nejdražší je PacketRAID 100G, kde se cena může vyšplhat až na 300 000 \$. Za tuto cenu je rychlost 100 Gbps a úložiště až 240 Tb.

Zařízení	Samostatné/potřeba PC	Cena	Rychlost	Úložiště	Porty
PacketRAID 100G	Samostatné	až 300000 \$	100 Gbps	8 - 240 Tb	2 - 8x QFSP28
n2disk	potřeba PC	1,999.95 €	40 Gbps	-	-
PROFISHARK 10G	potřeba PC	5 995 €	10 Gbps	-	2x SFP
FMADIO20-PORTABLE	Samostatné	od 9800 \$	20 Gbps	4 - 15 TB	2x 10G SFP+
Packet Expert 100G	Samostatné	od 4200 \$	100 Gbps	neuvedeno	2x 10G SFP+, 2x 100G QSFP28

Obr. 2.5: Srovnání technických parametrů a cenové dostupnosti

2.4 Odhad vnitřní architektury těchto zařízení

Vnitřní struktura komerčně dostupných záznamových zařízení nebývá běžně veřejně dokumentována, avšak na základě jejich vnějšího provedení a popsaných funkcí lze odhadnout pravděpodobné schéma jejich interní architektury.

Podle předpokládaného směru toku dat lze usuzovat, že pakety nejprve vstupují do zařízení prostřednictvím síťového rozhraní, zpravidla ve formě SFP+ nebo QSFP28 konektorů. Následně jsou data zpracovávána specializovanou hardwarovou jednotkou FPGA nebo ASIC, kde dochází k časově kritickým operacím, jako je timestampování, slicing, filtrování nebo případně rozdělování zátěže mezi více výstupními kanály (load balancing).

Za FPGA následuje mezivrstva v podobě operační paměti (RAM), která slouží k vyrovnávání krátkodobých přenosových špiček a minimalizaci rizika ztráty paketů. V této fázi je také obvykle generována PCAP hlavička a připravována struktura výstupních souborů.

Další zpracování zajišťuje řídicí procesorová jednotka (CPU), která se stará o adresaci, správu názvů souborů, zápis na disk a koordinaci ukládání. Vzhledem k nárokům na rychlost zápisu lze předpokládat použití NVMe SSD nebo diskového pole typu RAID, které zajistí dostatečnou propustnost i při vysoké zátěži.

2.5 Síťové adaptéry a rozhraní

Síťový adaptér tvoří zásadní prvek každého zařízení určeného pro zachytávání datového provozu. Jeho vlastnosti určují nejen maximální možnou přenosovou rychlost, ale i kompatibilitu s použitým rozhraním a schopnost efektivně spolupracovat se zbytkem systému. Výběr vhodného adaptéru je proto nezbytný pro dosažení spolehlivého záznamu bez ztráty paketů. Tato část se věnuje přehledu nejčastěji používaných typů síťových karet a rozhraní, která se v oblasti multi-gigabitového provozu běžně využívají.

2.5.1 Síťové karty

Síťové karty jsou nezbytnou součástí jakéhokoli zařízení vyžadující připojení do sítě, ať už se jedná o připojení bezdrátové nebo drátové. Dají se vybírat podle rychlosti přenosu síťových paketů, typu konektivity a aplikace.

Pro drátové připojení existují v základu dvě možnosti: metalické a optické. Metalické síťové karty pracují na protokolech 1000BASE-T, 10GBASE-T atd. podle rychlosti. Optické se využívají tam, kde je potřeba vyšších rychlostí nebo delších

vzdáleností přenosu. Bezdrátové síťové karty se vyskytují v laptopech a mobilních zařízeních a pracují ve spojení s Wi-Fi sítí.

Podle rychlosti je možné síťové karty dělit na nízkorychlostní a vysokorychlostní. Nízkorychlostní obsahují rychlosti do 1 *Gbps* a vysokorychlostní 10 *Gbps*, 25 *Gbps*, 40 *Gbps* až 100 *Gbps*. Takovéto rychlosti vyžadují pouze datová centra, velké společnosti nebo specializované aplikace. V každodenních zařízeních se běžně objevují karty s rychlostí 1 *Gbps*. [19, 20]

2.5.2 RJ45

RJ45 konektor, také známý jako 8P8C konektor, je jedním z nejrozšířenějších konektorů používaných v Ethernetových sítích. Nejvíce se používá při propojování počítačů nebo jiných zařízení na lokální síť LAN (Local Area Network). Konektor má 8 kontaktů v plastovém pouzdře s pojistkou proti náhodnému odpojení. Zapojit se dá podle dvou standardů T568A nebo T568B. Tyto standardy určují, jak jsou v konektoru uspořádány kroucené páry. Obecně je více rozšířený standard T568B. Konektor RJ45 je kompatibilní s kabely Cat5e, Cat6 a Cat7, které podporují různé přenosové rychlosti. Je navržen tak, aby zvládal přenosové rychlosti od Mbps až po 10 *Gbps* skrz standard 10GBase-T. Některé verze konektoru RJ45 mají v těle zabudovanou ochranu proti elektromagnetickému rušení, což zajišťuje integritu signálu a spolehlivý přenos dat. Ke chráněnému konektoru však musí být i chráněný kabel, jinak ochrana ztrácí smysl. [21]

2.5.3 SFP+

SFP je zkratka pro Small Form-factor Pluggable Transceiver. SFP+ je novější verze konektoru SFP. Jedná se o transceiverový modul, který využívá standardu 10GBASE-T (IEEE 802.3an-2006) nebo 10GBASE-SR a je kompatibilní s kabely Cat6 s konektory RJ45 i s optickými vlákny a konektory SC Duplex. Obvykle se používá v síťových zařízeních, jako jsou switche, routery nebo síťové karty. Dovoluje přenosové rychlosti od 10 *Gbps* do 16 *Gbps* jak na měděné kroucené dvoulince, tak na optickém vlákně. U měděné kroucené dvoulinky je omezená maximální vzdálenost přenosu na 100 *m*. Díky jeho malému form factoru je možné v jednom zařízení osadit více SFP+ konektorů vedle sebe než třeba větších XFP. [22, 23, 24]

2.5.4 Thunderbolt

Thunderbolt konektor je označení pro konektor více známý pod názvem USB-C. Podporuje 10 Gigabit Ethernet konektivitu skrze specializované adaptéry pro Thunderbolt 3 a Thunderbolt 4 porty. Samotná Thunderbolt technologie umožňuje rych-



Obr. 2.6: SFP+ Transceiver s konektorem RJ45

lost přenosu až 40 *Gbps*. Uživatel může dosáhnout rychlosti 10 *Gbps* při využití kabelů Cat6 nebo Cat6a, pokud použije Thunderbolt 3 to 10G Ethernet adaptér. Vzhledem k využití adaptéru se tento konektor pro multi-gigabit Ethernet konektivitu používá pouze tam, kde není dostupný jiný síťový konektor nebo je vyžadována multifunkčnost konektoru. [25, 26, 27]

2.6 Úložiště a paměťové subsystemy

Rychlost a spolehlivost úložného subsystemu hrají klíčovou roli při záznamu vysokorychlostního síťového provozu. Zpracovávaná data musí být ukládána s minimálním zpožděním a bez ztrát, což klade vysoké nároky na výběr vhodného typu paměti i způsobu jejího zapojení. Tato část se zaměřuje na dostupné technologie v oblasti úložišť, zejména na SSD, NVMe a RAID systémy, a hodnotí jejich využitelnost v kontextu navrhovaného záznamového zařízení.

2.6.1 SSD - Solid State Drive

SSD paměti představují moderní úložná zařízení založená na technologii flash paměti. Oproti tradičním pevným diskům (HDD) neobsahují žádné pohyblivé mechanické části, což přináší řadu výhod – zejména vyšší přenosové rychlosti, nižší

spotřebu energie a vyšší odolnost vůči mechanickému poškození. Nejčastěji využívají paměť typu NAND flash, která umožňuje rychlé čtení a zápis dat i bez nutnosti trvalého napájení, což řadí SSD mezi nevolatilní typy pamětí.

Nejrozšířenějším rozhraním pro připojení SSD disků je SATA (Serial Advanced Technology Attachment), původně navržené pro mechanické pevné disky, ale později adaptované i pro použití s SSD. SATA však naráží na technologické limity rozhraní, které umožňuje maximální teoretickou přenosovou rychlost 600 MB/s . Tato rychlost je dostatečná pro běžné použití v osobních počítačích, ale nedostačuje pro aplikace, které vyžadují záznam dat z multi-gigabitového síťového rozhraní, tedy přenosové rychlosti v řádu více než 1 GB/s .

Pro tyto náročnější účely se využívají rychlejší rozhraní, jako je například NVMe, které je podrobněji popsáno v následující kapitole. [28, 29]

2.6.2 NVMe SSD - Non-Volatile Memory Express SSD

NVMe (Non-Volatile Memory Express) je vysokorychlostní komunikační protokol, navržený speciálně pro přístup k nevolatilní paměti, jako jsou moderní SSD disky. Tento protokol pracuje přes sběrnici PCI Express (Peripheral Component Interconnect Express), což umožňuje podstatně vyšší přenosové rychlosti a nižší latenci než u starších protokolů, jako je AHCI používaný u SATA disků.

Díky využití více datových linek v rámci PCIe sběrnice mohou moderní NVMe SSD disky dosahovat přenosových rychlostí až kolem $7\,000\text{ MB/s}$ a více. To je výrazný rozdíl oproti SATA, které využívá pouze jednu linku s limitem 600 MB/s . NVMe rovněž nabízí nižší latenci, běžně kolem $25\text{--}30\text{ mikrosekund}$, což výrazně zvyšuje výkon při náročných operacích se soubory.

NVMe bylo navrženo s ohledem na vícejádrové procesory a paralelní zpracování dat, čímž umožňuje efektivní využití moderního výpočetního výkonu. Je kompatibilní s většinou současných operačních systémů. Přestože byly NVMe SSD původně využívány hlavně ve výkonných serverech a specializovaných systémech, dnes jsou běžnou součástí i moderních notebooků a osobních počítačů, kde výrazně zlepšují celkovou odezvu systému. [29, 30]

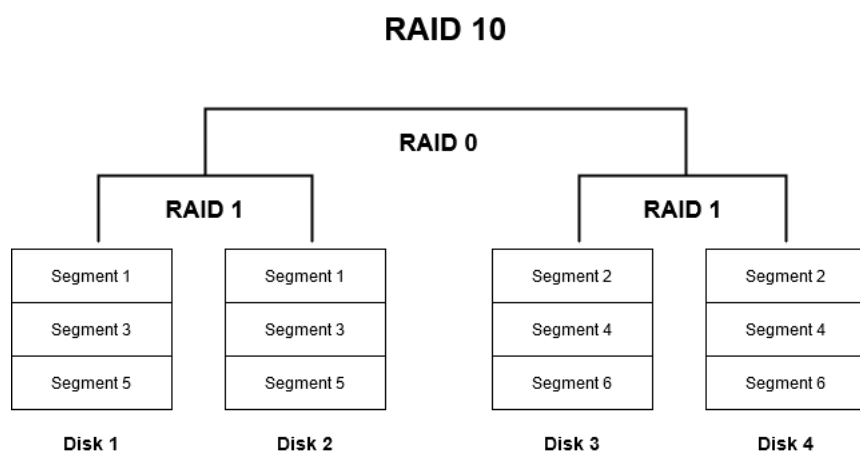
2.6.3 RAID - Redundant Array of Independent Disks

RAID technologie je použití více disků, jejich skládání a kombinování do jedné nebo více logických jednotek pro rychlejší ukládání a čtení dat. Dle použitého levelu se liší zapisovací a čtecí rychlost a stupeň zabezpečení dat v případě poruchy jednoho ze zapojených disků. Vyšších rychlostí zapisování a čtení je dosahováno pomocí dělení dat na menší bloky a rozprostření jejich ukládání na více disků. O bezpečnost se

stará duplikování dat. Pokud tedy jeden z disků selže, data jsou uložena alespoň na jednom dalším. [31]

Technologie RAID se dělí na několik levelů, kde každý z nich má jinou funkci.

- RAID 0 - Data jsou rozdělována na segmenty, které jsou následně ukládány na disky což zvyšuje rychlost zapisování a čtení. Pro tento level je minimální počet zapojených disků 2. RAID 0 ale neobsahuje žádnou ochranu dat, pokud tedy jeden z disků selže všechna data jsou ztracena.
- RAID 1 – Data nejsou segmentována, ale ukládají se identicky na dva (nebo více) disky současně. Tento způsob zajišťuje vysokou odolnost vůči selhání hardwaru – pokud jeden z disků selže, systém může nadále fungovat díky kopii na druhém disku. Nevýhodou je, že efektivní kapacita pole odpovídá pouze kapacitě jednoho disku, protože všechny disky obsahují stejná data. Zápisová rychlost může být nižší než u RAID 0, protože data se musí zapsat na všechny disky. Naopak čtecí rychlost může být vyšší, protože je možné číst paralelně z více disků. RAID 1 je vhodný tam, kde je prioritou ochrana dat před ztrátou
- RAID 5 – RAID 5 stripuje data (rozděluje je na bloky a rozkládá je napříč více disky), podobně jako RAID 0, ale zároveň k těmto datům přidává paritní bloky, které umožňují rekonstrukci dat v případě selhání jednoho z disků. Parita se neukládá vždy na jeden konkrétní disk, ale je rovnoměrně rozložena mezi všechny disky v poli. Pokud dojde ke ztrátě jednoho disku, jeho obsah lze dopočítat pomocí zbylých dat a paritních informací (obvykle pomocí XOR operace). Při selhání dvou disků současně však dochází ke ztrátě všech dat. RAID 5 tak představuje kompromis mezi výkonem, ochranou dat a využitelnou kapacitou – efektivní úložná kapacita je rovna součtu kapacit všech disků mínus jeden disk pro paritu. Nevýhodou je snížená rychlost zápisu, protože parita se musí vždy přepočítat a zapsat. RAID 5 vyžaduje minimálně tři disky.
- RAID 10 - Level 10 je kombinací RAID 0 a RAID 1. Data jsou nejdříve segmentována a poté duplikována a ukládána na disky. Minimální počet disků pro tento level je 4 viz. obrázek. [32]



Obr. 2.7: RAID 10

3 Návrh a realizace vlastního řešení

Na základě zhodnocení možných architektonických přístupů uvedených v kapitole 2 byla zvolena varianta založená na použití běžně dostupného notebooku v kombinaci s externí síťovou kartou. Tato část práce se zabývá návrhem konkrétního řešení s důrazem na výběr vhodného hardwaru, softwarového prostředí a způsobu, jakým lze zajistit spolehlivý záznam síťového provozu v prostředí s omezenými prostředky. Popsány jsou jednotlivé kroky vedoucí k sestavení funkčního prototypu včetně provedených úprav operačního systému a zvolených vývojových nástrojů.

3.1 Výběr hardwarové platformy

S ohledem na dostupnost a flexibilitu při vývoji byla zvolena hardwarová platforma založená na běžném notebooku doplněném o externí síťovou kartu. Toto řešení umožňuje snadné testování, úpravy i přenositelnost a zároveň poskytuje dostatečný výpočetní výkon pro záznam více-gigabitového síťového provozu. V této části je popsán konkrétní použitý hardware a úpravy, které byly nutné pro jeho plnohodnotné využití v navržené architektuře.

3.1.1 Notebook jako základ hardwarového řešení

Jako základ pro návrh záznamového zařízení byl zvolen notebook Lenovo IdeaPad Gaming 3 (15ACH6) vybavený procesorem AMD Ryzen 5 5600H a 16 GB DDR4 operační paměti. Toto zařízení bylo vybráno z důvodu dostatečné kapacity operační paměti, přítomnosti volného slotu pro NVMe SSD disk a dostupného rozhraní USB-C, které umožňuje připojení externí síťové karty potřebné pro zachytávání více-gigabitového síťového provozu. [40]

3.1.2 Výběr a připojení síťové karty

Vzhledem k tomu, že integrovaná síťová karta zvoleného notebooku Lenovo podporuje maximálně 1 Gbps, bylo pro účely více-gigabitového záznamu nutné použít výkonnější síťové rozhraní. K tomuto účelu byla zvolena externí 2,5 GbE síťová karta Axagon (ADE-25RC) připojitelná přes rozhraní USB-C 3.2 Gen 1. Toto řešení umožňuje dosažení vyšší přenosové rychlosti potřebné pro záznam více-gigabitového síťového provozu. [54]

3.1.3 Výběr a připojení datového disku

Jako úložné médium byl zvolen disk Samsung 990 EVO Plus ve formátu NVMe, který nabízí dostatečnou rychlost zápisu a zároveň dostatečnou kapacitu pro instalaci operačního systému i uchovávání datových záznamů z delších časových úseků. V notebooku Lenovo se nachází jeden volný slot typu PCIe 3.0 M.2 M-key 2280, do něhož lze tento disk připojit. Ačkoliv disk podporuje rozhraní PCIe 4.0, standard je zpětně kompatibilní, takže i při provozu na sběrnici PCIe 3.0 je dosažitelná rychlost zápisu dostatečná pro záznam více-gigabitového síťového provozu. [55]

3.2 Výběr operačního systému a jeho úprava

Jako operační systém pro záznam multi-gigabit Ethernet toku byl v této práci zvolen AlmaLinux – stabilní a spolehlivá distribuce založená na Red Hat Enterprise Linux (RHEL), která je vhodná pro produkční nasazení a umožňuje detailní konfiguraci systému včetně síťových a kernelových parametrů.

3.2.1 AlmaLinux

AlmaLinux je komunitou vyvíjený operační systém, který vznikl jako binárně kompatibilní náhrada za Red Hat Enterprise Linux (RHEL). Jeho vývoj byl motivován zejména ukončením podpory distribuce CentOS Linux, která byla oblíbenou bezplatnou alternativou k RHEL pro serverové nasazení. AlmaLinux je podporován především společností CloudLinux Inc. a dalšími více než 25 sponzory z oblasti enterprise softwaru.

Distribuce AlmaLinux využívá standardní linuxové jádro s optimalizacemi převzatými z RHEL a je poměrně „lehká“ – při instalaci bez grafického uživatelského rozhraní vykazuje nízkou paměťovou i výpočetní režii. Typická spotřeba paměti u čisté serverové instalace se pohybuje pod hranicí 200 MB RAM, což je příznivé pro úlohy náročné na přenos a zpracování síťového provozu.

Pro záznam síťového toku o rychlosti vyšší než 2,5 Gbps však není rozhodující pouze nízká systémová režie, ale zejména možnost úprav síťových parametrů jádra (např. `net.core.rmem_max`) a podpora efektivních zero-copy mechanismů pro příjem dat, jako je například `AF_PACKET`.

V porovnání s distribucí Ubuntu Server nabízí AlmaLinux nižší výchozí paměťovou zátěž, avšak Ubuntu má výhodu ve své rozsáhlé komunitě a lepší dokumentaci, což může být užitečné při ladění síťových nastavení. Ubuntu může být částečně lépe přednastaveno pro vysokorychlostní záznam, nicméně za cenu mírně vyšší spotřeby systémových prostředků.

Další alternativou je samotný Red Hat Enterprise Linux, který je s AlmaLinuxem prakticky identický z hlediska výkonu i jádra. Má sice mírně vyšší paměťovou náročnost, ale díky oficiální technické podpoře a dlouhodobé stabilitě se jedná o výbornou volbu pro záznam multi-gigabitového provozu v produkčním nasazení. Je však nutné zmínit, že se jedná o placený operační systém s uzavřeným licenčním modelem, což může představovat překážku pro použití v nízkonákladových nebo experimentálních řešeních. [65]

3.2.2 Úprava jádra a síťových parametrů

Jádro operačního systému, označované jako kernel, představuje klíčovou komponentu zajišťující komunikaci mezi hardwarem a softwarem počítače. V případě Linuxu se jedná o tzv. monolitické jádro, které běží v jednotném adresním prostoru a integruje širokou škálu funkcí, jako je správa procesů, paměti, zařízení, síťových operací nebo souborových systémů. Linuxové jádro plní úlohu správce systémových prostředků – přiděluje procesorový čas, alokuje paměť, řídí vstupně-výstupní operace a zajišťuje bezpečný přístup k zařízením i datům. Díky své modularitě lze do jádra dynamicky přidávat moduly (např. ovladače) bez nutnosti restartu systému.

Po instalaci operačního systému bylo nejprve nutné ověřit, zda je systém aktualizován na nejnovější verze všech balíčků a repozitářů, aby byla zajištěna maximální kompatibilita a stabilita. K tomu slouží příkaz:

```
sudo dnf update -y
```

Po dokončení aktualizace je nutné nově načíst všechna aktuální systémová nastavení do konfiguračních souborů, což je provedeno pomocí:

```
sudo sysctl --system
```

Distribuce *AlmaLinux* standardně obsahuje nástroj *tuned*, který slouží k automatické optimalizaci výkonu systému. Pro ověření jeho přítomnosti a případné doplnění byl použit příkaz:

```
sudo dnf install kernel-core tuned -y
```

Jak se očekávalo, systém oznámil, že všechny balíčky jsou již nainstalovány. Aby bylo zajištěno, že se optimalizované jádro načte i po restartu, byl proveden obnovovací zápis do zaváděcího zavaděče:

```
sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

Následně bylo přistoupeno k úpravě síťových parametrů jádra s cílem optimalizovat systém pro vysokorychlostní záznam síťového provozu. Tyto úpravy byly provedeny vytvořením nového konfiguračního souboru ve složce `/etc/sysctl.d/` s následujícími klíčovými parametry:

- `net.core.rmem_max` – maximální velikost bufferu pro příjem dat.
- `net.core.wmem_max` – maximální velikost bufferu pro odesílání.
- `net.core.rmem_default` – výchozí velikost přijímacího bufferu, pokud není přepsána aplikací.
- `net.core.wmem_default` – výchozí velikost výstupního bufferu.
- `net.core.netdev_max_backlog` – maximální počet paketů ve frontě síťového zařízení (NIC), než je zpracuje jádro.
- `net.ipv4.udp_mem = 2097152 4194304 8388608` – tříúrovňová hranice určující, kolik paměti může být celkově přiděleno jádrem pro všechny UDP sockety:
 - Minimální (pressure threshold): 2 GB – začíná se omezovat alokace.
 - Defaultní (soft limit): 4 GB – preferovaná mez.
 - Maximální (hard limit): 8 GB – žádná další alokace není povolena.
- `net.ipv4.udp_rmem_min` – minimální velikost přijímacího bufferu pro jednotlivý UDP socket.
- `net.ipv4.udp_wmem_min` – minimální velikost výstupního bufferu pro jednotlivý UDP socket.

```
net.core.rmem_max = 33554432
net.core.wmem_max = 33554432
net.core.rmem_default = 33554432
net.core.wmem_default = 33554432
net.core.netdev_max_backlog = 100000
net.ipv4.udp_mem = 2097152 4194304 8388608
net.ipv4.udp_rmem_min = 16384
net.ipv4.udp_wmem_min = 16384
```

Obr. 3.1: Nastavené hodnoty systémových bufferů

Nastavené hodnoty všech těchto bufferů jsou vidět na obrázku 3.1. Cílem těchto nastavení je zabránit ztrátám dat při vysokém síťovém toku, předejít přeplnění systémových bufferů a zvýšit stabilitu a výkonnost aplikace, která příchozí síťový provoz zpracovává. Po uložení upravených hodnot do konfiguračního souboru je nutné tyto nové parametry zapsat do běžícího systému. To se provede opětovným spuštěním příkazu:

```
sudo sysctl --system
```

Tím se načtou všechna nastavení definovaná ve výchozích i uživatelských souborech ve složce `/etc/sysctl.d/` a aktivují se bez nutnosti restartu.

Pro samotný vývoj záznamové aplikace je dále nezbytné nainstalovat skupinu balíčků *Development Tools*, která obsahuje základní nástroje a knihovny potřebné pro kompilaci a ladění aplikací v jazyce C/C++. Instalace se provede pomocí následujícího příkazu:

```
sudo dnf groupinstall "Development Tools"
```

3.2.3 Nastavení externí síťové karty

Pro správné fungování sítě při práci s jumbo rámci bylo nutné nastavit hodnotu MTU (Maximum Transmission Unit) externí síťové karty na 9000 bajtů. Toho lze dosáhnout pomocí několika kroků s využitím nástroje *nmcli*.

Nejprve je třeba zobrazit seznam dostupných síťových připojení a identifikovat správné zařízení:

```
nmcli connection show
```

Výstup může vypadat například takto:

```
Drátové připojení 1 8d253568-b1bc-4e8a-bdca-654e90060b7c ethernet  
enp6s0f3u1
```

Po nalezení správného síťového zařízení je možné změnit jeho hodnotu MTU na 9 000 pomocí následujícího příkazu:

```
nmcli connection modify "Drátové připojení 1" 802-3-ethernet.mtu 9000
```

Aby se změna projevila, je nutné dané připojení restartovat:

```
nmcli connection down "Drátové připojení 1" && nmcli connection up  
"Drátové připojení 1"
```

Pro ověření, že byla hodnota nastavena správně, lze využít příkaz:

```
ip link show enp6s0f3u1
```

V jeho výstupu by se měla nacházet položka `mtu 9000`, která potvrzuje úspěšné nastavení.

3.3 Vývoj vlastní aplikace

Pro záznam síťového provozu je možné využít jak volně dostupné, tak i komerční nástroje, mezi které patří například *tcpdump*, *Wireshark*, *n2disk* nebo vlastní aplikace využívající *AF_PACKET* sockety v operačním systému Linux. Přestože některé z těchto nástrojů, zejména ve spojení se specializovaným hardwarem (např. Napa-tech NIC), umožňují zachytávání provozu i při rychlostech přes 30 Gbps, ve většině případů – zejména na běžném hardwaru – narážejí na omezení způsobená použitím knihovny *libpcap*, která neumožňuje efektivní memory-mapped přístup k datům.

Nástroje jako *tcpdump* nebo *Wireshark* zpravidla dosahují maximální spolehlivé propustnosti v rozsahu 1–5 Gbps, přičemž jsou náchylné ke ztrátám paketů při vyšší zátěži. Pokud má být nástroj jako *tcpdump* použit při vyšších rychlostech, je nutné upravit jeho chování například využitím kruhového bufferu přes *PACKET_RX_RING*, který je součástí rozhraní *AF_PACKET*. V takovém případě však začíná dávat větší smysl vyvinout samostatnou aplikaci, která tento mechanismus využívá přímo a naplno. Vlastní řešení poskytuje nejen vyšší výkon a efektivitu, ale také plnou kontrolu nad funkcionalitou aplikace a možnost snadno doplnit další záznamové nebo analytické prvky podle konkrétních požadavků.

Kromě technických omezení je vhodné zohlednit i licenční aspekty. Například *Wireshark* je šířen pod licencí GNU GPLv2, která v případě jakýchkoli úprav vyžaduje, aby byl modifikovaný zdrojový kód při dalším šíření veřejně dostupný. To může být v některých komerčních scénářích nevhodné, například pokud úpravy obsahují unikátní know-how nebo specifické techniky zpracování paketů, které by si vývojář nepřál sdílet s konkurencí. [50, 51]

3.3.1 Optimalizace tcpdump pomocí Gulp a RX_RING

Při záznamu síťového provozu ve vysokých přenosových rychlostech může nástroj *tcpdump* trpět ztrátovostí paketů a nedostatečnou propustností. Pro zvýšení jeho výkonu se využívají dvě hlavní optimalizační techniky: nástroj Gulp pro efektivní přesměrování dat mezi procesy a mechanismus *RX_RING* pro nízkoúrovňovou práci s pamětí.

Gulp slouží jako prostředník mezi nástroji typu *dumpcap* a *tshark*, nahrazuje běžné unixové roury a umožňuje plynulejší přenos datového proudu. Využívá vlastní kruhový buffer, který minimalizuje ztráty způsobené zahlcením meziprocesové komunikace. Mezi jeho výhody patří vyšší efektivita ve srovnání s běžnými rourami a nižší ztrátovost při přenosu dat mezi nástroji. Nevýhodou je pevně daná velikost interního bufferu, který může přetéct při dlouhodobé zátěži, a také omezené možnosti konfigurace, neboť dostupná dokumentace pro přizpůsobení bufferu je velmi

omezená. Gulp se používá například následovně:

```
dumpcap -q -i lo -w - | gulp -c > /tmp/pipe  
tshark -r - < /tmp/pipe
```

Druhou metodou je využití mechanismu *RX_RING*, který v systému Linux umožňuje přímý přístup k síťovým paketům prostřednictvím paměťového mapování. V nástroji *tcpdump* lze tento přístup aktivovat například pomocí parametru *-B*, kterým se zvětší velikost vyhrazené paměti pro příchozí pakety. Zvýšení velikosti bufferu například na 4 *MiB* snižuje pravděpodobnost ztráty paketů v jádře, protože zpracovávající aplikace má více času na jejich zápis na disk. Použití v praxi může vypadat následovně:

```
tcpdump -B 4096 -i eth0 -w capture.pcap
```

Mezi výhody této metody patří přímé ladění na úrovni jádra a vysoká efektivita. Nevýhodou je nutnost běhu s oprávněním správce a omezená flexibilita nástroje *tcpdump*, který tento mechanismus využívá jen částečně. [56]

Z důvodu, že efektivní zpracování paketů v *tcpdump* při vyšší zátěži je možné pouze při využití nízkoúrovňových technik jako *RX_RING*, byla zvolena cesta implementace vlastní aplikace, která tento přístup využívá přímo a naplno. Tím je dosaženo vyšší výkonnosti a nižší ztrátovosti při více-gigabitových rychlostech. Zároveň takové řešení poskytuje větší kontrolu nad vnitřním chováním aplikace, možnost snadného rozšíření o další funkce (například vlastní timestampování, filtrování nebo deduplikaci) a lepší možnosti ladění a optimalizace v různých provozních podmínkách. [57]

3.3.2 Použité technologie

Socket typu *AF_PACKET* umožňuje v systému Linux zachytávat síťové rámce již na úrovni linkové vrstvy (Layer 2 dle modelu OSI), a to včetně všech hlaviček Ethernet rámce. Tento přístup je výhodný zejména v případech, kdy je potřeba detailní práce se síťovým provozem – například při detekci anomálií, vývoji monitorovacích nástrojů nebo při vytváření vlastní záznamové infrastruktury. Na rozdíl od běžně používaných socketů typu *AF_INET*, které pracují až na úrovni transportní vrstvy (např. TCP/UDP), umožňuje *AF_PACKET* zachytit i ne-IP provoz, například protokol ARP.

Sockety typu *AF_INET* nebyly pro tuto aplikaci použity z několika důvodů. Prvním je vyšší režie způsobená nutností průchodu paketů přes síťový stack jádra až na úroveň transportní vrstvy, což má negativní dopad na výkon a latenci aplikace. Druhým problémem je omezená podpora pro práci s jumbo rámci, které mohou

být kvůli fragmentaci neúplně zachyceny. Třetím a zásadním omezením je skutečnost, že sockety typu *AF_INET* nezachytávají kompletní Ethernetové hlavičky, což znemožňuje analýzu rámců na úrovni linkové vrstvy.

Z těchto důvodů bylo jako záznamové rozhraní zvoleno řešení postavené na *AF_PACKET* socketu v kombinaci s typem *SOCK_RAW*, které umožňuje efektivní, nízkoúrovňový a přesný záznam síťového provozu včetně všech potřebných metadat.

3.3.3 Struktura a funkce aplikace

Aplikace slouží k efektivnímu záznamu síťového provozu na vybraném síťovém rozhraní pomocí nízkoúrovňového socketu typu *AF_PACKET*. Využitím režimu *PACKET_RX_RING* a techniky *memory-mapped I/O* je dosaženo přímého přístupu k datům v paměťovém prostoru jádra, což eliminuje nutnost opakovaného volání systémových funkcí jako *recv()* a snižuje zatížení procesoru. Aplikace si postupně přebírá rámce z kruhového bufferu, kontroluje jejich dostupnost, provádí načtení a následný zápis dat do výstupního souboru ve formátu PCAP. Celý proces je řízen hlavní smyčkou, která se ukončí při zachycení signálu přerušení nebo při splnění jiné podmínky.

Během běhu programu je zároveň sledován počet úspěšně zpracovaných a zahozených paketů. Po ukončení běhu jsou vypočteny dodatečné metriky, jako je skutečná doba záznamu, celkový objem přenesených dat a průměrná přenosová rychlost. Výsledky těchto výpočtů slouží nejen k ověření funkčnosti aplikace, ale také k porovnání s jinými nástroji pro zachytávání paketů (např. tcpdump, Wireshark, n2disk). Díky použitému přístupu je aplikace vhodná pro testování výkonnosti záznamu provozu i ve vícenásobně gigabitových sítích, kde běžné nástroje selhávají kvůli přetížení nebo ztrátám paketů.

Před samotným spuštěním aplikace je nejprve nutné provést její překlad, k čemuž slouží nástroj *gcc*, který je součástí nainstalovaného balíčku *Development Tools*. Překlad zdrojového souboru probíhá pomocí příkazu:

```
gcc -o název_aplikace název_aplikace.c
```

Příkaz je třeba spustit v terminálu v adresáři, kde se nachází příslušný zdrojový kód aplikace. Výsledkem je vytvoření spustitelného souboru se zadaným názvem.

Samotné spuštění aplikace se poté provádí příkazem:

```
sudo ./název_aplikace název_síťového_rozhraní  
cesta_a_název_výstupního_souboru
```

Použití předpony `sudo` je v tomto případě nezbytné, neboť aplikace pracuje s `AF_PACKET` socketem, jehož používání vyžaduje administrátorská oprávnění. Bez `sudo` by aplikace neměla přístup k síťovému rozhraní na potřebné úrovni a nebyla by schopna zachytávat provoz.

3.3.4 Rozbor zdrojového kódu

Bezprostředně po načtení všech potřebných headerů je v programu definována velikost bufferů, které budou později využity při nastavení kruhového bufferu `RX_RING`. Dále je deklarována globální proměnná `running`, která slouží pro signalizaci ukončení běhu aplikace. Její hodnota je nastavena na 0 ve funkci `handle_sigint()`, která se aktivuje při přijetí signálu ukončení (např. `SIGINT`) a umožňuje tak bezpečné ukončení hlavní smyčky.

Následuje definice struktury hlavičky výstupního PCAP souboru, která obsahuje veškeré náležitosti specifikované v kapitole 1.3.1. Podobně je poté definována i struktura hlavičky jednotlivých paketů. Poté následuje vstup do hlavní funkce programu, kde jsou zpracovávány argumenty předané při spuštění. První argument představuje název síťového zařízení ve formátu, který lze zjistit například příkazem `ip link show` (např. `enp6s0f3u1`). Druhý argument slouží k zadání výstupní cesty a názvu souboru, např. `/home/Packet_Capture/captured.pcap`. Oba argumenty jsou uloženy do proměnných pro další použití.

Dále dochází k vytvoření `AF_PACKET RAW` socketu, který umožňuje zachytávat všechny Ethernet rámce (`ETH_P_ALL`). Následuje získání číselného indexu síťového rozhraní pomocí systémového volání `ioctl()` s parametrem `SIOCGIFINDEX`, které pracuje se strukturou `ifreq`. Tato hodnota je nezbytná pro správné nastavení adresy v další struktuře `sockaddr_ll`, která bude použita při volání funkce `bind()`. Struktura `tpacket_stats` je připravena pro pozdější čtení statistik o zachycených a zahozených paketech přímo z jádra systému.

Konfigurace `sockaddr_ll` slouží k nastavení síťového rozhraní, ke kterému má být socket připojen. Vlastní připojení je realizováno voláním `bind()`. Následuje nastavení parametrů kruhového bufferu `RX_RING`, kam se ukládají přijaté pakety. Pomocí indexů aplikace prochází buffer tak, že přečte paket, uloží jej do souboru, označí pozici v bufferu jako volnou a přejde na další rámec. Konkrétní hodnoty parametrů bufferu (`BLOCK_SIZE`, `FRAME_SIZE`, `FRAME_COUNT`) jsou definovány již na začátku programu.

Poté je buffer aktivován a pomocí funkce `mmap()` je mapován do paměti. Tato funkce definuje, že buffer má být přístupný pro čtení a zápis, změny budou sdíleny s jádrem a jako deskriptor musí být použit socket, na kterém byl buffer aktivován. Offset je nastaven na nulu, takže se mapování provádí od začátku alokovaného

prostoru.

Dalším krokem je vytvoření a otevření výstupního souboru na cestě zadané druhým argumentem. Následně je vytvořena hlavička PCAP souboru a zapsána na začátek výstupního souboru.

Následuje nastavení obsluhy signálů ukončení — například při stisknutí *Ctrl+C* nebo při automatickém ukončení po vypršení časového limitu (*timeout*). Při jejich zachycení je nastavena hodnota proměnné *running* na 0, čímž se hlavní smyčka ukončí a dojde k bezpečnému ukončení programu.

Poslední připravenou strukturou před vstupem do hlavní smyčky *while()* je proměnná využívaná k obcházení kruhového bufferu *RX_RING*, která slouží k iteraci přes jednotlivé rámce bufferu při jejich zpracování.

Bezprostředně před spuštěním hlavní smyčky je do terminálu vypsána základní informace o tom, na jakém síťovém rozhraní bude aplikace zachytávat provoz a kam bude výstupní PCAP soubor ukládán. Tím jsou zobrazeny hodnoty zadané uživatelem při spuštění programu prostřednictvím argumentů. Tento výpis slouží jako přehledné ověření správnosti parametrů ještě před zahájením záznamu.

Na počátku **hlavní nekonečné smyčky** se nejprve získá ukazatel na aktuální rámec v kruhovém bufferu *RX_RING*. Následně je připravena struktura, která specifikuje socket, na němž má aplikace čekat na příchozí data.

Poté se kontroluje, zda je aktuální rámec připraven pro zpracování. Pokud ano, podmínka přechází rovnou na zpracování paketu. Pokud nikoliv, je aktivováno čekání pomocí funkce *poll()*, která zablokuje provádění až do okamžiku, kdy jsou k dispozici nová data. Pokud během této doby (maximálně 100 *ms*) žádný paket nepřijde, funkce vrátí hodnotu -1 a provede se další iterace smyčky od jejího začátku.

Jakmile je paket připraven, získá se aktuální čas pro měření délky záznamu. Do proměnné *first_ts_sec* se uloží čas prvního paketu (pouze pokud je hodnota počítadla *packet_count* rovna nule), zatímco čas každého dalšího paketu se ukládá do proměnné *last_ts_sec*. Rozdílem těchto hodnot je později spočten celkový čas zápisu.

Následně je vytvořena hlavička PCAP záznamu pro konkrétní paket a spolu s daty paketu je zapsána do výstupního souboru. Pro účely měření propustnosti je velikost paketu přičtena k proměnné *total_bytes* a současně je inkrementována hodnota *packet_count*.

Po úspěšném zpracování je aktuální rámec označen jako znovu dostupný pro kernel, čímž je umožněno jeho přepsání dalším příchozím paketem. Index v rámci kruhového bufferu je posunut na další pozici a smyčka pokračuje znovu od začátku.

Po **ukončení hlavní smyčky** dochází k uvolnění alokované paměti kruhového bufferu pomocí funkce *munmap()*, která odmapuje dříve mapovanou oblast paměti.

Následně je uzavřen výstupní PCAP soubor a aplikace ukončí připojení k síťovému rozhraní uzavřením socketu prostřednictvím volání `close(sock)`;

Pro **porovnání s ostatními možnostmi** záznamu byla do aplikace přidána funkcionality pro základní statistické vyhodnocení úspěšnosti zachytávání paketů. Informace o počtu přijatých a zahozených paketů jsou získávány prostřednictvím systémového volání `getsockopt()` s parametrem `PACKET_STATISTICS`, které poskytuje přístup ke statistikám jádra pro daný `AF_PACKET` socket.

Pomocí této funkce je naplněna struktura `tpacket_stats`, která obsahuje dvě klíčové hodnoty:

- `tp_packets` – počet úspěšně zachycených paketů
- `tp_drops` – počet paketů, které byly zahozeny, například z důvodu nedostatečné kapacity přijímacího bufferu

Tyto hodnoty jsou zobrazeny pod nadpisem *Statistika zachytávání (kernel)*, což umožňuje jednoduché a rychlé vyhodnocení, zda během provozu docházelo ke ztrátám.

Do programu byla dále implementována funkcionality pro výpočet a výpis dodatečných statistik týkajících se samotného procesu záznamu. Zatímco předchozí statistika získaná z jádra systému pomocí `PACKET_STATISTICS` poskytovala pouze počet zachycených a zahozených paketů na úrovni kernelu, nově přidaná část programu umožňuje zjistit také skutečný počet úspěšně zapsaných paketů do výstupního souboru, reálný čas trvání měření, velikost zachycených dat a výpočet průměrné rychlosti (propustnosti) aplikace.

Reálný čas měření je odvozen z proměnných `first_ts_sec` a `last_ts_sec`, do kterých se ukládají časové značky prvního a posledního zapsaného paketu. Odečtením těchto hodnot je získán celkový čas zachytávání. Ačkoli byla aplikace nastavena na 20-sekundový běh, samotné zapisování paketů do souboru probíhalo přibližně po dobu 19 sekund. Tento rozdíl je způsoben prodlevou způsobenou inicializací a spuštěním aplikace.

Počet reálně zapsaných paketů je počítán pomocí proměnné `packet_count`, která je inkrementována v hlavní smyčce `while()`, a to po každém úspěšném zápisu paketu do souboru. Tím je zajištěno, že se do statistik promítají pouze skutečně uložené pakety.

Velikost přenesených dat je sledována pomocí proměnné `total_bytes`, do níž se po každém zápisu přičítá velikost datového obsahu příslušného paketu. Z této hodnoty je následně při výpisu odvozena celková velikost v megabajtech. Kombinací této hodnoty a doby trvání měření je nakonec vypočtena a do terminálu vypsána i průměrná přenosová rychlost aplikace, která slouží jako přímý ukazatel její výkonnosti v reálných podmínkách.

3.3.5 Přiložený zdrojový kód

Kompletní implementace aplikace pro záznam síťového provozu je součástí elektronické přílohy (složka `Kód aplikace` soubor `packet_capture_pcap.c`). Obsahuje plně funkční zdrojový kód napsaný v jazyce C. Kód byl testován na distribuci AlmaLinux 9.3 s kompilátorem `gcc` verze 11.2.0.

4 Testování a ladění výkonu

Operační systém byl úspěšně připraven a nakonfigurován pro potřeby záznamu multi-gigabitového síťového provozu. K zařízení je připojena externí síťová karta, jejíž hodnota MTU byla nastavena na 9 000, aby bylo umožněno zpracování paketů typu jumbo frames. Byla vyvinuta vlastní záznamová aplikace, navržená pro vysokou propustnost a efektivní využití dostupného síťového hardwaru.

V této kapitole bude popsán postup testování této aplikace, včetně způsobu generování síťového provozu, který má být zachytáván. Dále budou provedeny zátěžové testy při různých přenosových rychlostech, porovnání výkonu aplikace s dostupnými nástroji a případné optimalizace podle dosažených výsledků.

4.1 Generování datového provozu

Pro účely testování vyvíjené aplikace je nezbytné zajistit generování specifikovaného síťového toku, který umožní měřit rychlost zpracování paketů, ztrátovost při jejich zachytávání a současně poskytne možnost porovnání s dalšími dostupnými nástroji určenými pro zachytávání síťového provozu. Takto vytvořené testovací podmínky slouží k objektivnímu zhodnocení výkonnosti aplikace, její spolehlivosti při práci s vysokorychlostním síťovým tokem a k identifikaci případných slabých míst v implementaci.

4.1.1 Python skript

Python nabízí jednoduchý způsob, jak vytvořit definovaný tok UDP paketů a odeslat jej na síťové rozhraní pomocí zadané IP adresy. Takto vytvořený program může obsahovat implementaci počítadla odeslaných paketů a sledování jejich celkového objemu, což je vhodné pro porovnání s daty zachycenými aplikací.

Vlastní skript je postaven na několika základních částech:

- třetí řádek kódu vytváří socket typu *AF_INET*, *SOCK_DGRAM*, určený pro odesílání nebo příjem UDP paketů,
- následující řádek definuje cílovou IP adresu a port, kam budou pakety směřovat,
- další řádek nastavuje obsah a velikost odesílaných paketů,
- poté je inicializováno měření času pomocí proměnných pro začátek a konec časového intervalu (10 *sekund*),
- je definována proměnná pro průběžné sčítání velikosti odeslaných dat,
- ve smyčce pak skript odesílá pakety dokud neuplyne definovaný čas,

- nakonec je provedeno vyčíslení a výpis celkového objemu odeslaných dat za daný časový úsek.

Výpis 4.1: Jednoduchý UDP generátor provozu

```
import socket, time

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
addr = ("127.0.0.1", 5001)
packet = b"x" * 1472

start = time.time()
sent = 0

while time.time() - start < 10:
    sock.sendto(packet, addr)
    sent += len(packet)

print(f"Sent{sent/(1024*1024):.2f} MB in 10s")
```

Při testování tohoto řešení se však objevil problém spojený s použitím externí síťové karty Realtek, která nepodporuje zpětný přenos (loopback) dat na fyzické vrstvě. Pokud je adresa příjemce nastavena na "127.0.0.1", dochází ke zpracování dat výhradně v rámci interní smyčky systému (loopback interface) a paket vůbec neopustí síťový stack operačního systému. V kombinaci s Realtek síťovou kartou tak může dojít k tomu, že pakety určené na localhost nejsou zachyceny externím rozhraním a test nedává očekávané výsledky.

Tento problém by bylo možné vyřešit použitím druhého zařízení vybaveného další 2,5 *GbE* síťovou kartou, na které by byl UDP provoz směrován. Jednalo by se však o logisticky i technicky náročnější řešení, zvláště pokud je již k dispozici zařízení, jako je MikroTik RB5009, které obsahuje integrovanou funkci traffic generatoru (podrobněji popsáno v kapitole 4.1.3) a umožňuje jednoduše vytvářet skutečný fyzický síťový provoz bez potřeby dalšího počítače.

4.1.2 Scapy

Scapy je nástroj určený pro generování definovaného síťového provozu. Přestože je velmi flexibilní a snadno použitelný, jeho schopnosti v oblasti rychlosti přenosu jsou omezené. Podle dostupné dokumentace a praktických měření není Scapy vhodné pro testování aplikací zaměřených na multi-gigabitový záznam síťového provozu, jelikož jeho maximální propustnost se pohybuje přibližně kolem 1 *Gbps*.

4.1.3 Mikrotik RB5009

MikroTik RB5009 je výkonný a kompaktní router vhodný pro domácí i laboratorní použití. Díky robustnímu pasivnímu chlazení je zajištěn stabilní provoz i při vyšší zátěži. Zařízení nabízí tři různé možnosti napájení: prostřednictvím kteréhokoliv ethernetového portu (PoE-in), přes samostatný napájecí konektor, nebo pomocí svorkovnice umístěné na boční straně. Podporované napájecí stejnosměrné napětí se pohybuje v rozsahu 24–57 V.

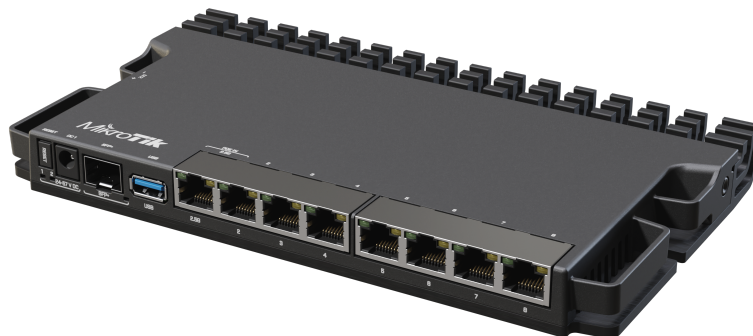
Router je vybaven osmi ethernetovými porty – sedmi porty s rychlostí 1 *Gbps* a jedním portem s rychlostí 2,5 *Gbps*. Dále obsahuje jeden SFP+ slot umožňující připojení optických modulů s rychlostí až 10 *Gbps* a jeden port USB 3.0. Díky montážním otvorům na bocích zařízení je možné RB5009 jednoduše instalovat do rackových skříní.[39, 38]

Pro konfiguraci zařízení MikroTik tak, aby generovalo uživatelem definované pakety, je možné využít webové rozhraní, které umožňuje základní a jednodušší nastavení přes běžný internetový prohlížeč. Pro pokročilejší a specifické konfigurace je však doporučeno použít nástroj Winbox. Tento program poskytuje podrobnější přístup k nastavení routeru prostřednictvím přehledného grafického rozhraní. Jelikož je Winbox určen primárně pro operační systém Windows, je pro jeho spuštění na Linuxu nutné využít nástroj Wine, který slouží jako kompatibilní vrstva umožňující provozování aplikací vyvinutých pro Windows na unixových systémech.

Wine je speciální aplikace (přesněji softwarová vrstva), která umožňuje spouštět programy určené pro Windows na jiných operačních systémech, hlavně na Linuxu a macOS. Název "Wine" je zkratka pro "Wine Is Not an Emulator" – což je takový vtip, protože Wine technicky nenapodobuje celý Windows systém jako emulátor, ale převádí volání Windows API přímo na nativní volání operačního systému, na kterém běží. Díky tomu je Wine často rychlejší než klasické emulátory nebo virtuální stroje.

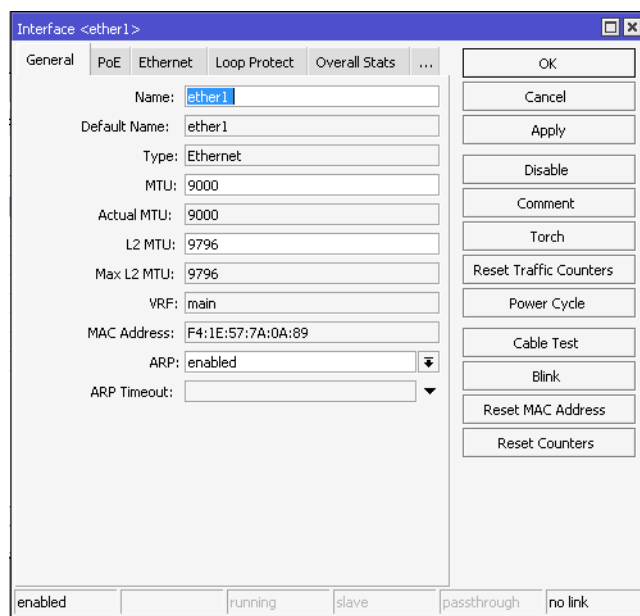
Zařízení RB5009 bylo vybráno proto, že umožňuje generování síťového provozu včetně paketů o velikosti odpovídající jumbo rámcům. Díky podpoře rychlostí až 10 *Gbps* je možné vytvořit dostatečně velký tok paketů pro testování aplikace. Tento přístup zároveň odstraňuje nutnost zatěžovat hardware, na kterém samotná aplikace běží, vytvářením síťového provozu. Navíc se tím vyřešil problém s odesíláním paketů na externí síťovou kartu, který se objevil při generování toku pomocí Python skriptu.

Pomocí aplikace Winbox bylo provedeno základní nastavení zařízení MikroTik tak, aby bylo schopné generovat požadovaný síťový tok na prvním ethernetovém portu. Tento port je – vedle SFP+ slotu – jediným rozhraním podporujícím přenosové rychlosti vyšší než 1 *Gbps*. Klíčovým krokem konfigurace bylo navýšení hodnoty MTU (Maximum Transmission Unit), která definuje maximální velikost datového rámce přenášeného bez fragmentace.



Obr. 4.1: Mikrotik RB5009[38]

Vzhledem k požadavku na generování odchozího provozu ve formátu jumbo rámců byla výchozí hodnota MTU (1 500 *bajtů*) navýšena na 9 000 *bajtů*, což odpovídá běžně používané hodnotě pro přenos jumbo rámců v prostředí vysokorychlostních sítí.



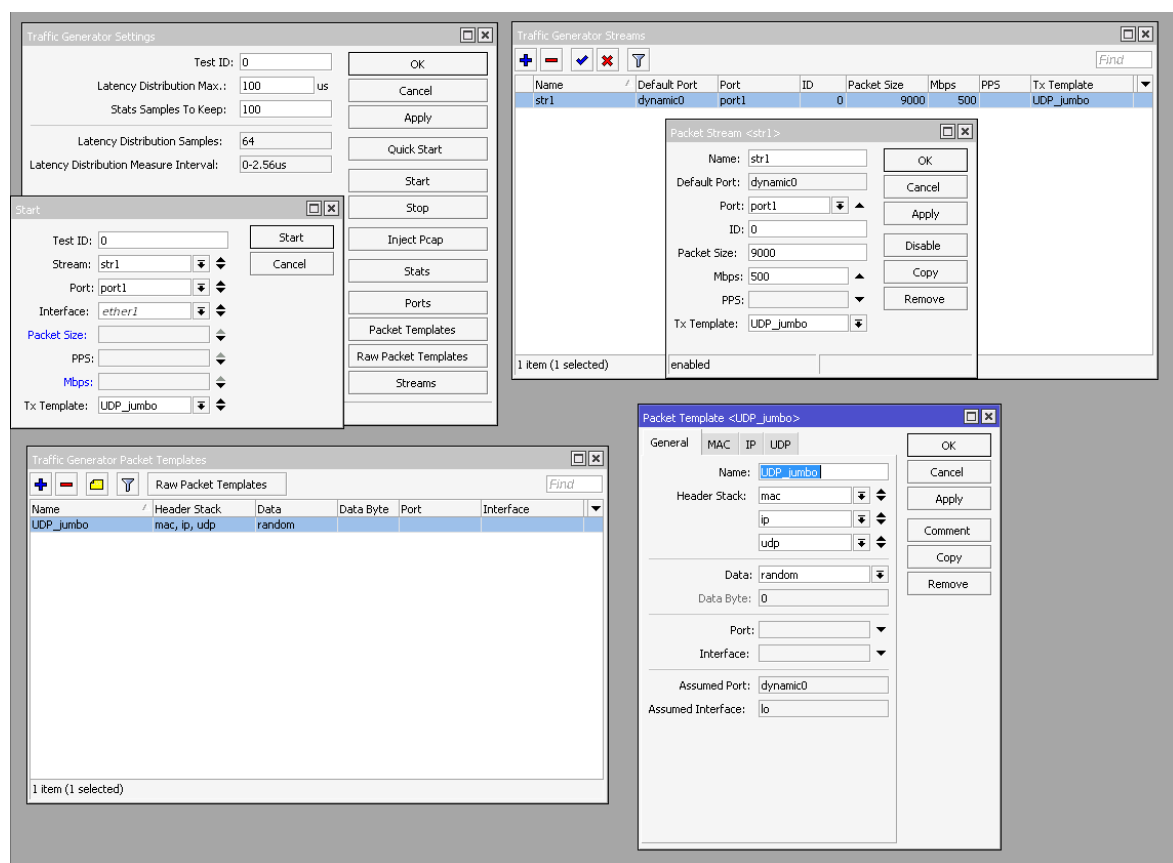
Obr. 4.2: Nastavení hodnoty MTU pro 2.5 Gb port

Následně bylo provedeno nastavení vestavěného nástroje Traffic Generator v roz-

hraní Winbox. Prvním krokem bylo vytvoření šablony paketů (packet template), která definovala základní parametry požadovaného síťového provozu. Šablona specifikovala cílovou MAC adresu, hlavičku IP protokolu a transportní protokol UDP. Jelikož obsah přenášených dat nebyl pro účely testování důležitý, byla zvolena volba generování náhodných dat (random payload).

Pro správnou funkci bylo dále nutné vytvořit datový proud (stream), v rámci kterého se definovala velikost generovaných paketů a požadovaná přenosová rychlost. Velikost paketu byla nastavena na 9 000 *bajtů* (odpovídající jumbo rámcům) a rychlost přenosu byla pro první měření stanovena na 500 *Mbps*. Jako šablona pro tento stream byl použit právě vytvořený template s názvem UDP_jumbo.

Pro samotné spuštění generování provozu byl vybrán port ether1, tedy první ethernetový port zařízení podporující přenosovou rychlost 2,5 *Gbps*. Počet generovaných paketů za sekundu nebyl specifikován, neboť celková přenosová rychlost byla určena velikostí paketů a požadovaným datovým tokem (500 *Mbps*). Kompletní konfigurace je znázorněna na příloženém obrázku.



Obr. 4.3: Nastavení traffic generatoru

4.1.4 Další možnosti

Dalšími možnostmi pro generování síťového provozu jsou nástroje jako *iperf*, *pktgen* nebo *MoonGen*. Nástroj *iperf* umožňuje generování UDP provozu s přenosovou rychlostí přibližně 2,3 Gbps. Nástroj *pktgen* je součástí linuxového jádra a podporuje i vyšší rychlosti, podobně jako *MoonGen*, který je určen pro přesné a časově kontrolované generování paketů.

Všechny tyto nástroje však sdílejí společný problém: generované pakety jsou odesílány virtuálně a je nutné specifikovat cílové rozhraní, kam mají být směrovány. Socket typu *AF_PACKET*, který aplikace používá pro záznam, vyžaduje přímý přístup k fyzickému rozhraní, a tudíž není možné zpracovávat provoz vytvářený čistě virtuálně bez skutečného hardwarového přenosu.

V testovacím prostředí byla použita externí síťová karta Realtek s podporou 2,5 GbE, která však nepodporuje funkci smyčky (loopback). V rámci dostupných možností se rovněž nepodařilo nalézt jiný spolehlivý způsob, jak vytvořit smyčku přes jiná zařízení či alternativní metody, což znemožnilo směrování provozu zpět do záznamové aplikace bez fyzického přeposlání dat skrz reálné síťové rozhraní.

4.2 Měření propustnosti při zátěži

V rámci testování je již k dispozici generovaný síťový tok s předem definovanými parametry. Následně bude postupně zvyšována zátěž cílové aplikace s cílem vyhodnotit její chování při různých přenosových rychlostech. Získané výsledky budou porovnány s výkony dalších dostupných nástrojů pro záznam síťového provozu.

4.2.1 Výsledky měření a porovnání v ostatních nástrojích

Pro první test aplikace byla na zařízení MikroTik zvolena rychlost generování paketů na hodnotu 500 Mbps. Na obrázku 4.4 je patrné, že kernel nezahodil žádný paket a že počet paketů zachycených kernelem je shodný s počtem paketů úspěšně zapsaných aplikací. To potvrzují i výpisy proměnných *tp_packets*, *tp_drops* a *packet_count*. Aplikace tedy při záznamu provozu o rychlosti 500 Mbps vykazuje nulovou ztrátovost, a bylo tak možné přistoupit k dalšímu testu s vyšší přenosovou rychlostí.

V dalším testu byla rychlost generovaného provozu zvýšena na 1 000 Mbps. I v tomto případě nebyl kernelem zahoděn žádný paket, což potvrzuje hodnota proměnné *tp_drops*. Došlo však ke zjištění rozdílu mezi počtem paketů zachycených kernelem (*tp_packets*) a počtem paketů zapsaných aplikací do výstupního souboru (*packet_count*). Konkrétně nebyly zapsány dva pakety, které byly v okamžiku ukončení aplikace ještě přítomny v bufferu, ale nedošlo k jejich zápisu. Výstup aplikace je vidět na obrázku 4.5.

```

^CStatistika zachytávání:
  Přijato paketů (kernel): 58622
  Zahozeno paketů:      0

--- Výkonové statistiky ---
Doba zachytávání: 8.348718 s
Skutečně zachyceno: 58622 paketů
Celková velikost dat: 502.62 MB
Průměrná rychlost: 0.505 Gbps

```

Obr. 4.4: První test s nastaveným tokem 500 *Mbps*

Tento jev lze přičíst vyšší přenosové rychlosti, při níž už není zaručeno, že všechny přijaté pakety budou včas zpracovány a zapsány při obdržení signálu *Ctrl+C*. Vzhledem k tomu, že tyto pakety nebyly kernelem zahozeny, nelze tento rozdíl považovat za skutečnou ztrátovost aplikace. Pokud by aplikace pokračovala v běhu, je pravděpodobné, že by došlo i k jejich zápisu.

```

^CStatistika zachytávání:
  Přijato paketů (kernel): 246097
  Zahozeno paketů:      0

--- Výkonové statistiky ---
Doba zachytávání: 20.902817 s
Skutečně zachyceno: 246055 paketů
Celková velikost dat: 2111.58 MB
Průměrná rychlost: 0.847 Gbps

```

Obr. 4.5: Druhý test s nastaveným tokem 1 000 *Mbps*

Při této rychlosti je zároveň patrné, že zařízení MikroTik nedokáže přesně generovat zadanou hodnotu přenosové rychlosti. Reálně naměřená rychlost byla nižší – při nastavené hodnotě 1 000 *Mbps* dosahoval skutečný datový tok pouze 0,847 *Gbps*. Tuto skutečnost potvrzuje i výstřížek obrazovky z aplikace Winbox, která v rámci nástroje *Traffic Generator* poskytuje vlastní měření statistik provozu (viz obrázek 4.6).

Další zvolenou rychlostí pro testování bylo 1 500 *Mbps*. Během doby trvání testu 19,13 *sekund* bylo zachyceno celkem 2 737,83 *MB* dat, což odpovídá průměrné přenosové rychlosti přibližně 1,2 *Gbps*, jak je vidět na obrázku výkonnostní statistiky 4.7. Naměřená hodnota opět neodpovídá požadované hodnotě 1 500 *Mbps*, která byla nastavena na zařízení MikroTik (viz obrázek 4.8).

Při této rychlosti nebyl zahozen ani jeden paket z celkového počtu 319 107 zachycených kernelem, což potvrzuje hodnota *tp_drops* rovna nule. Zvýšil se však počet paketů, které nebyly aplikací zapsány před jejím ukončením. Do výstupního

Seq	ID	Tx Packets	Tx Rate
312	0	11 696	842.1 Mbps
311	0	11 698	842.2 Mbps
310	0	11 661	839.5 Mbps
309	0	11 633	837.5 Mbps
308	0	11 698	842.2 Mbps
307	0	11 697	842.1 Mbps
306	0	11 662	839.6 Mbps
305	0	11 633	837.5 Mbps
304	0	11 698	842.2 Mbps
303	0	11 697	842.1 Mbps
302	0	11 634	837.6 Mbps
301	0	11 725	844.2 Mbps
300	0	11 634	837.6 Mbps
299	0	11 697	842.1 Mbps
298	0	11 696	842.1 Mbps
297	0	11 662	839.6 Mbps
296	0	11 633	837.5 Mbps

Obr. 4.6: Reálná rychlost při nastavení 1 000 *Mbps*

souboru bylo zapsáno pouze 319 083 paketů, což znamená, že 24 paketů zůstalo nezapsaných, pravděpodobně kvůli nedostatečnému času pro vyprázdnění bufferu při ukončení běhu programu.

```
^CStatistika zachytávání:
Přijato paketů (kernel): 319107
Zahozeno paketů: 0

--- Výkonové statistiky ---
Doba zachytávání: 19.138417 s
Skutečně zachyceno: 319083 paketů
Celková velikost dat: 2737.83 MB
Průměrná rychlost: 1.200 Gbps
```

Obr. 4.7: Třetí test s nastaveným tokem 1 500 *Mbps*

61	0	16 320	1175.0 Mbps
60	0	16 386	1179.7 Mbps
59	0	16 380	1179.3 Mbps
58	0	16 384	1179.6 Mbps
57	0	16 386	1179.7 Mbps
56	0	16 385	1179.7 Mbps
55	0	16 350	1177.2 Mbps
54	0	16 413	1181.7 Mbps
53	0	16 322	1175.1 Mbps
52	0	16 450	1184.4 Mbps
51	0	16 320	1175.0 Mbps
50	0	16 335	1176.1 Mbps
49	0	16 385	1179.7 Mbps
48	0	16 385	1179.7 Mbps

Obr. 4.8: Reálná rychlost při nastavení 1 500 *Mbps*

Při testu s nastavenou rychlostí 2 000 *Mbps* nedošlo opět k žádnému zahození paketů kernelem, hodnota *tp_drops* zůstala nulová. Celkem bylo zachyceno 434 824 paketů, z nichž bylo do výstupního souboru úspěšně zapsáno 434 728. Test probíhal po dobu 18,38 *sekund* a bylo během něj zaznamenáno 3 730,64 *MB* dat. Průměrná přenosová rychlost aplikace dosáhla hodnoty přibližně 1,702 *Gbps*.

Reálná rychlost přenosu generovaného zařízením MikroTik, která byla měřena pomocí jeho integrovaného nástroje *Traffic Generator*, je zobrazena na obrázku 4.10. I v tomto případě lze pozorovat rozdíl mezi nastavenou a skutečně dosaženou přenosovou rychlostí.

```
^CStatistika zachytávání:
Přijato paketů (kernel): 434824
Zahozeno paketů: 0

--- Výkonové statistiky ---
Doba zachytávání: 18.383560 s
Skutečně zachyceno: 434728 paketů
Celková velikost dat: 3730.64 MB
Průměrná rychlost: 1.702 Gbps
```

Obr. 4.9: Čtvrtý test s nastaveným tokem 2 000 *Mbps*

Seq	ID	Tx Packets	Tx Rate
130	0	23 104	1663.4 Mbps
129	0	23 040	1658.8 Mbps
128	0	22 939	1651.6 Mbps
127	0	22 955	1652.7 Mbps
126	0	22 976	1654.2 Mbps
125	0	23 040	1658.8 Mbps
124	0	23 074	1661.3 Mbps
123	0	23 075	1661.4 Mbps
122	0	23 040	1658.8 Mbps
121	0	22 976	1654.2 Mbps
120	0	23 040	1658.8 Mbps
119	0	23 134	1665.6 Mbps
118	0	23 011	1656.7 Mbps
117	0	23 098	1663.0 Mbps
116	0	23 003	1656.2 Mbps

Obr. 4.10: Reálná rychlost při nastavení 2 000 *Mbps*

Při nastavené rychlosti 2 500 *Mbps* se již začalo projevovat zahazování paketů kernelem. Konkrétně bylo zahozeno 524 paketů. Za dobu zápisu 18,24 *sekund* bylo reálně zapsáno pouze 516 836 z celkových zachycených 517 477 paketů, což odpovídá ztrátovosti přibližně 0,124 %.

Pro srovnání byl proveden identický test pomocí nástroje *tcpdump*. Při nastaveném čase 20 *sekund* stihl *tcpdump* zachytit celkem 553 295 paketů, avšak z tohoto počtu bylo 6 724 paketů zahozeno (viz 4.11), což odpovídá ztrátovosti 1,215 %. Tato

hodnota je násobně vyšší než u testované aplikace, což potvrzuje její lepší efektivitu při zachytávání dat v náročnějších podmínkách. Reálně vygenerovaná rychlost toku se pohybovala okolo 2 *Gbps*, jak ukazuje výstřížek z nástroje *Traffic Generator* v aplikaci Winbox (viz obrázek 4.13).

```
[filip@localhost Packet_Capture]$ sudo timeout 20 tcpdump -i enp6s0f3u1 -w dump.pcap
[sudo] heslo pro filip:
dropped privs to tcpdump
tcpdump: listening on enp6s0f3u1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
546539 packets captured
553295 packets received by filter
6724 packets dropped by kernel
[filip@localhost Packet_Capture]$
```

Obr. 4.11: Test záznamu 2 500 *Mbps* pomocí tcpdump

4.2.2 Optimalizace výkonu

Jelikož 2 *Gbps* představuje přibližnou hranici toho, co lze označit jako multi-gigabitový tok, a již při této úrovni dochází ke ztrátám, bylo nutné provést určité úpravy s cílem eliminovat zahazování paketů a zvýšit efektivitu záznamu.

```
^CStatistika zachytávání:
  Přijato paketů (kernel): 517477
  Zahozeno paketů:          524

--- Výkonové statistiky ---
Doba zachytávání: 18.246090 s
Skutečně zachyceno: 516836 paketů
Celková velikost dat: 4435.33 MB
Průměrná rychlost: 2.039 Gbps
```

Obr. 4.12: Pátý test s nastaveným tokem 2 500 *Mbps*

Nejprve byla provedena změna nastavení velikosti vytvářených bufferů na začátku programu. Jednalo se konkrétně o parametry *BLOCK_SIZE* (velikost jednoho bloku v bajtech), *FRAME_SIZE* (velikost jednoho rámce v kruhovém bufferu) a *FRAME_COUNT* (počet vytvořených rámců). Původní hodnoty těchto parametrů jsou znázorněny na obrázku 4.14, nové nastavení pak na obrázku 4.15.

Tato změna zajišťuje, že zapisování paketů již nebude omezeno nedostatečnou velikostí bufferů. V původním stavu totiž mohlo docházet k situacím, kdy kernel zahazoval pakety z důvodu zaplnění bufferu – tedy proto, že pro nově příchozí pakety nebylo k dispozici volné místo pro zápis do paměti.

Toto samotné však problém se ztrátovostí nevyřešilo a bylo nutné hledat další úpravy vedoucí ke zvýšení spolehlivosti záznamu. Dalším krokem bylo odstranění

2681	0	27 763	1998.9 Mbps
2680	0	27 699	1994.3 Mbps
2679	0	27 790	2.0 Gbps
2678	0	27 782	2.0 Gbps
2677	0	27 760	1998.7 Mbps
2676	0	27 764	1999.0 Mbps
2675	0	27 853	2.0 Gbps
2674	0	27 698	1994.2 Mbps
2673	0	27 729	1996.4 Mbps
2672	0	27 827	2.0 Gbps
2671	0	27 761	1998.7 Mbps
2670	0	27 716	1995.5 Mbps
2669	0	27 790	2.0 Gbps
2668	0	27 699	1994.3 Mbps
2667	0	27 828	2.0 Gbps
2666	0	27 648	1990.6 Mbps

Obr. 4.13: Reálná rychlost při nastavení 2 500 *Mbps*

```
// Nastavení velikostí bufferu a rámců pro zachytávání
#define BLOCK_SIZE 8192
#define FRAME_SIZE 2048
#define FRAME_COUNT 4096
```

Obr. 4.14: Původní hodnoty nastavení bufferů

```
// velikosti bufferu a rámců pro zachytávání
#define FRAME_SIZE 16384
#define BLOCK_SIZE (FRAME_SIZE * 64)
#define FRAME_COUNT 2048
```

Obr. 4.15: Změněné hodnoty nastavení bufferů

zbytečného čekání v programu, konkrétně volání funkce *usleep(10)*, které bylo prováděno pokaždé, když v daném okamžiku nebylo co zapisovat.

```
// Čekání, dokud není rámec připraven jádrem
if (!(hdr->tp_status & TP_STATUS_USER)) {
    usleep(10);
    continue;
}
```

Obr. 4.16: Uspávání programu pomocí funkce *usleep()*

Tato pauza měla původně zabránit nadměrnému vytěžování procesoru v případě nečinnosti. V prostředí vysokorychlostního provozu však způsobovala opačný efekt – pokud aplikace „usne“ ve chvíli, kdy není co zapisovat, ale mezitím dorazí nový

provoz, může dojít k přetečení bufferu dříve, než aplikace opět začne zapisovat. Výsledkem je růst hodnoty *tp_drops*, tedy ztrátovosti způsobené nedostatečným odčerpáváním dat z příjmového bufferu.

Proto bylo nutné tento blok nahradit efektivnějším řešením, které by nečekalo pevně stanovenou dobu, ale reagovalo okamžitě na příchozí provoz. Toho lze dosáhnout například použitím funkce *poll()*, která čeká na aktivitu na socketu. Pokud během čekací doby (např. 100 *ms*) dorazí data, pokračuje aplikace ve zpracování. Pokud nic nepříjde, proběhne nová iterace smyčky. Pokud jsou data připravena k zápisu, aplikace reaguje okamžitě. Tím se eliminuje zbytečné čekání a aplikace je schopna efektivně a včas reagovat na příchozí pakety.

Ani tato úprava však nevedla ke snížení ztrátovosti aplikace. Následovala tedy další optimalizace, tentokrát na úrovni nastavení jádra operačního systému. Prvotní konfigurace byla provedena již po instalaci systému, avšak některé systémové parametry bylo nutné upravit znovu.

Konkrétně byla zvýšena hodnota *net.core.rmem_max*, která určuje maximální velikost přijímacího socketového bufferu, z původní hodnoty 33 554 432 na 536 870 912 *bajtů*. Stejná hodnota byla nastavena také pro *net.core.rmem_default*, což je výchozí velikost bufferu přidělovaná nově vytvořeným socketům. Dále byla změněna hodnota *net.core.netdev_max_backlog*, což je maximální délka fronty nevyřízených paketů mezi síťovým ovladačem a zpracováním v jádře, ze 100 000 na 500 000.

Po těchto úpravách již k zahazování paketů během běhu aplikace nedocházelo, což potvrzuje, že příčinou problému byla nedostatečná velikost některého ze systémových bufferů.

Pro zvýšení rychlosti zpracování paketů a minimalizaci ztrátovosti při vysokých přenosových rychlostech byla do hlavní smyčky *while()* přidána vnitřní smyčka *for()*, která umožňuje dávkové zpracování více paketů najednou. V rámci této vnitřní smyčky byla přesunuta většina operací — mimo jiné výpočet časových údajů, vytvoření struktury hlavičky paketu, kontrola stavu přijatého rámce a zpracování jeho obsahu.

Zásadní změnou oproti původní verzi bylo zavedení vlastního paměťového bufferu, do kterého se postupně ukládá definovaný počet paketů (v tomto případě 16) spolu s jejich hlavičkami. Až po naplnění bufferu dochází k jeho hromadnému zápisu do výstupního souboru. Tento způsob snižuje počet systémových volání *write()* a zvyšuje tak efektivitu při práci s diskem, což má přímý vliv na propustnost aplikace. Nevýhodou může být zvýšení požadavků na paměť.

Při ukončení aplikace je alokovaná paměť pro tento buffer korektně uvolněna. V rámci optimalizace byly dále zpřehledněny názvy proměnných souvisejících s *RX_RING* bufferem a upraveny výpočty související s jeho velikostí a strukturou,

což přispívá k lepší čitelnosti kódu a snazší údržbě.

4.2.3 Vyhodnocení výkonu po optimalizaci aplikace

Pro zajištění plynulého chodu aplikace bylo prováděno opakované měření, a to několikrát po sobě za stejných podmínek, aby bylo možné ověřit stabilitu výsledků. Konkrétně byl test realizován pomocí funkce *timeout*, která ukončila zachytávání vždy po 20 *sekundách*. Tím bylo zajištěno, že výsledky budou mezi jednotlivými měřeními vzájemně porovnatelné. Generovaný tok byl nastaven na maximální možnou hodnotu datového portu zařízení MikroTik, tedy 2 500 *Mbps*, přičemž reálná přenosová rychlost dosahovala hodnoty přibližně 2,1 *Gbps*.

Na základě těchto měření lze konstatovat, že aplikace při zátěži okolo 2,1 *Gbps* stabilně a bezztrátově zachytává datový tok, jak je vidět na obrázku 4.17.

Pro porovnání byl stejný test proveden také s nástrojem *tcpdump*, u kterého byla zaznamenána ztrátovost přesahující 3 % (viz obrázek 4.18), zatímco vyvinutá aplikace vykázala ztrátovost nulovou.

```
--- Výkonové statistiky ---
Doba zachytávání: 18.534942 s
Skutečně zachyceno: 545671 paketů
Celková velikost dat: 4683.50 MB
Průměrná rychlost: 2.120 Gbps
[filip@localhost Packet_Capture]$ sudo timeout 20 ./packet_capture_pcap enp6s0f3u1 /home/filip/capture.pcap
[sudo] heslo pro filip:
Statistika zachytávání:
  Přijato paketů (kernel): 554612
  Zahozeno paketů: 0

--- Výkonové statistiky ---
Doba zachytávání: 18.789952 s
Skutečně zachyceno: 539740 paketů
Celková velikost dat: 4632.58 MB
Průměrná rychlost: 2.068 Gbps
[filip@localhost Packet_Capture]$ sudo timeout 20 ./packet_capture_pcap enp6s0f3u1 /home/filip/capture.pcap
Statistika zachytávání:
  Přijato paketů (kernel): 553418
  Zahozeno paketů: 0

--- Výkonové statistiky ---
Doba zachytávání: 18.920217 s
Skutečně zachyceno: 548362 paketů
Celková velikost dat: 4706.60 MB
Průměrná rychlost: 2.087 Gbps
[filip@localhost Packet_Capture]$ sudo timeout 20 ./packet_capture_pcap enp6s0f3u1 /home/filip/capture.pcap
Statistika zachytávání:
  Přijato paketů (kernel): 554212
  Zahozeno paketů: 0

--- Výkonové statistiky ---
Doba zachytávání: 18.941778 s
Skutečně zachyceno: 549485 paketů
Celková velikost dat: 4716.21 MB
Průměrná rychlost: 2.089 Gbps
```

Obr. 4.17: Replikování výsledků testu aplikace při 2 500 *Mbps*

Když bylo toto měření provedeno znovu s přidáním parametru *-B*, který mění velikost bufferu používaného knihovnou *libpcap* (na níž je nástroj *tcpdump* založen),

```

tcpdump: listening on enp6s0f3u1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
543821 packets captured
554419 packets received by filter
10578 packets dropped by kernel
[filip@localhost Packet_Capture]$ sudo timeout 20 tcpdump -i enp6s0f3u1 -w dump.pcap
dropped privs to tcpdump
tcpdump: listening on enp6s0f3u1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
530517 packets captured
554353 packets received by filter
23815 packets dropped by kernel
[filip@localhost Packet_Capture]$ sudo timeout 20 tcpdump -i enp6s0f3u1 -w dump.pcap
dropped privs to tcpdump
tcpdump: listening on enp6s0f3u1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
533292 packets captured
554616 packets received by filter
21295 packets dropped by kernel
[filip@localhost Packet_Capture]$ sudo timeout 20 tcpdump -i enp6s0f3u1 -w dump.pcap
dropped privs to tcpdump
tcpdump: listening on enp6s0f3u1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
533134 packets captured
554627 packets received by filter
21463 packets dropped by kernel
[filip@localhost Packet_Capture]$ sudo timeout 20 tcpdump -i enp6s0f3u1 -w dump.pcap
dropped privs to tcpdump
tcpdump: listening on enp6s0f3u1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
533712 packets captured
554490 packets received by filter
20751 packets dropped by kernel
[filip@localhost Packet_Capture]$ sudo timeout 20 tcpdump -i enp6s0f3u1 -w dump.pcap
dropped privs to tcpdump
tcpdump: listening on enp6s0f3u1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
533798 packets captured
554640 packets received by filter
20819 packets dropped by kernel

```

Obr. 4.18: Replikování výsledků testu tcpdump při 2 500 *Mbps* před přidáním parametru -B

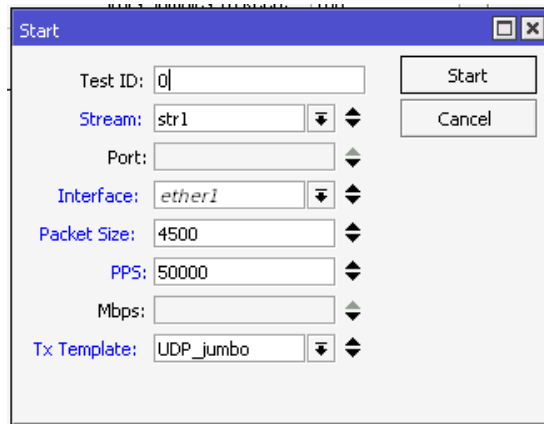
ztrátovost se snížila na 0 %. Stejný test byl následně proveden i pomocí nástroje *Wireshark* a rovněž zde byla naměřena nulová ztrátovost.

Lze však předpokládat, že při vyšších přenosových rychlostech by i tyto nástroje začaly vykazovat ztrátovost, a to z důvodu svých omezení v oblasti výkonu a efektivity zpracování vysokorychlostního síťového provozu.

4.3 Zatížení vyšším počtem paketů menší velikosti

Jako další test bylo provedeno zatížení aplikace větším počtem menších paketů. Konkrétně byl vytvořen datový tok s velikostí MTU 4 500 *bajtů*, tedy poloviční hodnotou oproti standardní velikosti jumbo paketů. Požadovaný počet přenesených paketů byl nastaven na 50 000 *PPS* (Packets Per Second), přičemž reálně bylo naměřeno přibližně 47 950 *PPS*. Nastavení tohoto testovacího toku je znázorněno na obrázku 4.19.

Také v tomto případě nedocházelo k zahazování paketů ze strany kernelu, nicméně oproti předchozím testům došlo k výraznému nárůstu počtu ztracených paketů na



Obr. 4.19: Nastavení rychlejšího toku s menšími pakety

počátku záznamu, tedy v čase inicializace aplikace. Tento jev je očekávaný, neboť v důsledku zdvojnásobeného přenosového tempa narůstá pravděpodobnost přepsání paketů v bufferu ještě před aktivací hlavní smyčky. Opakovaným měřením byla tato počáteční ztráta vyčíslena přibližně na 8000 paketů. Výsledek jednoho z těchto měření je zachycen na obrázku 4.20.

Pro ověření, že k uvedeným ztrátám dochází skutečně pouze před zahájením hlavní zapisovací smyčky, byly provedeny testy s různě dlouhými časovými úseky běhu aplikace. Ve všech případech byla výsledná ztráta konstantní, bez ohledu na délku záznamu. To potvrzuje, že k žádným dalším ztrátám během samotného běhu aplikace nedochází.

Lze tedy předpokládat, že ztráty vznikají pouze v průběhu inicializace aplikace, kdy kernel již přijímá a ukládá pakety do bufferu, ale samotná aplikace je zatím nezačala zpracovávat. Tato dočasná prodleva mezi aktivací socketu a spuštěním záznamové logiky způsobuje, že část příchozích dat je přepsána dříve, než k nim aplikace získá přístup.

Pro srovnání byl proveden stejný test pomocí nástrojů *tcpdump* a *Wireshark*. V případě *tcpdump* došlo při stejných parametrech přenosu ke ztrátě pouze zhruba 150 paketů, což ukazuje na odlišné chování při inicializaci a zpracování paketů mezi oběma nástroji. V případě nástroje *Wireshark* došlo ke ztrátě 2 274 paketů (obrázek 4.21).

Byl proveden další test s ještě menší velikostí paketů a vyšší přenosovou rychlostí vyjádřenou v počtu paketů za sekundu. Velikost jednoho paketu byla nastavena na 3000 bajtů a hodnota *PPS* na 70 000.

U vyvíjené aplikace došlo opět ke zvýšení počtu zahozených paketů v úvodní fázi, tedy ještě před zahájením hlavní zapisovací smyčky. U nástroje *tcpdump* byl zaznamenán větší rozdíl mezi počtem přijatých paketů podle filtru a skutečně za-

```
[filip@localhost Packet_Capture]$ sudo timeout 120s ./packet_capture_pcap enp6s0f3u1 /home/filip/Packet_Capture/capture.pcap
Zachytávám na rozhraní enp6s0f3u1 a zapisuji do /home/filip/Packet_Capture/capture.pcap.

--- Statistika zachytávání (kernel) ---
Přijato paketů:      5666385
Zahozeno paketů:     0

--- Výkonové statistiky ---
Doba zachytávání:    118.600165 s
Zapsáno paketů:      5658132
Celková velikost dat: 25461.18 MB
Průměrná rychlost:   1.717 Gbps

[filip@localhost Packet_Capture]$
```

Obr. 4.20: Měření propustnosti aplikace pro zátěž vyšším počtem paketů menší velikosti

Interfaces				
Interface	Dropped packets	Capture filter	Link type	Packet size limit
enp6s0f3u1	2274 (0.1%)	none	Ethernet	262144 bytes
Statistics				
Measurement	Captured	Displayed	Marked	
Packets	3188476	3188476 (100.0%)	—	
Time span, s	63.814	63.814	—	
Average pps	49965.1	49965.1	—	
Average packet size, B	4500	4500	—	
Bytes	14348018461	14348018461 (100.0%)	0	
Average bytes/s	224M	224M	—	
Average bits/s	1 798M	1 798M	—	

Obr. 4.21: Měření propustnosti nástroje Wireshark pro zátěž vyšším počtem paketů menší velikosti

psanými pakety do výstupního souboru. Naopak nástroj Wireshark vykázal v tomto testu zlepšení – ztrátovost poklesla zpět na 0 %, pravděpodobně v důsledku lepšího zpracování menších paketů v této konkrétní konfiguraci.

5 Návrh alternativního řešení

Tato kapitola se věnuje souhrnnému zhodnocení dosavadního řešení, které bylo postaveno na běžném notebooku v kombinaci s externí více-gigabitovou síťovou kartou. Cílem je posoudit praktickou použitelnost tohoto přístupu při záznamu síťového provozu ve vysokých přenosových rychlostech z hlediska výkonu, spolehlivosti i provozní flexibility. Na základě identifikovaných výhod a omezení bude následně navržena alternativní architektura využívající embedded platformu, která by mohla lépe vyhovovat požadavkům na dlouhodobý a stabilní provoz.

5.1 Shrnutí zkušeností s notebookem

Řešení postavené na běžném notebooku v kombinaci s externí síťovou kartou se ukázalo jako funkční varianta, která při vhodném softwarovém nastavení a optimalizaci (např. využitím *AF_PACKET* a *RX_RING*) umožňuje bezproblémový záznam síťového provozu při přenosových rychlostech okolo 2,1 Gbps. Výhodou tohoto přístupu je zejména dostupnost použitého hardwaru a nízké pořizovací náklady. Systém disponuje dostatečným výpočetním výkonem, volným slotem pro rychlé NVMe úložiště a rozhraním USB-C 3.2 Gen. 2 pro připojení externí více-gigabitové síťové karty.

Během testování se však ukázalo několik praktických omezení, která tuto architekturu činí méně vhodnou pro dlouhodobý nebo produkční provoz. Mezi hlavní nevýhody patří vyšší spotřeba energie, větší fyzické rozměry a omezená flexibilita při nasazení v prostorově omezených nebo plně autonomních prostředích. Dalším faktorem je omezený počet síťových rozhraní a nutnost softwarové podpory pro externí síťové adaptéry, což může v některých případech komplikovat nasazení.

Z těchto důvodů byla následně navržena alternativní varianta založená na embedded platformě, která by mohla lépe vyhovovat požadavkům na kompaktní rozměry, energetickou efektivitu a provozní spolehlivost.

5.2 Řešení s embedded počítačem

Za účelem zajištění přenositelnosti, jednoduchosti a cenové dostupnosti hardwaru pro záznamové zařízení s rozhraním multi-gigabitového Ethernetu byla zvolena platforma embedded počítačů. Při výběru vhodného zařízení je klíčové, aby obsahovalo všechny nezbytné komponenty pro spolehlivý záznam síťového provozu. Mezi tyto komponenty patří zejména slot pro NVMe SSD v provedení M.2 pro zajištění dostatečně rychlého zápisu, dostatečná kapacita operační paměti (RAM), síťový port

s minimální přenosovou rychlostí 2,5 *Gbps* a možnost napájení z DC zdroje, což umožňuje provoz zařízení i z bateriového napájení.

Pro záznam dat v řádu několika gigabitů za sekundu je zcela dostačující použití sběrnice PCIe verze 3.0, která umožňuje přenosové rychlosti až 3 500 *Mbps*. Pokud je požadována vyšší rezerva nebo budoucí rozšiřitelnost, lze využít i novější verzi PCIe 4.0, jejíž propustnost dosahuje přibližně 5 000 *Mbps*.

Výběr již hotového a komerčně dostupného zařízení s sebou nese výhody i nevýhody. Hlavní výhodou je úspora nákladů a času spojených s vývojem vlastního hardwarového řešení. Na druhou stranu však tato zařízení často nenabízejí jednoduchou rozšiřitelnost, což může být limitující v případě budoucích požadavků na úpravu nebo rozšíření funkcionality. Dále může být výzvou nalezení vhodného modelu, který by splňoval všechny požadavky na specifikace a konektivitu. Naštěstí je trh s embedded zařízeními v současnosti natolik rozvinutý, že i v nižších cenových kategoriích lze nalézt plně konfigurovatelné počítače. Většina těchto řešení nabízí M.2 slot pro NVMe disk již ve standardní výbavě, avšak často jsou vybavena pouze 1 *Gbps* LAN portem, což může být omezením při práci s požadovanými vyššími přenosovými rychlostmi.

Mezi možné výrobce embedded single-board počítačů vhodných pro záznam síťového provozu patří společnosti jako Advantech, Maxtang, Asus nebo Cincoze. Společnost Advantech nabízí převážně samotné základní desky s procesorem, rozhraními a konektory, bez přibalených periférií. Uživatel si musí samostatně zajistit úložná zařízení, rozšiřující moduly a vhodné šasi. Oproti tomu Maxtang nabízí pět produktových řad průmyslových embedded počítačů, které již zahrnují kompletní sestavu včetně skříně, avšak bez integrovaného datového úložiště.

Společnost Cincoze pak kromě zařízení s pevně danými specifikacemi poskytuje také široce konfigurovatelné modely. U těch je možné přizpůsobit parametry jako typ procesoru, kapacitu paměti RAM, typ a počet vstupně-výstupních portů, přítomnost předinstalovaného operačního systému nebo třeba volitelný aktivní chladič. V případě všech zmíněných výrobců je při objednávce nutné specifikovat požadovanou velikost operační paměti, obvykle v rámci výrobcem určeného rozsahu.

Pro správný výběr zařízení bylo nejprve nutné analyzovat nároky operačního systému a samotné záznamové aplikace na systémové prostředky, zejména spotřebu paměti RAM a zátěž procesoru. V první fázi bylo provedeno měření základního zatížení distribuce AlmaLinux v nečinném stavu, tedy bez spuštěné aplikace. Následně byla spuštěna záznamová aplikace a měření bylo zopakováno, aby bylo možné porovnat rozdíl ve využití systémových prostředků a posoudit náročnost běhu aplikace.

Oba stavy jsou zachyceny na následujících obrázcích: výchozí stav bez aplikace na obrázku 5.1, stav se spuštěnou aplikací na obrázku 5.2.

Z těchto dvou obrázků je patrné, že v nečinném stavu spotřebovává samotný ope-



Obr. 5.1: Zatížení zařízení v klidovém stavu



Obr. 5.2: Zatížení zařízení při spuštění aplikaci

rační systém přibližně 2,1 GB paměti RAM. Po spuštění záznamové aplikace celková spotřeba vzrostla na 4,5 GB, což znamená, že samotná aplikace vyžaduje přibližně 2,4 GB RAM. Tuto hodnotu je nutné zohlednit při výběru vhodného zařízení, na kterém bude provozováno zachytávání síťového provozu, zejména s ohledem na paměťovou kapacitu systému. Zvýšené využití paměti je způsobeno především alokací

kruhového bufferu (ring bufferu), jehož velikost byla na začátku programu nastavena na 2,0 *GiB*, což odpovídá přibližně 2,147 *GB*. Další část paměti je alokována pro batch buffer, který slouží ke zpracování více paketů najednou.

Vytížení jednotlivých jader procesoru je rovněž zaznamenáno na výše uvedených obrázcích. Ani při plném běhu aplikace nepřesáhla zátěž žádného z procesorových jader hranici 70 %. Je však patrné, že nejvyšší vytížení je soustředěno pouze na jedno jádro, zatímco ostatní zůstávají relativně nevytížená. To odpovídá skutečnosti, že aplikace byla navržena jako jednovláknová a zpracování paketů probíhá v rámci jednoho vlákna. Pro další zvýšení výkonnosti a lepší využití vícejádrových systémů by bylo do budoucna vhodné zvážit optimalizaci směrem k paralelnímu zpracování.

5.2.1 Advantech MIO-4370

Embedded zařízení MIO-4370 od společnosti Advantech představuje výkonnou a flexibilní platformu vhodnou i pro náročnější síťové aplikace. Podporuje až 32 *GB* paměti DDR5 RAM, je vybaveno dvěma 2,5 *GbE* Ethernet porty, až čtyřmi porty USB 3.2 Gen 2 a dvěma výstupy HDMI.

Uživatel má možnost volby z několika variant procesorů značky Intel, přičemž i nejnižší dostupná konfigurace podle výsledků benchmarkových měření dosahuje výkonu pouze mírně nižšího než procesor použitý v testovacím zařízení Lenovo (viz obrázek 5.3). Vzhledem k vytíženosti procesoru při testování (kapitola 5.2) by tedy i tento procesor měl být dostačující. [46]

Zařízení je dále dostupné ve variantě se dvěma sloty M.2 M-key 2280 PCIe Gen. 4, do kterých lze osadit NVMe SSD disky. Alternativně je možné využít jeden slot pro záznamové úložiště a druhý pro rozšiřující síťovou kartu s podporou přenosových rychlostí až 10 *Gbps*. [45, 47]

5.2.2 Advantech MIO-2364

MIO-2364 je levnější a kompaktnější varianta embedded počítače od společnosti Advantech. Zařízení podporuje až 16 *GB* paměti RAM a je vybaveno jedním 2,5 *GbE* Ethernet portem. Použitý procesor má dostatečný výkon pro záznam síťového provozu, avšak hlavním omezením tohoto modelu je absence NVMe kompatibilního PCIe rozhraní. Z tohoto důvodu není možné použít výkonné NVMe SSD disky, což může představovat zásadní překážku při snaze o dosažení potřebné rychlosti zápisu při multi-gigabitovém záznamu. [44]

	Intel Core i3-12100TE	AMD Ryzen 5 5600H
Price	Search Online	Search Online
Socket Type	FCLGA1700	FP6
CPU Class	Laptop, Mobile/Embedded	Laptop
Clockspeed	2.1 GHz	3.3 GHz
Turbo Speed	Up to 4.0 GHz	Up to 4.2 GHz
# of Physical Cores	4 (Threads: 8)	6 (Threads: 12)
Cache	L1: 320KB, L2: 5.0MB, L3: 12MB	L1: 384KB, L2: 3.0MB, L3: 16MB
TDP	35W	45W
Yearly Running Cost	\$6.39	\$8.21
Other	Intel UHD Graphics 730	with Radeon Graphics
First Seen on Chart	Q3 2022	Q2 2021
# of Samples	5	4139
CPU Value	94.9	0.0
Single Thread Rating (% diff. to max in group)	3139 (0.0%)	2919 (-7.0%)
CPU Mark (% diff. to max in group)	11859 (-29.2%)	16754 (0.0%)

Obr. 5.3: Srovnání procesorů testovacího zařízení a MIO-4370[46]

5.2.3 Advantech MIO-5379

Nejdražší, ale zároveň nejvýkonnější variantou embedded zařízení od společnosti Advantech je model s označením MIO-5379. Tento počítač je vybaven velmi výkonným procesorem a podporuje až 96 GB paměti RAM. Disponuje jedním 1 GbE portem a jedním 2,5 GbE portem. Stejně jako ostatní modely však nabízí pouze dva sloty M.2 M-key PCIe, které lze využít buď pro NVMe úložiště, nebo pro rozšíření o síťovou kartu s podporou 10 Gbps. [43]

Vyšší kapacita operační paměti a výpočetní výkon procesoru mohou být výhodné zejména v případech, kdy je kromě samotného záznamu požadováno i následné zpracování nebo analýza zachycených dat přímo na zařízení – buď v reálném čase, nebo po dokončení záznamu.

5.2.4 Maxtang MTN-FP750

Maxtang MTN-FP750 je embedded počítač dodávaný jako kompletní řešení včetně kompaktního šasi. Zařízení nabízí jeden 2,5 GbE Ethernet port, podporuje až 64 GB

paměti RAM a je osazeno procesorem s výkonem srovnatelným s procesorem použitým v testovacím zařízení.

Hlavním omezením tohoto modelu je přítomnost pouze jednoho PCIe slotu kompatibilního s NVMe úložišti. To znamená, že slot nelze současně využít pro vysokorychlostní SSD disk i pro rozšíření o rychlejší síťovou kartu (např. 10 *GbE*), což limituje možnosti nasazení v prostředí s požadavky na extrémní přenosové rychlosti.

Zařízení je napájeno pomocí stejnosměrného zdroje s napětím 20 V, což je vhodné i pro mobilní nebo úsporné nasazení mimo standardní napájecí infrastrukturu. [42]

5.2.5 Specializovaná zařízení pro vysokorychlostní síťový provoz

Většina zařízení, která již ve výchozí konfiguraci obsahují síťový port s podporou 10 *GbE*, je zároveň vybavena dostatečnou kapacitou operační paměti a výkonným procesorem. Tato zařízení jsou obvykle navržena tak, aby zvládala záznam a analýzu vysokorychlostního síťového provozu bez výrazného omezení výkonu. Výrobci je často přímo prezentují jako specializovaná řešení pro síťovou analýzu, monitorování provozu nebo bezpečnostní dohled v reálném čase. Příklad takového zařízení je PALMSHELL NeXT H2 SBC. [41]

5.3 Výběr PC a komponent

Jako nejvhodnější volba bylo zvoleno zařízení MIO-4370, a to především s ohledem na jeho podporu 2,5 *GbE* Ethernetu, možnost připojení NVMe úložišť a případné rozšíření síťového připojení až na rychlost 10 *GbE* prostřednictvím jednoho z dostupných M.2 PCIe slotů. Pomocí dostupných rozhraní, jako jsou 2× HDMI a 4× USB 3.2 Gen 2, lze k zařízení připojit periferní zařízení, například monitor, myš a klávesnici, a zajistit tak jeho přímé ovládání.

MIO-4370 navíc disponuje dostatečnou kapacitou operační paměti a výkonným procesorem, čímž splňuje požadavky na stabilní a efektivní zpracování dat v prostředí multi-gigabitových sítí. Vzhledem k výpočetnímu výkonu a rozšiřitelnosti je rovněž možné v budoucnu jednoduše přejít na ještě vyšší rychlosti zachytávání bez nutnosti výměny zařízení. Z praktického hlediska je výhodná i pořizovací cena, která v době zpracování této práce činila přibližně 568,1 *EUR*, což představuje velmi dobrý poměr cena/výkon. [45, 47]

5.3.1 NVMe disk

K zařízení MIO-4370 byl vybrán disk od společnosti Samsung 990 PRO s kapacitou 4 *TB*, což představuje dostatečný prostor jak pro operační systém, tak i pro přibližně

6 minut záznamu při přenosové rychlosti 10 *Gbps*. V případě potřeby je možné do stejného slotu osadit disk s vyšší kapacitou. Tento konkrétní model byl zvolen díky vhodnému poměru mezi kapacitou a cenou. Jeho pořizovací cena činí 7 024 Kč bez DPH (stav k květnu 2025).

Jak samotné zařízení MIO-4370, tak i vybraný disk podporují připojení přes rozhraní PCIe 4.0, což zajišťuje dostatečnou přenosovou rychlost pro dané účely. Pokud by nebylo nutné využít druhý slot pro 10 *GbE* síťovou kartu, je možné do něj připojit další disk a pro záznam využít jeden z integrovaných 2,5 *GbE* portů dostupných přímo na základní desce. [58]

5.3.2 Síťová karta 10 *GbE*

Pro dosažení vyšší záznamové rychlosti je nutné k desce připojit síťovou kartu s vyšší přenosovou kapacitou. K tomuto účelu lze využít jeden ze slotů typu M.2 M-key 2280, do kterého je možné instalovat síťovou kartu s podporou přenosové rychlosti až 10 *GbE*. Pro navrhované řešení byla zvolena karta osazená slotem pro transceiver SFP+, který umožňuje připojení jak metalickým, tak i optickým kabelem. Pořizovací cena této síťové karty činí 1 849 Kč s DPH (stav k květnu 2025). [60]

Je však třeba zohlednit, že osazením síťové karty do tohoto slotu dojde ke ztrátě jednoho místa pro připojení paměťového disku, což může negativně ovlivnit dostupnou úložnou kapacitu. S touto skutečností je nutné při návrhu zařízení počítat, zejména pokud je cílem rozšíření záznamové rychlosti na 10 *GbE*.

5.3.3 Paměť RAM

K zařízení je rovněž nutné zvolit vhodnou operační paměť. Embedded platforma podporuje připojení paměťového modulu typu DDR5 s frekvencí 4800 *MHz* ve formátu SODIMM a maximální kapacitou až 32 *GB*.

S ohledem na budoucí možné rozšiřování funkcionality zařízení a provoz dalších aplikací na pozadí byl zvolen modul s maximální dostupnou kapacitou Crucial SO-DIMM 32 *GB* DDR5 4800 *MHz* CL40. Tím je zároveň zajištěno, že běh záznamové aplikace nebude omezován nedostatkem operační paměti. Pořizovací cena tohoto modulu činí 1 586 Kč bez DPH (stav k květnu 2025). [59]

5.4 Shrnutí

Celkové náklady na navrhované embedded řešení lze shrnout následovně: základní deska Advantech MIO-4370 s cenou 568,1 *EUR*, dále paměťový modul Crucial SO-DIMM 32GB DDR5 4800MHz za 1 586 Kč bez DPH, NVMe disk Samsung 990 PRO

4TB za 7 024 Kč bez DPH a volitelná 10GbE síťová karta Ethernet NIC Card M.2 to 10Gb B+M KEY 82599-1 SFP Industrial Control Server Network Card prodávaná společností Yiwu Ranwen Technology za 1 849 Kč s DPH. Při aktuálním kurzu (cca 25 Kč/EUR) činí celková cena přibližně 20 000 Kč bez DPH, v závislosti na konkrétní konfiguraci a využití volitelných komponent.

V porovnání s architekturou založenou na běžném notebooku a externí síťové kartě představuje toto embedded řešení nejen levnější, ale i výkonnější variantu. To platí i po započítání nákladů na potřebné periferní komponenty a výrobu vhodné skříně pro kompletaci zařízení. Díky dostupnosti 3D tisku je možné prototypové řešení umístit do přesně navrženého krytu s minimálními náklady – cena tisku pouzdra o rozměrech přibližně 20×20×10 cm se běžně pohybuje do 2 000 Kč.

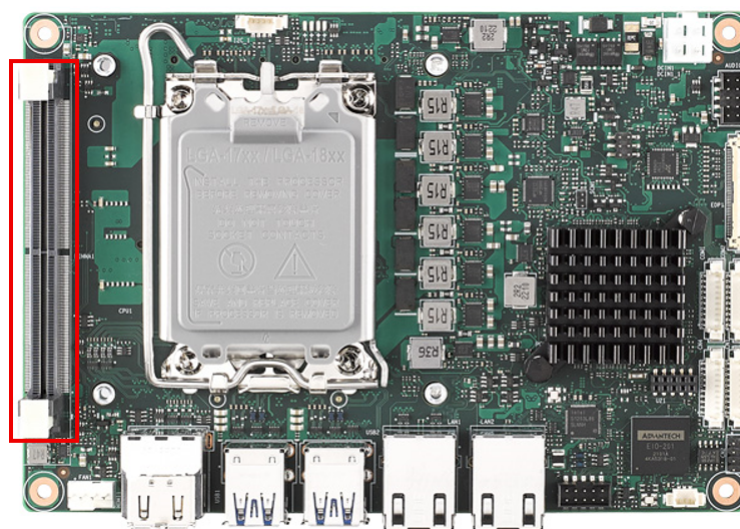
Embedded platforma MIO-4370 navíc umožňuje další zvyšování výpočetního výkonu díky patcovému uchycení procesoru, které umožňuje jeho případnou výměnu. Naopak možnosti upgradu operační paměti RAM a síťové konektivity jsou omezené, protože pro daný formát zařízení (COMe Compact s konektorem typu B) nejsou běžně dostupné síťové karty s rychlostí vyšší než 10 GbE. Pokud by byla požadována vyšší rychlost připojení (například 25 nebo 40 GbE), bylo by nutné přejít na jiný typ embedded platformy s rozhraním Thunderbolt 3/4, ke kterému se vyrábějí kompatibilní externí síťové karty.

Celkové náklady na plně funkční a výkonné zařízení včetně volitelných komponent a vyrobené skříně tak nepřesahují 25 000 Kč, což je stále výrazně méně než cena profesionálních komerčních řešení s obdobnými parametry.

6 Postup pro sestavení funkčního prototypu

Tato kapitola popisuje praktický postup sestavení a zprovoznění záznamového zařízení založeného na zvolené embedded platformě. Cílem je ukázat, jakým způsobem lze z jednotlivých komponent vytvořit plně funkční celek připravený pro provoz vyvinuté záznamové aplikace. Popis zahrnuje jak fyzické osazení komponent, tak i základní softwarovou konfiguraci.

Pokud jsou všechny potřebné komponenty k dispozici, je nejprve do vyhrazeného slotu na desce embedded počítače (viz obrázek 6.1) vložen paměťový modul typu DDR5 SODIMM, který byl zvolen v kapitole 5.3.3.

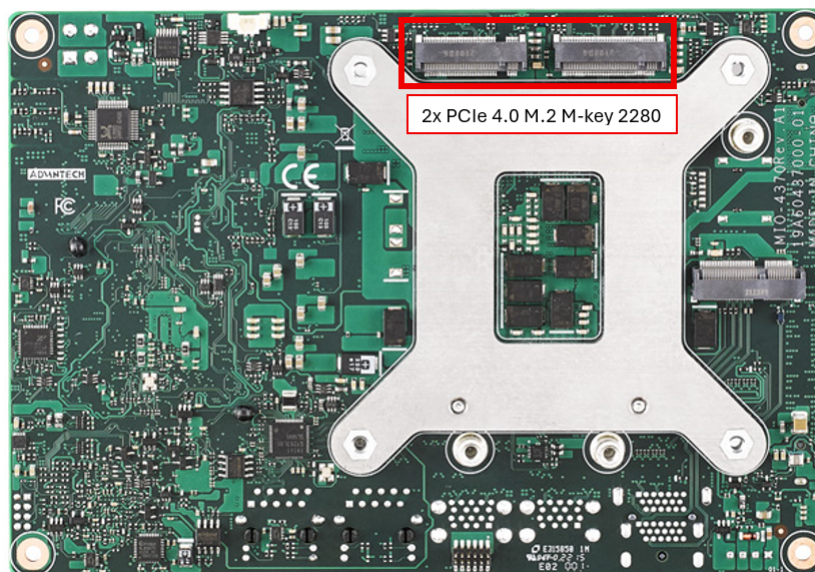


Obr. 6.1: Umístění slotu pro RAM paměť na desce embedded počítače MIO-4370 [45]

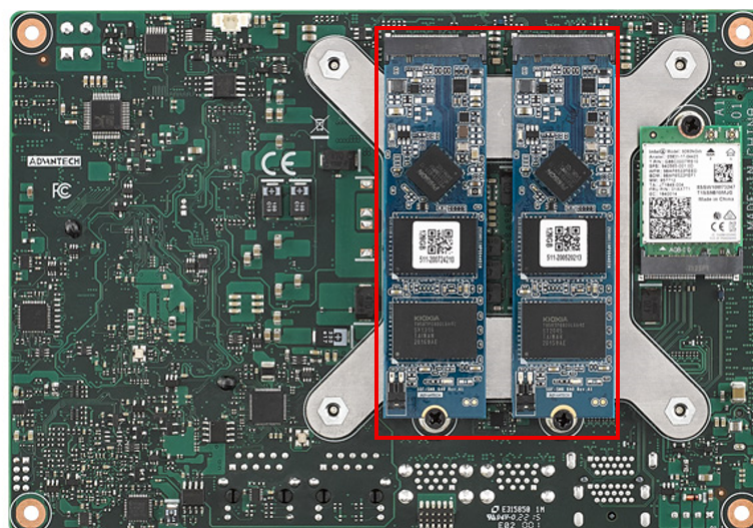
Deska je následně otočena, protože konektory s rozhraním PCIe 4.0 se nacházejí na její spodní straně (viz obrázek 6.2). Do těchto slotů je možné osadit buď dva paměťové disky typu NVMe, jak je znázorněno na obrázku 6.3, nebo kombinaci jednoho disku a jedné síťové karty s vyšší přenosovou rychlostí, například takové, jaká byla popsána v kapitole 5.3.2.

Po fyzickém sestavení zařízení a připojení nezbytných periférií (monitor, klávesnice, myš) prostřednictvím dostupných rozhraní (například 2× HDMI a 4× USB 3.2 Gen 2) může být přistoupeno k instalaci operačního systému. Pro účely této práce byl zvolen AlmaLinux, přičemž samotný proces jeho instalace není předmětem této práce.

Po úspěšné instalaci systému je nutné provést úpravy jeho konfigurace tak, aby bylo možné provozovat záznamovou aplikaci s požadovanou propustností a spoleh-



Obr. 6.2: Umístění slotů pro připojení NVMe disků nebo síťové karty 10 *GbE* [45]



Obr. 6.3: Ukázka připojení dvou NVMe disků k MIO-4370 [45]

livostí. Veškeré nezbytné úpravy systémových parametrů jsou podrobně popsány v kapitolách 3.2.2, 3.2.3 a 4.2.2.

Jakmile je konfigurace systému dokončena, je možné do vybraného adresáře zkopírovat záznamovou aplikaci, která je součástí elektronické přílohy této práce. Následně lze dle postupu uvedeného v kapitole 3.3.3 aplikaci přeložit a spustit. Ve stejné kapitole je rovněž popsán způsob zadání argumentů pro správné spuštění a příklady konkrétních příkazů.

Závěr

Výsledkem práce je plně funkční návrh softwarové části přenosného záznamového zařízení, podložený praktickými měřeními, analýzou současných technologií a návrhem vhodné hardwarové architektury.

Na začátku práce jsou popsány základní architektury běžně používané pro záznam vysokorychlostního síťového provozu a detailně rozebrány jednotlivé technologie, které se k tomuto účelu využívají – od metod přímého přístupu k síťovým datům až po způsoby jejich ukládání a analýzy.

V rámci rešerše byla provedena analýza dostupných profesionálních záznamových zařízení, jako jsou PacketRAID 100G, ProfiShark 10G nebo PacketExpert 100G. Byla zhodnocena z hlediska výkonu, ceny, rozměrů, mobility a složitosti použití. Výsledky ukázaly, že většina komerčních zařízení je cenově velmi náročná a často vyžaduje externí napájení či specializovaný hardware. Proto byl navržen vlastní koncept zařízení, který spojuje výkonný, ale dostupný embedded hardware a softwarové řešení přizpůsobené konkrétnímu účelu.

Původně byla zvolena architektura založená na běžném notebooku, ke kterému byla připojena externí síťová karta a NVMe úložiště. Toto řešení se ukázalo jako funkční a snadno dostupné, avšak s omezenou rozšiřitelností a méně stabilními výkonovými rezervami. Během návrhu a testování se proto ukázalo, že vhodnější volbou pro finální zařízení je využití embedded výpočetní platformy – z důvodu vyšší stability, lepší rozšiřitelnosti a možnosti trvalého zabudování všech komponent. Tomu odpovídá i závěrečná část práce, kde je podrobně popsán výběr konkrétních hardwarových prvků a postup sestavení kompletního záznamového zařízení s embedded počítačem jako výpočetní jednotkou.

Práce se zaměřila především na vývoj softwaru, protože se tato část ukázala jako kriticky důležitá pro spolehlivost celého řešení. Správné nastavení zachytávání, vyrovnávacích pamětí a zápisu se v praxi ukázalo jako klíčové pro dosažení nulové ztrátovosti, a proto se většina úsilí soustředila právě na tuto oblast.

Navržená aplikace byla testována na různých přenosových rychlostech až do 2,5 Gbps, přičemž při optimalizovaném nastavení systémových bufferů, odstranění blokující funkce *usleep()* a zavedení dávkového zápisu do souboru dosahovala nulové ztrátovosti. Aplikace dosahovala výborných výsledků zejména při záznamu spojitého toku s velkými rámci (např. jumbo pakety o velikosti 9000 bajtů), kde se ztrátovost pohybovala na úrovni 0 % i při rychlostech přes 2 Gbps. Při testech se zvýšeným počtem menších paketů (vyšší PPS) se však ukázalo, že ztráty mohou vznikat zejména v úvodní fázi běhu aplikace – tedy během inicializace, kdy jádro již zachytává provoz, ale aplikace ještě nezpracovává data. Ve srovnání s nástroji tcpdump a Wireshark byla aplikace stále velmi výkonná, ale rozdíly nebyly vždy jednoznačně ve prospěch

navrženého řešení – například při menších paketech a vyšším *PPS* vykazoval Wireshark v některých konfiguracích nulovou ztrátovost, zatímco vyvíjená aplikace znamenala několik tisíc zahozených paketů při spuštění. Tato zjištění naznačují, že efektivita záznamu závisí nejen na přenosové rychlosti, ale i na konkrétním průběhu toku a režii během startu aplikace.

Ztráty paketů pozorované při zahájení záznamu jsou způsobeny prodlevou mezi otevřením socketu a plným spuštěním hlavní zapisovací smyčky. V tomto časovém okně dochází k plnění bufferu jádrem, zatímco aplikace ještě nezačala data zpracovávat, což může vést k přetečení a zahození části příchozího provozu. Tento problém by bylo možné dále minimalizovat optimalizací inicializační sekvence aplikace, nebo zvýšením velikosti kruhového bufferu tak, aby nedošlo k přetečení ještě před zahájením zapisování. Alternativním řešením by mohl být přechod na novější verzi rozhraní *AF_PACKET* typu *TPACKET_V3*, které poskytuje lepší kontrolu nad bufferem a podporuje tzv. block-based processing, což umožňuje efektivnější práci se zahájením příjmu paketů a může zamezit ztrátám způsobeným právě v úvodní fázi běhu aplikace.

Vyšší přenosové rychlosti (například 10 *Gbps*) nebyly testovány z důvodu nedostupnosti potřebného vybavení – konkrétně síťových adaptérů a výkonnější embedded výpočetní platformy, která by umožňovala provoz při těchto parametrech.

Jedním z klíčových přínosů navrženého řešení je jeho výrazně nižší cena ve srovnání s komerčně dostupnými alternativami. Celkové náklady na sestavení zařízení činí přibližně 20 000 Kč bez DPH v základní konfiguraci (včetně desky MIO-4370, 32 GB RAM, 4 TB NVMe disku a volitelné 10 *GbE* karty). I po započítání periférií a výroby vhodné skříně pro umístění všech komponent – například pomocí 3D tisku, jehož cena pro rozměry okolo 20×20×10 cm běžně nepřesahuje 2 000 Kč – zůstává výsledná cena výrazně nižší než u srovnatelných profesionálních řešení. Zvolená embedded platforma navíc umožňuje zvýšit výpočetní výkon díky patičovému připojení procesoru. Další výrazné zlepšení výkonu formou upgradu RAM nebo rychlejší síťové karty však není možné, protože pro daný konektorový formát se výkonnější moduly nevyrábějí. Pokud by byl požadavek na vyšší síťovou propustnost (např. 25–40 *GbE*), bylo by nutné přejít na jiný model embedded počítače s rozhraním Thunderbolt 3/4, pro které jsou dostupné kompatibilní síťové karty. Celkové náklady na takové řešení by se i v rozšířené konfiguraci stále pohybovaly pod hranicí 25 000 Kč.

Ačkoli navržené řešení vykazuje velmi dobré výsledky při záznamu více-gigabitového síťového provozu, je třeba zmínit i některá rizika a technická omezení. Aplikace je v aktuální podobě jednovláknová a běží na jednom jádru procesoru, což zvyšuje nároky na výkon CPU, zejména u embedded zařízení s omezeným výpočetním výkonem. Využití více vláken by mohlo výrazně snížit zatížení a zvýšit stabilitu aplikace

při vyšších přenosových rychlostech. Dále je třeba počítat s možným výkonovým limitem funkce *memcpy()* při zápisu dat, která může být úzkým hrdlem na méně výkonných embedded systémech. Dalším potenciálním omezením je nutnost správného nastavení velikosti ring bufferu, která musí být zvolena s ohledem na konkrétní provozní scénář – příliš malá vede k přetečení a ztrátám, příliš velká zatěžuje paměť a zpomaluje inicializaci.

Za riziko lze také považovat ztrátovost paketů během startu aplikace, kdy dochází k plnění bufferu dříve, než začne probíhat jejich zpracování. Tento problém by bylo možné řešit zefektivněním úvodní fáze programu nebo zavedením časově řízeného zpoždění generace provozu. Pro celkové zlepšení efektivity a škálovatelnosti by se navíc do budoucna nabízel přechod na novější variantu rozhraní *AF_PACKET* – konkrétně *TPACKET_V3* – která umožňuje blokové zpracování paketů, lepší řízení bufferu a může přinést vyšší stabilitu při provozu v náročných síťových podmínkách.

Další rozšíření navrženého řešení by mohlo zahrnovat vytvoření grafického uživatelského rozhraní (GUI) nebo webového serveru, který by umožňoval zobrazovat reálné statistiky, sledovat stav běžící aplikace, využití systémových prostředků a zároveň umožnit vzdálené ovládání, například spuštění a zastavení záznamu. Praktickým vylepšením by také bylo automatické rozdělování zachytávaných dat do více souborů podle velikosti nebo času, což by usnadnilo práci s výstupními daty, jejich archivaci a analýzu.

Literatura

- [1] CLOUDFLARE, INC. *What is the OSI Model? Online*. CLOUDFLARE, INC. What is the OSI Model? 2018. Dostupné z: <https://www.cloudflare.com/learning/ddos/glossary/open-systems-interconnection-model-osi/>. [cit. 2024-09-30].
- [2] FORTINET. *OSI Model Explained*. Online. FORTINET. OSI Model Explained. 2021. Dostupné z: <https://www.fortinet.com/resources/cyberglossary/osi-model>. [cit. 2024-10-01].
- [3] AWS. *What is OSI Model?* Online. AWS. What is OSI Model? Dostupné z: https://aws.amazon.com/what-is/osi-model/?nc1=h_ls. [cit. 2024-10-01].
- [4] CBTNUETS. *What is the Ethernet Frame Format?* Online. CBTNUETS. What is the Ethernet Frame Format? 2024. Dostupné z: <https://www.cbtnuggets.com/blog/technology/networking/what-is-ethernet-frame-format>. [cit. 2024-10-03].
- [5] QSFPTTEK. *What Are Jumbo Frames? Definition and Configuration Guidance*. Online. QSFPTTEK. What Are Jumbo Frames? Definition and Configuration Guidance. Dostupné z: <https://www.qsfptek.com/qt-news/what-are-jumbo-frames.html>. [cit. 2024-12-16].
- [6] STONEFLY. *Ultimate Guide to JUMBO Frames: Configuration and Best MTU Size*. Online. STONEFLY. Ultimate Guide to JUMBO Frames: Configuration and Best MTU Size. 2023. Dostupné z: <https://stonefly.com/resources/jumbo-frames-configuration-and-best-mtu-size/>. [cit. 2024-12-16].
- [7] IBM. *Internet Protocol*. Online. IBM. Internet Protocol. 2021, 2024-08-27. Dostupné z: <https://www.ibm.com/docs/en/aix/7.1?topic=protocols-internet-protocol>. [cit. 2024-12-17].
- [8] MAX. *What Is a Basic Characteristic of the IP Protocol?* Online. MAX. What Is a Basic Characteristic of the IP Protocol? Dostupné z: <https://maxtrain.com/2024/02/05/what-is-a-basic-characteristic-of-the-ip-protocol/>. [cit. 2024-12-17].
- [9] GFG. *Transport Layer responsibilities*. Online. GFG. Transport Layer responsibilities. 2020, 2024-09-30. Dostupné z: <https://www.geeksforgeeks.org/transport-layer-responsibilities/>. [cit. 2025-01-03].

- [10] PUNNUB. *What is User Datagram Protocol (UDP)?* Online. PUNNUB. What is User Datagram Protocol (UDP)? 2024. Dostupné z: <https://www.pubnub.com/guides/udp/>. [cit. 2024-12-16].
- [11] CLOUDDNS. *UDP (User Datagram Protocol) explained in details.* Online. CLOUDDNS. UDP (User Datagram Protocol) explained in details. 2022. Dostupné z: <https://www.cloudns.net/blog/udp-user-datagram-protocol-explained-in-details/>. [cit. 2024-12-16].
- [12] NETRESEC. *What is a PCAP file?* Online. NETRESEC. What is a PCAP file? 2022. Dostupné z: <https://www.netresec.com/?month=2022-10&page=Blog&post=What-is-a-PCAP-file>. [cit. 2024-10-03].
- [13] ENDACE. *What is a PCAP file?* Online. ENDACE. What is a PCAP file? 2023. Dostupné z: <https://www.endace.com/learn/what-is-a-pcap-file>. [cit. 2024-10-03].
- [14] *PcapNG File Format.* Online. PcapNG File Format. 2013. Dostupné z: <https://pcapng.com/>. [cit. 2024-10-04].
- [15] WIRESHARK. *Extensible Record Format (ERF).* Online. WIRESHARK. Extensible Record Format (ERF). 2014. Dostupné z: <https://wiki.wireshark.org/ERF>. [cit. 2024-10-20].
- [16] WIRESHARK. *The world's most popular network protocol analyzer.* Online. WIRESHARK. The world's most popular network protocol analyzer. 2006. Dostupné z: <https://www.wireshark.org/>. [cit. 2024-12-14].
- [17] SCAPY. *About Scapy. Online.* SCAPY. About Scapy. 2016. Dostupné z: <https://scapy.readthedocs.io/en/latest/introduction.html>. [cit. 2025-01-03].
- [18] PENTEST. *What is Scapy?* Online. PENTEST. What is Scapy? 2023. Dostupné z: <https://pentestmag.com/what-is-scapy/>. [cit. 2024-12-20].
- [19] HASONS. *Network Cards: Unleashing the Full Potential of Your Internet Connection.* Online. HASONS. Network Cards: Unleashing the Full Potential of Your Internet Connection. 2024. Dostupné z: <https://hasonss.com/blogs/network-cards/>. [cit. 2024-12-12].
- [20] TRENTON SYSTEMS. *What is a NIC Card (Network Interface Card)?* Online. TRENTON SYSTEMS. What is a NIC Card (Network Interface Card)? 2024. Dostupné z: <https://www.trentonsystems.com/en-us/resource-hub/blog/nic-card>. [cit. 2024-12-15].

- [21] HOMEVIS. *Does RJ45 Support 10G?* Online. HOMEVIS. Does RJ45 Support 10G? Dostupné z: <https://howevision.com/does-rj45-support-10g/>. [cit. 2024-12-06].
- [22] ASCENTOPTICS. *The Ultimate Guide to SFP Modules, Connectors, and Transceivers for 1000BASE-T RJ45 SFP Technology*. Online. ASCENTOPTICS. The Ultimate Guide to SFP Modules, Connectors, and Transceivers for 1000BASE-T RJ45 SFP Technology. 2024. Dostupné z: https://ascentoptics.com/blog/the-ultimate-guide-to-sfp-modules/#Aspects_to_Consider_Data_Rate_Distance_and_Connector_type. [cit. 2024-12-02].
- [23] FS. *XFP vs SFP+: What Are the Differences?* Online. FS. XFP vs SFP+: What Are the Differences? 2022. Dostupné z: <https://community.fs.com/article/can-we-connect-xfp-and-sfp.html>. [cit. 2024-12-04].
- [24] PROFIBER NETWORKING. *Přehled typů optických konektorů*. Online. Přehled typů optických konektorů. 2020-11, 2023-05. Dostupné z: <https://www.profiber.eu/sk/aplikacie/prehled-typu-opticky-konektoru/>. [cit. 2024-12-07].
- [25] LUMENCI. *Thunderbolt Interface*. Online. LUMENCI. Thunderbolt Interface. Dostupné z: <https://lumenci.com/blogs/thunderbolt-interface/>. [cit. 2024-11-17].
- [26] EATON. *Thunderbolt 4 Explained*. Online. EATON. Thunderbolt 4 Explained. 2022. Dostupné z: <https://tripplite.eaton.com/products/thunderbolt-4>. [cit. 2024-11-17].
- [27] TECHTARGET. *USB 3.0 (SuperSpeed USB)*. Online. TECHTARGET. USB 3.0 (SuperSpeed USB). 2022. Dostupné z: <https://www.techtarget.com/searchwindowsserver/definition/USB-30-SuperSpeed-USB>. [cit. 2024-10-15].
- [28] SSSTC. *Comprehensive Guide to SSDs: Advantages and Key Features*. Online. SSSTC. Comprehensive Guide to SSDs: Advantages and Key Features. 2024, 2024-01-31. Dostupné z: <https://www.ssstc.com/knowledge-detail/guide-to-ssds-advantages-and-features/>. [cit. 2024-12-15].
- [29] IBM. *NVMe vs. SATA: What's the difference?* Online. IBM. NVMe vs. SATA: What's the difference? 2023. Dostupné z: <https://www.ibm.com/think/topics/nvme-vs-sata>. [cit. 2025-01-01].

- [30] NVMEXPRESS. *What is NVMe Technology?* Online. NVMEXPRESS. What is NVMe Technology? 2011, 2024-12-29. Dostupné z: <https://nvmexpress.org/>. [cit. 2024-12-29].
- [31] STELLAR. *Advantages and Disadvantages of RAID Levels*. Online. STELLAR. Advantages and Disadvantages of RAID Levels. 2020. Dostupné z: <https://www.stellarinfo.co.in/blog/advantages-and-disadvantages-popular-raid-systems/>. [cit. 2025-01-01].
- [32] RAID RECOVERY SPECIALISTS. *THE ADVANTAGES AND DISADVANTAGES OF SSD RAID*. Online. RAID RECOVERY SPECIALISTS. THE ADVANTAGES AND DISADVANTAGES OF SSD RAID. 2023. Dostupné z: <https://www.raiddatarecoveryspecialists.co.uk/blog/the-advantages-and-disadvantages-of-ssd-raid>. [cit. 2025-01-01].
- [33] THE PACKET COMPANY. *PacketRAID 100G*. THE PACKET COMPANY. The Packet Company [online]. [cit. 2024-10-29]. Dostupné z: <https://packet.company/packetraid100g>
- [34] Profitap HQ B.V. *ProfiShark 10G*. Online. PROFITAP HQ B.V. ProfiShark 10G. © 2024. Dostupné z: <https://www.profitap.com>. [cit. 2024-10-29].
- [35] NTOP. *N2disk™*. Online. NTOP. N2disk™. 2015. Dostupné z: <https://www.ntop.org/products/traffic-recording-replay/n2disk/>. [cit. 2024-10-03].
- [36] GL COMMUNICATIONS INC. *PacketExpert™ 100G - Comprehensive Ethernet/IP Testing Solution*. Online. GL COMMUNICATIONS INC. PacketExpert™ 100G - Comprehensive Ethernet/IP Testing Solution. 2023, 2024-07-29. Dostupné z: <https://www.gl.com/packetexpert-100g-ip-ethernet-network-tester.html>. [cit. 2024-10-15].
- [37] FMADIO. *FMADIO20-PORTABLE*. Online. FMADIO. FMADIO20-PORTABLE. 2022. Dostupné z: <https://www.fmad.io/10g-packet-capture-portable>. [cit. 2025-01-02].
- [38] MIKROTIK. *RB5009UG+S+IN*. Online. Dostupné z: https://mikrotik.com/product/rb5009ug_s_in. [cit. 2025-04-26].
- [39] MIKROTIK. *RB5009UG+S+IN*. Online. Dostupné z: <https://help.mikrotik.com/docs/spaces/UM/pages/78086262/RB5009UG+S+IN>. [cit. 2025-04-26].

- [40] LENOVO. *IdeaPad Gaming 3 15ACH6*. Online. 2024. Dostupné z: https://psref.lenovo.com/syspool/Sys/PDF/IdeaPad/IdeaPad_Gaming_3_15ACH6/IdeaPad_Gaming_3_15ACH6_Spec.pdf. [cit. 2025-05-10].
- [41] PALMSHELL. *NeXT H2*. Online. Dostupné z: <https://palmshell.io/next-h2>. [cit. 2025-05-20].
- [42] MAXTANG. *MTN-FP750*. Online. C2014-2025. Dostupné z: <https://www.maxtangpc.com/show-106-225-1.html>. [cit. 2025-05-21].
- [43] ADVANTECH. *MIO-5379*. Online. C1983-2025. Dostupné z: https://www.advantech.com/en-us/products/bc6786b7-b69b-4fdd-9b63-79180dbd5dcb/mio-5379/mod_132c88ac-36fc-47d8-8095-3a7b79b59443. [cit. 2025-05-21].
- [44] ADVANTECH. *MIO-2364*. Online. C1983-2025. Dostupné z: https://www.advantech.com/en-us/products/460a67de-a7c8-94dc-0809-336fd7570e46/mio-2364/mod_6e929c2c-6213-4586-b747-3c7b7fd0dabe. [cit. 2025-05-21].
- [45] ADVANTECH. *MIO-4370*. Online. C1983-2025. Dostupné z: https://www.advantech.com/en-us/products/b7b60b64-d034-4cdb-860b-538790b86961/mio-4370/mod_a400b96d-7de3-4b04-9a43-872174d74007. [cit. 2025-05-21].
- [46] PASSMARK SOFTWARE. *CPU Benchmarks*. Online. C2025. Dostupné z: <https://www.cpubenchmark.net/compare/4955vs4274/Intel-i3-12100TE-vs-AMD-Ryzen-5-5600H>. [cit. 2025-05-21].
- [47] MAUSER ELECTRONICS. *MIO-4370H-00A1*. Online. C2025. Dostupné z: https://eu.mouser.com/ProductDetail/Advantech/MIO-4370H-00A1?qs=ST9lo4GX8V2VrE5GqnbN%2FA%3D%3D&srsltid=AfmB0oqNKNZW2VygjXffZbVERbZYJ_w3mop-RJlrtq7i8L0BbCQGWI1I. [cit. 2025-05-25].
- [48] STACK OVERFLOW. *Libpcap to capture 10 Gbps NIC*. Online. 28. 2. 2019. Dostupné z: <https://stackoverflow.com/questions/7763321/libpcap-to-capture-10-gbps-nic>. [cit. 2025-05-30].
- [49] GITHUB. *Maximizing Packet Capture Performance with Tcpdump*. Online. 2020, 26. 11. 2020. Dostupné z: <https://satishdotpatel.github.io/maximizing-packet-capture-performance-with-tcpdump/>. [cit. 2025-05-30].

- [50] WIRESHARK. *Wireshark FAQ – Is Wireshark Open Source?*. Online. Dostupné z: <https://www.wireshark.org/faq.html#q1.2>. [cit. 2025-05-30].
- [51] GNU OPERATING SYSTEM. *GNU General Public License v2.0 – Frequently Asked Questions*. Online. Dostupné z: <https://www.gnu.org/licenses/old-licenses/gpl-2.0-faq.en.html>. [cit. 2025-05-30].
- [52] SILICOM LTD. *FPGA Solutions*. Online. Dostupné z: <https://www.silicom-usa.com/cats/fpga-based-cards/>. [cit. 2025-05-30].
- [53] MACNICA. *High speed packet capture solution*. Online. Dostupné z: <https://www.macnica.co.jp/en/business/semiconductor/manufacturers/napatech/products/134161/>. [cit. 2025-05-30].
- [54] DATART. *Síťová karta Axagon USB-C 3.2 Gen 1 – 2.5 Gigabit Ethernet (ADE-25RC) šedá*. Online. C2025. Dostupné z: <https://www.datart.cz/sitova-karta-axagon-usb-c-3-2-gen-1-2-5-gigabit-ethernet-ade-25rc-seda.html>. [cit. 2025-05-31].
- [55] ALZA. *Samsung 990 EVO Plus 1TB*. Online. C1994-2025. Dostupné z: <https://www.alza.cz/samsung-990-evo-plus-1tb-d12611064.htm>. [cit. 2025-05-31].
- [56] SERVERFAULT. *Lots of dropped packages when tcpdumping on busy interface*. Online. 2013, 9. 11. 2017. Dostupné z: <https://serverfault.com/questions/421789/lots-of-dropped-packages-when-tcpdumping-on-busy-interface>. [cit. 2025-05-31].
- [57] WIRESHARK. *Dumpcap/TCPDump packet loss when writing data to disk with high traffic*. Online. 2021, 12. 10. 2021. Dostupné z: <https://ask.wireshark.org/question/24768/dumpcaptcpdump-packet-loss-when-writing-data-to-disk-with-high-traffic/>. [cit. 2025-05-31].
- [58] ALZA. *Samsung 990 PRO 4TB*. Online. C1994-2025. Dostupné z: <https://www.alza.cz/samsung-990-pro-4tb-d7930228.htm>. [cit. 2025-05-31].
- [59] ALZA. *Crucial SO-DIMM 32GB DDR5 4800MHz CL40*. Online. C1994-2025. Dostupné z: <https://www.alza.cz/crucial-so-dimm-32gb-ddr5-4800mhz-cl40-d7987959.htm>. [cit. 2025-05-31].

- [60] FRUUGO. *Ethernet NIC Card M.2 to 10Gb BM Key 82599-1 SFP Industrial Control Server Network Card*. Online. C2006-2025. Dostupné z: <https://www.fruugo.cz/ethernet-nic-card-m2-to-10gb-bm-key-82599-1-sfp-industrial-control-server-p-344915042-754894574>. [cit. 2025-05-31].
- [61] WIKIPEDIA. *Microsoft Windows*. Online. 2005, 1. 6. 2025. Dostupné z: https://en.wikipedia.org/wiki/Microsoft_Windows. [cit. 2025-06-02].
- [62] WIKIPEDIA. *Linux*. Online. 2005, 8. 2. 2025. Dostupné z: <https://cs.wikipedia.org/wiki/Linux>. [cit. 2025-06-02].
- [63] WIKIPEDIA. *Linux (jádro)*. Online. 2006, 27. 5. 2025. Dostupné z: [https://cs.wikipedia.org/wiki/Linux_\(j%C3%A1dro\)](https://cs.wikipedia.org/wiki/Linux_(j%C3%A1dro)). [cit. 2025-06-02].
- [64] ŠUBRT, Pavel. 2025. Osobní sdělení.
- [65] ALMALINUX. *Free Linux OS for the community, by the community*. Online. Dostupné z: <https://almalinux.org>. [cit. 2025-06-02].

Seznam symbolů a zkratek

Zkratky:

GbE	gigabit ethernet – Gigabit Ethernet
Gbps	gigabit za sekundu – Gigabit per second
Mbps	megabit za sekundu – Megabit per second
OSI	Open Systems Interconnection
IP	Internet Protocol
URL	Uniform Resource Locator
DNS	Domain Name System
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
LAN	Local Area Network
IDB	Interface Description Block
CPU	Central Processing Unit
NVMe	Non-Volatile Memory Express
PCI Express	Peripheral Component Interconnect Express
SSD	Solid State Drive
HDD	Hard Disk Drive
SATA	Serial Advanced Technology Attachment
RAID	Redundant Array of Independent Disks
FPGA	Field Programmable Gate Array
PPS	pakety za sekundu – Packets Per Second
MTU	maximální velikost přenášeného rámce – Maximum Transmission Unit

CRC	cyklický redundantní součet – Cyclic redundancy check
FCS	Frame control sequence
DHCP	Dynamic Host Configuration Protocol
SODIMM	Small Outline Dual In-line Memory Module
GUI	Grafické uživatelské rozhraní – Graphical User Interface

Symboly:

W	watt
kg	kilogram
V	volt
A	ampér
Mb	megabit
MB	megabajt
Gb	gigabit
GB	gigabajt
Tb	terabit
TB	terabajt
ms	milisekunda
Kč	koruna česká

A ZDROJOVÝ KÓD APLIKACE (ELEKTRO- NICKÁ PŘÍLOHA)

Tato příloha je dostupná pouze v elektronické podobě na přiloženém CD.