정 보 보 호 학 회 지
제25권 제5호, 2015. 10

효율적인 Modbus 보안 적용 방안

권 태 연*, 이 옥 연**

요 약

산업제어 시스템의 통신 프로토콜 중 하나인 MODBUS/TCP는 오늘날 산업체에서 높은 점유율로 사용되고 있다. 단순한 구조로 다양한 기능을 가지고 있어 “hard real system”에서 주로 사용되고 있는 MODBUS/TCP의 보안 적용 방안 또한 많은 연구로 진행되어 왔다. DNP3, SSL, SCTP와 같은 프로토콜을 연동하거나 공개키를 이용한 전자서명 적용 등의 방법이 대표적인 MODBUS/TCP의 보안 방법은 충분한 메모리 크기, 고속 공개키 연산을 위한 하드웨어 가속기 등을 요구로 한다. 하지만 I/O 디바이스 등 열악한 환경에 이는 부담스러운 요소로 작용될 수 있다. 따라서, 본 논문에서는 이러한 열악한 환경에서 안전한 MODBUS/TCP 통신과 기존의 MODBUS/TCP와의 호환성을 고려한 보안 적용 방안을 소개한다. 또한 측정한 결과를 통해 본 논문에서 제시하는 E-ModbusSec가 “hard real system”에서 충분히 제 역할을 수행할 수 있음을 보이고자 한다.

Ⅰ. 서 론

2014년 12월 18일 영화 다이하드 4에서 볼 법한 ‘파이어세일’이 발생하였다. 이 날 발생한 공격은 한수원을 대상으로 일어난 것으로 직원정보에 이어 기술 자료까지 유출되는 사태가 발생하였다. 이 사건으로 산업제어시스템 보안에 대한 관심이 급증하였으며 관련 기관들이 활발히 연구개발을 진행하고 있다.

산업제어시스템은 제조, 생산, 발전 등에 필요한 산업공정 뿐만 아니라 폐수 처리, 기름 및 가스 파이프라인, 송전 및 배전 등에 필요한 기반 시설 공정을 바탕으로 한 시스템을 말하며, 건축, 공항 등에 사용되는 설비 공정을 바탕으로 원격 명령을 통해 제어 명령을 설비에 전달하고 설비 정보 수집한다. 이러한 산업제어시스템은 SCADA(supervisory control and data acquisition) system, DCS(Distributed control system) 등 다양한 유형들로 구성되어 있으며, PLC(Programmable Logic Controllers), 센서 등 다양한 요소들로 구성되어 있다.

산업제어시스템은 국가의 주요한 기반시설을 제어하는 시스템으로 구성되어 있어 높은 보안성을 요구하고 있으며, 기존의 독립망 운영에서 범용 표준기술이 적용됨에 따라 개방화가 점차적으로 확산되고 있어 취약성 증가에 따른 보안강화가 중요한 문제가 되고 있다.

본 논문에서는 산업제어시스템에서 요구하는 보안요구사항에 대한 분석과 대표적 통신 프로토콜인 MODBUS/TCP에 대한 소개를 한다. 또한 보안 방안을 적용하여 안전한 MODBUS/TCP를 구상하는 것을 목표로 한다.

Ⅱ. 산업제어시스템 보안요구사항

산업제어시스템과 IT시스템은 운영하는 데에 다양한 차이점이 존재한다. 따라서 기존 IT시스템에서 사용되던 보안 방법이 산업제어시스템에 적용되기 위해서는 산업제어시스템에서 요구하는 보안요구사항에 맞게 변경되어야 한다.

산업제어시스템 보안표준 및 보안요구사항 등 보안 정책을 위해 미국, 유럽, 일본 및 국내에서는 체계적인 기반을 마련하고 있으며, 특히 미국에서는 운영환경 및 범위, 운영분야, 개발자 및 운영자 등에 따라 차이를 두어 적용할 수 있는 가이드 문서를 제공하고 있다.

산업제어시스템은 일반적인 IT시스템과 가용성, 네트워크 구성, 시스템 수명 등 다양한 면에서 차이점을 가진다.

* 국민대학교 금융정보보안학과 (kte4567@kookmin.ac.kr)
** 국민대학교 금융정보보안학과 (oyyi@kookmin.ac.kr)

〔표 1〕 산업제어시스템과 IT시스템의 차이

구분	산업제어시스템	IT시스템
가용성	• 재부팅 비 허용 • 계획적인 절전	• 재부팅 허용
지연	• 실시간 • 응답 시간	• 비 실시간 • 응답 신뢰성
CIA 중요도 순서	무결성 가용성 기밀성	기밀성 무결성 가용성
네트워크 구성	• 독점적으로 지정된 유·무선을 포함한 복합 네트워크	• 표준 통신 프로토콜 • 제한된 무선 능력을 가진 유선 네트워크
시스템 수명	• 15~30년	• 3~5년

[표 1]에서 산업제어시스템과 일반적인 IT시스템의 차이를 분석하였다.

산업제어시스템은 전기, 철도, 교통 등 실시간 제어가 중요한 시스템들이 포함되어 있다. 주로 시급하고 위험한 물질을 다루기 때문에 갑작스런 오동작과 재부팅은 허용되지 않으며, CIA (Confidentiality, Integrity, Availability)의 중요도 순서가 IT시스템과 다르다. 또한 TCP/IP외 PROFIBUS, CAN, MODBUS, DNP3와 같은 제어시스템 전용 프로토콜을 사용한다.

산업제어시스템은 시스템이 구축된 후 약 15년에서 30년까지 지속적인 서비스를 제공해야한다. 따라서 운영상 문제점이 발견되어도 즉시 수정하기 어렵고, 별도의 시스템에서 안전성을 충분히 검증한 후 계획적인 수정을 수행해야 한다. 또한 원격 유지보수 및 자동 업데이트가 불가하기 때문에, 유지보수가 필요한 경우 개발업체에 의한 실제 제어시스템 내 유지보수를 수행해야 한다.

마지막 가장 중요한 산업제어시스템의 특징으로는 다수의 원격 장치와 통신하는 데에 실시간성을 요구하므로, 보안 통신을 하는 경우 경량/저비용의 보안적용을 필요로 한다는 점이다.

Ⅲ. MODBUS 내 취약성과 보안 적용 사례

이 장에서는 MODBUS 통신 프로토콜의 소개와 이 프로토콜이 가지는 알려진 취약성에 대해 분석한다. 또한 MODBUS 취약성을 보완하기 위해 제안된 보안 적용 사례에 대해 소개한다.

3.1. 용어 및 약어 설명

- ADU : Application Data Unit
- PDU : Protocol Data Unit
- MBAP Header : Modbus Application Protocol Header
- TS : TimeStamp

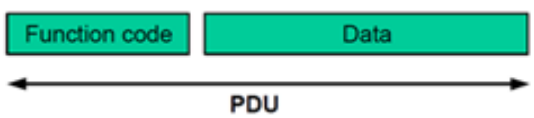
3.2. MODBUS

MODBUS는 1979년 Modicon에서 만든 시리얼 통신 프로토콜로서, 오늘날까지 전 세계의 산업체에서 가장 많이 사용되고 있는 대표적인 통신 프로토콜이다. 이 프로토콜은 본래 Modicon 자사의 controller 시리얼 통신을 위해 개발된 어플리케이션 계층 프로토콜이며, 2004년 이후부터 그 권리가 MODBUS organization으로 위임되었다.

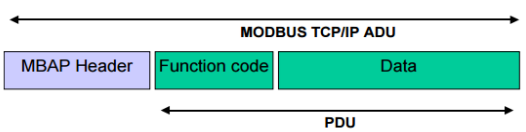
MODBUS 프로토콜은 단순하면서도 수행할 수 있는 기능과 활용될 수 있는 방식이 다양하다. 또한 통신 프로토콜 Specification이 모두 공개라는 장점을 가지고 있어 오랜 기간 동안 사용되어 왔으며, 오늘날 De Facto Standard로서 사용되고 있다.

MODBUS는 request-response방식의 프로토콜로 통신과정에서 사용되는 데이터는 MODBUS PDU의 Function code를 통해 구분한다.

MODBUS는 본래 시리얼 통신 프로토콜로서 개발되었지만, 통신 기술이 발달하여 산업용 이더넷이 선호됨에 따라 이더넷 기반의 MODBUS/TCP가 개발되었다. MODBUS/TCP는 기존의 MODBUS와 산업용 이더넷의 결합으로 산업제어시스템에서 현재까지 높은 점유율을 차지하고 있다.



〔그림 1〕 MODBUS의 PDU



〔그림 2〕 MODBUS/TCP의 ADU

3.3. MODBUS의 알려진 취약성

일반적으로 MODBUS 프로토콜에 관한 공격은 크게 Command를 이용한 Denial of Service 공격, 공격자에 의해 해킹된 HMI로부터 보내진 Illegal command 공격, Man-in-the-middle 공격으로 볼 수 있다. 세 가지 공격은 다양한 유형으로 수행되며, 공격 목적에 따라 [표 2]와 같이 정리된다.

또한, 통신에 관련된 공격에 대해 Only Serial 기반, Only TCP 기반, Serial&TCP 기반에 따라 공격 유형을 상세화 할 수 있다.

우선, Only Serial 기반으로 하는 공격은 주로 Function code가 조작됨으로 인해 발생한다. 공격자가 Function code를 조작하여 메시지를 전송하여 발생하는 공격 중에 Remote restart 공격이 대표적이다. Remote restart 공격은 field device의 serial line 포트를 초기화시키고, 강제적으로 device가 재부팅되도록 하는 공격이다. 이 공격으로 인해 device는 'Listen Only Mode'상태가 되어 request를 수신하더라도 response를 작성하지 않아 정상적인 통신이 불가하게 된다.

Only TCP 기반으로 하는 공격은 정당한 메시지에 잘못된 데이터를 삽입하는 형태로 많이 발생한다. 대표적인 공격으로는 Irregular TCP Framing 공격이 있다. Irregular TCP Framing 공격은 여러 개의 MODBUS 메시지를 하나의 TCP Frame에 담을 수 없는 것을 이용한 공격으로서 master unit이나 field device가 연결을 종료하도록 하는 부적절한 메시지를 만들기 위해, 메시지를 추가적으로 삽입하거나 수정하는 공격이다.

Serial&TCP 기반으로 하는 공격은 DoS를 유발하는 메시지를 작성하거나 탈취한 메시지를 이용한 재전송 공격의 형태로 존재한다. 대표적인 공격으로는 Broadcast Message Spoofing 공격, Baseline Response Replay 공격, Direct Slave Control 공격, Passive Reconnaissance 공격이 있다. Broadcast Message Spoofing 공격은 잘못된 broadcast 메시지를 slave들에게 보내는 공격이다. broadcast 메시지는 master의 전원이 인가될 때 slave들이 등록하게 하는 경우에 유일하게 사용되는데, 이러한 broadcast 메시지가 반복해서 사용될 경우 slave는 계속해서 등록하려 하므로 정상적으로 동작하지 못하게 된다. Baseline Response Replay 공격은 master와 field device 사이의 메시지를 탈취하여 이후 master에게 메시지를 재전송하는 공격이다. Direct Slave Control 공격은 master를 수행 불능 상태로 만들고 여러 field device들을 직접 제어하는 공격이다. 이는 master가 제어하는 field device의 ID가 노출되어 있기 때문에 발생하게 된다. 마지막으로 Passive Reconnaissance 공격은 공격자가 MODBUS 메시지를 수동적으로 읽는 공격이다. 중간에 공격자에 의해 메시지가 탈취되는 것을 막을 수는 없으나, 메시지의 내용이 그대로 노출되는 것은 문제이므로 메시지가 탈취되어도 바로 노출되지 않을 수 있는 방안이 필요하다.

〔표 2〕 공격 목적에 따른 MODBUS 취약점

공격 목적	공격 유형
기밀성 공격	MODBUS 메시지 내용을 읽는 것
무결성 공격	잘못된 데이터를 삽입하는 것
가용성 공격	Slave device들이 중요 기능을 상실하도록 하거나 재부팅 또는 갑작스러운 오작동을 발생시킴으로써 정상적인 동작이 어렵도록 하는 것

3.4. MODBUS 보안 적용 사례

MODBUS가 오랜 시간 사용되어 온 만큼 보안 적용 사례도 많이 연구되어 왔다. 주된 방법은 SSL을 연동하거나 DNP3와 같은 프로토콜을 연동하는 것이다. 하지만 MODBUS는 열악한 환경의 기기들이 통신하도록 개발된 프로토콜이기에 위와 같은 방법들은 자칫 부하가 클 수 있다. 따라서 본 논문에서는 다른 라이브러리와 연동하지 않으며 MODBUS 프레임을 크게 수정하지 않는 방법을 선택하였다. MODBUS 프로토콜 자체에 보안을 적용하는 대표적인 사례로 Igor Nai Fovino 외 3명이 저술한 "DESIGN AND IMPLEMENTATION OF A SECURE MODBUS PROTOCOL"가 있다.

3.4.1. Design and implementation of a secure modbus protocol

이 문서에서 말하는 산업 제어 시스템 내 발생한 핵심 공격은 인가되지 않은 명령 실행, MODBUS DoS

〔표 3〕 안전한 MODBUS의 조건

1. 인가되지 않은 개체는 메시지에 접근하지 못하도록 한다.
2. 인가되지 않은 개체는 메시지를 수정하지 못 하도록 한다.
3. 다른 개체를 모방하는 것을 못하도록 한다.
4. 어떠한 개체가 수행되고 있는 동작을 멈추는 것을 못 하도록 한다.
5. 어떠한 개체가 인가되지 않은 동작을 수행하기 위해 획득한 메시지를 재사용하는 것을 못하도록 한다.

공격, Man-in-the-middle 공격, 재전송 공격이 있다. 이에 따라 이 문서가 주장하는 안전한 MODBUS의 조건은 다음 [표 3]과 같다.

[표 3]를 위해 적용한 보안 종류는 무결성, 인증, 부인방지, 재전송 방지이며, 기밀성은 제외하였다. 무리한 기밀성 적용은 앞선 공격을 완화시킬 방법이 아니며, 일반적으로 암호알고리즘을 이용한 기밀성은 real-time 퍼포먼스에 영향을 줄 수 있는 오버헤드를 발생시킨다는 것이 그 이유이다.

본래의 MODBUS/TCP 프레임은 [그림 2]와 같다. 보안을 적용한 MODBUS/TCP 프레임은 [그림 2]와 같이 헤더 부분에 타임스탬프를 추가하고, 테일 부분에 해시 값에 대한 서명을 추가하여 무결성과 부인 방지를 제공함을 확인할 수 있다.

Ⅳ. E-ModbusSec

[3]에서 고려한 주요 공격은 메시지와 사용자의 정당성 위주이다. 이를 보완하기 위해 여러 보안 적용 방법이 사용되었으나, 이에는 몇 가지 문제가 되는 부분이 있다.

우선, [3]에서 제안한 내용에는 기밀성을 제공하는 부분이 없다. [3]에서는 앞선 공격에 대해서 기밀성이 크게 작용하지 않고, 적용한 보안 방법 중 인증을 위한 공개키 스킴을 사용하였기 때문에 암호알고리즘을 이용한 기밀성을 추가할 경우 오버헤드가 크게 발생된다고 설명한다. 그러나 알려진 MODBUS의 취약점 중에는 기밀성과 관련하여 발생하는 공격이 존재한다. 따라서 데이터 기밀성은 간과할 수 없는 문제이다.

이 장에서는 3.3.1절에서 소개한 논문을 참조하여 좀 더 효율적인 방안을 구상하였다. E-ModbusSec은 앞서 소개한 논문에서 사용한 E-Modbus 단어를 활용한 단어로, “Ethernet기반의 Secure Modbus”를 의미한다.

4.1. 용어 및 약어 설명

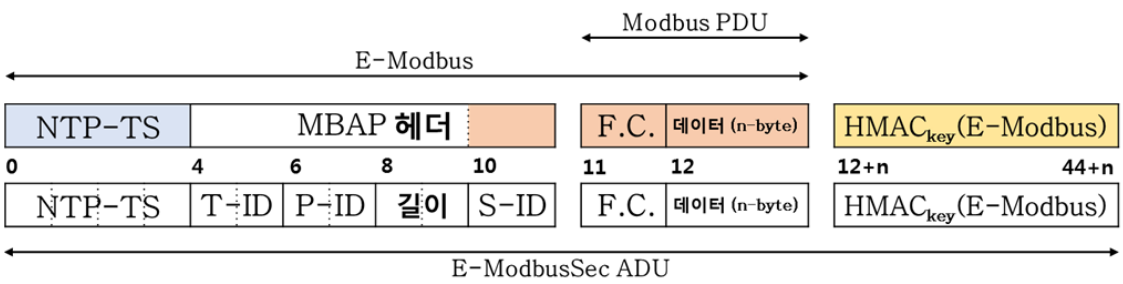
- NTP-TS : Network Time Protocol TS
- T-ID : Transaction Identification
- P-ID : Protocol Identification
- S-ID : Slave Identification
- F.C. : Function Code

4.2. E-ModbusSec 설계

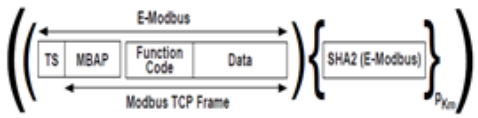
이 절에서는 기존의 MODBUS 프로토콜 방식을 따르면서 통신에 이용되는 데이터에 보안을 적용한 안전한 MODBUS 프로토콜을 설계한다. 또한 설계에 대한 근거를 앞서 언급한 취약점들에서 찾고자 한다.

E-ModbusSec 데이터 포맷은 [그림 3]과 같다. NTP-TS는 4바이트로 구성되며, 정해진 시간 내에서 유효한 값을 가지는 타임스탬프의 특징에 따라 앞서 소개한 Baseline Response를 포함한 재전송 공격을 방지할 수 있다.

HMAC은 해시 함수 기반의 메시지 인증 코드이다.



〔그림 3〕 E-ModbusSec 데이터 포맷



〔그림 4〕 보안을 적용한 MODBUS/TCP 프레임

〔표 4〕 적용한 보안 종류에 따른 보안 방법

보안 종류	보안 방법
무결성	SHA2 해시 함수
인증	RSA 기반 서명 스킴
부인 방지	RSA 기반 서명 스킴
재전송 방지	NTP 타임 스탬프

HMAC에 사용되는 해시 함수로는 최근 국내에서 경량 고속 구현이 가능하도록 개발된 LSH-256을 사용하였다. 해시 함수는 키가 사용되지 않으므로 그대로 적용하는 것은 악의적인 데이터 위/변조에 노출될 수 있다. 따라서 키를 이용하는 HMAC을 사용하였다. [3]에서는 이를 RSA 서명 스킴을 통해 무결성과 인증 및 부인방지 기능을 제공하고자 하였다. 공개키의 사용은 타원곡선 적용, 수학적 최적화, 하드웨어 가속기 등으로 길이나 속도 면에서 많은 발전이 이루어졌다. 하지만 하드웨어 가속기나, 난수 생성과 같은 방법의 공개키 사용은 I/O 디바이스 등의 열악한 Field device에 있어서 부담스러운 방안일 수 있다. 또한 MODBUS는 반드시 정해진 timeout시간(0.5초) 내에 response가 도달해야 하는 프로토콜이기에 향상되지 않은 공개키 적용 또한 부담이 될 수 있다.

따라서 본 논문에서는 “hard real-time system” 내 전자서명을 쓰지 못하는 환경에서의 실질적인 대안 방법으로 HMAC을 선정하였다. HMAC의 내부 해쉬 함수로 LSH를 사용하였으며, 이는 국내에서 개발한 경량 해쉬 함수이다.

마지막으로, 암호 알고리즘 적용은 앞서 언급한 데이터 기밀성 공격을 방지할 수 있다. 암호 알고리즘 적용에 있어 스트림 암호와 블록 암호를 비교하였을 때, 전송되는 실제 데이터 크기를 고려한 스트림 암호가 적합하다 판단될 수 있으나 일정 크기의 키 공간 내에서 키 갱신을 하고 간단한 XOR 연산으로 암호문을 생성하는 스트림 암호는 안전성이 높지 않다고 판단되어 블록암호가 적합하다고 판단하였다. 따라서 암호 알고리즘으로 블록암호를 적용하며, 보안 강도를 고려하여 128 이상의 강도를 가지는 암호 알고리즘 중 최근 국내에서 개발한 경량암호 LEA를 적용하였다.

암호화 데이터의 범위는 MBAP 헤더 중 마지막 구성요소인 S-ID, 그리고 Function code와 Data이다. 이는 S-ID가 노출됨으로써 발생하는 Broadcast Message Spoofing 및 Direct Slave Control 공격, Function code를 조작함으로 인해 발생하는 Remote restart 공격, 데이터가 노출됨으로 인해 발생하는 Passive Reconnaissance 공격을 방지하기 위함이다.

4.3. E-ModbusSec 구현

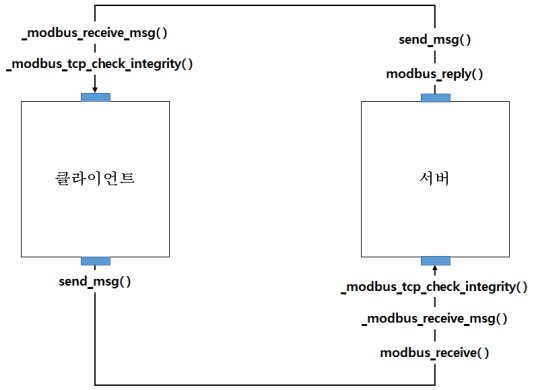
이 절에서는 Libmodbus의 구조에 대한 설명과 보안을 적용한 E-ModbusSec 성능에 관한 실험 및 평가를 한다. 본 논문에서는 MODBUS 프로토콜을 구현한 여러 오픈소스 중 C로 구현된 Libmodbus를 사용하였다.

4.3.1. LIBMODBUS

Libmodbus는 MODBUS를 사용하는 기기들이 송/수신하기 위한 소프트웨어 라이브러리이며 시리얼 기반의 RTU 또는 이더넷 기반의 TCP 통신을 제공한다.

〔표 5〕 Libmodbus v3.1.1에서 제공하는 함수 목록

Libmodbus v3.1.1			함수 코드
1-bit 단위 접근	Discrete Input	Read Discrete Inputs	02
	Coil	Read Coils	01
		Write Single Coil	05
		Write Multiple Coils	15
16-bit 단위 접근	Input Register	Read Input Register	04
	Holding Register	Read Holding Register	03
		Write Single Register	06
		Write Multiple Registers	16
		Read/Write Multiple Registers	23
		Mask Write Register	22

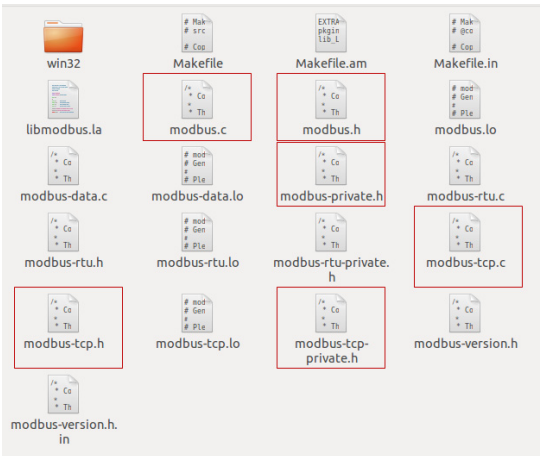


〔그림 5〕 libmodbus의 논리적 통신 구조 예

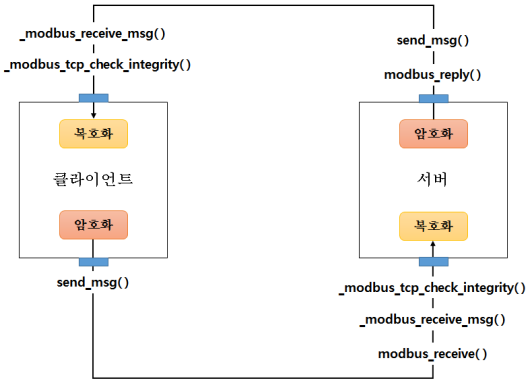
Libmodbus는 Linux, Mac OS X, FreeBSD, QNX, Windows 운영체제를 지원하며 현재 최신버전은 Libmodbus v3.1.2이다.

본 논문에서는 실험할 당시 최신 버전이었던 Libmodbus v3.1.1을 사용하였으며 Libmodbus v3.1.1은 데이터 접근에 따라 [표 5]와 같이 10가지 함수를 제공한다. 본래 MODBUS 프로토콜의 데이터는 [그림 2]와 같이 구성되지만 Libmodbus에서는 구현상의 편의를 위해 헤더영역, 메타데이터 영역, 데이터 영역으로 나뉜다. 또한 클라이언트와 서버의 사용 편의성을 위하여 API를 분리하였다. Libmodbus를 이용한 MODBUS/TCP 통신은 [그림 5]와 같이 논리적으로 표현할 수 있다.

[그림 5]는 하나의 함수를 예를 든 것으로, write_multiple_register 함수이다.



〔그림 6〕 수정이 필요한 libmodbus/src 하위 파일



〔그림 7〕 통신 흐름에 따른 보안 구성

4.3.2. 구현 방법

Libmodbus에서 구분된 데이터 영역에 따라 헤더영역에는 NTP 타임스탬프를 추가하고, 메타데이터와 데이터영역에는 LEA-CTR 적용, HMAC-LSH를 MAC으로 추가하였다. 위의 보안 스킴을 적용하기 위해서는 [그림 6]에 해당하는 파일에 대한 수정이 필요하다.

write_multiple-registers 함수를 예로 보안을 적용하여 데이터를 송신하고, 수신하는 과정은 [그림 7]과 같다.

4.3.3. 실험 환경

〔표 6〕 실험 환경

Desktop	
운영체제	Ubuntu 12.04 LTS 32-bit
CPU	Intel Core i5-3570 3.40GHz
컴파일러	i686-linux-gnu-gcc 4.6.3
Libmodbus	
버전	libmodbus v3.1.1

4.3.4. 실험 결과

〔표 7〕 Desktop에서 측정한 Libmodbus v3.1.1, E-ModbusSec 통신 속도 (단위 : ms)

함수	바이트 길이 (req / rsp)	Libmodbus v3.1.1
		E-Modbus Sec
		3.4.1절 방법
Read Discrete Inputs	12 / 12	0.89×10^{-4}
		0.68×10^{-4}
		20.1406×10^{-2}
Read Coils	12 / 10	1.21×10^{-4}
		0.78×10^{-4}
		20.4588×10^{-2}
Write Single Coil	12 / 12	1.79×10^{-4}
		1.96×10^{-4}
		19.5135×10^{-2}
Write Multiple Coils	18 / 12	1.02×10^{-4}
		0.74×10^{-4}
		14.4459×10^{-2}
Read Input Register	12 / 11	0.91×10^{-4}
		0.67×10^{-4}
		12.9467×10^{-2}
Read Holding Register	12 / 11	0.89×10^{-4}
		0.68×10^{-4}
		12.7265×10^{-2}
Write Single Register	12 / 12	0.87×10^{-4}
		0.68×10^{-4}
		9.1223×10^{-2}
Write Multiple Registers	19 / 12	0.93×10^{-4}
		0.69×10^{-4}
		17.0405×10^{-2}
Read/Write Multiple Registers	21 / 15	1.26×10^{-4}
		0.68×10^{-4}
		9.9960×10^{-2}

Libmodbus v3.1.1과 E-ModbusSec를 적용한 Libmodbus v3.1.1, 그리고 3.4.1절에서 소개한 Modbus 보안 방법을 적용한 Libmodbus v3.1.1의 비교 결과이다. 3.4.1절 방법을 구현하기 위해 openssl-1.0.2c의 RSA PSS를 사용하였다. 측정 결과, E-ModbusSec를 적용하였을 때, Libmodbus v3.1.1과 비교하여 0.44~4.79배 속도가 향상하였으며 3.4.1절에서 소개한 보안 방법의 경우 비교적 좋지 않은 성능결과가 도출되었다. 이는 보안 요소가 적용되지 않은 것과 적용한 것에 대한 비교 결과로 의미가 있는 결과이다.

〔표 8〕 메모리 요구량 비교

라이브러리	메모리 크기
libmodbus v3.1.1	111.4kB
E-ModbusSec	131.6kB
3.4.1절 방법	modbus : 122.9kB
	openssl : 2.7MB

〔표 9〕 Desktop에서 측정한 Libmodbus v3.1.1, E-ModbusSec 처리 속도 (단위 : cpb)

함수	바이트 길이 (req / rsp)	Libmodbus v3.1.1
		E-Modbus Sec
		3.4.1절 방법
Read Discrete Inputs	12 / 12	4265.88
		16491.17
		10411676.00
Read Coils	12 / 10	4561.23
		16600.33
		13211028.00
Write Single Coil	12 / 12	4130.67
		16253.25
		9728853.00
Write Multiple Coils	18 / 12	3621.63
		10920.22
		8286157.50
Read Input Register	12 / 11	4496.70
		16424.42
		11405225.00
Read Holding Register	12 / 11	4446.35
		16540.00
		10952972.00
Write Single Register	12 / 12	4113.67
		16393.42
		10626970.00
Write Multiple Registers	19 / 12	3504.52
		10508.89
		9039664.00
Read/Write Multiple Registers	21 / 15	3653.19
		21624.43
		17715498.00

V. 결 론

본 논문에서 소개한 E-ModbusSec은 HMAC, 암호 알고리즘 등을 적용하여 무결성과 기밀성을 제공할 뿐 아니라 키를 사용하였기에 키가 가지는 안전성 또한 함께 제공한다.

공개키를 적용하여 제공 가능한 출처인증은 산업제어시스템에서 중요한 보안 기능 요소이지만, hard real system에서 Field device를 구성하는 I/O 디바이스 등 열악한 환경에서는 공개키의 적용이 부담될 수 있거나 심하게는 보안 기능을 탑재하지 못할 수도 있다. 따라서 본 논문에서는 이러한 열악한 환경에서 효율적으로 제공 가능한 MODBUS 보안 방안을 소개하였으며, 이는 기존의 MODBUS 프레임과 동일한 구조로 구성되었기 때문에 호환성이 높다는 장점을 가진다. 또한 기존의 MODBUS 라이브러리의 메모리 요구량과 차이가 크지 않기 때문에 향후 더 많은 기능을 탑재할 수 있는 공간을 확보할 수 있다.

참 고 문 헌

[1] 128 비트 블록 암호 LEA, 한국정보통신기술협회, 2013.

[2] 고속 해시함수 LSH 규격서, 국가보안기술연구소, 2014.

[3] Design and implementation of a secure modbus protocol, Igor Nai Fovino, Andrea Carcano, Marcelo Masera and Alberto Trombetta, IFIP AICT 311, pp. 83-96, 2009.

[4] Modbus application protocol specificati on v1.1b3, Modbus organization, April 26, 2012.

[5] 산업용 네트워크 기술 동향과 적용사례, 김기준, 알에스오토메이션(주), 2010.

[6] Attack taxonomies for the Modbus protocol, Peter Huitsing, Rodrigo Chandia, Mauricio Papa, Sujeet Shenoi, IJCIP, 2008.

[7] Open modbus/tcp specification, Schneider Electic, March 29, 1999.

[8] Modbus messagin on tcp/ip implementation guide v1.0b, October 24, 2006.

[9] Modbus over serial line specification and implementation guide v1.02, Dec 20, 2006.

〈저자소개〉

권 태 연 (Taeyean Kwon)
학생회원

2014년 2월 : 국민대학교 수학과 졸업
2014년 3월~현재 : 국민대학교 금융정보보안학과 석사과정
관심분야 : 통신보안, 정보보안

이 옥 연 (Okyeon Yi)
정회원

1990년 2월 : 고려대학교 대수학 석사 졸업
1996년 8월 : University of Kentucky 대수학 박사 졸업
2001년 9월~현재 : 국민대학교 자연과학대학 금융정보보안학과 교수
관심분야 : 네트워크 보안, 정보보호, 산업제어시스템