VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

SYSTÉM PRO OCHRANU PŘED DOS ÚTOKY

SYSTEM FOR PROTECTION AGAINST DOS ATTACKS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **PAVEL ŠIŠKA**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JAN KUČERA**
SUPERVISOR

BRNO 2018

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Šiška Pavel**
Obor: Informační technologie
Téma: **Systém pro ochranu před DoS útoky**
System for Protection against DoS Attacks
Kategorie: Počítačové sítě

Pokyny:

1. Seznamte se s problematikou útoků typu odepření služby a firmwarem pro síťovou kartu COMBO-100G zaměřeným na ochranu před těmito útoky.
2. Navrhněte systém využívající prostředků hardwarové akcelerace, který umožní efektivně blokovat vybrané typy útoků. Při návrhu se zaměřte na modulární architekturu systému, která zajistí také snadnou rozšiřitelnost systému v budoucnosti.
3. Navržený systém implementujte. Proveďte integraci jednotlivých součástí a důkladně zdokumentujte programové rozhraní pro přidávání rozšiřujících modulů.
4. Ověřte funkčnost a výkonové parametry implementovaného řešení (propustnost, latenci).
5. Diskutujte dosažené výsledky a možnosti dalšího pokračování práce.

Literatura:
- Dle pokynů vedoucího.

Pro udělení zápočtu za první semestr je požadováno:
- Splnění bodů 1 a 2 zadání.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Kučera Jan, Ing.**, UPSY FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

prof. Ing. Lukáš Sekanina, Ph.D.
vedoucí ústavu

Abstrakt

Bakalářská práce se zabývá návrhem a implementací softwarové části systému pro ochranu před DoS útoky. Útoky typu Denial of Service jsou v dnešní době velmi rozšířené a jejich úspěšné provedení může způsobit nemalé finanční škody provozovatelům služeb i poskytovatelům připojení. Hlavním cílem této práce bylo vytvoření softwaru, zaměřeného na vysokou datovou propustnost, který poskytne účinnou ochranu proti těmto útokům, a který umožní nasazení v sítích o rychlosti až 100 Gbps. Klíčovou částí celého systému, vyvíjeného ve spolupráci se sdružení CESNET, je hardwarově akcelerovaná síťová karta, která zpracovává přijímaná síťová data na plné rychlosti linky a realizuje operace určené softwarovou částí. Úkolem softwaru je přitom periodické vyhodnocování získaných informací o síťovém provozu a řízení činnosti hardwarového akcelerátoru. V rámci práce byl proveden detailní návrh softwarové části systému a jeho implementace. Dosažené vlastnosti vytvořené implementace byly následně ověřeny v rámci laboratorního testování. Takto vytvořený systém byl přitom již v době psaní bakalářské práce pilotně nasazen v síťové infrastruktuře akademické sítě CESNET.

Abstract

This bachelor's thesis deals with the design and implementation of the software part of the system for protection against DoS attacks. Nowadays Denial of Service attacks are quite common and can cause significant financial damage to internet or service providers. The main goal of this thesis was to provide software, which is focused on high-speed data throughput and can provide efficient protection against these attacks in 100 Gbps networks. Key part of the system, which is being developed in cooperation with CESNET, is hardware-accelerated network interface card, which can process incoming network traffic at full wire-speed and does the operations laid down by the software part. The main task of the software is evaluation of the information about network traffic and managing actions of the hardware accelerator. The software part of the proposed system has been successfully implemented and the properties of the system have been verified in an experimental evaluation. During the work on this thesis the first implementation of the system has already been deployed in CESNET network infrastructure.

Klíčová slova

(D)DoS Protector, odepření služby, Ochrana před útoky, COMBO-100G, 100 Gbps, CESNET, vysokorychlostní sítě, zpracování síťových dat, filtrace paketů

Keywords

(D)DoS Protector, Denial of Service, Protection Against Attacks, COMBO-100G, 100 Gbps, CESNET, High-speed Networks, Network Traffic Processing, Packet Filtering

Citace

ŠIŠKA, Pavel. *Systém pro ochranu před DoS útoky.* Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jan Kučera

Systém pro ochranu před DoS útoky

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jana Kučery. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Pavel Šiška
17. května 2018

Poděkování

Rád bych poděkoval vedoucímu Ing. Janu Kučerovi za odborné vedení, poskytnutí množství cenných rad, připomínek a návrhů při tvorbě této bakalářské práce. Dále by jsem chtěl poděkovat pracovníkům z oddělení nástrojů pro monitoring a konfiguraci ze sdružení CESNET za poskytnutí odborné pomoci a podpory. V neposlední řadě děkuji také rodině a kamarádům za jejich podporu po celou dobu mého bakalářského studia.

Obsah

Kapitola 1

Úvod

Žijeme v době rychlého technického rozvoje, ve které jsou informační technologie nedílnou součástí každodenního života. Velký rozvoj se v posledních letech nevyhnul ani počítačovým sítím, zejména tedy internetu. Připojení k internetu a využívání služeb, jako je např. sdílení souborů, internetové bankovnictví, přehrávání videa či prohlížení webových stránek je dnes pro většinu populace již samozřejmostí. Občas se ale stane, že je některá služba nedostupná. Nedostupnost služby přitom může být způsobena hned několika důvody, např. problémem s připojením, pravidelnou údržbou systému, ale i možným kybernetickým útokem. Útoky, jejichž cílem je právě vyřazení služby z činnosti se nazývají DoS (Denial of Service).

Úlohou DoS útoků je především zamezení přístupu legitimním uživatelům k poskytovaným službám, což může jejich provozovatelům způsobit nemalé finanční ztráty. Efektivní ochranu proti takovým (často distribuovaným) útokům nelze jednoduše zajistit. Odhalení útoků zneužívajících konkrétních zranitelností služeb nebo vyčerpání systémových prostředků vyžaduje detailní analýzu síťového provozu. Proto je možné takové útoky řešit pouze v koncových sítích, kde není takový objem dat a je zde možné provádět daleko důkladnější analýzu, přímo na úrovni jednotlivých paketů. Naproti tomu cílem volumetrických DDoS útoků nebývají typicky samotná koncová zařízení, ale přímo celá síťová infrastruktura (firma, organizace), kde už útočník cílí na omezenou kapacitu linek. V případě zahlcení linky koncové sítě je filtrování provozu až přímo v koncové síti zcela neúčinné. Tento problém je proto potřebné řešit už na úrovni páteřní sítě, kde jsou linky s dostatečnou kapacitou, které nejsou takovým útokem zahlceny. Běžná síťová zařízení v současnosti však neposkytují dostatečný výkon pro pokročilé operace umožnující filtraci nebo alespoň omezení takového útoku při vysokých rychlostech (řádově až 100 Gbps). Řešením tohoto problému je využití systému, který pomocí hardwarově akcelerované síťové karty dokáže takto složité operace vykonávat i při plné rychlosti 100 Gbps. Sdružení CESNET v současné době takový systém vyvíjí. A právě vytvoření řídícího a obslužného softwaru pro tento systém je cílem této práce.

Klíčovou částí celého systému je hardwarově akcelerovaná síťová karta, která bude zpracovávat přijímaná síťová data na plné rychlosti linky a realizovat operace určené softwarovou vrstvou. Úkolem softwaru bude periodické vyhodnocování získaných informací o síťovém provozu a řízení činnosti hardwarových komponent. Při návrhu byl kladen důraz především na dosažení požadované datové propustnosti pro možnost nasazení systému v sítích o rychlostech až 100 Gbps. Software byl dle návrhu úspěšně implementován a následně otestován v laboratorních podmínkách. Celý systém je plně funkční. Bylo ověřeno správné fungování nejen vytvořeného obslužného softwaru, ale i systému jako celku.

Text práce je rozdělen do několika částí logicky strukturovaných podle jednotlivých etap řešení zadaného problému. Kapitola 2 poskytuje teoretický úvod k praktické části práce a zaměřuje se na popis fungování počítačových sítí, kybernetických útoků typu odepření služby a nakonec se věnuje síťovým COMBO kartám. Tyto karty využívají technologii FPGA pro hardwarově akcelerované zpracování síťových dat a tvoří jednu z částí systému pro ochranu před DoS útoky, jehož konceptuální návrh je detailněji popsán v kapitole 3. Tato kapitola také přibližuje způsob zapojení systému do infrastruktury sítě. Kapitola 4 se zabývá samotným návrhem softwarové části systému. Tato část tvoří hlavní náplň této práce. Jsou zde uvedeny nejen veškeré požadavky, které musí dané řešení splňovat, ale je zde také důkladně popsán detailní návrh architektury celé softwarové části. Následující kapitola 5 nabízí podstatné informace týkající se výsledné implementace softwarové části systému. Dosažené výsledky získané z měření při testování v laboratorních podmínkách systému jako celku a výsledky z jednotlivých částí implementace jsou popsané v kapitole 6. Závěrečná kapitola 7 shrnuje celkově dosažené výsledky práce a poskytuje pohled na její další možné budoucí směřování.

Kapitola 2

Teoretický rozbor

Tato kapitola vysvětluje základní pojmy, které tvoří teoretický základ k praktické části práce. Nejprve jsou popsány základní principy počítačových sítí. Následuje popis útoku typu odepření služby včetně jednotlivých metod útoku. Závěr kapitoly se věnuje popisu technologie FPGA, síťové kartě COMBO a především firmwaru síťové karty, který je důležitý při realizaci praktické části této práce.

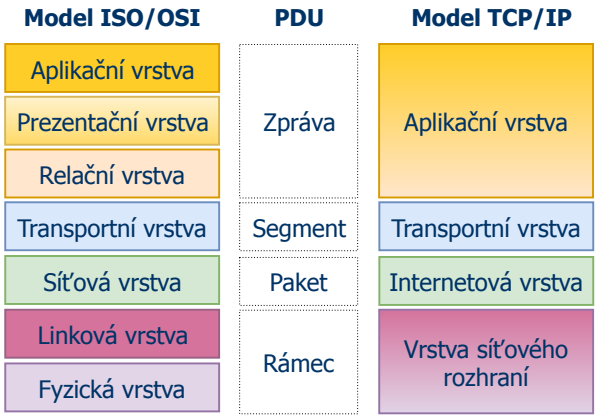
2.1 Principy a architektura počítačových sítí

Pojem počítačová síť je označení pro skupinu dvou a více zařízení, mezi kterými existuje logické spojení za účelem sdílení informací a zdrojů. Může se jednat jak o zdroje hardwarové, tak softwarové. Komunikace mezi zařízeními probíhá skrze přenosové médium. Jedná se typicky o síťové kabely (metalické, optické) nebo bezdrátové technologie (wifi, LTE). Přenos informací je realizován na principu binárního kódu (logická 0/1), který je fyzicky reprezentován hodnotou fyzikální veličiny (velikost elektrického napětí, vlnová délka elektromagnetického záření). Další nezbytnou součástí potřebnou pro komunikaci je síťový software, který zajišťuje přesuny dat, navazování spojení, zabezpečení apod. Není-li uvedeno jinak, vychází popis principů počítačových sítí z poznatků uvedených v literatuře [14].

Architektura počítačových sítí byla již od svého počátku navržena tak, aby odlišovala tři základní části komunikačního procesu – technologie pro přenos signálu, zajištění spolehlivého přenosu a aplikační vrstvu. Na základě tohoto rozdělení byl zaveden princip, tzv. řízení datové komunikace, který rozděluje komunikaci do několika vrstev, kde každá vrstva reprezentuje určitý krok v síťové komunikaci a vykonává jasně definovanou funkcionalitu. Daná vrstva k realizaci své funkcionality vždy využívá služeb nižší vrstvy a to bez toho, aniž by musela znát, jak či pomocí jakých protokolů jsou její funkce implementovány.

Pro popis síťové architektury se používá sedmivrstvý referenční model OSI (Open Systems Interconnect Reference Model), který byl definován v roce 1987 Mezinárodní standardizační organizací ISO (International Standards Organizatio). Na základě tohoto modelu má vytvořenou strukturu mnoho v současné době používaných protokolů, a proto lze tento model považovat za primární architekturu pro počítačovou komunikaci. Vzhledem na komplexnost modelu OSI byla v praxi implementována pouze část modelu.

Dalším modelem je TCP/IP, který se používá v drtivé většině dnešních počítačových sítí a také v té nejrozšířenější síti Internet. Model je založen na protokolech TCP (Transmission Control Protocol) a IP (Internet Protocol). Za vznikem modelu TCP/IP stojí projekt ARPAnet (Advance Research Projects Agency Network), který byl iniciován americkým



Obrázek 2.1: Porovnání referenčního modelu ISO/OSI a modelu TCP/IP

úřadem obrany (US Department of Defense, DoD) za dob studené války. Srovnání vrstev obou modelů je zobrazeno na obrázku 2.1. Architektura modelu TCP/IP je výrazně jednodušší než referenční model OSI a obsahuje pouze 4 vrstvy, narozdíl od 7 vrstev modelu OSI. Model TCP/IP spojuje služby prezentační a relační do jediné vrstvy aplikační. Na úrovni fyzického přenosu bitů spojuje fyzickou a linkovou vrstvu do vrstvy síťového rozhraní, která je obvykle implementována na síťové kartě. Každá vrstva definuje základní datovou jednotku pro přenos informací (PDU, Process Data Unit). Při přenosu dat po síti dochází na straně odesílatele k zapouzdření dat (encapsulation) vyšších vrstev do PDU nižších vrstev. Na straně příjemce potom dochází k opačnému procesu, tedy rozbalení dat (decapsulation), kde jsou data postupně rozbalena a předána vyšším vrstvám ke zpracování. Popis jednotlivých vrstev modelu TCP/IP od nejnižší vrstvy je popsán zde:

Vrstva síťového rozhraní je nejnižší vrstvou modelu TCP/IP. Vrstva zajišťuje ovládání síťových karet, skrz které přistupuje k přenosovému médiu (rádiové vlny, metalická a optická kabeláž) a provádí fyzický přenos dat. Vrstva také popisuje standardy pro fyzické médium a elektrické signály, které určují vlastnosti linky. Nejrozšířenějším protokolem na této vrstvě je Ethernet. Dalšími méně používanými protokoly jsou Frame Relay a Token Ring. Datové jednotky přenášeny touto vrstvo se nazývají rámce a pro jejich adresování se používá fyzická adresa. Na ethernetových sítích k adresaci slouží MAC (Media Access Control) adresa, která identifikuje síťovou kartu.

Internetová vrstva vytváří IP datagramy, takzvané pakety, a stará se o jejich správné doručení až na místo určení. Při doručování dat se usiluje o nalezení nejvhodnější cesty k adresátovi, tzv. doručení s největším úsilím (best-effort delivery), což ale nezaručuje spolehlivé doručení dat. V případě, že dojde po cestě ke koncovému uzlu ke ztrátě dat např. z důvodu výpadku uzlu či přeplněné fronty na směrovačích, vysílající uzel je o události informován a musí se postarat o opětovné přenesení dat. Proces hledání cesty ke koncovému uzlu se nazývá směrování a na této vrstvě se k tomu používá IP adresa. IP adresa je součástí IP protokolu, což je základní protokol této vrstvy. Dalším používaným protokolem je ARP (Address Resolution Protocol), který slouží k mapování IP adresy na MAC adresu, což je nezbytné pro správné fungování nižší vrstvy, která pracuje právě s MAC adresami. Protokol RIP (Routing Internet Protocol), potřebný pro získávání informací o směrování. Dále protokol ICMP (Internet Control Message Protocol), který slouží pro řízení toku dat a detekci nedosažitelných

uzlů a jako poslední uvedený protokol IGMP (Internet Group Management Protocol), který se používá pro přihlašování do multicastových skupin.

Transportní vrstva vytváří logické spojení mezi procesy nacházejícími se na koncových uzlech. Vrstva také zajišťuje rozdělení aplikačních dat na menší jednotky, tzv. segmenty a jejich posílání po síti. Základními protokoly na této vrstvě jsou TCP (Transmission Control Protocol) a UDP (User Datagram Protocol). Protokol TCP je spojovaný, tzn. že se před samotným odesláním dat musí nejprve ustanovit spojení. Ustanovení spojení vypadá následovně:

1. Klient iniciuje spojení se serverem. Klient pošle serveru první paket s nastaveným příznakem SYN (synchronizovat).
2. Server odpoví klientovi paketem, který má nastavený příznak SYN-ACK (synchronizovat potvrzení) a uloží si informaci o nadcházejícím spojení do interní datové struktury. Zároveň nastaví časovač, po který bude čekat na odpověd od klienta. Po tuto dobu bude udržovat informaci o spojení v paměti. Tomuto stavu se říká polootevřené spojení (half-open connection).
3. Klient odpoví serveru paketem s příznakem ACK (potvrzení). V tuto chvíli je spojení navázáno.

TCP protokol také garantuje spolehlivý přenos dat. Odesílaná data jsou označena sekvenčními čísly a jejich správné doručení je signalizováno příchodem potvrzení. TCP také disponuje mechanismy pro řízení toku dat a předcházení zahlcení. Protokol UDP na rozdíl od TCP neustanovuje spojení a nezajišťuje spolehlivý přenos dat. Komunikace tímto protokolem má nižší režii než TCP a umožňuje tak vyšší rychlost při přenosu dat. K adresaci komunikujících procesů slouží zdrojový a cílový port v záhlaví protokolů.

Aplikační vrstva je nejvyšší vrstvou modelu a její úlohou je posílání zpráv a interakce mezi komunikujícími aplikacemi. Vrstva také zajišťuje správnou reprezentaci dat a jejich kódování. Mezi protokoly pracující na této vrstvě se řadí protokol pro spolehlivý přenos souborů (FTP, File Transfer Protocol), protokol pro vzdálený přístup na počítač (TELNET, Terminal Emulation), protokol zajišťující přenos elektronické pošty (SMTP, Simple Mail Transfer Protokol), dále také protokol pro přenos webových stránek (HTTP, Hypertext Transfer Protocol) a další.

2.2 DoS útoky

Útok typu odepření služby neboli Denial of Service (dále jen DoS) se řadí do skupiny kybernetických útoků, jejichž cílem je znepřístupnění, znefunkčnění či omezení služby počítačového systému. Efektu odepření služby je možno dosáhnout dvěma způsoby. Prvním způsobem je využití vlastností jednotlivých síťových protokolů či chyby v programu cílové služby. Útočník pošle na cílovou službu speciálně upravené zprávy, které této vlastnosti či zranitelnosti využívají. Druhým způsobem je zahlcení služby posíláním obrovského množství zpráv, tím se spotřebovávají zdroje systému jako je paměť, šířka pásma a další. Služba po vyčerpání svých zdrojů není schopna vyřizovat příchozí požadavky a tyto zprávy jsou následně zahazovány. Ani v jednom z těchto případů se nejedná o ovládnutí služby útočníkem, ale o snahu narušit legitimní činnost služby. Služba postižená DoS útokem se jeví

jako pomalá nebo nedostupná. Nutno zmínit, že pokud služba vykazuje takovéto příznaky, ještě se nemusí jednat o DoS útok. DoS útoky se nejčastěji dělí na dva základní typy podle počtu zdrojů následovně:

Denial of Service (DoS): Tento typ útoku je charakteristický tím, že je iniciován pouze z jednoho útočícího zařízení. DoS útok je podskupinou ostatních typů útoků a často se tento název používá ve spojení se všemi útoky typu Denial of Service.

Distributed Denial of Service (DDoS): U tohoto typu útoku jsou útočící zařízení minimálně dvě, většinou však tisíce až statisíce. Síť takovýchto zařízení tvoří obvykle tzv. botnet, který je popsán dále v práci. Častým jevem je podvrhnutí zdrojových IP adres, tzv. IP spoofing. Velkou výhodou z pohledu útočníka je oproti DoS útoku několikanásobně vyšší síla, které útok může dosáhnout, a možnost kombinovat více útočících strategií najednou.

Jeden z prvních velkých DDoS útoků cílil v roce 1999 na univerzitní síť v Minnesotě, která byla na více jak dva dny nedostupná [7]. O zvýraznění DoS útoků se ve společnosti postaralo hlavně hnutí Anonymous [8], které se v roce 2010 podle článku [18] dostalo do povědomí hájením kauzy WikiLeaks. Hnutí Anonymous působí i v České republice, kde se hlásí k několika DoS útokům, které byly v minulosti vedeny [17][16]. V současné době se počet a síla DoS útoků neustále zvětšuje [10]. Jeden z největších útoků v historii dosahoval síly více než 1 Tb/s a podílel se na něm botnet více než 145 tisíc IoT[1] zařízení [20].

Obětí DoS útoku se může stát kdokoli s konektivitou k internetu, nicméně útočníci si oběti obvykle vybírají ze zištných důvodů:

Konkurenční boj: U tohoto motivu se firma, nebo jedinec snaží o znepřístupnění služby své konkurence z důvodu zvětšení svého podílu na trhu či kvůli poškození dobrého jména. Článek [19] popisuje útoky z Vánoc roku 2011.

Politický motiv: Tento motiv slouží k vyjádření nesouhlasu s konáním, názory či směřováním jedince, nebo organizace. Obětí útoku jsou obvykle webové stránky či služby vlastněné politickými stranami, vládami, mediálními weby či korporacemi. S politickým motivem je spojen i pojem hacktivismus[2].

Skrytí sekundárního útoku: Dalším motivem je odvedení pozornosti od jiné činnosti probíhající na síti. Obvykle se jedná o skrytí dalšího kybernetického útok.

Pomsta: Mstít se lze z více důvodů. Jedním z nich může být propuštění, kdy zaměstnanec cítí křivdu od svého bývalého zaměstnavatele, nebo se jedná například o nespokojeného zákazníka.

Vydírání: Obvykle za účelem finančního zisku, kdy útočník vydírá jedince či organizace tím, že jim znepřístupní webové stránky, nebo služby, které poskytují. Oběti jsou obvykle na těchto službách závislé a jejich nedostupnost jim může způsobit velké škody (internetové obchody).

[1] Internet of Things neboli internet věcí, moderní přístroje ovladatelné i na dálku pomocí internetu (kamery, žárovky, čidla).

[2] Nabourání počítačového systému za účelem politicky či společensky motivovaného důvodu. Hacktivistům jde o přilákání pozornosti k aktuálním politickým či sociálním tématům [15].

Uznání hackerské komunity: Posledním zde popsaným motivem je uznání provedeného útoku mezi hackery. Útočníci se předhání, kdo dokáže vyvinout větší sílu útoku. Dále také kdo dokáže znepřístupnit webové stránky či služby velkým organizacím a na jak dlouhou dobu.

Text následujícího odstavce popisující Botnet vychází z literatury [11], [5]. Botnet je skupina nebo síť zařízení, které jsou pod kontrolou tzv. C&C (command and control) serverů ovládaných útočníkem. Botnet lze využít pro plnění různých úkolů, které jim C&C servery dávají, jako je např. rozesílání spamu, těžba kryptoměn a dalších. I když má botnet mnoho využití, v této práci bude pojem botnet figurovat jako nástroj pro provádění DDoS útoků. Zařízení se stane součástí botnetu v okamžiku, kdy je infikováno škodlivým kódem tzv. malwarem. K infikování dochází například pomocí webové stránky, kdy po navštívení stránky začne stránka hledat zranitelnosti v systému k instalaci bota. Další z možností infikování je skrze připojený soubor v elektronické poště, nebo přímou instalací bota do systému. Takto postihnuté zařízení se obvykle nazývá zombie nebo bot. Po nainstalování se bot pokusí spojit s jedním z C&C serverů. Ke spojení je například použit aplikační protokol Internet Relay Chat (IRC), který slouží pro posílání textových zpráv mezi připojenými zařízeními. Komunikace probíhá skrze specifický kanál, který umožňuje zabezpečení pomocí hesla a šifrování. IRC kanál slouží ke kontrole a zasílání příkazů botům. Po úspěšném spojení se bot C&C serveru nahlásí a požádá o instrukce. Jakmile instrukce obdrží, začne zařízení provádět požadovanou činnost. Při DDoS útoku je možné na zařízení pozorovat příznaky, jako může být například zpomalení internetového připojení. Pokud bot instrukce neobdrží, přepne se do vyčkávacího módu, ve kterém se snaží nevzbuzovat zbytečnou pozornost.

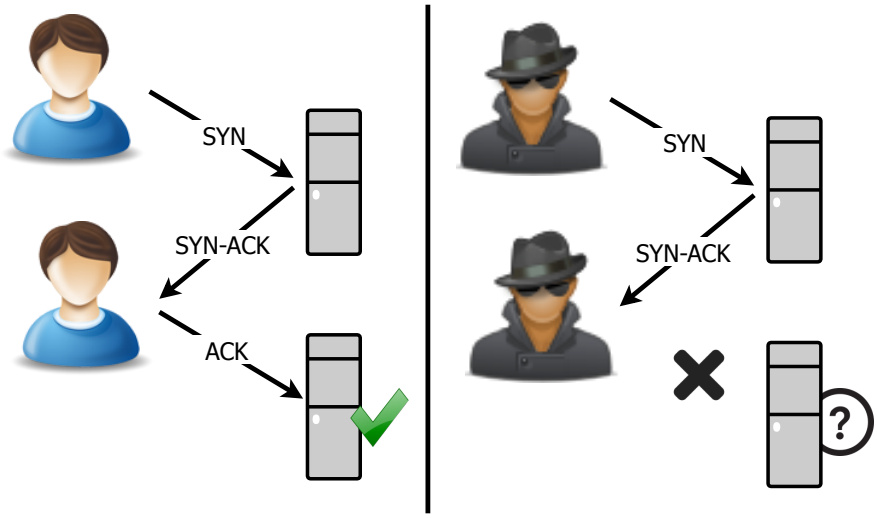
Rostoucí popularitě DoS útoků přispívá i to, že si je lze velice snadno objednat na internetu. Vlastníci botnetů nabízejí svá zotročená zařízení k pronájmu, případně přímo poskytují služby typu DoS. Potencionálním útočníkům tak jen stačí zadat cíl útoku, zaplatit a o zbytek už se postará někdo jiný.

Tabulka 2.1: Ceník DDoS útoků uvedený na stránkách [2]

Cena	Doba trvání	Síla
9.99$	600 sec	20 Gbps
24.99$	2000 sec	20 Gbps
34.99$	3600 sec	20 Gbps
49.99$	7200 sec	20 Gbps

Ceny se pohybují od jednotek dolarů v závislosti na zvolených parametrech, jak uvádí tabulka 2.1. Mezi tyto parametry obvykle patří síla útoku, typ útoku, doba trvání a umístění botnetu. Některé weby dokonce nabízejí testovací útok zdarma. Na internetu lze tyto weby najít pod názvy jako *stress test*, *booter* nebo *ddoser*. Některé typy útoků, které lze provést jsou popsány v následujícím textu:

TCP SYN Flood útok označovaný zkráceně jako „synflood“ je metoda útoku zneužívající vlastnosti protokolu TCP. Pro komunikaci tímto protokolem je nejdříve nutné navázat spojení mezi komunikujícími stranami, jak bylo popsáno v předchozí kapitole 2.1. Synflood útok spočívá právě ve zneužití navázání spojení, kdy útočník odesílá na server velké množství paketů s příznakem SYN, ale již neodpovídá serveru pakety s příznakem ACK. Server je tak nucen udržovat si velké množství spojení v poloo-tevřeném režimu, dokud jim nevyprší časovač. Počet takto udržovaných spojení je



Obrázek 2.2: Porovnání navázání spojení legitimního uživatele a útočníka.

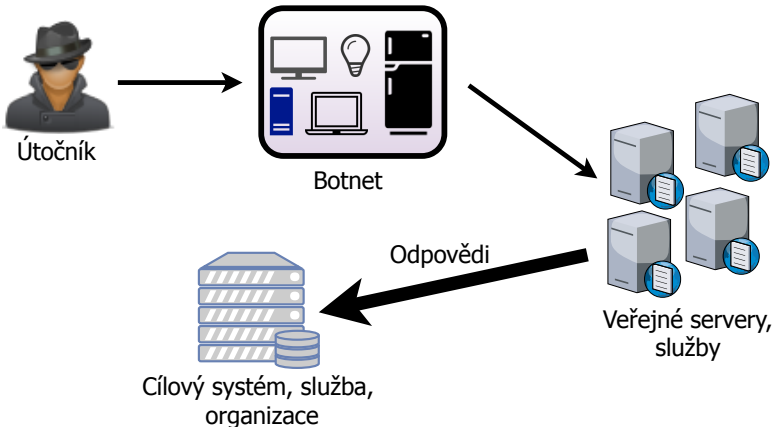
omezen a po jeho překročení začne server nově příchozí požadavky o spojení odmítat. Porovnání navázání spojení z pohledu legitimního uživatele a útočníka je zobrazeno na obrázku 2.2.

UDP flood útok využívá bezstavového transportního protokolu UDP. Útok typu UDP flood je jedním z nejjednodušších typů útoků. Cílem tohoto útoku je přetížení serveru nebo linky, vysíláním velkého množství zpráv na náhodné nebo konkrétní porty cílového systému. Systém se po přijetí zprávy snaží zjistit, zda na cílovém portu naslouchá nějaká aplikace. Pokud systém zjistí, že na požadovaném portu není spuštěna žádná aplikace, odpoví odesílateli ICMP zprávou `Destination Unreachable` vyjadřující nedostupnost. Snahou útočníka je tedy zahltit cílový systém UDP zprávami, které nemají přiřazenou žádnou aplikaci. Systém je tedy nucen na každou takovou zprávu odpovědět ICMP zprávou o nedostupnosti. Pokud je počet takovýchto UDP zpráv dostatečný, dojde k zahlcení systému nebo kapacity přenosového pásma. Velmi často jsou UDP zprávy generované s podvrženou zdrojovou IP adresou, aby útok nebylo možné snadno blokovat, nebo aby odpovědi od serveru nedostával zdroj útoku a nedošlo k jeho přetížení.

ICMP flood útok je jeden z nejstarších DoS útoků. Útok využívá protokolu ICMP, konkrétně zprávy typu *Echo*. Tento typ zprávy slouží ke zjištění, zda je vzdálené zařízení dostupné. Cílem útoku je zahlcení cílového systému či kapacity přenosového pásma. ICMP Echo funguje tak, že uživatel pošle zprávu typu *Echo request* na vzdálené cílové zařízení. Toto zařízení odpoví nazpět zprávou typu *Echo reply* o stejné velikosti, jako byla zpráva příchozí. Tímto je přenosové pásmo zatěžováno hned dvakrát. Poprvé zprávou typu Echo request a podruhé odpovědí typu Echo reply. U tohoto útoku je časté, že útočník podvrhne svou zdrojovou IP adresu, aby se vyhnul příjmu odpovědí typu Echo reply. Efektivita útoku roste s počtem zařízení zapojených do útoku.

Amplifikační, reflektované útoky typu DDoS mají za cíl zesílit svůj útočný potenciál před tím, než doputují k cílovému systému. Ukázka průběhu amplifikačního útoku je zobrazena na obrázku 2.3. K provedení útoku je obvykle využita síť botnet (horní část obrázku). Pro zesílení útoku útočník pomocí této sítě generuje dotazy obvykle na

veřejně dostupné služby, které disponují vysokou přenosovou kapacitou (pravá část obrázku). Tyto služby dokáží i na relativně malý dotaz odpovědět několikanásobně větší odpovědí. Tímto je schopen i útočník s linkou o malé kapacitě vyvolat velký útok. Poměru mezi velikostí odpovědi a dotazu se říká stupeň zesílení. K zesílení se obvykle zneužívají služby jako je systém doménovým jmen (Domain Name System, DNS), systém pro synchronizaci času (Network Time Protocol, NTP) nebo systém pro správu sítí (Simple Network Management Protocol, SNMP). Aby požadovaného efektu bylo dosaženo, musí útočník posílat takový typ dotazu, na který zneužitý systém dokáže odpovědět výrazně větší odpovědí. Dále je potřeba zajistit to, aby odpovědi nechodily zpět k útočníkovi, ale byly směrovány na cílový systém útoku (spodní část obrázku). Toho se dosáhne pomocí podvržení zdrojové IP adresy za IP adresu systému, na který má být útok směrován. Útoky, které využívají tento princip se na nazývají jako útoky s odrazem, případně reflektované útoky.



Obrázek 2.3: Ukázka průběhu amplifikačního útoku.

Amplifikační útoky lze potom dělit podle protokolu použitého k zesílení následovně:

- **DNS amplifikační útok** zneužívá vlastnosti systému doménových jmen DNS. Útočník pošle na DNS server dotaz (ANY) na kompletní záznamy k určitému doménovému jménu. DNS server vygeneruje odpověď, která může být až několikanásobně větší než dotaz a odešle ji na zařízení oběti. Útočník obvykle volí dotaz na takovou doménu, která obsahuje co nejvíce DNS záznamů. Pro příklad lze uvést dotaz o velikosti 68 bytů, který vygeneroval odpověď velkou 419 bytů, což je více než šestkrát větší zesílení [4]. Průměrný stupeň zesílení amplifikačního útoku systémem DNS se pohybuje v poměru 70:1 [3].
- **NTP amplifikační útok** využívá systému pro synchronizaci času NTP (Network Time Protocol). Útočník ve svém dotazu posílá příkaz `monlist`, který slouží k monitorování provozu na serveru. Tímto příkazem lze získat seznam posledních až šesti set adres, které se k danému serveru připojili. Poměr mezi velikostí odpovědi a dotazu se u tohoto typu útoku pohybuje v rozmezí 20:1 až 200:1 [3]. Útočník svůj útok tedy může zesílit až dvěstěkrát, což obvykle znamená, že systém oběti je zahlcený množstvím obrovských zpráv od systému NTP do takové míry, že jsou vyčerpány všechny dostupné zdroje a systém se stává nedostupným. K útoku lze využít pouze linuxové NTP servery, jejichž verze je

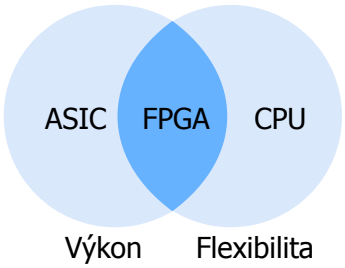
nižší než 4.2.8 a mají povolený příkaz `monlist` v konfiguraci. Od vyšších verzí byl příkaz `monlist` zcela odstraněn.

- **SNMP amplifikační útok** využívá systému SNMP (Simple Network Management Protocol), který se používá ke sběru statistik, testování stavu zařízení a případnému managementu síťových zařízení a serverů. Na jedné straně komunikace je monitorovací zařízení, které se nazývá SNMP manager, na druhé straně jsou monitorovaná zařízení, kterým se říká SNMP agenti. Princip útoku je založen na tom, že útočník posílá SNMP agentům dotazy typu SNMP GET s podvrženou zdrojovou IP adresou. Tento typ útoku je možné provádět pouze u SNMP verze jedna a dva. Od verze tři již tento útok není možné provádět, protože komunikace vyžaduje ověření prostřednictvím jména a hesla a zároveň se používá šifrované spojení.

2.3 Zařízení pro ochranu před DoS útoky

Zařízení pro ochranu před DoS útoky je vyvíjeno v rámci vědecko-výzkumné skupiny Liberouter, která spadá pod sdružení CESNET. Sdružení CESNET sdružuje vysoké školy a Akademii věd České republiky. Provozuje a rozvíjí také národní e-infrastrukturu pro vědu, výzkum a vzdělávání zahrnující počítačovou síť, výpočetní gridy, datová úložiště, prostředí pro spolupráci a nabízí širokou škálu služeb [1]. Zařízení je založeno na COMBO kartě s označením COMBO-100G, což je hardwarově akcelerovaná síťová karta (Network Interface Card, NIC) určená do PCI-Express slotu, která slouží ke zpracování síťových dat na vysokorychlostních linkách. Ke své akceleraci zařízení využívá technologii FPGA, která je blíže popsána dále v textu. Cílem této práce je vytvořit návrh a implementaci softwaru, který se bude starat o řízení tohoto hardwarového zařízení. Z toho důvodu je následující část věnována popisu tohoto zařízení a jeho rozhraní, které k práci s ním poskytuje.

Text následujícího odstavce vychází z literatury [12], [6]. Technologie programovatelného hradlového pole (FPGA, Field Programmable Gate Array) nabízí střední cestu mezi výkonem aplikačně specifických integrovaných obvodů ASIC (Application Specific Integrated Circuits) a flexibilitou základní výpočetní jednotky dnešních počítačů CPU (Central Processing Unit). Porovnání technologií ASIC, FPGA a CPU je vyobrazeno na obrázku 2.4. Aplikačně specifické integrované obvody jsou specializovány na vykonávání jedné kon-

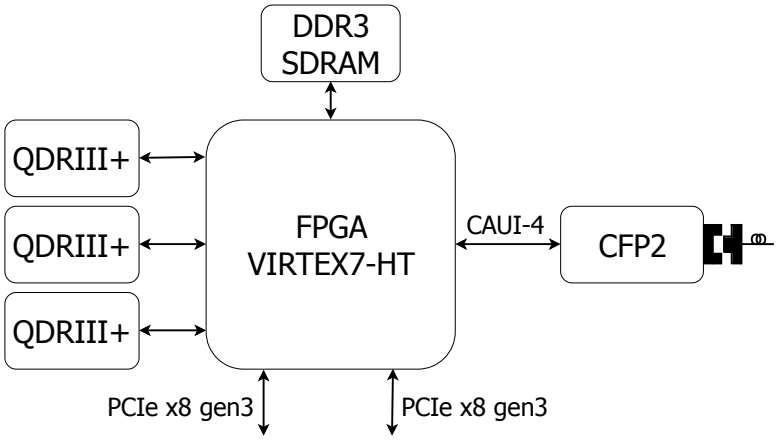


Obrázek 2.4: Porovnání technologií ASIC, FPGA a CPU

krétní aplikace. Návrh obvodu je optimalizovaný přímo pro potřeby aplikace, což poskytuje velkou rychlost a efektivitu při samotném vykonávání. Velkou nevýhodou je však nulová flexibilita, kdy po výrobě takového integrovaného obvodu již nelze provádět jakékoliv změny jeho struktury. Výroba ASIC obvodů je tak náchylná na chyby v návrhu, kdy se při zjištění

chyby musí vyrobit zcela nový integrovaný obvod. To může znamenat značné prodražení samotného vývoje. Vývoj aplikace pro tuto technologii je z důvodu zaměření na návrh dlouhý a počáteční náklady jsou velmi vysoké. Tato technologie se využívá především u náročných aplikací, vyžadující vysokou rychlost, malé rozměry a nízkou spotřebu. Obecný procesor (CPU) poskytuje oproti technologii ASIC vysokou flexibilitu, není omezen vykonáváním konkrétní aplikace, ale lze na něm vykonávat libovolnou aplikaci. Nevýhodou vysoké flexibility je relativně pomalá rychlost při vykonávání takovéto aplikace oproti hardwarovým řešením, které se specializují na konkrétní aplikaci. Programovatelná hradlová pole jsou konstrukční alternativou k aplikačně specifickým integrovaným obvodům. Mohou být naprogramovaná tak, aby umožnila realizaci libovolných uživatelsky definovaných sekvenčních a kombinačních obvodů. Největší výhodou oproti technologii ASIC je možnost rekonfigurace čipu, se kterou souvisí snazší a levnější vývoj aplikace. V případě opravy chyby, nebo rozšíření aplikace o nové vlastnosti není potřeba vyrábět nový hardwarový čip, ale stačí pouze čip přeprogramovat. Z tohoto důvodu je technologie FPGA používána ve velké míře pro vývoj prototypů a pro rychlé ověření funkčnosti návrhu vyvíjené aplikace. FPGA poskytuje oproti ASIC nižší výpočetní výkon, který se ale rychlým vývojem nových technologií v posledních letech pomalu srovnává s technologií ASIC. Nejrozšířenějším programovacím jazykem používaným k vývoji aplikací pro technologii FPGA je VHDL (VHSIC hardware Description Language) a System Verilog.

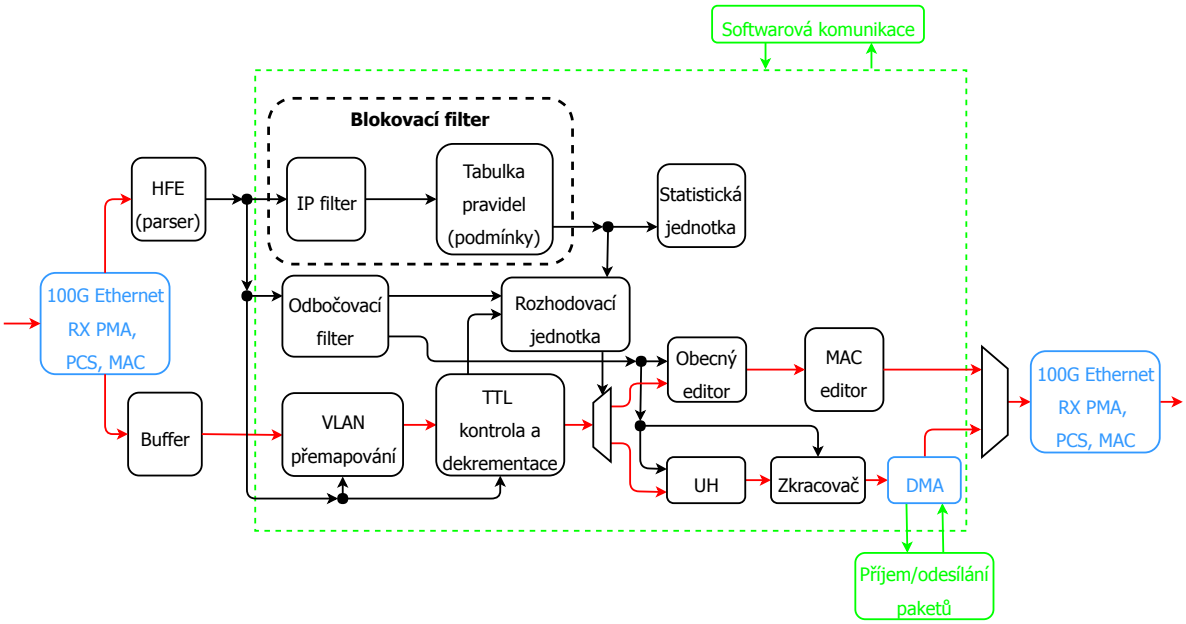
Akcelerovaná síťová karta COMBO-100G, jejíž základní blokové schéma je uvedeno na obrázku 2.5, se skládá z čipu FPGA Xilinx Virtex-7HT pro možnost pokročilého zpracování



Obrázek 2.5: Základní blokové schéma karty COMBO-100G. [9]

síťové komunikace. Dalšími komponentami jsou optické síťové rozhraní CFP2, rozhraní PCI-Express Gen3, tři moduly statických pamětí QDRIIIe a skupina modulů dynamických pamětí DDR3 SDRAM [9]. Čip FPGA Virtex-7HT poskytuje dostatečnou rychlost a výkon pro pokročilé zpracování síťové komunikaci při rychlosti 100 Gbps.

Následující část vychází z práce [12], která se popisovanou problematikou detailně zabývá. Za účelem dosažení požadované datové propustnosti využívá firmware karty principu zřetězeného zpracování (deep pipelined architecture). Firmware je rozdělen na několik samostatných bloků (modulů), jejichž schéma je zobrazeno na obrázku 2.6. Struktura firmwaru je rozdělena do tří základních cest. Datová cesta (označená červenou barvou), řídící cesta (označená černou barvou) a cesta určená pro konfiguraci (označená zelenou barvou) prostřednictvím softwarové rozhraní, které je popsáno dále v textu. Úlohou datové cesty je



Obrázek 2.6: Schéma architektury firmwaru.

přenos dat síťové komunikace (paketů) mezi jednotlivými moduly (komponentami) pracující s těmito daty. Takové moduly lze rozdělit do dvou skupin. První skupina využívá přijatá data (pakety) pro získání požadovaných informací v nich obsažených, ale nijak data nemodifikuje. Do této skupiny patří komponenta HFE (Header Field Extractor), která získává požadované informace z paketů. Další komponentou je vyrovnávací paměť (Buffer), sloužící k uchovávání paketů určených k dalšímu zpracování (levá část obrázku). Druhou skupinu tvoří komponenty, které na základě aktuálních instrukcí přijatých z řídící cesty provádí modifikaci a úpravu dat (paketů) v pořadí, jakém jsou zapojeny na obrázku (prostřední spodní část obrázku). Patří sem komponenta určená pro změnu položek VLAN, komponenta kontroly doby životnosti paketu (TTL, HOP LIMIT) a její dekrementace o jedničku. Následuje jednotka, která rozhoduje o dalším pokračování cesty. Na základě rozhodnutí jsou data dále zpracovávána buď v obecném editoru a editoru MAC adres, nebo v generátoru UH hlaviček a zkracovači paketů.

Úlohou řídící cesty je na základě informací získaných z paketů a nastavení řídících a stavových registrů prostřednictvím softwarové konfigurace určit akci, která má být s přijatými daty vykonána. Po určení akce jsou vygenerovány příkazy, které jsou rozeslány jednotlivým modulům, vykonávající změny v přijatých datech. Řídící cesta je tvořena následujícími moduly: HFE, odbočovací filter, IP filter, tabulka specifických podmínek, statistická jednotka a rozhodovací modul.

Konfigurační cesta je určená pro řízení, nastavení činností a chování jednotlivých modulů prostřednictvím softwarového rozhraní, které je nazváno `DCPROLIB`. Jedná se o knihovnu napsanou v jazyce C, která poskytuje funkce pro konfiguraci a komunikaci s jednotlivými moduly firmwarového jádra karty. Moduly, důležité s ohledem na praktickou část práce, jsou blíže specifikovány v následujícím textu:

Parser HFE (Header Field Extractor) získává potřebné informace o příchozím provozu. Tyto informace vyčítá z hlaviček jednotlivých protokolů, ve kterých jsou zapouzdřená přenášená data. Získané informace jsou následně pomocí rozhraní kompo-

nenty poskytovány dalším modulům ke zpracování. Těmito informacemi jsou například MAC adresy, IP adresy, číslo protokolu, fragmentace a další položky obsažené v hlavičkách paketů. Modul si také udržuje informaci o pozici vybraných hlaviček. Využívá k tomu hodnotu offsetu, která určuje vzdálenost od začátku dat.

Modul pro výměnu VLAN realizuje výměnu staré položky nacházející se ve VLAN hlavičce za položku novou podle zvolené konfigurace. Pozice hlavičky je získána z hodnoty offsetu, který poskytuje komponenta HFE.

Modul pro kontrolu TTL slouží k dekrementaci doby životnosti paketu, která je součástí IP hlavičky. Pozice hlavičky je jako u předchozího modulu získána pomocí offsetu z HFE komponenty. Modul také provádí kontrolu, zda nevypršela životnost paketu a nemá dojít k jeho zahození.

MAC editor uskutečňuje náhradu zdrojové a cílové MAC adresy za novou, případně provádí jejich vzájemnou výměnu. Hodnoty původních MAC adres jsou získány z komponenty HFE. U ethernetové hlavičky není nutné uchovávat její pozici, protože MAC adresy se vždy nacházejí na začátku dat.

Odbočovací filter slouží především k přesměrování specifické části síťového provozu do operační paměti počítače. Filter dokáže podle své konfigurace do operační paměti předávat buď celé pakety, nebo pouze takzvané UH hlavičky, které jsou popsány níže. Také dokáže zahazovat specifickou část síťového provozu.

Generátor UH v případě přeposílání síťového provozu do operační paměti počítače formou UH hlaviček tyto hlavičky vytváří. UH hlavičky tvoří několik položek s informacemi o paketu, jako jsou IP adresy, čísla portů, délka a další. Potřebné informace jsou získávány z komponenty HFE. Podrobný formát UH hlavičky je popsán v příloze A.

Blokovací filter se skládá ze dvou podjednotek a slouží k uložení IP adres a jim přidružených pravidel, která specifikují nežádoucí síťový provoz. První jednotka v pořadí je IP filter, která porovnává zdrojovou IP adresu získanou z HFE modulu s IP adresami uloženými v IP filtru. Pokud se nalezne shoda přistoupí se do druhé jednotky, kde se daný paket porovnává s přidruženým pravidlem IP adresy. Porovnávány mohou být například rozsahy portů, rozsahy délky paketu, čísla protokolů a další. O výsledku porovnání jsou informovány další moduly, jako je rozhodovací jednotka a statistická jednotka. Paket je v případě shody zahozen. Obě jednotky je možno dynamicky konfigurovat prostřednictvím softwarového rozhraní, bez nutnosti zastavení zpracování síťového přenosu.

Statistická jednotka má za úkol počítat počet paketů a bytů blokovaného síťového provozu. Hodnoty jsou počítány pro každou zdrojovou IP adresu v IP filtru zvlášť. Přičítaná velikost paketu je získána z komponenty HFE. Hodnoty čítačů lze vyčíst skrz softwarové rozhraní. Po jejich vyčtení jsou čítače vynulovány.

Kapitola 3

Konceptuální návrh systému

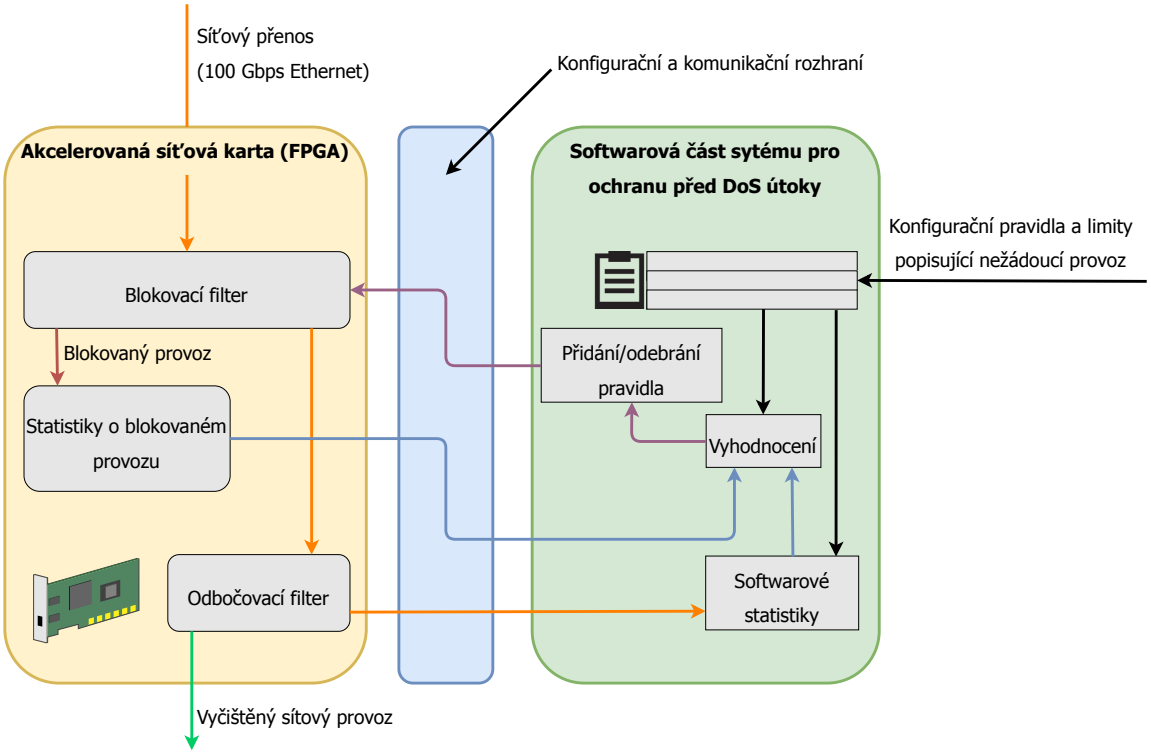
Kapitola se věnuje motivaci pro vznik systému. Druhá část je věnována konceptu systému pro řešení dané problematiky. Poslední část popisuje způsob zapojení systému do infrastruktury sítě.

3.1 Motivace a řešená problematika

Motivací ke vzniku zařízení byla skutečnost, že sdružení CESNET také pociťuje stoupající množství DDoS útoků ve své síti, kde v současné době nedisponuje žádným automatizovaným řešením tohoto problému. Proto vznikla myšlenka systému, který by byl schopný chránit koncové sítě a zařízení před zahlcením linek a vyčerpáním výpočetních zdrojů. Amplifikační, reflektované útoky typu DDoS často nemají za cíl pouze koncová zařízení, ale cílí přímo na samotnou infrastrukturu sítě oběti, kde vyčerpají dostupné zdroje dané sítě. Koncová síť pak není schopna se sama bránit, protože její přenosové linky jsou zahlcené. Proto je potřeba tento problém řešit už přímo na úrovni páteřní infrastruktury sítě, kde jsou linky s dostatečnou kapacitou, které nejsou útokem zahlceny. Ukazatelem takovýchto útoků obvykle bývá enormní počet paketů o velké délce proudící z typicky zranitelných veřejných služeb typu DNS, NTP atd. Běžná síťová zařízení, jako jsou routery, neposkytují dostatečné možnosti pro efektivní ochranu sítě. Obvykle disponují možností omezování síťového provozu (Rate limiting), nebo úplného zahazování specifického síťového provozu pomocí služby RTBH (Remotely Triggered Black Hole). Obě řešení jsou však velmi hrubá, a to hlavně z toho důvodu, že neumožňují dostatečně rozlišit provoz legitimního uživatele a útočníka. Může tak dojít k zahazování provozu legitimních uživatelů, což útočníkův záměr, odepření služby, ještě více podpoří. I proto vznikla snaha navrhnout takové řešení, které tento nedostatek dostatečně eliminuje a Rate limiting a RTBH budou sloužit pouze jako záloha, než jako primární zdroj ochrany sítě před DoS útoky.

3.2 Koncept systému

Konceptuální návrh systému je znázorněn na obrázku 3.1. Systém se bude skládat ze serveru, ve kterém bude připojena hardwarově akcelerovaná síťová COMBO karta (žlutá část obrázku). Z důvodu dosažení vysokých nároků na rychlost zpracování a datovou propustnost obstará většinu práce FPGA čip v kartě. Softwarová část systému (zelená část obrázku) bude skrz konfigurační rozhraní (modrá část obrázku) řídit firmware síťové karty a počítat statistiky příchozího síťového provozu k jednotlivým pravidlům. Ta budou popisovat nežá-



Obrázek 3.1: Konceptuální návrh systému pro ochranu před DoS útoky.

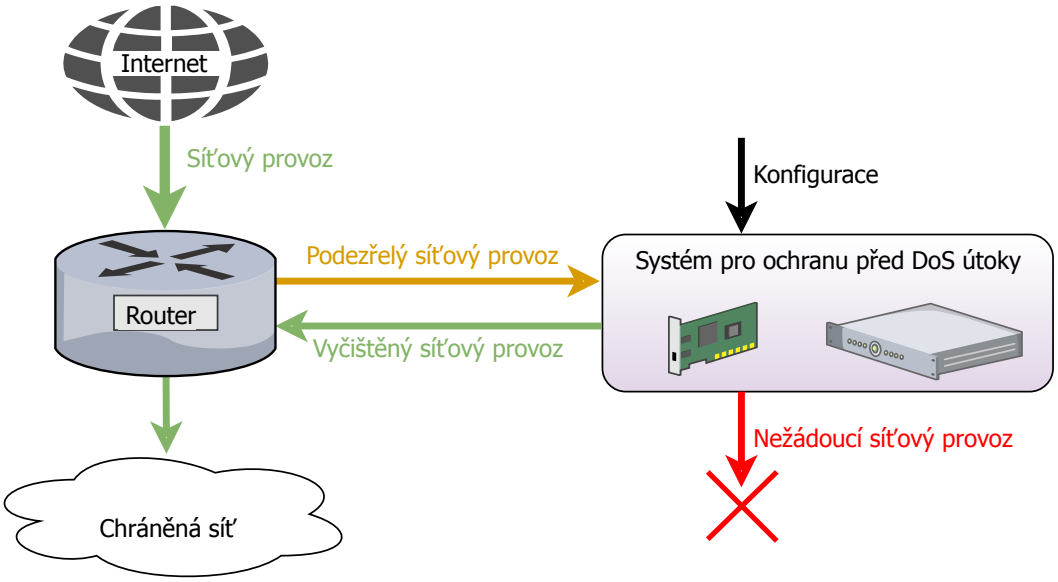
doucí síťový provoz a jeho hraniční limity. Získané hodnoty budou pravidelně předávány k vyhodnocení, kde se spojí s obdobnými statistikami počítanými ve firmwaru karty. Následně se provádí kontrola, zda nedošlo k překročení povolených limitů. V případě překročení hraničních hodnot se provedou úkony vedoucí k zablokování zdrojů nežádoucího provozu. Tyto zdroje budou předány do blokovacího filtru firmwaru karty, který bude porovnávat každý příchozí paket do systému s daty v něm uloženými. Provoz, který bude na základě nalezení shody filtrem zahozen bude započítán do statistik ve firmwaru karty, o kterých již byla zmínka výše.

Koncept předpokládá, že se na síťové rozhraní, pokud to bude možné, budou předávat pouze UH hlavičky, viz podkapitola 2.3, místo celých síťových dat. Tímto způsobem lze výrazně snížit objem dat přenášených do softwaru, protože UH hlavičky mají vždy velikost 48 bytů. Odpadne také nutnost získávání informací z paketů a jejich modifikace, protože tato činnost je při předávání UH hlaviček realizována hardwarovým akcelerátorem. Touto funkcionalitou se tak výrazně sníží výpočetní náročnost softwaru pro dosažení propustnosti až 100 Gbps. Tento princip ale nebude možné aplikovat u veškerého typu provozu, protože karta původní paket rovnou předává na své výstupní rozhraní bez možnosti ovlivnit softwarem jeho případné zahození, což nemusí být vždy vhodné.

Předpokládá se, že v budoucnu by softwarová část zařízení mohla být konfigurovatelná tak, aby softwarová část systému byla zcela nezávislá na firmwaru karty. Systém by tak mohl fungovat nad téměř libovolnou síťovou kartou. Z tohoto důvodu by měl software podporovat samotnou funkcionalitu firmwaru karty, aby tato nezávislost byla možná. Dalším důvodem, proč by měl software podporovat tuto funkcionalitu je předpoklad, že budou přibývat nové detekční moduly, které budou potřebovat zpracovávat, nebo vytvářet určitá síťová data zcela v softwarové části i při použití karty s hardwarovou akcelerací.

3.3 Zapojení systému v síti

Princip zapojení systému v síťové infrastruktuře je zobrazen na obrázku 3.2 a spočívá v tom, že všechen potencionálně škodlivý příchozí provoz z vnějších sítí bude směrován na centrální router, který jej odkloní do systému. Systém následně provede úkony k vyčištění provozu



Obrázek 3.2: Schéma zapojení systému pro ochranu před DoS útoky v síti.

a vrátí jej zpět na router, který vyčištěný provoz přepošle dál do sítě. Přesměrování provozu do systému a následně dále do síti bude realizováno na routeru technologií VRF (Virtual routing and forwarding), prostřednictvím které je možné realizovat směrování na základě hodnoty VLAN (virtuální LAN). Aby bylo možné od sebe vyčištěný a nevyčištěný provoz odlišit, bude nevyčištěný provoz přicházet ve VLAN X. Router tak pozná, že jej má odklonit do systému. Po vyčištění se změní hodnota VLAN z X→Y, což bude pro router informace, že se jedná o vyčištěný provoz. Systém se bude v síti jevit jako legitimní síťové zařízení, tzn. že, bude dekrementovat hodnotu TTL a HOP LIMIT, zahazovat pakety s TTL=1 a HOP LIMIT=1, odpovídat na dotazy typu ARP a NDP, mapovat již zmíněnou hodnotu VLAN na jinou a jako poslední bude přepisovat MAC adresy.

V současném návrhu řešení bude systém schopen přepisovat cílovou MAC adresu na pouze jednu pevně danou adresu. To je však omezující na možnosti způsobu zapojení systému v síti založené na infrastruktuře L2 switchů, kde by bylo nutné mít za systémem připojený router, který se bude starat o směrování dat dále do sítě. Upravený návrh řešení, který je popsán níže, řeší problémy se zapojením systému v L2 infrastruktuře, ale je nad rámec praktické části práce. Do budoucna je však plánována jeho implementace do systému. V upraveném návrhu bude směrování provádět přímo systém, který si bude uchovávat svou vlastní aktualizovanou směrovací tabulku, která bude mapovat adresy podsítí na MAC adresy. Ke každému odchozímu paketu se v tabulce podle cílové IP adresy vyhledá přidružená MAC adresa, která se následně vloží jako cílová MAC adresa paketu. Odpadne tak potenciální finanční náročnost, kde by kromě pořízení samotného zařízení byl potřebný i router.

Popsaný konceptuální návrh sytému byl vytvořený v rámci spolupráce se sdružením CESNET. Obsah a zaměření této práce je dále věnováno návrhu a implementaci softwarové části systému (zelená část obrázku 3.1).

Kapitola 4

Návrh softwaru

Kapitola popisuje podrobný návrh softwarové části systému. První část je věnována popisu požadavků a vlastností vycházející z konceptuálního návrhu systému na ochranu před DoS útoky. Druhá část je zaměřena na návrh architektury a detailnímu popisu jednotlivých částí softwarové části systému.

4.1 Požadavky na software

Na základě konceptuálního návrhu systému vyplývají na softwarové části systému základní požadavky a vlastnosti, potřebné pro správnou a bezproblémovou funkcionalitu celého systému jako celku.

Vysoká datová propustnost je vzhledem k místu a způsobu zapojení systému do sítě velice důležitá. Návrh systému počítá s nasazením v nejvyšší úrovni síťové infrastruktury, prostřednictvím které se vzájemně propojují jednotlivé podsítě a připojují se do globální sítě internet. Sítě na nejvyšší úrovni bývají typicky dimenzovány na vysoké přenosové kapacity, aby byl zajištěn bezproblémový síťový provoz připojených podsítí, jež bývají typicky dimenzovány na nižší přenosové kapacity. Infrastruktura sítě CESNET, pro kterou je návrh systému primárně zaměřen, pracuje na rychlosti 100 Gbps. Jednotlivé podsítě pak na rychlostech 1 až 10 Gbps. Z tohoto důvodu je důležité, aby systém dokázal pracovat na přenosové rychlosti 100 Gbps. Předpokládá se ale, že softwarová část nebude schopna při příjmu celých paketů a provozu tvořeném krátkými pakety dosahovat plné propustnosti 100 Gbps. Tento problém by měl být řešitelný samotným návrhem, který předpokládá, že se do softwarové části budou posílat pouze UH hlavičky.

Modifikace síťového provozu je nutná za účelem poskytnutí jednoduššího způsobu zapojení systému do síťové infrastruktury a pro úkony, které je třeba provádět, aby se systém jevil jako legitimní síťové zařízení. Tyto požadavky vyšly z komunikace vedené se síťovými administrátory, kteří se starají o správu síťové infrastruktury. Systém by měl poskytovat možnost modifikace síťových paketů, na základě zvolené konfigurace. Modifikována by měla být VLAN hlavička, sloužící k následnému směrování dat. Dále zdrojová a cílová adresa síťového rozhraní (MAC) a položka značící dobu platnosti paketu (TTL, HOP LIMIT).

Detekce začátku a konce útoků je nutná pro správné fungování systému. Je potřebné, aby si systém k jednotlivých pravidlům popisující nežádoucí provoz uchovával aktu-

ální statistiky, jako jsou počet zpracovaných paketů a bytů. Tyto statistiky budou následně periodicky vyčítány a porovnávány s nastavenými hraničními hodnotami v pravidlech, které budou určovat od jakého množství je provoz považován za nežádoucí. Při překročení hranice (začátek útoku) začne detekční modul provádět úkony vedoucí k potlačení nežádoucího provozu. Naopak při snížení množství provozu pod hraniční mez (konec útoku), musí detekční modul provést kroky, které zaručí, že daný síťový provoz nebude dále žádným způsobem regulován.

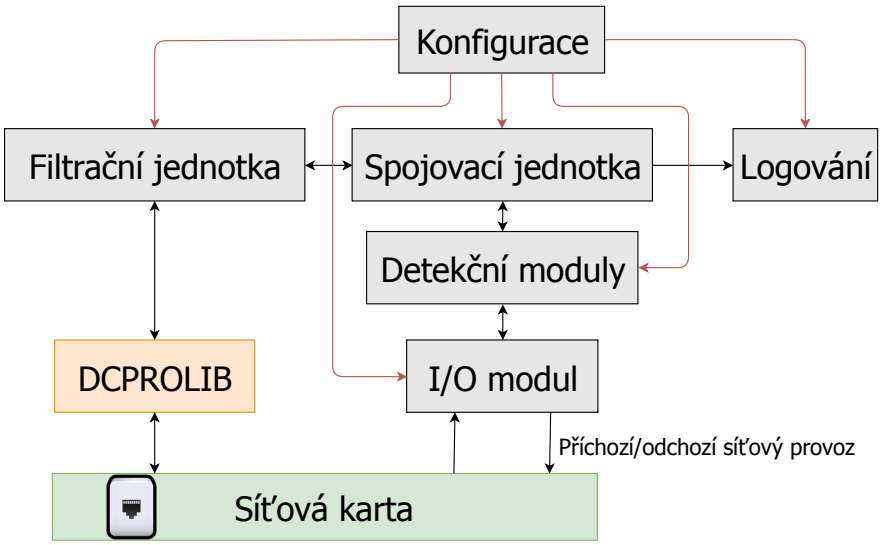
Správné blokování zdrojů nežádoucího provozu je důležité s ohledem na celkový koncept systému. Primárním cílem systému je sice ochrana koncové sítě nebo zařízení, ale je důležité, aby při potlačení nežádoucího provozu, pokud je to možné, nebyli limitování legitimní uživatelé, kteří mohou tvořit určitou část provozu. Pokud by se tak nedělo, tak by se samotný systém mohl stát zdrojem odepření služby. Systém by proto měl využívat takové algoritmy, které budou kompromisem mezi výpočetní složitostí algoritmu a snahou o rozlišení potencionálně legitimního uživatele od útočníka.

Informování o stavu systému je nezbytnou nutností vyplývající z komunikace se správci sítě, kteří vyžadují mít přehled o zařízeních připojených do síťové infrastruktury. Systém by měl proto provádět logování o událostech a o aktuálním stavu, jako je například aktuální zatížení systému, nebo počet aktivních DoS útoků.

Modularita systému je důležitá pro možné budoucí rozšiřování funkcionality systému. Návrh by měl být zaměřen na rozdělení systému do logických celků tak, aby byl kompromisem mezi flexibilitou a výkonností celé softwarové části.

4.2 Architektura softwaru

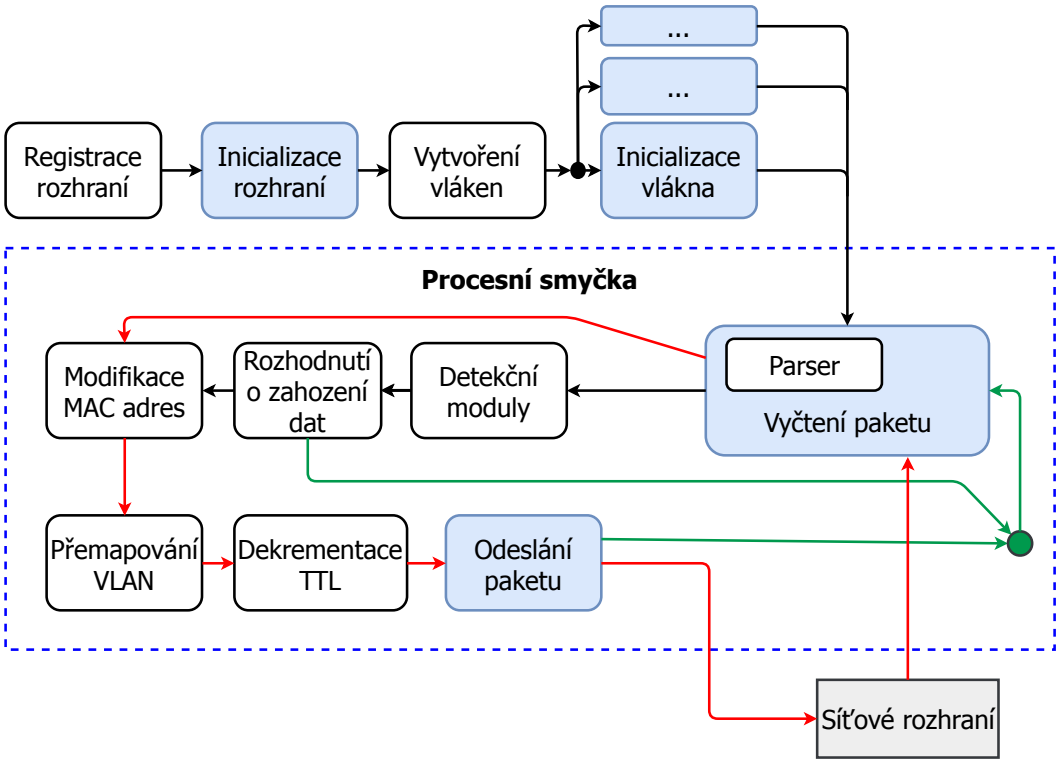
Architektura softwaru je znázorněná na obrázku 4.1. Návrh počítá s rozdělením na šest základních částí (šedé obdelníky v obrázku), které jsou blíže popsány dále v textu.



Obrázek 4.1: Schéma architektury softwaru

Konfigurace bude sloužit k nastavení chování a změně vlastností systému. Načtení konfigurace bude možné realizovat dvěma způsoby, příkazovou řádkou a strukturovaným konfiguračním souborem. Jednotlivé způsoby budou mít svou prioritu, aby bylo možné načítat konfiguraci z obou zdrojů současně. Priorita bude určovat, z jakého zdroje se hodnota konfigurační položky načte při zadání stejné položky v obou zdrojích. Vyšší priorita znamená načtení hodnoty z daného zdroje. Každá povinná položka bude mít v systému nastavenou svou výchozí hodnotu, která se jí přiřadí v případě, že tato položka nebude explicitně nastavena. Po načtení se provede u jednotlivých položek kontrola, zda zadaná hodnota odpovídá požadovanému formátu. Příkazová řádka bude mít nejvyšší prioritu. Všechny konfigurační položky zadávané skrz příkazovou řádku půjdou pro lepší přehlednost zadat v takzvaném dlouhém tvaru (long option). Položky ve strukturovaném konfiguračním souboru budou členěny do logických skupin a podskupin, aby byla zajištěna jejich snadná čitelnost.

I/O modul bude určený k příjmu, modifikaci a odesílání síťových dat. Schéma modulu je zobrazeno na obrázku 4.2. Návrh počítá s možností implementace více síťových



Obrázek 4.2: Schéma I/O modulu.

rozhraní. Proto bude modul poskytovat předpis obecných funkcí (vyplněné modré obdélníky v obrázku), které budou mít jasně definované vstupy a výstupy. Pro přidání možnosti práce s novým síťovým rozhraním, bude stačit tyto obecné funkce implementovat. Jako primární síťové rozhraní bude použito SZE, které je určeno pro práci s akcelerační COMBO kartou. Dalšími rozhraními by v budoucnu mohly být například PCAP, DPDK nebo NDP.

První krok modulu bude registrace rozhraní, jehož výběr a parametry pro práci s ním budou získané z konfiguračního modulu. Na základě výběru se přiřadí obecným funk-

cím implementace zvoleného rozhraní. Dalším krokem bude inicializace rozhraní, kde se budou provádět specifické úkony v závislosti na zvoleném rozhraní. Při práci s primárním SZE rozhraním zde proběhne konfigurace firmwaru skrz knihovnu `DCPROLIB`, kde se například provede nastavení MAC adresy systému, aby byl přijímán jen jí odpovídající provoz. Modul bude podporovat možnost vícevláknového zpracování síťových dat, a to v případě, že to zvolené rozhraní podporuje.

Další část, nazvaná procesní smyčka (modrý pruhovaný obdélník v obrázku), bude zajišťovat veškerý průchod a úpravu síťových dat v softwaru. Tuto část lze rozdělit do dvou základních cest: datová (červené šipky), ve které se pracuje s daty celých paketů a procesní (černé šipky), ve které se pracuje pouze s informacemi o paketu. Cesta paketu v softwaru započne jeho vyčtením ze síťového rozhraní, pomocí obecné funkce. Výstupem této funkce bude struktura, obsahující veškeré potřebné informace o paketu, jako jsou například: UH hlavička, typ přijatého paketu, pozice jednotlivých hlaviček a další. UH hlavička bude základním zdrojem informací o paketu v celém softwaru. Proto v případě přijetí celých síťových dat bude potřeba získat z paketu informace, které jsou v UH hlavičce obsaženy a následně je do ní uložit. O získání informací z hlaviček, ve kterých je paket zabalen, včetně pozic jednotlivých hlaviček, se bude starat submodul nazvaný Parser. Ten bude fungovat na stejném principu, jako již dříve popsaná komponenta HFE 2.3, implementována ve firmwaru karty. Parser bude navíc provádět kontrolu hodnoty doby životnosti paketu. Pakety s hodnotou jedna budou automaticky zahazovány. Následně se bude pokračovat procesní cestou, kde se získané informace o paketu předají připojeným detekčním modulům. Na základě informací o typu paketu, popř. informací z detekčních modulů, se rozhodne o tom, zda má být paket zahozen (rozhodnutí o zahození v obrázku), nebo předán dál pro další zpracování. Propuštěn dál bude moct být pouze paket, který nepřišel ve formě UH hlavičky. V případě zahození dat se procesní smyčka vrátí zpět na začátek (zelené šipky v obrázku) k vyčtení dalšího paketu.

Firmware karty v případě předávání celých paketů do softwaru neprovádí přepis MAC adres, proto je potřebné tuto funkcionalitu provádět softwarově. Stejně tak jako mapování VLAN a dekrementaci TTL v případě, že by software byl spuštěn s jinou síťovou kartou, která by tuto funkcionalitu nepodporovala. V případě, že se rozhodne o propuštění paketu, spojí se datová cesta s řídící a provede se následná modifikace dat, v závislosti na zvoleném rozhraní. V případě SZE tedy pouze přepis MAC adres. Návrh jednotlivých submodulů, které provádí úpravu paketu je popsán zde:

MAC editor vychází s návrhu stejnojmenné komponenty ve firmwaru karty. Bude se zde vykonávat nahrazení zdrojové a cílové MAC adresy za nové, podle zvolené konfigurace. Při vracení provozu zpět do sítě se provede modifikace rámců tak, že zdrojová MAC adresa bude nahrazena MAC adresou systému a cílová MAC adresa se nahradí MAC adresou zařízení, na které bude provoz směrován.

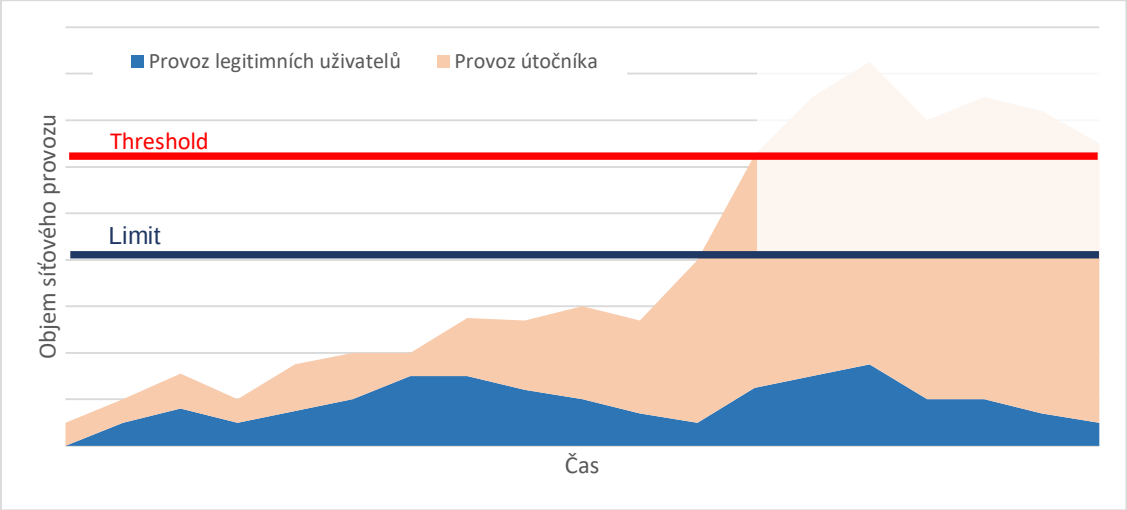
Přemapování VLAN vychází z návrhu implementovaném ve firmwaru karty. Modul bude provádět modifikaci VLAN hlavičky, konkrétně výměnu staré hodnoty VLAN ID (VLAN Identifier, VID) za hodnotu novou, podle zvolené konfigurace. Ta bude načítána ze souboru, jehož ukázka je v příloze C. Pokud paket bude obsahovat více VLAN hlaviček, bude se provádět úprava vnější hlavičky.

Dekrementace TTL bude snižovat hodnotu doby životnosti paketu TTL (IPv4) a HOP LIMIT (IPv6) o jedničku. U paketu IP verze 4 bude nutné při snížení TTL

provést i aktualizaci kontrolního součtu v IP hlavičce. Nová hodnota kontrolního součtu přitom nemusí být znovu vyčíslována na základě celé IP hlavičky, ale je možné provést jen inkrementální aktualizaci na základě znalostí předchozí hodnoty. Popis algoritmu pro inkrementální aktualizaci kontrolního součtu je popsán v RFC 1141 [13]. U paketu IP verze 6 není kontrolní součet v IP hlavičce obsažen, zde se pouze provede zmíněná dekrementace.

Posledním krokem v procesní smyčce bude samotné odeslání upraveného paketu dále do sítě. Odeslání se provede pomocí obecné funkce, po jejímž dokončení se přejde ke zpracování dalšího paketu (zelené šipky v obrázku).

Amplifikační modul spadá do skupiny detekčních modulů (zobrazené na obrázku 4.1), kde by měly být v budoucnu připojeny další detekční moduly, např. SYN Flood modul pro mitigaci synflood útoků (tento modul nebyl součástí této práce). Amplifikační modul bude sloužit k detekci amplifikačních DDoS útoků a k výběru zdrojových IP adres, které by měly být blokovány pro eliminaci útoku. Princip modulu je zobrazen na obrázku 4.3, kde oranžová část značí provoz útočníka a modrá část provoz legitimních uživatelů. Zobrazené hodnoty se vztahují k jednomu konkrétnímu pravidlu, kde se při



Obrázek 4.3: Princip amplifikačního modulu

překročení hranice threshold (v součtu obou provozů) detekuje začátek DDoS útoku. Následně se provede výběr zdrojových IP adres, které svou aktivitou nejvíce přispěly k překročení povolené hranice. Tyto IP adresy budou následně předány k zablokování, což bude mít za následek snížení objemu provozu pravidla na hodnotu limit. Koncové zařízení nebo podsíť tak bude před DDoS chráněna.

Aby bylo možné vykonávat úspěšnou detekci a výběr zdrojů nežádoucího síťového provozu, je potřebné modul vhodně nakonfigurovat. K tomuto účelu bude sloužit sada pravidel, která bude načítána skrz konfigurační soubor, jehož formát je zobrazen v příloze C. Pravidla budou specifikovat, jaký typ provozu a jaké jeho množství je považováno pro určitou podsíť za nežádoucí. Pravidla se budou skládat z několika položek, jejichž výčet a detailní popis je uveden zde:

```
<RULE> <DST-NET> <PROTOCOL> <SRC-PORT> <DST-PORT>
```

`<FRAGMENTATION> <TCPFLAGS> <LENGTH> <THRESHOLD> <LIMIT>`

`<RULE>` bude nepovinný parametr, který bude reprezentovat uživatelem definovaný název pravidla. Název pravidla bude textový řetězec o délce maximálně 100 znaků ve formátu `"slovo1 slovo2 slovo3"`.

`<DST-NET>` bude povinný parametr uvozený klíčovým slovem `dst net`, následovaný IP adresou s prefixem, která bude udávat chráněnou sít (zařízení). V jednom pravidlu bude možné zadat více prefixů, které budou oddělené mezerami (pro příklad `dst net 147.229.0.0/16 147.231.1.0/24`). Zápis více prefixů na jednom řádku bude ekvivalentní s rozepsáním pravidel na více řádků. Podporovány budou jak adresy IP verze 4, tak i IP verze 6. Zápis IP adresy bude muset být uváděn v jednoznačné formě (pro příklad 192.168.2.0/24 nebo 2001:db8:3c4d::/48).

`<PROTOCOL>` bude volitelný parametr reprezentovaný klíčovým slovem `protocol` a hodnotou protokolu. Hodnota značí číslo protokolu, jež bude nežádoucí provoz obsahovat v IP hlavičce. Hodnota může být vyjádřena číslem protokolu, nebo pro několik známých protokolů je možné uvést přímo jejich název `UDP`, `TCP`, `ICMP` a `SCTP`.

`<SRC-PORT>` bude volitelným parametrem, jež bude reprezentovat hodnotu zdrojového portu. Zápis bude uvozen klíčovým slovem `src port`, následovaný jedním z povolených vyjádření hodnoty. Povolený zápis bude buď přímo číslo zdrojového portu (exact match), operátor `>` (greater) nebo `<` (less) následovaný hodnotou, nebo rozsah (range) hodnot `menšíčíslo-vyššíčíslo`. Pro příklad je možné použít `src port 53`, `src port <1024` nebo `src port 1024-49151`. Číslo zdrojového portu je podporováno pouze pro protokoly `TCP=6`, `UDP=17` a `SCTP=132`, tzn. že v případě vynechání `<PROTOCOL>` je porovnáván zdrojový port pouze u těchto protokolů.

`<DST-PORT>` bude také volitelný parametr, jež bude uvozen klíčovým slovem `dst port` a bude značit číslo zdrojového portu u nežádoucího provozu. Zápis hodnoty za klíčovým slovem bude mít stejný formát a omezení jako port zdrojový.

`<FRAGMENTATION>` bude volitelným parametrem vyjadřujícím typ fragmentace nežádoucích dat. Parametr bude reprezentován klíčovým slovem `fragmentation` a hodnotou, která musí odpovídat jedné z následujících možností:

- `YES`, všechny fragmentované pakety se shodují včetně první a poslední části.
- `NO`, pouze nefragmentované pakety se shodují,
- `FIRST`, pouze fragmentované pakety obsahující první část paketu se shodují,
- `LAST`, pouze fragmentované pakety obsahující poslední část paketu se shodují,
- `MID`, pouze fragmentované pakety neobsahující první nebo poslední část paketu se shodují,
- `NOFIRST`, všechny fragmentované pakety vyjma první části paketu se shodují.

`<TCPFLAGS>` bude volitelný parametr značící příznaky (flagy) v TCP hlavičce. Zápis bude začínat klíčovým slovem `tcpflags`, následovaný hodnotou vyjádřenou jako řetězec složený z následujících písmen (`S`, `A`, `F`, `P`, `R`) nebo negace (`!`):

- `S`, pro SYN flag,
- `A`, pro ACK flag,
- `F`, pro FIN flag,
- `P`, pro PUSH flag
- `R`, pro RESET flag.

Negovaná hodnota každého flagu lze vyjádřit ! (vykřičníkem) před písmenem reprezentující flag. Pro příklad `S!A` se bude shodovat pouze s pakety obsahující SYN, ale neobsahující ACK. V případě zadání parametru `<TCPFLAGS>` bude nutné mít specifikovaný `protocol TCP`, jinak se bude jednat o neplatné pravidlo.

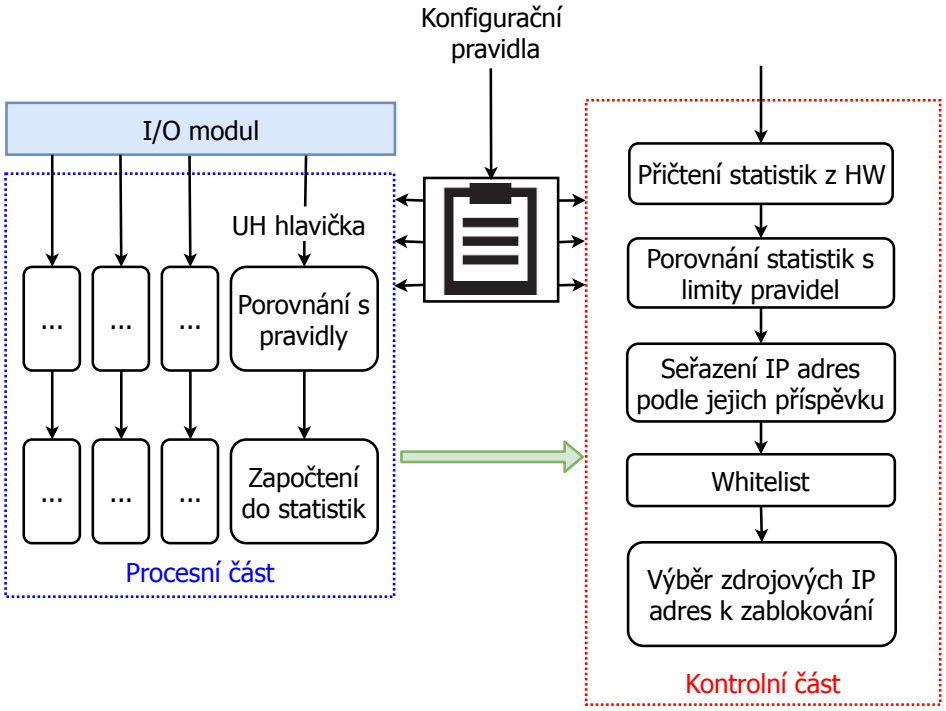
`<LENGTH>` bude volitelný parametr, který bude vyjadřovat délku paketu v bytech od MAC adres po cyklický redundantní součet (Cyclic redundancy check, CRC). Parametr bude reprezentován klíčovým slovem `length`, následovaný jedním z povolených zápisů hodnoty délky. Povolený zápis bude buď přímo velikost, operátor `>` nebo `<` následovaný hodnotou, nebo rozsah hodnot `menšíčíslo-vyššíčíslo`. Rozsah hodnot nemůže být použit spolu s operátory `<`, `>`. Pro příklad je možné použít `length 64`, `length >1526` nebo `length 64-1526`.

`<THRESHOLD>` bude povinný parametr začínající klíčovým slovem `threshold` následovaný hodnotou a jednotkou. Hodnota bude vyjadřovat, jaký datový průtok za vteřinu je považován za útok. Pokud bude hodnota thresholdu překročena, provoz bude zredukován na hodnotu `<LIMIT>`. Podporovány budou následující jednotky `pps`, `kpps`, `Mpps` pro paket rate a `Mbps`, `Gbps` pro bit rate. Převod jednotek je `k=1000`, `M=1000000` a `G=1000000000`. `<THRESHOLD>` a `<LIMIT>` budou muset mít nastavenou hodnotu buď pro pakety, bity nebo oboje zároveň.

`<LIMIT>` bude posledním povinným parametrem, který bude reprezentován klíčovým slovem `limit`, následovaný hodnotou a jednotkou. Hodnota bude určovat hranici, na kterou se má nežádoucí provoz omezit při překročení hranice `<THRESHOLD>`. Zápis hodnoty bude mít stejná pravidla jako u parametru `<THRESHOLD>`.

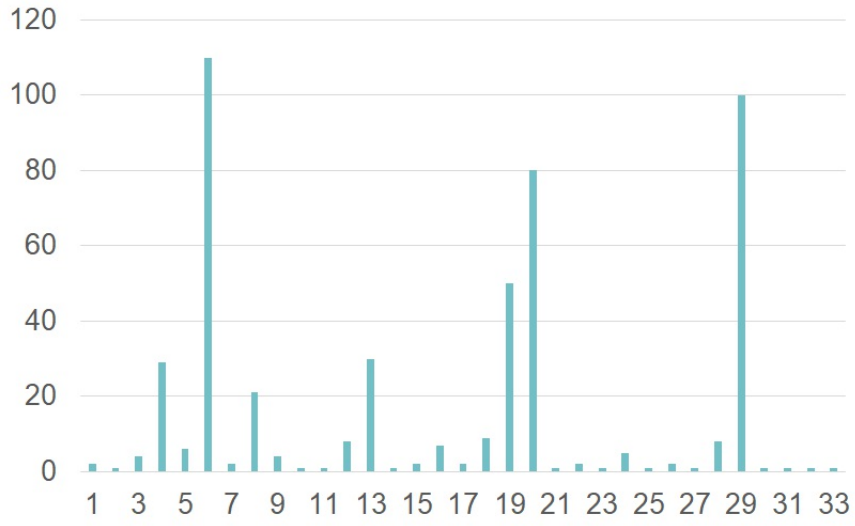
Schéma modulu je zobrazeno na obrázku 4.4. Modul se bude dělit na dvě hlavní části: procesní (levá strana obrázku) a kontrolní (pravá strana obrázku). Procesní část bude přijímat informace o aktuálním síťovém provozu od I/O modulu ve formě UH hlaviček. Informace získané z této hlavičky budou následně porovnány s jednotlivými pravidly, kdy v případě nalezení odpovídajícího pravidla budou aktualizovány statistiky pro dané pravidlo. Zároveň se pro každé pravidlo bude udržovat tabulka, která bude ukládat příspěvky viděných zdrojových IP adres, konkrétně počet viděných paketů a bytů. Celá procesní část bude podporovat možnost vícevláknového zpracování, v závislosti na konfiguraci.

Druhou částí modulu bude kontrolní část. V této části nejdříve proběhne sečtení statistik z jednotlivých procesních vláken, případně i statistik vedených ve firmwaru karty. Statistiky z firmwaru budou modulu předány skrz strukturu, kterou bude modul poskytovat k naplnění. O naplnění se postará spojovací část softwaru, tímto způsobem bude modul zcela nezávislý na použité síťové kartě a jejím firmwaru. Sečtené hodnoty budou následně porovnány s pravidly, kde se bude kontrolovat, zda nedošlo k překročení hranice `threshold`. V případě překročení bude potřeba zjistit, jaké IP adresy svou aktivitou nejvíce přispěly k překročení hranice `threshold`. Proto bude nutné spojit



Obrázek 4.4: Schéma amplifikačního modulu.

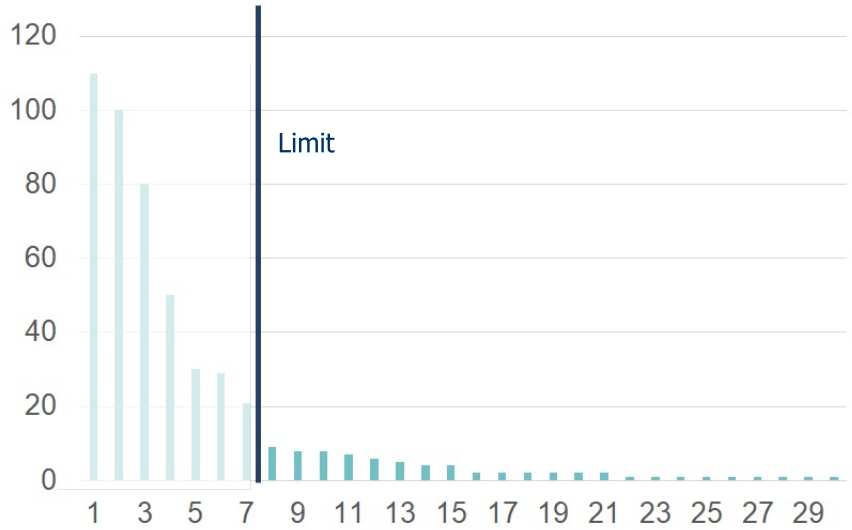
jednotlivé statistiky o příspěvcích zdrojových IP adres, vedených pro dané pravidlo, a to jak v samotném modulu, tak i z firmwaru karty. Po spojení bude tabulka obsahovat pouze unikátní IP adresy, duplikované adresy budou sloučeny. Ukázka možné výsledné tabulky je zobrazena na obrázku 4.5, kde osa X značí unikátní zdrojové adresy a osa Y pak jejich příspěvky. Poté co budou hodnoty spojeny, bude nutné provést



Obrázek 4.5: Unikátní zdrojové IP adresy po spojení procesních vláken a statistik z firmwaru

jejich seřazení. Řadit se bude sestupně od IP adres, které svou aktivitou přispěly nejvíce. Následně bude vybráno tolik zdrojových IP adres k zablokování, aby po odečtení

jejich příspěvku byl objem provozu snížen pod hranici `limit`. Princip je předveden na obrázku 4.6. Modul také bude podporovat vyloučení zvolených IP adres z blokování, tzv. whitelist, který bude načítán ze souboru. U každé IP adresy, která by měla být blokována, bude provedena kontrola, zda se nenachází na whitelistu. Pokud ano, daná IP adresa nebude vybrána k blokování. Do whitelistu bude možné zadávat nejen koncová zařízení, ale přímo celé podsítě. Ukázka konfiguračního souboru pro whitelist je zobrazena v příloze C. Vybrané IP adresy budou následně poskytnuty spojovací jednotce k dalšímu zpracování.



Obrázek 4.6: Seřazení zdrojových IP adres podle příspěvku a výběr adres k zablokování

Modul bude možné překonfigurovat za běhu. Nejprve se zkontroluje, zda nově nahraná konfigurace odpovídá požadovanému formátu, a následně se provede odstranění staré konfigurace a nahrání nové.

Spojovací jednotka bude zprostředkovávat komunikaci mezi amplifikačním modulem a filtrem, který je popsán níže v práci. Jednotka bude periodicky vyčítat informace o blokování z filtru. Těmito informacemi bude vyplňovat strukturu amplifikačnímu modulu, kterou tento modul poskytuje k získání informací o blokovaných IP adresách. Následně proběhne vyhodnocení amplifikačním modulem a poté spojovací jednotka vyčte IP adresy a jim přidružená pravidla, která tento modul požaduje zablokovat. Spojovací jednotka předá tyto informace filtru k zablokování.

Filter bude poskytovat rozšířenou funkcionalitu pro práci s IP filtrem ve firmwaru karty, jako bude možnost odebrání všech blokovaných IP adres nahrané konkrétním modulem, nebo odebrání všech blokovacích pravidel. IP filter ve firmwaru karty dokáže jednu IP adresu přiřadit pouze k jednomu blokovacímu pravidlu. V případě přiřazení jedné IP adresy k více blokovacím pravidlům, upraví filtr toto pravidlo tak, aby se pro danou zdrojovou IP adresu blokoval veškerý síťový provoz, nikoliv jen provoz odpovídající blokovacímu pravidlu. V případě, že dojde k odebrání blokovacího pravidla u IP adresy, která byla celkově blokována, filter automaticky nastaví specifické blokovací pravidlo, pokud toto pravidlo bude pro danou IP adresu už jako jediné.

Logování bude poskytovat informace o aktuálním stavu systému. Logování bude probíhat několika způsoby, v závislosti na zvolené úrovni poskytovaného detailu. Jednotlivé způsoby logování jsou popsány zde:

Syslog bude sloužit pro logování veškerých změn stavů systému (např. spuštění, zastavení, překročení tresholdu pravidla).

Logovací soubor zahrne veškeré zprávy logované také prostřednictvím Syslogu, a zároveň však bude logovat s vyšší úrovní detailu (např. výpis zablokovaných/odblokovaných IP adres při překročení tresholdu pravidla).

Stavový soubor bude poskytovat aktuální informace o stavu systému pomocí souboru, který se bude ve zvoleném intervalu atomicky přepisovat. Popis jednotlivých položek, které tento soubor bude obsahovat je zobrazen v příloze B.

Kapitola 5

Implementace

5.1 Implementace softwaru

Softwarová část systému, jejíž návrh byl popsán v předchozí kapitole 4, byla v rámci práce implementována v jazyce C. Prvním krokem byla implementace jednotlivých modulů softwarové části systému podle provedeného návrhu. Poté následovalo spojení jednotlivých části do jednoho celku. Významné části jsou popsány zde:

Obecné funkce v I/O modulu poskytují rozhraní pro budoucí rozšíření softwaru pro práci s novými rozhraními. Tyto funkce jsou implementovány jako ukazatele na funkce, jejich ukázka je v příloze D. Každé nové rozhraní bude muset povinně implementovat funkce `thread_init()`, `threads_count()`, `read_packet()` a `write_packet()`. Implementované funkce rozhraní se následně přiřadí obecným ukazatelům na funkce. Tyto funkce následně skryjí jakékoli použité rozhraní, a v programu se dále budou používat pouze tyto obecné funkce, bez nutnosti vědět, co se za nimi skrývá. Funkce `thread_init()` slouží k inicializaci daného vlákna a k přípravě přijmu a odesílání paketů. Funkce `threads_count()` vrátí počet vláken, které se mají vytvořit. Další povinná funkce `read_packet()` slouží k vyčtení paketu z daného rozhraní. Poslední povinnou funkcí je `write_packet()`, která umožňuje odeslání paketu. Další funkce, které jsou nepovinné, ale rozhraní je může implementovat jsou `init()`, `deinit()` a `thread_deinit()`. Funkce `init()` a `deinit()` slouží ke globální inicializaci a deinicializaci rozhraní, pro SZE rozhraní se zde provádí inicializace firmwaru karty. Poslední funkcí je `thread_deinit()`, která slouží k uvolnění paměti vzniklé při funkci `thread_init()`.

Práce se SZE rozhraním. Pro práci se SZE rozhraním se využívájí funkce, které poskytuje knihovna `libsze2.h`. Prvním krokem pro použití tohoto rozhraní je funkce `szedata_open()`, která umožňuje přístup k SZE driveru. Další funkcí je `szedata_subscribe()`, která slouží k výběru DMA kanálu, ze kterého chceme číst, popř. do kterého chceme zapisovat. Poslední inicializační funkcí je `szedata_start()`, která umožní čtení/zápis do zvoleného DMA kanálu. Pro čtení se využívá neblokující funkce `szedata_read_next_noblock()`, která vyčítá data vždy po jednom paketu. Pro odeslání paketu se používá funkce `szedata_burst_write_next()`, která umožňuje odeslání jednoho paketu, ten ale nemusí být odeslán okamžitě, protože se data určená k odeslání ukládají do bufferu, a k jejich odeslání dojde až je buffer zaplněný. To by mohlo způsobit zaseknutí paketu v kartě v případě, že by na vstupu nebyla žádná

nová data. Proto se při situaci, kdy nejsou na vstupu žádná data využívá funkce `szedata_burst_write_flush`, která tento buffer vyprázdní a data odešle.

Konfigurační soubor. Formát konfiguračního souboru byl jednou z věcí, které návrh přímo nespecifikoval. Návrh předpokládal použití takového formátu, který umožňuje strukturalizaci dat. Hlavními kandidáty byl formát YAML a XML. Pro lepší uživatelskou přehlednost byl nakonec zvolen formát YAML, jehož ukázka je zobrazena v příloze C.

5.2 Systémová služba

Z důvodu lepšího uživatelského ovládání byla vytvořena systémová služba, která nese název `ddosprotector`. Konfigurace služby se provádí pomocí konfiguračního souboru a služba podporuje následující operace:

start provede spuštění softwaru příkazem `service ddosprotector start`. Zároveň také provádí nahrání firmwaru do síťové karty.

stop ukončí běh softwaru příkazem `service ddosprotector stop`.

restart se provede příkazem `service ddosprotector restart`. Tento příkaze je ekvivalentní s provedením operací stop a následně start.

reload vynutí znovunačtení konfiguračních pravidel za běhu příkazem `service ddosprotector reload`.

status zjistí aktuální stav služby příkazem `service ddosprotector status`.

Kapitola 6

Dosažené výsledky

Kapitola popisuje výsledky získané při testování funkčnosti a měření reálných výkonnostních parametrů provedené implementace softwaru. Hlavním sledovaným kritériem byla datová propustnost realizovaného řešení, správná modifikace síťových dat a správné vyhodnocení útoku a jeho následná eliminace.

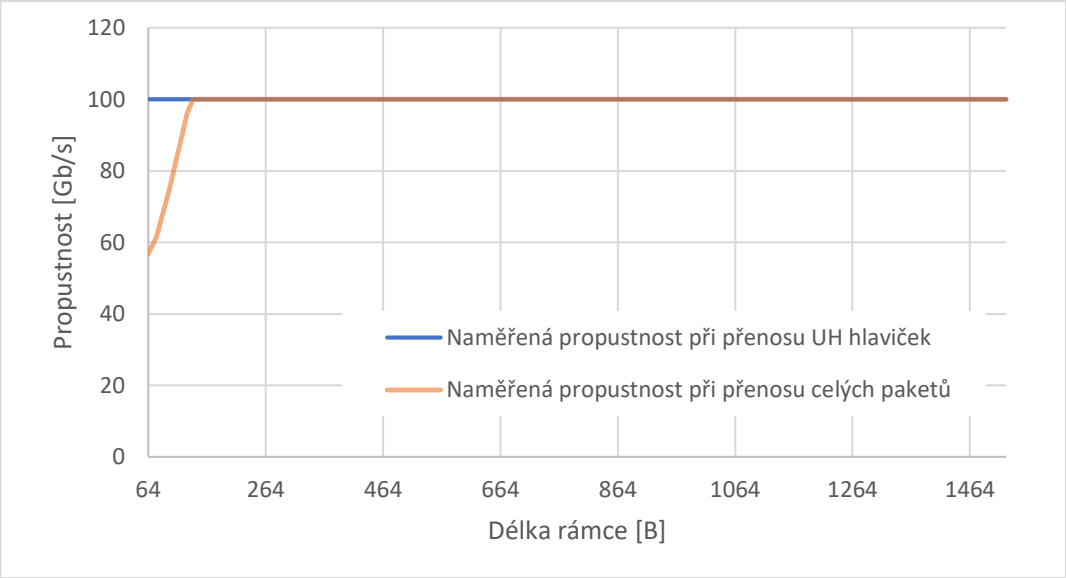
Veškerá měření probíhala na testovacím serveru, který byl osazen 12-jádrovým procesorem Xeon CPU E5-2620 s pracovní frekvencí 2.00 GHz. Server disponoval 64 GB operační paměti a FPGA kartou Combo–100G2Q. Karta byla v konfiguraci síťových portů 1x100 Gb/s.

6.1 Propustnost systému

Měření propustnosti dat probíhalo v laboratorním prostředí. Generování testovacího síťového provozu zajišťoval hardwarový tester Spirent TestCenter. Tento tester generuje umělý síťový provoz, kterým lze simulovat provoz reálné sítě. Pro účely tohoto testování byly generované ethernetové rámce se záhlavím protokolů IPv4 i IPv6, bez použití protokolu transportní vrstvy. Softwarová část byla nakonfigurována tak, že při testování nebyla v amplifikačním modulu aktivní žádná pravidla. Zdrojové adresy rámců byly generované rovnoměrně z rozsahu IP adres 192.85.1.1 – 192.85.1.255 a 2000::1 – 2000::255, čímž bylo vytvořeno 510 různých zdrojů provozu.

Měření probíhalo takovým způsobem, že na vstupní rozhraní systému byly posílané ethernetové rámce v takovém množství, že byla využita celá kapacita připojené 100G linky. Tato data byla firmwarem karty distribuována (pomocí hašování zdrojové IP adresy) na osm DMA kanálu, ze kterých softwarová část vyčítala osmi vlákny, každé pro jeden DMA kanál. Následně byly v pravidelných intervalech opakovaně vyčítány hodnoty čítačů přijatých rámců na vstupním rozhraní a hodnoty čítačů odeslaných rámců na výstupním rozhraní. Dosažená propustnost pak byla odvozena jako poměr z celkového množství příchozích rámců a počtu všech odchozích rámců. Uvedený způsob měření byl opakován pro různé délky ethernetových rámců v rozsahu 64 – 1526 bytů. Dále se rozlišovalo, zda byly pakety softwarové části předávány jako celá data, nebo ve formě UH hlaviček.

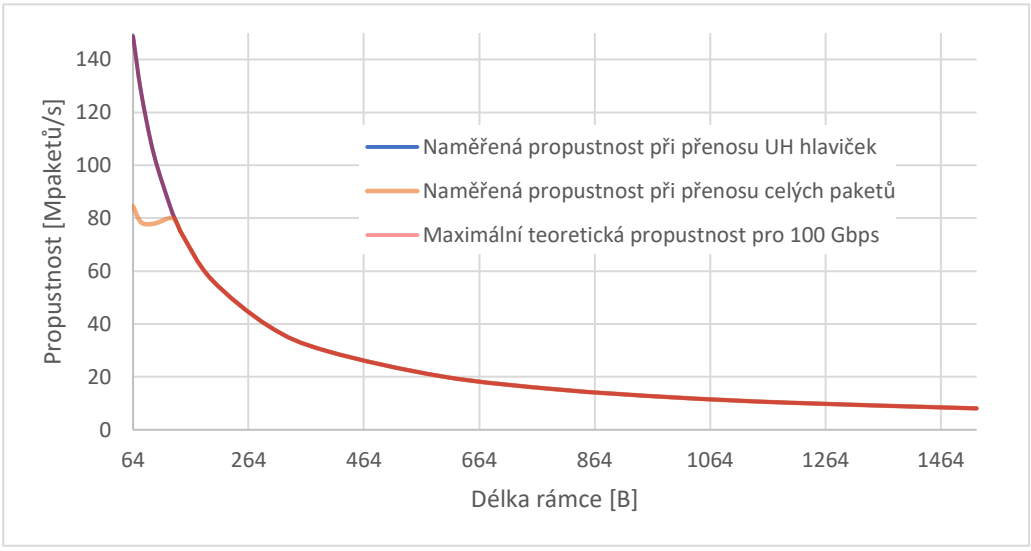
Výsledky měření jsou zobrazené v grafu 6.1. Graf znázorňuje datovou propustnost v Gbps v závislosti na délce přijímaných ethernetových rámců. Na grafu jsou znázorněny získaná data jak při přenosu UH hlaviček, tak při přenosu celých paketů do softwarové části systému. Modrá křivka znázorňuje datovou propustnost při přenosu UH hlaviček. Z grafu je vidět, že byla pro daný hardware serveru a počet procesních vláken dosažena plná datová



Obrázek 6.1: Propustnost systému v závislosti na délce rámce (v Gb/s)

propustnost 100 Gbps a to i při nejkratších paketech (64 B), kdy je počet zpracovávaných paketů největší. Oranžová křivka zobrazuje datovou propustnost při přenosu celých paketů, který je náročný jak z pohledu vytížení datové sběrnice, tak i z pohledu nutnosti získávat informace o paketu v softwarové části systému. O modifikaci a odesílání paketů se při tomto testu staral firmware karty. Na grafu lze pozorovat, že při nejkratších paketech dosahovala propustnost systému asi 56 Gbps. S vyšší velikostí přijímaných dat a tím pádem s menším počtem zpracovávaných paketů datová propustnost rostla. Při délce rámců 140 B bylo dosaženo plné datové propustnosti i při přenosu celých dat.

Na grafu 6.2 je znázorněn další pohled na výsledky provedeného měření. Graf znázorňuje



Obrázek 6.2: Propustnost systému v závislosti na délce rámce (v milionech paketů/s)

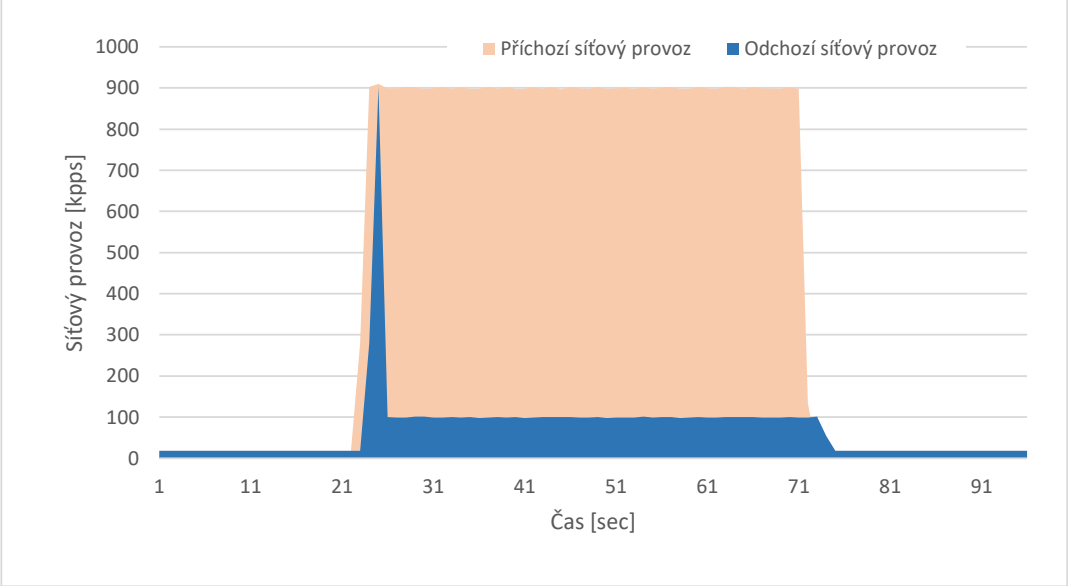
datovou propustnost v počtu přenesených paketů za jednu sekundu. V grafu lze vidět, že se červená křivka, znázorňující maximální teoretický počet přenesených paketů, překrývá s modrou křivkou, která vyjadřuje počet přenesených paketů při přenosu UH hlaviček. Oranžová křivka znázorňující počet přenesených paketů za sekundu při přenosu celých paketů má své maximum kolem 85 milionů paketů za sekundu.

6.2 Eliminace útoku

Pro otestování správné detekce a následné eliminace útoku byl vykonán test, jehož cílem bylo ověřit funkcionalitu jak softwarové, tak i firmwarové části systému. Při tomto testu byly posílány dva typy síťového provozu. První typ reprezentoval síťový provoz legitimních uživatelů, tvořilo jej celkem 60 unikátních zdrojových IP adres a v celkovém provozu tvořil 2 %. Druhý typ provozu reprezentoval provoz útočníka a útok typu DDoS, který chceme zablokovat. Tento typ provozu měl stejné parametry jako první typ, ale lišil se počtem zdrojových IP adres, kde tento typ tvořilo celkem 2000 unikátních zdrojových IP adres a pokrýval zbylých 98 % celkového provozu. Celkem bylo při útoku generováno 900 kpps a pravidlo reprezentující nežádoucí provoz mělo tento format.

```
dst net 147.229.0.0/16 protocol UDP src port 53 threshold 200kpps limit
    ↪ 100kpps
```

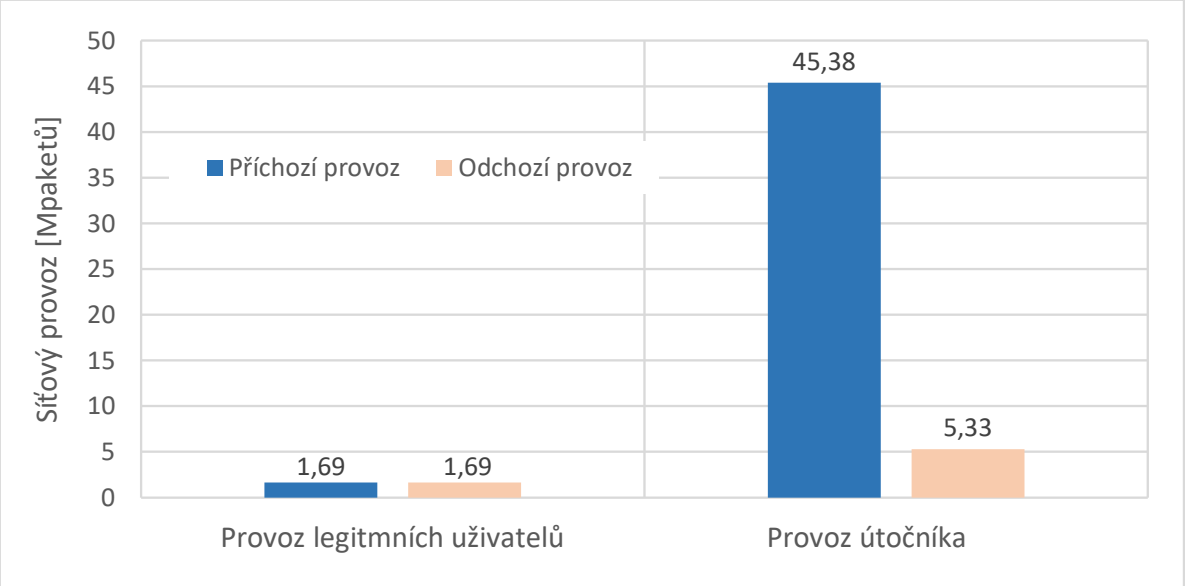
Výsledek měření je zobrazen na grafu 6.3. Graf znázorňuje poměr mezi příchozím a od-



Obrázek 6.3: Výsledky eliminace DDoS útoku

chozím síťovým provozem. Na ose X je zobrazen časový průběh měření. Ze začátku byl generován provoz pouze legitimními uživateli, poté začal generovat provoz i útočník a došlo k překročení hranice threshold. Systém útok správně detekoval a začal jej blokovat, až snížil objem provozu na hranici limit. Na obrázku lze pozorovat malá latence, než systém zareagoval na útok a začal jej blokovat. Tato latence je způsobena časovým oknem, nad kterým

detekční modul kontroluje získané statistiky. Časové okno u tohoto testu bylo nastaveno na jednu vteřinu. To znamená že detekční modul vteřinu sbíral data a následující vteřinu je vyhodnocoval. Do doby než proběhlo vyhodnocení (blokování) byl propouštěn veškerý provoz. Na dalším grafu 6.4 lze vidět, jaký provoz byl systémem blokován.



Obrázek 6.4: Výsledky eliminace DDoS útoku

Graf znázorňuje poměr mezi příchozím a odchozím síťovým provozem generovaný legitimními uživateli a útočníkem. Z hodnot lze vyčíst, že při úspěšném blokování DDoS útoku nebyl provoz legitimních uživatelů nijak ovlivněn.

6.3 Modifikace síťových dat

V laboratorním prostředí byly pomocí hardwarového testeru Spirent TestCenter realizované testy na ověření správné funkcionality jednotlivých částí pro úpravu síťového provozu. Realizovány byly tyto testy:

Test dekrementace a kontrola hodnoty TTL a HOP LIMITU sloužil k ověření, že se u paketů správně snižuje a kontroluje doba jejich životnosti. Test probíhal tak, že na server (systém) připojený k hardwarovému testeru byl posílán náhodně vygenerovaný síťový provoz. Následně provoz prošel přes systém, kde se provedla kontrolovaná činnost, a poté byl provoz vrácen zpět na tester. Na testeru probíhal záchyt příchozích síťových dat, které přicházely ze systému. Tato data byla následně zkontrolována, zda odpovídají očekávanému výsledku. Konkrétně byly prováděny dva testy, při kterých byla vypnuta tato funkcionalita ve firmwaru karty, aby ji bylo možné ověřit v softwarové části. V prvním testu se posílaly pakety, které měly dobu životnosti nastavenou na 1. Při tomto testu bylo ověřováno, že software tyto pakety zahodí. Při druhém testu byly posílány pakety, které měly dobu životnosti větší než 1. Tento test ověřoval, že pakety nebudou zahozené a že bude snížena doba jejich životnosti o jedna.

Test přepisu MAC adres kontroloval, zda správně probíhá nahrazení zdrojové a cílové MAC adresy. Test byl realizován prostřednictvím generování a zachytávání provozu,

podobně jako test při kontrole TTL. Pro ověření funkčnosti byl realizován jeden test, při kterém se kontrolovalo, zda systém provádí správně úpravu paketu. Konkrétně přepis zdrojové a cílové MAC adresy.

Test výměny VLAN ověřoval, zda probíhá správně výměna položky VLAN ID podle zvolené konfigurace. Test opět probíhal s vypnutou funkcionalitou ve firmwaru karty. Celkem byly prováděny dva testy. Prvním z nich se kontrolovalo, zda při výchozí konfiguraci, tj. přepisování se neprovádí, nedochází k modifikaci paketu. U druhého testu byly v konfiguračním souboru pokryty všechny možné kombinace hodnot položky VLAN ID a k nim náhodně vygenerované hodnoty, na které se má položka VLAN ID přepsat. Po přeposlání paketů přes systém byla provedena kontrola, zda byla uskutečněná výměna hodnoty VLAN ID a zda její hodnota odpovídá očekávané hodnotě podle konfigurace.

Kapitola 7

Závěr

Cílem bakalářské práce bylo navrhnout a implementovat softwarovou část systému pro ochranu před DoS útoky. Při návrhu a následné realizaci byl kladen důraz především na dosažení vysoké datové propustnosti 100 Gbps a také na splnění všech požadavků, které vyplynuly z analýzy dané problematiky.

V rámci řešení bakalářské práce jsem se nejprve řádně seznámil s architekturou a principy dnešních počítačových sítí v podobě vrstvových modelů, protokolů a principů použitých při síťové komunikaci. Následně jsem se podrobně věnoval problematice kybernetických DoS útoků, kde jsem získal informace o nejznámějších typech DoS útoků. Dále jsem se zaměřil na technologii programovatelných hradlových polí FPGA, která se používá pro hardwarovou akceleraci v síťových kartách COMBO. Tyto karty se využívají jako hardwarová část systému, vyvíjeného v rámci sdružení CESNET pro ochranu před DoS útoky. V neposlední řadě jsem se také zaměřil na samotné fungování karet a obeznámil se se specializovaným firmwarem těchto karet a rozhraním, které je pro práci s ním poskytováno.

Po získání vědomostí jsem realizoval podrobný implementační návrh softwarové části tohoto systému pro ochranu před DoS útoky. Následně jsem na základě realizovaného návrhu provedl implementaci softwarové části a důkladně ji zdokumentoval. Správnou funkcionalitu vytvořeného softwaru, ale i systému jako celku, jsem ověřil prostřednictvím laboratorního testování na hardwarově akcelerované kartě COMBO-100G2Q s využitím hardwarového testeru Spirent TestCenter, který umožňuje generování umělého síťového provozu na plné rychlosti 100 Gbps.

Už v době psaní bakalářské práce byla jedna z prvních verzí tohoto systému pro ochranu před DoS útoky, včetně nově implementované softwarové části, pilotně nasazena v síťové infrastruktuře akademické sítě CESNET. Systém mají aktuálně k dispozici síťový administrátoři, kteří na zařízení provádí své vlastní testy. K tomuto účelu je do systému přesměrován živý síťový provoz kolejí Liberecké technické univerzity. Dalším pokračováním této práce bude rozšiřování vlastností systému o nové způsoby detekce a eliminace dalších typů DoS útoků. Bude poskytována také podpora pro realizaci návrhů, které přicházejí od síťových administrátorů, kteří systém v současné době testují.

Literatura

[1] CESNET, zájmové sdružení právnických osob. [Online; navštíveno 25.3.2018].
URL https://www.cesnet.cz/sdruzeni/

[2] Free Booter. [Online; navštíveno 15.03.2018].
URL https://freebooter.co

[3] Imperva Incapsula: NTP Amplification | DDoS Attack Glossary. [Online; navštíveno 16.3.2018].
URL
https://www.incapsula.com/ddos/attack-glossary/ntp-amplification.html

[4] *Princip DNS Amplification útoku*. [Online; navštíveno 16.03.2018].
URL https://www.csirt.cz/page/2792/princip-dns-amplification-utoku/

[5] *What is a Botnet?* Cloudfare, [Online; navštíveno 2.5.2018].
URL https://www.cloudflare.com/learning/ddos/what-is-a-ddos-botnet/

[6] *FPGAs vs ASICs*. Gisselquist Technology, LLC, 10 2017, [Online; navštíveno 2.5.2018].
URL http://zipcpu.com/blog/2017/10/13/fpga-v-asic.html

[7] *History of DDoS Attacks*. Březen 2017, [Online; navštíveno 28.1.2018].
URL https://security.radware.com/ddos-knowledge-center/ddos-chronicles/ddos-attacks-history/

[8] cen: *Anonymous*. Listopad 2015, [Online; navštíveno 28.1.2018].
URL http://whatis.techtarget.com/definition/Anonymous

[9] Štěpán Friedl; Puš, V.; Matoušek, J.; aj.: Technical Report 7/2013 Designing a Card for 100 Gb/s Network Monitoring. CESNET, zájmové sdružení právnických osob, July 2013, [Online; cit. 25.3.2018].
URL https://www.cesnet.cz/wp-content/uploads/2014/02/card.pdf

[10] Kerner, S. M.: *DDoS Attacks: Growing, but How Much?* Duben 2013, [Online; navštíveno 28.1.2018].
URL https://www.esecurityplanet.com/network-security/ddos-attacks-growing-but-how-much.html

[11] Klubal, M.: *Problematika sítí typu botnet*. Mendělova univerzita v Brně, Provozně ekonomická fakulta, 2013.
URL http://is.mendelu.cz/zp/portal_zp.pl?prehled=vyhledavani;podrobnosti=55000;zp=37989;download_prace=1;lang=cz

[12] Kuka, M.: *Hardwarově akcelerované zařízení pro ochranu před DoS útoky*. Bakalářská práce, Vysoké učení technické v Brně, Fakulta informačních technologií, 2017.
URL http://www.fit.vutbr.cz/study/DP/BP.php?id=19924

[13] Mallory, T.; Kullberg, A.: *Incremental Updating of the Internet Checksum*. Leden 1990, [Online; navštíveno 7.4.2018].
URL https://tools.ietf.org/html/rfc1141

[14] Matoušek, P.: *Síťové služby a jejich architektura*. Publishing house of Brno University of Technology VUTIUM, 2014, ISBN 978-80-214-3766-1, 396 s.
URL http://www.fit.vutbr.cz/research/view_pub.php.cs?id=10567

[15] Rouse, M.: *hacktivism*. Červen 2007, [Online; navštíveno 28.1.2018].
URL http://searchsecurity.techtarget.com/definition/hacktivism

[16] Rouse, M.: *Anonymous napadli servery OSA, web české vlády i Evropského parlamentu*. Leden 2012, [Online; navštíveno 28.1.2018].
URL https://technet.idnes.cz/anonymous-napadli-servery-osa-web-ceske-vlady-i-evropskeho-parlamentu-1mp-/sw_internet.aspx?c=A120126_134112_sw_internet_nyv

[17] Rouse, M.: *Anonymous napadli weby Babišových firem, žádají zrušení evidence tržeb*. Srpen 2016, [Online; navštíveno 28.1.2018].
URL https://zpravy.idnes.cz/anonymous-napadli-weby-babisovych-firem-fd8-/domaci.aspx?c=A160801_192740_domaci_cen

[18] tdm, R., iDNES.cz: *Operace odplata: fanoušci WikiLeaks zahájili kybernetickou pomstu*. Prosinec 2010, [Online; navštíveno 28.1.2018].
URL https://zpravy.idnes.cz/operace-odplata-fanousci-wikileaks-zahajili-kybernetickou-pomstu-10b-/zahranicni.aspx?c=A101209_1496218_zahranicni_btw

[19] Vyleťal, M.: *Tisíce tuzemských e-shopů mělo výpadky, může za to masivní DDoS útok*. Prosinec 2011, [Online; navštíveno 28.1.2018].
URL https://www.lupa.cz/clanky/tisice-tuzemskych-e-shopu-melo-vypadky-muze-za-to-masivni-ddos-utok/

[20] Čmelík, M.: *Postřehy z bezpečnosti: rekordní DDoS útok 1,1 Tbps*. Říjen 2016, [Online; navštíveno 28.1.2018].
URL https://www.root.cz/clanky/postrehy-z-bezpecnosti-rekordni-ddos-utok-1-1-tbps/

Příloha A

Formát UH hlavičky

Tabulka A.1

Byty	**Velikost**	**Pole**	**Popis**
00-15	16B	SourceIP	Zdrojová IP adresa v network byte orderu. (IPv4 i IPv6, nuly pokud paket není IP). (IPv4 0-7B 0x0, 8-11B ipv4, 12-15B 0xffffffff)
16-31	16B	DestinationIP	Cílová IP adresa v network byte orderu. (IPv4 i IPv6, nuly pokud paket není IP). (IPv4 0-7B 0x0, 8-11B ipv4, 12-15B 0xffffffff)
32-33	2B	SourcePort	Číslo zdrojového TCP, UDP, SCTP portu (nuly v případě fragmentovaného paketu)
34-35	2B	DestinationPort	Číslo cílového TCP, UDP, SCTP portu (nuly v případě fragmentovaného paketu)
36-37	2B	Octets	Velikost paketu od začátku IP hlavičky do konce bez CRC.
38-38	1B	Protocol	L4 protokol (6=TCP, 17=UDP, ...)

Tabulka A.2

39-39	1B	IPVer/Flags/Fragmentation	Bity 0-1: Typ následujícího pole UH: 0 = následující pole je nevyužité, 1 = následující pole obsahuje MPLS, 2 = následující pole obsahuje jednu VLAN TCI, 3 = následující pole obsahuje dvě VLAN TCI
			Bit 2: Typ posledního pole UH: 0 = poslední pole UH je nevyužité, 1 = poslední pole UH obsahuje MPLS
			Bit 3: Verze protokolu IP (0 = IPv4, 1 = IPv6)
			Bit 4: Validita IP (0 = neIP paket, 1 = IP paket)
			Bit 5: Validita fragmentace (0 = následující bit není validní, 1 = následující bit je validní)
			Bity 6-7: Fragmentace (význam bitů: 11 = nefragmentovaný, 10 = první fragment, 00 = prostřední fragment, 01 = poslední fragment)
40-43	4B	Inner MPLS/VLAN	Nejvnitřnější (poslední) MPLS label nebo spodní 2B nejvnitřnější (poslední) VLAN TCI a horní 2B první VLAN TCI (nebo nuly). Pokud je jen jedna VLAN, budou vyplněné horní 2B. Pokud jsou 2 a více MPLS a nějaká VLAN, má VLAN v tomto poli přednost.
44-47	1B	TCP Flags	tcp příznaky FIN (0), SYN (1), RST (2), PSH (3), ACK (4), URG (5), ECE (6), CWR (7)

Příloha B

Ukázka stavového souboru

- **ddpStatsLastChange**: Datum a čas poslední změny souboru.
- **ddpIfcCount**: Počet aktivních interfaců síťové karty.
- **ddpLinkStatus**: Status linky.
- **ddpThreadsCount**: Počet aktivních procesních vláken.
- **ddpAvailableFilterSpace**: Počet IP adres, které je možné vložit do IP filteru.
- **ddpInOctets**: Celkový počet přijatých bytů ze všech interfaců.
- **ddpInPkts**: Celkový počet přijatých paketů ze všech interfaců.
- **ddpInReceivedPkts**: Celkový počet úspěšně přijatých paketů ze všech interfaců.
- **ddpInErrorPkts**: Celkový počet přijatých chybných paketů ze všech interfaců.
- **ddpInLostPkts**: Celkový počet ztracených paketů z příčiny přeplněných bufferů ze všech interfaců.
- **ddpOutOctets**: Celkový počet odeslaných bytů ze všech interfaců.
- **ddpOutPkts**: Celkový počet odeslaných paketů ze všech interfaců.
- **ddpOutErrorPkts**: Celkový počet paketů, které se kvůli chybě nepodařilo odeslat ze všech interfaců.
- **ddpIfc.ddpInOctets.X**: Počet viděných bytů na interfaci `X`.
- **ddpIfc.ddpInPkts.X**: Počet viděných paketů na interfaci `X`.
- **ddpIfc.ddpInReceivedPkts.X**: Počet přijatých paketů na interfaci `X`.
- **ddpIfc.ddpInErrorPkts.X**: Počet přijatých chybných paketů na interfaci `X`.
- **ddpIfc.ddpInLostPkts.X**: Počet ztracených paketů z příčiny přeplněných bufferů na interfaci `X`.
- **ddpIfc.ddpOutOctets.X**: Počet odeslaných bytů na interfaci `X`.

- **ddpIfc.ddpOutPkts.X**: Počet odeslaných paketů na interfaci `X`.
- **ddpIfc.ddpOutErrorPkts.X**: Počet paketů, které se kvůli chybě nepodařilo odeslat na interfaci `X`.
- **ddpTtlDroppedPkts**: Počet zahozených paketů z důvodu vypršení doby životnosti.
- **ddpTtlDroppedPkts/s**: Počet zahozených paketů za sekundu z důvodu vypršení doby životnosti
- **ddpAmpBlockedPkts**: Celkový počet paketů zablokovaných amplifikačním modulem.
- **ddpAmpBlockedOctets**: Celkový počet bytů zablokovaných amplifikačním modulem.
- **ddpAmpBlockedPkts/s**: Počet zablokovaných paketů za sekundu amplifikačním modulem.
- **ddpAmpBlockedMb/s**: Počet zablokovaných Mb za sekundu amplifikačním modulem.
- **ddpThreadsPkts**: Celkový počet paketů přijatých softwarovou částí.
- **ddpThreadsOctets**: Celkový počet bytů přijatých softwarovou částí.
- **ddpThreadsPkts/s**: Počet paketů za sekundu přijatých softwarovou částí.
- **ddpThreadsMb/s**: Počet Mb za sekundu přijatých softwarovou částí.
- **ddpThread.ddpInOctets.N**: Počet bytů přijatých v procesním vlákně `N`.
- **ddpThread.ddpInPkts.N**: Počet paketů přijatých v procesním vlákně `N`.
- **ddpThread.ddpInPkts/s.N**: Počet paketů za sekundu přijatých v procesním vlákně `N`.
- **ddpThread.ddpInMb/s.N**: Počet Mb za sekundu přijatých v procesním vlákně `N`.
- **ddpAmpNon-matchingOctets**: Počet bytů, které se neshodovali s žádným pravidlem v amplifikačním modulu.
- **ddpAmpNon-matchingPkts**: Počet paketů, které se neshodovali s žádným pravidlem v amplifikačním modulu.
- **dpAmpNon-matchingPkts/s**: Počet paketů za sekundu, které se neshodovali s žádným pravidlem v amplifikačním modulu.
- **ddpAmpNon-matchingMb/s**: Počet Mb za sekundu, které se neshodovali s žádným pravidlem v amplifikačním modulu.
- **ddpAmpCurrentActiveDDoS**: Počet aktivních DDoS útoků detekovaných amplifikačním modulem.

Příloha C

Ukázka konfiguračních souborů

YAML konfigurace

```
ddosprotector-common:
   window-size: 3
   receiver-mode: 'sze'
   src-mac-address: "aa:bb:cc:dd:ee:ff"
   dst-mac-address: "00:11:09:95:26:FE"
   vlan-file: "/etc/liberouter/ddosprotector/vlan.conf"
   daemon: Yes
   pid-file: "/cc/pid.file"

io-module:
   sze-device:
      device: "/cc/sze-dev.file"
      thread: 8
      node: 1

dcpro:
   filter-file: "/etc/liberouter/ddosprotector/filter.conf"
   design-xml-file: "/cc/design.file"
   combo-device: "/cc/dev.file"
   ibuf-min: 64
   ibuf-max: 16000

amplification-module:
   amp-rules-file: "/cc/syn_rules.file"
   amp-hash-size: 8192
   whitelist-file: "/cc/whitelist.file"


logs:
   status-file: "/cc/status.file"
   log-file: "/cc/log.file"
```

Soubor s konfiguračními pravidly pro amplifikační modul

```
// This is single line comment
"Serverovna FIT VUT" dst net 128.100.0.0/16 protocol UDP src port >53
    ↪ dst port 1024 length <1200 threshold 5Mpps 10Gbps limit 2250kpps
    ↪ 5Gbps

/*
* This is
* multi line
* comment
*/

"Brno - Kounicova" dst net 192.168.1.0/24 protocol 6 src port 53
    ↪ tcpflags S threshold 25000Mbps limit 12000Mbps

// 3 rules in a single line, statistics are updated per each prefix
    ↪ separately dst net 192.168.1.0/24
192.168.2.0/24 192.168.3.0/24 length 500-1526 fragmentation YES
    ↪ threshold 1Gbps limit 200Mbps
```

Ukázka konfiguračního souboru pro whitelist

```
10.0.0.0/8
172.16.0.0/12
192.168.0.0/16
fe80::/10
```

Ukázka konfiguračního souboru vlan editor

```
0
66 99
99 111
42 33
```

Příloha D

Struktura obecných funkcí I/O modulu

```
1
2  typedef struct io_function {
3
4      void *global_init_data;
5
6      /**
7       * @brief Globalni inicializace rozhrani.
8       */
9      ddp_ecode_t (*init)(void **global_init_data, void *program_config);
10
11     /**
12      * @brief Globalni deinicializace rozhrani.
13      */
14     void (*deinit)(void *global_init_data);
15
16     /**
17      * @brief Pocet vlaken, ktere maji byt vytvorene.
18      */
19     int (*threads_count) (void *config);
20
21     /**
22      * @brief Inicializace vlakna.
23      */
24     ddp_ecode_t (*thread_init)(unsigned thread_id, void *config,
           void~**data);
25
26     /**
27      * @brief Deinicializace vlakna.
28      */
29     void (*thread_deinit)(void *data);
30
31     /**
```

```
32      * @brief Vycteni paketu.
33      */
34     packet_read_code_t (*packet_read)(void *data, packet_t *p);
35
36
37     /**
38      * @brief Zapis paketu.
39      */
40     ddp_ecode_t (*packet_write)(void *data, unsigned char *packet,
           uint16_t length);
41
42 } io_function_t;
```

Příloha E

Obsah DVD

```
/
├── source/
│   └── ddosprotector/
├── text/
├── bp-xsiska12.pdf
└── README.txt
```

Adresář `source` obsahuje zdrojové kódy implementovaného řešení včetně dokumentace ve formátu Doxygen.

Adresář `text` obsahuje zdrojové soubory textu bakalářské práce pro její možnou úpravu a opětovné vysázení systémem LaTeX, včetně zdrojových souborů použitých obrázků.

Soubor `bp-xsiska12.pdf` obsahuje vysázený text bakalářské práce ve formátu PDF.

Soubor `README.txt` obsahuje informace o adrésářové struktuře a obsahu přiloženého DVD. Dále obsahuje pokyny pro umístění souborů do adresářové struktury repozitáře Dcpro a instrukce pro využití překladového systému.