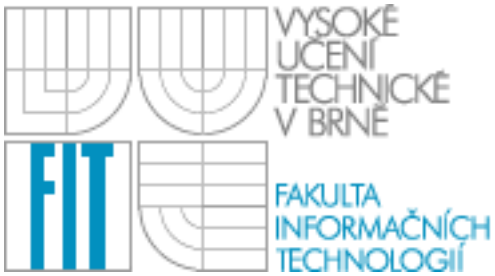

Dokumentácia k projektu pre predmet ISA

Neighbor Discovery Proxy

28. november 2011

Autor: Dávid Molnár, xmolna02@stud.fit.vutbr.cz

VUT FIT Brno

Obsah

1 Úvod

Premostenie viac rozhraní do jedného celku má niekoľko výhod. Jeden prefix podsiete je dostatočný pre podporu viac fyzických spojení. Nie je nutné prideliť čísla podsiete rôznych sietí. To znamená zjednodušenú správu. Premostenie niektorých typov médií však vyžaduje podporu sieťovej vrstvy.

Klasické premostenie na linkovej vrstve má nasledujúce obmedzenia (okrem iného):

- vyžaduje, aby všetky rozhrania podporovali promiskuitný režim
- vyžaduje, aby všetky rozhrania podporovali rovnaké linkové adresovanie

Neighbor Discovery Proxy (ND Proxy) vzniklo, aby umožnilo podporu premostenie za týchto okolností. Proxy, popísaný v tomto dokumentu na sieťovej vrstve pracuje s protokolom IPv6.

Požiadavky:

- podpora pripojenia viacerých segmentov s jedným prefixom podsiete
- podpora médií, ktoré nemôže byť premostené na linkovej vrstve
- nevyžaduje žiadne zmeny v existujúcich konfiguráciách smerovačov
- poskytuje plnú konektivitu medzi všetkými uzlami

2 Návrh aplikácie

Obidve rozhrania proxy by mali byť podľa [4] v režime „all-multicast“. To znamená, že by mali prijímať všetky pakety multicastového vysielania. Hoci nie všetky rozhrania podporujú promiskuitný režim, táto implementácia proxy skúsi dať rozhranie do promiskuitného režimu.

Proxy udržiava IPv6 Neighbor Cache, ktorý slúži na vyhľadanie linkovej adresy daného uzla podľa IPv6 adresy.

Keď je na rozhranie prijatý paket s akoukoľvek zdrojovou IPv6 adresou inou ako nešpecifikovanou, Neighbor Cache daného rozhrania je konzultované s cieľom nájsť položku pre zdrojovú IPv6 adresu. Ak sa žiadny záznam neexistuje, je vytvorený nový.

Prijatý paket je analyzovaný, či je to ICMPv6 Neighbor Discovery alebo Neighbor Solitation paket (ostatné typy v tejto implementácie neuvažujeme). Tieto pakety nesú adresy linkovej vrstvy a preto proxy nahradí tieto adresy so svojou linkovou adresou rozhrania, ktoré paket odosiela.

Ak je prijatý akýkoľvek iný IPv6 multicast paket, je odoslaný na druhé rozhranie bez zmeny (je použitá nová hlavička linkovej vrstvy). IPv6 Hop Limit nie je aktualizovaný a žiadne ICMP chyby nie sú poslané.

Ak je prijatý akýkoľvek iný IPv6 unicast paket, ktorý nie je lokálne určený, je odoslaný bez zmeny na druhé rozhranie (je použitá nová hlavička linkovej vrstvy) s linkovou adresou, ktorú našiel v Neighbor Cache rozhrania (na ktoré paket posielá) podľa cieľovej IPv6 adresy. Opäť IPv6 Hop Limet nie je aktualizovaný a žiadne ICMP chyby nie sú poslané. Keď sa nenájde cieľová IPv6 adresa v Neighbor Cache, paket je zahodený.

Hlavička linkovej vrstvy každého odoslaného paketa je modifikovaná nasledujúcim spôsobom:

- zdrojová adresa je adresa rozhrania, ktoré paket odosiela
- cieľová adresa je adresa v Neighbor Cache príslušný s cieľovou IPv6 adresou

Podľa kapitoly 4.4.1 v [4] proxy by mal poslať „Too Big Messages“. V tejto implementácie však to tak nie je.

Informácie o prijatých a odosielaných paketoch sa píšu na štandardný výstup.

3 Popis implementácie

Pakety zachytíme pomocou knižnice PCap a odosielame pomocou knižnice Libnet. Parametre príkazového riadku sú spracované pomocou funkcie `get_params`. Potom inicializujeme PCap pre oba rozhrania pomocou funkcie `init_pcap`. Nasleduje zisťovanie linkovej a IPv6 adresy oba rozhrania. Potom inicializujeme Libnet. V nekonečnej smyčke čakáme na príjazd paketov. Pomocou funkcie `select` rozhodneme, z ktorého rozhrania prichádzal paket.

Vo funkcii `packet_received` spracovávame pakety. Najprv kontrolujeme ich veľkosť. Ak cieľová IPv6 adresa paketu nie je nešpecifická, pridáme novú položku do Neighbor Cache (alebo ak položka už existuje, aktualizujeme ju). Pomocou funkcie `print_info` a `print_link_info` vypíšeme informácie (viď. nížšie) na štandardný výstup. Pomocou funkcie `cmp_mac_addr` a `cmp_in6_addr` kontrolujeme, či je paket pre nás. Ak áno, ukončíme spracovanie. Potom rozhodneme, či je cieľová IPv6 adresa multicastová. Ak nie, vyhľadáme v Neighbor Cache linkovú adresu pre cieľovú IPv6 adresu. Nasleduje kontrola, či je paket ICMPv6 NS alebo NA. Ak áno, zmeníme linkovú adresu v ICMPv6 payload. Nakoniec pomocou funkcie `libnet_build_ethernet` vyrobíme a funkciou

`libnet_write` odošleme paket.

4 Návod na použitie

Program funguje ako konzolová aplikácia, má iba textové ovládanie. Má dva povinné parametre (na ich poradí nezáleží):

`-i vnútorné_rozhranie` a `-o vonkajšie_rozhranie`.

Príklad na použitie: `ndpv6 -i eth0 -o eth1`.

K ukončeniu programu dôjde po prijatí signálu SIGTERM, SIGQUIT alebo SIGTERM.

4.1 Vzorový výstup

Pakety prichádzajúce (odchádzajúce) na vnútorné rozhranie:

```
IN: 10:02:24, 00:0c:29:a1:e5:89 > 00:0c:29:a1:d3:c2, fd00::1 > fd00::2
IN: time, srcmac > dstmac, ipv6src > ipv6dst
```

Pakety prichádzajúce (odchádzajúce) na vonkajšie rozhranie:

```
OUT: 10:02:30,00:0c:29:a1:d3:c2 > 00:0c:29:a1:e5:89, fd00::2 > fd00::1
OUT: time, srcmac > dstmac, ipv6src > ipv6dst
```

Pakety obsahujúce link-layer informácie (NS alebo NA pakety):

```
IN: 10:05:43, 00:0c:29:a1:e5:89 > 33:33:ff:00:00:02, fd00::1 >
ff02::1:ff00:2, ICMPv6 NS fd00::2, 00:0c:29:a1:e5:89
IN: time, srcmac > dstmac, ipv6src > ipv6dst, ICMPv6[NS|NA] target,
[slla|tlla]
```

Literatúra

[1] CARSTENS, T.: *Programming with pcap*, 2011, http://www.tcpdump.org/pcap.html

[2] SCHIFFMAN, M. D.: *Libnet 101*, *Part 1: The Primer*, Security Focus, 2000

[3] SCHIFFMAN, M.: *Building Open Source Network Security Tools: Components and Techniques*, Wiley, 2003, ISBN 0-47-1205443-3

[4] RFC 4389: *Neighbor Discovery Proxies (ND Proxy)*, http://tools.ietf.org/html/rfc4389

[5] Oracle VM VirtualBox®: *User Manual*, 2011, http://www.virtualbox.org/manual/UserManual.html; http://www.virtualbox.org/manual/ch06.html