ECTPN을 이용한 키복구 시스템의 명세 및 분석

고 정 호[†] · 강 상 승[††] · 전 은 아[†] · 이 강 수[†††]

요 약

암호 메시지의 수신자가 그의 개인키를 분실하여 메시지를 복구할 수 없을 때, 수신자의 개인키와 메시지의 세션키는 복구되어야만 한다. 본 논문에서는 캡슐화 방식의 키복구 시스템(EKRS, Encapsulation based Key Recovery System)을 제안하였으며, EKRS는 위임된 키복구와 권한의 분리, 프라이버시 보호, 난수키, n-way 복구 형식으로의 확장, 상업적 키복구의 유용성 등의 특성을 지니는 새로운 키 캡슐화 방식의 키복구 시스템이다. EKRS는 ECTPN(an Extended Cryptographic Timed Petri Net)의 도식적인 모델에 의해 정형적으로 명세된다. 안전한 정보흐름과 세션키의 신뢰성은 ECTPN의 도달성 그레프를 사용하여 검증하였다. 공개키 기반 구조에서 실행되는 EKRS는 CALS와 EC, EDI와 같은 웹 기반 응용의 보안 솔루션으로 사용될 수 있다.

Specification and Analysis of Key Recovery System using ECTPN

Jeong-Ho Ko[†] · Sang-Seung Kang[††] · Eun-Ah Jun[†] · Gang-Soo Lee[†††]

ABSTRACT

When a receiver of ciphertext message can not decrypt a message because he has lost his private-key, the private-key of receiver and session-key of the message should be recovered. In this paper, we developed an Encapsulation based Key Recovery System (EKRS). EKRS is a new key encapsulation based key recovery system which is characterized by secretly choice of KRA, randomized target keys, n-way recovery type, and useful for commercial key recovery. EKRS is formally specified by a pictorial model, an Extended Cryptographic Timed Petri Net (ECTPN). Secure information flow and reachability of a session-key are verified by using reachability graph of ECTPN. EKRS, executing over a Public Key Infrastructure, can be used as a security solution in Web based applications such as CALS, EC and EDI.

1. 서 론

키 관리와 암호 알고리즘은 정보보호 시스템의 핵심 기술이다. 키 관리는 키의 생성 및 분산, 검증, 저장, 사용, 유효기간 만료, 분실 또는 손상된 키의 복구활동 들을 의미한다[1]. 키복구의 측면에서, 개인 비밀의 유지 문제와 키복구 문제간에는 요구사항이 상충되므로, 개인 비밀의 보호와 정부의 이익간에는 민감한 문제가 발생할 수 있다.

암호 정보의 복구를 위한 기술인 키복구 방법들은 다음과 같이 분류된다[1, 2].

- 키 위탁(Key escrow, 예, 클리퍼 칩이나 위탁 암호 표준[3-5, 20]) : 사용자는 1개 이상의 키 위탁 에이전트에게 세션키(즉, 암호키)를 송신한다. 사용자가 키복구를 필요로 할 때, 키 위탁 에이전트는 저장된 키를 사용자에게 제공한다
- 신뢰된 제삼자(Trusted Third Party, 예, Yaksha

† 준 회 원 : 한남대학교 대학원 컴퓨터공학과
†† 정 회 원 : 한국전자통신연구원 전자상거래연구부 연구원
††† 종신회원 : 한남대학교 대학원 컴퓨터공학과 교수
논문접수 : 2000년 2월 25일, 심사완료 : 2000년 6월 7일

system[6, 7], ANSI X9.17) : 사용자는 TTP(또는, 키 분배 센터(KDC))로부터 그들의 세션키를 얻는다. 사용자가 키를 복구하기를 원할 때, TTP는 원래 세션키를 사용자에게 제공한다.

- 상업적 키 백업(Commercial key backup, 예 : AT&T CryptoBackup[8]) : 데이터를 암호화할 때, 세션키는 공개키로 암호화하여 데이터와 함께 유지한다. 사용자가 키복구 에이전트(KRA ; Key Recovery Agent)에게 개인키를 보낸다. 사용자가 키복구를 원할 때, KRA는 사용자에게 저장된 개인키를 제공한다.
- 키 캡슐화(Key encapsulation, 예 ; TIS Recovery-Key[9], CyKey[10], SecretAgent[11], IBM SKR[12], Binding Cryptography) : 송신자는 세션키를 지정된 KRA의 공개키로 암호화한 자료인 키복구정보(KRI ; Key Recovery Information)를 다른 정보와 함께 캡슐화하여 메시지를 구성한다. 수신자가 키복구를 원할 때, KRI를 KRA에게 보낸다. KRA는 수신자에게 복구된 세션키를 돌려준다.

키위탁 에이전트와 신뢰된 제삼자는 트랜잭션 오버헤드와 대량의 메모리를 요구하므로, 정보시스템상의 병목이 될 수 있다. 상업적 키 백업 방법은 이러한 오버헤드를 줄이지만, 암호화된 데이터에 직접 접근이 가능하므로, 사용자들은 그들의 개인키를 백업하는 것에 불안감을 느낄 수 있다. 특히, 키 위탁 방법은 개인 비밀의 침해와 바인딩 문제(예컨대, 정부에 의한 모니터링 없이 사용자간의 비밀통신 문제)가 발생할 수 있다[5]. 그러나, 키 캡슐화 방법에서는 키의 소유자가 키복구의 권한을 가지므로, 개인 비밀의 침해로부터 비교적 자유롭고 다른 방법에 비해 많은 장점을 가진다.

그러나, 기존의 키 캡슐화 방법에는 다음과 같은 문제점[9-11]들이 있다. 첫째, 키복구 시스템의 개발을 위해 필요한 구현 기술뿐만 아니라, 정형 명세와 분석 모델이 부족하다. 둘째, 키복구의 요청자는 하나 이상의 KRA와 직접 통신을 해야하므로[9, 10], 모든 사용자들은 복잡한 키복구 기능을 가져야 한다. 셋째, 다수의 KRA로 구성되는 n-way 키복구 시스템에 관한 해결책이 제안되지 않았다.

이러한 문제점들을 해결하기 위해 본 연구에서는 새로운 캡슐화 방식의 키복구 시스템(EKRS ; Encapsulation based Key Recovery System)을 개발하였다. EKRS는 확장된 암호시간형 페트리넷(ECTPN , Extended Cryptographic Timed Petri Net)을 사용하여 정형적으로 모델링하고 분석하였다. 페트리넷은 1964년 이래로 병행시스템과 실시간 시스템의 정형적이고 도식적인 모델링 및 분석모델로서 성공적으로 사용되어져왔다[13, 14]. 또한, 페트리넷은 정보보호 시스템의 정보흐름[15]과 암호 프로토콜[16]의 모델링 및 분석을 위해 사용되었다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 키 캡슐화 방법들의 특성들과 EKRS의 기능 요구사항을 기술하며, 3장에서 ECTPN에 의해 정형적으로 모델링된 새로운 EKRS를 보인다. 4장에서는 ECTPN의 도달성 그래프(reachability graph)에 의해 EKRS의 도달성(예, 복구가능성)을 분석한 결과를 보이며, 끝으로, 5장에서 본 논문의 요약과 결론을 맺는다.

2. 키복구 시스템

2.1 기존의 방법

2.1.1 TIS(Trusted Information System)의 상업적 키복구(CKR)[9]

DRC와 DRF는 각각 데이터 복구센터(data recovery center)와 데이터 복구필드(date recovery field)이다. DRF에는 버전 번호, DRC의 이름 및 DRC의 공개키 번호가 포함된다. RE(recovery enabled)에는 DRC의 공개키에 의해 암호화된 세션키와 사용자 id가 포함된다. TIS의 주요 특징인 접근 규칙 인덱스(ARI ; Access Role Index)는 어떤 접근규칙을 나타내는 번호이다. 접근규칙은 비상 접근시 검증된 권한을 획득하기 위한 절차이고, 접근을 얻기 위해 제한된 사람들의 집합을 정의한다.

2.1.2 CyLINK의 CyKey[10]

CyKey는 KRAU(key recovery authority)를 제외하면 CKR과 유사하다. KRR은 키복구 요청자이며 CA(certification authority)와 비슷한 기능을 갖는 KRAU는 단지 KRR을 위한 승인 기관으로서의 역할을 한다.

2.1.3 Information Security Corp의 Secret Agent[11]

KRF(key recovery field)는 KRR(key recovery requester)의 공개키와 KRA의 공개키에 의해서 암호화된 세션키에 의해 생성된다. 키복구 단계에서, KRF는

KRR의 개인키와 KRA의 개인키에 의해 복구되며, 두 에이전트 KRR과 KRA가 키복구 절차에 포함되므로 키복구의 보안성은 증가된다.

2.1.4 IBM의 SKR(two-phased cryptographic secure key recovery)[12]

SKR은 다음과 같은 단계들로 구성된다.

- 단계 1 : 송신자(UA)는 수신자(UB)의 비밀값(S)을 생성한다. 각 KRA를 위해, UA는 S의 해쉬 함수로 KG(key-generating key)를 생성하고 KRA의 공개키로 KG를 암호화한다.
- 단계 2 : 암호세션에 의하여 수행되는 UA는 KK(key-encrypting key)를 생성한다. 암호화된 KG와 다중 암호화된 세션키(K)는 암호화된 세션으로 전송된다.
- 복구과정 : 비밀값을 복구하기 위하여, 암호화된 KG와 공개 복구 정보는 KG를 복구하고 KK를 재생하고, 재생된 KK를 KRA에게 전송한다. 복구 기관은 K를 복구하기 위하여 KK를 사용한다. KG가 KK로부터 파생되지 않을 수 있으므로, 그들은 다중 암호 세션을 사용한다.

단계 1에서는 비밀키 연산에 비해 속도가 느린 공개키 연산이 필요하므로 세션그룹당 1회만 수행한다. 단계 2는 매 세션마다 수행되며 비밀키 연산을 이용한다. 즉, 단계 1의 결과는 단계 2에서 필요한 비밀키 암호키들(KG, KK, K)을 생성하기 위해 여러번 사용된다. KRA에게 제공되는 비밀값은 세션그룹의 수행동안 사용되는 KG이며, 비밀값은 SKR에 의해 보호되는 키를 암호로 날인하는데 사용되는 비밀값이 KK이다. KK는 키 유도절차로 이들 KG로부터 유도되며, SKR에 의해 보호되는 키와 결합된 KRI와 유도된 KK는 복구를 위한 기관에 전송된다. KG는 KK로부터 유도될 수 없기 때문에, 새로운 값이나 공개키의 암호 없이도 세션그룹의 수행동안 사용된다.

2.2 키복구의 필요성 및 요구사항

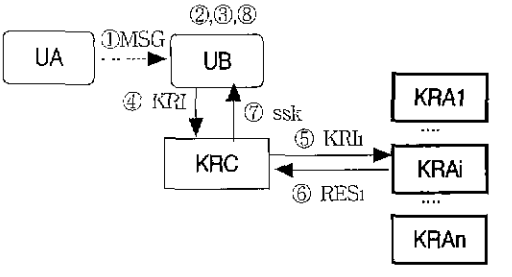
공개키 기반구조의 암호 메커니즘으로 사용자 A(UA)가 사용자 B(UB)에게 암호문을 보낸다고 가정할 때, 만약 UB가 그의 개인키를 잃어버렸거나 UA가 UB의 무효화된 공개키로 메시지를 암호화했다면, 세션키를 획득할 수 없으며, 암호문을 복호화할 수도 없게 된다. 따라서, UB는 세션키를 복구해야만 한다. 또한, 기관 내의 감사를 목적으로 하거나, 법 집행기관의 요구시에 이러한 키복구가 필요하다.

키복구 시스템의 최소한의 요구사항 및 가정 사항은 다음과 같다.

- EKRS는 PKI 기반에서 실행된다. 따라서, 모든 EKRS의 에이전트의 공개키와 인증서는 PKI의 KDC나 CA에 의해 안전하게 분배되어진다고 가정한다.
- 사용자의 키복구 요청시 안전하게 키복구가 수행되어지야 하며, 법 집행기관의 요구시에는 KRC와 연동하여 선택적으로 키복구를 할 수 있으며, 또한 사용자가 KRI를 조작할 수 없도록 시스템에서는 KRI 검증 기능을 가져야 한다.
- 복구의 대상은 데이터의 암호를 위한 비밀키인 세션키로 제한한다. 즉, 데이터전체의 복구는 하지 않으며, 세션키의 크기가 데이터보다 상대적으로 작기 때문에 키 캡슐화 방법을 적용할 수 있다.
- 정형적으로 명세되고 분석 및 검증되어야 한다.

2.3 캡슐화 키복구 시스템(EKRS)

본 시스템은 사용자시스템(UA, UB)과 키복구 센터(KRC), 키복구 에이전트(KRA)의 3가지 서브시스템으로 구성하며, (그림 1)에서 세션키의 기본적인 복구절차와 프레임워크를 보인다.



① MSG = SC(*data*, *ssk*) + PC(*ssk*, puk_{UB}) + KRI(where, KRI = merge(KRI_1 = PC(PC(fork(*ssk*, *i*), puk_{KRA_i}), puk_{KRC}))
②, ③ prk_{UB}를 잃거나 변경되었을 때, 새로운 키쌍(prk_{UB}'와 puk_{UB}')을 생성하고, CA에 puk_{UB}'를 등록한다. 그리고 KRC에게 키복구를 요청(UB나 법 집행기관의 요청)
④ KRC에게 KRI의 힘께 키복구 요청
⑤ KRI = PC(KRI, prk_{KRC}), KRI_i = divide(KRI), KRI의 함께 각 KRA_i에게 부분키 복구 요청
⑥ RES_i = PC(KRI_i, prk_{KRA_i})
⑦ ssk = join(RES_i)
⑧ MSG = SC(*data*, *ssk*)
(주) ssk : 세션키, prk_i : i의 개인키, puk_i : i의 공개키, SC(x, y) : DES(x, y)와 같은 비밀키 암호 함수(x는 데이터, y는 키), PC(x, y) : RSA와 같은 공개키 암호 함수, merge(), divide(), fork() and join()은 3.2절에서 정의한다

(그림 1) *n*-way EKRS의 프레임워크

EKRS의 키복구 시나리오는 다음과 같다.

- 단계 1(초기화) : PKI를 통해 모든 에이전트의 공개키를 분배한다.
- 단계 2(통신과 KRI 생성) . UA는 KRA 집단내에서 임의로 하나 이상의 KRA를 선택(즉, 권한의 분리)하고, 선택된 KRA의 공개키를 사용하여 KRI를 생성한다. KRI는 암호화 통신 메시지내에 캡슐화하여 UB에게 전송된다. UB가 자신의 개인키(prk_{UB})를 분실했거나, 혹은 UA가 UB의 만료된 공개키로 암호화하여 메시지를 전송했다면, 암호화된 메시지를 복구할 수 없다. 이때, UA가 선택한 KRA는 KRC 이외에는 누구도 알 수는 없다
- 단계 3(키복구) ' 메시지를 복구하려는 UB는 새로운 키쌍(puk_{UB}', prk_{UB}')을 생성하고, CA에게 새 공개키(puk_{UB}')를 등록한다. CA는 새 공개키가 포함된 새로운 인증서($cert_{UB}$)를 UB에게 발급한다. UB는 KRC에게 KRI와 복구요청서 및 새 인증서를 가지고 KRC에게 복구 요청(즉, 위임된 복구 요청)을 한다 나머지 키복구 절차는 (그림 1)의 ④~⑧의 과정을 통해 진행된다.

3. EKRS의 ECTPN 모델링

3.1 ECTPN의 정의

3.1.1 ECTPN 구조

프로토콜의 명세와 분석모델인 ECTPN은 페트리넷(Petri Net)[13]을 변형하여 정의되었다. 또한, TPN(Timed Petri Net)[14]과 CTPN(Cryptographic Timed Petri Net)[16]을 변형하여 정의되었으며, 다음과 같이 정의한다.

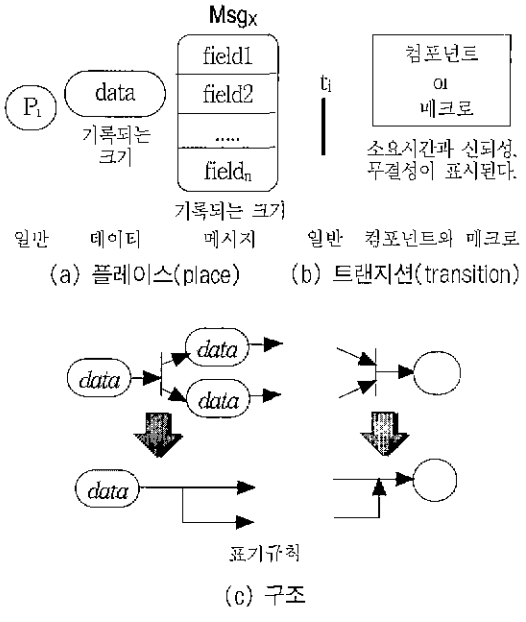
$$ECTPN = (P, T, I, O, M)$$

- P는 0개 이상의 정규 및 데이터, 메시지 플레이스와 같은 플레이스 클래스의 모임이다. 플레이스의 크기는 데이터와 메시지 플레이스의 경우에는 임의로 플레이스에 기록된다.
- T는 0개 이상의 정규 및 컴포넌트, 매크로 트랜지션(transition)과 같은 트랜지션 클래스의 모임이다. 트랜지션의 수행시간(즉, 소요시간)은 임의로 트랜지션에 기록된다.
- $I \subset P \times T$는 입력 함수의 집합(즉, 트랜지션으로 입력되는 화살표의 집합)이다.
- $O \subset P \times T$는 출력 함수의 집합(즉, 트랜지션으로부터 출력되는 화살표의 집합)이다.
- $M \subset P \times NI$는 마킹들의 집합(NI가 음이 아닌 정수의 집합일 때)이다.

데이터나 메시지 플레이스에 큰 점으로 표시되는 토큰(token)은 플레이스에 표시되는 데이터의 가용성(예를 들면, 개인키나 메시지는 사용 가능하다)으로서 해석한다 그러나, 일반 플레이스의 토큰은 일반 페트리넷의 경우에 제어와 자료의 추상적인 개체로서 해석한다.

컴포넌트나 매크로 트랜지션은 실행시간이 표시될 수 있는 RSA나 DES, exclusive-or, merge, divide, send. receive와 같은 함수나 컴포넌트로서 해석된다. 일반 트랜지션은 정규 페트리넷의 경우에 '즉간 트랜지션'(즉, 점화시간이 0인 트랜지션)이다. 일반 트랜지션은 입력 플레이스의 논리로 해석된다.

ECTPN의 가독성을 향상시키기 위해 다음과 같은 표기법들을 추가로 정의한다.



(그림 2) ECTPN의 표기법 정의

3.1 2 마킹(marking)과 점화(firing) 규칙

- 모든 플레이스는 초기화와 초기 키분배에 의해 플

레이스내에 토큰(즉, 키)이 놓일 때 마킹된다. 마킹된 플레이스들의 집합을 마킹셋(marking set)이라 하며, ECTPN에 의해 모델링된 시스템의 상태공간 내의 한 상태를 나타낸다

- 트랜지션의 모든 입력 플레이스에 마킹되면 그 트랜지션은 가용상태(enabled)를 나타낸다.
- 두 트랜지션의 입력 플레이스 집합들이 중첩되면 두 트랜지션들은 상호배타적으로 가용상태가 된다.
- 상호배타적이 아니며 동시에 가용상태에 있는 트랜지션들은 동시에 점화한다.
- 점화중인 트랜지션은 트랜지션에 부여된 소요시간이 지난 후에 모든 출력 플레이스에 마킹하며, 결과적으로, 새로운 마킹셋을 형성한다

3.1 3 ECTPN의 특성

ECTPN은 페트리넷의 확장된 종류가 아니고, 일종의 *colored Petri net*이다. ECTPN은 EKRS의 도식적인 명세와 분석(예, 도달성, 성능)을 위하여 정의한 것이다. ECTPN은 CTPN[16]과 TPN(Timed Petri net[14]) 뿐만 아니라, 페트리넷[13]의 모든 특성들을 그대로 상속받는다. ECTPN 모델은 EKRS의 자료흐름과 제어흐름을 통합한 명세모델이라 할 수 있다.

3.2 트랜지션과 플레이스의 정의

3.2.1 플레이스 정의

- 키 플레이스(key place) : *ssk*(세션키), *prk*(개인키), *prk'*(키복구를 위한 새로운 개인키), *puk*(공개키), *puk'*(키복구를 위한 새로운 공개키), *rk*(Vernam 암호를 위한 난수 키), *ik* (*ssk*를 위한 중간 키), *s-prk* (서명을 위한 개인키), *cert*(CA의 인증서), *sig*(서명), *s-puk*(서명검증을 위한 공개키), *krr*(키복구 요청 ; key recovery request)
- 메시지 플레이스(message place) : *msg*(메시지), KRI (키복구 정보 ; key recovery information)
- 데이터 플레이스(data place) : *data*(데이터), *req*(키복구 요청서), *int* (무결성 값 , integrity value)

3.2.2 컴포넌트 트랜지션

EKRS의 기능적 컴포넌트는 다음의 트랜지션으로 모델링된다.

- 연산 컴포넌트 트랜지션 : *merge*(연결), *divide*(분할), ⊕(Exclusive-OR), =?(동일 체크)
- 통신 컴포넌트 트랜지션 . *send*(송신 기능), *receive* (수신 기능)
  (단, 키복구가 저장된 암호 데이터에게 요구되어 질 때 *send/receive* 트랜잭션은 *write/read* 트랜잭션으로 각각 대치됨)
- 암호 컴포넌트 트랜지션

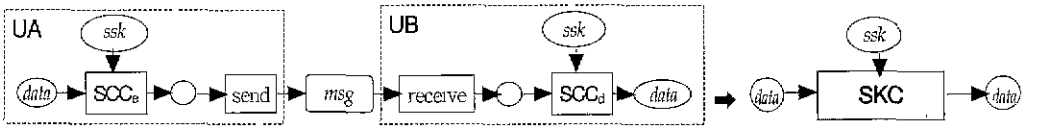
  비밀키 암호 컴포넌트(SCC) = {DES, IDEA, RC-4, …}, SCC_e(data, ssk), SCC_d(data, ssk)
  공개키 암호 컴포넌트(PCC) = {RSA, DH, Elliptic Curve, …}, PCC_e(data, puk), PCC_d(data, prk)
  해쉬 컴포넌트(HAC) = {SHA, MD5, …}, HAC(data)
  전자서명 컴포넌트(DSC) = {KCDSA, …}, DSC_s(data, s-prk, cert), DSC_v(data, s-pk, cert, sig) (단, e : 암호, d : 복호, s : 서명, v 서명검증, DES, IDEA, RC-4, RSA, DH, SHA, MD5, KCDSA(Korea Certification based Digital Signature Algorithm[17]))

이러한 컴포넌트 트랜지션은 다음의 매크로 트랜지션에서 사용되며, (그림 3)은 기본적인 매크로 트랜지션을 나타내며, KCDSA[17]와 전자서명이 포함된 매크로 트랜지션은 (그림 4)에서 보인다

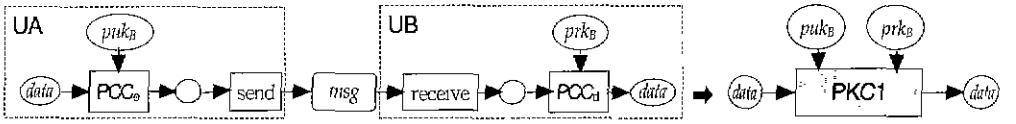
3.2.3 매크로 트랜지션

매크로 트랜지션은 EKRS의 상위 수준 설계를 위해 사용된다("X := Y"는 "X는 Y에 의해 정의된다"라는 의미임).

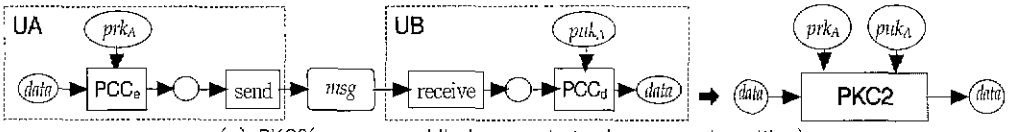
- 비밀키 암호 매크로 트랜지션(SKC)
  SKC(*data*, *ssk*)::= SCC_d(*receive*(A, *send*(B, SCC_e(*data*, *ssk*))), *ssk*)
- 단일 공개키 암호 매크로 트랜지션(PKC1, PKC2)
  PKC1(*data*, puk_B, prk_B)::= PCC_d(*receive*(A, *send*(B, PCC_e(*data*, puk_B))), prk_B)
  PKC2(*data*, prk_A, puk_A)::= PCC_d(*receive*(A, *send*(B, PCC_e(*data*, prk_A))), puk_A)
- 이중 공개키 암호 매크로 트랜지션(PKC3)
  PKC3(*data*, prk_A, puk_A, prk_B, puk_B)::= PCC_d(PCC_d(*receive*(A, *send*(B,PCC_e(PCC_e(*data*, prk_A), puk_B))), prk_B), puk_A)
- 무결성과 기밀성 암호 매크로 트랜지션 (ISC)
  ISC(*data*, *ssk*, prk_A, puk_A, prk_B, puk_B)'' =
  if PKC2 (HAC(*data*), prk_A, puk_A) =
  HAC(SKC(*data*, PKC3(*ssk*, prk_A, puk_A, prk_B, puk_B)))
  then *data* **else**error.



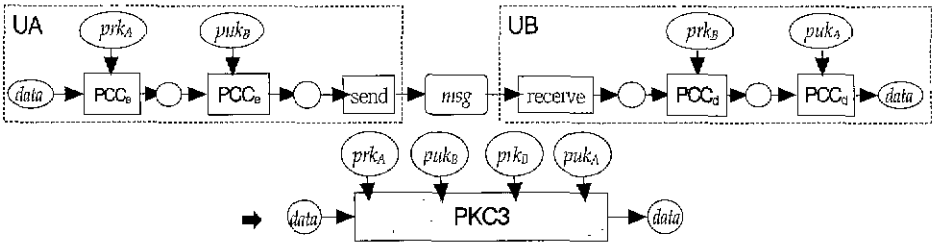
(a) SKC(secret key cryptographic macro transition)



(b) PKC1(one way public key cryptographic macro transition)

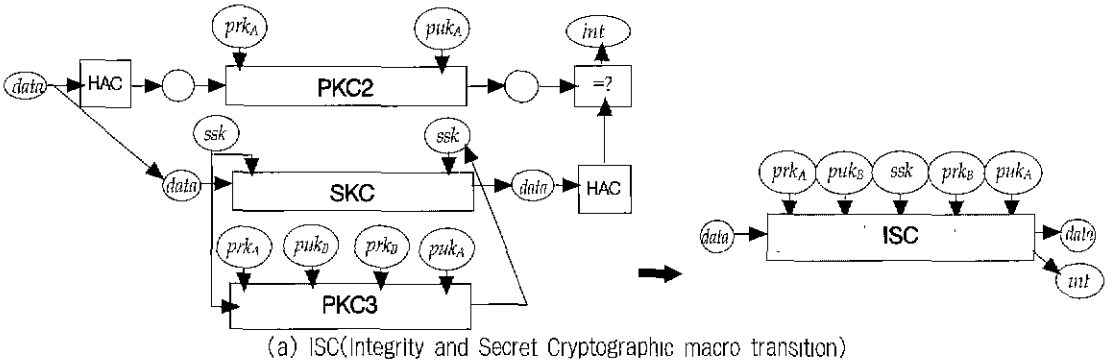


(c) PKC2(one way public key cryptographic macro transition)

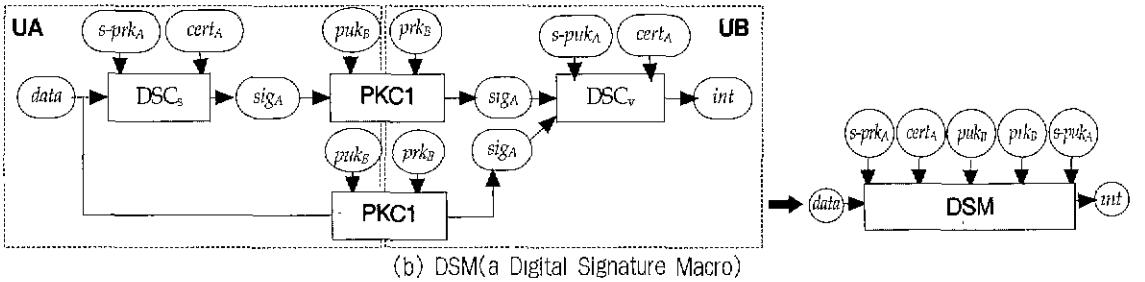


(d) PKC3(two way public key cryptographic macro transition)

(그림 3) 기본적인 매크로 트랜지션



(a) ISC(Integrity and Secret Cryptographic macro transition)



(b) DSM(a Digital Signature Macro)

(그림 4) 통합된 매크로 트랜지션

◦ 전자서명 매크로 (DSM)

SIG(*data, s-prk$_A$, cert$_A$, puk$_B$, prk$_B$, s-puk$_A$*)::=
DSC$_v$(PKC1(DSC$_s$(*data, s-prk$_A$, cert$_A$*), *puk$_B$, prk$_B$*),
PKC1(*data, puk$_B$, prk$_B$*), *s-puk$_A$, cert$_A$*)

컴포넌트 트랜지션은 기존의 암호 컴포넌트를 사용하여 구현할 수 있으며, 매크로 트랜지션은 ODBC(*open data base connectivity*)와 유사한 개념의 OCC(*open cryptographic connection*)로써 상위수준 암호와 통신을 위한 API라 할 수 있다. 매크로 트랜지션은 (그림 5)의 규칙을 사용하여 구현될 수 있다.

3.3 2-way EKRS의 ECTPN 모델 결과

EKRS는 암호와 통신 클래스 라이브러리의 컴포넌트를 사용하여 개발하였으므로, ECTPN 모델은 EKRS에 대한 API로써 간주할 수 있다. KRA를 두 개 갖는 2-way EKRS의 개념적 모델은 (그림 6)과 같으며 상세한 ECTPN 모델은 본 논문에서 생략하였다.

3.4 *n*-way EKRS로의 확장

2-way EKRS는 fork와 join기능을 사용하여 *n*-way EKRS(즉, *n*개의 KRA를 가진 EKRS)로 확장할 수 있다.

fork(*ssk, n*) = *x1, ... x$_i$, ... x$_n$* (*x1* = (*ssk* ⊕ *rk1*), *x2* = (*rk1* ⊕ *rk2*), ..., *x$_i$* = (*rk$_{i-1}$* ⊕ *rk$_i$*), ..., *x$_n$* = *rk$_{n-1}$*일 때),
join(*x1, ... x$_i$, ... x$_n$*) = *x1* ⊕ *x2* ⊕ ..., *x$_i$* ... ⊕ *x$_n$*, 단, *rk$_i$*는 난수이다.

ssk는 (***ssk***⊕*rk1*)⊕(*rk1*⊕*rk2*)⊕ . ., (*rk$_{i-1}$*⊕*rk$_i$*), ...⊕ (*rk$_{n-2}$*⊕ *rk$_{n-1}$*)⊕*rk$_{n-1}$* = ***ssk***이기 때문에, *n*-way로 확장하더라도 ssk는 보존된다. 다음 (그림 7)은 *n*-way로 확장했을 때의 세션키 복구의 흐름을 보인다.

4. EKRS의 도달성 분석

4.1 도달성 그래프(reachability graph)

도달성 그래프(RG)는 다음과 같이 정의한다.

RG = <N, LA>

여기서, N은 노드들의 집합이며, LA는 두 노드들을 연결하는 화살표의 집합이다. 노드는 동시에 마킹된 플레이스 집합이며, ECTPN의 한 상태를 나타낸다. 예를 들면, $t_i(p_{i1}, \ldots p_{ik} \ldots) \| t_j(p_{j1}, \ldots p_{jk} \ldots)$는 t_i와 t_j가 동시에 점화를 종료하는 것을 의미하고, p_{ik}, p_{jk}는 t_i, t_j의 각 출력 플레이스를 의미한다. 점화 가능한 트랜지션인 N_i가 점화될 때, 다음과 같이 트랜지션에 부여된 시간 중에 가장 긴 시간이 소요되며, 그때 새로운 노드 N_j가 생성된다.

$LA_{ij}[t_1(p_{11}, \ldots p_{1k} \ldots) \| t_2(p_{21}, \ldots p_{2k} \ldots) \| \ldots \| t_n(p_{n1}, \ldots p_{nk} \ldots)] : N_i \rightarrow N_j$ *max*(*delay*(*t1*), ... *delay*(t_i), ... *delay*(t_n)).
$p_{i1}, \ldots, p_{ik} \in N_i = \{mp_{i1}, mp_{i2}, \ldots mp_{ik}\}$

여기서, *mp*는 마킹된 플레이스(marked place)이며, *delay*(t_i)는 t_i의 소요시간을 나타낸다.

2-way EKRS에 대한 ECTPN의 도달성 그래프는 (그림 8)과 같다.

4.2 복구 시나리오와 소요시간, 도달성

복구 시나리오(SC_i)는 도달성 그래프상의 한 경로이다. 만일 $SC_i = \{N_{i1}, LA_{i1}, \ldots, N_{ik}, LA_{ik}, \ldots, N_{im}\}$($LA_{ik}$는 N_{ik}에서 N_{ik+1}으로의 연결선일 때)라면, SC_i와 *delay*(SC_i)는 다음과 같이 계산된다.

delay(LA_{ik}) = *max*(*delay*(*t1*), ..., *delay*(t_i), , *delay*(t_n)), $t_i \in LA_{ik}$일 때,

$$delay(SC_i) = \sum_{k=1}^{m} delay(LA_{ik})$$

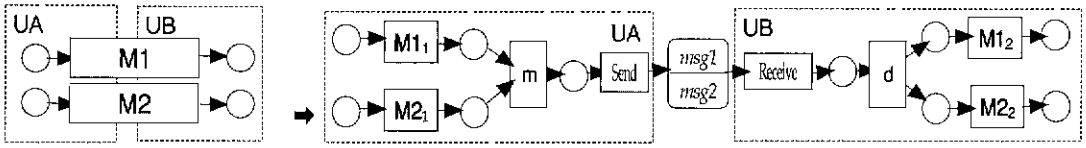
다음의 표는 본 논문에서 구현한 2-way EKRS의 각 트랜지션의 소요시간과 플레이스의 용량을 비교하였다. 단, 통신소요시간(send, receive, 각 컴포넌트의 통신시간)은 제외한다.

〈표 1〉 트랜지션의 소요시간과 플레이스의 용량

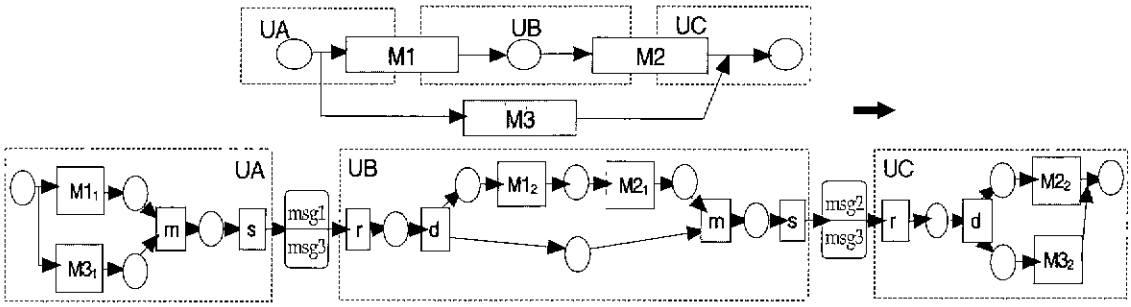
트랜지션(*t*)	소요시간(*delay*(*t*))	플레이스(*p*)	용량(*size*(*p*))
t1, t4, t11, t12 (ISC)	*48.57ms*	*data1, data2*	-
		ssk1, ssk2, ssk3, rk, ik1, ik2	*16 byte*
t2, t3, t7, t8 (PKC2)	*12ms*	*ik3, ik4*	*128 byte*
		ik5, ik6	*64 byte*
t5, t6, t9, t10 (PKC1)	*12ms*	*ik7, ik8*	*16 byte*
		req$_B$	*321 byte*
t13, t14, t15 (PKC3)	*36ms*	*req$_{KRC}$*	*226 byte*
		cert$_B$	*1024 byte*

(그림 8)의 통신소요시간을 제외한 총 복구 시간은

delay(SC_i) = 12(*t2*) 48.57(*t4*)+12(*t5*)+48.57(*t11*)+36(*t13*)+36(*t15*)+48.57(*t1*) = 241.71*ms*이다.



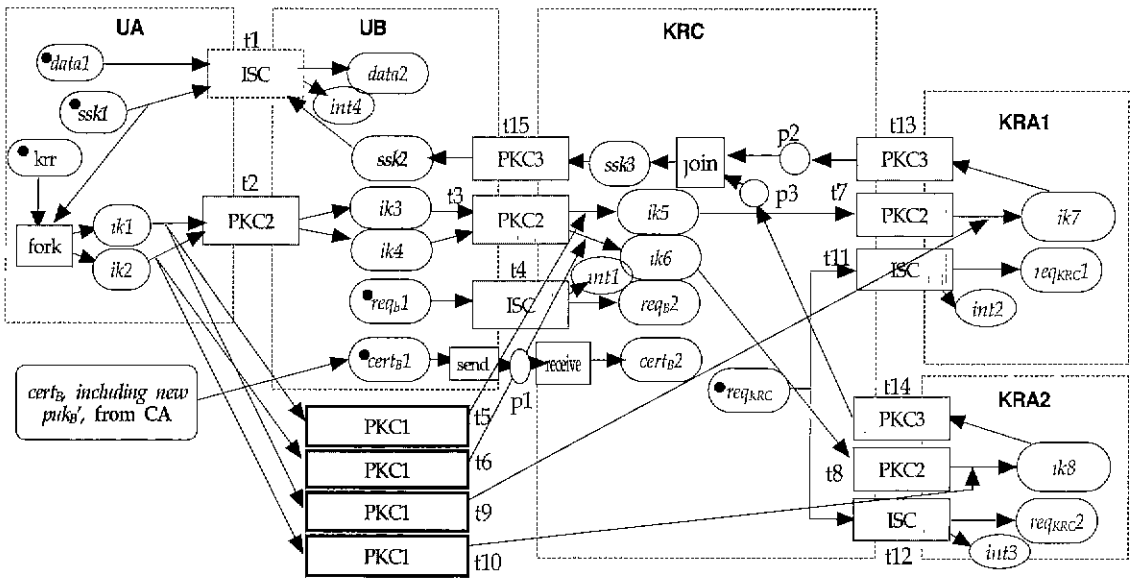
(a) 일괄적 통신(M1과 M2는 동시에 통신됨)



(b) 추이적 통신

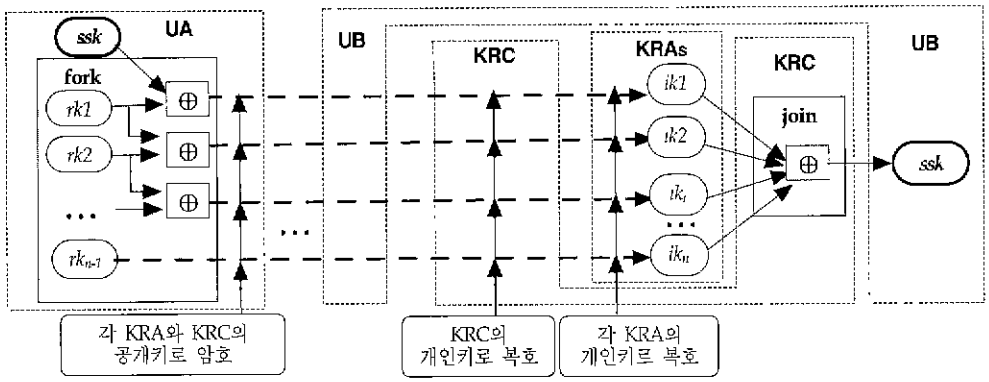
(주) Mi 는 매크로 트랜지션. M_{i1}과 M_{i2}는 송신자와 수신지네의 부분적인 계산(예, RSA, DES).
m : merge 트랜지션, d : divide 트랜지션, s : send 트랜지션, r : receive 트랜지션

(그림 5) 매크로 트랜지션의 구현규칙

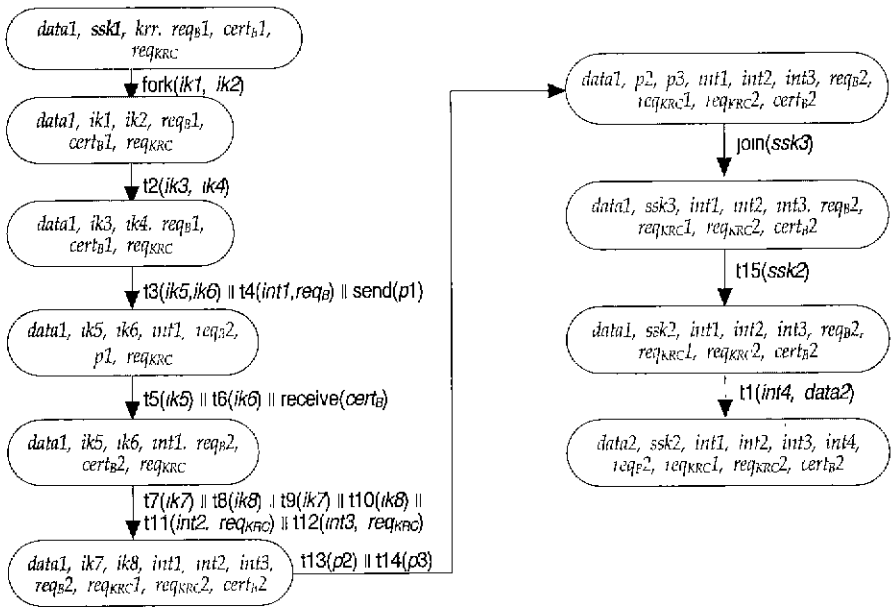


- 단언(assertions) : $data1 = data2$, $ssk1 = ssk2 = ssk3$, $ik1 = (ssk1 \oplus rk)$, $ik2 = rk$, $ik3 = PCC_e(PCC_e(ik1, puk_{KRA1}\ puk_{KRC}), ik4 = PCC_e(PCC_e(ik2, puk_{KRA2}), puk_{KRC})$, $ik5 = PCC_e(ik1, puk_{KRA1})$, $ik6 = PCC_e(ik2, puk_{KRA2})$. $ik7 = ik1\ \ ik8 = ik2$, $ssk3 = ik7 \oplus ik8$, $int1 = int2 = int3 = int4 = true$, $cert_B1 = cert_B2$, $req_n1 = req_n2$, $req_{KRC} = req_{KRC}1 = req_{KRC}2$
- 단, CA로부터 공개키 인증서의 분배절차는 생략하며, KRC와 KRA들은 사용자의 공개키를 미리 알 필요는 없다.

(그림 6) 2-way EKRS의 개념적 모델



(그림 7) *n*-way EKRS에서 세션키 복구의 흐름



(그림 8) 2-way ECTPN 모델의 도달성그래프

또한, ECTPN의 도달성(reachability)은 곧 EKRS 상의 세션키(*ssk*)의 복구성(recoverability)으로 해석할 수 있으며, 소요시간은 EKRS의 성능으로 해석할 수 있다. 예를 들어, UA의 초기마킹으로부터 UB의 *ssk2* 플레이스로 도달가능하다면, *ssk1*은 EKRS에서 복구가능하다는 것이다. 따라서, EKRS의 모든 트랜지션이 도달성 그래프를 사용하여 초기상태에서 도달가능하다는 것을 보일 수 있다

5. 요약과 결론

EKRS의 특성을 요약하면 다음과 같다.

- 기존의 캡슐화된 키복구 방법들과 비교한 결과는 <표 2>에 수록하였다.
- 하나 이상의 KRA는 각 통신 때마다 송신자(UA)에 의해 비밀히 선택되며, 수신자(UB)는 선택된 KRA의 식별자와 공개키에 대해 알 필요가 없으므로, 기

존의 방법들에 비해 KRA의 공모에 대한 가능성이 낮다.

〈표 2〉 기존의 방법과 비교

캡슐화 키복구 방법	수행 가능한 도식적 모델링	도달성 (복구성) 분석	세션키 분리	KRA 수	KRA 지정	새로운 공개키 분배절차
TIS's CKR[9]	N/A	N/A	N/A	1	고정	N/A
CyLINL's CyKey[10]	N/A	N/A	N/A	1	고정	N/A
ISC's SecretAgent[11]	N/A	N/A	N/A	1	고정	N/A
IBM's SKR[12]	N/A	파라미터 집중 구조	N/A	2이상	고정 or 사용자에 의해	N/A
NIST's SRKRP[19]	N/A	N/A	N/A	1	N/A	N/A
EKRS	ECTPN 모델	도달성 그래프 구조	무작위 방식 (randomized)	n	송신자에 의해 n개의 KRA 선택	공개키 인증서에 의해

- 키복구는 UB의 개인키를 잃어버렸을 때뿐만 아니라, UB가 갱신되기 이전의 공개키로 암호화된 메시지를 수신했을 때도 요구 할 수 있다

- 위임형 키복구형태이며, 각 사용자가 KRA들에게 직접 접촉하지 않고, 키복구를 KRC에게 위임한다. 따라서, KRC는 사용자 도메인보다는 안전한 복구 도메인을 형성하며 보안감시자의 역할을 한다. 또한, 복구 로그파일의 관리하기가 용이하다.

- 권한의 분리와 개인의 프라이버시 보호를 제공한다. KRI는 KRC나 KRA들에게 득점될 수 없으며, 사용자(송신자와 수신자)는 복구기능을 임의로 선택할 수 있다 그러므로, 사용자는 서로 안전하게 통신할 수 있는 것이다

- 난수키 방식 . n-way EKRS의 경우, 세션키(128 비트)는 n개의 부분키(즉, $128/n$ 비트의 길이)로 물리적으로 분할(이때, 암호 복잡성($2^{-128/n}$)이 지수적으로 감소)하는 것이 아니라, n개의 128 비트짜리 중간키들을 세션키와 $n-1$개의 난수키를 exclusive-or 연산하여 생성한다. 그 결과, 비도는 보존되며, 공격자는 128 비트짜리 세션키나 n개의 KRA의 각 개인키(즉, $344\times n$ 바이트)를 공격해야 한다. 후자의 공격은 n개의 KRA들이 모두 공모하기 전에는 불가능한 것이다.

- EKRS는 ECTPN 모델을 정형적으로 모델링하고 분석하였다 ECTPN 모델은 다양한 모델링기능(예, 암호 컴포넌트 및 매크로 트랜지션)과 분석기능을 가지며 다른 정보보호시스템에도 적용할 수 있는 모델이다.

- 컴포넌트기반 소프트웨어공학 기술(CBSE)에 의한 구현 : OMG의 CORBA와 MS의 DCOM/OLE. SUN의 JavaBeans로 시작된 CBSE 기술은 현재의 정보기술이라 할 수 있다 본 연구에서는 새로운 암호알고리즘[18]을 개발한 것이 아니라, 기존의 암호 라이브러리와 매크로 트랜지션 개념을 통해서 EKRS를 구현하였다. 따라서, 암호 알고리즘을 쉽게 교체할 수 있는 등 시스템의 유지보수성이 증가된다.

- NIST의 SRKRP[19] 표준들을 준수하여 개발하였다.

결론적으로, EKRS는 위임된 복구 요청과 권한의 분리, 개인의 프라이버시 보호, 난수 키, n-way 복구 형태로 확장성 및 상업적 키복구 형태에 유용하다는 특징을 가진 새로운 캡슐화기반의 키복구 시스템이다 특히, EKRS는 도식적 모델인 ECTPN에 의해 정형적으로 명세되었으며, ECTPN 모델은 도식적인 모델링과 분석뿐만 아니라, 컴포넌트기반으로 구현하는데 유용하다. 본 연구의 향후 연구과제로는 EKRS를 적용할 수 있는 솔루션 즉, PKI기반에서 수행될 수 있는 CALS와 EC, EDI와 같은 웹기반의 응용에서 사용할 수 있는 보안솔루션을 개발하는 것이다.

참 고 문 헌

[1] Technology Committee of Key Recovery Alliance, Cryptographic Information Recovery using Key Recovery, A Working Paper. Version 1.2, http : // www.kra.org, Aug. 1997

[2] Dorothy E. Denning and Dennis K. Branstad, "A Taxonomy for Key Escrow Encryption Systems," Communications of the ACM, pp.34-40, Vol.39, No. 3, 1996

[3] Ravi Ganesan, "How To Use Key Escrow," Communications of the ACM, Vol.39, No.3, pp.33, Mar. 1996.

[4] Jingmin He and Ed Dawson, "A New Key Escrow Cryptosystem," Lecture Notes in Computer Science. Vol.1029, pp.105-113, 1995.

[5] Yung-Cheng Lee ; Chi-Sung Laih, "On the key recovery of the Key Escrow System," Proceedings of 13th Annual Computer Security Applications Conference, pp.216-220, 1997.

[6] Ravi Ganesan, "The Yaksha Security System," Communications of the ACM. Vol.39, No.3, pp.55-60. Mar. 1996.

[7] Jefferies. N., Mitchell, C. and Walker, M., "A Proposed Architecture for Trusted Third Party Services," Lecture Notes in Computer Science, Vol. 1029, pp.98-104, 1995.

[8] D. P. Maher, Crypto Backup and Key Escrow, Communications of ACM. Vol.39, No.3, pp.48-53, Mar. 1996.

[9] Stephen T. Walker, Steven B. Lipner, Carl M. Ellison and David M. Balenson, "Commercial Key Recovery," Communications of the ACM, Vol.39, No.3, pp.41-47, 1996.

[10] M. Markowitz and R. Schlafly, "Key Recovery in SecretAgent," Digital Signature, 1997

[11] "CyKey, A Key Recovery System for Commerical Environments," Cylink Corp., http : //www.cylink.com, 1998.

[12] R. Gennaro, et. al., "Secure Key Recovery," IBM Thomas J. Watson Research Center, 1999

[13] James Peterson, J., 'Petri Nets Theory and the Modeling of Systems'. Prentice Hall. 1982.

[14] Zuberak, W , "Timed Petri Nets : Definitions, Properties, and Applications," Microelectronics and Reliability, Vol.31, pp.627-644, 1991.

[15] Varadhajan, V., "Petri Net based Modeling of Information Flow Security Requirements," Proc. of the Computer Security Foundations Workshop III, pp.51-61, 1990

[16] Gang-Soo Lee and Jin-seok Lee, "Petri Net based models for specification and analysis of Cryptographic Protocols." Journal of systems and software, Vol.37, pp.141-159, 1997.

[17] 'Korean Certification-based Digital Signature Algorithm', 한국정보보호센터, 1997.

[18] B. Scheneier, Applied Cryptography(second edition). Wiley& Sons, 1996.

[19] "Requirements for Key Recovery Products, (Final Report)," Federal Information Processing Standard for Federal Key Management Infrastructure, http://csrc.nist.gov/key recovery/, Nov. 1998.

[20] 채승철, 이임영, "안전한 키 위탁 시스템에 관한 연구", 한국통신정보보호학회논문지, 제9권, 제2호, pp. 83-92. Jun. 1999.

고 정 호

e-mail : jhko@se.hannam.ac.kr

1997년 한남대학교 컴퓨터공학과 졸업(학사)

1999년 한남대학교 대학원 컴퓨터공학과 졸업(공학석사)

1999년~현재 한남대학교 대학원 컴퓨터공학과 박사과정

관심분야 : 전자상거래, 정보보호, 소프트웨어 컴포넌트

강 상 승

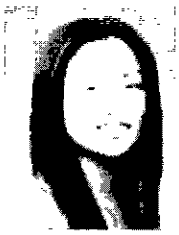
e-mail : kss@etri.re.kr

1997년 경북대학교 전자공학과 졸업(학사)

1999년 경북대학교 대학원 전자공학과 졸업(공학석사)

1999년~현재 한국전자통신연구원 전자상거래연구부 연구원

관심분야 : 전자상거래, 정보보호 응용 기술



전 은 아

e-mail : penguin@se.hannam.ac.kr

1999년 한남대학교 컴퓨터공학과 졸업(학사)

1999년~현재 한남대학교 대학원 컴퓨터공학과 석사과정

관심분야 : 암호응용, 정보보호, CALS/EC, 소프트웨어 공학

이 강 수

e-mail : gslee@eve.hannam.ac.kr

1981년 홍익대학교 컴퓨터공학과 졸업(학사)

1983년 서울대학교 대학원 전산학과 졸업(이학석사)

1989년 서울대학교 대학원 전산학과 졸업(이학박사)

1985년~1987년 국립대전산업대학교 전자계산학과 전임강사

1992~1993년 미국일리노이대학교 객원교수

1995년 한국전자통신연구원 초빙연구원

1998~1999년 한남대학교 멀티미디어학부장

1987년~현재 한남대학교 컴퓨터공학과 정교수

관심분야 : 소프트웨어공학, 병행시스템 모델링 및 분석, 정보보호시스템 평가, 멀티미디어교육 커리큘럼