Maciej TWARDY [2], **Grzegorz SUŁKOWSKI** [2], **Kazimierz WIATR** [1,2]
[1] AKADEMIA GÓRNICZO-HUTNICZA
[2] ACK CYFRONET AGH

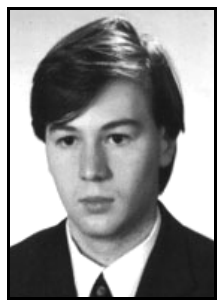
Filtrowanie adresów sieciowych w sprzętowym systemie bezpieczeństwa typu Firewall

Mgr inż. Maciej TWARDY

Ukończył studia na Wydziale Elektrotechniki, Automatyki, Informatyki i Elektroniki Akademii Górniczo-Hutniczej w Krakowie. Od 2005 roku kieruje Działem Archiwizacji i Bezpieczeństwa Danych w Akademickim Centrum Komputerowym CYFRONET AGH. Jego zainteresowania związane są z szeroko pojętą informatyką oraz projektowaniem układów cyfrowych w oparciu o układy reprogramowalne.

e-mail: Maciej.Twardy@cyfronet.pl

Mgr inż. Grzegorz SUŁKOWSKI



Ukończył studia na Wydziale Elektrotechniki, Automatyki, Informatyki i Elektroniki Akademii Górniczo-Hutniczej w Krakowie. Od 2006 roku jest Konstruktorem Systemów Obliczeniowych w dziale Archiwizacji i Bezpieczeństwa Danych w Akademickim Centrum Komputerowym CYFRONET AGH. Jego zainteresowania skupiają się wokół algorytmów obliczeniowych oraz ich realizacji w układach reprogramowalnych.

e-mail: Grzegorz.Sulkowski@cyfronet.pl

Streszczenie

W niniejszym artykule zaprezentowano wyniki praktycznej realizacji sprzętowego klasyfikatora adresów sieciowych opartego o dedykowaną pamięć TCAM (ang. *Ternary Content-Addressable Memory*). Opracowana metoda implementacji pamięci TCAM charakteryzuje się dużą szybkością pracy oraz znacznie efektywniejszym wykorzystaniem zasobów układów FPGA w porównaniu do komercyjnych wersji oferowanych przez firmę Xilinx.

Słowa kluczowe: systemy bezpieczeństwa informatycznego, układy programowalne, języki opisu sprzętu, Ethernet, firewall.

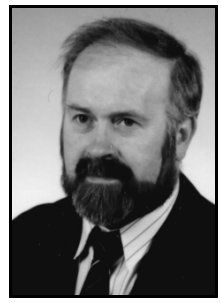
Network address filtering in a hardware Firewall security system

Abstract

The paper presents the results of practical realization of a network address and protocol type classifier based on Ternary Content-Addressable Memory (TCAM). The first section deals with a subject of packet classification. The second one describes the packet classifier internal structure, characterizing in details each of the elements included in the classifier, according to the block diagram of Fig. 1. The address filter architecture (shown in Fig. 2) assumed by the authors is discussed in the third section. The fourth section contains some details concerning the TCAM cells array functionality and implementation method. The last section summarizes the results obtained. The new TCAM architecture based on RAM16X1S storage elements adopted by the authors is much more effective than the commercial solution generated by the Xilinx COREGenerator software. The device resources requirements are over two times lower than the resources required by the COREGenerator version. This significant reduction causes improvements in overall timing characteristics. The estimated maximum operating frequency for the address and protocol type filter is 160 MHz. It means that the module can analyze about 160 million packets per second. The research work is in line with the rapidly developing trend towards using reprogrammable logic for securing data transfer in information technology networks.

Keywords: IT Security Systems, Programmable Logic, Hardware Description Language, Ethernet, Firewall.

Prof. dr hab. inż. Kazimierz WIATR



Studia AGH Kraków (1980), dr nauk technicznych (1987), dr habilitowany (1999) i profesor (2002). Profesor zwyczajny na AGH w Krakowie oraz Dyrektor Akademickiego Centrum Komputerowego CYFRONET AGH. Prowadzone prace badawcze dotyczą systemów wizyjnych, systemów wieloprocesorowych, rekonfigurowanych systemów obliczeniowych i sprzętowych metod akceleracji obliczeń. Jest autorem trzech monografii, w tym najnowsza Akceleracja obliczeń w systemach wizyjnych (2003).

e-mail: wiatr@agh.edu.pl

1. Wstęp

Proces filtracji pakietów w module klasyfikatora sytemu Firewall dokonywany jest na podstawie informacji zawartych w nagłówkach przetwarzanych pakietów. Decyzję o akcji podejmowanej dla poszczególnych pakietów (retransmisja bądź blokowanie) klasyfikator podejmuje analizując zgodność adresów oraz portów źródłowych i docelowych, jak również typu protokołu transmisji, ze wzorcem zapisanym w definicji reguł bezpieczeństwa. Ze względu na specyfikę implementacji klasyfikatora w logice reprogramowalnej FPGA proces weryfikacji danych podzielony jest na dwie części: adresy sieciowe wraz z typem protokołu oraz analizę wartości portów.

Najbardziej popularną metodą klasyfikowania adresów jest wykorzystanie pamięci trójwartościowych TCAM (ang. *Ternary Content-Addressable Memory*). Wynika to ze specyficznych własności tego typu pamięci, a przede wszystkim z ich zdolności do przechowywania informacji o wartościach nieistotnych (ang. *don't care*), oznaczanych w opisach znakiem gwiazdki „*". Taka funkcjonalność idealnie odpowiada potrzebom klasyfikatora adresów sieciowych. Definicje reguł bezpieczeństwa w części dotyczącej adresacji pakietów złożone są bowiem z dwóch elementów: 32-bitowego adresu sieciowego protokołu IP (ang. *Internet Protocol*) oraz 32-bitowej maski podsieci (ang. *Subnetwork Mask*), wyodrębniającej z adresu IP część sieciową oraz część hosta. Pamięć TCAM umożliwia zapisanie tych dwóch wartości dla poszczególnych reguł bezpieczeństwa i bardzo szybkie uzyskanie informacji o trafieniu (w przeciągu jednego cyklu zegarowego).

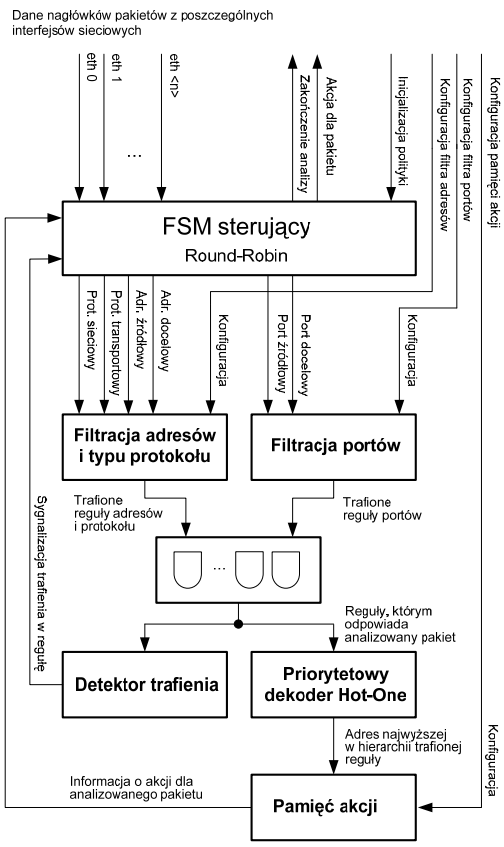
2. Struktura wewnętrzna klasyfikatora pakietów

Poglądowy schemat blokowy modułu klasyfikatora pakietów przedstawiono na rys. 1. Dane niezbędne do oceny zgodności przetwarzanych pakietów z obowiązującym schematem polityki bezpieczeństwa trafiają do klasyfikatora ze specjalnych bloków pamięci ramkowej, szczegółowo opisanych w pozycji [1]. Klasyfikator jest w stanie analizować informacje pochodzące z wielu interfejsów sieciowych, przy czym sumaryczny strumień danych wynikający z ich szybkości pracy nie powinien przekraczać maksymalnej wydajności modułu. Praca wielokanałowa realizowana jest przy wykorzystaniu algorytmu karuzelowego (ang. *Round-Robin*), który cyklicznie sprawdza dostępność nowych deskryptorów bezpieczeństwa na wejściu klasyfikatora. Deskryptor bezpieczeństwa z aktywnego wejścia, udostępniany do bloku filtrów składa się z następujących elementów (pół nagłówka pakietu):

a) typu protokołu sieciowego o długości 16 bitów,
b) typu protokołu transportowego o długości 8 bitów,
c) adresu źródłowego o długości 32 bitów,
d) adresu docelowego o długości 32 bitów,
e) numeru portu źródłowego o długości 16 bitów,
f) numeru portu docelowego o długości 16 bitów.

Pierwsze cztery pola o łącznej długości 88 bitów trafiają do modułu filtrującego adresy oraz typ protokołu. Dwa ostatnie zaś, o łącznej długości 32 bitów, przekierowywane są do modułu filtrującego porty. Na wyjściu każdego z filtrów dostępna jest w formie binarnej niekodowanej informacja o regułach, którym odpowiada aktualnie analizowany pakiet. Ze względu na podział funkcjonalny, spowodowany specyfiką implementacji filtrów w logice reprogramowalnej FPGA, w celu otrzymania ostatecznego zestawu aktywnych reguł, konieczne jest przeprowadzenie iloczynu logicznego wektorów pochodzących z obu modułów filtrujących. Wynik iloczynu jest zamieniany w priorytetowym dekoderze „gorącej jedynki" (ang. *hot one*) na adres binarny najwyższej położonej w hierarchii trafionej reguły. Na jego podstawie z pamięci akcji odczytywana jest informacja o dalszym postępowaniu z analizowanym pakietem. W obecnej implementacji możliwe są dwa scenariusze: odrzucenie bądź akceptacja i w jej efekcie retransmisja pakietu. Równocześnie blok detektora trafienia generuje sygnał o wystąpieniu przynajmniej jednej reguły, której odpowiada analizowany pakiet. Jest on niezbędny dla funkcjonowania głównego automatu sterującego. W wypadku gdyby nie istniała ani jedna odpowiednia reguła, pakiet domyślnie ulega odrzuceniu.

Ostatecznie sygnał o zakończeniu analizy wraz z potwierdzeniem akcji trafiają z modułu klasyfikatora do bloku pamięci ramkowej. Dla pakietów zaakceptowanych rozpoczyna się wówczas procedura przesyłania danych z pamięci do interfejsu nadawczego, zaś w przypadku odrzucenia pakietu, odpowiadająca mu strona pamięci zostaje zwolniona [1].



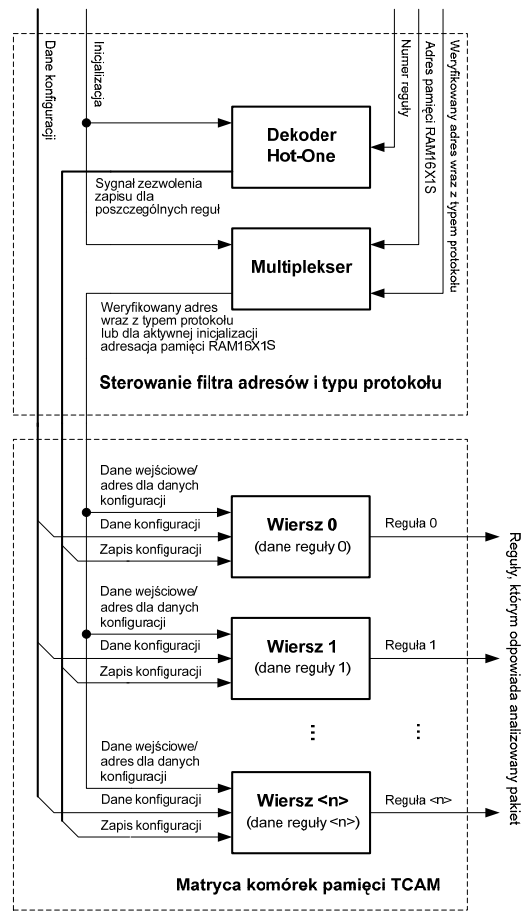
Rys. 1. Schemat blokowy modułu klasyfikatora pakietów
Fig. 1. Block diagram of the packet classifier

3. Moduł filtrujący adresy sieciowe oraz typ protokołu

Jak wspomniano we wstępie, moduł filtrujący adresy sieciowe oraz typ protokołu został zrealizowany w oparciu o pamięci trójwartościowe TCAM. Jego schemat blokowy przedstawiono na rys. 2. Zasadniczym elementem filtra jest matryca komórek pamięci TCAM (opisana szerzej w następnym rozdziale) uzupełniona o niezbędne elementy sterujące.

Część deskryptora bezpieczeństwa o długości 88 bitów, zawierająca informację o adresach sieciowych oraz typie protokołu, zostaje podana na wejście multipleksera bloku sterującego. Jeżeli sygnał inicjalizacji polityki bezpieczeństwa jest w niskim stanie logicznym, matryca pamięci TCAM pracuje w trybie odczytu. Na jej wejście poprzez multiplekser trafiają dane deskryptora. Z kolei na wyjściu pamięci pojawia się informacja o regułach, którym odpowiada weryfikowany pakiet. Czas trwania procesu odczytu jest typowa dla TCAM i wynosi jeden cykl zegara.

Jeżeli sygnał inicjalizacji jest w stanie wysokim, pamięć przechodzi do trybu zapisu. Wówczas na wejścia matrycy komórek podawane są z multipleksera adresy inkrementowane w zakresie od 0 do 15, służące zapisaniu wewnętrznych wartości poszczególnych komórek pamięci TCAM danymi konfiguracji odpowiadającymi definicjom poszczególnych reguł bezpieczeństwa. Na podstawie numeru reguły podawanego na wejście bloku sterującego, dekoder „gorącej jedynki" generuje sygnał zezwalający na zapis odpowiedniego wiersza. Proces zapisu definicji pojedynczej reguły zajmuje 16 cykli zegarowych.

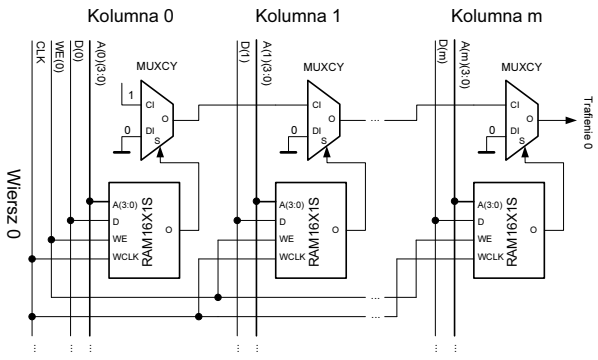


Rys. 2. Schemat blokowy filtra adresów i typu protokołu
Fig. 2. Block diagram of the address and protocol type filter

4. Matryca komórek pamięci TCAM

Ponieważ wynikowa informacja o trafionych regułach generowana jest na podstawie iloczynu wektorów pochodzących z dwóch niezależnych bloków filtrujących, niezbędne jest, aby każdy z nich dostarczał informację wyjściową w formie binarnej niekodowanej (każdej regule odpowiada dedykowane wyjście sygnałowe). Ten wymóg funkcjonalny narzuca duże ograniczenia odnośnie sposobu realizacji matrycy komórek TCAM, uniemożliwiając wykorzystanie algorytmów grupowania filtrów [2], bądź złożonych kaskad bloków LUT [3].

Jako referencyjną metodę implementacji pamięci TCAM na początku realizacji projektu wybrano rozwiązanie wykorzystujące komercyjne oprogramowanie COREGenerator firmy Xilinx. Jednak ze względu na brak możliwości zmiany wewnętrznej struktury otrzymanego w ten sposób modułu pamięci autorzy opracowali alternatywną koncepcję realizacji TCAM, pozwalającą zarówno na optymalizację wydajnościową, jak również minimalizację ilości niezbędnych zasobów sprzętowych. Opiera się ona na wykorzystaniu generatorów funkcji z konfigurowalnych bloków logicznych CLB (ang. *Configurable Logic Blocks*) układów Virtex 2 Pro firmy Xilinx, pracujących jako pamięci RAM16X1S (jednowejściowa pamięć RAM przechowująca 16 wartości jednobitowych). Każdy z wierszy TCAM zbudowany jest z kaskady elementarnych pamięci RAM16X1S włączonych do łańcucha przeniesień (ang. *Carry Chain*) zbudowanego przy pomocy multiplekserów MUXCY, wchodzących w skład bloków CLB. Strukturę pojedynczego wiersza opisywanej pamięci TCAM przedstawiono na rys. 3.



Rys. 3. Struktura pojedynczego wiersza pamięci TCAM
Fig. 3. Structure of one TCAM memory row

Do poszczególnych elementów RAM16X1S zapisane zostają odpowiednio przygotowane definicje reguł bezpieczeństwa. Przykładowa konfiguracja wewnętrzna pojedynczej czterowejściowej komórki pamięci TCAM wykrywającej adres „10**", gdzie „*" oznacza wartość nieistotną, została przedstawiona w tabeli nr 1.

Tab. 1. Przykładowa konfiguracja pojedynczej pamięci RAM16X1S
Tab. 1. Exemplary single RAM16X1S internal configuration

Adres	Wartość	Adres	Wartość
0000	0	**1000**	**1**
0001	0	**1001**	**1**
0010	0	**1010**	**1**
0011	0	**1011**	**1**
0100	0	1100	0
0101	0	1101	0
0110	0	1110	0
0111	0	1111	0

Dla wszystkich kombinacji danych wejściowych rozpoczynających się wartością „10" pamięć zwraca jedynkę logiczną. Chcąc uzyskać większą szerokość szyny wejściowej, konieczne jest połączenie niezbędnej ilości elementów RAM16X1S (wymagana szerokość wejścia podzielona przez 4) w łańcuch przeniesień. Każde z wyjść pojedynczych pamięci steruje multiplekserem MUXCY. Jeżeli pamięć zwraca wartość '1' jest ona propagowana przez MUXCY do kolejnej pozycji w łańcuchu, pod warunkiem, że na wyjściu poprzedzającego multipleksera również występowała jedynka logiczna. Sygnał trafienia w regułę jest aktywny jedynie wówczas, kiedy wyjścia wszystkich pamięci RAM16X1S w łańcuchu przeniesień są w stanie wysokim.

Ponieważ szerokość szyny danych modułu filtrującego adresy i typ protokołu, jak wspomniano wcześniej, wynosi 88 bitów, pojedynczy wiersz pamięci TCAM musi zawierać 22 elementy RAM16X1S. Konfiguracja wewnętrzna filtra jest przygotowywana na podstawie tabeli zawierającej reguły bezpieczeństwa za pomocą specjalnie opracowanego oprogramowania zarządzającego. Każda z reguł jest konwertowana do odpowiedniej postaci i przesyłana do wewnętrznej pamięci reguł Firewall'a. Stamtąd, w momencie zainicjowania ładowania polityki bezpieczeństwa, dane są przepisywane do odpowiednich komórek pamięci TCAM.

5. Wyniki i wnioski

Opisane rozwiązanie zostało zaimplementowane i przetestowane praktycznie przy pomocy płyty uruchomieniowej Digilent XUP z układem Virtex II Pro XC2VP30. W tabeli nr 2 przedstawiono porównanie wykorzystania zasobów wykorzystywanych przez różne warianty realizacji pamięci TCAM o szerokości 88 bitów zawierającej 32 wiersze danych.

Tab. 2. Wykorzystanie zasobów sprzętowych układu Virtex II Pro XC2VP30 oraz charakterystyka wydajnościowa dla różnych wariantów implementacji pamięci TCAM
Tab. 2. Virtex II Pro XC2VP30 device utilization summary and performance characteristic for different types of TCAM implementation

Typ zasobów Parametry czasowe	Pamięć TCAM oparta o RAM16X1S	Xilinx CORE Generator TCAM
Liczba bloków Slice:	810 (5%)	1795 (13%)
Min. okres zegara:	3,880 ns	7,668 ns
Maks. częstotliwość:	257,732 MHz	130,416 MHz
Min. czas ustalenia syg. wej. przed zmianą zegara:	6,121 ns	7,977 ns
Max. czas ustalenia syg. wyj. po zmianie zegara:	3,293 ns	3,461 ns

Opracowana metoda implementacja pamięci TCAM bazująca na elementach RAM16X1S jest ponad dwukrotnie bardziej efektywna od strony zapotrzebowania na zasoby sprzętowe w porównaniu do wersji modułu pamięci z Xilinx COREGeneratora. Współczynnik liczby bloków Slice na pojedynczy wiersz TCAM wynosi:
a) 25,3 dla wersji z pamięciami RAM16X1S,
b) 56,1 dla wersji komercyjnej.

Dodatkową korzyścią płynąca ze zredukowania niezbędnych zasobów logiki reprogramowalnej jest uzyskanie lepszych parametrów czasowych. W związku ze sposobem funkcjonowania filtra o ostatecznej wydajności przetwarzania danych decyduje minimalny czas ustalenia poziomu sygnału wejściowego przed zmianą zbocza zegara. Na tej podstawie teoretycznie wyliczona maksymalna częstotliwość pracy modułu filtrującego wynosi około:
a) 163 MHz dla wersji TCAM z pamięciami RAM16X1S,
b) 125 MHz dla wersji komercyjnego modułu TCAM.

Ponieważ proces odczytu pamięci TCAM zajmuje jeden cykl zegara, maksymalna przepustowość opracowanego filtra adresów i typu protokołu to około 160 milionów pakietów na sekundę. W sieci 10Gb Ethernet maksymalna ilość ramek na sekundę wynosi 14 880 952,38 fps (ang. *frame per second*), stąd też moduł filtrujący bez problemu i ze znacznym zapasem wydajności może znaleźć zastosowanie do analizy ruchu sieciowego we współczesnych sieciach teleinformatycznych o dużych przepustowościach.

Praca naukowa finansowana ze środków na naukę w latach 2006-2008 jako projekt badawczy.

6. Literatura

[1] G. Sułkowski, M. Twardy, K. Wiatr: Wielościeżkowe równoległe przetwarzanie danych w sprzętowym systemie bezpieczeństwa klasy Firewall, Konferencja KNWS'08, Miesięcznik Przegląd Telekomunikacyjny nr 6, Wydawnictwo SIGMA NOT, Warszawa 2008, s. 726-728.
[2] E. Spitznagel, D. Taylor, J. Turner: Packet Classification Using Extended TCAM, Proceedings of the 11th IEEE International Conference on Network Protocols, p.120, November 04-07, 2003.
[3] H. Qin, T. Sasao, J. T. Butler: Implementation of LPM Address Generators on FPGAs, Architectures and Applications, Second International Workshop, ARC 2006.

Artykuł recenzowany