情 報 保 護 學 會 誌
第19卷 第5號, 2009. 10

Proxy Re-encryption 기술

송 유 진*, 박 광 용**

요 약

최근 대용량 데이터의 급속한 생성, 유통으로 인해 데이터 서비스 사용자가 증가하고 있다. 이때, 대용량 데이터를 분산 데이터베이스 시스템에 저장·관리하는 경우, 데이터에 대한 위협문제가 발생된다. 본 논문에서는 대용량 데이터를 암호화하여 관리할 때 복호권한을 위임함으로써 보다 안전하게 데이터를 관리하는 Proxy Re-encryption 기법에 대해 검토한다.

Ⅰ. 서 론

인터넷의 발전과 대용량 데이터의 급속한 유통으로 데이터 서비스 사용자가 급격히 증가하고 있다. 이러한 인터넷 서비스 데이터의 지속적인 증가로 인해 시스템의 구축비용과 확장성이 인터넷 서비스 업체의 경쟁력 확보에 중요한 요소가 되고 있다.

또한, 인터넷 서비스 데이터량의 지속적인 증가로 대량의 원시 데이터로부터 정보를 가공 처리하는 과정, 체계화된 정보의 저장 관리 및 유용한 정보를 추출하기 위한 분석 등에 분산 컴퓨팅 기술을 적용하는 움직임이 활발히 진행되고 있다.

구글, 야후 등 글로벌 인터넷 서비스 업체들은 인터넷 서비스 플랫폼의 중요성을 인식하고 자체 연구 개발을 수행, 저가 상용 노드를 기반으로 한 대규모 클러스터 기반의 분산컴퓨팅 플랫폼 기술을 개발 활용하고 있다[12][13].

대용량 데이터 처리 및 저장 관리가 필요한 대표적인 어플리케이션으로는 인터넷 서비스 분야 외에 예를 들면, 비즈니스 인텔리전스 등 다른 응용 영역으로 확대하여 클라우드 서비스로 제공하려는 비즈니스 모델이 제시되고 있다.

이와 같이 분산 컴퓨팅 환경에서 다양한 데이터 서비스가 가능해지면서 대용량 데이터의 분산관리가 주요 이슈로 떠오르고 있다.

한편, 대용량 데이터의 다양한 이용 형태로부터 악의적인 공격자나 내부 사용자에 의한 보안 취약성 및 프라이버시 침해가 발생할 수 있다. 향후, 전자정부나 민간기업 등에서 취급하는 정보의 양이 점차 대규모화됨에 따라 개인정보 등의 프라이버시 관련 정보를 안전하게 저장해야 한다.

따라서 데이터의 안전한 저장·관리 문제 해결이 시급한 실정으로서 보안상의 위험성을 고려하여 데이터를 암호화한다. 그러나 암호화 방식은 키 분배 및 관리 운용상의 제약이 존재할 수 있다. 본 논문에서는 프록시 재암호화(Proxy Re-encryption) 기법을 이용하여 암호화된 데이터를 복호할 때 복호권한을 위임하여 다른 사용자가 복호할 수 있도록 한다. 이러한 권한 위임은 현재 대용량 데이터를 관리할 때 데이터의 안전한 분산관리 문제에 응용될 가능성이 있다.

본 논문에서는 대용량 데이터를 암호화하여 관리할 때 복호권한을 위임함으로써 보다 안전하게 데이터를 관리하는 Proxy Re-encryption 기법에 대해 검토한다. 본 논문은 다음과 같이 구성된다. 2장에서는 관련 연구로써 Proxy Re-encryption에 대해 검토하고 3장에서는 복호권한이 위임가능한 ID기반 Proxy Re-encryption 방식에 대해 검토한다. 4장에서는 결론을 맺는다.

Ⅱ. 관련 연구

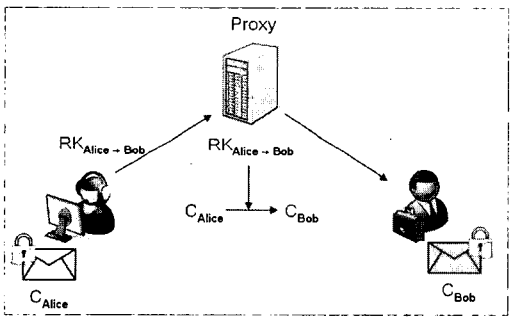
현재 Proxy Re-encryption은 많은 연구가 이루어지

이 논문은 2009년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. 2009-0087849)
* 동국대학교 정보경영학과 (song@dongguk.ac.kr)
** 동국대학교 전자상거래협동과정 (freemickey@dongguk.ac.kr)

고 있다[4][5][6][7][8][9][10][11][14][15].

Proxy Re-encryption이란 Proxy가 Alice의 공개키로 암호화된 암호문을 Bob의 비밀키로 복호할 수 있도록 암호문을 변환하는 방식이다. Proxy는 암호문을 변환하기 위한 키(re-encryption key)를 이용하여 기존의 암호문을 복호하지 않고 암호문을 변환할 수 있기 때문에 Proxy는 평문이나 Alice의 비밀키를 알지 못한다. 이러한 방식은 암호 메일의 전송, 파일 시스템 등에 응용할 수 있다.

메일 전송을 예로 들면, Alice의 부재 또는 비밀키를 분실했을 경우, Alice의 암호 메일을 Proxy가 Bob의 암호 메일로 변환하여 메일을 전송하는 것이 가능하다. 이때 Proxy는 암호문을 복호하지 않고 Alice의 암호문을 Bob의 암호문으로 변환만 한다. 또한, Bob도 Alice의 비밀키를 이용하지 않고 Bob 자신의 비밀키로 암호문을 복호할 수가 있다.



〔그림 1〕 Proxy Re-encryption기반 mail 전송 예

2.1 Mambo, Okamoto[4]

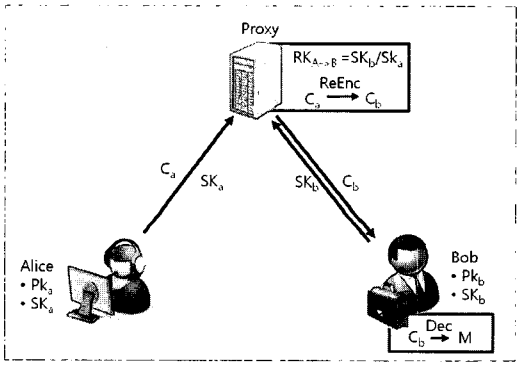
암호문의 복호 권한을 위임하는 방식은 1997년 Mambo, Okamoto[4]에 의해 처음으로 제안되었다. 이방식은 Alice의 공개키로 암호화된 암호문을 Bob이 복호할 수 있도록 암호문을 변환한다. 그러나 Alice만 암호문을 변환할 수 있기 때문에 Alice가 부재인 경우에는 암호문을 변환할 수가 없다.

2.2 Blaze, Bleumer, Strauss[5]

복호 권한을 위임하는 방식이 제안된 후 1998년에 Blaze, Bleumer, Strauss[5]에 의해 "atomic proxy cryptography"가 제안되었다(BBS 방식). BBS 방식은 암호문을 변환하는 Proxy가 re-encryption key를 이용하여 Alice의 암호문을 Bob의 암호문으로 변환한다. BBS 방식의 이점은 Alice가 암호문을 변환하지 않고 Proxy가 변환하는 것이다.

예를 들어 Alice가 부재인 경우나 비밀키를 잃었을 경우 Proxy가 암호문을 변환하여 Bob이 복호 할 수 있다. 여기서 Proxy는 암호문을 변환할 뿐 평문을 볼 수는 없다.

BBS 방식의 re-encryption key는 Alice의 비밀키와 Bob의 비밀키로부터 생성된다. 그래서 Proxy가 Alice의 암호문으로부터 Bob의 암호문으로 변환하는 것이 가능할 뿐만 아니라, Bob의 암호문으로부터 Alice의 암호문에 변환할 수도 있다. 이것을 Bidirectional Proxy Re-encryption라고 한다.



〔그림 2〕 BBS 방식

BBS 방식은 ElGamal 암호시스템에 기반을 두고 re-encryption key $RK_{A \to B}$의에 의해 암호문을 변환한다. re-encryption key $RK_{A \to B}$의 사용으로 평문을 복원하지 않고 하나의 비밀키에서 다른 키로 재암호화된다.

BBS 방식은 다음의 4가지 알고리즘으로 구성된다.

- Key Generation : Alice와 Bob의 공개키와 비밀키, re-encryption key $RK_{A \to B}$를 생성
- Encryption : 평문 m을 공개키로 암호화하여 암호문 C_a를 생성
- Decryption : 암호문 C_a를 공개키에 대칭되는 비밀키로 복호화하여 평문 m을 생성
- Re-encryption : Decryption단계를 거치지 않고 re-encryption key $RK_{A \to B}$를 사용하여 다른 공개키로 재암호화

(1) Key Generation:

$<g> = G$ of prime order q.

- 비밀키 $SK_a = a \in Z_q^*$와 $SK_b = b \in Z_q^*$를 랜덤하게 생성한다.
- 공개키 $PK_a = g^a$와 $PK_b = g^b$를 생성한다.
- 재암호화키 $RK_{A \to B} = b/a = b \cdot a^{-1} (\mathrm{mod} q)$를 생성한다.

(2) Encryption:

- 평문 $m \in G, r \in Z_q^*$을 랜덤하게 선택한다.
- 공개키 PK_a로 평문을 암호화한다.

$C_a = (g^r \cdot m, g^{ar})$

(3) Decryption:

- 비밀키 SK_a에 의해 평문 m을 복호한다.

$$m = \frac{g^r \cdot m}{(g^{ar})^{1/a}}$$

(4) Re-encryption:

- 암호문 C_a를 Proxy가 재암호화키 $RK_{A \to B}$를 이용하여 C_b로 재암호화한다.

$C_a = (g^r \cdot m, g^{ar})$

$C_b = (g^r \cdot m, (g^{ar})^{RK_{A \to B}})$

$= (g^r \cdot m, (g^{ar})^{b/a})$

$= (g^r \cdot m, g^{br})$

- 재암호화된 C_b는 비밀키 SK_b에 의해 복호가 가능하다.

$$m = \frac{g^r \cdot m}{(g^{br})^{1/b}}$$

BBS Proxy Re-encryption 알고리즘은 ElGamal에 기반하고 있기 때문에 암호문은 의미론적으로 안전하다. 따라서 a 또는 b를 알지 않고서 키서버는 $RK_{A \to B} = b/a$와 Z_q^*의 임의의 요소를 구별할 수 없다. 그러나, 본 방식은 Proxy와 Bob이 결탁(Collusion)하여 Alice의 비밀키가 유출되거나 다른 사용자의 re-encryption key가 생성되는 등 문제가 있다. 따라서, 다음의 문제가 개선되어야 한다.

1) Bidirectionality : Proxy는 $(RK_{A \to B})^{-1} = a/b$를 계산할 수 있다. 즉, Alice의 키로 Bob의 메시지를 재암호화하는 것을 가능하게 한다.
2) Collusion : Proxy가 Alice와의 공모를 통해 Bob의 비밀키 SK_b를 알아낼 수 있다. 반대로 프록시와 Bob이 Alice의 비밀키 SK_a를 알아내기 위해 공모할 수 있다.
3) Re-encryption Key Generation : $RK_{A \to B}$를 생성하기 위해서는 Alice와 Bob의 비밀키가 모두 필요하다.

2.3 Unidirectional Proxy Re-encryption

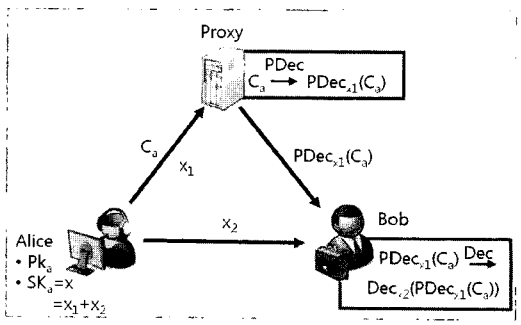
BBS 방식의 개선점은 다음과 같다.

1) Unidirectionality : Alice를 통해서 생성된 Bob의 re-encryption key는 Bob으로부터 Alice에 의해 재암호화하는 것이 허용되면 안된다.
2) Collusion-resistance : Alice와 프록시는 Bob의 비밀키에 대한 어떠한 것도 습득할 수 없어야 한다.
3) No pre-sharing : Alice가 Bob에 대한 re-encryption key를 생성하기 위해서는 SK_a와 PK_b가 요구되어야 한다. 즉, Bob의 비밀키는 필요하지 않다.

2.3.1 Dodis, Ivan[6]

Dodis, Ivan[6]은 2003년에 BBS방식을 개선하여 unidirectional Proxy encryption 기능을 가지는 방식을 제안했다(DI 방식). Unidirectional Proxy encryption은 Proxy에 대해 Alice의 암호문으로부터 Bob의 암호문으로 변환할 수 밖에 할 수 없다고 하는 성질을 가진다.

통상적인 Proxy Re-encryption의 개념은 Proxy가 Alice의 암호문을 Bob의 암호문으로 재암호화하는 것이다. 하지만 DI방식은 Alice의 비밀키를 2개의 키로 분할한다. 하나는 Proxy에게 전달하여 암호문의 변환에



〔그림 3〕 DI 방식

이용하고 나머지는 Bob에 전달하여 Proxy가 변환한 암호문의 복호에 이용하는 방식이다.

이 때문에 몇가지 문제점이 발생하는데, 첫 번째가 키분배의 문제이다. 분리된 비밀키를 안전한 채널로 전송하는 것이 쉽지 않을 것이다. 두 번째는 Proxy와 Bob의 결탁이다. Proxy와 Bob이 결탁한다면 Alice의 비밀키는 그대로 노출된다.

DI의 Proxy Re-encryption의 알고리즘은 다음과 같다.

(1) Key Generation:

- Alice의 비밀키 $SK=x\in Z_q$와 공개키 $PK=y$를 생성, 여기서 $y=g^x$이다(이때, $g\in Z_p^*$).
- Alice의 비밀키 $SK=x$를 x_1과 x_2로 분할한다 ($x=x_1+x_2$).
- x_1는 Proxy에게, x_2는 Bob에게 전달한다.

(2) Encryption:

- $r\in Z_q$은 랜덤하게 선택하고 평문 m을 공개키 $PK=y$로 암호화한다.

$$C=Enc_y(m)=(g^r, m\cdot g^{xr})$$

(3) Proxy Decryption:

- 암호문 C를 Proxy가 비밀키 조각 x_1를 이용하여 C'로 재암호화한다.

$$C=(g^r, m\cdot g^{xr})$$

$$C'=\left((g^r)^{x_1}, m\cdot g^{xr}\right)$$

(4) Decryption (Bob):

- C'는 비밀키 조각 x_2에 의해 복호가 가능하다.

$$C'=\left((g^r)^{x_1}, m\cdot g^{xr}\right)$$
$$=\left((g^r)^{x_1+x_2}, m\cdot g^{xr}\right)$$

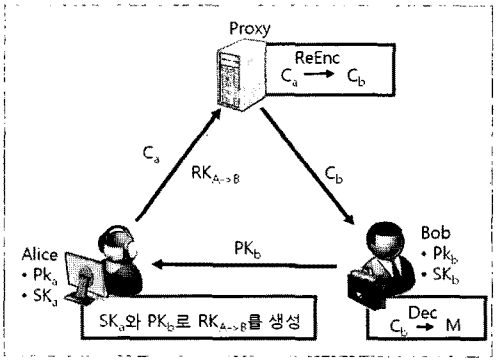
$$Dec_{x_2}(C')=\frac{m\cdot g^{xr}}{(g^r)^x}=m$$

여기서, DI방식은 Alice가 비밀키를 분할한 후 Proxy와 Bob에게 전달하여 변환된 암호문을 복호화할 수 있다. 이는 기존의 Proxy Re-encryption에서 Alice의 암호문을 재암호화하여 Bob의 비밀키로 복호하는 개념과는 다르다. 즉, Bob은 자신의 비밀키로 복호화하는 것이 아니라 Alice의 비밀키 조각으로 복호한다. 그리고 암호화시 Proxy도 Alice의 비밀키 조각으로 암호문을 변환할 뿐이다. 따라서, DI방식은 BBS의 Bidirectionality을 개선한 것이다.

2.3.2 Ateniese 등[7]

2005년에 Ateniese등[7]은 BBS방식이나 DI방식을 개선한 unidirectional Proxy Re-encryption을 제안했다 (AFGH 방식).

즉, Alice가 Bob에게 복호권한을 위임하기 위해서 만든 Re-encryption key는 Bob이 Alice의 암호문을 재암호화하는데 활용되어서는 안된다. Ateniese 등의 방식에서 re-encryption key는 Bob의 공개키와 Alice의 비밀키를 조합하여 생성된다. 또한, re-encryption key의 생성은 Alice 자신이 생성할 수 있다.



〔그림 4〕 AFGH 방식

AFGH 방식은 다음의 5가지 알고리즘으로 구성된다.

(1) Key Generation:

$<g>=G_1$ of prime order q.

- 비밀키 $SK_a=a\in Z_q^*$와 $SK_b=b\in Z_q^*$를 랜덤하게 생성한다.
- 공개키 $PK_a=g^a$와 $PK_b=g^b$를 생성한다.
- 임의의 난수 $r\in Z_q^*$을 생성한다.
- $Z=e(g,g)$
- 재암호화키 $RK_{A\rightarrow B}=(g^b)^{1/a}=g^{b/a}$를 생성한다.

(2) Encryption:

- 평문 $m\in G_2$을 공개키 PK_a로 암호화한다.

$$C_a=(Z^r\cdot m, g^{ra})$$

(3) Decryption (Alice):

- 암호문 C_a를 비밀키 SK_a로 복호화한다.

$$m=\frac{Z^r\cdot m}{e(g^{ra},g^{1/a})}=\frac{Z^r\cdot m}{Z^r}$$

(4) Re-encryption:

- 암호문 C_a를 Proxy가 재암호화키 $RK_{A\to B}$를 이용하여 C_b로 재암호화 한다.

$$C_a=(Z^r\cdot m, g^{ra})$$
$$C_b=(Z^r\cdot m, e(g^{ra}, RK_{A\to B}))$$
$$=(Z^r\cdot m, e(g^{ra}, g^{b/a}))$$
$$=(Z^r\cdot m, Z^{rb})$$

(5) Decryption (Bob):

- 재암호화된 C_b는 비밀키 SK_b에 의해 복호가 가능하다.

$$m=\frac{Z^r\cdot m}{(Z^{rb})^{1/b}}$$

AFGH방식은 다음의 이점을 제공한다.

1) 사전에 비밀키를 공유하지 않는다. 즉, Alice는 오직 SK_a와 PK_b를 사용해서 $RK_{A\to B}$를 계산할 수 있게 된다. Bob은 Alice 또는 제 3자와 SK_b를 공유할 필요가 없다.
2) 양방향적이지 않다. Re-encryption key를 만드는데 오직 Bob의 공개키와 Alice의 비밀키가 사용된다. Re-encryption key를 역으로 사용할 수 없다.
3) Proxy의 부정을 방지한다. Proxy가 어떤 당사자의 비밀키를 도출하거나 재암호화하여 메시지를 읽을 수 없다. 즉, Proxy는 단순한 재암호화기능만 수행할 수 밖에 없다.
4) 단방향적이다. Proxy는 $g^{b/a}$로부터 $g^{a/b}$를 계산할 수 없다. 그리고 Bob의 암호문은 Alice를 위해 재암호화될 수 없다. Proxy는 재암호화기능 외에 아무런 권한이 없다.
5) 알고리즘이 결탁을 방지한다. Proxy는 Alice와 결탁한다하더라도 $RK_{A\to B}$로부터 Bob의 비밀키 b를 유추하기 어렵다.

2.3.3 Green 와 Ateniese[8]

2006년에는 Green, Ateniese[8]에 의해 Proxy Re-encryption 기능을 가지는 IBE가 제안되었다(GA방식). GA방식은 multi-use성을 가지고 있다. Multi-use성은 Alice의 암호문으로부터 Bob의 암호문으로 변환한 후에 Bob의 암호문으로부터 Chris의 암호문으로 변환할 수 있는 성질이다.

그러나, 기존의 방식은 공모 공격에 약한 결점을 가진다. 공모 공격은 Proxy와 Bob이 공모 하는 것에 의해 Alice의 비밀키를 유출하거나 다른 re-encryption key를 생성할 수가 있다. 또 GA방식의 multi-use 성에 대해 암호문을 변환할 때마다 암호문의 길이가 늘어나는 단점을 가진다.

Ⅲ. ID 기반 Proxy Re-encryption

3.1 쌍선형 사상

2개의 순회군(Cyclic Group) G_1, G_2에 대해 쌍선형 사상 $e: G_1\times G_2\to G_T$(G_T는 쌍선형 사상의 출력 공간)는 다음의 성질을 갖는다.

- 쌍선형성(bilinear) : 모든 $u\in G_1$, $v\in G_2$ 및 모든 $a,b\in Z$에 대해 $e(u^a,v^b)=e(u,v)^{ab}$가 성립된다.
- 비퇴화성(non-degenerate) : $G_x(x=1,2)$의 생성원 $g\in G_x$에 대해 $e(g,g)\neq 1$이다.
- 계산가능성(computable) : 모든 $u\in G_1$, $v\in G_2$에 대해서 $e(u,v)$를 계산하는 효율적인 알고리즘이 존재한다.

3.2 Complexity Assumptions

ID 기반 암호는 Bilinear Diffie-Hellman (BDH) Assumption 문제를 계산하는 것이 어렵다는 가정하에 랜덤 오라클 모델에서 CPA(선택평문공격)에 대해 안전하다는 것이 증명되었다. ID 기반 암호방식의 안전성 근간이 되는 BDH 가정을 검토하면 다음과 같다.

- Decisional BDH Assumption

임의로 g, g_a, g_b, $g_c\in G$, $T\in G$를 설정한다. $\{g, g_a, g_b, g_c, e(g,g)_{abc}\}$와 $\{g, g_a, g_b, g_c, T\}$를 다항식 시간내의 알고리즘에 의해 1/2 이상의 확률로 식별할 수 없다.

- Computational BDH Assumption

임의로 g, g_a, g_b, $g_c\in G$를 설정한다. 이 값보다 $e(g,g)^{abc}$를 다항식 시간내의 알고리즘에 의해 산출할

수 없다.

3.3 ID 기반 암호방식

ID 기반 암호는 1984년 A. Shamir[1]에 의해 처음 제안된 방식으로 송신자가 암호화시 수신자의 ID를 이용하여 암호화 한다. 이때, 암호문 수신자는 PKG로부터 비밀키를 도출하여 복호하는 방식이다. ID와 같이 수신자를 식별할 수 있는 공개된 정보(이메일, IP 주소 등)와 같은 사용자의 잘 알려진 정보로부터 공개키를 생성한다.

Boneh Franklin의 ID 기반 암호방식[3]은 다음 4가지 알고리즘으로 구성된다.

(1) Setup(k) : 보안 파라미터 k를 입력하여 그 값에 대응하는 공개 파라미터 $params$와 마스터 키 s를 출력하는 알고리즘
- $[q, G_1, G_2, e] \leftarrow G(k)$, $P \leftarrow G_1$, $s \leftarrow Z_q^*$, $P_{pub} = sP$을 생성한다.
- $H_1 : \{0,1\}^* \rightarrow G_1^*$고 $H_2 : G_2 \rightarrow \{0,1\}^n$이다.
- $params = \langle q, G_1, G_2, e, n, P, P_{pub}, H_1, H_2 \rangle$, 마스터 키는 s이다.

(2) KeyGen(ID, $params$, s) : 마스터 키 s와 수신자의 ID를 입력하여 그 ID에 대응하는 비밀키 d_{ID}를 출력하는 알고리즘
- 비밀키 $d_{ID} = sQ_{ID}$를 생성한다. 여기서, $Q_{ID} = H_1(ID)(\in G_1^*)$이다.

(3) Enc($params$, ID, m) : 공개 파라미터 $params$와 수신자의 ID와 평문 m을 입력하여 그 평문에 대응하는 암호문 c를 출력하는 알고리즘
- $Q_{ID} = H_1(ID)$와 $r \leftarrow Z_q^*$을 랜덤하게 선택하고 $c = \langle rP, m + H_2(g_{ID}^r) \rangle$를 계산한다. 여기서, $g_{ID} = e(Q_{ID}, P_{pub})$이다.

(4) Dec($params$, $c = \langle U, V \rangle$, d_{ID}) : 비밀키 d_{ID}와 암호문 c를 입력하여 암호문에 대응하는 평문 m (대응이 없는 경우는 $\perp$)을 출력하는 알고리즘 $m = V + H_2(e(d_{ID}, U))$

Proxy Re-encryption 기능을 가지는 ID 기반 암호의 시스템에서는 공개 파라미터와 비밀키를 생성하는 PKG, 암호문을 변환하는 Proxy, 암호문을 작성하는 사용자인 송신자, 암호문을 복호 하는 사용자인 수신자 (Alice, Bob 등)로 구성된다.

3.4 ID 기반 Proxy Re-encryption[11]

3.4.1 알고리즘 개요

문헌[11]은 Proxy Re-encryption 기능을 가지는 ID 기반 암호이다. ID 기반 암호의 Setup, KeyGen, Enc, Dec의 4개의 알고리즘에 re-encryption key generating key를 출력하는 RGKG(Re-encryption key Generating Key Generation), re-encryption key의 생성을 하는 RG (Re-encryption key Generation)와 암호문을 변환하는 Re-encrypt로부터 완성되는 7개의 알고리즘으로 구성된다. 또한, re-encryption key의 생성을 송신자가 생성하여 송신자가 암호문 복호자를 결정할 수가 있다.

(1) Setup(k) : 보안 파라미터 k를 입력하여 공개 파라미터 $params$와 master key s를 생성한다.

(2) Extract($params$, s, ID) : 공개 파라미터 $params$, master key s, 사용자 개별의 ID를 입력으로 ID에 대응하는 사용자 비밀키 d_{ID}를 생성한다, 이는 PKG나 관리자 등 신뢰할 수 있는 기관이 행한다.

(3) RGKG($params$) : 공개 파라미터 $params$를 입력하여 re-encryption key와 암호문의 생성에 사용하는 re-encryption key generating key a를 생성한다.

(4) RG(ID_A, ID_B, a) : 각 사용자의 ID_A, ID_B, 랜덤 re-encryption key generating key a를 입력하여 re-encryption key $\pi_{A \rightarrow B}^a$를 계산한다.

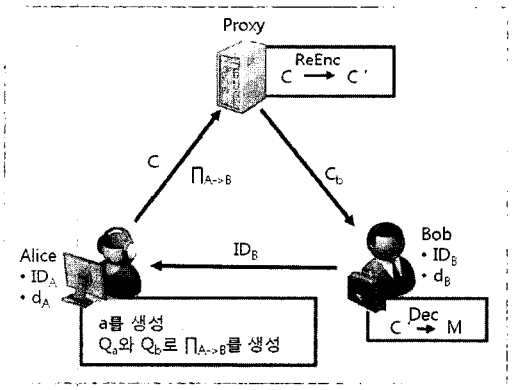
(5) Encrypt($params$, ID, a, m) : 공개 파라미터 $params$, 사용자 개별의 ID, re-encryption key generating key a, 평문 m를 입력하여 암호문을 C를 생성한다. (즉, 송신자가 RGKG에서 출력된 re-encryption key generating key a로부터 ID를 공개키로 평문 m을 암호화하여 암호문 C를 생성)

(6) Re-encrypt(C, $\pi_{A \rightarrow B}^a$) : Proxy가 암호문 C와

re-encryption key $\pi^a_{A\to B}$ 를 입력하여 암호문 C 를 C'로 변환한다. 여기서, Proxy는 암호문 변환만을 하는 기관이다.

(7) Decrypt($params$, d_{ID}, C') : 공개 파라미터 $params$, 사용자의 비밀키 d_{ID}, 암호문 C'를 입력으로서 평문 m으로 복원한다.

여기서, re-encryption key generation key는 re-encryption key나 암호문을 작성하는데 필요한 키이다. Re-encryption key generation key는 re-encryption key의 생성과 암호문의 생성에는 필요하지만, 복호에는 필요가 없다. RGKG는 종래의 Proxy Re-encryption에 없는 알고리즘이며 re-encryption key generation key를 생성한다. 또한, Encrypt는 암호문을 작성할 경우에 ID 기반 암호와 달리 공개키 ID외에 re-encryption key generation key를 필요로 한다.



〔그림 5〕 ID기반 Proxy Re-encryption

3.4.2 알고리즘 상세

(1) Setup(k):

- 보안 파라미터 k를 입력으로 위수 q인 군 G_1, $\hat{e}: G_1 \times G_1 \to G_2$를 선택
- H_1와 H_2는 해쉬 함수이며 $H_1 : \{0,1\}^* \to G_1$, $H_2 : G_2 \to \{0,1\}^n$
- $P \in G_1$를 랜덤 선택, $s \in Z_q^*$를 랜덤 선택, $P_{pub} = sP$를 계산
- 공개 파라미터 $params = \langle q, G_1, G_2, \hat{e}, P, P_{pub}, H_1, H_2 \rangle$ master key=s를 출력

(2) Extract($params$, s, ID):

- 공개 파라미터 $params$, master key s, 사용자의 $ID \in \{0,1\}^*$를 입력
- $Q_{ID} = H_1(ID) \in G_1$로부터 $d_{ID} = sQ_{ID}$를 계산해 출력

(3) RGKG($params$):

- 공개 파라미터 $params$을 입력으로 re-encryption key generating key $a \xleftarrow{R} Z_q^*$를 생성

(4) RG(ID_A, ID_B, a):

- 각 사용자의 $ID_A \in \{0,1\}^*$, $ID_B \in \{0,1\}^*$, re-encryption key generating key a를 입력
- $Q_A = H_1(ID_A) \in G_1$, $Q_B = H_1(ID_B) \in G_1$를 계산해서 re-encryption key $\pi^a_{A\to B} = -aQ_A + aQ_B$를 출력
- re-encryption key $\pi^a_{A\to B}$는 ID_A로 암호화된 암호문을 ID_B로 암호화된 암호문으로 변환할 수가 있음

(5) Encrypt($params$, ID_A, a, m):

- 공개 파라미터 $params$, $ID_A \in \{0,1\}^*$, re-encryption key generating key a, 평문 $m \in \{0,1\}^n$를 입력
- $Q_A = H_1(ID_A) \in G_1, r \xleftarrow{R} Z_q^*, G_1, k \xleftarrow{R} \in G_2$ 와 $g_A = \hat{e}(P_{pub}, ID_A)$를 계산하여
  $C_1 = rP_{pub}$
  $C_2 = arP$
  $C_3 = k \oplus g_A^{ar}$
  $C_4 = m \oplus H_2(k)$
- 암호문 $C = \langle C_1, C_2, C_3, C_4 \rangle$를 생성
- 암호문 C는 공개키 ID_A로 암호화되고 있으므로, 비밀키 d_A로 복호 할 수 있음.

(6) Re-encrypt(C, $\pi^a_{A\to B}$):

- 암호문 C, re-encryptionkey $\pi^a_{A\to B}$를 입력
- 암호문 $C = \langle C_1, C_2, C_3, C_4 \rangle$를 변환
  $C_1 = rP_{pub}$
  $C_2 = arP$
  $C_3' = C_3 \oplus \hat{e}(C_1, \pi_{A\to B})$
  $C_4 = m \oplus H_2(k)$

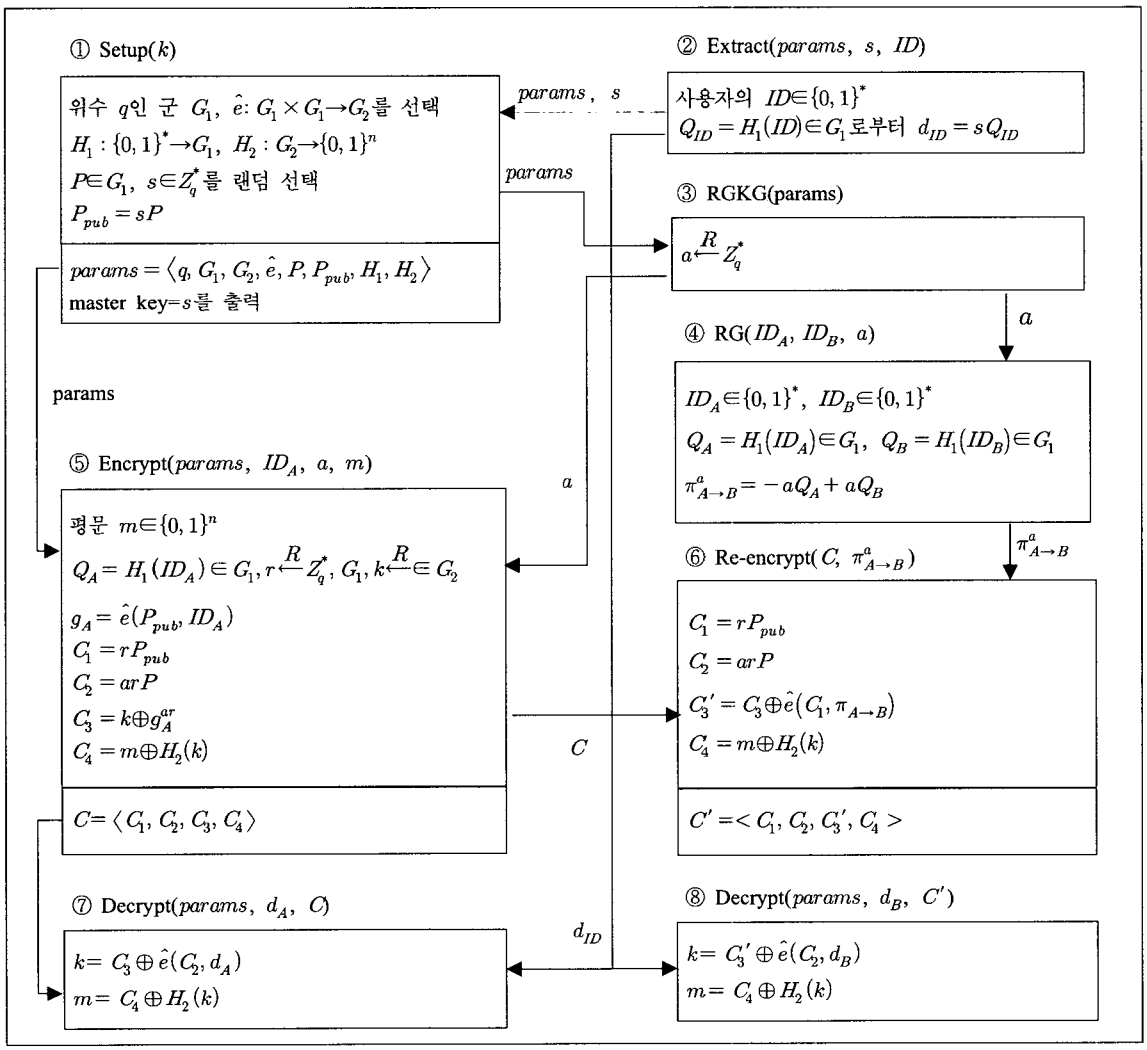
- 암호문 $C' = < C_1, C_2, C_3', C_4 >$ 를 출력

(7-1) Decrypt($params$, d_A, C):
- 공개 파라미터 $params$, 비밀키 d_A, 암호문 $C = < C_1, C_2, C_3, C_4 >$를 입력
- $k = C_3 \oplus \hat{e}(C_2, d_A)$를 계산하여 $m = C_4 \oplus H_2(k)$를 계산하는 것에 의해 평문 m을 복원

(7-2) Decrypt($params$, d_B, C'):
- 공개 파라미터 $params$, 비밀키 d_B, 암호문 $C' = < C_1, C_2, C_3', C_4 >$ 를 입력
- $k = C_3' \oplus \hat{e}(C_2, d_B)$를 계산하여 $m = C_4 \oplus H_2(k)$를 계산하는 것에 의해 평문 m을 복원
- 암호문 C의 복호에서도 암호문 C'의 복호에서도 같은 Decrypt 알고리즘을 사용

Ⅳ. 결 론

지금까지 Proxy Re-Encryption의 다양한 방법들에 대해 검토해 보았다. [표 1], [표 2]는 검토된 암호방식의 비교를 나타낸다. 본 논문에서는 데이터를 안전하게 관리하기 위해서 Proxy Re-encryption 방식을 이용하



〔그림 6〕 알고리즘 구성도

〔표 1〕 매개 변수와 키생성

방식	공통 매개변수	ID변환	공개키	복호키	재암호화키
BBS	$g \in Z_p^*$	-	$PK_a = g^a$ $PK_b = g^b$	$SK_a = a \in Z_q^*$ $SK_b = b \in Z_q^*$	$RK_{A \to B} = b/a$
DI	$g \in Z_p^*$	-	$PK = y$ $y = g^x$	$SK = x$ $x = x_1 + x_2$	$RK_{A \to B} = x_1$
AFGH	$g \in Z_p^*$	-	$PK_a = g^a$ $PK_b = g^b$	$SK_a = a \in Z_q^*$ $SK_b = b \in Z_q^*$	$RK_{A \to B} = (g^b)^{1/a}$ $= g^{b/a}$
鈴木, 齊藤	P, sP, a	$Q_A = H_1(ID_A) \in G_1$ $Q_B = H_1(ID_B) \in G_1$	-	$d_A = sQ_A$ $d_B = sQ_B$	$\pi^a_{A \to B} = -aQ_A + aQ_B$

〔표 2〕 암호화/복호화

방식	암호화	재암호화	복호화
BBS	$C_a = (g^r \cdot m, g^{ar})$	$C_b = (g^r \cdot m, (g^{ar})^{b/a})$ $= (g^r \cdot m, g^{br})$	$\frac{g^r \cdot m}{(g^{br})^{1/b}} = m$
DI	$C = (g^r, m \cdot g^{xr})$	$C' = ((g^r)^{x_1}, m \cdot g^{xr})$	$\frac{m \cdot g^{xr}}{(g^r)^x} = m$
AFGH	$C_a = (Z^r \cdot m, g^{ra})$	$C_b = (Z^r \cdot m, e(g^{ra}, g^{b/a}))$ $= (Z^r \cdot m, Z^{rb})$	$\frac{Z^r \cdot m}{(Z^{rb})^{1/b}} = m$
鈴木, 齊藤	$C_1 = rsP$ $C_2 = arP$ $C_3 = k \oplus e(sP, ID_A)^{ar}$ $C_4 = m \oplus H_2(k)$	$C_1 = rsP$ $C_2 = arP$ $C_3' = C_3 \oplus \hat{e}(C_1, \pi_{A \to B})$ $C_4 = m \oplus H_2(k)$	$C_3' \oplus \hat{e}(C_2, d_B) = k$ $C_4 \oplus H_2(k) = m$

여 암호문에 대한 복호권한을 다른 사람에게 위임함으로써 데이터를 분산관리할 때 안전하게 관리할 수 있음을 검토하였다. 향후에는 저장 데이터의 형태/종류에 따라 복호권한을 위임하는 Type and Identity 기반 방식과 속성기반 Proxy Re-encryption 방식에 대해 검토할 예정이다.

참고문헌

[1] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," In Advances in Cryptology. Crypto '84, LNCS 293, pp. 341-349, 1984.

[2] R. Sakai, K. Ohgishi and M. Kasahara. "Cryptosystems based on pairing," Symposium on Cryptography and Information Security, Japan, 2000.

[3] D. Boneh and M. Franklin, "Identity based encryption from the Weil pairing," Proc. of Crypto 2001, LNCS 2139, Springer-Verlag, pp. 213-229, 2001.

[4] M. Mambo and E. Okamoto, "Proxy cryptosystems: Delegation of the power to decrypt ciphertext," IEICE Trans. Fund Electronics Communications and Computer Science, 1997.

[5] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," In Advances in Cryptology. EUROCRYPT'98, volume 1403 of LNCS, pp. 127-144, 1998.

[6] Y. Dodis and A. Ivan, "Proxy cryptography revisited," In Network and Distributed System Security Symposium, February 2003.

[7] G. Ateniese, K. Fu, M. Green, and S. Hohenberger.

"Improved Proxy Re-encryption Schemes with Applications to Secure Distributed Storage," In Network and Distributed System Security Symposium, pp. 29-33, 2005.

[8] M. Green and G. Ateniese. "Identity-based proxy re-encryption," Cryptology ePrint Archive, Report 2006/473, 2006.

[9] 松尾俊彦. "ID ベース暗号のためのプロキシー再暗 号化システム," Symposium on Cryptography and Information Security, Japan, 2007.

[10] 扇裕和. "ID-base暗号によるデータ共有のための proxy cryptography," Symposium on Cryptography and Information Security, Japan, 2007.

[11] 鈴木秀輔, 齊藤泰一, "Proxy Re-encryption 機能をもつIDベース暗号," Symposium on Cryptography and Information Security, 2008.

[12] Jeff Dean, "Handling Large Datasets at Google: Current Systems and Future Directions," Data-Intensive Computing Symposium, 2008.

[13] Raghu Rmakrishan, "Sherpa: Cloud Comptuin of the Third Kind," Data-Intensive Computing Symposium, 2008.

[14] 구우권, 황정연, 김형중, 이동훈, "CCA 안전성을 제공하는 ID기반 프락시 재암호화 기법", 대한전자공학회지, 전자공학회논문지-CI, 제46권 제1호, pp. 64-77, 2009.

[15] D. Boneh, E.J. Goh, T. Matsuo, "Proposal for P1363.3 Proxy Re-encryption" (http://grouper.ieee.org/groups/1363/IBC/submissions/NTTDataProposalfor-P1363.3-2006-09-01.pdf).

〈著者紹介〉

송 유 진 (Youjin Song)
정회원

1982년 2월: 한국항공대학교 전자공학과 학사
1987년 8월: 경북대학교 대학원 석사
1995년 3월: 일본 Tokyo Institute of Technology(동경공업대학) 정보보호학 공학박사
1988년~1996년: 한국전자통신연구원 선임연구원
2003년~2005년: 미국 University of North Carolina at Charlotte 연구교수
2006년 7월~8월: 일본 정보보호대학원대학(IISEC) 객원교수
1996년~현재: 동국대학교 정보경영학과/대학원 교수
2005년~현재: 동국대학교 부설 전자상거래연구소 소장
1998년~현재: 한국정보보호학회 이사
2006년~현재: 국제e-비즈니스학회 이사
2006년~현재: 한국사이버테러정보전학회 이사
2001년: ICISC2001 운영위원장
2003년: 하계CISC2003 프로그램 위원장
2006년: CISC-S2006 공동 프로그램 위원장
2007년: 한국정보시스템학회 추계학술발표대회 공동 조직위원장
<관심분야> Secret Sharing, Privacy Protection, 전자상거래 응용보안 (Location Privacy, 디지털컨텐츠 보호, SCM/CRM 보안 등), Context Aware Application Security

박 광 용 (Kwangyong Park)
학생회원

2008년 2월: 동국대학교 전자상거래학과 졸업
2008년 3월~현재: 동국대학교 석사과정(전자상거래 기술전공)
<관심분야> 암호이론, 데이터베이스 보안, 유비쿼터스 프라이버시 보호