LUPA.CZ

Lupa.cz » **IPv6 Mýty a skutečnost, díl V. - Zjednodušené hlavičky**

IPv6 Mýty a skutečnost, díl V. - Zjednodušené hlavičky

10. 3. 2011 6:25 Tomáš Podermański, Matěj Grégr

Seriál Pohněme s IPv6

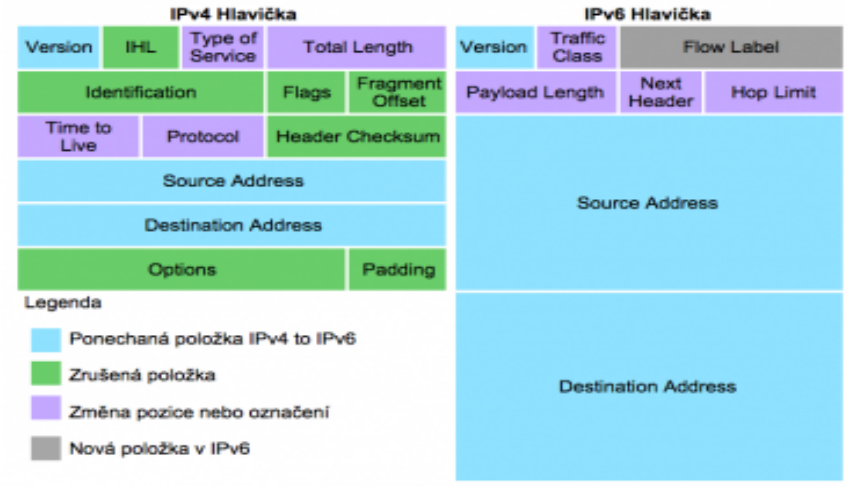
- IPv6 Mýty a skutečnost: díl III. - podpora end-to-end služeb
- IPv6 Mýty a skutečnost, díl IV. - Podpora autokonfigurace
- IPv6 Mýty a skutečnost, díl V. - Zjednodušené hlavičky
- IPv6 Mýty a skutečnost, díl VI. - Bezpečnostní mechanizmy
- IPv6 Mýty a skutečnost, díl VII. - Podpora Multicast a anycast provozu

Všechny díly seriálu

Jedním z požadavků na protokol IPv6 bylo výrazné zjednodušení hlaviček IP protokolu. Snahou bylo eliminovat co nejvíce položek, a v maximální míře tak usnadnit práci směrovačům. Dalším požadavkem bylo zavedení mechanizmu umožňujícího budoucí rozšiřování protokolu o další vlastnosti.

Šetří se, kde se dá: úsporná IPv6 hlavička

Zkušenosti získané provozováním protokolu IPv4 vedly k efektivnějšímu návrhu struktury základní IPv6 hlavičky. Základní porovnání položek IPv4 a IPv6 hlavičky je zachyceno na následujícím obrázku:



Porovnání položek základní IPv4 a IPv6 hlavičky

Jak vidíme, některé položky byly zrušeny bez náhrady. Jedná se zejména o položku **kontrolního součtu**. Kontrolní součet je zpravidla řešen na úrovni linkové (nižší) a transportní (vyšší) vrstvy, takže jej lze v IP hlavičce bez problému vynechat.

Další položky doznaly spíše kosmetických změn. K pouhé terminologické změně došlo u položky Time to

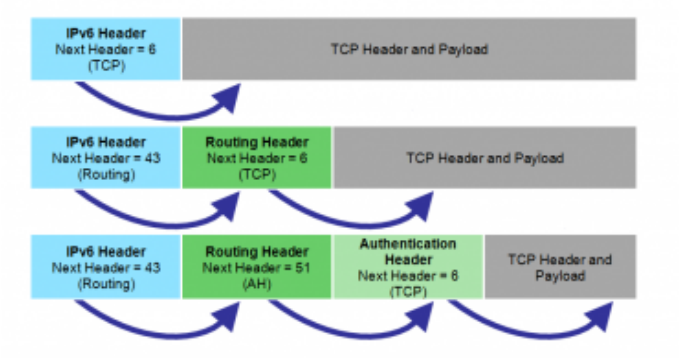
Live (**TTL**), která je nahrazena polem **Hop Limit**. Stejně jako v IPv4 je hodnota snížena při průchodu každým směrovačem a v případě, že je dosaženo hodnoty 0, je paket zahozen.

Technologickým garantem seriálu Pohněme s IPv6 je CZ.NIC .

Každá úspora něco stojí: rozšířené hlavičky

Další položky jako volby (***Options***) a posun fragmentu (***Fragment Offset***) nebyly zrušeny, ale přesunuty do tzv. rozšiřitelných hlaviček. Obdobnou změnu doznala i položka identifikace protokolu vyšší vrstvy (***Protocol***). V případě IPv4 jsme v této položce nalezli hodnotu odpovídající typu protokolu transportní vrstvy – tedy *TCP*(6), *UDP*(17), *ICMPv6*(58), *GRE*(47), *OSPF*(89) dle čísel přídělených organizací IANA .

Právě zde přichází IPv6 s nejvýznamnější změnou ve formátu hlaviček. Na položce již nenajdeme konkrétní číslo protokolu vyšší vrstvy. Položka ***Next Header*** (další hlavička) obecně odkazuje na typ hlavičky, která následuje. Další hlavička už nemusí nést pouze protokol vyšší vrstvy, ale další rozšiřující volby, které mohou ovlivňovat způsob zpracování datagramu v průběhu transportu anebo na koncovém zařízení.



Rozšiřující hlavičky IPv6

Vlastnímu datovému obsahu (například hlavička UDP) může předcházet řada volitelných hlaviček, které jsou zřetězeny do lineárního seznamu. Jednotlivé rozšiřující hlavičky by neměly být v paketu umístěny zcela nahodile. Standardy přesně specifikují pořadí rozšiřujících hlaviček tak, aby jejich zpracování bylo pro jednotlivá zařízení co nejjednodušší. Hlavičky, které zajímají směrovač, musí být v seznamu jako první a hlavičky určené pro příjemce jako poslední.

V případě, že se v budoucnu vyskytne potřeba rozšíření vlastností protokolu, nadefinuje se další typ hlaviček. Takhle jsou už v současné době řešena některá rozšíření protokolu, jako např. volitelné šifrování na úrovni síťové vrstvy (IPSec), podpora mobility (Mobility Header) anebo multuhoming (Shim6).

Na první pohled je zřejmé, že oba požadavky, tj. celkové zjednodušení základní hlavičky v podobě redukce položek a možnost dalšího rozšiřování v podobě zřetězených hlaviček, bezezbytku naplnily oba původní požadavky. Z nového uspořádaní hlaviček však plynou některé často opomíjené komplikace. Pojďme se na ně podívat.

Zpracování zřetězených hlaviček

První problém představuje samotné **řetězení hlaviček**. Skutečnost, že se jedná o lineární seznam, který nelze procházet jinak než sekvenčně, značně komplikuje zpracování takového paketu každému zařízení, které zpracovává protokoly vyšší vrstvy. Nemusí se nutně jednat pouze o firewally, IDS a IPS systémy, ale

také například všechny přepínače podporující efektivní práci s multicastem (*MLD Snooping*), kontrolu podvržených oznámení směrovače (*RA a DHCPv6 Snooping*, viz. předchozí díl seriálu), atd. Pokud chceme zpracovávat jakékoliv informace na úrovni vyšší protokolové vrstvy (*TCP, UDP, ICMPv6*), nezbytně musíme procházet všechny položky zřetězených hlaviček.

To nám ani tak nevadí při zpracování v software, ale výrazným způsobem se komplikuje návrh hardware. Ukažme si problém na již dříve zmíněném mechanizmu pro efektivní práci s multicastem (*MLD snooping*). Základní klasifikace řídících zpráv je typicky prováděna na úrovni hardware (například na vstupním portu přepínače). Po základní předfiltraci pouze na MLD zprávy následuje analýza MLD zpráv a nastavení parametrů pro MLD snooping, což řeší procesor přepínače (zpravidla nevalného výkonu). Jak vidíme, kvalitní předfiltrace se správným zpracováním rozšiřujících hlaviček se týká téměř každého síťového zařízení, které podporuje pokročilejší síťové funkce.

V případě protokolu IPv4 jsme byli schopni získat informace vyšších vrstev v jednom až dvou taktech v deterministickém čase. V IPv6 je nutné procházet lineární seznam s předem neznámým počtem položek a jejich datovou délkou. Nepříjemným důsledkem je nedeterministický čas zpracování jednotlivých paketů, což může představovat značný problém výkonově kritických systémů.

Zatím to však vypadá, že výrobcům se daří problém zpracování hlaviček docela úspěšně řešit, nicméně řada problémů může být zatím skrytá, díky relativně malým datovým provozům na protokolu IPv6 a také tím, že podpora na úrovni hardware je zatím implementována zejména do dražších zařízení. U přepínačů pohybující se v nižších cenových hladinách bychom podporu klasifikace IPv6 v hardware hledali zatím marně.

Cílené přeuspořádání hlaviček

IPv6 přesně definuje, v jakém pořadí mají být jednotlivé rozšiřující hlavičky uspořádány. Na začátku musí být umístěny hlavičky nesoucí informace pro transportní zařízení (například směrovače) a na konci seznamu hlavičky nesoucí informace pro koncová zařízení. Slušné a spořádané systémy bezesporu nebudou mít problém toto pravidlo dodržet. Zde se však otevírá široké pole pro nové typy útoků zaměřených na **cílené přeuspořádání hlaviček** a **záměrné prodlužování seznamu hlaviček**. Jednotlivá zařízení se s nečekanými kombinacemi dokážou vypořádat různým způsobem. Možností je spousty: od plného propuštění, zahození, až po nějaké pěkné buffer overflow nebo crash zařízení.

Ti, co by si mysleli, že tento problém je pouze hypotetický, si můžou otestovat svá zařízení (ještě jednou upozorňuji: **pouze SVÁ zařízení!**) nástrojem isic6 . Tento jednoduchý nástroj určený pro testování IPv6 zařízení dokáže generovat IPv6 pakety dle zadaných parametrů včetně možnosti nahodilého generování hlaviček.

Použití je velice jednoduché. V příkazové řádce nastavíte několik málo parametrů a už pouze sledujete, co se děje se souvisejícími zařízeními. Během několika málo minut zjistíte, jak plně zatížit CPU směrovače, a tím vyřadit z provozu signalizační protokoly (VRRP, SPT, směrovací protokoly). Dalšími experimenty odhalíte vhodnou kombinaci hlaviček umožňující dočasné vyřazení filtrace na úrovni ACL atd.

Hop-by-hop směrování (RH0)

Další typ útoků je velice dobře znám už z IPv4 světa, kde patří mezi jakousi „klasiku“. Podstata útoku je opřena o zneužití mechanizmu tzv. **hop-by-hop směrování**, který se občas také označuje „source routing“ (neplést s jinými nesouvisejícími mechanizmy, které se také označují jako „source routing“). V rámci tohoto mechanizmu je možno v hlavičce paketu předepsat cestu, kudy chceme, aby po síti putoval. Útočník si s využitím tohoto mechanizmu může dopravit datagram na některý důvěryhodný uzel (například v DMZ), a následně už bez omezení pokračovat dále ve vnitřní síti. Další problém, který tento mechanizmus přináší, je možnost zahlcení linky. Pokud se do odesílaného paketu vhodně předepíšou mezilehlé adresy, je možné na jedné lince docílit cyklického putování paketu tam i zpět a tímto ji doslova ucpat.

Ve světle těchto zkušeností bylo prostřednictvím draftu s výstižným názvem I draft-jabley-ipv6-rh0-is-evil navrženo zrušení a nařízeno nekompromisní zahazování všech datagramů obsahujících rozšiřující hlavičku RH0. Následně v roce 2007 prostřednictvím RFC 5095 byla tato rozšiřující hlavička definitivně

zrušena a zpracování paketu nařízeno v poněkud měkčí podobě.

Zdálo by se, že problémem rozšiřující hlavičky RH0 se z dnešního pohledu nemusíme nějak zvlášť zatěžovat. Řada základních implementací IPv6 ovšem vznikla ještě před zrušením tohoto mechanizmu, a naprostá většina systémů jej dnes podporuje. Zpracování rozšiřující hlavičky RH0 je možné v některých systémech potlačit. Na směrovačích typicky direktivou *no ipv6 source-route*, v některých UNIX-like systémech je možné mechanizmus konfigurovat prostřednictvím *sysctl*. Naštěstí řada dostupných implementací firewallu umožňuje klasifikaci a případnou filtraci datagramů obsahující hlavičku RH0.

Jak si jistě mnozí pamatují, obdobný mechanizmus je také implementován v protokolu IPv4. Již před lety byl považován za zdroj potenciálních problémů, a proto byl správci systémů zpravidla deaktivován. Bohužel se tato zkušenost nijak nepromítla do původního návrhu IPv6. Vznikl tak mechanizmus, který byl nejdříve složitě navrhován, následně výrobci zařízení a SW pracně implementován, aby byl správci systémů všemožně potlačován. Jako dovršení všeho byl v konečném důsledku zcela zrušen. Názorně můžeme vidět, jakými pozoruhodnými slepými uličkami se může občas vývoj protokolu ubírat.

Fragmentace paketů

Dalším, také již dříve známým typem útoku, je útok na fragmentaci paketů. Fragmentace je proces, který přijde ke slovu v případě, kdy potřebujeme transportovat sítí větší datagram, než umožňují jednotlivá zařízení a propoje umístěné v cestě. V takovém případě je datagram rozdělen odesílatelem na jeden nebo více částí, ty jsou očíslovány a na straně příjemce opět rekonstruovány. Fragmentované pakety jsou řešeny prostřednictvím speciální rozšiřující hlavičky (č. 44). V rámci této hlavičky je přenášena identifikace fragmentů patřících k sobě (*Identification*), pozice fragmentu v původním datagramu (*Fragment offset*) a příznak, zda je příslušný fragment posledním v pořadí (*More*). Pro aplikaci je celý proces zcela transparentní a o vše se postarají vrstvy na síťové úrovni protokolu.

Skutečnost, že je původní datagram rozdělen na více kousků, způsobuje vrásky na čele nejednomu zařízení, které se zabývá inspekcí přenášených dat. Pro potřeby analýzy musí zařízení provádět rekonstrukci původního datagramu (*Fragment Reassembling*), anebo alespoň udržovat stavovou informaci k jednotlivým fragmentovaným tokům. Zde vzniká opět pole pro vedení útoků v podobě záměrně přeuspořádaného pořadí jednotlivých fragmentů, vyčerpání zdrojů vytvořením velkého množství toků s rozdílnou identifikací atd. V principu se však tyto typy útoků příliš neliší od obdobných útoků známých z IPv4.

Na rozdíl od IPv4, kde mohlo k fragmentaci docházet na kterémkoliv transportním zařízení, mohou v IPv6 fragmentaci provádět pouze koncové uzly. V protokolu IPv4 nebylo nezbytně nutné mít k dispozici zvláštní prostředky pro určování velikosti jednotlivých fragmentů. Směrovač na základě MTU definovaných na svých rozhraních dokázal identifikovat, zda je nutno přistoupit k fragmentaci datagramu či nikoliv.

V případě IPv6 koncové uzly nemohou předem tušit, jak velký paket je možné transportovat, a musí proto mít prostředek, jak tuto informaci zjistit. Problém řeší algoritmus objevování MTU cesty (RFC 1981). V případě, že některý z transportních uzlů v síti není schopen datagram příslušné velikosti přenést, je vyslána ke zdroji dat ICMPv6 zpráva informující o tom, že paket je příliš velký (*Packet too big – ICMPv6 type 2*), a současně je doplněna informace o maximální velikosti datagramu, kterou lze transportovat. Zdrojový uzel na základě této informace zmenší velikost odeslaného datagramu (například rozdělením na více fragmentů) a pokusí se datagram odeslat znova. Vzhledem k tomu, že zpráva informující o velkém datagramu je přenášená protokolem ICMPv6, **nelze u IPv6 beztrestně blokovat veškerý ICMPv6 provoz na firewallu**, jak se tomu mnohdy dělo v případě ICMP(v4). Případná bezpečnostní politika musí definovat specifická pravidla pro jednotlivé typy ICMPv6 zpráv, kde je zohledněn účel jednotlivých typů zpráv.

Protokol, anebo rozšiřující hlavička?

Jak jsme si již v úvodu řekli, pole další hlavička (*Netx Header*) může obsahovat dva základní typy dat. Je to buď rozšiřující hlavička (*Routing Header, Mobility, Shim6 atd.*), v jejímž těle je vždy zaznamenán odkaz na další hlavičku, anebo identifikace protokolu vyšší vrstvy (*TCP, UDP, ICMPv6, GRE atd.*). V případě definice protokolu transportní vrstvy již není informace o navazující hlavičce uvedená a předpokládá se, že

definice protokolu je posledním prvkem v zřetězeném seznamu. Takto vypadá uspořádaní poměrně jednoduše a logicky, ale je zde skryt jeden malér. Z vlastní definice ať už rozšiřující hlavičky nebo protokolu není možné identifikovat, zda se jedná již o konečnou položku seznamu či nikoliv. Za normálních okolností to nevadí. Zařízení zpracovávající hlavičky (*Firewall, IDS, IPS, NetFlow Sonda*) přesně ví, jak s jednotlivými položkami zacházet. Problém ovšem nastává v případě, že zařízení dostane za úkol zpracovat rozšířenou hlavičku anebo protokol, který dosud nezná. V takovém případě se může rozhodnout dvojím způsobem:

- Zahodit paket, kde je odkaz na neznámou hlavičku a zdroji je vygenerována ICMPv6 zpráva (*unrecognized Next Header*). Tímto je však narušena původní myšlenka rozšiřování hlaviček, protože s příchodem nového rozšíření se nový typ hlavičky musí naučit všechny systémy. Tento způsob zpracování hlaviček je doporučený pro konfiguraci na úrovní korporátního firewallu.
- Předpokládat, že neznámá hodnota v položce další hlavička (*Next Header*) je rozšiřující hlavičkou a pokračovat ve zpracování do doby než narazíme na systému známou hodnotu. Zde ovšem vzniká problém s rozšiřitelností nových protokolů. V případě, že ono neznámé číslo je nějakým novým protokolem (např. nové lepší TCPv2 či GREv2) bude jeho obsah nesprávně interpretován jako rozšiřující hlavička na základě těchto dat vyvodit nesprávné závěry. V současné době ve výchozím stavu velké množství firewallů zpracovává hlavičky právě tímto způsobem.

Řešení problému není zcela jednoduché. Čistá forma řešení je rozšířit rovněž hlavičky protokolů o dva bajty tak, aby i zde byla položka další hlavička (*Next Header*) a poslední záznam v seznamu by byl ukončen speciální hodnotou (*No Next Header*). To však naráží na praktickou neproveditelnost v podobě zásahu do hlaviček všech protokolů transportní vrstvy. Podrobněji se problémem zabývá prezentace s názvem *IPv6 Extension Header – Processing Implications* , HAGEN PAUL PFEIFER.

Toky

Zcela novou položkou v základní IPv6 hlavičce je identifikace toků. V této položce by měly být pod vhodným číslem identifikovány datagramy toku, které spolu souvisí. Například datagramy jedné TCP relace, data multicast streamu atd. V současné době je tato položka specifikována prostřednictvím RFC 3697 , které specifikuje tvorbu identifikace toku. Ve většině IPv6 paketů je tato položka nastavena na hodnotu 0 (nespecifikovaný tok) a většina systémů tuto položku zatím ignoruje.

Možnosti obrany

Jak jsme si již řekli, rozšířené hlavičky potenciálně nabízejí nové možnosti pro narušení bezpečnostní politiky. Je tedy přirozené, že zařízení typu firewall atd. musí být rozšířena o prostředky umožňující klasifikaci a následnou filtraci vybraných typů hlaviček.

Většina systémů podporujících filtraci na úrovni hlaviček (Linuxový netfilter, FreeBSD ipfw) umožňuje zatím klasifikovat pouhou přítomnost či nepřítomnost příslušné rozšiřující hlavičky z předem definovaného výčtu. Pokud byste chtěli na úrovni ip6tables blokovat datagramy obsahující například rozšiřující hlavičku *shim6* (140), máte smůlu. Můžete ovšem politiku postavit tak, že propouštíte pouze ty typy hlaviček, vůči kterým máte důvěru, a vše ostatní zahazujete. Na úrovni korporátního firewallu by mělo být toto nastavení samozřejmostí.

Podpora filtrace hlaviček v komerčních systémech je zatím různá – někde není podporována vůbec, někdy je možné blokovat vybrané typy hlaviček například prostřednictvím ACL (*Access Control List*).

Závěr

Zjednodušené IPv6 hlavičky jsou velice často předkládány jako výrazné vylepšení protokolu IPv6. Často je však opomíjen fakt, že rozšiřující hlavičky, které otevírají cestu pro přidávání nových vlastností protokolu,

si vybírají nemalou daň v podobě nových bezpečnostních rizik a složitějšího zpracování paketů. Dá se předpokládat, že s postupnou penetrací IPv6 a jeho atraktivitou pro útočníky budou stále intenzivněji zdokonalovány bezpečnostní prvky řešící tuto problematiku.

V současné době je vhodné mít na paměti existenci rozšířených hlaviček při výběru síťových zařízení (např. Firewally, IPS, IDS). Při definici bezpečnostní politiky je pak potřeba zpracovat nejen rovinu protokolů a služeb, ale také politiku zpracování rozšířených hlaviček. Nepříjemným faktem zůstává, že možnosti filtrace jsou v mnohých systémech zatím značně omezené a zdaleka nedosahují komfortu, na který jsme zvyklí při definici bezpečnostních politik, například na úrovni služeb. Provozní praxe si však jistě vynutí postupné zdokonalování podpory těchto prostředků v jednotlivých systémech a bezpečnostních zařízeních.

Tomáš Podermański

Autor pracuje jako správce metropolitní sítě VUT v Brně. Podílí se na řešení projektů zaměřenýchna bezpečnost a monitoring sítí. Je aktivním členem evropského projektu GÉAN3 v aktivitě Campus Best Practice.

Matěj Grégr

Studuje na Fakultě informačních technologií VUT v Brně. Snaží se porozumět počítačovým sítím a teoretické znalosti pak (ne)úspěšně uplatňovat v praxi jako správce kolejní sítě VUT.

Seriál Pohněme s IPv6

- IPv6 Mýty a skutečnost: díl III. - podpora end-to-end služeb
- IPv6 Mýty a skutečnost, díl IV. - Podpora autokonfigurace
- IPv6 Mýty a skutečnost, díl V. - Zjednodušené hlavičky
- IPv6 Mýty a skutečnost, díl VI. - Bezpečnostní mechanizmy
- IPv6 Mýty a skutečnost, díl VII. - Podpora Multicast a anycast provozu

Všechny díly seriálu