SCIS 2003 The 2003 Symposium on Cryptography and Information Security
Hamamatsu,Japan, Jan.26-29,2003
The Institute of Electronics, Information and Communication Engineers

内部不正者を考慮したグループ鍵共有方式に関する考察

On Secure Group Key Agreement Protocol Robust against Insider Attacks

安東　学* 松浦　幹太* 馬場　章*
Manabu Ando Kanta Matsuura Akira Baba

あらまし グループ鍵共有方式において，エンティティ認証などの付加的なセキュリティ要件を満たす場合，プロトコルが極めて複雑になる．これまでに様々なグループ鍵共有方式が提案されているが，これらは必ずしも付加的なセキュリティ要件を満たしているわけではない．特に，Key Confirmation を保証できるグループ鍵共有方式はあまり考察されていない．著者らは Kim らが提案した，木構造と Diffie-Hellman 鍵共有方式に基づくグループ鍵共有方式に着目した．Kim らは，内部不正者を考慮していないため，能動的な内部不正者による攻撃が可能となる．これは Key Confirmation がプロトコルにおいて保証されないことを利用した攻撃である．本論文では，Kim らの方式を拡張し，Key Confirmation 保証機能付きのグループ鍵共有方式を提案する．また，提案プロトコルは能動的な内部不正者に対し，ある仮定のもとでは安全であることを示す．

キーワード グループ鍵共有方式，Key Confirmation, 内部不正者

1 はじめに

グループでの秘密通信を行いたい場合，通常グループに属する参加者全員が同一の鍵を共有し，その鍵でメッセージを暗号化する．このとき鍵共有方式は，2 者間での鍵共有と比べて複雑になる．プロトコルそのものが複雑になるだけではなく，要求されるセキュリティ要件を満たす仕組みもまた複雑になる．例えば，2 者間で Diffie-Hellman 鍵共有 [1] を行う際に，エンティティの認証を同時に行いたい，という要求がある．これに対し，[4][5] 等で提案されている方式を適用することができる．しかし，グループ鍵共有において同様の機能を満たすには，より複雑なプロトコル設計になってしまう．従って，2 者間での鍵共有方式にもたらされていたセキュリティ要件を，これまでに提案されてきた方式は必ずしも満たしているわけではない．

これまで，様々なグループ鍵共有方式が提案されている．グループ鍵共有方式は 2 種類の方針がある．中央管理型と分散型である．前者の例としては，Diffie-Hellman 鍵共有方式を応用し，管理者が参加者に鍵を配送する方式 [3]，木構造を利用してグループ鍵を配送する方式 [16]，一方向性ハッシュ関数と木構造を利用してグループ鍵を配送する方式 [11]，一定期間毎に管理者がグループ鍵を更新し，参加者に配送する方式 [14] 等がある．後者の例としては，Diffie-Hellman 鍵共有方式を利用し，参加者の積極的参加により鍵を生成する方式 [5]，参加者がリレー式に通信を行い，フィードバックを受けて鍵を生成する方式 [15]，参加者各自が木構造を管理し，一方向性ハッシュ関数により鍵を生成する方式 [2]，Diffie-Hellman 鍵共有方式と木構造を利用して鍵を生成する方式 [8][12] 等がある．これらの方式に共通している問題点として，他の参加者と同一の鍵を共有しているのかどうか確認できない，ということがある．つまり，Key Confirmation がプロトコル実行において保証されない，という問題がある．[3] では，Key Confirmation を保証できる方式が提案されているが，中央管理型の方式である．著者らは，中央管理型については，[12] に指摘されているように，セキュリティ上危険であるとの立場をとっている．また，[5] においても，Key Confirmation を保証できるが，これは Challenge-Response をプロトコルの最後に付け加えたものであり，効率面に問題がある．

従って，本論文では (1) 管理者権限を排除した分散型グループ鍵共有方式，(2)Key Confirmation を保証できる方式，を考える．(1) については，上記した通り多数のプロトコルが既に提案されている．この中から本論文では，Kim らが提案した方式を取り上げる．このグループ鍵共有方式は安全性と効率面，実現性などから総合的に

* 東京大学大学院情報学環・学際情報学府，〒 113-0033 東京都文京区本郷 7-3-1，Interfaculty Initiative in Information Studies, Graduate School of Interdisciplinary Information Studies, Univ. of Tokyo, 7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan

判断し，優れているものであると考えられる．この方式は，木構造と Diffie-Hellman 鍵共有方式 [1] に基づくグループ鍵共有方式 [6][8][12] である（以後，Kim らの方式を KPT 方式を記述する）．KPT 方式では，管理者権限を所持するエンティティを極力減らし，各エンティティの積極的協力により，グループ鍵を生成する．また，ランダムオラクルを仮定すると，能動的な盗聴者がグループ鍵を生成できないということが証明されている [7].

しかし，KPT 方式では内部不正者に対する考慮が十分ではない．特に，Key Confirmation が保証されないために，能動的な内部不正者による攻撃が可能となる．内部不正者はある特定の参加者に対してのみ，偽の情報を送り，他の参加者とは異なる鍵を生成させることができる．このような攻撃は，[3] においても管理者による能動的攻撃として指摘されており，無視することはできない．

本論文では，KPT 方式を拡張し，Key Confirmation を保証できるグループ鍵共有方式を提案する．Key Confirmation は，鍵を共有後に全参加者による Challenge-Response によって保証できるが，この方法は参加者数に比例して通信量が増加する．また，鍵共有後であると，実際に共有した鍵と Challenge-Response に利用した鍵とのリンクを保証することが困難である．従って，鍵共有プロトコル実行中に Key Confirmation を保証できる仕組みが必要である．提案プロトコルは，KPT 方式実行中に Key Confirmation を保証することができる．また，Diffie-Hellman 問題の解決が困難であることを仮定すると，提案プロトコルはある種の能動的な内部不正者に対して安全であることを証明する．

本論文では，まず 2 章で KPT 方式の概要と問題点を示す．この問題点を受け，3 章と 4 章で Key Confirmation 保証機能付きグループ鍵共有方式について述べる．5 章では，提案プロトコルの安全性について考察する..

2 KPT 方式とその問題点

2.1 KPT 方式の概要

ここで，KPT 方式で用いられる表記を整理する．

U_i	参加者 i
p	大きな素数
g	Z_p^* における原始元
r_i	参加者 U_i が生成する乱数
$K_{(i,j)}$	U_i, U_j 間で共有される鍵
$BK_{(i,j)}$	$BK_{(i,j)} = g^{K_{(i,j)}} \bmod p$

表 1: KPT 方式における表記

KPT 方式では，完全二分木を用意し，各参加者を葉に対応させる．図 1 に例を示す．管理者権限を所持する参加者を設定しないため，木は各参加者が独自に所有する．変更があった際には各自の責任で更新する．鍵共有

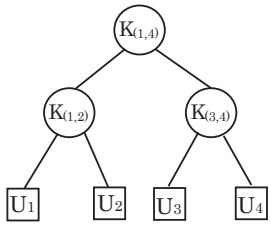

図 1: 木構造の例

には以下のようなプロトコルを実行する．なお，[6] と [12] では鍵共有プロトコル自体に違いはない．

Step 1. U_i は乱数 r_i を生成し，$x_i = g^{r_i} \bmod p$ を計算する．U_i は x_i を全参加者へ送信する．また，他の参加者から x を受信する．

Step 2. U_i は隣接する葉に対応する参加者 U_{i+1} から受け取った x_{i+1} と r_i から，$K_{(i,i+1)} = g^{r_i r_{i+1}} \bmod p$ を生成する．また，$BK_{(i,i+1)} = g^{g^{r_i r_{i+1}}} \bmod p$ を生成し，全参加者へ送信する．

Step 3. U_i は隣接するノードに対応する $BK_{(i+2,i+3)}$ と自らが生成した $K_{(i,i+1)}$ から，$K_{(i,i+3)} = BK_{(i+2,i+3)}^{K_{(i,i+1)}} \bmod p$ を生成する．

以降，*Step 2* と *Step 3* を繰り返すことにより，全参加者と鍵を共有することができる．

参加者の変更に関するイベントが発生した場合，一時的な管理者 (Sponsor) を設定し，変更部分を全参加者に対しブロードキャストする．他の参加者はこれを受けて，自らが所持している木を更新する（詳細は [6] を参照）．[12] では，管理者権限を完全に排除しているため，参加者変更イベント時にも全ての BK がブロードキャストされる．

2.2 KPT 方式の問題点

KPT 方式では，グループ鍵を更新する場合，各参加者の BK をブロードキャストする．また，中間ノードに対応する BK についても各参加者によりブロードキャストされる．例えば，図 1 において，$BK_{(1,2)}$ は U_1 と U_2 のみ生成することができ，U_1, U_2 により他の参加者へブロードキャストされる．このとき，能動的な内部不正者による攻撃が成功する．ここで，内部不正者 I_2 とし，図 1 における U_2 の葉に対応しているとする．また，攻撃対象を U_1 とする．U_1 は正規の動作によりプロトコルを実行する．

Step 1'. U_1 は $x_1 = g^{r_1} \bmod p$ を計算する．U_1 は x_1 を全参加者へ送信する．また，I_2 から $x_2' = g^{r_2'} \bmod p$

を受け取る．I_2 は他の参加者に対し，$x_2 = g^{r_2} \bmod p$ を送信する．

Step 2'. U_1 は $K'_{(1,2)} = g^{r_1 r'_2} \bmod p$ を生成する．また，$BK'_{(1,2)} = g^{g^{r_1 r'_2}} \bmod p$ を生成し，全参加者へ送信する．ここで，I_2 はこの通信内容を $BK_{(1,2)}$ にすり替え，他の参加者へ転送する．

このような攻撃により，U_1 は正常にプロトコルを実行したにも関わらず，他の参加者と同一のグループ鍵を共有することができない．これは，Key Confirmation が保証されないことを利用した攻撃である．

また，KPT 方式では，管理者権限を極力排除しているが，参加者変更イベント時に Sponsor と呼ばれる一時的管理者が設定される．Sponsor は全参加者に変更部分を知らせる．このことにより問題が発生する．例え

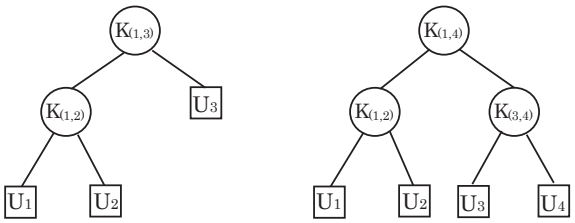

図 2: 能動的な Sponsor による攻撃

ば，図 2 において U_4 が加入することを考える．このとき，Sponsor は U_3 である．U_3 は U_4 から $g^{r_4} \bmod p$ を受け，$K_{(3,4)} = g^{r_3 r_4} \bmod p$ を更新する．更に，$BK_{(3,4)} = g^{K_{(3,4)}}$ となり，$BK_{(3,4)}$ を U_1, U_2 へ送る．このとき，U_3 が攻撃対象を U_1 とし，U_1 に対し $BK'_{(3,4)}$ を送ると，U_1 は他の参加者と同一の鍵を共有することができない．また，他の参加者にとっては正当な動作を行っているために，不正があったことに気がつかない．この攻撃もまた，Key Confirmation がプロトコル実行中に保証されないことを利用した攻撃である．

[12] のように，管理者権限を完全に排除しても同様な攻撃が可能となる．更に，[12] の方法では通信量が参加者数に比例して増加する．グループ鍵を共有するのに全ての BK は必要ない．[12] の方式では過剰な通信負担が懸念される．これは，参加者のサブグループへの分割と識別子により解決することができる．つまり，サブグループ番号とサブグループ内における識別子から参加者を一意に特定できるようにし，この識別子により通信相手を決定する．その際に余分な通信を極力削減できるようなポリシーによって通信相手を決定することが重要になる．

3 提案プロトコルのモデル

本論文では，KPT 方式に Key Confirmation 保証機能を加える．従って，モデルは KPT 方式とほぼ同様になる．本節では，変更する部分について述べ，提案プロトコルで使用する表記を整理する．

3.1 表記と仮定

提案プロトコルでは，N 個の葉をもつ完全二分木を用意する (ただし，$N = 2^l$)．各参加者は木の葉に対応する．参加者は仮想的なサブグループを構成する．サブグループは 2 者から成り立つ．従って，参加者数 N，サブグループ数 $N/2$ となる．提案プロトコルに必要な表記を表 2 にまとめる．

U_{ij}	参加者の識別子．i はサブグループ番号，j はサブグループ内での番号を示す．$1 \leqq i \leqq N/2$, j={0,1}.
p, q	大きな素数．ただし，$q \mid p-1$
g	Z_p^* における位数 q の元
r_{ij}, t_{ij}	参加者 U_{ij} が生成する乱数
$X_{ij \to lm}$	U_{ij} が生成し，転送先を U_{lm} に指定した値
SK_{ij}	参加者 U_{ij} の秘密鍵
PK_{ij}	参加者 U_{ij} の公開鍵，$PK_{ij} = g^{SK_{ij}} \bmod p$.
$h()$	一方向性ハッシュ関数
$K_{(ij,lm)}$	U_{ij}, U_{lm} 間で共有される鍵
$BK_{(ij,lm)}$	$BK_{(ij,lm)} = g^{K_{(ij,lm)}} \bmod p$

表 2: 表記

3.2 通信相手について

前節で指摘したように，[12] では必要のない BK の送受がある．本論文では，この余分な通信を削減するために，以下のような通信ポリシーを定める．なお，U_{i0} は左の葉，U_{i1} は右の葉に対応する．

参加者 U_{i0}**:** サブグループ内の $U_{i(j+1)}$ と通信し，鍵 $K_{(ij,i(j+1))}$ を共有する．その後，U_{i0} $(1 \leqq i \leqq N/2)$ と通信することにより，参加者全体のグループ鍵を共有する．

参加者 U_{i1}**:** サブグループ内の $U_{i(j+1)}$ と通信し，鍵 $K_{(ij,i(j+1))}$ を共有する．その後，U_{i1} $(1 \leqq i \leqq N/2)$ と通信することにより，参加者全体のグループ鍵を共有する．

このポリシーに基づいて通信することにより，[12] と同様にグループ鍵を共有することができる．更に，余分な通信を減少させることができる．また，[12] での安全性を保っている．しかし，Key Confirmation は保証されないままである．

4 Key Confirmation 保証機能付きグループ鍵共有プロトコル

4.1 準備

表 2 の中で，p, q, g はプロトコル参加者全体の公開情報である．また，参加者 U_{ij} は公開鍵 PK_{ij} を TA(Trusted Authority) へ登録する．TA は公開情報と ID の組み合わせを厳重に保管する．正規の参加者は，この公開情報を参照することができる．また，TA による不正はないものと仮定する．従って，内部不正者と TA による結託はないものとする．

4.2 提案プロトコルの概要

以下では，U_{ij} を始動者とする．通信モデルに従い，U_{ij} は $U_{i(j+1)}$ と通信を行う．*Step 1* では，KPT 方式と同様の手続きが行われる．つまり，2 者間での Diffie-Hellman 鍵共有プロトコルと同じ手続きである．

Step 1.
$U_{ij} \rightarrow U_{i(j+1)}$:
$x_{ij} = g^{r_{ij}} \bmod p$
$U_{ij} \leftarrow U_{i(j+1)}$:
$x_{i(j+1)} = g^{r_{i(j+1)}} \bmod p$

Step 2 における手続きも KPT 方式と同様である．すなわち，Diffie-Hellman 鍵共有プロトコルにおける鍵確立手続きである．さらに，*Step 2* では $K_{(ij,i(j+1))}$ に対応する $BK_{(ij,i(j+1))}$ の準備も行う．

Step 2.
U_{ij}:
$K_{(ij,i(j+1))} = g^{r_{ij} r_{i(j+1)}} \bmod p$
$BK_{(ij,i(j+1))} = g^{g^{r_{ij} r_{i(j+1)}}} \bmod p$

Step 3 において，U_{ij} は $U_{i(j+1)}$ から受け取った値 $x_{i(j+1)}$ に対する署名を生成する．このとき，内部不正者が U_{ij} の公開鍵から署名を偽造できないようにする工夫が必要である．ここでは，$e_{ij\rightarrow n(j+1)}, y_{ij\rightarrow n(j+1)}$ が $x_{i(j+1)}$ に対する署名となっている．検証者は U_{ij} の公開鍵と $z_{ij\rightarrow n(j+1)}$ から $g^{t_{ij}} \bmod p$ を復元することによって署名を検証する．$e_{ij\rightarrow n(j+1)}$ のハッシュ関数の入力値である $PK_{n(j+1)}^{t_{ij}}$ により，この署名は $PK_{n(j+1)}$ の所有者のみが検証できることになる．

Step 3.
U_{ij}:
$for(n = 1; n \leqq N/2; n{+}{+})$ {
$t_{ij} \in_R Z_q$
$e_{ij\rightarrow n(j+1)} = h(x_{i(j+1)}, BK_{(ij,i(j+1))}, PK_{n(j+1)}^{t_{ij}})$
$y_{ij\rightarrow n(j+1)} = t_{ij}/(t_{ij} + e_{ij\rightarrow n(j+1)} SK_{ij}) \bmod q$
$z_{ij\rightarrow n(j+1)} = (g^{t_{ij}})^{y_{ij\rightarrow n(j+1)}} \bmod p$
}

Step 4 では，U_{ij} は $U_{i(j+1)}$ に対し，署名部分を送信する．

Step 4.
$U_{ij} \rightarrow U_{i(j+1)}$:
$e_{ij\rightarrow n(j+1)}, y_{ij\rightarrow n(j+1)}, z_{ij\rightarrow n(j+1)}$

Step 5 において，$U_{i(j+1)}$ は U_{ij} から受け取った署名部分と $BK_{(ij,i(j+1))}$ を結びつける．

Step 5.
$U_{i(j+1)}$:
$w_{i(j+1)\rightarrow n(j+1)} = BK_{(ij,i(j+1))} \cdot z_{ij\rightarrow n(j+1)}$

Step 6 において $U_{i(j+1)}$ は，$BK_{(ij,i(j+1))}$，$x_{i(j+1)}$ とともに，U_{ij} から受け取った署名部分 $e_{ij\rightarrow n(j+1)}, y_{ij\rightarrow n(j+1)}$ を $U_{n(j+1)}$ へ転送する．

Step 6.
$U_{i(j+1)} \rightarrow U_{n(j+1)}$:
$e_{ij\rightarrow n(j+1)}, y_{ij\rightarrow n(j+1)}, BK_{(ij,i(j+1))}, x_{i(j+1)},$
$w_{i(j+1)\rightarrow n(j+1)}$

Step 7 において，$U_{n(j+1)}$ は (1) KPT 方式における鍵共有のための手続き，(2) 署名の検証を行う．(1) は $U_{i(j+1)}$ から受け取った $BK_{(ij,i(j+1))}$ を利用し，$K_{(ij,n(j+1))}$ を生成する．この手続きは KPT 方式と同様である（2.1 節参照）ため，ここでは省略する．(2) は $U_{i(j+1)}$ から転送された U_{ij} による署名の検証である．ここでは，U_{ij} の公開鍵と署名 $e_{ij\rightarrow n(j+1)}, y_{ij\rightarrow n(j+1)}$ から，$g^{t_{ij}} \bmod p$ を復元する．更に，$U_{n(j+1)}$ 自身が所持する秘密鍵を利用し，ハッシュ関数の入力値 $PK_{n(j+1)}^{t_{ij}}$ を復元する（復元した値を $v_{n(j+1)}$ と書くことにする）．$v_{n(j+1)}$ と $U_{i(j+1)}$ から受け取った $x_{i(j+1)}, BK_{(ij,i(j+1))}$ をハッシュ関数に入力し，$e_{ij\rightarrow n(j+1)}$ と等しくなれば，署名の検証に合格する．検証が不合格の場合，プロトコル全体を終了させる．

Step 7.
$U_{n(j+1)}$:
$v_{n(j+1)} = \left(PK_{ij}^{e_{ij\rightarrow n(j+1)} y_{ij\rightarrow n(j+1)}} \cdot w_{i(j+1)\rightarrow n(j+1)} / BK_{(ij,i(j+1))}\right)^{SK_{n(j+1)}} \bmod p$
Check,
$e_{ij\rightarrow n(j+1)} \stackrel{?}{=} h(x_{i(j+1)}, BK_{(ij,i(j+1))}, v_{n(j+1)})$

5 安全性分析

提案プロトコルでは，U_{ij} は $U_{i(j+1)}$ から受け取った値 $x_{i(j+1)}$ に対し，転送相手にしか検証できない署名を添付している．この署名の検証に合格するということは，(1) $U_{i(j+1)}$ が U_{ij} へ送った $x_{i(j+1)} = g^{r_{i(j+1)}} \bmod p$ に使用された $r_{i(j+1)}$，(2) $U_{i(j+1)}$ が $U_{n(j+1)}$ へ送った $BK_{(ij,i(j+1))} = g^{g^{r_{ij} r_{i(j+1)}}} \bmod p$ に使用された $r_{i(j+1)}$ が等しいことを示す．KPT 方式における能動的な内部

不正者 I は，(1) と (2) の値を異なるものにし，攻撃対象のみが同一の鍵を共有できないようにしている（2.2 節参照）．従って，(1) と (2) が等しいと証明されれば，同一の鍵が共有されたことを保証することができる．

上記のような内部不正者 I は，提案プロトコルで同様の攻撃を行う場合，署名部分について偽造を行おうとすることが予想される．署名部分を偽造せずに (2) の値のみを変更するのでは，署名検証時に不合格となり，攻撃が成功しない．また，署名部分は正規の手段で入手することができる．従って，内部不正者 I は (2) の値を変更し，署名部分を偽造することによって攻撃を成功させようとする．よって，能動的な内部不正者による署名部分の偽造が，ある仮定のもとでは困難であることが証明できれば，提案プロトコルは安全であるといえる．本節では，ある種の能動的な内部不正者による署名偽造が困難であることを示す．

5.1 内部不正者 I の定義

2.2 節で示したように，内部不正者 I_{ij} は通信相手によって異なる値を送信し，攻撃対象が他の参加者と同一の鍵を共有できないようにする．*Step 1* において，I_{ij} は $U_{i(j+1)}$ に対し $x'_{ij} = g^{r'_{ij}} \bmod p$ を送信する．しかし，*Step 6* において，I_{ij} は U_{nj} に対し，$BK_{(ij,i(j+1))} = g^{g^{r_{ij}r_{i(j+1)}}} \bmod p$ と $x_{ij} = g^{r_{ij}} \bmod p$ を送信する．

提案プロトコルでは，$e_{i(j+1)\to nj}, y_{i(j+1)\to nj}, w_{ij\to nj}$ の値が署名の役割を果たしているため，I_{ij} はこれらの値も偽造し，$BK_{(ij,i(j+1))}$ や x_{ij} と辻褄が合うようにしなければならない．本論文では提案プロトコルに対する上記の攻撃を “攻撃 1” と書くことにする．

なお，I_{ij} はプロトコルでやり取りされるメッセージを全て盗聴することができる攻撃者とする．つまり，通信ポリシーに従うと受け取ることのないメッセージであっても，盗聴することによって受け取ることができる攻撃者である．

5.2 安全性証明

“攻撃 1” を成功させるには，I_{ij} は $U_{i(j+1)}$ から受け取った $e_{i(j+1)\to nj}$ を改ざんしなければならない．$e_{i(j+1)\to nj}$ を辻褄の合うように改ざんできれば，$y_{i(j+1)\to nj}$, $z_{i(j+1)\to nj}$ はそのまま転送しても “攻撃 1” に成功する．$e_{i(j+1)\to nj} = h(x_{ij}, BK_{(ij,i(j+1))}, PK_{nj}^{t_{i(j+1)}})$ であるから，I_{ij} は $e'_{i(j+1)\to nj} = h(x'_{ij}, BK'_{(ij,i(j+1))}, PK_{nj}^{t_{i(j+1)}})$ を生成することができれば “攻撃 1” に成功する．

I_{ij} が $e_{i(j+1)\to nj}$ の入力値 $x_{ij}, BK_{(ij,i(j+1))}, PK_{nj}^{t_{i(j+1)}}$ を生成できるのであれば，$e'_{i(j+1)\to nj}$ を生成し，“攻撃 1” に成功する．$e_{i(j+1)\to nj}$ の入力値の内，I_{ij} が自力で計算できるのは $x_{ij}, BK_{(ij,i(j+1))}$ である．また，PK_{nj} を入手することもできる．従って，$PK_{nj}^{t_{i(j+1)}}$ が生成できれば “攻撃 1” に成功する．

定理 5.1. オラクル C が公開値と通信路上を流れる値から $PK_{nj}^{t_{i(j+1)}}$ を計算し “攻撃 1” に成功するのであれば，Diffie-Hellman 問題を解くことのできるアルゴリズム A^C が存在する．

Proof. A^C は以下のような計算を行う．

1. A^C は，U_{nj} の公開鍵 $PK_{nj} = g^{SK_{nj}} \bmod p$ を入手する．
2. A^C は，$U_{i(j+1)}$ から正当な手続きで送られてきた $y_{i(j+1)\to nj}$，$z_{i(j+1)\to nj}$ を得る．$y_{i(j+1)\to nj}$ の逆数を計算し，$z_{i(j+1)\to nj}$ から，$g^{t_{i(j+1)}} \bmod p$ を得る．
3. A^C は，オラクル C に対し，$g^{t_{i(j+1)}} \bmod p$ と $PK_{nj} = g^{SK_{nj}} \bmod p$ を送る．
4. オラクル C は，$g^{t_{i(j+1)}} \bmod p$ と PK_{nj} を入力とし，$PK_{nj}^{t_{i(j+1)}}$ を出力することができる．

オラクル C の出力結果は，$PK_{nj}^{t_{i(j+1)}} = g^{SK_{nj}t_{i(j+1)}} \bmod p$ である．従って，アルゴリズム A^C は，オラクル C からの出力結果を得ることによって，$g^{t_{i(j+1)}} \bmod p$ と $g^{SK_{nj}} \bmod p$ の Diffie-Hellman 問題を解くことのできるアルゴリズムとなっている． □

この定理から，Diffie-Hellman 問題の解決が困難であることを仮定したとき，提案プロトコルは能動的な内部不正者 I_{ij} による “攻撃 1” に対して，安全であるといえる．

6 おわりに

本論文では，KPT 方式というグループ鍵共有方式において，能動的な内部不正者による攻撃が可能であることを示し，この攻撃に耐性のあるプロトコルを提案した．KPT 方式では，Key Confirmation が保証されないことを利用し，能動的な内部不正者が攻撃対象の鍵共有を妨害し，他の参加者と同一の鍵を共有できないようにするような攻撃が可能となる．提案プロトコルでは，KPT 方式に Key Confirmation 保証機能を加えることにより，各参加者が同一の鍵を共有しているかどうか確認することができる．結果として，内部不正者によるある種の能動的攻撃が困難になった．また，Diffie-Hellman 問題の困難さを仮定すると，提案プロトコルはある種の能動的な内部不正者に対し，安全であることを示した．

今後の課題は，木構造の複雑化に関する考察である．本論文では，木構造を完全二分木に限定した．しかし，実際の世界におけるグループの構成はより複雑である．グループ内サブグループへの多重所属など，二分木で表現することのできないグループ構成が考えられる．多様なグループ構成を実現でき，かつ安全なグループ鍵共有方式の実現が今後の課題である．

参考文献

[1] W. Diffie and M. E. Hellman: “New Direction in Cryptography,” *IEEE Trans. Information Theory*, vol.IT-22, pp.644-654, 1976.

[2] L. Dondeti, S. Mukherjee, and A. Samal: “DISEC: A Distributed Framework for Scalable Secure Many-to-many Communication,” In *Proc. of the 5th IEEE Symposium on Computers and Communications (ISCC2000)*, pp.693-698, 2000.

[3] 廣瀬勝一，池田克夫: “離散対数問題に基づく会議鍵配布システムとその安全性,” 1997 年暗号と情報セキュリティシンポジウム, SCIS’97-21E, 1997.

[4] S. Hirose and S. Yoshida: “An Authenticated Diffie-Hellman Key Agreement Protocol Secure against Active Attacks,” In *Proc. of PKC’98*, LNCS1431, pp.135-148, Springer-Verlag, 1998.

[5] M. Just and S. Vaudenay: “Authenticated Multi-Party Key Agreement,” In *Proc. of Advances in Cryptology – ASIACRYPT ’96*, LNCS1163, pp.36-49, Springer-Verlag, 1996.

[6] Y. Kim, A. Perrig, and G. Tsudik: “Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups,” In *Proc. of 7th ACM Conference on Computer and Communications Security*, pp.235-244, 2000.

[7] Y. Kim, A. Perrig, and G. Tsudik: “Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups,” Technical Report 2, USC Technical Report00-737, 2000.

[8] Y. Kim, A. Perrig, and G. Tsudik: “Communication-Efficient Group Key Agreement,” In *Information Systems Security, Proceedings of the 17th International Information Security Conference IFIP SEC’01*, 2001.

[9] K. Matsuura and H. Imai: “Protection of Authenticated Key-Agreement Protocol against a Denial-of-Service Attack,” In *Proc. of 1998 International Symposium on Information Theory and Its Applications (ISITA’98)*, pp. 466-470, 1998.

[10] 松浦幹太，今井秀樹: “鍵共有プロトコルのサービス妨害攻撃対策,” 第 21 回情報理論とその応用シンポジウム予稿集，pp.719-722, 1998.

[11] D. A. McGrew, and A. T. Sherman: “Key Establishment in Large Dynamic Groups Using One-Way Function Trees,” Technical Report No. 0755, TIS Labs at Network Associates, Inc., 1998.

[12] A. Perrig: “Efficient Collaborative Key Management Protocols for Secure Autonomous Group,” In *CrypTEC’99*, pp.192-202, 1999.

[13] C. P. Schnorr: “Efficient Identification and Signatures for Smart Cards,” In *Proc. of Advances in Cryptology – CRYPTO’89*, LNCS 435, pp.239-252, Springer-Verlag, 1990.

[14] S. Setia, S. Koussih, S. Jajodia, and E. Harder: “Kronos: A Scalable Group Re-keying Approach for Secure Multicast,” In *Proc. of IEEE Symposium on Security and Privacy*, pp.215-228, 2000.

[15] M. Steiner, G. Tsudik, and M. Waidner: “CLIQUES: a New Approach to Group Key Agreement,” In *Proc. of the 18th International Conference on Distributed Computing Systems (ICDCS’98)*, pp.380-387, 1998.

[16] C. K. Wong, M. Gouda, and S. S. Lam: “Secure Group Communications Using Key Graphs,” In *Proc. of ACM SIGCOMM*, pp.68-79, 1998.