情 報 保 護 學 會 誌
第 14 券 第 3 號, 2004. 6

# NIST 800-56 키 설정 스킴에 관한 연구*

원 동 규**, 곽 진**, 주 미 리***, 양 형 규****, 원 동 호*****

요 약

네트워크 기술의 발전으로 사회 여러 분야에서 인터넷을 이용하여 다양한 서비스들이 제공되고 있다. 이러한 서비스가 널리 확산됨에 따라 인터넷 상에서 전송되는 정보의 기밀성과 무결성을 보장하기 위한 암호기술이 크게 주목을 받고 있다. 현재 널리 사용되고 있는 암호 알고리즘으로는 FIPS 197(Federal Information Processing Standard)에서 정의한 AES(Advanced Encryption Standard)와 FIPS 46-3에 채택된 Triple DES(Data Encryption Standard), FIPS 198에 정의된 HMAC 등이 있다. 이러한 알고리즘들은 시스템 사이의 상호 운용을 위해 제정되었으며, 공유 키 재료(Shared keying material) 설정이 사전에 이뤄져야 한다. 키 재료의 설정은 신뢰기관에 의해 분배가 가능하나, 객체의 수가 증가함에 따라 키 재료 분배 작업이 지수적으로 증가하게 되는 문제점이 있다. 그러므로, 본고에서는 이러한 키 재료 분배의 문제점과 효율적인 키 설정 스킴에 대하여 기술하고 있는 NIST 800-56을 분석하고자 한다.

## Ⅰ. 서 론

정보통신기술의 발전은 사회·경제·생활속에서 다양한 형태로 영향을 끼치게 되었다. 특히, 실생활에서의 상거래 및 금융서비스들이 사이버 공간(인터넷)에서 전자상거래·인터넷 뱅킹·전자결제서비스 등으로 제공됨에 따라 사용자들에게는 효율성과 편리성을 제공하게 되었다. 그러나, 이러한 서비스들은 사이버공간의 비대면적인 특성으로 인해 사용자 정보의 노출 및 위조 그리고 전송되는 정보의 변조와 같은 역기능들이 발생하게 되었다. 현재, 역기능들을 방지하기 위하여 전송정보의 무결성 및 기밀성, 사용자 인증, 부인방지를 제공하는 암호기술이 널리 사용되고 있다.

암호기술에 사용되는 암호 알고리즘으로는 AES와 3DES (Triple DES), HMAC 등이 있으며, 암호 알고리즘을 사용하기 위해서는 먼저 암호통신을 하고자하는 시스템들간의 키 재료 설정과정이 선행되어야만 한다. 키를 설정하기 위한 방법들은 여러 표준(ANSI, IEEE, FIPS)에서 정의하고 있으며, 최근 ANSI X9. 표준을 기반으로 한 NIST 800-56이 발표되었다. 이에 본고에서는 NIST 800-56에서 발표한 키 설정 스킴들을 살펴보고, 그 중에서도 키 동의 방식을 중심으로 분석하고자 한다.

- ANSI X9.42 : 이산대수문제를 기반으로 하는 키 동의
- ANSI X9.63 : 타원곡선을 기반으로 하는 키 동의와 키 전송
- ANSI X9.44 : 소인수 분해 문제를 기반으로 하는 키 동의와 키 전송

본 고의 구성은 다음과 같다. 먼저 제 2장에서는 용어정리와 키 설정 스킴의 기반이 되는 암호학적 요소에 대하여 살펴보고, 제 3장에서는 키 동의 스킴에 대해 분석한다. 그리고 마지막으로 4장에서 결론을 맺는다.

## Ⅱ. 암호학적 요소

본 장에서는 용어정의와 키를 생성하는데 사용되는 도메인 파라미터(domain parameter), 개인·공개

* 본 연구는 대학 IT 연구센터 육성·지원 사업의 연구결과로 수행되었음
** 성균관대학교 정보통신공학부 정보통신보호연구실 ({dkwon, jkwak}@dosan.skku.ac.kr)
*** 국가보안기술 연구소 (mrjoo@etri.re.kr)
**** 강남대학교 컴퓨터미디어 공학부 부교수 (hkyang@kangnam.ac.kr)
***** 성균관대학교 정보통신공학부 정교수 (dhwon@dosan.skku.ac.kr)

키 쌍의 생성과 검증방법, 키 유도함수(KDF : Key Derivation Function), 프리미티브(primitive) 등에 관하여 기술한다.

## 1. 용어정의

- 키 설정(key establishment)
  : 두 개체에 의해 공유 키 재료를 설정하는 과정으로 키 전송이나 키 동의에 의해 만들어진다.
- 키 재료(key material)
  : 키, 초기 벡터, 정보로써 암호 키 관계를 유지하거나 설정하기 위해 필요한 데이터
- 공유 비밀(shared secret)
  : 키 동의 스킴에 의해 계산되는 값으로써 키 유도 함수의 입력으로 사용된다.
- 고정 키(static key)
  : long term 키로써 다른 암호 키 설정구조에 사용된다.
- 일회용 키(ephemeral key)
  : 다른 암호 키를 생성하기 위해 바로 전에 생성하여 사용하고, 다른 암호 키 생성 후에 삭제되는 short term 키
- 공유 키 재료(shared keying material)
  : 공유 비밀이나 정보를 위해 키 유도 함수에 적용하여 유도된 키 재료

## 2. 도메인 파라미터

도메인 파라미터는 공개정보로써, 생성된 키 쌍과 정확히 상응되도록 관리·유지되어야 하며, 하나의 도메인 파라미터 집합은 여러 개의 키 설정 스킴에 사용될 수 있다. 유한체(FFC : Finite Field Cryptography)와 타원곡선(ECC : Elliptic Curve Cryptography)의 도메인 파라미터를 살펴보면 다음과 같다.

### 2.1 유한체 상의 도메인 파라미터

FFC 스킴을 위한 도메인 파라미터는 $p$, $q$, $g$, [*SEED*, *pgenCounter*]의 형태로 구성된다.

- $p$ : large prime field order
- $q$ : the prime subgroup order
- $g$ : $GF(p)^*$상의 $q$를 위수(order)로하는 순환 서브그룹의 원시원소
- *SEED*, *pgenCounter* : $p$, $q$를 생성·검증과정에서 사용되는 값

〔표 1〕 FFC Equivalent Strengths

| 비트의 안전성 | 80 | 112 | 128 | 192 | 256 |
|---|---|---|---|---|---|
| $q$의 비트 길이 | 160 | 224 | 256 | 384 | 512 |
| $p$의 비트길이 | 1024 | 2048 | 3072 | 8192 | 15360 |

$p$, $q$를 선택하는데 미 연방 정부는 5단계로 구성된 안전수준을 정하고 시간·용량 그리고 안전성을 고려하여 파라미터를 선택하도록 하였다. 표 1은 5단계의 안전수준을 표로 나타낸 것이다. 이는 선택된 $p$, $q$를 사용하여 생성한 하나의 키를 깨기 위해선 현재 알려진 최선의 공격방법으로 대략 $2^{80}$, $2^{112}$, $2^{128}$, $2^{192}$, $2^{256}$번의 연산을 수행해야 함을 나타내고 있다. $p$, $q$를 생성하고, 검증하는 방식으로 Shawe-Taylor algorithm을 사용한다.

### 2.2 타원곡선 상의 도메인 파라미터

ECC 스킴을 위한 도메인 파라미터는 $q$, *FR*, $a$, $b$, [*SEED*], $G$, $n$, $h$의 형태로 구성된다.

- $q$ : field size
- *FR* : indication of the basis
- $a, b$ : field elements
- *SEED* : 비트 문자(랜덤하게 생성)
- $G$ : 원시원소($x_G$, $y_G$)
- $n$ : $G$의 위수(order)
- $h$ : 인수(cofactor)

타원곡선의 경우, 유한체 상에서와 마찬가지로 5단계의 안전수준을 선택하여 사용하며, 이는 유한체 상의 비트 안전성과 동일하다. 미 연방 정부에서 사용되는 타원곡선의 인수는 65,536과 같거나 작다.

〔표 2〕 ECC Equivalent Strengths

| 비트의 안전성 | 80 | 112 | 128 | 192 | 256 |
|---|---|---|---|---|---|
| $n$의 최소 비트 길이 | 160 | 224 | 256 | 384 | 512 |
| ECC 인수 $h$의 최대 값 | 65,536 | 65,536 | 65,536 | 65,536 | 65,536 |

### 2.3 도메인 파라미터 검증

개체에 의해 사용되는 도메인 파라미터 집합의 수학적 검증은 안전한 키의 사용과 연관 있다. 그러므로 사용할 도메인 파라미터에 대해 아래와 같은 방법 중

최소한 하나의 방법을 통해 검증해야 한다.

① 정의된 요구사항에 맞는 도메인 파라미터를 생성한다.
② 정의된 도메인 파라미터 검증을 수행한다.
  a. FIPS 186-3 : 키의 크기 검증
  b. ANSI X69 : 인수, 키의 크기 검증
③ 제 삼의 신뢰기관으로부터 ①, ②의 방법으로 검증된 도메인 파라미터를 사용한다.

### 3. 개인키/공개키

고정·일회용 키는 똑같은 프리미티브를 사용하여 생성된다. 유한체 상에서 개인·공개키는 위에서 정의한 $p$, $q$, $g$, [ *SEED*, *pgenCounter*]의 도메인 파라미터를 사용하여 생성하는데, 개인키는 승인된 랜덤 수 생성기를 사용하여 통계적으로 유일하고 예측할 수 없는 값으로 선택된다. 타원곡선 상에서는 $q$, *FR*, $a$, $b$, [ *SEED*], $G$, $n$, $h$의 도메인 파라미터를 사용하여 개인·공개키를 생성한다.

공개키는 안전한 키 설정을 위해 수학적 검증이 요구된다. 공개키 검증방식은 공개키를 생성한 소유자(owner) 혹은 공개키의 수신자(recipient)인지에 따라 다르며, 이를 살펴보면 다음과 같다.

■ 소유자의 공개키 검증

① Owner Full Validation
  : 소유자가 공개키 검증을 수행한다.
② TTP(Trusted Third Party) Full Validation
  : 소유자는 제 삼의 신뢰기관으로부터 공개키 검증 수행을 보장을 받는다.
③ Owner Generation
  : 개인키로부터 공개키를 생성한다.
④ TTP Generation
  : 제 삼의 신뢰기관이 개인·공개키 쌍을 생성하고 소신자에게 전송해 준다.

■ 수신자의 공개키 검증

① Recipient Full Validation
  : 수신자가 공개키 검증을 수행한다.
② TTP Full Validation
  : 수신자는 제 삼의 신뢰기관으로부터 공개키 검증을 보장받는다.
③ TTP Generation
  : 제 삼의 신뢰기관이 개인·공개키 쌍을 생성하고 수신자에게 전송해 준다.

공개키의 검증 방식은 공개키가 고정 공개키(static public key)인지 일회용 공개키(one-time public key)인지에 따라서 검증방식에 약간의 차이가 있다. 고정 공개키의 경우 앞에서 언급한 방식으로 검증되고, 일회용 공개키의 경우에는 소유자와 수신자에 따라 다음과 같은 방법으로 수행된다. 소유자의 경우 소유자 자신이 직접 키를 생성했기 때문에 공개키에 대한 검증이 보장된다. 수령자의 일회용 공개키 검증 방식의 경우 유한체와 타원곡선으로 다시 분류되는데, 수신자의 공개키 검증 방식의 ①, ②번은 그대로 사용하고, 타원곡선상에서는 수신자와 TTP가 부분적 공개키 검증 방식을 사용하여 검증하는 방법을 사용한다.

유한체와 타원곡선 상의 공개키 검증방식을 살펴보면 아래와 같다.

■ 유한체 상에서의 공개키 검증

· 입력
① ( $p$, $q$, $g$, [ *SEED*, *pgenCounter*])
② $y$ : FFC 공개키

· 과정
① $2 \le y \le p-2$ 검증
② $y^q = 1 \bmod p$ 검증

■ 타원곡선 상에서의 공개키 검증

· 입력
① ( $q$, *FR*, $a$, $b$, [ *SEED*], $G$, $n$, $h$)
② $Q' = (x_Q', y_Q')$ : ECC 공개키

· 과정
① $Q'$가 무한 0점에 존재하지 않음을 검증
② $x_Q'$와 $y_Q'$가 [0, $p-1$]사이의 정수인지, $q=p$은 소수인지, $x_Q'$와 $y_Q'$가 $q=2^m$의 경우 $m$ 비트 길이를 가지는지 검증
③ 만약 $p=q$인 소수라면,
$(y_Q')^2 \equiv (x_Q')^3 + ax_Q' + b \pmod p$ 검증

$q=2^m$이라면, $GF(2^m)$에서 아래 식 검증

$(y_Q')^2+x_Q'y_Q'=(x_Q')^3+a(x_Q')^2+b$

④ $nQ'=O$ 검증

타원곡선 상에서의 공개키 검증 방식에서 ④번 과정은 생략할 수 있다.(타원곡선 상에서의 부분적 공개키 검증 방식)

생성된 개인·공개키 쌍은 다음의 요구사항에 따라 관리·유지되어진다.

■ 고정·일회용 키 쌍에 대한 일반적인 요구사항

① 개인·공개키 쌍은 정의된 도메인 파라미터와 정확히 연관을 가져야 하며, 키 쌍은 하나의 파라미터만을 사용해야 한다.
② 각 개인키는 랜덤하며, 예측할 수 없어야 한다.
③ 개인키는 노출, 수정 등에 대한 공격에 보호되어야 한다.
④ 공개키는 수정, 치환 등에 대해 보호되어야 한다.

■ 고정 키 쌍에 대한 요구사항

① 일회용 키를 생성하기 이전에 고정 키를 생성해야 한다.
② 고정 키의 수령자는 공개키와 도메인 파라미터 사이의 관련사항을 보장해야 한다. → 신뢰기관이 서명한 공개키 인증서 검증
③ 고정 키는 여러 키 설정 스킴에서 사용 될 수 있다. 하지만 개인·공개키 쌍은 다른 목적으로 사용 될 수 없다.
④ 고정 키는 수령자가 신뢰할 수 있어야 한다. → 소유자는 신뢰기관의 인증서를 가지고 있어야 함
⑤ 소유자와 수령자는 공개키 검증을 수행할 수 있어야 한다. → 공개키 인증서
⑥ 소유자와 수령자는 고정 공개키에 상응하는 개인키를 소유하고 있음을 확인할 수 있어야 한다.

■ 일회용 키 쌍에 대한 요구사항

① 일회용 개인키는 한번, 암호 키 설정 프리미티브의 계산에 사용된다.
② 일회용 키 쌍은 사용되기 바로 직전에 생성되어야 한다.
③ 공유비밀(shared secret)이 생성된 후에 바로 파기해야 한다.
④ 수령자는 일회용 공개키를 검증할 수 있어야 한다.

## 4. 키 유도 함수 (KDF)

키 유도 함수는 공유비밀로부터 키 재료를 유도하기 위한 수단으로 사용된다. NIST 800-56에서는 키 유도 함수로서 두 가지 방법을 정의하고 있다.

### 4.1 Concatenation 키 유도 함수

· 함수 요청 : *kdf*( *Z*, OtherInput)

※ OtherInput : *U*, *V*, *keydatalen*, *hashlen*, [*SharedInfo*]

· 입력

a. *Z* : 공유비밀의 비트열
b. *U*, *V* : 참가 객체들의 ID
c. *keydatalen*
: 키 재료의 길이, $hashlen\times(2^{32}-1)$
d. *hashlen*
: 키 재료를 유도하기 위한 해쉬의 비트 길이
e. *SharedInfo*
: 옵션 비트 문자열로써 각 객체가 공유한 데이터로 구성

· 과정

a. 초기 32비트, 끝의 문자열 $00000001_{16}$
b. $j=\lceil keydatalen/hashlen \rceil$
c. $i=1$부터 $j$까지
c.1 $Hash_i=H(Z\|counter\|U\|V\|[SharedInfo])$
c.2 *counter* 증가
d. $(keylen/hashlen)$이 정수면 $Hhash=Hash_j$,
아닐 경우,
$Hash_j$의 $(keydatalen-(hashlen\times(j-1)))$ 좌측 비트로 설정
e. 유도된 키 재료 = $Hash_1\|\|..\|\|Hash_{j-1}\|\|Hhash$

### 4.2 ASN.1 키 유도 함수

- ASN.1 DER encoding 기반의 키 유도함수
- *hashlen* : 해쉬함수의 출력 길이
max *hashlen* : 해쉬함수 입력의 최대 길이

· 함수 요청 : *kdf*(*Z*, *OtherInput*)

※ *OtherInput* : *keydatalen*, *hashlen*, *OtherInfo*
*OtherInfo* : *AlgorithmID*, *counter*, *PartyUInfo*, *PartyVInfo*, [*SuppPrivInfo*], [*SuppPubInfo*]

- 입력
a. $Z$ : 공유비밀의 비트열
b. *keydatalen*
: 키 재료 길이인 정수, *hashlen*×($2^{32}-1$)
c. *OtherInfo*
: ASN.1 DER encoding에서 정의한 비트열
c.1 키 설계 정보
c.1.1 *AlgorithmID*
: 예) 120-bit AES, 80-bit HMAC key
c.1.2 *counter*
: 32비트 8문자열으로 초기값은 $00000001_{16}$
c.2 *PartyUInfo*
: initiator의 공개정보 비트열
c.3 *PartyVInfo*
: recipient의 공개정보 비트열
c.4 (Optional) *SuppPrivInfo*
: 상호객체가 알고있는 비밀정보
c.5 (Optional) *SuppPubInfo*
: 상호객체가 알고있는 공개정보

- 과정 : 키 유도함수 계산
a. $d = \lceil keydatalen/hashlen \rceil$
b. 초기 $counter = 00000001_{16}$
c. $i$=1부터 $d$
c.1 $h_i = H(Z||OtherInfo)$
c.2 *counter*를 정수로 변경
c.3 *counter* 증가
c.4 *counter*를 8진수로 변경
d. $DerivedKeyingMaterial = h_1||h_2||...||h_d$의 *keydatalen* 왼쪽 비트열

## 5. DLC 프리미티브(Primitive)

### 5.1 FFC DH 프리미티브

유한체와 Diffie-Hellman 알고리즘을 기반한 dhHybrid1, dhEphem, dhHybridOneFlow, dhOneFlow, dhStatic 스킴에 사용된다.

- 입력
a. ($p, q, g,$ [*SEED*, *pgenCounter*])
: 도메인 파라미터
b. $x_A$ : 한 객체의 개인키
c. $y_B$ : 다른 객체의 공개키

- 과정
a. $Z = y_B^{x_A} \bmod p$
b. $Z$=1이면 실패
c. $Z$ 출력

### 5.2 ECC CDH 프리미티브

타원곡선과 Diffie-Hellman 알고리즘을 기반한 Full Unified Model, Ephemeral Unified Model, One-Pass Unified Model, One-Pass Diffie-Hellman, Static Unified Model 스킴에 사용된다.

- 입력
a. ($p, FR, a, b,$ [*SEED*], $G, n, h$)
: 도메인 파라미터
b. $d_A$ : 한 객체의 개인키
c. $Q_B$ : 다른 객체의 공개키

- 과정
a. $P = hd_AQ_B$를 계산한다.
b. $P$=0이면 실패
c. $Z = x_P$, $x_P$는 $P$의 x좌표

### 5.3 FFC MQV 프리미티브

유한체와 MQV 알고리즘에 기반한 MQV2, MQV1 스킴에 사용된다.

- 입력
a. ($p, q, g,$ [*SEED*, *pgencounter*])
: 도메인 파라미터
b. $x_A$ : 자신의 고정 개인키
c. $y_B$ : 다른 객체의 고정 공개키
d. $r_A$ : 자신의 두 번째 개인키
e. $t_A$ : 자신의 두 번째 공개키
f. $t_B$ : 다른 객체의 두 번째 공개키

· 과정

a. $w = \lceil \|q\|/2 \rceil$

b. $T_A = (t_A \bmod 2^w) + 2^w$

c. $S_A = (r_A + T_A x_A) \bmod q$

d. $T_B = (t_B \bmod 2^w) + 2^w$

e. $Z = ((t_B(y_B{}^{T_B}))^{S_A}) \bmod P$

f. $Z = 1$이면 실패

### 5.4 ECC MQV 프리미티브

타원곡선과 MQV 알고리즘에 기반한, Full MQV, One-Pass MQV 스킴에 사용된다.

· 입력

a. $(q, FR, a, b, [SEED], G, n, h)$
: 도메인 파라미터

b. $d_{s,A}$ : A의 고정 개인키

c. $Q_{s,B}$ : B의 고정 공개키

d. $d_{e,A}$ : A의 두 번째 개인키

e. $Q_{e,A}$ : A의 두 번째 공개키

f. $Q_{e,B}$ : B의 두 번째 공개키

· 과정

a. $implicitsig_A = (d_{e,A} + avf(Q_{e,A}) d_{s,A}) \bmod n$

b. $P = H(implicitsig_A)(Q_{e,B} + avf(Q_{e,B}) Q_{e,B})$

c. $P = 0$이면 실패

d. $Z = x_P$

## Ⅲ. 키 동의(Key Agreement)

본 표준에서 정의한 키 동의 스킴은 크게 세 가지로 구분된다. 키 동의에 참여한 각 개체가 일회용 키 쌍을 생성하는 Two Ephemeral keys 스킴과 키 동의를 시작하는 한 개체만이 일회용 키를 생성하는 One Ephemeral keys 스킴, 마지막으로 일회용 키들을 사용하지 않는 Zero Ephemeral keys 스킴이다. 위의 세 가지 카테고리는 위의 표 3과 같이 서브 카테고리(Subcategory)로 나누어진다.

### 1. 두 개의 일회용 키를 사용하는 스킴, C(2)

각 객체는 공유비밀을 유도하기 위해서 일회용 키 쌍을 생성하고, 그 공개키를 이용하여 키를 설정하는 스킴이다. 이 스킴은 참여한 객체가 고정 키와 일회용

〔표 3〕 키 동의 스킴

| Category | Subcategory |
|---|---|
| C(2): 두 개의 일회용 키 | C(2,2)<br>: 각 객체가 고정 키 쌍과 일회용 키 쌍을 생성 |
| | C(2,0)<br>: 각 객체는 일회용 키 쌍만을 생성 |
| C(1): 하나의 일회용 키 | C(1,2)<br>: initiator는 일회용 키 쌍을 생성한다. initiator와 responder는 고정 키 쌍을 가진다. |
| | C(1,1)<br>: initiator는 일회용 키 쌍을 생성하지만 고정 키는 가지고 있지 않고, responder는 오직 고정 키 만을 가진다. |
| C(0): 일회용 키를 사용 안함 | C(0,2)<br>: 각 객체는 오직 고정 키만을 가진다. |

〔표 4〕 dhHybrid1 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | · 고정 개인키 $x_U$<br>· 고정 공개키 $y_U$ | · 고정 개인키 $x_V$<br>· 고정 공개키 $y_V$ |
| 일회용 데이터 | · 일회용 개인키 $r_U$<br>· 일회용 공개키 $t_U$ | · 일회용 개인키 $r_V$<br>· 일회용 공개키 $t_V$ |
| 입력 | $(p, q, g)$,<br>$x_U, y_V, r_U, t_V$ | $(p, q, g)$,<br>$x_V, y_U, r_V, t_U$ |
| 계산 | $Z_s$<br>: U의 고정 개인키와 V의 고정 공개키를 사용하여 FFC DH로부터 계산<br><br>$Z_e$<br>: U의 일회용 개인키와 V의 일회용 공개키를 사용하여 FFC DH로부터 계산<br><br>$Z = Z_e \| Z_s$ | $Z_s$<br>: V의 고정 개인키와 U의 고정 공개키를 사용하여 FFC DH로부터 계산<br><br>$Z_e$<br>: V의 일회용 개인키와 U의 일회용 공개키를 사용하여 FFC DH로부터 계산<br><br>$Z = Z_e \| Z_s$ |
| 키재료 유도 | $kdf(Z, OtherInput)$ | $kdf(Z, OtherInput)$ |

| 객체 U | | 객체 V |
|---|---|---|
| | U의 고정 공개키, 일회용 공개키 → | |
| | ← V의 고정 공개키, 일회용 공개키 | |
| ① 자신의 고정·일회용 개인키와 객체 V의 고정·일회용 공개키를 사용 공유비밀을 생성 | | ① 자신의 고정·일회용 개인키와 객체 U의 고정·일회용 공개키를 사용 공유비밀을 생성 |
| ② 공유비밀을 사용하여 키 유도함수로부터 공유 키 재료를 생성한다. | | ② 공유비밀을 사용하여 키 유도함수로부터 공유 키 재료를 생성한다. |

〔그림 1〕 일반적인 C(2,2) 키 동의 스킴

키를 모두 사용하는 C(2,2)와 일회용 키 쌍만을 생성하여 사용하는 C(2,0)로 나눠진다.

### 1.1 C(2,2) 스킴

객체는 같은 도메인 파라미터로 키 쌍을 생성한다. 그림 1은 고정 키와 일회용 키를 모두 사용하는 C(2,2) 키 동의 스킴을 나타낸다. NIST 문서에서 언급한 C(2,2)스킴은 다음과 같다.

- dhHybrid1, C(2,2, FFC DH)

각 객체는 고정 키를 가지고 있고, 고정 키에 사용된 동일한 파라미터를 사용하여 일회용 키를 생성한다. 생성한 고정·일회용 공개키를 서로 나누어 가지고 키 동의 스킴을 수행한다. 각 공개키에 대한 인증서가 있어야 한다.

- Full Unified Model, C(2,2, ECC DH)

위의 dhHybrid1과 방법은 동일하다. Full Unified Model 키 동의 스킴은 표 5와 같다.

〔표 5〕 Full Unifeid Model 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | • 고정 개인키 $d_{s,U}$<br>• 고정 공개키 $Q_{s,U}$ | • 고정 개인키 $d_{s,V}$<br>• 고정 공개키 $Q_{s,V}$ |
| 일회용 데이터 | • 일회용 개인키 $d_{e,U}$<br>• 일회용 공개키 $Q_{e,U}$ | • 일회용 개인키 $d_{e,V}$<br>• 일회용 공개키 $Q_{e,V}$ |
| 입력 | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,U}$, $Q_{s,V}$, $d_{e,U}$, $Q_{e,V}$ | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,V}$, $Q_{s,U}$, $d_{e,V}$, $Q_{e,U}$ |
| 계산 | $Z_s$<br>: U의 고정 개인키와 V의 고정 공개키를 사용한 ECC CDH로부터 계산<br>$Z_e$<br>: U의 일회용 개인키와 V의 일회용 공개키를 사용한 ECC CDH로부터 계산<br>$Z = Z_e \| Z_s$ | $Z_s$<br>: V의 고정 개인키와 U의 고정 공개키를 사용한 ECC CDH로부터 계산<br>$Z_e$<br>: V의 일회용 개인키와 U의 일회용 공개키를 사용한 ECC CDH로부터 계산<br>$Z = Z_e \| Z_s$ |
| 키재료 유도 | $kdf(Z, OtherInput)$ | $kdf(Z, OtherInput)$ |

- MQV2, C(2,2, FFC MQV)

ANSI X9.42에서 정의하고 있으며, 위의 방식과 동일한 사전 과정을 수행한다. MQV2, C(2,2, FFC MQV) 키 동의 스킴은 아래 표 6과 같다.

- Full MQV, C(2,2, ECC MQV)

ANSI X9.63에서 정의하고 있으며, 위의 방식과 동일한 사전 과정을 수행한다. Full MQV, C(2,2, FFC MQV) 키 동의 스킴은 표 7과 같다.

### 1.2 C(2,0) 스킴

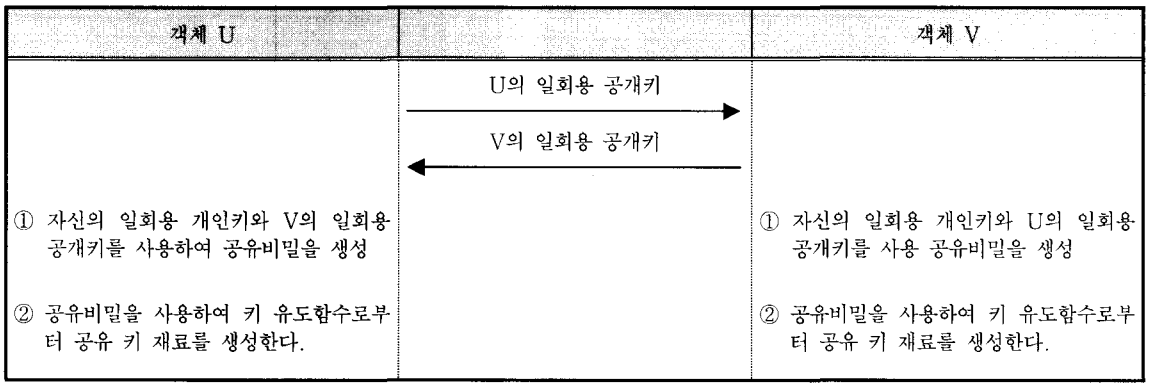
키 동의 스킴에 참여한 객체는 같은 도메인 파라미터로 키 쌍을 생성한다. C(2,0) 스킴은 고정 키를 사용하지 않고 일회용 키만을 사용하여 Diffie-

〔표 6〕 MQV2 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | • 고정 개인키 $x_U$<br>• 고정 공개키 $y_U$ | • 고정 개인키 $x_V$<br>• 고정 공개키 $y_V$ |
| 일회용 데이터 | • 일회용 개인키 $r_U$<br>• 일회용 공개키 $t_U$ | • 일회용 개인키 $r_V$<br>• 일회용 공개키 $t_V$ |
| 입력 | $(p, q, g)$, $x_U, y_V, r_U, t_V, t_U$ | $(p, q, g)$, $x_V, y_U, r_V, t_U, t_V$ |
| 계산 | $Z$<br>: U의 고정 개인키, V의 고정 공개키, U의 일회용 (개인·공개키) V의 일회용 공개키를 사용한 FFC MQV로부터 계산 | $Z$<br>: V의 고정 개인키, U의 고정 공개키, V의 일회용 (개인·공개키) U의 일회용 공개키를 사용한 FFC MQV로부터 계산 |
| 키재료 유도 | *kdf(Z, OtherInput)* | *kdf(Z, OtherInput)* |

〔표 7〕 Full MQV 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | • 고정 개인키 $d_{s,U}$<br>• 고정 공개키 $Q_{s,U}$ | • 고정 개인키 $d_{s,V}$<br>• 고정 공개키 $Q_{s,V}$ |
| 일회용 데이터 | • 일회용 개인키 $d_{e,U}$<br>• 일회용 공개키 $Q_{e,U}$ | • 일회용 개인키 $d_{e,V}$<br>• 일회용 공개키 $Q_{e,V}$ |
| 입력 | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,U}$, $Q_{s,V}$, $d_{e,U}$, $Q_{e,V}$, $Q_{e,U}$ | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,V}$, $Q_{s,U}$, $d_{e,V}$, $Q_{e,U}$, $Q_{e,V}$ |
| 계산 | $Z$<br>: U의 고정 개인키, V의 고정 공개키, U의 일회용 (개인·공개키) V의 일회용 공개키를 사용한 ECC MQV로부터 계산 | $Z$<br>: V의 고정 개인키, U의 고정 공개키, V의 일회용 (개인·공개키) U의 일회용 공개키를 사용한 ECC MQV로부터 계산 |
| 키재료 유도 | *kdf(Z, OtherInput)* | *kdf(Z, OtherInput)* |




〔그림 2〕 일반적인 C(2,0) 키 동의 스킴

〔표 8〕 dhEphem 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | 없음 | 없음 |
| 일회용 데이터 | • 일회용 개인키 $r_U$<br>• 일회용 공개키 $t_U$ | • 일회용 개인키 $r_V$<br>• 일회용 공개키 $t_V$ |
| 입력 | $(p, q, g)$, $r_U$, $t_V$ | $(p, q, g)$, $r_V$, $t_U$ |
| 계산 | $Z$<br>: U의 일회용 개인키, V의 일회용 공개키를 사용하여 FFC DH로부터 계산 | $Z$<br>: V의 일회용 개인키, U의 일회용 공개키를 사용하여 FFC DH로부터 계산 · |
| 키재료 유도 | *kdf(Z, OtherInput)* | *kdf(Z, OtherInput)* |

Hellman 키 분배를 수행한다. 그림 2은 C(2,0) 키 동의 스킴이다.

■ dhEphem, C(2,0, FFC DH)

ASNI X9.42에서 정의한 방식으로 일회용 키 쌍만을 생성하여 키 동의 스킴을 수행한다.

■ dhEphem Unified Model, C(2,0, ECC CDH)

ASNI X9.63에서 정의한 방식으로 일회용 키 쌍만을 생성하여 키 동의 스킴을 수행한다.

〔표 9〕 Ephemeral Unified Model 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | 없음 | 없음 |
| 일회용 데이터 | · 일회용 개인키 $d_{e,U}$<br>· 일회용 공개키 $Q_{e,U}$ | · 일회용 개인키 $d_{e,V}$<br>· 일회용 공개키 $Q_{e,V}$ |
| 입력 | $(q, FR, a, b, [SEED], G, n, h)$, $d_{e,U}$, $Q_{e,V}$ | $(q, FR, a, b, [SEED], G, n, h)$, $d_{e,V}$, $Q_{e,U}$ |
| 계산 | $Z$<br>: U의 일회용 개인키, V의 일회용 공개키를 사용한 ECC CDH로부터 계산 | $Z$<br>: V의 일회용 개인키, U의 일회용 공개키를 사용한 ECC CDH로부터 계산 |
| 키재료 유도 | $kdf(Z, OtherInput)$ | $kdf(Z, OtherInput)$ |

## 2. 한 개의 일회용 키를 사용하는 스킴, C(1)

키 동의 스킴을 시작하는 객체만이 일회용 키 쌍을 생성하고, 그 공개키를 다른 객체에게 전송해 주는 방식이다. 이 스킴은 참여한 객체가 고정 키와 일회용 키를 모두 사용하는 C(1,2)와 초기 시작 객체는 일회용 키 쌍만을 생성하여 사용하고, 다른 객체는 고정 키 만을 사용하는 C(1,1)로 나눠진다.

### 2.1 C(1,2) 스킴

객체는 같은 도메인 파라미터로 키 쌍을 생성한다. 그림 3은 스킴을 처음 시작하는 객체만이 일회용 키를 사용하는 C(1,2) 키 동의 스킴을 나타내고 있다.

〔표 10〕 dhHybridOneFlow 키 동의 스킴

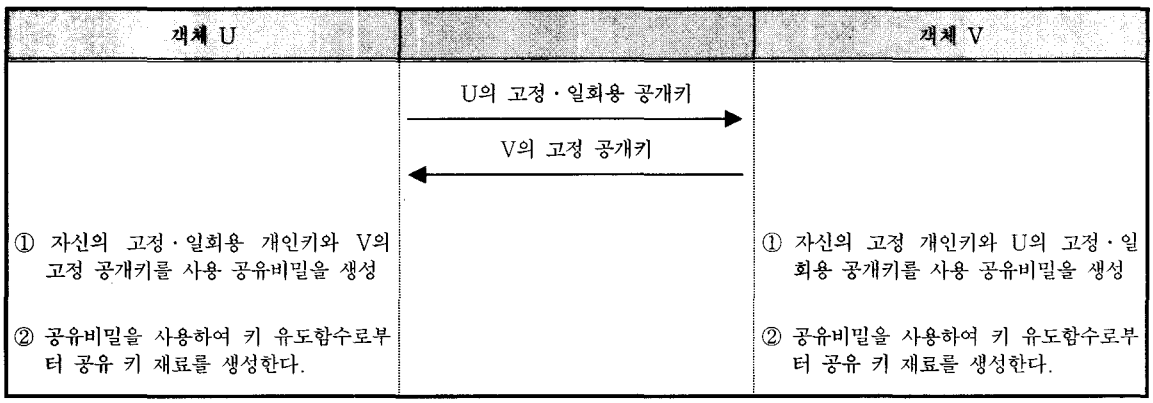
| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | · 고정 개인키 $x_U$<br>· 고정 공개키 $y_U$ | · 고정 개인키 $x_V$<br>· 고정 공개키 $y_V$ |
| 일회용 데이터 | · 일회용 개인키 $r_U$<br>· 일회용 공개키 $t_U$ | 없음 |
| 입력 | $(p, q, g)$, $x_U, y_V, r_U$ | $(p, q, g)$, $x_V, y_U, t_U$ |
| 계산 | $Z_s$<br>: U의 고정 개인키와 V의 고정 공개키를 사용한 FFC DH로부터 계산<br>$Z_e$<br>: U의 일회용 개인키와 V의 고정 공개키를 사용한 FFC DH로부터 계산<br>$Z = Z_e \| Z_s$ | $Z_s$<br>: V의 고정 개인키와 U의 고정 공개키를 사용한 FFC DH로부터 계산<br>$Z_e$<br>: V의 일회용 개인키와 U의 일회용 공개키를 사용한 FFC DH로부터 계산<br>$Z = Z_e \| Z_s$ |
| 키재료 유도 | $kdf(Z, OtherInput)$ | $kdf(Z, OtherInput)$ |

■ dhHybridOneFlow, C(1,2, FFC DH)

ASNI X9.42에서 정의한 방식으로 일회용 키 쌍만을 생성하여 키 동의 스킴을 수행한다.

■ One-Pass Unified Model, C(1,2, ECC CDH)

ASNI X9.63에서 정의한 방식으로 기본 구조는




〔그림 3〕 일반적인 C(1,2) 키 동의 스킴

〔표 11〕 One-Pass Unified Model 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | · 고정 개인키 $d_{s,U}$<br>· 고정 공개키 $Q_{s,U}$ | · 고정 개인키 $d_{s,V}$<br>· 고정 공개키 $Q_{s,V}$ |
| 일회용 데이터 | · 일회용 개인키 $d_{e,U}$<br>· 일회용 공개키 $Q_{e,U}$ | 없음 |
| 입력 | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,U}$, $Q_{s,V}$, $d_{e,U}$ | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,V}$, $Q_{s,U}$, $Q_{e,U}$ |
| 계산 | $Z_s$<br>: U의 고정 개인키와 V의 고정 공개키를 사용한 FFC DH로부터 계산<br>$Z_e$<br>: U의 일회용 개인키와 V의 고정 공개키를 사용한 FFC DH로부터 계산<br>$Z = Z_e \| Z_s$ | $Z_s$<br>: V의 고정 개인키와 U의 고정 공개키를 사용한 FFC DH로부터 계산<br>$Z_e$<br>: V의 일회용 개인키와 U의 일회용 공개키를 사용한 FFC DH로부터 계산<br>$Z = Z_e \| Z_s$ |
| 키재료 유도 | *kdf(Z, OtherInput)* | *kdf(Z, OtherInput)* |

dhHybridOneFlow 방식과 동일하다.

■ MQV1, C(1,2, FFC MQV)

ASNI X9.42에서 정의한 방식으로 기본 구조는 dhHybridOneFlow 방식과 동일하다.

■ One-Pass MQV, C(1,2, ECC MQV)

〔표 12〕 MQV1 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | · 고정 개인키 $x_U$<br>· 고정 공개키 $y_U$ | · 고정 개인키 $x_V$<br>· 고정 공개키 $y_V$ |
| 일회용 데이터 | · 일회용 개인키 $r_U$<br>· 일회용 공개키 $t_U$ | 없음 |
| 입력 | $(p, q, g)$, $x_U, y_V, r_U$ | $(p, q, g)$, $x_V, y_U, t_U$ |
| 계산 | $Z$<br>: U의 고정 개인키, V의 고정 공개키, U의 일회용 개인·공개키 V의 고정 공개키를 사용한 FFC MQV로부터 계산 | $Z$<br>: V의 고정 개인키, U의 고정 공개키, V의 고정 개인·공개키 U의 일회용 공개키를 사용한 FFC MQV로부터 계산 |
| 키재료 유도 | *kdf(Z, OtherInput)* | *kdf(Z, OtherInput)* |

ASNI X9.63에서 정의한 방식으로 기본 구조는 dhHybridOneFlow 방식과 동일하다.

## 2.2 C(1,1) 스킴

키 동의 스킴에 참여한 객체들은 같은 도메인 파라미터로 키 쌍을 생성한다. 그림 4은 키 동의 스킴을 처음 시작하는 객체는 고정 키를 사용하지 않고 일회용 키를 생성·사용하고, 다른 객체는 고정 키만을 사용하여 키를 설정하는 키 동의 스킴을 나타내고 있다.

■ dhOneFlow, C(1,1, FFC DH)

ASNI X9.42에서 정의한 방식으로 객체 V는 고정 키를 사용하고 객체 U는 일회용 키 쌍을 생성하여 키 동의 스킴을 수행한다.

| 객체 U | | 객체 V |
|---|---|---|
| | V의 고정 공개키 ← | |
| | → U의 일회용 공개키 | |
| ① 자신의 일회용 개인키와 V의 고정 공개키를 사용 공유비밀을 생성 | | ① 자신의 고정 개인키와 U의 일회용 공개키를 사용 공유비밀을 생성 |
| ② 공유비밀을 사용하여 키 유도함수로부터 공유 키 재료를 생성한다. | | ② 공유비밀을 사용하여 키 유도함수로부터 공유 키 재료를 생성한다. |

〔그림 4〕 C(1,1) 키 동의 스킴

〔표 13〕 One-Pass Unified Model 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | • 고정 개인키 $d_{s,U}$<br>• 고정 공개키 $Q_{s,U}$ | • 고정 개인키 $d_{s,V}$<br>• 고정 공개키 $Q_{s,V}$ |
| 일회용 데이터 | • 일회용 개인키 $d_{e,U}$<br>• 일회용 공개키 $Q_{e,U}$ | 없음 |
| 입력 | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,U}$, $Q_{s,V}$, $d_{e,U}$, $Q_{e,U}$ | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,V}$, $Q_{s,U}$, $Q_{e,U}$, $Q_{e,V}$ |
| 계산 | $Z$<br>: U의 고정 개인키, V의 고정 공개키, U의 일회용 개인·공개키 V의 고정 공개키를 사용한 ECC MQV로부터 계산 | $Z$<br>: V의 고정 개인키, U의 고정 공개키, V의 고정 개인·공개키 U의 일회용 공개키를 사용한 ECC MQV로부터 계산 |
| 키재료 유도 | $kdf(Z, OtherInput)$ | $kdf(Z, OtherInput)$ |

〔표 14〕 dhOneFlow 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | 없음 | • 고정 개인키 $x_V$<br>• 고정 공개키 $y_V$ |
| 일회용 데이터 | • 일회용 개인키 $r_U$<br>• 일회용 공개키 $t_U$ | 없음 |
| 입력 | $(p, q, g)$, $y_V, r_U$ | $(p, q, g)$, $x_V, t_U$ |
| 계산 | $Z$<br>: U의 일회용 개인키와 V의 고정 공개키를 사용한 FFC DH로부터 계산 | $Z$<br>: V의 고정 개인키와 U의 일회용 공개키를 사용한 FFC DH로부터 계산 |
| 키재료 유도 | $kdf(Z, OtherInput)$ | $kdf(Z, OtherInput)$ |

〔표 15〕 One-Pass Diffie-Hellman 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | 없음 | • 고정 개인키 $d_{s,V}$<br>• 고정 공개키 $Q_{s,V}$ |
| 일회용 데이터 | • 일회용 개인키 $d_{e,U}$<br>• 일회용 공개키 $Q_{e,U}$ | 없음 |
| 입력 | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,U}$, $Q_{s,V}$, $d_{e,U}$, $Q_{e,U}$ | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,V}$, $Q_{s,U}$, $Q_{e,U}$, $Q_{e,V}$ |
| 계산 | $Z$<br>: U의 일회용 개인키, V의 고정 공개키를 사용한 ECC CDH로부터 계산 | $Z$<br>: V의 고정 개인키, U의 일회용 공개키를 사용한 ECC CDH로부터 계산 |
| 키재료 유도 | $kdf(Z, OtherInput)$ | $kdf(Z, OtherInput)$ |

■ One-Pass Diffie-Hellman, C(1,1, FFC DH)

ASNI X9.63에서 정의한 방식으로 위의 dhOneFlow 방식과 동일하다.

### 3. 고정 키만을 사용하는 스킴, C(0)

키 동의 스킴에 참여한 객체들이 고정 키만을 사용하여 공유비밀을 유도하는 방식이다.

### 3.1 C(0,2) 스킴

그림 5는 고정 키만을 사용하여 공유비밀을 유도하는 키 동의 스킴을 나타내고 있다.

■ dhStatic, C(0,2, FFC DH)

| 객체 U | | 객체 V |
|---|---|---|
| | U의 고정 공개키 → | |
| | ← V의 고정 공개키 | |
| ① 자신의 고정 개인키와 V의 고정 공개키를 사용 공유비밀을 생성 | | ① 자신의 고정 개인키와 U의 고정 공개키를 사용 공유비밀을 생성 |
| ② 공유비밀을 사용하여 키 유도함수로부터 공유 키 재료를 생성한다. | | ② 공유비밀을 사용하여 키 유도함수로부터 공유 키 재료를 생성한다. |

〔그림 5〕 C(0,2) 키 동의 스킴

〔표 16〕 dhStatic 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | • 고정 개인키 $x_U$<br>• 고정 공개키 $y_U$ | • 고정 개인키 $x_V$<br>• 고정 공개키 $y_V$ |
| 일회용 데이터 | 없음 | 없음 |
| 입력 | $(p, q, g)$, $x_U, y_V$ | $(p, q, g)$, $x_V, y_U$ |
| 계산 | $Z$<br>: U의 고정 개인키와 V의 고정 공개키를 사용한 FFC DH로부터 계산 | $Z$<br>: V의 고정 개인키와 U의 고정 공개키를 사용한 FFC DH로부터 계산 |
| 키재료 유도 | *kdf(Z, OtherInput)* | *kdf(Z, OtherInput)* |

ANSI X9.42에서 정의한 방식으로 객체 U와 객체 V 모두 고정 키를 사용하여 공유비밀을 유도하는 스킴이다.

■ Static Unified Model, C(0,2, ECC CDH)

ASNI X9.63에서 정의한 방식으로 위의 dhStatic 방식과 동일하다.

〔표 17〕 Static Unified Model 키 동의 스킴

| | 객체 U | 객체 V |
|---|---|---|
| 고정 데이터 | • 고정 개인키 $d_{s,U}$<br>• 고정 공개키 $Q_{s,U}$ | • 고정 개인키 $d_{s,V}$<br>• 고정 공개키 $Q_{s,V}$ |
| 일회용 데이터 | 없음 | 없음 |
| 입력 | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,U}$, $Q_{s,V}$ | $(q, FR, a, b, [SEED], G, n, h)$, $d_{s,V}$, $Q_{s,U}$ |
| 계산 | $Z$<br>: U의 고정 개인키, V의 고정 공개키를 사용한 ECC CDH로부터 계산 | $Z$<br>: V의 고정 개인키, U의 고정 공개키를 사용한 ECC CDH로부터 계산 |
| 키재료 유도 | *kdf(Z, OtherInput)* | *kdf(Z, OtherInput)* |

## Ⅳ. 결 론

본 고에서는 키 설정 표준문서인 ANSI X9.를 기반으로 제정된 NIST 800-56 문서를 중심으로 시스템간의 암호통신을 위해 수행되는 키 재료 설정 과정에 대해 분석하였다. NIST 800-56은 키 설정을 수행하기 위한 기본요소인 도메인 파라미터, 공개키·개인키 쌍 생성 및 검증방법, 키 유도 함수, 프리미티브 등에 대해 기술하고 있으며, 키 설정 스킴으로서 키 동의와 전송방법에 대해 기술한다. 본고에서는 설정에 관하여 자세히 분석하였으며, 특히 키 설정 중 활용도가 높은 키 동의 방식을 기반으로 하여 분석하였다.

본 연구는 향후 키 설정 스킴에 대한 연구를 진행함에 있어서 보다 효율적이고 안전한 암호통신을 위한 기반자료로서 널리 활용할 수 있을 것으로 기대된다.

## 참 고 문 헌


〔1〕 NIST SP 800-57 (Draft), Key Management Guideline, January 2003

〔2〕 ANSI X9.42-2001, Public Key Cryptography for the Financial Services Industry : Agreement of Symmetric Keys Using Discrete Logarithm Cryptography

〔3〕 ANSI X9.44 (Draft), Public Key Cryptography for the Financial Services Industry : Agree-ment the Key Transport Using Factoring-Based Cryptography, December 2003

〔4〕 ANSI X9.63-2001, Public Key Cryptography for the Financial Services Industry : Key Agreement and Key Transport Using Elliptic Key Cryptography

〔5〕 ANSI X9.69, "Framework for Key Manage-ment Extensions", ANSI, 1998

〔6〕 ANSI X9.80-2002, Prime number Generation, Primality Testing and Primality Certificates (Revised)

〔7〕 FIPS 197, Advanced Enryption Standard, November 2001

〔8〕 FIPS 198, The Keyed-Hash Message Authentication code(HMAC), March 2002

〔9〕 IEEE P1363, "Standard Specifications For Public Key Cryptography", IEEE, 2001

[10] PKCS #3, "Diffie-Hellman Key-Agreement Standard" RSA research, 1999
[11] B. Schneier, Applied Cryptography (Second Edition), John Wiley & Sons, Inc., 1996

## 〈著 者 紹 介〉

**원 동 규 (Dongkyu Won)**
**학생회원**

2003년 2월 : 인천시립대학교 전자공학과(공학사)
2003년 3월~현재 : 성균관대학교 정보통신공학부 석사과정

**곽 진 (Jin Kwak)**
**학생회원**

2008년 8월 : 성균관대학교 바이오 메카드로닉스 공학과 졸업(공학사)
2003년 2월 : 성균관대학교 대학원 전기전자 및 컴퓨터 공학부 졸업(공학석사)
2003년 3월~현재 : 성균관대학교 정보통신공학부 박사과정

**주 미 리 (Miri Joo)**
**학생회원**

1996년 2월 : 성균관대학교 정보공학과 졸업(공학사)
1998년 2월 : 성균관대학교 대학원 정보공학과(공학석사)
2003년 2월 : 성균관대학교 대학원 전기전자 및 컴퓨터 공학부 졸업(공학박사)
2001년~현재 : 현재 국가보안기술연구소 연구원

**양 형 규 (Hyungkyu Yang)**
**정회원**

1983년 2월 : 성균관대학교 전자공학과 졸업(공학사)
1985년 2월 : 성균관대학교 대학원 전자공학과 졸업(공학석사)
1984년 12월~1991년 2월 : 삼성전자 선임 연구원
1995년 2월 : 성균관대학교 대학원 정보공학과 졸업(공학박사)
1995년 3월~현재 : 강남대학교 컴퓨터미디어공학부 부교수

**원 동 호 (Dongho Won)**
**종신회원**

1976년~1988년 : 성균관대학교 전자공학과 (학사, 석사, 박사)
1978년~2003년 : 한국전자통신연구소 전임 연구원, 일본 동경공대 객원연구원, 성균관대학교 교학처장, 전기·전자 및 컴퓨터공학부장, 정보통신대학원장, 국무총리실 정보화추진위원회 자문위원, 한국정보보호학회 이사, 부회장, 수석부회장, 회장
현재 : 성균관대학교 정보통신공학부 교수, 성균관대학교 연구지원처장, 한국정보보호학회 명예회장, 정통부지정 정보보호인증기술연구센터 센터장