프로토콜의 단계를 고려한 RFID 인증프로토콜 검증

정장영○ , 홍영식
동국대학교 컴퓨터공학과
{ sd109○, hongys } @dongguk.edu

RFID Authentication Protocol Verification With Protocol Phase

JangYoung Chung, YoungSik Hong
Department of Computer Engineering, Dongguk University

요 약

RFID처럼 개방형 네트워크의 특성상 프로톨콜이 노출 되는 문제점이 발생한다. 또한 광범위한 지역에서 RFID사용과 임시적인 RFID사용은 비용 상 문제점이 있다. 본 논문은 RFID의 요소 중 DB(Server)를 제외한 리더와 태그 간의 통신을 통한 인증 프로토콜을 제안하며, 악의적인 사용자에 의해서 이용될 수 있는 정보노출의 문제점을 해결하기 위해 기존 제안된 방식에서 프로토콜의 단계를 고려한 프로토콜을 제안한다. 또한 리더와 태그만의 구성으로 비용을 절감하고자 하며, AVISPA를 이용하여 제안한 프로토콜의 안전성을 검증 한다

1. 서 론

RFID는 기존 사용되었던 바코드에 비해 비, 눈, 안개 등의 환경적인 영향에 따른 인식오류 없이 인식속도가 빠르고 교통, 은행, 물류, 의료 등의 많은 분야에서 사용되는 기술이다. 이러한 RFID는 일반적으로 DB(Server), 리더, 태그로 구성이 되며, DB는 태그에 대한 정보를 저장하는 곳이며, 리더와 DB는 SSL같은 안전한 통신망을 이용한다.

리더는 태그의 정보를 받아 식별하는 장치로서 DB에서 태그의 정보를 전달하는 역할을 수행한다.

그리고 RFID의 특성상 RF를 사용하기 때문에 악의적인 사용자에게 노출될 가능성이 높아 공격받을 가능성이 높다. 이로 인해 기존에 제시되었던 기법으로는 인증 및 복제 방지, 위치추적 등의 문제점을 해결할 수가 없었으며 [6]의 제 1 프로토콜, 제 2 프로토콜의 경우 시뮬레이션을 통한 보안성 검증은 이뤄 졌으나 프로토콜의 노출빈도가 높은 단점이 있다..[13]

그리고 전장, 고속도로와 같이 일시적 또는 사용범위가 광범위 하여 비용이 많이 드는 경우 중앙 DB를 이용한 RFID를 사용하기가 어렵고 비용이 많이 들기 때문에 Serverless 환경에서 태그의 인증프로토콜 관한 연구가 진행되고 있다.[3][6]

기존 제안방식들은 Serverless 환경에 적합하지 않았으며 프로토콜의 단계를 높은 제안방식들이 많았다. 이는 관련연구2장에서 살펴보겠다. [2][3][4][5][13][14]

3장에서는 본 논문에서 제시한 RFID 인증프로토콜을 설명하고자 한다. 4장은 제안프로토콜에 대하여 AVISPA를 이용하여 본 논문에서 제안한 인증프로토콜에 대한 보안성과 프로토콜단계를 분석하고 기존연구과 비교한다. 마지막으로 결론과 향후 연구 과제를 제시한다.

2. 관련연구

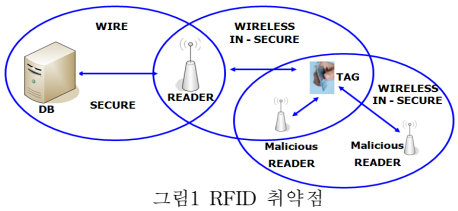

그림1 RFID 취약점

그림 1과 같이 리더와 태그사이에 통신은 무선으로 이루어지기 때문에 도청이 가능하다. 때문에 모든 문제의 원인이 되어 복제, 위치 추적 등의 문제가 발생하게 된다. 이 문제를 막기 위해 HASH-LOCK, RAN DOMIZED-HASH-LOCK, [3],[6] Kill Command 등의 여러 가지 기법이 소개 되었다. 하지만, 비용이나, RFID 특성상 이 문제를 해결하기가 어렵고, 도청으로 인한 여러 가지 문제점이 발생한다.

[14]에서 고려하였듯이 프로토콜 단계를 고려하여 악의적인 사용자에 의해서 공격노출 빈도를 낮추어야 하지만 기존 몇몇의 프로토콜은 노출빈도가 높은 단계를 가지고 있다.

HASH-LOCK기법은 태그의 *ID*의 노출을 막기 위한 방

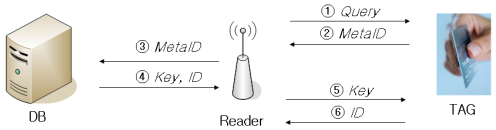

그림 2 HASH-LOCK 프로토콜

법으로 $MetaID = Hash(ID)$와 Key, ID가 매번 같은 정보를 전송하기 때문에 노출이 될 경우 복제와, 위치추적의 위험성이 있다. 또한 Serverless환경에서는 $MetaID$를 이용하기 때문에 부하가 적지만, 도청이 가능하며, 태그의 Key, ID를 쉽게 얻을 수 있고, 이를 이용하여 위치추적, 불법 복제 등

의 문제가 발생한다. 프로토콜의 단계에서 *MetaID*, *Key*, *ID*가 모두 노출이 된다.

RANDOMIZED-HASH-LOCK프로토콜은HASH-LOCK의 기법의 문제점을 보완한 방식이다. 태그에서 난수 r을 생성하고 $r\,H(ID||r)$ 전송하지만 태그의 메모리 용량의 한계로 인하여 r의 범위가 작다. 또한 리더에서 모든*ID*에 r에 대한 계산을 해야 하는 단점이 발생한다.

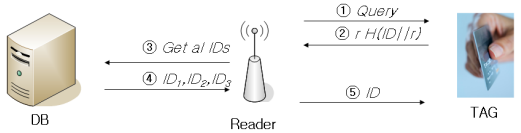

그림 3 RANDOMIZED-HASH-LOCK 프로토콜

그리고 악의적인 사용자에 의해서 $r, H(ID\|r)$이 복제될 경우 상호인증이 이뤄지지 않아 리더는 태그의 복제여부를 알 수가 없다. 그리고 Serverless환경에서 리더의 계산량이 많아지며, 프로토콜의 단계에서$r, H(ID\|r)$, *ID*가 그대로 노출이 되 버린다.

Chiu C. et.al은 Serverless 환경에서 적용한 *Challenge & Response*방식 프로토콜과 리더에서 태그를 인증하는 프로토콜 방식 2가지 인증프로토콜을 제안하였다.

*Challenge & Response*방식은 리더와 태그사이에 태그의 넘버와 리더의*ID*, 랜덤넘버 n_i를 주고 받는다. $[h(f(r_i,t_i)\|n_i,n_j)]_b$를 *CA*와 약속된 전송 비트 b크기만

$$L_i = \begin{cases} f(r_i,t_1) : id_1 \\ \cdots\cdots\cdots : \cdots. \\ f(r_i,t_n) : id_n \end{cases}$$

그림 4 리더의 태그 LIST

보내고 나머지 비트에 대한 질의($quees_r^1, \cdots\cdots quees_r^k$)를 전송한다. 리더에서는 질의에 대한 비트($ans_r, quees_t^1, \cdots\cdots quees_t^k$,)를 전송한다.

Challenge & Response 방식에서 태그의 한계 상 난수생성기에서 n_j를 보내는 것이 아니라 일정 난수 테이블에서 보내기 때문에 범위가 적은 n_j를 통하여 같은 $[h(f(r_i,t_i)\|n_i,n_j)]_b$를 보내게 된다. 이때 위치가 추적 될 수 있다.

리더에서 태그를 인증하는 타입의 프로토콜은 *Challenge & Response*방식과 같이 태그의 넘버와 리더의 *ID*, 랜덤넘버n_i를 주고받으며, $h(f(r_i,t_j))_m$, $h(f(r_i,t_j)\|n_i,n_j)\oplus id_j$를 통하여 리더는 태그를 인증하게 된다. 하지만 제안한 방식에서도 리더가 일정한 n_i, r_i를 전송하게 된다면 특정 태그는 $h(f(r_i,t_j))_m$ 를 전송하게 되어 위치가 추적될 수 있다.

또한 여기서는 태그가 리더를 인증하는 기법이 없어 문제가 발생한다.

[6]의 논문은 [3]의 논문의 보안의 문제점을 보완한 프로토콜로서 해쉬함수와 *xor*을 이용한 2단계과정의 프로토콜을 제안하였다. 태그는 반드시 일정 지역에서 인증을 받기 위해서는 시작점에 있는 리더로부터 갱신을 받으며,

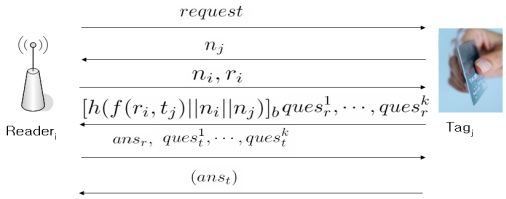

그림 5 *Challenge & Response* 타입 프로토콜

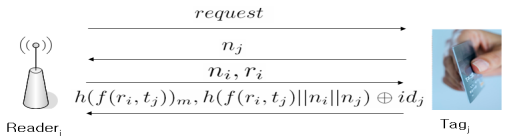

그림 6 리더에서만 태그를 인증하는 타입 프로토콜

리더는 태그의 *ID*, *Key*와 사용되는 리더들의 *ID*와 *Key*를 가지고 있다. 그리고 각 리더들은 인증을 받은 태그의 *ID*와 함께 *TimeStamp*를 저장한다. 이유는 태그를 인증할 때마다 *ID*가 변하기 때문에 마지막으로 사용된 태그의 *ID*를 파악하여 일정주기로 각 리더의 태그의*ID*를 갱신하기위해 사용된다.

그림 7과 같이 리더에서 Query를 보내게 되면 태그는 리더와 태그 사이에 약속된 해쉬함수에 자신의 N_i, K_i를 해쉬함수를 통하여 암호화 한다. 이 값이 리더에게 전송되면 리더는 N_i, K_i를 비교하여 태그의 *ID*를 해쉬하여 보내게 되며 이와 함께 리더의 R_i, RK_i를 태그의 *Key*로 *xor*하여 전송하게 된다. 그 후에 태그는 자신의 *ID*와 *Key*를 리더의 RK_i로*xor*연산하여 갱신 하게 된다. 만약 태그가 리더의 지역에서 벗어나 다시 들어오게 된다면, 마지막에 사용된 *ID*와 *Key*를 사용하게 된다. 그림 7은 제1 프로토콜은 태그의*ID*와 *Key*를 갱신 받는 과정이다. 만약 초기 단계의 프로토콜에서 *ID* 갱신을 받지 않은 상태에서 기존 *ID*의 갱신 없이 리더들이 있는 일정지역에서 사용을 하게 된다면 복제태그를 선별해낼 수 있다.

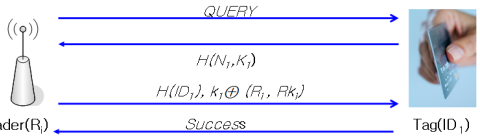

그림 7 초기 단계 사용되는 제1프로토콜

일정지역에서 인증을 받을 경우 사용되는 제2프로 토콜의 절차는 다음과 같다. *Query*를 보내게 되고그 후에 태그에서는 초기 리더로부터 받은 R_i와 그리고 자신의 *ID*를 해쉬하고 $H(R_i)$, $H(ID_1)\oplus H(R_i)$와 같이 보내게 된다. 이것을 태그가 받으면 태그는 $H(R_i)$를 이용하여 자신이 가지고 있는 리더의 리스트와 비교하여 있는지 확인한다.

확인이 끝난 후에는 $H(K_1')\oplus H(R_i)$, $H(K_1)\oplus H(R_i)$를 태그에게 보내서 자신이 가지고 있는 *Key*와 비교하게 된다. 이 절차 후에는 자신의 $H(ID_1)\oplus N_1$를 보내서 리더에서 태그에 대한 *ID*확인하는 것으로 상호인증과정을 마치게 되고, 리더는 태그의 *ID*를 갱신 하기 위해 자신

R_i	리더의 ID
ID_i	태그의 ID
K_i	태그와 리더사이에 비밀키
RK_i	리더와 리더사이에 비밀키
N_i	태그의 넘버
$\oplus$	xor
H	$Hash\ Function$

표 1 표기법

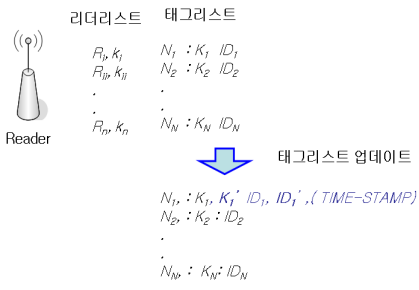

그림 8 리더의 태그와 리더 리스트

$$ID' = ID \oplus RK_i$$
$$Key' = Key \oplus RK_i$$

그림 9 ID, KEY 갱신

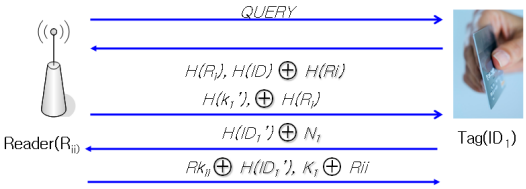

그림 10 일정지역 안에서 인증을 받을 경우에 사용되는 제2 프로토콜

의 RK와 R_i, $RK_{ii} \oplus H(ID')$, $K_1 \oplus R_{ii}$를 전송하며, 그림 9 처럼 자신의 ID와 Key를 갱신 한다.

사용자의 프라이버시를 보호하는 방법으로서 RFID리더가 Kill command를 명령하면 태그의 기능을 상실하게 되는 기법이다. 단점은 재사용이 불가능하여 많은 비용이 든다. 최근 Adi Shamir에 의해서 태그의 전력 소모량을 이용하여 Kill Password 획득하는 방법이 제시되었다.[4][11]

3. 제안프로토콜

Serverless환경에서는 기존에 제안되었던 방식들을 적용하기가 어렵고 보안상 취약점이 있다. 또한 기존 제안된 [3]의 경우 상호인증의 과정이 없고, 일정한 넘버를 전송하게 되면 태그에서는 같은 결과를 전송하기 때문에 위치추적, 리더에 대하여 복제가 가능하게 된다. [6]의 경우 [3]의 제안한방식의 보안의 문제점을 보완하였지만 프로토콜 단계에 있어 기존 제안된 방식에 비하여 프로토콜의 단계가 많은 단점이 있었다. 이러한 단점을 보완하기 위하여 기존 [6]에서 제안된 프로토콜을 이용하되 프로토콜의 단계를 고려하여 정보 노출의 빈도를 줄이고자한다.

[6]의 제안 목표처럼 전방향 보안성, 기밀성, 익명성, 상호인증, 불추적성, 무결성을 보장하고자 한다.

불법 복제된 태그를 검출하기위해 모든 리더들은 일정주기를 기준으로 바뀐 태그의 ID를 갱신 하고, 자신의 Key

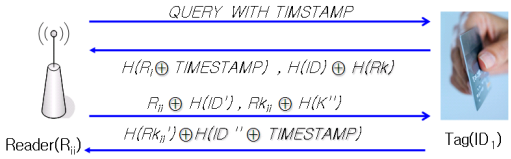

그림 11 제안 제2 프로토콜

를 교체한다. 이 과정은 전 방향 안전성과 불 추적성을 위한 것이다.

태그는 반드시 시작점에 있는 리더에게서 태그의 ID와 Key의 갱신을 반드시 받아야 한다. 리더는 DB의 역할을 동시에 수행해야하기 때문에 저장기능과 계산 기능을 가진다.

제1프로토콜 경우 $Query$와 $Success$단계를 제외하면 상호인증을 위한 2단계 프로토콜을 사용하기 때문에 기존 프로토콜을 사용한다.

제2프로토콜의 경우 총 $Query$를 사용하는 단계를 포함 5단계의 프로토콜을 사용하게 되는데 제안하는 프로토콜의 경우 기존 논문에서 사용된 프로토콜의 절차를 4단계로 줄여 악의적인 사용자에 의해서 정보 노출의 빈도를 줄이고자 한다.

본 논문에서 제안하는 제2프로토콜의 수정은 그림11과 같다. 기존 제안된 방식과 달리 $TimeStamp$의 난수성을 이용하여 절차를 줄이도록 하였다. 절차는 리더에서 $Query$와 함께 $TimeStamp$를 보낸다. $Query$와 $TimeStamp$를 받은 태그는 자신이 가지고 있던 이전단계의 리더의 ID인 Ri에 $TimeStamp$를 xor하여 해쉬를 취하고 자신의 본래 ID의 해쉬값과 리더의 키 RK의 해쉬값을 xor하여 보낸다.

그 후 리더는 $TimeStamp$의 값을 알고 있으므로 자신의 리더리스트의 모든 리더들의 ID에 $TimeStamp$를 취 하여 전송된 $H(R_i\ xor\ Time\ stamp)$값을 비교하여, 자신의 리스트중에 지금 현재의 태그의 ID와 Key를 리더의 ID와 Key로 xor하여 전송한다. 태그는 전송받은 정보를 이용하여 해쉬한 리더의 ID에 다음단계에서 쓰일 태그의 ID와 $TimeStamp$를 xor하고 해쉬하여 전송하게 된다.

이때, 리더는 기존ID와 다음단계에 쓰일 태그의 ID의 갱신의 정보를 통하여 태그를 인증할수 있으며, 태그는 자신의 본래의ID정보를 전송하고 리더로부터 자신의 현재 ID를 전송받아 리더를 인증하게 된다.

4. 실험결과 및 분석

4.1 실험 환경

본 논문에서 제시한 프로토콜은 AVISPA (Autom ated Validation of Internet Security Protocols andApplications) [4][11]를 이용하여 기존의 프로토콜과 제안 프로토콜을 시뮬레이션 하여 비교하였다.

AVISAPA는 HERMES, BAN logic, GNY logic, SvO logic, FDR, CAPSL, ISL, AAPA 등의 기존 검증도구에 비해 프로토콜의 세션구분과 노드의 표현과 공격자의 새로운 KEY나 nonce등에서 향상된 표현을 가능하게 한다.

HERMES는 간단한 프로토콜은 빠르게 검증할 수 있으나. AVISPA에 비해 세션이 늘어날수록 느리고 검증의 문제점이 발생하였다.[12]

AVISPA는 정형 검증을 통하여 분석하는 도구로서 HLPSL의 입력으로 IF로 변화되어 4가지의 모듈(OFMC, CL-AtSe, SATMC, TA4SP)에 입력되어 결과를 보여준다.

구분	HASH - LOCK①	RANDOMIZED-HASH LOCK②	Serverless -Challenge & Response 방식③	Serverless -리더에서만 태그를 인증하는 방식④	[6] 제1프로토콜 ⑤	[6] 제1프로토콜 제2프로토콜 통 합 ⑥	제1프로토콜 제안 제2 프로토콜 통합 ⑦
OFMC	UNSAFE	UNSAFE	UNSAFE	UNSAFE	SAFE	SAFE	SAFE
CL-AtSe	UNSAFE	UNSAFE	UNSAFE	UNSAFE	SAFE	SAFE	SAFE
SATMC	INCONCLUSIVE	UNSAFE	SAFE	INCONCLUSIVE	SAFE	SAFE	SAFE
TA4SP	INCONCLUSIVE	INCONCLUSIVE	INCONCLUSIVE	INCONCLUSIVE	INCONCLUSIVE	INCONCLUSIVE	INCONCLUSIVE

표 2 실험결과

시뮬레이션은 제1 프로토콜의 독립적인 실험은 제외하고 제1프로토콜과 제안된 제2프로토콜을 통합하여 검증하였다.

(단위 : 시간(s))

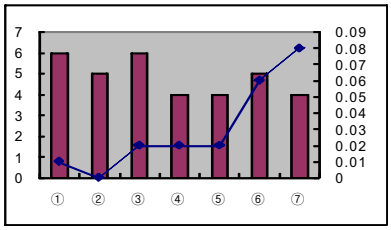

그림 12 프로토콜단계비교 및 OFMC 검증 측정시간

4. 2 결과 및 분석

AVISPA검증결과는 표2와 같으며, 그림 12의 경우 AVISPA 측정 시간을 나타낸 것으로 1과2,3,4,5,6의 경우 판정까지 0~0.06s가 측정되었다. 본 논문에서 제안하는 프로토콜 7번의 경우 SAFE 판정까지는 0.08s가 측정이 되었는데 이는 기존 제안된 방식에 비해서 높게 측정되었다, 이는 제안하는 프로토콜이 기존의 제안된 방식보다 프로토콜의 단계는 적어졌으나 복잡성이 높기 때문이다.

그림 12와 같이 최소는 1단계 최대는 2단계에 적은 프로토콜단계를 거쳐 인증을 받는 구조 이면서, ①,②,③,④의 프로토콜들이 제안하지 못 했던 상호인증(Mutual Authentication)이 가능한 구조를 가지고 있다. 그리고 기존 제안된 프로토콜에 비교하여 최소1단계에서 최대 2단계 적은 프로토콜 단계를 보여주고 있다.

[6]의 논문처럼 일정지역 안에서 사용되는 프로토 콜의 경우 리더의 *Key*로써 자신의 *ID*와 *Key*를 갱신한다. 그리고 리더의 *Key*와 저장하고 있는 태그의 *ID*도 일정주기로 갱신이 되기 때문에 전방향보안성, 해쉬함수를이용한 기밀성, 그리고 태그의 *ID*는 매 세션마다 갱신이 되어 익명성 및 불추적성, 무결성을 보장하며 프로토콜의 단계를 고려하였기 때문에 노출빈도를 줄이게 되었다.

5. 결론 및 향후 연구

본 논문에서는 RFID의 3개의 기본구조 중 DB를 제외한 환경에서 리더와 태그만으로 프로토콜의 단계를 고려하여 인증하는 프로토콜을 AVISPA를 이용하여 검증하였다. 이를 통하여 RFID 시스템의 안정을 확보함과 동시에 비용을 감소시킬 수 있어서 RFID의 사용을 확대할 수 있을 것으로 예상된다.

또한 프로토콜의 단계를 기존논문에 비해 줄여 악의 적인 공격자에 의해서 정보의 노출로 인해 피해를 줄일 것으로 예상된다.

향후 연구에서는 2단계에 인증프로토콜을 간소화 시키고 TA4SP의 결과가 불분명하게 나왔기 때문에 이에 관한 연구와 제1•2프로토콜을 통합하게 될 경우 경량화에 문제점을 보완하며 태그들 중 특정 태그의 찾기 위한 연구를 진행하고자한다.

참고문헌

[1] F. Jenings, "Active RFID Standard - Why Issues and Solution",RIFD/USN KOREA 2007 International Conference, NOV, 2007.

[2] A. Jules, "RFID Security and Privacy : A Research Survey ", IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL24.NO2, FEB.2006.

[3] Chiu C. Tan,Bo Sheng, Qun Li, "Severless Search and Authentication Protocols for RFID", Fifth Annual IEEE International Conference on Volume , Issue , 19-23 March 2007.

[4] 강미영, 오정현, 이송희, 최진영, "AVISPA을 이용한 RFID 보안 Protocol의 명세 및 검증" , 한국정보과학회 추계학술대회, 2007.

[5] 오정현, 김현석, 최진영, "RFID 보안Protocol의 취약성 분석 및 설계" 한국정보과학회 가을학술발표논문집 Vol33,No.2(C), 2000.

[6] 정장영, 홍영식 "Serverless 환경에서 RFID 인증프로토콜 검증" 한국정보과학회 2008 종합학술대회 논문집 제35권 제1호(A), 2008

[7] H. Kim, "Security and Privacy Issues in RFID", RIFD/USN KOREA 2007 International Conference, NOV, 2007.

[8] S.A.Weis, S. E Sarma, R.L. Rivest, D.W.Engles, "Security and Privacy Aspects of Low-Cost Radio-Frequency Identification System", Security in Pervasive Computing 2003, LNCS 2802, pp201-212, Springer-Verlag Heidelberg, 2004.

[9] http://www.epcglobalinc.org/

[10] http://www.autoidlabs.org/

[11] http://www.karus.or.kr/

[12] http://www.avispa-project.org/

[13] M. Hussain, D. Seret " A Comparative study of Security Protocols Validation Tools: HERMES vs. AVISPA" , ICACT 2006, 2006.

[14] Xiang Zhang, Baciu, G."Low Cost Minimal Mutual Authentication Protocol for RFID" Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on