VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

GENERÁTOR IPV6 PREFIXOVÝCH SAD

IPV6 PREFIX SETS GENERATOR

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **KIRILL UTKIN**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JIŘÍ MATOUŠEK, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačových systémů (UPSY) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Utkin Kirill**
Program: Informační technologie
Název: **Generátor IPv6 prefixových sad**
IPv6 Prefix Sets Generator
Kategorie: Počítačové sítě

Zadání:

1. Nastudujte princip generování IPv6 prefixů popsaný v článku: K. Zheng, B. Liu. *V6Gene: A Scalable IPv6 Prefix Generator for Route Lookup Algorithm Benchmark.* Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA'06), 2006.
2. Proveďte implementaci nastudovaného přístupu v jazyce Python.
3. Seznamte se s pravidly pro přidělování IPv6 adres.
4. Navrhněte vlastní způsob generování IPv6 prefixů, který bude využívat znalost pravidel pro přidělování IPv6 adres.
5. Navržený způsob generování implementujte v jazyce Python.
6. Ověřte vlastnosti implementovaného generátoru a srovnejte je s vlastnostmi přístupu prezentovaného v článku z bodu 1 zadání.
7. V závěru diskutujte možnosti dalších vylepšení Vámi navrženého a implementovaného generátoru.

Literatura:

- K. Zheng, B. Liu. *V6Gene: A Scalable IPv6 Prefix Generator for Route Lookup Algorithm Benchmark.* Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA'06), 2006.
- RIPE Network Coordination Centre. *IPv6 Address Allocation and Assignment Policy* [online]. Rev. 21 May 2012 [cit. 2012-10-31]. K dispozici na: <http://www.ripe.net/ripe/docs/ipv6policy.html>.
- American Registry for Internet Numbers. *ARIN Number Resource Policy Manual* [online]. Rev. 31 July 2012 [cit. 2012-10-31]. K dispozici na: <https://www.arin.net/policy/nrpm.html>.
- APNIC. *APNIC Internet Number Resource Policies* [online]. Version 005, December 2017 [cit. 2018-09-06]. K dispozici na: <https://www.apnic.net/community/policy/resources>.

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Matoušek Jiří, Ing., Ph.D.**
Vedoucí ústavu: Sekanina Lukáš, prof. Ing., Ph.D.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 26. října 2018

Abstrakt

Vzhledem k aktivnímu rozvoje protokolu IPv6 dochází k nárůstu počtu IPv6 prefixů ve směrovacích tabulkách. Vzhledem k tomuto faktu je zapotřebí se vyvíjet nové vyhledávací algoritmy. Avšak, efektivní testování takových algoritmů je závislé na velikosti prefixových sad, které v dnešní době se skládají jen z malého počtu záznamů. Cílem této bakalářské práce je návrh a implementace generátoru IPv6 prefixů, jenž bude vycházet z alokačních politik pro přidělování adresového prostoru. Návrhu vlastního generátoru předcházelo nastudování a implementace generátoru V6Gene. S cílem ověření funkčnosti implementovaných generátorů byly použity metody srovnání rozložení délek a hodnot úrovní prefixů mezi reálnou a vygenerovanou sadou prefixů. Nakonec generátory byly porovnány z pohledu rychlosti generování a paměťové spotřeby.

Abstract

Due to the fast adoption of IPv6 protocol, number of IPv6 prefixes in routing tables are incessantly increasing. Based on this fact, development of new lookup algorithms is required. However, testing of those algorithms is highly dependent on size of datasets, which are not large enough for this purpose at the moment. Design and implementation of generator of IPv6 prefix sets, which will be based on currently using address allocation policies, is the main goal of this bachelor's thesis. Implementation of generator was preceded by study and implementation of the generator V6Gene. Validation of generated datasets were performed by comparing length distribution and level distribution of prefixes with the real world datasets. Finally, speed of the generating process and memory usage were compared for implemented generators.

Klíčová slova

generátor prefixových sad IPv6, IPv6 prefix, IPv6 adresa, pravidla přidělování IPv6 adres, V6Gene

Keywords

IPv6 prefix set generator, IPv6 prefix, IPv6 address, IPv6 address assignment policies, V6Gene

Citace

UTKIN, Kirill. *Generátor IPv6 prefixových sad.* Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jiří Matoušek, Ph.D.

Generátor IPv6 prefixových sad

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jiřího Matouška, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Kirill Utkin
16. května 2019

Poděkování

Rád bych poděkoval Ing. Jiřímu Matouškovi, Ph.D. za odborné rady a vstřícnost při konzultacích v průběhu řešeni této bakalářské práce.

Obsah

Kapitola 1

Úvod

Protokol IPv4 (z angl. *Internet Protocol version 4*) byl vyvinut na počátku sedmdesátých let 20 století, aby byla umožněna snadnější komunikace a sdíleni informaci mezi vládními výzkumníky a akademiky ve Spojených státech amerických. V těchto letech počet zařízení připojených v sítí nebyl příliš velký, a proto autoři protokolu nevěnovali dostatečnou pozornost bezpečnostním požadavkům a QoS (z angl. *Quality of Service*). IPv4 adresa má délku 32 bitů (nebo 4 bajty) a tato délka je postačující k vytvoření 2^{32} unikátních adres podporovaných protokolem IPv4. Nutné je však poznamenat, že určitá část z těchto adres je rezervována pro vnitřní potřeby protokolu a tudíž nejsou globálně směrovatelné v rámci Internetu [10]. Kromě toho dřívější metody přidělovaní IPv4 adres neprokázaly dostatečnou efektivitu, vzhledem k tomu, že kvůli omezujícímu způsobu dělení adresového prostoru s využitím tříd docházelo ke vzniku velkého množství nevyužitých IPv4 adres v jednotlivých nově připojovaných sítích. To vedlo k rychlému vyčerpaní adresového prostoru IPv4 a v roce 2011 došlo k přidělení posledních adresových bloků IPv4 adres jednotlivým regionálním registrátorům (RIR, z angl. *Regional Internet Registry*) [15]. Je také očekávané, že ke konečnému vyčerpaní adresového prostoru, který distribují regionální registrátoři, už může dojít v průběhu následujících dvou let [9]. Problém nedostatku IPv4 adres byl dočasně vyřešen zrušením tříd IP adres, přidělováním délek prefixu sítí podle reálné potřeby a překladem privátních adres na omezený počet veřejných adres pomocí mechanismu NAT (z angl. *Network Address Translator*) [7]. Nakonec rychlý růst počtu síťových zařízení a aktivní vývoj mobilních sítí, vedly k vývoji protokolu nové generace.

Na začátku 90. let byl navržen a následně vyvíjen protokol IPv6 (z angl. *Internet Protocol version 6*), který vyřešil výše zmíněné nedostatky a rozšířil vlastnosti protokolu IPv4. Nová verze protokolu výrazně zvyšuje dostupný adresový prostor. Adresy protokolu IPv6 mají délku 128 bitů, což umožňuje vytvořit celkem 2^{128} unikátních adres.

Vzhledem k tomu, že komunikující stanici stanice mohou nacházet v různých sítích, je nezbytné určitým způsobem směrovat síťový provoz. Proces směrování síťového provozu slouží pro vyhledávání nejoptimálnější cesty a přenos paketu od zdrojové stanice ke stanice cílové. Na základě IP adresy cílové stanici probíhá vyhledávání tzv. nejdelšího shodného prefixu (angl. *Longest Prefix Match*) ve směrovací tabulce směrovače. Tím pádem výkonnost vyhledávacího algoritmu je závislá na délce uložených prefixů ve směrovací tabulce. Vzhledem k tomu, ze IPv6 adresy mají čtyřikrát větší délku, vyhledávací algoritmy používající pro vyhledávání prefixů v tabulkách složených z IPv4 adres nevykazují dostatečnou efektivitu vyhledávání ve směrovacích tabulkách IPv6 [22]. Avšak vývoj nových vyhledávacích algoritmů vyžaduje sady prefixů pro testování výkonnosti. Vzhledem k tomu, že protokol

IPv6 stále se nachází na ranní etapě rozvoje, současné směrovací tábulky IPv6 nemohou poskytnout dostatečné množství záznamů pro účely testování.

Hlavním cílem této bakalářské práce bylo nastudovat princip generování sad prefixů pomocí nástroje V6Gene a navrhnout vylepšení tohoto řešení na základě znalosti politik přidělování adresového prostoru. Teoretická část práce je dělena do kapitol 2 a 3. Kapitola 2 se zabývá detailnějším popisem protokolu IPv6, především způsobu reprezentaci IPv6 prefixu. Také je zde ukázaná hierarchie internetových registrátorů a podrobné rozebraný alokační politiky pro přidělování adresového prostoru IPv6. Kapitola 3 je rozdělena na dvě části. První část této kapitoly se věnuje popisu datové struktury trie, která se používá pro reprezentaci adresového prostoru ve směrovacích tabulkách. Zde jsou také vysvětleny vlastností popsané struktury, jelikož jejich pochopení je nezbytně k provádění analýzy sad prefixů a realizaci celé praktické části této práce. V závěru první části jsou také vysvětleny základní operace, které je se stromovou strukturou možné provádět. Druhá část této kapitoly se zabývá vysvětlením postupů pro provádění analýzy prefixových sad, prováděním samotnou analýzou a vyhodnocením dosažených výsledků. Úkolem praktické části bakalářské práce je implementace generátorů IPv6 prefixů. Implementaci generátoru dle vlastního návrhu uvedené v kapitole 5, předchází kapitola 4, která popisuje vlastností a implementaci generátoru V6Gene. Kapitola 6 se zabývá problematikou testování implementovaných nástrojů.

Kapitola 2

Protokol IPv6

2.1 Textová reprezentace IPv6 adresy

IPv6 adresy mají délku 128 bitů a k jejich zápisu se používají šestnáctková čísla po čtveřicích oddělených dvojtečkami, což znamená že každé 4 bity mohou být reprezentovány jednou hexadecimální číslicí v rozsahu od 0_{16} $\left[0000_2\right]$ až do f_{16} $\left[1111_2\right]$. Existují několik možnosti řetězcové reprezentace IPv6 adresy mezi které patří [8]:

1. Nejčastěji používaná forma reprezentace IPv6 adresy je `x:x:x:x:x:x:x:x`, kde symbol '`x`' označuje skupinu, skládající se z jedné až čtyř číslic ve šestnáctkové soustavě. Příklad zápisu adresy v takové podobě je následující:

   ```
   2001:0db0:0000:123a:0000:0000:0000:0030
   ```

   Zápis úvodních nulových bitů u jednotlivých čtveřic není podmínkou, je však nutné uvést minimálně jednu číslici v rámci jedné skupiny (výjimkou je případ popsaný v dalším bodě). Na základě tohoto pravidla, výše uvedený zápis IPv6 adresy může být upraven do následujícího tvaru:

   ```
   2001:db0:0:123a:0:0:0:30
   ```

2. Po zkrácení adresy uvedeným způsobem může dojít ke vzniku jedné nebo několika po sobě jdoucích skupin, skládajících se pouze z nulových hodnot bitů. Daná posloupnost může být nahrazena dvojicí dvojteček, avšak pouze jednou, aby bylo možné jednoznačné zjistit, kolik bitů bylo zkráceno a převést zkrácenou adresu do původní podoby v případě potřeby. Kupříkladu, IPv6 adresa ve tvaru:

   ```
   2001:db0:0:123a:0:0:0:30
   ```

   může být přepsaná do následujícího tvaru:

   ```
   2001:db0:0:123a::30
   ```

Vzhledem k velkému množství způsobu IPv6 byla navržená řada doporučení. Tyto doporučení definuji tzv. kanonický tvar IPv6 adresy. Pravidla pro zápis adresy dle kanonického tvaru jsou definovaný následovaně [11]:

- Velikost písmen používaných pro označení skupin bitů IPv6 adresy v šestnáctkové soustavě nehraje roli, tehdy velká a malá písmena jsou si ekvivalentní. Je však doporučeno využití malých písmen.
- Zkratka "`::`" musí být použita pro nejdelší skupinu nulových bitů. IPv6 adresa může také obsahovat dvě skupiny nulových bloků stejně délky. V tomto případě zkrátka "`::`" musí být použita pro první skupinu zleva.
- Počáteční nuly čtveřice vždy mají být vynechany.

2.2 Textová reprezentace IPv6 prefixu

Textová reprezentace IPv6 prefixu se zapisuje podobným způsobem jako IPv4 prefix v CIDR (z angl. *Classes Inter-Domain Routing*) notaci. Adresování CIDR poskytuje více flexibilní způsob dělení sítě na jednotlivé podsítě. Zapsaný adresový prefix s využitím této notaci se skládá ze dvou základních částí [8]:

`ipv6 adresa` / `délka prefixu`

Význam jednotlivých částí je následující:

- `ipv6 adresa` - reprezentace IPv6 adresy v textové podobě. Představuje prefix, který určuje celou síť nebo podsíť. Detailnější popis textové reprezentaci uveden v podkapitole 2.1.
- `délka prefixu` - nezáporné celé číslo v desítkové soustavě, které určuje počet významných bitů IPv6 adresy.

Zbývajících 128 - d bitů, kde d označuje délku prefixu, slouží pro označení identifikátoru rozhraní v rámci dané sítě.

2.3 Typy IPv6 adres

Podobně jako u protokolu IPv4, protokol IPv6 používá adresy typu unicast a multicast. V porovnaní s protokolem IPv4, není v IPv6 definovaný adresy typy broadcast. Úlohu broadcastových adres u protokolu IPv6 plní adresy multicastové [8]. Novinkou protokolu IPv6 jsou adresy typu anycast.

Unicast

Stejné jako u protokolu IPv4, adresa typu unicast se používá pro jednoznačnou identifikaci síťového rozhraní v rámci sítě. U protokolu IPv6 rozlišujeme čtyři další typy unicastových adres:

- **Loopback** - speciální adresa mající tvar ::1/128, která představuje zpětnovazební smyčku a používá se pro komunikaci stanice sama s sebou. Adresa typu loopback nemusí být přiřazená žádnému síťovému rozhraní v síti. U IPv4 adresa `127.0.0.1` má stejný význam.

- **Global Unicast** - adresa, která jednoznačně identifikuje síťové rozhraní v rámci celého internetu. Přidělování těchto adres probíhá z adresového prostoru `2000::/3` (001 v binární notaci).
- **Link-Local Unicast** - adresa, která plní úlohu privátních adres a má prefix `fe80::/10`. Adresy typu anycast se používají pouze v rámci lokální sítě nebo fyzické linky a nejsou směrovatelné. Každé síťové rozhraní má přidělenou adresu typu link-local. Tento typ adres je hraje důležitou roli v protokolu IPv6, neboť se používají pro automatickou konfiguraci zařízení v koncových sítích, objevování sousedů nebo pro komunikaci s dalšími zařízeními v rámci lokální sítě.
- **Unspecified** - speciální adresa mající tvar `::/128`, která nemusí být přiřazena žádnému síťovému rozhraní. Používá se jako zdrojová IP adresa rozhraní, kterému nebyla ještě přidělena IPv6 adresa. Pakety s nespecifikovanou zdrojovou adresou nejsou směrovatelné směrovačem.

Multicast

Adresy typu multicast identifikují skupinu síťových rozhraní. Paket zaslaný na tuto adresu bude doručen všem zařízením patřícím do odpovídající skupiny. Adresy typu multicast tvoří obecný prefix `ff::/8`.

Anycast

Podobně jako adresy typu multicast, anycast adresa specifikuje množinu rozhraní. Však, paket zaslaný na tuto adresu je doručen pouze jednomu konkrétnímu síťovému rozhraní, které z pohledu principů směřování je nejbližší. Přiřazování adres tohoto typu probíhá z rozsahu adres typu global unicast (tudíž z adresového prostoru reprezentovaného prefixem `2000::/3`).

2.4 Přidělování adresového prostoru IPv6

Na začatku této podkapitoly jsou definovany cíle správy adresového prostoru IPv6, které mají být splněny, aby bylo možné předejít neefektivní distribuci adresového prostoru (viz 2.4.1). Samotný proces přidělování adres má komplexní charakter a může být rozdělen na dvě části. Nejprve je nutné provést přidělení prefixu sítě. Takový prefix může být přidělen pouze odpovídajícím internetovým registratorem. Internetový registrátor je účastníkem internetové komunity, který je zodpovědný za distribuci adresového prostoru svým uživatelům nebo dalším internetovým registrátorům (viz 2.4.2). Také je nezbytné provádět přidělení adres koncovým zařízením v rámci sítě. Tuto činnost možné provést za pomocí mechanismu autokonfiguraci (viz 2.4.4).

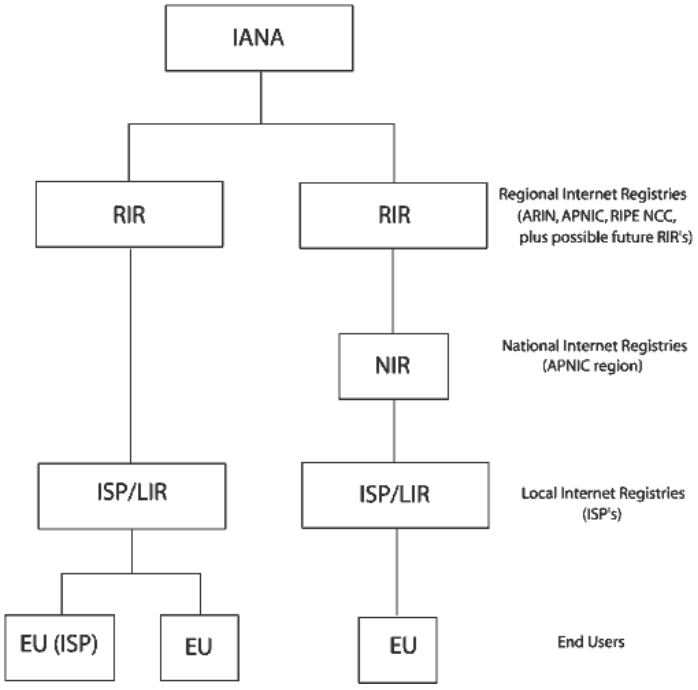
2.4.1 Cíle správy adresového prostoru IPv6

Adresový prostor IPv6 je veřejným zdrojem, který vyžaduje předvídatelné spravování s ohledem na dlouhodobý rozvoj internetu. Zodpovědné řízení zahrnuje řadu cílů, se kterými je nutné počítat při přidělování adresového prostoru. Mezi tyto cíle patří [21]:

- **Unikátnost** - každé přiřazení a/nebo přidělování adresového prostoru musí být celosvětově unikátní.
- **Registrace** - přidělený adresový prostor musí být registrován v databázi, která je přístupná pro jednotlivé členy internetové komunity, mezi které se řadí internetové registrátory. Registrace je nezbytná pro zajištění unikátnosti při přidělování adresového prostoru, však registrace by měla probíhat v souladu s dodržováním ochrany osobních údajů registrovaných entit.
- **Agregace** - adresový prostor musí být hierarchicky rozdělován s ohledem na topologii sítí. Politiky jednotlivých internetových registrátorů by měly vyhledat možnost s co nejmenší fragmentací adresového prostoru.
- **Konzervativnost** - ačkoli IPv6 adresový prostor je extrémně velký, internetový registrátoři by se měli vyvarovat přidělování zbytečné velkého adresového prostoru. Všechny požadavky na přidělení by měly být detailně zdůvodněny v odpovídající dokumentaci.
- **Spravedlnost** - všechny politiky, týkající se přidělování veřejného adresového prostoru, by měly být rovnoprávné a poctivé pro existující a budoucí účastníky internetové komunity bez ohledu na jejich národnost, velikost, lokaci atd.
- **Minimalizování režie** - Adresový prostor by měl být přidělen tak, aby v budoucnu nedocházelo ke vzniku velkého množství požadavků na dodatečné přidělení.

2.4.2 Hierarchie internetových registrátorů

V současně době existuje několik úrovní hierarchií internetových registrátorů. Kořenovým registrátorem v hierarchii je organizace IANA (z angl. *Internet Assigned Numbers Authority*), která přiděluje adresové bloky IP adres jednotlivým regionálním registrátorům. Regionální registrátoři pak distribuuji adresový prostor mezi jednotlivé lokální (LIR, z angl. *Local Internet Registry*) nebo nacionální (NIR, z *angl. National Internet Registry*) registrátory. NIR je organizace, která přiděluje adresové bloky určité délky lokálním registrátorům na státní úrovni a vyskytuje se převážně v asijsko-pacifickém regionu [21]. Lokální registrátory obvykle plní role ISP (z angl. *Internet Service Provider*) a poskytují koncovým uživatelům (EU, z angl. *End User*), případně menším ISP, přístup do sítě Internet. Na obrázku 2.1 je uvedeno, jak vypadá hierarchická struktura registrátorů. Uspořádaní různých typu registrátorů do hierarchie způsobí agregovatelnost přidělených prefixů.



Obrázek 2.1: Hierarchické uspořádaní internetových registrátorů [21]

2.4.3 Alokační politiky přidělovaní adresového prostoru IPv6

Jak bylo popsané v předchozí podkapitole, existuje hierarchie internetových registrátorů zapojených do procesu přidělování adresového prostoru. Tato podkapitola se zabývá vysvětlením alokačních politik jednotlivých registrátorů, jež se nacházejí na různých úrovních této hierarchie.

IANA

IANA je celosvětová organizace, která je řízena dobročinnou organizaci ICAAN (z angl. *Internet Corporation for Assigned Names and Number*). IANA plní 3 základní funkce, mezi které patří [20]:

- **Správa adresového prostoru** - IANA spravuje adresový prostor IPv4 a IPv6 a přiděluje adresové bloky IP adres jednotlivým RIR.
- **Správa DNS** - IANA spravuje kořenové servery systému DNS (z angl. *Domain Name System*) a spravuje domény .int, arpa a řadu dalších.
- **Správa protokolů** – IANA upravuje a publikuje parametry IETF (z angl. *Internet Engineering Task Force*) protokolů a přiděluje unikátní jména a čísla pro internetové protokoly.

Alokační politiky přidělovaní adresového prostoru, který spravuje IANA, organizacím typu RIR jsou definovaný následujícím způsobem [19]:

- Minimální velikost IPv6 adresového bloku, který získává RIR od IANA, je `/12`.

- IANA přidělí dostatečné velký adresový prostor na základě požadavků RIR. Velikost přiděleného adresového prostoru by měla být dostačující pro splnění kladených požadavků v době následujících 18 měsíců.
- IANA dovoluje RIR definovat vlastní politiky přidělování IPv6 adres svým klientům tak, aby bylo zajištěny cíle, uvedené v podkapitole 2.4.1

RIR má právo požádat o přidělení dodatečného adresového prostoru od IANA v následujících případech:

- Dostupný adresový prostor, který RIR už má k dispozici, je méně než 50% z původní předěleného `/12` bloku.
- Velikost volného adresového prostoru IPv6 adres je menší než adresový prostor, který je potřeba přidělit v průběhu následujících 9 měsíců.

IANA přiděluje bloky adres jednotlivým RIR z rozsahu adres typu global unicast `2000::/3`.

RIR

Regionální registrátory jsou organizace, které spravují a distribuují veřejný adresový prostor získaný od organizaci IANA v rámci příslušných regionů. Současně existuje pět regionálních registrátorů mezi které patří:

- **AFRNIC** (z angl. *African Network Information Center*) - Afrika.
- **ARIN** (z angl. *American Registry for Internet Numbers*) - Severní Amerika.
- **APNIC** (z angl. *Asia-Pacific Network Information Centre*) - Asie, Austrálie a Océanie.
- **LACNIC** (z angl. *Latin America and Caribbean Network Information Centre*) - Střední a Jižní Amerika.
- **RIPE NCC** (z ang.l *Réseaux IP Européens Network Coordination Centre*) - Evropa, Rusko a Blízký východ.

Kvůli tomu, že alokační politiky jednotlivých RIR se významně liší v určitých bodech, dále budou popsány alokační politiky, které jsou definovany organizací RIPE RCC [21]:

- Minimální velikost IPv6 adresového bloku, který přiděluje RIR svým klientům, je stanovena na `/32.`
- Pro vyžádaní o přidělení adresového prostoru od RIR je podmínkou plnit úlohu LIR. Žádající registrátor musí také ukázat alokační plán přidělování adresového prostoru jiným organizacím a/nebo EU na následující dva roky.

LIR může požádat o dodatečné přidělení adresového prostoru od RIR. Dojde-li k dodatečnému přidělení adresového prostoru, bude přidelený adresový blok mít stejnou veikost jako již přidělený. Při dodatečném přidělování RIR bude snažit přidělit adresový blok, který bude navazovat na už přidělené adresy. Podminky pro schválení požadavku na přidělení dodatečného prostoru jsou stanoveny následovaně:

- Registrátor LIR musí být schopen ukázat, že již přidělený adresový prostor je nedostatečný. Takový požadavek musí být doprovázen přiložením odpovídající dokumentaci.
- K zisku dodatečného přidělení adresového prostoru je nezbytné, aby žádající o tuto činnost registrátor dodržoval hodnotu HD (z angl. *Host-Density*) poměru rovnou 0.94 nebo větší. Tato hodnota slouží pro měření efektivity přiřazení IPv6 adres. Pro výpočet hodnoty HD-poměru se používá vzorec [6]:

$$HD - poměr = \frac{log(počet\ přidělených\ IPv6\ adres)}{log(maximálni\ počet\ přidělitelných\ IPv6\, adres)}$$

LIR / NIR

Internetový registrátor LIR provádí přidělení adresového prostoru EU nebo menším ISP. Neexistují žádná pravidla pro přidělování adresového prostoru, který přiděluje LIR svým klientům. Ačkoli každá LIR může definovat vlastní politiky pro přidělování, je stanoveno, že adresový blok přidělený koncovému uživateli musí mít velikost alespoň `/64`. Zároveň s tím, každé přidělení adresového bloku kratšího než `/48` musí být registrováno LIR nebo podřízeným ISP u odpovídajícího RIR pro provádění přepočtu hodnoty HD-poměru.

EU

Koncový uživatel je entitou, která pouze získává adresový prostor od jiných registrátorů, nacházejících na vyšších úrovních hierarchii. Tato entita se nachází v obchodních nebo právních vztazích s poskytovatelem služeb. Příkladem služeb může například být přidělení adresového prostoru nebo poskytnutí přístupu do Internetu.

2.4.4 Automatická konfigurace IPv6 adres

Po připojení zařízení k síti je nezbytně získat informaci o parametrech, za pomocí kterých by bylo možné získat přistup do sítě Internet a provádět veškerou komunikaci. Potřebné informace mohou být získaný za pomocí mechanismu autokonfiguraci. Protokol IPv6 nabízí dvě možné varianty automatického přidělování IP adresy koncovému zařízení.

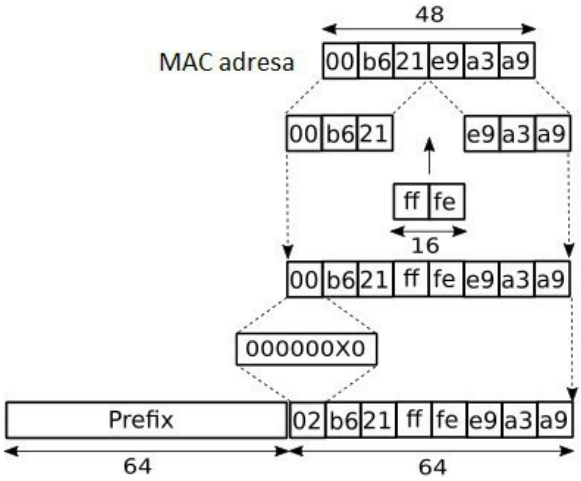
Stavová konfigurace

Při stavové konfiguraci IPv6 adres se používá protokol DHCPv6 (z angl. *Dynamic Host Configuration Protocol version 6*) a DHCPv6 server, který uchovává informace o připojených zařízeních a dostupných adresách v rámci dané síti. DHCPv6 server poskytuje IP adresu, adresu výchozí brány, délku prefixu sítě, informace o DNS serverech a řadu dalších parametrů. Rozlisujeme tři základní režimy fungovaní DHCPv6 [5]:

- **Dynamické přidělování** - adresa je přidělena klientovi na určitý časový interval. Klient však má kdykoliv možnost požádat o nové přidělení.
- **Automatické přidělování** - adresa je přiřazena klientovi na dobu neurčitou a za předem definovaného adresového prostoru.
- **Statické přidělování** - klient zvolí IP adresu sám a informuje o tom DHCPv6 server.

Bezstavová konfigurace

Na rozdíl od stavové konfigurace, bezstavový způsob konfigurace SLAAC (z angl. Stateless auto configuration) nevyžaduje DHCPv6 servery v síti a umožňuje klientovi vygenerovat adresu na základě vlastní MAC (z angl. *Media Access Control Address*) adresy síťového rozhraní a informací získaných od směrovačů. Směrovač v pravidelných intervalech šíří tzv. RA (z angl. *Router Advertisement*) zprávy na adresu odpovidajíci adrese typu multicast, určenou pro všechny uzly v rámci odpovídající sítě. Tyto zprávy obsahují prefix dané sítě. Na základě těchto údajů vznikne nová adresa, ve které první 64 bity tvoří prefix sítě získané z RA zprávy, zbývající bity jsou reprezentovány pomocí EUI-64 (z angl. *Extended Unique Identifie*) identifikátoru. EUI-64 je identifikátorem odvozeným ze 48bitové MAC adresy síťového rozhraní klienta. Takový 64-bitový identifikátor vzniká vložením dvou oktetu `ff-fe` uprostřed MAC adresy síťového rozhraní. Poté bude sedmý bit zleva invertovan na opačnou hodnotu. Hodnota tohoto bitu slouží pro označení globální jednoznačností EUI-64 identifikatora. Proces vytvoření EUI-64 identifikátoru je znázorněn na obrázku 2.2.



Obrázek 2.2: Automatická konfigurace adresy IPv6 [1]

Unikátnost vytvořené adresy je pak ověřena za využití NDP (z angl. *Neighbor Discovery Protokol*). Pokud vygenerovaná adresa je již přidělena jinému zařízení v sítí, pak klient nemá možnost vytvořenou adresu nastavit na vlastní internetové rozhraní a musí čekat na další RA zprávu od směrovače. Případně klient má právo požádat o opakované vysílání RA zprávy zasíláním zprávy typu RS (z angl. *Router Solicitation*).[12][14]

Kapitola 3

Reprezentace a analýza adresového prostoru IPv6

Tato kapitola se zabývá popisem reprezentace adresového prostoru ve směrovačích a analýzou veřejné dostupných prefixových sad. V první části kapitoly je popsaná datová struktura, která může být použita pro ukládání prefixů ze směrovacích tabulek, a základní vlastnosti jejích prvků. Druhá část kapitoly se věnuje popisu metod analýzy skutečných sad IPv6 prefixů, které vycházejí ze znalosti vlastností struktury popsané v první části této kapitoly. Na konci kapitoly jsou uvedeny důvody pro návrh generátoru IPv6 prefixů.

3.1 Reprezentace adresového prostoru ve směrovačích

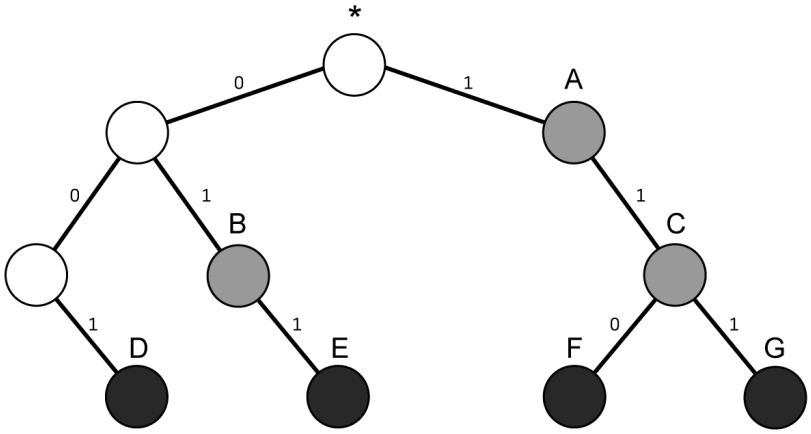
Zatímco adresový prefix se zapisuje v hexadecimální podobě, pro uložení hodnoty prefixu ve směrovacích tabulkách se používá binární reprezentace. Aby bylo možné urychlit proces vyhledávání odpovídající hodnoty prefixu ve smerovací tabulce směrovače a efektivní provádět aktualizací záznamů umistených v této tabulce, je nezbytné zvolit vhodnou datovou strukturu. Taková struktura se jmenuje trie a je popsaná (včetně její vlastností) v rámci této podkapitoly.

3.1.1 Datová struktura binární prefixový strom

Binární prefixový strom neboli trie je dynamická stromová datová struktura, která je tvořená množinou uzlů. Název struktury vychází z anglického slova *retrieval*, což znamená vyhledávání. Tato struktura a její modifikace se často používají pro uložení prefixů ve směrovacích tabulkách. Ve vrcholu stromu je umístěn uzel, který se jmenuje kořen. Tento uzel reprezentuje libovolný prefix. Uzel označený jako kořen má nulovou hloubku a je jediným uzlem stromu který nemá žádného rodiče. Každý uzel uložený ve stromu má maximálně dva potomky, kterým říkáme levý a pravý potomek. Jednotlivé bity jsou reprezentovány hranou, kde hrana vlevo reprezentuje nulovou hodnotu bitu a hrana vpravo reprezentuje bit s hodnotou jedna. Na obrázku 3.1 je vyobrazen binární prefixový strom reprezentující sadu prefixů uvedenou v tabulce 3.1.

Označení uzlu	Prefix
A	1*
B	01*
C	11*
D	001*
E	011*
F	110*
G	111*

Tabulka 3.1: Ukázková sada prefixů uspořádaná od nejkratšího prefixu k nejdelšímu



Obrázek 3.1: Prefixový strom, sestavený na základě ukázkové prefixové sady.

Ve stromu se mohou vyskytovat tři základní typy uzlů. Uzel, který nemá žádného dalšího potomka, se nazývá prefixovým listovým uzlem (z angl. *Prefix leaf node*) a je označen na obrázku 3.1 černou barvou. Uzly reprezentující uložené prefixy ve stromu, ale mající odkaz na dalšího potomka, jsou označeny šedou barvou a nazývají se prefixové uzly (z angl. *Prefix node*). Posledním typem uzlů jsou tzv. vnitřní uzly (z angl. *Internal node*), které jsou označeny na obrázku 3.1 bílou barvou. Na rozdíl od listových a prefixových uzlů, vnitřní uzly nereprezentují žádný prefix uloženy ve stromu, ale jen jeden bit prefixu a pouze odkazují na svého potomka. Tím pádem je prefix uložený ve stromové struktuře reprezentován cestou od kořenového uzlu až do určitého prefixového uzlu. Pro každý prefixový a prefixový listový uzel lze určit jeho hloubku a hodnotu úrovně v rámci stromové struktury. Kvůli tomu, že tyto vlastnosti jsou velmi důležité pro zkoumání struktury adresového prostoru, je nutné jejich popsat detailněji.

- **Hloubka** prefixového uzlu je dána počtem předchůdců na cestě z kořene do hledaného uzlu ve stromu. Označení |Depth(i)| pak slouží pro vyjádření počtu prefixů, které mají hloubku rovnou hodnotě i. Jinými slovy, hloubka prefixového uzlu udává délku prefixu uloženého ve stromu [22]. Například, uzly stromu z obrázku 3.1 označené písmeny `A`, `B` a `D` mají hloubkou 1, 2 a 3. Jak už bylo zmíněno dříve, kořen stromu, který je označen symbolem `"*"`, má hloubku rovnou hodnotě 0. Maximální hloubka celého stromu je určena na základě délky nejdelšího uloženého prefixu. Tehdy hloubka stromu z obrázku 3.1 se rovná 3.

- Další vlastností prefixového uzlu je hodnota udávající **úroveň** uzlu. Ze zvoleného uzlu `A` může vést více cest do prefixových listových uzlů. Ze všech možných cest, zvolíme pouze tu, která obsahuje největší počet prefixových uzlů. Počet těchto uzlů s vyloučením uzlu `A` udává úroveň zvoleného uzlu `A`. Označení |Level(i)| pak definuje počet prefixových uzlů majících hodnotu úrovně rovnou i. Tímto parametrem se dá vyjádřit úroveň hierarchie určité podsítě [22]. Úroveň uzlu `F`, `A` a `C` z obrázku 3.1 se rovná 0, 2 a 1. Zároveň platí, že |Level(0)| = 4 , |Level(1)| = 2 a |Level(2)| = 1.

3.1.2 Operace nad prefixovým stromem

V předchozí podkapitole byla popsána struktura prefixového stromu a vlastnosti jednotlivých uzlů. Tato podkapitola představuje operace, které lze nad stromem provádět.

Základní operací je vyhledávání prefixu ve stromu. Vyhledávání probíhá na základě celé cílové nebo zdrojové IP adresy a hledá se tzv. nejdelší shodný prefix ve stromu, který dané adrese odpovídá. Tato operace probíhá iterativním způsobem po jednotlivých bitech prefixu, přičemž se začíná od kořenového uzlu stromu. Zvolení hrany je určeno v závislosti na aktuální hodnotě bitu prefixu. V průběhu průchodu prefixovým stromem mohou být nalezeny další prefixové uzly. Tyto uzly je třeba ukládat zapamatovány kvůli tomu, mohou reprezentovat nejdelší shodný prefix na cestě. Pokud totiž bude nalezen uzel, který neobsahuje hranu s odpovídající hodnotou bitu, proces vyhledávání se zastaví a bude použit poslední , a tudíž nejdelší zapamatovaný prefix. Uvažujme proces vyhledávání ve stromu z obrázku 3.1 pro adresu začínající 011. Vyhledávání začíná od kořene stromu a na základě hodnoty nejméně významného bitu pokračuje směrem dolů po levé hraně. Jelikož hodnotou dalšího bitu je 1, vyhledávání pokračuje v pravém podstromu. V tento okamžik je nalezen prefixový uzel `F`, který musí být zapamatován jako doposud nejdelší nalezený shodný prefix. Na základě hodnoty třetího bitu vyhledávání pokračuje po pravé hraně. Nalezený uzel I je listovým prefixovým uzlem stromu, a proto bude vyhledávání zastaveno a prefix reprezentovaný tímto uzlem bude označen jako nejdelší shodný prefix.

Vložení nového uzlu do stromu probíhá téměř stejně jen s tím rozdílem, že na konci tohoto procesu dojde k vytvoření nového prefixového uzlu. Pro vložení uzlu `I` do stromu je nutné na začátku vyhledat uzel `F`, který by v tomto případě ještě neobsahoval odkaz na dalšího potomka. Nakonec bude vytvořena nová hrana reprezentující hodnotu bitu 1 a vytvořený uzel bude propojen s uzlem `F`. Vložením prefixu do stromové struktury může dojít k vytvoření nových vnitřních uzlů. Ku příkladu, pro vložení prefixu 101 bude na začátku vyhledán prefixový uzel `B`. Z důvodu, že vyhledaný uzel ještě nemá hranu reprezentující hodnotu bitu nula, dojde k vytvoření této hrany a nového vnitřního uzlu. Z tohoto vnitřního uzlu pak vznikne hrana reprezentující bit s hodnotou jedna a dojde k vytvoření prefixového uzlu, sloužícího pro označení vkládaného prefixu. Pokud vkládaný prefix již existuje ve stromu, operace vložení se zastaví.

Operace odstranění prefixu ze stromu, stejně jako při vkládání, začíná vyhledáváním. Nejprve je nutné nalézt uzel reprezentující odstraňovaný prefix. Pokud takový uzel má alespoň jednoho potomka, postačí označit nalezený uzel jako vnitřní, a tím bude odstranění uzlu dokončeno. V případě, že je prefix reprezentován listovým uzlem, proběhne na začátku odstranění takového uzlu ze stromu a poté budou odstraněny všechny vnitřní uzly, které nemají žádného potomka.

Vzhledem k tomu, že v rámci jedné iterace lze získat pouze jednu hodnotu bitu uloženého prefixu, časová složitost uvedených operací bude mít lineární charakter. Tudíž v nejhorším případě bude nutné provést 128 kroků pro získání uloženého prefixu ve stromové struktuře.

3.2 Analýza IPv6 prefixových sad

Tato podkapitola je věnovaná analýze sad prefixů na základě vlastností popsaných v první části této kapitoly.

3.2.1 Metodika analýzy

Pro provádění analýzy bylo nutné využít sady prefixů získané z reálných směrovacích tabulek. Jako zdroj dat byla zvolena databáze IPv6 adres, kterou spravuje organizace CAIDA [4]. Sady získané z této databáze, se skládají z IPv6 prefixů získaných z databáze adres projektu RouteViews Univerzity v Oregonu [16]. V průběhu analýzy byly prozkoumány vzorky dat z roku 2007[18] a z roku 2019 [17], které jsou uloženy ve složce *CAIDA_datasets*(viz A).

Kvůli tomu, že stažené prefixové sady nebyly vhodně formátované, byl napsán skript v jazyce Python 3, který provádí formátování stažených souborů a jejich uložení do složky *formated_datasets* (viz A). Takto formátované sady prefixů pak mohou být použity jako vstupní parametr jednotlivých generátorů.

K provedení analýzy bylo nutné vytvořit další skript, který zpracuje formátovaný vstupní soubor obsahující IPv6 prefixy a na základě obsahu načteného souboru vytvoří binární prefixový strom. Dále provede výpočet rozložení délek a úrovní načtených prefixů. Pro reprezentaci výsledků jsou pomocí knihovny Matplotlib[1] vytvořeny grafy, které se ukládají do složky *statistics/* dle nazvu zkoumané sady (viz A) jako soubory s příponou .png.

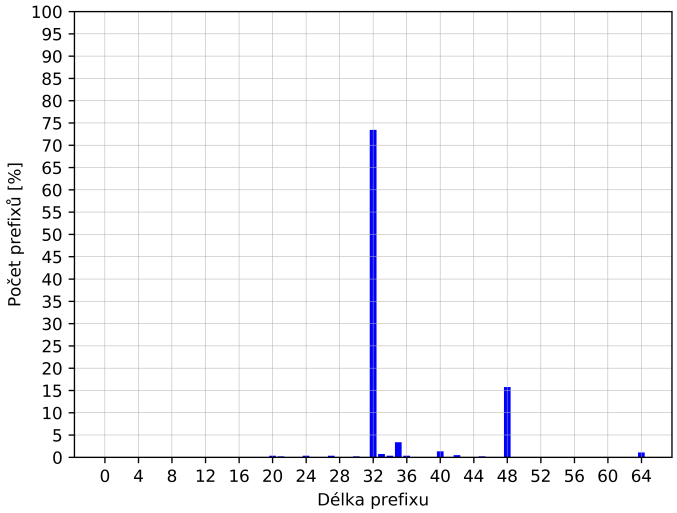
3.2.2 Provedení analýzy

Analyza prefixových sad je rozdělena na dvě základní částí. Každá část se zaměřuje na sledování jiných vlastností. Na začátku je sledováno rozložení délky prefixů v prefixových sadách. Ze získaných informací by mělo být možné posoudit, nakolik jsou jednotlivými registrátory dodržována pravidla pro přidělování IPv6 adres. V rámci druhé části je sledováno rozložení prefixů podle úrovní. Na základě tohoto parametru lze zjistit, jak se mění struktura sítě.

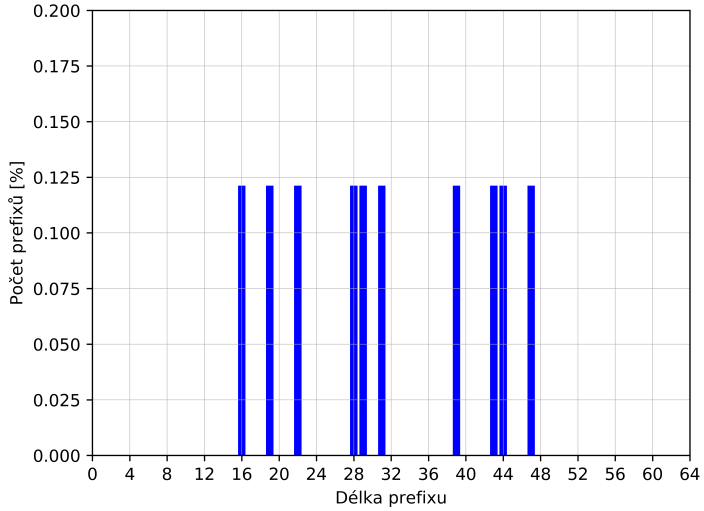
Rozložení delek prefixů

Reálné datové sady IPv6 prefixů obsahují prefixy o různých délkách. Z popisu alokačních politik, uvedených na konci kapitoly 2, je možné předpokládat, že většina prefixů získaných z dřívějších zdrojů dat, bude mít délku v rozsahu odpovídajícím délce prefixu LIR nebo ISP. Na obrázku 3.2 je znázorněno rozložení délek prefixů získaných z datové sady roku 2007. Pro větší přehlednost, rozložení délek prefixů, které mají příliš malé procentuální zastoupení v referenční sadě prefixů, je uvedeno na obrázku 3.3.

[1] https://matplotlib.org



Obrázek 3.2: Rozložení délek prefixů v datové sadě roku 2007

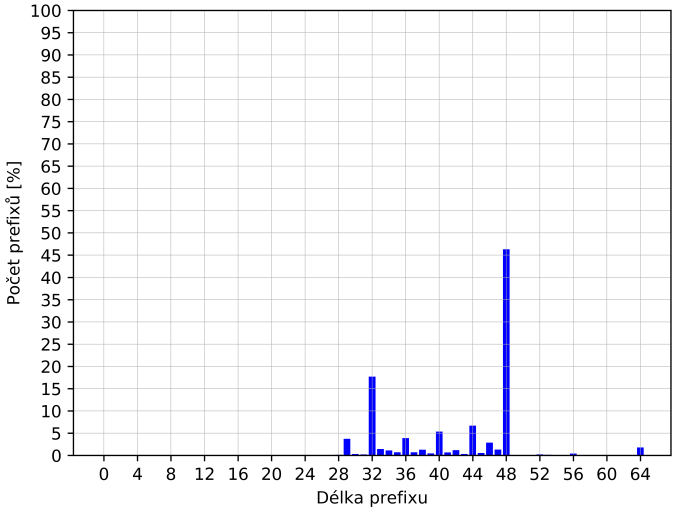


Obrázek 3.3: Rozložení délek prefixů, majících nízkou četnost výskytu v sadě z roku 2007

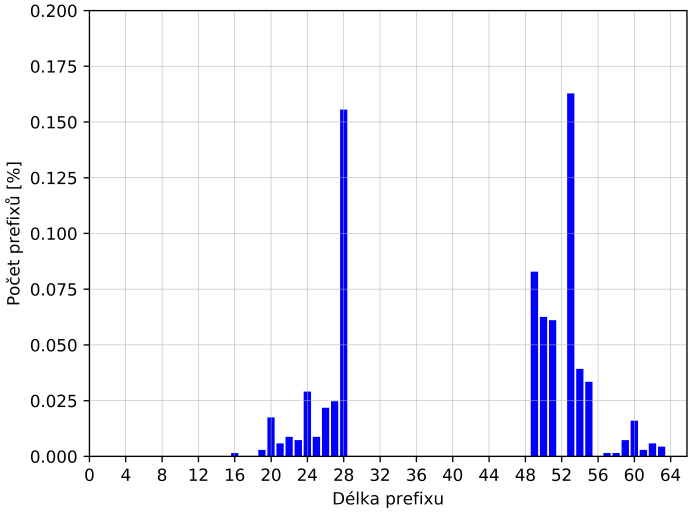
Vzhledem a aktivnímu rozvoje IPv6 je možné očekávat, že rozložení délek prefixů se bude v čase měnit tak, že nárůst počtu prefixů délky `/32` se zpomalí a bude častěji docházet k přidělování jednotlivých bloků adres organizacím, které se nacházejí na nižších úrovních hierarchie internetových registrátorů. To povede k nárůstu počtu prefixů v rozsahu délek `/48` až `/64`. Na obrázcích 3.4 a 3.5 je viditelně, že adresový prostor se výrazně změnil v průběhu následujících dvanácti let.

Na obrázku 3.4 můžeme pozorovat nárůst prefixů majících délku v rozsahu `/40` až `/48`. Tento jev může být způsoben tím, že došlo ke vzniku nových ISP, kteří na základě zdůvodněných požadavků žádali o přidělení většího adresového prostoru.

Kvůli tomu, že posledních 64 bitů adresy se používá pro identifikaci rozhraní v síti, neměly by se prefixy s délkou v rozsahu `/65` až `/128` ve směrovacích tabulkách vyskytovat. Přesto se takové prefixy vyskytují v datové sadě roku 2019. Jejich počet je ale relativně malý. Na základě nastudovaných alokačních politik, prefixy takové délky by neměli vyskytovat ve směrovacích tabulkách. Z těchto důvodů prefixy tyto prefixy nebyly uvedeny ve výsledném grafu.



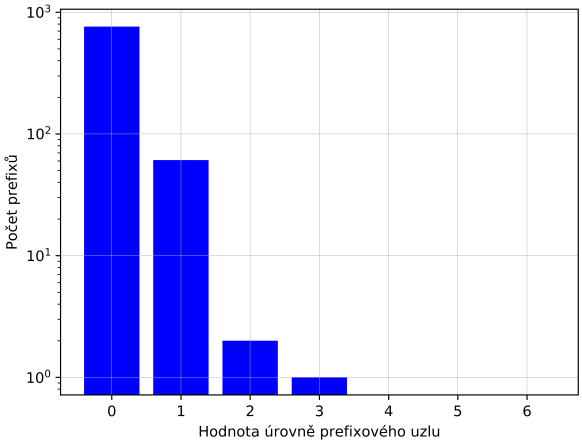
Obrázek 3.4: Rozložení délek prefixů v datové sadě roku 2019



Obrázek 3.5: Rozložení délek prefixů, majících nízkou četnost výskytu v sadě z roku 2019

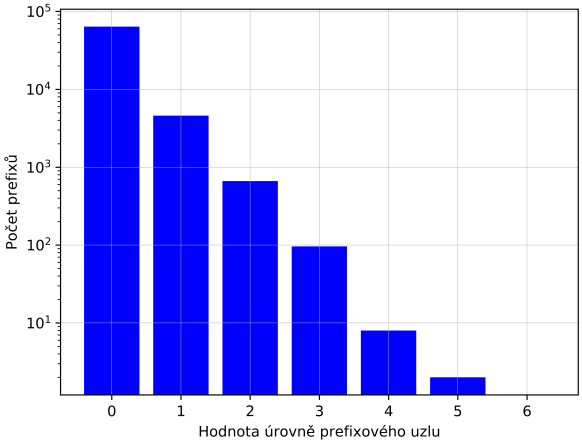
Rozložení úrovní prefixů

Další metodou analýzy je zkoumání struktury adresového prostoru na základě rozložení úrovní prefixů. Obrázek 3.6 ukazuje rozložení hodnot úrovní prefixů z datové sady roku 2007. Je patrné, že většina prefixů má hodnotu úrovně rovnou 0. To znamená, že přidělené prefixy jsou ve stromu reprezentovaný pomocí prefixových uzlů, které nemají žádného následníka.



Obrázek 3.6: Rozložení úrovní prefixů v datové sadě roku 2007

Maximální úroveň prefixů datové sady roku 2019 je větší a rovná se hodnotě 5, což může svědčit o tom, že došlo ke vzniku nových podsítí v rámci již existujících sítí. Zároveň však platí, že většina prefixů má nulovou úroveň. Je tudíž možné předpokládat, že v průběhu přidělování adresových bloků novým ISP nebo EU se mohou hodnoty úrovně uzlů postupně zvyšovat. Výsledné rozložení úrovní prefixů sady z roku 2019 je uvedeno na obrázku 3.7.



Obrázek 3.7: Rozložení úrovní prefixů v datové sadě roku 2019

3.3 Důvody pro návrh generátoru prefixových sad

Vzhledem k tomu, že protokol IPv6 používá pro reprezentaci IP adres 128 bitů a poskytuje tak mnohem větší adresový prostor, algoritmy pro směrování paketů v sítích IPv4 nevykazují dostatečnou efektivitu v sítích IPv6 .

Z výsledků analýzy rozložení délek prefixů uvedených v podkapitole 3.2.2 je patrné, že většina prefixů v dnešních směrovacích tabulkách má délku `/32` nebo `/48`. Udržení takového rozložení délek prefixů by však nemělo trvat dlouhou dobu. Lze předpokládat, že struktura adresového prostoru se výrazně změní a bude mnohem častěji docházet k přidělení adresového prostoru směrem k EU, což bude mít za následek narůst prefixů majících délku v rozsahu `/56` až `/64`.

Dalším významným problémem pro vývoj výkonných směrovacích algoritmů a jejich testování je malé množství záznamů ve dnešních IPv6 směrovacích tabulkách na rozdíl od směrovacích tabulek IPv4 [2].

Z těchto důvodů je nutné navrhnout nástroj, který by dokázal poskytnout dostatečně velké IPv6 tabulky, jež budou odpovídat struktuře IPv6 tabulek v budoucnosti. Výsledné IPv6 tabulky by pak mohly být použity pro efektivní testování nových směrovacích algoritmů.

Kapitola 4

Návrh a implementace generátoru V6Gene

Tato kapitola se věnuje návrhu a implementaci generátoru IPv6 prefixů. Zaprvé bylo nutné nastudovat princip generování IPv6 prefixů popsaný v článku [22] (viz podkapitola 4.1). Dále byla provedena implementace nastudovaného přístupu v jazyce Python, jejímuž popisu se věnuje podkapitola 4.3. Z vygenerované sady prefixů pak byly odhaleny nedostatky implementovaného generátoru (viz podkapitola 5.1), které byly následně eliminovány při návrhu vlastního generátoru IPv6 prefixů popsaného v kapitole 5.

4.1 Popis generátoru

Generátor V6Gene byl vyvíjen a popsán v roce 2006 Zhengem a Liu. Princip generování je v nastrojí V6Gene založen na základě existujících sadách prefixů a vychází ze znalosti pravidel pro přidělování prefixů IPv6 adres. Tento nástroj má za úkol rozgenerovat existující sady prefixů, které v době vzniku tohoto přístupu se skládali z velmi malého počtu záznamů do stavu co nejpřesněji zodpovídajícího budoucností. Proces generování prefixů zahrnuje tři základní fáze [22]:

1. V **inicializační** fázi generátor načítá a zpracovává vstupní parametry. Následně je načten vstupní soubor obsahující vzorové IPv6 prefixy. Na základě této prefixové sady pak je sestaven binární prefixový strom, který reprezentuje množinu prefixů uložených ve směrovací tabulce a používá se v následující fázi.

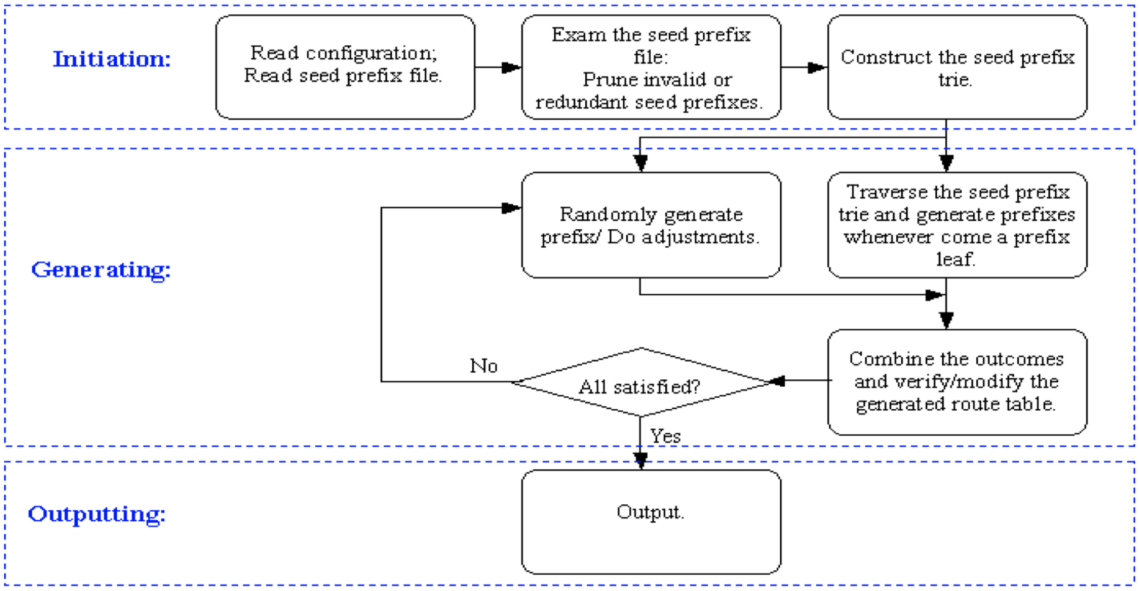
2. Fáze **generování** simuluje proces přidělování IPv6 adresových bloků a je rozdělena na dva paralelní procesy:

   - Na začátku probíhá simulace přidělování nových prefixů od již existujících lokálních registrátorů směrem ke koncovým uživatelům. Pokud je v průběhu procházení prefixovým stromem nalezen listový prefixový uzel, který byl vytvořen v době sestavení stromu, bude spuštěn proces generování nových prefixů v souladu se vstupními parametry.
   - Ve druhém procesu probíhá náhodné generování simulující přiřazování adresového prostoru lokálním registrátorům, kteří doposud neexistovali. Takové prefixy se generují bez ohledu na vstupní prefixovou sadu. Počet prefixů vytvořených v této fázi je ovlivněn hodnotou vstupního parametru `RGR` (z angl. *Random generation ratio*)

Po dokončení obou procesů proběhne kontrola vygenerovaných prefixů, v rámci které jsou odstraněny duplicitní prefixy a prefixy, které neodpovídají vstupním parametrům.

3. Ve **výstupní** fáze probiha formátovánaní výsledné sady prefixů do podoby souboru obsahujícího vstupní prefixy. Na zavěr, formatovaná sada prefixů bude přesměrovaná na odpovidajicí výstup.

Průběh generování je znázorněn na obrázku 4.1



Obrázek 4.1: Algoritmus generátoru V6Gene [22]

4.2 Návrh generátoru

Inicializační fáze generátoru zahrnuje načtení a zpracování hodnot vstupních parametrů. Parametry by měly specifikovat celkový počet prefixů na výstupu a definovat množství prefixů, které bude vygenerováno na základě obsahu vstupního souboru. Je nutné také zavest parametr, který bude specifikovat kolik prefixů určité délky se mají objevit ve výstupní sadě prefixů. Posledním parametrem, který by měl ovlivnit strukturu stromu je požadavek na rozložení prefixů podle hodnoty úrovně. Dalším krokem inicializační fáze je ověření souboru obsahujícího vzorové prefixy. Z tohoto souboru je zapotřebí odstranit nevalidní prefixy. Kvůli tomu, že se vstupní prefixová sada může skládat z dat získaných z různých zdrojů, mohou se v ní objevit duplicitní prefixy, které by měly být rovněž odstraněny. S ohledem na výsledky analýzy datové sady uvedené v podkapitole 3.2.2 a alokačních politik popsaných v podkapitole 2.4.3 je nutné také odstranit prefixy, jejichž délka je větší než `/64`. Z upravené sady prefixů pak bude sestaven binární prefixový strom. Uzly stromu by měly obsahovat informaci o délce uloženého prefixu a jeho úrovni, aby bylo možné dodržovat rozložení specifikované vstupními parametry.

Ve fázi generování je nutné pro vytvoření nových prefixů navrhnout algoritmus generování skupiny nových bitů. Za předpokladu, že vložením nového uzlu může dojít ke přepočítání úrovní uzlů na cestě do vkládaného uzlu, je potřeba navrhnout určitý algoritmus.

Takový algoritmus má za úkol zkontrolovat, zdali nedošlo k vytvoření uzlu majícího úroveň vyšší než maximální povolená, a provést její přepočítání v případě potřeby. Na konci první fáze generování proběhne kontrola duplicitních prefixů a prefixů, které nesplňují požadavek na rozložení prefixů podle hodnot úrovní. Pro zjednodušení provádění kontroly takových prefixů ve stromu by bylo možné provádět odstranění uzlu v případě potřeby už v průběhu vložení generovaného uzlu do stromu. Konečně je nutné zakázat generování nových prefixů z uzlů, které byly přidány do stromu v průběhu generovacího procesu.

V druhé části procesu generování probíhá generování náhodně. Aby nový prefix patřil do rozsahu adres typu global unicast, je nutné nastavit první tři bity nového prefixů na odpovídající hodnotu. Zbývající počet bitů bude vygenerován stejným způsobem jako v předchozí fázi. Pokud po dokončení první fáze generování bude počet prefixů nedostatečný, bude spuštěno náhodné generování, které se pokusí dosáhnout požadovaného počtu prefixů.

Výstupní fáze se pouze stará o převedení prefixů do hexadecimální podoby a předává výslednou sadu na výstup.

4.3 Implementace generátoru

Tato podkapitola se zabývá popisem implementaci generátoru. Nastroj byl implementován jako konzolová aplikace v jazyce Python verze 3.6. Pro statistickou analýzu výsledků byly vytvořeny grafy za použití externí knihovny Matplotlib spolu s knihovnou NumPy[1].

4.3.1 Vstupní parametry generátoru

Pro spuštění generátoru se používá soubor `V6Gene.py` s následujícími parametry:

- `--input=`
  Povinný parametr, který definuje cestu do referenčního souboru obsahujícího prefixy pro sestavení binárního prefixového stromu.

- `--output=`
  Volitelný parametr, který specifikuje cestu do výstupního souboru, do nějž se ukládají vygenerované prefixy. Při spuštění bez tohoto parametru budou všechny vygenerované prefixy (včetně prefixů ze vstupní prefixové sady) předány na standardní výstup.

- `--prefix_quantity=`
  Kladné celé číslo určující počet prefixů, které se objeví ve výstupní sadě. Počet prefixů definovaný tímto parametrem musí být větší než počet prefixů ve vstupní datové sadě. Parameter je povinný.

- `--rgr=`
  Je povinným parametrem určujícím poměr počtu prefixů, které budou vygenerovány bez ohledu na vstupní prefixovou sadu. Poměr se zadává jako desetinné číslo v rozsahu <0, 1>. Pokud je hodnota parametru `--rgr` nastavena na 0, nebude proces náhodného generování prefixů spuštěn a není možné garantovat, že se na výstupu objeví požadovaný počet prefixů.

- `--depth_distribution_path=`
  Parametr definuje cestu do souboru obsahujícího výsledné rozložení délek prefixů.

[1] https://www.numpy.org

Soubor je tvořen záznamy ve tvaru `D:N` uloženými na jednotlivých řádcích se znakem ‘,’ na konci. Jednotlivé položky záznamu lze popsat následujícím způsobem:

  - `D` - nezáporné celé číslo v rozsahu <0, 64>, které reprezentuje délku prefixu.
  - `N` - nezáporné celé číslo definující počet prefixů ve výsledné sadě s délkou D.

  Tento parametr nelze kombinovat s jiným vstupním parametrem `--depth_distribution`.

- `--depth_distribution=`
  Parametr definuje výsledné rozložení délek prefixů ve výstupní prefixové sadě. Tento parametr nemůže být použit spolu s parametrem `--depth_distribution_path`. Hodnota parametru se skládá ze záznamů majících stejný tvar jako u parametru `--depth_distribution_path`, jen s tím rozdílem, že jednotlivé záznamy jsou odděleny pouze znakem ","na konci.

- `--level_distribution_path=`
  Parametr definuje cestu do souboru obsahujícího výsledné rozložení prefixů podle hodnot úrovní. Soubor se skládá ze záznamů ve tvaru `L:N` na jednotlivých řádcích se znakem ",’"na konci. Význam jednotlivých položek těchto záznamů je následující:

  - `L` - nezáporné celé číslo v rozsahu <0, 6>, které reprezentuje úroveň prefixu.
  - `N` - nezáporné celé číslo, které určuje počet prefixů s hodnotou úrovně rovnou L.

  Stejně jako i parametru `--depth_distribution_path`, ani tento parametr nelze kombinovat s parametrem `--level_distribution`.

- `--level_distibution=`
  Parametr definuje výsledné rozložení prefixů podle úrovní. Hodnota parametru se skládá ze záznamů majících stejný tvar jako u parametru `--level_distribution_path`, jen s tím rozdílem, že jednotlivé záznamy jsou odděleny pouze znakem ‘,’ na konci. Tento parametr nelze kombinovat s parametrem `--level_distribution_path`.

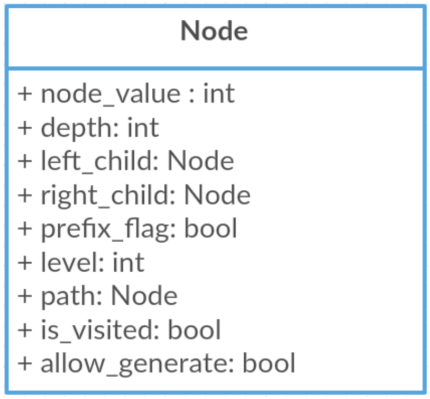
- `--graph`
  Volitelný parametr, který určuje, jestli po dokončení běhu generátoru budou vytvořeny statistické grafy. Vytvořené grafy budou uloženy do složky `statistics`, pro každý generátor zvlášť.

- `-h, --help`
  Výpis nápovědy na standardní výstup.

4.3.2 Inicializace

V inicializační fázi probíhá načtení a ověřování jednotlivých vstupních parametrů. V případě nalezení hodnoty, která je mimo očekávaný rozsah, dojde k ukončení běhu programu s výpisem odpovídající chybové hlášky. Poté následuje načtení vstupního souboru obsahujícího vzorové prefixy, přičemž každý prefix ze vstupního souboru musí být umístěn na samostatném řádku. V průběhu načítání prefixů probíhá rozhodování, zdali má hodnota prefixu správný formát zápisu a patří do rozsahu prefixů typu globální unicast. Načtený prefix se ukládá jako objekt třídy `IPv6Address`, která je definována v knihovně ipaddress. Pokud načtený prefix nespadá do rozsahu prefixů typu globální unicast nebo jeho délka přesahuje hodnotu 64, tak je v tomto případě odstraněn. Po kontrole je hodnota prefixu

konvertována do binární podoby a poté uložena do datové struktury typu množina. Pomocí této struktury je zajištěna unikátnost načtených prefixů. Po ověření prefixů ze vstupního souboru je ve výsledku z těchto prefixů vytvořen binární prefixový strom. Samotný strom je reprezentován třídou `Trie`, která je odvozená z abstraktní třídy `AbstractTrie` a implementuje metody `add_node()` a `generate_node()`. Základním prvkem stromové struktury jsou jednotlivé uzly, které jsou reprezentovány třídou `Node`. Struktura této třídy je uvedena na obrázku 4.2.



Obrázek 4.2: Třída Node

Význam jednotlivých položek třídy `Node` je následující:

- `node_value` - reprezentuje hodnotu uloženou v uzlu.
- `depth` - celé nezáporné číslo určující hloubku uzlu.
- `left_child` - ukazatel na levého potomka uzlu.
- `right_child` - ukazatel na pravého potomka uzlu.
- `prefix_flag` - indikátor, který slouží pro označení uzlů reprezentujících uložený prefix ve stromu.
- `level` - udržuje informaci o hodnotě úrovně uzlu.
- `path` - ukazatel na předchozí uzel uložený ve stromu (včetně vnitřního). Používá se pro usnadnění procesu přepočtu nebo ověření hodnoty úrovně prefixových uzlů nacházejících se na cestě k vkládanému uzlu.
- `is_visited` - signalizuje, zdali byl uzel navštíven při průchodu stromové struktury.
- `allow_generate` - signalizuje, zdali může uzel být použit pro generování nových prefixů.

V průběhu vložení uzlu do stromu probíhá kontrola, jestli vložený uzel nemá větší než maximální povolenou úroveň. Pokud k takové situaci dojde, proces generování nebude spuštěn a program skončí chybovým hlášením. Na konci inicializační fáze je vytvořena datová struktura typu seznam, ve které jsou uloženy informace o tom, kolik prefixů určité délky musí vzniknout v průběhu procesu generování.

4.3.3 Generování

První fází generování je procházení binárního prefixového stromu. Při nalezení listového uzlu stromu je zavolána metoda `generate_prefixes()`, která má za úkol vygenerovat a vložit nový prefix. Na začátku proběhne výpočet délky nového prefixu a to na základě délky stávajícího prefixu. Aby bylo zajištěno dodržování pravidel pro přidělování adres, která jsou uvedena v kapitole 2, je nutné definovat délky prefixů, které mohou vzniknout z prefixu původního (viz tabulka 4.1).

Délka prefixu ve stromu	**Délka generovaného prefixu**
/12 - /31	/32 - /47
/32 - /47	/48 - /63
/48 - /63	/64
/64	generování není povoleno

Tabulka 4.1: Závislost délky generovaného prefixu na délce prefixu uloženého ve stromu

Vložení nového prefixu do stromu je realizováno metodou `add_node()`, která vloží generované bity do stromu ve formě uzlů. Poslední vložený uzel bude označen jako prefixový. Tímto dochází ke vzniku nového prefixu.

Před pokračováním procesu generování prefixů je nutné ověřit, zdali vložený prefix splňuje následující požadavky:

- Vygenerovaný prefix musí být unikátní. Dojde-li k vygenerování již existujícího prefixu, vyskytne se výjimka `PrefixAlreadyExists`.
- Vložením nového prefixového uzlu nesmí dojít k překročení maximální povolené úrovně stromu. Pokud taková situace nastane, vyskytne se výjimka `MaximumPrefixLevel`.
- Délka nově vygenerovaného prefixu musí být větší než délka prefixu, ze kterého byl spuštěn proces generování, a zároveň musí respektovat požadavky dané rozložením délek prefixů specifikovaného vstupním parametrem.

Dojde-li k nesplnění alespoň jednoho z výše uvedených požadavků, vygenerovaný prefix nebude vložen do stromu a pokračuje proces vyhledávání dalšího listového uzlu. Bude-li hodnota úrovně prefixového uzlu větší než maximální povolená, je nutné vložený uzel ze stromu odstranit. Zároveň s tím je nutné také odstranit vnitřní uzly, které vznikly vložením prefixového uzlu. Proces odstranění prefixového uzlu je řízen pomocí algoritmu 1.

Algorithm 1 Odstranění uzlu ze stromu

```
Input
    node: uzel, který je nutné odstranit ze stromu
procedure DELETE_NODE_FROM_TRIE(node):
    path = získej všichní uzly nachazející se před uzlem node
    child = path[0]
    for counter = 0 to len(path) do
        if (childisnode) or (child není prefixovým uzlem and child je listovým uzlem) then
            parent = path[counter + 1]
            if child je pravým potomkem uzlu parent then
                Odstraň pravého potomka uzlu parent
            else
                Odstraň levého potomka uzlu parent
            end if
            child = parent
             continue
        end if
         break
    end for
end procedure
```

Vložení nového prefixového uzlu do stromu může způsobit změnu hodnot úrovně jednotlivých uzlů na cestě, na kterou byl vytvořený prefix uložen. Proces přepočítání hodnot úrovně takových uzlů (včetně hodnoty úrovně uzlu vkládaného) je řízen algoritmem 2.

Algorithm 2 Vypočet hodnoty úrovně vloženého uzlu

```
Input
    node: vkládaný uzel do stromu
procedure CALCULATE_LEVEL(node):
    full_path = získej prefixové uzly nachazející se před uzlem node
    new_level = 0
    if node není listovým uzlem then
        new_level = získej největší hodnotu úrovně ze všech uzlů v seznamu full_path
        if new_level + 1 > maximalní povolená úroveň then
            raise MaximumLevelException
        end if
        new_level = new_level + 1
    end if
    if vlozením uzlu node dojde ke změně hodnoty úrovně
       alesnpoň jednoho uzlu z full_path a ta bude větší než maximální povolená then
        raise MaximumLevelException
    end if
    if node není listovým uzlem then
        Nastav hodnotu úrovně uzlu node na hodnotu new_level
    end if
    if je nutné přepočitat hodnotu úrovně alespoň jednoho uzlů v seznamu full_path then
        Přepočitej hodnotu patřičných uzlů
    end if
end procedure
```

Po dokončení první fáze generování se může stát, že počet vygenerovaných prefixů bude menší, než bylo požadováno (například vzhledem k tomu, že došlo ke vzniku duplicitních prefixů). V tomto případě budou zbývající prefixy vygenerovány v průběhu náhodného generování. Během druhé fáze generování jsou nové prefixy generovány náhodně, tedy bez ohledu na množinu již existujících prefixů reprezentovaných binárním prefixovým stromem. S ohledem na to, že se ve směrovacích tabulkách mohou vyskytnout pouze adresy typu global unicast, jsou první tři bity generovaného prefixu nastaveny na hodnotu 001, aby bylo zajištěno přidělení prefixu z adresového prostoru `2000::/3`. Zbývající bity vkládaného prefixu se generují zcela náhodným způsobem. Na konci procesu generování proběhne ověření vytvořeného prefixu (zejména bude ověřená jeho unikátnost a hodnota úrovně).

4.3.4 Výstup

Na konci běhu programu je realizována konverze generovaných prefixů do hexadecimální podoby a uložení výsledku do odpovídajícího souboru specifikovaného vstupním parametrem `--output`. V opačném případě je výsledná sada prefixů vypsaná na standardní výstup.

4.4 Testování výsledné implementace

V této podkapitole jsou uvedeny výsledky testování generátoru IPv6 prefixů implementovaného dle [22]. Kvůli tomu, že neexistuje veřejně dostupná referenční implementace od autorů generátoru, nebylo možné porovnat sadu prefixů vygenerovanou pomocí referenční

implementace generátoru se sadou, která byla vygenerována implementovaným generátorem. Proto bylo nutné navrhnout vlastní přístup k testování výsledné implementace.

4.4.1 Metodika testování

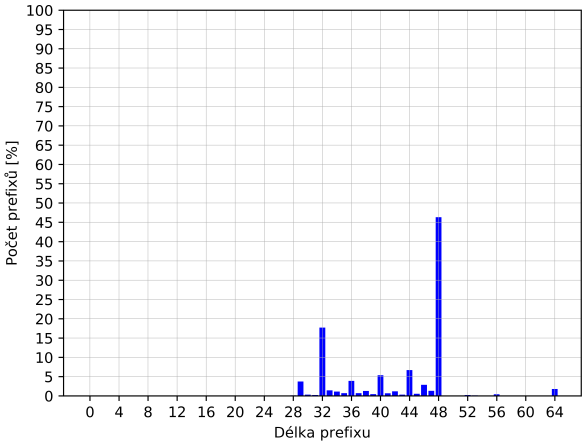
Metodika testování generátoru vychází ze znalosti obsahu směrovacích tabulek v různých časových intervalech. Z důvodu, nemožnosti jednoznačně určit, jak budou vypadat tyto tabulky v budoucnosti, byla použita sada prefixů reprezentující stav adresového prostoru v minulosti a na základě této sady byl vygenerován určitý počet nových prefixů. Vlastnosti výsledné sady prefixů pak mohou být porovnány s vlastnostmi aktuální verze dat získaných ze stejné databáze. Z těchto sad prefixů je nezbytné odvodit závislosti, které pak bude možné předat aplikaci ve formě vstupních parametrů.

4.4.2 Parametrizace

Pro sestavení prefixového stromu v inicializační fázi slouží datová sada IPv6 prefixů z roku 2007, která je uložena v souboru *formated_datasets/dataset2007*. Pro určení celkového počtu prefixů na výstupu byla použita datová sada z roku 2019 ze souboru *formated_datasets/dataset2019*. Z obsahu tohoto souboru byla také stanovena hodnota parametru `--depth_distribution`. Při podrobnějším zkoumání rozložení délek prefixů bylo zjištěno, že by se na výstupu mělo objevit 3111 nových prefixů, které mají délku odpovídající délce prefixů regionálních registrátorů. Tyto prefixy nemohou vzniknout z uzlů binárního prefixového stromu, a proto bylo nutné zvolit vhodnou hodnotu parametru `--rgr`. Procentuální zastoupení náhodně generovaných prefixů tedy bylo nastaveno na hodnotu 0.05 (neboli 5%). Na základě analýzy datové sady prefixů z roku 2019, která je uvedena v kapitole 3.2.2, bylo zišteno rozložení prefixů podle úrovně. Toto rozložení slouží jako hodnota vstupního parametru `--level_distribution`. Je očekáváno, že úroveň prefixových uzlů ve stromu by neměla přesáhnout hodnotu 5 (viz 3.7) Celý experiment může být spušten pomocí skriptu `v6gene_expirement.py` ve složce *experiments* (viz A)

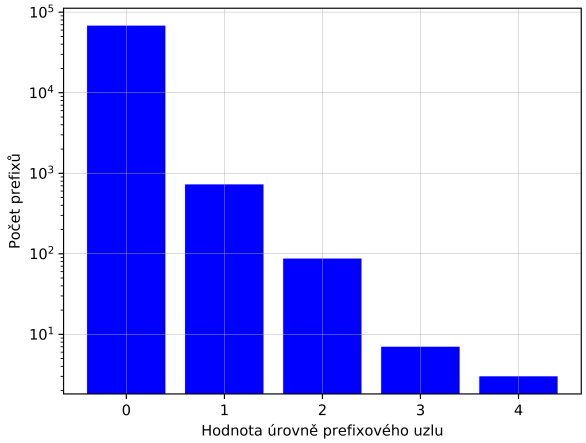
4.4.3 Testování generátoru

Z obrázku 4.3 je patrné, že rozložení délek prefixů ve výsledné sadě přesně odpovídá rozložení délek prefixů v reálné prefixové sadě (viz 3.4).



Obrázek 4.3: Rozložení délek prefixů v generované sadě prefixů

Výsledné rozložení prefixů podle hodnoty úrovně lze pozorovat na obrázku 4.4. Je viditelné, že maximální hodnota úrovně uzlu ve výsledné sadě prefixů je menší, než bylo očekáváno. Generovaná sada prefixů také obsahuje mnohem větší počet prefixů s hodnotu úrovně 0. Tento jev by mohl být způsoben tím, že se v průběhu první fáze generování nepodařilo splnit požadavek na generování určitého počtu prefixů, a proto bylo zbývající množství těchto prefixů vygenerováno zcela náhodně. Takové chování by mohlo mít za následek například vznik prefixových uzlů ve stromu, reprezentujících prefixy typu EU nebo ISP patřící k žádnému LIR.



Obrázek 4.4: Rozložení hodnot úrovně prefixů ve výsledné sadě prefixů

Kapitola 5

Návrh a implementace vlastní verze generátoru

Tato kapitola se zabývá návrhem a implementací vlastní verze generátoru IPv6 prefixů. Návrh nové verze generátoru vychází z přístupu popsaného v kapitole 4 a odhalených nedostatků tohoto přístupu (viz podkapitola 5.1). Navržená vlastní verze generátoru by měla poskytnout výsledky co nejvíce odpovídající realitě. Návrh generátorů vychází ze znalostí alokačních politik uvedených v kapitole.

5.1 Motivace

Z popisu generátoru V6Gene uvedeného v podkapitole 4.1 je patrné, že generování nových prefixů může být spuštěno pouze z listových uzlů stromu. Toto omezení však není odůvodněno ve článku [22]. Odstraněním tohoto omezení by mělo mít za následek narůst počtu větvení uvnitř prefixového stromu. Další důležitou vlastností generátoru V6Gene je zákaz generování nových prefixů z uzlů přidaných do stromu v průběhu procesu generování. Jinými slovy dochází k přidělení adresového prostoru organizacím, které nikdy nepřidělí adresový prostor svým klientům. Takové chování přitom na žádné z úrovní hierarchie registrátorů, kromě úrovně koncových uživatelů, neodpovídá realitě.

Jak bylo zmíněno v podkapitole 4.2, po dokončení generování prefixů na základě obsahu vstupního souboru může stále platit požadavek na generování nových prefixů. Dojde-li k takové situaci, pak bude tento požadavek obsloužen generátorem náhodných prefixů. Kvůli tomu, že v této fázi probíhá generování prefixů bez ohledu na vstupní datovou sadu, může nastat situace, kdy bude nový prefixový uzel vygenerován z uzlu odpovídajícího registrátorovi, který by na základě pravidel přidělování adresového prostoru takový prefix přidělit neměl. Příkladem je přidělení prefixu o délce `/64` koncovému uživateli od registrátora na úrovni RIR. Z těchto důvodů bylo nutné navrhnout jiný způsob generování prefixů, který je podrobnější popsán v další podkapitole.

5.2 Návrh generátoru

Stejně jako u generátoru V6Gene bude proces generování nových prefixů sestávat ze tří fází. V průběhu první fáze budou načteny a ověřeny vstupní parametry generátoru. Za účelem zjednodušení parametrizace generátoru byly odstraněny vstupní parametry `--level_distribution` a `--level_distribution_path`. Dále byl odstraněn vstupní para-

metr `RGR`. Počet prefixů, které mohou vzniknout pouze v průběhu náhodného generování, je totiž možné vypočítat na základě rozložení délek prefixů obsažených ve vstupním souboru a zadaného rozložení délek prefixů ve výstupní prefixové sadě. Nakonec byl navrhnut jiný způsob zápisu parametru pro určení maximální povolené hodnoty úrovně prefixových uzlů. V průběhu vytváření prefixového stromu je také nutné provádět ukládání prefixových uzlů reprezentujících prefixy z načteného souboru do samostatné datové struktury, což umožní vyhnout se jejich vyhledávání v prefixovém stromu v průběhu generování nových prefixů. Lze očekávat, že takový způsob výběru počátečních uzlů pro generování nových prefixů pozitivně ovlivní rychlost celého generování.

Na základě znalosti nedostatků generátoru V6Gene, uvedených v podkapitole 5.1, bylo nutné kompletně přepracovat fázi generování nových prefixů tak, aby:

- bylo povoleno generování nových uzlů z libovolného prefixového uzlu;
- bylo možné provádět generování nových prefixů z uzlů, které byly přidány do stromu v průběhu procesu generování;
- generování náhodných prefixů bylo omezeno pouze na prefixy odpovídající délce prefixů na úrovni RIR.

Na rozdíl od implementace generátoru V6Gene bude nejprve provedeno generování prefixů pomocí náhodného generátoru, aby pak bylo možné použít takto vytvořené prefixy v průběhu generování postaveného na základě obsahu vstupního souboru.

Výstupní fáze generátoru zůstane kompletně zachována.

5.3 Implementace

Tato podkapitola se zaměřuje na samotnou implementaci vlastní verze generátoru IPv6 prefixů. Generátor byl implementován ve formě konzolové aplikace v jazyce Python verze 3.6.

5.3.1 Vstupní parametry generátoru

Pro spuštění generátoru se používá soubor `IPv6Gene.py` s následujícími parametry:

- `--input=`
  Povinný parametr, který definuje cestu do vstupního souboru obsahujícího prefixy pro sestavení binárního prefixového stromu.
- `--output=`
  Volitelný parametr, který specifikuje cestu do výstupního souboru, do kterého se ukládají vygenerované prefixy. Při spuštění bez tohoto parametru jsou všechny vygenerované prefixy (včetně prefixů ze vstupního souboru) předány na standardní výstup.
- `--prefix_quantity=`
  Kladné celé číslo určující počet prefixů, které se objeví ve výstupní sadě. Počet prefixů definovaný tímto parametrem musí být větší než počet prefixů ve vstupní datové sadě. Parametr je povinný.
- `--depth_distribution_path=`
  Parametr definuje cestu do souboru obsahujícího cílové rozložení délek prefixů. Soubor

je tvořen záznamy ve tvaru D:N uloženými na jednotlivých řádcích se znakem‘,’ na konci, kde::

  - `D` - celé číslo v rozsahu <0, 64>, které reprezentuje délku prefixu.
  - `N` - celé číslo definující počet prefixů s délkou D ve výsledné sadě.

  Tento parametr nelze kombinovat s jiným vstupním parametrem `--depth_distribution`.

- `--depth_distribution=`
  Parametr definuje cílové rozložení délek prefixů ve výstupní prefixové sadě. Tento parametr nemůže být použit spolu s parametrem `--depth_distribution_path`. Hodnota parametru se skládá ze záznamů majících stejný tvar jako u parametru parametru, popsaného výš jen s tím rozdílem, že jednotlivé záznamy jsou odděleny pouze znakem‘,’ na konci.

- `--max_level=`
  Celé číslo v rozsahu <0, 7> určující maximální povolenou úroveň prefixu, která se muže ve stromu vyskytnout. Jde o povinný parametr.

- `--graph`
  Volitelný parametr, který určuje, jestli budou po dokončení běhu generátoru vytvořeny statistické grafy. Vytvořené grafy budou uloženy do složky `stats_output`, pro každý generátor zvlášť.

- `--stats`
  Výpis statistických informací v průběhu procesu generování na standardní výstup.

- `-h, --help`
  Výpis nápovědy na standardní výstup.

5.3.2 Inicializace

Jednotlivé kroky v inicializační fázi, s výjimkou vytvoření prefixového stromu, zůstaly zachované. Hlavní rozdíl při sestavení prefixového stromu spočívá v tom, že v době vložení prefixového uzlu do stromu bude vkladaný uzel uložen také do datové struktury typu slovník. Tato struktura se skládá ze záznamů majících tvar `D:S`. Význam jednotlivých položek je následující:

- `D` - celé číslo v rozsahu <1, 4>, které reprezentuje úroveň internetového registrátora v hierarchii registrátorů určenou podle délky prefixu. Přehled jednotlivých organizačních úrovní je uveden v tabulce 5.1.

Úroveň v hierarchii IR	**Délka prefixu**
1 - RIR	/12 - /31
2 - LIR	/32 - /47
3 - ISP	/48 - 63
4 - EU	/64

Tabulka 5.1: Přehled organizačních úrovní podle délky prefixu

- `S` - seznam všech prefixových uzlů, které patří do odpovídající úrovně registrátorů.

Podle délky vkládaného prefixu bude určen klíč `D` ve slovníku. Vytvořený prefixový uzel pak bude vložen podle vypočtené hodnoty klíče do odpovídajícího seznamu `S`.

5.3.3 Generování prefixů

Při detailnějším zkoumání sad prefixů z roku 2007 a 2019 bylo zjištěno, že v těchto sadách neexistují prefixy s délkou kratší než `/16` (výjimkou je pouze prefix s délkou `/0`, který slouží pro reprezentaci celého adresového prostoru). Zároveň s tím, prefixová sada z roku 2019 v porovnaní s sadou z roku 2007 má velký nárůst prefixů s délkou v rozsahu `/12` až `/31`. Pro uspokojení požadavku na generování prefixů s délkou odpovídající úrovni RIR registrátorů bude nejprve spuštěno generování náhodných prefixů. Takto vytvořené uzly uloženy do globálního uložiště prefixových uzlů reprezentovaného pomocí slovníku (viz podkapitola 5.3.2). Využití náhodného generátoru prefixů bude omezeno pouze na prefixy, jejichž délka spadá do výše uvedeného rozsahu.

Ve druhé části proběhne generování prefixů na základě obsahu vstupního souboru a již vygenerovaných prefixů z předchozího kroku. Podle délky nového prefixu bude ve slovníku vybrána položka, která udržuje informace o uzlech nacházejících se v hierarchii internetových registrátorů o jednu úroveň výše. Pro určení uzlu, z nějž bude generován nový prefix, se používá funkce `randint()` ze standardní knihovny `random`. Nakonec bude zavolána metoda `generate_new_bits()`, která má za úkol vygenerovat skupinu nových bitů, které pak budou připojeny k prefixu reprezentovanému vybraným uzlem. Před vložením vygenerovaného prefixu do stromu bude provedeno ověření, zdali vygenerovaný prefix splňuje požadované vlastností (viz podkapitola 4.3.3). Tím dojde k vytvoření prefixu a jeho uložení do stromové struktury. Procesy přepočítaní hodnoty úrovně prefixu a odstranění uzlu v případě potřeby lze provést podle již uvedených algoritmů 1 a 2. Než bude proces generování prefixů pokračovat dál, je nutné umístit vytvořený prefix do slovníku, přičemž položka ve slovníku bude vybrána na základě délky vytvořeného prefixu.

Proces generování pokračuje tak dlouho, dokud není vygenerované požadované množství prefixů. Aby bylo možné předejít uváznutí generátoru v případě, kdy nemůže být požadavek na vytvoření prefixu z vybraného uzlu splněn, je nezbytné omezit počet pokusů pro generování z prefixových uzlů stromu na určitou hodnotu. Experimentálně bylo zjištěno, že je optimální povolit maximálně 6 pokusů. Dojde-li k využití všech možných pokusů pro zvolený uzel, bude další generování z takového uzlu zakázáno.

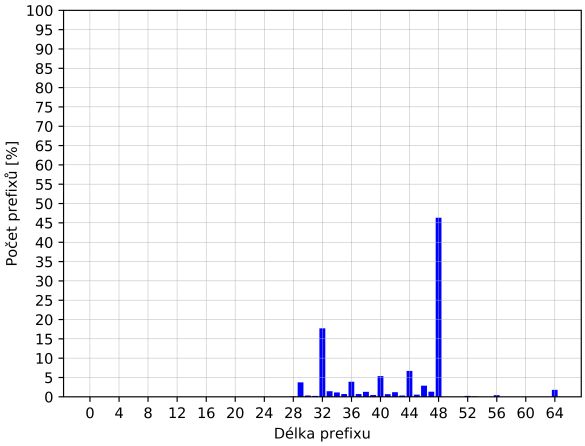
5.3.4 Výstup

Ve výstupní fázi generátoru proběhne převádění prefixů uložených ve stromu do hexadecimální podoby a jejich předání na odpovídající výstup určený podle vstupního parametru aplikace.

5.4 Testování výsledné implementace

Testování implementovaného generátoru lze provést stejným způsobem jako u generátoru V6Gene (4.4.1). Pro určení hodnoty parametru `--depth_distribution` byla opět použita datová sada roku 2019. Maximální povolená úroveň prefixu ve stromu, která se zadává pomocí parametru `--max_level`, byla stanovena na hodnotu 5. Binární prefixový strom byl vytvořen na základě obsahu sady prefixů z roku 2007. Na obrázku 5.1 lze vidět, že rozložení

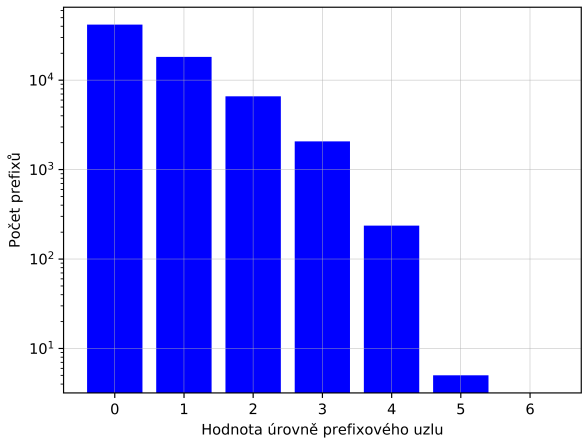
délek prefixů ve vygenerované sadě prefixů, získané pomocí navrženého generátoru, přesně odpovídá rozložení délek prefixů v reálné prefixové sadě z roku 2019 (viz 3.4).



Obrázek 5.1: Rozložení délek prefixů v výstupní sadě

Rozložení prefixů podle hodnoty úrovně je znázorněno na obrázku 5.2. V porovnání s rozložením délek prefixů v vytvořené sadě pomocí implementovaného generátoru V6Gene 4.4 je patrný nárůst počtu prefixů s hodnotou úrovně větší než nula. Zároveň s tím došlo k poklesu počtu prefixů s hodnotou úrovně rovnou nule. Tento jev je dán odlišným způsobem generování prefixových uzlů, které mohou vznikat z libovolného uzlu stromu a ne jen z listových uzlů.

Z obrázku 5.2 je patrně, že na rozdíl od adresového prostoru reprezentovaného sadou prefixů, jež byla vytvořena s využitím generátoru V6Gene, obsahuje adresový prostor vytvořený generátorem dle vlastního návrhu více jednotlivých podsítí, které byly vytvořeny buď z referenčních prefixů nebo z prefixů vytvořených v průběhu procesu generování. Experiment lze provést za pomocí skriptu `ipv6gene_experiment.py` ve nacházejícím složce *experiments* (viz A).



Obrázek 5.2: Rozložení úrovní prefixů v datové sadě roku 2019

Kapitola 6

Srovnání implementovaných generátorů

Závěrečná kapitola této práce je věnována analýze výsledků získaných za využití implementovaných generátorů a porovnání efektivity těchto nástrojů. Z grafů uvedených na obrázcích 4.3 a 5.1 vyplývá, že rozložení délek prefixů ve výsledných sadách získaných pomocí implementovaných generátorů přesně odpovídají rozložení délek prefixů v datové sadě z roku 2019. Na rozdíl od prefixového stromu, vytvořeného za pomocí prvotní verze generátoru, strom, který byl vytvořen generátorem dle vlastního návrhu, má odlišnou strukturu (viz 4.4 a 5.2). Tudíž bylo nutné navrhnout další postupy pro porovnání generátorů, aby bylo možné posoudit o jejích kvalitě.

První část kapitoly se věnuje návrhu možných postupů, pomocí kterých lze srovnání provést. Další podkapitoly popisují samotný proces porovnání generátorů.

Testování bylo provedeno na zařízení MacBook Pro v prostředí operačního systému macOS Mojave, s využitím procesoru Intel Core i7 o frekvenci 3.5 GHz a 16 Gb paměti.

6.1 Postupy srovnání

Implementované generátory se používají binární prefixový strom pro reprezentaci adresového prostoru. Nicméně vzhledem k tomu, že generátory přistupují k procesu generování prefixových uzlů zcela odlišným způsobem, bylo by zajímavé detailněji prozkoumat strukturu vytvořených stromů reprezentujících výsledný adresový prostor (viz podkapitola 6.2). Pro účely analýzy by například bylo možné se podívat na počet uzlů ve stromech, které nemají na cestě před sebou žádný jiný uzel reprezentující prefix registrátora na vyšší úrovni hierarchie. Je také mohou ve stromu vyskytnout uzly, které byly vygenerovány bez ohledu na alokační politiky uvedené v kapitole 2. Další důležitou vlastností, kterou by bylo vhodné zvážit, je čas potřebný pro generování výsledných sad prefixů a jeho závislost na velikosti těchto sad. V neposlední řadě je nezbytné se podívat i na paměťovou spotřebu implementovaných přístupů. Podrobnější popis testování efektivity generátorů je popsán v podkapitole 6.3.

6.2 Srovnání struktur prefixových stromů

V průběhu testování generátoru V6Gene vznikla domněnka, že využití náhodného generátoru může značně ovlivnit kvalitu výsledné sady prefixů. Při generování prefixových uzlů

bez ohledu na obsah referenční sady existuje vysoká šance vytvoření prefixu, k jehož vzniku by nemělo podle nastudovaných alokačních politik dojít. Proto bylo rozhodnuto detailněji prozkoumat uzly stromu, které vznikly v průběhu procesu generování. Jako vstup byla u obou generátorů opět použita sada prefixů z roku 2007. Na jejím základě byl vytvořen prefixový strom, který byl následně rozgenerován do stavu sady z roku 2019. Po dokončení běhu generátoru bylo zapotřebí se podívat na jednotlivé prefixové uzly stromu a zjistit úroveň registrátora, který přidělil prefix reprezentovaný zkoumaným uzlem.

Nejprve byly prozkoumány uzly prefixového stromu, jenž byl vytvořen pomocí generátoru V6Gene. Tabulka[1] 6.1 obsahuje informace o počtů prefixů neodpovídajících nastudovaným alokačním politikám z hlediska jejich příslušnosti do jednotlivých organizačních úrovní.

Úroveň registrátora	**Počet prefixů**
RIR	3130
LIR	15621
ISP	710
EU	64

Tabulka 6.1: Počet prefixů z výstupní sady, které nedodržují alokační politiky. Sada byla vygenerována pomocí implementovaného generátoru V6Gene

Z výsledků analýzy sad prefixů uvedených v kapitole 3.2.2 je patrné, že směrovací tabulky neobsahují prefixy o délce kratší než 16 bitů. Zároveň nebyly nalezeny politiky přidělování adresového prostoru od registrátorů jiným organizacím, které se nacházejí na stejné úrovni hierarchie. Proto může docházet k vytvoření prefixů o délce odpovídající úrovni RIR pouze v průběhu náhodného generování. To svědčí o nemožnosti výskytu uzlů ve stromu reprezentujících poskytovatele adresového prostoru pro registrátory typu RIR. Toto platí pro oba generátory.

Podle tabulky T1 lze usoudit, že stromová struktura vytvořená za pomocí implementovaného nástroje V6Gene obsahuje na úrovních LIR, ISP a EU určité množství uzlů nedodržujících alokační politiky. Výskyt takových uzlů na uvedených úrovních je možné opět odůvodnit využitím náhodného generátoru, jelikož náhodné generování může být u generátoru V6Gene také spuštěno v následujících případech:

- Simulace přidělení adresového prostoru doposud neexistujícím organizacím na úrovni LIR. [22]
- Převzetí existujícího požadavku na generování prefixů, které se nepodařilo vytvořit na základě prefixového stromu (viz podkapitola 4.2). Příkladem je generování prefixů pro ISP a EU.

Výsledek analýzy uzlů prefixového stromu vytvořeného pomocí vlastního generátoru je uveden v tabulce 6.2.

[1] Sloupec **Počet prefixů** v tabulkách 6.1, 6.2 a 6.3 udává počet prefixů, které byli přiděleny bez ohledu na alokační politiky.

Úroveň registrátora	Počet prefixů
RIR	3130
LIR	667
ISP	54
EU	2

Tabulka 6.2: Počet prefixů z výstupní sady, které nedodržují alokační politiky. Sada prefixů byla vytvořena za použití generátoru dle vlastního návrhu

Navzdory tomu, že prefixový strom sestavený v době běhu generátoru dle vlastní implementace, obsahuje mnohem menší počet takových uzlů (s výjimkou uzlů nacházejících se na úrovni RIR), je překvapivé, že se tyto uzly ve stromu objevily. Jelikož využití náhodného generátoru je omezeno pouze na vytvoření prefixů RIR, nemělo by docházet k vytváření uzlů, které byly přiděleny od organizací nacházejících se více než o jednu úroveň výše v hierarchii registrátorů (výjimkou jsou pouze uzly, které zastupují prefixy EU). Při podrobnějším zkoumání vstupní sady prefixů bylo zjištěno, že tato sada už obsahovala stejný počet prefixů na stejných úrovních registrátorů přidělených bez ohledu na alokační politiky. Ke vzniku takových prefixů v reálné sadě mohlo pravděpodobně docházet v rané etapě rozvoje protokolu IPv6, když ještě alokační politiky pro přidělení adresového prostoru nebyly dostatečně přesně definovány. Výsledky analýzy prefixových uzlů stromu vytvořeného na základě sady z roku 2007 lze nalézt v tabulce.

Úroveň registrátora	Počet prefixů
RIR	17
LIR	667
ISP	54
EU	2

Tabulka 6.3: Počet prefixů v sadě z roku 2007, které nedodržují alokační politiky

6.3 Testování efektivity implementovaných generátorů

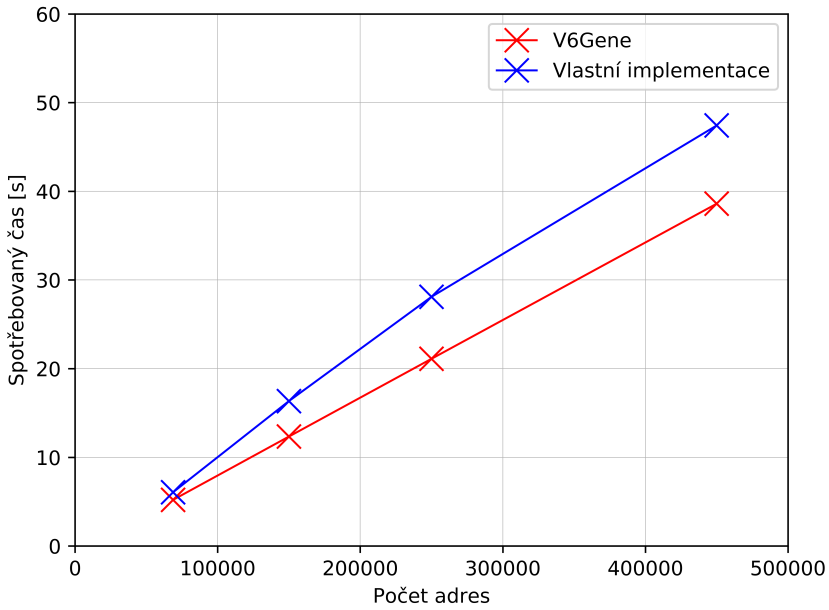
Generování prefixových uzlů je u generátoru V6Gene založeno na principu průchodu prefixovým stromem. Generátor dle vlastního návrhu na rozdíl od generátoru V6Gene zanedbává proces vyhledávání patřičného uzlu ve stromu a jen vybírá náhodný uzel, ze kterého je povoleno vytvoření nového uzlu o požadované délce. Lze předpokládat, že tato odlišnost ovlivní celkový čas potřebný pro generování. Dalším parametrem, který by mohl ovlivnit rychlost generátoru, je počet prefixů ve výstupní datové sadě. Vzhledem ke zcela odlišným způsobům ukládání vytvořených uzlů je také nezbytné se podívat na paměťovou náročnost implementovaných nástrojů.

Jako vstup byla u obou generátorů použita sada prefixů z roku 2007, ze které byly předem odstraněny duplicitní a nevalidní hodnoty prefixů.

6.3.1 Srovnání rychlosti generování

Nejprve bylo pro testování rychlosti generátorů provedeno generování 68 780 prefixů(velikost obsahu referenční sady z roku 2019 po odstranění duplicitních a nevalidních záznamů). Nicméně generování se povedlo provést přibližně za stejný čas pro oba typy generátorů a

to z důvodu malého množství záznamů. Bylo tudíž nutné vyzkoušet generování sad obsahujících poněkud větší počet záznamů. Pro účely testování rychlosti implementovaných generátorů byly vygenerovnány sady prefixů, které se skládají z 150 000, 250 000 a 400 000 unikátních hodnot. Rozložení délek prefixů pro tyto sady se nacházejí ve složce *distributions/big_datasets*. Výsledky testování jsou vyobrazeny na obrázku 6.1.

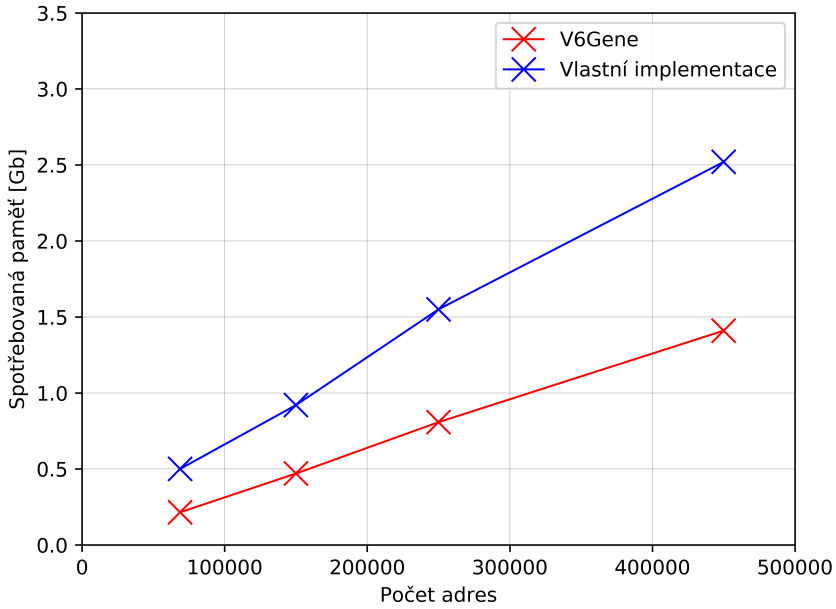


Obrázek 6.1: Porovnání rychlosti generátorů. Generování proběhlo na základě sady prefixů z roku 2007

6.3.2 Paměťová náročnost generátorů

Na základě toho, že uzly stromu obsahují ukazatele na své potomky a zpětný ukazatel na předcházející uzel, má datová struktura trie dost velké nároky na použitou paměť. Implementované generátory používají tuto datovou strukturu především pro ukládání prefixových uzlů a provedení kontroly, zdali vkladný uzel dodržuje požadavky kladené na hodnotu úrovně. Na rozdíl od generátoru V6Gene, který ukládá vytvořené uzly pouze do stromové struktury, navržený generátor ukládá vytvořené uzly zvlášť (viz 5.3.3), aby pak bylo umožněno jejich použití v průběhu procesu generován. Analýza efektivity generátorů z hlediska paměťové spotřeby probíhala stejným způsobem jako srovnání rychlosti generování uvedené v podkapitole 6.3.1. Pro měření paměťové náročnosti byla použita externí knihovna memory_profiler[2]. Tato knihovna však výrazně zpomaluje běh programu, a proto byla po dokončení testování ze zdrojových kódů odstraněna.Výsledky testování jsou vyobrazeny na obrázku 6.2. Větší spotřeba paměti u navrženého generátoru je důsledkem zvoleného způsobu ukládání prefixů během jejich generování.

[2] https://pypi.org/project/memory-profiler/



Obrázek 6.2: Paměťová náročnost generátorů. Generování proběhlo na základě sady prefixů z roku 2007

Kapitola 7

Závěr

Cílem této bakalářské práce bylo vytvoření generátoru IPv6 sad prefixů. Výstupní prefixové sady se skládají z prefixů, vytvořených v souladu nastudovanými alokačními politikami přidělování IPv6 adresového prostoru. Takto vytvořené sady prefixů by bylo možné použít pro testování výkonnosti vyhledávacích algoritmů. Návrhu a implementaci vlastní verze generátoru předcházelo nastudovaní a implementace již existujícího přístupu k řešeni teto problematiky uvedeného ve článku [22]. V teoretické častí práce je uveden způsob reprezentaci prefixu IPv6 a alokační politiky organizaci RIPE NCC, na základe kterých probíhá přidělení IPv6 adresového prostoru v určitých regionech. Problematikou reprezentaci adresového prostoru se zabývá kapitola 3. V teto kapitole byla popsaná stromová struktura trie a její podstatné vlastnosti. Na základě znalosti vlastnosti teto struktury byla pak provedena analýza dat, získaných z reálných směrovacích tabulek.

Praktická část práce nejprve se zaměřuje na implementaci generátoru V6Gene. Implementace tohoto nastroje je uvedena v kapitole 4. Je patrně, že úpravou odhalených nedostatků implementaci generátoru V6Gene lze dosáhnout kvalitnějších výsledků. Z těchto důvodů byl následné navržen a implementován vlastní generátor IPv6 prefixů. Popisu tohoto generátoru se věnuje kapitola 5.

Vzhledem k tomu, ze internetové registrátory na úrovni RIR mají odlišné politiky přidělovaní adresového prostoru, by bylo možné rozšířit výslednou implementaci vlastního generátoru o další parametr, který umožni provádět generovaní i na základě pravidel obstaních regionálních registrátorů. Dalším možným rozšířením je prováděni generování IPv6 prefixů s využitím pravidel popsaných v [13] a [3].

Ačkoli implementované generátory se přistupují k generovaní zcela odlišným způsobem, oba typy generátoru se používají datovou strukturu trie pro reprezentaci adresového prostoru a ukládaní prefixů. Jak ukázala analýza, která byla popsaná v kapitole 6, vlastní verze generátoru má větší paměťovou spotřebu a vyžaduje vice času pro dokončení generovaní větších prefixových sad než je to u implementovaného generátoru V6Gene. Avšak generátor dle vlastního návrhu poskytuje kvalitní výsledky. Jelikož dalším možným vylepšením generátoru je zvoleni jiného způsobu pro ukládaní a reprezentaci adresového prostoru.

Literatura

[1] *IPv6 Address Auto Configuration*. [Online; navštíveno 14.05.2019].
URL http://bucarotechelp.com/networking/standards/81090802.asp

[2] BGP Routing Table Analysis Reports. [Online; navštíveno 17.02.2019].
URL http://bgp.potaroo.net

[3] Blanchet, M.: *A Flexible Method for Managing the Assignment of Bits of an IPv6 Address Block*. April 2003, [Online; navštíveno 21.03.2019].
URL https://tools.ietf.org/html/rfc3531

[4] CAIDA Data Server. [Online; navštíveno 11.02.2019].
URL http://data.caida.org/datasets/routing/routeviews6-prefix2as/

[5] *DHCP Address Allocation Methods*. [Online; navštíveno 10.05.2019].
URL https://docs.paloaltonetworks.com/pan-os/8-1/pan-os-admin/networking/dhcp/dhcp-addressing/dhcp-address-allocation-methods

[6] Durand, A.; Huitema, C.: *The Host-Density Ratio for Address Assignment Efficiency: An update on the H ratio*. November 2001, [Online; navštíveno 21.03.2019].
URL https://tools.ietf.org/html/rfc3194

[7] Hagen, S.: *IPv6 Essentials*. O'Reilly Media, 2006, ISBN 0596100582.

[8] Hinden, R.; Deering, S.: *IP Version 6 Addressing Architecture*. February 2006, [Online; navštíveno 21.03.2019].
URL https://tools.ietf.org/html/rfc4291

[9] Huston, G.: *IPv4 Address Report*. [Online; navštíveno 15.04.2019].
URL https://ipv4.potaroo.net

[10] Internet Assigned Numbers Authority (IANA): *IANA IPv4 Special-Purpose Address Registry*. 2009, [Online; navštíveno 14.04.2019].
URL https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xhtml

[11] Kawamura, S.; Kawashima, M.: *A Recommendation for IPv6 Address Text Representation*. August 2010, [Online; navštíveno 01.02.2019].
URL https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xhtml

[12] Liu, D.; Barber, B.; DiGrande, L.: *Cisco CCNA/CCENT Exam 640-802, 640-822, 640-816 Preparation Kit*. Cisco Press, 2009, ISBN 978-1-59749-306-2.

[13] Nartenand, T.; Huston, G.; Roberts, L.: *IPv6 Address Assignment to End Sites*. March 2011, [Online; navštíveno 21.03.2019].
URL https://tools.ietf.org/html/rfc6177

[14] Odom, W.; Wilkins, S.: *Cisco CCNA Routing and Switching 200-120 Official Cert Guide and Simulator Library*. Cisco Press; 1 edition, 2004, ISBN 9781587204661.

[15] RIPE Network Coordination Centre: *RIPE NCC Receives Final /8 of IPv4 Address Space from IANA*. February 2011, [Online; navštíveno 27.02.2019].
URL https://www.ripe.net/publications/news/announcements/ripe-ncc-receives-final-8-of-ipv4-address-space-from-iana

[16] Routeviews - University of Oregon Route Views Project: *University of Oregon Route Views Archive Project*. June 2004, [Online; navštíveno 11.02.2019].
URL http://routeviews.org

[17] The CAIDA UCSD: *datasets/routing/routeviews6-prefix2as/2019/04* . [Online; navštíveno 14.04.2019].
URL
http://data.caida.org/datasets/routing/routeviews6-prefix2as/2019/04/

[18] The CAIDA UCSD: *routeviews6-prefix2as/2007/04* . [Online; navštíveno 14.04.2019].
URL
http://data.caida.org/datasets/routing/routeviews6-prefix2as/2007/04/

[19] The Internet Corporation for Assigned Names and Number (ICAAN) *Internet Assigned Numbers Authority (IANA) Policy For Allocation of IPv6 Blocks to Regional Internet Registries*. September 2006, [Online; navštíveno 30.02.2019].
URL https:
//www.icann.org/resources/pages/allocation-ipv6-rirs-2012-02-25-en

[20] The Réseaux IP Européens Network Coordination Centre *The Internet Registry System*. April 2014, [Online; navštíveno 30.02.2019].
URL https://www.ripe.net/participate/internet-governance/internet-technical-community/the-rir-system

[21] The Réseaux IP Européens Network Coordination Centre (RIPE) *IPv6 Address Allocation and Assignment Policy*. July 2018, [Online; navštíveno 30.02.2019].
URL https://www.ripe.net/publications/docs/ripe-707

[22] Zheng, K.; Liu, B.: *V6Gene: A scalable IPv6 prefix generator for route lookup algorithm benchmark*. Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA'06), vol. 1. IEEE. April 2006, 2006, ISBN 0-7695-2466-4.

Příloha A

Obsah přiloženého paměťového média

- `README.txt` - Popisuje instalaci knihoven, potřebných pro spuštění generátorů a návod k překladu.
- `doc/` - obsahuje samotný text bakalářské práce a zdrojové kódy LaTeX
- `src.zip` - obsahuje zdrojové kódy a všechní potřebné soubory pro běh generátorů, dodatečné skripty a datové sady, jenž byly použity pro testování