SCADA 시스템에서 무리수 기반의 스트림 암호화 통신 프로토콜 연구

김성종*, 김형주*, 김형민**, 전문석*
*숭실대학교 일반대학원 컴퓨터공학과
**숭실대학교 정보과학대학원 정보보안학과
e-mail:yesksj724@ssu.ac.kr; hyungjoo.kim@ssu.ac.kr; ovsm@naver.com;
mjun@comp.ssu.ac.kr

Stream cipher communication protocol based on Irrational numbers in SCADA System

Seong-Jong Kim*, Hyung-Joo Kim*, Hyung-Min Kim**, Moon-Seog Jun*
*Dept of Computer Engineering, Soong-sil University
*Dept of Information Security, Soong-sil University

요 약

SCADA(Supervisory Control and Data Acquisition) 시스템은 국가중요기반망인 전력, 가스 등을 제어하기 위해 사용되어 왔다. 최근에는 기존의 폐쇄적인 통신 환경과 달리 외부망과 연결된 개방 통신 환경을 사용함에 따라 안전한 통신을 위한 암호화 기술이 연구되고 있다. 기존 시스템에 적용되고 있는 프로토콜들은 공유키를 갱신하기 위해 수동적으로 설정해야했다. 본 논문에서는 SCADA 시스템을 구성하는 Server와 RTU가 상호 인증하여 무리수 기반의 스트림 암호화 통신과 타임스탬프를 이용한 키 갱신 프로토콜을 제시하여 재사용공격과 가장공격이 불가능한 실시간 암호화통신 프로토콜을 연구하였다.

I. 서론

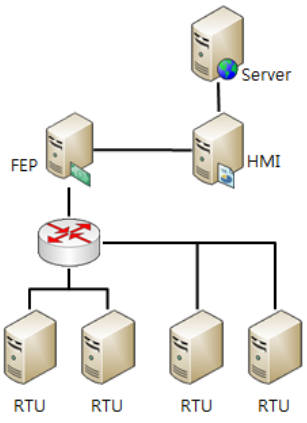
SCADA 시스템은 전기, 가스, 수도, 교통 등 주요기반시설을 감시하는 시스템으로 특정 장소의 상태정보 데이터를 원격수집장치(RTU:Remote Terminal Unit)로부터 실시간으로 수집, 기록, 표시하며 중앙 제어 시스템이 RTU를 감시하고 제어하는 시스템이다. 과거의 SCADA 시스템은 외부망과 분리되어 독립적으로 운용되었고 제어 시스템에 특화된 프로토콜을 사용하여 사이버 공격으로부터 안전하다고 판단했다. 하지만 최근에는 자동화 효율을 높이고 상호운영성을 높이기 위해 외부망과 인터넷 연결로 악의적인 사용자가 올바른 RTU의 ID를 가장하여 서버 관리자에게 조작된 정보를 보내서 불법적인 변경을 유도하거나 서버의 메세지가 악의적으로 조작될 경우 실시간으로 정보를 처리해야하는 SCADA 시스템에 혼란을 초래할 우려가 있다[1].

따라서, 본 논문에서는 SCADA 시스템의 서버와 RTU 간의 상호인증을 통하여 올바른 RTU와의 통신을 하고 서로간의 실시간 통신을 위해 무리수 기반의 스트림 암호화 일대일키(Pair-wise key) 공유 프로토콜을 제안한다.

II. 관련 연구

2.1 SCADA 시스템

SCADA 시스템은 전력·가스·항공 등 활용되는 대상과 목적에 다라 네트워크 구성환경과 구조가 달라질 수 있지만 일반적인 구조는 다음 (그림 1)과 같다. SCADA 시스템은 기본적으로 서버, HMI (Human machine interface), 전단처리기(front-end processor, FEP), RTU 등으로 구성된다.



(그림 1) 일반적인 SCADA 시스템

SCADA 서버는 RTU에서 전송되는 계측정보를 수집·분석하고, HMI를 통해 내려진 제어 명령을 RTU로 전달하는 장치이다. HMI는 다양한 수집정보를 SCADA 서버관리자에게 제공하며, 관리자가 내린 제어명령을 RTU로 전달한다. FEP는 SCADA 서버와 다른 프로토콜을 사용하는 RTU도 SCADA 서버와 통신할 수 있도록 프로토

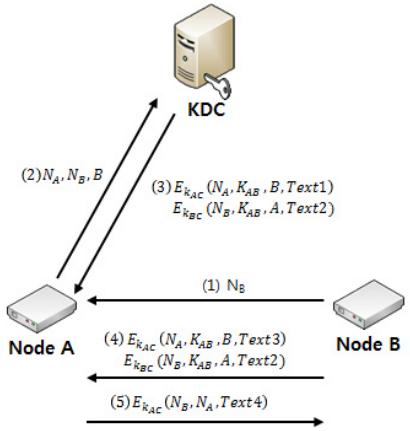
콜을 변환해주는 역할을 수행한다[2].

2.2 SKE(Sandia Key Management)

SKE는 Beaver 등이 제안한 SCADA 시스템에서의 키 관리 프로토콜로서 이 방식에서는 세션 키를 설정할 때 5가지 요소들을(LTK, FLAG, ID, TVP, LEN)미리 공유하고 있어야 한다. 서버와 RTU는 LTK를 수동으로 공유하고 있으며 Seed값을 랜덤비트열과 해쉬(Hash)하여 GK(General Key)를 생성하여 LTK로 암호화한 다음 RTU로 전송된다. 이 방식을 사용할 경우 특별한 상황이 일어나야 수동적으로 LTK를 갱신해야하고, 특정 RTU의 ID를 복제하여 사용할 경우 악의적인 사용자가 통신을 조작할 수 있는 취약점이 있다[3].

2.3 SKMA(Key Mangement Architecture for SCADA)

SKMA는 2006년에 R. Dawson 등에 의해 제안되었다. 시스템에 새로운 RTU가 등록되어있을 경우, RTU간의 세션 키 설립을 통해 안전한 통신을 제공하고, 대칭키 알고리듬으로만 이루어져 있으며, 노드는 SCADA 시스템을 구성하는 모든 장치가 될 수 있다. 사용되는 키는 3가지로 첫 번째로 노드 A와 KDC의 사전공유 키, 두 번째로 노드 A와 B의 사전공유 키로써 노드 추가시 생성한다. 마지막으로 세션키가 있다.



(그림 2) Server와 RTU 사이의 상호 인증과정

(그림 2) SKMA는 노드와 노드 간에 키를 설립할 때 두 노드간의 공유키를 생성하기 위해 KDC와 통신을 해야하고 항상 KDC와 통신해야하기 때문에 고속의 대칭키 암호 알고리듬을 사용한다. 하지만 중간 노드의 통신량이 많고 KDC의 부하가 많이 걸린다는 단점을 가지고 있다[4].

III. 제안 프로토콜

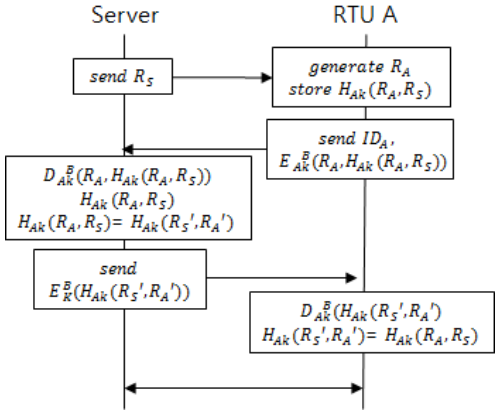
본 논문에서는 서버와 특정 원격지 RTU간의 상호 인증을 하고 무리수 기반의 스트림암호화 통신 프로토콜을 제안한다. 제안 프로토콜에서 쓰이는 용어는 아래 <표 1>과 같다.

<표 1> 제안 프로토콜 용어 정의

Notation	Description
R_S	서버의 의사난수
R_A	노드의 의사난수
N^1	첫번째 R_S, R_A 논리합 결과
ID_A	노드의 아이디
ID_S	서버의 아이디
A_K	서버와 RTU의 사전 공유 키
$H_{Ak}()$	키 A_K를 이용하는 해쉬함수
$E^B_{Ak}()$	키 A_K를 이용한 블록암호화
$D^B_{Ak}()$	키 A_K를 이용한 블록복호화
$E^S_{Ak}()$	키수열 A_K를 이용한 스트림암호화
$D^S_{Ak}()$	키수열 A_K를 이용한 스트림복호화
Ak^1_{stream}	무리수를 이용해 나온 첫번째 키
T	타임스탬프
REQ_{KCHNG}	키수열 변경 요청 메세지
REQ_{ACK}	키수열 변경 응답 메세지
n	1부터 시작하고 1라운드당 1상승

3.1 상호 인증과정

상호 인증과정은 다수의 RTU가 서버에 접속하여 데이터의 기밀성을 필요한 RTU에 대해서 서버이외에는 어떠한 정보도 제공하지 않도록 Server와 RTU 사이에 상호 인증하는 과정이고 다음 (그림 3)와 같다.



(그림 3) Server와 RTU 사이의 상호 인증과정

- 단계 1 : 서버 S는 ID_S를 난수생성기에 적용하여 랜덤수열 R_S를 RTU A에게 전달한다. A는 ID_A를 난수생성기에 적용하여 랜덤수열 R_A값을 생성하여 상호인증을 위한 $H_{Ak}(R_A, R_S)$을 생성하여 저장한다.

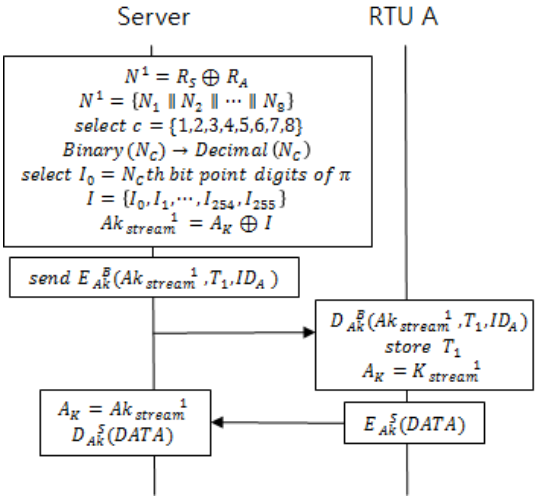
- 단계 2 : A는 ID_A와 $E^B_{Ak}(R_A, H_{Ak}(R_A, R_S))$를 S로 전송하고 S는 복호화 후 A에게 받은 R_A와 자신의

R_S를 가지고 $H_{Ak}(R_S', R_A')$를 계산하여 A에게서 받은 $H_{Ak}(R_A, R_S)$를 비교하여 일치하면 상호 인증이 성립된다.

- 단계 3 : 서버 S에서 $E_{Ak}^{B}(H_{Ak}(R_S', R_A'))$하여 A에게 전달하고 A가 복호화하여 S에게서 받은 해쉬값을 비교하여 일치하면 상호 인증이 성립됨을 확인하고 상호 인증과정을 완료한다.

3.2 Pair-wise키 생성과정

RTU들은 관리자가 직접 관리할 수 없는 환경에서 위치해있기 때문에 악의적인 물리적 조작으로 공유키가 유출될 수 있다. 이런 이유로 사전 공유키는 첫 번째 Pair-wise키 생성과정까지만 쓰이고, SCADA 시스템의 특징인 실시간 전송의 기밀성을 위하여 무리수의 무한소수를 사용하여 스트림 암호화에 사용되는 Pair-wise키를 생성한다. 본 논문에서는 대표적인 무리수인 π를 사용하였다[5][6]. 서버는 π의 소수점 2^{32}번째 자리수열을 안전한 곳에 저장한다. 세부동작 과정은 다음 (그림 4)과 같다.



(그림 4) Pair-wise키 설립과정

- 단계 1 : S는 $R_S \oplus R_A$하여 256bit길이의 결과 N^1을 추출한다. N^1을 8개의 블록으로 나눠 32bit길이의 블럭 하나를 선택하고 나머지 블럭들은 저장한다. 선택된 2진 수열N_C는 10진수로 변환한다.

$$N^1 = \{N_1 \| N_2 \| N_3 \| N_4 \| N_5 \| N_6 \| N_7 \| N_8\}$$

- 단계 2 : 선택된 N_C에 해당하는 π의 소수점자리 N_C번째 bit를 I_0로 선택하여 256bit 길이의 키를 갖는 이진수열 $I^1 = \{I_0, I_1, I_2, \ldots, I_{253}, I_{254}, I_{255}\}$를 추출한다.

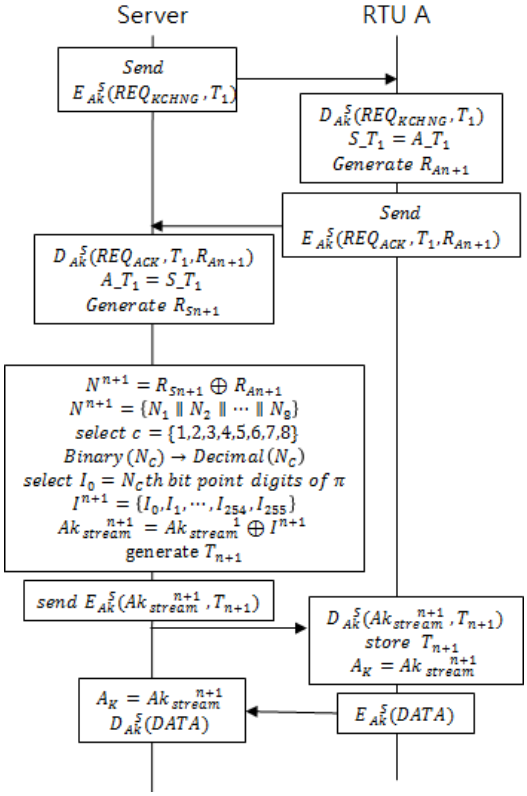
- 단계 3 : 공유키 A_K와 이진수열 I를 XOR연산을 하여 256bit의 Pair-wise키 Ak_{stream}^{1}을 추출하여 데이터의 안전한 전송을 위한 $E_{Ak}^{S}()$의키로 사용한다.

$$Ak_{stream}^{1} = A_k \oplus I$$

- 단계 4 : S는 $E_K^B(K_{stream}^{1}, T_1)$를 A로 전달하고 A는 복호화하여 T_1을 저장하고, Ak_{stream}^{1}을 이용해 S와 실시간 스트림암호화를 사용해 전달한 다음 사전 공유키 A_K를 삭제한다.

3.3 Pair-wise키 갱신 과정

Pair-wise키는 스트림 암호화 키로 쓰이기 때문에 아무리 매우 긴 주기를 가진 키 수열이라 해도 오랜 시간이 지나게 되면 반복된 주기를 가지게 되는 취약점이 드러날 수 있다. 이러한 점을 보완하기 위해 일정 시간이 지난 만큼 Pair-wise키의 갱신과정이 필요하다. pair-wise키 갱신에 대한 세부적인 동작 과정은 다음 (그림 5)와 같다.



(그림 5) Pair-wise키 갱신과정

- 단계 1 : S는 A에게 $E_{Ak}^{S}(REQ_{KCHNG}, T_1)$을 전송한다. A는 $D_{Ak}^{S}(REQ_{KCHNG}, T_1)$하여 A가 가지고 있는 T_1과 비교하여 일치하면 256bit 길이의 R_{An+1}을 생성한다. $E_{Ak}^{S}(REQ_{ACK}, T_1, R_{An+1})$ 후 S에게 전송하여 S가 복호화하면 상호 타임스탬프가 일치하는 것을 확인하고 R_{Sn+1}를 생성한다.

<표 2> 기존 키관리 프로토콜과 제안 프로토콜의 비교표

구 분	SKE	SKMA	제안 방식
RTU간 통신	불허	허용	불허
공유 키 갱신	수동 갱신	KDC를 이용하여 갱신	타임스탬프
세션 키 설정	LTK 이용	LTK 이용	랜덤수열 이용
장단점	통신방식의 이원화	LTK 갱신시 KDC통신 과부하	무리수의 무한소수점 계산이 어려움 및 대용량 저장공간 필요

- 단계 2 : S는 $R_{Sn+1} \oplus R_{An+1}$ 계산한 N^{n+1} 을 32bit의 8개 블록으로 나눈 다음 특정 하나의 블럭을 선택한 후에 서버에 저장된 π의 소숫점 자리에서 부터 256개의 이진수열 I^{N+1} 와 Ak^{1}_{stream}을 XOR 연산하여 새로운 키수열 Ak^{n+1}_{stream}을 생성하여 A에게 $E^{S}_{Ak}(Ak^{n+1}_{stream}, T_{n+1})$를 전송하여 pair-wise키를 갱신힌다.

IV. 제안 프로토콜의 성능분석

본 논문에서는 기존의 키 관리 프로토콜과 비교하여 제안 프로토콜의 차이는 아래의 <표 2>에 작성되어 있다. 본 논문의 제안 프로토콜의 단점으로 무리수의 2^{32}번째 소수점 자리수까지 기억을 하기 위해서는 최소 512MB이상의 큰 용량을 필요로 한다. 이러한 문제로 서버와 RTU의 1:1통신 또는 서버와 서버간의 통신에서 적합하며, RTU간 통신은 저장 공간이 부족하기 때문에 불가하다.

4.1 재사용 공격

제안 프로토콜에서 사용되는 공유키 값 A_K는 사전에 공유가 되어있다. 공격자가 초기 A_K값을 나중에 알게 되더라도, 서버와 RTU 간의 새로운 Pair-wise키 생성 후 제거하기 때문에 재사용 공격에 안전하다. 또한 이 프로토콜에서는 대표적인 무리수 값인 π를 사용하고 있지만, 관리자에 의해서 다양한 무리수의 값을 사용할 수 있다. 어떠한 무리수의 무한소수를 사용하는지 알고 있어도 해당 무한소수의 소수점자리 최대 2^{32}번째 비트에 위치한 256개의 비트수열을 찾아야 한다. 만일 찾더라도 현재 공유하고 있는 키를 알 수 없으면 복호화를 할 수 없으므로 재사용 공격이 불가능 하다.

4.2 가장 공격

제안된 프로토콜에서는 다수의 RTU들이 서버에 접근할 때마다 서버와 각각의 RTU가 랜덤수열(R_s, R_A)을 사전 공유키를 이용한 해쉬 값을 생성하여 비밀 정보의 노출을 예방하고 있다. 따라서 제 3자가 복제된 RTU의 ID 값을 가지고 서버에 접근할 경우 올바른 RTU와 해쉬 값을 다르게 계산하기 때문에 서버와의 통신이 불가능하다.

V. 결론

본 논문에서는 대용량 저장 공간이 있는 서버와 RTU간, 서버와 서버 통신구조에서 실시간 스트림암호화를 통해 기밀성과 무결성을 보장한다. 통신 당사자 간의 랜덤수열을 알고 있다고 하더라도, 서버에서 사용되는 무리수 값을 알 수 없기 때문에 키를 쉽게 유추할 수 없고, 키 값을 알게 되더라도 일정간격마다 키를 갱신하기 때문에 복호화를 할 수 없어 재사용공격이 불가능하다. 본 논문에서 제안한 무리수 기반의 Pair-wise Key 생성 프로토콜을 이용하여 SCADA 시스템 외에도 저장 공간이 충분한 환경이라면 사용될 수 있을 것으로 기대된다.

참고문헌

[1] 김영진, 이정현, 임종인 “SCADA 시스템의 안정성 확보방안에 관한 연구” 정보보호학회논문지, 제 19권 제 6호, 2009. 12, pp. 145-152

[2] 김인중, 정윤정, 고재영, 원동호 “중요핵심기반시설(SCADA)에 대한 보안 관리 연구” 통신학회논문지, 2005.8 Vol.30 no.8C pp.838-848.

[3] B. Cheryl, G. Donald, N. William and T.Mark, “Key Management for SCADA.” Sandia National Labatory. Mar. 2002.

[4] D. Robert. B. Colin, D. Ed and G. N. Juan, “SKMA a key management architecture for SCADA systems,” Australasian Information Security Workshop, vol. 54, pp.138-192, 2006.

[5] Hossein Ghodosi, Chris Charnes, Josef Pieprzyk, Rei Safavi-Naini “Pseudorandom Sequences obtained from Expansions of Irrational Numbers” Proc. of Cryptography Policy and Algorithms Conference. 1995, pp.165-177.

[6] Jing Wang, Guo-ping Jiang “Irrational-based Cryptosystem” Chaos-Fractals Theories and Applications (IWCFTA), 2010 International Workshop on, pp.139-143.