HDLC 프로토콜 기반 DLMS 트래픽 모니터링 시스템

송병권*, 김새벽**, 김경헌0, 정태의***, 김진철****, 김영억****
*서경대학교 정보통신공학과, 0서경대학교 정보통신공학과,
**서경대학교 전자 컴퓨터 공학과,
***서경대학교 컴퓨터과학과,
***한전kdn(주) 정보통신연구그룹

e-mail: bksong@skuniv.ac.kr*, realgb@nate.com**, rudebrain@gmail.com0
tejeong@skuniv.ac.kr***, kjc@kdn.com****, yekim@kdn.com****

DLMS Traffic Monitoring System base on HDLC Protocol

Byung-Kwon Song*, Sae-Byuk Kim**, Kyeong-Hun Kim0, Tae-Eui Jeong***,
*Information Communication Engineering, SeoKyeong University,
0Information Communication Engineering, SeoKyeong University,
**Dept of Electronic and Computer Engineering, Seokyeong University,
***Computer Science, SeoKyeong University,
****Information Communication Research group

요 약

기존의 전력량계의 원격 검침을 위한 DLMS/COSEM(Device Language Message Specification / Companion Specification for Energy Meters) 시스템에서의 실제 송, 수신되는 데이터를 수집하여 이를 표현 및 분석 할 수 있는 프로그램을 구현하여 이로 하여금 실제 검침원에게 시스템에서의 송, 수신 되는 데이터의 흐름을 한 눈에 볼 수 있게 하였다. 또한 데이터의 실제 값들을 분석하여 이들을 가시적으로 볼 수 있어 오류 발생 시에 세부적으로 데이터를 분석하여 오류 원인을 찾아 낼 수 있다.

1. 서론

실생활 내에서의 프로토콜의 쓰임새는 다양하다. 프로토콜은 실생활 어디에서나 통신이라는 단어가 언급만 되어도 그 존재가 필요 있다고 생각해도 무방하다. 예를 들어 흔히 우리가 사용하는 핸드폰이나 인터넷 전화서만 보아도 프로토콜은 존재하며, 이는 통신 산업이 발전하면 할수록 필수적 요소임에 틀림없다.

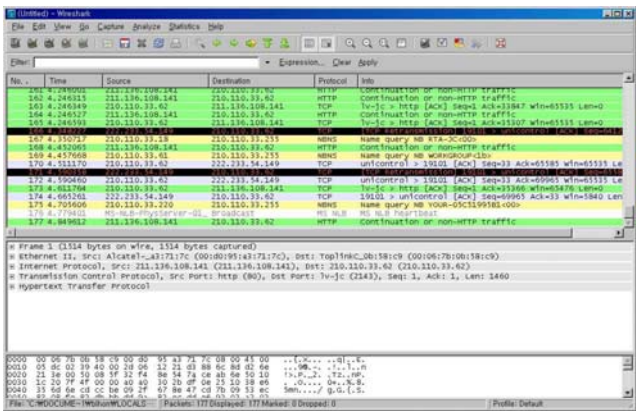
이러한 통신 프로토콜의 쓰임은 여러 산업 분야에도 적용이 된다. 산업 전력 IT 분야에서는 각 장비들의 원격 검침을 위하여 다양한 프로토콜을 사용하고 있다. 그 중에서 전기 계량기 값을 검침하기 위해서 쓰이는 DLMS(Device Language Message Specification) 방식이 그것이다.

이에 COSEM(Companion Specification for Energy Meters) 인터페이스를 적용한 DLMS/COSEM 통신 프로토콜은 IEC 62056 국제 규격을 기반으로 하는 차세대 전력량계 통신 프로토콜로서 현재 전 세계적으로 각광을 받고 있다. 이 프로토콜은 전기는 물론 가스와 수도 등에도 적용되고 있으며, 유럽에서는 전력량계의 표준 프로토콜로 점차 확대되고 있는 추세에 있다[1].

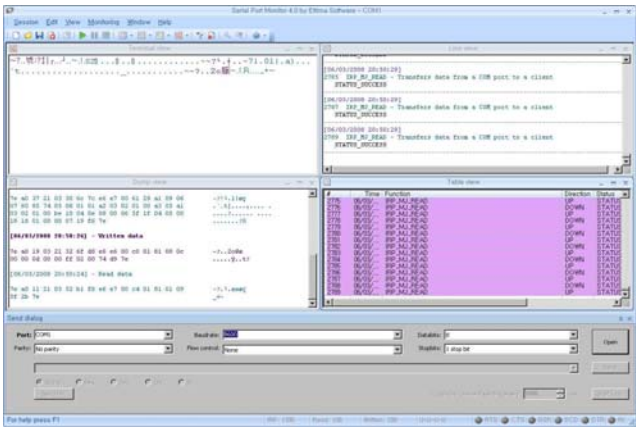
※ 본 연구는 한전KDN(주) 위탁연구 및 중기청 산학협력 연구비로 수행되었음.

본 논문에서는 이러한 DLMS/COSEM 통신 프로토콜에서의 송, 수신되는 데이터 값들은 많은 정보를 포함하고 있다. 이러한 정보는 실제 사용자에게 직접적으로 보이지 않은 채 정해진 결과만을 표현해 준다. 이러한 결과 이전에 실제 사용되는 데이터를 획득, 이를 분석함으로써 전문가에게 있어서 좀 더 신뢰있는 흐름을 보여 줄 수 있으며, 또한 오류 발생 시에 이를 분석함으로써 빠른 대응이 가능하다.

2. 관련 연구



(그림 1) Wireshark

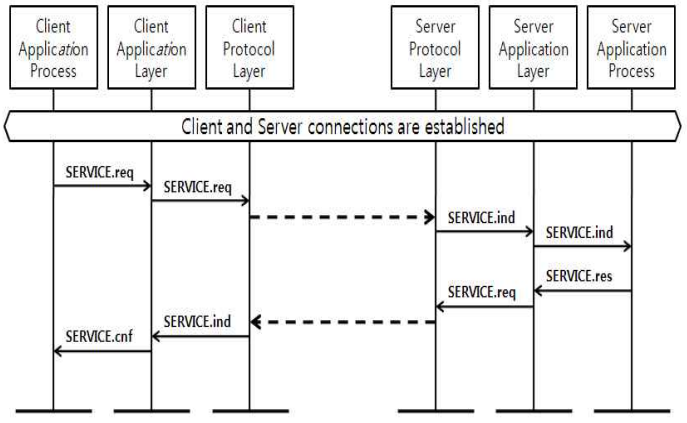


(그림 2) Serial port monitor

현재 가장 많이 쓰이는 트래픽 모니터링 시스템으로는 Wireshark(그림 1), Serial port monitor(그림 2) 등이 있다. (그림 1)의 WireShark는 이더넷상의 거의 모든 패킷을 캡처해 이를 출력, 분석 해 주는 툴이다. 하지만 WireShark 에서는 단지 이더넷 상의 패킷만을 캡쳐 및 분석을 지원해준다는 단점이 있다. 본 논문에서 표현하고자 하는 DLMS/COSEM 인터페이스에서의 TCP/IP 프로토콜을 사용하는 구조는 표현함에 있어서 문제가 없어 보이나 이를 분석하는 기능이 없으며, 또한 HDLC 프로토콜의 RS-232C cable 을 통한 통신을 지원하지 않는다. 따라서 WireShark 에서의 HDLC 프로토콜의 패킷 캡쳐 링은 불가능하다는 단점이 존재한다.

이더넷 기반이 아닌 serial cable 환경에서의 트래픽 모니터링 프로그램으로는 (그림 2)의 Eltimasoft사의 Serial Port Monitor 프로그램이 있다. 이 Serial Port Monitor 프로그램은 시리얼 통신상의 실제 트래픽을 수집 및 분석을 해 준다. 하지만 Serial Port Monitor 프로그램은 특정 프로토콜에 실제 데이터를 ASCII 문자로 표현해 주는 기능이 고작이다. 본 논문에서 구현한 프로그램은 위에서 설명한 두 가지 프로그램을 HDLC 프로토콜 분석에서 가지는 여러 가지 단점을 보완한 결과를 보여주려 한다.

3. DLMS/COSEM system



(그림 2) DLMS/COSEM Service Primitive

위의 (그림 2)에서는 양족 끝에 위치한 Application process 간의 기본적인 송, 수신 동작을 논리적인 흐름으로 표현하고 있다. COSEM 인터페이스 클래스를 사용하여 이루어지는 전기계량 장치와의 통신은 클라이언트/서버 패러다임을 기초로 한다. 이러한 환경에서의 통신은 결과적으로 클라이언트와 서버 Application 프로세스 간의 통신이다. 이 서비스는 클라이언트와 서버 어플리케이션 프로세스 간의 메시지 교환(Service.requests/.responses)을 통해 제공된다. 이러한 Application process 의 명령들은 하위 층 프로토콜의 Application layer와의 연결이 이루어지면서 클라이언트 응용계층의 2개의 어플리케이션 서비스 요소(ACSE, xDLMS)의 도움을 받아 APDU로 조합된다[2]. 이러한 APDU는 다음의 프로토콜 환경에 의해 각각의 헤더가 붙게 된다. 해당 ADPU는 IEC 61334-4-41에 정의된 내용을 따른다[3]. 대표적인 APDU 형태로는 COSEM APDU에 속하는 AARQ, AARE 가 있으며, ACSE APDU에 속하는 RLRQ, RLRE등이 있다.

3. HDLC Protocol Format

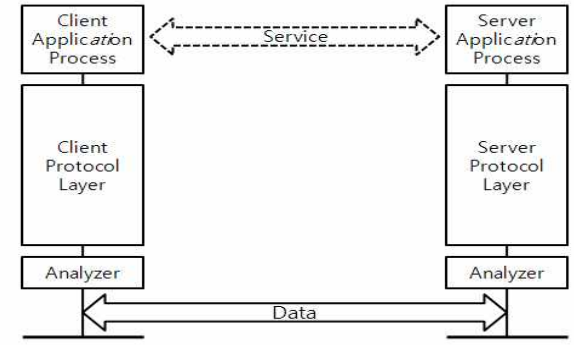
Flag	Frame format	Dest. Address	Src. Address	Control	HCS	Information	FCS	Flag
1	2	1, 2, 4	1	1	2		2	1

(그림 3) HDLC 프레임의 구조(Frame Type 3)

(그림 3)에서는 Frame Type3의 형태를 지니는 HDLC 프로토콜 프레임의 구조를 보여주고 있다. 시작과 마지막의 Flag Field는 비동기 전송의 그것과 같이 $01111110_{(2)}$의 값을 지니게 된다. HDLC 프레임 헤더는 비동기 시작과 끝을 알리는 Flag, 프레임 포맷과 프레임 길이를 알리는 FrameFormat, Address Field, 상태제어 목적을 가지는 Control, 오류제어를 위한 checksum field 가 있다. HDLC에서는 Information field 부분에 실제 APDU가 대입이 된다.

4. DLMS/COSEM monitering program

4.1 HDLC Monitoring 과정



(그림 4) 분석 과정

(그림 4)에서는 Application process 로부터 명령이 내려지는 순간부터 전송이 되는 순간까지의 대략적인 통신 흐름을 보여주고 있다. 실제 사용자가 느끼기에는 Client Applicaion Process 로부터 Server Applicaion Prosess 까지 바로 서비스 프리미티브가 전달되는 것처럼 느껴지게 하기 위해서 Protocol Layer를 지나서 물리계층에서 직접적으로 데이터가 전달되어야 한다.

4.3 Traffic Monitor Architecture

다음은 C++ 형태를 닮은 의사코드(pseudo code)로 표현된 분석기 구조이다.

입력 : 완성된 Packet(or Frame) PACK,
출력 : Packet(PACK) 의 분석 화면.

```
/*트래픽 모니터: 실제 송, 수신되는 데이터 값을 표현*/
void Monitor(Packet PACK)
{
        if( read(F) == TRUE )
                Print the PACK packet.;
                Analyzer(PACK);
        else wait to packet
}

/* 패킷 분석기 : 패킷(or 프레임)을 읽은 뒤 직접 분석한다. */
void Analyzer(Packet PACK)
{
  if( PACK == TCPformat )
    Cut off the TCP Header.
    print the WPDU Header information and
    Cut off it.
    AnalyzeAPDU( PACK );
  else if ( PACK == UDPformat )
    Cut off the UDP Header.
    print the WPDU Header information and
     Cut off it.
    AnalyzeAPDU( PACK );
  else if ( PACK == HDLCformat )
    print the HDLC frame header information and
    Cut off it.
    AnalyzeAPDU( PACK );
  else return False;
}

/* APDU 분석기 : 헤더정보를 떼어낸 APDU를 분석하여 표현한다. */
AnalyzeAPDU( Packet PACK)
{
  APDUtype(PACK);//first byte is APDU Tag.
    print another APDU field informations
    by APDU type.
}
```

4.4 Monitoring

```
7E A0 2F 03 21 10 17 DD E6 E6 00 60 21 80 02 02 84 A1 09 06 07 60 85 74 05 08 01
 01 BE 10 04 0E 01 00 00 00 06 5F 1F 04 00 00 18 98 04 00 A9 32 7E

 ---- Send     ----
7E : Start flag
A0 : FrameType
2F : FrameLength
03 : Destination address
21 : Source address
10 : Control field
17 DD : HCS(header check sequence)

 ** Information Field **
E6 : Format Identifier
E6 : Group Identifier
00 : Group Length

60 : AARQ Tag
21 : Length
80 02 02 84 : Version
A1 09 06 07 60 85 74 05 08 01 : app_ctxt_name
01 : app_ctxt
BE 10 04 0E 01 00 00 00 06 5F 1F 04 00 : xdlms_ctxt
00 18 98 : Conformance
04 00 : max_apdu_size_recv

A9 32 : FCS(Frame check sequence)
7E : End flag
```

(그림 5) Send APDU analyze view

(그림 5)에서는 HDLC 에서의 AARQ를 전송한 후의 실제 형성된 데이터 프레임의 형태와 각 field 부분을 실제 데이터에 대비하여 분석된 화면을 보여주고 있다. (그림 5)에서의 순서는 실제 프레임의 첫 바이트로부터의 오름차순으로 분석된 화면이며, AARQ Tag를 통해 해당 인자들의 데이터 값들의 의미를 표기하였다.

```
7E A0 37 21 03 30 6C 7C E6 E7 00 61 29 A1 09 06 07 60 85 74 05 08 01 01 A2 03 02
 01 00 A3 05 A1 03 02 01 00 BE 10 04 0E 08 00 06 5F 1F 04 00 00 18 18 01 00 00 0
7 19 F6 7E

 ---- Receive    ----
Type : HDLC Frame
7E : Start flag
A0 : FrameType
37 : FrameLength
21 : Destination address
03 : Source address
30 : Control field
6C 7C : HCS(header check sequence)

 ** Information Field **
E6 : Format Identifier
E7 : Group Identifier
00 : Group Length

61 : AARE Tag
29 : Length
09 06 07 60 85 74 05 08 01 : app_ctxt_name
01: Context
A2 03 02 01 : asso_result
00 : xdlms check
A3 05 A1 03 02 01 : source_diag
00 : reason
BE 10 04 0E 08 00 06 5F 1F 04 00 : user info
00 18 18 : ConfBlock
01 00 : InfoSize
00 07 : Context

19 F6 : FCS(Frame check sequence)
7E : End flag
AARQ Success
```

(그림 6) Receive APDU analyze view

(그림 6)에서는 AARQ전송후의 응답인 AARE의 전송 화면을 보여준다. (그림 5)와 마찬가지로 실제 수신된 데이터를 바이트 단

위로 출력하였으며, AARE Tag를 통해 해당 인자들의 데이터 값들의 의미를 표기 하였다.

5. 결론 및 향후 연구

실제 DLMS/COSEM이 적용이 되는 전력량계의 원격 검침분야에 있어서는 적은 오류율이 보장되는 시스템이 구축되어야 할 것이다. 하지만 어느 시스템이든지 실제 구동 시에는 수도 없이 많은 여러 환경과 맞부딪쳐야 할 것이다. 실제 프로토콜을 설계 및 분석하는 학생들은 프로토콜을 분석하기 위하여 WireShark등과 같은 프로토콜 분석기를 사용하여 트래픽을 모니터링 한다. 어느 프로토콜이던지 실제 데이터를 보는 것만큼이나 확실한건 없다고 생각하며, 그러한 관점에서의 DLMS/COSEM 시스템의 트래픽 모니터링 프로그램은 반드시 필요하다.

본 논문에서 구현된 프로그램은 관련연구에서 보인 WireShark나 Serial port monitor 에 비해서 보이는 면이 적다고 생각하나 이 두 트래픽 모니터 프로그램이 가지지 못한 부분을 채워보았다. 좀 더 연구가 진행됨에 따라 단지 보이는 것만이 아닌 실제 검침원이 사용하기 용이하게 구현 하여야 했으면 하는 바람이 있다.

향후 연구로는 실제 HDLC 프로토콜을 기반을 두어 운영되는 DLMS에 현 프로그램을 구동시켜 보는 것이며, 또한 TCP/IP, UDP환경에서 구동되는 DLMS/COSEM 시스템에도 현 프로그램을 적용시켜 얼마나 정확히 구현이 되었고 실제 구동되는 프로토콜들에 대해서 동기가 얼마나 되었는지도 알아야 할 것이라 생각된다. 또한 위에서 잠시 언급했듯이 검침원이 조작하기 쉬운 인터페이스의 개발이 필요하다고 여겨지는 바이다.

참고문헌

[1] 한국전기연구원 Blog, http://blog.naver.com/
[2] 산업자원기술표준원, 전기 계량-검침과 종별 계량,부하 제어를 위한 데이터 교환, KSC IEC 62056-53, 2005
[3] IEC, Data link layer using HDLC protocol, IEC 62056-46, 2007
[4] IEC, COSEM transport layers for IPv4 networks, IEC 62056-47, 2006
[5] IEC, COSEM Application layer, IEC 62056-53, 2006
[6] IEC, Object identification system (OBIS), IEC 62056-61. 2006
[7] IEC, Interface classes, IEC 62056-62. 2006
[8] Wireshark, http://www.wireshark.org/
[9] Eltima soft ware, http://www.eltima.com/