情報保護學會論文誌
第15卷 第5號, 2005. 10

RFID 시스템에서 강제 충돌을 이용한 태그 정보 보호 기법

박 주 성,† 강 전 일, 양 대 헌‡
인하대학교

A Method to Protect the Information of Tag Using Forced Collision Mechanism in RFID System

Ju-Sung Park,† Jeonil Kang, Dae-Hun Nyang‡
INHA University

요 약

이 논문에서는 RFID 시스템에서 태그가 갖고 있는 일련의 정보들을 허가받지 않은 리더에 의한 공격으로부터 보호하기 위한 방법을 제안한다. 이 방법에서 허가받지 않은 리더에게는 랜덤하게 분포된 거짓 비트열을 삽입한 정보를 제공하여 태그의 원래 정보를 보호하고 반대로 허가받은 리더에게는 ownership tag라는 별도의 태그로 하여금 리더의 질의에 대한 응답으로 거짓 비트가 전송될 때에 강제로 충돌을 발생시키게 하여 리더가 거짓 비트의 존재와 위치를 파악하여 태그 원래 정보를 알아낼 수 있도록 한다.

ABSTRACT

We propose a method to protect the tag's information from illegal reader in RFID system. In this method, a special tag called ownership tag makes intentionally collisions during singulation. By the forced collision of an ownership tag a reader can obtain the information of ordinary tags. Whereas a reader can not find any information on tag's ID without the ownership tag.

Keywords : *RFID, Privacy, Information leakage, Tag, Collision*

Ⅰ. 서 론

RFID(Radio Frequency Identification)는 크게 태그와 리더 그리고 데이터베이스 시스템으로 구성된 시스템으로서 기존의 바코드 시스템을 대체할 수 있는 시스템이다. 이 시스템의 최대 강점은 각 각의 태그가 포함하고 있는 적은 용량의 메모리에 제품마다 각자의 정보를 보유할 수 있다는 것이다. 기존의 바코드 체계는 각 품종마다 고유의 번호를 제공하여 제품인식에만 사용한 반면 이 태그는 품종이 아닌 각 제품 한 개당 서로 다른 식별자를 제공할 수 있고 그로 인하여 각 제품마다 고유의 정보를 포함 할 수 있다는 것이다. 그러나 이런 장점에도 불구하고 가격 경쟁력이 바코드에 비하여 떨어지기 때문에 가격이 낮아야한다는 제약 문제가 남아있고, 낮은 생산원가를 맞추기 위하여 자원의 한계가 발생하게 된다.[8] 따라서 RFID 시스템은 자원의 한계에 따른 몇 가지 문제를 안고 있으며, 그 중 가장 많은 논란의 소지를 갖고 있는 것이 프라이버

접수일 : 2005년 3월 29일 ; 채택일 : 2005년 10월 12일
† 주저자, security77@seclab.inha.ac.kr
‡ 교신저자, nyang@inha.ac.kr

시 문제이다. 프라이버시 문제는 대부분의 태그가 자체 전원이 없는 수동형 모델을 채택하고 있고, 자원적인 제약으로 인하여 태그 안에서 어떤 특정 연산이나 동작을 통하여 태그의 정보를 보호하기 힘든 데서 기인한다. 따라서 태그의 정보를 보호하기 위해서는 특별한 정보 보호 조치가 필요하다.

RFID 시스템 내에서 태그 정보 보호 그리고 개개인의 프라이버시 문제에 대하여 여러 가지 해결책을 제시하고자 많은 연구가 활발하게 이루어지고 있다. 그 중 블로커 태그는 충돌을 강제적으로 발생시키고 태그의 정보를 보호하는 목적을 갖고 있다는 것에서 ownership tag와 유사하다.

블로커 태그는 리더의 어떤 질의에도 'yes'라는 응답을 전송하여, 리더와 태그사이의 통신을 방해함으로써 태그의 정보를 보호한다. 블로커 태그는 두 개의 안테나로 구성되어, 한쪽 안테나에서는 '1'을 다른 쪽 안테나에서는 '0'을 전송하여 어떤 질의에 대해서도 'yes'라는 대답을 가능하게 한다.

많은 해결책 중에서 블로커 태그는 이진탐색트리 방식에 기초하여 모든 노드에서 강제적인 충돌을 일으키는 알고리즘으로 유일하게 암호학적 알고리즘을 구현하지 않고 태그의 정보를 보호하며 구현하는데 많은 비용이 들지 않고, 쉽게 태그의 정보를 보호하는 장점이 있다. 그러나 블로커 태그의 알고리즘을 공격자가 역이용한다면 즉, 태그의 정보를 보호하기 위한 목적이 아닌 정상적인 통신을 방해하기 위해 사용한다면, 리더에 대한 서비스거부 공격도 가능하다. 또한, 블로커 태그는 보호하고자 하는 태그의 주위에 같이 존재할 때만 태그의 정보를 보호한다. 만약 소비자의 사소한 실수 (분실, 별도의 소지 등)가 발생한다면 태그의 정보는 보호받지 못한다. 즉, 블로커 태그는 집을 지키기 위해 문을 잠그기보다는 경비원을 배치하는 것과 같다. 경비원만 없다면 집은 문이 열린 상태로 방치되는 상태가 된다.[3]

이 논문에서는 태그의 정보와 소비자의 프라이버시를 보호하기 위한 새로운 방법을 제안한다. 새로운 방법은 블로커 태그와 유사한 방법으로 충돌을 이용하지만 추구하는 목적이 정 반대의 개념이므로, 이에 따라서 구현 시에 블로커 태그 방법이 갖고 있는 몇 가지의 문제점을 해결할 수 있다. 중요한 것은 이 논문에서 제시하는 새로운 방법이 블로커 태그 방법의 개선안이 아니라는 것이다. 새로운 방법은 충돌을 이용하는 것이 블로커태그와 유사할 뿐 같은 목적을 갖고 있지도 또한, 구현 원리도 다른 것이다. 즉, 전혀 새로운 개념의 태그 정보 누출 방지 알고리즘이다.

Ⅱ장에서는 일반 태그에 대한 간략한 설명과 ownership tag에 대한 기술적인 설명을 하고자한다. Ⅲ장에서는 제시된 방법에 대한 분석 및 평가를 하며, Ⅳ장에서는 결론을 도출한다.

Ⅱ. Ownership Tag

2.1 Overview

Ownership tag는 블로커태그의 단점을 보완하는 방법이나 블로커태그 방식의 개선안이 아니다. 다만 ownership tag가 강제 충돌을 이용하는 측면에서 블로커태그와 유사하며, 구현 시 블로커태그가 갖고 있는 문제를 해소할 수 있다. 이 논문에서 제시하는 새로운 방법의 목적은 태그의 정보가 악의적인 리더에 읽혀지지 않게 보호하고자 함이며, 태그의 정보가 바르게 읽혀지기 위한 일련의 키를 리더에게 제공할 때 단순히 키 정보를 제공하는 것이 아니라 강제적인 충돌을 이용해서 리더에게 알려주는 것이다. 이를 구현할 때에 블로커태그와 마찬가지로 다른 암호학적인 알고리즘을 사용할 필요가 없다. 다만, 일반적인 RFID 시스템에 추가되는 요구사항이 존재하며 다음 장에서 자세하게 기술한다.

일반적인 RFID 시스템은 리더와 태그사이에 질의와 응답을 통해 태그의 비트열을 리더가 알아내고 그 비트열로 백엔드 데이터베이스 시스템에서 해당하는 정보를 찾아내는 방식이다. 이러한 통신을 하는데 있어, 가장 많이 사용하는 방식이 트리기반 방식과 알로하 방식이 있다. 트리기반 방식은 트리를 따라 질의와 전송이 이루어지는 방식이며 알로하방식은 전체 비트열을 한 번에 전송하는 방식이다. 일반적인 태그들은 한 개의 안테나와 하나의 ROM으로 이루어져 있으며, 질의에 따라 안테나를 이용해 해당 비트를 답변한다.

2.2 가정

2.2.1 일반 태그 I

RFID시스템에서 태그는 대략적으로 몇 가지로 구분할 수 있다. 사용된 메모리에 따른 구분과 형태에 따른 구분이 대부분의 구분법이다. 우선 메모리의 구분에 따르면, RAM을 쓰는 가장 흔한 형태부

터 EEPROM, FRAM, OTP 등으로 구분되어 진다. 또한, 형식에 따르면 능동형과 수동형으로 구분되어 진다. 능동형은 자체 전원 공급 장치가 존재하며 데이터 전송거리가 긴 반면에 수동형은 그렇지 못하다. 하지만, 가격이 저렴하여 대부분의 RFID 시스템에서는 수동형을 선호한다.[2,6]

이 논문에서 일반적 태그는 다음과 같이 구성된다고 가정한다.

1. 태그의 형식은 수동형을 사용 한다.
2. 태그는 일반적인 RFID의 태그와 동일하다. 다만, 태그 내에 한 개의 OTP메모리가 추가되며, 이 OTP메모리에는 삽입될 거짓 비트가 저장된다.

2.2.2 리더 및 통신 방법

리더는 다음과 같이 가정한다.

1. 리더와 태그는 이진 탐색 트리 알고리즘을 사용하여 통신한다고 가정한다.

   RFID 시스템에는 대표적으로는 이진탐색기법, slotted Aloha가 있으며 그 중 이진 탐색 기법은 충돌에 대한 정확한 위치가 리더 내에 파악되어 있어야 하며, 리더는 태그에게 정해진 양의 비트의 전송을 요구하고 충돌이 발생하지 않았을 경우 다음 정해진 양의 비트의 전송을 요구한다. 충돌이 발생했을 경우 충돌한 위치를 기록하여 서브트리를 검색하게 되는 방식이다. 이 방법의 장점은 확정적인 데이터 전송에서 찾아볼 수 있다. 리더는 태그가 보내는 모든 데이터를 사용하게 되므로 충돌로 인하여 불필요하게 사라지는 데이터는 없는 것과 같다. 또한 많은 수의 태그가 존재하거나 적은 수의 태그가 존재하여도 이에 따른 전체 성능을 선형적인 수식에서 예측할 수 있다. 반면 단점은 리더가 태그에게 태그가 리더에게 보내는 데이터만큼이나 많은 명령 질의를 전송해야 한다는 것이다. 또한 태그는 리더에게 데이터를 전송하기 위하여 전력을 충전하고 방전하는 과정이 지나치게 빈번하다는 것도 문제점을 갖고 있다.
2. 이 논문에서는 리더와 태그사이의 통신만을 다룬다.

   RFID 시스템을 구성하는 3가지 요소 중 백엔드 데이터베이스에 대한 것은 다루지 않는다.
3. 이 논문에서는 리더와 태그사이에 채널을 이용한 능동적인 공격은 고려하지 않는다.

   어떤 종류의 응용모델에서는 태그가 전송하는 데이터를 공격자가 도청하지 못하며, 충돌 또한 감지하지 못한다. 이에 따른 메시지를 끼워 넣거나 가로채는 공격은 이러한 응용모델에서는 가능하지 않다.[5] 만약, 공격자가 태그와 리더사이의 통신을 사이에 간섭을 할 수 있다 하더라도 이것은 이 논문에서 해결하고자 하는 정보누출과는 크게 관련성이 없다.

2.2.3 Ownership tag

Ownership tag는 다음과 같이 가정한다.

1. ownership tag는 OTP(One Time Programmable user memory)를 사용한다고 가정한다.

   Ownership tag에는 리더가 제공하는 거짓 비트를 저장해야 할 메모리가 필요함으로 쓰기가 가능해야 한다. 그러나 여러 번의 쓰기가 필요하지 않으므로 여러 번 쓰기가 가능한 고가의 FRAM등을 사용할 필요가 없다. 또한 RFID 시스템에서 태그 가격의 목표는 대략 0.1$ 정도를 형성하는 것이기 때문에 가격이 비싼 EEPROM이나 FRAM은 사용을 하지 않는다. 대신 한 번의 쓰기가 지원되며 가격이 비교적 저렴한 OTP를 사용한다.
2. ownership tag는 강제적으로 충돌을 발생시키기 위해 두 개의 안테나를 갖는다고 가정한다.

2.3 시스템 요구사항

일반 태그에는 원래 일반 태그들이 갖고 있는 안테나 한 개와 ROM한개 이외에 거짓 비트를 삽입하기 위해서 일반 태그에 OTP메모리 하나를 추가한다. 모든 일반 태그들은 (같이 구매된 상품의 태그들) 처음 물건을 구매할 때 안전한 리더로부터 부여되는 동일한 거짓 비트열을 이 OTP메모리에 저장한다. 또한, 모든 태그가 동일한 거짓 비트들을 보유함에도 불구하고 리더의 질의에 응답으로 보내는 비트는 각자 다르게 하기 위해 일반 태그에 3개의 포인터를 추가한다. 만약 모든 태그가 동일한 거짓 비트들을 보유하고 그 거짓 비트를 질의에 대한 응답으로 사용한다면, 이 논문에서 제시하는 방식의 목적인 이동 중에 혹은 사용 중에 소비자가 읽혀지

표 1. Ownership Tag Scheme 구현을 위한 시스템 요구사항

Ownership Tag Scheme		
부분	구성요소	역할 및 비고
리더	리더	이진탐색트리 기법에 의하여 태그의 정보를 읽어 들인다.
태그	안테나 1개	질의의 수신과 응답의 발신을 위해 사용된다.
	ROM 1개	태그의 정보를 저장한다.
	OTP메모리 1개	거짓 비트들의 위치 값을 저장한다.
	포인터 3개	data_pointer는 태그 정보를 담고 있는 메모리 셀을 가리키고, bogus_pointer는 거짓 비트열을 담고 있는 메모리 셀을 가리킨다. input_bogus_pointer는 태그 데이터부분의 맨 마지막을 초기 값으로 갖고, 한 셀씩 감소하면서 삽입되는 거짓 비트를 지시한다. (data_pointer, bogus_pointer, input_bogus_pointer)
ownership tag	안테나 2개	충돌을 발생시키기 위해서 2개의 안테나가 필요하다.
	OTP 메모리 1개	태그의 거짓 비트열과 같은 비트열을 저장한다.
	포인터 1개	질의를 청취하며, 태그의 거짓 비트에 맞춰 진행한다.

고 싶지 않은 즉, 소비자에게 허가받지 않은 리더가 단순히 질의를 전송하고 그 응답을 저장하는 것만으로도 태그의 정보가 노출될 수 있기 때문이다. 허가받지 않은 리더는 몇 개의 태그들이 전송한 응답을 보고 n번째 비트들이 모두 1 또는 0 임을 알 수 있고 이것으로 인하여 허가받지 않은 리더는 태그의 원래 정보를 알아낼 수 있다. 이를 방지하기 위해 3개의 포인터가 동작원리에 의해 동작하면서 같은 거짓 비트열임에도 불구하고 거짓비트의 위치에서 각자 다른 값을 질의의 응답으로 전송하게 된다. 자세한 동작원리는 다음 장에서 설명한다.

Ownership tag라는 특수한 목적을 갖는 태그는 강제적인 충돌을 발생시키기 위한 두 개의 안테나를 갖고, 질의를 청취하면서 진행하는 하나의 포인터를 갖고 마지막으로 거짓 비트가 삽입될 OTP 메모리를 갖는다.

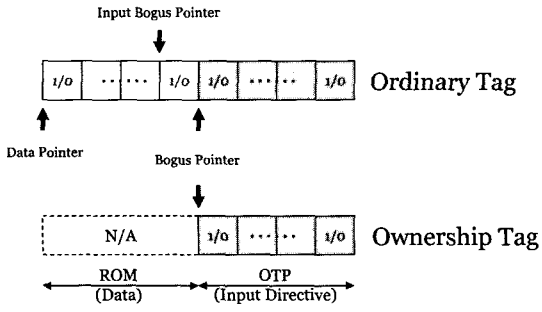


그림 1. 일반 태그와 ownership tag의 구성도

2.3.1 초기화

- input_bogus_pointer는 데이터를 저장한 메모리의 끝 위치 값으로 초기화 된다.
- bogus_pointer는 거짓 비트를 저장한 메모리의 첫 위치 값으로 초기화된다.
- data_pointer는 데이터를 저장한 메모리의 첫 위치 값으로 초기화 된다.
- ownership tag의 포인터를 메모리 첫 위치 값으로 초기화 한다.
- 리더는 모든 상품의 태그에 대한 정보를 알아내는 질의 동작을 모두 종료하고, 이에 해당하는 정보를 백 엔드 데이터베이스에서 찾아 상품명과 가격 등 상품에 대한 정보를 알아낸다. 이 처리과정이 끝나고 리더는 일반 태그의 빈 OTP메모리에 완전 랜덤하게 생성한 거짓 비트들을 삽입한다. 그리고 같은 거짓 비트열을 ownership tag에도 삽입한다.

2.4 작동방법

2.4.1 일반 태그

일반 태그는 리더의 질의가 홀수 번째인가 짝수 번째인가를 판단한다. 이것은 태그가 따로 계산할 필요가 없이 첫 번째에는 데이터부분이 응답하고 다음번에는 bogus bit가 삽입된 OTP부분으로 한 번씩 교대로 작동하는 것으로 쉽게 가능하다. 만약 홀수 번째의 질의라면 태그는 데이터가 저장된 ROM

에서 data pointer가 가리키고 있는 셀의 해당 비트를 응답으로 전송한 뒤 포인터를 한 셀 증가시킨다. 짝수 번째라면, 거짓 비트가 삽입 된 OTP에서 bogus_pointer가 가리키고 있는 해당 비트 값이 무엇인지 체크한다.

이 거짓 비트 값이 0이라면 태그는 다시 데이터가 저장된 ROM으로 가서 data pointer가 가리키고 있는 비트를 질의에 대한 응답으로 전송하고 포인터를 한 셀 증가 시킨다. 1일 경우, 태그는 input_bogus_pointer가 가리키고 있는 셀의 비트 값을 질의에 대한 응답으로 전송한 뒤, 포인터를 한 셀 감소시킨다. 이렇게 하는 이유는 여러 개의 태그들이 하나의 같은 거짓 비트 값을 갖고도 각 태그마다 실제 전송되는 거짓 비트 값이 다르게 하기 위함이다. 만약, 여러 개의 태그들이 각자 같은 거짓 비트 값만을 갖고 그것을 실제 전송하는 거짓 비트로 사용한다면, ownership tag가 존재하지 않을 경우에도, 어떤 리더라도 몇 개의 태그들의 데이터를 분석한다면 쉽게 거짓 비트의 위치를 알아낼 수 있게 되기 때문이다. n개의 태그가 같은 위치에 같은 비트 값을 갖게 될 확률은 다음과 같다.

비트 값이 1일 확률 + 비트 값이 0일 확률은

$$= \left(\frac{1}{2}\right)^n + \left(\frac{1}{2}\right)^n = \frac{1}{2^{n-1}} \text{ 이다.}$$

그러므로 태그의 개수가 커질수록 허가받지 않은 리더가 일반적인 질의를 통해서 ownership tag가 존재하지 않을 경우에도 태그에 포함된 거짓 비트를 쉽게 알아낼 수가 있게 된다. 따라서 이 논문에서는 input_bogus_pointer를 이용하여 거짓 비트가 삽입될 위치에 태그가 각자 갖고 있는 데이터를 역순으로 이용하여 같은 거짓 비트열을 갖고 있음에도 불구하고 다른 거짓 비트로 응답을 하게 된다. 즉, 태그가 리더로부터 부여 받는 것은 거짓 비트의 삽입 위치만을 나타내는 것이며, 태그가 각자 갖고 있는 데이터들이 실제 삽입되는 거짓 비트가 되는 것이다. 위와 같이 작동하면, 태그들은 각자 다른 거짓 비트를 같게 되므로, 허가 받지 않은 리더가 태그가 ownership tag가 존재하지 않을 때, 일반 태그에게 질의를 하는 것만으로 태그의 원래 데이터를 알아낼 수가 없게 된다.

그림 2는 일반 태그가 ownership tag scheme에서 작동하는 방식을 의사코드로 표현한 것이다.

```
loop ∞
 receive_Query_n^th
  if n is odd
    go to data part
     if data_pointer^th bit is EOF
       break loop
     else
       send response data_pointer^th bit
       increase data_pointer
  else
      go to bogus part
       if bogus_pointer^th bit is 0
         increase bogus_point
         go to data part
         send response data_pointer^th bit
         increase data_pointer
       else
          increase bogus_pointer
          go to data part
          send response input_bogus_pointer^th bit
          decrease input_bogus_pointer
```

그림 2. ownership tag scheme에서 일반태그의 의사코드

위 방법은 함께 구입된 모든 일반 태그 모두가 동일한 거짓 비트열을 갖고, 이것은 출현 위치를 같게 만들어 준다. 그러므로 ownership tag는 어떤 태그와도 함께 동작할 수 있다. 또한, 같은 위치이면서 일반 태그 각자가 보유한 데이터에 따라 삽입되는 거짓 비트는 달라진다. 따라서 'hole problem'을 완벽하게 해결할 수 있다.

2.4.2 Ownership tag

Ownership tag는 리더에서 전송하는 질의를 청취하면서, 일반 태그와 함께 순차적으로 진행한다. ownership tag는 리더가 갖고 있는 데이터는 모르지만, 삽입된 거짓 비트를 알기 때문에 어떤 질의에 거짓 비트가 혹은 태그의 데이터가 응답으로 전송되는가를 알 수 있다. ownership tag는 순차적으로 진행하는 도중에 거짓 비트가 질의에 대한 응답으로 전송되는 시점에 강제적으로 충돌을 발생시킨다. 충돌이 발생하면, 리더는 충돌에 의한 재 질의를 전송하며, ownership tag는 이 질의에 대해서는 침묵한다. 이렇게 침묵을 하는 이유는 리더로 하여금 방금 전의 충돌이 ownership tag에 의한 강제 충돌임을 알려주기 위한 것이다. 리더는 이런 충돌 즉, 재 질의에 침묵하는 충돌에 대해서는 방금 전 충돌위치가 거짓비트의 위치임을 알아채고, 해당 위치를 무시하며 진행한다.

즉, ownership tag가 하는 일은 일반태그에 맞춰 진행하는 것과 거짓 비트의 위치에서 강제 충돌

을 일으키는 일뿐이다.

2.4.3 리더

리더는 태그의 데이터를 읽고 거짓 비트를 만들어 삽입시키는 동작을 하고, 충돌이 나면 충돌 위치에서 해당 태그에게 충돌에 따른 재 질의를 전송하며 충돌 위치에서 해당 태그가 침묵을 한다면 ownership tag가 고의적으로 일으킨 충돌로 간주하고 해당 비트를 무시하며 진행한다. 그러나 일반 태그에 랜덤하게 생성된 거짓 비트를 데이터 사이에 삽입하는 방식이 갖고 있는 앞서 4.1절에서 언급한 문제 이외에 또 다른 문제가 남아있다. 만약, 삽입된 위치(level)가 완전한 이진트리가 구성되는 위치라면 ownership tag가 거짓 비트의 위치를 알리기 위해 강제로 일으킨 충돌이 리더에 의해 정상적인 충돌로 인식되면서 삽입된 거짓 비트가 정상 비트로 인식하는 문제가 발생한다. 따라서 앞선 리더의 동작 이외에 추가적인 동작을 추가하여 문제를 해결하고자 한다.

이 논문에서 리더와 태그간의 통신 방식으로 가정한 이진탐색트리 알고리즘의 경우 충돌이 발생하면 그 위치를 저장하고, 다음 비트로 이동하며, 이런 과정으로 탐색을 해서 마지막 비트에 도달하면 그 전 충돌 위치로 되돌아가서 탐색을 다시 시작하며 이 경우 b bit길이의 태그에서 마지막 b번째에서 충돌이 나려면 첫 비트(level 0)부터 $b-1$번째 비트까지 같고 마지막 b번째 비트가 다른 태그가 2개 이상 존재해야 한다. 따라서 태그의 수와 충돌이 발생가능한 위치는 그림 3에서 보는 것처럼 $h1$에 국한되어 있다. n개의 태그가 존재하는 경우, $h1$은 충돌이 일어날 확률이 높은 부분으로 이것은 $\lceil \log_2 n \rceil$ 정도가 될 것이다.

위 특성과 함께 리더가 태그의 정보를 먼저 읽은 다음 거짓 비트를 삽입하는 것에 착안하면 리더가 거짓비트의 위치를 알리고자 발생한 강제 충돌을 정상 충돌로 인식함으로써 발생하는 문제는 해결 가능하다.

무작위로 추출한 것처럼 보이지만, 사실 먼저 읽어 들인 태그들의 데이터를 보고, 완벽한 이진트리를 구성하는 높이($h1$)를 피해주기만 하면 된다. 약간의 처리 시간이 소모되긴 하지만, 평균적으로 완벽한 이진트리를 구성하는 높이는 완벽한 무작위로 생성된 트리가 128개 일 경우, 확률적으로 대략 앞쪽의 7-bit를 피하면 완벽한 이진트리가 나오지 않는다. 그러므로 몇 개의 위치만 피해준다면 나머지 삽입 가능한 위치에는 무작위로 추출한 비트열을 ownership tag에 삽입할 수 있다. 다시 말해서, 그림 3에서 $h1$ 부분만 피한다면 $h2$부분에서는 적어도 어느 한곳에서는 거짓 비트의 삽입의 감지가 가능하고, 이에 따라 허가 받은 리더는 모든 태그에 대해서 어느 곳에 거짓 비트가 삽입 되었는지 알 수가 있고, 정확한 태그 정보를 알아낼 수 있다.

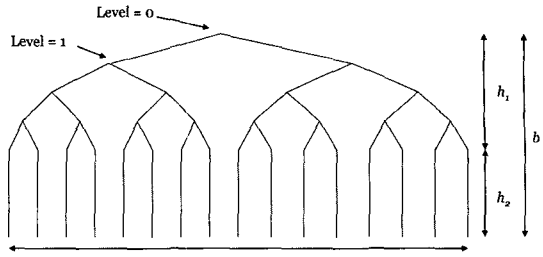


그림 3. 이진 탐색 트리의 작동 모델[1] ($h1$:완전 이진트리가 구성되는 부분, $h2$:완전 이진트리가 구성되지 않는 부분, n: 태그의 개수, b: 태그의 비트수)

리더는 먼저 구입한 물품의 일반 태그들을 읽고 완전이진트리가 생기는 부분은 체크한 후, 일반 태그에 삽입 전에 앞서 체크된 부분에 들어갈 거짓 비트를 0으로 만들고 나머지 부분은 랜덤하게 생성한다.

그림 4는 위 문제를 해결하기 위하여 모든 태그가 충돌이 나는 위치에 거짓 비트가 삽입되지 않고, 거짓 비트를 생성하기 위한 리더의 동작을 의사코드로 표현한 것이다.

III. 분 석

이 절에서는 아무런 추가적인 연산이 없는 상태 즉, 보통의 이진 탐색 트리를 이용한 리더와 태그

```
..... binary tree walking process .....

finish binary tree walking process
loop∞
 pop collision position list
 if n is max
  break loop
 if all tag occur collision in n^th bits
  n^th bogus bit is 0
  push the bogus bit list
 else
  make a bogus bit randomly
  push the bogus bit list
```

그림 4. bogus hiding 문제를 해결하기 위한 리더의 의사동작코드

표 2. 수식에 사용된 약어 설명

약자	설명
b	태그의 비트수
m	b 깊이를 표시할 수 있는 최소 비트수 ($\lceil \log_2 b \rceil$)
n	태그의 개수
BT(bit time)	1bit를 전송하는데 걸리는 시간
IBT(Inter Bit Time)	BT와 BT사이의 지연되는 시간

사이에서의 통신에 드는 시간과 비용과 ownership tag를 이용하였을 때, 그리고 ownership tag가 존재하지 않을 때 의 시간과 비용을 비교해보고자 한다.

3.1 이진 탐색 트리를 이용한 리더와 태그사이의 통신에 대한 분석

이진탐색트리 알고리즘은 [1]에서 논의된 것을 바탕으로 한다.

BT의 합은 다음과 같다.

$2n(b-\log_2 n)+(m+1)(n-2)$ BT

IBT의 합은 다음과 같다.

$2n(b-\log_2 n)+2(n-2)$ BT

3.2 Ownership tag에 대한 분석

그림 5에서 보다시피 ownership tag가 RF필드에 존재하여 강제로 충돌을 일으키는 경우 BT는 강제적 충돌당 (m+1), IBT는 3만큼 증가한다.

거짓 비트가 삽입된 트리는 $h1'$와 $h2'$로 사상할 수 있으며, $h1'$에는 거짓 비트가 삽입되지 않아야 하며, 이 논문에서도 $h1'$에는 거짓 비트가 없다고 가정

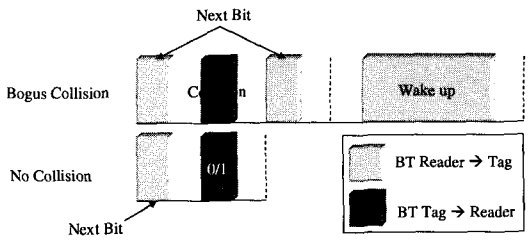


그림 5. 거짓 비트에 의한 추가 BT, IBT 예시

한다. $h2'$에 c개의 거짓 비트가 삽입된다. 그러므로 추가되는 충돌 횟수는 c와 태그의 개수 n에 비례한다. 그러므로 ownership tag가 RF필드에 없을 때의 BT와 IBT를 BT_{off}와 IBT_{off}라 하고, RF필드에 있을 때의 BT를 BT_{on}, IBT를 IBT_{on}이라 하면,

$$BT_{on} = BT_{off} + cn(m+1)$$
$$IBT_{on} = IBT_{off} + 3cn$$

(BToff와 IBToff는 같은 길이 태그의 이진탐색 트리 알고리즘에서 BT와 IBT와 같다.)

Ownership tag를 구현하면 표 3과 같은 성능을 얻을 수 있을 것으로 보인다.

128bits의 일반적 데이터들 사이에 1 bit부터 n bits까지 거짓 비트가 삽입되었을 때의 암호학적 강도는 다음의 값을 갖는다. MIN{$_{128}C_n$, 2^n} 그런데 이 그래프에서 $_{128}C_2$는 항상 2^n보다 작기 때문에 항상 $_{128}C_2$의 값을 갖게 된다.

그림 6에서는 위의 수식을 따른 그래프를 나타내고 있다. 그래프 맨 위의 직선은 2^{128} bits 비밀키의 복잡도를 나타내고, 맨 아래의 직선은 2^{64} bits 비밀키의 복잡도를 나타내고 있다. 포물선은 ownership tag에 대한 복잡도를 나타낸다. 위 그래프에 따르면 일반 데이터가 128bits일 때, 삽입되는 거짓 비트의 수가 64 bits정도에서 최대의 복잡도를 나타내고 있다. 완전히 랜덤하게 비트열을 생성한다고 가정했을 때, '0'과 '1'의 출현 확률이 1/2이

표 3. 일반 태그와 ownership tag의 BT, IBT의 비교 (n=100)

Tag Length	Ordinary tags		Protected tags without ownership tag		Protected tags with ownership tag	
	BT	IBT	BT	IBT	BT	IBT
64, (+32)	12157	11667	18655	18067	44255	28255
128, (+64)	25055	24467	37953	37267	95553	57153
256, (+128)	50573	50067	76451	75667	204451	114851
512, (+256)	102051	101267	153349	152467	434949	230149

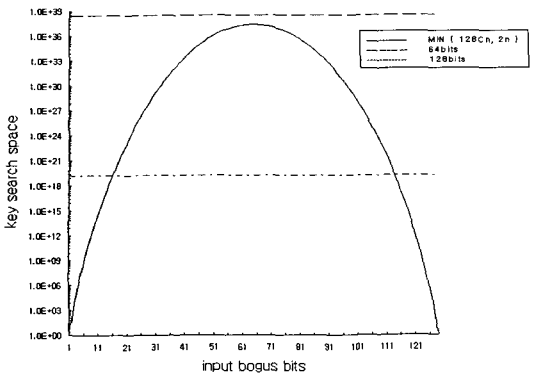


그림 6. 64bits, 128bits 키와 ownership tag에 삽입되는 비트수에 따른 key search space 비교 그래프

고, ownership tag가 갖고 있는 특수한 방식에 의하여, '0'은 null로 인식되어 진다. 따라서 평균적으로 삽입되는 거짓비트의 수는 64bit 정도가 될 것이다.

Ⅳ. 결 론

사용자는 자신의 태그의 정보가 필요한 곳에서만 리더에게 자신의 일반 태그와 ownership tag를 제시하는 것으로 태그 정보의 누출을 막을 수 있다. 즉, 사용자가 자신의 태그 정보를 읽게 허가한 리더만 태그의 정보를 올바르게 읽을 수 있으며, 그렇지 않은 리더에게는 ownership tag를 감추는 것으로 태그 정보를 보호할 수 있다. 예를 들자면, 소비자가 상점에서 물건을 구매하고 물건에 대한 태그와 구입한 상품들에 대한 ownership tag를 상점으로부터 제공받는다. 소비자는 제공받은 ownership tag가 악의적인 리더에게 노출되지 않도록 보관한 뒤, 집으로 간다. 만약 집까지 가는 도중에 어떤 악의적인 리더가 태그의 정보를 알아내고자 공격을 한다면, 악의적인 리더는 거짓 비트가 섞인 쓸모없는 정보를 얻게 될 뿐이다. 만약 소비자가 태그의 정보를 알아야 할 필요가 있다면 혹은 태그의 정보가 읽혀져야 한다고 판단한다면, 자물쇠를 열기위해 열쇠를 사용하는 것처럼 ownership tag를 노출시키는 것으로 원하는 목적을 이룰 수 있다. 이 논문에서 제시하는 ownership tag scheme은 어려운 암호학적 알고리즘의 적용 없이 또한, 하드웨어적으로 적은 비용으로 간단하게 구현이 가능하다. 기능, 속도 그리고, 보안 강도측면에서 적용이 가능하다. ownership tag는 수행 시간적 측면에서 표 4에서 비교한 것처럼 수행속도는 다소 느리다. 그러나 충돌을 이용하는 방식인 블로커 태그의 경우, ownership tag보다 훨씬 느리다는 것을 알 수 있다.(블로커 태그는 모든 비트에서 충돌을 일으킨다.) 또한 보안적인 강도 역시 삽입된 거짓비트의 수가 64bit 정도 일 때, 128bit의 비밀키 방식의 복잡도와 근접해있는 것을 그림 6에서 알 수 있다.

서비스 거부 공격에 대해 블로커 태그보다 덜 적합하다. 왜냐하면, 공격자가 ownership tag를 역이용하여 서비스 거부 공격을 시도할 경우, ownership tag는 블로커 태그처럼 모든 비트에서 충돌을 일으키지 않고 선택적으로 충돌을 일으키기 때문에 서비스 거부 공격에 악용될 요소가 적다. 또한, ownership tag 방식은 분실이나 도난에 대해 비교적 안전하다. 블로커 태그가 사용자의 실수 등에 약점을 보이는 반면 ownership tag는 일반 태그가 혼자 존재 할 때 보호받으므로 사용자의 사소한 실수에도 일반 태그는 보호 된다.

이진탐색트리뿐만 아니라 알로하 알고리즘에서도 적용가능하다. 현재 RFID 시스템에서 가장 널리 사용되고 있는 충돌방지 기법은 이진탐색트리와 알로하 방식이다. 이 논문에서는 이진탐색트리 기법을 사용한다는 가정 하에 태그정보 보호하기 위한 기법을 설명하였다. 그러나 ownership tag 기법은 알로하에서도 역시 적용가능하다. 알로하는 모든 태그들이 리더에게 데이터를 보내기 위하여 경쟁하는 과정을 거치게 된다. 태그가 미디어의 상태를 체크하기 어려운 이유로 RFID 시스템에서는 시간을 슬롯(또는 프레임)으로 나누고 동기화된 시간에 데이터를 전송하게 된다. 이런 알로하 기법 고유의 특성 때문에 ownership tag를 알로하기법 하에서 적용하기 위해서는 비트 동기화 즉, 보통의 일반 태그와 ownership tag와의 비트의 순서를 맞추기 위한 사전 작업이 필요하다.

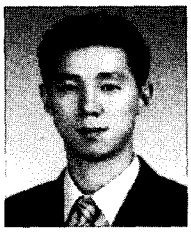
다른 용도로 사용이 가능하다. ownership tag 기법의 목적은 태그의 정보를 보호하는데 있다. 그러나 이 기법을 활용한다면 태그정보의 보호뿐 아니라, 위조방지 혹은 영수증 등 여러 가지로 사용될 수 있다.

참 고 문 헌

[1] 강전일, 박주성, 양대헌, "미리 분배된 난수를 이용한 빠른 충돌 방지 알고리즘" *한국통신학회*

논문지 Vol30 No3, 2005년 3월
[2] 강전일, 박주성, 양대헌, "RFID 시스템에서의 프라이버시 보호기술", *정보보호학회지* 제14권 6호 pp.28-36, 2004.12
[3] Ari Juels and Ronald L. Rivest and Michael Szydlo, "The Blocker Tag : Selective Blocking of RFID Tag for Consumer Privacy", RSA Laboratory pp. 71-79, May 1997.
[4] Ari Juels, John Brainard, "Soft Blocking : Flexible Blocker Tags on the Cheap", RSA Laboratory, MIT
[5] Stephen A. Weis, "Security and Privacy in Radio-Frequency Identification Devices", MIT, 2003
[6] Klaus Finkenzeller, 이근호 외 3명 공역, "RFID HANDBOOK", 2nd Edition in Korea, 2004, ISBN 89-314-2769-7
[7] Stephen A. Weis, Sanjay E. Sarma, Ronald L. Rivest, Daniel W. Engels, "Security and Privacy Aspect of Low-Cost Radio Frequency Identification Systems", In Security in Pervasive Computing, 2003
[8] 유성호, 김기현, 황용호, 이필중, "상태기반 RFID 인증 프로토콜", *정보보호학회* 논문지 14권 6호 pp.57-68, 2004. 12

〈著者紹介〉



박 주 성 (Ju-sung Park)
2004년 2월: 인하대학교 컴퓨터 공학과 졸업
2004년 3월~현재: 인하대학교 정보통신대학원 석사과정
〈관심분야〉 RFID 보안

강 전 일 (Jeon-il Kang) 학생회원
2003년 2월: 인하대학교 컴퓨터 공학과 졸업
2004년 3월~현재: 인하대학교 정보통신대학원 석사과정
〈관심분야〉 RFID 보안

양 대 헌 (DaeHun Nyang) 정회원
1994년 2월: 한국과학기술원 과학기술 대학 전기 및 전자 공학과 졸업
1996년 2월: 연세대학교 컴퓨터 과학과 석사
2000년 8월: 연세대학교 컴퓨터 과학과 박사
2000년 9월~2003년 2월: 한국전자통신연구원 정보보호연구본부 선임연구원
2003년 2월~현재: 인하대학교 정보통신대학원 조교수
〈관심분야〉 암호이론, 암호프로토콜, 인증프로토콜, 무선 인터넷 보안