TCAM을 이용한 하드웨어 기반 메커니즘에서의 TCP 상태기반 패킷 필터기의 설계 및 구현

이승복° 신동렬
성균관대학교 컴퓨터공학과
{sblee1°, drshin}@ece.skku.ac.kr

Design and Implementation of TCP stateful packet filter in Hardware-based mechanism using TCAM

Seoung-Bok Lee° , Dong-Ryeol Shin
School of Information and Communication Engineering, Sung Kyun Kwan University

요 약

인터넷 네트워크에 존재하는 방화벽(Firewall) 또는 라우터(Router) 장비에서의 패킷 필터 기능은 모든 방화벽 장비의 기본적인 기능이 될 수 있다. 하지만 최근에 등장한 세션기반의 악의적 침입과 바이러스의 출현으로 패킷 필터기는 단순한 정적 패킷 필터 기능이 아닌 상태기반 패킷 필터의 동적 패킷 필터 기능을 요구하게 되었다. 또한 최근에 인터넷 속도가 급증하는 환경변화에 맞추어 방화벽 장비의 TCP 패킷 처리기능은 매우 빠른 처리속도를 요구하고 있다. 이에 우리는 매우 빠른 고속의 TCP 상태기반 패킷 필터 처리를 요구하는 에지(Edge)급 라우터의 방화벽 옵션카드를 만들기 위해 하드웨어 기반의 TCAM(Ternary CAM) 관리를 이용한 TCP 세션 상태기반 (Stateful) 패킷 필터기를 구현하였으며, TCAM 제어와 패킷의 상태기반 검사 등 모든 기능처리는 FPGA(Field Programmable Gate Array)를 이용한 하드웨어 로직(Logic) 및 상태기(State Machine)로 구현하였다. 그리고 본 논문의 구현방식을 적용한 방화벽 옵션카드는 인-라인(In-line) 모드로 구성될 경우 1GHz 이상의 Wire Speed를 만족하는 처리성능을 보여주었다.

1. 서 론

인터넷 네트워크에 존재하는 방화벽은 공격자로부터 네트워크를 보호하는데 사용되며, 인터넷 네트워크의 가장자리(에지:Edge)에 독립적으로 위치하거나 에지-급 라우터에 그 기능이 내장되어 어떤 패킷에 대한 접근을 허용할 것인지에 대한 보안정책을 수행하는 기능을 담당한다. 최근의 인터넷 망에서 네트워크 속도는 꾸준히 증가하고, 낮은 지연율(Low Latency)을 가져야 하는 음성, 동영상 등의 새로운 서비스들은 더 높은 처리율(Throughput)을 요구하고 있는 반면에, 인터넷 서비스 가입자는 점점 더 빠른 서비스를 요구하고 있다 [4].

그리고, 한편으로는 최근에 등장한 세션기반의 악의적 침입과 바이러스의 출현 및 갈수록 교묘해지는 공격자에 대응하기 위하여 패킷 필터기는 단순한 정적 패킷 필터 기능이 아닌 상태기반 패킷 필터의 동적 패킷 필터 기능을 요구하게 되었다 [3].

상태기반 패킷 필터는 세션테이블과 상태테이블, 2개의 테이블을 필요로 한다. 본 논문에서는 고속의 상태기반 패킷 필터기 구현을 위하여 하드웨어 기반의 검색 알고리즘을 가지며 높은 처리율을 가지는 TCAM (Ternary CAM)을 세션 테이블로 구성하였으며, 고속을 지원하는 SRAM 메모리를 상태 테이블로 구성하였다. 그리고 하드웨어 상태기를 이용하여 TCAM의 세션 테이블과 SRAM의 상태테이블을 관리하도록 구현하였으며, 실제 라우터 방화벽 옵션카드를 라우터에 탑재하여 시험한 결과 1GHz 이상의 Wire Speed를 만족시키는 TCP 상태 기반 패킷 필터의 검색성능을 확인할 수 있었다.

2. 관련 연구

2.1 패킷 필터

패킷 필터는 네트워크에 대한 접근 규칙과 정책들을 설정하는 수단을 제공하며, 네트워크 안으로 들어오고 밖으로 나가는 모든 패킷은 이들 규칙에 의해 검사되고 패킷들은 보안정책에 위배될 경우 폐기되거나 거절된다. 그리고 하나의 규칙은 이를 만족하는 패킷을 위해 가져야 하는 액션을 가지며, 보통 한 개의 규칙을 정의하는데 사용되는 TCP/IP 헤더의 필드들은 발신지주소, 목적지주소, 발신지포트, 목적지포트, 프로토콜 필드 등의 어떤 조합이 될 수 있고, TCP 플래그, 시퀀스 번호와 같은 보조필드들을 포함할 수도 있다. 이와 같은 패킷 필터는 크게 2가지로 구분할 수 있는데 정적 필터(Static Filter) 과 동적 필터 또는 상태기반 필터(Stateful Filter) 로 나뉘어 질 수 있다 [3][4].

정적 필터(Static filter)는 규칙들의 우선순위화 된 목록를 가지며, 모든 규칙들은 관리자에 의해서 생성되고, 데이터베이스에 저장된다. 이들 규칙은 직접 관리자 입력 없이 변화하지 않으므로 낮은 업데이트 율을 가진다 [3].

상태기반 필터는 상태의 개념을 더 가진다. 한 개의 규칙은 한 개의 세션이 방화벽의 보호된 네트워크에서 외부의 망으로 시도될 때 동적으로 발생된다. 상태기반(Stateful) 패킷 필터를 가지는 방화벽이 보호된 네트워크 밖에 있는 목적지로 보내지는 요청을 발견할 때, 방화벽은 그 요청 패킷의 정확한 발신지와 목적지 IP주소, 정확한 발신지와 목적지 포트, 그리고 프로토콜을 가지는 한 개의 세션을 동적으로 만들어 낸다. 지금 보호된 네트워크로 들어오기를 시도하는 패킷은 누구나 상태기반 방화벽에 저장된 한 개의 세션과 정확히 매칭되어야만 한다. 이 규칙은 세션이 종료되거나 미리 정해진 Idle Time의 기간이 종료될 때까지 세션목록에서 유지되어 요구 받지 않은 어떤 패킷이라도 네트워크 안으로 들어오는 것을 방지한다 [3][4].

2.2 라우터 방화벽 (Router Firewall)

대부분의 라우터에서는 고속의 패킷을 처리하기 위해 고성능의 네트워크 프로세서(Network Processor: NP)를 가지는데 패킷 데이터 처리의 대부분을 담당한다. 보통 라우터의 네트워크 프로세서는 다양한 PHY Framer로부터 패킷을 수신해서 처리하고, 경우에 따라서는 Framer의 일부 기능도 수행할 수 있으므로 네트워크 프로세서에는 항상 많은 부하가 걸리게 된다. 그리고 라우터는 일반적으로 패킷 필터 기능으로서 ACL(Access Control List)기능을 가진다. [2] 이 기능은 주로 상태기반이 아닌 관리자의 미리 정해진 정책에 의한 패킷의 허용여부를 결정하는 것으로서, 고속의 패킷 분류(Classification) 처리를 위해 하드웨어 알고리즘 기반인 TCAM Co-Processor가 사용될 수 있다.

라우터에서의 ACL(Access Control List), 패킷 포워딩 등에서 사용되는 Range Match 기법과 달리 TCP 상태기반 패킷 필터 링은 Exact Match 기법을 적용한다. 정확히 매칭되는 값만이 현재 패킷에 대한 이전세션의 히스토리(History) 성분을 가지기 때문이다.

에지-급 라우터의 구조에서 방화벽 옵션을 적용할 때, 네트워크 프로세서 기반에서의 상태기반 패킷 필터를 수행할 경우 라우터는 상당한 과부하로 인해 시스템 성능저하가 심화될 수 있다. 또한 상태기반 패킷 필터를 처리하기 위해서는 먼저 수신한 패킷의 이전 세션이 존재하는지를 확인하기 위해 세션 테이블을 검색하는 과정이 필요하며, 이 결과에 따라 1 개의 패킷 데이터 처리를 위해 새로운 테이블을 추가하거나 삭제 또는 변경하는 등의 테이블에 대한 여러 번의 접근 과정이 요구되는 경우도 있다.

이에 본 논문에서는 라우터 방화벽의 시스템 성능저하를 막으면서 라인속도(Line Speed)를 제공할 수 있는 해법으로서, FPGA와 TCAM을 이용한 완전한 하드웨어 기반의 TCP 상태기반 패킷 필터를 구현하기 위한 방법을 제공한다.

2.3 TCP 상태기반 검사 (Stateful Inspection)

체크 포인트(Check Point)사의 Firewall-1 제품은 처음엔 SYN 패킷이 입력될 때에만 규칙기반 검사를 행하여 허용이면 연결상태 테이블에 연결등록 및 저장하고 허용되지 않은 패킷은 폐기되는 형식이었다. 이후 TCP 3-Handshaking 패킷인 SYN_ACK, ACK는 이전의 연결상태 테이블이 존재하면 검사되지 않고 그대로 통과되었다. 이와 같은 방식은 도스공격(Dos Attack) 이나 SYN Flooding 공격에 매우 취약하여 이들 공격을 방지하기 위한 새로운 방법들이 소개되었다 [1].

그래서 현재의 대부분의 방화벽 시스템에서는 SYN 패킷일 경우 규칙기반 검사를 수행하고 난 후 세션 테이블에 미완성 상태로 등록이 되며, 이후의 TCP 3-Handshaking 패킷은 미완성 상태의 세션테이블에 적용되고 연결상태 테이블에 등록된다.

본 논문의 구현에 있어서 규칙기반 검사는 구현되지 않았다. 이미 라우터 방화벽 옵션 카드는 기본적인 정적 필터 링(Static Packet Filtering)기능인 ACL 기능을 가지고 있으며, 이를 통과한 패킷 만이 방화벽 옵션카드로 입력되기 때문이다.

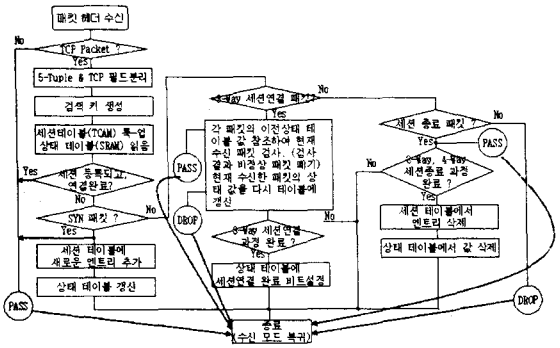

그림 1. TCP 상태기반 검사 흐름도

TCAM과 고속의 하드웨어 로직 제어기를 이용한 상태기반 패킷 필터를 구현하기 위해, 본 논문에서는 그림 1. 과 같은 흐름도를 가지는 메커니즘으로 설계하였으며, TCP 패킷에 한정해 패킷 필터 기능을 수행하고 나머지는 그냥 통과 시킨다. 전체적 흐름에 대한 설명은 아래와 같다.

(1) 패킷 필터기는 입력되는 모든 패킷에 대해 이전 세션의 존재여부를 확인하기 위해 TCAM 룩-업(Lookup)을 수행하고 이에 대칭되는 상태 테이블 주소를 얻기 위해 SRAM 상태 값 데이터 읽기를 수행한다. 패킷을 룩-업 한 결과 세션 연결테이블에서 "Established" 비트 값이 나타나면 이미 이전 세션에서 등록된 패킷이므로 그냥 통과 된다.

(2) 룩-업해서 상태 테이블로부터 읽은 값이 아직 미완성의 TCP 3-Handshaking 값을 나타내면, 필터기는 기본적인 TCP 3-Handshaking의 각 단계에 맞는 정확한 패킷 인지 검사한 후에 정상이면 상태 테이블을 현재 수신한 값으로 갱신하고 포워딩 (Forwarding) 블록으로 통과 신호를 보내며 아니면 폐기 신호를 내보낸다.

(3) 이전세션에 대한 룩-업 실패 및 상태 테이블 결과값이 존재하지 않는 최초의 SYN 패킷 일 경우, TCAM 테이블 관리기는 새로운 세션을 나타내는 엔트리(Entry)로서 TCAM에 등록하고 상태테이블에도 상태 값을 SYN 플래그로 갱신한다.

(4) TCP 3-Handshaking 단계가 끝나고 연결설정 완료상태에 이르면 상태테이블에 "Established" 값으로 갱신한다.

(5) 3-Way, 4-Way 방식의 TCP 세션 종료를 감지하여 기존에 등록된 모든 세션테이블과 상태테이블을 찾아서 해당되는 세션정보들을 모두 삭제한다.

2.4 TCAM (Ternary CAM)

TCAM은 매우 비싼 가격과 전력소모의 단점도 있지만, 빠르고 일정한 룩-업 시간을 보장하는 이점으로 인해 라우터, 방화벽 등 여러 네트워크 장치에서 사용되고 있다. [8]

TCAM을 제외한 다른 많은 검색 알고리즘들은 룩-업을 위해 더 많은 메모리 액세스 횟수를 가지는데, 이는 메모리 액세스 Latency를 감추려고 사용하는 Pipeline 구조의 사용을 어렵게 하기 때문에 Throughput 성능을 제한시킨다. 더욱이 대부분의 검색 알고리즘은 2-Dimension 구조이거나 더 높은 Dimension 구조의 휴리스틱 알고리즘을 사용하므로 고정적이지 못한 성능을 제공하는데 반해 TCAM은 항상 O(1)의 클럭 사이클에서 매칭 규칙을 찾으므로 가장 빠른 룩-업 시간을 제공한다 [6][8].

본 논문에서 사용한 TCAM은 IDT사의 9Mbit를 가지는 IDT75K62100이며, IPv4 패킷일 경우 최대 144 비트 기준으로 64K 세션을 지원할 수 있고, IPv6일 경우 288 비트 기준으로 최대 16K 세션까지 지원 가능하다. 상태 테이블로 사용되는 SSRAM은 512K * 36비트 구조이며, Pipeline 구조를 지원한다. 그리고 TCAM은 72비트의 입력을 가지며 최대 100MHz까지 동작한다.

3. H/W 기반의 TCP 상태기반 패킷 필터 설계

3.1 TCP 상태기반 패킷 필터링의 설계

인터넷 망에서 보호되는 서버 측과 보호 되지 않은 외부 망 사이에 존재하는 에지-급 라우터 방화벽은 TCP 양방향 통신 중간에 위치하므로 1 개의 세션에 대해 2 개의 입력포트를 가진다. 이에 본 논문의 구현에서는 1개의 세션에 대해 2개의 세션 테이블과 2개의 상태 테이블을 가진다. 이 때 세션 테이블은 각각의 방향에 해당하는 엔트리를 가지며, 상태 테이블은 1 개의 세션 양방향에 2개의 상태를 가지므로 항상 동일한 값을 유지하도록 구현되었다.

그림 2. 는 1개의 세션에 대해 양방향 패킷이 입력되었을 때를 처리하기 위한 제어 흐름도 이다. TCAM에서 양방향 테이블을

관리하기 위하여 2개의 엔트리를 가짐으로 인해 메모리 낭비가 있지만 1 개의 엔트리를 구성하는 것 보다 더 빠른 처리가 가능하다. 순방향으로 진행된 TCP 패킷에 대한 응답이 역방향으로 진행하여 패킷 필터에 입력될 때, 이전 세션과 정확히 Exact Matching이 되도록 하기 위해 입력 패킷의 헤더 중에서 IP 와 TCP 의 발신자/목적지 필드를 서로 바꾸어 준다.

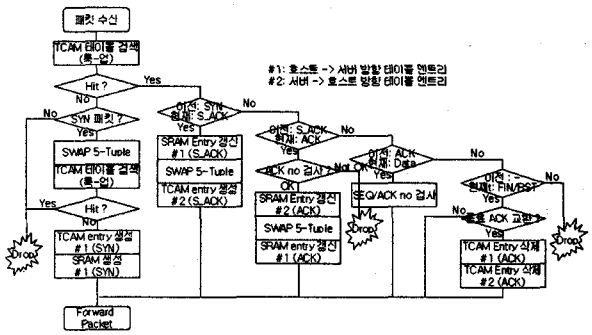

그림 2. 양방향 패킷를 위한 TCAM 엔트리 제어 흐름도

그림 2. 의 TCAM 엔트리 제어 흐름도에 대한 동작설명은 아래와 같다.

1) 현재의 입력된 패킷에 대해 룩-업(Lookup)을 수행하여 이전 세션의 상태 값들을 읽어와서 수신한 패킷을 통과시킬 것인지 폐기할 것인지를 결정한다.
2) 룩-업(Lookup)결과로서 수신한 패킷이 TCP 3-Handshaking SYN 이라면 세션등록을 위한 새로운 세션의 TCAM 엔트리를 추가하며, 수신한 패킷이 TCP종료를 알리는 것이라면 해당 세션의 TCAM 엔트리를 삭제 한다.
3) TCP 3-Handshaking 단계일 때 발신지 A에서 목적지 B로 SYN 패킷이 전송될 경우, A 에서 B로의 세션에 대한 정보뿐만 아니라, B 에서 A로의 세션에 대한 정보도 등록하여 양방향 세션관리가 이루어지도록 하며, 세션등록 정보를 삭제할 때에도 동시에 두 개의 세션모두 삭제한다.
4) TCP 3-Handshaking 단계이고, 아직 세션이 확립되지 않은 패킷을 수신할 경우 TCP/IP 헤더의 5-Tuple중에서 프로토콜 필드를 제외한 발신지 IP 주소와 목적지 IP 주소, 그리고, TCP 발신지 포트와 TCP 목적지 포트 필드들을 각각의 IP, TCP에 대해서 필드 위치교환(Swap)을 수행한다.

3.2 TCP 상태기반 검사의 주요 흐름도

그림 3. 은 입력 패킷 헤더의 5-Tuple 필드를 이용하여 최초 TCAM을 룩-업 한 후에 이번 패킷에 대한 이전의 상태 플래그 값을 상태 테이블에서 읽어와서 현재의 상태 플래그 값과 비교하는 메커니즘을 보여준다

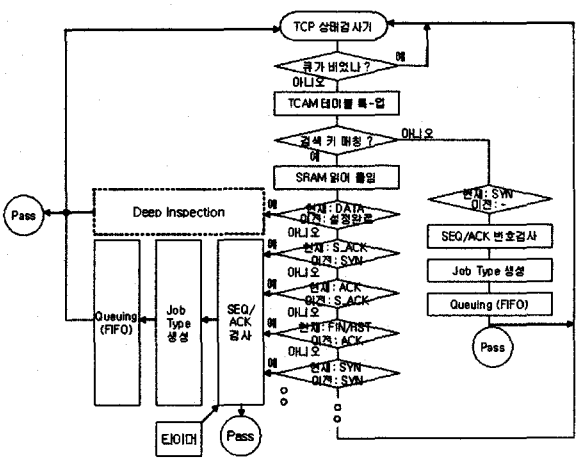

그림 3. TCP 상태기반 세션 검사

그림 3. 에서 만약 룩-업 결과가 최초의 SYN 패킷이거나, 다른 연결설정 패킷이라면 SEQ 번호와 ACK 번호를 검사하고 나서 TCAM 테이블 관리를 위하여 별도의 JOB Type을 부여하고 이를 큐에 저장한다. 연결설정이 완료된 패킷은 추가적인 패킷의 정밀 검사(Deep Packet Inspection)을 수행하는 방법이 있지만 본 논문의 구현에서는 적용되지 않았다.

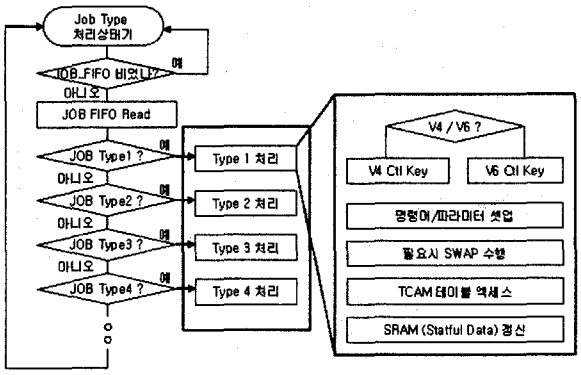

그림 4. Job Type별 테이블 관리도

그림 4. 는 그림 3. 에서 Job 큐(Queue)에 저장된 것을 읽어들여 해당 Job type별 처리동작을 나타낸 것이다. Job Type은

그림 3. 에서 이전과 현재 세션의 상태 플래그를 서로 비교함으로써 얻어지며, 그림 2. 에서 각각의 세로 줄 흐름도가 1 개의 Job type이 될 수 있다. 각각의 Job Type은 그림 2. 에서와 같이 서로 다른 TCAM 액세스 횟수를 가진다.

4. 구 현

그림 5. 는 FPGA로 구현한 기능블록도 이다. FPGA 내부 클록 (Clock)은 100MHz를 사용하였으며 TCAM으로 입력되는 검색 키나 제어 키 부분을 제외하고는 모두 32 Bit 데이터 처리를 수행한다. 내부적으로 TCAM으로 입력되는 키(Key) 처리부분은 대부분 병렬로 처리하여 매우 적은 지연율을 가지며, 패킷의 고유 ID와 Sequence 번호, Acknowledge 번호, 플래그 필드 등을 저장하는 상태테이블인 SRAM은 32비트 3개를 사용하여 총 96비트의 병렬을 제공하여 1 개의 세션에 대한 상태 값을 한번에 모두 읽어 들이거나 갱신하도록 하였다.

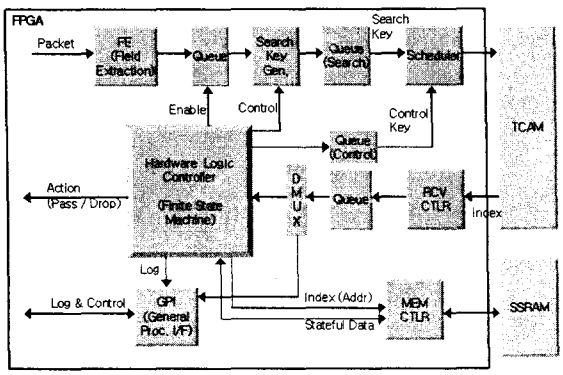

그림 5. 하드웨어 로직 기반의 컨트롤러와 TCAM을 이용한 TCP 상태기반 패킷 필터의 구조

그림 5. 의 그림에 대한 설명은 다음과 같다. 패킷의 헤더부분이 HFE (Header Field Extraction)로 입력되면, 패킷 필터기는 이 헤더를 연속적으로 파싱 (Parsing)하고 이를 이용하여 룩업 키를 만들어 TCAM 엔트리를 룩-업(lookup) 한다. 룩-업(Lookup) 결과가 통과(Pass) 또는 폐기(Drop)이면 더 이상의 TCAM 액세스는 없지만, TCP 3-Handshaking 단계에서의 패킷들은 세션 설정이 완료될 때까지 한 번 이상씩 TCAM을 액세스 하게 된다.

HFE에서는 패킷의 IP 버전을 검사하여 검색 키로 사용되는 필드-값들이 파싱(Parsing)되어 큐에 저장되며, IPv4 와 IPv6는 서로 다른 길이의 필드들을 가지므로 1 개의 패킷에 대해 서로 다른 깊이의 큐 사이즈를 가진다.

파싱 큐에 저장된 필드들은 하드웨어 로직-컨트롤러에 의해 인에이블(Enable) 되어 검색 키 생성 부로 전달되며, 이들 필드들은 검색 키 생성 부에서 TCAM 입력 형식에 맞게 검색 키를 생성하고 이와 관련된 TCAM 파라미터들을 동시에 생성시킨다.

검색 키 큐에서는 스케줄러에 의해 검색 키와 제어 키들이 TCAM으로 입력되기 전 대기상태로 존재하며, 출력신호가 오면 연속적으로 출력된다.

8 stage 파이프라인 구조의 TCAM으로부터 출력되는 결과 값들을 수신하기 위해, 8 개의 수신기가 존재하며 이 들 수신기는 결과 값들을 모두 수집하여 수신 큐에 저장한다. 그리고 테이블 관리 제어기에서 생성한 제어 키에 대한 결과값은 입력 패킷의 룩-업의 수신 큐와 다른 수신 큐나 레지스터에 저장된다.

DeMux에서는 시험용이나 기타 목적을 위한 TCAM제어를 위한 결과값과 입력 패킷의 룩-업 결과값, 그리고 테이블 관리제어기의 제어 키 결과값에 따른 경로를 제공해 준다.

하드웨어 로직 제어기(Logic Controller)에서는 TCAM 테이블 관리를 위한 상태기 로직과 TCP 3-Handshaking 단계에서의 패킷 검사 등을 위한 로직으로 구성되며, 라인 속도의 패킷 필터 결과 값을 패킷의 시퀀스 번호와 함께 포워딩 부분으로 전달해 준다.

본 논문에서 사용한 FPGA는 자일링스(Xilinx)사의 XC2VP70 시리즈이며 본 논문의 구성을 위한 로직 점유율은 전체 FPGA 내의 20%정도이다. 그리고 자일링스 시스템 Tool을 이용하여 FPGA 내부에 MicroBlaze CPU를 구현할 수 있으며, 본 논문에 설명되지 않은 로깅 기능 등은 이 CPU 기능을 사용하여 구현되었지만 본 논문에는 포함되지 않았으며, 본 논문의 구현을 위해 코딩(Coding)한 하드웨어 언어는 Verilog이다.

5. 성능 측정

SmartBit-6000 패킷 발생기 와 모니터링, 그리고 Firewall 옵션 소프트웨어인 WebSuite/Firewall 을 사용하여 TCP 양 종단 사이에 본 논문의 구현장치인 TCP 상태기반 패킷 필터를 내장하는 라우터 방화벽을 설치하여 시험 하였다.

이 시험에서 1Gbps의 라인속도에 이르는 TCP 세션 테이블(Session Table)의 룩-업(Lookup) 처리속도를 나타냈으며, 동시세션 연결 능력(Concurrent Connection

Capacity)은 단 방향으로 IPv4인 경우 초당 32768 세션, IPv6인 경우 초당 16384 세션을 처리할 수 있었으며, 이는 TCAM의 최대 엔트리(Entry) 수만큼 지원함을 의미한다.

6. 결 론

TCP 상태기반 패킷 필터를 하드웨어 기반으로 구현하였을 때 TCAM이 지원하는 용량 내에서는 상태검사를 위한 룩-업 처리 속도가 라인속도를 제공함을 보았으며, TCAM의 용량을 확장하면 동시 최대 연결능력은 동일한 크기만큼 증가할 것이다. 본 논문에서는 TCP 상태기반 패킷 필터에 관한 구현내용이지만 향후에 본 논문에서 구현한 필터 구현방법을 기반으로 세션기반에서의 패턴 매칭을 이용한 바이러스 차단기 등을 사용하는 하드웨어 기반의 IPS(Intrusion Protection System), IDS(Intrusion Detection System) 시스템 등의 보안장비 응용에도 활용될 수 있을 것이다.

참고 문헌

[1] Noureldien A. & Izzeldin M. "A stateful Inspection Module Architecture", IEEE 2000.

[2] Chris Roeckl "Stateful Inspection Firewall", white paper, Juniper Networks, http://www.juniper.net/solutions/literature/white_papers/wp_firewall.pdf

[3] An Examination of Firewall Architectures, www.cs.plu.edu/courses/CompSec/arts/cfirewall.pdf

[4] Addressing Next-Gen Firewall Design Challenges Part I: Firewall Basics And Static Firewalls, www.analogzone.com/iot_0516.pdf

[5] Wasti, S., 2001 "Hardware Assited Packet Filtering Firewall" Proceedings of the 2000-2001 Grad Symposium, CS Dept, University of Saskatchewan, 11 April 2001

[6] Kai Zheng, hao Che, TCAM-based Distributed Parallel Packet Classification Algorithm with Range-Matching Solution

[7] IDT corporation, IDT75K62100 datasheet

[8] Pankaj Gupta, An Algorithm for Performing Routing Lookups in Hardware http://klamath.stanford.edu/~pankaj/thesis/chapter2.pdf