Fakulta informatiky a informačných technológií, STU Bratislava

Softvérový prepínač

(Počítačové siete 2)

Autor: Adam Sloboda
Odbor: Informatika
Semester: zimný 2009/2010
Dátum: 6. októbra 2009

Obsah

Zoznam obrázkov

1 Zadanie

Softvérový analyzátor navrhnutý v rámci cvičení z predmetu Počítačové siete I rozšírte o ďalšie funkcie (vysielanie, filtrovanie, monitorovanie ....) tak, aby výsledným riešením bol návrh a implementácia softvérového multilayer prepínača (switch). Prepínač má byť iba dvojportový (dve sieťové karty, port 1, 2), ovládanie interfejsov realizujte príslušnými paketovými ovládačmi.

Prepínač navrhnite a implementujte tak, aby:

- poskytoval štatistické informácie vrstvy 2-4RMOSI o počte PDU, ktore budu zretelne zobrazovat spravne fungovanie switchu. (nezabudnut RESET tlacitko)
- zobrazoval smerovaciu tabuľku: MAC adresa – port; Prepínač sa svoju smerovaciu tabuľku učí.
- filtroval komunikáciu v rozsahu analyzátora z PS1 t.j. vrstva 2-4 RMOSI, aplikacie (well known ports+lubovolny). Riešenie navrhnite ako zoznam pravidiel tak, aby bolo možné realizovať aj lubovolnu kombináciu filtrov, napr. pre nejakú IP povoliť iba http komunikaciu a pre nejaku MAC zakazat "ping". Filtre rozlisujte v smere ïn/out"per port switcha.(Napr. Host A sa nedostane von na web (HTTP), ale u neho beziaci appache server bude dostupny)

Pri spracovaní koncepcie návrhu prepínača rozoberte obe techniky prenosu rámcov (Store & Forward, Cut-Through) ako aj možnosť návrhu viacportového prepínača. Vo výslednom návrhu a implementácii si zvoľte návrh iba dvojportového prepínača a iba jednej techniky prenosu – jej výber však zdôvodnite.

Softvérový most implementujte pod OS Windows príp. Linux.

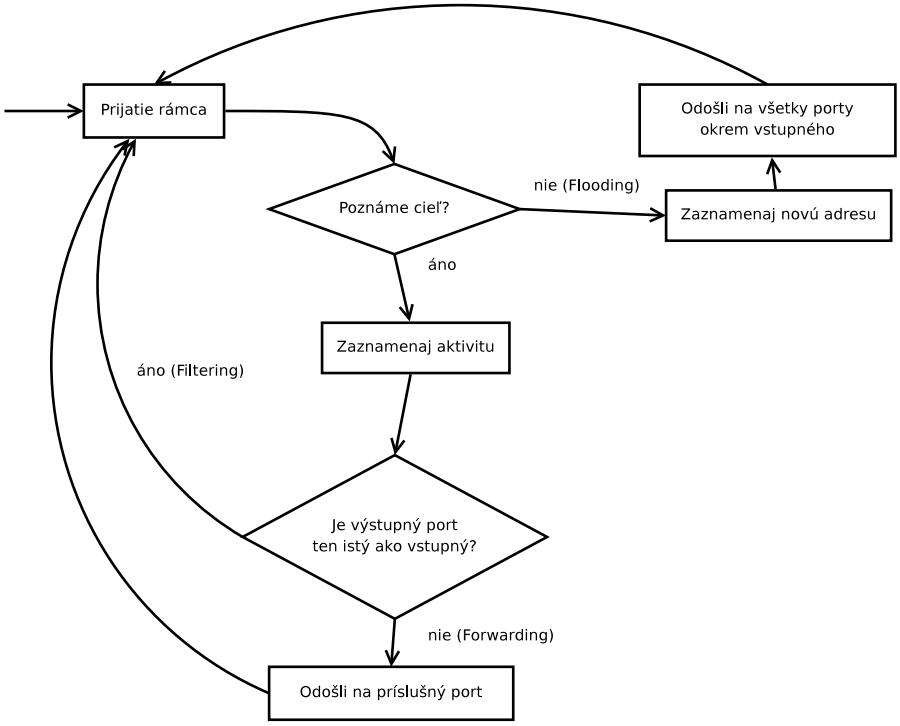
2 Riešenie

2.1 Analýza

Prepínač je zariadenie, ktoré pracuje na druhej (data-link) vrstve OSI sieťového modelu. Na ethernet LAN sieti to znamená prijatie ethernet rámcov a odoslanie na príslušný výstupný port.

Správanie transparentného prepínača v každej situácii sa dá opísať piatimi procesmi:

1. *Learning* - zaznamenanie zdrojovej MAC adresy k vstupnému portu
2. *Flooding* - ak nevie kde sa nachádza cieľová MAC adresa, posiela rámec na všetky porty okrem vstupného
3. *Forwarding* - pokiaľ už cieľovú MAC má zaznamenanú, posiela na príslušný port
4. *Filtering* - pokiaľ je cieľová MAC na tom istom segmente, rámec sa nemusí posielať
5. *Aging* - odstraňovanie záznamov z tabuľky ak za danú dobu nebol zaznamenaný rámec pochádzajúci z tejto adresy



Obrázok 1: Spracovanie rámca

2.1.1 Špeciálne požiadavky

Vzhľadom na to, že na počítači beží operačný systém s plne funkčným network stackom (rámce nezávisle od nás spracováva a odpovedá na ne), bude nutné ignorovať rámce, ktoré pochádzajú z alebo sú určené pre lokálne zariadenia. Na funkcionalitu prepínača to nemá žiaden vplyv.

2.2 Návrh

Na základe požiadaviek vieme rozlíšiť základné samostatné súčasti prepínača:

- tabuľka známych adries (s príslušným portom) - asociatívne pole MAC adresa - port, ktoré pomáha určiť kam rámec poslať
- port - spracuje prichádzajúci a odchádzajúci rámec, uchováva vlastné štatistiky
- filter - podľa zadaných pravidiel zamedzí prijatiu alebo odoslaniu rámca

Tieto tri súčasti sa dajú implementovať ako triedy. *Tabuľka* je globálna inštancia zdieľaná všetkými portami. Pre každý port sa vytvorí inštancia triedy *Port*. Každý port potrebuje vlastnú inštanciu triedy *Filter*.

Port načíta rámec a podľa tabuľky MAC adries sa rozhodne na aký port sa má odoslať (*Flooding, Forwarding, Filtering*). Tiež aktualizuje štatistiky a tabuľku adries (*Learning*). Proces *Aging* sa vykonáva periodicky nad celou tabuľkou.

Načítanie býva realizované dvoma metódami:

- *Cut-through* metóda miesto načítania celého rámca začne odosielať tak skoro ako je to možné – po načítaní cieľovej adresy (a bez skontrolovania kontrolnej sumy). Metóda cut-through by mala znížiť latenciu na minimum, no nemôže byť aplikovaná pri prechode rámca z pomalšieho portu na rýchlejší.
- *Store & forward* je pre nás vhodnejšia metóda, pretože znamená menšiu réžiu. Tiež potrebujeme načítať rámec až po hlavičku štvrtej vrstvy aby sme mohli aplikovať niektoré filtre (TCP/UDP port). Prijímanie rámca taktiež nedokážeme ovplyvniť bežnými programovými prostriedkami.

Filter si ukladá pravidlá, ktoré určujú aké rámce má odfiltrovať. Je potrebné rozlíšiť vstupné a výstupné pravidlá, preto je potrebné ich uchovávať zvlášť.

2.2.1 Pravidlá filtrovania

Pravidlo zadané switchu bude slovné a na jednom riadku (`#` bude označovať komentár do konca riadku). Syntax pravidla je:

```
DEVICE in/out [icmp/ip/tcp/udp/any]
[ from/to ip/mac ADDRESS ]*
[ from/to tcp/udp/port PORT ]*

# IP:  ip[/mask]
# MAC: xxxxxxxx alebo xx:xx:xx:xx:xx:xx
```

Pokiaľ nie je zadané politika pre filter, pravidlá zakazujú rámce vyhovujúce pravidlám a povoľujú všetky ostatné. Politika sa dá zmeniť nasledujúcim pravidlom:

```
DEVICE in/out [policy allow/deny]

# allow: povolí ostatné rámce
#  deny: zakáže ostatné rámce
```

Táto syntax umožňuje zadať akúkoľvek kombináciu hodnôt.

2.2.2 Viacportový prepínač

Návrh bude fungovať správne s akýmkoľvek počtom portov. Použitie viacerých portov nekladie žiadne ďalšie požiadavky na implementáciu.

2.3 Implementácia

2.3.1 Vývojové prostredie

Ako vývojová platforma bol zvolený operačný systém GNU/Linux. Zadanie bolo implementované v jazyku *Ruby*[1] (verzia 1.8.7) s použitím knižnice na prácu so sieťovými zariadeniami *Pcap*[2] (verzia 0.9.8). Na zistenie MAC adries lokálnych zariadení je použitý Ruby modul *macaddr*[3]. Program používa GUI toolkit *GTK+*[4].

Ruby/Pcap[5] (verzia 0.6) prepojenie jazyka Ruby s knižnicou Pcap postrádalo metódu na posielanie paketov, preto ju bolo nutné doimplementovať (patch je súčasťou zdrojových kódov).

2.3.2 Opis implementácie

Program prijíma názvy zariadení, ktoré sa majú použiť ako porty prepínača. Pokiaľ nie je žiadne zariadenie zadané, vypíše všetky nájdené.

Program vytvorí hlavné okno programu, ktoré slúži na zobrazovanie informácií a stavu prepínača (tabuľku MAC adries, prijaté pakety, štatistiky portov, pravidlá filtrovania). Taktiež umožňuje zadávať pravidlá filtrovania, zastaviť a reštartovať prepínač.

Hlavé funkčné triedy sú `LookupEntry`, `LookupTable`, `Port`, `Filter` a `Rule`.

`LookupTable` je globálna tabuľka (associative array; indexovaná MAC adresou), v ktorej je zaznamenaný port na ktorom sa nachádza a časová značka (`LookupEntry`). Táto tabuľka je periodicky kontrolovaná a po uplynutí doby od poslednej aktualizácie položky určenej konštantou `AGING` vymaže položku z tabuľky.

`Port` je trieda, ktorá vytvorí vlákno. Vo vlákne je cyklus, ktorý načíta paket, aktualizuje tabuľku MAC adries, skontroluje vstupný filter a pošle packet na príslušné výstupné porty, kde sa skontroluje výstupný filter.

[1] `http://www.ruby-lang.org/`
[2] `http://www.tcpdump.org/`
[3] `http://codeforpeople.com/lib/ruby/macaddr/`
[4] `http://www.gtk.org/`
[5] `http://www.goto.info.waseda.ac.jp/~fukusima/ruby/pcap-e.html`

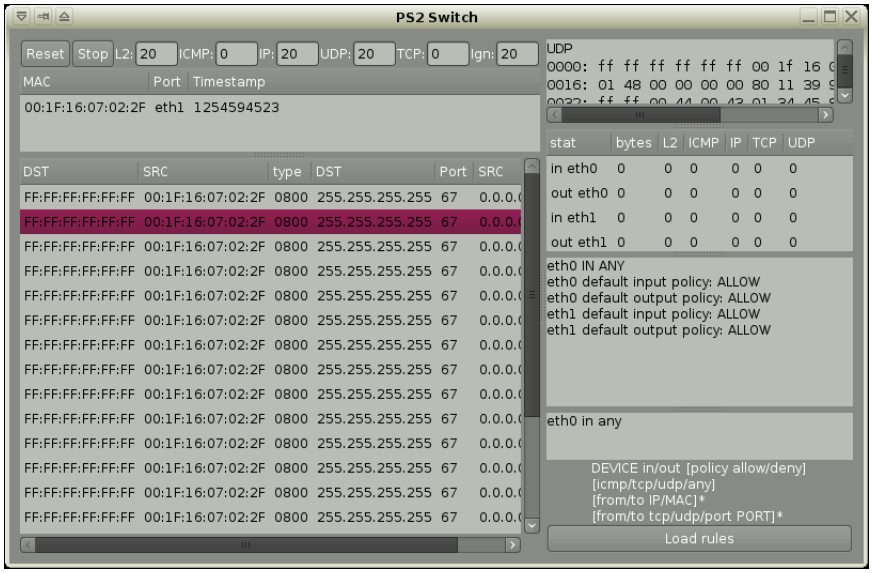
Trieda `Rule` prijíma reťazec, ktorý predstavuje pravidlo filtrovania a metódou `match` dokáže porovnať packet s týmto pravidlom. Výsledkom je vstupné alebo výstupné pravidlo, ktoré kontroluje všetky zadané hodnoty v pakete.

2.3.3 Konštanty programu

`AGING` vyjadruje koľko sekúnd musí uplynúť od poslednej aktivity pred odstránením položky v globálnej tabuľke MAC adries. `AGING_CHECK` vyjadruje v milisekundách periódu aktivácie *Aging* procesu.

2.3.4 Hlavné okno programu

V ľavej časti sú tlačidlá Reset a Stop. Pod nimi je tabuľka MAC adries a tabuľka prijatých rámcov:



Obrázok 2: Okno programu

Prijaté rámce sú označené farbami:

- bez farby - ignorovaný (lokálne zariadenie)
- červená - odmietnutý na vstupe
- oranžová - odmietnutý na výstupe (aspoň jedným z portov)
- žltá - nebol poslaný, pretože pochádza z rovnakého segmentu (Filtering)
- zelená - odoslaný do siete (v prípade viacerých výstupných portov to znamená, že nebol blokovaný ani jedným)

Napravo je umiestnené zobrazenie rámca, štatistiky portov, aktívne pravidlá a politiky, vstup pre pravidlá, syntax pravidiel a tlačítko, ktoré načíta nové pravidlá.

Veľkosť zobrazovaných prvkov sa dá meniť kliknutím na okraj a presunutím na požadovanú pozíciu.

2.3.5 Viacportový prepínač

Implementácia by mala pracovať správne nezávisle od počtu portov.

3 Zhodnotenie

Návrh bol správny a implementácia riešenia je plne funkčná. Implementácia podporuje ľubovoľné množstvo portov. Nevýhoda je, že je implementovaný vo vyššom programovacom jazyku, z čoho vyplývajú väčšie nároky na výpočtový výkon než umožňuje hardvér bežných prepínačov.

Navrhnutý filter dokáže meniť základnú politiku a porovnávať rámce s hodnotami bežne používaných polí. Taktiež umožňuje ľubovoľne kombinovať pravidlá a politiky na vstupe a výstupe portov. Syntax je zapamätateľná a je možné ju ľahko rozšíriť aj o podporu iných polí a protokolov druhej až štvrtej vrstvy OSI modelu.

Použitie multiplatformných knižníc Pcap a GTK+ by malo zabezpečiť prenositelnosť na iné operačné systémy.

A Zdroje a odkazy

- How LAN Switches Work (Transparent Bridging: The Process)
  `http://computer.howstuffworks.com/lan-switch11.htm`
- The Myth of Cut-Through Switching in Datacenter Networks
  `http://www.hpcwire.com/features/The-Myth-of-Cut-Through-Switching-in-Datacenter-Networks-39391297.html?viewAll=y`
- Ruby/Pcap extension library manual
  `http://www.goto.info.waseda.ac.jp/~fukusima/ruby/pcap/doc/index.html`