논문 2009-46CI-1-9

콘텐츠 보안 시스템용 트래픽 패턴 매칭 하드웨어

(A Traffic Pattern Matching Hardware for a Contents Security System)

최 영*, 홍 은 경*, 김 태 완**, 백 승 태**, 최 일 훈**, 오 형 철***

(Young Choi, Eun-Kyung Hong, Tae-Wan Kim, Seung-Tae Paek, Il-Hoon Choi, and Hyeong-Cheol Oh)

요 약

본 논문에서는 고성능 네트워크 응용에서 사용하기 위한 트래픽 패턴 매칭 하드웨어를 제안한다. 제안하는 트래픽 패턴 매칭 하드웨어는 고속 망에서 다양한 종류의 정보 유출이나 침입을 차단하기 위한 콘텐츠 보안 시스템에서 사용할 목적으로 설계되었다. 제안하는 하드웨어는 헤더 검색부와 스트링 패턴 매칭부로 구성되었다. 헤더 검색부의 하드웨어 구현에는, 흔히 TCAM(Ternary CAM) 구현이 사용되지만 하드웨어나 메모리 비용과 전력 소모 면에서 비효율적이므로, 본 논문에서는 비교기 배열과 HiCuts 트리에 기반을 둔 구현 기법을 채택하고 이를 수정하여 적용하였다. Xilinx FPGA XC4VSX55을 사용한 구현에서, 제안된 설계는 TCAM 구현에 비하여 FPGA 슬라이스 사용을 약 26%까지 그리고 블록 RAM의 사용을 약 58%까지 절약할 수 있었다. 스트링 패턴 매칭부의 설계에서는 하드웨어 면에서 효율적이며, 충돌 발생률을 감소시킬 수 있도록 구성을 바꿔 전력 소모를 감소시킬 수 있는 셀룰러 오토마타형 해싱 모듈을 설계하여 사용하였다.

Abstract

This paper presents a traffic pattern matching hardware that can be used in high performance network applications. The presented hardware is designed for a contents security system which is to block various kinds of information drain or intrusion activities. The hardware consists of two parts: the header lookup and string pattern matching parts. For implementing the header lookup part in hardware, the TCAMs(ternary CAMs) are popularly used. Since the TCAM approach is inefficient in terms of the hardware and memory costs and the power consumption, however, we adopt and modify an alternative approach based on the comparator arrays and the HiCuts tree. Our implementation results, using Xilinx FPGA XC4VSX55, show that our design can reduce the usage of the FPGA slices by about 26%, and the Block RAM by about 58%. In the design of string pattern matching part, we design and use a hashing module based on cellular automata, which is hardware efficient and consumes less power by adaptively changing its configuration to reduce the collision rates.

Keywords: contents security system, header lookup, string pattern matching, HiCuts tree, hashing

* 학생회원, 고려대학교 전자정보공학과
(Dept. of Elec. & Info. Eng., Graduate School, Korea Univ.)
** 정회원, (주) 소만사
(Somansa Co., Ltd)
*** 정회원, 고려대학교(조치원) 전자및정보공학과
(Dept, of Elec. & Info. Eng., Korea Univ. at ChoChiWon)
※ 본 논문은 정보통신연구진흥원과 반도체설계교육센터(IDEC)의 지원을 받아 수행되었습니다.
접수일자: 2008년12월10일, 수정완료일: 2009년1월12일

I. 서 론

최근 인터넷의 상용화 수준과 망의 속도가 급증함에 따라, 고속 망을 통한 정보의 유출을 방지하고 침입을 차단하기 위한 콘텐츠 보안 시스템(Contents Security System)의 필요성이 증대되고 있다. 현재 콘텐츠 보안 시스템은 주로 소프트웨어 상에서 구현되고 있지만, OC-192나 OC-768에 이르는 고속망에서는 하드웨어 지

원이 불가피하다[1~3]. 본 논문에서는 김병구 등[2]에서 제안된 침입 탐지 시스템을 응용하여 콘텐츠 보안 시스템을 구현하는 과정에서 설계한 트래픽 매칭 하드웨어를 소개한다. 본 논문에서 소개하는 하드웨어가 사용될 콘텐츠 보안 시스템 (이하 "콘텐츠 보안 시스템"이라고 함)에서는, 기존의 침입 탐지 시스템에 비해 정보의 유출을 방지하는 기능이 강조되어, 사용자가 보다 다양하고 많은 규칙을 정의할 것을 기대한다. 이에 따라 본 논문에서는 트래픽 패턴 매칭 하드웨어의 구현 비용 및 에너지 소모를 줄이고, 보다 많은 규칙을 효율적으로 처리할 수 있도록 함에 설계의 초점을 두었다.

콘텐츠 보안 시스템은 패킷 분석 부시스템(Subsystem)과 콘텐츠 분석 부시스템 및 시스템 관리 부시스템으로 구성된다. 이와 같이 3개의 부시스템으로 구현되는 전체 시스템은 윈도우즈 OS기반의 소프트웨어 부분과 FPGA로 구현되는 하드웨어 부분으로 구현된다. 하드웨어 부분이 FPGA로 구현되는 것은 콘텐츠 보안 시스템의 특성상, 설치되어 사용되는 중에 보안 규칙 등의 설정을 바꾸기 용이하게 하기 위함이다.

콘텐츠 보안 시스템을 구성하는 3개의 부시스템 중에서 패킷 분석 부시스템에 포함되는 트래픽 패턴 매칭부는 패킷에 적용할 분류 규칙("통과", "차단 또는 분석")을 정한다. 이 트래픽 패턴 매칭부는, 패킷의 헤더를 살펴보고 지정된 규칙에 저촉되는 패킷을 찾는 헤더 검색(Header Lookup)부와, 패킷에 포함된 페이로드(Payload)에서 지정된 시그니처(Signature)를 찾아 그에 대응하는 분류 규칙을 정하는 스트링 패턴 매칭(String Pattern Matching)부로 구성되는데 하드웨어 비용이나 동작 속도 면에서 가장 설계가 중요한 부분 중의 하나이다. 본 논문에서는 이 트래픽 패턴 매칭부의 효율적인 설계를 제안한다.

헤더 검색의 구현에는 다양한 기법이 제안되어 있는데, 이 중 TCAM(Ternary Content Addressable Memory)을 사용하는 기법이 설계가 간단하고 처리 속도가 빨라 하드웨어 구현에서 빈번히 사용된다. 그런데 TCAM 구현은 하드웨어 비용이 많이 들고 전력 소모가 크며 메모리의 낭비가 크다는 단점을 갖는다[3~4]. 본 논문에서는 TCAM의 단점을 극복하기 위하여 Kennedy 등[3]이 제안한 기법을 보다 많은 필드를 처리하여야 하는 콘텐츠 보안 시스템에 맞추어 수정하여 적용한 설계를 제안하고자 한다.

스트링 패턴 매칭에는 김병구 등[2]에서 제안된 설계를 적용하면서, 본 논문에서는 스트링 패턴 매칭에서 집중적으로 사용되는 해싱 모듈의 효율적인 설계를 제안하고자 한다.

본 논문의 구성은 다음과 같다. 제II장에서는 콘텐츠 보안 시스템의 전반적인 시스템 구성에 대하여 소개하고, 제III장에서 트래픽 패턴 매칭부의 설계를 제안하며, 제IV장에서 결론을 맺는다.

II. 콘텐츠 보안 시스템

콘텐츠 보안 시스템은 네트워크를 통한 정보의 유출을 방지하고 침입을 차단하기 위한 시스템으로서 윈도우즈 OS기반의 소프트웨어 부분과 FPGA기반의 하드웨어 부분으로 구성되어 있다. 본 장에서는 본 콘텐츠 보안 시스템을 기능 측면에서 아래와 같이 3개의 부시스템으로 나누어 설명한다.

- 패킷 분석 부시스템
- 콘텐츠 분석 부시스템
- 시스템 관리 부시스템

1. 패킷 분석 부시스템

패킷 분석 부시스템은 PCI 보드 상의 FPGA를 이용하여 하드웨어로 구현 되어 있으며 내부 구조는 <그림 1>과 같다.

rx_if 모듈은 FPGA의 외부에 있는 PHY/MAC 모듈로부터 받은 패킷을 큐(queue)에 저장하는 역할을 하며, tx_if 모듈은 PCAP 모듈이나 FPGA 외부의 PCI_IF 모듈로부터 패킷을 입력으로 받아 FPGA 외부의 PHY/MAC 모듈로 보내는 역할을 한다. 큐는 PM과

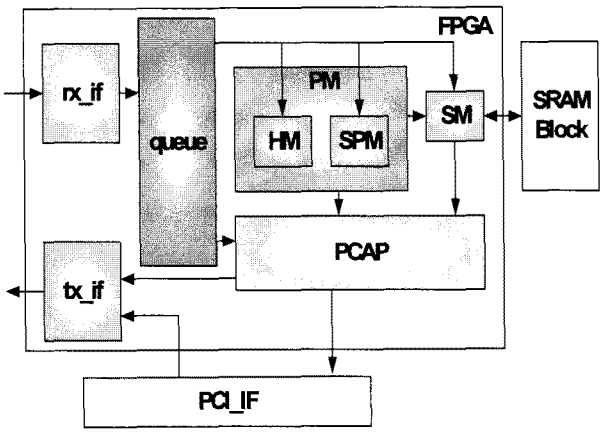


그림 1. 패킷 분석 부시스템의 블록도
Fig. 1. Block diagram of the packet analysis subsystem.

SM 모듈이 패킷에 대한 분석을 수행하는 동안 패킷을 임시로 보관하는 역할을 하며 분석이 끝나면 패킷을 PCAP 모듈로 보낸다.

PM 모듈은 트래픽 패턴 매칭부로서, 패턴 분류 규칙("통과", "차단 또는 분석")을 정하여 이를 SM 모듈과 PCAP 모듈에 알리는 역할을 하며 HM 모듈과 SPM 모듈로 구성되는데, 하드웨어 비용이나 동작 속도 면에서, 가장 설계가 중요한 모듈 중의 하나이다. 헤더 검색을 수행하는 HM 모듈은 패킷의 헤더를 살펴보고 지정된 규칙에 저촉되는 패킷을 찾는 역할을 하고, 스트링 패턴 매칭을 수행하는 SPM 모듈은 패킷에 포함된 페이로드에 지정된 시그니처를 찾아 그에 대응하는 분류 규칙을 정하는 역할을 한다. 본 논문에서는 이 PM 모듈의 설계를 제안하며, 다음 장에서 상세하게 설명한다.

SM 모듈은 PM 모듈에서의 결정에 따라, <소스주소, 목적주소, 소스포트, 목적포트, 프로토콜>의 5-tuple로 이루어진 세션 정보를 외부 SRAM에 업데이트한다. 이 정보는 세션의 분석 및 관리를 위한 것으로서, 예를 들어 PM에 의해 차단 패킷이라고 분석된 패킷이 속한 세션의 다음 패킷들을 차단하는데 사용한다.

PCAP 모듈은 SM과 PM으로부터 받은 분석 결과를 종합하며 패킷을 tx_if모듈로 보내거나(통과) PCI_IF 모듈를 통해 콘텐츠 분석 부시스템으로 보낸다.

2. 콘텐츠 분석 부시스템

콘텐츠 분석 부시스템은 패킷 분석 부시스템으로부터 "차단 또는 분석"으로 분류된 패킷을 받아들여 원래의 콘텐츠로 복원작업을 수행한다. 복원된 콘텐츠는 소프트웨어에 의해 좀 더 정교하게 분석되고, 이는 다시 패킷 분석 부시스템에 의해 외부로 전달된다. 예를 들어, "Security ID Number"라는 시그니처를 패킷 분석 부시스템의 SPM 모듈에서 검색한다고 했을 때 이러한 시그니처가 단지 내용상의 필요에 의해 패킷의 페이로드에 존재 하는 것인지 아니면 어떤 이가 악의에 의해 외부로 보내는 것인지에 대한 조사가 필요하다. 이러한 결정을 콘텐츠 분석 소프트웨어가 내리게 되는 것이다.

3. 시스템 관리 부시스템

시스템 관리 부시스템은 패킷 분석 부시스템의 SPM 모듈에 시그니처를 설정하거나, HM 모듈에 분류자(Classifier)를 설정하거나, 콘텐츠 분석 부시스템에 분석 정책을 설정하는 등, 분석 로그 및 시스템 로그 등의 전반적인 시스템 인터페이스 환경을 제공한다.

Ⅲ. 트래픽 패턴 매칭부(PM)의 설계

본 장에서는 콘텐츠 보안 시스템의 핵심 기술이 되는 헤더 검색과 스트링 패턴 매칭 기능을 구현하는 하드웨어를 제안한다.

1. 헤더 검색(HM) 모듈[5]

헤더 검색에는 다양한 알고리즘이 제안되어 있는데, 하드웨어 구현에는 TCAM을 이용하는 구현이 흔히 사용된다. TCAM 구현은 설계가 간단하고 처리 속도가 빨라 하드웨어 구현에서 빈번히 사용되고 있으나, 하드웨어 비용이 많이 들고 전력 소모가 크며 메모리의 낭비가 크다는 단점을 갖는다[3~4].

본 논문에서는 TCAM의 단점을 극복하기 위하여 Kennedy[3] 등이 제안한 기법 (이하 "KWL 기법"이라고 함)을 콘텐츠 보안 시스템을 위한 헤더 검색 모듈의 설계에 적용하였다. 그런데 콘텐츠 보안 시스템에서는 비교적 많은 헤더 필드를 사용하여 KWL 기법을 그대로 적용하면 (주어진 FPGA 자원에 비하여) 지나치게 넓은 RAM을 사용하게 된다. 이를 피하기 위하여 본 논문에서는 아래의 가항에서 설명된 바와 같이 KWL 기법을 수정하여 적용하였다.

가. 하드웨어 알고리즘

KWL 기법은 HiCuts(Hierarchical Intelligent Cuttings) 알고리즘[3~4]을 기반으로 한다. 알고리즘은 규칙들의 집합으로부터 결정 트리(이하 "HiCuts 트리"라 함)를 구성한다. 패킷 헤더의 각 필드에 따른 지역적 분할을 지정된 횟수만큼 실시한 후, 배정된 규칙을 지정된 개수보다 적게 갖는 잎(Leaf)의 수가 가장 많은 필드를 루트로 설정한다. 이후 루트가 아닌 노드(Node)가 포함하고 있는 규칙들을 같은 방법으로 분할해 나간다[3].

본 논문의 설계에서, HiCuts 알고리즘을 이용한 패킷 분류 기능을 흔히 사용되는 TCAM 기법 대신 KWL 기법으로 구현하는 것이 타당한 이유는 다음과 같다. 첫째, 본 논문의 콘텐츠 보안 시스템에서 헤더 검색 모듈은 주파수가 125MHz인 클록의 매 클록마다 일정량(8비트)의 패킷 데이터를 수신하여 처리하게 된다. 따라서 데이터를 수신하는 시간 내에 검색을 수행하면 되는데

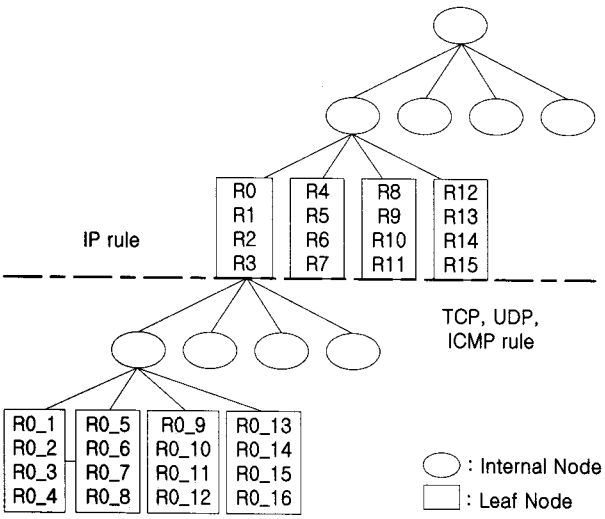


그림 2. 많은 헤더 필드를 포함하는 규칙을 위한 HiCuts 트리의 계층 구조

Fig. 2. Hierarchical structure of the HiCuts tree for rules with many header fields.

이는 KWL 기법을 사용하여도 구현이 가능하다 (본 절의 다항 참조). 둘째, 본 논문의 콘텐츠 보안 시스템의 구현에서 KWL 기법이 하드웨어(메모리, 레지스터 및 기타 MUX나 임의조합회로)를 더 효율적으로 사용하는 것으로 분석된다. 이는 TCAM이 기본 자원으로 제공되지 않는 FPGA상에 TCAM 기법을 구현하기 위해서는, 많은 양의 레지스터와 MUX가 소요되며, 내부 메모리를 사용하여 TCAM 입력 크기에 따른 분할된 구조로 설계하려면 규칙의 개수에 따라 내부메모리의 폭과 길이가 각각 비례하여 증가하기 때문이다.

한편, 많은 수의 헤더 종류와 헤더의 필드를 포함한 규칙을 구현해야 하는 경우, KWL 기법은 (주어진 FPGA 자원에 비하여) 지나치게 넓은 RAM을 필요로 하는 문제가 생긴다. 본 논문에서는, <그림 2>의 예에서 보인 바와 같이 HiCuts 트리가 계층 구조를 갖도록 구성하여 이 문제를 해결하였다. 이러한 구조는 내부 메모리에 적재되는 잎 데이터의 구조에 하나의 주소 공간을 포함시켜 구현할 수 있다.

나. 설계

구현된 하드웨어는, <그림 3>과 같이, 크게 패킷 파서 모듈(Packet Parser)과 검색 모듈(Searcher) 그리고 HiCuts 트리 모듈로 구성되어 있다. 패킷 파서는 입력으로 들어오는 패킷에서 헤더 데이터를 추출하여 검색 모듈에 공급한다. 검색 모듈은 블록 RAM으로 구현된 HiCuts 트리 모듈로부터 노드(내부 노드와 잎 노드)의

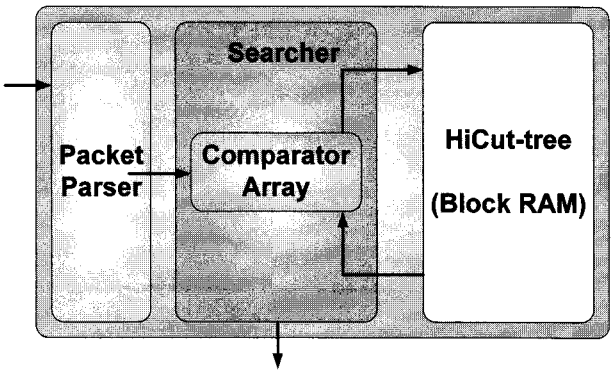


그림 3. 헤더 검색 모듈의 블록도

Fig. 3. Block diagram of the HM module.

데이터를 읽어 해당 데이터를 내부의 비교기 배열(Comparator array)에서 비교한다. HiCuts 트리는 내부 메모리에 구현되며, 소프트웨어에 의해 미리 준비된 노드 데이터를 가지고 있다.

(1) 패킷 파서 모듈

패킷 파서는 IP나 TCP 또는 UDP 및 ICMP 헤더의 필드 데이터들을 입력 패킷으로부터 추출하는데, 추출된 필드 값들은 하나의 입력 패킷에 대한 패킷 분류가 완료될 때까지 보관되어야 한다. 이는 HiCuts 트리 구축 방법에 기인한 것으로, 예를 들면 루트에서 어떤 필드가 언제 사용될지 예측할 수 없기 때문이다.

모든 필드의 출력 값은 유효 여부에 관한 비트(Valid Bit)를 가지고 있어 패킷 파서의 출력을 입력으로 받는 검색 모듈이 각 필드의 값의 유효 여부를 알 수 있게 설계하였다.

(2) 검색 모듈

검색 모듈은 HiCuts 트리 모듈로부터 노드 데이터를 받고 패킷 파서로부터 각 헤더의 필드 값들을 받아, HiCuts 트리를 내부 노드로부터 해당 규칙이 있는 잎 노드까지 검색한다.

본 논문의 설계에서 검색 모듈은 <그림 4>에 도시된 바와 같이, 4개의 비교기 집합으로 구성된 비교기 배열을 가지며, 노드에 따라 혹은 규칙의 종류(IP, TCP, UDP, ICMP)에 따라 비교기의 입력을 선택하는 장치(Input Allocator)와, 비교 결과에 따라 다음에 읽을 데이터를 결정하고 찾는 헤더의 검출 여부를 결정하는 장치(Match Logic)로 구성되어 있다. 각 비교기 집합은 총 12개의 비교기로 구성되며, 규칙의 종류에 따라 노드의 매칭을 수행한다.

(3) HiCuts 트리 모듈

HiCuts 트리 모듈은 내부 메모리로 구현되며 내부 노드와 잎 노드의 데이터를 가지고 있다. 내부 메모리의 하나의 워드(Word)는 1280 비트로 구성되는데, 이는 4개의 규칙으로 구성된 1개의 잎 노드의 데이터 또는 16개의 내부 노드의 데이터를 저장할 수 있다.

하나의 잎 노드에 배정되는 규칙의 개수와 각 노드의 자식 노드의 개수를 모두 4로 제한하였는데, 내부 노드 데이터를 저장할 때 메모리 이용의 효율을 높이고자, 메모리 주소의 하위 2개 비트를 할당하여 노드를 참조할 때 4개의 자식 노드 중 하나를 지정할 수 있도록 하였다. 또한 데이터에 nodeleaf라는 비트를 두어 현재 읽어온 데이터가 잎 데이터인지 혹은 내부 노드 데이터인지를 나타내었다. 잎 데이터는 4개의 규칙 외에 1개의 다음 주소를 가지고, <그림 2>와 같이 수정된 HiCuts 트리의 2단 계층 구조를 지원한다.

내부 노드의 데이터는 필드 형(Field type)과 해당 필드에 대한 분할 정보를 포함한다. 필드 형은 4가지 헤더의 필드 중 어느 필드에 의해 다음 자식 노드가 분할되는 지를 나타내며, 분할 정보는 자식 노드의 주소 정보와 지역 분할의 영역을 나타내는 정보로 구성된다.

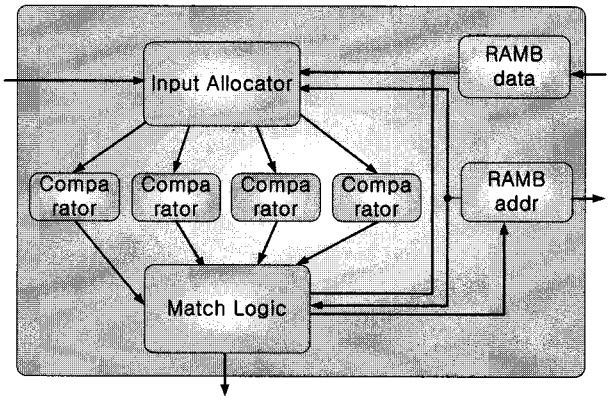


그림 4. 검색 모듈의 블록도
Fig. 4. Block diagram of the Searcher module.

다. 평가

설계된 헤더 검색 모듈을 Verilog HDL로 작성하여 Cadence NC-verilog Simulator를 이용하여 동작을 검증하고, 콘텐츠 보안 시스템의 하드웨어 부분이 구현될 Xilinx FPGA XC4VSX55에 구현하여 131MHz(worst case)까지 동작할 수 있음을 확인하였다. 또한 구현 결과를 흔히 사용되는 TCAM 구현과 비교하였다.

<그림 5>는 비교에 사용된 TCAM 구현을 도시한

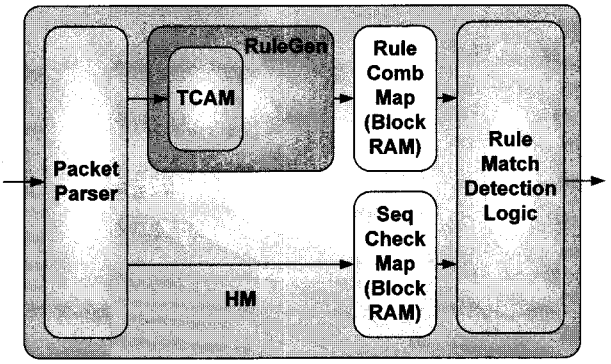


그림 5. 헤더 검색 모듈의 TCAM 구현
Fig. 5. TCAM implementation of the header lookup module.

표 1. 하드웨어 비용 비교
Table 1. Comparison of hardware costs.

	TCAM	본 논문의 설계
Slices	8,494	6,215
LUT	10,845	10,406
Block RAM	96	40

것이다. 사용하는 FPGA는 TCAM을 기본 자원으로 제공하지 않으므로, TCAM 기능은 프리미티브 모듈인 SRL16을 이용하여 구현하였다. 하나의 TCAM 모듈은 입력의 크기에 따라 3에서 9개까지의 SRL16 모듈로 구현되는데, TCAM의 수는 모든 규칙 내의 무정의(don't care)가 아닌 값을 가지고 있는 필드의 수와 같아야 한다. 이들 TCAM의 묶음이 메모리 주소를 생성하여, 참조된 메모리의 데이터를 이용하여 실질적인 규칙을 찾는다. <그림 5>와 같이 설계된 TCAM 구현도 제안하는 설계와 동일한 방법으로 동작을 검증하고 FPGA로 구현하였다.

TCAM을 이용한 구현은 1024개의 부분 규칙*을 적용할 수 있도록 구현하였으며, 제안하는 설계는 1024개의 규칙을 적용할 수 있도록 구현하였다. <표 1>은 두 구현의 하드웨어 비용을 비교한 것이다. 표에서 볼 수 있듯이, 제안된 설계가 약 26%의 Slice와 약 4%의 LUT, 그리고 약 58%의 블록 RAM을 절약할 수 있다.

2. 스트링 패턴 매칭(SPM) 모듈

콘텐츠 보안 시스템에서 성능에 큰 영향을 미치는 또

* 전체 규칙 중 하나의 필드에 해당하는 것

하나의 부분이 패킷 페이로드에 대한 스트링 패턴 매칭 부이다. 스트링 패턴 매칭부에서 각 시그니처 페이로드는 정해진 크기의 부스트링(Substring) 단위로 나뉘어 하드웨어 메모리인 테이블에 저장(등록)된다. 예를 들어, "/etc/inetd.conf"는 "/etc/"와 "inetd.c" 그리고 "onf"로 나뉘어 각각 해당 테이블에 저장된다. 매칭 과정이 시작되면 유입되는 패킷의 페이로드 부분을 지정된 크기의 부스트링 단위로 이동해 가면서 메모리에 해당 부스트링이 저장되어 있는지를 검색한다[2~7]. 그런데 메모리의 적재 내용을 효율적으로 찾기 위하여 <그림 6>에 도시한 바와 같이 해싱 기법이 흔히 사용된다. <그림 6>에서 해당 부스트링은 9비트* 주소 값으로 바뀌어 대응하는 저장 정보를 빠르게 찾는다. 등록된 부스트링이면 선행하는 부스트링 정보가 저장된 주소 (Lead_ptr)을 이용하여 계속 확인하여 나아가고, 전체 스트링이 등록된 스트링인지 아닌지를 결정한다. Flag 비트는 해당 부스트링이 스트링의 어느 부분(머리, 중간, 또는 꼬리)인지를 표시한다. 본 장에서는 스트링 패턴 매칭에서 집중적으로 사용되는 해싱 모듈의 설계를 제안한다.

가. 해싱 모듈 (9-bit Hash)

본 논문에서는 Chowdhury 등[6]이 제안한 Cellular Automata(이하 "CA"라 함)기반의 해시 함수 중 비교적 효율적인 해시 함수(이하 "CA7-Hash"이라 함)를 기반으로 해싱 모듈을 설계하였다.

CA는 D 플립플롭 배열과 다음 상태를 결정하는 조합회로로 구성된 규칙적인 구조로 하드웨어 비용 면에서 효율적인 구현이 용이하다[6]. 콘텐츠 보안 시스템의 패턴 매칭에는 3~7 바이트의 키를 사용하므로 이에 대응하는 구현이 필요하지만 본 논문에서는 4바이트 크기의 키에 대해서만 설명한다.** 해시 값인 주소는 9비트로, 즉 표의 주소 공간의 크기를 512로, 가정하나 설계 기법은 다른 크기에도 적용이 가능하다.

나. 설계

<그림 7>은 CA7-Hash의 구현을 도시한 것이다. CA7은 16x2 배열의 32개의 셀로 구성되며, 각 cell은 한 개의 D 플립플롭과 그 D 플립플롭의 다음 상태를 결정하는 조

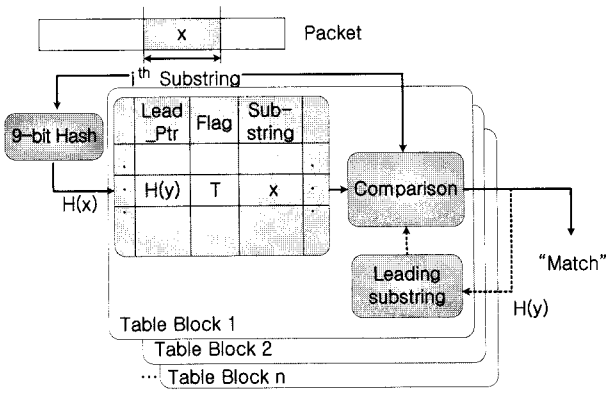


그림 6. 스트링 패턴 매칭의 구현
Fig. 6. Implementation of string pattern matching.

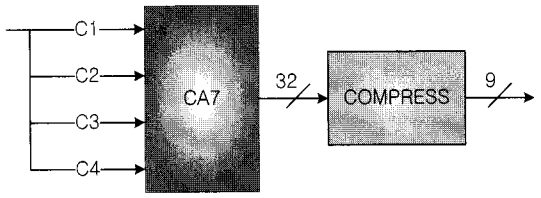


그림 7. CA7-Hash의 구현
Fig. 7. Implementation of CA-Hash.

합회로로 구성된다. 다음 상태를 결정하는 셀간의 접속을 수식(1)과 같이 정의할 수 있는데[6], 각 cell의 다음 상태는 4개의 이웃 셀과 자신의 현재 값에 의존한다. 자신과 이웃 셀을 Self(S)와 Top(T)과 Left(L) 및 Bottom(B), 그리고 Right(R)로 표기하여 각 셀의 의존 관계를 (S T L B R)의 형태로 표기하는데, 예를 들어 수식(1)에서 13이란 숫자는 이진수로 01101이므로, 해당 셀의 다음 값이 자신의 위치에서 Top, Left, Right의 현재 값들을 XOR연산을 한 결과 값이 된다.

$$CA7 = \begin{bmatrix} 29 & 29 & 13 & 29 & 13 & 29 & 13 & 29 & 13 & 29 & 13 & 29 & 13 & 29 & 13 & 29 \\ 29 & 29 & 13 & 29 & 13 & 29 & 13 & 29 & 13 & 29 & 13 & 29 & 13 & 29 & 13 & 29 \end{bmatrix} \quad (1)$$

<그림 7>에서 CA7은 4개의 바이트 입력 (C1~C4)으로부터 32비트의 출력을 산출하는데, COMPRESS는 이 32비트에서 9비트를 선택하는 블록이다. 본 논문에서는 32비트 중에서 선택된 비트를 1로 나타내어 16진수로 표현한 값으로서 COMPRESS 함수를 정의하였다. 이 COMPRESS 함수도 해싱의 성능에 영향을 주므로, 다양한 선택에 대하여 생성되는 주소의 충돌 발생률을 모의실험하여 충돌 발생률이 가장 낮은 구성을 선택하여 설계하였다. COMPRESS 함수를 적절히 선택하면, <그림 7>의 구현이 김병구 등[2]에서 사용된 해싱 모듈보다 낮

* 512-entry 테이블의 경우임.
** 입출력이 32비트 보다 큰 경우에는 Chowdhury 등[6]이 제안된 바와 같이 동일 구조를 병렬로 접속한다.

은 충돌 발생률을 얻을 수 있다[7].

다. 평가

설계된 해상 모듈도 헤더 검색 모듈과 동일한 방법으로 동작을 검증하고 FPGA로 구현하였다. Linux 파일 경로 데이터에서 임의로 선택된 100개의 입력 집합을 사용하여 실험하였는데, <그림 8>은 메모리 적재율(Load ratio)*와 COMPRESS 함수에 따른 충돌 발생률을 가장 좋은 성능을 보인 10개의 COMPRESS 함수에 대하여 보인 것이다. <그림 8>에서 볼 수 있듯이 메모리 적재율에 따라 최적의 COMPRESS 함수가 다르다. 본 논문에서는 메모리 적재율에 따라 최적의 COMPRESS 함수를 선택하도록 설계하였다.

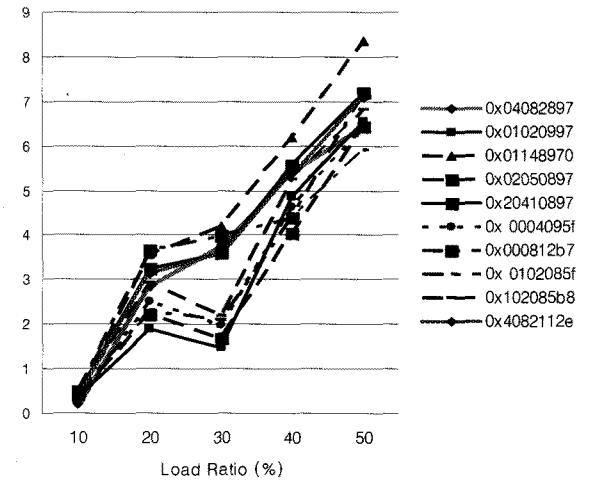


그림 8. 메모리 적재율과 COMPRESS 함수에 따른 충돌 발생률(%)

Fig. 8. Collision rates(%) depending on the load ratio and the COMPRESS function.

IV. 결 론

본 논문에서는 고성능 네트워크 응용에서 사용하기 위한 트래픽 패턴 매칭 하드웨어를 제안하였다. 헤더 검색부는 흔히 사용되는 TCAM 구현 대신, 비교기 배열과 HiCuts 트리에 기반을 둔 구현 기법을 채택하여 하드웨어 비용을 현저하게 감소시켰다. 이 과정에서, 많은 수의 헤더 종류와 헤더의 필드를 포함한 규칙을 처리해야하는 콘텐츠 보안 시스템의 특성에 맞추어 HiCuts 트리를 2단계층 구조로 수정하여 적용하였다. 스트링 패턴 매칭부의 설계에서는 하드웨어 면에서 효율적이며 전력 소모를 최소화할 수 있는 셀룰러 오토마타형 해싱 모듈을 설계하였다.

본 논문에서 제안된 설계는, 많은 양의 하드웨어가 사용되는 고성능 네트워크 하드웨어에서 중요한, 하드웨어 비용과 전력 소모량을 현저하게 감소시킬 수 있다. 제안된 트래픽 패턴 매칭 하드웨어는 현재 개발되고 있는 콘텐츠 보안 시스템에 실장되어 검증될 것이며, 보다 고속화되는 환경에 맞추어 지속적으로 개선되어야 한다.

* 주어진 메모리 공간에서 데이터가 적재되어 있는 공간의 비율.

참 고 문 헌

[1] 신승원, 강동호, 김기영, 장종수, "DPI 기술 분석", 정보통신동향분석, 제19권 제3호, 117-124쪽, 2004년 6월

[2] 김병구, 윤승용, 오진태, 장종수, "하드웨어 기반의 고성능 침입탐지 기술", 정보통신동향분석, 제22권 제1호, 51-58쪽, 2007년 2월

[3] A. Kennedy, X. Wang, and B. Liu, "Energy efficient packet classification hardware accelerator", in Proc. of IEEE Int. Symp. on Parallel and Distributed Processing, pp.1-8, Apr. 2008.

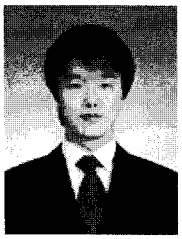
[4] P. Gupta and N. McKeown, "Algorithms for packet classification," IEEE Network, Vol. 15 No. 2, pp.24-32, Apr 2001.

[5] 최영, "패킷 분류를 위한 하드웨어 가속기," 석사 학위논문, 고려대학교 대학원, 2009.

[6] D. Chowdhury, I. Gupta, and P. Chaudhuri. "A Low-Cost High-Capacity Associative Memory Design Using Cellular Automata", IEEE Trans on Computers. Vol.44, No.10, pp.1260-1264, Oct. 1995.

[7] 홍은경, 백승태, 최일훈, 오형철. "고성능 콘텐츠 필터링 시스템을 위한 해싱", 대한전자공학회 추계종합학술대회 논문집, 제31권 제2호, pp. 183-184, Nov 29, 2008.

저 자 소 개



최 영(학생회원)
2007년 고려대학교 전자및정보 공학 학사
2009년 현재 고려대학교 대학원 전자정보공학과 재학
<주관심분야 : 컴퓨터 산술, SoC 설계>

홍 은 경(학생회원)
2008년 고려대학교 전자및정보 공학 학사
2009년 현재 고려대학교 대학원 전자정보공학과 재학
<주관심분야 : 컴퓨터 산술, SoC 설계>

김 태 완(정회원)
1999년 서울대학교 산업공학 학사
2009년 현재 (주)소만사
<주관심분야 : 네트워크 보안, 프로토콜 분석, 데이터베이스 보안, 개인정보 보호>

백 승 태(정회원)
1999년 고려대학교 정보공학 학사
2009년 현재 (주)소만사
<주관심분야 : 네트워크 보안, 프로토콜 분석, 데이터베이스 보안, 개인정보 보호>

최 일 훈(정회원)
1995년 서울대학교 컴퓨터 과학 학사
2007년 연세대학원 산업정보경영 석사
2009년 현재 (주)소만사
<주관심분야 : 네트워크 보안, 프로토콜 분석, 데이터베이스 보안, 개인정보 보호>

오 형 철(정회원)
1982년 서울대학교 전자공학 학사
1984년 한국과학기술원 전기및전자공학 석사
1984~1987년 (주)금성반도체 연구소
1993년 Univ. of Maryland of College Park 전기공학 박사
2009년 현재 고려대학교 전자및정보공학과 교수
<주관심분야 : 컴퓨터 구조, 컴퓨터산술, SoC, 내장형 시스템>