UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

IMPLEMENTÁCIA PROTOKOLU IPV6 V PRODUKČNOM PROSTREDÍ

2014 **Anna LIPTAJOVÁ**

UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
NÁZOV FAKULTY PRÍRODOVEDECKÁ FAKULTA

IMPLEMENTÁCIA PROTOKOLU IPV6 V PRODUKČNOM PROSTREDÍ

BAKALÁRSKA PRÁCA

Študijný program:	slovenský jazyk a literatúra - informatika
Pracovisko (katedra/ústav):	Ústav informatiky
Vedúci práce:	RNDr. JUDr. Pavol Sokol

Košice 2014

Anna LIPTAJOVÁ

Abstrakt v štátnom jazyku

V súčasnom digitálnom svete existuje veľké množstvo zariadení pripájajúcich sa k Internetu. So stúpajúcim počtom zariadení klesá možnosť prideliť každému z nich jedinečnú IP adresu. Aktuálny protokol IPv4 totiž umožňuje vytvoriť len približne štyri miliardy IP adries. Sprísnenie kritérií na prideľovanie voľných IP adries alebo používanie NAT je len prostriedkom na oddialenie problému. Skutočné riešenie prichádza s novým protokolom sieťovej vrstvy – IPv6. Protokol IPv6 so sebou prináša okrem obrovského adresného priestoru aj celý rad zmien. Zásadnou zmenou je napríklad zmena formátu základnej hlavičky alebo zrušenie ARP protokolu, či zavedenie bezstavovej automatickej konfigurácie. Široké spektrum zmien podnecuje potencionálnych útočníkov k testovaniu bezpečnosti IPv6 siete. Cieľom tejto práce je preto poukázať na možné nedostatky a bezpečnostné riziká zavedenia protokolu IPv6 do praxe. Výstupom je aplikácia schopná simulovať bezpečnostné útoky vedené na IPv6 sieť a overiť tak jej bezpečnostné aspekty. Na simuláciu útokov sme použili testovaciu sieť vytvorenú v priestoroch ŠDaJ UPJŠ v Košiciach. Uvedená aplikácia poskytuje testovanie IPv6 siete prierezom bezpečnostných rizík rozdelených do jednotlivých bezpečnostných okruhov. Aplikácia je napísaná modulárne, čo umožňuje doplniť ju o ďalšie typy útokov a v neposlednej miere umožňuje aj modelovanie vlastných IPv6 paketov na otestovanie budúcich bezpečnostných hrozieb.

Abstrakt v cudzom jazyku

There is a large number of devices connecting to the Internet in the current digital world. The possibility of assigning an unique IP address to each one device decreases with increasing count of devices. The current IPv4 protocol enables to create approximately four billion IP addresses. One of the way for time displacement of actual problem with the free IP addresses is usage of a stricter criteria for allocation of IP and/or usage of NAT. The real solution comes with usage of a new protocol of network layer – IPv6. The IPv6 protocol brings expect with a huge address space many changes .A new format style of basic header, a deletion of ARP protocol and an automatic stateless configuration are the main changes compare to previous type of IP version. This wide spectrum of the improvements stimulate the potential attackers to testing of stability and security of IPv6 network. The aims of this bachelor thesis are focus to the possible weaknesses and the security risks of IPv6 usage in practice. The main output of this thesis is application for simulation of IPv6 network attacks and in this way test its security aspects. We used test network formed in area of ŠDaJ UPJŠ in Košice to simulate attacks. Our application provides IPv6 network testing with focus to security risks divided into the several security fields. The application has been created by modular programming way and for this reason it is possible to extend its to another attack types, but mainly it is possible to model own IPv6 packets for test of the future possible security damage efforts.

Obsah

Zoznam skratiek a značiek

ICMP	Internet Control Message Protocol
ICMPv6	Internet Control Message Protocol verzie 6
IPv4	Internet protokol verzie 4
IPv6	Internet protokol verzie 6
RFC	Request For Comments
TCP	Transmission Control Protocol

Slovník termínov

Bezpečnostná hrozba je možnosť potencionálneho porušenia bezpečnostných pravidiel.

Bezpečnostný útok je zámerný pokus o obídenie bezpečnostných bariér zabezpečeného informačného systému.

Informačná bezpečnosť je schopnosť siete alebo informačného systému ako celku odolať s určitou úrovňou spoľahlivosti náhodným udalostiam alebo nezákonnému či zákernému konaniu.

RFC dokumenty sú dokumenty, ktoré obsahujú normy na riadenie Internetovej komunikácie.

Sieťový smerovač je sieťové zariadenie, ktoré umožňuje prenos dát medzi viacerými počítačovými sieťami.

Úvod

Svetová populácia neustále rastie. V súčasnosti (v roku 2014) žije na svete približne 7 miliárd ľudí. Z toho 2,5 miliardy ľudí reálne používa Internet, a to za pomoci približne 9,6 miliárd zariadení. Každé zariadenie v sieti potrebuje svoju jednoznačnú IP adresu, vďaka ktorej ho vieme identifikovať. Protokol IPv4 nám však poskytuje celkovo len približne 4 miliardy IP adries. Otázkou je, čo sa stane, ak sa tieto adresy minú?

Sprísnenie kritérií na prideľovanie voľných IP adries alebo používanie NAT je len prostriedkom na oddialenie problému. Skutočné riešenie prichádza s novým protokolom sieťovej vrstvy – IPv6. Jeho vývoj začal začiatkom 90. rokov, kedy bolo predstavené RFC 1883 – Internet Protocol Version 6 – Specification. Protokol IPv6 nielenže dlhodobo vyriešil problém s nedostatkom voľných IP adries, ale zároveň zmenil aj niekoľko zabehnutých pravidiel stanovených protokolom IPv4. Zásadnou zmenou je zmena formátu hlavičky – niektoré voľby hlavičky IPv4 sa zrušili, funkcia iných prešla do voliteľných rozširujúcich hlavičiek. Nový spôsob adresácie priniesol celkovo $3{,}4\times10^{38}$ voľných IPv6 adries. Správy ICMPv6 sa rozšírili o niekoľko nových typov a zaviedli sa ako povinná súčasť komunikácie. K stavovej automatickej konfigurácii pomocou DHCP servera pribudla aj bezstavová automatická konfigurácia založená na ohláseniach smerovačov. Namiesto ARP protokolu sa na vyhľadávanie linkových adries začala používať metóda objavovania susedov. Protokol IPv6 so sebou prináša aj nové možnosti – mobilita umožňuje dostupnosť domácej adresy pomocou mobilného uzla, IPSec sa na druhej strane orientuje na zvýšenie bezpečnosti prenášaných paketov. Veľa zmien však so sebou prináša aj určité negatíva – prevádzka protokolu IPv6 nebola ešte úplne a natrvalo spustená, čoho dôsledkom je, že mnoho bezpečnostných rizík a bezpečnostných zraniteľností sa objaví až po jej úplnom zavedení do praxe.

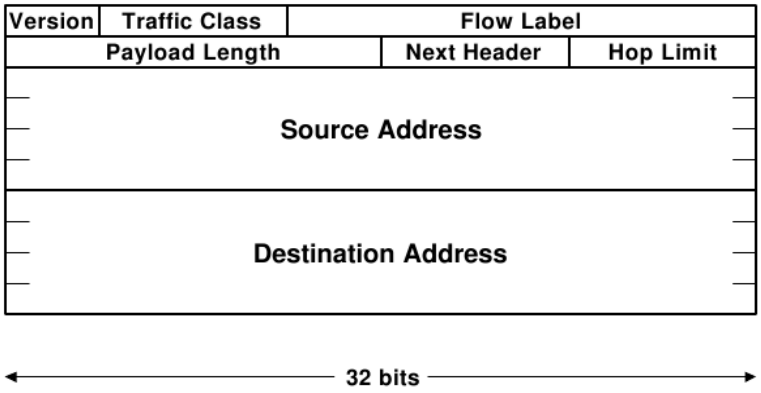
Cieľom našej bakalárskej práce je implementovať protokol IPv6 na smerovačoch, prepínačoch a sieťových serveroch produkčnej siete a zároveň analyzovať známe bezpečnostné riziká používania tohto protokolu. Jadro práce je rozdelené do jednotlivých kapitol podľa typov bezpečnostných hrozieb. Každá kapitola sa venuje určitým typom bezpečnostných problémov, najmä však tým, ktoré so sebou prináša zavedenie protokolu IPv6. Záver každej kapitoly je doplnený o zoznam bezpečnostných opatrení na čiastočné, resp. úplné riešenia bezpečnostných hrozieb. Posledná kapitola sa zameriava na popis návrhu a implementácie testovacej siete, ktorej cieľom je otestovať niektoré bezpečnostné hrozby.

1 Základná charakteristika protokolu IPv6

Protokol IPv6 je protokolom tretej sieťovej vrstvy TCP/IP, ktorá zabezpečuje prenos dát medzi zariadeniami v Internete. Vznikol na začiatku 90. rokov, kedy bolo už zrejmé, že počet voľných IP adries, ktoré poskytuje protokol IPv4, nebude dostačujúci. Nový protokol mal spĺňať viacero predpokladov, počnúc vyhovujúcou veľkosťou adresného priestoru, cez podporu multicastu, anycastu a unicastu, jednotou adresného priestoru v rámci Internetu aj vnútornej siete, hierarchickým smerovaním v súlade s hierarchickým usporiadaním adries, zvýšenie bezpečnosti, podporu služieb so zabezpečenou kvalitou, optimalizáciu pre vysokorýchlostné smerovanie, automatickú konfiguráciu, podporu mobility a v neposlednom rade hladký a plynulý prechod od protokolu IPv4 k novému protokolu [1]. Výsledkom týchto požiadaviek bol v roku 1995 vydaný dokument RFC 1883 Internet Protocol, Version 6 (IPv6) Specification, kde sa uviedli základné princípy protokolu IPv6. Neskôr bol tento dokument doplnený o ďalšie špecifikácie uvedené v dokumente RFC 2460.

1.1 Popis protokolu

Protokol IPv6 prichádza s novou štruktúrou základnej hlavičky, navrhnutou tak, aby zefektívnila manipuláciu s paketmi. Jej veľkosť je fixná (40 bytov). Funkcionalitu základnej hlavičky však môžeme doplniť voliteľnými rozširujúcimi hlavičkami (napr. fragmentáciou alebo smerovaním). Rozširujúce hlavičky majú presne stanovené poradie v akom majú nasledovať za základnou hlavičkou. Za rozširujúcimi hlavičkami sa k paketu pripája identifikátor prenášaných dát (napr. TCP segment) [1].



Obr. 1: Základná hlavička datagramu IPv6 [2]

V protokole IPv6 hrá hlavnú úlohu **nový spôsob adresovania**. Vďaka dostatočnej dĺžke IPv6 adresy (128 bitov) vieme adresovať 3,4 $\times 10^{38}$ zariadení. Za štandardnú formu zápisu IPv6 adresy považujeme formát x:x:x:x:x:x:x:x, kde x predstavuje skupinu jedného až štyroch číslic šestnástkovej sústavy [3].

Adresu IPv6 možno klasifikovať do troch kategórií, v závislosti na tom, či identifikuje len jedno rozhranie alebo celú skupinu rozhraní [4] :

- **unicast adresa** identifikuje práve jedno zariadenie, čo znamená, že paket zaslaný na túto adresu bude doručený konkrétnemu zariadeniu, ktoré adresa identifikuje,
- **multicast adresa** identifikuje skupinu zariadení, čo znamená, že paket zaslaný na túto adresu bude doručený všetkým členom skupiny,
- **anycast adresa** identifikuje skupinu zariadení, pričom paket zaslaný na túto adresu bude doručený členovi skupiny, ktorý sa nachádza najbližšie k odosielateľovi paketu.

Ďalšie rozdelenie IPv6 adresy je založené na rozsahu jej použitia a súčasne aj na základe prefixu [4] :

- **lokálna linková adresa** (fe80::/10) slúži na jednoznačné rozpoznanie zariadenia v lokálnej sieti. Každé zariadenie s IPv6 adresou má aspoň jednu lokálnu linkovú adresu. Využívajú sa napríklad pri stavovej automatickej konfigurácii pomocou DHCPv6,
- **lokálna miestna adresa** (fec0::/10) mala identifikovať zariadenie v rámci miestnej podsiete (napr. podsieť organizácie), avšak v súčasnosti sa už nepoužíva. Dokonca RFC 3879 s názvom Deprecating Site Local Addresses odporúča nepodporovať tento druh adries pri implementácii [24],
- **globálna jedinečná adresa** identifikuje svojho nositeľa v rámci Internetu. Z tohto dôvodu by mala byť celosvetovo jednoznačná.

V rámci automatickej konfigurácie nám IPv6 **prináša bezstavový typ automatickej konfigurácie** založený na **ohlásení smerovačov (Router Advertisement)**. Ohlásenie smerovača posiela v náhodných časových intervaloch každý smerovač, do každej siete, v ktorej je pripojený. Najskôr prebehne určenie vlastnej adresy – vytvorí sa vlastná lokálna linková adresa. Potom sa pomocou objavovania susedov overí duplicita tejto lokálnej linkovej adresy a následne uzol počká na ohlásenie smerovača alebo si ho vyžiada pomocou tzv. **výzvy pre smerovač** [1]. Takýto spôsob automatickej konfigurácie prináša so sebou niekoľko výhod. Na pridelenie IP adresy nepotrebujeme DHCP server, čo znamená, že konfigurácia sieťových nastavení klienta prebieha automaticky, hneď

po zapojení zariadenia do siete. Veľkú výhodu predstavuje pre aplikácie, ktoré vyžadujú zabezpečené pripojenie bez ďalších sprostredkovateľov alebo pre aplikácie, ktoré potrebujú zvýšiť rýchlosť prenosu dát. Použitie bezstavovej automatickej konfigurácie sa považuje za nákladovo výhodné a vhodné aj pre wireless pripojenia [5].

Ďalšiu novinku v protokole IPv6 predstavuje **rozpoznávanie susedov**. Má bohaté využitie nielen pri zisťovaní linkových adries, ale aj pri detekcii duplicitných adries, hľadaní smerovačov, pri presmerovaní a zisťovaní dostupnosti susedov, či zisťovaní prefixov a parametrov siete počas automatickej konfigurácie. Napríklad pri zistení linkovej adresy sa najskôr zreťazí posledných 24 bitov hľadanej IP adresy s prefixom pre skupinové adresy, čím sa vytvorí skupinová adresa vyzývaného uzla. Následne sa pošle ICMP správa tzv. **výzva susedovi**, pričom sa spätne očakáva správa , tzv**. ohlásenie suseda**, ktorá v sebe nesie hľadanú linkovú adresu [1].

K ďalším výhodám protokolu IPv6 patrí aj zvýšenie bezpečnosti pomocou IPSec alebo zavedenie mobility pre lepšiu komunikáciu medzi mobilnými zariadeniami. Týmto výhodám budeme venovať pozornosť v nasledujúcich podkapitolách.

1.2 Základné rozdiely medzi protokolmi IPv4 a IPv6

Protokol IPv6 vzniká ako efektívnejšia a inovatívnejšia náhrada protokolu IPv4. Znižuje sa zložitosť, zavádzajú sa nové postupy a vylepšuje sa funkcionalita. Preto je dôležité pri jeho implementácii rozlišovať spoločné ale aj odlišné prvky oproti predchádzajúcemu protokolu IPv4.

Ústredný rozdiel medzi protokolom IPv4 a IPv6 predstavuje už spomenutá základná hlavička paketu. Hlavička protokolu IPv6 má minimalizovaný počet parametrov – niektoré parametre známe z IPv4 sa odstránili zo základnej hlavičky, napríklad [6]:

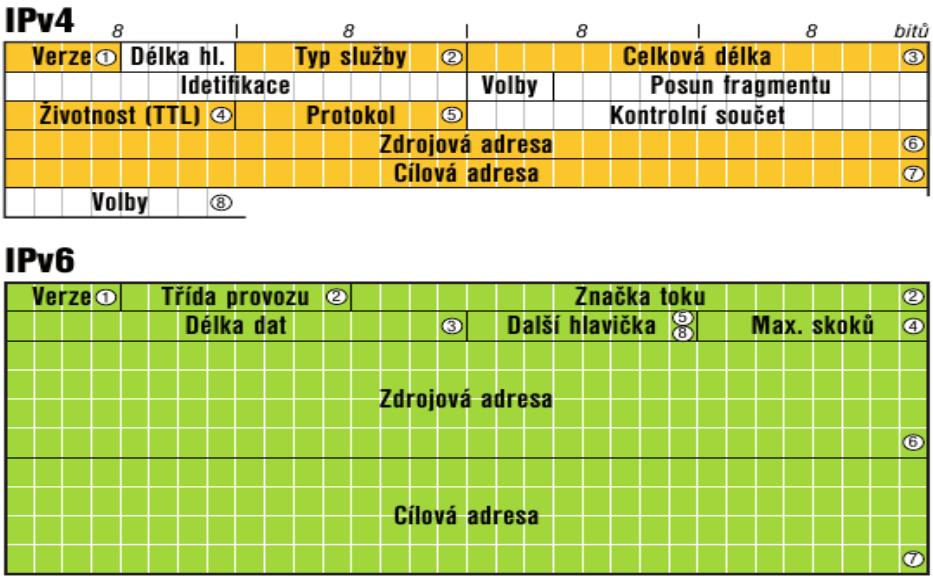
- parameter **IHL (Internet Header Length)** určujúci dĺžku samotnej hlavičky v bitoch,
- parametere určené pre fragmentáciu ako **Identification, Flags a Fragment Offset**,
- parameter **Header Checksum**, čiže kontrolný súčet,
- voliteľné nastavenia **Options**,
- parameter **Padding**.

Niektoré parametre sa v rámci základnej hlavičky modifikovali, napríklad [6]:

- parameter **Type of Service**,
- parameter určujúci finálnu dĺžku datagramu – **Total Length**,
- parameter **TTL**,
- parameter **Protocol**,

- parametre **zdrojovej a cieľovej adresy**.

Nezmenený formát si zachovala voľba Version, ktorá definuje verziu použitého IP protokolu [6]. Výsledkom minimalizácie volieb v základnej hlavičke je konštantná veľkosť hlavičky IPv6 , 40 bytov, čo predstavuje dvojnásobok hlavičky IPv4 [1].



Obr. 2 : Rozdiely medzi základnou hlavičkou IPv4 a IPv6. Čísla v jednotlivých poliach zodpovedajú zaradeniu v jednotlivých protokoloch. Oblasti bez zafarbenia neboli v základnej hlavičke IPv6 použité [1]

Očakávanou zmenou oproti protokolu IPv4 je obrovský počet IP adries, ktorý nám protokol IPv6 poskytuje. Dokonca sa uvádza, že adresný priestor je dostatočne veľký na to, aby pokryl 155 miliárd IPv4 pripojení na každý milimeter štvorcový zemského povrchu, vrátane oceánov [7]. Badateľná je zmena v typológii adries – v protokole IPv6 už nenájdeme broadcastové adresy – ich funkcionalitu preberá multicast.

Zmena protokolu IPv4 na protokol IPv6 zasiahne aj ostatné sieťové protokoly. Príkladom je **protokol ICMP**, slúžiaci na ohlasovanie chybových stavov a podávanie informácii o stave prenášaného paketu. Verzia ICMPv6 prislúchajúca protokolu IPv6 je obohatená o niekoľko nových typov správ. V porovnaní s protokolom IPv4 je v protokole IPv6 prítomnosť ICMPv6 nevyhnutnosťou. Správy ICMPv6 sú potrebné napríklad pri objavovaní susedov alebo pri ohláseniach smerovača, pri statickej autokonfigurácii alebo v prípade, ak má server viacero adries, či DNS záznamov [8].

Stavová konfigurácia DHCPv6 prebieha podobným spôsobom ako automatická konfigurácia DHCP, avšak na komunikáciu medzi klientom a serverom sa nevyužívajú broadcastové adresy. Rovnako sa nepoužíva väzba na nižšie linkové vrstvy, nakoľko každá stanica je schopná určiť si sama svoju linkovú adresu. K zjednodušeniu komunikácie prispieva aj DUID (DHCP Unique Identifier) – identifikátor, ktorý pomáha identifikovať účastníkov DHCP komunikácie [1].

K zisteniu linkovej adresy komunikačného partnera sa v protokole IPv4 využíval ARP protokol (Address Resolution Protocol). V IPv6 ho nahrádza protokol **NDP (Neighbor Discovery Protocol)**, teda tzv. objavovanie susedov.

Základné rozdiely medzi protokolom IPv4 a IPv6		
Oblasť	**IPv4 protokol**	**IPv6 protokol**
Základná hlavička paketu	20 bytov	42 bytov
	veľkosť v závislosti od volieb	fixná veľkosť
	fragmentácia aj na smerovačoch	fragmentácia len u odosielateľa
	kontrolný súčet	bez kontrolného súčtu
	voľby v základnej hlavičke	voľby v rozširujúcich hlavičkách
Adresa	32 bitov	128 bitov
	xxx.xxx.xxx.xxx	xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
	desiatková sústava	šestnástková sústava
	4 294 967 296 možných adries	$3,4 \times 10^{38}$ možných adries
	unicast, multicast, broadcast	unicast, multicast, anycast
Chybové a informačné správy	ICMP	ICMPv6 (rozšírený o niekoľko typov správ)
	Voliteľný	nevyhnutný (objavovanie susedov, automatická konfigurácia..)
Preklad IP adries na MAC adresy	protokol ARP	protokol NDP (rozpoznávanie susedov)
Multicast	protokol IGMP	protokol MLD
Automatická konfigurácia	Stavová (DHCP)	Stavová (DHCPv6) a bezstavová (pomocou RA)
DNS	záznam typu A	záznam typu AAAA
Podpora IPSec	Voliteľná	požadovaná

Tab. 1: Základné rozdiely medzi protokolom IPv4 a IPv6

1.3 Mobilita

Pod podporou **mobility v IPv6** rozumieme dostupnosť domáceho sieťového pripojenia pomocou mobilných zariadení, ako sú notebooky, mobilné telefóny a podobné zariadenia. Vychádza z predpokladu, že každé mobilné zariadenie má domácu sieť – a v nej má registrovanú svoju domácu adresu [1]. V rámci IPv4 siete by mal byť mobilný uzol identifikovateľný domácou adresou, nezávisle na mieste, kde sa práve nachádza. Ak sa však nachádza ďaleko od domovskej stanice, nie je možné komunikovať pomocou domácej adresy, pretože sa odlišuje prefix súčasnej adresy od domovskej adresy [4].

V mobilite protokolu IPv6 rozlišujeme tri základné termíny, ktoré sú kľúčové pre mobilnú komunikáciu [4]:

- **mobilný uzol (mobile node)** - sa nazýva zariadenie, ktoré je schopné zmeniť svoju lokalizáciu, pričom je do siete pripojené pomocou domáceho pripojenia,
- **domáci agent (home agent)** – je zariadenie, ktoré lokalizujeme na domácej adrese a ktoré má informácie o aktuálnej polohe mobilného zariadenia. Ak sa mobilné zariadenie nachádza mimo domácej siete, domáci agent zachytí pakety určené pre mobilné zariadenie a pošle ich pomocou tunelovania na registrovanú **CoA adresu (care of adress)**. Pod CoA rozumieme dočasnú adresu, ktorú dostane mobilné zariadenie mimo domácej lokalizácie. Keď je smerovacia optimalizácia úspešná, odpovedajúci uzol môže už priamo komunikovať s mobilným uzlom, pričom sa už nevyužíva tunelová komunikácia Domáci agent je zvyčajne implementovaný na domácom smerovači,
- **odpovedajúci uzol (correspondent node)** - je zdieľaný uzol na komunikáciu s mobilným uzlom, môže byť fixný alebo tiež mobilný. Jedná sa teda o zariadenie, ktoré inicializuje komunikáciu.

Obr. 3 : Princíp fungovania mobility

1.4 IPSec v protokole IPv6

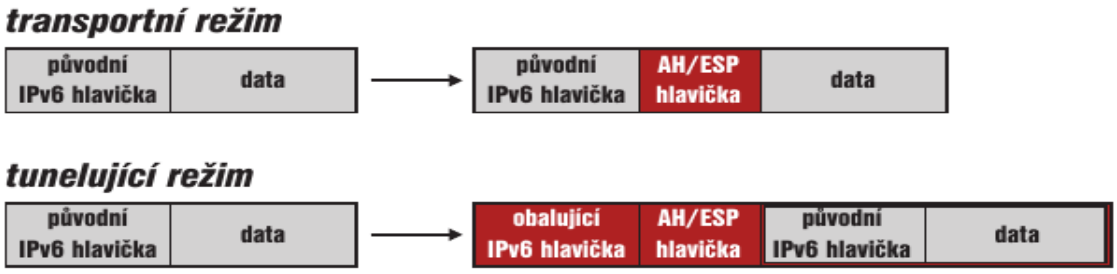
Pôvodný návrh internetového protokolu nemal žiadne prostriedky na zabezpečenie prenosu jednotlivých paketov. Až neskôr, v roku 1998 organizácia IETF (Internet Engineering Task Force) zaoberajúca sa vývojom internetových štandardov, definovala v RFC sadu protokolov zvaných ako IP Security (IPSec). **IPSec** si môžeme predstaviť ako konštrukčný rámec zahŕňajúci rôzne možnosti pre šifrovanie a overovanie paketov. Nevynucuje sa použitie konkrétneho šifrovacieho algoritmu, ale naopak, IPSec je otvorený novým možnostiam a novým nápadom na zabezpečenie bezpečnej komunikácie pomocou IPv6 protokolu [9].

IPSec nám poskytuje dve základné služby – autentifikáciu a šifrovanie. Podstatou **autentifikácie** je overiť identitu odosielateľa paketu a zabezpečiť, aby paket po ceste do cieľovej stanice nezmenil svoj obsah. Úlohou **šifrovania** je utajiť obsah prenášaných dát pred útočníkom a to tak, aby zašifrované dáta mohol odšifrovať len príjemca paketu [1]. Okrem toho zabezpečuje aj ďalšiu funkcionalitu – kontroluje duplicitu a poškodenie dát.

Realizácia IPSec v rámci IPv6 prebieha pomocou rozširujúcich hlavičiek **Authentication Header (AH)** a **Encapsulating Security Payload (ESP).** AH sa podieľa na autentifikácii dát a ich odosielateľa, zabezpečuje ochranu pred duplicitnými paketmi. ESP poskytuje okrem autentifikácie aj šifrovanie dát, pričom môže byť použitá aj samostatne alebo v kombinácii s AH. Výber jednotlivých spôsobov použitia IPSec protokolu závisí od stanovenia bezpečnostných požiadaviek pre danú sieť [10].

Bezpečnostné hlavičky IPSec je možné zavádzať v nasledujúcich režimoch [1] :

- **transportný režim** – šifrovaná je len časť paketu nasledujúca za hlavičkou ESP, zvyšná časť spolu so základnou hlavičkou tak nepodlieha bezpečnostným opatreniam IPSec (a teda je možné zistiť zdrojovú aj cieľovú adresu paketu),
- **tunelový režim** – do šifrovania je zahrnutý celý paket, pričom po zašifrovaní je ešte ošetrený novou základnou hlavičkou. Zdrojovú a cieľovú adresu preto nie je možné zistiť. Útočník sa v lepšom prípade dozvie IP adresy bezpečnostných brán, medzi ktorými je IPSec tunel vytvorený.



Obr. 4 : Porovnanie zavádzacích režimov [1]

Problematike IPSec sa bližšie venovať nebudeme, keďže ide o problematiku, ktorej rozsah niekoľkonásobne presahuje rozsah tejto práce. V našej práci sa venujeme bezpečnostným hrozbám protokolu IPv6, ktoré buď nesúvisia s IPSec, alebo s nimi súvisia len okrajovo.

1.5 Interoperabilita medzi protokolmi

Je zrejmé, že prechod medzi protokolom IPv4 a IPv6 nebude krátkodobá záležitosť a pravdepodobne budú určitú dobu oba protokoly fungovať súbežne. Pod **súbežným fungovaním** rozumieme prostredie, kde izolované IPv6 domény vzájomne komunikujú cez sieť postavenú na IPv4 protokole. Na zabezpečenie **interoperability protokolov IPv4 a IPv6** bolo vyvinutých niekoľko druhov mechanizmov rozdelených do troch skupín [9] :

- tunelové mechanizmy,
- prekladové mechanizmy,
- dual-stack mechanizmy.

Tunelové mechanizmy pomáhajú komunikovať izolovaným IPv6 doménam cez sieť fungujúcu na protokole IPv4. **Prekladové mechanizmy** umožňujú prekladať komunikáciu medzi IPv4 a IPv6 uzlami. **Dual-stack mechanizmy** používajú na komunikáciu IPv4 a IPv6 zásobníky, v závislosti od typu komunikácie [9].

Pod **tunelovaním** rozumieme mechanizmus, kedy jeden protokol je zapuzdrený vo vnútri iného protokolu, aby mohol prejsť sieťou, cez ktorú by v bežnej situácii neprešiel (ako napríklad IPv6 paket cez sieť založenú na IPv4 protokole). Proces prebehne jednoducho, pridaním hlavičky IPv4 pred paket IPv6. Pomocou tejto hlavičky potom paket prejde IPv4 sieťou. Ak sa však dostane do siete podporujúcej protokol IPv6, hlavička IPv4 sa zahodí [7].

Tunelové mechanizmy sa delia na dva druhy – tunely, ktorých konečný bod je manuálne nakonfigurovaný a tunely, kde konečný bod vzdialeného tunela je detekovaný automaticky. **Manuálne konfigurované tunely** musia byť nakonfigurované na oboch stranách, no napriek tomu sú jednoduchšie. **Tunely s automatickou detekciou** konečného bodu sú zložitejšie, avšak často preferovanejšie. Patrí tu **Automatic Tunneling** – prvý zo spôsobov automatického tunelovania, v ktorom jedna IPv6 adresa zodpovedá práve jednej IPv4 adrese. V súčasnosti sa považuje za zastaraný. Ďalej tu patria tunely **6over4** a **ISATAP**, ktoré riešia tunelovanie v rámci jednej organizácie alebo vnútornej siete, 6over4 pritom využíva IPv4 sieť s podporou multicast, ISATAP prezentuje IPv4 sieť ako NBMA (Non-broadcast Multiple Access) sieť, pričom kóduje IPv4 adresy ako časť identifikátora rozhrania IPv6. **Tunel TEREDO** umožňuje zavádzať tunelovacie

mechanizmy v rámci NAT (Network Address Translation), tunel **6to4** funguje podobne ako ISATAP, umožňuje vytvárať tunely medzi oboma stranami [7].

SIIT (Stateless IP/ICMP Translation) protokol zabezpečuje konektivitu medzi zariadeniami, ktoré akceptujú iba IPv6 pakety a zariadeniami, ktoré akceptujú iba IPv4 pakety, a to pomocou prekladového mechanizmu. Neuchováva však informácie pri každej relácii ani nešpecifikuje, ako dlho je adresa priradená k hostiteľovi. V rámci NAT siete funguje mechanizmus NAT-PT (Network Address Translation – Port Translation), ktorý umožňuje mapovacie tabuľky, v ktorých sa IPv6 adresy viažu na adresy IPv4 [4].

2 Bezpečnostné hrozby spojené s implementáciou protokolu IPv6

Táto kapitola predstavuje jadro celej práce a zaoberá sa bezpečnostnými hrozbami, ktoré ohrozujú IPv6 protokol. V úvode kapitoly rozoberáme niekoľko pojmov z informačnej bezpečnosti. Následne je kapitola rozdelená do niekoľkých podkapitol vzhľadom na typy bezpečnostných hrozieb. Súčasťou každej podkapitoly je zhrnutie bezpečnostných opatrení a odporúčania pre správcov využívajúcich protokol IPv6.

2.1 Základné pojmy z oblasti informačnej bezpečnosti

Pred rozoberaním jednotlivých hrozieb a opatrení v rámci protokolu IPv6 je nevyhnutné si vyjasniť niekoľko základných pojmov z oblasti informačnej bezpečnosti, z ktorých budeme vychádzať v našej práci.

Za **bezpečnostnú hrozbu (threat)** považujeme možnosť potencionálneho porušenia bezpečnostných pravidiel, ktorá existuje v rámci zraniteľnosti systému [11]. Bezpečnostná hrozba vyvoláva **bezpečnostný incident**, ktorý chápeme ako porušenie bezpečnostných požiadaviek buď priamym pôsobením na údaje, zasiahnutím prostredia, alebo informačno-komunikačnej technológie, ktorá údaje spracováva. Na to, aby bezpečnostná hrozba nastala, musí existovať v zavedenom systéme **zraniteľnosť (vulnerability)** [25].

Zraniteľnosť informačného systému sa prejavuje ako chyba v systémovom návrhu, implementácii, v prevádzke alebo v riadení, ktorá môže byť využitá k porušovaniu pravidiel bezpečnostnej politiky [11].

Pod **útokom** rozumieme zámerný pokus o obídenie bezpečnostných bariér zabezpečeného systému za účelom zmeny systémových zdrojov a ovplyvnenia ich prevádzky v rámci aktívneho útoku alebo využitia informácii zo systému bez zneužitia systémových zdrojov v rámci pasívneho útoku. Útok môžeme rozdeliť na vnútorný a vonkajší, v závislosti od prístupnosti zdrojov pre útočníka [11].

Negatívne dôsledky naplnenia bezpečnostnej hrozby predstavuje **miera dopadu**, ktorú je možné vyjadriť kvantitatívne (množstvom prostriedkov, ktoré je potrebné vynaložiť na odstránenie následkov bezpečnostného incidentu) alebo kvalitatívne pri nemerateľnom dopade [25].

Pod **bezpečnosťou informačného systému** rozumieme prijatie bezpečnostných opatrení, ktoré zabraňujú útokom na systémové zdroje alebo informácie. Súhrn všetkých pravidiel a princípov bezpečnosti systému predstavuje jeho **bezpečnostná architektúra** [11].

2.2 Rozdelenie bezpečnostných hrozieb

Existuje viacero rozdelení bezpečnostných hrozieb protokolu IPv6. V odbornom článku **IPV4/IPV6 security and threat comparisons** nachádzame rozdelenie bezpečnostných hrozieb v závislosti od ich spojenia s predchádzajúcim protokolom.

K bezpečnostným hrozbám **spoločným pre protokol IPv4 aj IPv6** radíme [12]:

- útoky zamerané na zachytávanie dát prenášaných v sieti (sniffing),
- útoky vedené na aplikačnú vrstvu (vírusy alebo trójske kone),
- útoky spôsobené neautorizovanými zariadeniami v sieti,
- Man-in-the-Middle útoky,
- DoS útoky.

Rovnako dôležité je vymedziť **útoky špecifické iba pre IPv6 protokol** [12] :

- prieskumné útoky v sieti IPv6,
- bezpečnostné hrozby spojené so smerovacou hlavičkou IPv6,
- bezpečnostné hrozby spojené s fragmentáciou IPv6 paketu,
- bezpečnostné hrozby spojené s ICMPv6 a multicastom,
- SEND a CGA mechanizmy,
- bezpečnostné problémy týkajúce sa prechodových mechanizmov.

Pre účely tejto záverečnej práce sme rozdelili bezpečnostné hrozby protokolu IPv6 nasledovne:

- Bezpečnostné hrozby spojené s prieskumnými útokmi
- Bezpečnostné hrozby spojené so smerovacou hlavičkou
- Bezpečnostné hrozby spojené s fragmentáciou
- Bezpečnostné hrozby spojené s protokolom ICMPv6
- Bezpečnostné hrozby spojené s protokolom Neighbor Discovery Protocol (NDP)

2.3 Bezpečnostné hrozby spojené s prieskumnými útokmi

2.3.1 Prieskumný útok

Prvotnú fázu akéhokoľvek útoku predstavuje takzvaný prieskumný útok. Pod **prieskumným útokom** (skenovaním) rozumieme neautorizovaný spôsob zhromažďovania informácii o

zariadeniach v sieti. Takto získané informácie umožnia útočníkovi spoznať zraniteľnosť siete a následne mu umožnia stanoviť najjednoduchší spôsob ako uskutočniť úspešný útok na cieľ. Skenovanie sa nepovažuje za útok sám o sebe, je však nevyhnutnou súčasťou útoku, preto je **defenzíva voči nemu súčasťou hĺbkovej ochrany systému** [9].

2.3.2 Bezpečnostné hrozby a navrhované protiopatrenia

Prieskumné útoky sú už dobre známe z protokolu IPv4. V rámci protokolu IPv6 nachádzame útoky **identické** s protokolom IPv4 ale aj útoky **špecifické** len pre protokol IPv6. Mnoho nástrojov na skenovanie IPv4 sietí prešlo úpravami, aby sa ich využitie rozrástlo aj o IPv6 siete (ping6, traceroute6, nmap6). Niektoré prieskumné útoky známe z IPv4 však už nie je možné použiť na protokol IPv6. Príkladom je priame skenovanie siete, ktoré by v prípade protokolu IPv6 trvalo nekonečne dlhú dobu.

V našej práci sme rozdelili prieskumné útoky do nasledujúcich kategórii:

1. Základné prieskumné útoky
2. Prieskumné útoky zamerané na skenovanie siete
3. Prieskumné útoky špecifické pre prechodové mechanizmy

Uvedené rozdelenie sme vykonali na základe zamerania jednotlivých informácii, ktoré môže útočník uskutočnením prieskumných útokov získať. Základné útoky poskytujú informácie o konkrétnom zariadení, zatiaľ čo útoky zamerané na skenovanie poskytujú údaje o celkovej štruktúre počítačovej siete. Poslednú kategóriu tvoria útoky vedené na zariadenia využívajúce prechodové mechanizmy.

K **základným prieskumným útokom** zaraďujeme prehľadávanie registrov (whois), kontrolu DNS záznamov, sniffovanie a zbieranie informácii z verejne prístupných zdrojov (ako sú napríklad webové vyhľadávače). Radíme tu aj prieskumné útoky, ktoré využívajú predchádzajúce, už uskutočnené útoky na sieť [9].

Je dôležité si uvedomiť, že prieskumným útokom založeným na zbieraní informácii je ťažké zabrániť. Preto treba zvážiť každú informáciu, ktorú o svojej sieti zverejňujeme. Ako obranu voči základným prieskumným útokom sa odporúča dbať na zabezpečenie aplikačnej vrstvy. V prípade, že používame webový server, je vhodné použiť odporúčané bezpečnostné nastavenia (vypnutie výpisu adresárov, vypnutie zobrazovania informácii o webovom serveri, filtrovanie zlých URL požiadaviek, zavedenie reštrikcií a prípadné použitie niektorých bezpečnostných modulov ako sú rewrite modul, mod-evasive alebo mod-security). V tabuľke č. 2 uvádzame niektoré zo

spomenutých bezpečnostných nastavení. Ich platnosť sa vzťahuje nielen na protokol IPv4, ale aj na protokol IPv6.

Odporúčané bezpečnostné nastavenia pre webový server Apache 2	
Bezpečnostné opatrenie	Konfigurácia alebo vykonaný príkaz
Vypnutie výpisu adresárovej štruktúry	<Directory /var/www/> Options **-Indexes** FollowSymLinks MultiViews AllowOverride None Order allow,deny allow from all </Directory> <Directory /> Options **None** Order allow,deny Allow from all </Directory>
Vypnutie zobrazovania informácii o webovom serveri	**ServerToken Prod** (vypnutie bannera) **ServerSignature Off** (vypnutie riadku s číslom verzie a ServerName)
Filtrovanie zlých URL požiadaviek	apt-get install **libapache-mod-security mod-security-common**
Predchádzanie HTTP-DoS a DDoS, Brute Force	Apt-get install **libapache2-mod-evasive**
Zavedenie reštrikcií k špecifickej adrese alebo sieti	<Directory /yourwebsite> Options None AllowOverride None Order deny,allow Deny from all Allow from **yournetwork** </Directory>

Tab. 2: Bezpečnostné nastavenia webového servera Apache 2

V protokole IPv4 je pomerne jednoduché identifikovať zariadenia v sieti pomocou skenovania. Veľkosť adresného priestoru je totiž značne menšia ako v protokole IPv6. Z toho dôvodu **prieskumný útok založený na skenovaní** siete je záležitosťou niekoľkých minút. V protokole IPv6 by takýto útok trval podstatne dlhšie, preto je útočník nútený využiť niekoľko iných metód, ktoré protokol IPv6 ponúka. Prvou možnosťou je spoliehať sa na **sekvenčnú adresáciu** uzlov v sieti. Ak sú adresy prideľované dynamicky s použitím MAC adresy zariadenia, stáva sa skenovanie

náročnejším, avšak nie nezdolateľným. Stačí, aby útočník poznal identifikačné čísla výrobcov sieťových rozhraní a oblasť kombinácii sa zníži o 24 bitov [21].

Skupinové (**multicast**) adresy majú v IPv6 dôležitú úlohu, či už pri objavovaní susedov alebo detekcii duplicitných adries. Môžu byť však využité na prieskumný útok zameraný na skenovanie zariadení patriacich do tej istej skupiny. Stačí, ak útočník odošle ICMPv6 správu na lokálnu linkovú multicast adresu (napríklad ff02::1) a ako odpoveď dostane ICMPv6 správy od všetkých účastníkov skupinovej komunikácie obsahujúce ich lokálne linkové adresy [22].

Intenzívny proces skenovania môže vplývať na sieť dvomi spôsobmi – buď dôjde k obsadeniu značnej šírky pásma, alebo sa vyčerpajú systémové prostriedky na smerovacích zariadeniach, ktoré vyhľadávajú MAC adresy. Na obmedzenie prieskumných útokov založených na skenovaní je preto potrebné efektívne **zabezpečiť adresáciu siete** - uzly v sieti by nemali byť identifikovateľné sekvenčne, a zároveň by ich identifikátor nemal byť podobný ako identifikátor celej podsiete [9].

Úspešnou prevenciou voči skenovaniu je aj používanie **CGA** (Cryptographically Generated Addresses) pre adresovanie uzlov v rámci siete. Hlavnou úlohou CGA adries je totiž vytvorenie jednoznačného identifikátora rozhrania za pomoci asymetrických kryptografických metód. Používa sa spracovanie verejného kľúča spolu s ďalšími položkami pomocou SHA-1 hašovacej funkcie. Prvých 64 bitov výsledku sa použije ako identifikátor rozhrania. CGA adresy majú hlavné využitie aj pri SEND (Secure Neighbor Discovery) [1].

Alternatívnou možnosťou je použitie tzv. **pull modelu**, ktorý sa používa pri detekcii duplicitných adries - DAD (Duplicate Address Detection). Pull model zabezpečuje prideľovanie adries pomocou **hašovacej funkcie (hash function)**. Ide o jednocestnú funkciu pre prevod vstupného reťazca dát na krátky výstupný reťazec. Hašovacia funkcia vytvára pre rovnaký vstup stále rovnaký výstup. Ak centrálny server kontrolujúci zariadenia v sieti vyhodnotí, že hashovacia funkcia pre dve rôzne adresy je identická, jedná sa o duplicitu adries. Uvedený mechanizmus generovania adries umožňuje univerzálnejšie usporiadanie siete a garantuje jedinečnosť vygenerovaných adries [26].

Riešenie uvedenej bezpečnostnej hrozby skenovania skupinovej siete pomocou multicastu predstavuje **kontrola ICMPv6 správ** pomocou firewallu. Odporúčaným nastavením v rámci firewallu je pridanie pravidla, ktoré v časti vstupu na stroj, resp. v časti pre preposielanie paketov (napríklad pri sieťovom smerovači) bude kontrolovať správy protokolu ICMPv6 typu echo-request a pri doručení takýchto paketov ich firewall zahodí. Pravidlo by v prípade firewallu iptables mohlo vyzerať nasledovne : *ip6tables –A INPUT –p icmpv6 - - icmpv6-type echo-request –j DROP*. V prípade CISCO zariadení by sme mohli použiť rozšírený typ prístupových zoznamov (možnosť

zadefinovať aj typ protokolu, konkrétne porty a pod.) s nasledujúcim pravidlom: *access-list 101 deny icmp any any echo*.

Možným riešením je aj úplný **zákaz ICMPv6** správ cielených na adresu multicastu, avšak v danom prípade je potrebné použiť namiesto bezstavovej automatickej konfigurácie, konfiguráciu pomocou DHCPv6, prípadne manuálnu konfiguráciu [22].

V závere tejto podkapitoly ešte spomenieme kategóriu prieskumných útokov špecifických pre IPv6 protokol. Prechodové mechanizmy, či už Teredo, ISATAP alebo 6to4 generujú používateľom ľahko identifikovateľné IP adresy. Napríklad Teredo používa na generovanie IPv6 adries IPv4 adresu a číslo UDP portu, cez ktorý sa IPv6 paket zabalený do IPv4 prenáša [23].

Prehľad bezpečnostných hrozieb a ich riešení			
Typ bezpečnostnej hrozby	Bezpečnostná hrozba	Navrhované riešenie	Použitá implementácia
Bezpečnostné hrozby spojené s prieskumnými útokmi	**základné útoky** (prehľadávanie registrov, DNS záznamov, hľadanie informácii na Internete, predchádzajúce útoky)	Dôkladná selekcia zverejnených informácii, ochrana aplikačnej vrstvy, dôkladné zabezpečenie webového servera	Na cieľovom zariadení nie je nainštalovaný webserver, porty potrebné pre web komunikáciu sú zakázané, zariadenie je adresované mimo študentskej siete
	skenovanie siete (pri sekvenčnom usporiadaní adries, pri dynamickom generovaní pomocou MAC, skenovanie multicastu)	zavedenie nesekvenčnej adresácie uzlov v sieti, použitie kryptograficky generovaných adries	ip6tables -A INPUT -p icmpv6 - - icmpv6-type echo-request -j DROP ip6tables -A FORWARD -p icmpv6 - - icmpv6-type echo-request -j DROP
	Prechodové mechanizmy (napríklad TEREDO) generujú ľahko rozpoznateľné IPv6 adresy	Obmedzenie používania prechodových mechanizmov	Testovacia sieť je založená na priamej IPv6 komunikácii, nepoužíva prechodové mechanizmy

Tab. 3 : Bezpečnostné hrozby spojené s prieskumnými útokmi

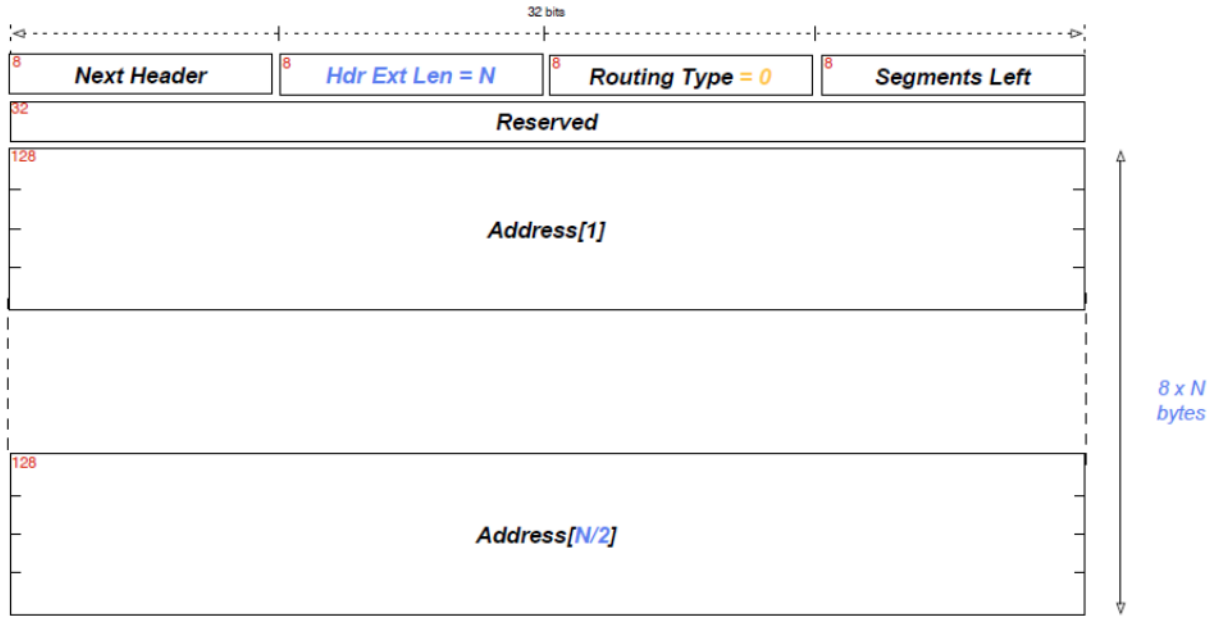
2.4 Bezpečnostné hrozby spojené so smerovacou hlavičkou IPv6

2.4.1 Smerovacia hlavička paketu IPv6

Každý paket je štandardne smerovaný na svoju cieľovú adresu. **Smerovacia hlavička (Routing Header, RH)** umožňuje ovplyvniť cestu paketu k cieľovej adrese a rozšíriť ju o niekoľko destinácii. Štruktúra RH hlavičky je popísaná v RFC 2460 [6], pričom parameter **typ smerovania** definuje typ smerovacej hlavičky. Okrem experimentálnych hlavičiek existujú dva základné typy smerovacej hlavičky. Prvým je **RH0 (Routing Header type 0)** hlavička, ktorá predstavuje rozširujúcu

hlavičku IPv6 typu 43 (smerovacia hlavička) variantu 0 a plní podobnú funkciu ako smerovanie v protokole IPv4. Druhým typom je **RH2 (Routing Header type 2)** hlavička, pri ktorej sú definované nastavenia smerovania pre mobilitu.

Hlavička RH0 definuje uzly siete, ktorými bude paket prechádzať. Ako prvá sa v hlavičke uvedie adresa prvého z prechádzajúcich uzlov, potom nasledujú ostatné adresy, až ako posledná je zapísaná cieľová adresa paketu. V položke „zostávajúce segmenty“ sa zapíše zostávajúci počet adries, ak je položka nulová. To znamená, že paket dorazil do cieľa. Smerovacia hlavička typu 0 bola zavedená hlavne z dôvodu testovania dosiahnuteľnosti spojení medzi jednotlivými adresami. Jej funkcionalita je v overovaní funkčnosti spojenia. Tento typ smerovacej hlavičky však môže poslúžiť k útokom zahltenia prenosových trás. Preto sa v súčasnosti doporučuje filtrovať pakety so smerovacou hlavičkou typu 0. V tomto prípade, ak cieľový IPv6 uzol obdrží hlavičku smerovania typu 0, je povinný ju ignorovať, pokiaľ je počet zostávajúcich segmentov nulový. V prípade, ak je nenulový, musí sa paket zahodiť a ohlásiť ako chybný.



Obr. 5 : Rozširujúca hlavička smerovania typu 0 [27]

Typ RH2 bol navrhnutý pre zabezpečenie mobility. Ide o zovšeobecnenú verziu typu 0, keď má paket ešte navyše dočasnú adresu, ktorá sa mení podľa siete, v ktorej sa paket nachádza. Koncovou adresou je pevná adresa mobilného uzlu. Najskôr sa ale paket musí dopraviť na dočasnú adresu. Smerovacia hlavička tohto typu umožňuje uložiť len jednu adresu, a to domácu adresu mobilného uzlu, ktorému je paket určený, čo výrazne obmedzuje jej zneužiteľnosť [1].

2.4.2 Bezpečnostné hrozby a navrhované protiopatrenia

Použitie smerovacích hlavičiek protokolu IPv6 so sebou prináša nasledujúce bezpečnostné hrozby :

1. bezpečnostné hrozby spojené so zneužitím smerovacej hlavičky RH0
2. bezpečnostné hrozby spojené s filtrovaním smerovacích hlavičiek

Zneužitie smerovacej hlavičky typu 0 spočíva v tom, že útočník odošle RH0 paket na hostiteľa, ktorý spracováva hlavičky paketov a má dôveryhodný vzťah s cieľovým zariadením. Hostiteľ tento paket spracuje a na základe smerovacej hlavičky ho odovzdá ďalej. V závere cieľová stanica obdrží packet a vygeneruje odpoveď útočníkovi (v prípade že útočník použil ako zdrojovú adresu reálnu IPv6 adresu). Pomocou RH0 útoku možno rapídne zvýšiť sieťovú prevádzku v počítačovej sieti. Výsledkom je aj **DoS (Denial of Service) útok**, ktorý vzniká presmerovaním oneskorených slučiek na obeť útoku v rovnakom čase.

Voči tomuto útoku existuje niekoľko bezpečnostných opatrení. Prvé je možnosť využitia firewallu **(**napríklad Cisco ASA verzie 8.0). **Firewall** v štandardnom režime blokuje pakety RH0. Firmware verzie IOS 12.04 a vyššie verzie zabezpečujú ochranu pred RH0 pomocou príkazu, ktorý zabráni smerovaču prepravu RH0 paketov, ktoré majú svoje IP rozhranie adries v reťazcoch spolu so smerovacou hlavičkou. Týmto príkazom je *no ipv6 source-route*. Pre úplne blokovanie RH0 útokov je potrebné nakonfigurovať **ACL (Access Control List)** pre všetky rozhrania sieťového smerovača (globálne, fyzické, loopback, a pod.) tak, aby RH0 paket neprešiel sieťovým smerovačom, hoci obsahuje vo svojom záhlaví IP adresu smerovača. Blokované by mali byť RH0 pakety posielané priamo do smerovača, ale aj tie, ktoré cez smerovač len prechádzajú. Je potrebné aplikovať ACL na všetkých fyzických rozhraniach smerovača.

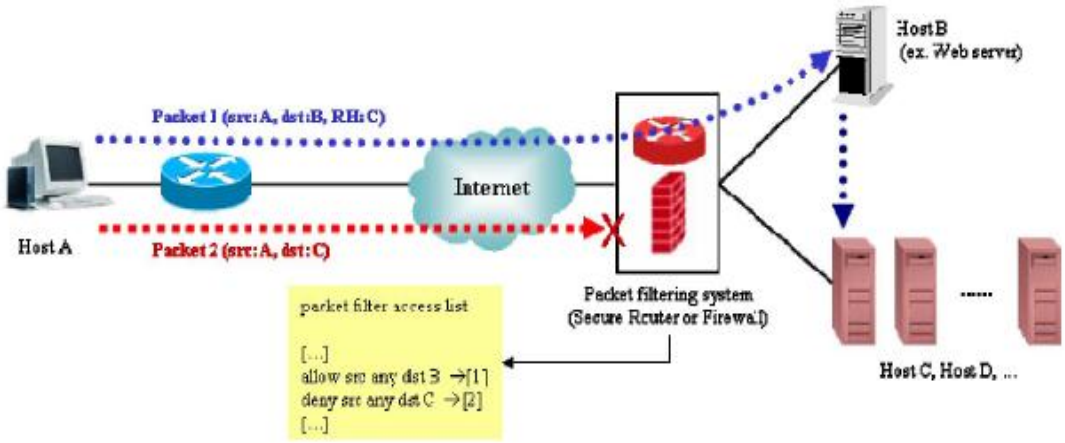
Ďalšie riešenie sa ponúka v dôkladnejšom skúmaní paketov v zariadeniach po ceste. Smerovače a firewally sa zamerajú na packety, ktoré majú vo svojej hlavičke opakovanie tých istých adries po ceste, čo predznamenáva RH0 útok. Mnohé operačné systémy majú implementované filtrovanie RH0 paketov [9].

OS	Host	Router	*Deactivable?*
Linux 2.6	dropped	processed	**no**
FreeBSD 6.2	**processed**	processed	**no**
NetBSD 3.1	**processed**	processed	**no**
OpenBSD 4.0	**processed**	processed	**no**
Cisco IOS	n/a	processed	yes
Cisco PIX	n/a	dropped	n/a
Juniper RTR	n/a	processed	**no**
Netscreen FW	n/a	dropped	n/a
Windows XP SP2	dropped	n/a	n/a
Windows Vista	dropped	n/a	n/a

Remark #1: by "Deactivable" we do not consider firewalling, only sysctl or equivalent means
Remark #2: red indicates a problem, bold and red a big one

Obr. 6 : Zachytávanie RH0 paketov v sieťových smerovačoch [27]

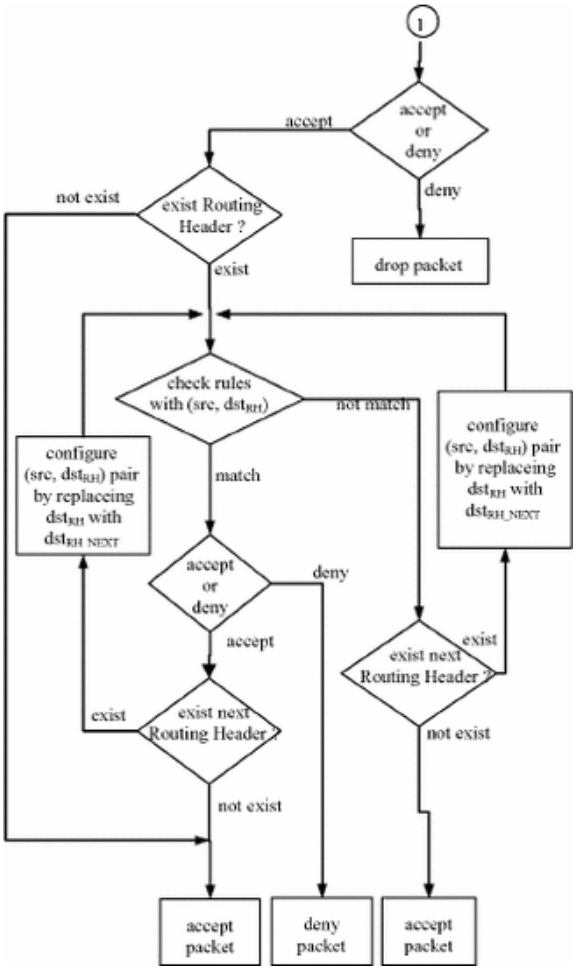
Útok pomocou obchádzania filtrovacích pravidiel je znázornený na obrázku č. 7. K dispozícii sú tri druhy zariadení. Zariadenie **Host B** je poskytovateľ verejných služieb, preto je otvorený pre prístup mimo siete. Zariadenie **Host C** naopak využíva filtráciu packetov, a teda nie je dosiahnuteľný mimo príslušnú sieť, nakoľko poskytuje súkromné služby. Útočník , **Host A**, môže napadnúť sieť pomocou packetu 1 alebo packetu 2, pričom packet 2 smerujúci k zariadeniu C je zachytený podľa filtrovacích pravidiel. Avšak packet 1 smerujúci k zariadeniu B príde do cieľa a odtiaľ sa pomocou smerovacej hlavičky dostane k zariadeniu C, čo predstavuje obchádzanie filtrovacích pravidiel [13].



Obr. 7 : Útok pomocou obchádzania filtrovacích pravidiel [13]

Bezpečnostným opatrením voči tomuto útoku je **rozšírenie filtrovacích pravidiel** a modifikácia používateľského rozhrania. Ide o menej efektívne riešenia. Efektívnejšie je využitie **DPA algoritmu (Detour Protection Algorithm)**, ktorého cieľom je rozšíriť kontrolnú oblasť filtrovacích pravidiel o adresy zahrnuté v smerovacej hlavičke. Aby bol paket regulárne filtrovaný, je nutné skontrolovať aj ďalšie parametre ako zdrojovú a cieľovú adresu, zdrojový a cieľový port a použitý protokol. Preto hĺbková analýza paketu môže pozostávať z viacerých filtrovacích pravidiel.

Názorný príklad použitia algoritmu je uvedený na obrázku č. 8. V uvedenej schéme prijatý paket prechádza najskôr základnými pravidlami firewallu. Potom nastupuje kontrola pomocou DPA algoritmu. Ak sa medzi rozširujúcimi hlavičkami nachádza smerovacia hlavička typu 0, aplikujú sa filtrovacie pravidlá kontrolujúce zdrojovú a cieľovú adresu. Ak je zdrojová aj cieľová adresa vyhodnotená ako bezpečná, paket znova prechádza základnými filtrovacími pravidlami a ak sa v smerovacej hlavičke stále nachádzajú adresy pre smerovanie, hodnoty zdrojovej a cieľovej adresy sa aktualizujú na hodnoty adries uvedených v smerovacej hlavičke. Tento proces sa opakuje až pokým smerovacia hlavička neobsahuje smerovacie adresy.



Obr. 8 : Schéma DPA algoritmu s úvodným pravidlom vo firewalle [13]

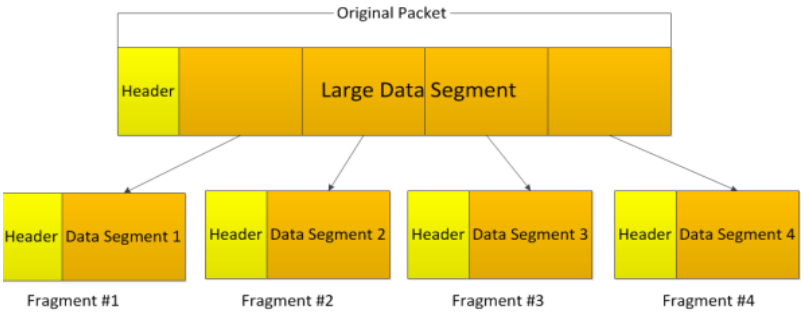
Prehľad bezpečnostných hrozieb a ich riešení			
Typ bezpečnostnej hrozby	Bezpečnostná hrozba	Navrhované riešenie	Použitá implementácia
Bezpečnostné hrozby spojené so smerovacou hlavičkou	Dos útok	Filtrovacie pravidlá	-A INPUT -m rt --rt-type 0 -j DROP
			-A FORWARD -m rt --rt-type 0 -j DROP
			-A OUTPUT -m rt --rt-type 0 -j DROP
		Použitie ACL pre všetky rozhrania sieťového smerovača	Router(config-ipv6-acl)#deny ipv6 any any routing-type 0
		Linux verzie 2.6.20.9 a vyššie	Linux bublinka-ipv6 3.2.0-4-686-pae
	Obchádzanie filtrovacích pravidiel	DPA algoritmus	

Tab. 4 : Bezpečnostné hrozby spojené so smerovacou hlavičkou

2.5 Bezpečnostné hrozby spojené s fragmentáciou IPv6 paketu

2.5.1 Fragmentačná hlavička IPv6 paketu

Fragmentačná hlavička sa podieľa na fragmentácii paketov, ktoré svojou veľkosťou presahujú hodnotu MTU. Odporúča sa, aby každé zariadenie po ceste malo MTU vo veľkosti 1280 oktetov a vyššie [6]. V rámci IPv4 mohol pakety fragmentovať každý smerovač po ceste, avšak v protokole IPv6 úlohu fragmentovania preberá odosielateľ paketu. Všetky hlavičky, ktoré predchádzajú fragmentačnej hlavičke, sú považované za nefragmentovateľnú časť dát. Fragmentovať sa môžu teda len dáta, ktoré nasledujú za fragmentačnou hlavičkou. Fragmentovateľná časť sa potom rozdelí fragmenty vo veľkosti násobkov ôsmych bajtov, ktoré sa následne odošlú do cieľovej adresy ako samostatné pakety. Každý z týchto fragmentov má jedinečný identifikátor a poradie v rámci pôvodného paketu. Na základe údajov z fragmentačnej hlavičky si príjemca v závere poskladá pôvodný paket [1].



Obr. 9 : Paket s veľkosťou presahujúcou MTU sa rozdelí na menšie fragmenty [28]

2.5.2 Bezpečnostné hrozby a navrhované protiopatrenia

Na bezpečnostné hrozby vyplývajúce z fragmentácie mysleli už autori RFC 2460 (Internet Protocol, Version 6 Specification), v ktorom navrhli možné riešenia neprimeranej fragmentácie [6] :

- ak sa do 60 sekúnd od doručenia prvého fragmentu nedostavia do cieľa aj ostatné fragmenty, paket sa neposkladá do pôvodnej verzie a doručené pakety sa zahodia. V prípade, že bol doručený prvý fragment (Offset určujúci poradie fragmentu je rovný nule), na adresu odosielateľa sa odošle ICMPv6 správa typu 3 s kódom 1 (Fragment Reassembly Time Exceeded),
- v prípade, že dĺžka doručeného fragmentu nie je násobkom ôsmych oktetov a fragmentačná hlavička obsahuje príznak M s hodnotou 1 (t.j. uvedený fragment nie je posledným fragmentom), fragment musí byť zahodený a odosielateľovi sa pošle ICMPv6 správa typu Parameter Problem s kódom 0,
- ak dĺžka a posun fragmentu nadobúdajú hodnoty také, že Payload Length presahuje veľkosť 65 535 oktetov, fragment bude zahodený a odosielateľovi sa pošle ICMPv6 správa typu Parameter Problém s kódom 0.

Počas fragmentácie však môže nastať niekoľko ďalších situácií, ktoré môžu byť vyhodnotené ako potencionálne bezpečnostné hrozby.

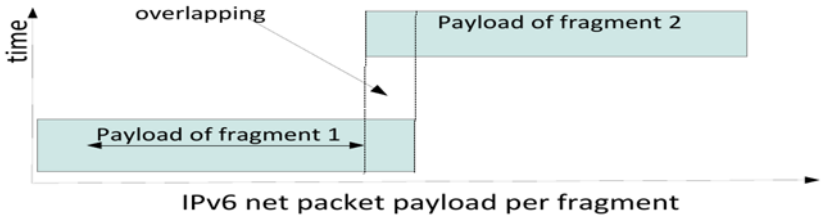
Jednou z nich je situácia, keď útočník odosiela **paket, ktorý má minimálnu veľkosť**. Ako už bolo spomenuté, protokol IPv6 požaduje, aby veľkosť paketov bola 1280 oktetov a viac. Ak je veľkosť MTU nevyhovujúca, úlohu fragmentácie a opätovného poskladania paketu preberá vrstva pred IPv6. Nakoľko však zavádzanie protokolu IPv6 prebieha počas dlhého časového horizontu, musíme počítať so skutočnosťou, že paket sa pomocou prechodových mechanizmov (napr. tunel ipv6to4) dostane do siete s MTU nižšou ako je určené v prípade IPv6 protokolu. Pre daný prípad je potom menšia veľkosť IPv6 paketu výhodou. Otázkou je, čo nastane, ak veľkosť paketu nezodpovedá požiadavkám IPv6 siete. Vytvoríme paket, ktorého veľkosť je 56 bytov. Pozostáva zo základnej hlavičky (40 bytov), fragmentačnej hlavičky (8 bytov) a z hlavičky prislúchajúceho protokolu, napríklad ICMPv6 (taktiež vo veľkosti 8 bytov). Testovaním sa zistilo, že takto vytvorené pakety bez problémov prejdú bezpečnostnými pravidlami a odosielateľ naspäť dostane odpoveď vo forme ICMPv6 Echo Reply správy [14]. Najvhodnejším riešením tejto bezpečnostnej hrozby je použitie filtrovacích pravidiel zameraných na kontrolu dĺžky fragmentu. Príkladom možnej implementácie je filtrovacie pravidlo *ip6tables -A INPUT -m frag --fraglen number –j DROP* určené pre smerovače alebo *ip6tables -A FORWARD -m frag --fraglen number –j DROP* určené pre konkrétny server. Parameter number reprezentuje za nami zvolenú číselnú dĺžku fragmentu.

Na podobnom princípe funguje aj takzvaný **atomický fragment**, ktorý je prvým a súčasne aj posledným prvkom fragmentácie (hodnota posunu aj príznak M identifikujúci posledný prichádzajúci fragment nadobúdajú hodnotu 0). Podľa výsledkov uvedených v štúdii [16] všetky testované operačné systémy akceptovali takto vytvorený fragment bez ohľadu na to, či sa jedná o komunikáciu priamo v IPv6 sieti alebo v sieti využívajúcej prechodové mechanizmy. Skutočné riziko atomických fragmentov si vysvetlíme pri rozoberaní nasledujúcej bezpečnostnej hrozby. Riešením uvedenej bezpečnostnej hrozby je použitie filtrovacieho pravidla : *ip6tables -A INPUT -m frag - - fragfirst - - fraglast –j DROP*. Dané filtrovacie pravidlo otestuje, či prijatý fragment je súčasne prvým aj posledným fragmentom. Ak je filtrovacie pravidlo splnené, daný fragment sa zahodí.

Voľba identifikácie fragmentu oproti predchádzajúcemu protokolu IPv4 zdvojnásobila svoju veľkosť na 32 bitov. Hlavnou úlohou identifikácie je v prvom rade jednoznačne odlíšiť fragmenty pochádzajúce z rôznych paketov, pričom sa do úvahy berie životnosť paketu, tranzitný čas od zdroja k cieľovej stanici a aj približný čas opätovného zloženia paketu v cieľovej stanici. Ako sa uvádza v dokumente RFC 6145 s názvom IP/ICMP Translation Algorithm [15], ak paket prechádza sieťou 6to4, posledných 16 bitov tejto voľby tvorí identifikáciu fragmentu pre IPv4 sieť. Autor článku [16] poukazuje na skutočnosť, že väčšina operačných systémov má **problém generovať identifikačné čísla fragmentu** – v Linuxových jadrách do verzie 3.1 sa identifikácia fragmentu negeneruje individuálne pre každú destináciu – ak útočník dokáže predpovedať tieto identifikačné čísla, môže spôsobiť DoS útok vytvorením obdobných paketov. Riešenie tejto bezpečnostnej hrozby prišlo v zmene implementácie identifikátorov fragmentov nového linuxového jadra – prvá hodnota identifikátora sa vygeneruje náhodne, potom sa inkrementuje zvlášť pre každú odlišnú destináciu.

Nová bezpečnostná hrozba vzniká **kombináciou atomického fragmentu a problému generovania identifikátorov** – zaslanie atomického paketu pomôže útočníkovi identifikovať nasledujúci identifikátor. Výsledkom je už spomenutý DoS útok ale aj iné útoky, ako napríklad skenovanie portov. Riešenie prichádza s novým dokumentom RFC 6946 (Processing of IPv6 „Atomic“ Fragments) [17], kde sa uvádza spôsob ako predísť útokom spôsobeným zasielaním atomických fragmentov. Ak zariadenie obdrží takýto paket, musí ho spracovať v izolácii od ostatných paketov, a to bez ohľadu na to, či majú rovnakú zdrojovú adresu alebo iné identifikačné prvky.

Ďalšou bezpečnostnou hrozbou protokolu IPv6 zameranou na fragmentáciu IPv6 paketov je **prekrývanie fragmentov**. Situáciu bližšie vysvetľuje obrázok č. 10, kde je zobrazený paket rozdelený do dvoch fragmentov o veľkosti 1280 bytov.



Obr. 10 : Prekrývajúce sa fragmenty [14]

Testovaním predchádzajúceho útoku sa zistilo, že väčšina novších operačných systémov je voči nemu imúnna. Náchylnosť k tomuto však zaznamenali operačné systémy Ubuntu 10.4 a OpenBSD. Spracovaním takého to typu paketov môže dôjsť k zbieraniu informácii alebo zahlteniu zdrojov IDS (Intrusion Detection System) [14].

Prehľad bezpečnostných hrozieb a ich riešení			
Typ bezpečnostnej hrozby	Bezpečnostná hrozba	Navrhované riešenie	Použitá implementácia
Bezpečnostné hrozby spojené s fragmentáciou IPv6 paketu	Fragment minimálnej veľkosti	Použitie filtrovacích pravidiel	ip6tables -A INPUT -m frag - -fraglen number –j DROP ip6tables -A FORWARD -m frag - -fraglen number –j DROP
	Atomický fragment	Použitie filtrovacích pravidiel	ip6tables -A INPUT -m frag - -fragfirst - - fraglast –j DROP ip6tables -A FORWARD -m frag - -fragfirst - - fraglast –j DROP
	Problém s generovaním identifikátorov	Na koncovom zariadení použiť operačný systém Linux s novšou verziou jadra (3.1 a vyššie)	Linux bublinka-ipv6 3.2.0-4-686-pae
	Kombinácia atomických fragmentov a chybných identifikátorov	Kombinácia spomenutých riešení	ip6tables -A INPUT -m frag - -fragfirst - - fraglast –j DROP Linux bublinka-ipv6 3.2.0-4-686-pae
	Prekrývanie fragmentov	Na koncovom zariadení nepoužívať v texte spomenuté linuxové distribúcie	Debian GNU/Linux 7 (wheezy)

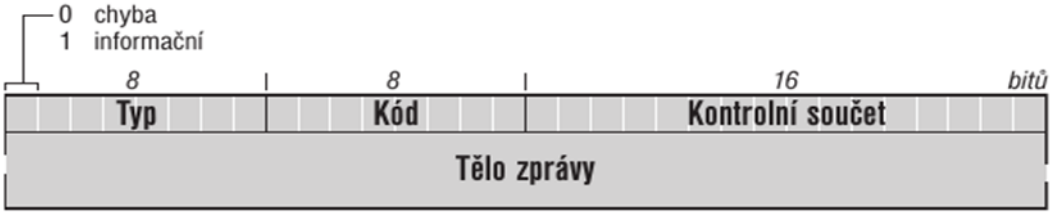
Tab. 5: Bezpečnostné hrozby spojené s fragmentáciou IPv6 paketu

2.6 Bezpečnostné hrozby spojené s protokolom ICMPv6

2.6.1 Protokol ICMPv6

ICMP (Internet Control Message Protocol) je dôležitou súčasťou internetovej komunikácie. Slúži ako režijný protokol, ktorého úlohou je informovať účastníkov komunikácie o chybách, ktoré sa vyskytli pri spracovaní paketov, pričom plní aj ďalšie funkcie ako napríklad zisťovanie

dosiahnuteľnosti zariadení [1]. Nový protokol IPv6 si vyžadoval pozmenenie pôvodného ICMP protokolu tak, aby spĺňal požiadavky na komunikáciu v IPv6 sieti. Preto vzniká dokument RFC 4443 s názvom Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification [18], ktorý definuje jeho novú verziu – ICMPv6 obohatenú o niekoľko nových typov správ a príslušných kódov. Implementácia ICMPv6 protokolu v IPv6 sieti je povinná.



Obr. 11 : Formát ICMPv6 správ [1]

2.6.2 Bezpečnostné hrozby a navrhované protiopatrenia

V predchádzajúcej verzii ICMP protokolu bolo už známych niekoľko druhov bezpečnostných útokov, ktoré mohli obmedziť funkcionalitu siete. Jedným z nich bol aj takzvaný DoS útok vedený pomocou ICMP správ na cieľové zariadenie. ICMPv6 je však voči takýmto typom útokov odolný vďaka niekoľkým novým kvantitatívnym parametrom – a to určením minimálneho časového odstupu medzi ICMPv6 správami alebo určením maximálneho podielu z celkovej šírky pásma, ktorý môžu ICMPv6 správy zabrať počas komunikácie. Ďalším efektívnym mechanizmom na zabránenie DoS útokov je použitie IPSec a zabalenie obsahu ICMPv6 správy do autentizačnej hlavičky [1].

V knihe IPv6 Security [9] nachádzame nasledujúce situácie, ktoré môžu byť vyhodnotené ako bezpečnostné riziká:

- chybové správy ICMPv6 môžu obsahovať **časti chybných paketov**,
- ICMPv6 správy typu 138 (Router Renumbering) obsahujú **rozsahy multicastu**,
- ICMPv6 **správy o prekročení Hop limitu** informujú útočníka o hraničných hodnotách Hop limitu.

Ak **minimálna MTU** predstavuje 1280 bytov, je vysoká pravdepodobnosť, že časť chybového paketu sa prenesie v ICMPv6 správe – tieto dáta môžu byť súčasťou utajenej komunikácie (convert channels). Pod pojmom **covert channel** rozumieme schopnosť použiť paket prenášaný po sieti na prenos dát, ktoré do paketu nepatria. Príkladom sú nadbytočné dáta ktoré môže útočník pripojiť za rozširujúcimi hlavičkami – z toho dôvodu vznikla rozširujúca hlavička typu No Next Header (59),

ktorá signalizuje, že sa jedná o poslednú hlavičku. Dáta, nachádzajúce sa za touto hlavičkou musia byť ignorované. Riešením spomenutej bezpečnostnej hrozby je preveriť obsah paketu pomocou firewallu, avšak odhalenie takéhoto typu paketov nie je jednoduché. Odporúča sa kontrolovať legitimitu zdrojovej IPv6 adresy a spoliehať sa na komunikačnú dôveryhodnosť odosielateľa [21].

Router Renumbering Option definovaný v RFC 2894 (Router Renumbering for IPv6), umožňuje meniť prefixy adries na sieťových smerovačoch. Používa pri tom ICMPv6 správy typu 138, ktoré obsahujú informácie o rozsahu multicastu (multicast site scope addresses) a site local destination addresses, ktoré sa nepoužívajú a často sú filtrované. Riešením je autentifikácia takéhoto typu správ pomocou IPSec, čo však stojí určitú námahu, nakoľko tieto pakety môžu byť odchytené na iných zariadeniach a implicitne zahodené. Filtrovať takýto typ správ môžeme v ip6tables pomcou pravidla *ip6tables -A INPUT -p icmpv6 - -icmpv6-type router-renumbering –j DROP*

Väčšina paketov s ICMPv6 správou má nastavený hop limit na maximálnu možnú hodnotu, a to 255 skokov. Ak smerovač dostane paket s hodnotou hop limitu rovnou 1, zníži ju na 0 a paket sa zahodí. Práve vtedy vzniká situácia, kedy sa naspäť k odosielateľovi paketu odošle ICMPv6 **správa o prekročení hop limitu** (ICMPv6 Time Exceeded kód 0). Útočník tak môže predpokladať, že ide o bežné správanie zariadení a preto bude generovať veľké množstvo paketov s hraničným Hop limitom. Riešením uvedenej situácie je napríklad zamedzenie odosielania ICMPv6 správ tohto typu. Použijeme na to filtrovacie pravidlo *ip6tables -A OUTPUT -p icmpv6 --icmpv6-type time-exceeded –j DROP*. Ďalšie riešenie sa ponúka vo filtrovaní paketov, ktoré majú nízku hodnotu TTL, a to v platnosti pre smerovače *ip6tables –A FORWARD –m ttl –ttl-lt number -j DROP*, tak aj pre konkrétne zariadenia *ip6tables –A INPUT –m ttl –ttl-lt number -j DROP*. Parameter number reprezentuje nami zvolenú číselnú hodnotu TTL.

K uvedeným útokom pripojíme aj zasielanie správ **ICMPv6 typu 137 (ICMPv6 Redirect Message)**. Tieto správy používajú na určenie najvýhodnejšej cesty paketu do cieľovej stanice. Ich zaslaním odosielateľ ovplyvňuje spôsob, akým sú pakety smerované. Útočník môže poslať falošnú ICMPv6 správu do miestnej siete, čím môže ovplyvniť smerovacie tabuľky jednotlivých uzlov v sieti. Dôsledkom je následné presmerovanie komunikácie.

Riešením je vypnúť podporu ICMPv6 správ na koncových zariadeniach. V linuxových operačných systémoch tak môžeme urobiť nastavením sysctl : *net.ipv6.conf.all.accept_redirects = 0*

Prehľad bezpečnostných hrozieb a ich riešení			
Typ bezpečnostnej hrozby	Bezpečnostná hrozba	Navrhované riešenie	Použitá implementácia
Bezpečnostné hrozby spojené s ICMPv6 protokolom	ICMPv6 správy obsahujúce convert channels	Použiť filtrovacie pravidlá na firewalle v závislosti od už predtým uskutočnenej komunikácie	ip6tables -A INPUT -p ipv6-icmp --icmpv6-type 1 -j ACCEPT ip6tables -A INPUT -p ipv6-icmp --icmpv6-type 2 -j ACCEPT (povolenie len určitých typov správ) ip6tables -A icmpv6-filter -p icmpv6 -j DROP (ostatné typy sa zahodia)
	ICMPv6 správy Router Renumbering	Použitie filtrovacích pravidiel	ip6tables -A INPUT -p icmpv6 - -icmpv6-type router-renumbering –j DROP
	ICMPv6 správy o prekročení hop limitu	Použitie filtrovacích pravidiel na odosielanie ICMPv6 správ typu Time Exceeded	ip6tables -A OUTPUT -p icmpv6 --icmpv6-type time-exceeded –j DROP
		Použitie filtrovacích pravidiel na zahadzovanie paketov s nízkym hop limitom	iptables –A INPUT –m ttl –ttl-lt number -j DROP iptables –A FORWARD –m ttl –ttl-lt number -j DROP
	ICMPv6 správy typu Redirect	Nastavenia na koncových zariadeniach (sysctl)	net.ipv6.conf.all.accept_redirects = 0

Tab. 6: Bezpečnostné hrozby spojené s ICMPv6 protokolom

2.7 Bezpečnostné hrozby spojené s protokolom Neighbor Discovery Protocol (NDP)

2.7.1 Neighbor Discovery Protocol (NDP)

Neighbor Discovery Protocol (NDP) má v IPv6 bohaté využitie. Rieši radu problémov spojených s interakciou medzi zariadeniami pripojenými do rovnakej siete. Používa sa pri [1] :

- zisťovaní linkových adries uzlov v rovnakej linkovej sieti,
- rýchlej aktualizácii neplatných položiek a zisťovaní zmien v linkových adresách,
- hľadaní smerovačov,
- presmerovaní,
- zisťovaní prefixov, parametrov siete, a údajov potrebných pre automatickú konfiguráciu,
- overovaní dosiahnuteľnosti susedov,
- detekcii duplicitných adries.

Jeho bližšiu charakteristiku podáva RFC 4861 (Neighbor Discovery for IPv6). NDP často pri svojej funkcionalite využíva ICMPv6 protokol – v tabuľke č. 5 sú popísané ICMPv6 správy, ktoré NDP protokol využíva [19].

Typ ICMPv6 správy	Popis
Router Solicitation (RS)	**Výzvu smerovača** posiela zariadenie, ktoré požaduje od smerovača zaslanie správy RA
Router Advertisement (RA)	**Ohlásenie smerovača** zasiela každý smerovač v pravidelných intervaloch, alebo po výzve RS za účelom oboznámiť zariadenia v sieti s parametrami siete ako sú prefix siete, MTU alebo hop limit
Neighbor Solicitation (NS)	**Výzva susedovi** je správa, ktorú posiela zariadenie v sieti, ak potrebuje zistiť MAC adresu suseda. Slúži pri detekcii duplicitných adries v sieti, pri zavádzaní automatickej konfigurácie, ale aj ako spôsob testovania dostupnosti susedov
Neighbor Advertisement (NA)	**Ohlásenie suseda** sa posiela ako odpoveď na správu NS, pričom v sebe nesie informáciu o linkovej adrese odosielateľa. Ak zariadenie zmenilo MAC adresu, je možné túto správu zaslať nevyžiadane
Presmerovanie	Informuje o zmene trasy paketu

Tab. 7 : ICMPv6 správy, ktoré používa NDP protokol

2.7.2 Bezpečnostné hrozby a navrhované protiopatrenia

NDP protokol je potrebné dobre zabezpečiť hlavne v otvorených sieťach, kde pripojenie nevyžaduje autentifikáciu užívateľov. Bezpečnostné hrozby týkajúce sa NDP protokolu môžeme rozdeliť do dvoch základných kategórií v závislosti od ich zamerania [20] :

- bezpečnostné hrozby **spôsobujúce DoS útok,**
- bezpečnostné hrozby **spôsobujúce presmerovanie** sieťovej komunikácie.

Útoky patriace do druhej kategórie môžu byť použité ako prostriedok na vyvolanie útokov prvej kategórie.

Do prvej kategórie radíme napríklad útok spôsobený **falošným konfiguračným prefixom**, kedy útočník odošle falošnú správu RA informujúcu príjemcov o prefixe, ktorý je neplatný. Zariadenia v sieti môžu pri automatickej konfigurácii brať túto správu do úvahy a na základe chybného prefixu si tak vytvoriť adresu. Výsledkom je, že zariadenie nebude schopné odoslať paket do siete a ani ho prijať, lebo jeho sieťová adresa bude neplatná.

K bezpečnostným hrozbám spôsobujúcim DOS útok patrí aj útok, kedy útočník **vytvára adresy podsiete podľa prefixu** a zasiela im pakety s ND správami. Posledný smerovač po ceste sa potom tieto správy pokúša spracovať, čo však spôsobuje jeho zdržanie v smerovaní paketov, ktoré

zaslali legitímni užívatelia siete. Uvedený útok je špecifický tým, že útočník nepotrebuje byť priamym členom siete, stačí mu poznať prefix, podľa ktorého vytvorí adresy.

Do tejto kategórie radíme aj útok využívajúci pri automatickej konfigurácii **kontrolu duplicitných adries v sieti**. Útočník odpovedá na každú požiadavku, v ktorej smerovač kontroluje, či je daná adresa obsadená. Novým zariadeniam v sieti je potom problematické prideliť sieťovú adresu.

Pre druhú kategóriu je charakteristický útok, pri ktorom útočník použije lokálnu linkovú adresu prvého smerovača v sieti na vytvorenie paketu s ICMPv6 hlavičkou presmerovania a odošle ho zariadeniu pripojenému v sieti. Dané zariadenie takto vytvorený paket presmerovania akceptuje, nakoľko jeho zdrojová adresa pochádza z lokálnej linkovej adresy smerovača.

Efektívnu obranu na uvedené útoky predstavuje **SEND (Secure Neighbour Discovery)** definovaný v RFC 3971 (SEcure Neighbor Discovery). Jeho úlohou je poskytnúť dostatočné bezpečnosť pomocou nasledujúcich zabezpečení [1]:

- CGA adresy,
- RSA podpis,
- časová značka,
- unikát.

Kryptograficky generované adresy (CGA) sme už spomenuli v podkapitole 2.3.2, kde sme uviedli spôsob ich vytvárania a ich využitie pri ochrane pred skenovaním siete. Vzhľadom k jednosmernosti hašovacej funkcie použitej v CGA útočník nie je schopný vytvoriť už k existujúcej adrese dátovú štruktúru s vlastnými kľúčmi.

RSA podpis (RSA Signature) umožňuje každú správu súvisiacu s objavovaním susedov digitálne podpísať. Najdôležitejšie parametre tohto bezpečnostného riešenia je hash kľúča (Key Hash), ktorým vieme identifikovať verejný kľúč a vlastný digitálny podpis (Digital signature). Algoritmom RSA sa podpisuje zdrojová a cieľová adresa.

Časová značka predstavujúca aktuálny čas a **unikát** pozostávajúci z náhodných dát poskytujú ochranu pred opakovaním, aby si útočník nemohol ukladať staršie platné ICMPv6 správy a znova ich odosielať.

Hlavnou výhodou SEND v porovnaní so štandardnými bezpečnostnými mechanizmami ako je IPSec, je jeho jednoduchosť a minimálna réžia. Pokým IPSec vytvára bezpečnostné asociácie a pomocou protokolov si vymieňa kľúče a algoritmy, pri SEND sa odosiela správa rozšírená o CGA, ktorá overuje, či odosielateľ skutočne disponuje uvedenou adresou a párom kľúčov, ktoré sa k nej viažu [1].

Ako možné implementačné riešenie uvádzame postup pri zavádzaní SEND mechanizmu pre CISCO zariadenia [29]:

1. enable (prístup do EXEC módu, v prípade potreby používateľ zadá požadované heslo)
2. configure terminal (vstup do globálneho konfiguračného módu)
3. ip http server (konfigurácia HTTP servera)
4. crypto pki trustpoint name (voliteľná deklarácia certifikačnej autority)
5. ip-extension [multicast | unicast] {inherit [ipv4 | ipv6] | prefix ipaddress | range min-ipaddress max-ipaddress} (voliteľný krok, určuje, či IP rozšírenia budú zahrnuté v požiadavke certifikátu, alebo budú generované pomocou CA)
6. revocation-check {crl | none | ocsp} (voliteľný krok, ktorý nastavuje metódu prerušenia)
7. exit (návrat do globálneho konfiguračného módu)
8. crypto pki server name (nastavenie PKI servera a vstup do konfiguračného módu pre server)
9. grant auto (automatická podpora certifikačných požiadaviek)
10. cdp-url url-name (nastavenie URL adresy, v prípade ak zariadenie používa CRL)
11. no shutdown (povolenie certifikačného servera)

3 Aplikácia na testovania bezpečnosti IPv6 prostredia

Cieľom tejto kapitoly je priblížiť návrh a implementáciu testovacej aplikácie, ktorej úlohou je testovanie bezpečnosti IPv6 siete. Aplikácia vychádza z teoretických základov predchádzajúcej kapitoly a zameriava sa na prieskumné útoky, útoky na hlavičky IPv6 paketu a útoky na fragmentáciu paketu. Súčasťou tejto kapitoly je aj uvedenie dvoch najpoužívanejších nástrojov na testovanie bezpečností IPv6 sietí a zhrnutie výhod a limitácií testovacej aplikácie.

3.1 Súvisiace riešenia

Cieľom tejto podkapitoly je oboznámiť čitateľa o už existujúcich nástrojoch a aplikáciách, ktoré slúžia na testovanie bezpečnosti IPv6 siete. Hlavnou úlohou týchto nástrojov je upozorniť na bezpečnostné riziká spôsobené zavedením nového protokolu do praktického používania.

3.1.1 SI6 Network´s IPv6 Toolkit

SI6 Networks' IPv6 Toolkit (A security assessment and troubleshooting tool for the IPv6 protocols) je sada nástrojov vytvorená za účelom posudzovania bezpečnosti IPv6 siete a riešenia problémov v sieti, ktoré vznikajú zanedbaním bezpečnostných opatrení. Aplikácia poskytuje nástroje na analýzu IPv6 adresy, fragmentácie, tzv. jumbo paketov, testovanie bezpečnosti ICMPv6 správ. Taktiež využíva prieskumné útoky ako aj útoky spojené so smerovacími hlavičkami a s protokolom NDP. Uvedená sada nástrojov je pre záujemcov dostupná na adrese http://www.si6networks.com/tools/ipv6toolkit/.

3.1.2 THC-IPV6

THC-IPv6 (The Hackers Choice IPv6 Toolkit) je koncept zahŕňajúci úplnú sadu nástrojov určených na testovanie zabezpečenia IPv6 siete. Obsahuje aj generickú triedu na generovanie paketov určených na testovanie bezpečnostnej politiky siete. Ponúka nám široký výber rôznych typov bezpečnostných hrozieb. Nájdeme tu nástroje určené na prieskumné útoky (alive6, trac6), nástroje určené na DoS útoky (dos-new-ip6, denial6), prípadne útoky spojené s ICMPv6 a s objavovaním susedov (flood_router6). Uvedená sada nástrojov je dostupná na adrese https://www.thc.org/thc-ipv6/.

3.2 Návrh aplikácie

Dôležitým krokom v našej práci bolo zvolenie si vhodného spôsobu implementácie útokov. Testovacia aplikácia mala byť v prvom rade vytvorená modulárne, aby sme s pribúdajúcimi IPv6 útokmi vedeli rozšíriť jej zameranie. Výhodou navrhnutej aplikácie je okrem jej modularity aj schopnosť použiť na testovanie už dostupné programy a príkazy z operačného systému.

3.2.1 Základné rozhranie

Štruktúru aplikácie môžeme rozdeliť do jednotlivých vrstiev. Každá vrstva je nositeľom jedinečnej funkcionality, pričom však aktívne komunikuje aj s ostatnými vrstvami:

1. **Vrstva grafického rozhrania**
   - statická časť
   - dynamická časť
2. **Vrstva bezpečnostných útokov**
   - prieskumné útoky
     - vedené s pomocou nástroja Scapy
     - vedené s pomocou nástrojov poskytnutých operačným systémom
   - útoky zamerané na smerovanie
   - útoky zamerané na fragmentáciu
   - generická trieda

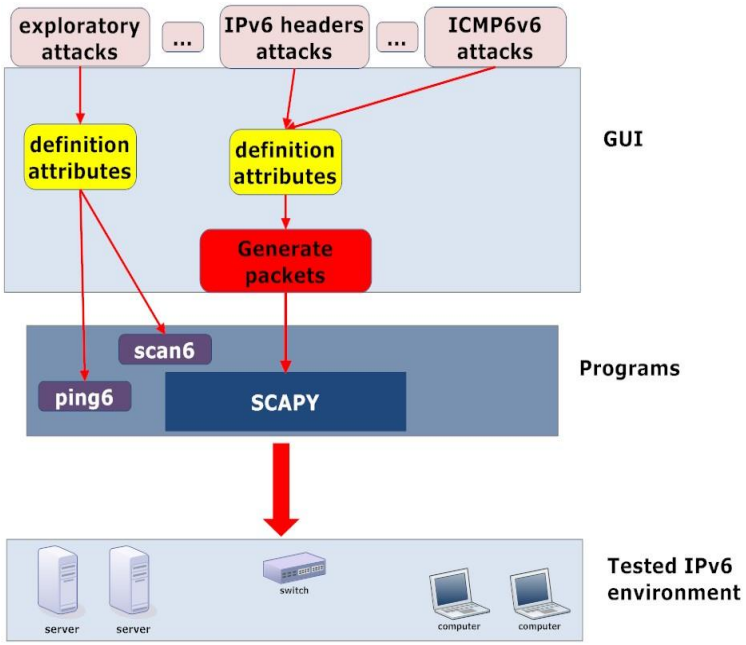
Vrstvu grafického rozhrania v našej aplikácii zabezpečuje knižnica Kivy. Zvolili sme si túto knižnicu hlavne na základe multiplatformového využitia a možnosti inštalovať aplikáciu aj na mobilné zariadenia. Vrstva grafického rozhrania pozostáva zo statickej a dynamickej časti. **Statická časť** je časť grafického rozhrania, ktorá sa počas behu aplikácie nemení. Obsahuje základné grafické rozhranie aplikácie, a jednotlivé grafické prvky potrebné na generovanie tried útokov. Nachádza sa v súbore s príponou *kv*. **Dynamická časť** sa naopak mení v závislosti od priebehu bezpečnostných útokov. Zahŕňa grafické prvky, ktorých úlohou je prehľadne zobrazovať výstupy pochádzajúce z vrstvy bezpečnostných útokov. Dynamická časť grafickej vrstvy je súčasťou hlavnej triedy testovacej aplikácie.

Vrstvu bezpečnostných útokov predstavujú individuálne triedy pomenované podľa príslušnosti k jednotlivým typom útokov.

Na vytvorenie uvedenej aplikácie sme použili nasledujúce nástroje a knižnice:

- programovací jazyk **Python 2.7**

- knižnicu **Kivy 1.7.1** určenú na vytvorenie grafického rozhrania
- nástroj na generovanie IPv6 paketov – **Scapy 2.02** a potrebné pluginy (tcpdump, graphviz, pyx, svglib)
- nástroje prístupné z operačného systému (ping6, traceroute6)



Obr. 12: Návrh testovacej aplikácie.

Hlavnou úlohou je zabezpečiť komunikáciu medzi požívateľom a testovacím prostredím. Používateľ komunikuje prioritne pomocou grafického rozhrania. Grafické rozhranie využíva používateľom zadané atribúty na generovanie tried, ktoré komunikujú s programovou časťou. Programová časť ďalej aplikuje používateľské požiadavky na testovaciu sieť.

3.2.2 Testovacie moduly

V aplikácii sa nachádzajú už vyššie spomenuté 4 typy testovacích modulov :

- testovací modul prieskumných útokov,
- testovací modul útokov spojených so smerovaním,
- testovací modul útokov spojených s fragmentáciou,
- testovací modul generickej triedy.

Testovací modul prieskumných útokov má za úlohu zistiť o cieľovom zariadení základné informácie ako napríklad jeho IPv6 adresu, dostupnosť zariadenia, doménové meno, porty, na ktorých je prístupná sieťová komunikácia, alebo počet zariadení, ktoré paket musí prejsť aby sa dostal do cieľa. Takto získané informácie budú následne použité na zabezpečenie nasledujúcich bezpečnostných testovacích modulov. Tento testovací modul sme rozdelili do dvoch častí.

Prvá časť reprezentuje už spomenuté základné informácie. Používateľ zadá do vstupného poľa doménové meno alebo IPv6 adresu a označí informácie, ktoré chce získať. Pri označení niektorých typov informácií (ping, TCP nmap) sa môže otvoriť dialóg s požadovanými nastaveniami (ako je počet odosielaných paketov alebo číslo portu, na ktorý chceme pristúpiť). Po zatvorení všetkých dialógov môže používateľ zahájiť útok. Výstupné aplikácie sa následne zobrazia v dialógovom okne.

Druhá časť sa zameriava na traceroute6 útok. Do vstupného poľa zadá používateľ IPv6 adresu alebo doménové meno cieľovej stanice. Tento útok umožňuje overiť viacero zariadení súčasne, preto môžeme na vstup očakávať aj viacero doménových mien alebo IPv6 adries, oddelených čiarkou. Ďalej môže používateľ zadať voliteľné hodnoty ako počet odosielaných paketov alebo povoliť možnosť zobrazenia grafického výstupu uskutočneného útoku. Výstupom je tabuľka obsahujúca zoznam zariadení, ktoré pakety navštívili po ceste do cieľovej stanice. Ak si používateľ zvolil možnosť grafického výstupu útoku, po kliknutí na vygenerované tlačidlo sa zobrazí grafické zobrazenie navštívených destinácii. Výhodou grafického zobrazenia je, že používateľ môže prehľadne vidieť štruktúru jednotlivých sietí a podsietí, ktorými pakety prechádzali.

Testovací modul útokov spojených so smerovaním pozostáva z vytvorenia základnej smerovacej hlavičky typu 0. Používateľ zadá IPv6 adresu cieľovej stanice do prvého vstupného poľa. Adresy, ktoré patria do smerovacej hlavičky užívateľ zadá do druhého vstupného poľa a oddelí ich novým riadkom. Výstupom je dialógové okno, ktoré informuje o úspešnosti útoku.

Testovací modul útokov spojených s fragmentáciou pozostáva z vytvorenia atomického fragmentu, ktorý sa odošle na cieľové zariadenie. Používateľ na vstup zadáva iba IPv6 adresu cieľového zariadenia, pričom výstup je dialógové okno, ktoré informuje o úspešnosti útoku.

Testovací modul generickej triedy obsahuje základné metódy na vytvorenie IPv6 paketu, t.j. metódy na vytvorenie základnej hlavičky a prípadných rozširujúcich hlavičiek protokolu IPv6. Používateľ si najskôr zvolí typ rozširujúcej hlavičky, prípadne ICMPv6 segment, ktorý chce vygenerovať. Následne sa po stlačení tlačidla vo vopred vyznačenej oblasti vygeneruje požadovaný objekt, ktorý je možné ťahaním posúvať. Po kliknutí na rozširujúcu hlavičku sa otvorí dialógové okno a používateľ môže nastaviť parametre rozširujúcej hlavičky. V prípade ICMPv6 segmentu dialógové okno iba zobrazuje informácie o type ICMPv6 správy. Vygenerované objekty je potrebné

rozložiť do označenej plochy podľa stanoveného poradia, definovaného v RFC 2460. Po odoslaní paketu sa ako výstup vygeneruje dialógové okno, obsahujúce informácie o úspešnosti alebo neúspešnosti útoku.

3.2.3 Výhody a obmedzenia navrhovaného riešenia

Navrhované riešenie prináša so sebou niekoľko výhod. Medzi dominantné výhody patrí **grafické rozhranie**, ktoré zabezpečuje jednoduché ovládanie aplikácie. Grafické prostredie je navrhnuté tak, aby ho bolo možné používať aj na **mobilných zariadeniach** (ako je napríklad Android alebo iOS). Vyššie uvedené súvisiace riešenia neobsahujú grafické rozhranie.

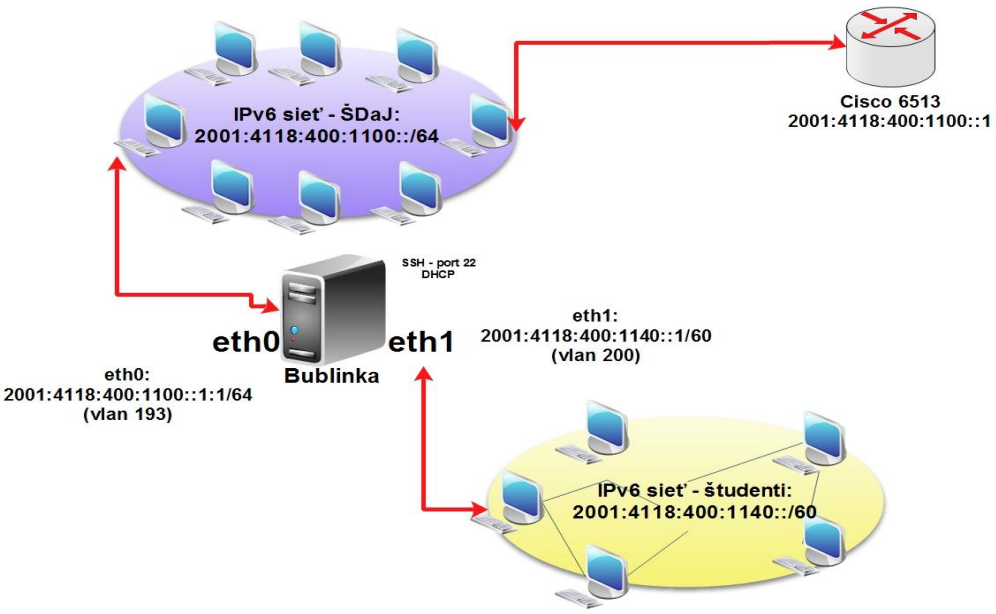
Z funkčného hľadiska predstavuje významnú výhodu **modularita aplikácie.** Pri postupnom zavádzaní protokolu IPv6 vznikajú vždy nové námety na testovanie bezpečnosti. Generická trieda umožňuje užívateľovi poskladať niekoľko typov útokov v závislosti od použitých rozširujúcich hlavičiek, a zároveň je možné ju jednoducho doplniť o nové bezpečnostné hlavičky. Modul na vytváranie nových typov útokov obsahuje THC-IPv6. V SI6 táto funkcionalita chýba.

3.3 Implementácia aplikácie

Implementácia aplikácie prebehla v niekoľkých fázach. V prvej fáze sme navrhli implementačné a testovacie prostredie. Bližšie ho popisujeme v kapitole 3.3.1. Následne sme pokračovali implementáciou základného rozhrania aplikácie, ktoré popisujeme v kapitole 3.3.2. Ako posledným sme venovali pozornosť implementáciám jednotlivých testovacích modulov. Tie rozoberáme v závere tejto kapitoly.

3.3.1 Implementačné a testovacie prostredie

Jedným zo sekundárnych cieľov našej práce bolo vytvoriť prostredie vhodné na testovanie bezpečnostných útokov uvedených v teoretickej časti práce. Priestory na vytvorenie uvedenej počítačovej siete nám poskytli Študentské domovy a jedálne UPJŠ v Košiciach.



Obr. 13 : Architektúra testovacej počítačovej siete IPv6

Pri testovaní útokov je dôležité vedieť, v akej sieti sa nachádza zariadenie útočníka a zariadenie, na ktoré bude vedený útok. V našom prípade sa zariadenie útočníka nachádza v sieti s prefixom 2001:4118:400:1100::/64 . Cieľové zariadenie je nakonfigurovaný smerovač, ktorý prepája dve siete (sieť IPv6-ŠDaJ a IPv6 sieť- študenti). Na oboch zariadeniach je nainštalovaný operačný systém Linux, odlišujú sa však v linuxových distribúciach – zariadenie útočníka používa Fedoru 19, zatiaľ čo cieľové zariadenie používa Debian 7.

3.3.2 Implementácia základného rozhrania

Základné rozhranie aplikácie pozostáva z uvedených nástrojov a knižníc:

- programovací jazyk **Python 2.7**,
- knižnica **Kivy 1.7.1** určená na vytvorenie grafického rozhrania.

Ako základný pilier aplikácie sme si vybrali programovací jazyk **Python** a to hlavne pre jeho jednoduchú implementáciu a v neposlednom rade pre skutočnosť, že prevažná väčšina nástrojov použitá pri testovaní bezpečnostných hrozieb je napísaná pomocou tohto programovacieho jazyka.

Použitá knižnica Kivy nám umožňuje vytvárať aplikácie vhodné pre rôzne prostredia operačných systémov – od Linuxov, cez Windows až po operačné systémy mobilných zariadení, ako Android a iOS. Takto vytvorené grafické rozhranie je možné zapísať staticky do súboru s

príponou kv alebo pomocou Pythonu použiť dynamický spôsob generovania jednotlivých prvkov. Túto skutočnosť sme využili aj pri písaní našej aplikácie – statickú časť grafickej vrstvy sme zapísali do súboru s príponou kv, dynamickú časť grafickej vrstvy sme generovali v hlavnej triede.

3.3.3 Implementácia testovacieho modulu – prieskumné útoky

Účelom tohto testovacieho modulu je poskytnúť základné informácie o cieľovom zariadení. Zo získaných informácii môže potom útočník čerpať v nasledujúcich útokoch.

Testovací modul svoje fungovanie používa uvedené technológie:

- **scapy 2.02** a potrebné pluginy (tcpdump, graphviz, pyx, svglib),
- **nástroje prístupné z operačného systému** (napr. ping6, traceroute6).

Scapy je nástroj určený na manipuláciu s IPv6 paketmi. Pomocou tohto nástroja je útočník schopný vytvoriť základnú hlavičku a aj rozširujúce hlavičky podľa vlastných kritérií a nastavení. Takto vytvorený paket alebo súbor paketov útočník odošle na cieľové zariadenie. Scapy disponuje aj generovaním grafov, čo sme neskôr využili pri traceroute6 útoku.

Prieskumné útoky vo vytvorenej aplikácii sú vedené z doménového mena alebo IPv6 adresy. Z praktického hľadiska sme rozdelili prieskumné útoky do viacerých častí. Prvá časť predstavuje útoky určené na zistenie základných informácii ako IP adresy, doménového mena a dostupnosti cieľového zariadenia, prípadne preverovanie otvorených portov. Na uvedené útoky sme použili pythonovú knižnicu Socket ale aj nástroje prístupné zo základnej výbavy operačného systému (ping6). Podstatnú časť prieskumných útokov sme však vytvorili za pomoci nástroja Scapy.

V aplikácii sme ďalej použili už vytvorený **nástroj traceroute6 z rozhrania Scapy**, slúžiaci na identifikáciu zariadení v sieti a to od odoslania paketu až po jeho doručenie k cieľovému zariadeniu. Nástroje typu traceroute odosielajú pakety s obmedzeným počtom hopov, pričom počet hopov sa inkrementuje po každej odozve. Traceroute6 v Scapy však nečaká na odpoveď od každého uzla, ale odošle všetky pakety s naraz, čoho dôsledkom je nemožnosť definovať trvanie útoku. Z toho dôvodu je zadanie maximálnej hodnoty TTL nevyhnutnosťou. Hodnota maximálnej TTL je bez zadania užívateľa 34 hopov. Scapy ďalej poskytuje možnosť stanoviť cieľový port, osloviť viacero cieľov súčasne a graficky zobraziť výsledok prieskumného útoku. Pri implementácii grafického zobrazenia výsledku traceroute6 sme narazili na vývojovú chybu v balíčku Scapy verzie 2.02. Zle pomenovanie triedy spôsobilo, že generovanie grafického zobrazenia cesty IPv6 paketu do cieľovej stanice nebolo možné. Chybu v súbore *../scapy/layers/inet.py* sme opravili a umožnili sme tak našej aplikácii vytvárať grafické zobrazenia.

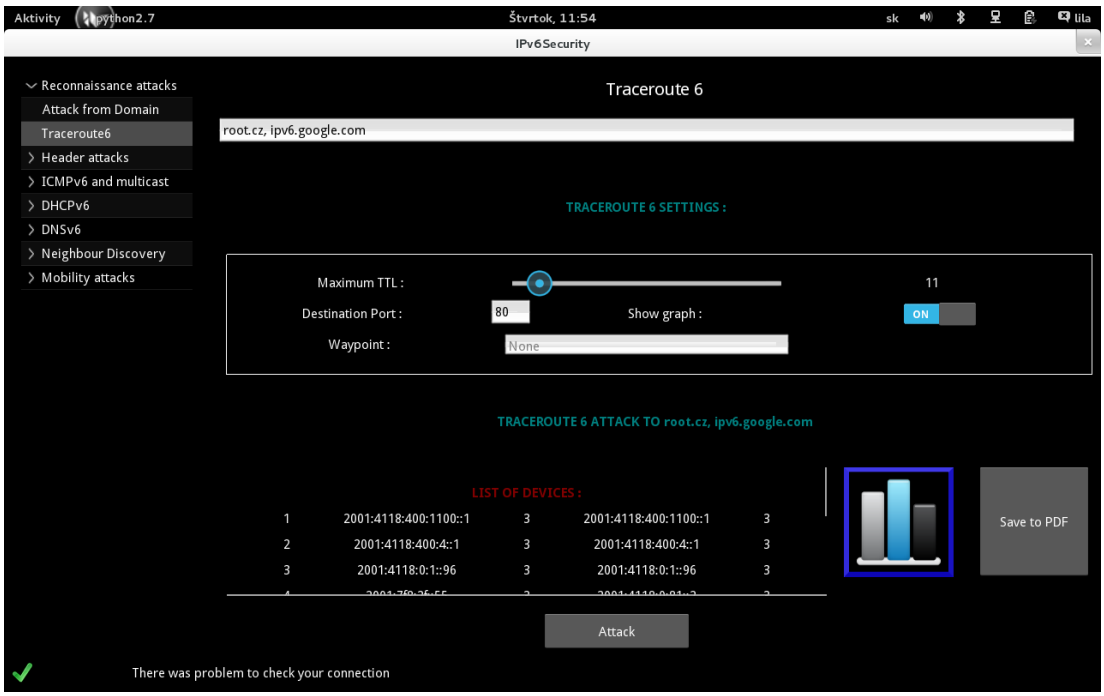
Je potrebné nahradiť nasledujúcu časť kódu v súbore../*scapy/layers/inet.py* :

if not (ICMP in r and r[ICMP].type == 11) and not (conf.ipv6_enabled and scapy.layers.inet6.IPv6 in r and ***ICMPv6TimeExceeded*** *in r):*

nasledujúcou časťou :

if not (ICMP in r and r[ICMP].type == 11) and not (conf.ipv6_enabled and scapy.layers.inet6.IPv6 in r and ***scapy.layers.inet6.ICMPv6TimeExceeded*** *in r):*

čím dosiahneme nápravu existujúcej chyby.



Obr. 14: Ukážka aplikácie – prieskumné útoky

3.3.4 Implementácia testovacieho modulu – útok na smerovanie IPv6 paketu

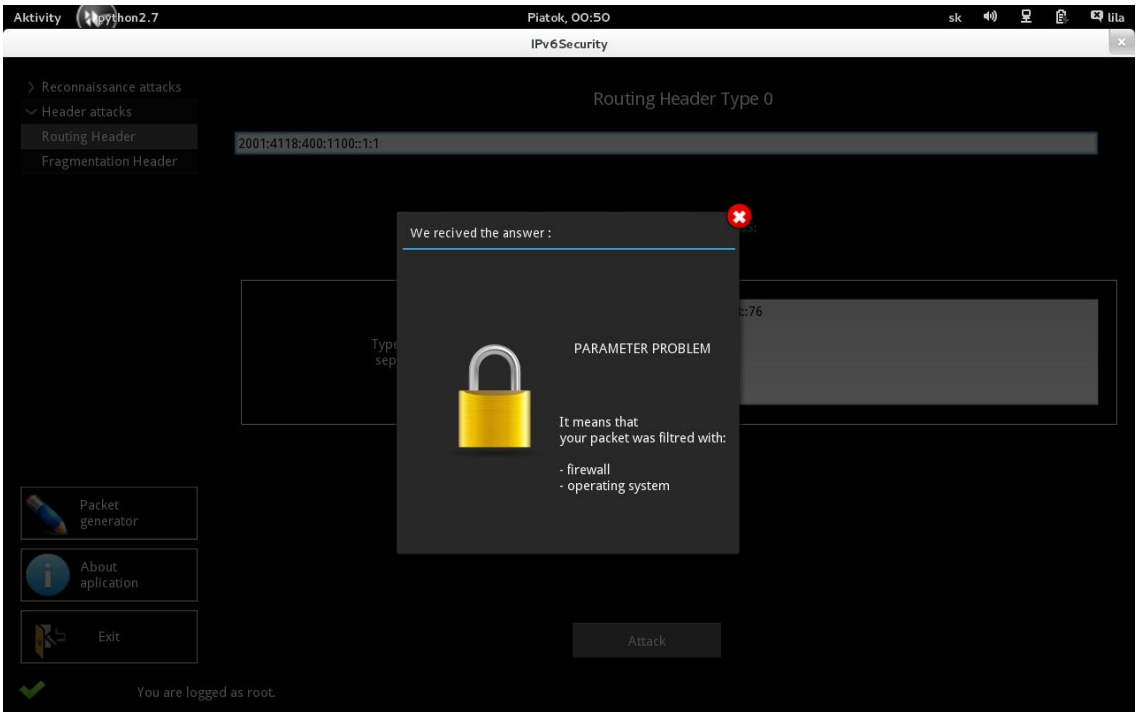
Účelom tohto testovacieho modulu je overiť filtrovacie pravidlá na zachytávanie paketov obsahujúcich smerovaciu hlavičku typu 0.

Testovací modul na svoje fungovanie používa :

scapy 2.02 a potrebné pluginy (tcpdump, graphviz, pyx, svglib)

Testovací nástroj Scapy v tomto prípade slúži na vytvorenie paketu pozostávajúceho zo základnej hlavičky, smerovacej hlavičky a ICMPv6 segmentu (ICMv6 Echo Request). Cieľová IPv6 adresa zadaná v prvom z dvoch vstupných polí patrí do základnej hlavičky, zatiaľ čo zoznam adries zadaný v druhom vstupnom poli priradíme do smerovacej hlavičky. Odoslaním paketu

očakávame spätnú odpoveď v ICMPv6 správe typu Echo Reply, a to v prípade ak bol útok úspešný. Filtrovanie odoslaného paketu nám napovie spätná správa typu ICMPv6 Parameter Problem.



Obr. 15: Ukážka aplikácie – útoky spojené so smerovacou hlavičkou

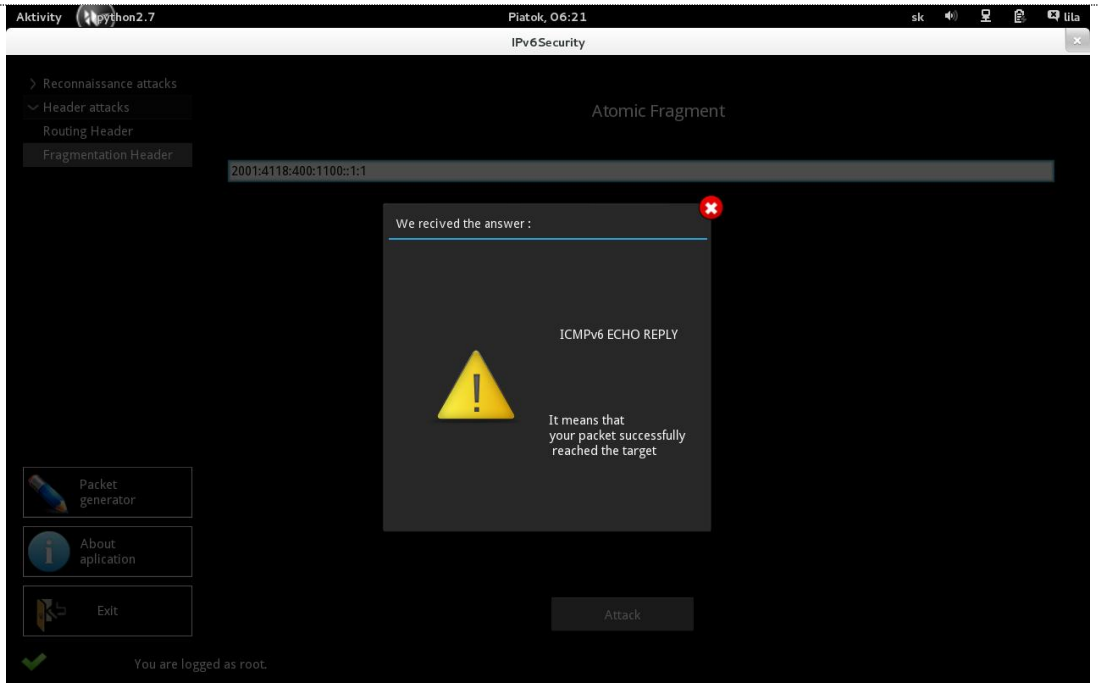
3.3.5 Implementácia testovacieho modulu – útok spojený s fragmentáciou

Účelom tohto testovacieho modulu je overiť bezpečnostné nastavenia týkajúce sa fragmentácie, a to zaslaním atomického fragmentu.

Testovací modul svoje fungovanie používa :

scapy 2.02 a potrebné pluginy (tcpdump, graphviz, pyx, svglib)

Útok spojený s fragmentáciou využíva Scapy na vytvorenie paketu, ktorý obsahuje takzvaný atomický fragment. Atomický fragment je fragment, ktorý je prvým a zároveň aj posledným prvkom fragmentácie. Jeho bližší popis sme uviedli v teoretickej časti. Okrem IP adresy cieľového zariadenia, nie je potrebné nič zadávať. Výsledný paket pozostáva zo základnej hlavičky, fragmentačnej hlavičky (kde hodnoty M príznaku a posunu sú nastavené ako nulové) a napokon z ICMPv6 (Echo Request) segmentu. Ak je útok úspešný, používateľ obdrží ICMPv6 Echo Reply.



Obr. 16: Ukážka aplikácie – útoky spojené s fragmentáciou IPv6 paketu

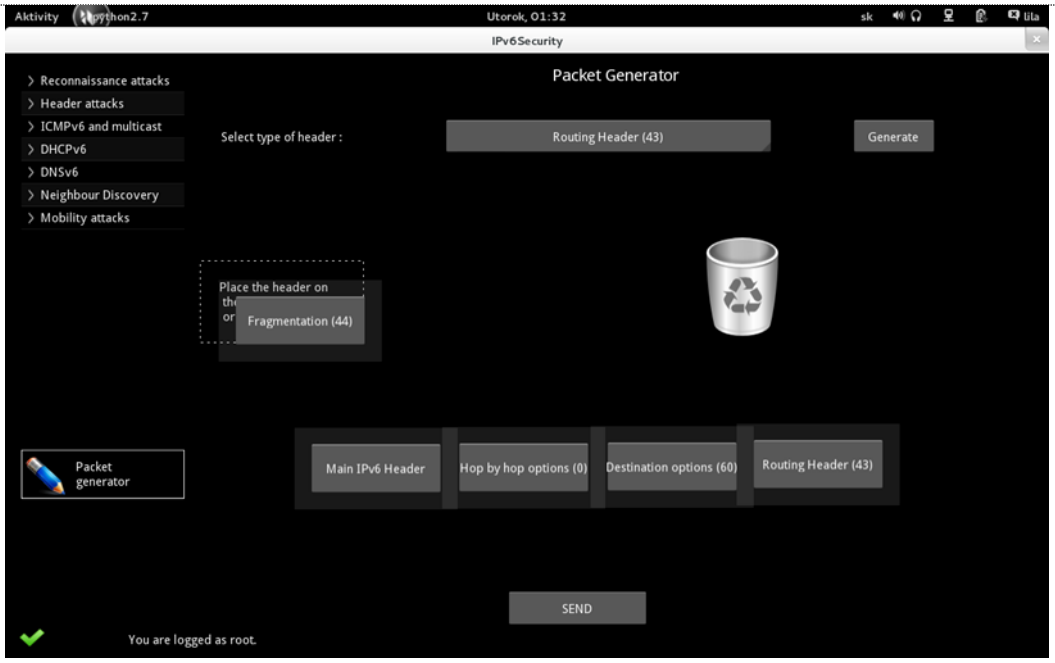
3.3.6 Implementácia testovacieho modulu – generická trieda

Účelom tohto testovacieho modulu je vytvorenie generickej triedy, ktorá umožní používateľovi vytvárať vlastné útoky.

Testovací modul svoje fungovanie používa :

scapy 2.02 a potrebné pluginy (tcpdump, graphviz, pyx, svglib)

Generická trieda využíva Scapy ako nástroj na generovanie paketov v závislosti od požiadaviek útočníka. Disponuje základnou hlavičkou, 4 typmi rozširujúcich hlavičiek a ICMPv6 segmentom. Po vytvorení potrebných hlavičiek a vyplnení jednotlivých parametrov je potrebné hlavičky hierarchicky usporiadať podľa poradia stanoveného RFC 2460. Miesto, kde sa majú usporiadané pakety nachádzať je farebne vyznačené. Ak sa útočník rozhodne hlavičku nepoužiť, môže ju jednoducho zahodiť do koša. Scapy umožňuje nastaviť parametre pre jednotlivé rozširujúce hlavičky. Odporúčame vytvárať pakety zakončené s ICMPv6 segmetom, aby sme si mohli byť istí, že paket, ktorý sme poslali došiel do cieľa a cieľ naňho primerane reagoval.



Obr. 17: Ukážka aplikácie – prieskumné útoky

Záver

Hlavnou témou tejto záverečnej práce bola problematika implementácie protokolu IPv6 v produkčnom prostredí. Pred zavedením protokolu do bežného používania je dôležité poznať jeho jedinečné vlastnosti ale aj odlišnosti voči predchádzajúcej, aktuálne používanej verzii protokolu. Vzájomnému porovnaniu protokolov sme sa venovali v úvodnej kapitole s názvom Základná charakteristika protokolu IPv6. Medzi jedinečné vlastnosti protokolu sme zaradili špecifický formát hlavičky, spôsob adresácie. Spomenuli sme tiež prostriedky stavovej aj bezstavovej automatickej konfigurácie, mobilitu či IPSec. Rozdiely medzi protokolom IPv4 a IPv6 sme znázornili v prislúchajúcej tabuľke. Vzájomný vzťah medzi protokolmi sme načrtli aj priblížením problematiky interoperabilných mechanizmov.

Jedným z hlavných cieľov tejto práce bolo poukázať na potrebu zabezpečenia sieťových pripojení používajúcich protokol IPv6. Danou problematikou sme sa zaoberali v nasledujúcej kapitole. Najskôr sme sa oboznámili so základnými pojmami operujúcimi so sieťovou bezpečnosťou, následne sme uviedli niekoľko typov delení bezpečnostných hrozieb, z ktorých sme vychádzali pri vytváraní vlastnej typológie. Bezpečnostné hrozby sme rozdelili na 5 základných typov.

Prvým typom bezpečnostných hrozieb sú hrozby spojené s prieskumnými útokmi, ktoré sú zamerané na získane informácií o cieľových zariadeniach. Často sa vedie diskusia, či zhromažďovanie informácií o zariadeniach v sieti môžeme považovať za útok, avšak ako sme neskôr zistili pri implementácii útokov do testovacej aplikácie, takto získané informácie predstavujú pre útočníka značnú výhodu. Bezpečnostné opatrenia spomenuté v práci však často nestačia na odstránenie tohto typu útokov, nakoľko vyhľadávacie nástroje na Internete a registre whois stále poskytujú dostatočné množstvo informácií o sieťových pripojeniach.

Druhý typ bezpečnostných hrozieb predstavujú hrozby zamerané na bezpečnosť smerovacej hlavičky IPv6 paketu. Smerovacia hlavička IPv6 paketu typu 0 predstavuje zvýšené riziko vzniku DoS útokov, preto sa aj odporúča filtrovať pakety obsahujúce tento typ hlavičky. V práci uvádzame útok, pomocou ktorého sa útočník vyhne filtrovacím pravidlám. Zároveň navrhujeme aj dodatočné riešenie – rozšírenie filtrovacích pravidiel alebo použite DPA algoritmu.

Bezpečnostné hrozby spojené s fragmentáciou paketu sa venujú možným bezpečnostným rizikám, ktoré môžu nastať pri vytváraní fragmentov alebo pri ich opätovnom skladaní do pôvodného paketu. Medzi takéto situácie patrí napríklad nesprávne generovanie identifikátorov fragmentov, nesprávna veľkosť fragmentov alebo vytváranie prekrývajúcich sa fragmentov. Voči

niektorým z uvedených útokov sa vieme bezpečne ubrániť nastavením správnych filtrovacích pravidiel.

Štvrtým typom bezpečnostných hrozieb sú hrozby spojené s ICMPv6 správami. Do tejto kategórie radíme situácie, kedy sa ICMPv6 správy stávajú prostriedkom na prenášanie chybných dát, alebo pomáhajú útočníkovi zistiť dôležité sieťovej konfigurácie (ako sú rozsahy multicastu a hodnoty hop limitov). Riešením je použitie filtrovacích pravidiel firewallu na niektoré typy ICMPv6 správ, pripadne použitie autentifikácie a šifrovania hlavičiek pomocou IPSec.

Posledný typ nami zvolených bezpečnostných hrozieb sa zaoberá NDP protokolom. V závislosti od zamerania hrozieb sme tento typ rozdelili na podkategóriu hrozieb vyvolávajúcich priamo zahltenie zdrojov (DoS) a podkategóriu hrozieb zameraných na presmerovanie paketov. NDP v rámci IPv6 protokolu zaberá široké pole pôsobnosti, preto je potrebné zamedziť útokom čo možno v najvyššej miere. Častým riešením je použitie SEND protokolu.

Poslednú kapitolu tejto práce tvorí návrh a implementácia aplikácie na testovanie bezpečnostných zraniteľností IPv6 sietí. Podstatou aplikácie je testovanie zraniteľností IPv6 protokolu v rámci počítačovej siete tým, že uskutočníme útok a zistíme, či bol, resp. nebol úspešný. V rámci aplikácie sme implementovali základné prieskumné útoky (preklad IPv6 adries na doménové mená a naopak, ping6, traceroute6, TCP nmap6), útok spojený so smerovaním pomocou smerovacej hlavičky typu 0, útok zameraný na fragmentáciu (tzv. atomický fragment) a napokon generickú triedu, schopnú vytvoriť IPv6 paket na základe vopred stanovených požiadaviek.

Zoznam použitej literatúry

[1] SATRAPA, P. IPv6: internetový protokol verze 6. 3., aktualiz. a dopl. vyd. Praha:CZ.NIC, 2011, 407 s. CZ.NIC. ISBN 978-80-904248-4-5.

[2] DE GHEIN, L. Basic IPv6. In: Cisco Systems [online]. 2001 [cit. 2013-09-20]. 196 s. Dostupné z: https://docs.google.com/viewer?url=http%3A%2F%2Fwww.6net.org%2Fevents%2Ftraining-2003%2Fipv6-basics.pdf

[3] RFC 4291. In: IETF [online]. 2007 [cit. 2013-09-21]. Dostupné z: http://tools.ietf.org/rfc/rfc4291.txt

[4] MUN, Y. and Hyewon, K. L. Understanding IPv6. 2nd ed. New York: Springer, 2005, xvi, 279 p. ISBN 03-872-5429-3.

[5] KAUSHIK, D. Stateless Auto Configuration. Ipv6.com [online]. 2008 [cit. 2014-05-02]. Dostupné z: http://ipv6.com/articles/general/Stateless-Auto-Configuration.htm

[6] RFC 2460. In: IETF [online]. 2007 [cit. 2013-09-27]. Dostupné z: http://tools.ietf.org/rfc/rfc2460.txt

[7] BEIJNUM, I. Running IPv6. Berkeley: Apress, 2006, xix, 266 s. ISBN 15-905-9527-0.

[8] STOCKEBRAND, B. IPv6 in practice: a Unixer's guide to the next generation Internet. 1st ed. New York: Springer, 2006, xxiv, 390 p. ISBN 35-402-4524-3.

[9] HOGG, S. and VYNCKE, E. IPv6 security. Indianapolis: Cisco Press, 2009, xxi, 540 s. ISBN 978-1-58705-594-2.

[10] MINOLI, D. and KOUNS, J. Security in an IPv6 environment. Boca Raton: CRC Press, 2009, xvi, 272 p. ISBN 978-142-0092-295.

[11] RFC 2828. In: IETF [online]. 2000 [cit. 2013-12-20]. Dostupné z: http://tools.ietf.org/rfc/rfc2828.txt

[12] DURDAGI, E. and BULDU, A. 2010. IPv4/IPv6 security and threat comparisons. Procedia Soc. Behav. Sci., 2: 5285-5291.

[13] JAEDEOK L. and YOUNGKI, K.: Protection Algorithm against security holes of IPv6 routing header. Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference , vol.3, no., pp.2004-2007, 20-22 Feb. 2006

[14] ATLASIS, A.: Attacking ipv6 implementation using fragmentation. In: Black Hat

Europe[online]. 2012[cit. 2014-02-20]. Dostupné z:
http://media.blackhat.com/bh-eu-12/Atlasis/bh-eu-12-Atlasis-AttackingIPv6-WP.pdf

[15] RFC 6145. In: IETF [online]. 2011 [cit. 2014-02-20]. Dostupné z:
http://tools.ietf.org/rfc/rfc6145.txt

[16] ATLASIS, A. Attacking ipv6 implementation using fragmentation : one year later. In: Black Hat Europe [online]. 2013[cit. 2014-02-20]. Dostupné z:
https://www.troopers.de/wpcontent/uploads/2013/01/TROOPERS13Fragmentation_Overlapping_Attacks_Against_IPv6_One_Year_Later-Antonios_Atlasis.pdf

[17] RFC 6946. In: IETF [online]. 2013 [cit. 2014-03-20]. Dostupné z:
http://tools.ietf.org/rfc/rfc6946.txt

[18] RFC 4443. In: IETF [online]. 2006 [cit. 2014-04-20]. Dostupné z:
http://tools.ietf.org/rfc/rfc4443.txt

[19] BARBHUIYA, F. A., BANSAL G., et al. Detection of neighbor discovery protocol based attacks in IPv6 network. Networking Science, 2013, Volume 2, Number 3-4, Page 91

[20] ARKKO, J., AURA, T., KEMPF, J., MANTYLA, V. M., NIKANDER, P., ROE, M.
2002. Securing IPv6 neighbor discovery and router discovery. In Proceedings of the 2002 ACM Workshop on Wireless Security (WiSe). ACM Press, Atlanta, GA, USA, 77–8

[21] DEGEN, S. and HOLTZER, A., et.al. Testing the securitiy of IPv6 implementations [online].2014[cit.2014-02-22]. Dostupné z:
https://www.tno.nl/downloads/testing_the_security_of_IPv6_implementations.pdf

[22] PHILIHANTO, A. A Complete Guide on IPv6 Attack and Defense. 1st ed. [ebook].SANS Institute.[cit.2014-04-28]. Dostupné z :
http://www.sans.org/reading_room/whitepapers/detection/complete-guide-ipv6-attack-defense_33904.pdf

[23] RFC 5157. In: IETF [online]. 2008 [cit. 2014-04-28]. Dostupné z:
http://tools.ietf.org/rfc/rfc5157.txt

[24] RFC 3879. In: IETF [online]. 2004 [cit. 2013-04-28]. Dostupné z:
http://tools.ietf.org/rfc/rfc3879.txt

[25] OLEJÁR, D., BUBÁK, M., HUDEC, L., KOPÁČIK, I., ORAVEC, I.,

SALLER, E., SOVIŠ, F., STANEK, M., STANKO, J. Informačná bezpečnosť. Bratislava: MF SR, 2013.

[26] YAO, G. , BI, J., WANG, S., ZHANG, Y., LI, Y. A Pull Model IPv6 Duplicate Address Detection. In: 35th Annual IEEE Conference on Local Computer Networks [online]. Denver, Colorado, 2010 [cit. 2014-04-29]. Dostupné z: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5735746&tag=1

[27] BIONDI, P. and EBALARD, A. IPv6 Routing Header Security. In: EADS Corporate Research Cent [online]. Suresnes, FRANCE, 2007 [cit. 2014-04-29]. Dostupné z: http://people.su.se/~jj/junk/012.pdf

[28] ROSE, J. G. Advantages & Security, IPv6 Advantages and Security Considerations of Utilizing the IPv6 Protocol [online]. 2010 [cit. 2014-04-30]. Dostupné z: http://sysmincomputing.files.wordpress.com/2010/12/ipv6securitypaper.pdf

[29] IPv6 Secure Neighbor Discovery. Cisco [online]. 2012 [cit. 2014-05-02]. Dostupné z: http://www.cisco.com/en/US/docs/ios-xml/ios/sec_data_acl/configuration/15-2mt/ip6-send.html

Prílohy

Príloha A: Grafická vrstva – statická časť

Príloha B: Grafická vrstva – dynamická časť (traceroute6)

Príloha C: Testovací modul – generická trieda

Príloha D: CD nosič s testovaciou aplikáciou

Príloha A – Grafická vrstva – statická časť

```
<RoutingHeader>
        BoxLayout:
                orientation: 'vertical'
                pos_hint: {'x': 0, 'y': 0}
                padding: 20
    spacing: 30
    txt_inpt:txt_inpt
                Title:
                        text: 'Routing Header Type 0'
                TextInput:
                        id: txt_inpt
                        hint_text: 'Enter IPv6 address of destination'
                        #size_hint: .7, .125
                        size_hint: 1, None
                        pos_hint: {'y': .9}
                        height: 30
                        multiline: False
                Inform:
                        text: 'ROUTING HEADER SETTINGS:'


                GridLayout:
                        id:addresses
                        cols: 2
                        canvas.before:
                                Color:
                                        rgb: 1,1,1
                Line:
                rectangle: self.x+10,self.y-20,self.width+10,self.height+40
      Label:
        text: 'Type the list of addresses \n separated with Enter :'
                        TextInput:
                                id:addresses
```

```
                input_type: 'text'
                auto_indent: True

    BoxLayout:
    BoxLayout:
        spacing: 40
        padding: 20
        orientation: 'horizontal'
        pos_hint: {'x': .35}

        Button:
            size_hint: None, None
            size: root.width/6,root.height/17
            text: 'Attack'
            on_press: root.create_rh(addresses.text, txt_inpt.text)
```

Príloha B – Grafická vrstva – dynamická časť (traceroute6)

```
class Traceroute6Attack (FloatLayout):
    def __init__(self, **kwargs):
        super(Traceroute6Attack, self).__init__(**kwargs)

    def open_result_box(self, BoxLayout, target, maxTTL, port, graph):
        BoxLayout.clear_widgets()
        BoxLayout.add_widget(Label(text="TRACEROUTE 6 ATTACK TO "+target, spacing=10,
size_hint=(1,None), color=(0, 255, 255, .5),bold=True ))

        count_targets=1
        #multiple targets
        if ',' in target:
            count_targets=target.count(',')+1
            target_strings=target.split(',')
            print target_strings
            for i in range (0, len(target_strings)):
                target_strings[i]=target_strings[i].strip()
            output=getTraceroute6(target_strings,int(maxTTL), port, graph.active)
        else:
            output=getTraceroute6(str(target),int(maxTTL), port, graph.active)
        #one target

        result=GridLayout(cols=2, spacing=20)
        BoxLayout.add_widget(result)

        g1=GridLayout(cols=1, spacing=30, size_hint_y=None)
        g1.bind(minimum_height=g1.setter('height'))
        g1.add_widget(Label())

        if isinstance(output, basestring):
            g1.add_widget(Label(text="ERROR :"+str(output), color=(255, 0, 0, .9), bold=True))
```

```
        else:
            g1.add_widget(Label(text='LIST OF DEVICES :', color=(255, 0, 0,.5), bold=True))
            output=output[3:]
            for i in range (0, len(output)):
                outputStrings=output[i].split()
                g_temp=GridLayout(cols=3*count_targets)
                for x in outputStrings:
                    g_temp.add_widget(Label(text=str(x)))
                g1.add_widget(g_temp)
        sw = ScrollView(size_hint=(None, None), size=(700,150))
        sw.add_widget(g1)
        result.add_widget(sw)

        def on_button_press(instance):

            #ak stlacime tlacidlo Graf
            help_layout=FloatLayout()
            bubble = Popup (title='TRACEROUTE GRAPH :',content=help_layout,
auto_dismiss=False,size_hint=(None, None), size=(800, 700), pos_hint={'x':0.1, 'y':0})

            self.add_widget(bubble)
            close_button = Button(size_hint=(None, None),size=(32,32), pos_hint={'y':1.04, 'x':.99},
spacing=0)
            close_button.background_normal = "img/close_icon.png"
            close_button.background_down = "img/close_icon.png"
            def close_bubble(instance):
                self.remove_widget(bubble)

            help_layout.add_widget(close_button)
            close_button.bind(on_press=close_bubble)
            graph_image=Image(source="tmp/output.png",pos_hint={'x':0, 'y':0})
            graph_image.reload()
            help_layout.add_widget(graph_image)

        g2=GridLayout (cols=2,rows=1, spacing=30)
```

```
if graph.active:
    graph_button=Button( size_hint=(None, None), size=(128, 128))
    graph_button.background_normal="img/graph_bordered.png"
    graph_button.background_down="img/graph_bordered2.png"
    g2.add_widget(graph_button)
    graph_button.bind(on_press=on_button_press)

g2.add_widget(Button(text='Save to PDF', size_hint=(None, None), size=(128, 128)))

result.add_widget(g2)
```

Príloha C – Testovací modul – generická trieda

```
import logging
from scapy.layers.inet6 import traceroute6, IPv6, IPv6ExtHdrHopByHop,\
    IPv6ExtHdrDestOpt, IPv6ExtHdrRouting, IPv6ExtHdrFragment
from scapy.layers.inet import traceroute, ICMP
from scapy.layers.inet6 import ICMPv6TimeExceeded
from StringIO import StringIO
from scapy.layers import inet6
logging.getLogger("scapy.runtime").setLevel(logging.ERROR)
from scapy.all import *
from scapy.layers.inet import IP, TCP, UDP, IPV6_ADDR_GLOBAL

def createIPv6():
    return IPv6()

def createHopByHopHeader():
    return IPv6ExtHdrHopByHop()

def createDestinationOptionsHeader():
    return IPv6ExtHdrDestOpt()

def createRoutingHeader():
    return IPv6ExtHdrRouting()

def createFragmentationHeader():
    return IPv6ExtHdrFragment()

def setIPv6(ipv6_object, src, dst, plen, fl, tc, hlim, nh):
    ipv6_object.src=src
    ipv6_object.dst=dst
    ipv6_object.plen=plen
    ipv6_object.fl=fl
    ipv6_object.tc=tc
```

```
    ipv6_object.hlim=hlim
    ipv6_object.nh=nh
    return ipv6_object

def setHopByHopHeader(hbh_object, nh, length, autopad, options):
    hbh_object.nh=nh
    hbh_object.len=length
    hbh_object.autopad=autopad
    hbh_object.options=options
    return hbh_object

def setDestinationOptionsHeader(dstopt_object, nh, length, autopad, options):
    dstopt_object.nh=nh
    dstopt_object.len=length
    dstopt_object.autopad=autopad
    dstopt_object.options=options
    return dstopt_object

def setRoutingHeader(rh_object, nh, length, rhtype, segleft, reserved, addresses):
    rh_object.nh=nh
    rh_object.len=length
    rh_object.type=rhtype
    rh_object.segleft=segleft
    rh_object.reserved=reserved
    rh_object.addresses=addresses
    return rh_object

def setFragmentationHeader(fh_object, nh, res1, offset, res2, m, idfrag):
    fh_object.nh=nh
    fh_object.res1=res1
    fh_object.offset=offset
    fh_object.res2=res2
    fh_object.m=m
    fh_object.id=idfrag
    return fh_object
```

```
def sendFinalPacket(header_list):
    s=header_list[0]
    for x in header_list:
        if header_list.index!=0:
            s=s/x
    send(s)
```