Grzegorz SUŁKOWSKI* , Maciej TWARDY*, Kazimierz WIATR*,**

Wielościeżkowe równoległe przetwarzanie danych w sprzętowym systemie bezpieczeństwa klasy *Firewall*

W pierwszym etapie prac, realizowanych w ramach projektu badawczego dotyczącego sprzętowej implementacji systemu bezpieczeństwa klasy *Firewall*, autorzy skoncentrowali się na przygotowaniu odpowiedniej platformy testowo-uruchomieniowej oraz opracowaniu elementów warstwy komunikacyjnej, umożliwiającej eksploatację systemu w infrastrukturze sieci Ethernet [4][5]. Aby w pełni wykorzystać potencjał układów reprogramowalnych **FPGA** [6], było konieczne odmienne od klasycznej implementacji *software*'owej spojrzenie na architekturę wewnętrzną systemu bezpieczeństwa. Wprawdzie w pewnych elementach nie sposób uniknąć *stricte* sekwencyjnego przetwarzania danych, co wynika wprost z charakteru pracy *Firewalla*, jednak było możliwe takie zoptymalizowanie jego sprzętowej wersji, aby sekwencyjność ta była niwelowana przez mechanizmy potokowości czy też zrównoleglenie przetwarzania danych. W całościowym ujęciu takie podejście zapewnia skompensowanie spadków wydajności w przepływie danych, przy jednoczesnym zachowaniu bardzo wysokiego poziomu bezpieczeństwa systemu z racji braku jakichkolwiek elementów *software*'owych. Prowadzone prace badawcze wpisują się w dynamicznie rozwijający się nurt wykorzystywania logiki reprogramowalnej w obszarach zabezpieczania transmisji danych w sieciach informatycznych [2][3].

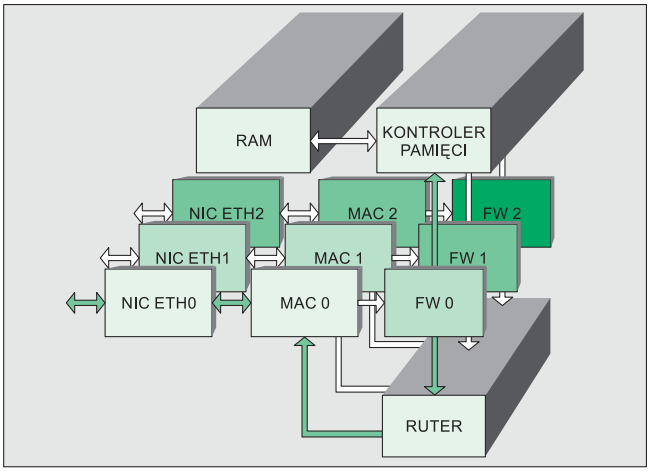
W dalszej części artykułu szerzej zostanie zaprezentowana przyjęta przez autorów koncepcja architektury wewnętrznej *Firewalla.*

ARCHITEKTURA *FIREWALLA* – CHARAKTERYSTYKA OGÓLNA

Podstawowymi założeniami, przyjętymi przy opracowywaniu koncepcji architektury sprzętowego *Firewalla*, było przede wszystkim zachowanie dużego bezpieczeństwa systemu oraz zmaksymalizowanie jego wydajności. O ile spełnienie pierwszego z założeń jest już osiągalne przez przeniesienie funkcjonalności *Firewalla* do logiki reprogramowalnej, wyeliminowaniem tym samym oprogramowania mającego luki wewnętrzne, to w przypadku zwiększenia wydajności konieczne jest drobiazgowe przeanalizowanie słabych stron konwencjonalnych rozwiązań po to, aby móc je wyeliminować dodatkowymi mechanizmami, jakie zyskujemy po przejściu w implementację sprzętową. Powszechnie eksploatowane programowe systemy bezpieczeństwa wykorzystują w swym działaniu procesory ogólnego przeznaczenia, które stanowią główny punkt ograniczający efektywność. Procesor w takim przypadku jest wykorzystywany nie tylko do realizacji algorytmów analizy pakietów, ale również przez system operacyjny *Firewalla*, aplikacje zarządzające, obsługę komunikacji sieciowej itp. Tak znaczne obciążenie procesora zadaniami pobocznymi w stosunku do zasadniczego celu działania urządzenia skutkuje jego przeciążeniami i ograniczeniem maksymalnej przepływności transmisji. Co więcej, można obserwować eskalację tego problemu przy dodawaniu kolejnych interfejsów sieciowych, generujących dodatkowe obciążenie związane z rutingiem pakietów.

Taka sytuacja sprowokowała autorów do poszukiwań sprzętowych rozwiązań eliminujących wymienione problemy. Opracowana koncepcja architektury *Firewalla* opiera się na założeniu tworzenia dedykowanych kanałów komunikacyjnych, tak jak w przypadku mechanizmu mikrosegmentacji, wykorzystywanego praktycznie we wszystkich obecnych na rynku przełącznikach ethernetowych. Każdy taki kanał zawiera w sobie pełny tor przetwarzania i analizy bezpieczeństwa danych. Poszczególne moduły *Firewalli* dla wszystkich istniejących ścieżek komunikacyjnych opierają się w takim przypadku na jednym, globalnym zestawie reguł bezpieczeństwa. W momencie ich modyfikacji wszelkie zmiany są propagowane do każdego modułu weryfikującego przetwarzane dane. Jeżeli w systemie bezpieczeństwa zostaną zainstalowane więcej niż dwa interfejsy sieciowe (*Network Interface Card*), konieczne jest zaimplementowanie dodatkowego bloku, realizującego ruting pakietów. Optymalnym rozwiązaniem byłby w tej sytuacji oczywiście ruter w pełni sprzętowy, natomiast zagadnienie to wykracza poza tematykę prowadzonych prac, mogąc stać się z racji obszerności tematem oddzielnych badań. W obecnym stadium realizacji projektu autorzy zakładają minimalną wersję konfiguracji systemu, zawierającą dwa interfejsy sieciowe, która zapewnia jednak zweryfikowanie funkcjonalności przyjętej koncepcji architektury sprzętowego *Firewalla*, nie blokując zarazem możliwości jego dalszej rozbudowy o moduł rutujący.

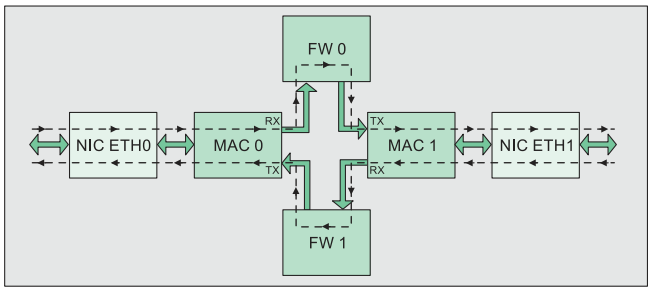
Ogólny schemat koncepcji architektury *Firewalla* z równoległym przetwarzaniem wielościeżkowym przedstawiono na rys. 1. Poszczególne karty interfejsów sieciowych są obsługiwane przez dedykowane sprzętowe bloki **MAC** (*Media Access Control*), realizujące funkcjonalności wymagane przez standard opisujący sieć



Rys. 1. Schemat architektury sprzętowego *Firewalla*. Oznaczenia: RAM – blok pamieci, MAC – blok dostępu, NIC – interfejs sieciowy, FW – moduł weryfikacji reguł bezpieczeństwa

* ACK Cyfronet AGH, Kraków,
e-mail: Grzegorz.Sulkowski@cyfronet.pl,
Maciej.Twardy@cyfronet.pl,
** Akademia Górniczo-Hutnicza, Kraków,
e-mail: wiatr@agh.edu.pl

Ethernet 802.3. Dane z odbieranych ramek trafiają następnie do dedykowanych modułów weryfikacji reguł bezpieczeństwa (bloki **FW**). Ramki, które zostały zaakceptowane, są przekazywane w kolejności do modułu rutera, przełączającego strumień danych do toru transmisyjnego odpowiedniej karty sieciowej. W przypadku dwóch kart sieciowych moduł ten nie występuje.



■ **Rys. 2. Przepływ danych w *Firewallu* z dwoma interfejsami sieciowymi. Oznaczenia jak na rys. 1**

Bloki FW poszczególnych torów komunikacyjnych pobierają reguły bezpieczeństwa z głównej pamięci reguł przez specjalny kontroler pamięci. Taka komunikacja występuje tylko w momencie pierwszego uruchomienia *Firewalla* lub po każdorazowej zmianie polityki bezpieczeństwa. Zmiany te nie występują jednak w praktyce na tyle często, aby opisana koncepcja dystrybucji reguł rzutowała na spadek wydajności przetwarzania danych.

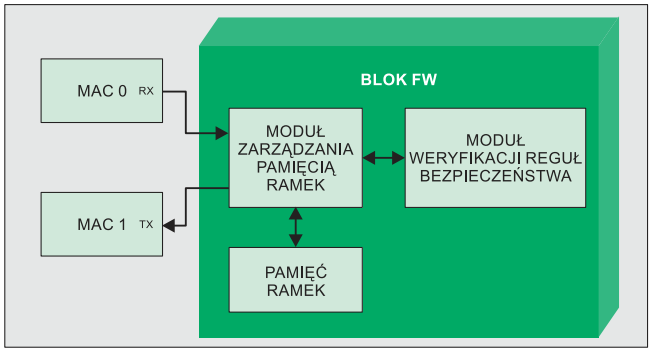
Przepływ danych w systemie z dwoma interfejsami sieciowymi przedstawiono na rys. 2.

BUDOWA BLOKU FW

Każdy z bloków FW składa się z trzech zasadniczych elementów:

- modułu weryfikacji reguł bezpieczeństwa,
- pamięci ramek,
- modułu zarządzania pamięcią ramek.

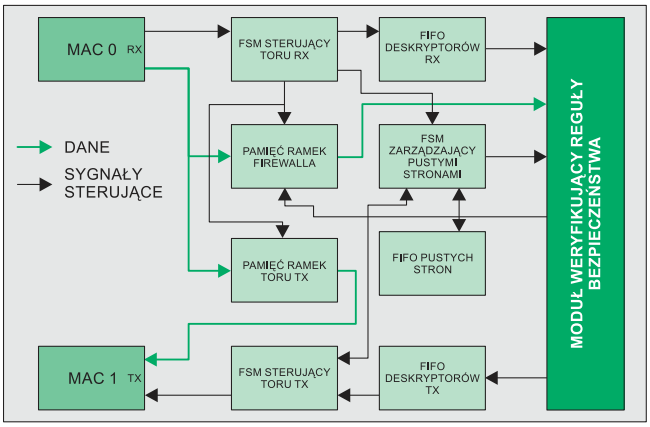
Schemat wewnętrzny bloku FW przedstawiono na rys. 3. Pamięć ramkowa, zbudowana z wykorzystaniem wewnętrznych zasobów pamięci blokowej *Block SelectRAM+* układu *Virtex II Pro*



■ **Rys. 3. Schemat wewnętrzny bloku FW**

firmy Xilinx, jest podzielona na strony. Każdej stronie odpowiada pojedynczy blok pamięci **BRAM** o rozmiarze 2304 bajtów. Rozmiar taki umożliwia zapisanie danych do pojedynczej strony ramki ethernetowej o maksymalnej dopuszczonej standardem długości, wynoszącej 1518 bajtów [1]. Liczba stron pamięci ramek dla danego bloku FW jest parametryzowana i zależy od typu zastosowanego układu FPGA. Opracowana koncepcja organizacji pamięci ramkowej oraz modułu zarządzającego zwiększa wydajność pracy *Firewalla* przez umożliwienie transmisji ramek zweryfikowanych pod względem zgodności z regułami bezpieczeństwa, niezależnie od analizowania bieżąco napływających danych. Taki stan osiągnięto dzięki wykorzystaniu dwóch niezależnych pamięci ramkowych: jednej dla modułu analizującego ramki, a drugiej dla toru transmisyjnego **TX** właściwej karty sieciowej. Dane z każdej odebranej ramki trafiają z modułu MAC – kontrolera sieci Ethernet – do obydwu pamięci jednocześnie, pod tym samym adresem strony. W tym samym momencie utworzony zostaje deskryptor ramki, zawierający informację o jej długości i numerze strony, pod jaką została ona zapisana. Deskryptor jest zapisywany do kolejki **FIFO RX**, z której następnie zostaje pobierany przez moduł analizy bezpieczeństwa. Silnik *Firewalla* ma więc niezbędny zestaw informacji, aby rozpocząć pobieranie nagłówka ramki ethernetowej, na podstawie którego dokona weryfikacji jej zgodności ze schematem reguł bezpieczeństwa. Taka analiza, w zależności od liczby reguł wprowadzonych do *Firewalla* oraz zastosowanych algorytmów wyszukujących, może trwać przez czas przekraczający odbiór ramki ethernetowej o minimalnej długości (60 bajtów). W najbardziej niekorzystnym przypadku, kiedy w trakcie weryfikacji bezpieczeństwa jest odbierany ciąg krótkich następujących po sobie ramek, pamięć może ulec przepełnieniu (zabraknie wolnych stron), co może skutkować utratą części przesyłanych danych. Aby temu zapobiec, można w trakcie analizowania ramek transmitować wszystkie te, które zostały już zaakceptowane przez moduł weryfikujący, zwalniając tym samym strony w pamięci ramek.

Po poprawnym zatwierdzeniu ramki moduł weryfikujący generuje deskryptor, który zostaje zapisany do kolejki **FIFO TX**. Tor nadawczy kontrolera MAC sięga do niej, pobierając dane o kolejnych ramkach, wymagających przesłania, niezależnie od zadań wykonywanych przez moduł weryfikujący. Zakończenie transmisji danej ramki przez tor TX kończy się zwolnieniem odpowiedniej strony w pa-



■ **Rys. 4. Schemat blokowy pamięci ramek**

mięci ramkowej i próbą pobrania kolejnego deskryptora z FIFO TX, o ile istnieją jakiekolwiek dane do przesłania. Zwalnianie stron następuje nie tylko po zakończeniu transmisji ramki przez tor TX kontrolera MAC, lecz również w momencie odrzucenia danej ramki w module weryfikującym reguły bezpieczeństwa. Lista pustych stron pamięci ramkowej jest przechowywana w specjalnym FIFO, zarządzanym przez dedykowany automat stanów (*Finite State Machine*).

Schemat blokowy pamięci ramek przedstawiono na rys. 4.

* * *

Testy praktyczne przedstawionej architektury sprzętowego *Firewalla* przeprowadzono przy wykorzystaniu płyty uruchomieniowej **Digilent XUP** z układem *Virtex II Pro XC2VP30*. Konfigurację sprzętową uzupełniały dwie karty sieciowe oparte na układzie DP83865DVH, pracujące w trybie *Fast Ethernet*. Pamięć ramek dla obydwu torów transmisyjnych miała 16 stron. Ponieważ moduł weryfikujący reguły bezpieczeństwa jest w trakcie opracowywania, było konieczne jego zastąpienie specjalnym blokiem symu-

lującym analizę pól ramek ethernetowych oraz przeszukiwanie tablicy reguł. W praktyce sprowadziło się to do pobierania z pamięci ramkowej danych nagłówka, następnie wprowadzenia opóźnienia symulującego jego weryfikację, a na koniec zatwierdzaniua każdej z odebranych ramek.

Tabela 1. Wykorzystanie zasobów sprzętowych

Zajętość zasobów układowych *Virtex II Pro* XC2VP30-7		
	Liczba wykorzystanych elementów	**Utylizacja procentowa**
Liczba elementów Slices	222 z dostępnych 13 696	1%
Liczba elementów z przerzutnikami	260 z dostępnych 27 392	1%
Liczba czterowejściowych komórek LUT	409 z dostępnych 27 392	1%
Liczba bloków pamięci BRAM	70 z dostępnych 136	51%

Wykorzystanie zasobów układu FPGA dla dwóch bloków FW (z pominięciem kontrolerów sieciowych MAC) przedstawiono w tabeli 1. Orientacyjna maksymalna szybkość pracy pamięci ramkowej wynosi około 350 MHz.

Kolejnym istotnym etapem prowadzonych prac będzie opracowanie modułu weryfikującego reguły bezpieczeństwa.

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

LITERATURA

[1] 802.3 IEEE *Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks*, 2002
[2] Loinig J., Wolkerstorfer J., Szekely A. *Packet Filtering in Gigabit Networks Using FPGAs,* Austrochip 2007 – Proceedings of the 15th Austrian Workshop on Microelectronics, 2007
[3] Moscola J., Lockwood J., Loui R. P., Pachos M., *Implementation of a Content-Scanning Module for an Internet Firewall*, Proceedings of IEEE Symposium on Field-Programmable Custom Computing Machines, 2003
[4] Sułkowski G., Twardy M., Wiatr K.: *Implementacja systemu bezpieczeństwa typu Firewall dla potrzeb sieci Ethernet w oparciu o układy reprogramowalne FPGA*, Konferencja KNWS'07, Kwartalnik Pomiary, Automatyka i Kontrola, Warszawa, nr 5, 2007
[5] Sułkowski G., Twardy M., Wiatr K.: *Implementacja standardu sieci Ethernet IEEE 802.3 w układach FPGA na potrzeby systemu bezpieczeństwa typu Firewall*, Konferencja: Reprogramowalne Układy Cyfrowe 2007, Kwartalnik Pomiary, Automatyka i Kontrola nr 7, 2007
[6] Wiatr K.: *Akceleracja obliczeń w systemach wizyjnych*, Wydawnictwa Naukowo-Techniczne, Warszawa 2003

Artykuł recenzowany

Jacek TKACZ, Marian ADAMSKI**

Wykorzystanie komputerowego wnioskowania w projektowaniu kombinacyjnych układów sterowania

Dotychczasowe badania, prowadzone nad komputerowym wnioskowaniem symbolicznym [1, 12, 13], uwidoczniły jego przydatność do rozwiązywania różnorodnych problemów kombinatorycznych metodami formalnymi. Dokładne algorytmy analizy funkcji logicznych silnie nieokreślonych, zwłaszcza przedstawionych w postaci symbolicznej, opisane w literaturze przedmiotu są bardzo pracochłonne [6, 7]. W artykule przedstawiono zastosowanie wnioskowania symbolicznego Gentzena do uzyskiwania już uproszczonego układu kombinacyjnego na podstawie przykładu o charakterze akademickim, zaczerpniętego z literatury. Jądrem systemu wnioskującego jest system dowodzenia twierdzeń w zdaniowym rachunku sekwentów Gentzena [4].

Należy tu zaznaczyć, że celem badań nie jest poszukiwanie narzędzia, znacznie sprawniejszego od minimalizatorów zawartych w systemach profesjonalnych, gdzie poszukuje się rozwiązań suboptymalnych, wprowadzając wyrafinowaną heurystykę. Przedstawione w pracy oprogramowanie ma charakter uniwersalny, daje rozwiązania dokładne, nawet bez interakcji z projektantem, a na dodatek dokumentacja (protokół) wnioskowania jest formalnym dowodem poprawności analizy lub syntezy układu kombinacyjnego. Nie oznacza to jednak, że w trakcie badań eksperymentalnych projektant nie może wprowadzić elementów heurystyki, dzieląc na przykład skomplikowane zadania na podzadania [7]. Największą zaletą proponowanej metody jest jej elastyczność, nieoceniona zwłaszcza w badaniach naukowych.

*** Uniwersytet Zielonogórski, Instytut Informatyki I Elektroniki, e-mail: J.Tkacz@iie.uz.zgora.pl, M.Adamski@iie.uz.zgora.pl**

SYSTEM WNIOSKOWANIA GENTZENA

Rozpatrywane są sekwenty w logice zdań. Sekwentem nazywa się uporządkowaną parę (Γ,Δ) skończonych zbiorów formuł $\Gamma = \{A1, A2, \ldots, Am\}$, $\Delta = \{B1, B2, \ldots, Bn\}$. W tradycyjnym ujęciu zbiory te są uporządkowane jako ustalone ciągi formuł. Zamiast pisać (Γ, Δ) stosuje się notację $\Gamma \mid\!- \Delta$. Intuicyjnie definiując sekwent A1, A2, …, Am |-B1, B2, …, Bn podaje się, że jest on spełniony dla wartościowań v wtedy i tylko wtedy, gdy dla tych samych wartościowań formuła (A1*A2*…*Am) –> (B1+B2+…+Bn) jest spełniona. W przyjętej notacji symbol –> oznacza implikację, symbol + dysjunkcję (alternatywę niewyłączającą), natomiast symbol * jest symbolem koniunkcji.

Wykorzystując figury wnioskowania (zwane również regułami wnioskowania) eliminuje się stopniowo wszystkie spójniki logiczne występujące w formułach A i B, tworząc drzewo dowodu. Przykładowymi figurami wnioskowania są eliminacja spójnika koniunkcji po prawej stronie sekwentu i eliminacja spójnika dysjunkcji po lewej stronie sekwentu (1).

$$\frac{\Lambda/\text{-}\Theta\Phi^*\Psi\Gamma}{\Lambda/\text{-}\Theta\Phi,\Gamma} \quad \frac{\Lambda/\text{-}\Theta,\Psi\Gamma}{\Theta,\Phi+\Psi,\Gamma/\text{-}\Pi} \quad \frac{\Theta,\Phi+\Psi,\Gamma/\text{-}\Pi}{\Theta,\Phi,\Gamma/\text{-}\Pi} \quad \frac{\Theta,\Psi,\Gamma/\text{-}\Pi}{} \tag{1}$$

Intuicyjnie jasny jest sposób eliminacji spójnika negacji '/', polegający na przeniesieniu negowanej formuły na przeciwną stronę sekwentu (2).

$$\frac{\Theta,\Phi,/\Psi,\Gamma/\text{-}\Lambda}{\Theta,\Phi,\Gamma/\text{-}\Lambda,\Psi} \quad \frac{\Lambda/\text{-}\Theta,\Phi,/\Psi,\Gamma}{\Lambda,\Psi/\text{-}\Theta,\Phi,\Gamma} \tag{2}$$