다중계층 프로토콜의 적합성시험 방안

박 용 범† · 김 명 철†† · 김 장 경†††

요 약

Multi-protocol로 구성된 시험대상의 프로토콜에 대한 적합성 여부를 시험하기 위해서는 각 계층에 대한 시험이 모두 수행되어야 한다. ISO9646에 의하면 multi-protocol 시험대상의 최상위 프로토콜에 대해서는 단일 계층 시험방법(single-layer test method)이 사용된다. 최상위 프로토콜을 제외한 나머지 프로토콜은 인터페이스가 개방되지 않으므로 각각에 대해 단일 계층 내포 시험방법(embedded test method)을 반복하여 적용한다. 그리고 기존의 내포 시험방법에 대한 연구는 multi-protocol 시험대상 내의 단일 계층을 시험하는데 초점을 맞추고 있다.

본 논문에서는 multi-protocol Implementation Under Test(IUT)에서 기존의 방법과는 달리 하나의 시험 스위트에서 두개 이상의 계층에 대한 시험을 동시에 수행하는 방안을 제안한다. 또한 이 방법에서는 프로토콜 간의 인터페이스가 개방되지 않은 multi-protocol 시험대상을 가정한다. 제안된 방법을 간략화 한 TCP/IP에 적용하여 시험을 작성하는 예를 보이고 기존의 시험방법에 의한 시험과 시험경우의 수, 시험 이벤트의 수 그리고 시험 coverage 관점에서 비교한 결과, multi-protocol 시험방법을 사용하면 프로토콜을 별도로 시험하는 기존의 방법에 비해 동일한 시험 커버리지(coverage)를 가지면서도 시험경우의[1] 수와 behavior line의 수 그리고 시험 이벤트의 수가 줄어들게 된다. 또한 제안된 시험방법은 multi-protocol의 어떤 계층에서 오류가 발생하였는지를 정확히 밝혀 낼 수 있다.

Conformance Testing of Multi-protocol IUTs

Yong-Bum Park† · Myung-Chul Kim†† · Jang-Kyung Kim†††

ABSTRACT

To declare conformance of multi-protocol Implementation Under Test(IUT), every layer of the multi-protocol IUT should be tested. According to ISO9646, single-layer test method is applied to testing the highest layer of multi-protocol IUT and single-layer embedded test method is used for the layers other than the highest layer because the interfaces between layers are not exposed. So far the conventional researches have focused on testing layer by layer all the protocols in a multi-protocol IUT.

This paper proposes a new method for testing a multi-protocol IUT. The proposed test method assumes that a multi-protocol IUT is under test and that the interfaces between the layers can not be controlled or observed by the tester. We apply the proposed test method to TCP/IP and compare the application results with those of the existing test methods in terms of various criteria such as the number of test cases, the number of test events and test coverage. It turns out that the proposed test method significantly reduces the number of test cases as well as the number of test events while providing the same test coverage. In addition, the proposed test method shows the capability to locate the layer that is the source of failure in testing multi-protocol IUTs.

† 정 회 원 : 한국전자통신연구원 표준시험연구팀 선임연구원
†† 정 회 원 : 한국정보통신대학원대학교 교수
††† 정 회 원 : 한국전자통신연구원 네트워크장비시험센터 센터장
논문접수 : 1999년 4월 19일, 심사완료 : 1999년 9월 27일

1) 시험경우(test case), 시험 이벤트(test event), behavior line에 대한 정의는 [1] 참조.

1. 서 론

프로토콜에 대한 적합성시험을 위한 방법론과 체계는 ISO/IEC JTC1을 통해 국제표준으로 제정되어 사용되고 있다[ISO9646]. 이 표준에서는 프로토콜에 대한 적합성시험의 개념, 시험방법과 시험 명세, 그리고 시험 실현과 제3자 시험을 위한 요구사항 등을 정의 하였으며, 시험 명세를 기술하기 위한 언어인 Tree and Tabular Combined Notation(TTCN)을 기술하고 있다. 이 표준은 단일 계층을 시험 대상으로 시험하는 것으로서 여러 계층으로 구성된 multi-protocol - 프로토콜의 집합 또는 프로파일을 시험하기 위해서는 단일 계층 시험을 반복하여 수행하는 것으로 규정하고 있으며 이를 "incremental testing of a multi-protocol IUT" 라고 한다.

표준 [1]에 의하면 프로토콜에 대한 시험은 특정 계층에 대하여 수행되고 있으며, 그 계층의 하위 계층은 서비스 제공자의 역할을 한다. 이러한 환경에서는 특정 계층에 대한 적합성 시험에서 오류가 발생하였을 경우 그 오류의 원인이 어느 계층에서 발생한 것인지를 알아내는 것이 어려우므로, 일반적으로 하위 계층들은 안정되어 있다고 가정하여 모든 오류의 원인은 시험대상 계층에 의한 것이라고 가정하고 시험을 진행한다.

multi-protocol에서 최상위를 제외한 프로토콜을 시험하는 방법으로 단일 계층 내포 시험방법이 사용된다. 이 방법에 의한 시험규격은 시험하고자 하는 계층과 그의 상위 계층들에 대한 프로토콜 행위를 포함하여 기술되나 프로토콜 간의 인터페이스에 대한 제어와 관찰은 포함되지 않는다. 즉 N_1부터 N_m 까지의 계층으로 구성된 시험대상에서 $N_i(1 <= i < m$, i는 정수) 계층을 시험하기 위한 시험규격은 N_i PDU 뿐만 아니라 N_{i+1} 부터 N_m 프로토콜의 PDU를 포함한다. 내포 시험방법의 시험 범위, 시험 시퀀스 생성 등에 대한 기존 연구([5], [6], [7], [8])가 있었으나 이들은 모두 multi-protocol IUT에서 단일 계층에 대한 시험에 초점을 맞추고 있다.

내포 시험방법은 "testing in context"라고도 하며 시험대상은 두개의 FSM으로 구성되어 있다고 할 때 시험대상 프로토콜에 대한 FSM을 "component"라고 하고 또 다른 하나의 FSM을 "context"라 한다. context는 시험의 대상이 아닌 프로토콜들의 집합으로서 시험기가 직접 제어하거나 관찰할 수 없는 내부 인터페이스를 통해 component와 서로 통신한다. 이러한 시험방법에서는 context에는 오류가 없는 것으로 가정한다. [5]과 [6]에서는 "testing in context"를 communicating FSM으로 모델링하고 시험을 생성하는 방안을 제시하였다. 내포 시험방법에서는 프로토콜 사이의 인터페이스를 직접 제어하거나 관찰할 수 없음으로 인하여 시험 커버리지에 많은 제약이 따른다. [7]에서는 오류 모델(fault model)을 기반으로 하여 내포 시험방법의 시험 커버리지를 평가하는 방법과 도구를 제안하였다. [8]에서는 내포 시험방법에 의한 시험 스위트의 최소화하는 방안을 제안하였다. 이러한 연구들은 모두 context에는 오류가 없다는 가정 하에서 단일 계층에 해당하는 component에 대한 시험에만 초점을 맞추고 있다.

본 논문에서는 multi-protocol IUT에서 상위 프로토콜에 적용하는 단일계층 시험방법과 하위 프로토콜에 적용되는 내포 시험방법을 조합하여 적용함으로써, 하나의 시험 스위트에서 두개 이상의 계층에 대한 시험을 동시에 수행하는 방법을 제안한다. 이 제안된 방법은 기존 방법에 비해 다음과 같은 장점을 갖는다.

- 최하위 PCO의 제어와 관찰을 통해 프로토콜 간의 인터페이스를 간접적으로 제어하고 관찰하여 내포된 프로토콜과 상위 프로토콜을 한꺼번에 시험한다.
- 두개의 프로토콜을 하나의 시험 스위트로 시험함으로써 시험규격 작성과 수행에 필요한 오버헤드를 줄인다.
- 프로토콜 사이의 인터페이스가 개방되지 않은 multi-protocol을 시험할 수 있다.

이 논문의 구성은 다음과 같다. 2장에서 multiprotocol 적합성 시험방법을 제시한다. 3장에서 이 방법을 TCP/IP 프로토콜에 적용한 예를 보이고 기존의 단일 계층 시험 그리고 내포 시험방법에 의한 시험과의 비교 분석을 기술한다. 마지막으로 4장에서 본 논문과 관련된 결론을 맺고 향후 계획을 기술한다.

2. Multi-protocol 적합성 시험방법

표준 [1]에 의하면 multi-protocol에 대한 시험은 단일 프로토콜에 대한 시험을 반복하여 수행한다. 이 경

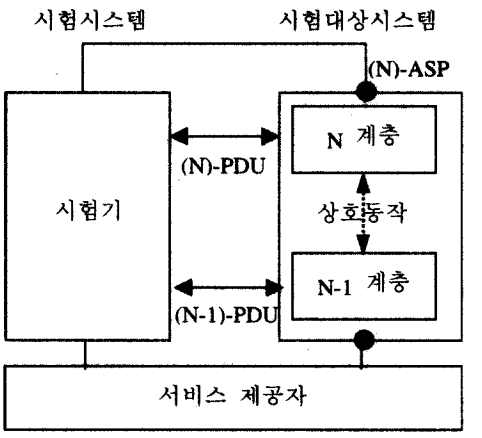
우 하나의 프로토콜을 시험할 때 그 이외의 다른 프로토콜에는 오류가 없는 것으로 가정하고 있다. 그러나 두개의 프로토콜을 동시에 시험하여 오류의 원인이 어느 프로토콜에 있는 것인지를 정확히 알 수 있다면 한 번의 시험으로 multi-protocol의 적합성 여부를 판단할 수 있을 것이다. 이것이 이 논문의 주요한 아이디어이다.

multi-protocol의 예로는 B-ISDN ATM Adaptation Layer(AAL) 계층을 들 수 있다[2]. AAL에서 최상위 계층인 Service Specific Coordination Function(SSCF) 아래에는 Service Specific Connection Oriented Protocol (SSCOP)과 Common Part Convergence Sublayer (CP-AAL)이 있으며 SSCOP의 인터페이스는 개방되어 있지 않고 SSCF와 CP-AAL을 통해서만 접근이 가능하다. 따라서 SSCOP에 대한 시험을 위해서는 내포 시험방법이 사용되어야 한다. multi-protocol의 또 다른 예로는 계층 2와 계층 3을 통합하여 성능을 개선하고자 하는 IETF의 Multiprotocol Label Switching(MPLS)을 들 수 있다[3]. MPLS를 구현한 상용 라우터에서 계층 2와 계층 3사이의 인터페이스는 개방되지 않으며, 따라서 이에 대한 시험을 위해서는 역시 내포시험방법이 사용되어야 한다.

앞에서 설명한 바와 같이 프로토콜 시험에 대한 기존의 연구들은 단일 계층의 시험을 목표로 하고 있다. 따라서 multi-protocol IUT의 시험에 대한 연구에서도 어느 한 프로토콜을 시험대상으로 하며 나머지 프로토콜은 오류가 없다고 가정하고 시험을 수행하므로 프로토콜 수 만큼의 시험 스위트를 필요로 한다.

multi-protocol에 대한 시험에서는 최상위와 최하위의 인터페이스만이 개방되어 있어서 프로토콜 간의 인터페이스에 대한 직접적인 제어와 관찰은 불가능하다. 그러나 multi-protocol의 행위는 최하위 인터페이스의 메시지 제어 및 관찰과 프로토콜 간의 상호동작의 관계에 의해 유추할 수 있다. 이러한 점에 착안하여 본 논문에서는 하나의 시험 시나리오에서 두개의 계층에 대한 시험을 동시에 수행하는 방안을 제안한다.

(그림 1)과 같은 시험 구성에서 N 계층 프로토콜과 N-1 계층 프로토콜에 대한 시험을 대상으로 본 논문에서 제안하는 multi-protocol 시험방법을 설명한다. 이 그림에서 N 계층의 상위와 N-1 계층의 하위 인터페이스는 개방되어 있으나 N 계층과 N-1 계층 사이의 인터페이스는 직접적으로 제어 및 관찰할 수 없다.



(그림 1) Multi-protocol 시험 대상의 시험 구성

(그림 1)과 같은 시험대상에 대해 ISO9646 표준에 의거한 기존의 시험방법에서는 N 계층 프로토콜과N-1 계층 프로토콜을 각각 시험한다. 즉 N 계층 프로토콜에 대해서는 단일 계층 시험방법을 사용하고 N-1 계층 프로토콜에 대해서는 내포 시험방법을 사용한다. N 계층의 단일 계층 시험방법에 의한 시험 이벤트는 (N)-ASP와 (N)-PDU로 기술되고, N-1 계층의 내포 시험방법에 의한 시험 이벤트는 (N)-ASP와 (N-1)-PDU로 기술된다.

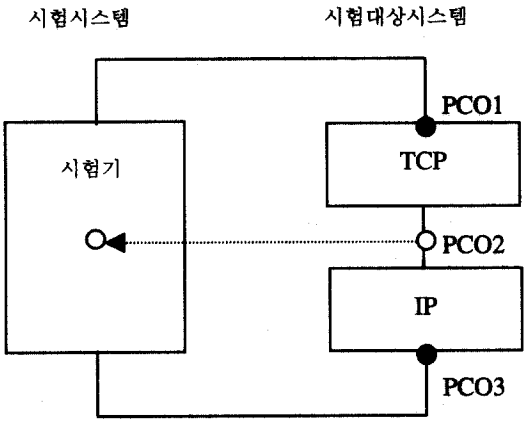
이 본문에서 제안하는 multi-protocol 시험방법을 위해서는 다음과 같은 환경이 설정된다.

- 각 계층의 프로토콜은 FSM으로 표현된다.
- multi-protocol 시험대상의 최상위 및 최하위 인터페이스는 개방되어 있다.

multi-protocol에 대한 시험방법에서는 상위 계층에 대한 단일 계층 시험방법과 하위 계층에 대한 내포 시험방법을 조합하여 시험 스위트를 표현한다. 따라서 (그림 1)과 같은 시험 구성에서 시험 이벤트는 (N)-ASP, (N)-PDU, (N-1)-PDU를 모두 사용하여 표현된다. 특히, (N)-PDU는 (N-1)-PDU의 사용자 데이터 필드에 포함되어 표현된다.

표준 [1]의 시험방법에서는 IUT의 상하위 인터페이스를 통해 IUT의 프로토콜 행위를 제어하고 관찰하여 시험을 진행하며 이 두개의 인터페이스를 PCO라 한다. 이 논문에서 제안한 multi-protocol 시험방법의 PCO에서는 한 PCO를 통해 입 출력되는 이벤트가 두개 계층

의 프로토콜 행위를 포함한다. 예를 들어 TCP/IP가 결합된 multi-protocol인 경우 IP의 프로토콜 행위에 대한 PCO내의 한 이벤트에는 TCP의 PDU가 포함되어 있으며, multi-protocol 시험방법에서는 IP PDU에 포함된 TCP PDU의 제어와 관찰도 수행할 수 있다. 따라서 multi-protocol 시험방법에서의 PCO 구성을 TCP/IP를 예로 하여 나타내면 (그림 2)와 같다.



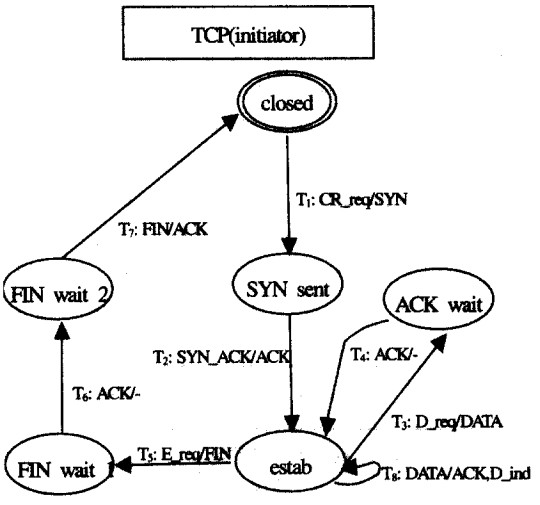
(그림 2) Multi-protocol 시험방법에서의 PCO

(그림 2)에서 PCO1과 PCO3는 직접적인 접근이 가능하다. 그러나 PCO2는 개방되지 않은 가상의 PCO이다. PCO1의 시험 이벤트는 TCP Abstract Service Primitive (ASP)로 표현되고 PCO3의 시험 이벤트는 IP PDU로 표현된다. IP PDU의 데이터(payload) 필드는 TCP PDU를 포함하고 있으므로 PCO3에서는 IP PDU에 대한 제어와 관찰 뿐만 아니라 PCO2에서 수행 할 수 있는 TCP PDU에 대한 제어와 관찰도 간접적으로 이루어 진다. 따라서 multi-protocol 시험방법에서는 2개의 PCO로써 3개의 PCO를 가지고 있는 것처럼 TCP와 IP의 프로토콜 행위를 동시에 시험 할 수 있다.

3. Multi-protocol 시험방법의 TCP/IP에 대한 적용

이 장에서는 TCP/IP multi-protocol에 대해 이 논문에서 제안한 multi-protocol 시험방법의 적용 예를 설명하고 이를 기존의 단일 계층 시험 그리고 내포 시험 방법에 의한 시험과 비교 분석한다.

3.1 간략화 한 TCP/IP의 FSM



트랜지션 표기법:

<트랜지션> := <식별자> : <입출력>
<식별자> := T_i, i는 정수
<입출력> := <입력> / <출력>
<입력> := <TCP ASP> | <TCP PDU> | -
<출력> := <TCP ASP> | <TCP PDU> | -
- : NULL
<TCP ASP> := CR_req | D_req | D_ind
<TCP PDU> := SYN | SYN_ACK | FIN | DATA | ACK

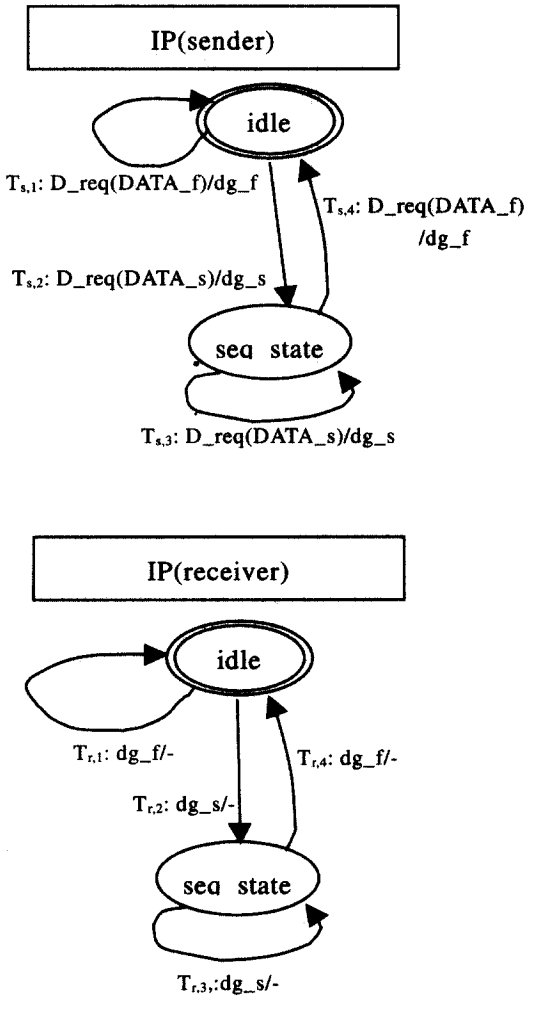
(그림 3) 간략화 한 TCP 프로토콜의 FSM

(그림 3)은 [4]의 TCP프로토콜에 대한 상태 기계를 간략화 하여 표현한 FSM이다. TCP는 연결 관리, 타이머 관리, 오류 제어를 통한 신뢰성 있는 데이터 전송, 흐름 제어 등을 제공한다. (그림 3)에서는 TCP의 연결 설정 기능과 사용자 데이터를 송수신하는 기능만을 표현하였으며, 연결 설정 기능에서의 타이머 관련 상태와 데이터 송신에서 오류에 대한 상태들은 이 논문에서는 생략하였다. 또한 이 논문에서는 TCP의 initiator만으로 multi-protocol 시험 방법을 설명하였다. TCP의 responder에 대해서도 initiator와 동일하게 시험을 기술할 수 있다.

IP는 연결 관리 기능이 없고 단순한 데이터 송수신

만을 지원하므로 상태(state)가 없이 표현될 수 있다. 그러나 IP에서 fragmentation을 하는 경우에는 IP 프로토콜에 상태를 부여하여 표현하는 것이 유용하다.

즉, TCP의 요청에 의해 IP가 데이터그램을 송수신하는 경우 TCP가 요청한 데이터의 크기에 따라 fragmentation을 수행하게 되고 이 경우 마지막 fragment와 그 이외의 fragment에 대한 프로토콜 행위가 달라지게 되므로 상태를 사용하는 것이 필요하다. (그림 4)는 이러한 IP 프로토콜을 FSM으로 표현한 것이다. (그림 4)에서 IP의 datagram은 fragmentation 여부에 따라 dg_f와 dg_s로 구분하였다. 즉 fragmentation이 필요 없거나 fragmentation 된 경우의 마지막 fragment는 dg_f에 해당하고 그 이외의 경우는 dg_s에 해당한다. 따라서



트랜지션 표기법:
<트랜지션> := <식별자> : <입출력>
<식별자> := $T_{<역할>, i}$, i는 정수
<역할> := s | r
* s : sender, r: receiver
<입출력> := <입력> / <출력>
<입력> := <IP ASP> | <IP PDU>
<출력> := <IP ASP> | <IP PDU>
<IP ASP> := D_req
<IP PDU> := dg_s | dg_f
* dg_s : fragmentation 한 경우 마지막을 제외한 fragment에 해당하는datagram
* dg_f : dg_s를 제외한 datagram

(그림 4) IP 프로토콜의 FSM

여기에서 사용하는 IP는 데이터그램 송신과 수신 그리고 fragmentation만을 제공한다.

위에서 제시한 TCP와 IP는 서로 연계하여 수행되며, TCP와 IP의 상호동작은 TCP의 PDU 크기에 따라 달라진다. 즉 (그림 3)의 SYN, SYN_ACK, FIN, ACK 등의 TCP PDU는 (그림 4)의 dg_f PDU에 포함되어 전달되고 DATA PDU는 fragmentation 필요여부에 따라 dg_f PDU 또는 dg_s PDU에 포함되어 전달된다. 따라서 TCP FSM의 트랜지션 T_3의 수행은 DATA의 크기에 따라 IP FSM의 트랜지션 $T_{s,1}$과 상호동작 하거나 트랜지션 $T_{s,2}$, $T_{s,3}$ 그리고 $T_{s,4}$와 상호동작 한다.

3.2 TCP FSM의 트랜지션에 대한 시험경우

이 절에서는 multi-protocol 시험방법에 의한 시험경우에 대해 설명한다. 이 논문에서 사용하는 시험경우 이름의 표기법은 다음과 같다.

<시험경우 이름> := <시험방법>_tc_<트랜지션>
<시험방법> := m | s | e
* m: multi-protocol 시험방법, s: 단일 계층 시험방법, e: 단일 계층 내포 시험방법
<트랜지션> := T_i | $T_{<역할>, i}$, i는 정수
<역할> := S | R
* S: sender, R: receiver

(그림 3)의 트랜지션 T_3와 (그림 4)의 IP FSM의 트랜지션과의 상호동작은 송신할 DATA의 크기에 따라 두 가지로 나타난다. 따라서 multi-protocol 시험방법에

서 (그림 3)의 트랜지션 T_3를 시험하기 위한 시험경우는 아래와 같이 두개(m_tc_T_{3_1}, m_tc_T_{3_2})이다. 이 논문에서는 간략화 한 TTCN을 사용하여 시험경우를 표현한다.

m_tc_T_{3_1}: 트랜지션 T_3에 대한 시험경우 - 1

```
+m_tc_T2 /* preamble to put the IUT to state estab */
  PCO1 ! D_req(DATA) /* small DATA, no fragmentation */
    PCO3 ? dg_f(DATA) /* PCO2 observation included */
```

m_tc_T_{3_1}에서 첫번째 behavior line은 IUT를 트랜지션 T_3를 수행할 수 있도록 "estab" 상태로 보내는 시험 스텝(test step)이다. 두번째 behavior line은 IP의 fragmentation이 필요 없는 작은 데이터를 송신하는 것으로 TCP FSM의 트랜지션 T_3의 수행을 위한 입력이다. 이 입력으로 TCP FSM의 트랜지션 T_3가 수행될 뿐만 아니라 세번째 behavior line에서 보이는 바와 같이 (그림 4)의 IP FSM의 트랜지션 $T_{s,1}$이 연계되어 수행된다. 이때 PCO3에서 dg_f의 수신을 관찰 할 뿐만 아니라 dg_f의 데이터 필드(payload)에 포함된 TCP의 DATA PDU를 조사함으로써 PCO2에 대한 관찰이 이루어 진다. 결국 이 시험경우는 서로 다른 프로토콜에 존재하는 두개의 트랜지션(TCP의 T_3와 IP의 $T_{s,1}$)을 한꺼번에 시험하게 된다. 이 상황을 정리하면 <표 1>과 같다.

〈표 1〉 m_tc_T3_1의 behavior line과 트랜지션의 관계

behavior	transition of TCP	transition of IP
PCO1 ! D_req(DATA)		
PCO3 ? dg_f(DATA)	T_3	$T_{s,1}$

m_tc_T_{3_2}: 트랜지션 T_3에 대한 시험경우 - 2

```
+m_tc_T2
  PCO1 ! D_req(DATA) /* large DATA, fragmentation */
    PCO3 ? dg_s(DATA_s) /* PCO2 observation included */
      PCO3 ? dg_s(DATA_s) /* PCO2 observation included */
        PCO3 ? dg_f(DATA_f) /* PCO2 observation included */
```

m_tc_T_{3_2}에서 두번째 behavior line은 IP의 fragmentation이 발생할 만큼 큰 데이터를 송신하는 것이다. 따라서 이 behavior line과의 상호동작에 의해 (그림 4)의 IP FSM의 트랜지션 $T_{s,2}$, $T_{s,3}$, $T_{s,4}$가 수행 되며 IP datagram 내의 TCP의 DATA PDU를 조사함으로써 PCO2에 대한 관찰이 이루어 진다. 이 시험경우에서 보이는 바와 같이 multi-protocol 시험방법에 의한 시험경우에서는 상위 계층의 하나의 자극(stimulus)으로 인해 하위계층에서 여러 번의 관찰을 하여야 하는 경우가 발생한다. 이 시험경우가 cover하는 트랜지션은 <표 2>에서 보이는 바와 같이 TCP FSM의 트랜지션 T_3와 IP FSM의 트랜지션 $T_{s,2}$, $T_{s,3}$, 그리고 $T_{s,4}$이다.

〈표 2〉 m_tc_T3_2의 behavior line과 트랜지션의 관계

behavior	transition of TCP	transition of IP
PCO1 ! D req(DATA)		
PCO3 ? dg f(DATA s)		Ts,2
PCO3 ? dg f(DATA s)		Ts,3
PCO3 ? dg f(DATA f)	T3	Ts,4

앞의 두 시험경우는 송신자 역할을 하는 TCP의 프로토콜 행위를 시험하는 것이다. 수신자 역할을 하는 TCP의 행위에 대한 시험은 (그림 3)의 트랜지션 T_8에 대한 시험경우로 설명될 수 있다. 아래의 시험경우는 IP에서 fragmentation이 필요한 큰 데이터를 TCP가 수신하는 경우이다.

m_tc_T_{8_2}: 트랜지션 T_8에 대한 시험경우

```
+m_tc_T2
  PCO3 ! dg_s(DATA_s)
    PCO3 ! dg_s(DATA_s)
      PCO3 ! dg_f(DATA_f) /* large DATA, fragmentation,
      PCO2 control included */
        PCO3 ? dg_f(ACK)
```

m_tc_T_{8_2}에서 TCP 트랜지션 T_8을 시험하기 위한 자극(stimulus)은 IP의 프로토콜 행위에 대한 3개의 behavior line으로 표현되었다. 그리고 이들 3개의 behavior line에 의한 자극의 결과는 TCP에서 ACK의 수신을 관찰하는 것으로 나타난다. 이 시험경우는 multi-protocol 시험방법에 의한 시험경우에서 하위 계층에 대한 여러 자극의 결과가 상위 계층에서는 하나의 관찰로 나타나는 예를 보여준다. 이 시험경우가 cover하는 트랜지션은 TCP FSM의 트랜지션 T_8과 IP FSM의 트랜지션 $T_{r,2}$, $T_{r,3}$, $T_{r,4}$ 그리고 $T_{s,1}$이다.

위에서 예를 든 multi-protocol 시험방법을 (그림 3)의 TCP FSM의 모든 트랜지션에 대해 적용한 결과는 부록 1에 나타내었다.

3.3 단일 계층 시험에 의한 TCP 시험경우

이 절에서는 단일계층 시험방법에 의한 시험경우에

대해 설명한다. (그림 2)의 시험 구조에서 최상위 프로토콜 TCP는 시험기가 상하위 인터페이스를 모두 접근할 수 있으며 단일 계층 시험방법으로 시험할 수 있다[1]. 단일 계층 시험방법에서는 IUT의 상하위 인터페이스에 해당하는 두개의 PCO만이 존재하며 (그림 2)에서 TCP를 시험하는 경우 PCO는 PCO1과 PCO2이다. (그림 3)의 트랜지션 T_3를 시험하기 위한 시험경우는 아래와 같다.

s_tc_T_3: 트랜지션 T_3에 대한 시험경우

```
+s_tc_T2
  PCO1 ! D_req(DATA)
    PCO2 ? DATA
```

여기서는 TCP의 프로토콜 행위만을 제어하고 관찰하므로 IP의 동작은 다루지 않는다. 따라서 3.2절의 m_tc_$T_{3,2}$에서 사용되었던 dg_f, dg_s 등의 IP PDU는 사용되지 않는다. s_tc_T_3에서 두번째 behavior line은 DATA의 크기에 따라 IP의 fragmentation이 발생될 수도 있다. 그러나 IP에서는 reassembly 이후의 최종적인 DATA 만을 TCP에 전달하게 된다. 따라서 IP의 행위를 제어하거나 관찰하지 않는 단일 계층시험방법에서는 multi-protocol 시험경우와 달리 IP FSM의 트랜지션 $T_{s,1}$ 또는 트랜지션 $T_{s,2}$, $T_{s,3}$, $T_{s,4}$에 대한 관찰이 이루어지지 않는다.

위에서 예를 든 단일 계층 시험시험방법을 (그림 3)의 TCP FSM의 모든 트랜지션에 대해 적용한 결과는 (부록 2)에 나타내었다.

3.4 내포 시험방법에 의한 IP 시험경우

내포시험방법에 의한 IP FSM에 대한 시험경우의 예는 다음과 같다. 내포 시험방법에서 사용되는 PCO는 (그림 2)의 시험구성에서 PCO1과 PCO3이다.

e_tc_$T_{s,2}$: 트랜지션 $T_{s,2}$에 대한 시험경우

```
+e_tc_Ts,1
  PCO3 ! dg_f(SYN_ACK)
    PCO3 ? dg_f(ACK)
      PCO1 ! D_req(DATA) /* large DATA, fragmentation */
        PCO3 ? dg_s(DATA_s)
```

위의 시험경우는 IP를 시험하기 위한 것이다. 그러나 두번째 또는 세번째 behavior line과 같이 IP를 주어진 상태로 이동시키기 위해서 TCP 계층의 행위를 시험경우에 나타내어야 할 필요가 있다.

위에서 예를 든 단일 계층 내포 시험시험방법을 (그림 4)의 IP FSM의 모든 트랜지션에 대해 적용한 결과는 부록 3에 나타내었다.

3.5 multi-protocol 시험방법과 기존 시험방법의 비교

위의 절에서 각각 기술한 시험경우에서 보이는 바와 같이 multi-protocol 시험방법에 의한 시험경우는 단일 계층 시험방법에 의한 시험경우와 내포 시험방법에 의한 시험경우를 포함하며 그 관계는 다음과 같다.

$m\_tc\_T_1 = s\_tc\_T_1 + e\_tc\_T_{s,1}$
$m\_tc\_T_2 = s\_tc\_T_2 + e\_tc\_T_{s,1} + e\_tc\_T_{r,1}$
$m\_tc\_T_{3\_1} = s\_tc\_T_3 + e\_tc\_T_{s,1}$
$m\_tc\_T_{3\_2} = s\_tc\_T_3 + e\_tc\_T_{s,2} + e\_tc\_T_{s,3} + e\_tc\_T_{s,4}$
$m\_tc\_T_4 = s\_tc\_T_4 + e\_tc\_T_{s,1}$
$m\_tc\_T_5 = s\_tc\_T_5 + e\_tc\_T_{s,1}$
$m\_tc\_T_6 = s\_tc\_T_6 + e\_tc\_T_{r,1}$
$m\_tc\_T_7 = s\_tc\_T_7 + e\_tc\_T_{r,1} + e\_tc\_T_{s,1}$
$m\_tc\_T_{8\_1} = s\_tc\_T_8 + e\_tc\_T_{r,1} + e\_tc\_T_{s,1}$
$m\_tc\_T_{8,2} = s\_tc\_T_8 + e\_tc\_T_{r,2} + e\_tc\_T_{r,3} + e\_tc\_T_{r,4} + e\_tc\_T_{s,1}$

위의 시험경우의 관계 중에서 multi-protocol 시험방법에 의한 시험경우 m_tc_$T_{3,2}$는 단일 계층시험 방법에 의한 TCP 시험경우 s_tc_T_3와 단일 계층 내포시험방법에 의한 IP 시험경우 e_tc_$T_{s,2}$, e_tc_$T_{s,3}$, e_tc_$T_{s,4}$를 포함한다. 이러한 시험경우의 관계를 각 방법에 대해 시험경우 수, behavior line의 수, 이벤트 수 그리고 시험 적용범위를 정리하면 다음과 같다.

〈표 3〉 각 방법에 의한 시험경우 비교

비교기준 \ 시험방법	제안된 다중계층 프로토콜 시험방법	기존 시험방법		
		TCP에 대한 단일 계층 시험방법	IP에 대한 단일 계층 내포 시험방법	기존 두 방법의 합
시험경우 수	10	8	8	16
Behavior lines 수	32(2+3+3+5+3+3+2+3+3+5)	22	23	45
이벤트 수	64(2+4+6+8+8+6+7+9+6+8)	48	47	95
시험 커버리지	모든 트랜지션			모든 트랜지션

위의 표에서 보이는 바와 같이 multi-protocol 시험방법은 단일 및 내포 시험방법과 비교할 때 동일한 시험 커버리지를 제공하면서도 시험경우 수가 현저히 줄어드는 것을 알 수 있다. 또한 시험경우에 포함된 behavior line의 수가 적어지므로 시험경우의 크기가 작아지고 시험경우를 작성하는 오버헤드(overhead)가 줄어들며, 시험경우에 포함된 이벤트의 수도 적어지므로 시험수행 시간을 줄일 수 있는 장점이 있다. 이러한 기존 시험방법의 오버헤드는 상위 프로토콜에 대한 단일 계층 시험에서 수행되는 하위 계층 FSM의 트랜지션이 하위 계층에 대한 내포 시험방법에서 중복되어 수행되기 때문이다. 그러나 multi-protocol 시험방법에서는 두 계층에 대한 시험을 한꺼번에 수행함으로써 트랜지션의 중복 수행을 줄일 수 있고 그 결과로 시험경우에서 behavior line과 시험 이벤트의 수가 줄어들게 된다.

multi-protocol 시험방법의 또 다른 장점으로는 정확한 오류 위치를 알아낼 수 있다는 것이다. 기존 시험방법에서는 시험대상 계층의 하위 계층들은 안정되어 있어서 모든 오류의 원인은 시험대상 계층에 의한 것이라고 가정하고 시험을 진행한다. 실제로 오류는 하위계층으로 인해 발생할 수도 있으며, multi-protocol 시험방법에서는 이 경우에 어떤 계층에서 오류가 발생하였는지를 정확히 밝혀 낼 수 있다.

multi-protocol 시험방법의 단점으로는 하위 계층 프로토콜의 오류에 대한 대응 행위를 시험할 수 없다는 것을 들 수 있다. 이는 시험기가 시험대상 프로토콜의 인터페이스를 직접 접근할 수 없고 상위 계층 프로토콜을 경유하여 시험 대상 프로토콜을 간접적으로 제어 및 관찰하기 때문에 발생하는 문제점으로 내포 시험방법에서도 동일하게 존재하는 문제이다.

4. 결론 및 향후 계획

본 논문에서는 multi-protocol로 구성된 프로토콜을 시험하기 위한 multi-protocol 시험방법을 제안하였다. 이 방법은 기존의 단일 계층 시험방법과 내포 시험방법을 조합하여 사용함으로써 하나의 시험스위트로써 두개의 프로토콜로 구성된 시험대상을 시험할 수 있다. multi-protocol 시험방법을 사용할 때의 구체적인 시험경우 작성 방법을 설명하기 위해 간략화 한 TCP/IP 프로토콜에 적용하고 기존 시험방법에 의한 시험경우와 비교 분석하였다. multi-protocol 시험방법을 사용하면 기존의 단일 계층 시험방법과 내포 시험방법을 사용하여 두개의 프로토콜을 별도로 시험하는 방법에 비해 시험경우의 수와 behavior line의 수 그리고 시험 이벤트의 수가 줄어들게 된다. 또한 제안된 시험방법은 multi-protocol의 어떤 계층에서 오류가 발생하였는지를 정확히 밝혀 낼 수 있다.

이 논문에서는 두개의 계층을 가지는 multi-protocol 시험대상에 대한 multi-protocol 시험방법의 적용 만을 보였다. multi-protocol 시험방법을 세 개 이상의 계층으로 이루어진 multi-protocol 시험대상에 대해 적용할 수 있도록 일반화하는 것이 향후의 연구과제가 될 것이다. ATM AAL, MPLS 등 다수의 프로토콜이 통합된 multi-protocol에 실제로 적용하기 위한 연구가 필요하다. 또한 한 FSM의 여러 트랜지션과 다른 FSM의 여러 트랜지션이 서로 혼합되어 수행되는 경우 등 프로토콜 간의 복잡한 상호 동작에 대한 형식적인 정의와 시험 방안에 대한 연구가 필요하다.

부 록

1. Multi-protocol 시험방법에 의한 TCP 시험경우

m_tc T_1: test case for T_1

```
PCO1 ! CR_req
  PCO3 ? dg_f(SYN) /* PCO2 observation included */
```

covered transitions: T_1, $T_{s,1}$

m_tc T_2: test case for T_2

```
+m_tc_T1
  PCO3 ! dg_f(SYN_ACK) /* PCO2 control included */
    PCO3 ? dg_f(ACK) /* PCO2 observation included */
```

covered transitions: T_2, $T_{r,1}$, $T_{s,1}$

m_tc $T_{3,1}$: test case for $T_{3,1}$

```
+m_tc_T2
  PCO1 ! D_req(DATA) /* small DATA, no fragmentation */
    PCO3 ? dg_f(DATA) /* PCO2 observation included */
```

covered transitions: T_3, $T_{s,1}$

m_tc $T_{3,2}$: test case for $T_{3,2}$

```
+m_tc_T2
PCO1 ! D_req(DATA) /* large DATA, fragmentation */
  PCO3 ? dg_s(DATA_s) /* PCO2 observation included */
    PCO3 ? dg_s(DATA_s) /* PCO2 observation
```

included */
PCO3 ? dg_f(DATA_f) /* PCO2 observation included */

covered transitions: T_3, $T_{s,2}$, $T_{s,3}$, $T_{s,4}$

m_tc_T_4: test case for T_4
+m_tc_T_3
PCO1 ! D_req(ACK)
PCO3 ? dg_f(ACK) /* PCO2 observation included */

covered transitions: T_4, $T_{s,1}$

m_tc_T_5: test case for T_5
+m_tc_T_2
PCO1 ! E_req
PCO3 ? dg_f(FIN) /* PCO2 observation included */

covered transitions: T_5, $T_{s,1}$

m_tc_T_6: test case for T_6
+m_tc_T_5
PCO3 ! dg_f(ACK) /* PCO2 control included */

covered transitions: T_6, $T_{r,1}$

m_tc_T_7: test case for T_7
+m_tc_T_6
PCO3 ! dg_f(FIN) /* PCO2 control included */
PCO3 ? dg_f(ACK) /* PCO2 observation included */

covered transitions: T_7, $T_{r,1}$, $T_{s,1}$

m_tc_$T_{8,1}$: test case for $T_{8,1}$
+m_tc_T2
PCO3 ! dg_f(DATA) /* small DATA, no fragmentation, PCO2 control included */
PCO3 ? dg_f(ACK)

covered transitions: T_8, $T_{r,1}$, $T_{s,1}$

m_tc_$T_{8,2}$: test case for $T_{8,2}$
+m_tc_T_2
PCO3 ! dg_s(DATA_s)
PCO3 ! dg_s(DATA_s)
PCO3 ! dg_f(DATA_f) /* large DATA, fragmentation, PCO2 control included */
PCO3 ? dg_f(ACK)

covered transitions: T_8, $T_{r,2}$, $T_{r,3}$, $T_{r,4}$, $T_{s,1}$

2. 단일 계층 시험방법에 의한 TCP 시험경우

s_tc_T_1: test case for T_1
PCO1 ! CR_req
PCO2 ? SYN
covered transitions: T_1

s_tc_T_2: test case for T_2
+s_tc_T_1
PCO2 ! SYN_ACK
PCO2 ? ACK
covered transitions: T2

s_tc_T_3: test case for T_3
+s_tc_T_2
PCO1 ! D_req(DATA)
PCO2 ? DATA
covered transitions: T3

s_tc_T_4: test case for T_4
+s_tc_T_3
PCO1 ! D_req(ACK)
PCO2 ? ACK
covered transitions: T4

s_tc_T_5: test case for T_5
+s_tc_T_2
PCO1 ! E_req
PCO2 ? FIN
covered transitions: T5

s_tc_T_6: test case for T_6
+s_tc_T_5
PCO2 ! ACK
covered transitions: T6

s_tc_T_7: test case for T_7
+m_tc_T_6
PCO2 ! FIN
PCO2 ? ACK
covered transitions: T7

s_tc_T_8: test case for T_8
+s_tc_T_2
PCO2 ! DATA
PCO2 ? ACK
covered transitions: T8

3. Embedded 시험방법에 의한 IP 시험경우

e_tc_$T_{s,1}$: test case for $T_{s,1}$
PCO1 ! CR_req
PCO3 ? dg_f(SYN)

e_tc_$T_{s,2}$: test case for $T_{s,2}$
+e_tc_$T_{s,1}$
PCO3 ! dg_f(SYN_ACK)

PCO3 ? dg_f(ACK)
PCO1 ! D_req(DATA) /* large DATA, fragmentation */
PCO3 ? dg_s(DATA_s)

e_tc_$T_{s,3}$: test case for $T_{s,3}$
+e_tc_$T_{s,2}$
PCO3 ? dg_s(DATA_s)

e_tc_$T_{s,4}$: test case for $T_{s,4}$
+e_tc_$T_{s,3}$
PCO3 ? dg_f(DATA_f)

e_tc_$T_{r,1}$: test case for $T_{r,1}$
PCO3 ! dg_f(SYN)
PCO1 ? CR_ind

e_tc_$T_{r,2}$: test case for $T_{r,2}$
+e_tc_$T_{r,1}$
PCO1 ! CR_resp
PCO3 ? dg_f(SYN_ACK)
PCO3 ! dg_f(ACK)
PCO3 ! dg_s(DATA_s)

e_tc_$T_{r,3}$: test case for $T_{r,3}$
+e_tc_$T_{r,2}$
PCO3 ! dg_s(DATA_s)

e_tc_$T_{r,4}$: test case for $T_{r,4}$
+e_tc_$t_{r,4}$
PCO3 ! dg_f(DATA_f)
PCO1 ? DATA_ind

참 고 문헌

[1] ISO 9646, "Information Technology OSI Conformance Testing Methodology and Framework," 1992.

[2] ITU-T Recommendation Q.2110, "B-ISDN ATM Adaptation Layer Service Specific Connection-Oriented Protocol(SSCOP)," 1994.

[3] IETF draft-ietf-mpls-framework-02.txt, "A Framework for Multiprotocol Label Switching," 1997.

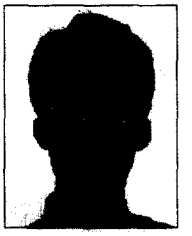
[4] W. Richard Stevens, TCP/IP Illustrated, Addison-Wesley, 1994.

[5] Alexandre Petrenko, Nina Yevtushenko, Gregor v. Bochmann and Rachida Dssouli, "Testing in context : framework and test derivation," Computer Communications 19, pp.1236-1249, 1996.

[6] Alexandre Petrenko and Nina Yevtushenko, "Fault detection in embedded components," IWTCS '97, pp.272-287, Cheju Island, Korea, 1997.

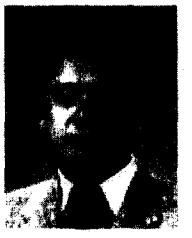
[7] Jinsong Zhu, Son T. Voung and Samuel T. Chanson, "Evaluation of test coverage for embedded system testing," IWTCS'98, pp.111-126, Tomsk, Russia, 1998.

[8] Nina Yevtushenko and Ana Cavali, "Test suite minimization for testing in context," IWTCS '98, pp.127-145, Tomsk, Russia, 1998.



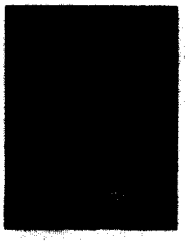
박 용 범

e-mail : ybpark@netc.etri.re.kr
1986년 경북대학교 전자공학과 졸업(학사)
1989년 한국과학기술원 전산학과 졸업(석사)
1995년~1996년 미국 국립표준연구원 객원연구원
1989년~현재 한국전자통신연구원 선임연구원
관심분야 : protocol engineering, wireless LAN, and mobile computing



김 명 철

e-mail : mckim@icu.ac.kr
1982년 아주대학교 전자공학과 졸업(학사)
1984년 한국과학기술원 전산학과 졸업(석사)
1993년 University of British Columbia 전산학과 졸업(박사)
1984년~1997 한국통신 연구개발단
1998년~현재 한국정보통신대학원대학교 교수
관심분야 : Internet, protocol engineering, telecommunications, and mobile computing



김 장 경

e-mail : jkkim@netc.etri.re.kr

1980년 연세 대학교 전자 공학과 졸업 (학사)

1989년 Iowa State University Computer Engineering (공학석사)

1992년 Iowa State University Computer Engineering (공학박사)

1980년~1986 국방 과학 연구소 연구원

1994년~1995 미국 University of Maryland 파견 국제 공동 연구 수행

1992년~현재 한국전자통신연구원 네트워크장비시험센터 센터장

관심분야 : 통신망 프로토콜 상호 연동 표준, 정보통신 표준 시험 기술, High Speed Network