Česká společnost uživatelů otevřených systémů EurOpen.CZ
Czech Open System Users' Group
www.europen.cz

38. konference

Sborník příspěvků

U Bednáře
Pavlov
8.–11. května 2011

Sborník příspěvků z 38. konference EurOpen.CZ, 8.–11. května 2011

Plzeň 2011. První vydání.
Editor: Vladimír Rudolf
Sazba a grafická úprava: Ing. Miloš Brejcha – Vydavatelský servis, Plzeň
e-mail: servis@vydavatelskyservis.cz

Tisk: Typos, tiskařské závody, s. r. o.
Podnikatelská 1160/14, Plzeň

Příspěvky neprošly redakční ani jazykovou úpravou.

ISBN 978-80-86583-21-1

Obsah

Data mining v RapidMineru

Miroslav Skrbek

E-mail: mskrbek@prf.jcu.cz

Abstrakt

S dostupností stále většího množství dat v elektronické podobě roste také potřeba analýzy těchto dat za účelem získávání využitelných informací. Dobývání znalostí je oborem, který zahrnuje velké množství metod z oblasti databází, statistiky a umělé inteligence, které dovedou hledat vztahy a zákonitosti v datech a vytvářet jejich modely. Tento článek představuje otevřenou platformu RapidMiner určenou pro data mining, prediktivní analýzu a podporu manažerského rozhodování, která integruje široké spektrum algoritmů a nástrojů pro analýzu, modelování a vizualizaci dat.

Úvod

Informatizace společnosti je doprovázena rostoucím množství dat uložených v podnikových databázích, databázích státní správy, databázích finančního sektoru, u obchodních řetězců, u mobilních operátorů a dalších subjektů. Enormní objemy dat pak lze nalézt u poskytovatelů internetových služeb, zejména vyhledávačů, sociálních sítí, www a e-mailových prostorů. Vedoucí pracovníci a manažeři si stále více, a celkem logicky, uvědomují význam těchto dat, která mohou poskytnout důležité informace pro řízení podniku, reklamní účely, práci s klienty, zvyšování efektivity výroby apod. Také si postupně uvědomují i cenu takových dat, která po vytěžení znalostí mohou zásadně ovlivnit podnikové rozhodování (Bussines Intelligence) a přinést zisky.

KDD

Dobývání znalostí z databází (nebo také vytěžování dat) se označuje zkratkou KDD (Knowledge Discovery in Databases). Vytěžování dat musí provádět odborník se znalostí potřebných metodik, metod a nástrojů. Odborník, častěji tým odborníků, podrobí dostupná data procesu dobývání znalostí a na základě výsledků vypracuje zprávu, kde budou jednoduše a srozumitelnou formou prezentovány závěry analýzy a využitelné informace.

Obor KDD je vystaven na třech pilířích [1]

- databáze,

- statistické metody,
- umělá inteligence (strojové učení),

které tvoří teoretický základ oboru a jsou hlavním zdrojem metod pro výběr, analýzu a zpracování dat.

Proces dobývání znalostí se skládá z fáze selekce, předzpracování, transformace dat, dolování znalostí, vizualizace a interpretace výsledků [1]. Ve fázi selekce je třeba z datového skladu vybrat relevantní podmnožinu dat. Zde se výrazně uplatňují databázové techniky. V nejjednodušším případě se jedná o vykonání jednoho nebo sady SQL příkazů. Pokud je však nutno získávat data z více druhů databází, například SQL databází a současně fulltextových databází (dokumenty, e-maily, www stránky, apod.), může být proces výběru dat velmi složitý, časově náročný, vyžadující specializované konverzní a extrakční programy.

Fáze předzpracování a transformace dat je přípravou dat pro analytické metody. Řeší se zde především problémy s chybějícími hodnotami a odlehlými hodnotami, konverze datových typů, kódování kategorických proměnných, doplňování meta dat. Nad daty se provádí řada transformací. Velmi častá je normalizace dat. Některé analytické metody mohou také požadovat odstranění trendu, výpočet statistických charakteristik, při velkém množství atributů je nezbytná redukce dimenze datové matice, případně nasazení metod pro výběr a výpočet relevance atributů (feature selection a feature ranking).

Fáze dolování znalostí se opírá o metody modelování, klasifikace a shlukování dat. Zde se s výhodou uplatňují statistické metody a metody umělé inteligence, zejména metody založené na strojovém učení. Dolování znalostí je náročný iterativní proces, který se neobejde bez asistence odborníka, který na základě výstupů dolování zasahuje do celého procesu s cílem získat co nejlepší výsledek analýzy. Zásahy mohou mít formu jak úprav parametrů použitých metod, tak přímo i výměny analytických metod, včetně zásahů do procesu předzpracování a transformace dat. Výsledkem dolování znalostí jsou opět data, která je třeba dále interpretovat, vyvodit z nich závěry a získat tak využitelné informace. Klíčovým nástrojem pro tuto fázi je vizualizace dat. Obecně mohou být vizualizační techniky jednoduché (graf závislosti y na x), nebo také velmi pokročilé využívající 3D zobrazení s širokými možnostmi současné počítačové grafiky. Mapování hodnot nejen na souřadnice bodů, ale na různé parametry bodu jako je velikost, barva, tvar, průsvitnost dovoluje zobrazovat více-dimenzionální data.

Vytěžování dat je složitý časově náročný proces, zejména pokud je nutné zpracovávat velké objemy dat. Pro jednotlivé fáze procesu je jistě možné nalézt individuální nástroje. To je vhodné pro malé soubory dat, ale ne pro velké soubory dat. Lze například kombinovat SQL dotazy nad databází spolu s programem OpenOffice Calc nebo nebo SQL dotazy s knihovnami algoritmů napsanými v jazyce C, C++, Java. Automatizace analýzy dat se v tomto případě typicky

opírá o skripty shellu použitého operačního systému. Mnohdy se ale nevyhneme nutnosti průběžných zásahů uživatele (například nahrát soubor, spustit výpočet, uložit soubor), což nedovoluje proces analýzy vždy plně automatizovat. Různorodost použitých nástrojů přináší mnoho nekompatibilit v datových formátech, které je třeba často řešit vlastními konverzními programy. Pak každá změna procesu zpracování dat vyžaduje nezanedbatelnou práci a čas navíc. To samozřejmě odvádí pozornost od vlastního procesu analýzy a prodlužuje potřebnou dobu pro vytěžení dat.

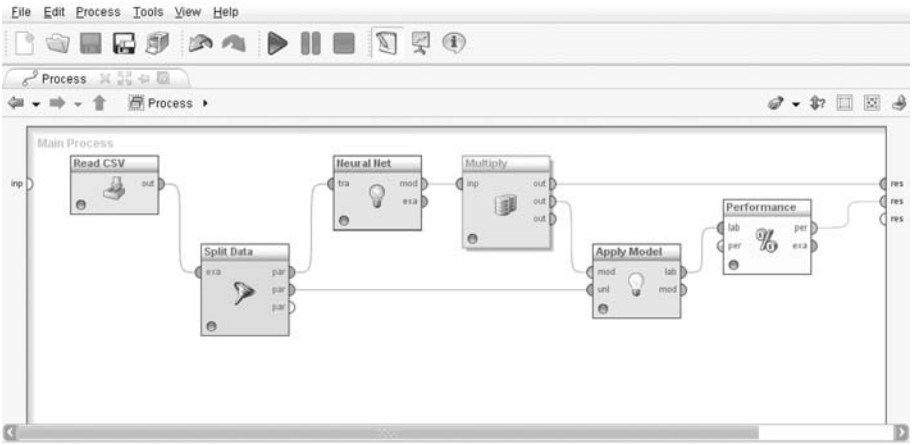
Pokročilé nástroje pro KDD jsou laboratořemi s jednotnou reprezentací dat, kde lze snadno a hlavně rychle sestavovat a upravovat postupy analýzy. V oblasti open-source se pro podporu KDD v posledních několika letech vyvinul nástroj RapidMiner. RapidMiner je KDD platforma určená pro podporu Bussiness intelligence. Dnes je k dispozici v open-source a také komerční verzi. Je vyvíjena v jazyce Java. Zárodek této platformy vznikl jako projekt YALE (Yet Another Learning Environment), kde byla definována jeho základní funkcionalita. Přeměna na RapidMiner přinesla především grafický popis procesů zpracování dat.

Rapid Miner

RapidMiner lze charakterizovat jako nástroj pro zpracování, modelování a vizualizaci dat. Integruje velké množství algoritmů z oblasti databází, statistiky a umělé inteligence, které jsou zapouzdřené do podoby operátorů, kterých je v současné době více než pět set. Argumenty (vstupy) a funkční hodnoty (výstupy) operátorů jsou datové objekty, které mohou obsahovat data, modely, výsledky a další. Nepostradatelnou součástí platformy jsou vizualizační nástroje. Ty nejsou zapouzdřeny do operátorů, jako je tomu u jiných platforem, ale jsou integrovány přímo do datových objektů a vyvolávány z prohlížeče výsledků. RapidMiner disponuje grafickým designérem pro návrh schémat zpracování dat (viz obr. 1). Operátory jsou ve schématech zobrazeny jako boxy se vstupy a výstupy, které lze snadno myší propojit spoji. Přes spoje si operátory předávají datové objekty. Schéma má podobu dataflow grafu výpočtu. Dokumentaci k RapidMineru je možné nalézt v [2].

Reprezentace, import a export dat

RapidMiner implementuje třídu ExampleSet, která reprezentuje data ve formě tabulky. Operátory si předávají instance této třídy a modifikují data v nich. Kromě vlastních dat obsahují instance třídy ExampleSet také meta data. Meta data jsou například informace o názvech atributů (sloupců tabulky), typech jednotlivých atributů (integer, real, nominal, ...), jejich roli při zpracování v operátorech, minimální/maximální hodnoty, průměrná hodnota atributu, výčet nominálních hodnot a další.



Obr. 1 Schéma procesu zpracování dat v RapidMineru

Data lze do RapidMineru importovat z databází a ze souborů v různých formátech. Podporován je například formát CSV, Excel, BibTex, z databází například PostgreSQL, MySQL a Oracle. Jsou rovněž k dispozici odpovídající operátory pro export dat.

Operátory a popis procesu zpracování dat

Operátory RapidMineru lze zařadit do následujících hlavních kategorií

- Import/export dat
- Transformace dat
- Modelování
- Validace
- Reportování
- Řízení procesu zpracování

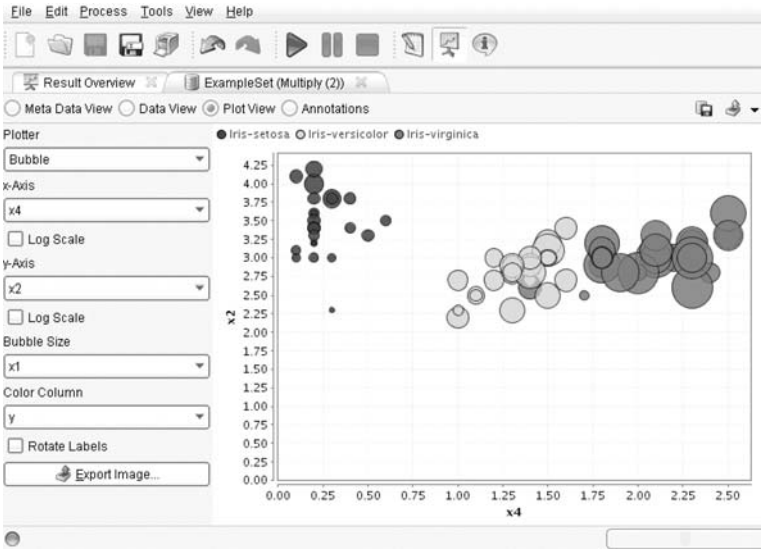
Kromě těchto základních skupin zde najdeme skupiny operátorů specializovaných na určitou oblast, například operátory pro zpracování textu (Text Processing), časových řad (Series) a informací z Webu (Web Mining).

Operátory pro transformaci dat se uplatňují hlavně ve fázi předzpracování a transformace dat. Je zde podpora pro normalizaci a transformaci hodnot, konverzi typů, nahrazování chybějících hodnot, detekci odlehlých hodnot, filtraci dat a redukci atributů.

RapidMiner podporuje širokou škálu modelů a klasifikátorů, například regresní modely (lineární, polynomiální a logistická regrese), bayesovské klasifikátory, rozhodovací stromy, asociační pravidla, neuronové sítě a operátory pro shlukovou analýzu. Operátory tohoto druhu nalezneme v kategorii Modelování.

Do kategorie Validace jsou zařazeny operátory pro cross-validaci, metody pro statistické vyhodnocení výkonnosti a statistické porovnání modelů. Operátory pro reportování podporují tvorbu zpráv, kde jsou přehledně textově a graficky prezentovány výsledky data miningu. Reporty je možné ukládat do souborů v různých formátech, typicky v PDF.

Operátory řízení procesu podporují podmíněné větvení, cykly. Z operátorů lze taktéž vytvářet funkční bloky (Subprocesses), které je možné jako podprocesy opakovaně zařazovat do procesu zpracování. Operátory typu Loop (smyčka), Branch (podmíněné větvení) dovolují opakované, případně podmíněné, provádění podprocesů. To dovoluje snadnou implementaci různých optimalizačních technik. Například lze vytvořit množinu modelů (kde počet je proměnný), modely otestovat a následně vybrat nejlepší model. Jiným příkladem může být vytvoření množiny modelů spojených do ensemblu s lepšími výslednými vlastnostmi oproti dílčím modelům, případně nalezení optimálního nastavení parametrů algoritmů zapouzdřených v operátorech.



Obr. 2 Vizualizace dat v RapidMineru

Vizualizace dat

Každé schéma procesu zpracování dat má na pravé straně přípojné body označené jako `res` (viz obr. 1 pravý okraj schématu). K těmto bodům je možno připojovat výstupy z operátorů. Datové objekty (data, modely, výsledky), které jsou poslány do těchto přípojných bodů jsou zobrazeny v prohlížeči výsledků. Každý objekt má specifickou formu vizualizace jak v textové, tak grafické formě.

Nejbohatší vizualizaci má třída ExampleSet, která se skládá z několika pohledů

- Meta Data
- Data View
- Plot View
- Annotations.

Meta Data a Data View poskytují informace o datech v textové podobě, Data View je zobrazením dat v podobě tabulky. Plot View (viz obr. 2) je laboratoří, kde si data můžeme zobrazit v různých typech grafů a vybrat si, jaké atributy k jakým souřadnicím přiřadíme. Protože se obvykle zpracovávají vícedimenzionální data je grafické zobrazení dat klíčové nejen pro názornou prezentaci, ale hlavně pro vlastní proces analýzy. Grafické zobrazení nám přiblíží strukturu dat, ukáže shluky, jejich vzájemné polohy a pomůže přinést určitou představu o datech, kterou můžeme s výhodou zhodnotit při návrhu procesu zpracování dat.

Externí knihovny a vlastní operátory

Množství operátorů, zvláště s pokročilými algoritmy, určuje schopnosti celé platformy. Pro zvýšení počtu operátorů vývojáři, kromě vlastní implementace algoritmů, využívají externí knihovny tak, že operátor se stane rozhraním pro algoritmus v externí knihovně. Takto byly zařazeny algoritmy systému Weka[1] a rozhraní pro prostředí statistického jazyka R[2].

RapidMiner je možné rozšířit i o vlastní operátory. Operátory, stejně jako celá platforma, jsou napsány v Javě. Je nutné zdědit bázovou třídu pro operátor požadovaného druhu, definovat parametry operátoru a přepsat výkonné metody, které implementují proces zpracování dat. U datových objektů, konkrétně modelů, je třeba doplnit výkonnou metodu aplikace operátoru (modelu) na data (na instanci třídy ExampleSet) a vizualizaci modelu.

[1] Weka, Data Mining Software in Java, http://www.cs.waikato.ac.nz/ml/weka
[2] The R Project for Statistical Computing, http://www.r-project.org/

Závěr

RapidMiner je sofistikovaná platforma pro data mining. Podporuje celý proces KDD od importu dat, přes předzpracování, modelování a vyhodnocování, až po vizualizaci dat a tvorbu reportů. Grafická forma popisu procesu zpracování dat dovoluje rychlé získání výsledků, bez nutnosti programování. Je vhodný jak pro použití ve firemní praxi, tak pro výukové a vědecké účely.

Literatura

[1] Berka, P.: *Dobývání znalostí z databází.* Nakladatelství ACADEMIA, 2003. ISBN 80-200-1062-9.

[2] *RapidMiner 5.0 – User Manual*, Technická dokumentace k programu RapidMiner. Rapid-I GmbH, 2010.
http://garr.dl.sourceforge.net/project/rapidminer/1.%20RapidMiner/5.0/rapidminer-5.0-manual-english_v1.0.pdf

Historie a současnost IPv6

Pavel Satrapa

E-mail: Pavel.Satrapa@tul.cz

Abstrakt

V tomto příspěvku se podíváme na ušlechtilá předsevzetí, která IPv6 dostalo do vínku, jak se je podařilo naplnit a co IETF dělá ke zlepšení dlouhodobě váznoucího nasazení nového internetového protokolu.

IPv6 – plány a skutečnost

IPv6 má úspěšně za sebou svá dětská léta – koncem loňského roku oslavilo patnácté narozeniny. Pojďme se podívat, jak se naplnily věštby pronesené sudičkami Stevem Deeringem a Robertem Hindenem u jeho kolébky.

Připomeňme, že hlavní motivací vzniku IPv6 v první polovině devadesátých let byla prognóza vyčerpání adres stávajícího IPv4, k němuž mělo podle tehdejších odhadů dojít kolem roku 2002 až 2003. Jelikož bylo poměrně dost času, rozhodli se autoři nového protokolu nenavrhnout jen minimalistické rozšíření IPv4 s větším adresním prostorem, ale zcela nový protokol, který nabídne řadu úžasných nových funkcí. Stanovili si následující cíle:

- *Rozsáhlý adresní prostor s de facto nekonečnou životností.*

  Tohle se jednoznačně povedlo. 128 bitů dlouhé adresy poskytují monstrózní počet možných adres, jehož vyčerpání se zdá být neuskutečnitelné. Jako drobnou sabotáž lze vnímat, že celá polovina adresy byla věnována identifikátoru rozhraní, tedy rozlišení strojů v rámci jedné podsítě. To je na jedné straně brutálně předimenzováno, na straně druhé to zjednodušuje některé mechanismy.

  V současnosti se přidělují jen adresy s binárním prefixem 001, to znamená jen 1/8 dostupného prostoru. Pokud by se později tempo spotřeby řádově zrychlilo, je stále prostor definovat pro valnou většinu adresního prostoru jinou strukturu – s kratším identifikátorem rozhraní a větším počtem podsítí.

- *Tři druhy adres – individuální, skupinové a výběrové.*

  Splněno. Ovšem nedá se říci, že by v tomto bodě IPv6 získalo nějakou výhodu proti IPv4. V něm zmiňované tři druhy adres existují také a úroveň jejich podpory je víceméně srovnatelná.

- *Jednotné adresní schéma pro Internet i vnitřní sítě.*

  Podařilo se. Zde má IPv6 díky své mnohem delší adrese jednoznačnou výhodu. Zatímco v IPv4 se dnes bojuje o každý bit v adrese sítě, podsítě a rozhraní, IPv6 si může zahýřit. Není třeba používat privátní rozsahy (přestože jsou pro zájemce k dispozici v podobě ULA) a také struktura adresy může být jednotná.

  Sice se ustoupilo od zcela unifikovaného přístupu RFC 3177, které doporučovalo sítím přidělovat prefixy délky 48 bitů, dalších 16 b měla být adresa podsítě a závěrečných 64 b adresa rozhraní. Podle nového RFC 6177 rozhodují o délce prefixu sítě internetové registry, ovšem v praxi se až na výjimky přidělují délky 48, 56 nebo 64 bitů. Celkově je adresní prostor IPv6 výrazně soudržnější než IPv4.

- *Hierarchické směrování.*

  IPv6 těží z toho, že je mladší a začalo se nasazovat až v době, kdy vládl CIDR a hierarchické směrování bylo prioritou. IPv4 si nese dědictví chaotického přidělování prefixů z minulosti a do budoucna se na jeho stavu nepochybně podepíše vyčerpání adres, které povede k převodům částí adresního prostoru na jiné vlastníky. Směrovací tabulky IPv4 se díky tomu budou drobit stále více a více. Současný hardware je ovšem bez problémů zvládá.

- *Bezpečnostní prvky přímo v základním protokolu.*

  IPsec se svými hlavičkami AH pro autentizaci a ESP pro šifrování sice je oficiálně součástí protokolu, ovšem definice existuje i pro IPv4 a nemám pocit, že by se kvalita implementací mezi oběma protokoly nějak dramaticky lišila.

- *Podpora pro služby se zajištěnou kvalitou.*

  Pro tento účel zařadili autoři do základní IPv6 hlavičky položky *Třída provozu* a *Značka toku*. První se používá pro klasifikaci paketů v mechanismech typu DiffServ. Ohledně toků panuje bezradnost připomínající TOS v IPv4. Ve vlastní definici IPv6 je pouze řečeno, že budou popsány později, ovšem reálně k tomu dosud nedošlo. Vzácné výjimky, jako například RFC 3697, představují jen minimální pokrok a dosud neexistuje představa, jak vlastně koncept toků uchopit a využít.

- *Optimalizace pro vysokorychlostní směrování.*

  Snaha o rychlejší průchod směrovači se odrazila ve zjednodušení formátu IPv6 hlavičky a odstranění položek, které komplikovaly a zpomalovaly zpracování (kontrolní součet, fragmentace). V praxi se ovšem prokázala některá úskalí konceptu zřetězení rozšiřujících hlaviček, jimiž lze doplňovat dodatečné informace. Správně by měly být uspořádány tak, aby se jimi směrovač mohl co nejdříve přestat zabývat a datagram odeslat. Požadavek äbuď tolerantní k příchozím paketůmô ovšem vede k tomu, že směrovače přesto zkoumají celý řetěz hlaviček. Důsledkem jsou DoS útoky využívající enormně dlouhé sekvence rozšiřujících hlaviček, které již byly zaznamenány.

- *Automatická konfigurace, pokud možno plug-and-play.*

  Plug-and-play se podařilo v podobě automatické bezstavové konfigurace, která je tak jednoduchá a báječná, až se z toho nejednomu správci sítě, jenž by rád měl svou infrastrukturu pod kontrolou, ježí hrůzou vlasy na hlavě. Až bude implementováno jeho poslední rozšíření o informace pro DNS podle RFC 6106, nebude mu už nic podstatného chybět.

  Hořkou pilulkou v oblasti automatických konfigurací je neutěšený stav DHCPv6. Vznikalo překvapivě dlouho a jeho výsledná podoba budí rozpaky. Nahrazení MAC adresy při rozpoznávání klientů identifikátorem DUID znatelně komplikuje jeho použitelnost. Výměna síťové karty nyní sice nezpůsobí změnu identifikátoru příslušného stroje, zato se DUID změní při reinstalaci systému. Nemluvě o tom, že pokud na jeden počítač nainstalujete několik operačních systémů, bude mít každý z nich odlišný identifikátor. Chybějící možnost předat adresu implicitní brány také DHCPv6 na atraktivitě nepřidá.

  Počítače si navíc vytvářejí krátkodobé náhodné adresy podle RFC 4941 s cílem chránit soukromí uživatelů, které jsou ovšem pro správce obtížně uchopitelné. Je třeba se smířit s tím, že IPv6 sítě se hůře kontrolují. Chce-li mít správce přehled o zapojených počítačích a jejich uživatelích, je pravděpodobně nejschůdnější cestou uvolnit pravidla z hlediska adres a kontrolovat uživatele, například pomocí IEEE 802.1X.

- *Podpora mobility.*

  Komunikace s mobilními zařízeními je v IPv6 nepochybně vyřešena lépe než v předchozím protokolu. Reálného rozšíření se však zatím příliš nedočkala a zůstává jen papírovou předností.

- *Hladký a plynulý přechod z IPv4.*

  Tuto oblast nelze hodnotit jinak, než jako naprosté selhání. Podle původních představ nemělo k vyčerpání IPv4 adres vlastně vůbec dojít. Během uplynulých let měl Internet postupně přejít ze starého protokolu na nový a o IPv4 neměl být touto dobou prakticky žádný zájem. Ve skutečnosti počátkem února 2011 IANA alokovala poslední volné bloky IPv4 adres a životnost jejich zásob na úrovni regionálních registrů se počítá spíše na měsíce než na roky. V době konání konference podle všeho bude asijský APNIC již bez adres, evropskému RIPE NCC je předpovídáno vyčerpání na přelomu let 2011/2012.

  Hlavním problémem je, že poskytovatelům připojení a obsahu chybí motivace. Nasazením CIDR, NATů a přísnějších pravidel pro přidělování adres se spotřeba adresního prostoru IPv4 výrazně zpomalila. Ve druhé polovině 90. let se zdálo, že k vyčerpání dojde nejdříve kolem roku 2020, takže není kam spěchat a IPv6 bylo většinou poskytovatelů odloženo na neurčito. Po roce 2000 se prognózy začaly postupně zhoršovat, Internet však mezitím opět významně narostl a byl ještě o něco méně ochotný se měnit. Ekonomické faktory – nasazení IPv6 znamená náklady, jejichž návratnost je ovšem problematická – stály a dosud stojí proti novému protokolu.

  Přestože vznikla řada přechodových mechanismů, vůle k jejich použití byla spíše malá. V současnosti se poměrně běžně lze setkat s různými variantami tunelů, které umožňují přenos IPv6 běžným IPv4 Internetem. Manuální tunely, 6to4, ISATAP či Teredo nejsou žádnou vzácností a umožňují přenášet IPv6 prakticky kamkoli, bez vazby na síť poskytovatele.

  Hodně palčivá je otázka vzájemného překladu protokolů. Jestliže IPv6 není shora kompatibilní s IPv4, potřebujeme alespoň nějaký překladač mezi nimi. Dokud se IPv6 uživatelé nedostanou k IPv4 zdrojům, budou se novému protokolu bránit, protože je omezuje. V této oblasti má IETF co dohánět. Postarší NAT-PT (RFC 2766), který se právě o překlad mezi IPv4 a IPv6 snažil, byl v RFC 4966 odsunut na smetiště dějin bez náhrady. V současnosti skupina Behave usilovně pracuje na jeho náhradě, zjednodušené verzi nazvané NAT64, ovšem tato snaha přichází hodně pozdě.

IPv6 v uplynulých letech dozrálo a praktické zkušenosti s ním se odrazily v nových generacích jeho dokumentů. Dočkalo se implementace v nejdůležitějších operačních systémech a platformách. Často je implicitně zapnuto a pokud je k dispozici, dostává přednost. Přesto je v praxi využíváno jen minimálně. Statistika největšího evropského peeringového centra AMS-IX vykazuje ve špičkách téměř 2,5 Gb/s IPv6 provozu. To nevypadá jako špatné číslo, ovšem v porovnání se 750 Gb/s IPv4 se jedná zhruba o třetinu procenta celkových datových toků procházejících infrastrukturou AMS-IX.

Co na to IETF?

Nabízí se otázka, jakým způsobem se IETF snaží situaci zlepšit a posunout IPv6 dál. V současnosti jsou relevantní zejména následující pracovní skupiny:

- *IPv6 Maintenance (6man)* má na starosti základní specifikace. Jak název napovídá, nesnaží se o heroický vývoj, spíše o jejich údržbu a postupné vylepšování.
- *IPv6 Operations (v6ops)* se zabývá otázkami praktického nasazení IPv6. Popsala například způsob přidělování prefixů IPv6 adres (RFC 6177), požadavky na základní zabezpečení domácích směrovačů (RFC 6092) nebo scénáře pro nasazení IPv6 v sítích poskytovatelů Internetu (RFC 6036), abychom jmenovali několik příkladů z poslední doby.
- *Behavior Engineering for Hindrance Avoidance (Behave)* řeší otázky NATů a překladu protokolů. Pro budoucnost IPv6 je velmi důležitá, protože právě zde vzniká specifikace NAT64 a spřízněného DNS64, které by měly zajistit vzájemný překlad mezi IPv6 a IPv4.
- *Softwires* se orientuje na přenos jednoho protokolu po druhém, tedy na otázky tunelování. Hledá způsoby, jak v sítích současných ISP nabídnout zákazníkům IPv6 a jak do budoucna zpřístupnit IPv4, pokud páteřní sítě pro nedostatek adres přejdou na IPv6.
- *Site Multihoming by IPv6 Intermediation (Shim6)* vyvíjí metody pro připojení sítě k několika poskytovatelům, tedy IPv6 multihoming.
- *IPv6 over Low power WPAN (6lowpan)* je spíše okrajová pracovní skupina s velmi úzce vymezeným záběrem. Je jím přenos IPv6 po sítích vycházejících ze standardu IEEE 802.15.4 (např. MiWi).

Pro blízkou budoucnost a nasazení IPv6 jsou pravděpodobně nejdůležitější skupiny *Behave* a *Softwires*, na jejichž půdě vznikají mechanismy pro koexistenci obou verzí IP. Podívejme se na ně podrobněji.

NAT64 a DNS64

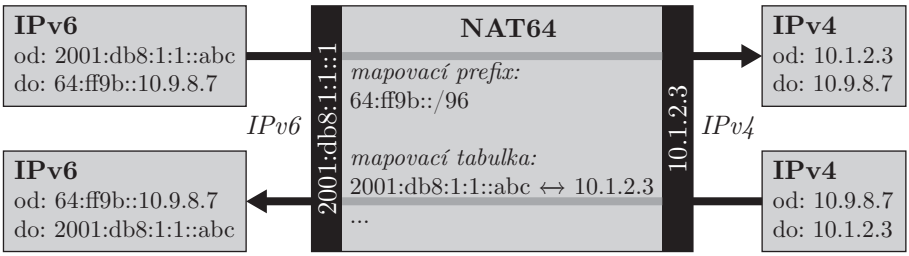
Základem pro překlad paketů mezi IPv4 a IPv6 je letité *Stateless IP/ICMP Translation Algorithm (SIIT)* definované v RFC 2765, jehož aktualizace je na spadnutí – viz [7]. Stanoví celkem jednoduchá pravidla, podle kterých se vzájemně převádějí jednotlivé položky v hlavičkách IPv4 a IPv6 datagramů, včetně řady různých omezení – zejména rozšiřující hlavičky nelze ve druhém protokolu

často vyjádřit. SIIT ovšem nechává nedořešenou klíčovou otázku, a sice jak mapovat adresy. Je zjevné, že zejména vyjádření IPv6 prostřednictvím mnohem kratšího IPv4 nebude snadné.

Tuto oblast doplňuje mechanismus označovaný jako *NAT64* a popsaný v [2]. Stručně řečeno funguje podobně jakou současné IPv4 NATy – udržuje si tabulku se vzájemným mapováním IPv4 a IPv6 adres. Na rozdíl od NAT-PT, jehož následníkem má být, je NAT64 koncipován jako asymetrický. Navázat spojení půjde jen ve směru z IPv6 do IPv4, nikoli opačně (podobně jako současné NATy umožňují zahájit spojení jen směrem z koncové sítě ven). Jakmile je spojení navázáno a vytvořena příslušná položka v mapovací tabulce, komunikace pokračuje bez omezení v obou směrech.

Mapovací tabulka je ve stutečnosti potřeba jen ve směru z IPv4 do IPv6 (proto ona jednosměrnost). Mapování IPv4 adres do IPv6 je triviální – je pro něj definován určitý prefix (lze použít 64:ff9b::/96 podle RFC 6052 nebo libovolný jiný, který přidělí správce sítě) a IPv4 adresy se jednoduše připojují za něj. Zároveň NAT64 šíří pomocí běžných směrovacích protokolů do IPv6 sítě informaci, že právě jeho prostřednictvím je dostupná síť s tímto prefixem.

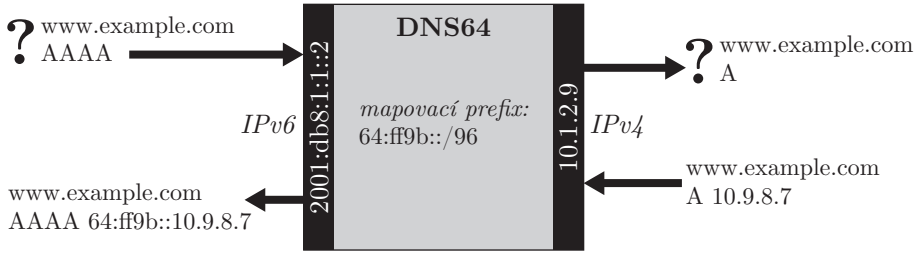
Jakmile mu dorazí po IPv6 paket s takto vytvořenou cílovou adresou, vyzvedne z ní IPv4 adresu a datagram přeloží do IPv4 podle pravidel SIIT. Pokud pro odesilatele ještě nemá založenou položku v mapovací tabulce, přidělí mu některou ze svých IPv4 adres v kombinaci s vhodným číslem portu a údaje si zaznamená do mapovací tabulky. Jinak použije již existující položku. Příslušné údaje vloží do odeslaného IPv4 datagram jako odesilatele. Dorazí-li odpověď, má v mapovací tabulce uloženo, jak přeložit její cíl z IPv4 do IPv6. Odesilatelovu IPv4 adresu mapuje jednoduše připojením za pevně daný prefix.



Obr. 1: Základní funkce NAT64

Díky popsanému mechanismu mohou IPv6 klienti z koncových IPv6 sítí komunikovat s IPv4 servery, což je hlavním úkolem NAT64. Zbývá ovšem dořešit, jak získají cílovou adresu. Typicky ji dodá DNS, a proto je třeba vhodně zasáhnout i do jeho práce. Toto má na starosti specifikace *DNS64*, která se s NAT64 vhodně doplňuje. DNS64 je typicky implementováno jako součást místního rekurzivního serveru, který obsluhuje klienty v dané síti.

IPv6 klienti pochopitelně kladou DNS dotazy ohledně IPv6 adres, tedy poptávají záznamy typu AAAA. Dorazí-li na takový dotaz úspěšná odpověď, bude standardním způsobem předána tazateli a není třeba sahat k žádným mimořádnostem. Jestliže ale příchozí odpověď znamená, že hledané jméno sice existuje, ale není pro ně k dispozici žádný záznam typu AAAA, server implementující DNS64 se zkusí zeptat na záznamy typu A pro stejné jméno. Když uspěje, převede je na záznamy AAAA, IPv4 adresy spojí s prefixem pro jejich mapování (musí tedy znát prefix používaný NAT64) a pošle klientovi takto upravenou odpověď.



Obr. 2: DNS64 a jeho úpravy DNS dotazů a odpovědí

Klient následně odešle paket na adresu s mapovanou IPv4 adresou, jeho paket dorazí zařízení implementujícímu NAT64 a bude zpracován výše popsaným postupem.

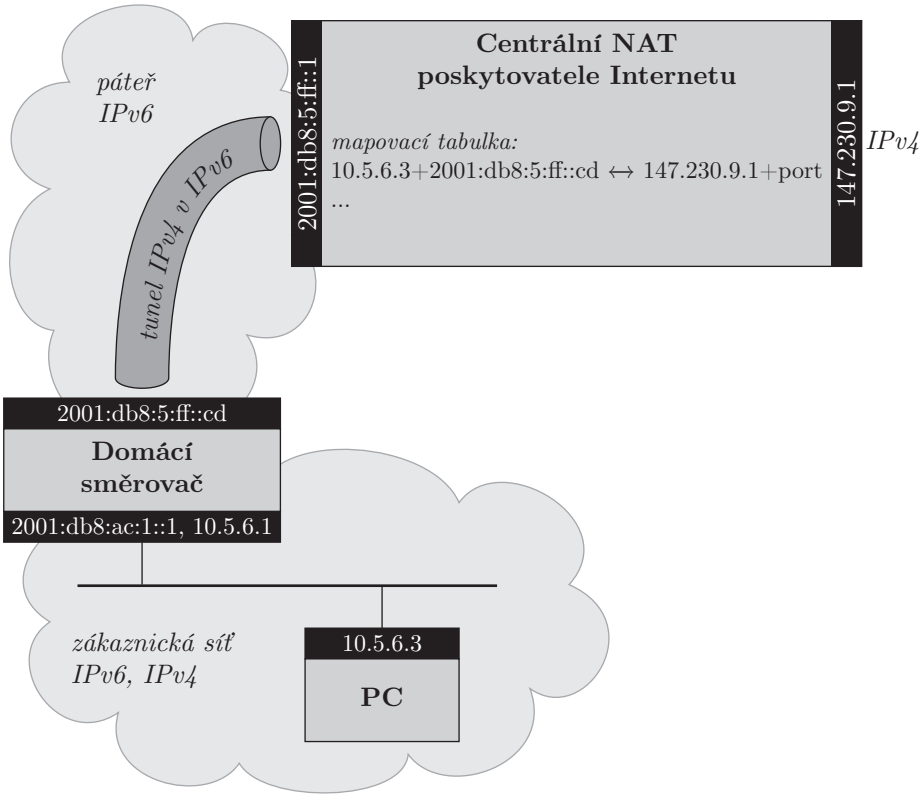
Dual-Stack Lite

Jestliže skupina *Behave* se snaží zpřístupnit IPv6 uživatelům IPv4 zdroje pomocí překladu paketů, skupina *Softwires* usiluje o podobné cíle odlišnou metodou – provozováním koncových sítí s oběma protokoly i v případech, kdy poskytovatelova síť nabízí jen jeden. Vzniká zde několik různých specifikací, za nejvýznamnější ovšem považuji *Dual-Stack Lite*.

Předpokládá, že poskytovatelé připojení ve svých páteřních sítích postupně přejdou na IPv6, protože veřejné IPv4 adresy pro ně nebudou mít k dispozici a používání neveřejných s několika stupni NATování bude v provozu neúnosně složité. Dual-Stack Lite i v tomto prostředí umožní nabídnout domácím uživatelům kromě IPv6 také IPv4. Sice s neveřejnými adresami, ovšem ani dnes nic výrazně jiného nemají (aneb 10.0.0.0/8 do každé rodiny).

Myšlenka je taková, že dnešní model, kdy každý máme doma jednoduchý NAT, bude nahrazen jedním (či několika málo) centrálním NATem na straně poskytovatele. Domácí směrovač bude IPv6 doručovat nativně, zatímco IPv4

datagramy zabalí a tunelem pošle po IPv6 centrálnímu NATu. Ten bude připojen k IPv4 Internetu a neveřejné IPv4 adresy klientů bude mapovat na své vlastní (v kombinaci s čísly portů), podobně jako je mapují současné zákaznické NATy. K rozlišení koncových počítačů bude kromě IPv4 adres používat i IPv6 adresy domácích směrovačů zajišťujících tunelování, takže snadno lze používat stejné neveřejné IPv4 adresy u různých zákazníků.



Obr. 3: Přeprava IPv4 datagramu podle Dual-Stack Lite

Poskytovatel připojení tedy nemusí rozvádět IPv4 do celé své sítě. Navíc díky sdílení stejných IPv4 adres mnoha zákazníky dokáže svou omezenou zásobu veřejných IPv4 adres využít efektivněji. Podrobnější popis najdete v [6].

Co bude dál?

Předvídat budoucnost je samozřejmě ošidné. Aktuálně se zdá, že mnohaletá fáze přešlapování a okukování pozvolna přechází do skutečného nasazování IPv6. Má-li ve větší míře uspět, musí být splněny následující dvě podmínky:

- *Uživatelé IPv6 nesmí být znevýhodněni.*

  To v praxi znamená, že musí mít přístup ke zdrojům a službám poskytovaným po IPv4. Nezáleží příliš na tom, zda toho dosáhneme překladem protokolů či dual-stack přístupem s vhodným tunelováním. Dosažitelnost IPv4 je ale klíčová, protože řada provozovatelů síťových služeb a zdrojů evidentně nehodlá v nejbližší době přejít na nový protokol a pokud bude IPv6 Internet nabízet zlomek toho co jeho IPv4 předchůdce, těžko se o něj kdo bude zajímat.

  Současné aktivity IETF vzbuzují naději, že v dohledné době vzniknou řešení. Klíčová samozřejmě bude rychlost implementace nových standardů a jejich dostupnost v běžně používaných síťových prvcích.

- *Poskytovatelům se musí vyplatit.*

  Nasazení IPv6 vyžaduje náklady na vybavení i na personál. Nejsou velké. Vybavení podléhá přirozené obměně a vzdělávání zaměstnanců nepředstavuje dramatické částky. Na druhé straně i nevelké náklady musí něco přinést, aby měly smysl.

  Tady pro IPv6 pracuje vyčerpání adres. Ve světě IPv4 nás v dohledné době nečeká nic krásného – všeobecně se předpokládá, že adresy se budou kupovat a sítě budou běžně NATovány na několika úrovních. Jejich provoz bude složitější, a proto méně spolehlivý a dražší. Během pár let se IPv6 může posunout ze současné komplikace do pozice jednoduššího a levnějšího protokolu.

Potřeba přistupovat ke zdrojům ze světa IPv4 je klíčová především pro budoucnost, až bude problém získat IPv4 adresu. V současné době nás obvykle nic nenutí stavět sítě podporující výlučně IPv6. Daleko častěji se vyskytuje společně s IPv4 v režimu dual stack, kdy síť má přístup jak k IPv4, tak k IPv6 Internetu.

Takové nasazení je reálně použitelné již dnes, přestože některé složky IPv6 dosud nejsou dotaženy. Na TU v Liberci nativně provozujeme oba protokoly paralelně skoro v celé síti již více než 5 let. Máme v DNS AAAA záznamy pro univerzitní web i poštovní servery. Z vnitřních strojů TU přichází kolem 10 % dopisů a 15–30 % WWW požadavků po IPv6. Vstoupili jsme do IPv6 projektu Google, takže pro jeho stroje dostáváme z DNS záznamy typu AAAA a komunikujeme s nimi přednostně po IPv6. Nefunkční IPv6 by proto znamenalo bezprostřední

omezení řady uživatelů. K problémům tohoto typu však nedochází. Podle našich zkušeností je IPv6 dostatečně zralé a stabilní k produkčnímu nasazení.

Že zájem o ně přestává být zanedbatelný dokládají údaje ze zprávy o stavu domény *cz* za loňský rok, viz [5]. Jejích přibližně 750 tisíc poddomén obsahuje 150 tisíc záznamů s IPv6 adresami DNS serverů (20,3 % záznamů tohoto typu), přes 60 tisíc poštovních serverů (8,6 %) a necelých 40 tisíc web serverů (5,2 %). To považuji za nejlepší důkaz, že IPv6 se začíná konečně stávat realitou.

Literatura

[1] Bagnulo, F., Matthews, P., Beijnum, I. van: *Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers.* RFC 6146 (před vydáním), IETF, 2011.

[2] Bagnulo, F., Sullivan, A., Matthews, P., Beijnum, I. van: *DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers.* RFC 6147 (před vydáním), IETF, 2011.

[3] Baker, F., Li, X., Yin, K.: *Framework for IPv4/IPv6 Translation.* RFC 6144 (před vydáním), IETF, 2011.

[4] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., Li, X.: *IPv6 Addressing of IPv4/IPv6 Translators.* RFC 6052, IETF, 2010.

[5] kolektiv autorů: *Domain Report 2010.* CZ-NIC, 2011.
http://www.nic.cz/files/nic/doc/Domain_Report_2010_CZ.pdf

[6] Durand, A., Droms, R., Woodyatt, J., Lee, Y.: *Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion.* draft-ietf-softwire-dual-stack-lite-07, IETF, 2011.

[7] Li, X., Bao, C., Baker, F.: *IP/ICMP Translation Algorithm.* RFC 6145 (před vydáním), IETF, 2011.

Tak nám došly. A co dál?

Ondřej Filip

E-mail: ondrej.filip@nic.cz

Chybka

Lidé, kteří vymysleli internetové standardy, byli rozhodně geniální a měli obrovskou nadčasovou vizi. Ale i na jejich případě lze ukázat, že člověk je tvor omylný a že i do jejich úžasného návrhu se vloudila chybička. Tou chybou byla volba délky IP adresy, kterou stanovili na 32 bitů, tedy zhruba na 4 miliardy adres. Vzhledem k tomu, že počet připojených zařízení nám dramaticky roste a že adresy rozhodně nejsou 100% využitelné, je zřejmé, že nám musí jednou dojít. A tato fáze se odehrává právě nyní. Ale začneme od začátku.

Správa adres

Správa IP adres je hierarchická. Nejvyšším správcem je organizace IANA, která přiděluje velké bloky pěti regionálním registrům (RIR), což jsou AfriNIC, APNIC, ARIN, LACNIC, a RIPE NCC a RIRy zase přidělují adresy dále lokálním registrům (LIR), což jsou obvykle jednotliví poskytovatelé služeb. V případě IPv4 rozdávala IANA bloky velikosti /8, nebo-li 16,8 miliónů adres. Tento systém fungoval tak, že každý RIR měl určitou zásobu adres a zároveň měl stanovenu mez, pod kterou zásoba neměla klesnout. Pokud k tomu došlo, RIR okamžitě požádal o určitý počet bloků /8 tak, aby mu adresy vydržely přibližně 18 měsíců. V praxi to vypadalo tak, že velké registry (RIPE NCC, ARIN, APNIC) měly dolní mez na dvou blocích a žádaly o další 2 bloky. Malé registry měly tuto mez pod jedním blokem a alokovaly jeden blok.

Poslední bloky

Tento systém fungoval až do prvního února, kdy IANA přidělila 2 bloky APNICu a zbylo jí 5 posledních bloků. Ohledně těchto bloků existovala dohoda, že každý RIR dostane po jednom. To nastalo již o dva dny později a tím IANA definitivně přestala přidělovat IPv4 adresy. Vedle těchto pěti bloků, ještě existovalo jedno přidělení adres. Ještě před vznikem RIRů se adresy alokovaly přímo

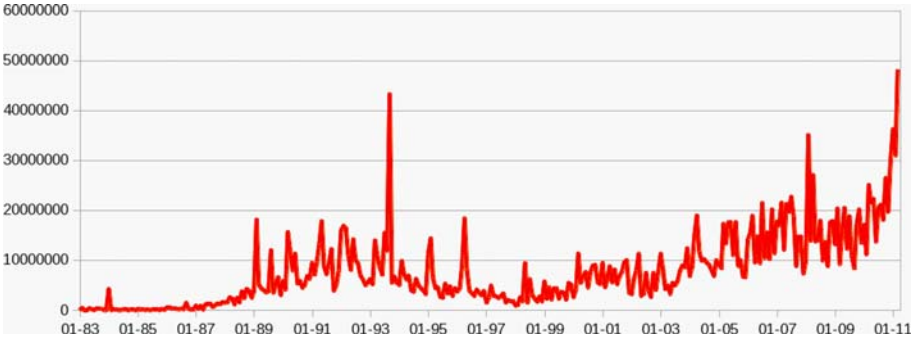
z registru IANA, který podobně jako dnešní RIRy prostě postupně parceloval některé bloky. Po vzniku RIRů se od této praxe upustilo a bloky použité pro tyto alokace byly označeny jako LEGACY a byly přiděleny ke správě jednotlivým RIRům, které je nechávaly s výjimkou bloku 188/8 u RIPE NCC netknuté. I tyto adresy byly rozděleny rovnoměrně mezi RIRy, tedy kromě onoho zmíněného bloku. Bohužel rozhodnutí rovnoměrné rozdělit posledních IPv4 adresy nebylo vůbec racionální. Totiž každý LIR spotřebovává adresy zcela jinou rychlostí. Jen pro ilustraci, AfriNICu byly za celou historii přiděleny 4 bloky. Naopak APNIC jich dostal 47. A tak vlivem rozdělení posledních IPv4 adres došlo k tomu, že zatímco Afričané mají zásobu IPv4 adres na poměrně dlouho (cca 4 roky), v regionu Asie-Pacifik adresy fakticky došly v půlce dubna.

Asie-Pacifik

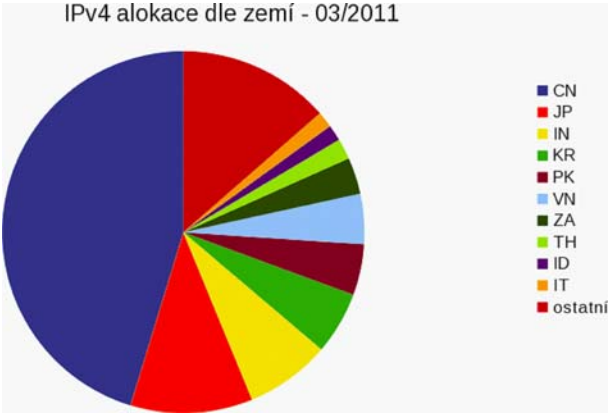
Tedy pro upřesnění, nějaké IPv4 adresy v APNICu ještě zbývají, ale pro jejich přidělování se uplatňuje poměrně dramatické pravidlo. To říká, že od zmíněného data je možné žádat pouze o alokace minimální velikosti, což je v současné době /22 nebo-li 1024 adres, a zároveň, že každý člen může žádat z tohoto bloku pouze jednou. Dlužno podotknout je, že těchto malých alokací lze provést až 16 384, což je násobně více, než je v současnosti členů APNICu. To znamená, že volné IPv4 adresy v tomto regionu budou ještě hodně dlouho, byť už APNIC fakticky nepřiděluje.

Rychlost alokace

IPv4 adresy nakonec došly překvapivě dříve, než se čekalo. LIRové poměrné dlouho ignorovaly fakt, že zásoby IPv4 adres se ztenčují, ale ke konci vzaly RIRy doslova útokem a začaly alokační dostihy. Vysokou alokační rychlost v závěru nejlépe ilustruje obr. 1.



Obr. 1



Obr. 2

Drtivý podíl na tomto tempu měl právě region Asie-Pacifik a v něm konkrétně Čína, což ukazuje koláčový graf (obr. 2) za letošní březen.

Díky konci adres v Asii nám alokační rychlost rozhodně poklesne. Ale dá se očekávat, že stejně hysterie se bude opakovat i v dalších regionech, kde IPv4 adres ubývá, jde především o Evropu a Severní Ameriku. U nás v Evropě mohou adresy dojít ještě letos.

A co dál?

Alespoň v jedno regionu adresy došly. A co bude dál? Díky faktu, že v některých částech světa adresy jsou a v jiných ne, dojde rozhodně k tomu, že se objeví nějaké polo-legální cestičky, jak převést IPv4 adresy z málo aktivních regionů do těch aktivnějších. Tímto by mohlo v příštím roce dojít k postupnému vyprázdnění všech RIRů. Trh se fakticky zablokuje, protože noví hráči nebudou moci vstupovat. Přestanou vznikat nové LIRy pro IPv4 adresy. Tato fáze jistě přesvědčí mnohé hráče, že přechod na IPv6 je zcela nezbytný a je pravděpodobné, že rozvoj IPv6 bude značně akcelerovat. S IPv4 adresami se bude obchodovat a jejich cena bude rychle vzrůstat. Díky tomuto obchodování dojde v silnému růstu velikosti globální směrovací tabulky. Některé směrovače nebudou mít dostatečnou kapacitu a tak jejich majitelé mohou začít některé prefixy filtrovat. Celková spolehlivost Internetu tak může z pohledu koncových uživatelů klesat. Díky nedostatku adres budou někteří ISP nuceni nasazovat carrier grade NATy, což bude v konečném důsledku opět znamenat zhoršení kvality služeb. Lze jen obtížně určit, jak dlouho bude toto období překupování adres trvat, ale stoupající cena IPv4 adres a jejich špatná „obchodovatelnost“ v podstatě okamžitě

znamená, že ne všechny projekty bude moci na IPv4 provozovat. Ruku v ruce s tímto bude klesat kvalita IPv4 připojení a IPv6 bude čím dál více rozšířené. Hlavně P2P služby a media rich weby budou mnohem spolehlivěji fungovat na protokolu IPv6, což by mohlo nastartovat poptávku běžných koncových uživatelů po tomto protokolu. Není pravděpodobné, že se objeví nějaká nová technologie, která by život IPv4 mohla prodloužit a tak v tomto období budou ISP i poskytovatelé služeb prakticky donuceni k přechodu na IPv6. Od určité hranice tohoto přechodu se začnou objevovat služby, které budou specificky využívat protokolu IPv6 a začne se objevovat pouze IPv6 obsah. Samozřejmě IPv4 internet s námi bude ještě hodně dlouho, ale od určitého momentu bude úplně stačit přístup pomocí NAT64 či podobných technologií a hlavní jádro Internetu bude tvořit IPv6.

Závěr

Jinými slovy, IPv6 nás čeká, ať už se nám to líbí či ne, nemá smysl strkat hlavu do písku. Je tedy nutné se na nový protokol připravit, obnovovat hardware i software právě s ohledem na tuto skutečnost. Pozvolný a pečlivě naplánovaný přechod je jistě mnohem lepší varianta, než křečovité dodělávání na poslední chvíli.

Implementace IPv6 v prostředí univerzitní sítě VŠB-TU Ostrava

Martin Pustka

E-mail: Martin.Pustka@vsb.cz

Abstrakt

Příspěvek je věnován nasazení protokolu IPv6 v prostředí univerzitní počítačové sítě Vysoké školy Báňské – Technické univerzity Ostrava. V příspěvku jsou shrnuty použité technologie, návaznosti na poskytované služby a také zkušenosti, které jsme získali v průběhu let v souvislosti s nasazováním protokolu IPv6 a podporou IT správců i uživatelů naší univerzity.

Popis počítačové sítě VŠB

Počítačovou síť VŠB lze stručně charakterizovat jako metropolitní počítačovou síť a páteřními trasami s kapacitou 10Gbps. Tato síť má přibližně 20 tis. uživatelů, 10 tis. registrovaných pevných stanic a také 250 bezdrátových přístupových bodů, které ve špičkách obsluhují okolo 1 200 uživatelů. Samotná počítačová síť obsahuje přibližně 600 aktivních prvků.

Jako v podstatě každá akademická síť, tak i tato je svým charakterem více lokální institucionální sítí. Ovšem k nezanedbatelnému množství specifických akademických uživatelů musí být v určitých situacích také sítí poskytovatele Internetu. Díky této dvojí roli nelze v síti zcela jednoduše aplikovat pravidla obvyklá v lokálních sítí. Stejně tak nelze funkci sítě zjednodušit na pouhé poskytování Internetu.

První implementace IPv6 spadají do doby okolo r. 2000, kdy protokol IPv6 začal být nasazován v počítačové síti CESNET. Tehdy nebylo možno z technických důvodů nasadit dual-stack směrovače, a proto byl do pozice směrovače nasazen PC směrovač s OS Linux.

Přibližně od let 2006–2007, kdy proběhl technologický upgrade páteřní sítě, je protokol IPv6 postupně rozšiřován v celé síti.

Topologie počítačové sítě

Připojení počítačové sítě VŠB do sítě Internet je zajišťováno prostřednictvím sítě CESNET2. Primární připojení je realizováno 10 Gbps okruhem, záložní pak okruhem s kapacitou 1 Gbps.

Při návrhu a realizaci IPv6 sítě je naší snahou držet se následujících pravidel:

- jednotná topologie sítě (IPv4 i IPv6),
- dual-stack směrovače,
- podpora přechodových mechanismů,
- výstavba nových sítí s podporou IPv4 i IPv6,
- nepoužívat IPv4 privátní adresy a technologie NAT,
- nebudovat sítě pouze s protokolem IPv6.

Pro směrování uvnitř sítě je použit interní směrovací protokol OSPFv3. Jeho nastavení je shodné jako u IPv4. Oproti IPv4 je navíc použita agregace adresních prostorů lokalit.

Pro směrování mezi sítí VŠB a sítí CESNET2 je použit protokol BGP.

Všechny směrovače podporují prokoly IPv4 i IPv6. Loopback rozhraní mají IPv4 i IPv6 adresy, které lze rovnocenně použít i pro vzdálenou správu.

Adresace

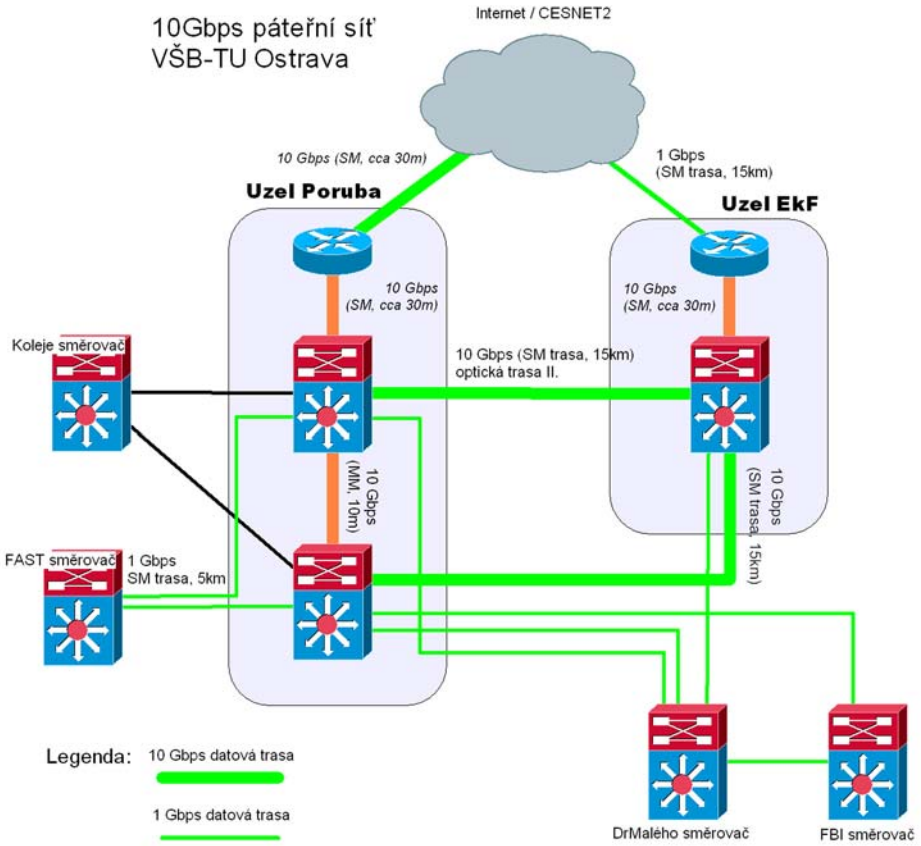
VŠB má přidělen adresní IPv6 prostor 2001:718:1001::/48. Bereme-li v úvahu, že koncovým sítím přidělujeme sítě s délkou prefixu 64 bitů, tak máme k dispozici 65 536 podsítí. Každé lokalitě byla přidělena podsíť s délkou prefixu 56 bitů, která je v počítačové síti ve směrovacím protokolu OSPF propagována jako jeden agregovaný prefix.

Lokality adresujeme 2001:718:1001:0x00::/56, kde X je číslo lokality.

U centrálních služeb, které jsou poskytovány v jedné síti, je snahou zachovat zapamatovatelnost IPv4 i IPv6 adresy. Tedy adresa 158.196.149.9 má IPv6 ekvivalent 2001:718:1001:149::9.

Přechodové mechanismy

Cílem návrhu celé sítě je impelemtovat protokol IPv6 tak, aby by; dostupný ve všech koncových sítích. To však nelze realizovat okamžitě a zcela jednoduše.



Obr. 1

Proto jsou v počítačové síti podporovány přechodové mechanismy. V našem případě se jedná zejména o podporu ISATAP tunelů.

Díky centralizované správě počítačové sítě a vyžadovanému DHCP (pro IPv4) jsme schopni koncovým stanicím přidělit i doménu vsb.cz. Windows systémy se pokoušejí vytvořit si nestavový tunel s ISATAP serverem, který je dostupný pod FQDN isatap.<doména>, v našem případě tedy isatap.vsb.cz.

ISATAP server je zprovozněn na jednom z páteřních prvků Cisco Catalyst 6500. Konfigurace je následující (uvedena je pouze příslušná část):

```
nterface Tunnel0
 description ISATAP IPv6 tunnel interface
 ipv6 address 2001:718:1001:700::1/64
```

```
ipv6 nd prefix 2001:718:1001:700::/64
tunnel source Loopback1
tunnel mode ipv6ip isatap
```

Použité technické prostředky

V pozici hraničních směrovačů jsou nasazeny moderní směrovače Cisco ASR 1004/RP1 osazené 10 Gbps rozhraními.

Páteřní síť tvoří prvky Cisco Catalyst 6500 ve dvou technických variantách. Prvky s požadavkem na vyšší výkon jsou osazeny supervisory SUP720-3B, prvky u nichž není požadavek na vysoký výkon jsou osazeny méně výkonnými supervisory SUP32. Všechny tyto prvky jsou provozovány v režimu dual-stack, použit je nemodulární IOS ve verzi 12.2(33)SXI.

V pozicích agregačních L2 přepínačů jsou nasazovány prvky Cisco Catalyst 3560/3750. Tyto prvky jsou také provozovány s podporou IPv4/IPv6. Na nich jsou uplatňovány některé bezpečnostní IPv4/6 mechanismy, používány jsou IOSy ve verzích 12.2(55)SE.

Jako přístupové přepínače pak prvky Cisco Catalyst 2950/2960. Starší prvky Cisco Catalyst 2950 IPv6 vůbec nepodporují ani pro správu, ani v oblasti jimi přepínaného provozu. Modernější prvky Cisco Catalyst 2960 podporu pro přepínaný provoz mají, část vzdálené správy lze přes IPv6 také realizovat.

Bezdrátová síť

Bezdrátová WiFi síť je centralizovaná a pro její řízení je použito zařízení Cisco WiSM WLAN service module. Bezdrátové přístupové body utvářejí k centrálnímu řídicímu modulu tunelové spojení v němž je tunelován provoz stanic připojených k příslušnému bezdrátovému přístupovému bodu. Tento přístup nám umožňuje nasadit shodné politiky i služby ve všech bodech a to včetně protokolu IPv6. A to i v lokalitách, které nejsou přímou součástí sítě VŠB a kde IPv6 není lokálně podporováno.

Bezdrátová síť je tvořena, jak již bylo zmíněno výše, dvěma fyzickými WiSM moduly provozovanými v modulárních systémech Cisco Catalyst 6500. Tyto dva moduly jsou tvořeny čtyřmi logickými řídicími moduly. Každý logický modul má maximální kapacitu 150 bezdrátových bodů. Fyzické moduly jsou z důvodu redundance umístěny ve dvou geografických lokalitách.

Problémem v takto rozsáhlé bezdrátové síti jsou oznamování cizích směrovačů (RA, Router Advertisement), které generují Windows systémy s nastaveným sdílením připojení k Internetu. Tato oznámení bohužel nelze v současných verzích programového vybavení řídicích modulů filtrovat. Proto je provoz bez-

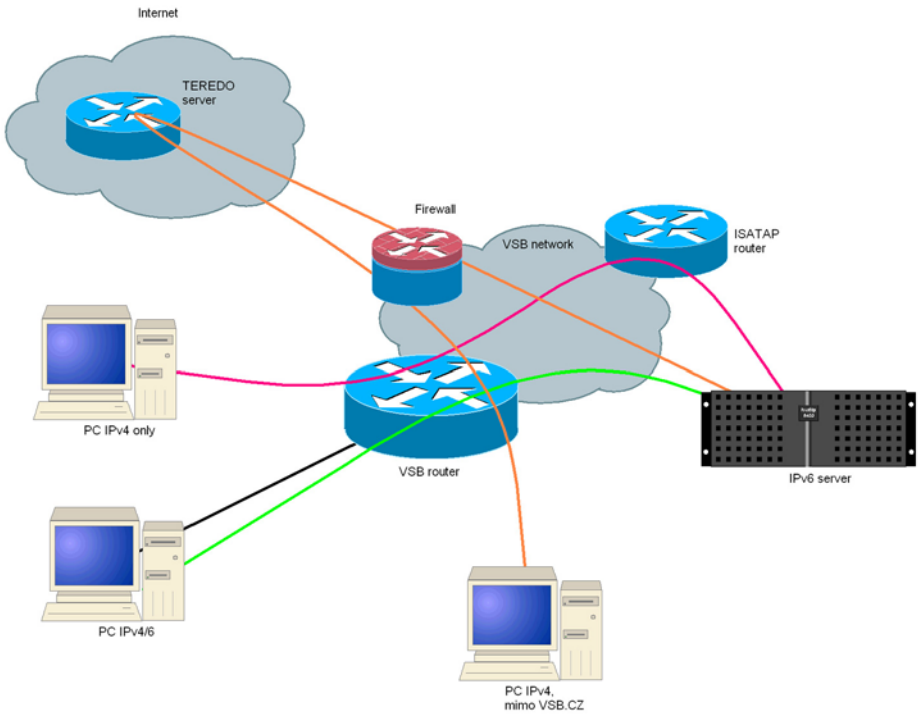
drátových sítí vyveden také na dva systémy, které okamžitě po detekci těchto RA paketů pošlou do sítě jejich zneplatnění. Pro toto zneplatňování je použit open-source produkt RAMON, který je provozován v prostředí OS Linux.

Koncové stanice

Moderní operační systémy koncových stanic obvykle IPv6 podporují. Je ovšem nutné používat bezstavovou konfiguraci klientů SLAAC (Stateless Address Autoconfiguration), protože zdaleka ne všechny operační systémy podporují DHCPv6.

Zde narážíme na problém, že nemůžeme generovaným IPv6 adresám tvořit DNS a rDNS záznamy, protože nevíme, jakou adresu budou stanice využívat.

Stanice v sítích, kde není protokol IPv6 přímo dostupný, mohou získat konektivitu prostřednictvím ISATAP tunelu (viz část o přechodových mechanismech). Provoz ISATA serveru odráží naší snahu eliminovat využívání IPv6 konektivity získané z tunelových serverů mimo síť VŠB.



Obr. 2

Dalším problémem je provozování virtuálních systémů na pracovních stanicích. Tyto instance virtuálních OS jsou obvykle ze síťového pohledu zprovozněny jako systémy za IPv4 NATem. Uživatelé se spokojí s velmi snadným zprovozněním virtuálního systému, což jim ve valné většině případů dostačuje. Ale tento virtuální systém má problém s přístupem k IPv6 konektivitě, protože:

- nemá přístup k fyzické síti VŠB a nezíská nativní IPv6 adresu,
- nevyužívá ISATAP tunely na isatap.vsb.cz, protože neví, že je zapojen doméně vsb.cz (DHCP je poskytováno virtualizačním řešením).

Jedinou cestou, kterou virtualizovaný systém s OS Windows automaticky použije, jsou TEREDO tunely. To však znamená kvalitativně horší IPv6 konektivitu z pohledu propustnosti i latencí a také mohou nastat problémy s přístupem k lokálním službám, protože tento systém má IPv6 adresu z rozsahu mimo VŠB.

Serverové systémy

Serverovým systémům přidělujeme IPv6 adresy staticky. DHCPv6 opět nelze vždy použít, protože některé OS jej nepodporují. V sítích serverových systémů vypínáme podporu autokonfigurací na našich směrovačích. Důvodem je, že i přes staticky nastavenou adresu si některé OS nastaví i IPv6 adresu přes SLAAC. To způsobovalo v praxi problémy, protože byla zaznamenána spojení, ve kterých nebyla jako zdrojová IPv6 adresa použita staticky nastavená IPv6 adresa. Toto spojení může být druhou komunikující stranou zamítnuto jako nepovolené.

Zmíníme-li serverové systémy provozované ve virtuální infrastruktuře, tak zde problémy nejsou. Systémy jsou řízeně zapojeny do infrastrukury počítačové sítě, jejich provoz je přepínán virtuálními přepínači a směrován na směrovačích VŠB.

Centrální serverové IPv6 služby

IPv6 je podporováno u některých centrálních serverových služeb. Prostřednictvím IPv6 jsou dostupné DNS servery (ISC BIND), SMTP server (Postfix), WWW server (Apache, Tomcat). IPv6 podporují od počátku také systémy zapojené do centrální Active Directory domény.

Při nasazování nových služeb je použití IPv6 zvažováno a pokud nejsou důvody pro nenasazení, tak nasazeno také je.

Přístup k IPv6 službám mimo síť VŠB

Mnozí poskytovalé služeb na Internetu už dnes poskytují své služby prostřednictvím protokolu IPv6. Mezi největší tyto poskytovatele patří v našem případě Google. Na počátku roku 2010 bylo domluveno zpřístupnění služeb serverů Google (Google, YouTube, Blogger, ...) i po IPv6. Toto selektivní zpřístupnění je realizováno poskytováním AAAA záznamů domluveným adresním rozsahům. Bližší infromace o způsobu zpřístupnění Google IPv6 zdrojů naleznete na stránkách Google [1].

Lze říci, že z dlouhodobého hlediska nejsou s přístupem k těmto externím zdrojům přes IPv6 žádné problémy. Běžní uživatelé ani neví, že k těmto zdrojům přistupují prostřednictvím IPv6.

Vyskytnou-li se ovšem problémy s IPv6, tak jsou obvykle reportovány stylem „Seznam jede, Google ne“.

Závěr

Lze konstatovat, že všichni výrobci hardware i programového vybavení mají problém s rychlou implementací IPv6 do svých produktů. Někdo je v některých oblastech lepší, někdo horší, ale podpora IPv6, zejména v oblasti bezpečnosti, kde tato úroveň nedosahuje úrovně implementací IPv4. I když podpora IPv6 v nějakém produktu je, tak se v praxi občas ukazuje, že není zcela dobrá, popř. ji prvek realizuje neakcelerovaně v CPU, takže výkon takových řešení je výrazně degradován.

Dalším problematickým bodem je evidence stanic. Bude-li k dispozici dostatečná podpora DHCPv6 v operačních systémech, stejně bude nutné postavit systém evidencí koncových stanic jiným způsobem než v IPv4.

Problémem, který je společný snad všem sítím, je náhrada starších prvků, které IPv6 nepodporují a podporovat nikdy nebudou. Jedná se o finančně nákladné operace a obvykle je potřeba je rozložit do více let.

Literatura

[1] *Access Google services over IPv6.* dostupné na Internetu na adrese http://www.google.com/ipv6

Infrastruktura IPv6 na otevřených systémech

Pavel Šimerda

E-mail: pavel.simerda@netinstall.cz

Opensource operační systémy v roli síťových prvků

Opensource operační systémy jako Linux a BSD se už tradičně drží v přední linii, pokud jde o podporu síťování TCP/IP. Není překvapením, že zastávají přední pozice i v implementaci „nového Internetu“, tedy protokolu IPv6.

Linux si získal pověst nejoblíběnějšího operačního systému pro levné routery. Ty můžete potkat v nejrůznějších podobách, od kancelářských krabiček na stůl, až po profesionální řešení pro poskytovatele bezdrátového připojení k Internetu. Ani BSD takový věhlas v malých síťových prvcích nezískalo.

Takový linuxový operační systém, často označovaný jako firmware, je obvykle sestavován samotným výrobcem či dodavatelem hardware. Nejčastější jsou SOHO krabičky s malým množstvím voleb.

Jádro operačního systému a mnohé další nástroje podléhají některé verzi licence GPL, která uživateli zajišťuje určité výsady, či jazykem GPL určité svobody. V praxi to znamená, že takový firmware časem někdo upraví a vylepší.

Mikrotik RouterOS

RouterOS není zdaleka celý opensource, ale je postavený na Linuxu a některých dalších službách. Přes všechny chyby je možné tento software považovat za ukázku, jak by se to i v opensource světě mělo dělat.

RouterOS je zajímavý tím, že má jednotné příkazové i grafiké konfigurační rozhani. Vše na první pohled působí jako jedna velká databáze nastavení. Desktopové konfigurační rozhraní WinBox (funguje v Microsoft Windos i ve Wine) je velice příjemnou alternativou k nastavení pomocí SSH. Pravda, vzhledem k zacílení Mikrotiků na provozovatele menších sítí, je SSH spíše tou vzácnější metodou.

Ani tak se konfigurace pomocí SSH nijak nepodobá konfiguraci běžné linuxové distribuci. Shell je hierarchický, působí spíše jako příkazový frontend ke

konfigurační databázi, který se strukturou víceméně přesně shoduje s frontendem grafickým.

S IPv6 si velmi dobře poradí RouterOS ve verzi 5, která vyšla teprve nedávno. Pro provozovatele sítí, kteří ještě nedávno kvůli nedostatečné podpoře IPv6 vypínali, to může být impulsem alespoň pro počáteční testy.

Mikrotik RouterOS se používá v produktech téže firmy nebo je možnost ho provozovat i na běžném PC. Použít se dá jak k běžným zákaznickým přípojkám, tak i ke stavbě páteře menší sítě s dynamickým routingem pomocí OSPF a BGP.

OpenWRT

Na malých krabičkách se těší velké oblibě firmware OpenWRT. To je firmware spíše vhodnější pro hračičky případně pro složitější konfiguraci. Bylo tu i několik návrhů, jak dodat OpenWRT nějaké pěkné ovládání, ale zatím se nic z toho. Přesto se nebojím přát tomuto opensource projektu další rozvoj a světlé zítřky.

Pro základní konfiguraci používá OpenWRT jednotný formát UCI. Konfiguraci lze ovládat stejnojmeným programem, a nastavování se pak alespoň částečně přibližuje tomu na Mikrotiku a dobře se nad ním staví jakékoli rozhraní, třeba zrovna to webové.

Správa se u OpenWRT zatím většinou provádí v unixovém shellu a textovém editoru přes protokol SSH.

IPv6 je v OpenWRT samozřejmostí, stejně jako v serverových i desktopových distribucích. Ani IPsec se nenechá zahanbit a pokročilá funcionalita se instaluje v balíčcích.

OpenWRT se obvykle nahrává místo dodávaného firmware, ale prodávají se i desky s předinstalovaným OpenWRT.

IPv6 – bezpečnostní hrozby (aneb IPSec to srovná)

Tomáš Podermański

E-mail: tpoder@cis.vutbr.cz

Abstrakt

Již v prvotních návrzích protokolu IPv6 bylo řešení otázky bezpečnosti stanoveno jako jeden z klíčových cílů. Požadavky kladené na bezpečnost sítí se však během 15 let vývoje protokolu IPv6 výrazně změnily. Tyto změny se ovšem do návrhů nového protokolu příliš nepromítly a IPv6 dnes přináší zcela nové bezpečnostní hrozby. Ty navíc mnohdy mají přesah do stávající IPv4 infrastruktury. Společně s rostoucím množstvím IPv6 zařízení v sítích začínají být otázky bezpečnosti IPv6 stále více aktuální. Cílem přednášky je poukázat na některá bezpečnostní rizika, která s sebou zavádění IPv6 přináší a naznačit možné způsoby jejich řešení.

IPv6 – všelék v oblasti bezpečnosti

IPv6 poskytuje mnohem lepší zabezpečení pro aplikace a sítě. Vysoké úrovně zabezpečení je dosaženo využitím protokolu IPSec, který zajišťuje autentizaci a šifrování přenášených dat s využitím kryptografických metod. Podpora mechanizmu IPSec je povinně požadována jako součást základní implementace IPv6. Další bezpečnostní prvek IPv6 je dán velkým adresovým prostorem, nad kterým není možné provádět sekvenční scanning a vyhledávat zařízení zapojená v IPv6 síti. IPv6 poskytuje lepší úroveň zabezpečení, než jaká byla možná v sítích na bázi IPv4.

Tímto krátkým odstavcem, jak vytaženým z reklamní brožurky, bývá v různých diskusních skupinách a dokumentech řešena mnohdy celá problematika bezpečnosti IPv6. Realita však není tak optimistická, jak by se mohlo na první pohled zdát, a protokol IPv6 obsahuje řadu bezpečnostních problémů. Podívejme se na některé z nich.

Horizontální skenování

Skenování představuje základní prostředek útočníka k odhalení případných bezpečnostních nedostatků. Výstupem skenování bývá množina adres zařízení, která jsou v síti aktivní a tedy vhodná k vedení potenciálních útoků. Skenování tedy představuje první útočníkův krok a je vstupní branou pro vykonávání dalších záškodnických akcí. Ze strany správců je vyvíjeno poměrně velké úsilí k eliminaci těchto aktivit. V případě IPv4 se skenování provádělo poměrně snadno. Útočník si vybral potenciálně zajímavou síť a postupně otestoval jednu adresu za druhou. Vzhledem k omezenému množství adres se nejednalo o časově nijak náročnou operaci.

V případě IPv6 je situace odlišná. Pro koncové sítě je vyhrazený adresový prostor v délce 64 bitů. Taková kombinace nám dává 1.8×10^{19} adres. Prostými propočty zjistíme, že pokud bychom chtěli provést skenování na takové síti hrubou silou (*Brute Force*), trvalo by 28 let, než bychom nalezli první aktivní IPv6 adresu. Skenování by navíc muselo být vedeno s intenzitou 1 milion testů za sekundu, což by vyžadovalo datové pásmo o šířce 400 Mb/s. S klidem tedy můžeme říci, že tato varianta je mimo hru.

Je nereálné, že by se různé záškodnické a hackerské skupiny spokojily s takovýmto závěrem, a skenování sítí zkrátka přestaly provádět. Vzniknou tedy snahy, jak poměrně velký adresový prostor zúžit a zaměřit se na ty adresy, u nichž je větší pravděpodobnost výskytu. Jaká možná zjednodušení se dají použít?

- **Služby poskytující informace o adresách:** DNS, Dynamické DNS, Whois, NetFlow záznamy, logy serverů, NI Query (RFC 4620 [2]) atd.
- **Odhadování na základě předpokládaného výskytu adres:** EUI 64, sekvenčně přidělované adresy, tzv. *well-known MAC adresy.*
- **Odhalování na základě existujících IPv4 adres:** v různých tunelovacích systémech, na serverech atd. IPv6 adresy serverů bývají dnes často konfigurovány tak, že se současná IPv4 adresa serveru zapíše hexadecimálně do spodních 32 bitů IPv6 adresy.

Názorná ukázka provedení takového skenování byla prezentována na konferenci 27th Chaos Communication Congress [4]. Výsledek provedených optimalizací je poměrně zajímavý. Nalezeno bylo cca 2 000 aktivních IPv6 adres v průběhu 20 vteřin.

Výše uvedené komplikace platily pouze v případě snahy identifikovat aktivní IPv6 adresy ze vzdálených sítí. Pokud se ale útočník dostane na úroveň lokální sítě, je v naprosto odlišné situaci. Zde není problém odposlouchávat informace signalizačního protokolu ICMPv6 v multicastové skupině `FF02::1`, a z této komunikace sestavit přehled o všech aktivních IPv6 uzlech v lokální síti. Není

potřeba dokonce žádného sofistikovaného nástroje. Pro začátek můžete vyzkoušet příkaz `ping6 FF02::1`, a poté prohlédnout cache sousedů (*Neigbor Cache*). Bezpečně v ní najdete *Link-Local* adresy všech zařízení v lokální síti.

Vzdálené skenování sítě je v IPv6 prostředí o něco náročnější a vyžaduje kombinování informací z více zdrojů. Lze tedy říci, že oproti IPv4 zde dochází k jisté komplikaci, nicméně vhodným postupem lze dosáhnout velice dobrých výsledků. Skenování na úrovni lokálních sítí je naopak operace velice jednoduchá a proveditelná jakýmkoliv uzlem připojeným v příslušné lokální síti.

Bezpečný IPSec

Rozšíření protokolu v podobě IPSec umožňuje volitelně autentizovat a zabezpečovat provoz pomocí rozšiřujících hlaviček AH (*Authentication Header*) a ESP (*Encapsulation Security Payload*), kde se hlavička AH používá k autentizaci a pro kontrolu síťové adresy a ESP pro šifrování obsahu. Hlavičky mohou být použity současně, pokud požadujeme autentizaci i šifrování, anebo zvlášť. Povinně implementovaná musí byt pouze hlavička ESP [5]. Do budoucna se plánuje vypuštění hlavičky AH a implementace IPSec budou používat pouze hlavičky ESP. Hlavička ESP šifruje obsah datagramu, takže účinně zabraňuje odposlechu, nicméně informace z hlaviček IP datagramů (například zdrojová a cílová IP adresa) mohou být útočníkem bez problému odposlouchávány.

IPSec může obecně pracovat ve dvou režimech. První varianta řeší zabezpečení komunikace mezi koncovými uzly. Další varianta použití IPSec-u je tunelovací režim. Ten je použitelný zejména pro propojování celých sítí. Data vstupující do zařízení realizujících tunelovací režim jsou obalena vnější zabezpečenou vrstvou a předána protější straně tunelu, která zajistí opačný proces.

Samotný proces šifrování a autentizace vyžaduje další podpůrné mechanizmy. Jedná se zejména o správu bezpečnostních asociací, pravidelnou výměnu klíčů, správu certifikačních cest. Případné zájemce lze odkázat na publikaci Pavla Satrapy IPv6 [14], kde je problematika IPSecu popsána velice precizně.

IPSec a aplikace

IPSec, jak už to s novinkami bývá, nepřináší pouze výhody v podobě možnosti bezpečného transportu dat, ale také jisté komplikace. První riziko pramení z faktu, že samotná data jsou šifrovaná. Tímto se veškerá zařízení v podobě firewallu, IDS, IPS, monitorovacích sond stávají prakticky slepá. Obsah IP datagramu je šifrovaný a výše uvedené systémy se nedostanou dál než na úroveň IPv6 hlavičky. Zcela utajeny jim zůstanou informace o protokolu vyšší vrstvy – nedokáží identifikovat, zda v šifrovaném obsahu je přenášena webová TCP relace, UDP datagramy útočící na DNS server, záškodnická ICMPv6 zpráva anebo

další rozšiřující hlavičky. Lze namítnout, že úplně stejnému typu problému jsme vystaveni při jakékoliv šifrované komunikaci (např. https, ssh, imaps). To je do jisté míry pravda, ani tam nedokážeme provádět detailní inspekci obsahu, nicméně máme k dispozici informace o provozované službě a směru komunikace. V rámci firemní bezpečnostní politiky dokážeme například definovat, že služba https (443) na serveru s adresou a.b.c.d (či spíše `a:b::c:d`) je v pořádku, zatímco provoz služby na portu 445 (*Microsoft-DS*) již nikoliv. V případě IPSec tuto rozlišovací schopnost ztrácíme.

Ochrana před IPSec-em

Z výše uvedeného plyne doporučení, které se na první pohled mnohým může jevit poněkud zvláštně. **Provoz protokolu IPSec by měl být blokován na úrovni korporátního firewallu.** Obecně lze problém přirovnat k letištnímu provozu, kdy vstupní kontrola zcela jistě nedovolí nikomu vstoupit do letadla s nedobytným olověným bezpečnostním kufříkem, aniž by znala jeho obsah. Případné výjimky pro provoz IPSec-em by měly vznikat jen pro ta zařízení, o kterých je bezpečně známo, že jsou sama o sobě dostatečně zabezpečena a nemohou tak posloužit jako útočníkova přestupní stanice do vnitřní sítě. IPSec je realizován jako jedna z rozšiřujících hlaviček a tedy pro filtraci je možné použít běžné prostředky [10]. V některých implementacích firewallu je možno specifikovat i další parametry, jako například Index asociace [18], a tím omezit provoz jen na správcem schválené IPSec asociace.

Další spornou výhodou IPSecu je, že pro aplikace pracuje zcela transparentně. To přináší velkou výhodu tvůrci aplikace, který využívá běžná volání systému pro práci se sítí, a zabezpečením se nemusí nijak zabývat. To, co je výhodou, je současně obrovským nedostatkem. Aplikace tímto ztrácí možnost ověřit, zda komunikuje zabezpečeně a s důvěryhodnou protistranou. Dotazovací dialog na ověření certifikátu, který vám nabízí například webový prohlížeč při vstupu do internetového bankovnictví, je najednou skryt před očima aplikace. Nemůžete si byt jisti, zda vám nějaký útočník nepodstrčil prostřednictvím DNS jinou IP adresu, kde není bezpečnostní politika prostřednictvím IPSec definována, a vy tedy komunikujete s útočníkem nezabezpečeně.

IPSec rozhodně nepatří mezi zavrženíhodné technologie a v IP sítích si postupně nachází své místo. Díky tomu, že byl integrován jako volitelná nadstavba do protokolu IPv4, existují již praktické zkušenosti, které mohou být využity ke zlepšení standardů a zdokonalení stávajících implementací. Oproti původní představě rozhodně nelze IPSec považovat za komplexní řešení bezpečnostních problémů budoucího Internetu, ale pouze jako prostředek ke zvýšení úrovně zabezpečení mezi komunikujícími uzly tam, kde je to požadováno.

Autokonfigurace a její zabezpečení

Jedním ze základních požadavků na protokol IPv6 byla podpora automatické konfigurace koncových uzlů sítě. Původní idea autokonfigurace vycházela z představy, kdy se IPv6 zařízení připojí do sítě a vše potřebné se nakonfiguruje automaticky, aniž by byla vyžadována nějaká další interakce na straně uživatele. Ačkoliv již v době původních návrhů IPv6 byl v sítích docela běžně využíván protokol DHCP, rozhodli se tvůrci protokolu IPv6 vydat odlišnou cestou.

Ústředním prvkem automatické konfigurace koncových zařízení se stala technika označovaná jako bezstavová autokonfigurace (zkráceně SLAAC). Směrovač v síti v pravidelných intervalech informuje všechny připojené uzly v síťovém segmentu, v jaké síti se nacházejí a který směrovač mají použít pro pakety, které mají putovat do Internetu (RA – *Router Advertisement*). Prosté oznamování by samozřejmě nebylo dostatečně pružné. Z toho důvodu může nově připojené zařízení vyslat do sítě požadavek (RS – *Router Solicitation*) se žádostí o informaci, ve které síti se nachází a kudy vede cesta ven. Celý mechanizmus autokonfigurace je součástí specifikace *Neighbor Discovery for IP Version 6* (RFC 2461 [7]), a veškerá komunikace probíhá s využitím protokolu ICMPv6.

Pokud si pečlivě prostudujete specifikaci informací předávaných v rámci oznámení směrovače (*Router Advertisement*), zjistíte, že se v něm kromě definice typů, voleb a časů platností nachází pouze definice prefixů sítí, ve které se nacházíme. Pro plnohodnotnou komunikaci v síti bychom ovšem potřebovali vědět některé další „drobnosti“ – jako například IPv6 adresy rekurzivních DNS serverů. Taková informace však v oznámeních směrovače obsažena není. Problém absence rekurzivních DNS serverů v SLAAC je znám už poměrně dlouhou dobu. Snahy o vyřešení problému se prakticky rozešly do tří směrů. Rozšíření informací předávaných v rámci SLAAC o rekurzivní DNS servery v podobě RFC 6106 [14], specifickými anycastovými adresami (draft-ietf-ipv6-dns-discovery-07 [3]) a prostřednictvím DHCPv6 serveru. První varianta v podobě rozšíření autokonfiguračních informací byla přijata nedávno a zatím scházejí implememntace v koncových systémech. Řešení opřené o specifické anycastové adresy ztratilo svůj význam společně se zrušením Site-Local adres. V praxi tedy zůstává poslední možnost, a to přidělení adres prostřednictvím DHCPv6 serveru.

DHCPv6 rozlišuje dva základní režimy. První tzv. **bezstavové (Stateless) DHCPv6** je prakticky pouhou nadstavbou na dříve popsaný mechanismus autokonfigurace (SLAAC). K tomuto účelu slouží v oznámeních směrovače (*Router Advertisement*) speciální příznak O – other, který informuje klienta, že si v příslušné síti má zažádat o další informace související s parametry připojení prostřednictvím bezstavového DHCPv6.

Stavové (Stateful) DHCPv6 se svým chováním poněkud více podobá DHCP, které známe z IPv4. Klient zjistí z oznámení směrovače (*Router Advertisement*), že v síti má získávat adresu pomocí DHCPv6 (nastaven příznak M).

Klient prostřednictvím multicastu požádá DHCPv6 server o přidělení adresy. Ten klientovi přidělí adresu na určitou dobu, přidělení se potvrdí atd. To ovšem klientovi nezabrání, aby si rovněž nakonfiguroval adresy, které získal na základě oznámení směrovače. Ve výsledku má tedy klient nakonfigurovánu jak IPv6 adresu prostřednictvím SLAAC, tak adresu získanou z DHCPv6 serveru.

Na první pohled by se mohlo zdát, že mechanizmus SLAAC můžeme zcela deaktivovat a vše nechat pouze na DHCPv6. Tato myšlenka je bezpochyby správná – až na jeden detail. Prostřednictvím DHCPv6 dokážeme předat všechny potřebné parametry kromě toho nejdůležitějšího, a to **výchozí brány** (*default route, gateway*). Výsledkem jsou tedy dvě techniky autokonfigurace koncových zařízení, kde v současné době jedna nemůže fungovat bez druhé a v sítích je nutno provozovat jak SLAAC, tak DHCPv6.

Bezpečnost autokonfigurace v IPv4 sítích

Na první pohled je patrné, že proces konfigurování IP adresy je přímo lákadlem pro nejrůznější typy útoků. Dobře promyšlený zásah do procesu autokofigurace může útočníkovi umožnit podvržením odpovědi buď přesměrovat celý provoz na PC útočníka, anebo podvrhnout IP adresy rekurzivních DNS serverů. V mnoha případech ani nemusí jít o cílený útok, ale například o nehodu, kdy si uživatel připojí do sítě třeba WiFi router s předkonfigurovaným DHCP serverem.

Z tohoto důvodu postupně vznikly v IPv4 světě některé mechanizmy znemožňující nebo alespoň komplikující tyto typy útoků. Ochrana je v ideálním případě implementována na koncovém portu přepínače, kde se připojuje uživatel nebo síť zákazníka. Různí výrobci používají mírně modifikované označení pro jednotlivé typy ochran, ale obecně se můžeme setkat s následujícími:

DHCP Snooping: V rámci přepínače jsou explicitně definovány důvěryhodné porty, ze kterých mohou přicházet odpovědi z DHCP serveru (tzv. *trusted port*). Předpokládá se, že někde v hloubi za trusted portem je umístěn DHCP server. V případě, že na port, který není definován jako trusted, dorazí odpověď DHCP serveru, je rovnou zahozena. Případný DHCP server spuštěný na klientském systému (ať už je tam záměrně, nebo nedopatřením) tak neohrozí ostatní klienty v síti, protože odpovědi se nedostanou dál než za port, kde je tato ochrana aktivována.

Dynamic ARP protection: Představuje další stupeň ochrany navazující na DHCP Snooping. V rámci komunikace klienta s DHCP serverem si přepínač „odposlechne“ tuto komunikaci a uloží si do databáze dvojici MAC adresa – IP adresa. Tuto databázi pak použije na untrusted portech k inspekci ARP paketů. Tímto se eliminují útoky zaměřené na podvržení záznamů v ARP tabulkách (poisoned ARP cache).

Dynamic IP Lockdown: Dalšího stupně pak dosáhneme tím, že je na *untrusted* portech prováděna inspekce zdrojové MAC a IPv4 adresy na všech paketech vstupujících do portu. Tímto je znemožněno podvrhávaní zdrojové IPv4 anebo MAC adresy (*Address Spoofing*). Další velice často ceněnou vlastností tohoto mechanizmu je, že klient prakticky nemůže komunikovat, pokud si nejdříve nezažádá o IP adresu prostřednictvím DHCP serveru.

Můžeme vidět, že ochranné prostředky v IPv4 světě se postupem času rozvinuly do docela vysoké úrovně zabezpečení. U jednotlivých výrobců se mírně liší chování a použitá terminologie, nicméně v principu jsou možnosti velice podobné.

Bezpečnost autokonfigurace v IPv6 sítích

Zřejmě nikoho nepřekvapí, že proces autokonfigurace je možné v IPv6 zneužít obdobným způsobem jako v IPv4. Je tedy na místě také IPv6 zabezpečit vhodnými ochrannými mechanizmy. Celkovu situaci nám ovšem poněkud komplikuje skutečnost, že v sítích musíme provozovat jak bezstavovou konfiguraci (SLAAC), tak protokol DHCPv6. Podívejme se na ekvivalentní možnosti ochrany a nabízená řešení v IPv6.

Savi

Podobný mechanizmus, jaký jsme si popsali v případě DHCP snoopingu pro IPv4, se snaží řešit draft draft-ietf-savi-dhcp-07 [1]. Omezuje se pouze na DHCPv4 a DHCPv6 a nijak neřeší problematiku podvržení oznámení směrovače (*Router Advertisement*). Některé prvky mechanizmu SAVI jsou implementovány z prvcích H3C.

SEND – Secure Network Discovery

Způsob, který se snaží problematiku řešit zcela jinou cestou. Systém je postavený na podepisování paketů kryptografickými metodami [16]. Kromě směrovače nevyžaduje podporu na úrovni aktivních prvků sítě. Vlastní ověřování validity prostřednictvím certifikátu zprávy se odehrává až na koncovém systému. IPv6 adresa koncového systému je dána výsledkem kryptografické funkce (a vida, máme tu další autokonfigurační metodu). Použití SENDu se přímo vylučuje z použití s EUI 64 adresami a s *Privacy Extensions*. Velkou výhodou SENDu je, že řeší nejen problematiku autokonfigurace, ale i ostatní bezpečnostní problémy *Network Discovery* protokolu (RFC 2461 [7]). Další výhodou je nezávislost na infrastruktuře, může tedy dobře sloužit i například ve WiFi sítích.

Základním nedostatkem SENDu je, že vyžaduje podporu infrastruktury veřejného klíče podle X 509. Pro správné fungování musíte mít nainstalován certifikát autority, která vydává certifikáty pro směrovače. Zatím není zcela zřejmé, zda by si organizace musely pro směrovače nakupovat a obnovovat komerční certifikáty, jejichž autority již budou v klientských systémech předinstalovány, anebo si uživatel před připojením bude muset příslušný certifikát obstarat. V takovém případě je ovšem organizačně snazší dát uživateli papírek s IPv6 adresou, kterou si nastaví ručně. Se SENDem jsou také spojeny ještě další bezpečnostní rizika [10]. SEND je patentovanou technologií firmy Cisco, proto asi nikoho nepřekvapí, že je implementován zejména v zařízeních této firmy.

RA Guard

Další alternativou, která ovšem řeší pouze problematiku falešných oznámení směrovače, je IPv6 Router Advertisement Guard [16], který je zatím ve fázi draftu. V principu se podobá řešení DHCP snoopingu, avšak aplikuje se pro *Router Advertisement* pakety. Snaží se blokovat falešná oznámení směrovače, ideálně na připojovacím portu zařízení, které je neoprávněně generuje. Je to tedy technologie na úrovni přepínače. Kromě prostředků, které mají usnadnit počáteční konfiguraci přepínačů (učící režim), otevírá cestu k integraci se SENDem. V tomto režimu přepínač pracuje jako tzv. *node-in-the-middle*, kde přepínač s aktivovaným RA Guardem použije informaci ze SENDu pro ověření validity paketů, a vůči připojenému koncovému systému se již tváří jako běžné oznámení směrovače (*Router Advertisement*). Jak lze již z názvu tušit, RA Guard nijak neřeší problematiku zabezpečení DHCP nebo DHCPv6. Implementaci již můžeme najít v některých zařízeních Cisco.

Všechny výše uvedené možnosti jsou dnes použitelné spíše sporadicky. Nejsou buď naimplementovány na klientech, nebo schází podpora v zařízeních. Dalším problémem je to, že pokud mají být ochranné prvky skutečně smysluplné, musí být umístěny co nejblíže koncovému systému. To mnohdy může představovat výměnu kompletní síťové infrastruktury, což je asi záležitost, kterou kvůli zavádění IPv6 nebude ochoten jen tak někdo podstoupit. Nezbývá tedy nic jiného, než se porozhlédnout po nějakém dostupnějším řešení, které alespoň zmírní případné dopady snahy ochromit mechanizmus autokonfigurace v IPv6.

Access Listy na přepínači

Uvedené řešení publikoval Petr Lampa v rámci pracovní skupiny CESNET [6]. Předpokládá, že na aktivním prvku máme možnost konfigurovat Access Listy pro IPv6.

```
01: ipv6 access-list block-ra-dhcp
02:     10 deny icmp any any 134 0
03:     20 deny udp any eq 547 fe80::/64 eq 546
04:     30 permit ipv6 any any
05:     exit
06: interface 1-44
07:     ipv6 access-groupblock-ra-dhcp in
08: exit
```

Uvedený access list provede blokaci všech ICMPv6 zpráv typu 134 (zprávy RA, řádek č. 2) a blokaci provozu na cílový port 546 (dhcpv6-client, řádek č. 3). Následně jsou pravidla aplikována na vstupu portů, kde jsou připojeni klienti (řádek č. 7). Tímto je možné eliminovat výskyt falešných směrovačů a DHCPv6 serverů. Uvedený příklad je použitelný na přepínačích HP, nicméně řešení je obecně použitelné na jakékoliv platformě podporující IPv6 ACL.

Pasivní monitorování

Další možností je detekce falešných oznámení směrovačů (*Router Advertisement*). Příliš nás neochrání proti promyšlenému a cílenému útoku, ale lze jím aspoň detekovat chybně nakonfigurované klienty. K tomuto řešení se budeme muset uchýlit v případě, že nelze použít některou z výše uvedených možností. Pro mnohé sítě to bude po dlouhou dobu jedno z mála použitelných řešení [8]. Všechny nástroje pro detekci falešných oznámení směrovače pracují na shodném principu. Připojí se do multicastové skupiny `ff02::1`, kde se uvedené zprávy šíří, a tím mají možnost sledovat všechna oznámení vyskytující se na síti. O nežádoucím stavu může pak informovat správce, vyvolat nějakou automatizovanou akci (Ndpmon [8], Ramond [12]), nebo dokonce zpět do sítě vyslat zprávu rušící platnost falešného oznámení (Rafixd [19]).

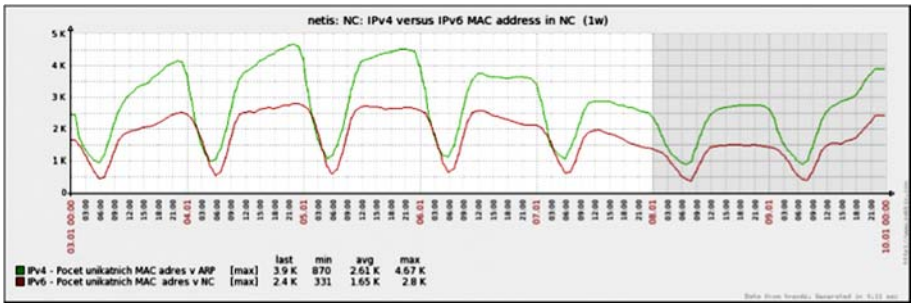
Autokonfigurace a přesah do IPv4 infrastruktury

Jak jsme si již popsali výše, prostředky autokonfigurace a možnosti ochrany jsou mnohdy ještě ve stavu vývoje. Můžeme si tedy říci: *„Fajn, IPv6 se stále vyvíjí, počkáme tedy, až se věci ustálí, a až pak se budeme IPv6 nějak zabývat, a až přijde ta správná doba, tak do sítě nasadíme to, co bude běžně dostupné na většině zařízení. Stejně to IPv6 zatím nikdo nepoužívá.“* Proti tomuto pragmatickému postoji jistě nelze nic namítat až do doby, než se začneme zabývat dopadem IPv6 na stávající infrastrukturu. **Síť bez řízeného IPv6 se stává zranitelnou současně i pro služby provozované po IPv4.** Možný způsob si ukážeme na následujícím scénáři.

Pro jednoduchost si představme síť (např. středně velkého poskytovatele u vás ve městě, nebo firemní síť) provozovanou s přiměřenou mírou zabezpečení (DHCP snooping atd.). Pro jednoduchost předpokládejme, že na síti zatím není nijak konfigurována podpora IPv6.

- Do takové sítě přijde útočník se svým PC a vyšle do sítě falešné oznámení směrovače (*Router Advertisement*). Zde už má první možnost si přesměrovat IPv6 provoz na své vlastní síťové rozhraní. To by zatím ještě tolik nevadilo, protože by na sebe stáhl pouze IPv6 provoz, kterého je jako šafránu. Tedy zatím škody minimální.
- V rámci oznámení směrovače ponechá nastaven příznak M (managed), kterým donutí všechna zařízení podporující DHCPv6 požádat DHCPv6 server o další konfigurační údaje. Ten samozřejmě útočník připravil na svém zařízení. Záměrně v položce rekurzivních DNS uvede IPv6 adresu (či lépe několik adres) svého PC. Díky tomu, že rekurzivní DNS servery jsou používány střídavě bez ohledu na protokol, bude každý n-tý požadavek vyřizován naším podstrčeným DNS serverem. Pravděpodobnost dotazování našeho falešného DNS serveru zvýšíme prostým předáním většího množství adres DNS serveru v DHCPv6 odpovědi.
- Od tohoto okamžiku je pokračování naprosto triviální. Místo A, resp. AAAA záznamu, jako je www.csas.cz, www.seznamka.cz, www.facebook.com, bude jednoduše falešný DNS server nabízet vlastní IP adresu. Pak už je vše záležitostí vhodné konfigurace proxy serveru, vhodného vložení se do komunikace atd. Představivosti se meze nekladou.

Jak vidíte, takový typ útoku je proveditelný naprosto triviálním způsobem. Nepotřebujete k tomu žádné extra znalosti ani dovednosti. Nemusíte naprogramovat jediný řádek kódu. Vše dokáže vytvořit uživatel s lehce nadprůměrnými dovednostmi. Je pouze otázkou času, kdy obdobných typů chyb začnou využívat organizované skupiny (možná se už tak děje).



Obr. 1

Zde bych rád upozornil na graf počtu IPv6 zařízení ve studentské síti Vysokého učení technického v Brně [12]. Horní křivka (zelený průběh) zachycuje počet unikátních MAC adres v ARP tabulce. Vyjdeme-li z předpokladu, že všechna zařízení v síti v současné době mají alespoň jednu IPv4 adresu, pak můžeme tento průběh prohlásit za celkový počet zařízení v síti. Dolní křivka (červený průběh) zachycuje počet unikátních MAC adres v neighbour cache, tj. ve statistice je zachyceno každé zařízení, které má nakonfigurovánu alespoň jednu IPv6 adresu. Z uvedených průběhů vyplývá, že na síti již dnes máme více než polovinu zařízení podporujících IPv6. Všechna tato zařízení jsou potenciálním cílem na obdobný typ útoků, přičemž vůbec nezáleží, zda v síti IPv6 využíváte či nikoliv. Nezáleží dokonce na tom, zda je síť zabezpečená proti útokům po IPv4, díky IPv6 lze toto zabezpečení elegantním způsobem obejít.

Závěr

V současné době příliš nezáleží na tom, jaký je váš vztah k IPv6, zda si o protokolu myslíte, že je to slepá ulička vývoje, anebo skvělý nápad. Jak názorně ukazují statistiky, protokol IPv6 je tady, a to na více než na polovině zařízení v síti. Bylo by jistě velice naivní si myslet, že všechna tato zařízení jsou dokonale zabezpečená proti nejrůznějším formám zneužití. Situace je o to složitější, že řada chyb v implementacích IPv6 jednak dosud nebyla odhalena, anebo jejich řešení stojí absolutně na okraji zájmu výrobců hardware a software. Časté upínání se k technologii IPSec-u zanechává ve stínu mnohem závažnější bezpečnostní problémy protokolu IPv6. Většina bezpečnostních rizik známých z protokolu IPv4 existuje současně i v IPv6, navíc doplněna o zcela nové možnosti.

Paradoxně se dostáváme do stavu, kdy protokol IPv6 se nepoužívá k tomu, k čemu byl primárně určen (tedy ke komunikaci), ale na mnoha místech vytváří jakási zadní vrátka umožňující snadný vstup do síťové infrastruktury. Co je horší, IPv6 neohrožuje pouze samotnou IPv6 infrastrukturu (to by asi ani tolik nevadilo, když ji téměř nikdo nepoužívá), ale výrazným způsobem může ovlivnit chod již existujících zařízení a služeb využívajících IPv4. Prostá ignorace IPv6 může poměrně zásadním způsobem ohrozit jak samotný chod sítě, tak i chod služeb na ní provozovaných. **Neřízená penetrace IPv6 v koncových sítích vede k řadě problémů a je tedy mnohem lepší začít se věnovat zavádění řízenému.** Sice se tím nevyřeší všechny problémy, nicméně alespoň jejich velká část se zmírňuje a umožňuje mít nad sítí větší kontrolu.

Literatura

[1] BI, J., WU, J., YAO, G., BAKER, F.: *SAVI Solution for DHCP*: Draft-ietf-savi-dhcp-07 [online]. 2010. [cit. 19. 4. 2011]. Dostupné na internetu: https://datatracker.ietf.org/doc/draft-ietf-savi-dhcp/

[2] CRAWFORD, M., HABERMAN, B. *IPv6 Node Information Queries*: RFC 4620 [online]. 2006. [cit. 19. 4. 2011]. Dostupné na internetu: http://tools.ietf.org/html/rfc4620

[3] DURAND, A., HAGINO, J., THALER, D.: *Well known site local unicast addresses to communicate with recursive DNS servers*: Draft-ietf-ipv6-dns-discovery-07 [online]. 2010. [cit. 19. 4. 2011]. Dostupné na internetu: http://tools.ietf.org/html/draft-ietf-ipv6-dns-discovery-07

[4] HEUSE, M.: *Recent advances in IPv6 insecurities* (přednáška na 27th Chaos Communication Congress) [online]. 2010. [cit. 19. 4. 2011]. Dostupné na internetu: http://www.youtube.com/watch?v=c7hq2q4jQYw

[5] KENT, S., SEO, K.: *Security Architecture for the Internet Protocol*: RFC 4301 [online]. 2005. [cit. 19. 4. 2011]. Dostupné na internetu: http://tools.ietf.org/html/rfc4301

[6] LAMPA, P.: *Detekce routeru a problémy* [online]. 2010. [cit. 19. 4. 2011]. Dostupné na internetu: http://www.cesnet.cz/ipv6/wg/p/1006-detekce-routeru.pdf

[7] NARTEN, T., NORDMARK, E., SIMPSON, W.: *Neighbor Discovery for IP Version 6 (IPv6)*: RFC 2461 [online]. 1998. [cit. 19. 4. 2011]. Dostupné na internetu: http://tools.ietf.org/html/rfc2461

[8] GRÉGR, M., ROGUE ROUTER DETECTION [online]. 2011. [cit. 19. 4. 2011]. Dostupné na internetu: https://ow.feide.no/_media/geantcampus:2011-gn3na3t4-ipv6-gregr.pdf

[9] *NDPMon*: IPv6 Neighbor Discovery Protocol Monitor [online]. 2009. [cit. 19. 4. 2011]. Dostupné na internetu: http://ndpmon.sourceforge.net/

[10] PODERMAŃSKI, T., GRÉGR, M.: IPv6 Mýty a skutečnost, díl V. – Zjednodušené hlavičky. *Lupa.cz* [online]. 2011. [cit. 19. 4. 2011]. Dostupné na internetu: http://www.lupa.cz/clanky/ipv6-myty-a-skutecnost-dil-v-zjednodusene-hlavicky/

[11] PODERMAŃSKI, T., GRÉGR, M.: IPv6 Mýty a skutečnost, díl VI. – Bezpečnostní mechanizmy. *Lupa.cz* [online]. 2011. [cit. 19. 4. 2011]. Dostupné

na internetu: http://www.lupa.cz/clanky/
ipv6-myty-a-skutecnost-dil-vi-bezpecnostni-mechanizmy/

[12] PODERMAŃSKI, T.: IPv6 Mýty a skutečnost, díl I. – Jak jsme na tom. *Lupa.cz* [online]. 2011. [cit. 19. 4. 2011]. Dostupné na internetu: http://www.lupa.cz/clanky/ipv6-myty-a-skutecnost-dil-i-jak-jsme-na-tom/

[13] *RAMOND* [online]. 2009. [cit. 19. 4. 2011]. Dostupné na internetu: http://ramond.sourceforge.net/

[14] SATRAPA, P.: DNS zařazeno do bezstavové konfigurace. Konečně. *Lupa.cz* [online]. 2010. [cit. 19. 4. 2011]. Dostupné na internetu: http://www.lupa.cz/clanky/dns-zarazeno-do-bezstavove-konfigurace-konecne/

[15] SATRAPA, P.: *IPv6*. 2. vyd. Praha : CZ.NIC. 2008. 357 s. ISBN 978-80-904248-0-7. Dostupné na internetu: http://knihy.nic.cz/files/nic/edice/pavel_satrapa_ipv6_2008.pdf

[16] SATRAPA, P.: RA-Guard: pomáhat a chránit sousedy v IPv6. *Lupa.cz* [online]. 2010. [cit. 19. 4. 2011]. Dostupné na internetu: http://www.lupa.cz/clanky/ra-guard-pomahat-a-chranit-sousedy-v-ipv6/

[17] SATRAPA, P.: SEND chrání objevování sousedů pro IPv6. *Lupa.cz* [online]. 2010. [cit. 19. 4. 2011]. Dostupné na internetu: http://www.lupa.cz/clanky/send-chrana-objevovana-souseda-pro-ipv6/

[18] *Security Parameter Index* [online]. 2010. [cit. 19. 4. 2011]. Dostupné na internetu: http://en.wikipedia.org/wiki/Security_Parameter_Index

[19] *Strattg/Rafixd* [online]. 2011. [cit. 19. 4. 2011]. Dostupné na internetu: https://github.com/strattg/rafixd

Příliš mnoho Internetu, aneb „Za síť čistější“

Petr Břehovský

E-mail: petrbreh@gmail.com

Staré časy

Pamatujete ještě na časy, kdy uživatel sám rozhodoval o tom, jaká data z Internetu získá a jaká data do Internetu poskytne?

Pamatujete na služby jako je Gopher, Veronica, FTP, Archie, nebo dokonce Ftpmail a Webmail?

Pamatujete na doby, kdy byla celá vaše organizace připojena do Internetu 64kb pevnou linkou?

Některé ze služeb, které jsme v minulosti využívali přetrvávají dodnes. Uveďme některé příklady.

Gopher vyhledávač Veronica 2:
gopher://gopher.floodgap.com/1/v2
Příklad gopher serveru:
gopher://gopher.meulie.net/11/gopher
Vyhledávač souborů poskytovaných FTP servery Archie:
http://www.physik.uni-oldenburg.de/Docs/net-serv/archie-gate.html
http://archie.icm.edu.pl/archie-adv_eng.html
Anonymní FTP server:
ftp://ftp.funet.fi

FTP mail umožňuje získat soubory z FTP serverů pomocí elektronické pošty. Zašleme-li na adresu FTPmail serveru následující dopis:
reply email@adresa
connect ftpserver
binary
compress
uuencode
get pub/soubor.zip
quit

Dostaneme jako odpověď zprávu s přiloženým souborem *soubor.zip* ze serveru *ftpserver*.

email@adresa je naše adresa

ftpserver je server na kterém se nachází soubor
pub/soubor.zip je soubor, který si chceme nechat zaslat
Více informací lze získat na:
http://w2.eff.org/Net_culture/Net_info/EFF_Net_Guide/EEGTTI_HTML/eeg_221.html

WEB mail umožňuje získat pomocí elektronické pošty dokonce WEB stránky:
http://www.www4mail.org/

Máte nápad jak ještě v dnešní době použít FTP mail a WEB mail?

Ve volných chvílích můžeme pomocí telnetu shlédnout dokonce i film v ASCII (Star Wars):
telnet://towel.blinkenlights.nl

Časy kdy výše uvedené služby kralovaly, jsou však nenávratně pryč.

Nové časy

Nyní již nerozhodujete o tom, co z Internetu získáte vy sami, ale vyhledávač, WEB server a někdy i nezávisle na vůli uživatele sám klient.

Vyhledávač

Google vytváří pro každého uživatele jiný obraz Internetu. Nezískáváme tedy obraz takový jaký ve skutečnosti je, ale výsledky vyhledávání jsou uzpůsobeny geografické poloze uživatele, resp. většinou jeho mateřskému jazyku.

Pokud si chceme sami vybrat, jaký tento obraz bude, musíme použít některý z volně dostupných proxy serverů, nebo si vhodně vybrat odchozí uzel některé z anonymizačních sítí (např. TOR) tak, abychom vypadali jako uživatel pocházející z té které oblasti.

Mnohokrát pomůže také Privoxy (o Privoxy bude řeč dále) filtr *hide-accept-language* a *hide-user-agent*.

```
# Chceme se vydávat za Kanaďana.
{+hide-accept-language{en-ca} \
+hide-user-agent{Mozilla/5.0 (X11; U; OpenBSD i386; en-CA;
rv:1.8.0.4) Gecko/20060628 Firefox/1.5.0.4} \
}
```

Některé další nevýhody vyhledávačů

Identifikace uživatele

Každý vyhledávač má zcela jistě profil našeho počítače (IP adresa, prohlížeč a jeho nastavení, operační systém), podle kterého nás může poměrně přesně

odlišit od ostatních uživatelů Internetu. Ve spojení např. s e-mail kontem může tento profil mapovat na konkrétní osobu. Pravděpodobně je možné mapovat počítač a uživatele pomocí jeho profilu, i když „cestuje“ (mění IP adresy).

Dá se tomu nějak bránit? Jistě. Můžeme použít některou z anonymizačních sítí jako je například TOR v kombinaci s maskováním WEB klienta pomocí, ano hádáte správně Privoxy. K tomu se výborně hodí

Privoxy Actions *add-header, change-x-forwarded-for, client-header-filter a client-header-tagger*. V případě Googlu je také možné použít specializovaný proxy server Scroogle (http://scroogle.org).

Analýza uživatelských dat

Cílená reklama, která se objevuje v pravé části stránky uživatelského rozhraní Gmailu neoddiskutovatelně svědčí o tom, že Google naše emaily čte. Úvahu o tom jak je to s ukládanými soubory nechám na každém z vás.

Zde pomůže pouze šifrování silnou šifrou.

Existují vůbec nějaké alternativy?

V roce 1956 napsal Isaac Asimov skvělou krátkou povídku *The Last Question* (česky *Poslední otázka*), která se zabývá centralizací výpočetní techniky dovedenou až k absolutní hranici. Může být dnešní Google Multivacem z Asimovovi povídky? Zatím asi ne, protože existuje mnoho velmi zajímavých vyhledávačů, které si s Googlem nezadají, a ve speciálních případech se mnohdy dokonce hodí mnohem lépe. Uveďme několik skupin takových vyhledávačů s konkrétními odkazy.

Vyhledávače založené na umělé inteligenci:
ChaCha: http://www.chacha.com/

Vyhledávače poskytující výsledky hledání v tzv. Clusterech (skupinách):
Carrot: http://search.carrot2.org/
Quintura: http://www.quintura.com/
Yippy: http://search.yippy.com/

Vyhledávače založené na doporučeních:
MusicMap: http://www.music-map.com/

Metavyhledávače (oslovují za uživatele více vyhledávačů najednou):
Metacrawler: http://www.metacrawler.com/
Dogpile: http://www.dogpile.com/
Zuula: http://www.zuula.com/

Ne snad přímo vyhledávač, ale v každém případě věc, která působí téměř zázračně:
Wolfram: http://www.wolframalpha.com/

Webserver

Webservery jsou kapitola sama o sobě. Každému z nás se jistě mnohokrát stalo, že mu byla konkrétní informace o délce několika řádek ASCII textu poskytnuta spolu s nicneříkajícím ba obtěžujícím balastem zvícím několika desítek, možná i stovek kilobytů.

Srovnejme například titulní stránku iDnes.cz s obrázky a bez obrázků (měřen počet paketů programem netstat).

S obrázky:

```
Kernel Interface table
Iface    MTU     RX-OK    TX-OK
eth0     1500    797      821
```

a bez obrázků:

```
Iface   MTU     RX-OK    TX-OK
eth0    1500    200      231
```

Tato data je možné zobrazit také v mnohem spektakulárnější podobě pomocí programu Etherape (http://etherape.sourceforge.net/).

Co s tím?

Samotné prohlížeče pomocí některých svých nastavení (Automaticky zobrazovat obrázky, Blokovat vyskakovací okna atd.) jistě odvádějí dobrou práci, ovšem ideální je lokální proxy s většinou mnohem bohatší funkcionalitou. Skvělým příkladem může být například Privoxy (http://www.provoxy.org).

Privoxy umožňuje skoro libovolně automatizovaně zpracovávat HTTP transakce, je ho tedy možné použít například k:

- Blokování reklam
- Blokování bannerů
- Blokování zákeřných serverů
- Automatické editaci HTML kódu stránek

Jádrem programu Privoxy jsou *Actions*, což jsou vlastně direktivy, které Privoxy říkají jak má nakládat s HTTP transakcemi. Každá *Action* má unikátní jméno a funkci.

Actions se uvádějí v konfiguraci Privoxy a jsou následovány jedním, nebo více URL na které jsou aplikovány. URL mohou být (a většinou jsou) uvedeny ve formě paternů.

Základní *Actions* jsou *Blocks* a *Filters*. Pomocí *Blocks* můžeme nevyžádaný obsah blokovat, pomocí *Filters* můžeme obsah měnit. *Blocks* blokují server, nebo

stránku, *Filters* umožňují filtrovat, nebo modifikovat přímo obsah stránky. Můžeme tak například nahradit sprosté slovo jeho oficiálním anatomickým významem.

Pokud budeme chtít definovat vlastní *Actions*, je vhodné je uvádět v konfiguračním souboru *user.action*, který je zachováván při aktualizaci privoxy. Stejně tak *Filters* je vhodné definovat v *user.filter*.

Uveďme příklad filtru, který zabrání zobrazení obrázků ve stránce, pokud se mají stahovat z jiného serveru, než je server hostující danou stránku. Vycházejme ze zjednodušeného předpokladu, že obrázky poskytované domovským serverem jsou do HTML kódu vloženy jako *img src="/obrazky/image.gif"* a obrázky stahované z jiného serveru jako
img src="http://server.example.com/obrazky/image.gif".

Zkontrolujeme, zda jsou v hlavním konfiguračním souboru Privoxy (*config*) uvedeny soubory pro uživatelské konfigurace:
actionsfile user.action
filterfile user.filter

Vytvoříme filtr, který bude v HTML kódu přijaté stránky nahrazovat odkazy (resp. část *img src="http://*) na obrázky na externích WEB serverech textem NECHCICIZIOBRAZKY:

```
FILTER: breh Ostranuje ref. Na externi obrazky
s/<img\s*src="http:\/\/*/NECHCICIZIOBRAZKY/g
```

kde *breh* je jméno filtru, následuje komentář, a druhý řádek definuje substituci (tak jak ji známe např. z vi editoru). $\backslash s*$ nahrazuje libovolný počet mezer a tabulátorů, obrácená lomítka $\backslash/\backslash/$ ruší speciální význam lomítek a *g* znamená, že substituci provádíme globálně, tj. v rámci celé řádky.

Samozřejmě to není ideální řešení, protože mezi *img* a *src* nemusí být jenom mezery a tabulátory, a mezi *"* a *http* naopak mezery být mohou, ale jako jednoduchou ilustraci problematiky budeme uvedený příklad považovat za dostačující...

Filtr uvedeme v souboru *user.filter* a do *user.action* na něj vytvoříme odkaz spolu výrazem (*.cz*), který definuje, na co má být celá akce aplikována:

```
{ +filter{breh} }
.cz
```

V žádném případě však není nutné, aby se běžný uživatel Privoxy věnoval psaní filtrů a akcí. Už v implicitní konfiguraci jsou ochranná funkce programu dostačující, protože obsahuje poměrně velké množství pečlivě připravených filtrů a akcí (viz konfigurační soubory *default.action* a *default.filter*). Pokud však přesto vyžadujeme přísnější filtrování, můžeme si pomocí konfigurační stránky Privoxy dostupné lokálně pomocí odkazu http://config.privoxy.org/ resp. http://config.privoxy.org/edit-actions-list?f=0 vybírat mezi třemi (seřazeno podle

„přísnosti“) předdefinovanými úrovněmi filtrování: *Cautious* (implicitní), *Medium* a *Advanced*.

Editaci nastavení musíme ale nejdříve povolit v konfiguračním souboru (*config*) nastavením *enable-edit-actions* na *1*.

Klient

Klient je to, co na první pohled patří nám, to co nás v Internetu reprezentuje jako entitu. Nebo by to tak alespoň mělo být. Realita je však někdy taková, že naši softwaroví klienti reprezentují někoho jiného, nebo spíše zájmy někoho jiného. To je v dnešní době jedna z největších nočních můr uživatele Internetu (nebo alespoň moje). Každodenně na nás dotírá různý škodlivý kód (malware), který číhá za odkazy na WEB stránkách, v souborech na USB discích, v přílohách e-mailů, nebo se dokonce sám šíří ve formě červů. Silná trojka: Pravidelné aktualizace, Antivir a Firewall nás poměrně spolehlivě chrání. Ovšem co se děje v útrobách počítače a na jeho síťovém rozhraní stejně pořádně nevíme.

Síťová analýza

V poznání toho co se opravdu děje nám však mohou pomoci některé základní a jednoduché nástroje. V první řadě se jedná o analyzátor síťového provozu. Každý jistě zná Tcpdump (http://www.tcpdump.org/), nebo Wireshark (http://www.wireshark.org/), ale jistě není v silách běžného uživatele sledovat rychle rolující výpis paketů a hledat v něm něco podezřelého.

Mnohem efektivnější je pakety nějakým způsobem vizualizovat. Například pomocí již zmíněného Etherape (http://etherape.sourceforge.net/), nebo VisualSnifferu (http://www.biovisualtech.com/vindex.htm). Tyto programy je možné mít neustále spuštěny a poočku sledovat, jestli se neděje něco neobvyklého.

Pokud se něco neobvyklého děje, je nutné provést podrobnější analýzu typu: „Kdo safra vlastně všechna ta data posílá?“

Pokud známe lokální port a protokol podezřelé komunikace, které zjistíme analyzátorem, nebo programem *netstat*, můžeme pomocí programů *fuser* a *ps* zjistit jaký program se na ní podílí:

```
# fuser 8118/tcp
8118/tcp:             2536
# ps -p 2536
  PID TTY          TIME CMD
  2536 ?        00:00:00 privoxy
```

Vyšší úroveň paranoie vyžaduje hlubší analýzu, která může spočívat ve spuštění síťového analyzátoru už při startu samotného systému těsně po aktivaci

síťového rozhraní, v ukládání všech nachytaných paketů do souboru a v jejich následném zkoumání.

`/etc/init.d/networking:`

... kráceno...

```
*)
   echo "Usage: /etc/init.d/networking {start|stop}"
   exit 1
    ;;
esac
nohup /usr/sbin/tcpdump -n -nn -s0 -w /home/breh/start.pcap &
exit 0
```

DNS sinkhole

Další zajímavou možností jak blokovat, nebo zkoumat připojení zákeřných programů do Internetu je DNS sinkhole. DNS sinkhole představuje takovou konfiguraci DNS serveru, kdy je server masterem pro domény, které hostí servery pro zákeřné programy (CC servery, servery sbírající informace atd.) a pro všechny počítače v těchto doménách vrací definovanou IP adresu. Touto adresou může být 127.0.0.1, nebo IP adresa serveru, který slouží k detekci napadených počítačů v kontrolované síti.

DNS sinkhole může běžet jak lokálně na osobním počítači, tak na DNS serveru sítě.

Uveďme příklad instalace a konfigurace tohoto systému na lokálním počítači.

Instalace DNS serveru BIND:

```
# apt-get install bind9
```

Konfigurace serveru:
Do *named.conf.local* (nebo *named.conf*, pokud nevyužíváme *named.conf.local*) uvedeme domény, které chceme blokovat:

```
zone "lumpove.com" { type master; file  "/etc/bind/db.blocked"; };
```

Všimněte si, že se stáváme masterem pro domény, které blokujeme (seznam domén obsahujících zákeřné servery můžeme pro nekomerční použiti získat například na http://www.malwaredomains.com/)

Soubor *db.blocked* je společný pro všechny blokované domény a vypadá následovně:

```
$TTL    600
@   IN      SOA     localhost root (
                    2011041600
                    3H      ;refresh
```

```
                        15M     ;retry
                        1W      ;expire
                        1D )    ; min ttl
    1D      IN      NS      @
;
                        A       127.0.0.1
*   5       IN      A       127.0.0.1
```

hvězdička znamená, že blokujeme všechny A záznamy dané domény (www1.lumpove.cz, www2.lumpove.cz atd.)

Pokud se bude jednat o serverovou konfiguraci, upravíme odpovídajícím způsobem *SOA* větu a pokud máme server, který bude vyhodnocovat pokusy o připojení nakažených klientů k zákeřným serverům, nahradíme 127.0.0.1 jeho IP adresou.

Restartujeme DNS server:

```
# /etc/init.d/bind9 restart
```

A na klientu nezapomeneme upravit konfigurační soubor resolveru (*/etc/resolv.conf*) tak, aby ukazoval na server s DNS sinkhole. V případě že, provozujeme sinkhole lokálně:

```
nameserver 127.0.0.1
```

Je třeba si uvědomit, že tento systém neblokuje škodlivý kód, který se svými servery komunikuje pomocí IP adres! Zato je poměrně spolehlivý v případě že zákeřný kód používá v současné době hojně využívané Fast-flux DNS techniky.

A ještě poslední poznámka. Nezapomeňte kontrolovat *resolv.conf*, případně konfigurační soubory BINDu některým ze systémů pro kontrolu integrity souborů. Nelze totiž vyloučit, že se je některý ze sofistikovanějších škodlivých programů pokusí upravit k obrazu svému.

Poznámka na závěr

Poslední dobou dochází k přesunu osobních dat z počítačů na mobilní telefony. V telefonech máme nejenom kontakty, ale i poznámky, dokumenty, přístupová hesla a používáme je v systémech elektronického bankovnictví, k platbám za jízdenky, a není daleko doba, kdy pomocí nich budeme platit všude. Telefony používáme k přístupu k Internetu, čtení pošty a k navigaci. Prostřednictvím mobilu může stále více a více organizací monitorovat náš pohyb. V neposlední řadě telefony používáme i k telefonování, které se při volání na podvodná čísla může řádně prodražit.

Naše životně důležité údaje se rychle přesouvají z osobních počítačů do mobilů a spolu s nimi tam musíme přesunout i nástroje na jejich ochranu.

Drupal CMS

Vojtěch Kusý

E-mail: vojtech.kusy@fsv.cvut.cz

Abstrakt

Drupal patří spolu se systémy Wordpress a Joomla! mezi tzv. velkou trojku opensource systémů pro správu obsahu (CMS) napsaných v PHP. Drupal pohání cca 1,7 % webu, což ho sice v žebříčku absolutního počtu nasazení těchto systémů řadí až na třetí místo, ale v poslední době se mu z těchto tří systémů daří nejúspěšněji pronikat i do té nejvyšší „enterprise“ sféry – nasazují ho nadnárodní korporace i velké neziskové a vládní organizace. Ve svém příspěvku stručně popíši historii tohoto systému a zajímavé koncepty, kterými se odlišuje od ostatních systémů a které také většinou zásadním způsobem ovlivnily další směřování jeho vývoje. Kromě modulární architektury, systému uzlů (nodes), systému háků (hooks) a základních knihoven jádra Drupalu popíši i několik zásadních doplňkových modulů jako jsou Content Construction Kit (CCK), Views nebo Features. Představím novinky z aktuální verze Drupalu, zejména koncept tzv. entit a pokusím se nastínit i další pravděpodobný vývoj tohoto systému.

Stručná historie aneb jak vznikl Drupal

Původním autorem Drupalu je Belgičan Dries Buytaert [*drís bytárt*], který tento systém začal vytvářet na koleji univerzity v Ghentu mezi lety 1998 a 1999 jako systém pro komunikaci a sdílení zpráv s ostatními studenty, se kterými pracoval na společném univerzitním projektu. Po dokončení studia uvolnil tento systém v roce 2000 pod licencí GNU GPL a umístil ho volně ke stažení na internet. A protože měl tento software úspěch a zájem o něj rostl, rozhodl se ho Buytaert dále rozšiřovat o funkce CMS systémů jako je moderování obsahu, granulární správa oprávnění apod. Zaregistroval pro projekt vlastní doménu – a tak vznikl Drupal. Drupal verze 1.0.0 vyšel 15. 1. 2001. [1]

Základní charakteristiky

Drupal (http://drupal.org/about) je vysoce modulární open source content management system (CMS) nebo také content management framework (CMF) napsaný v jazyce PHP. Základními charakteristikami tohoto systému jsou otevřenost, modularita a snadná rozšiřitelnost – téměř každý aspekt Drupalu lze ovlivnit pomocí modulů nebo pomocí šablon v tématech vzhledu. Díky otevřené modulární architektuře je dosaženo velké interoperatibility, což je důležité např. při integraci systému Drupal s jinými systémy ať už externími (Facebook, Google API) nebo interními (mapový server, ERP systém). Pro mnohé subjekty je rozhodujícím momentem také kvalitní dokumentace a důsledně dodržované *coding standards* (standardy pro psaní zdrojových kódů) a bezpečnost, na kterou je kladen velký důraz.

Další významnou charakteristikou je to, že Drupal je, a téměř od samého počátku byl, komunitní projekt[1]. Např. jen verze Drupalu 7.0 obsahovala záplaty od 954 různých lidí (nezahrnuje kód z předchozích verzí). [2]

Jádro systému

Jádro systému je tvořeno základními knihovnami Drupalu a povinnými moduly. Kromě povinných modulů obsahuje základní instalace ještě volitelné moduly, témata vzhledu a instalační profil.

Do základních knihoven patří základní funkce jako např. *drupal_http_request()*, která dokáže poslat HTTP požadavek a přijmout výsledek, funkce *drupal_goto()*, která způsobí přesměrování na URL zadanou v parametru nebo různé funkce pro práci s řetězci v UTF-8 kódování, které Drupal interně používá. Kromě jednotlivých funkcí obsahují základní knihovny i základní programovací rozhraní (Application Programming Interface, API) pro práci se systémem – např. Schema API, pomocí kterého lze definovat datovou strukturu úložiště dat nezávisle na konkrétním databázovém systému, Forms API, pomocí kterého se v Drupalu tvoří formuláře, Menu API, které slouží k přiřazování cest v URL k jednotlivým akcím a stránkám v systému (tzv. *routování*) a další.

Nad těmito knihovnami jsou postaveny povinné moduly, v Drupalu 6 mezi ně patří následující: Block, Filter, Node, System, User. Ve verzi Drupalu 7 mezi ně patří: Filter, Node, System, User, Field, Field SQL Storage a Text.

[1] Dries Buytaert má stále v projektu vůdčí pozici a je i současným prezidentem mezinárodní Drupal Asociace, ale sám již téměř neprogramuje, jeho současná pozice je spíše vizionářská a je hlavní správce projektu – schvaluje záplaty a zahrnuje je do jádra. Snaží se vést s celou komunitou jakýsi dialog. Dává sice najevo, jak by si Drupal představoval on, ale snaží se i naslouchat a iniciativu nechává na ní. Viz jeho přednáška na konferenci TED, Managing the Chaos of a Thousand Voices, http://www.technologyreview.com/business/37079/page1/}

- Modul Block slouží k zobrazení obsahu ve formě bloků. Od Drupalu 7 nepatří mezi povinné moduly, protože se objevily i alternativní způsoby – systém bloků jádra lze tedy nahradit jiným systémem (např. doplňkovým modulem Panels nebo Boxes).
- Modul Filter slouží k filtrování výstupu obsahu, který byl vložen uživateli (vstup se v Drupalu ukládá nezměněný, k filtrování dochází až na výstupu, tedy až při zobrazování obsahu).
- Modul Node (Uzel) definuje API pro data obsahu a definuje také výchozí typy obsahu Story (Článek) a Page (Stránka). Umožňuje také administrátorovi vytvářet další typy obsahu.
- Modul System slouží k základní systémové konfiguraci Drupalu.
- Modul User obsahuje registrační a přihlašovací systém pro uživatele.
- Field, Field SQL Storage a Text jsou součástí Fields API a umožňují k uzlům, uživatelům a dalším prvkům obsahu připojovat vlastní datová pole (více viz Modul CCK).

Kromě těchto základních modulů obsahuje základní instalace systému Drupal ještě asi 20 volitelných modulů např. Comment, Blog, Database logging, Forum, Help, OpenID, Poll, Search, Taxonomy a další. Každý z těchto modulů obohacuje Drupal o další funkcionalitu[2].

Systém háků a architektura systému

Jádro systému komunikuje s moduly pomocí tzv. háků (hooks). Háky jsou speciální funkce, které mají předem definovaný vstup a výstup a při jejich vytváření je nutné dodržet konvenci ve tvaru <*název modulu*>_<*název háku*>. Tyto háky se volají v určitých momentech běhu systému, čímž je umožněna reakce modulů na daný stav systému (jedná se o návrhový vzor pasivní pozorovatel neboli *passive observer*). Např. pokud modul implementuje hook_boot() jako mujmodul_boot(), je tato funkce volána při každém požadavku na systém na Drupal. Háky obvykle vykonají nějaký kód a skončí (např. hook_cron) nebo vrací nějakou hodnotu (např. hook_menu – vrací položky pro účely routování

[2]Důvody, proč jsou volitelné moduly v jádře, jsou dva – jednak jsou dlouhodobě vyhodnoceny jako hojně používané a zajišťují základní funkce CMS systému nebo můžou sloužit jako vzorová implementace nějakého API, či případu užití (*use case*). Aktuálně se např. vede diskuse o odstranění modulu OpenID ze základní instalace, protože ze statistik vyplývá, že je zapnutý jen na 2,5 % Drupal instalacích, nicméně zřejmě v jádru zůstane, dokud se neobjeví jiný lepší otevřený standard, na kterém by bylo možné dobře otestovat přihlašování a registraci do Drupalu přes autoritu třetí strany – viz diskuse Remove OpenID from core, http://drupal.org/node/556380.

a zároveň také odkazy do menu). Kromě toho existují ještě tzv. *alter hooks*, ty dovolují modulům upravit hodnoty, které již předtím vrátily jiné moduly. Např. hook_menu_alter – umožňuje upravit routy a položky menu. Asi nejpoužívanější alter hook je hook_form_alter() kterým je možné upravit libovolný formulář poskytovaný jádrem drupalu nebo přidanými moduly.

Příklad implementace hook_init() v souboru hello.module:

```
<?php
/**
 * Implementace hook_init().
 * Pozdravíme přihlášeného uživatele při každém načtení stránky.
 */
function hello_init() {
  if (user_is_logged_in()) {
    global $user;
    drupal_set_message(t('Hello %name!', array('%name' => $user->name)));
  }
}
?>
```

Kouzlo tohoto návrhového vzoru spočívá v naprosté jednoduchosti implementace. Programátor vůbec nemusí znát objektové programování, aby byl schopen se do systému zapojit se svým kódem. Až do verze Drupalu 6 (tedy až na jednu okrajovou výjimku, třídu XML parseru), se v celém systému nevyskytovala žádná objektová třída, celý běh systému je obsluhován čistě procedurálním kódem – místo objektů se používá datový typ asociativního pole s definovanou strukturou dle příslušné části API, anebo případně základní objekt třídy *stdClass* (což je ale v jazyce PHP téměř to samé jako asociativní pole).

V Drupalu 7 již byly některé subsystémy přepsány do objektů (např. databázová vrstva nebo tzv. entity, které nahradily dřívější ploché objekty *node*, *user*, *comment* apod. plnohodnotnými objekty – entitami) a tento hybridní styl programování – používání objektového a procedurálního programování podle typu úlohy – bude charakteristický i pro další verze Drupalu. Procedurální kód je ideální pro svou jednoduchost a také lepší výkon. Objekty zas vynikají při řešení složitějších úloh, kde se mohou uplatnit některé typické objektové návrhové vzory jako např. návrhový vzor *strategie*, nebo pokud někde vyžadujeme striktní rozhraní.

Díky *hooks* a obzvláště díky *alter hooks*, lze v Drupalu upravit téměř cokoliv a to bez toho, aby bylo nutné upravovat originální soubory. V Drupalu platí zásada *don't hack core*, tedy nezasahovat do jádra, pokud to není nezbytně nutné. Díky tomu je kód Drupalu snadno udržovatelný, nemusíme se obávat každé bezpečnostní aktualizace, která by nás při nedodržení této zásady nutila buď aktualizovaný kód ručně přenést, nebo aktualizaci raději vůbec nedělat, abychom si nerozbili funkčnost webu.

Co se týká architektury, tak Drupal na rozdíl od většiny webových frameworků neimplementuje návrhový vzor MVC (Model-View-Controller) ale PAC (Presentation-Abstrastraction-Control). Rozdíl je v tom, že v modulech nenajdeme žádný „model“, protějšek modelu – *abstraction* je pouhou abstrakcí nad daty a veškerá logika je soustředěna v *Control*, který odpovídá řadiči (controlleru) z návrhového vzoru MVC. Drupal neobsahuje ani něco na způsob ORM (objektově relační mapování), struktura databáze nenese žádnou informaci o objektech v systému, slouží jen jako „hloupé“ úložiště dat a metadat. Další významný rozdíl je v tom, že architektura navržená podle návrhového vzoru PAC je již z podstaty vícevrstvá. Lze si to představit tak, že vyrenderovanou stránku v Drupalu lze reprezentovat jako stromový graf. Tam, kde je kořen, je prvek celé stránky a jeho potomky jsou jednotlivé oddíly stránky, které se dále rozvětvují až na atomické složky v listech tohoto stromu, což jsou jednotlivé části obsahu, které se na stránce objevují, obalené a zpracované různými šablonami a dodané prostřednictvím různých modulů. [3, 4]

Modul CCK

Modul CCK (Content Creation Kit, http://drupal.org/project/cck) rozšiřoval Drupal až do verze 7 o možnost vytvářet vlastní typy obsahu a přiřazovat k nim přes grafické rozhraní vlastní datová pole. CCK pole obsahují 3 vrstvy:

1. *field*, starají se o ukládání i nahrávání dat pole z úložiště,
2. *widget* neboli formulářový prvek pro editaci pole a jeho validace a
3. *formatter* neboli formátovač, který řeší jak data reprezentovat uživatelům.

Tento modul způsobil v Drupalu revoluci, nebo spíše evoluci, která posunula Drupal výrazně směrem k frameworku. Použití CCK je snadný způsob, jak vytvářet typy obsahu – datové struktury – na míru a to i bez znalosti programování.

CCK je jedním ze základních modulů, které jsou nainstalované prakticky na každém webu postaveném na Drupalu, kromě těch snad úplně nejjednodušších, kde si uživatelé vystačí pouze s obsahovými typy Článek a Stránka.

Od Drupalu 7 je velká část jeho funcionality přenesana do jádra, jako tzv. Fields API a další doplňující moduly: Text, Number, File, List, Field UI, Field SQL storage, pro základní použití není třeba instalovat modul CCK.

Tuto funkcionalitu nabízí i konkurenční CMS jako Joomla a v poslední hlavní verzi i Wordpress, ale v Drupalu byla tato funkcionalita dostupná již v roce 2004 (ještě pod názvem Flexinode, od roku 2006 pak již jako CCK) a hlavně Drupal v tomto případě významně těží ze své interoperability – CCK je podporováno jak jádrem systému tak prakticky i všemi z přibližně 6 tisíců modulů, které jsou dostupné na http://drupal.org/project/modules.

Od uzlů k entitám

Uzel je základní jednotkou obsahu v Drupalu. Je to základní konceptuální objekt, který abstrahuje od konkrétních typů obsahu jako článek nebo produkt. Až do verze 6 obsahovalo jádro několik základních objektů: jednalo se zejména o *Node, User, Taxonomy Term* a *Comment*. Uzel resp. konceptuální objekt Node jako abstraktní reprezentant obsahu byl vždy v popředí zájmu a hák *hook_nodeapi*, jehož implementací mohly doplňkové moduly zasahovat do procesu nahrávání, zobrazování, ukládání i mazání uzlů je dosud jedním z nejpoužívanějších.

Díky modulu CCK, který popisuji v předchozím oddílu, bylo snadné v uzlu vytvořit libovolnou datovou strukturu a to velmi rychle, bez programování. Při vývoji webů je často potřeba kromě hlavních datových struktur obsahu, které chceme veřejně prezentovat, vytvořit také pomocné struktury[3]. Bohužel použití CCK a Node API na takový úkol často znamenalo, že se jen zaměnila jedna množina problémů za jinou. Drupal totiž vyžaduje, aby každý uzel měl název, svoji URL a volitelně tělo textu, dále standardně nerozlišuje práva přístupu k zobrazení obsahu a také standardně obsah uzlů indexuje pro vyhledávání na webu. Pokud nám některá z daných vlastností nevyhovovala (často všechny), bylo nutné hledat cesty jak standardní chování obejít a násilně potlačit mnoho funkcionality, kterou Drupal pro uzly nabízí.

Již dlouhou dobu bylo jasné, že je nutné abstrahovat ještě o úroveň výš. Proto v Drupalu 7 vznikl koncept entit. Téměř vše je v Drupalu 7 entita a pokud má vlastnost „fieldable“, lze k ní připojovat i vlastní datová pole, stejně jak je popsáno níže u modulu CCK. Uzel už je jen jednou z implementací Entity, stejně tak jako User, Taxonomy Term nebo Comment. Entity je samozřejmě možné implementovat i v doplňkových modulech a existují již i moduly, které práci s entitami ještě více usnadňují např. Entity API (http://drupal.org/project/entity), která dovede zcela bezpracně řešit rutinní CRUD[4] operace a urychlit tak vývoj vlastních doplňkových modulů na bázi entit.

Modul Views

Modul Views http://drupal.org/project/views) je po CCK druhým nejvýznamnějším modulem. Umožňuje vytvářet pohledy na data uložená v databázi CMS Drupal a od verze 3 již dokonce i mimo databázi – od této verze dovolují Views pracovat s téměř libovolnými externími daty, zejména s daty různých webo-

[3] Jako příklad mě napadá typ „adresa pobočky“ prodejního řetězce. Adresy nechceme nikdy zobrazit samostatně, chceme je vypsat v přehledu poboček a pak u detailu produktů, pokud jsou dostupné na nějaké pobočce.

[4] CRUD = Create, Read, Update, Delete – rutinní operace ve všech CMS systémech: vytvořit, načíst, aktualizovat, smazat

vých služeb: výsledky vyhledávání Wolfram Alpha, fotky ze služby Flickr, videa z Youtube nebo výsledky získané pomocí Yahoo Query Language (YQL).

Jak to funguje? Views poskytují administrační rozhraní, kde si při tvorbě pohledu v prvním kroku zvolíme, jaké objekty chceme zobrazit (např. uzly, komentáře, uživatele, taxonomie, externí data, ...), v dalším kroku nám Views nabídnou rozsáhlý, ale přehledný formulář, kde lze díky Ajaxu poměrně pohodlně nastavit mnoho parametrů od stylu zobrazení (neformátovaná data, tabulka, seznam, ...) nebo oprávnění přístupu k tomuto pohledu na data, přes relace na další objekty, vybrání konkrétních datových polí a jejich formátování, až po řazení a seskupování dat a také filtry. Zmíněné filtry lze navíc přepnout do stavu „exposed“ neboli „vystavené“, což umožní uživatelům filtr ovládat pomocí formuláře a tím měnit zobrazená data.

Views také umí filtrovat data v pohledu pomocí tzv argumentů, což umožňuje filtrovat data např. pomocí parametrů v URL, na které je pohled zobrazen.

Každý pohled může mít libovolný počet „displejů“, což jsou různé varianty toho samého pohledu, které určují, jak bude pohled použit. Typickými příklady displejů jsou Stránka nebo Blok. Displej Stránka tedy zobrazí pohled jako hlavní obsah stránky na určité URL (kterou musíme zadat), displej Displej typu blok Blok nám zase umožní zobrazit data na libovolném místě na našem webu v podobě bloku. Existují i další typy displejů, některé ve formě doplňkových modulů a pluginů do Views, které umožňují data „zobrazit“ i v jiných formátech než HTML, např. jako CVS, XML, JSON, RSS zdroj apod.

Modul Features

Předchozí dva moduly, CCK a Views, se spolu s jádrem Drupalu staly základními nosníky a téměř neoddělitelnou součástí. Dalo by se říci, že se jedná o nepřiznanou část jádra Drupalu (což tedy v případě CCK a Drupalu 7 již neplatí).

První verze CCK a Views byly vytvořeny pro Drupal 4.7 a způsobily velkou změnu ve směřování jádra Drupalu a významně také ovlivnily způsob, jakým jsou weby postavené na Drupalu vytvářeny. Do té doby se v Drupalu vyskytovaly moduly, které měly charakter komponent, jak je známe např. z CMS Joomla!. Byly to velké funkční celky – např. modul Events: přidal typ obsahu *event* (událost), stránky s kalendářem a vestavěné listování událostmi, bloky s kalendářem událostí pro aktuální měsíc apod. Takovéto moduly, až na čestné výjimky jako je třeba sada e-commerce modulů Übercart, zcela vymizely. Repozitář přídavných modulů na Drupal.org se naopak začal plnit různými miniaturními moduly, které poskytují často pouze API, nebo nějakým způsobem rozšiřují Views nebo CCK (resp. Fields). Možnosti řešení, které poskytuje kombinace CCK a Views jsou tak široké, že jimi lze postihnout drtivou většinu toho, co lze po CMS chtít, a navíc umožňují, aby si tvůrci webů přizpůsobili datové struktury obsahu i jejich zobra-

zení přesně na míru, jak potřebují. Tvořit nějaké komplexní moduly jako ucelené komponenty pro řešení specializovaných úkolů začalo být nežádoucí, protože nebylo možné poskytnout dostatečnou flexibilitu jak výsledné řešené ovlivnit.

Posun od specializovaných modulů k „vše lze vyřešit pomocí CCK a Views“ však přinesl také jednu poměrně významné úskalí: veškerá struktura webu je držena v databázi a zdrojový kód modulů obsahuje pouze API.

Tento přístup přinesl spoustu problémů, všem kdo to s webovým vývojem mysleli vážně. Co když je potřeba nějakou část takto vytvořené funkcionality použít jinde? Co když na tvorbě webu potřebuje pracovat více lidí nebo je nutné mít více vývojových serverů? Jak přenášet změny z vývojového prostředí do produkčního?

Částečně tyto problémy řeší přímo moduly CCK i Views, jelikož obsahují rozhraní pro export konfigurace do textové podoby a umístění do vlastního modulu, z kterého je možné konfiguraci pak opět načíst.

Nicméně skutečnou odpověď na tyto problémy přinesl až modul Features (http://drupal.org/project/features). Modul *Features* obsahuje administrační rozhraní, pomocí kterého lze exportovat veškerou potřebnou konfiguraci, která je za normálních okolností uložena v databázi (zejm. typy obsahu s jejich nakonfigurovanými CCK poli a příslušné pohledy modulu Views) a uložit ji ve formě modulu, který obaluje tyto exporty. Takto vytvořenou *feature* lze poté aktualizovat, verzovat pomocí nástrojů na správu zdrojových kódů[5], distribuovat na další weby[6] a pokud ji umístíme na server, Drupal si umí pomocí modulu Update Status ohlídat i její aktualizaci, tak jako to umožňuje u standardních modulů a témat. Existuje i několik serverů, kde lze hotové *features* stáhnout, tato funkcionalita ale zatím nebyla zabudována do webu drupal.org, jelikož modul *Feature* je doplňkový modul, nikoliv modul jádra.

Momentálně existuje několik konkurenčních projektů, které se snaží Features nahradit a posunout možnosti tohoto přístupu ještě dále, nebo tento problém uchopit z trochu jiného úhlu pohledu. Jedná se např. o moduly Snapit (http://drupal.org/project/snapit) a nejnověji Apps (http://drupal.org/project/apps).

Instalační profily a distribuce

Myšlenka přenášet funkční celky konfigurace není ale nová a už od verze Drupalu 5 je možné vytvářet tzv. instalační profily. Drupal 5 a 6 obsahoval v základu výchozí instalační profil pod názvem *Default*. Drupal 7 obsahuje dokonce dva:

[5] neboli SCM, source code management

[6] Jak správně vytvářet dobře přenositelné a znovu použitelné *features* popisuje standard Kit (http://drupal.org/project/kit)

Standard a *Minimal*. Kromě těchto standardních profilů je možné stáhnout i další profily ze stránky http://drupal.org/project/installation+profiles[7].

Instalační profily jsou podobně jako Features uloženou konfigurací s možností některé parametry upravit během instalace. Rozdíl je v tom, že Feature se zaměřuje na určitou dílčí funkcionalitu (např. fotogalerie) a Feature můžeme také doinstalovat kdykoliv během životního cyklu webu, zatímco instalační profil se využije jen při čisté instalaci nového webu.

Instalační profily a Features se vzájemně doplňují. Pokud bychom chtěli vyvíjet web pomocí *code driven development*[8], použijeme pravděpodobně oboje.

Kromě instalačních profilů můžeme nalézt i tzv. distribuce. Distribuce se od instalačního profilu odlišuje tím, že je to již hotový produkt – balík obsahující Drupal jádro (někdy i mírně upravené), instalační profil a všechny potřebné závislosti: doplňkové moduly, témata a případně i lokalizační soubory.

Následuje výčet populárních distribucí, které stojí za pozornost:

- OpenAtrium – http://openatrium.com/ – groupware distribuce určená pro intranety (původně vyvinuto jako intranet pro mezinárodní pracovní skupiny Světové banky)
- OpenPublish – http://openpublishapp.com/ – distribuce určená pro publikační portály
- OpenPublic – http://openpublicapp.com/ – distribuce určená pro státní organizace
- Pressflow – http://pressflow.org/ – distribuce/fork Drupalu uzpůsobená pro vyšší výkon
- Tattler (app) – http://tattlerapp.com/ – agregační nástroj pro monitorování zadaných témat v libovolných veřejných zdrojích a sociálních sítích
- Acquia Drupal – http://acquia.com/products-services/acquia-drupal – distribuce poskytuje jádro a vybrané moduly a témata, které tato společnost primárně podporuje ve svých placených support programech
- Drupal Commons – http://acquia.com/products-services/drupal-commons-social-business-software – distribuce určená pro rychlé vybudování komunitního/sociálního webu

[7] Např. pro Drupal 5 existoval český instalační profil, který nainstaloval Drupal kompletně v češtině.

[8] Více informací lze nalézt v prezentaci Code driven development: using Features effectively in Drupal 6 and 7, http://www.slideshare.net/nuvoleweb/code-driven-development-using-features-effectively-in-drupal-6-and-7

Užitečné nástroje pro vývoj a správu

Mezi důležité nástroje, které velmi usnadňují práci při vývoji webů na Drupalu patří balík modulů s názvem Devel (http://drupal.org/project/devel), který obsahuje moduly na generování testovacího obsahu, pomůcky na sledování výkonu a funkce pro ladění.

Velmi užitečným nástrojem je Drush neboli Drupal Shell (http://drupal.org/project/drush), což ale není doplňkový modul, ale jedná se o samostatný nástroj, který umožňuje s Drupalem pracovat z příkazové řádky a urychlit tak řadu úkonů[9].

Zajímavým nástrojem je také Aegir Hosting System (http://community.aegirproject.org/), který je určen pro profesionální hosting Drupalu. Pomocí tohoto nástroje lze přes webové rozhraní zakládat a klonovat či aktualizovat instalace Drupalu, přenášet je a synchronizovat mezi servery a další užitečné funkce jak pro vývoj Drupalu tak pro poskytování software – Drupal webů – jako služby[10].

Drupal 8

Během konference DrupalCon Chicago 2011, proběhla klíčová přednáška Driese Buytaerta, kde představil svou vizi pro Drupal 8. Jednou z klíčových věcí kterou by chtěl změnit je, aby Drupal přestal být orientován primárně na HTML, ale aby byl nativně schopný poskytovat výstup i v jiných formátech stejně tak jako v HTML. V souladu s touto vizí by se chtěl hodně zaměřit na mobilní web. Takové rozhodnutí podepřel studií, která předpodkládá, že počet mobilních zařízení s internetem v blízké budoucnosti převýší počet klasických osobních počítačů a laptopů. Dále by chtěl dosáhnout lepší interoperability a integrace se systémy třetích stran, jelikož jeho dlouhodobou vizí je vybudovat z Drupalu dostatečně flexibilní platformu, aby mohla zastat co nejvíc úkolů v podnikové infrastruktuře a umožnit tak Drupal masivně nasazovat i v korporátní sféře jako jakýsi jednotící prvek místo specializovaných řešení pro každý úkol. [5, 6]

Zajímavé postřehy uvedl i Károly Négyesi[11], na konferenci Drupal Developer Days Bruxelles 2011. Nejzásadnější myšlenkou jeho přednášky asi bylo, že by téměř všechny hlavní komponenty systému měly být *pluggable* – nahraditelné, což by vyřešilo mnoho problémů a mimo jiné by to také znamenalo další posun k objektově orientovanému programování, protože systémové *pluginy* by

[9] Drush podporuje i mnoho doplňkových modulů, které ho rozšiřují o další funkce (implementují hook_drush_command), sada dostupných příkazů pro Drush je tedy vždy individuální, podle toho, v kterém webu se zrovna v konzoli nacházíme.

[10] SaaS, Software as a Service

[11] Károly Négyesi jeden z hlavních vývojářů, který se podílí na vývoji jádra již několik let, je např. architektem současného menu systému a routování.

musely být navrženy oproti striktnímu rozhraní. S růstem obliby NoSQL a také díky klesajícím cenám hardware se také uvažuje o změně konceptu CRUD na CRA(P) dle hesla *„RAM is the new disk and disk is the new tape“* [12]. Tedy, že místo operací *Create/Read/Update/Delete*[13] bude Drupal standardně operovat s obsahem, či s entitami dle konceptu *Create/Read/Archive (Purge)*[14]. Ve zkratce to znamená, že veškerý obsah se bude archivovat, vše bude mít revize a staré, archivované záznamy se budou po čase mazat, pokud tak zvolíme. Z hlediska škálovatelnosti a z hlediska přenosu obsahu mezi servery se také počítá také se zavedením univerzálního unikátního identifikátoru (Universally Unique Identifier, UUID) pro všechny entity v Drupalu 8[15]. [7]

Závěr

Ve svém příspěvku jsem se snažil vysvětlit podstatu fungování Drupalu a zajímavé koncepty, které ovlivnily jeho vývoj. Klíčové jsou zejména jeho snadno rozšiřitelná a modulární architektura, která byť založena převážně na strukturovaném programování umožňuje i pomocí jednoduchého kódu vytvářet poměrně komplexní aplikace, pevně nastavené a vyžadované standardy pro psaní kódu jak pro jádro tak i pro doplňkové moduly a důraz na bezpečnost.

Drupal nelze považovat za hotový produkt, je to spíše polotovar, platforma pro postavení vlastního CMS. Za plnohodnotné produkty lze považovat až distribuce, jejichž některé příklady jsem uvedl. Podobně je tomu s moduly, CCK a Views nejsou produkty či aplikace, které lze rovnou použít, jsou to nástroje, pomocí kterých se musí potřebná funkcionalita vytvořit – a pokud ji chceme jako produkt distribuovat dále, může nám v tom velmi pomoci modul Features, který dokáže efektivně pracovat s importy i exporty takto připravených aplikací.

Na konci příspěvku jsem se snažil nastínit jaké jsou současné vize směřování Drupalu, která je aktuálně upřena k webovým službám, mobilnímu webu a větší podpoře škálovatelnosti.

V samotném příspěvku mi na to již nezbyl prostor, ale důležitou součástí je také komunita okolo tohoto systému, která staví na kultuře otevřenosti, sdílení znalostí a společné práci. Důkazem síly této komunity jsou DrupalCony, velké mezinárodní konference s několikatisícovou účastí, které se pořádají dvakrát do roka (vždy jednou v Evropě a jednou v USA) a desítky menších akcí, které pořádají lokální týmy po celém světě každý měsíc.

[12] RAM (operační paměť) je nový disk a disk je nová páska
[13] Vytvořit/Načíst/Aktualizovat/Smazat
[14] Vytvořit/Číst/Archivovat (Pročistit)
[15] UUID je dnes již dostupné jako doplňkový modul (http://drupal.org/project/uuid).

Literatura

[1] *Drupal history as seen by Dries*, [On-line], [rev. 2011–01–20], [cit. 2011–04–13]. URL: http://drupal.org/node/297669

[2] *Contributors for Drupal 7 – Final Numbers*, [On-line], [rev. 2011–01–04], [cit. 2011–04–14]. URL: http://growingventuresolutions.com/blog/contributors-drupal-7-final-numbers

[3] *Drupal programming from an object-oriented perspective*, [On-line], [rev. 2009–12–25], [cit. 2011–04–14]. URL: http://drupal.org/node/547518

[4] *MVC vs. PAC*, [On-line], [rev. 2006–12–31], [cit. 2011–04–14]. URL: http://www.garfieldtech.com/blog/mvc-vs-pac

[5] Buytaert, D.: *Keynote-Dries Buytaert*, DrupalCon Chicago : 2011, 2011–03–08, [On-line], URL: http://chicago2011.drupal.org/keynote-dries-buytaert

[6] *Acquia product strategy and vision*, [On-line], [rev. 2011–01–24], [cit. 2011–04–14]. URL: http://buytaert.net/acquia-product-strategy-and-vision

[7] Négyesi, K., *Drupal 8, First Things First*, Drupal Developer Days Bruxelles 2011, 2011–02–05, [On-line], URL: http://bxl2011.drupaldays.org/node/318

Java CMS

Josef Krupicka

E-mail: aragorn@civ.zcu.cz

Abstrakt

Redakčních systémů napsaných v PHP, Ruby a dalších skriptovacích jazycích jsou stovky. V oblasti Javy je výběr výrazně chudší a slušných open source řešení je pouze několik. Příspěvek se pokusí identifikovat, proč tomu tak je a seznámí s těmi nejzajímavějšími.

Když se řekne redakční systém, tak většině přijde na mysl „svatá“ trojice Drupal, Joomla a WordPress. Dalších systémů napsaných v PHP jsou desítky, kdežto těch, které jsou napsané v Javě a mají aktivní vývoj je okolo deseti. CMS používajících Python, .NET a Ruby jsou v na tom obdobně jako ty napsané v Javě a jejich počty a nasazení se ani zdaleka nepřibližuje PHP systémům. Neexistuje přesná statistika, která by řekla, kolik webů používá daný redakční systém. Na následujícím obrázku je velký vidět rozdíl mezi tím jak jsou vyhledávané nejznámější PHP a Java CMS.

Obr. 1: Porovnání četnosti hledaných CMS

Dále budou popsány tři hlavní důvody, proč jsou PHP CMS o tolik rozšířenější než ty napsané v Javě.

Hosting

Zde je nejspíš největší důvod proč jsou Java CMS tak málo rozšířené. Sehnat použitelný Java hosting, který by byl zdarma, je nemožné. Placené hostingy existují, ale obsahují celou řadu omezení, přičemž to největší z nich je paměť, kterou lze přiřadit webovému serveru (např. Apache Tomcat). Hostingy nabízejí standardně 64 MB paměti, což je na provoz jakékoliv webové aplikace napsané v Javě velmi málo. Další paměť lze dokoupit, ale za nemalé částky.

PHP v tomto má nespornou výhodu, protože sehnat kvalitní hosting s podporou Apache, PHP a MySQL není velký problém. Navíc většina firem poskytujících hosting dnes již nabízí možnost snadné instalace nějakého známého CMS. Proto jsou PHP redakční první a vlastně jedinou použitelnou volbou pro uživatele, kteří chtějí rychle a levně rozjet jednoduchou webovou prezentaci. Jedničkou je v tomto případě WordPress. Java v této oblasti nemá bohužel čím konkurovat a v blízké době se to nezmění a nejspíš tomu nepomůže ani Google App Engine.

Rozšiřitelnost a modifikovatelnost

Při nasazování CMS většinou nestačí základní funkce vybraného systému a je pak potřeba vytvořit nebo integrovat dodatečné moduly (ankety, diskuze, eshop, dotazníky, ...). Pro známé PHP systémy uvedené výše existují stovky různých modulů, které lze snadno nainstalovat, nakonfigurovat a použít v rámci vytvářeného webu. Většinou k tomu mají i pěkné rozhraní, přes které se tyto kroky snadno provedou. Navíc není nutné po instalaci nového modulu nutné restartovat celý systém. U Javy je situace opět o dost horší a těch důvodů je několik.

Vzhledem k malé velikosti komunity a většinou nepříliš kvalitní dokumentace je počáteční čas, potřebný k proniknutí do „vnitřností“ systému celkem velký. Typicky má systém třívrstvou architekturu a nejprve je potřeba proniknout do použitých technologií. Datová vrstva většinou používá některý z ORM (objektově relační mapovaní) frameworků. Prezentační vrstva je napsáná v Java Server Pages (JSP) nebo některém oblíbených šablonovacích knihoven (Freemarker, Velocity, ...). Celé to spojuje Spring framework nebo Enterprise Java Beans. Dále je potřeba dobře znát API, které je možné v modulech používat a zjistit, jak se modul instaluje a konfiguruje.

Pokud už někdo vytvoří nějaký kvalitní modul pro jeden systém, tak ho bývá velmi složité plně integrovat do jiného systému. Situaci v této oblasti mohl změnit portletový standard JSR-168 a jeho následovník JSR-268. V těchto standardech je definován portlet jako aplikace, poskytující specifickou část obsahu, která je vložena do portálové stránky. Za celou dobu existence těchto standardů však nevznikla aktivní komunita vyvíjecí portlety, které by šlo snadno využívat v rámci různých portálů. Systémy (Liferay, eXo Plaform, HippoCMS), které

jsou postaveny na těchto standardech, obsahují celou řadu zajímavých aplikací. Bohužel tyto portlety nejsou nasaditelné na jiných portálových řešeních, protože používají proprietární API daného systému.

Velikost komunity

Jediným použitelným ukazatelem o rozšíření CMS je velikost komunity, která se okolo daného systému vytvořila. Komunity okolo nejrozšířenějších PHP systémů jsou velké a velmi aktivní.Zájemce o daný systém tak má k dispozici:

- spoustu informací o možnostech systému
- kvalitní šablony, které může snadno upravit a nasadit
- dostatek rozšiřujících modulů

Systémy napsané v Javě mají výrazně menší komunity. Během těch několika let nevyprofiloval nějaký leader v této oblasti, okolo kterého by se vytvořila pevné jádro vývojářů, které by táhlo vývoj kupředu. Např. známý, kvalitní a lety prověřený CMS Plone napsaný v Pythonu má aktivní komunitu, která tento systém posouvá dál. U Java systémů je většinou vývoj veden firmou, která profituje z nasazování daného systému, poskytování placené podpory a školení. Většina Javovských systémů má dvojitou licenci, kdy open source verze poskytuje pouze základní funkce. Pokud chcete při nasazení využít např. integraci s LDAPem, nebo možnost klastrovat a rozložit zátěž je k dispozici placená enterprise verze, která obsahuje více funkcí a kvalitní podporu.

Zajímavé CMS napsané v Javě

V roce 2006 byla vytvořena analýza systémů napsaných v Javě (Krupička, 2006) pro účely webu Katedry informatizace a výpočetní techniky ZČU , ve které bylo probráno 20 tehdy dostupných CMS. Většina z nich byla pro reálné nasazení nepoužitelná a do finálního výběru se dostala pouze čtveřice kandidátů. Ve finále bylo vybráno OpenCMS, o kterém bude dále zmínka. Od té doby se situace příliš nezměnila a jejich počet se ještě zmenšil (u řady systému zcela skončil vývoj). Níže je trochu více probrána trojice CMS, které jsou v současnosti asi nejvíce rozšířené a obsahují zajímavé funkce.

OpenCMS

Jeden z nejstarších redakčních systémů napsaných v Javě, jehož první verze vyšla už v roce 1999. OpenCMS byl nejdříve vyvíjen komunitou vývojářů, která se však

po čase rozpadla. Zdrojové kódy převzala firma *Alcacon Software*, jejíž vývojáři dále pracují na jádře systému. Komunita je složena především z firem, které tento systém nasazují a poskytují služby a rozšiřující moduly. Těch je celá řada a lze je celkem snadno integrovat do použitých šablon. Systém od začátku staví na XML. Obsah OpenCMS je ukládán do virtuálního souborového systému (VFS), který je uložen v databázi. Zajímavou vlastností je možnost vytvářet symbolické odkazy, které mají stejnou funkci jako v souborových systémech. Obsah VFS je přístupný přes WebDAV. K dispozici je také dobré API, které umožňuje s VFS (a nejen s ním) snadno pracovat a vytvářet tak snadno rozšiřující moduly. OpenCMS jako jediné má i skriptovací rozhraní, kterým lze provádět dávkové úpravy obsahu.

Velkou výhodou systému je propracované administrátorské rozhraní, přes které lze snadno spravovat celý web. Editoři mohou dělat úpravy obsahu rovnou při procházení stránek v prohlížeči a v nové verzi OpenCMS číslo 8, by se mělo vytváření webu ještě zjednodušit. Šablony se v OpenCMS vytvářejí přes JSP a je potřeba mít dobré znalosti Javy a vnitřního API k vytvoření kvalitní a použitelné šablony. Počáteční investice do naučení mechanismů systému se však vyplatí a je možno pak OpenCMS využít jako kvalitní aplikační platformu a vytvářet v něm rychle pěkné aplikace a snadno je integrovat do výsledného webu.

OpenCMS je využíván k provozování webových prezentací velkých bank, telekomunikačních firem atd. Tento systém byl úspěšně použit k vytvoření univerzitního webu Západočeské univerzity a běží na něm i řada menších webových prezentací. Na jedné instalaci lze provozovat desítky webů a usnadňuje se tak jejich správa. Díky konfigurovatelným šablonám a skriptovacímu rozhraní pak není problém snadno vytvářet kostry nový webů a zpřístupnit je editorům.

Magnolia

První verze tohoto systému vyšla v roce 2003. Systém se skládá z aplikace pro tvůrce obsahu a aplikace, která tento obsah zobrazuje. Aplikace mohou běžet na různých a tím snadno škálovat při větší zátěži. Jako jeden z mála redakčních systému používá Apache Jackrabbit, což je referenční implementace standardu JSR-170 (Java Content Repository). Velkou výhodou je také pěkně zpracované administrační rozhraní, které poskytuje všechny funkce systému pro tvůrce šablon, editory obsahu a administrátory. Administrátoři zde mají přístup k definici vlastních typů obsahu, dávkových úkolů a workflow. Zajímavým prvkem je možnost instalovat rozšiřující moduly přes integrované Magnolia store, ve kterém se nacházejí aplikace od tvůrců systému a třetích stran. Editoři zde mohou snadno definovat strukturu a obsah webu. Tvůrci obsahu navíc mohou editovat obsah stránek přímo při prohlížení stránek a nemusejí se tak přepínat do administrátorského rozhraní. Pro editory je tak velmi snadné rychle provést požadovanou změnu. Další silnou vlastností systému je snadné vytváření šablon, ke kterému na rozdíl od většiny jiných CMS nejsou potřeba velké technické znalosti. Přes admi-

nistrátorské rozhraní lze také definovat jednoduché online formuláře, přes které lze vytvářet např. registrační formuláře, dotazníky , atd.. Magnolie je vhodná k vytváření velkých firemních prezentací a tvůrci na svých stránkách udávají informaci o více než 2 500 nasazení.

Liferay

Momentálně nejrozšířenější open source portál postavený na standardu JSR 168. Ve vývoji od roku 2000 firmou Liferay, která táhne vývoj aplikace vpřed. Liferay je vhodný k vytváření intranetových portálů a nabízí velkou spoustu portletů. V poslední verzi obsahuje i velmi kvalitní sadu portletů, které poskytují pokročilé funkce správy obsahu. Navíc má systém dobrou architekturu a využívá osvědčené Java frameworky a knihovny, díky čemuž je lze na tomto systému provozovat velké internetové prezentace, které musejí zvládnout velkou zátěž. Např. český web T-Mobile běží na Liferayi.

Další výhodou Liferaye je pěkné uživatelské rozhraní, ve kterém lze celkem snadno spravovat obsah webu. Jako první přišli s možností drag & drop vkládáním portletů na stránku. Pro vývojáře je podstatné, že je k dispozici kvalitní dokumentace, velká a živá komunita a také o tomto systému již vyšla celá řada knih. Navíc většinu z předinstalovaných 60 portletů lze celkem snadno upravit pro potřeby daného nasazení systému a existuje jasný postup, jak tyto úpravy provádět. Je také možné zasahovat do řady vnitřních mechanismů systému bez nutnosti zásahu do samotných zdrojových kódů. Vytvořená rozšíření jsou jasně oddělená od jádra systému, čímž se značně usnadňuje upgrade. Administrátoři mohou systém provozovat na většině dostupných aplikačních a databázových serverech a lze ho snadno napojit na LDAP. Opět je tento systém dostupný v open source a enterprise edici.

Závěr

V následující tabulce jsou stručně shrnuty výhody systému napsaných v Javě a PHP. Nevýhody je těžké sepisovat a vždy je při výběru nutné brát ohled k čemu je daný systém vhodný. Java CMS jsou dle všeho vhodnější na tvorbu specializovaných a velkých internetových prezentací, ve kterých je potřeba dodělat celou řadu rozšiřujících modulů, které jsou specifické pro daný web.

Výhody

JAVA	PHP
Dobře navržená architektura	Rychlá a snadná instalace na víceméně jakémkoliv hostingu
Vhodné pro robustní a výkonná řešení	Jednoduché uživatelské rozhraní zaměřené na editory a rychlé vytváření webu
Možnost instalace na většinu dostupných databází	Snadná instalace rozšiřujících modulů a šablon
Kvalitní administrační rozhraní, ve kterém se však může editor obsahu těžko orientovat	Velký počet uživatelů a více dostupných informačních zdrojů
Podpora standardů v oblasti CMS (JCR, CMIS)	Velké množství kvalitních šablon a modulů (diskuze, obrázkové galerie, ...)
Lze použít jako aplikační platformu a snadno dodělat jakoukoliv požadovanou funkcionalitu	

Jak bylo zmíněno výše, je velká škoda, že se za těch přibližně 11 let, kdy se používají redakční systémy, nevyprofiloval jeden, okolo kterého by se shromáždila velká, aktivní komunita a který by poskytoval kvalitní rozšiřující moduly.

Literatura

[1] Krupička, J.: *Publikační systémy postavené na Java technologii.* Plzeň : 2006.