平成30年度　修 士 論 文

和文題目

管理者への情報漏えいを防止する
グループ鍵共有方式の提案と実装

英文題目

Proposal for a Group Key Sharing Method that Prevents Information Leakage to an Administrator and its Implementation

情報工学専攻　渡邊研究室

(学籍番号: 173426013)

菅沼　良一

提出日: 平成 31 年 1 月 29 日

名城大学大学院理工学研究科

概要

インターネットを介してあらゆる情報をやり取りする機会が増加している．ここでは 1 対 1 の通信だけではなく，1 対多や多対多での通信も多く行われる．この場合，端末間でグループを生成し，通信が行われる．この時，通信内容がグループ外の端末に漏洩しないよう，グループ鍵を用いて暗号化通信を行うことが考えられる．グループ鍵の管理は鍵管理装置で集中的に管理する方法が多い．しかし，こののの方式では，サーバ管理者がグループ鍵を所有していることによる情報漏洩の可能性が懸念される．そこで本稿では，配送経路が異なる 2 つの乱数をグループメンバに配布し，それらを基にグループ鍵を生成する方式を提案する．これにより，管理が容易でかつサーバ管理者にも内容を読み取られることがないセキュアなグループ通信が可能となる．また，提案方式の一部を実装し，仮想環境において動作検証と計測を行った．この結果から，実用可能な時間で実行できることを確認することで，セキュアなグループ鍵共有が実現できることを示した．

目 次

第1章　序論

ネットワーク技術・インフラの発展により，世界中のあらゆる場所でインターネットが利用可能な社会となった．企業用途，個人用途双方でインターネットを介した情報共有の重要性が増してきている．インターネットを介した情報共有の方法として，チャットや遠隔会議などが挙げられる．これらは，1 対 1 の通信だけでなく，1 対 N や，N 対 N といった，グループ通信を行うことも多い．そのため，グループで行われる通信を実現するための手法は必要不可欠である．また，それらの用途で使用されるアプリケーションは，機密性が高い内容を取り扱うケースも多くあるため，グループメンバの認証とパケットの暗号化による安全な通信を行えることが必須である．

安全なグループ通信を行う方法として，グループメンバそれぞれが，お互いの鍵を共有して暗号化通信を行う手法が考えられる．しかしこの手法では，各グループメンバは，すべてのグループメンバの鍵を所有している必要があり，グループで共有するデータを，各グループメンバに対応する鍵を用いて暗号化するため，非常に非効率的である．グループでの暗号化通信を行う場合，グループ鍵を用いることにより，鍵管理が容易になることが知られている．グループ鍵は，グループ内で共有される鍵であり，これを用いて暗号化を行うため，グループで共有するデータの暗号化は，一度のみでよい．グループ鍵を用いるコミュニケーションシステムでは一般的に鍵管理要件を満たす必要があるとされている [9]．鍵管理要件とは，グループ鍵を運用するにあたって満たすべき項目であり，外部からの攻撃に対しての安全性や，グループメンバの参加や退会に際して行われるべき処理及びグループ鍵の更新方法を規定している．しかし，近年になってメッセージアプリケーションのセキュリティ評価を行っている非営利団体 EFF（Electric Frontier Foundation）は，現状の主流なチャットアプリケーションのセキュリティが極めて脆弱であることを指摘していた [1]．EFF の評価項目の中には，グループ鍵を管理者が読めないよう暗号化されているかどうかという項目があり，多くのアプリケーションはこの項目を満たしていない．この項目は，2013 年にアメリカの国家安全保障局 NSA と連邦捜査局 FBI が，Google や Facebook など，アメリカに拠点を置く IT 企業のサーバから，不正にデータを入手していた [2] ことが発覚したことからも，重要性が高いことがわかる．即ち，管理者も 1 つのセキュリティホールになりうる．

グループ鍵を共有する代表的な実現手法として，GSAKMP(Group Secure Association Key Mangagement Protocol) [8] が挙げられる．GSAKMP は，GKMP(Group Key Management Protocol) を改良した，RFC4535 によって標準化されている鍵管理プロトコルであり，鍵管理要件を満たす．GSAKMP では，鍵を管理するサーバである GCKS(Group Controller Key Server) を用いて，グループ鍵の生成，更新，配送を行っており，各端末が別々のグローバル/プライベート空間に所属している場合にも適用できる．しかし，GSAKMP では，GCKS によってグループ鍵の生成，更新，配送が行われているため，管理者はグループの通信内容を取得することができるという課題がある．

また，管理者がグループメンバを定義する為，不正なメンバを混入させることができるという課題もある．さらに，すべてのグループメンバが公開鍵証明書を所有する必要があるため，証明書の管理コストがかかる．

グループ管理者に関するリスクをなくす方法として，鍵管理サーバを用いないグループ鍵管理方式（分散型鍵管理方式）がある [3] [4] [5] [6]．この方式では一般に鍵管理要件を満たすための鍵管理プロトコルが複雑になる．また，複数のプライベート空間に渡って通信を行う，実環境での運用が難しい．これらの代表例として，Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups [7](以下 SFKA-DCG) が提案されている．この方式は，グループ鍵を，2 分木のキーツリーを用いて管理する．グループメンバの数が N の時，各グループメンバは，$N+1$ 個の鍵を管理する．これらの鍵を用いて DH 鍵交換のプロトコルのような冪乗演算を行うことで，全てのグループメンバはグループ鍵となる値を共有できる．グループ管理サーバを用いないので，管理者への情報漏洩リスクはない．また，グループメンバの変更や鍵更新の際，全てのグループメンバに対し，パケットをブロードキャストするため，各グループメンバが異なるプライベート空間に存在している場合への適用が難しい．

そこで，本論文では鍵管理が容易なサーバ型鍵管理に着目し，鍵管理要件に加えて，グループ管理者であってもグループ鍵を取得できず，グループメンバが異なるプライベート空間に所属していても適用可能なグループ鍵共有方式を提案する．また，提案方式では，鍵の共有を行う際に公開鍵証明書を用いて認証を行い通信する方式とは別に，セキュアな E2E（End-to-End）経路を用いる方式を用意することで，条件付きではあるが，公開鍵証明書の管理コストが不要となる．提案方式は，グループ管理サーバ GMS(Group Management Serever) と，グループメンバによって構成されている．提案方式では，GMS とグループメンバのそれぞれが乱数を生成し，それらを用いて，グループメンバがグループ鍵を生成する．これらの乱数はそれぞれ別の経路によって配布される．そのため，グループ管理者であってもグループ鍵を知ることができない．また，GMS が生成，配布する乱数は，グループメンバの変更時や，あらかじめ決められたタイミングで更新され，各メンバに配布される．これらの処理を行うことで，提案方式では，鍵管理要件を満たすことができる．また，端末が所属する空間を問わず，グループ鍵共有を行うことができる．これらの提案方式の一部を実装し，動作検証と評価を行い，実用上問題ない性能で実現できることを確認した．

以後，2 章では関連技術とその課題について述べる．3 章では提案方式について詳細に説明する．4 章では提案法式の実装について説明を行い，5 章では定量的評価と定性的評価の 2 つの側面から評価し，最後に 6 章でまとめる．

第2章　関連技術

本章では，グループ鍵を用いる場合に重要な鍵管理要件を述べる．既存技術として集中管理型の GSAKMP（Group Secure Association Key Management Protocol）と，分散型鍵管理の SFKA-DCG(Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups) の概要と課題について述べる．

2.1　鍵管理要件

グループ鍵管理プロトコルの目的は，機密性や認証のために必要なデータを最新の暗号化状態でグループメンバに提供することであり，一般的に以下のような鍵管理要件が存在する [9].

- 鍵はあらかじめ定めた期間で定期的に更新を行う．
- 鍵データは正規ソースからのみ入手可能で，正しいグループメンバのみに送られる．
- 鍵管理プロトコルはリプレイ攻撃*1や DoS（Denial of Service）攻撃に対して安全である．
- 参加や退会が容易であり，新たに参加したメンバはグループに参加する前の鍵データへアクセスできない（後方秘匿性）．また退会したメンバはそれ以降の鍵データへアクセスすることができない（前方秘匿性）．

2.2　GSAKMP

2.2.1　GSAKMP の概要

GSAKMP(Group Secure Association Key Management Protocol) は，グループコミュニケーションにおけるセキュリティフレームワークである．GSAKMP では，アクセス制御ルールを決定し，このルールに基づいたユーザ認証を行うことで，グループの確立を行う．GSAKMP は，GO(Group Owner)，グループ鍵などを管理するサーバである GCKS(Group Controller Key Server)，GM(Group Member) の 3 つの要素によってグループ管理を行う．GSAKMP では，GC/KS と GM となるユーザ端末が公開鍵証明書を所有していることを前提としており，これを用いて GCKS-GM 間の認証を行う必要がある．

GO は鍵の更新処理や，グループの招待方法などを含む，セキュリティポリシー及び，グループの開始を GCKS に通知する役割を持つ．GSKS は GO が設定したポリシーを基に，鍵生成，鍵配

*1 ユーザのログイン時や参加申請時にネットワークに流れるデータを盗聴し，そのデータを認証サーバへ送ることで不正な通信をする行為．

布，鍵の再生成及び，GM の管理を行う．このとき，GM は，ポリシーに基づいた，適切なグループ鍵の運用が求められる．GCKS は，GM の登録や退会等の管理ができる．

2.2.2 GSAKMP の鍵共有方式

GSAKMP におけるグループ鍵共有シーケンスを図 1 に示す．新たにグループへ参加する Node は，GO から招待を受けていることが前提である．即ち，管理者がメンバを決定する．

（1）Request to Join

GO から招待を受けた Node は GCKS へ Request to Join を送信する．これはグループへの参加を要求するメッセージであり，このメッセージの中にはユーザの公開鍵証明書や招待されたときに付与されているグループ ID などが含まれている．

（2）Key Download

参加申請を受け取った GCKS は新たに参加するユーザの公開鍵証明書の検証を行う．この検証が成功し，正しいユーザであることを確認した場合 GCKS からユーザへ，GTPK（Group Traffic Protection Key）と Rekey Key を配布する．GTPK（Group Traffic Protection Key）はグループ通信のデータを暗号化するグループ鍵であり，Rekey Key は，定期的なグループ鍵の更新や，グループメンバの変更の際に，GCKS からのマルチキャストパケットを暗号化するための鍵である．

（3）Key Download Ack

2 つの鍵を受け取ったユーザは応答を返す．これらの一連の処理が完了したタイミングで招待されたユーザは当該グループのグループメンバとなる．

グループ鍵の更新の際，GCKS はグループ鍵を更新した後，Rekey Key を用いた暗号化を行い，マルチキャストパケットを用いて，グループメンバ全体に配布を行う．グループ鍵の更新の際，グループの変化がない場合は，グループ鍵のみ配送を行う．GM の参加や退会が行われた場合，グループ鍵と Rekey Key の双方を配送する．退会したユーザはそれ以降の通信内容が閲覧できず，新規参加ユーザは参加する前の通信内容を閲覧できないため，前方秘匿性と後方秘匿性の両方を満たすことができる．また，GSAKMP では，鍵共有シーケンスメッセージに対しシーケンス番号を付与することで，リプレイ攻撃対策を行う．さらに，Cookie [10] と呼ばれるデータを用いて，GCKS のメンバ登録処理以前に，GM の認証を行うことで DoS 攻撃対策を行う処理を，オプションで定義することができる．このため，GSAKMP は鍵管理要件を満たすことができる．

2.2.3 GSAKMP の課題

GSAKMP では GCKS と GM の両者が公開鍵証明書を所有しているため相互認証を行い，鍵管理要件を満たす点が特徴である．そのため，外部のコミュニケーションサーバを利用したアプリケーションの通信においても，GCKS によって配布されるグループ鍵を用いてコンテンツを暗号化できるため，第三者に情報を読み取られることはない．しかし，GSAKMP はサーバ管理者を信用したプロトコルであるため，GCKS は，グループ鍵と Rekey Key を知っていることから，サーバ管理者にグループメンバの通信内容を閲覧される可能性がある．また，GO が鍵の更新処理やメ

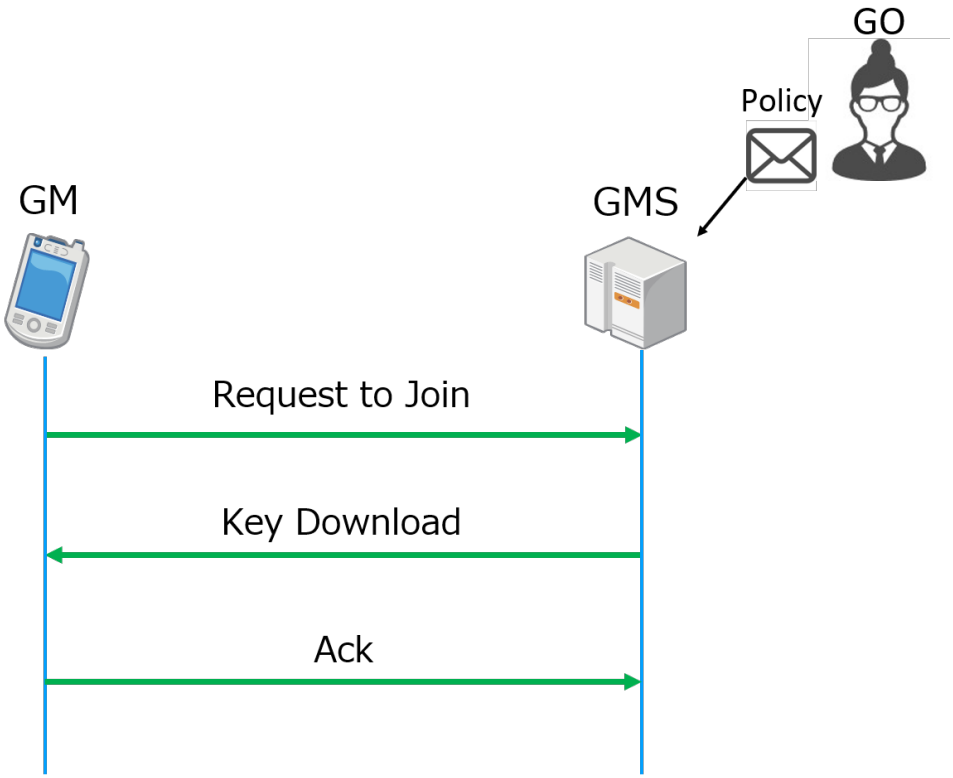

図 1　GSAKMP におけるグループ鍵共有シーケンス

ンバの招待方法の設定を行うことができるため，サーバ管理者が悪意のあるメンバをグループに混入させることも可能である．また，すべてのグループメンバが公開鍵証明書を所有させる必要があるため，証明書の管理コストが必要となる．

2.3　SFKA-DCG

2.3.1　SFKA-DCG の概要

SFKA-DCG は，分散型の鍵管理方式であり，2 分木のキーツリーを用いてグループ鍵を管理する．利用される用語を以下に示す．

M_i　　：グループメンバ
h　　：キーツリーの高さ
$< l,v >$　：深さ l，v 番目のツリーノード
N　　：グループメンバ数
p　　：素数
$f()$　　：$\alpha^k \mod p$
$K_{<l,v>}$　：$< l,v >$ の鍵 (詳細は後述)
$BK_{<l,v>}$　：$f(K_{<l,v>})$

$K_{<l,v>}$ と，$BK_{<l,v>}$ (Blinded Key) を用いた冪乗演算を行うことにより，ルートノードの鍵 $K_{<0,0>}$ をグループ全体で共有する．$K_{<0,0>}$ をハッシュ関数にかけることによりグループ鍵を生成する．グループメンバの参加や退会が発生すると，スポンサーと呼ばれる特定のグループメンバが，変更が行われたノードの K 及び BK の再計算を行い，全てのグループメンバに向けて，BK を含めたパケットのブロードキャストを行う．ブロードキャストされたパケットを受信した各グループメンバは，自身が所有している K, BK を用いてグループ鍵生成を行う．グループメンバが変更する度，$K_{<0,0>}$ が変更されるため，前方秘匿性と後方秘匿性を満たすことができる．グループに参加するユーザは，鍵共有シーケンスにおける受信パケットの検証を，公開鍵証明書を用いて行う必要がある．

2.3.2 SFKA-DCG の鍵共有方法

SFKA-DCG によって生成されるキーツリーを，図 2 に示す．グループメンバは，キーツリーの葉ノードに対応しており，各グループメンバは，自身の対応する葉ノードからルートまでのパス上に存在する K と，全てのノードの BK を所有している．また，すべての $K_{<l,v>}$ は，以下のように計算される．

$$\begin{aligned} K_{<l,v>} &= (BK_{<l+1,2v+1>})^{K_{<l+1,2v>}} \mod p \\ &= (BK_{<l+1,2v>})^{K_{<l+1,2v+1>}} \mod p \\ &= \alpha^{K_{<l+1,2v>}K_{<l+1,2v+1>}} \mod p \\ &= f(K_{<l+1,2v>}K_{<l+1,2v+1}) \end{aligned} \tag{2.1}$$

これは，$<l,v>$ の子ノードの K と BK を用いて，$K_{<l,v>}$ を計算できることを示している．

図 2 の M_1 を例にとると，M_1 は，$\{K_{<3,0>}, K_{<2,0>}, K_{<1,0>}, K_{<0,0>}\}$ と，$\{BK_{<0,0>}, BK_{<1,0>}, \ldots, BK_{<3,5>}\}$ を所有している．これらのうち，$\{K_{<2,0>}, K_{<1,0>}, K_{<0,0>}\}$ は，$K_{<3,0>}$ と，$\{BK_{<3,1>}, BK_{<2,1>}, BK_{<1,1>}\}$ を用いて $l = 2$ の K から順に計算することで導出可能である．

2.3.3 SFKA-DCG の課題

Rekey 処理のために BK のブロードキャストを行うため，グループメンバは自分以外のメンバに対してコネクションを張り続ける必要がある．このため，異なるプライベート空間に所属する端末によってグループが形成されている場合，複数の E2E 通信経路の管理の点などから，グループの規模が大きくなるにつれ，実運用がより困難になるといった課題がある．また，全てのグループメンバが公開鍵証明書を取得していることを前提としていることから，公開鍵証明書の管理コストが必要となる．

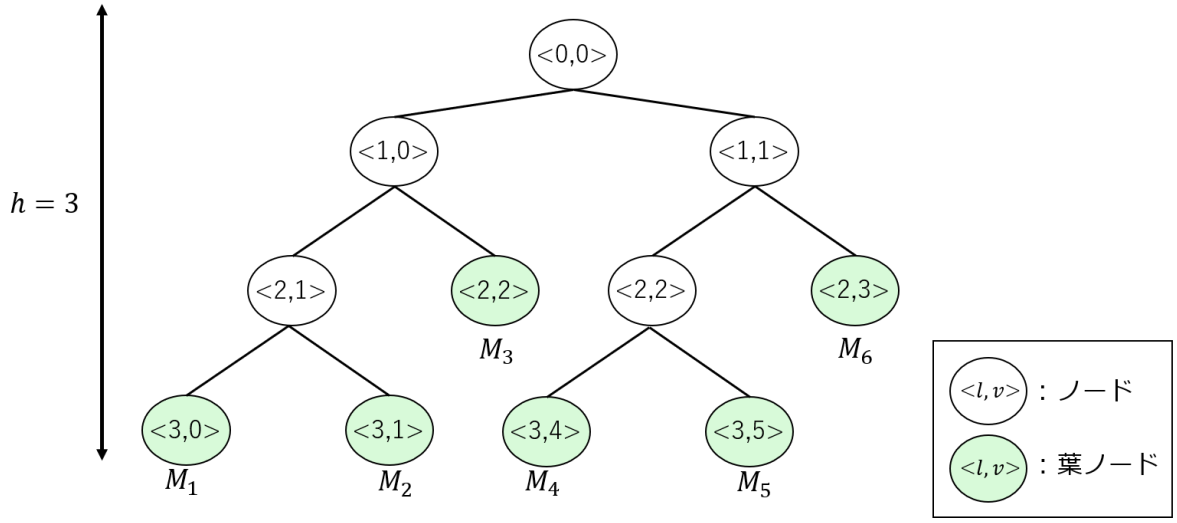

図 2　SFKA-DCG によるキーツリー

第3章　提案方式

本章では，提案する鍵共有方式について述べる．

3.1　提案方式の概要

本提案では，グループ鍵の運用が容易なサーバ型に着目し，グループ管理サーバを用いるが，管理者がグループ鍵を取得することができない仕組みを導入する．管理装置を用いるため，鍵の更新や管理が容易であるという集中方式の利点をそのまま継承する．グループ鍵（GK）の生成は，管理装置で生成される乱数（RN_{gms}）と，ユーザサイドで生成される乱数（RN_{node}）を，異なる配送経路でグループメンバに配送し，各グループメンバがそれらの乱数を組み合わせ，そのハッシュ値を取ることで行われる．グループ管理者が知っているのは管理装置が生成した RN_{gms} のみであり，GK はグループメンバ側で生成するため，管理者に通信内容を読み取られることがない．

RN_{gms} は，定期的に，かつ，グループメンバの参加，退会時に鍵の更新を行う．これにより，前方秘匿性と後方秘匿性を実現できる．

RN_{node} は，グループメンバを招待する際に，グループ招待者と招待される端末間で共有される．RN_{node} は GK 生成の要素であるため，グループメンバ以外に知られることが無いよう，安全に配送を行われる必要がある．これを実現する方式として，公開鍵証明書を用いて認証，暗号化を行い RN_{node}の 配送を行う，公開鍵証明書方式と，エンド端末間で安全な E2E（End-to-End）経路を生成して RN_{node} の配送を行う，E2E 方式が存在する．これらの RN_{node} 共有方式については以降の章で説明を行う．GK の更新は，RN_{gms} 主導で行われるため，RN_{node} の有効期間は 1 年などの長期間でよい．提案方式のシステム構成を，図 3 に示す．提案方式は，GMS(Group Management Server) と，GM(Group Member) によって構成される．GMS はグループを管理するサーバであり，グローバル空間上に設置され，GMS-GM 間で共有される RN_{gms} の生成，配送や，グループに関する情報の管理を行う．GMS が管理するグループの情報として，グループ ID，RN_{gms}(バージョン情報含む) の他，ログインステータス，FQDN(端末名) や，アドレス情報などがある．GMS は，グループメンバの参加や退会が発生した場合，RN_{gms} を更新し，ログインステータスがオンの端末に対して，再生成した RN_{gms} を，各メンバに向けて配送する．最後に，RN_{gms} を受信したグループメンバが，$hash(RN_{node}|RN_{gms}|$ グループ名）（RN_{node}，RN_{gms}，グループ名のハッシュ値）をとることで，グループ鍵を生成する．

GM はグループに参加しているメンバであり，GM 間で共有する乱数 (RN_{node}) の配送や，GK の生成，GMS に向けての新規メンバの通知，グループメンバの招待を行う．

グループメンバへの招待は，招待権限をもつメンバのみが行うため，GMS による悪意のあるグループメンバの混入を防ぐことができる．

なお，公開鍵は暗号化アルゴリズム上セキュリティの課題がないことを前提とする．

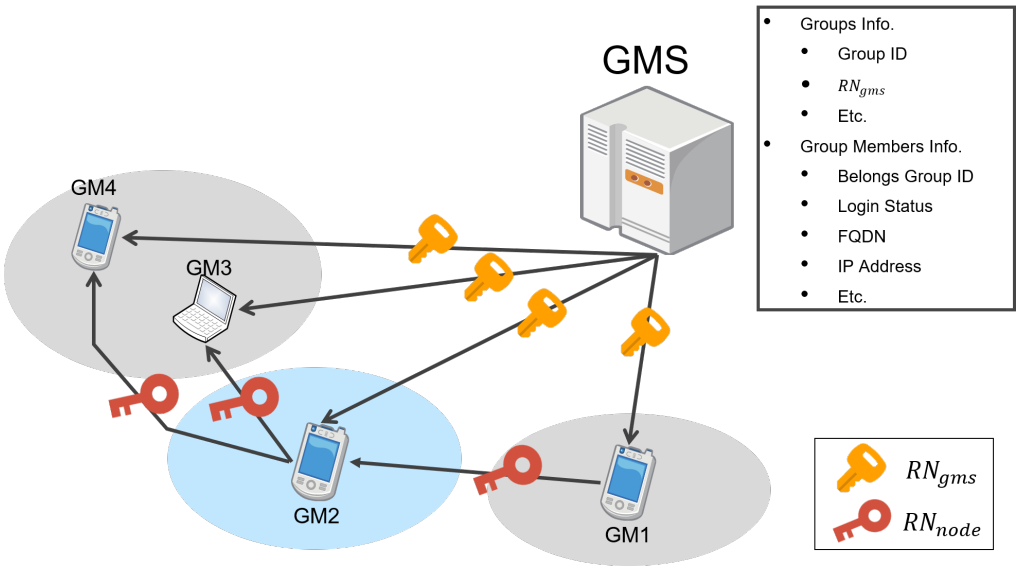

図 3　提案方式のシステム構成

3.1.1　GMS が所有するグループ情報

GMS が管理する情報は大別して，グループに関する情報と，グループメンバに関する情報がある．

グループに関する情報は以下の通りである:

GroupID　　: グループを識別するための値

RN_{gms}　　: グループメンバと GMS 間で共有される乱数．GMS によって生成．

$RN_{gms}version$　　: GMS によって生成される乱数のバージョン

グループメンバに関する情報は以下の通りである:

GroupID　　: 所属グループを識別するための値

LoginStatus　　: GMS とのコネクションが確立されているかを示す値

GroupMemberID　　: グループメンバを識別する値 (FQDN 等)

InviterFlag　　: 招待者権限を所有しているかを示す値

3.1.2　グループ鍵生成要素について

提案方式では，$hash(RN_{node}|RN_{gms}|GN)$ によってグループ鍵を生成する．グループ鍵の構成要素については以下の通りである．

RN_{node} : グループメンバ間でのみ共有される乱数．一人目のグループメンバが生成
RN_{gms} : 既出であるため省略
GN : グループメンバ間でのみで共有される．一人目のグループメンバが定義したグループ名

RN_{node} と GN は，グループ生成時に，一人目のグループメンバが生成を行い，後述する RN_{node} 共有シーケンスによって新規のグループメンバに配布，共有される．RN_{gms} は，グループメンバの参加，退会，あらかじめ決められたタイミングで，鍵の生成及び更新を行い，GMS によってグループメンバに配布される．

3.1.3 DoS 攻撃とリプレイ攻撃対策

DoS 攻撃対策として，GMS はセキュリティツールの導入やアドレスによるフィルタリングを行う．リプレイ攻撃に対応するため，後述する鍵共有におけるシーケンスパケットには，シーケンス番号が付与する．

3.2 鍵共有方式

以降では，グループに招待する端末を招待者，グループに招待される端末を被招待者とする．また，RN_{gms} 共有を安全に行うための前提として，グループ鍵の要素である RN_{gms} を安全に配送するために，GMS-端末間では公開鍵証明書やパスワードを利用した認証が完了しており，共通鍵による暗号化通信を行えることとする．

グループ鍵共有では，グループ生成シーケンス，RN_{node} 共有シーケンス，RN_{gms} 共有／ GK 生成シーケンスが存在する．これらのシーケンスで取り扱う記号を以下に示す．

GN : 一人目のグループメンバが定義したグループ名．
GID : GMS がグループを識別するための識別子．GM 内で GN と結び付けられている．
GMI_{first} : グループメンバのアドレス等の情報．
$GMI_{invitee}$: 被招待者のアドレス等の情報．
$Cert_{inviter}$: 招待者の公開鍵証明書．
$Cert_{invitee}$: 被招待者の公開鍵証明書．
$PW_{inviter}$: 招待者のパスワード．
$PW_{invitee}$: 被招待者のパスワード．
RN_{gms}^{vN} : バージョン N の，RN_{gms}．

3.2.1 グループ生成

図 4 にグループ生成シーケンスを示す．グループを作成する端末は GN を定義した後，GMI を Group Generation Request として GMS へ送信する．GMS は，GID の生成と GMI の登録を行い，

GID を応答（Ack）として返す．その後，グループを最初に生成した招待者は RN_{node} を生成することでグループ生成を完了する．

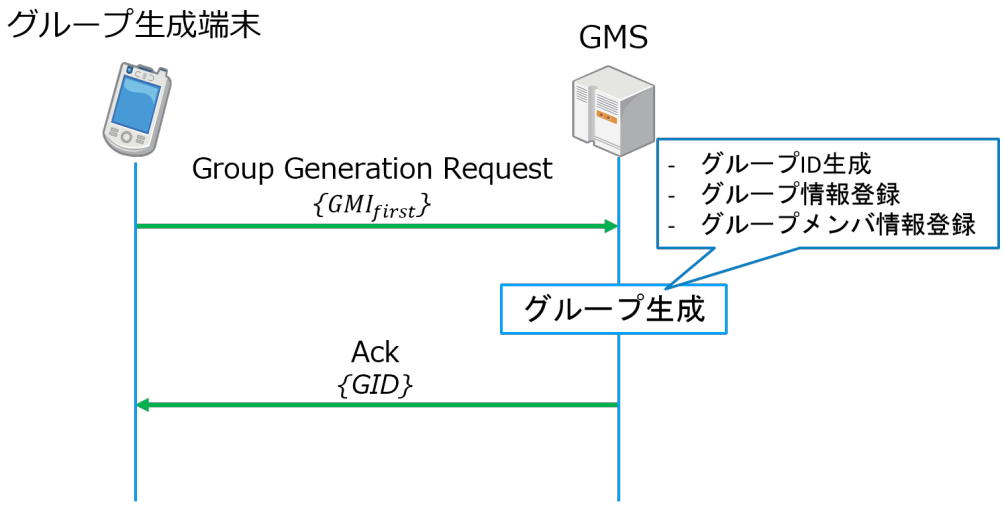

図 4　グループ生成シーケンス

3.2.2　RN_{node} の共有

RN_{node} の共有を実現する方式として，ユーザ端末に公開鍵証明書を所有させる公開鍵証明書方式と，E2E(End-to-End) 通信が可能なセキュアネットワークを使用する E2E 通信方式の 2 通りが考えられる．公開鍵証明書方式では，RN_{node} を共有する際に，中継用サーバを経由させることができるが，公開鍵証明書方式の取得，管理にコストがかかる．一方，E2E 通信方式では，公開鍵証明書を取得する必要はないが，E2E 通信を行うセキュアなネットワークが必要となる．

(1) 公開鍵証明書方式

図 5 に公開鍵証明書方式での RN_{node} 共有シーケンスを示す．招待者-被招待者間で，公開鍵証明書を用いた認証（TLS などを用いる）が完了していることを前提とする．招待者と被招待者は，公開鍵を用いた認証と暗号化ができるため，中継用サーバを介した RN_{node} の共有が可能である．招待者と被招待者は事前にパスワードを共有することで，端末を操作しているユーザの正当性を確認することも可能である．招待者は被招待者に対し，グループ招待パケットとして，Group Invitation Request を送信する．このパケットには，グループ生成時にユーザが定義するグループ名と，GMS によって生成されるグループ ID が含まれる．この時，公開鍵認証処理を連続で行わせることで，端末に過剰な負荷をかけることを目的とした DoS 攻撃の対象となることを防ぐため，招待者側の公開鍵証明書は送付しない．

被招待者は，事前に共有したパスワードを入力し，招待の応答として，Group Invitation Response を送信する．Group Invitation Response には被招待者の公開鍵証明書や入力したパスワードを含める．応答を受け取った招待者は，被招待者の公開鍵証明書やパスワードの検証を行う．検証が成功した場合，RN_{node} を被招待者の公開鍵を用いて暗号化し，RN_{node} Distribution として，招待者の

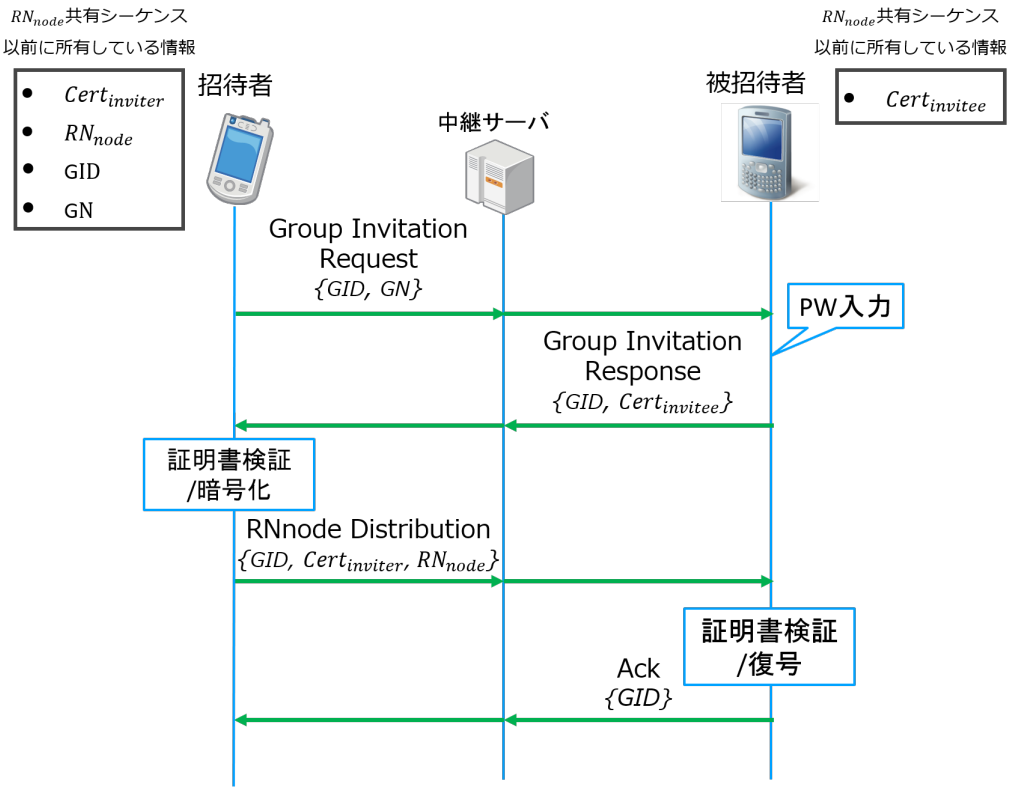

図 5　公開鍵証明書方式の RNnode 共有シーケンス

公開鍵証明書とともに被招待者へ送信する．このパケットには，被招待者に招待者権限を与えるかどうかのフラグも含める．被招待者は，招待者の公開鍵証明書を検証し，正しい場合，自身の秘密鍵を用いて RN_{node} の復号を行う．最後に招待者に対して応答を返すことにより，RN_{node} の共有が完了する．

(2)E2E 通信方式

図 6 に E2E 方式での RNnode 共有シーケンスを示す．セキュアな E2E 通信方式の例として，NTMobile [12] [13] [14] が挙げられる．招待者と被招待者は，お互いの認証を行うために，各自のパスワードを事前に設定し，共有しておく．

招待者と被招待者は，セキュアな E2E 通信経路を確立するためのシグナリングを行い (Secure Communication Signaring)E2E 経路を確立する．これ以降のすべての通信は，セキュアな E2E 経路を使用して行う．

招待者は被招待者に対して，グループ ID とグループ名を含めた Group Invitation Request を送信する．招待を受信した被招待者は，事前に共有したパスワードを入力し，Group Invitation Response として，招待者へ送信する．応答を受け取った招待者は，入力されたパスワードをもとに被招待者の正当性を確認した後，被招待者側のパスワードを入力し，RN_{node} と，招待者権限の有無を示すフラグともに，被招待者へ送信する (RNnode Distribution)．RN_{node} を受信した被招待者は，最後の応答 (Ack) を返して，共有シーケンスを終了する．

公開鍵証明書方式とは異なり，一度経路を確立してしまえば，証明書の認証を行うことなく RN_{node}

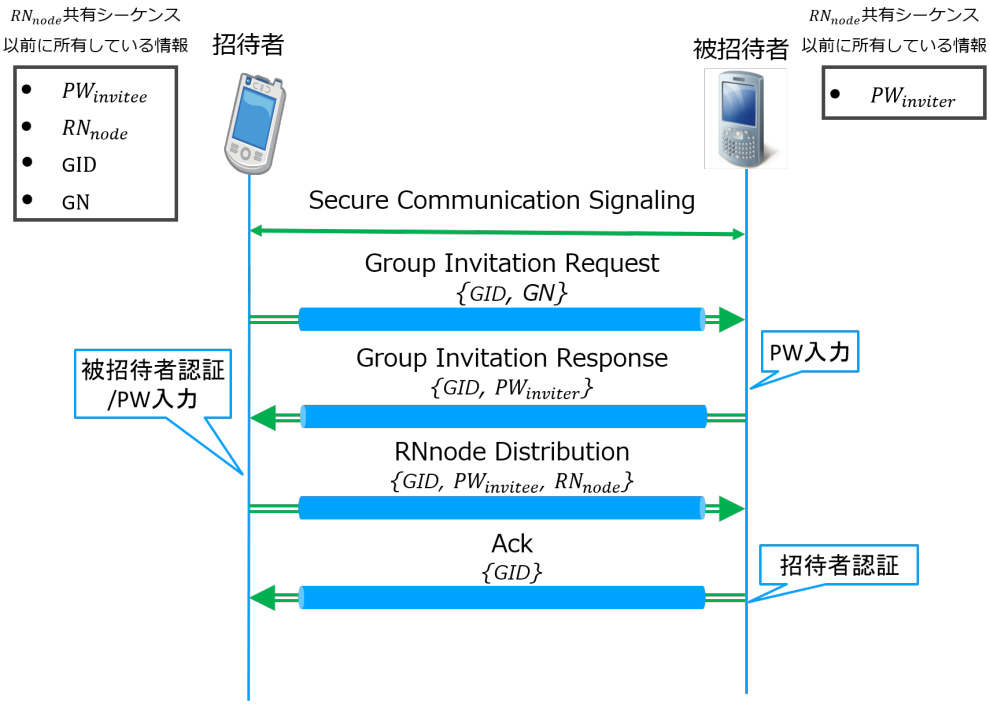

図 6　E2E 方式の RNnode 共有シーケンス

を共有することが可能である．

3.2.3　RN_{gms} の共有と GK の生成

図 7 に RN_{gms} の共有シーケンスを示す．RN_{node} の共有が完了すると，招待者は，グループ ID や被招待者の情報 (IP アドレスや FQDN，招待者権限の有無など) を含めた情報を，Group Member Notification パケットとして GMS へ送信する．Group Member Notification を受信した GMS は，当該グループの RN_{gms} を生成し，被招待者の情報と共にデータベースに登録する．次に，招待者と GMS 間で，グループ鍵共有シーケンス実行前に共有しておいた共通鍵を用いて RN_{gms} を暗号化し，被招待者を含めた全ての GM に対して配送する (RN_{gms} Distribution)．RN_{gms} を受信した GM は，GMS に対して応答 (Ack) を送信する．最後に，RN_{node} と RN_{gms} とグループ名のハッシュ値をとることで GK を生成する．GMS が所有している情報は RN_{gms} のみであり，RN_{node} と $GroupName$ を得ることはできない．そのため，サーバ管理者がグループ鍵を入手することなく，当該グループメンバのみで GK の共有が可能となる．

3.2.4　プロトコルレベルで DoS 攻撃の対策を行う場合の RN_{gms} 共有

3.2.3 の方式では，GMS への DoS 攻撃対策をプロトコルレベルで行っていない．プロトコルレベルで DoS 攻撃対策を施す場合の RN_{gms} 共有シーケンスを，図 8 に示す．初めに，招待者が生成した Nonce 値を Group Member Notification に付与して，GMS へ送信する．Group Member Notification を受信した GMS は乱数を生成し，乱数と受信した Nonce 値を用いて Cookie を生成する．その後，Cookie Download として，Cookie を招待者に対して送信する．招待者は，受信した Cookie 値を Group Member Notification に付与して，再度送信を行う．GMS は，Cookie 値や $GMI_{invitee}$ 情報が

正しいかを認証し，成功した場合は，RN_{gms} の生成／更新や，被招待者のメンバ登録を行い，グループメンバ全体に対して，RN_{gms} の配送を行う．これらの処理により，GMS は Cookie による認証が成功するまで，DB へのアクセスを行わないため，DoS 攻撃の影響を緩和することができる．

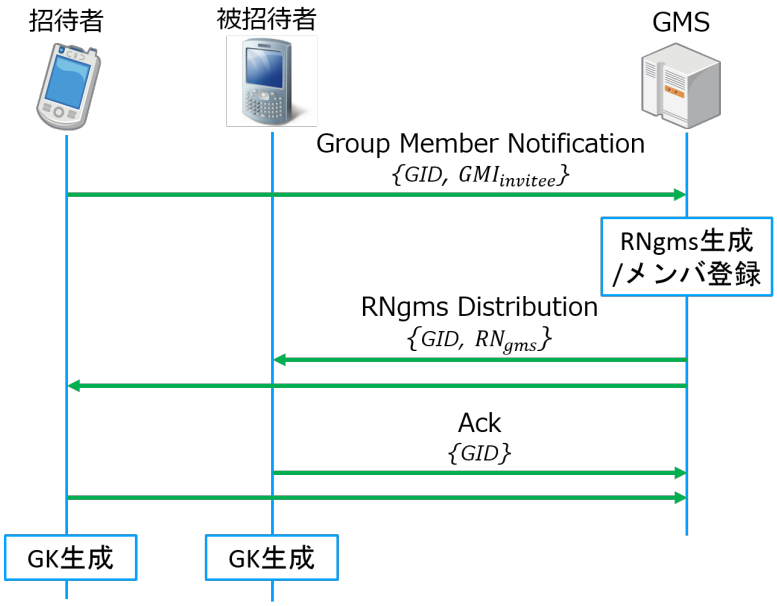

図 7　RN_{gms} 配布/GK 生成シーケンス

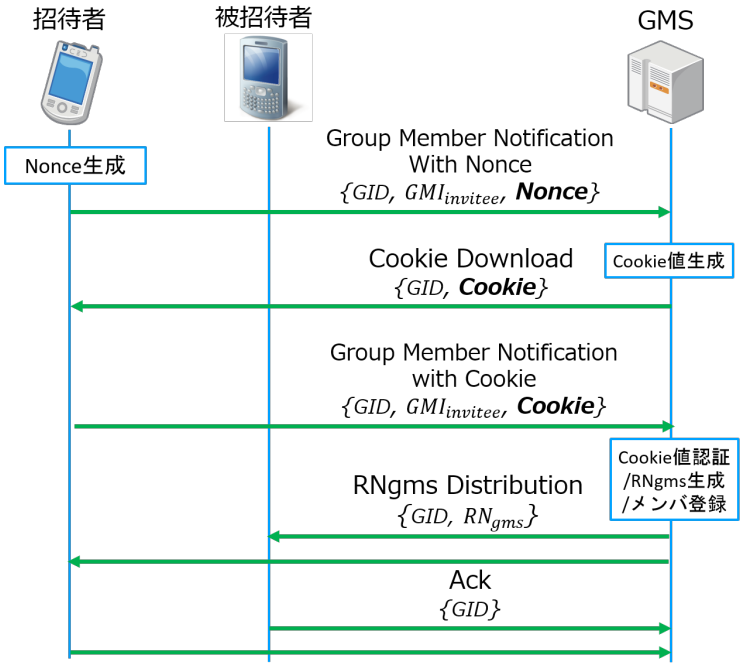

図 8　DoS 攻撃対策を施した RN_{gms} 配布シーケンス

3.3 RN_{gms} の更新

鍵管理要件を満たすため，GMS は RN_{gms} の更新を行う必要がある．RN_{gms} の更新は，一定の時間が経過した場合や，グループメンバの参加と退会の際に行われる．一定時間が経過した際の鍵更新は，GMS が RN_{gms} の更新を行い，RN_{gms} Distribution を行うことで実現する．参加と退会処理に関しては，以降で説明する

3.3.1 参加

グループへの参加は，3.2.3 の処理が実行されることで行われる．グループメンバが参加したタイミングで RN_{gms} は更新されるため，新たに参加するメンバは，参加前の RN_{gms} によって生成された GK を知ることができない．これにより後方秘匿性を満たすことができる．

3.3.2 退会

図 9 に，GM を退会する際の処理が行われる際の構成図を示す．この図では，GM3 が RN_{node} を所持したまま退会した場合を想定している．RN_{gms}^{v1} は，更新前の RN_{gms} であり，RN_{gms}^{v2} は更新後の RN_{gms} である．以下に，GM3 が退会する際の処理を示す．

① GM3 は GMS に対して，自身の退会通知である Group Withdrawal Notification を送信する．この時，Group Withdrawal Notification 中の識別子は，GM3 自身のものを格納する．
② 退会通知を受けて，GM3 に関しての情報を，GMS の DB から削除する．
③ RN_{gms} の更新 (RN_{gms2}) を行い，GM3 以外のグループメンバに配送する．

最後に，各 GM が RN_{gms2} を用いて GK を生成する．退会したユーザは，RN_{node} を所有したまま退会した場合でも，RN_{gms2} を知ることができないため，前方秘匿性を確保することができる．

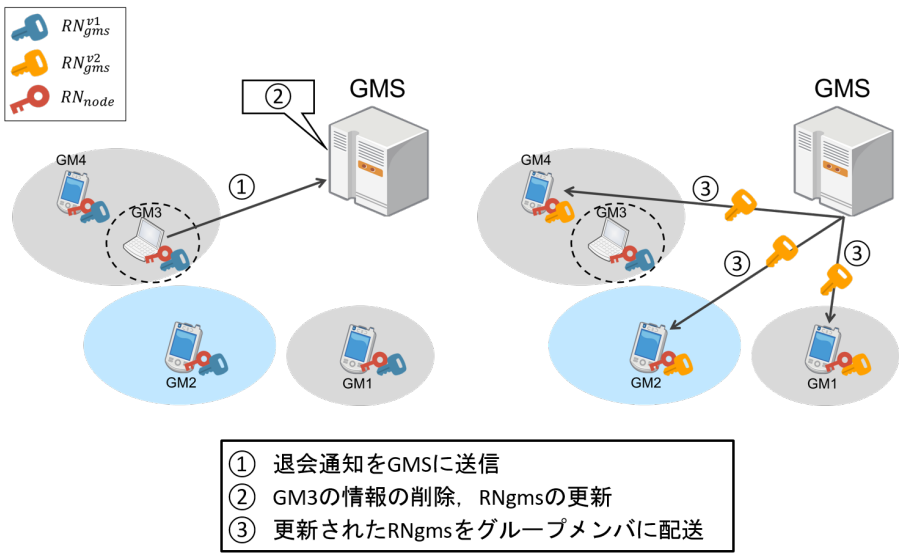

図 9　メンバ退会時の RN_{gms} 更新処理

3.4 RN_{gms} のバージョン管理機能

RN_{gms} は GMS によって配送されるが，ネットワークに接続されていない端末や，電源が入っていなかった端末が存在していた場合，RN_{gms} の更新が行われず，最新の GK を生成できない問題が発生する．図 10 にバージョン管理機能を用いた RN_{gms} の更新処理を示す．GM1 が電源をオフにしている間にグループメンバの変更が発生し，GMS において RN_{gms}^{v1} が更新され，RN_{gms}^{v2} になったケースを想定する．

招待者から GMS へ Group Member Notification が送信されると，GMS において RN_{gms}^{v1} が RN_{gms}^{v2} に更新され，GMS から各グループメンバへ配布されるが，電源がオフである GM1 には RN_{gms}^{v2} を配布できない．そのため，GM1 の電源をオンにした時，GM1 から GMS へ RN_{gms} Inquiry を送信する．これのメッセージは，GM1 が所有している RN_{gms} と，最新の RN_{gms} のバージョンが一致しているかを確認するものである．これを受け取った GMS はユーザ認証を行うとともに，GM1 が所有している RN_{gms} のバージョンを確認する．図 10 では，GM1 が所有している RN_{gms} は RN_{gms}^{v1} であり，最新の RN_{gms} である，RN_{gms}^{v2} とはバージョンが異なっていることが確認される．バージョンが異なっていた場合，GMS は最新の RN_{gms} を，RN_{gms} Inquiry 送信した端末に対して RN_{gms} Distribution として送信する．RN_{gms}^{v2} を受け取った GM は GMS へ応答を返し，GM が $hash(RN_{node}|RN_{gms}^{v2}|GN)$ を行い，新しい GK を生成することでグループメンバ間の暗号化通信が可能となる．

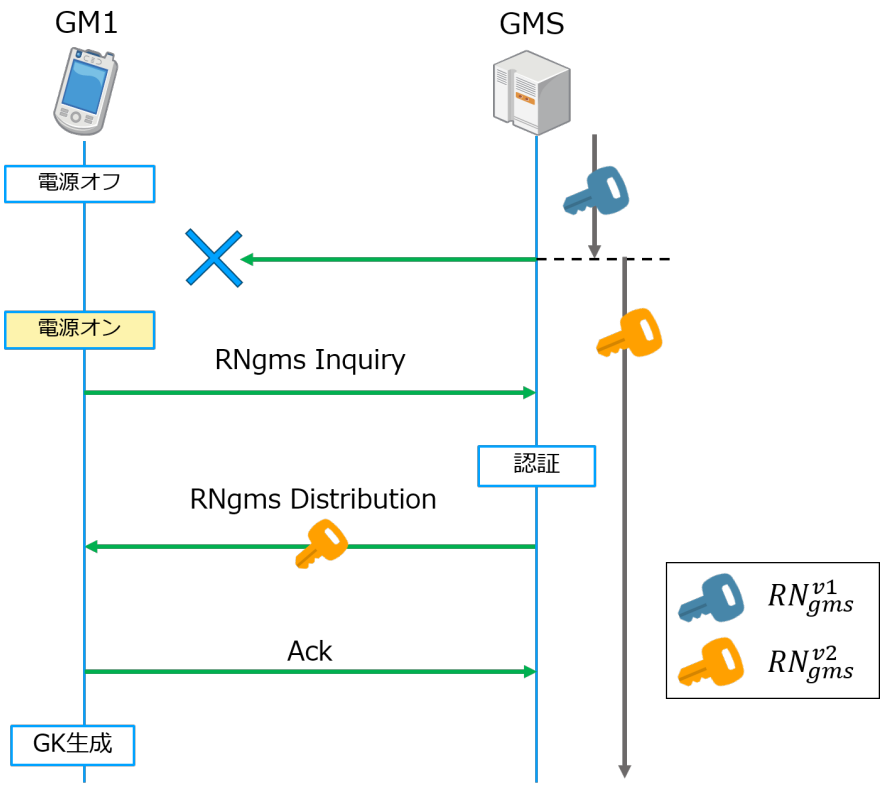

図 10　バージョン管理機能を用いた RN_{gms} の更新処理

第4章　実装

本章では，提案方式の一部を実装したためその動作検証，性能評価，および既存技術との比較について述べる．今回実装した方式は，E2E 通信方式である．本方式を実装するにあたり，セキュアな E2E 通信実現する方式として，NTMobile を使用した．

4.1　NTMobile

本論文の提案内容と関連しないため，NTMobile の詳細な説明は省略する．

NTMobile に関する機器を以下に示す．

- NTM 端末
  NTMobile の機能を持った端末
- DC(Direction Coordinator)
  NTM 端末のアドレス情報の管理や，E2E 通信用のトンネルの構築指示を行う
- AS(Account Server)
  NTM 端末の認証や，DC-NTM 端末間の暗号化通信における共通鍵の配布を行う
- RS(Relay Server)
  NTM 端末間で，E2E 通信ができない場合や，一般端末との通信を行う際に，パケットを中継する役割を担う

ここで，NTM 端末は，事前に AS に端末情報を登録しているものとする．

NTM 端末は初回起動時に AS にログインし，DC の FQDN 及び DC との通信に用いる共通鍵を取得する．NTM 端末-AS 間では，メールアドレスとパスワードを用いた認証や，公開鍵を用いた認証を行う．次に，NTM 端末-DC 間では，NTM 端末は DC に対して自身の実 IP アドレス等の情報を送信し，DC がそれらを登録した後，仮想アドレスを NTM 端末に通知する．以降は NTM 端末-DC 間でキープアライブが行うことで，常に双方向通信を行える状態にする．

NTMobile を用いた通信を行う際は，上に記した処理が行われた後，DC がトンネル経路の構築指示を行うことで，エンド端末間でセキュアなトンネル経路が生成される．

NTMobile では，制御メッセージや，カプセル化パケットに対しシーケンス番号や MAC（Message Authentication Code）が付与されており，パケットの改竄検知や，リプレイ攻撃対策がとられている．

また，現在研究室で実装されている NTMobile に関する機器は，C 言語によって記述されている．尚，今回は，RN_{gms} 共有部分は DoS 攻撃対策を付与していない方式で実装している．

4.2　鍵共有方式の実装

本提案方式における，3.2.1 グループ生成，3.2.2 RN_{node} の共有，3.2.3 RN_{gms} の共有と GK の生成の 3 つのシーケンスを，C 言語を用いて，GMS は Ubuntu14.04 上で，招待者/被招待者は Ubuntu16.04 上で実装した．尚，今回の実装では，RN_{gms} のバージョン管理機能や，あらかじめ決められた期間での RN_{gms} の更新の実装は行っていない．本実装の GMS は，今後 NTMobile 側に提案方式の機能を追加することを想定し，DC に鍵共有機能を追加することで実現している．

4.2.1　DC 上に実装した Group Management Server

図 11 に DC 上に実装した GMS のモジュールの構成を示す．現在実装されている DC は，トンネル構築の指示を行うための NTMobile デーモンと，NTM 端末のアドレス情報を動的に登録するために BIND を利用した DNS サーバ，トンネル構築指示のために必要なアドレス情報などを登録するためのデータベースによって構築されている．今回は DC の NTMobile デーモンに，グループ鍵共有を行うための Group Management モジュールのプロトタイプ実装を行った．Group Management モジュールは，グループ鍵共有シーケンス用の制御パケットの処理や，グループに関する情報をデータベースに登録する役割を持つ．

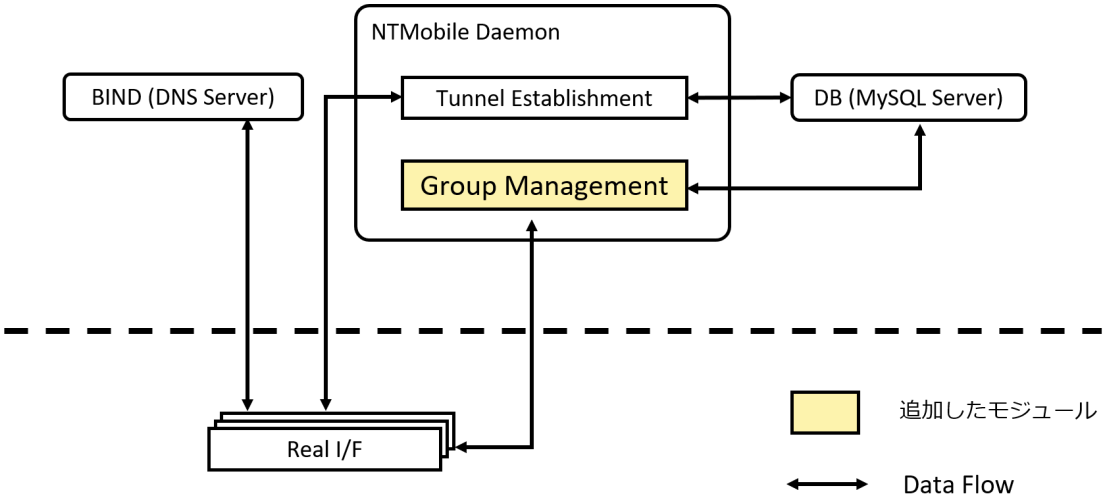

図 11　DC 上に実装した GMS のモジュールの構成

4.2.2　グループメンバ用端末

図 12 にグループメンバ用端末モジュールの構成を示す．グループメンバ用端末は，NTM 端末上にグループ鍵共有機能を実装することで実現している．端末の NTMobile の機能は，NTMobile をアプリケーションレイヤで使用可能にする，NTMFW（NTMobile Framework）[15] を用いて実装されている．NTMFW モジュールの詳細な説明は，本論文の趣旨から外れるため，追加及び変更を加えたモジュールのみの説明とする．

NTM 端末上に，グループメンバ（招待者，被招待者）間の制御パケットを処理する Key Negotiation between GM モジュールと，RN_{node} の生成や GK 生成など，鍵に関する処理を行うための Key モジュール，NTMobile を起動し，E2E 通信経路確立処理を起動するための NTMobile Initialization モジュールを作成した．NTMFW に対して，GMS―グループメンバ間の制御パケットを処理する Key Negotiation between GMS モジュールを追加した．また，Packet Manipulation モジュールに対して変更を加えた．Packet Manipulation モジュールは，パケットの MAC 付与や検証及び暗号化／復号を行い，受信したパケットの種類応じて，その種類に対応したモジュールに処理を振り分ける機能を持つ．このモジュールに対し，受信したパケットが，グループ鍵共有に関する制御パケットであった場合，Key Negotiation between GMS モジュールに処理を振り分けるように変更を加えた．

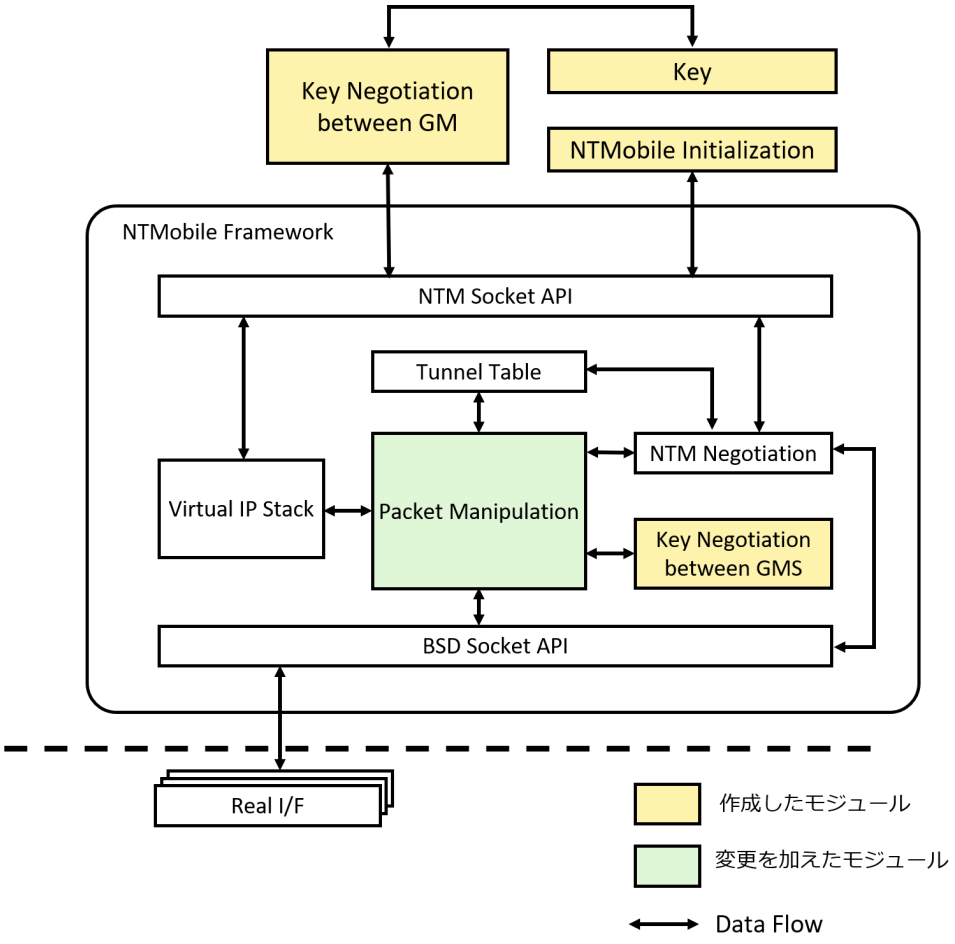

図 12　グループメンバ用端末のモジュール構成

第5章　動作検証と評価

実装した提案方式 (E2E 通信方式) のグループ鍵共有の動作検証と性能評価を仮想マシンを用いて行った．また，既存技術との比較を行った．

5.1　動作検証

動作検証を行った際の，ホストマシンの構成と仮想環境の構成を，それぞれ表 1，表 2 に示す．また，検証時のネットワーク構成を図 13 に示す．

仮想マシン Virtual Box[⋆1] を用いて，1 台のホストマシン上に，GM1(招待者)，GM2(被招待者)，GMS(DC)，AS の 4 台を構築した．全てのマシンは同一ネットワーク上に所属している (図 13)．

以上の環境の下で，グループ鍵共有処理を実行し，グループ鍵が GM 1 と GM2 で共有されていることを確認した．

表 1　ホストマシンの構成

	ホストマシン
OS	Windows10 64bit
CPU	Intel Core i7-7700
Memory	8GB

表 2　仮想環境の構成

	GMS	GM
OS	Ubuntu 14.04	Ubuntu 16.04
Kernel version	3.13.0-24-generic	4.15.0-36-generic
Memory	1GB	1GB

5.2　性能評価

性能評価では，図 14 に示す箇所の処理時間を計測した．計測の際は，clock_gettime() を用いて，各箇所においける試行回数 10 回の平均時間を取得した．

[⋆1] https://www.virtualbox.org/

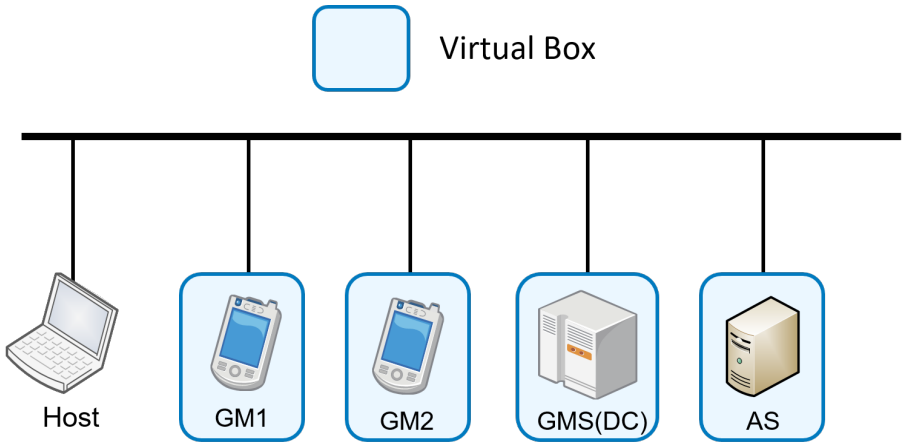

図 13　動作検証環境のネットワーク構成

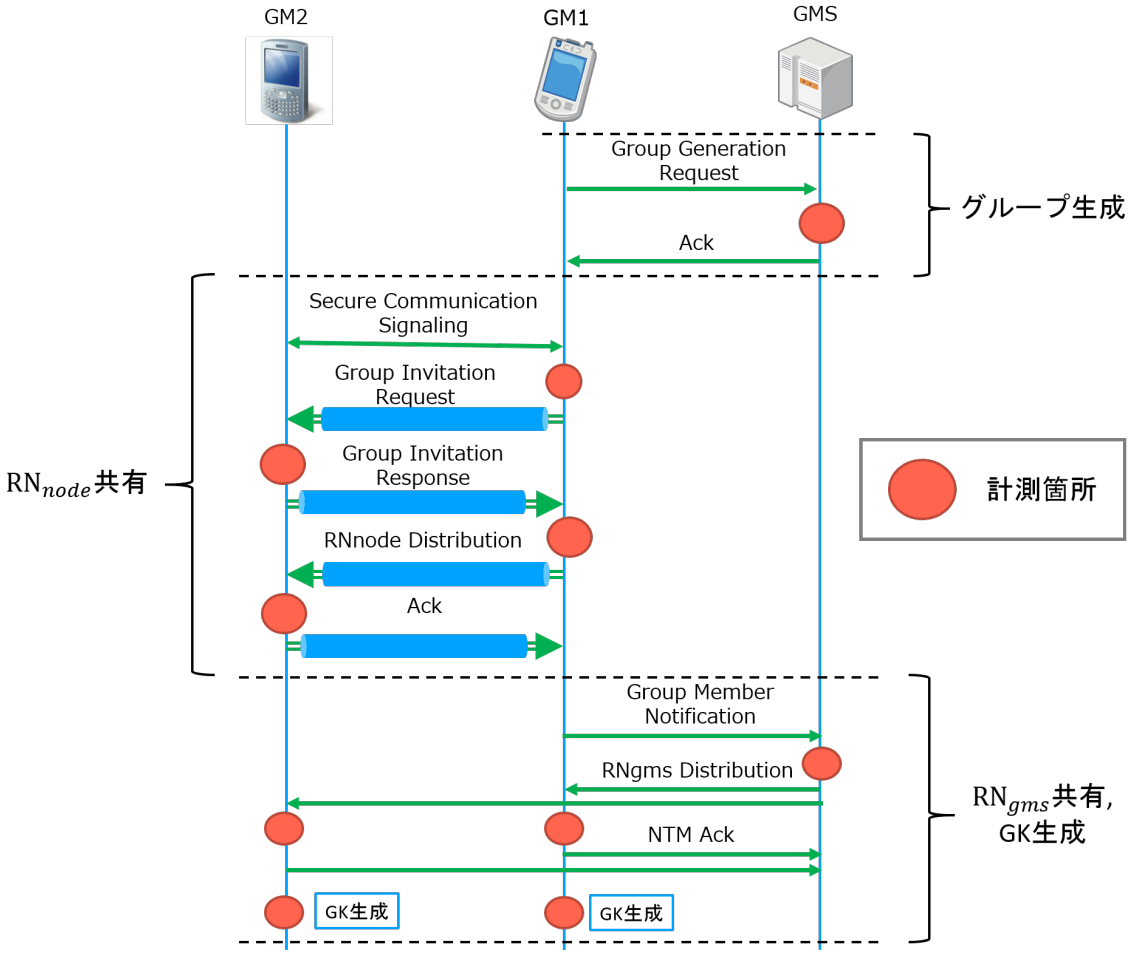

図 14　処理時間の計測箇所

5.3 乱数共有部の実装と計測

本論文にて提案する方式の 1 つである公開鍵証明書方式の RN1 共有部を実装し仮想環境において動作検証を行った．表 3 - 表 5 は，グループ生成，RN_{node} 共有，RN_{gms}共有と GK 生成 の 3 つのシーケンスについて，各機器ごとの処理時間を計測した結果である．また，表 6 は，RTT を 25ms[⋆1] とし，シーケンスごとの処理時間を導出したものであり，各シーケンスにおいて，最初のシーケンス制御パケット（Group Generation，Group Invitation Request，Group Member Notification）を送信した機器が，最後の制御パケットを受信するまでの時間である．表 3 は，GMS における，グループ生成と RN_{gms} 共有における測定結果であり，DB アクセスの処理時間と全体の処理時間を示す．表 4 と，表 5 は，招待者と被招待者における，RN_{gms} 共有及び RN_{node} 共有部分の測定結果であり，NTMoble の処理にかかった時間 (表中の NTM) と全体の処理時間を示している．

表 3 から，GMS の処理時間のうち，60〜70%が DB（MySQL 5.5.61）へのアクセスに要する時間であることがわかる．これは，DB のチューニングや，GMS の性能を向上させることで対応できる．

表 4 と表 5 から，RN_{gms} の共有に比べ，RN_{node} 共有の処理時間が長いことがわかる．これは，全体の処理時間のうち，約 80%を占める NTMobile が原因である．RN_{node} 共有は，グループ招待時に行うのみであり，半永久的に更新されることはない．

今回の動作検証は，インターネットの RTT を約 25ms と仮定しても，もっとも通信回数が多い RN_{node} 共有に要する時間は，約 50ms であり，グループ鍵共有シーケンスにかかる時間に比べ，ネットワーク遅延が支配的である．このことから，提案方式の処理時間は十分短いと考えられる．また，被招待者がグループに参加し，GK を共有するまでの時間である，RN_{node} 共有と，RN_{gms} 共有及び GK 生成の時間に着目すると，合計で約 90ms である．これは，グループに参加したいユーザが，グループに参加するまでに要する時間を示している．Web サイトの応答時間について，遅延が 1000ms 以内であれば，途切れることなく作業が進んでいると感じることから [16]，提案方式のグループ鍵共有は，ユーザの作業を妨げることのない処理時間の許容範囲内であり，実用上問題ないと考えられる．

表 3　GMS の処理時間計測結果

	DB アクセス [ms]	全体 [ms]
グループ生成	1.92	3.01
RN_{gms} 共有	5.11	7.17

[⋆1] 3G 回線による (www.softbank.jp) への 4 日間の Ping による実測値．

表 4　招待者の処理時間計測結果

	NTM[ms]	全体 [ms]
RN_{node} 共有	2.58	3.25
RN_{gms} 共有と GK 生成	-	0.374

表 5　被招待者の処理時間計測結果

	NTM[ms]	全体 [ms]
RN_{node} 共有	3.04	3.72
RN_{gms} 共有と GK 生成	-	0.393

表 6　シーケンス別の処理時間（**RTT25ms** 想定）

	処理時間
グループ生成	28.0
RN_{node}共有	57.0
RN_{gms}共有と *GK* 生成	32.9

5.4　関連技術との比較

表 7 にグループ鍵の共有に着目した関連技術との比較を示す．項目は以下の 3 つであり，比較対象は GSAKMP 及び SFKA-DCG とした．

（1）管理者が読めないように暗号化されているか．
（2）鍵管理要件を満たしているか．
（3）鍵管理が容易であるか
（4）公開鍵証明書管理コスト

評価項目（1）はサーバ管理者がセキュリティホールになり得ることから必要な項目であり，評価項目（2）は一般的にグループ鍵を使用する際に必要となる項目である．評価項目（3）は，実環境で運用する際に，鍵の共有，更新処理が容易であるかを表す項目であり，複数のプライベート空間に渡って通信が行われる場合も想定する．評価項目（4）は，公開鍵証明書に管理コストがかかるかを示す項目である．

表 7　関連技術との比較

	項目（1）	項目（2）	項目（3）	項目（4）
GSAKMP	×	○	○	×
SFKA-DCG	○	△	×	△
提案方式	○	○	○	△

GSAKMP はグループ鍵の生成や各種攻撃に対する対策が定義されており，更新期間も設定されているため前方秘匿性や後方秘匿性を満たすため鍵管理要件を満たしている．また，サーバによる管理を行っているため，項目（3）を満たしている．しかし，鍵サーバ GCKS においてグループ鍵 GTPK と更新鍵 Rekey Key のどちらも生成しているため，サーバ管理者が通信内容を読み取れる恐れがあり評価項目 (1) を満たしていない．また，すべてのグループ端末が公開鍵証明書を持っ

ている必要があるため，項目（4）をみたしていない．SFKA-DCG は，キーツリーとべき乗演算によって，グループ鍵を生成する方式であり，グループメンバに変更があった場合は，特定のグループメンバが自身が，所有している鍵 (K) の更新と，キーツリーノードの鍵 (K，BK) の再計算を行い，再計算後の鍵をグループメンバ全体で共有することで，前方秘匿性や後方秘匿性を満たし，適切な鍵更新期間を設定できる．しかし，DoS 攻撃やリプレイ攻撃の対策は取られておらず，鍵管理要件をすべて満たしていない．さらに，鍵の更新を行う際に，グループメンバ全体へブロードキャストを行わなければならず，複数の異なるプライベート空間で通信が行われる実環境では，運用が難しいことから，項目 3 を満たしていない．項目（4）については，セキュアな E2E 経路を用いることで公開鍵証明書を用いることなく実現できるが，そのような経路を確保できない場合は，公開鍵証明書をすべてのグループメンバに所有させる必要がある．

提案方式では，RN1 をユーザ間で安全に共有しグループ鍵を生成するのでサーバの管理者がグループ鍵を生成することができず，管理者が読めないようにコンテンツの暗号化を行うことができる．評価項目（2）において RN_{gms} をあらかじめ定めた期間により更新を行い，かつ新たなメンバの参加や退会ごとに RN_{gms} の更新を行っているため鍵管理要件を満たしている．また，サーバによる管理を行っているため，項目（3）も満たしている．項目（4）については，SFKA-DCG と同様の理由である．

第6章　結論

既存のグループ鍵共有方式では，サーバ管理者にグループ通信の内容を傍受される恐れや，グループメンバが増加するに従い，管理する鍵数とそれに伴うグループ鍵導出のための演算回数が増加してしまうため，大規模グループでの使用が不向きであるものがあった．そこで本論文では，管理サーバを用いた，生成元が異なる 2 つの乱数 (RN_{gms}, RN_{node}) からグループを生成させるグループ鍵共有方式を提案した．乱数は GMS と GM で生成され，それぞれ，GMS-GM 間と GM-GM 間の，別々の経路で GM に配送され，それらを用いて GK を生成することで，GMS にグループ通信の内容を傍受されることなく通信を行うことができる．GM の参加や退会時に，RN_{gms} の更新を行うことにより，前方/後方秘匿性を満たすことができる．さらに，RN_{gms} のバージョン管理機能を利用することで，何らかの理由で GM が GMS と通信できない場合でも，グループ鍵の更新を行えるようにした．

また，提案方式の，グループ鍵生成，RN_{node} 共有，RN_{gms} 共有と GK 生成の実装と動作検証及び性能測定を行った．さらに，他の提案方式との定性評価を行った．定性評価では，関連技術との比較を行い，提案方式がよりセキュアで鍵管理コストが低いグループ鍵共有を行えることを示した．

謝辞

本研究を進めるにあたり，多大なるご指導，ご鞭撻を賜りました，指導教官である名城大学理工学研究科情報工学専攻　渡邊晃教授に，心から感謝いたします．　また，本研究を進めるにあたり，適格なご意見並びにご助言を賜りました，名城大学理工学研究科情報工学専攻　鈴木秀和准教授に心から感謝いたします．
本論文を作成するにあたり，快く副査を引き受けていただきました名城大学理工学研究科情報工学専攻　吉川雅弥教授，旭健作准教授に心より感謝致します．
最後に，本研究を進めるにあたり，数々の有益なご助言を賜りました，渡邊研究室および鈴木研究室の諸氏に感謝致します．

参考文献

[1] Electronic Frontier Foundation: Secure Messaging Scorecard. https://www.eff.org/ja/node/82654 (2018 年 12 月 20 日アクセス).

[2] Edward Snowden: the whistleblower behind the NSA surveillance revelations — US news — The Guardian https://www.theguardian.com/world/2013/jun/09/edward-snowden-nsa-whistleblower-surveill (2019 年 1 月 5 日アクセス).

[3] Michael Steiner, Gene Tsudik and Michael Waidner: Key Agreement in Dynamic Peer Groups, IEEE Transactions on Parallel and Distributed Systems, vol.11, pp.769–780 (2000).

[4] Jim Alves-Foss: An Efficient Secure Authenticated Group Key Exchange Algorithm for Large And Dynamic Groups, IN PROC. 23 RD NATIONAL INFORMATION SYSTEMS SECURITY CONFERENCE, pp254–266 (2000)

[5] Y. Kim , A. Perrig , G. Tsudik: Group Key Agreement Efficient in Communication, IEEE Transactions on Computers, vol.53, pp.905–921 (2004).

[6] R.K. Balachandran ; B. Ramamurthy ; Xukai Zou ; N.V. Vinodchandran: CRTDH: An Efficient Key Agreement Scheme for Secure Group Communications in Wireless Ad Hoc Networks, Communications, ICC 2005. 2005 IEEE International Conference, vol.2, pp.1123–1127 (2005)

[7] Y. Kim , A. Perrig , G. Tsudik: Simple and fault-tolerant key agreement for dynamic collaborative groups, Proceedings of the 7th ACM conference on Computer and communications security, pp.235–244 (2000).

[8] H. Harne, U. Meth, A. Colegrove, G. Gross: GSAKMP: Group Secure Association Key Management Protocol, RFC4535, IETF (2006).

[9] M. Baugher, R. Canetti, L. Dondeti, F. Lindholm: Multicast Security (MSEC) Group Key Management Architecture,RFC 4046,IETF (2005).

[10] P.Karn, W.Simpson: Photuris: Session-Key Management Protocol, RFC2522, IETF (1999).

[11] ITpro Special 広域ネットワークで注目の SD-WAN http://special.nikkeibp.co.jp/atcl/ITP/17/nttpc0314/

[12] 上醉尾一真，鈴木秀和，内藤克浩，渡邊晃:IPv4/IPv6 混在環境で移動透過性を実現する NTMobile の実装と評価，情報処理学会論文誌，Vol. 54，No. 10，pp. 2288–2299 (2013).

[13] H. Suzuki, K. Naito, K. Kamienoo, T. Hirose and A.Watanabe: NTMobile: New End-to-End Communication Architecture in IPv4 and IPv6 Networks, Proceedings of the 19th Anuual International Conference on Mobile Computing and Networking (Mobicom2013)，pp. 171–174 (2013).

[14] 納堂博史，杉原史人，鈴木秀和，内藤克浩，渡邊　晃 : NTMobile の実用化に向けた統合的枠組の検討，情報処理学会研究報告モバイルコンピューティングとユビキタス通信研究会（MBL），Vol. 2015 MBL 77, No. 20, pp. 1–8 (2015).

[15] 納堂博史，鈴木秀和，内藤克浩，渡邊　晃：エンドツーエンド通信をアプリケーションレベルで可能にする通信ライブラリの実現と評価，情報処理学会論文誌，TBD.

[16] Meggin Kearney: RAIL モデルでパフォーマンスを計測する — Web — Google Developers https://developers.google.com/web/fundamentals/performance/rail (2019 年 2 月 5 日アクセス)

研究業績

国際会議（査読あり）

（1）R. Suganuma, H. Suzuki, K. Naito and A. Watanabe: Proposal on Application Layer Multicast, Based on a Ring Shaped Route, *The Tenth International Conference on Mobile Computing and Ubiquitous Networking (ICMU 2017)* , pp. 79 – 80 Toyama, Japan, Oct.2017.

研究会・大会等（査読なし）

（1）菅沼良一，納堂博史，鈴木秀和，内藤克浩，渡邊 晃 : NTMobile におけるマルチキャスト機能の実現，情報処理学会第 79 回全国大会講演論文集，pp. 285 – 286, Sep. 2017.

（2）菅沼良一，納堂博史，鈴木秀和，内藤克浩，渡邊 晃 : IP アドレスに基づきリング状経路を生成するアプリケーションレイヤマルチキャストの提案，情報学ワークショップ 2017(WiNF2017) 論文集, No.PC-29，Nov.2017.

（3）菅沼良一，鈴木秀和，内藤克浩，棚田慎也，渡邊 晃 : 物理ネットワークを意識した リング状アプリケーションレイヤマルチキャストの提案，情報処理学会第 80 回全国大会講演論文集, pp. 275 – 276，Mar.2018.

（4）菅沼良一，納堂博史，棚田慎也，鈴木秀和，内藤克浩，渡邊 晃:リング状経路を用いたアプリケーションレイヤマルチキャストの提案，マルチメディア，分散，協調とモバイル (DICOMO2017) シンポジウム論文集，pp. 1715 – 1720，Jun. 2017.