情報保護學會論文誌
第14券 第6號, 2004. 12

상태기반 RFID 인증 프로토콜*

유 성 호[a)†‡], 김 기 현[a)], 황 용 호[a)], 이 필 중[b)]
포항공과대학교[a)], 포항공과대학교/KT[b)]

Status-Based RFID Authentication Protocol*

Sung Ho Yoo[a)†‡], KiHyun Kim[a)], Yong Ho Hwang[a)], Pil Joong Lee[b)]
POSTECH[a)], POSTECH/KT Research Center[b)]

요 약

근래에, Radio Frequency Identification (RFID) 시스템은 생산, 공급망관리, 재고관리 등의 분야에서 유용하게 사용될 만한 기술로서 산업계에서 많은 관심을 받고 있다. 가까운 미래에는 저렴한 가격의 RFID 태그나 스마트 라벨이 현재 사용되고 있는 바코드 대신 상품관리에 활용될 것으로 알려져 있다. 그러나 작고 값싼 RFID태그의 생산과, 사용자 프라이버시 보호를 위한 안전한 인증프로토콜의 개발은 아직 해결해야 할 문제들이다. 반도체 기술의 발전에도 불구하고, 태그의 계산과 저장능력은 제한되어 있으므로 기존 다른 시스템에 적용되던 암호시스템의 적용은 현실적으로 쉽지 않다. 그러므로 RFID시스템에서 사용할 수 있도록 더 작은 저장 공간과 더 작은 계산량이 소요되는 새로운 프로토콜의 개발이 필요하다. 본 논문에서는 위치트래킹 공격과 스푸핑 공격에 안전한 RFID인증 프로토콜을 제안한다. 제안한 프로토콜은 기존 인증프로토콜보다 데이터베이스에서의 계산량이 상당히 감소하였으므로 프라이버시 보호를 위한 실질적인 해결책으로 사용이 가능하다.

ABSTRACT

Recently, Radio Frequency Identification (RFID) systems stands in the spotlight of industry as a common and useful tool in manufacturing, supply chain management (SCM) and stock management. In the near future, low-cost RFID Electronic Product Codes (EPC) or smart-labels may be a practical replacement for optical barcodes on consumer items. However, manufacturing cheap and small RFID tags, and developing secure RFID authentication protocols are problems which need to be solved. In spite of advances in semiconductor technology, computation and storage ability of the tag are so limited that it is difficult and too expensive to apply existing crypto-systems to RFID tags. Thus it is necessary to create a new protocol which would require less storage space and lower computation costs and that is secure in the RFID system's environments. In this paper, we propose a RFID authentication protocol that is secure against location tracking and spoofing attacks. Our protocol can be used as a practical solution for privacy protection because it requires less computations in database than the previous RFID authentication protocol.

Keywords : *RFID, Authentication protocol, Location privacy*

Ⅰ. 서 론

RFID (Radio Frequency IDentification) 시스템은 무선 주파수를 이용하여 물리적 접촉 없이 개체에 대한 정보를 읽거나 기록하는 자동인

접수일 : 2004년 8월 17일 ; 채택일 : 2004년 11월 22일
* 본 연구는 대학 IT 연구센터 육성·지원사업, 교육부 두뇌 한국 21사업의 지원으로 수행되었음.
† 주저자, ‡ 교신저자 : shyoo@oberon.postech.ac.kr

식기술 시스템이다. 바코드에 비해서 저장능력이 뛰어나며 비접촉식이므로 리더와의 시야확보를 고려할 필요도 없으며 인식 속도가 빨라 물류시스템에서 바코드를 대체할 인식 시스템으로 많은 연구가 진행되고 있다. 근래에는 교통요금 지불시스템, 가축관리, 산업 자동화, 의료분야 등에서도 일부 활용되고 있다.

그러나 RFID시스템이 물리적 접촉 없이도 인식이 가능하다는 특징은 안전성과 프라이버시측면에서 기존에 발생하지 않았던 여러 문제들을 발생시킨다. 예를 들어, 식별 가능한 정보를 그대로 전송하는 태그의 경우, 태그와 리더사이의 통신내용은 제 3자에 의해 쉽게 도청이 가능하다. 또한 공격자는 도청한 정보를 바탕으로 위치트래킹 등의 공격을 행할 수 있으며 이는 소비자의 프라이버시 침해로 직결된다. 그러므로 RFID시스템을 사용한 여러 응용에 대한 연구뿐만 아니라 시스템 적용을 통해 발생 가능한 여러 보안, 프라이버시 문제를 해결하는데도 많은 연구가 이루어 져야 한다.

프라이버시 문제를 해결하기 위해서 가장 간단하면서도 확실한 방법은 태그가 부착된 상품의 판매 이후 계산대에서 태그를 파괴하거나, 'kill' 명령어[1, 2]를 사용하여 태그를 사용하지 못하게 하는 것이다. 그러나 홈 네트워킹이나 위치기반서비스(Location Based Service)등 RFID시스템의 활용범위가 넓어지면서 단순히 태그를 무력화시키는 방법은 여러 문제점을 낳게 되었다. 그러므로 정당한 리더와 데이터베이스는 태그를 이용하여 사용자에게 유용한 서비스가 가능하도록 하되, 그 외의 개체들은 태그에 대한 어떠한 정보도 얻을 수 없게 함으로써 사용자의 프라이버시를 보장하도록 하는 방법들이 제안되고 있다. 이를 위해 기존 무선 환경에서 제공하던 프로토콜의 적용을 고려할 수 있으나 RFID태그 가격이 낮아야 한다는 제약사항으로 인해 이것이 현실적으로는 불가능하다. 태그의 상용화를 위해서는 수동형 태그의 가격이 5센트 미만이어야 하며 5센트짜리 태그를 만들기 위해서는 IC(Integrated Circuit) 가격이 2센트를 초과해서는 안 된다[3]. 낮은 가격은 게이트의 수를 제한하므로 DES[4], AES[5], SHA-1[6], HAVES[7] 같은 알고리즘들의 사용 가능 여부도 아직까지는 불투명하다(물론 고가의 태그에는 이미 적용되고 있다). 이의 해결을 위해서는 낮은 가격에 좋은 성능을 가지는 IC를 개발/생산하는 것도 중요하지만, 자원의 소모가 적으면서도 안전한 암호 알고리즘의 개발과 아울러 최소의 자원을 사용하면서도 안전한 프로토콜의 개발도 필수적이다[3,8].

본 논문에서는 RFID태그 자원의 한계를 고려하여 최소한의 연산/저장자원을 사용하면서도 사용자의 프라이버시를 보호하기 위해 기존에 제안된 다른 프로토콜[9,10,11]보다 개선된 프로토콜을 제안한다. Ⅱ장에서는 RFID시스템에 대한 위협요소 및 설계 시 고려사항에 대해 설명하고, Ⅲ장에서는 기존에 제안된 프로토콜들에 대한 설명과 특징, 문제점을 기술한다. Ⅳ장은 본 논문에서 제안하는 프로토콜에 대해 설명하고, Ⅴ장은 Ⅱ장에서 기술한 위협요소에 대해 제안한 프로토콜이 안전함을 보인다. 마지막으로 기존 프로토콜과의 비교와 함께 Ⅵ장에서 결론을 맺도록 한다.

1. 용어정의

ID : 태그의 고유 식별자
n : 데이터베이스가 보유한 ID의 갯수
IK : *ID*검색 시 사용되는 index key.
cnt : 카운터
flag : 이전 세션의 상태를 나타내는 값
TID : Transaction의 ID
LST : 마지막으로 성공한 transaction의 ID
ΔTID : $LST - TID$
AE : DB에서 연관된 행을 가리키는 지시자
$h(\ )$: 해쉬 함수
$L(\ )$: 입력 값의 왼쪽 반을 출력하는 함수
$R(\ )$: 입력 값의 오른쪽 반을 출력하는 함수
|| : 문자열 연접 연산
xor : Exclusive or 연산

Ⅱ. 위협요소 및 설계 시 고려사항

RFID시스템은 리더와 태그간의 통신이 무선으로 이루어지며 tag의 연산능력과 저장 공간의 제약으로 인해 많은 취약점들을 가지므로 여러 위협에 노출되기 쉽다. 이러한 취약점들은 공격자가

기존 다른 시스템에서 보다 더 적은 노력으로도 원하는 목적을 달성할 수 있게 한다. 지금부터 RFID시스템에 대해 공격자가 행할 수 있는 공격 방법들에 대해 알아보고 이런 공격들에 대비하기 위해 RFID 인증프로토콜을 설계함에 있어 고려해야 할 사항들에 대해서 알아본다[12].

RFID시스템에서 발생할 수 있는 주요 공격방법은 아래와 같다.

- 도청 (Eavesdropping) : 태그와 리더간의 통신방식은 무선이므로 공격자는 큰 노력 없이도 통신내용을 엿들을 수 있다. 엿들은 내용은 이후 설명할 여러 공격방법의 기본정보로 활용이 가능하다. RFID시스템에서 도청공격은 불가피함을 가정해야 하나, 공격자가 도청을 통해서 얻은 내용을 통해 다른 공격에 활용 가능한 어떠한 정보도 얻을 수 없도록 해야 한다.
- 통신내용분석 (Traffic analysis) : 공격자는 도청을 통해서 얻은 내용을 분석하여 리더의 질의에 대한 태그의 응답을 예측할 수 있다. 이렇게 예측한 정보를 통해 공격자는 태그의 이동경로를 트래킹하는 공격 등에 활용할 수 있으며 이는 태그소유자의 프라이버시를 침해하는 주요원인이 된다.
- 위치트래킹 (Location Tracking) : 위치트래킹 공격이란 공격자가 공격자 혹은 악의적인 리더가 태그의 위치변화를 감지함으로써 태그 소유자의 이동경로를 파악하는 방법으로 사용자의 프라이버시를 침해하는 유형중의 하나이다. 위치트래킹에 안전하기 위해서는 태그가 부착된 상품을 판매한 후, 물리적으로 태그를 파괴하거나, 'kill' 명령어[1,2]를 사용하거나, 블로커 태그[12]를 이용함으로써 위치트래킹 공격에 안전할 수 있다. 위의 방법들은 태그의 ID를 알고 있는 데이터베이스까지도 태그의 이동경로를 파악할 수 없으므로 사용자의 위치프라이버시를 완벽하게 보장할 수 있다는 장점이 있으나, 이후 홈네트워킹이나 위치기반서비스 등에서 태그의 정보를 활용해야 하는 경우 사용이 불가능하거나 용이하지 않다. 그러므로 근래에는 판매 이후에 태그를 여러 응용서비스에서 활용해야 하는 경우, 정당한 데이터베이스의 경우 태그의 위치를 파악할 수 있도록 하고 정당한 데이터베이스를 제외한 개체들은 태그의 위치뿐만 아니라 태그의 이동경로조차 알지 못하게 함으로써 사용자의 위치프라이버시를 보장하는 방법을 사용하고 있다[13].
- 스푸핑 (Spoofing) : 스푸핑이란 정당하지 않은 개체를 정당한 것처럼 속여 인증과정을 통과하는 방법이다. 스푸핑은 그 대상에 따라 두 가지로 구분할 수 있다. 먼저 공격자가 태그로 위장하여 정당한 리더를 속이는 방법과, 반대로 공격자가 리더로 위장하여 태그를 속이는 방법이 있는데, 첫 번째의 경우 값싼 물품에 부착된 태그의 정보를 이용하여 비싼 물품의 구매 시에 리더를 속이거나, 건물 출입허가가 있는 사용자의 출입증에 부착된 태그로부터 정보를 획득하여 출입통제기를 속이는데 사용 가능한 공격방법이다. 두 번째 방법은 태그 내에 정보를 가지고 있을 경우, 공격자는 정당한 리더인척 하여 태그의 정보를 알아낼 수 있다.
- 메시지 유실 (Message loss) : 공격자의 고의 또는 시스템상의 문제로 인해 태그와 리더간에 주고받는 통신내용의 일부가 유실될 수 있다. 이는 인증세션의 비정상적인 종료뿐만 아니라 메시지 유실로 인해 둘 사이의 동기가 어긋날 경우 자칫 데이터베이스가 태그의 ID를 잃어버리는 경우가 발생할 수도 있다.
- 서비스 거부(Denial of Service) : RFID 시스템이 정상적으로 작동하지 못하도록 하기 위해 특정 주파수를 갖는 방해전파를 방출하는 등의 공격방법을 말한다.
- 물리적 공격 (Physical attack) : 태그는 생산가격의 제한으로 인해 고가의 시스템에 사용되는 고가의 메모리나 칩을 사용하기가 힘들다. 그러므로 프로브공격[1]이나, TEMPEST공격[2] 등에 취약하다.

1) 칩에 탬퍼 방어 패키지를 제거하여 직접 IC Chip에 프로브를 해 중요정보를 해석하는 수법

2) 통신장비 및 컴퓨터에서 방출되는 전자파를 분석하여 이들 사이에 송수신되는 내용을 도청할 수 있는 공격법

위에서 설명한 위협요소들 중에서 서비스 거부 공격과 물리적 공격은 RFID시스템의 기계적/물리적인 특성에 기인한 공격방법이며 인증프로토콜 설계 시 고려할 수 있는 사항이 아니므로 본 논문에서는 언급하지 않는다.

위에서 열거한 공격에 안전하며 RFID태그의 특성과 관련하여 인증프로토콜을 설계함에 있어 고려해야 할 사항은 아래와 같다.

- 태그소유자의 프라이버시 침해 방지.
- 통신상의 내용을 공격자가 엿듣더라도, 공격자는 유용한 어떤 정보도 얻을 수 없어야 함.
- 위치트래킹을 방지하기 위해서는 정당한 데이터베이스와 리더이외의 개체에게는 태그의 이동경로를 파악할 수 있는 어떠한 정보도 제공해선 안 됨.
- 태그에 저장된 내용은 정당하지 않은 리더가 읽거나 써서는 안 됨.
- 스푸핑 공격에 안전하기 위해 상대 개체의 응답을 추측하여 공격자가 응답내용을 임의로 생성할 수 없어야 함.
- 태그의 연산능력과 저장능력을 고려하여 인증에 필요한 계산 양과 저장 공간을 최소화해야 함.
- RFID시스템에서의 인증은 여러 개의 태그에 대해 짧은 시간 내에 이루어져야 하므로 서버가 인증을 처리하는데 소요되는 시간을 최소화해야 함.

Ⅲ. 기존 RFID 인증프로토콜

이제부터 기존에 제안된 RFID 인증프로토콜에 관해 알아보고, 각각의 프로토콜에 대해 위에서 언급한 위협요소에 대한 안전성여부를 설명하도록 한다.

Henrici와 Muller는 [9]에서 해쉬에 기반하여 *ID*를 갱신함으로써 위치트래킹 공격을 방지하는 프로토콜을 제안하였다(그림 1). 상품제조자는 $h(ID)$, *ID*, *TID*, *LST*, *AE*를 저장할 수 있는 데이터베이스를 구축하고 태그에는 *ID*와 *TID*, *LST*를 저장한다. 질의를 받은 태그는 *TID*를 1 증가시키고 $h(ID)$, $T=h(TID \text{ xor } ID)$, ΔTID를 계산하여 리더에게 전송한다. 데이터베이스는 $h(ID)$로 *ID*를 검색하여 해당 *TID*에 ΔTID를 더하여 T'를 계산한다. 그림 1 에서 ③의 T와 T'가 일치하면 데이터베이스는 Q를 계산하여 전송하고 *ID*를 갱신하기 위해 랜덤하게 생성한 R과 xor 연산을 수행한다. ⑤를 받은 태그 역시 Q'를 계산하여 Q와 일치할 경우 자신의 *ID*를 갱신하게 된다. *AE*는 이전 *ID*에 대한 정보를 가짐으로써 시스템상의 문제 또는 공격자의 고의로 인한 메시지의 유실에 안전하도록 하였다.

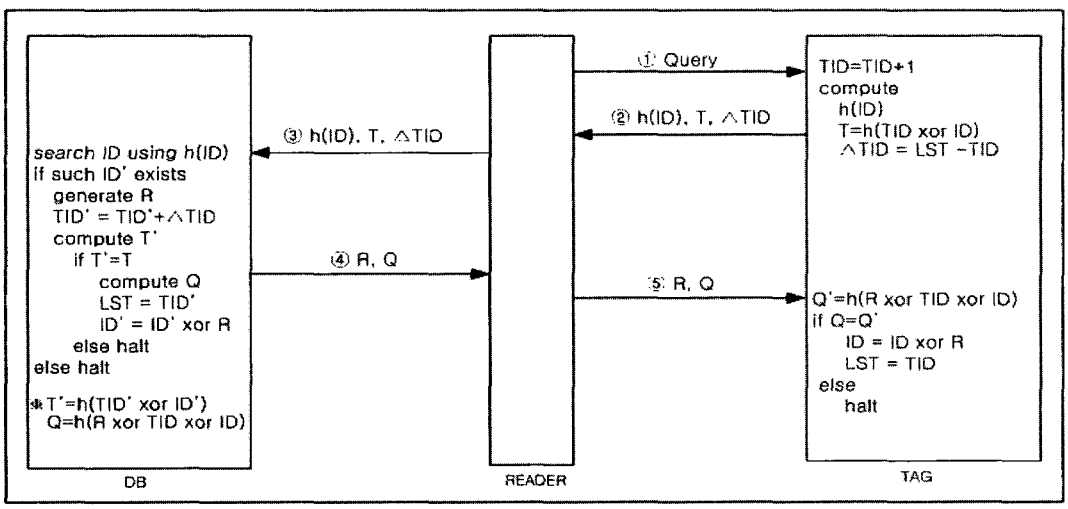


그림 1. 해쉬기반 ID 변형 프로토콜[9]

이 프로토콜은 인증이 완료될 경우 ID가 갱신되므로 위치트래킹 공격에 안전한 듯 보이지만 태그와 데이터베이스 사이에 정상적이지 않은 인증의 경우(예. 공격자가 공격의 목적으로 태그에게 질의를 하는 경우) 태그는 항상 동일한 $h(ID)$를 응답하므로 공격자는 태그의 위치를 트래킹 할 수 있다. 또한 스푸핑 공격에도 안전하지 못한데 공격자는 질의를 통해 ②를 얻어낼 수 있으며 태그가 정상적인 인증세션을 열기 이전에 데이터베이스와의 세션에서 ②를 전송하게 되면 데이터베이스는 공격자를 정당한 태그로 인증할 수밖에 없게 된다[10]. 또한 공격자가 세션 중간에서 리더가 태그에게 전송하는 ⑤의 값에서 R을 연속된 0으로 이루어진 문자열로 주고 Q 대신 T를 전송하면 태그는 에러를 감지하지 못하며 ID를 ID xor 0으로 갱신하므로 다음 인증 시, 서버는 $h(ID)$로 기존 ID정보는 찾을 수 있으나 태그와 데이터베이스에 저장된 기존 ID에 대한 LST가 일치하지 않으므로 태그는 인증을 받지 못하게 된다.

황 영주 등은 위에서 설명한 [9]의 문제점인 스푸핑 공격에 대해 안전하고, 태그의 해쉬 횟수를 3번에서 2번으로 줄인 프로토콜[10]을 제안하였다(그림 2). 리더는 의사난수생성기를 이용하여 랜덤한 S를 생성하여 태그에게 질의를 보낸다. 질의를 받은 태그는 $h(ID)$와 $R=h(ID||S)$를 생성 후, R의 왼쪽 반인 $L(R)$을 리더를 통해 데이터베이스에 전송한다. 데이터베이스는 $h(ID)$를 통해 ID를 검색하여 리더로부터 받은 S와 함께 R'를 생성한다. 생성한 R'의 왼쪽 반인 $L(R')$과 전송받은 $L(R)$을 비교하여 일치할 경우 정당한 태그로 인증하고 ID를 R'로 xor 하여 갱신한 후, R'의 오른쪽 반인 $R(R')$을 전송한다. 태그는 $R(R)$과 $R(R')$를 비교하여 일치할 경우 ID를 R과 xor를 수행하여 갱신한다. 이 프로토콜은 리더가 랜덤한 S를 질의하며 S에 해당하는 R을 공격자가 만들어 낼 수 없으므로 [9]에서 발생 가능한 스푸핑 공격에 안전하며, 인증에 사용되는 값이 해쉬한 결과의 반이여서 해쉬를 두 번만 수행하므로 효율적이다.

그러나 이 프로토콜 역시 위치트래킹 공격에 여전히 안전하지 못한데 이는 [9]에서와 같은 이유이다. 또한 이 프로토콜은 태그가 데이터를 가지는 시스템에 적용될 경우, 공격자는 리더인 척 가장하여 태그를 속이고 태그 내의 데이터를 얻을 수 있는 스푸핑 공격을 행할 수 있다. 공격자는 리더와 태그사이의 통신을 도청하여 ①, ②, ⑤를 얻고, ⑤는 태그에게 주지 않고 가로챈다고 가정한다. 그럼 태그의 ID는 변경되지 않을 것이고 공격자가 ①과 동일한 S를 태그에게 질의하고 ②의 응답에 대해 도청한 ⑤를 태그에게 주면 태그는 리더를 인증하고 데이터를 주게 된다.

Ohkubo 등은 위치트래킹 공격에 안전하며, 전방위 안전성도 보장되는 해쉬체인 프로토콜[11]을 제안하였다(그림 3). 데이터베이스에는 ID와

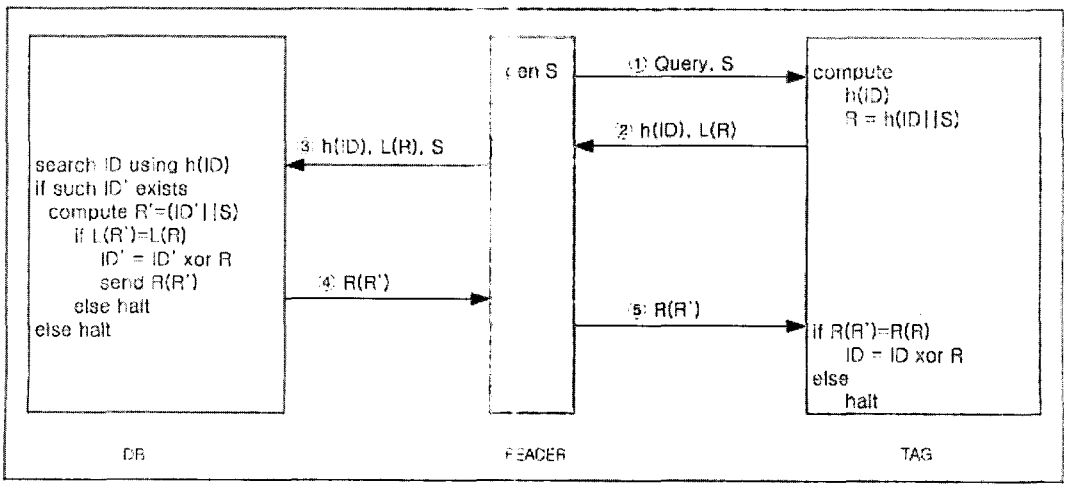


그림 2. 향상된 해쉬기반 ID 변형 프로토콜[10]

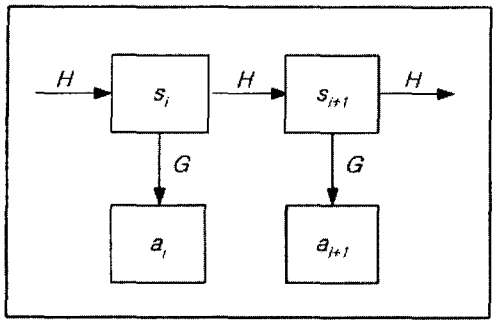


그림 3. 해쉬체인 프로토콜[11]

비밀 값 s_1이 저장되며 태그에는 동일한 s_1값을 저장하고, 두개의 해쉬함수인 H, G를 구현한다. 리더의 질의에 대해 태그는 $G(s_i)$를 수행하여 리더에게 응답하며 자신의 비밀 값인 s_i는 $H(s_i)$를 통해 s_{i+1} 으로 갱신한다. 리더의 질의에 대해 태그는 매번 다른 응답을 하므로 공격자는 태그의 이동경로를 파악할 수 없게 된다.

그러나 이 프로토콜에서 서버는 태그의 ID와 초기 비밀 값인 s_1만을 가지고 있으므로 태그로부터 온 $a_i = G(s_i)$값에 해당하는 ID를 검색하기 위해서는 최악의 경우, 데이터베이스가 보유한 모든 s_1에 대해서 H와 G를 i번[3] 수행해야 한다. 더군다나 태그로부터 잘못된 응답이 왔을 경우, 데이터베이스는 보유한 모든 ID에 대해 무한번의 해쉬를 수행할 가능성도 배재할 수 없다. 또한 해쉬체인 기반 기법은 일 방향 인증으로서 태그가 리더를 인증하지 못하므로 태그의 동작을 제어하는 리더의 명령을 수행함에 있어 문제가 발생하게 된다[15].

Ⅳ. 제안 프로토콜

지금부터는 Ⅱ장에서의 설계 시 고려사항을 바탕으로 모든 위협요소에 대해 안전한 프로토콜을 제안한다. 제안할 프로토콜은 기존 프로토콜들의 가장 큰 문제점인 위치트래킹 공격에 안전하면서도 일반적인 경우에는 서버에서 해쉬연산 없이 ID를 검색할 수 있으며 이전 인증세션이 정상적으로 종료되지 않은 경우, $\sqrt{n}$(n은 데이터베이스가 보유한 ID의 개수)번의 해쉬연산으로 ID를 검색할 수 있으므로 기존에 위치 트래킹 공격을 방지하는 프로토콜인 해쉬체인 프로토콜[11]보다 훨씬 효율적이다.

제안하는 프로토콜의 가장 큰 특징은 이전 인증 세션의 성공 여부에 따라 동작방법이 달라진다는 점이다. 이전 인증세션에서 리더로부터 정당한 응답을 받아 인증이 정상적으로 완료된 경우와, 공격자의 고의 또는 시스템상의 문제로 인해 정당한 응답을 받지 못해 인증이 정상적으로 종료되지 않은 경우로 나뉘어 지며 그 상태에 따라 태그와 데이터베이스의 동작방법이 달라진다. 태그는 이전 인증세션의 상태에 따라 정상종료인 경우 $flag$를 0으로, 비정상 종료인 경우 1로 저장한다. 리더로부터 질의를 받은 태그는 $flag$의 값에 따라 응답의 내용과 해당 연산을 달리 하게 된다.

제안 프로토콜에서 태그는 리더의 질의에 대해 매번 다른 내용을 응답하므로 공격자는 태그의 위치를 트래킹 할 수 없다. 이전 세션이 정상적으로 종료된 경우 데이터베이스에서 ID검색의 효율성을 위해 태그는 $h(ID)$값을 응답한다.

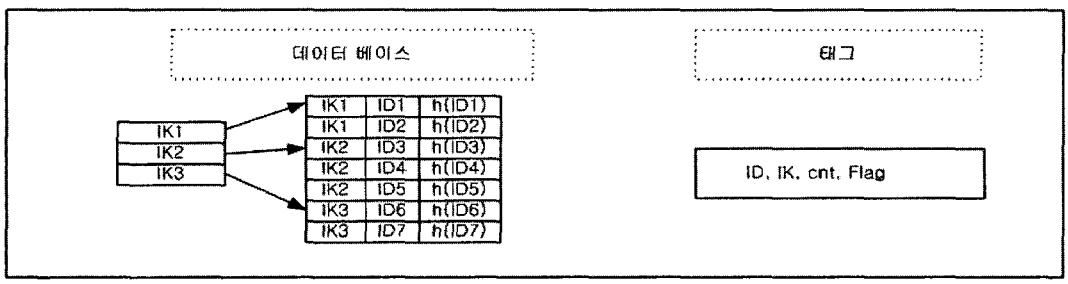


그림 4. 데이터베이스 구조와 태그 저장정보

3) 데이터베이스는 i값을 알 수 없으므로 [11]에서의 설명만으로는 데이터베이스가 ID를 검색하기는 불가능하다.

이 경우 데이터베이스는 *ID* 검색에 해쉬연산이 필요치 않으므로 매우 효율적이다. 정상적인 세션이 종료되면 태그의 *ID*는 변경되므로 *h(ID)*도 매번 바뀌게 되며 공격자는 *h(ID)*를 도청하더라도 값이 매번 바뀌므로 태그의 이동경로를 파악할 수 없게 된다. 공격자가 위치 트래킹 공격 등 여러 목적을 위해 태그에게 질의를 하는 경우 혹은 통신상의 문제로 인해 태그가 비정상적으로 세션을 종료했을 경우 태그는 *flag*를 1로 변경한다. 비정상적인 세션의 종료 뒤에는 태그의 ID가 변경되지 않으므로 *h(ID)*값은 매번 동일하게 되며 이동 경로가 파악된다. 그러므로 본 프로토콜에서는 *flag*가 1일 경우에 *ID*의 해쉬 값인 *h(ID)* 대신 리더의 질의에 따라 매번 변경되는 값인 *cnt*와 데이터베이스에서 *ID*를 검색하는데 사용되는 *IK*를 병합한 후 해쉬하여 응답을 하도록 설계하였다. 매번 변경되는 값인 *cnt*는 태그의 계산능력이 제한적임을 고려, 매번 랜덤하게 생성하지 않고, 최초 랜덤하게 생성된 *cnt* 값을 태그에 저장하고 질의시마다 1씩 증가시켜 사용하도록 하였다. *cnt* 값은 해쉬의 입력으로 사용되므로 1씩 증가함을 공격자가 알더라도 해쉬의 결과를 보고 *cnt* 값이 1 증가되었음을 알지 못한다. 주의할 점은 *cnt* 의 사이즈가 작을 경우, 태그는 연속적인 공격자의 질의에 대해 동일한 *cnt*에 대한 응답을 하게 되므로, 위치 트래킹이 가능하게 되며, 예측한 *cnt*에 대해 공격자는 스푸핑 공격을 할 수도 있게 된다. 본 프로토콜에서 사용가능한 태그의 메모리 사용량은 *ID*길이의 2배로 정하였으며, *ID*를 제외하면 *flag*의 길이 1bit를 제외한 나머지 메모리를 *IK*와 *cnt*가 나누어 사용하게 된다. 유비쿼터스 ID 센터[4]에서 제안한 ucode에 따르면 *ID*길이는 128bit이며 128bit씩 확장될 수 있다. *ID* 길이를 128bit로 했을 경우 본 프로토콜에서는 *IK*와 *cnt*가 127bit 메모리용량을 사용할 수 있으며, 예로 *cnt*를 64bit로 할 경우, 공격자는 태그에게 2^{64}번의 질의와 해쉬연산을 수행해야만 위치를 트래킹 할 수 있게 된다. 그러나 2^{64}번의 해쉬연산은 polynomial time안에 수행하기 불가능하므로 *cnt*를 64bit로 사용하면 위치트래킹 공격에 안전하다. 데이터베이스는 *ID*의 해쉬값 (*flag*=0인 경우) 또는 *IK* (*flag*=1인 경우)를 사용하여 태그의 *ID*를 알아낼 수 있으며 해당 *ID*로 계산해낸 값과 태그로부터 받은 값을 비교하여 일치할 경우 태그를 인증하며, 태그를 인증한 뒤에는 태그에게 태그와 데이터베이스만이 계산해 낼 수 있는 값을 보내 태그가 데이터베이스를 인증할 수 있도록 한다.

4) http://www.uidcenter.org/

1. 사전준비단계

상품제조자는 *IK*, *ID*, *h(ID)*를 저장할 수 있는 데이터베이스를 준비한다. 또한 태그의 *ID*를 데이터베이스에 효과적으로 검색할 수 있게 해 주는 *IK*와 리더의 질의 시 마다 매번 값이 증가하는 *cnt*, 초기값을 0으로 갖는 *flag*를 상품에 부착된 태그에 저장한다(그림 4 참조).

2. 인증단계

이 단계에 대한 설명은 그림 5, 6에서 태그와 리더, 데이터베이스가 주고받는 통신내용을 기준으로 하여 설명한다.

①. 태그를 인증하기 위해 리더는 의사난수생성기를 사용하여 난수 *S*를 생성하고 태그에게 질의를 한다. 이때 리더는 매번 다른 *S*를 태그에게 보낸다고 가정한다. 태그는 가지고 있는 정보와 *S*를 사용하여, $T=h(ID||cnt)$, $Q=h(ID||T||S)$를 계산하고 이전 세션의 상태에 따라 *flag*가 0인 경우 *h(ID)*를 계산하고, *flag*가 1인 경우 $h(T||IK)$를 계산한다. 리더가 임의의 *S*를 보내는 것은 태그가 응답할 Q를 공격자가 만들 수 없게 하여 스푸핑 공격과 재생공격을 막기 위해서이다. 또한 이전 세션의 상태에 따라 *h(ID)*와 $h(T||IK)$를 구분해서 보내는 이유는, 악의적인 리더나 공격자가 질의를 통해 태그의 정보를 얻으려고 하는 가능성을 없애기 위해서이다. 공격자로부터 질의가 온 경우 공격자는 태그에게 정당한 응답을 주지 못하므로 태그의 *ID*는 변경되지 않는데 이때 *h(ID)*를 응답하게 되면 위치 트래킹이 가능해진다. 반면에 매번 $h(T||IK)$를 보내게 되면 값이 매번 바뀌므로 위치트래킹 공격에는 안

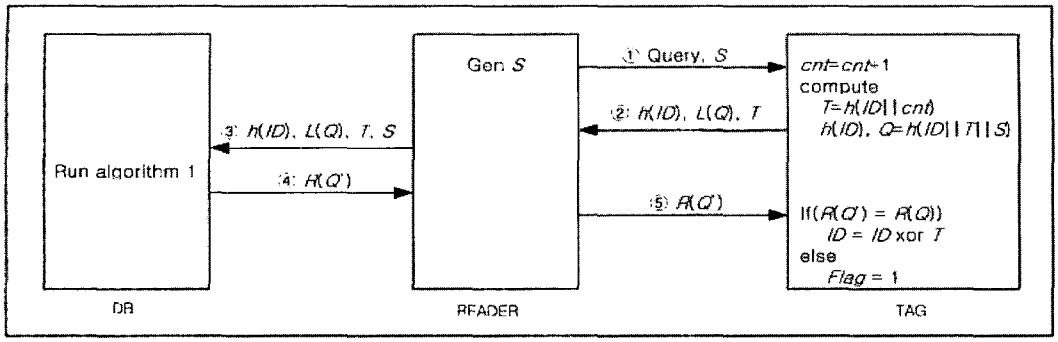


그림 5. flag가 0인 경우

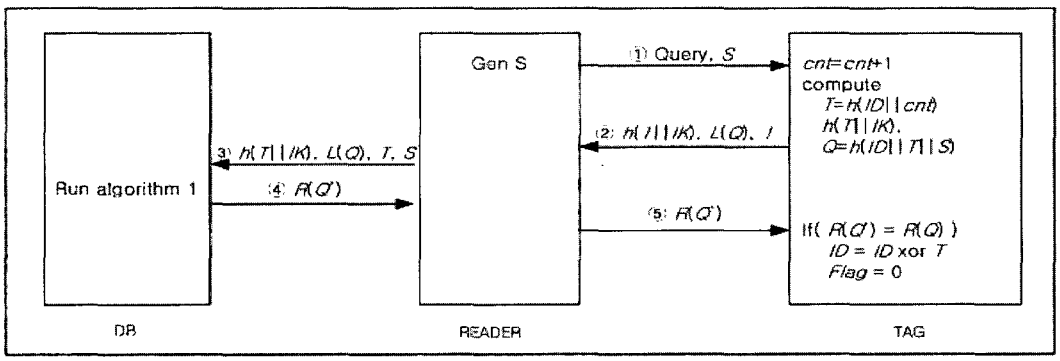


그림 6. flag 가 1인 경우

전하지만 서버에서 IK를 이용한 해쉬연산을 통해서 ID를 검색해야 하므로 $h(ID)$로 ID를 검색하여 응답하는 경우보다 서버의 부하가 많아진다. 그러므로 상태에 따라 응답을 달리함으로써 위치 트래킹 공격에도 안전하며, 서버의 효율도 증가시키도록 설계하였다. T 또한 매번 값이 변하게 되는데 그 이유는 태그가 리더로부터 질의를 받을 때 마다 cnt가 증가하기 때문이다. ID가 동일하더라도 cnt값이 변하므로 해쉬 값인 T는 공격자가 추측할 수 없는 값으로 매번 바뀌게 된다. Q는 T와 S를 해쉬한 값이며 리더의 질의마다 바뀌므로 이 또한 공격자가 추측할 수 없다.

②. 태그는 계산된 $h(ID)$(또는 $h(T||IK)$), $L(Q)$, T를 리더에게 전송한다. 위에서 설명한 바와 같이, 태그가 전송하는 내용을 공격자가 도청하여 트래픽을 분석하더라도 전송되는 값이 모두 해쉬연산의 결과이므로 공격자는 이전의 값을 추측할 수 없고 따라서 다른 공격에 사용가능한 어떠한 정보도 얻을 수 없게 되며 위치트래킹도 불가능하게 된다.

③, ④. 리더는 태그에게 받은 정보와 S를 데이터 베이스에 전달한다. ③을 전달받은 데이터베이스는 그림 7의 알고리즘을 수행한다. 이때 ③은 태그의 이전 세션의 상태에 따라 값이 달라지며, 데이터 베이스는 태그의 이전 세션상태가 어떤지를 알지 못하므로 우선 이전 세션의 상태가 정상임을 가정하고(즉 $h(ID)$ 가 전송되었음을 가정하고), ③의 첫 번째 값($h(A)$로 표기)으로 데이터베이스의 $h(ID)$와 일치하는 ID(ID'로 표기)가 있는지 검색한다(그림 7의 1라인). 일치하는 $h(ID)$가 있을 경우, 해당 ID'로 Q'를 계산하여 $L(Q')$와 ③의 $L(Q)$를 비교한다(그림 7의 2라인~5라인). 일치할 경우, ID'에 T를 xor연산하여 ID'를 변경하고 $R(Q')$를 리더에게 전송한 후, $h(ID')$를 데이터베이스에 저장한다. 만약 $h(A)$가 $h(ID)$가 아닌 $h(T||IK)$의 형태였을 경우, 2라인에서의 다른 분기인 10라인 이후를 수행하게 된다. 데이터베이스는 태그로부터 받은 T에 IK 각각을 차례대로 붙여 해쉬를 수행하면서 결과값을 $h(A)$와 비교한다. 일치하는 IK(IK'로 표기)가

```
Input: h(A), L(Q), T, S
1. Search h(A) in DB
2. if h(A) exists
3.    Get ID' for the h(A) from ID column in DB
4.    compute Q' = h(ID'||T||S)
5.    if L(Q) = L(Q')
6.       ID' = ID' xor T
7.       compute h(ID') and store it
8.       send R(Q')
9.    else
10.      halt
11. else
12.    compute h(T||IK) and compare with h(A)
       until finding IK' s.t h(T||IK')=h(A)
13.    if such IK' exists
14.       compute Q=h(ID||T||S), L(Q')
          until finding ID' related to the IK' s.t L(Q')=L(Q)
15.       if such ID' exists
16.          ID' = ID' xor T
17.             compute h(ID') and store it
18.          send R(Q')
19.       else
20.          halt
21.    else
22.       halt
```

그림 7. Algorithm 1

발견되면 IK'에 해당하는 ID들에 대해 $L(Q')$를 계산하여 $L(Q)$와 일치하는 ID'가 있는지 비교한다. 일치하는 ID'가 있는 경우 T와 xor를 수행하여 ID'를 변경하고 $R(Q')$를 리더에게 전송한다.

⑤. 리더는 데이터베이스로부터 받은 $R(Q')$를 태그에게 전달한다. 태그는 받은 $R(Q')$와 자신이 계산한 $R(Q)$를 비교하여 일치할 경우, 리더를 인증하고 자신의 ID를 T와 xor하여 ID값을 변경하고, $flag$를 0으로 저장한다. 만약 리더로부터 응답이 오지 않거나, $R(Q')$가 R(Q)와 일치 하지 않을 경우 태그는 $flag$값을 1로 갖는다.

본 프로토콜에서는 $flag$가 0인 경우 $h(ID)$로만 ID를 검색하므로 데이터베이스에서는 해쉬연산을 수행할 필요가 없다. $flag$가 1인 경우, ID를 검색하기 위한 해쉬 계산량을 최소화하기 위해 $\sqrt{n}$개의 ID가 같은 값의 IK를 가지도록 하여 총 $\sqrt{n}$개의 서로 다른 IK를 가지도록 한다. 그럼 데이터베이스에 존재하는 IK의 수는 $\sqrt{n}$개이므로 평균 $\sqrt{n}/2$번의 $h(T||IK)$연산을 수행하면 태그가 가지는 IK와 동일한 값을 갖는 IK'를 찾을 수 있다. 위와 같이 IK'를 찾은 후에 IK'에 해당하는 ID들에 대해서 $Q' = h(ID||T||S)$를 계산하고 $L(Q')$가 태그로부터 받은 $L(Q)$와 동일한 값을 가지면 태그의 ID를 찾은 것이 된다. 이 때, IK'에 해당하는 ID는 $\sqrt{n}$개 존재하므로 평균 $\sqrt{n}/2$번 $h(ID||T||S)$연산을 하면 ID'을 찾을 수 있다. 따라서 평균적으로 $\sqrt{n}$번의 해쉬연산을 통해서 태그의 ID를 얻을 수 있다.

V. 위협에 대한 안전성

여기서는 II장에서 소개한 위협요소에 대해 우리가 제안한 프로토콜이 안전함을 설명한다.

표 1. 프로토콜 비교

구분	해쉬기반 ID변형프로토콜[9]	개선된 해쉬기반 ID변형 프로토콜[10]	해쉬체인 기반 프로토콜[11]	제안 프로토콜
위치트래킹 공격에 안전	X	X	O	O
스푸핑 공격에 안전	X	X	O	O
메시지 복구 가능 여부	O	O	X	O
태그 메모리 사용량 (L은 ID의 길이)	3L	1L	1L	2L
태그 계산량	해쉬 3회	해쉬 2회	해쉬 2회 (두 해쉬는 다름)	해쉬 3회
데이터베이스의 계산량 (평균 해쉬 연산횟수)	없음	없음	$i * n$ (i는 s_i의 갱신횟수)	평상시 없음 이전세션이 비정상인 경우 $\sqrt{n}$

- 도청/통신내용분석 : 공격자는 도청한 내용 혹은 태그에게 질의를 하여 얻은 응답을 분석함으로써 위치트래킹 공격이나 스푸핑 공격에 활용할 수 있다. 그러나 본 프로토콜에서는 누가 질의를 했던 간에 태그의 응답은 항상 다르며, 그 내용은 모두 해쉬의 결과값이므로 해쉬의 일방향성으로 인해 공격자는 어떠한 정보(ID, cnt, ...)도 얻을 수 없다.
- 위치트래킹 : 본 프로토콜에서는 태그가 $flag$ =0인 상태에서는 매번 ID가 갱신되므로 $h(ID)$, $L(Q)$, T 값이 매번 달라지며 따라서 공격자는 위치를 트래킹할 수 없다. 공격자가 계속해서 태그에게 질의를 하는 경우 태그는 $flag$=1인 상태를 유지하는데 이때 cnt가 매번 증가하므로 T 또한 매번 변경되며 T 값이 변경됨에 따라 $h(T||IK)$와 Q 또한 값이 변경되어 위치트래킹은 불가능해 진다. $flag$ 0과 1이 반복되는 경우 역시 위의 내용이 번갈아 응답되므로 태그는 위치 트래킹 공격에 안전하다.
- 스푸핑 : 본 프로토콜에서 공격자는 리더를 속이기 위해 태그인 척 또는 태그를 속이기 위해 리더인 척 할 수 없다. 공격자가 태그인 척 하기 위해서는 그림 5, 6의 ②, ⑤를 가로챈 후, 그 내용을 수정하여 정당한 리더에게 보내야 하는데, 정당한 리더는 매번 랜덤한 S를 질의하므로 공격자는 ID, cnt를 모르는 상태에서는 S에 대한 Q를 만들어 낼 수 없으므로 데이터베이스는 속지 않는다. 공격자가 리더인척 하는 경우, 먼저 공격자는 S를 생성하거나 이전에 도청한 S를 태그에게 질의한다(이때 공격자는 도청한 S에 해당하는 그림 5, 6의 ②, ⑤를 모두 알고 있다고 가정한다). 태그는 S에 대해 $L(Q)$를 생성하게 되는데 공격자는 S에 해당하는 이전의 $R(Q')$를 알아도 해쉬 함수의 일방향성 때문에 현재의 $R(Q')$를 만들어 낼 수 없으므로 태그를 속일 수 없다. 또한 공격자가 추측하여 생성한 $R(Q')$가 정당한지를 확인하기 위해 태그를 이용하려고 해도 태그에게는 $R(Q')$를 한번 이상 보낼 수 없게 되므로[5] 이 방법으로 태그를 속이는 것 또한 불가능하다.
- 메시지 유실 : 본 프로토콜에서는 [9]에서 제안한 AE를 이용한 메시지 유실에 대비한 스킴을 적용할 수 있다. 그림 5, 6의 ⑤가 유실된다 하더라도 데이터베이스는 이전 ID와 해당 정보를 저장하고 있으므로, 메시지 유실에 대해 안전하다. ②가 유실되는 경우에도 태그는 ⑤를 받지 못할 경우 이전 세션을 비정상으로 판단하고 $flag$를 1로 설정하므로 둘 사이에 동기화는 유지된다.

Ⅵ. 결론 및 향후과제

지금까지 사용자의 프라이버시를 보호하기 위한 기존 RFID 인증프로토콜에 대해 알아보았으며, 그들의 단점을 보완하기 위해 우리가 제안한 프로토콜을 설명하였다. 끝으로 아래의 표 1은 제안한 프로토콜과 기존프로토콜을 공격에 대한 안전성, 효율성 측면에서 비교한 것이다.

본 논문에서 제안한 프로토콜은 기존에 제안된 프로토콜들이 가지는 스푸핑공격이 가능한 취약점을 보완하였으며, 위치프라이버시를 보장하도록 설계되었다. 또한 위치프라이버시를 보장하는 기존의 프로토콜[11]이 서버 계산량의 과다로 인해 실제 시스템에서 적용이 불가능함에 반해, 본 프로토콜은 서버의 부하를 현저히 줄임으로써, 안전하고 효율적인 RFID인증 프로토콜이 요구되는 시스템에 쉽게 적용이 가능하다.

향후에는, 태그의 ID가 노출되었을 경우에도 이전의 위치기록을 공격자가 알 수 없도록 하기 위해 전방위 안전성을 보장하는 기능을 프로토콜에 추가시키는 연구를 진행해 나갈 생각이다.

참 고 문 헌

[1] Sanjay Sarma, Stephen Weis, and Daniel Engels. "Radio-frequency identification systems," CHES'02, LNCS 2523, pp. 454 - 469, August 2002.

5) 정당하지 않은 R(Q')를 받은 태그는, 정당한 R(Q')를 받을 때 까지 기다리지 않고, 새로운 세션이 시작되어 리더로부터 질의가 오지 않으면 응답을 하지 않는 상태가 된다.

[2] Sanjay Sarma, Stephen Weis, and Daniel Engels. "RFID systems, security and privacy implications," Technical Report MIT-AUTOID-WH-014, AutoID Center, 2002. Available from http://www.autoidcenter.org

[3] Sanjay E. Sarma. "Towards the fivecent tag," MIT Auto ID Center, Technical Report MIT-AUTOID-WH-006,2001. Available from http://www.autoidcenter.org

[4] Federal Information Processing Standards (FIPS), "Data Encryption Standard (DES)," NIST, Technical Report 46-2, January 1988.

[5] Federal Information Processing Standards (FIPS). "Advanced Encryption Standard (AES)," NIST, Technical Report 197, November 2001.

[6] Federal Information Processing Standards (FIPS). "Secure Hash Standard (SHA-1)," NIST, Technical Report 180-1, April 1995.

[7] 윤호선, 류종호, 김락현, 윤이중, 염흥렬, "고속 동작 가능한 해쉬알고리즘 (HAVES)의 제안," 한국정보보호학회논문지 8권 4호, pp. 3-15, 1998.

[8] Stephen Weis. "Security and Privacy in Radio-Frequency Identification Devices," Masters Thesis MIT, May 2003.

[9] Dirk Henrici, and Paul Müller. "Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers," PerSec'04, pp. 149-153, March 2004.

[10] 황영주, 이수미, 이동훈, 임종인. "유비쿼터스 환경의 Low-Cost RFID 인증프로토콜," CISC'S04, pp. 120-122, June 2004.

[11] Miyako Ohkubo, Koutarou Suzuki, and Shingo Kinoshita. "Cryptographic approach to 'privacyfriendly' tags," RFID Privacy Workshop MIT, November 2003.

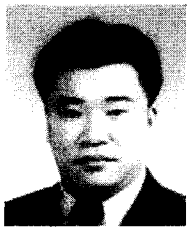
[12] Stephen Weis, Sanjay Sarma, Ronald Rivest, and Daniel Engels. "Security and privacy aspects of low-cost radio frequency identification systems," SPC'03, pp 454-469, March 2003.

[13] Ari Juels, Ronald Rivest, and Michael Szydlo. "The blocker tag : Selective blocking of RFID tags for consumer privacy," 8th ACM Conference on Computer and Communications Security, pp. 103 - 111, 2003.

[14] Alastair Beresford, and Frank Stajano. "Location Privacy in Pervasive Computing," IEEE Pervasive Computing 2003, pp. 46-55, 2003.

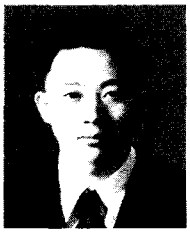
[15] 이근우, 오동규, 곽진, 김승주, 원동호. "Low-Cost RFID 시스템을 위한 Improved Hash Chain 프로토콜," CISC'S04, pp. 155-160, June 2004.

〈著者紹介〉

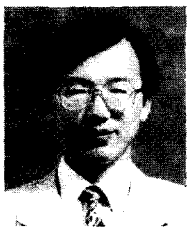


유 성 호 (Sung Ho Yoo) 학생회원
2000년 2월 : 숭실대학교 컴퓨터학부 학사
2000년 1월~현재 : 포스데이타 재직
2003년 3월~현재 : 포항공과대학교 정보통신대학원 석사과정
〈관심분야〉 정보보호, Ubiquitous Computing

김 기 현 (KiHyun Kim) 학생회원
2003년 2월 : 포항공과대학교 전자전기공학과 학사
2003년 3월~현재 : 포항공과대학교 전자전기공학과 석사과정
〈관심분야〉 정보보호, 암호이론



황 용 호 (Yong Ho Hwang) 학생회원
2001년 2월 : 홍익대학교 전자전기제어공학과 졸업(공학사)
2001년 3월~현재 : 포항공과대학교 전자전기공학과 박사과정
〈관심분야〉 정보보호, 암호이론, 암호 프로토콜



이 필 중 (Pil Joong Lee) 정회원
1974년 2월 : 서울대학교 전자공학과 학사
1977년 2월 : 서울대학교 전자공학과 석사
1982년 6월 : U.C.L.A System Science, Engineer
1985년 6월 : U.C.L.A Electrical Engineering, Ph.D.
1980년 6월~1985년 8월 : Jet Propulsion Laboratory, Senior Engineer
1985년 8월~1990년 2월 : Bell Communications Research, M.T.S.
1990년 2월~현재 : 포항공과대학교 전자전기공학과 교수
1996년 2월~1997년 2월 : NEC Research Institute 방문 연구원
2000년 9월~2003년 8월 : 포항공과대학교 정보통신 연구소장 (정보통신 대학원장 겸임)
2004년 1월~2004년 12월 : 한국정보보호학회 회장
2004년 1월~2004년 12월 : KT 연구소 방문 연구원
〈관심분야〉 정보보호전반