그리드 환경에서 노드별 성능정보를 고려한 효율적인 RFID 태그 판별에 관한 연구

신명숙* · 이준**

A Study on the Efficient RFID Tag Identification considering Performance Information of Individual Nodes in a Grid Environment

Myeong-Sook Shin* · Joon Lee**

요 약

RFID는 차세대 유비쿼터스 환경에서 중요한 기술적인 위치를 차지할 것으로 주목받고 있다. 또한 향후 다양한 분야에서 새로운 시장을 창출할 것으로 기대되고 있다. RFID 시스템을 보편화하기 위해서는 무엇보다도 무선 통신으로 인한 프라이버시 보호 문제의 해결과 대량의 태그를 처리할 수 있는 빠른 처리 능력이 요구된다. 이 논문에서는 프라이버시 보호 기법의 필수 보안 요건 3가지를 모두 만족하면서 태그를 빠르게 판별할 수 있는 효율적인 태그 판별 방법을 제안한다. 이 방법은 먼저 Hash-Chain의 병행성을 분석하여 그리드 환경으로 이식하고 그리드 환경에서 각 노드의 성능 측정을 한다. 그런 다음 그 측정 결과를 이용하여 SP들을 분할하여 각 노드별로 태그를 판별할 수 있는 SP분할 알고리즘을 제안하고 그리드환경에서 구현하고자 한다.

ABSTRACT

RFID is recognized to technically occupy important position in ubiquitous computing environment and expected to create new markets in a variety of fields from now on. In order to generalize RFID system, it is required to solve the problem of privacy invasion and expedite lots of tags We suggest efficient RFID Tag Identification to identify tags quickly on the satisfaction with 3 security requirements of privacy protection in this paper. This methods are transferred to Grid environment through parallel analysis of Hash-Chain, and we measure performance of each nodes under the Grid environment. Then, We'll suggest SP-Division Algorithm to identify tags with each nodes and implement it in a Grid environment.

키워드

Computational Grid, MPICH-G2, Hash-Chain scheme, RFID, Privacy protection
계산 그리드, MPICH-G2, 정보보호, 해시 체인 기법, RFID, Privacy protection

Ⅰ. 서 론

RFID 시스템은 개개인은 물론 사회 전반에 편리성과 유용성을 제공하고 있기 때문에 다양한 분야에서 빠르게 발전하고 현실화되고 있다. 하지만 RFID를 사용하였을 때 개인의 프라이버시 침해라는 심각

* 조선대학교 전자정보공과대학 컴퓨터공학부(sms-sy@nate.com)
** 교신저자 : 조선대학교 전자정보공과대학 컴퓨터공학부 교수(jlee@chosun.ac.kr)
접수일자 : 2011. 08. 22 심사(수정)일자 : 2011. 09. 24 게재확정일자 : 2011. 10. 12

한 이슈가 제기되고 있다[1][2].

사용자 프라이버시 침해 문제를 해결하기 위해 많은 연구들이 진행 중에 있으며, 이 중에서 프라이버시 보호를 위한 필수 보안 요건인 기밀성, 불구분성과 전방보안성에 가장 안전한 Hash-Chain 기법[3]을 이용한 연구들이 진행 중에 있다. 그러나 Hash-Chain 기법은 백엔드 서버에서 태그를 판별하는 데 소요되는 시간이 많이 걸리므로 백엔드 서버에서의 계산량이 어느 정도 이상 많아지면 실시간으로 태그를 판별하는 것이 불가능해진다.

따라서 Hash-Chain 기법은 RFID 태그의 취약점인 사용자 프라이버시 보호를 위한 필수 보안 요건을 만족하지만, 효율성 측면에서 태그 판별 시간을 줄이기 위한 방법이 필요하다.

본 논문에서는 그리드 환경에서 각 노드의 시스템 상태를 반영하여 SP들을 분할하고 Hash-Chain 기법의 분석을 통해 이 분할된 SP들을 병행처리[4]가 가능하도록 그리드에 이식한다. 그리고 기본적으로 k개의 노드가 있을 경우, 각 노드에 SP(Startpoint)들을 균등하게 분할하여 적용한다[5]. 하지만 이 방법은 이질적인 시스템으로 구성되는 그리드 환경에서는 최적화된 성능을 얻을 수 없었다. 이를 해결하기 위한 방법으로 노드의 성능에 따라 SP들을 다양하게 분할하는 SP 분할 알고리즘을 제안하고, 이를 그리드 환경에서 구현한다.

II. 본 론

2.1. 2.1 RFID

RFID는 IC 칩을 내장한 태그에 축척된 정보를 무선 주파수를 이용하여 원격에서 인식하는 방식이다. 또한 언제, 어디서나, 자동 확인 또는 위치 추적이 가능하여 정보 갱신 및 수정이 가능하다는 점이다[6].

2.2 개인 프라이버시

RFID 시스템 도입으로 인한 개인 정보 침해 문제의 심각성은 개인 신상정보의 단순한 수집, 이용보다 개인의 여러 가지 거래 내용, 사회활동 내용과 신상정보를 조합함으로써 본인도 모르는 사이에 분석 활용될 수 있다는 점이다.

2.3 Hash-Chain 기법

M. Ohkubo 등이 제안한 기법으로 일방향 해시함수를 사용하여 안전한 프라이버시 보호가 보장되는 기법이다[3].

백엔드 시스템에는 ID_t와 해시 시드 값 $s_{t,1}$이 저장되며 태그에도 동일한 $s_{t,1}$값을 저장하고, 두 개의 해시함수 H와 G로 구현한다. 리더의 질의에 대해 태그는 $a_{t,i} = G(s_{t,i})$를 수행하여 리더에게 응답하며 자신의 시드 값인 $s_{t,i}$는 $H(s_{t,i})$를 통해 $s_{t,i+1}$로 갱신한다. 그러나 이 기법은 서버에서 태그를 판별하기 위한 계산량이 많다는 문제점이 있다.

2.4 그리드 컴퓨팅

그리드 컴퓨팅이란 지리적으로 분산된 컴퓨터를 결합시켜 발생 장소를 의식하지 않고 원하는 만큼 사용할 수 있게 한다는 것에서 나온 것 이다. 그리드 시스템은 계산 그리드, 데이터 그리드, 액세스 그리드로 분류할 수 있다[7].

계산 그리드는 많은 자원을 연결하여 계산을 해결할 수 있게 하고, 데이터 그리드는 원격지의 분산된 자료들을 통합하여 분석할 수 있게 해 주는 그리드다. 액세스 그리드는 분산처리를 필요로 하는 어플리케이션을 위한 그리드다.

2.5 글로버스 툴킷

글로버스 툴킷은 그리드 서비스를 제공하는 Middleware로서 계산 그리드를 구축하는데 필요한 기본적인 기술이다.

2.6 MPICH-G2

MPICH-G는 MPI에 글로버스가 제공하는 서비스를 가미함으로서 그리드 컴퓨팅에서의 병렬 프로그래밍 환경의 기초를 구성하였으며, MPICH-G2 (Grid-enable MPI Chameleon 2)를 개발하였다.

III. 제안한 방법

기존의 Hash-Chain 기법은 안전한 보안성이 보장되지만 분산 환경에서 엄청난 태그 수의 증가로 인해 태그를 판별하는데 막대한 계산 시간이 요구되는 문제점이 있다. 이러한 문제점을 해결하기 위해서 노드의 성능에 따라 SP들을 다양하게 분할하는 SP분할 알고리즘을 제안한다.

3.1 그리드 환경으로의 이식

RFID 프라이버시 보호를 위해 적용한 Hash-Chain 기법은 그림 1과 같이 하나의 태그를 판별하기 위한 계산에서, 백엔드 시스템에서는 모든 $1 \leqq t \leqq m$와 에 대해서 $a_{t,i}^{'} = G(H^{i-1}(s_{t,1}))$를 계산한다.

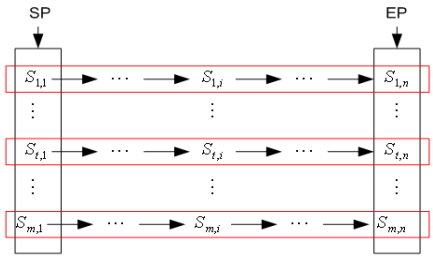

그림 1. 서로 다른 SP에 대해 독립적
Fig. 1 Independence from mutually different SP

Hash-Chain 기법에서 SP로부터 EP를 계산하는 과정은 그림 1과 같이 서로 다른 SP에 대해 독립적이다. 또한 태그 판별 과정도 서로 종속성이 없이 독립적이다.

3.2 SP균등분할 알고리즘

본 논문에서는 계산 그리드를 이용하여 문제를 해결하기 위해서는 제기된 문제의 병행성을 분석한 후 k개의 노드로 SP들을 균등하게 분할한다.

여기서 SP[i]는 분할된 SP들의 개수이고, m은 전체 태그 개수이다. 또한 k는 노드 수이며 i는 {1, 2, 3…k}이다. 동시에 수행될 수 있는 노드 수가 많을수록 그리드의 활용도는 높아진다.

노드별로 균등하게 SP들 분할은 Hash-Chain 계산 테이블에서 k개의 노드로 SP들을 균등하게 분할하는 방법으로서 선택할 SP들의 개수는 m/k이 된다. 그 이후의 과정은 선택된 SP들을 이용하여 각 노드에서 독립적으로 태그 판별 과정을 수행한다. k 개의 Slave에서는 m/k 개의 SP들에 대하여 Hash-Chain을 계산하면서 Master에서 받은 $a_{t,i}$와 비교하면서 일치하는 값을 찾는다. 여기서의 검색 방식은 순차검색을 이용한다. 결국 Slave는 m/k 개의 SP들에 대하여 Hash-Chain 계산을 하면서 $a_{t,i} = a'_{t,i}$를 비교하여 일치하는 값을 찾게 되면 해시 시드 값 $s_{1,1}, s_{2,1}, \cdots, s_{t,i}, \cdots, s_{m,1}$ 에 해당하는 식별 정보 ID_t를 Master에게 전송하고 프로토콜을 종료하게 된다.

3.3 SP 분할 알고리즘

기본적으로 SP들을 균등하게 분할하여 적용한 방법은 이질적인 시스템으로 구성되는 그리드 환경에서는 최적화된 성능을 얻을 수 없었다. 이를 해결하기 위한 방법으로 노드의 성능에 따라 SP들을 다양하게 분할하는 SP분할 알고리즘을 제안한다.

SP분할 알고리즘에서 사용되는 변수와 관련된 식들은 다음과 같다.

표 1. SP 분할 알고리즘에서 사용되는 변수 정의
Table 1. Definition of Parameters used in SP-Division Algorithm

Parameter	Content
m	Total Number of SPs
k	Total Number of Nodes
i	{1, 2, 3, …, k}
node[i]	Used Number of Nodes
perf[i]	Inverse Number of Performance Measurement Result of node[i]

식 (1)에서 각 노드에 대한 성능 지수 perf[i]는 해시 계산 시간의 역수이다.

$$perf[i] = \frac{1}{node[i]} \tag{1}$$

각 노드들의 성능을 측정하여 SP들을 분할하는 SP분할 비율은 식 (2)와 같다.

$$SP\ 분할비율 = \frac{perf[i]}{\sum_{i=1}^{k} perf[i]} \tag{2}$$

SP 분할비율에 따른 SP들의 분할은 식 (3)과 같다.

$$SP[i] = ROUND\left(\frac{m \bullet perf[i]}{\sum_{i=1}^{k} perf[i]}\right) \quad (3)$$

그림 2에서와 같이 백엔드 서버에서의 태그판별시간을 단축하기 위해서는 SP분할 알고리즘을 이용하여 각 노드별로 동시에 수행함으로써 대량의 SP를 처리할 때 태그판별처리 시간을 단축할 수 있는 모델을 제공한다.

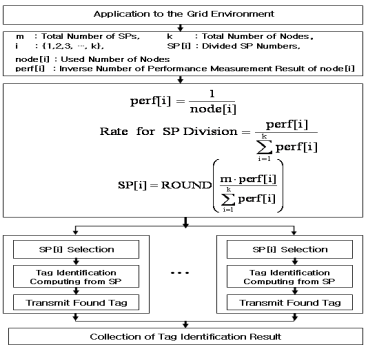

그림 2. SP 분할 알고리즘을 이용한 흐름도
Fig. 2 Flowchart using SP-Division Algorithm

기존 방법과 마찬가지로 태그/리더-그리드시스템 간의 통신은 무선 전파를 이용한 통신으로써 도청의 가능성이 있는 것으로 간주한다.

Ⅳ. 실험 및 분석

4.1 실험 환경

성능평가를 위한 실험 환경은 표 2와 같이 구성된다. 그리고 Hash-Chain 연산에 사용한 일방향 해시 함수는 128bit의 md4, md5를 이용하였으며 검색 방법은 순차 검색을 이용하였다.

그리드 환경에서 메시지 전달은 Master와 Slave 사이의 통신을 위해서 필요하다. 본 논문에서는 Master와 Slave 사이의 통신을 위해 MPICH-G2를 이용하였으며 MPI_Recv()와 MPI_Send() 함수를 이용하여 메시지 교환을 한다.

RFID 태그의 샘플 데이터는 16byte의 시드 값을 갖는 임의의 데이터를 생성하여 사용한다. 전체 태그의 개수 m, 즉 SP는 1,000, 2,000, 3,000, 4,000개로 증가시키면서 실험하였고 최대 Hash-Chain의 길이 n은 1,000으로 실험하였다. 성능평가 방법은 Hash-Chain 길이(n)와 태그판별개수는 고정된 상태에서 SP 개수와 노드 수를 증가하면서 태그판별시간을 평가하였다. 단 테스트는 각각 100번씩 수행하여 평균으로 산출하였다.

표 2. 구현을 위한 하드웨어 및 소프트웨어 환경
Table 2. Environment of Software and Hardware for Implement

Item		Content
H/W	Master	Intel Xeon 5130 2.0G, dual core 4.0GB
	Slave1	Pentium4 2.4G 512M
	Slave2	Pentium4 3.0G 512M
	Slave3	Pentium4 2.8G 512M
	Slave4	Intel Xeon 5130 2.0G, dual core 4.0GB
S/W	OS	RadHat Linux 9.0, Kernel Version : 2.4.20-8
	Middleware	Globus Toolkit 2.2.2
	MPI	MPICH-G2
	Language	C

4.2 보안 분석

기존의 기법들과 제안한 방법에 대해 필수 보안 요건인 기밀성, 불구분성, 전방보안성 등의 보장 여부는 아래와 같이 보장한다.

기밀성은 태그가 송신하는 값은 일방향 해시 함수 G의 계산 결과 값 $a_{t,i}$는 물품에 대한 어떠한 정보도 가지고 있지 않기 때문에 문제가 되지 않는다. 또한 태그의 내부에도 물품에 대한 직접적인 정보가 들어있지 않기 때문에 공격자가 태그의 송신정보를 통해서 물품에 대한 유용한 정보를 획득할 가능성은 없다.

불구분성은 공격자가 태그의 다음 단계의 송신 값 $a_{t,i+1}$을 현재까지의 모든 송신 값 $a_{t,i}$를 가지고 예측하기 불가능하다. 이는 일방향 해시 함수 G의

입력 값 $s_{t,i+1}$을 공격자가 알아내기는 어렵기 때문이다. $s_{t,i+1}$을 예측할 수 있으려면 $s_{t,i}$ 값을 $a_{t,i}$ 알아내야 하는데 $s_{t,i}$ 값은 외부로 전송하지 않고, G 해시 함수를 거쳐 계산된 결과를 전송하기 때문에 일방향 해시 함수의 특성상 $s_{t,i}$를 예측하는 것은 계산적으로 불가능하다.

전방보안성은 태그가 물리적인 공격을 당하는 경우에는 현재의 시드 값 $s_{t,i}$가 노출이 되지만, 공격자는 $s_{t,j}$ 값을 알아낼 수 없다. 이는 일방향 해시 함수 H의 일방향성을 근거로 한다. 또한, 공격자는 태그가 전송하는 값 $a_{t,i}$를 알아내었다하더라도 G 해시 함수의 일방향성을 근거로 $s_{t,i}$를 알아내기는 어렵다.

4.3 실험 결과

그림 3, 그림 4, 그림 5는 Hash-Chain 길이를 1,000번으로 고정하고 SP들의 총수는 1,000, 2,000, 3,000, 4,000으로 증가시킨다. 노드 수를 2에서 4까지 확장하면서 단일 노드, 균등분할, SP 분할 알고리즘 간의 각각의 태그 판별 시간을 비교한 그래프들이다.

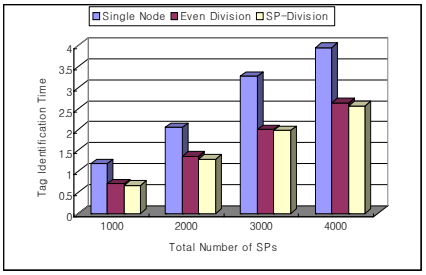

그림 3. SP들의 총수에 따른 성능 비교 : 노드 수(2)
Fig. 3 Performance Comparison by the Total Number of SPs : Node Numbers(2)

그림 3에서는 Hash-Chain 길이 1,000번, 노드 수 2는 고정하고 SP들의 총수는 1,000, 2,000, 3,000, 4,000으로 증가하면서 측정한 결과는 단일 노드와 비교하여 SP들의 총수가 1,000개 일 때 41%, 43%, 2,000개 일 때 33%, 37%, 3,000개 일 때 39%, 40%, 4,000개 일 때 33%, 35%가 향상되었다.

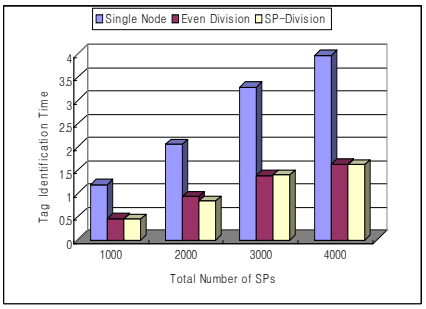

그림 4. SP들의 총수에 따른 성능 비교 : 노드 수(3)
Fig. 4 Performance Comparison by the Total Number of SPs : Node Numbers(3)

그림 4는 위와 같은 조건하에서 단지 노드 수를 3으로 하여 측정한 결과는 단일 노드와 비교하여 SP들의 총수가 1,000개 일 때 41%, 62%, 2,000개 일 때 54%, 59%, 3,000개 일 때 72%, 75%, 4,000개 일 때 83%, 84%가 향상되었다.

또한 그림 5도 같은 조건하에서 노드 수를 4로 측정한 결과를 단일 노드와 비교하면 SP들의 총수가 1,000개 일 때 70%, 73%, 2,000개 일 때 68%, 69%, 3,000개 일 때 70%, 74%, 4,000개 일 때 66%, 67%가 향상되었다.

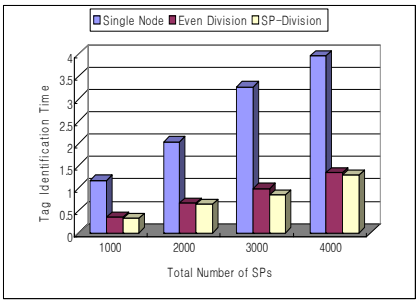

그림 5. SP들의 총수에 따른 성능 비교 : 노드 수(4)
Fig. 5 Performance Comparison by the Total Number of SPs : Node Numbers(4)

따라서 위의 결과와 같이 SP들의 총수와 노드 수에 따라 태그 판별 시간을 비교 분석하면 SP들의 총수와 노드 수가 증가할수록 향상률이 증가함을 볼 수 있었다.

Ⅴ. 결 론

본 논문에서는 효율적인 RFID 태그 판별을 위해서 노드의 성능에 따라 SP들을 다양하게 분할하는 SP 분할 알고리즘을 제안하여 구현하였다.

제안한 방법의 구현 결과, Hash-Chain 길이 1,000번으로 고정된 상태에서 노드 수 2, 3, 4로 확장하면서 SP들의 총수가 2,000개일 때의 성능을 비교해보면 단일 노드와 균등분할 알고리즘을 비교하면 33%, 54%, 68%로 향상되고, 단일 노드와 SP 분할 알고리즘을 비교하면 37%, 59%, 69%로 성능이 향상되었다. 테스트는 각각 100번씩 수행하여 가장 먼저 찾은 시간의 합을 100으로 나누어 태그 판별 시간을 산출하였다.

결과적으로 위에서 제시한 결과를 분석해보면 SP들의 총수의 증가는 물론 노드 수가 증가하면 할수록 단일 노드에 비해서 SP 분할 알고리즘 성능이 향상됨을 확인할 수 있었다. 따라서 많은 계산 능력을 필요로 하는 문제를 그리드 환경에 적용하고자 할 때 본 논문에서 제안한 그리드 환경을 도입하여 SP 분할 방법을 사용한다면 이질적인 시스템 구성으로 인한 성능 저하 문제를 제거함으로써 최적화된 성능을 얻을 수 있을 것이다.

향후 연구 방향으로는 다양한 네트워크로 연결된 확장된 그리드 환경에서 SP 분할 알고리즘을 적용하였을 경우의 성능 향상 정도의 연구가 요구된다.

참고 문헌

[1] Good, N., J. Han, E. Miles, D. Molnar, D. Mulligan, L. Quilter, J. M. Urban and D. Wagner, “Radio frequency ID and privacy with information goods”, Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society, October, Washington, U.S.A., ACM Press, pp. 41-42, 2004.

[2] 김성철, 양동훈, 송찬후, 전형순, “RFID 확산에 따른 정보보호 문제와 기업의 대응전략“, 정보통신정책학회논문지, Vol. 12 No. 1, pp. 149-168, 2005.

[3] M. Ohkubo, K. Suzuki, and S. Kinoshita. “Cryptographic approach to "privacy-friendly" tags“, In RFID Privacy Workshop, MIT, USA, Nov., 2003.

[4] Hyung-Jun Kim, Sung-up Jo, Yong-won Kwon, So-Hyun Ryu, Yong-je Woo, Chang-Sung Jeong, and hyoungwoo Park, “Fast Parallel Algorithm for Volume Rendering and Its Experiment on Computational Grid”, ICCS 2003, LNCS 2657, pp. 610-618, 2003.

[5] 신명숙, 이준, "계산 그리드를 이용한 대량의 RFID 태그 판별 시간 단축 방법", 전자정보통신기술학회논문지, Vol. 5 No. 5, pp. 547-554, 2010.

[6] 이동민, “RFID 기반 상품의 효율적 라이프사이클관리를 위한 통합시스템 설계”, 대한산업공학회 학술지, Vol. 19, No. 4, pp. 333-341, 2006.

[7] 이춘희, “그리드 컴퓨팅(Grid Computing)”, 정보처리학회 학회지, Vol. 10, No. 01, pp. 0109~0120, Jan., 2003.

저자 소개

신명숙(Myeong-Sook Shin)

1992년 2월 광주대학교 전자계산학과 졸업(공학사)
1996년 2월 광주대학교 대학원 컴퓨터학과 졸업(공학석사)
2008년 8월 조선대학교 대학원 전자정보공과대학 컴퓨터학과 졸업(공학박사)
조선대학교 전자정보공과대학 컴퓨터공학부 겸임교수
※ 관심분야 : 시스템소프트웨어, 유비쿼터스컴퓨팅, 정보보호

이준(Joon Lee)

1979년 2월 조선대학교 전자공학과 졸업(공학사)
1981년 2월 조선대학교 대학원 전자공학과 졸업(공학석사)
1997년 2월 숭실대학교 대학원 전자계산학과 졸업(공학박사)
조선대학교 전자정보공과대학 컴퓨터공학부 교수
※ 관심분야 : 운영체제, 정보보호, 유비쿼터스컴퓨팅