Koszek-Matyja Network Simulator

Wojciech A. Koszek
wkoszek@FreeBSD.czest.pl
Piotr Matyja
piotr-matyja@o2.pl

Wprowadzenie

Głównym powodem do stworzenia programu KMNSIM była chęć wiarygodnego odtworzenia zachowania sieci Ethernet w sposób wirtualny, bez konieczności faktycznej ingerencji w strukturę i konfigurację jakiejkolwiek fizycznej sieci.

Dzięki programowej emulacji zachowania sieci Ethernet, możliwe jest symulowanie połączeń między komputerami-hostami, koncentratorami, przełącznikami oraz routerami.

2009-06-16: funkcjonalność dot. routerów nie jest jeszcze zaimplementowana

Architektura symulatora

Symulator stworzony został w modułowy sposób. Taka architektura niesie za sobą wiele zalet, od choćby łatwej możlwości testowania oprogramowania, a na rozszerzalności kończąc. W początkowej fazie projektowania problemem stała się sama złożoność możliwej do zasymulowania sieci oraz ilość możliwych do wykonania połączeń.

Naszym zdaniem niemożliwe byłoby dokładne przetestowanie programu, gdyby został wykonany on w sposób typowy dla aplikacji z graficznym interfejsem użytkownika. Zdecydowaliśmy się toteż na skorzystanie z architektury oprogramowania znanego z systemów rodziny UNIX, gdzie główną funkcjonalność realizuje aplikacja tekstowa, uruchamiania w trybie wstadowym. Wynik działania programu zapisywany jest do plik (bądź plików), których poprawność można łatwo zweryfikować. Na podstawie generowanego pliku interfejs graficzny, uruchamiany jako osobny, niezależny program, jest w stanie zwizualizować użytkownikowi poszczególne kroki czasowe symulacji.

Natywny graficzny interfejs użytkownika nie jest jeszcze zaimplementowany; istnieje jednakże możliwość wykorzystania pakietu inżynierskiego Graphviz to podejrzenia w postaci graficznej plików DOT, które reprezentują graf połączeń między elementami sieciowymi.

Zasada działania

Sposób w jaki działa program jest dość prosty. Użytkownik tworzy opis sieci w pliku tekstowym przy pomocy zdefiniowanego języka opisu sieci. Język ten zastępuje typowe, graficzne metody konfiguracji. Jest jednocześnie formatem służącym do zachowywania struktury symulowanej sieci.

Proces konfiguracji–tworzenia pliku wejściowego–rozpoczynamy od specyfikacji wszystkich elementów sieci. Następnie dokonujemy połączeń między nimi w kontekście spinania "wirtualnego" kabla. Poprawna konfiguracja poszczególnych urządzeń oraz posiadania choć jednego aktywnego połączenia umożliwiają rozpoczęcie procesu symulacji.

Każde urządzenie sieciowe do którego można się podłączyć (host, hub, switch, router) posiada pewną ilość interfejsów.

Podstawową jednostką nadawania jest host. Posiada on możliwość nadawania i odbioru danych z sieci, które skierowane zostały do niego. Minimalną ilością hostów nadających jest 1. Host posiada jeden interfejs, który w celu aktywacji musi zostać skonfigurowany adresem fizycznym MAC, adresem IP oraz maską sieciową.

Hub jest pośrednikiem w wymianie pakietów między hostami/routerami. Hub posiada pewną z góry określoną ilość interfejsów. Owe interfejsy nie mogą ulec konfiguracji. Są one tylko przekaźnikiem napływających ramek.

Obecna implementacja zakłada, że zarówno hub jak i switch ma 8 portów; Specyfikacja dowolnej ilości portów wymaga kosmetycznych zmian w kodzie symulatora.

Rola switcha jest analogiczna do roli huba, z jedyną funkcjonalną różnicą, że zamiast przekazywać odebrany pakiet na wszystkie porty, switch dokonuje routingu w warstwie drugiej przekazując pakiet tylko na port, z którego host/router docelowy będą w stanie odebrać dane.

Połączenie między dwoma urządzeniami możliwe jest po specyfikacji 4 parametrów: nazwy i numeru interfejsu miejsca źródłowego połączenia oraz nazwy i numeru interfejsu miejsca przeznaczenia.

Na tym etapie proces tworzenia pliku konfiguracyjnego jest ukończony.

Program interpretuje ów plik oraz sprawdza jego poprawność. W przypadku błędnej składni pliku konfiguracyjnego, program przerywa działanie oraz informuje użytkownika o popełnionych błędach. W przeciwnym razie program rozpoczyna symulację. Symulacja polega na wykonaniu wyspecyfikowanej przez użytkonika ilości kroków, po którym nastąpi koniec symulacji.

Ostateczną fazą działania jest wypisanie wyników symulacji na ekran. Program wspiera kilka formatów wyjściowych – wcześniej wspomniany format Dot pakietu Graphviz, format tekstowy użyteczny do obserwacji faktycznej **implementacji** symulatora lub, najbardziej użyteczny, plik tekstowy obrazujący zachowanie sieci w poszczególnych krokach symulacji.

Opis języka konfiguracyjnego: `host`

Pierwszym elementem potrzebnym do symulacji jest host. Każdy **host** posiada kolejkę nadawczo-odbiorczą. W przypadku wysyłania danych umieszcza w buforze nadawczym dane dot. miejsca źródłowego i przeznaczenia, po czym oznacza dane jako gotowe do wysłania. Automatycznie wraz ze stworzeniem hosta `NAZWA` stworzony zostanie interfejs o nazwie `NAZWA` i numerze `0` (analogicznie do nazewnictwa interfejsów w większości systemów operacyjnych).

Określenie miejsca docelowego, a co za tym idzie, późniejsza decyzja o akceptacji do przetwarzania pakietu dokonywana jest na bazie treści pakietu.

Proponowana składnia dla dodawania hosta

Spowoduje dodanie hosta do sieci	`host newhost1 create`
Usuwa host z sieci	`host mojhost1 remove`

Opis języka konfiguracyjnego: iface

Konfiguracja interfejsu może odbywać się w przypadku, gdy dany interfejs należy do hosta bądź routera.

Konfiguruje ustawienia warstwy fizycznej	`iface NAZWA NUMER mac AA:BB:CC:DD:EE:FF`
Konfiguruje adres IP hosta	`iface NAZWA NUMER ip IP`
Konfiguruje maskę sieciową hosta	`iface NAZWA NUMER netmask NETMASK`

W przypadku hosta `NUMER` musi być równy 0 – host posiada tylko jeden interfejs. Z kolei w przypadku huba/switcha możliwe jest wyspecyfikowanie innych interfejsów w obrębie dostępnych portów. Przykładowo, dla hosta `h1` możliwymi komendami są:

Konfiguruje ustawienia warstwy fizycznej	`iface h1 0 mac 11:22:33:44:55:66`
Konfiguruje adres IP hosta	`iface h1 0 ip 192.168.1.1`
Konfiguruje maskę sieciową hosta	`iface h1 0 netmask 255.255.255.0`

Opis języka konfiguracyjnego: hub

Pierwszym elementem styczności hosta z siecią jest **hub**. Hub jest urządzeniem, które posiada wiele portów Ethernet. Ruch odebrany na jednym z portów jest przekierowywany na wszystkie inne dostępne porty. W celu stworzenia huba w sieci korzystamy z polecenia:

Spowoduje dodanie huba do sieci	`hub NAZWA create`
Usuwa huba z sieci	`hub NAZWA remove`

Na przykład stworzenie hub'a `hu1` możliwe jest dzięki poleceniom:

Spowoduje dodanie huba `hu1` do sieci	`hub hu1 create`
Usuwa huba `hu1` z sieci	`hub hu1 remove`

Opis języka konfiguracyjnego: switch

W celu stworzenia switcha w sieci korzystamy z polecenia:

Spowoduje dodanie huba do sieci	`switch NAZWA create`
Usuwa huba z sieci	`switch NAZWA remove`

Stworzenie switchaa `sw1` możliwe jest dzięki poleceniom:

Spowoduje dodanie switcha `sw1` do sieci	`switch sw1 create`
Usuwa switch `sw1` z sieci	`switch sw1 remove`

Opis języka konfiguracyjnego: router

Router to host posiadający wiele interfejsów sieciowych o różnych adresach IP. Router posiada również tablicę routingu czyli spis miejsc osiągalnych poprzez określony interfejs sieciowy. Router posiada możliwość przekierowania danych z jednego interfejsu do drugiego – dzieje się tak w przypadku, w którym miejsce przeznaczenia wysłanego z hosta pakietu znajduje się w tablicy routingu routera. Oznacza to, że router wie, na który interfejs przekierować pakiet i właśnie to dokonuje.

Proponowana składnia dla dodawania routera

Stworzenie routera	`router NAZWA create`
Usunięcie routera	`router NAZWA remove`
Ustala trasę routingu	`router NAZWA route IP NETMASK NUMER-INTERFEJSU`

Ta część jest jeszcze niezaimplementowana.

Aktywne elementy sieci, które posiadają możliwość nadawania danych mogą posiadać przypisane do siebie zadanie. Zadanie służy do wymuszenia pewnego ustalonego ruchu danych w strukturze sieci i jest zasadniczą właściwością pracy symulatora. Zadanie zostanie wykonane po rozpoczęciu symulacji.

Język konfiguracyjny: pozostałe połączenia

Na koniec pozostała nam najbardziej użyteczna grupa instrukcji, a mianowicie ta specyfikująca poszczególne parametry dotyczące procesu symulacji. Zmiany parametrów możemy dokonać poprzez:
Ustawia zmienną NAZWA na wartość WARTOSC `set NAZWA WARTOSC`

Zaimplementowana funkcjonalność to:
Ustawia ilość kroków symulacji (domyślnie:10) `set simtime ILE-KROKOW`

Notki implementacyjne

Symulator został napisany w języku ANSI C. Wybór na tę technologię padł z powodu łatwej dostępności nagłówków, implementujących struktury pakietów sieci Ethernet oraz najlepszą z dostępnych dróg na zapewnienie modularności (biblioteki dzielone) naszego projektu.

Architektura symulatora w chwili obecnej używa uproszczonych nagłówków Ethernet, IP oraz ICMP. Możliwe jest jednak uzyskanie PEŁNEJ zgodności ze strukturą rzeczywistych pakietów i wykorzystanie replik w symulatorze.

Faktem jest, że symulator rzeczywiście pracuje w kontekście "wysyłania", "odbierania", "analizy" i "przetwarzania" rzeczywistych pakietów. Wiadomość ICMP REQUEST jest faktyczną porcją danych docierającą do interfejsu hosta, którą ten musi przeanalizować.

Plik nagłówkowy `queue.h` został zapożyczony z projektu FreeBSD i służy do wygodnego zarządzania strukturami danych (listy, kolejki) wykorzystanymi w projekcie.

Do gruntownego rozszerzenia!

Zakończenie

Życzymy miłego użytkowania i czekamy na uwagi.

Wojciech A. Koszek
`wkoszek@FreeBSD.czest.pl`
Piotr Matyja
`piotr-matyja@o2.pl`