실시간 보안 데이타베이스 시스템에서 병행수행 제어를 위한 얼림 기법

(A Freezing Method for Concurrency Control in Secure Real-Time Database Systems)

박 찬 정 † 한 희 준 †† 박 석 †††

(Chanjung Park) (Heejun Han) (Seog Park)

요 약 실시간 응용을 위한 데이타베이스 시스템은 각 트랜잭션에 부여된 시간 제약사항을 만족해야 한다. 일반적으로, 트랜잭션 스케줄러는 종료시한이라는 형태로 각 트랜잭션의 시간 제약사항을 표현하며 이는 그 트랜잭션의 우선순위로 사용이 된다. 최근, 보안이 많은 실시간 응용에서 중요한 요구사항이 되고 있다. 많은 시스템에서 기밀성을 띤 정보가 서로 다른 인가등급을 가진 사용자에 의해 공유된다. 시간 제약사항과 동시에 기밀 데이타를 관리하는 응용에서 진보된 데이타베이스 시스템의 사용이 증가하면서 시간 제약사항과 보안 요구사항을 만족하는 병행수행 제어 프로토콜의 개발이 요구되어지고 있다. 본 논문에서는 보안 요구사항과 실시간 요구사항을 보장하는 두 개의 병행수행 제어 프로토콜들을 제안한다. 제안하는 프로토콜들은 다중버전 로킹에 기반을 두고 있지만 두 가지 제약사항을 만족시키기 위해서 얼림이라는 새로운 기법을 사용한다. 또한, 제안하는 프로토콜의 정확성을 증명하고 기존의 로킹에 기반을 둔 프로토콜들보다 높은 병행수행 정도를 제공함을 증명한다. 마지막으로, 몇 가지 예제를 통해 다른 프로토콜들과 성능 분석을 수행하여 성능 향상이 있음을 보인다.

키워드 : 다중버전 병행수행 제어, 다단계 보안, 실시간 데이타베이스 시스템, 트랜잭션 관리

Abstract Database systems for real-time applications must satisfy timing constraints associated with transactions. Typically, a timing constraint is expressed in the form of a deadline and is represented as a priority to be used by schedulers. Recently, security has become another important issue in many real-time applications. In many systems, sensitive information is shared by multiple users with different levels of security clearance. As more advanced database systems are being used in applications that need to support timeliness while managing sensitive information, there is an urgent need to develop concurrency control protocols in transaction management that satisfy both timing and security requirements. In this paper, we propose two concurrency control protocols that ensure both security and real-time requirements. The proposed protocols are primarily based on multiversion locking. However, in order to satisfy timing constraint and security requirements, a new method, called the FREEZE, is proposed. In addition, we show that our protocols work correctly and they provide a higher degree of concurrency than existing multiversion protocols. We present several examples to illustrate the behavior of our protocols, along with performance comparisons with other protocols. The simulation results show that the proposed protocols can achieve significant performance improvement.

Key word : multiversion concurrency control, multilevel security, real-time database systems, transaction management

---

· 본 논문은 한국과학재단 목적기초연구(R04-2001-00121)지원으로 수행되었음.

† 정 회 원 : 제주대학교 컴퓨터교육과 교수
cjpark@cheju.ac.kr

†† 비 회 원 : 매크로임팩트
heezuni@macroipact.com

††† 종신회원 : 서강대학교 컴퓨터학과
spark@dblab.sogang.ac.kr

논문접수 : 2001년 7월 12일
심사완료 : 2002년 3월 7일

1. 서 론

데이타베이스 시스템(database system)은 미리 정의된 연산들의 집합과 데이타베이스의 효율적인 관리 및 이러한 데이타에 대한 연산들로 구성된 트랜잭션들을 위한 트랜잭션 관리를 제공하는 시스템을 의미한다[6]. 그리고, 실시간 데이타베이스 시스템(real-time database system)이란 트랜잭션이 종료시한(deadline)과 같은 시간적 제약사항을 갖는 시스템으로, 시스템의 정확성은 논리적 일관성 뿐만 아니라 결과가 생성되는 시간에 의존한다[1][9][19][20]. 한편, 다단계 보안 데이타베이스 시스템(multilevel secure database system)은 하나 이상의 인가등급 (clearance level)을 가진 사용자들에 의해 공유되고, 하나 이상의 분류등급(classification level)을 가진 데이타를 관리하는 시스템이다[11][12][15][17][18]. 트랜잭션 관리자는 데이타베이스의 논리적인 일관성을 보장해야 할 뿐만 아니라 하위등급을 갖는 사용자에게로 기밀성을 지닌 상위등급의 데이타가 흐를 수 있는 통로인 비밀채널(covert channel)을 제거하여야 한다. 비밀채널의 방지를 위해서 항상 하위등급을 갖는 트랜잭션들이 상위등급을 갖는 트랜잭션들에 의해 지연되지 않도록 처리하고 있다[3].

최근 보안에 대한 관심이 고조되면서 기존의 실시간 응용에서도 보안 요구사항이 중요한 문제로 대두되고 있다. 따라서 시간적 제약사항과 보안 요구사항을 모두 갖는 응용을 지원하기 위한 데이타베이스 시스템의 개발이 요구되어지고 있고[7][21], 그 결과 실시간 보안 데이타베이스 시스템(real-time secure database system)이 제안되었다. 예를 들어, 실시간 데이타베이스 시스템이 필요한 대표적인 응용 분야로 주식거래시스템을 들 수가 있다. 주식거래시스템에서는 일반적으로 주가가 급격하게 변화한다. 이러한 환경에서 어느 한 사용자가 자신의 주식을 팔거나 새로운 주식을 사고자 한다면, 이에 대응되는 매매작업은 적절한 시간 내에 처리되어져야 한다. 또한, 시스템 내의 데이타는 임의의 사용자에게 누출되어져서는 안된다. 따라서, 주식거래시스템의 데이타베이스 시스템은 시간적인 제약사항과 보안 요구사항을 만족시켜야 한다.

실시간 데이타베이스 시스템을 위한 병행수행 제어 프로토콜이나 다단계 보안 데이타베이스 시스템을 위한 병행수행 제어 프로토콜들은 많이 제안되어 왔다. 하지만, 실시간 보안 데이타베이스 시스템을 위한 병행수행 제어 프로토콜들은 그다지 많지 않고 제안된 일부 프로토콜들은 두 요구사항을 완전히 지원하지 못하고 있다. Son의 연구[16]에서 SRT-2PL(secure real-time two phase locking) 프로토콜이 제안되었다. 이 프로토콜은 각 데이타 항목에 대해서 같은 보안등급 트랜잭션을 위한 기본 복사본(primary copy)과 높은 보안등급 트랜잭션의 하향 판독을 위한 이차 복사본(secondary copy)을 유지한다. 임의의 트랜잭션들이 자신과 같은 등급의 한 데이타 항목에 대해 기록연산을 수행하면, 데이타 일관성 유지를 위해 기록으로 인한 갱신 내용들을 갱신큐(update queue)라는 자료구조를 사용하여 저장한다. 일정 기간이 지난 후, 갱신 큐의 내용을 이차 복사본에 반영하는데 이 과정에서 비밀채널은 방지할 수 있으나 만일, 이차 복사본을 판독하고 있는 우선순위가 높은 상위 트랜잭션이 존재하는 경우, 우선순위가 높은 상위 트랜잭션을 취소시켜 우선순위 역전현상(priority inversion)[1]이 발생할 수 있다. 한편, Son의 다른 연구[22][23]에서는 노이즈(noise) 개념을 도입하여 보안 요구사항이 실시간 요구사항과 어떤 방식으로 통합될 수 있는지를 제시하였지만 완전하게 두 가지 요구사항을 만족시키지는 못하고 있다. 즉, 부분적인 요구사항의 만족으로 인해 보안 요구사항과 실시간 요구사항 모두가 완전하게 만족되어야 하는 환경에서 문제점이 발생한다. George[9]의 연구에서는 보안 요구사항은 정확성 기준으로 실시간 요구사항은 성능에 대한 요구사항으로 간주하여, 항상 높은 우선순위를 갖는 트랜잭션이 낮은 보안등급을 갖게 하고 있다. 만일, 이러한 우선순위 할당 정책을 채택하게 되면, 보안과 실시간 요구사항을 쉽게 만족시킬 수 있다. 이러한 방식은 어떤 면에서 매우 합당한 방식일지 모르나, 상위 보안등급 트랜잭션이 임의의 우선순위를 할당받을 수 있는 상황을 고려하기 위한 좀더 일반적인 접근방법이 요구된다.

기존의 프로토콜들은 시간적 제약사항과 보안 요구사항을 만족시키는 것이 프로토콜의 주된 목적이었기 때문에, 실제 응용에서 발생되는 트랜잭션들의 특성에 대한 정보를 이용하지 않았다. 판독전용 트랜잭션(read-only transaction : ROT)은 다량의 데이타 항목에 대해 장기간 판독 연산만을 수행하는 성질을 가지기 때문에, 갱신 트랜잭션의 수행을 상당히 오랜 기간 동안 지연시키거나 갱신 트랜잭션에 의해 쉽게 취소될 수가 있다[6]. 따라서, 판독전용 트랜잭션을 갱신 트랜잭션과 구분하여 처리하게 되면, 데이타 충돌을 감소시켜 시스템의 병행수행 정도를 높여준다. 사실 여러 실시간 응용에

1) 우선순위 역전현상(priority inversion)은 우선순위가 높은 트랜잭션이 우선순위가 낮은 트랜잭션에 의해 연산의 처리가 지연되는 현상을 의미한다.

서 판독전용 트랜잭션이 높은 비중을 차지하고 있기 때문에, 실시간 데이타베이스 시스템에서 판독전용 트랜잭션을 효과적으로 처리하는 것이 상당히 중요시되었다[14][24].

본 논문에서는 실시간 보안 데이타베이스 시스템을 위한 새로운 프로토콜인 B-Freeze 프로토콜과 판독전용 트랜잭션을 고려한 R-Freeze 프로토콜 제안한다. 제안하는 프로토콜들은 단일복사본 직렬화 가능성을 보장하며 비밀채널과 우선순위 역전현상을 제거하여 보안 요구사항과 실시간 요구사항을 모두 만족한다. 또한, 프로토콜들은 다중버전에 기반을 두고 있고 모든 요구사항을 만족시키기 위해 얼림(freeze)이라는 새로운 기법을 사용한다. 일반적으로, 병행수행 제어 프로토콜이 다중버전에 기반을 두고 있으면 보다 많은 저장공간을 차지하게 된다. 따라서, 같은 단일 버전에 기반을 둔 프로토콜들 혹은 2 개의 버전을 사용하는 프로토콜들도 제안되었지만, 이들은 모두 한 가지 요구사항의 희생을 초래할 수밖에 없다[16][22].

본 논문의 구성은 다음과 같다. 2절에서는 실시간 보안 데이타베이스 시스템의 정확성 기준을 기술하고 제안하는 프로토콜의 보안모델 및 제안하는 프로토콜의 기반이 되는 다중버전 2단계 로킹 (multiversion two phase locking : MV2PL) 프로토콜에 대해 기술한다. 3절에서는 제안하는 두 개의 프로토콜의 특성을 기술한다. B-Freeze 프로토콜은 트랜잭션의 특성을 고려하지 않은 반면, R-Freeze 프로토콜은 트랜잭션을 판독전용과 갱신 트랜잭션으로 분리하여 처리한다. 또한, 몇 가지 예제를 통해 제안하는 프로토콜들이 트랜잭션을 처리하는 과정을 살펴보고 기존의 다중버전에 기초를 둔 프로토콜보다 높은 병행수행 정도를 보장함을 보인다. 4절에서는 제안하는 프로토콜들의 정확성을 증명하고 5절에서는 다른 프로토콜들과 성능을 분석하여 제안하는 프로토콜이 보다 나은 성능을 가짐을 보인 후, 마지막으로 6절에서 결론을 맺는다.

2. 관련연구

이 절에서는 전통적인 데이타베이스 시스템을 위한 프로토콜들 중, 다중버전에 기초를 둔 병행수행 제어 프로토콜을 위한 정확성 기준에 대해서 살펴본 후, MV2PL 프로토콜[5]을 기술한다. 또한, 실시간 보안 데이타베이스 시스템을 위한 보안 모델과 스케줄러가 가져야 할 특성들을 기술한다.

전통적인 데이타베이스 시스템을 위한 논리적 일관성 기준은 *직렬화 가능성(serializability)*이다. 임의의 두 트랜잭션 T_i와 T_j의 연산으로 이루어진 수행기록 H에서 T_i의 연산이 T_j의 연산을 모두 선행하거나 모두 후행하면 그 수행기록은 '*직렬(serial)이다*' 라고 *한다*. 또한, T_i와 T_j가 병행수행하여 생성된 수행기록 H'의 결과가 두 트랜잭션이 직렬로 수행된 결과들 - 예를 들면, T_iT_j 혹은 T_jT_i-중에 하나와 일치일 때, H'는 *직렬화 가능(serializable)하다*고 한다[6].

한편, 다중버전 병행수행 제어 프로토콜을 위한 정확성 기준은 단일-복사본(one-copy) 직렬화 가능성으로, 다중버전 수행기록인 H와 임의의 두 트랜잭션 T_i, T_j에 대해서, 연산 $r_i[x_j]$가 H에 존재할 때 – 즉, T_j가 기록한 x의 버전 x_j를 T_i가 판독하였을 때 – 항상 T_j가 T_i이거나 혹은 가장 최근에 x를 생성한 트랜잭션이라면 H는 *1-직렬*이라고 하고, 다중버전 수행기록 H가 *단일-복사본 직렬화 가능(one- copy serializable : 1SR)*하다는 것은 H가 특정한 1-직렬인 다중버전 수행기록 H'과 동치[2]일 때를 의미한다[6].

다중버전에 기초를 둔 프로토콜들 중에서 MV2PL 프로토콜은 판독-기록 연산간의 충돌과 기록-기록 연산간의 충돌을 제거하는 대신, 공인(certifying) 로크라는 새로운 로크를 정의하였다[5]. 따라서, 각 데이타 항목은 여러 개의 비공인(uncertified) 버전을 갖게 되지만, 단일-복사본 직렬화 가능성을 만족시키기 위해서 트랜잭션들은 항상 가장 최근에 공인된 데이타 항목들만을 판독한다. [5]에 프로토콜의 모든 성질이 기술되어져 있고 그 중에서 본 논문에서 참조하고자 하는 성질을 기술하면 다음과 같다. 성질에서 $r_i[x_j]$는 트랜잭션 T_i가 T_j에 의해 생성된 x의 버전 x_j를 판독함을 나타내고, $w_i[x_i]$는 T_i가 x의 새로운 버전인 x_i를 기록함을 나타내며 c_i란 T_i의 공인 연산을 의미한다.

【성질 1】 모든 트랜잭션 T_i에 대해서, T_i의 모든 판독연산 혹은 기록연산들은 공인연산을 선행하며 T_i의 공인연산은 완료되기 전에 수행된다.

【성질 2】 같은 등급 트랜잭션에 대해서, 수행기록 H에 $r_k[x_j]$ 연산과 $w_i[x_i]$ 연산이 포함되어 있다면, c_i가 $r_k[x_j]$ 연산을 선행하거나 후행한다. 즉, 판독연산은 x를 기록한 트랜잭션의 공인연산에 의해 순서화된다.

MV2PL 프로토콜은 표 1과 같이 어떤 트랜잭션이 데이타를 판독하고 있는 경우, 다른 트랜잭션들은 같은 데이타 항목에 대해서 공인 연산을 수행할 수 없다. MV2PL 프로토콜은 다단계 보안 데이타베이스 시스템

2) 임의의 다중버전 수행기록 H,H'에 대해서 두 수행기록이 모두 같은 연산들로 구성되어 있고 같은 기록-관계를 가질 때, H와 H'은 동치(equivalent)이다.

을 위한 프로토콜로 사용할 수 없다. 왜냐하면, 상위등급 트랜잭션이 어떤 하위등급 데이타 항목 x에 대해 판독연산을 수행하고 있을 때, 다른 하위등급 트랜잭션이 x에 대해 새로운 데이타 버전을 생성한 후 공인연산을 수행하려 한다면, 하위등급 트랜잭션의 공인연산은 상위등급 트랜잭션의 판독연산에 의해 지연될 수밖에 없고, 이로 인해 비밀채널이 발생할 수 있다. 따라서, 하위등급 트랜잭션이 지연되는 현상을 제거하기 위한 새로운 방법이 요구된다.

표 1 MV2PL 호환성표

T_R \ T_H	R	W	C
R	Y	Y	N
W	Y	Y	Y
C	N	Y	N

T_H : 로크 소유 트랜잭션
T_R : 로크 요구 트랜잭션
R : 판독연산
W : 기록연산
C : 공인연산
Y : 공유 가능
N : 공유 불허

본 논문에서 제안하는 병행수행 제어 프로토콜은 다음과 같은 벨-라파둘라(Bell-LaPadula : BLP) 모델[4]을 가정하고 있다.

(1) *판독연산을 위한 단순 특성(simple property)* : 모든 트랜잭션은 자신이 갖는 보안등급에 의해 지배(dominated)되는 데이타 항목만을 판독할 수 있다. 즉, 트랜잭션 T의 보안등급을 L(T)라 하고 데이타 항목 x의 보안등급을 L(x)라고 하면, L(T)≥L(x) 일 때, T는 x를 판독할 수 있다.

(2) *기록연산을 위한 제한된 ★-특성(restricted star property)* : 모든 트랜잭션은 자신이 갖는 보안등급과 같은 등급을 갖는 데이타 항목에 대해서만 기록연산을 수행할 수 있다. 즉, L(T)=L(x) 일 때, T는 x를 기록할 수 있다.

한편, 스케줄러가 다음과 같은 성질을 만족할 때, 그 스케줄러를 *보안(secure)* 스케줄러라 정의한다[10].

(1) *값 보안(value security)* : 스케줄러에 의해 생성된 임의의 수행기록을 구성하는 모든 트랜잭션들에 대해서, 만일 트랜잭션이 판독연산을 수행한 경우, 판독한 데이타의 값은 자신보다 상위등급을 가진 트랜잭션들에 의해 영향 받지 않으면, 그 스케줄러는 값 보안 성질을 만족한다.

(2) *지연 보안(delay security)* : 스케줄러에 의해 생성된 임의의 수행기록을 구성하는 모든 트랜잭션들에 대해서, 각 트랜잭션의 연산이 자신보다 상위등급을 가진 트랜잭션의 연산에 의해 지연되지 않으면, 그 스케줄러는 지연 보안 성질을 만족한다.

(3) *회복 보안(recovery security)* : 임의의 스케줄을 구성하는 모든 트랜잭션들에 대해서, 트랜잭션간에 교착상태가 발생하였다면, 상위등급 트랜잭션에 상관없이 발생된 것이어야 하고, 교착상태에서 회복하기 위해 취해진 행동도 상위등급의 트랜잭션에 의해 영향 받지 않아야 한다. 이와 같은 성질을 만족하면, 스케줄러는 회복 보안 성질을 만족한다.

3. 얼림 기법을 이용한 프로토콜

이 절에서는 프로토콜을 제안하기 전에 시간 요구사항과 보안 요구사항간에 발생하는 충돌을 살펴본 후, 두 개의 새로운 프로토콜들을 정의하고 몇 가지 예제 수행기록을 통해 프로토콜들의 특징을 기술한다.

표 2 임의의 두 트랜잭션 T_i와 T_j간의 가능한 우선순위와 보안등급 관계

트랜잭션 / 경우	트랜잭션 T_i		트랜잭션 T_j	
	우선순위	보안등급	우선순위	보안등급
경우 1	P_{Eq}	L_{Eq}	P_{Eq}	L_{Eq}
경우 2	P_{Eq}	L_{Low}	P_{Eq}	L_{High}
경우 3	P_{Eq}	L_{High}	P_{Eq}	L_{Low}
경우 4	P_{High}	L_{Eq}	P_{Low}	L_{Eq}
경우 5	P_{High}	L_{Low}	P_{Low}	L_{High}
경우 6	P_{High}	L_{High}	P_{Low}	L_{Low}

표 2는 실시간 보안 데이타베이스 시스템에서 임의의 두 트랜잭션간에 데이타 항목의 공유로 인한 충돌이 발생하게 되었을 때, 두 트랜잭션이 가질 수 있는 우선순위와 보안등급의 조합을 나타낸다. P_{Eq}, P_{High}, P_{Low}는 모두 우선순위를 나타내며, P_{High} > P_{Low}이다. L_{Eq}, L_{High}, L_{Low}는 보안등급으로 L_{High} > L_{Low}이다. 경우 1은 T_i와 T_j가 모두 같은 우선순위와 같은 보안등급을 갖는 경우이다. 따라서, 별도의 기법이 필요 없이 기존의 프로토콜로 충돌을 해결할 수 있다. 경우 2는 T_i와 T_j의 우선순위는 같으나 T_i보다 T_j의 보안등급이 높은 경우로 보안 요구사항을 만족시키기 위해서 낮은 등급의 트랜잭션인 T_i를 지연시키지 않고 트랜잭션을 스케줄하면 된다. 경우 3은 경우 2의 반대로 낮은 등급의 트랜잭션인 T_j를 지연시키지 않고 트랜잭션을 스케줄하면 된다.

경우 4는 T_i와 T_j가 같은 보안등급을 갖는 반면, T_i의 우선순위가 높으므로 우선순위 역전현상을 방지하기 위해서 T_i를 먼저 스케줄하면 된다. 경우 5의 경우에는 T_i가 높은 우선순위를 가지면서 낮은 보안등급을 갖기 때문에 시간 요구사항과 보안 요구사항을 모두 만족시키기 위해서는 T_i를 먼저 처리하면 된다. 마지막으로, 경우 6의 경우에는 T_i의 우선순위가 T_j보다 높아 T_i를 지연시켜서는 안되고, 또한 T_j의 보안등급이 T_i보다 낮기 때문에 비밀채널의 방지를 위해서 T_j도 T_i에 의해 방해받아서는 안된다. 즉, 경우 6에서 시간 요구사항과 보안 요구사항간에 충돌이 발생하게 되며 이와 같은 상황을 *SP(security and priority)*-충돌이라고 정의하겠다. SP-충돌을 해결하기 위해서 새로운 방법이 모색되어야 하며, 본 논문에서는 얼림 기법을 정의하여 SP-충돌을 해결한다.

3.1 B-Freeze 프로토콜

B-Freeze 프로토콜은 기본적으로 MV2PL 프로토콜[5]을 기반으로 하였고, 트랜잭션의 특성을 고려하지 않고 다음 정의와 같이 직접-얼림과 간접-얼림을 통해 SP-충돌을 해결한다. 임의의 한 트랜잭션 T가 판독하는 데이타 항목들의 집합을 RS(T)라 정의하고, 트랜잭션 관리자의 선 처리 작업에 의해 트랜잭션 시작 시, 모든 T는 RS(T)를 부여받는다고 가정한다. 그리고 트랜잭션 T의 보안등급은 L(T)로, 우선순위는 P(T)라고 한다.

【정의 1】 직접-얼림(Directly Frozen)

임의의 두 개의 활성 트랜잭션 T_i와 T_j가 같은 데이타 항목 x에 대해서 판독, 기록 로크를 각각 소유하고 있다고 가정한다. 만일, $L(T_i)>L(T_j)$, $P(T_i)>P(T_j)$ 이고 T_j는 T_i가 판독로크를 소유하고 있어도 공인로크를 획득하여 공인 연산을 수행하거나, 반대로 T_j가 공인로크를 소유하고 있어도 T_i가 판독로크를 획득하여 판독연산을 수행하면, T_i는 T_j에 의해 직접-얼림을 당한다고 한다. ■

【정의 2】 간접-얼림(Indirectly Frozen)

임의의 활성 트랜잭션 T_i는 다음과 같은 조건을 만족할 때 다른 활성 트랜잭션 T_j에 의해 간접-얼림을 당한다고 정의한다. 첫째, $L(T_i)\geq L(T_j)$, $P(T_i)>P(T_j)$이고 둘째, $RS(T_i)\cap RS(T_j)\neq\varnothing$이고 셋째, T_i는 직접-얼림 혹은 간접-얼림을 당한 트랜잭션이다. ■

【정의 3】 얼림점(Frozen points)

임의의 활성 트랜잭션 T_i의 얼림점 집합인 $fp(T_i)$는 (t, TID)들의 쌍으로 이루어진 집합이며, 이 때 t는 시간을 나타내고 TID는 트랜잭션의 식별자를 나타낸다. 만일, T_i가 T_j에 의해 직접-얼림을 당한 트랜잭션이라면, 시간 t는 T_j의 공인연산이 수행된 시간을 의미하고 TID는 0이 된다. 만일, T_i가 T_j에 의해 간접-얼림을 당한 트랜잭션이라면, T_j의 각 얼림점 (t, TID)에 대해서, T_i는 다음과 같은 얼림점들을 갖는다. $fp(T_i)=\{(t, u(TID))|(t, TID)\in fp(T_j)\}$. 여기서 함수 u는 얼림점 정의 함수로서 임의의 한 얼림점 (t, TID)에 대해서, 만일 $TID\neq 0$ 이면, $u(TID)=TID$ 이고 $TID=0$이면, $u(TID)=j$로 정의된다. ■

다음 그림 1의 수행기록 H_1의 트랜잭션 T_1, T_2, T_3, T_4에 대해서, $L(T_1)>L(T_2)>L(T_3)>L(T_4)$이고 $P(T_1)>P(T_2)>P(T_3)>P(T_4)$라 가정하자. 또한, 임의의 데이타 항목 d의 보안등급을 $L(d)$라고 할 때, $L(x)=L(T_4)$, $L(y)=L(T_3)$, $L(z)=L(T_2)$이라 하자.

시간 1에서 T_2는 하위 데이타 항목 x에 대해 판독로크를 획득하고, 시간 3에서 하위등급 트랜잭션인 T_4가 공인 연산을 수행한다. 따라서, T_2는 T_4에 의해 직접-얼림을 당하고, $fp(T_2) = \{(3, 0)\}$이 된다. 한편, $L(T_1) > L(T_2)$이고 $RS(T_1) \cap RS(T_2) \neq \varnothing$이며, T_2는 직접-얼림을 당한 트랜잭션이기 때문에, T_1은 T_2에 의해 간접-얼림을 당한다 ($fp(T_1) = \{(3, 2)\}$).

그림 2는 B-Freeze 프로토콜에 대한 호환성 표이다. B-Freeze 프로토콜의 규칙과 특성을 기술하면 다음과 같다. 우선, 모든 데이타 항목 x에 대해서, x가 공인될 때 타임스탬프 $ts(x)$를 부여받고, 모든 트랜잭션 T는 수행을 시작할 때, 그의 판독연산 집합인 RS(T)를 부여받는다.

【규칙 1】 같은 우선순위를 갖는 트랜잭션들에 대해서는 MV2PL 프로토콜 규칙을 적용한다. 하지만, 같은 보

트랜잭션 \ 시간	1	2	3	4	5	6	7	8	9	10	11	12
T_1								$r_1[y_0]$	$r_1[z_0]$	c_1		
T_2	$r_2[x_0]$						$r_2[y_0]$				$w_2[z_2]$	c_2
T_3				$r_3[x_4]$	$w_3[y_3]$	c_3						
T_4		$w_4[x_4]$	c_4									

그림 1 수행기록 H_1

T_R \ T_H	Read	Write	Certify
Read	Y	—	—
Write	Y	—	—
Certify	A	—	—

(a) T_H는 T_R보다 같거나 낮은 우선순위를 갖고, 높은 보안등급을 갖는다.

T_R \ T_H	Read	Write	Certify
Read	Y	—	—
Write	Y	—	—
Certify	Y*	—	—

(b) T_H는 T_R보다 높은 우선순위를 갖고, 높은 보안등급을 갖는다.

T_R \ T_H	Read	Write	Certify
Read	Y	Y	Y*
Write	—	—	—
Certify	—	—	—

(c) T_H는 T_R보다 낮은 우선순위를 갖고, 낮은 보안등급을 갖는다.

T_R \ T_H	Read	Write	Certify
Read	Y	Y	N
Write	—	—	—
Certify	—	—	—

(d) T_H는 T_R보다 같거나 높은 우선순위를 갖고, 낮은 보안등급을 갖는다.

T_H : 로크소유 트랜잭션 T_R : 로크요청 트랜잭션
Y : 허용 N : 불허 A : 취소(로크소유 트랜잭션의 취소) — : 불가능
Y* : 허용하나 높은 보안등급 트랜잭션이 낮은 보안등급 트랜잭션에 의해 얼림을 당한다. (SP-충돌)

그림 2 B-Freeze 프로토콜의 호환성 표

안등급을 가지나 상이한 우선순위를 갖는 트랜잭션들이 같은 데이타 항목에 대해 충돌을 하게 된다면, 우선순위 역전현상을 방지하기 위해 높은 우선순위를 갖는 트랜잭션을 우선 처리한다. 즉, 실시간 데이타베이스 시스템을 위한 프로토콜을 사용하면 된다. 한편, 같은 우선순위를 갖지만 상이한 보안등급을 갖는 트랜잭션들이 충돌을 하게 된다면, 비밀채널의 방지를 위해 하위등급을 갖는 트랜잭션을 지연시키지 않고 우선 처리를 한다. 즉, 다단계 보안 데이타베이스 시스템을 위한 프로토콜을 사용하면 된다.

【규칙 2】 임의의 트랜잭션 T가 하위등급 데이타 항목 x를 판독하려고 할 때, 만일 T보다 높은 우선순위를 갖는 트랜잭션 T_H가 x에 대해 공인로크를 가지고 있다면, T_H는 보안모델에 의해 T보다 낮은 보안등급을 가진다. 따라서, T는 시간요구사항과 보안요구사항을 모두 만족시키기 위해서 지연되고 T_H가 먼저 수행된다. 만일, T보다 낮은 우선순위를 갖는 트랜잭션 T_H가 x에 대해 공인로크를 가지고 있다면, SP-충돌이 발생하게 된다. 따라서, T와 T_H는 두 가지 요구사항을 만족시키기 위해서 지연되거나 취소될 수 없다. 즉, 다음과 같은 두 가지 연산이 수행되어진다. (i) T의 얼림점 집합인 fp(T)가 공집합이라면, T는 지연되지 않고 가장 최근에 공인된 x의 버전을 판독한다. 그리고 T는 T_H에 의해 직접-얼림을 당하고 (T_H의 공인시간, 0)이 fp(T)에 삽입된다. (ii) 공집합이 아니라면, T는 fp(T)에 있는 얼림점 (t, TID)들 중에서 가장 작은 시간 t를 갖는 얼림점에 따라서 t이전에 공인된 가장 최신에 x의 버전을 판독한다.

【규칙 3】 T가 데이타 항목 x에 대해서 공인연산을 수행하려 할 때, 만일 다른 트랜잭션 T_H가 x에 대해 판독로크를 가지고 있다면, 보안모델에 의해 T_H의 보안등급은 T와 같거나 높다. 만일, T와 T_H의 보안등급이 같다면, T는 두 트랜잭션의 우선순위에 의해 처리되며, T_H의 보안등급이 T보다 높다면, 우선순위를 비교한다. 만일, T의 우선순위가 T_H보다 높다면, 두 가지 요구사항을 만족시키기 위해 T_H가 취소되고 T는 공인연산을 수행한다. 하지만, T의 우선순위가 낮다면, SP-충돌이 발생하게 되고 비밀채널을 제거하기 위해 T_H를 취소시키는 것이 아니라, T가 x에 대해 공인연산을 수행하도록 로크를 허용하고, 대신 T_H는 T에 의해 직접-얼림을 당하게 한다. 즉, 상위등급 트랜잭션들은 얼림점을 갖게 된다.

【규칙 4】 T가 완료하려고 할 때, T에 의해 간접-얼림을 당한 활성 트랜잭션이 존재한다면, 그 활성 트랜잭션의 얼림점 집합에서 TID가 T인 얼림점을 제거한다.

다음은 하위등급 트랜잭션 혹은 같은 등급 트랜잭션에 의해 임의의 트랜잭션 T가 어떤 방식으로 얼림점을 부여받는지에 관한 규칙이다.

【규칙 5】 T가 수행을 시작하기 전에 T와 보안등급이 같거나 낮으면서 같은 데이타 항목을 판독하고 있고 직접 혹은 간접-얼림을 당한 트랜잭션이 존재한다면, 그 트랜잭션의 얼림점 집합을 모두 상속받는다. 상속받는

규칙은 정의 3과 같다. 만일, 이와 같은 트랜잭션이 존재하지 않는다면, T는 공집합인 얼림점 집합을 갖는다.

【규칙 6】 T가 수행중일 때, 시간 t_{now}에 어떤 트랜잭션 T_i에 의해 직접-얼림을 당하게 되면, T는 (t_{now}, 0)이라는 새로운 얼림점을 가지게 된다.

【규칙 7】 T가 수행중일 때, 시간 t_{now}에 어떤 다른 트랜잭션 T_i가 T_j에 의해 직접-얼림 혹은 간접-얼림을 당했을 때, $L(T) \geq L(T_i)$이고 $RS(T) \cap RS(Ti) \neq \varnothing$이면, T는($t_{now}$, i) 라는 새로운 얼림점을 갖게 된다.

위에 정의한 규칙에 의해 제안한 프로토콜은 같은 등급의 트랜잭션들에 대해서 MV2PL 프로토콜의 특성과 같은 특성을 갖는다. 2절에서의 성질 1과 성질 2 이외에 추가적으로 다음과 같은 성질을 갖는다.

【성질 3】 임의의 수행기록 H에 있는 모든 판독연산 $r_k[x_j]$에 대해서, 만일 j≠k라면 c_j는 $r_k[x_j]$를 선행한다. 그렇지 않다면, $w_k[x_k]$는 $r_k[x_k]$를 선행하고 BLP 모델에 의해 $L(x) = L(T_k)$이다.

【성질 4】 H에 있는 모든 $r_k[x_j]$와 $w_i[x_i]$에 대해서(i, j, k는 서로 상이함), 만일 $c_i < r_k[x_j]$이고 $L(T_i)=L(T_k)$이면, $c_i < c_j$이다. 즉, 같은 등급의 데이타를 판독하는 트랜잭션은 자신이 기록한 데이타를 판독하거나 가장 최근에 공인된 데이타 버전을 판독한다.

【성질 5】 H에 속한 모든 $r_k[x_j]$에 대해서, 만일 j≠k라면, c_j는 $r_k[x_j]$를 선행하고 $L(T_k) \geq L(x)$이다. 그렇지 않다면, $w_k[x_k]$는 $r_k[x_k]$를 선행하며 $L(x)=L(T_k)$이다. 즉, 모든 트랜잭션은 공인된 데이타 버전을 판독한다.

【성질 6】 모든 $w_i[x_i]$와 $w_j[x_j]$ 연산에 대해서, c_i는 c_j를 선행하거나 c_j는 c_i를 선행한다. 만일, $L(T_k)=L(T_i)$라면, MV2PL의 가장 최근에 공인된 버전을 판독한다는 성질로부터 쉽게 증명된다. 한편, $L(T_k)<L(T_i)$과 같은 경우는 BLP 모델에 의해 발생되지 않는다. $L(T_k)>L(T_i)$ 라면, 비밀채널의 방지를 위해 하위등급 트랜잭션 T_i는 T_k가 판독로크를 소유하고 있더라도 공인연산을 수행한다. 하지만 T_k가 직접-얼림을 당한 트랜잭션이므로 T_k는 T_i가 공인연산을 수행한 시간 이후에 생성된 데이타 항목들을 판독하지 않는다. 게다가 보안등급이 다르므로 T_i와 T_k간의 공인연산 충돌은 존재하지 않는다. 따라서, T_k는 직렬화 순서상 T_i를 선행한다.

B-Freeze 프로토콜에 의해서 생성될 수 있는 수행기록의 한 예를 살펴보자. 그림 3의 수행기록 H_2의 트랜잭션 T_1, T_2, T_3, T_4, T_5에 대해 $L(T_5)<L(T_4)<L(T_3)<L(T_2)<L(T_1)$이고, $P(T_5)<P(T_4)<P(T_3)<P(T_2)<P(T_1)$이며 $L(x)=L(T_4)$, $L(y)=L(T_5)$, $L(z)=L(T_3)$이라 하자. 이 수행기록은 MV2PL프로토콜에 의해서 생성될 수 없다. 하지만, 얼림 기법을 사용하면 가능한데, 시점 3에서 $fp(T_3)=\{(3, 0)\}$이 되고, T_2가 T_3에 의해 간접-얼림을 당하기 때문에 시점 4에서 T_2는 $fp(T_2)=\{(3, 3)\}$으로 시작한다. 시점 7에서 T_2와 T_3은 T_5에 의해서 직접-얼림을 당하므로 $fp(T_3)=\{(3, 0), (7, 0)\}$이 된다. 그리고 $fp(T_2)=\{(3, 3), (7, 0)\}$이 된다. T_1이 시작하는 시점 8에서, $RS(T_1) \cap RS(T_3) \neq \varnothing$이므로 T_1은 T_3에 의해 영향을 받는다. 따라서, $fp(T_1)=\{(3, 3), (7, 2)\}$가 되고(정의 3에 의해서), T_1은 x_0를 판독하게 된다. 시점 11에서 T_3이 완료되면, $fp(T_1)=\{(7, 2)\}$로 된다. 따라서, T_1은 z_3을 판독하게 되고 수행기록은 $T_2T_3T_1T_4T_5$로 수행된 결과와 동치이다.

3.2 R-Freeze 프로토콜

판독전용 트랜잭션은 어떠한 데이타 항목에 대해서도 오직 판독 연산만을 수행하고, 결코 갱신 연산을 수행하지 않는 트랜잭션이다[6]. 실시간 응용 분야에서, 판독전용 트랜잭션은 레이더 추적 시스템에서 비행기의 움직임을 추적하거나, 주식 거래 시스템에서 주가의 변동을 모니터 하는데 사용된다. 판독전용 트랜잭션은 일반적인 병행수행 제어 프로토콜로 처리하여 직렬화 가능성을 보장할 수 있기 때문에, 예전에는 실시간 응용에서 판독전용 트랜잭션에 대한 별다른 처리를 하지 않았다. 그러나 판독전용 트랜잭션은 일반적으로 많은 양의 데이타 항목에 대해서 장기간 수행되는 특성을 가지기 때문에, 일반적인 병행수행 제어 프로토콜을 사용할 경우, 갱신 트랜잭션의 수행을 상당히 오랜 기간 동안 지연시키거나 갱신 트랜잭션에 의해 쉽게 취소될 수가 있다[6]. 따라서, 판독전용 트랜잭션을 갱신 트랜잭션과 구분하여 처리하게 되면, 데이타 충돌을 감소시켜 시스템

시간 / 트랜잭션	1	2	3	4	5	6	7	8	9	10	11	12	13
T_1								$r_1[x_0]$				$r_1[z_3]$	c_1
T_2				$r_2[y_0]$					c_2				
T_3		$r_3[x_0]$				$r_3[y_0]$				$w_3[z_3]$	c_3		
T_4	$w_4[x_4]$		c_4										
T_5					$w_5[y_5]$		c_5						

그림 3 예제 수행기록 H_2

의 병행수행 정도를 높여주어 여러 연구에서 판독전용 트랜잭션을 효과적으로 처리할 수 있는 병행수행 제어 프로토콜을 소개하였다[17][24].

실시간 보안 데이타베이스에서도 판독전용 트랜잭션의 수행이 존재하는데, 지금까지 제안되었던 병행수행 제어 프로토콜들은 보안 요구사항과 실시간 요구사항을 만족시키는데 주된 목적을 두었기 때문에, 판독전용 트랜잭션에 대한 특별한 처리를 하지 않고 있다. 이 절에서는 판독전용 트랜잭션의 수행을 고려한 새로운 병행수행 제어 프로토콜을 제안한다. 제안하는 프로토콜은 B-Freeze 프로토콜을 기초로 하고 있다.

3.2.1 판독전용 트랜잭션에 대한 B-Freeze 프로토콜의 문제점

본 논문에서는 실시간 보안 환경에서 병행수행 제어 프로토콜로 B-Freeze 프로토콜을 사용했을 경우, 판독전용 트랜잭션과 관련되어 발생될 수 있는 문제점을 크게 다음의 두 가지로 지적한다. 첫째, 보안 등급이 높고 우선순위가 낮은 판독전용 트랜잭션은 갱신 트랜잭션에 의해 자주 취소되어 종료시한 내에 수행이 완료되기가 어렵다. 둘째, 판독전용 트랜잭션으로 인한 불필요한 간접-얼림은 트랜잭션의 판독 데이타 항목에 대한 최근성 정도를 떨어뜨린다. 다음은 이러한 두 가지 문제가 발생되는 상황을 나타낸다.

문제점 1 : 판독전용 트랜잭션에 대한 잦은 취소 문제

첫번째 문제는 B-Freeze 프로토콜에서는 판독전용 트랜잭션이 취소될 확률이 매우 높다는 것이다. 그림 4의 수행기록 H_3을 보면, T_1의 우선순위는 T_2보다 낮고 보안등급은 T_2보다 높을 때, T_1은 시간 4에서 T_2와 데이타 충돌을 발생시키기 때문에 취소되어 재시작된다. 그런데, 판독전용 트랜잭션은 그 특성상 다량의 데이타 항목에 대해 장기간 판독하기 때문에, 수행기록 H_3과 같이 우선순위가 높고 보안등급이 낮은 갱신 트랜잭션에 의해 데이타 충돌이 발생하여 취소될 확률이 매우 높다. 결과적으로, 데이타 충돌이 많은 상황에서는 판독전용 트랜잭션의 수행을 주어진 종료시한 내에 완료하기가 더욱 힘들어진다. 이러한 문제가 발생되는 근본적인 원인은 B-Freeze 프로토콜에서 SP-충돌인 상황에서만 직접-얼림을 사용하기 때문이다. 즉, 표 2의 경우 5에서 높은 보안등급과 낮은 우선순위를 가진 로크 소유 트랜잭션인 T_H가 판독전용 트랜잭션일 때, T_H를 취소시키지 않고 직접-얼림을 시킨다면 보다 많은 판독전용 트랜잭션들이 종료시한 내에 완료될 수 있을 것이며, 이와 같은 상황을 ***직접-얼림의 추가상황***이라 정의한다.

시간 / 트랜잭션	1	2	3	4
T_1	$r_1[x_0]$	$r_1[y_0]$		*abort*
T_2			$w_2[x_2]$	c_2

그림 4 수행기록 H_3

문제점 2 : 판독전용 트랜잭션에 의한 불필요한 간접-얼림 문제

그림 5의 수행기록 H_4에서 $L(T_1)>L(T_2)>L(T_3)>L(T_4)$이고 $P(T_1)>P(T_2)>P(T_3)>P(T_4)$이라 가정한다. 또, $L(x)=L(T_4)$이고 $L(y)=L(T_3)$이다. 시간 8에서 T_1은 T_2에 의해 간접-얼림을 당한다. 따라서, T_1의 판독 데이타 항목은 시간 3 이전에 공인된 버전이어야 한다. 하지만, 사실상 T_1은 시간 8 이전에 공인된 보다 최근의 버전을 판독해도 된다. 왜냐하면, T_1을 간접-얼림시키는 트랜잭션 T_2가 데이타 항목에 대해서 판독 연산만을 수행하기 때문에 T_1와 T_2사이의 수행 순서를 T_2T_1로 보장할 수 있기 때문이다. 이와 같이, 판독전용 트랜잭션은 다른 활성 트랜잭션에 대해서 불필요하게 간접-얼림을 당하여, 결과적으로 트랜잭션들이 불필요하게 구 버전의 데이타를 판독하게 한다. 즉, T_2가 판독전용 트랜잭션이라면 T_1은 T_2에 의해서 간접-얼림을 당하지 않고 최신의 데이타를 판독할 수 있다. 이와 같은 상황을 고려한 것이 ***간접-얼림의 예외상황***이다. 따라서, 판독전용 트랜잭션에 의해 다른 판독전용 트랜잭션들이 간접-얼림을 당하지 않는다고 간접-얼림의 예외상황을 정의한다.

3.2.2 R-Freeze의 규칙 및 성질

이 절에서는 R-Freeze에 대해 소개한다. 트랜잭션 T가 판독하는 데이타 항목들의 집합을 RS(T), 기록하는 데이타 항목들의 집합을 WS(T)라 정의하고, 트랜잭션 관리자의 선 처리 작업에 의해 트랜잭션 시작 시 모든 T는

시간 / 트랜잭션	1	2	3	4	5	6	7	8	9	10	11	12
T_1								$r_1[y_0]$	$r_1[z_0]$			c_1
T_2	$r_2[x_0]$						$r_2[y_0]$			$r_2[z_0]$	c_2	
T_3				$r_3[x_4]$	$w_3[y_3]$	c_3						
T_4		$w_4[x_4]$	c_4									

그림 5 수행기록 H_4

RS(T)과 WS(T)를 부여받는다고 가정한다. 트랜잭션 T의 보안등급은 L(T), 우선순위는 P(T)라고 가정한다. 제안하는 프로토콜이 B-Freeze 프로토콜을 기반으로 하였기 때문에 R-Freeze에 관하여 아래에 나오는 정의의 일부분은 B-Freeze 프로토콜의 얼림 기법에서 인용하였다.

【정의 4】 임의의 트랜잭션 T에 대해서, WS(T)가 공집합이면 T는 **판독전용 트랜잭션**이라고 정의한다.■

【정의 5】 임의의 두 활성 트랜잭션 T_i와 T_j가 같은 데이타 항목 x에 대해서 판독, 기록 로크를 각각 소유하고 있다고 가정한다.

경우 1 : 만일, $L(T_i)>L(T_j)$, $P(T_i)>P(T_j)$이고 T_j는 T_i가 판독로크를 소유하고 있어도 공인로크를 획득하여 공인연산을 수행하거나 T_i는 T_j가 공인로크를 소유하고 있어도 판독연산을 수행한다면, T_i는 T_j에 의해 **직접-얼림**을 당한다고 정의한다.

경우 2 : 만일, $L(T_i)>L(T_j)$, $P(T_i)<P(T_j)$이고 트랜잭션 T_i가 판독전용 트랜잭션일 때, T_j는 T_i가 판독로크를 소유하고 있어도 공인로크를 획득하여 공인 연산을 수행한다면, T_i는 T_j에 의해 **직접-얼림**을 당한다고 정의한다.(이와 같은 상황을 **직접-얼림의 추가상황** 이라고 정의한다.) ■

【정의 6】 임의의 두 활성 트랜잭션 T_i와 T_j에 대해서, T_i는 다음과 같은 조건을 만족할 때 다른 트랜잭션 T_j에 의해 간접-얼림을 당한다고 정의한다. (i) $L(T_i) \geq L(T_j)$, $P(T_i) > P(T_j)$ 이고 (ii) $RS(T_i) \cap RS(T_j) \neq \varnothing$이고 (iii) T_j는 직접-얼림 혹은 간접-얼림을 당한 트랜잭션이고, (iv) T_i가 판독전용 트랜잭션일 때, T_j는 판독전용 트랜잭션이 아니다. (T_j가 판독전용 트랜잭션일 때, 이와 같은 상황을 **간접-얼림의 예외상황** 이라고 정의한다.) ■

다음은 판독전용 트랜잭션을 고려함으로 인해 발생하게 되는 B-Freeze 프로토콜 규칙 이외의 추가적인 규칙이다.

【규칙 8】 트랜잭션 T의 수행 중에 만일, T_i가 T_j에 의해서 직접-얼림을 당하고, $L(T) \geq L(T_i)$이고 $RS(T) \cap RS(T_i) \neq \varnothing$이고 T_i가 판독전용 트랜잭션이 아니

```
/* R, W, 와 C를 각각 판독, 기록, 공인연산이라고 하자. */

case 1: 임의의 트랜잭션 Tj가 데이타 항목 x에 대해서 판독연산을 요청
    if ( 로크 소유 트랜잭션 Tj가 x에 공인로크를 가짐 ) then
        if ( Tj 가 Ti보다 높거나 같은 우선순위를 가짐 ) then
            if ( Tj가 Ti보다 낮거나 같은 보안등급을 가짐 ) then
                로크 요청 트랜잭션 Ti를 지연시킨다 ;
            end if
        else
            /* 로크 요청 트랜잭션은 로크 소유 트랜잭션에 의해 직접-얼림 당한다. */
            얼림점(현재시간 tnow, 0)을 로크 요청 트랜잭션인 Ti의 얼림점 집합에 추가한다 ;
        end if
    end if
    Ti에게 판독로크를 승인한다 ;

case 2: 임의의 트랜잭션 Tj가 데이타 항목 x에 대해서 기록연산을 요청
    /* 기록연산은 다른연산과 충돌관계를 갖지 않는다.*/
    Ti에게 기록로크를 승인한다 ;

case 3: 임의의 트랜잭션 Tj가 데이타 항목 x에 대해서 공인연산을 요청
    if (같은 등급의 트랜잭션들 중에서 x에 대해 판독 혹은 공인 로크를 획득하고 있는
        트랜잭션이 존재하지 않음) then
        Ti에게 공인로크를 승인한다 ;
    else /* 판독이나 공인 로크를 가지고 있는 트랜잭션이 존재한다. */
        if (판독로크를 가진 트랜잭션 reader들이 존재함) then
            if (Ti가 reader들 보다 높은 우선순위를 가짐) then
                    if (reader들이 판독전용트랜잭션임) then
                        reader들을 얼림시킨다 ;
                        Ti에게 공인로크를 승인한다 ;
                    else
                        reader들을 취소한다 ;
                        Ti에게 공인로크를 승인한다 ;
                    else if
            else
                if (Ti의 보안등급이 낮음) then
                    활성중인 reader들을 얼림시킨다 ;
                    Ti에게 공인로크를 승인한다 ;
                else /* 보안등급이 같다. */
                    로크 요청 트랜잭션 Ti를 지연시킨다 ;
                end if
            end if
        else /* 공인로크 소유자인 Tj가 존재한다. */
            /* 논문에서 채택한 보안모델에 의해서, Ti와 Tj는 같은 보안등급 가진다. */
            if (Ti가 Tj보다 높은 우선순위를 가짐) then
                Tj의 공인로크를 기록로크로 전환한다 ;
                Ti에게 공인로크를 승인한다 ;
            else
                Ti를 Tj가 공인할 때까지 대기시킨다 ;
            end if
        end if
    end if
```

그림 6 R-Freeze 프로토콜의 의사코드

라면, (T_j의 공인시간, i)가 fp(T)에 삽입된다. 이것은 T_i가 T_j에 의해서 간접-얼림을 당하는 것을 의미한다. (규칙 7에서 간접-얼림의 예외상황의 보완)

【규칙 9】 T가 데이타 항목 x에 대해서 공인연산을 수행하려 할 때, 만일 다른 트랜잭션 T_H가 x에 대해 판독로크를 가지고 있다면, 보안모델에 의해 T_H의 보안등급은 T와 같거나 높다. 만일, T와 T_H의 보안등급이 같다면, T는 두 트랜잭션의 우선순위에 의해 처리되며, T_H의 보안등급이 T보다 높다면, 우선순위를 비교한다. 만일, T의 우선순위가 T_H보다 높고 T_H가 판독전용 트랜잭션이 아니라면, T_H가 취소되고 T는 공인연산을 수행한다. 하지만, T의 우선순위가 낮거나 T_H가 판독전용 트랜잭션이라면, T_H를 취소시키는 것이 아니라, T가 x에 대해 공인연산을 수행하도록 로크를 허용하고, 대신 T_H는 T에 의해 직접-얼림을 당하게 한다. (규칙 3에서 직접-얼림 추가상황의 보완)

그림 6은 제안하는 프로토콜을 의사코드로 표현한 것이다.

그림 7의 수행기록 H_5는 R-Freeze 프로토콜에 의해서 생성된 것이다. H_5에서 $L(T_1) > L(T_2) > L(T_3) > L(T_4)$, $P(T_1) > P(T_3) > P(T_4) > P(T_2)$ 이고 $L(x) = L(T_4)$, $L(y) = L(T_3)$, $L(z) = L(T_2)$이다. 그리고 각 데이타 d의 초기버전은 d_0이다.

T_2는 판독전용 트랜잭션이기 때문에, 시간 3에서 T_4에 의해서 취소되지 않고 직접-얼림을 당하여 그 수행을 계속한다(직접-얼림의 추가상황). 그 대신 이후의 하향판독은 시간 3이전에 공인된 버전을 판독해야 되므로 시간 7에서 데이타 항목 y의 y_0버전을 판독하게 된다. 시간 8에서 T_1은 얼림을 당한 트랜잭션 T_2가 판독전용 트랜잭션이고, 자신도 판독전용 트랜잭션이기 때문에 간접-얼림을 당하지 않는다(간접-얼림의 예외상황). 따라서 데이타 항목 y의 최근 버전인 y_3을 판독하게 된다.

4. 정확성 증명

이 절에서는 제안하는 프로토콜이 단일-복사본 직렬화 가능성을 보장하고, 우선순위 역전을 발생시키지 않고, 비밀채널을 제거함을 증명한다.

4.1 직렬화 가능성 보장

일반적으로, 다중버전에 기반을 둔 병행수행 제어 프로토콜의 정확성을 증명하기 위해서 그 프로토콜이 생성하는 모든 수행기록은 단일-복사본 직렬화 가능성을 만족하여야 한다.

[정리 1]임의의 다중버전 수행기록 H의 MVSG (H, ≪)가 비순환적이라면, H는 단일-복사본 직렬화 가능하다[6]. ■

임의의 데이타 항목 x에 대한 버전순서 ≪는 x의 완료된 버전들의 전역 순서화(total order)를 나타낸다. 임의의 다중버전 수행기록 H와 H에 대한 버전순서 ≪가 주어졌을 때, 다중버전 직렬화 그래프 MVSG(H, ≪)는 H에 있는 완료된 트랜잭션들을 노드로 갖고 다음과 같이 두 가지 타입의 에지(충돌 에지와 버전순서 에지)를 갖는 그래프이다 : (i) T_i에서 T_j로의 충돌 에지는 H의 프로젝션 C(H)가 특정 데이타 항목 x에 대해 $w_i[x_i]$와 $r_j[x_i]$ 연산을 포함할 때 발생한다. (ii) 임의의 두 연산 $r_k[x_j]$와 $w_i[x_i]$ (이 때, i, j, k는 서로 상이함)에 대해, 만일, $x_i \ll x_j$일 때, $T_i \rightarrow T_j$로의 버전순서 에지가 발생한다. 그렇지 않으면, $T_k \rightarrow T_i$로의 버전순서 에지가 발생한다. ■

[정리 2] 제안한 프로토콜에 의해 생성된 모든 수행기록은 1SR이다.

증명 : $T=\{T_1, T_2, \ldots, T_n\}$을 트랜잭션의 집합이라고 할 때, T를 기반으로 제안된 프로토콜에 의해 생성된 수행기록을 H라고 하고 트랜잭션 T_j의 공인연산을 c_j라고 나타낸다. 그리고 버전순서 $x_i \ll x_j$는 $c_i < c_j$연산을 함축한다고 정의한다. 첫째, T_i가 선택하는 버전들을 살펴보았을 때, 만일 T_i가 얼림점을 갖지 않는다면, 버전을 선택하는 규칙은 MV2PL과 같이 가장 최근에 완료된 버전을 선택한다.

그렇지 않다면, T_i의 얼림점 정의에 의해서 T_i가 판독하는 버전은 $fp(T_i)$에 속하는 얼림점 중에서 가장 작은 공인 시간 t 이전에 생성된 것들이다. 이는 정의 3을 위배하는 트랜잭션이 존재하지 않음을 의미한다. 둘째,

시간 / 트랜잭션	1	2	3	4	5	6	7	8	9	10	11	12
T_1								$r_1[y_3]$	$r_1[z_0]$			c_1
T_2	$r_2[x_0]$						$r_2[y_0]$			$r_2[z_0]$	c_2	
T_3				$r_3[x_4]$	$w_3[y_3]$	c_3						
T_4		$w_4[x_4]$	c_4									

그림 7 수행기록 H_5

MVSG(H, ≪)에 속하는 모든 에지 $T_i \rightarrow T_j$에 대해서, 그들이 공인순서로 되어 있음을 보임으로써 MVSG(H, ≪)가 비순환적임을 증명한다. 즉, $T_i \rightarrow T_j$ 이면, $c_i < c_j$ 임을 보인다. 우선, $T_i \rightarrow T_j$ 가 직렬화 그래프 SG(H)[3] 에 속하는 에지라고 가정한다. 이 에지는 두 트랜잭션간의 판독관계(read-from relationship)를 나타낸다. 그러면, 모든 트랜잭션들은 항상 공인된 데이타 버전을 판독하기 때문에 $c_i < c_j$이다. 다음은 서로 상이한 i, j, k에 대해서 H의 연산 $w_i[x_i]$와 $w_j[x_j]$, $r_k[x_j]$를 고려해 본다. 그러면, 두 가지 경우가 가능하다: (i) $x_i \ll x_j$ 와 (ii) $x_j \ll x_i$ 이다. 첫 번째 경우는 MVSG(H,≪)에 $T_i \rightarrow T_j$ 버전순서 에지를 추가시킨다. 버전순서≪의 정의에 의해서 $c_i < c_j$이다. 두 번째 경우는 MVSG(H, ≪)에 $T_k \rightarrow T_i$ 로의 버전순서 에지를 추가시킨다. 그러면, [성질 4]에 의해서 $c_i < c_j$이거나 $c_k < c_i$이다. 첫 번째 경우는 버전순서 ≪의 정의에 의해서 $c_j < c_i$를 의미하기 때문에 불가능하므로 $c_k < c_i$이다. MVSG(H, ≪)에 속하는 모든 에지들이 공인연산 순서로 되어 있기 때문에 MVSG(H, ≪)는 비순환적이다. 따라서, H는 단일-복사본 직렬화 가능하다. ■

4.2 보안 요구사항 만족

[정리 3] 제안한 프로토콜은 상위등급 트랜잭션의 연산 때문에 하위등급 트랜잭션의 연산을 결코 지연시키거나 취소시키지 않음으로써 비밀채널을 방지한다.

증명 : 본 논문에서 가정하고 있는 보안모델에 의해서 임의의 트랜잭션은 자신과 같은 등급의 데이타 항목 혹은 하위등급 데이타 항목만을 판독할 수 있으며, 같은 등급의 데이타 항목만을 기록할 수 있다. 임의의 두 트랜잭션 T_i와 T_j에 대해서 그들의 보안등급 $L(T_i)$와 $L(T_j)$가 $L(T_i) > L(T_j)$라고 가정한다. 만일 T_i와 T_j가 서로 충돌관계에 있는 트랜잭션이라면, 보안모델에 의해 Tj는 어떤 특정 데이타 항목 x에 대해서 공인연산을 수행하려 하고 T_i는 같은 데이타 항목 x에 대해서 하향 판독연산을 수행하려는 것이다. 그러면, 두 가지 경우가 가능하다. : (i) $P(T_i)<P(T_j)$와 (ii) $P(T_i)>P(T_j)$. (i) 경우에는, $L(T_i)>L(T_j)$이고 $P(T_i)<P(T_j)$ 이기 때문에, Tj는 보안 요구사항과 실시간 요구사항을 만족시키기 위해서 블록 될 수 없다. 그대신, T_i가 판독전용 트랜잭션인 경우 얼림을 당하고, 그렇지 않다면 T_j에 의해서 블록 된다. (ii)의 경우에는, $L(T_i)>L(T_j)$이고 $P(T_i)>P(T_j)$ 이기 때문에, SP-충돌이 발생한다. 비밀채널과 우선순위 역전을 방지하기 위해서, T_i와 T_j가 모두 블록되어서는 안된다. 그대신, T_i는 T_j에 의해 얼림을 당한다. 그러므로 T_j는 T_i에 의해서 결코 블록 되지 않는다. 따라서 제안된 프로토콜은 보안 요구사항을 만족한다.■

[정리 4] 제안한 프로토콜은 보안기준을 만족한다. 즉, 값 보안, 지연 보안, 회복 보안성질을 만족한다.

증명 : 제안한 프로토콜이 보안기준을 만족함을 보이려면, 낮은 보안등급을 갖는 트랜잭션 T가 높은 보안등급으로 갖는 임의의 상위등급 트랜잭션에 의해 지연되거나 취소되지 않고 수행결과가 변하지 않음을 증명하면 된다. 값 보안을 증명하기 위해 임의의 두 트랜잭션 T_i와 T_j에 대해서 T_i의 보안등급이 T_j의 보안등급보다 높다고 가정한다. 우선, 본 논문이 가정하고 있는 보안정책인 BLP모델에 의해 T_i는 T_j가 판독하는 값을 기록할 수 없으므로 영향을 미칠 수 없고 T_i의 얼림점은 [정의 1]과 [정의 2]에 의해서 T_j의 버전 선택에 영향을 미칠 수 없다. 따라서, T_j는 임의의 상위등급 트랜잭션인 T_i가 존재하는지의 유무에 상관없이 버전을 선택하게 되므로 값 보안 성질을 만족한다. 한편, [정리 3]에서 이미 제안한 프로토콜이 지연 보안 성질을 만족함을 증명하였다. 마지막으로, 같은 등급 트랜잭션간의 연산은 로킹기법을 사용하였지만 서로 등급이 다른 트랜잭션간은 하향 판독연산만이 가능하고 이는 타임스탬프 기법의 변형인 개선된 얼림 기법을 이용하여 버전을 선택하고 있기 때문에, 상위등급 트랜잭션에 의해서는 교착상태가 발생하지 않는다. 따라서, 제안한 프로토콜은 회복보안 성질도 만족한다. ■

4.3 실시간 요구사항 만족

[정리 5] 제안한 프로토콜은 상위 우선순위 트랜잭션을 하위 우선순위 트랜잭션에 의해 지연시키거나 블록시키지 않으므로, 우선순위 역전이 발생하지 않는다.

증명 : 충돌관계에 있는 임의의 두 트랜잭션 T_i와 T_j에 대해서 그들의 우선순위 $P(T_i)$와 $P(T_j)$가 $P(T_i) > P(T_j)$라고 가정한다. 그러면, 두 가지 경우가 가능하다. 첫 번째 경우는 $L(T_i) < L(T_j)$인 경우이다. T_i와 T_j가 모두 동일한 데이타 항목 x에 대해서 접근하고자 한다면, 보안모델에 의해서 T_i가 공인연산을 수행하려 할 때 T_j는 판독연산을 하게 되는 경우이다. T_j의 보안등급과 T_i의 우선순위를 고려하여, 비밀채널과 우선순위를 방지하기 위해서 T_j를 블록시킨다. 물론, T_j가 판독전용 트랜잭션인 경우, Tj를 블록 시키지 않고 직접-얼림 시켜 구 버전의 데이타 항목을 판독하게 하고 T_i는 블록

3) 임의의 수행기록 H에 대한 직렬화그래프 SG(H)는 트랜잭션들을 노드로 갖고 데이타 항목에 대한 충돌을 갖는 두 트랜잭션의 관계를 에지로 표현한 방향성 그래프이다.

되지 않는다. 두 번째 경우는 $L(T_i) > L(T_j)$인 경우이다. 보안 모델에 의해서 T_j가 공인연산을 수행하려 할 때 T_i가 판독연산을 하게 되는 경우로, SP-충돌을 발생시킨다. 이때는, T_i가 T_j에 의해 얼림을 당하게 하므로, 상위 우선순위 트랜잭션 T_i는 블록 되지 않는다. 대신 T_i는 T_j가 공인연산을 수행한 시간 이전에 공인된 구 버전의 데이타 항목을 판독하게 된다. 이와 같이, 모든 경우에 대해서 제안된 프로토콜은 실시간 요구사항을 만족한다. ■

5. 성능 평가

본 논문에서 제안한 R-Freeze 프로토콜은 판독전용 트랜잭션과 갱신 트랜잭션 사이의 충돌을 해결하여 보다 많은 판독전용 트랜잭션들이 주어진 종료시한 내에 수행을 마칠 수 있도록 한다. 따라서 R-Freeze 프로토콜은 판독전용 트랜잭션의 처리 성능과 전체 트랜잭션의 처리 성능 측면이 모두 평가되어져야 한다. 평가의 대상은 R-Freeze, B-Freeze, SRT-2PL 프로토콜이고, 평가의 기준은 트랜잭션의 재시작 비율과 평균 처리시간, 그리고 종료시한-위반 비율이다.

5.1 시뮬레이션 모델

제안한 프로토콜의 성능을 평가하기 위해서, 본 논문에서는 시뮬레이션 툴로 AweSim[2]을 사용하였다. 그림 8과 같이 트랜잭션 생성부에서 트랜잭션이 생성되면 트랜잭션 스케줄러에 제출되고 트랜잭션 스케줄러는 다음의 공식에 의해 종료시한과 우선순위를 부여한다. 본 논문에서는 각 트랜잭션에 대해서 소프트 종료시한을 가정하기 때문에, 트랜잭션이 종료시한을 위반하더라도 취소되지 않는다.

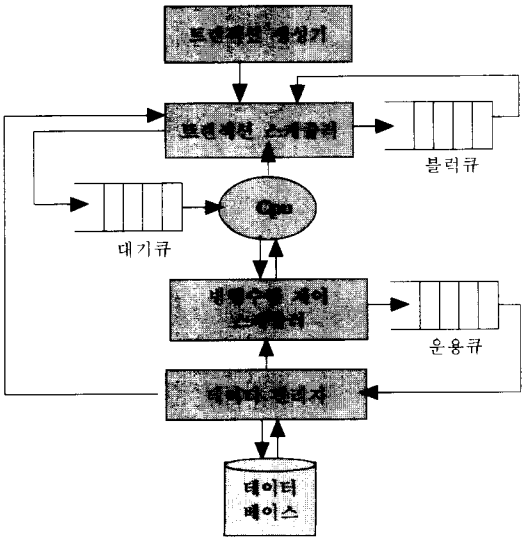

그림 8 시뮬레이션 모델

표 3에 나와 있는 시뮬레이션에서 사용되는 파라미터의 초기값은 [13]에 나와있는 값에 기초하여 설정하였다.

표 3 시뮬레이션 파라미터

파라미터	값
데이타베이스에 존재하는 데이타 항목의 수	100
수행되는 트랜잭션의 수	50
트랜잭션 수행에 부여되는 여유시간	10
보안등급 수	4
트랜잭션이 가지는 연산의 수	30~50
데이타 항목의 접근에 대한 I/O 시간	25ms
연산이 CPU에 머무는 시간	10ms
트랜잭션이 재수행될 때, 지연되는 시간	10ms

【공식 1】 $DL(T) = AT(T) - Slack * N(T) * (CPUTIME + DISKIOTIME)$

【공식 2】 $P(T) = DL(T) * 100$

【공식 3】 *재시작 비율* = $\frac{\text{재시작 트랜잭션의 수}}{\text{전체 트랜잭션의 수}}$

【공식 4】 *평균 처리시간*=

$$\frac{\sum_{i=1}^{N} \text{Finish Time}(T_i) - \text{Start Time}(T_i))}{N}$$

N은 수행된 전체 트랜잭션의 수, StartTime(T_i)와 FinishTime(T_i)는 트랜잭션 T_i의 시작 시간과 종료 시간을 각각 나타낸다.

【공식 5】 *종료시한-위반 비율*=100*

$$\frac{\text{늦어진(tardy) 트랜잭션의 수}}{\text{수행된 전체 트랜잭션의 수}}$$

여기서 DL(T)는 트랜잭션 T의 종료시한을 의미하고, AT(T)는 T가 도착한 시간을 나타낸다. 또, Slack은 실시간 데이타베이스 시스템에서 각 트랜잭션의 종료시한에 여유를 주기 위해서 사용된다. N(T)는 트랜잭션에 포함된 판독 또는 기록 연산의 수이며, CPUTIME은 각각의 판독이나 기록 연산이 CPU에 머무는 시간이고, DISKIOTIME은 디스크 I/O 시간이다. 마지막으로 P(T)는 트랜잭션 T의 우선순위를 의미하며 큰 값을 가질수록 우선순위는 낮아진다. 재수행 되는 트랜잭션의

수를 보이기 위해서 공식 3을 사용하였고 트랜잭션당 평균 처리시간을 계산하기 위해서 공식 4를 사용하였다. 그리고 자신의 종료시한을 위배하는 트랜잭션의 비율을 종료시한-위반 비율이라 정의하고 이를 계산하기 위해서 공식 5를 사용하였다.

5.2 평가 분석

그림 9는 트랜잭션의 도착 간격시간이 변화함에 따라, 판독전용 트랜잭션의 재시작 비율이 어떻게 변하는지를 보여 주고 있다. 도착 간격시간이 적어지면 트랜잭션 사이의 데이타 충돌이 발생할 확률이 높아지기 때문에, 도착 간격시간이 커짐에 따라 전체적으로 재시작 비율이 낮아지는 하향곡선을 그리게 된다. 트랜잭션의 도착 간격시간이 적어질수록, SRT-2PL과 B-Freeze 프로토콜은 판독전용 트랜잭션의 재시작 비율이 심각하게 높아진다. 하지만, 제안하는 프로토콜은 판독전용 트랜잭션을 결코 취소시키지 않기 때문에 트랜잭션의 도착 간격시간에 상관없이, 판독전용 트랜잭션의 재시작 비율은 항상 0이다. 따라서, 그림 10에서도 전체 트랜잭션의 재시작 비율은 제안한 프로토콜이 다른 프로토콜보다 낮은 재시작 비율을 가진다.

그림 11에서는 트랜잭션의 도착 간격시간이 변화함에 따라, 판독전용 트랜잭션의 처리시간이 어떻게 변하는지를 보여 주고 있다. 트랜잭션이 재시작 되지 않는다면, 그 만큼 빨리 처리되기 때문에 제안하는 프로토콜이 SRT-2PL과 B-Freeze 프로토콜보다 판독전용 트랜잭션을 빠르게 처리한다. 제안하는 프로토콜은 판독전용 트랜잭션을 재시작 시키지 않지만, 트랜잭션의 도착 간격시간이 작아지면 트랜잭션의 연산들이 그림 9의 대기큐에 머무르는 시간이 많아지기 때문에 도착 간격시간이 커짐에 따라 하향곡선을 그리게 된다. 판독전용 트랜잭션의 평균 처리시간은 전체 트랜잭션의 평균 처리시간에 영향을 끼치게 되어 그림 12와 같이 전체적으로도 제안하는 프로토콜이 나은 성능을 보이게 된다.

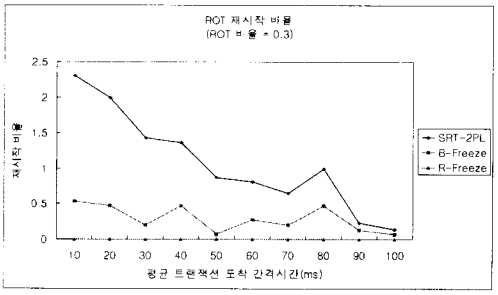

그림 9 ROT의 재시작 비율

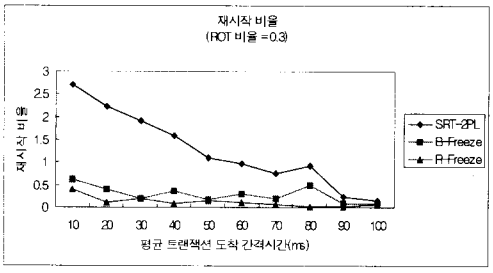

그림 10 트랜잭션 재시작 비율

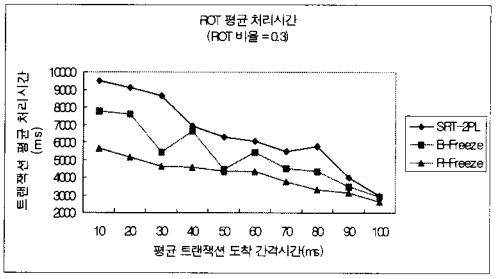

그림 11 ROT의 처리시간

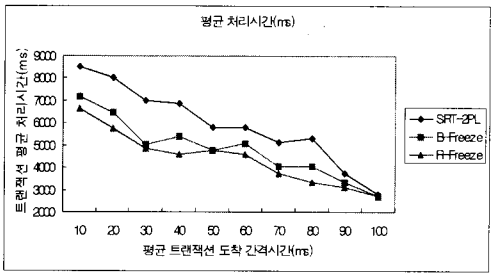

그림 12 트랜잭션의 처리시간

그림 13은 트랜잭션의 도착 간격시간이 변화함에 따라, 판독전용 트랜잭션의 재시작 비율이 어떻게 변하는지를 보여 주고 있다. 트랜잭션의 처리시간이 빨라지면, 그 만큼 자신의 종료시한 내에 수행을 완료하는 확률이 높아지기 때문에 제안하는 프로토콜이 SRT-2PL과 B-Freeze 프로토콜보다 낮은 종료시한-위반 비율을 가진다. 결과적으로 그림 14처럼 전체 트랜잭션의 종료시한-위반 비율도 제안하는 프로토콜이 비교 대상 프로토콜 보다 낮은 값을 가진다.

이상의 평가 분석에서 제안하는 프로토콜을 실시간

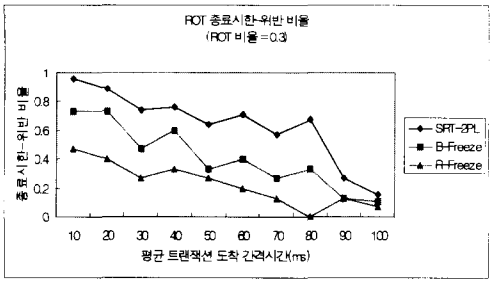

그림 13 ROT의 종료시한-위반 비율

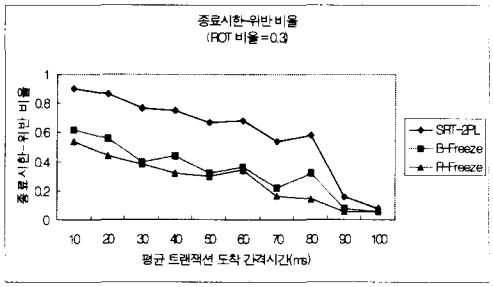

그림 14 트랜잭션의 종료시한-위반 비율

보안 데이타베이스 시스템의 병행수행 제어 프로토콜로 사용하게 되면, B-Freeze 프로토콜을 사용했을 때 보다, 판독전용 트랜잭션과 전체 트랜잭션에 대한 재시작 비율, 처리시간, 종료시한-위반 비율 측면에서 모두 나은 성능을 가진다는 것을 알 수 있다. 그러나 제안하는 프로토콜은 직접-얼림의 추가상황과 간접-얼림의 예외상황을 두기 위해서 각 트랜잭션별로 기록집합을 추가적으로 저장해야 하고, 병행수행 제어 시, 항상 트랜잭션이 판독전용 트랜잭션인지 검사를 해야 하는 추가비용이 든다. 만약, 판독전용 트랜잭션의 수행이 전체 트랜잭션 수행에 비해서 극히 미비하고, 판독전용 트랜잭션의 수행이 시스템에서 그렇게 큰 의미를 가지지 못하여 재시작 되어도 별다르게 해로운 점이 없는 환경이라면, 오히려 B-Freeze 프로토콜을 사용하는 것이 더 바람직할 수 있다. 하지만, 판독전용 트랜잭션이 전체 트랜잭션의 수행 중 상당 부분을 차지하고, 판독전용 트랜잭션의 수행을 시스템에서 매우 중요하게 여기는 환경에서, 제안하는 프로토콜은 B-Freeze 프로토콜보다 나은 성능을 보인다.

6. 결 론

판독전용 트랜잭션을 다른 갱신 트랜잭션과 구별하면 판독전용 트랜잭션을 더욱더 효과적으로 처리할 수 있다. 그러나 현재 실시간 보안 데이타베이스 시스템의 병행수행 프로토콜들은 데이타베이스의 논리적 일관성, 보안 요구사항, 시간적 제약사항을 만족시키는데 그 주된 목적을 두었고, 실제로 발생되는 트랜잭션의 특성에는 관심을 두지 않았다. 결과적으로, 이들 프로토콜을 사용하게 되면, 판독전용 트랜잭션과 갱신 트랜잭션을 구별하여 처리하지 않기 때문에 판독전용 트랜잭션의 수행을 보장하지 못하는 경우가 발생한다.

본 논문에서 제안한 프로토콜들은 얼림 기법을 이용하여 실시간 보안 데이타베이스 시스템의 시간 요구사항과 보안 요구사항을 모두 만족시키면서 기존의 다중버전에 기초를 둔 프로토콜에 비해 병행수행 정도를 향상시키는 새로운 프로토콜이다. 특히, R-Freeze 프로토콜은 판독전용 트랜잭션의 수행을 보장한다. R-Freeze는 B-Freeze 프로토콜의 얼림 기법을 개선한 것으로, 직접-얼림의 추가상황과 간접-얼림의 예외상황을 두어, 판독전용 트랜잭션이 갱신 트랜잭션에 의해서 재시작되지 않게 할 뿐만 아니라, 불필요한 간접-얼림을 당하지 않게 하여 판독하는 데이타의 최근성 정도를 높여주었다. 또한, 얼림 기법을 기반으로 하였기 때문에, 제안한 프로토콜은 직렬화 가능성을 만족함을 증명하였다.

성능 평가에서 드러난 바와 같이 본 논문에서 제안한 프로토콜은 판독전용 트랜잭션의 수행에 있어서 B-Freeze 프로토콜 보다 나은 결과를 보였고, 전체적인 트랜잭션의 수행 결과도 우수하였다. 본 논문에서 제안된 프로토콜은 판독전용 트랜잭션의 수행이 중요시되고, 판독하는 데이타의 수가 많고, 그 수행기간이 긴 환경일수록 기존의 프로토콜보다 더욱더 우수한 성능을 보일 것으로 기대된다. 본 논문에서는 병행수행 제어 프로토콜의 수준에서 제안된 프로토콜의 구현과 성능평가를 고려하였다. 제안한 프로토콜이 다중버전에 기초하였기 때문에, 앞으로는 다중버전을 효율적으로 관리하기 위한 버퍼 관리 기법과 쓰레기 수집 기법에 대한 연구가 필요하다.

참 고 문 헌

[1] Abott, R. K. and H. Garcia-Molina, "Scheduling Real-Time Transactions : A Performance Evaluation," *ACM Trans. on Database Systems*, Vol. 17 (No. 3), pp. 513-560, 1992.

[2] Alan, A., B. Pritsker, J. O'Reilly, and D. K. Laval, *Simulation with Visual SLAM and Awesim*, System Publishing Corporation, Indiana, 1997.

[3] Atluri, V., S. Jajodia, T. F. Keefe, C. McCollum,

and R. Mukkamala, "Multilevel Secure Transaction Processing : Status and Prospects," *Proceedings of IFIP WG11.3 10th Annual Working Conference on Database Security*, pp. 79-98, 1996.

[4] Bell, D. E. and L. J. LaPadula, *Secure Computer Systems : Mathematical Foundations*, Technical Report MTR-2997, The Mitre Corporation, Bedford, 1973.

[5] Bernstein, P. A. and N. Goodman, "Multiversion Concurrency Control - Theory and Algorithms," *ACM Transactions on Database Systems*, Vol. 8 (No. 4), pp. 465-483, 1983.

[6] Bernstein, P. A., V. Hadzilacos, and N. Goodman, *Concurrency Control and Recovery in Database Systems*, Addison-Wesley, 1987.

[7] Chaney, C. and S. Son, "Supporting the Requirements for Multilevel Secure and Real-Time Databases in Distributed Environments," *Proceedings of IFIP 11th Working Conference on Database Security*, pp. 57-71, 1997.

[8] George, B. and J. Haritsa, "Secure Transaction Processing in Firm Real-Time Database Systems," *Proceedings of the ACM SIGMOD*, pp. 462-473, 1997.

[9] Haritsa, J., M. Carey, and M. Livny, "Dynamic Real-Time Optimistic Concurrency Control," *Proceedings of the 11th IEEE Real-Time Systems Symposium*, pp. 94-103, 1991.

[10] Jajodia, S. and V. Atluri, "Alternative Correctness Criteria for Concurrent Execution of Transactions in Multilevel Secure Database Systems," *Proceedings of IEEE Symposium on Security and Privacy*, pp. 216-224, 1992.

[11] Jajodia, S., L. V. Mancini, and I. Ray, "Secure Locking Protocols for Multilevel Secure Database Management Systems," *Proceedings of IFIP 10th Working Conference on Database Security*, pp. 177-194, 1996.

[12] Keefe, T. F. and W. T. Tsai, "Multiversion Concurrency Control for Multilevel Secure Database Systems," *Proceedings of the 10th IEEE Symposium on Research in Security and Privacy*, pp. 369-383, 1990.

[13] Kumar, V., *Performance of Concurrency Control Mechanism in Centralized Database Systems*, Prentice-Hall, 1986.

[14] Lam, K., S. H. Son, V. Lee, and S. Hung, "Using Separate Algorithms to Process Read-Only Transactions in Real-Time Systems," *IEEE Real-Time Systems Symposium*, pp. 50-59, 1998.

[15] McDermott, J. and S. Jajodia, "Orange Locking : Channel-Free Database Concurrency Control via Locking," *Proceedings of IFIP 6th Working Conference on Database Security*, pp. 267-284, 1995.

[16] Mukkamaka, R. and S. Son, "A Secure Concurrency Control Protocol for Real-Time Databases," *Proceedings of IFIP 9th Working Conference on Database Security*, 1995, pp. 235-253.

[17] Park, Chanjung and Seog Park, "Alternative Correctness Criteria for Multiversion Concurrency Control and its Applications in Advanced Database Systems," Proceedings of the 9th International Workshop on DEXA, pp. 864-869, 1998.

[18] Park, Chanjung and Seog Park, "SMVL: A Concurrency Control Protocol for Real-Time Secure Database Systems," Journal of Electrical Engineering and Information Science, Vol. 2 (No. 5), 1997. 10.

[19] Shu, L. C. and M. Young, *Correctness Criteria and Concurrency Control for Real-Time Systems : A Survey*, Technical Report SERC-TR-131-P, Indiana, 1992.

[20] Son, S., S. Park, and Y. Lin, "An Integrated Locking Protocol," *Proceedings of the 8th International Conference on Data Engineering*, pp. 527-534, 1992.

[21] Son, S. and B. Thuraisingham, "Toward a Multilevel Secure Database Management System for Real-Time Applications," *Proceedings of IEEE Workshop on Real-Time Applications*, pp. 131-135, 1993.

[22] Son. S, R. David, and B. Thuraisingham, "An Adaptive Policy for Improved Timeliness in Secure Database Systems," *Proceedings of IFIP 9th Working Conference on Database Security*, pp. 223-233, 1995.

[23] Son, S., C. Chaney, and N. Thomlinson, "Partial Security Policies to Support Timeliness in Secure Real-Time Databases," *IEEE Symposium on Security and Privacy*, pp. 136-147, 1998.

[24] 이주형, 박 석, "뷰 일관성에서 판독전용 트랜잭션을 위한 실시간 낙관적 병행수행 제어기법", *한국 데이타베이스 학술대회 논문집*, pp. 104-109, 1999.

박 찬 정

1988년 서강대학교 전자계산학과(학사). 1990년 한국과학기술원 전산학과(석사). 1998년 서강대학교 전자계산학과(박사). 1990년 3월 ~ 1999년 9월 한국통신 멀티미디어연구소 전임연구원. 1999년 9월 ~ 현재 제주대학교 컴퓨터교육과 조교수. 2000년 1월 ~ 현재 한국정보과학회 데이타베이스논문지 편집위원. 관심분야는 데이타보안, 트랜잭션 관리, 웹기반교육, 교육평가

한 희 준

1997년 서강대학교 전자계산학과(학사). 2000년 서강대학교 컴퓨터학과 학사(석사). 2000년 1월 ~ 2002년 2월(디지탈드림), 검색엔진 및 B2B솔루션 개발. 2002년 2월 ~ 현재(매크로임팩트), SAN 환경을 위한 스토리지 인프라스트럭쳐 소프트웨어 개발. 관심분야는 실시간 데이타베이스 시스템, 실시간 운영체제, 병렬처리시스템

박 석

정보과학회논문지 : 데이타베이스
제 29 권 제 1 호 참조