単一の鍵で多重帰属できるグループファイル共有システム

佐々木 啓[†1]　長澤 悠貴[†1]　脇田 知彦[†1]
毛利 公美[†2]　白石 善明[†1]　野口 亮司[†3]

グループウェアの代表的な機能にグループファイル共有があるが，ファイルをサービス提供者のサーバに保管するため，サービス提供者の不正が懸念される．利用者側でファイルを暗号化することでサービス提供者が閲覧できないようにできるが，利用者は所属するグループが増えるほど鍵管理の負担が大きくなる．我々は既に，グループメンバの変更が容易で，利用者が単一の鍵で複数グループに所属して暗号化ファイルを復号できる方式を提案している．本論文では，我々が提案しているグループファイル共有プロトコルの実装について述べる．まず，プロトコルを組み込んだシステムの機能と，鍵構造を管理するためのデータベースについて示す．次に，実装したグループファイル共有システムの各主体の操作画面について説明する．実装したシステムでは，サービス提供者及びグループに所属しない利用者は共有ファイルを閲覧できないことを確認した．

A Group File Sharing System Enabling Multiple Association by Single Key

Kei Sasaki[†1]　Yuuki Nagasawa[†1]　Tomohiko Wakita[†1]
Masami Mohri[†2]　Yoshiaki Shiraishi[†1]　Ryoji Noguchi[†3]

Groupware is more popular as a method of information sharing in the organization. Group file sharing system is one of the representative function in groupware. In this system, users may have uneasiness in illegal act of the service provider. A solution for the uneasiness is that shared file is encrypted and stored in service server then group member shares its decryption key. In this case, user has the same number of group decryption key as multiple associating groups. We have already proposed a group file sharing protocol and group management protocols using ElGamal threshold cryptosystem and secret sharing scheme, which can decrypt file decryption key shared in any associated groups by one group key. This paper, we give a system construction and DB for implementing our proposed protocol. In addition, we show operation view of each entity and implemented user-friendly GUI.

1. はじめに

SaaS 型のグループウェアサービスを用いた組織内情報共有が浸透し始めている[1][2]．グループウェアの機能の 1 つにファイル共有があり，これを使うと組織のグループ内でファイルを共有することができる．このとき，共有するファイルはグループウェアサービス提供者側のサーバに保管されるため，サービス提供者の不正が懸念される．不正への対応策としてはサービス利用者が共有するファイルを暗号化する方法がある．その一つに，共通鍵暗号方式でファイルを暗号化し，公開鍵暗号方式でファイルの復号に必要になる鍵を共有する方法があるが，メンバの一人がグループから離脱する際に，公開鍵／秘密鍵ペアの再生成と再配布，ファイルの復号に用いる鍵の再暗号化が必要になることや，複数グループに多重帰属するメンバの管理する鍵が増えるなどの理由で鍵管理のコストが大きくなる．そのために多くのグループファイル共有サービスは通信路を暗号化するのみでファイルを暗号化していない．

我々は柔軟なグループ構築が可能で，かつ利用者が複数のグループに所属する場合でも鍵管理の負担を軽減できる暗号プロトコル[3]を提案し，サービスとして実利用可能なものであることをシミュレーションにより確認している[4]．本論文ではこのプロトコルに基づくグループファイル共有システムの実装について述べる．

本論文の構成は次のようになる．2 章でクラウド型グループファイル共有システムの概要とファイル共有に必要な機能について述べ，SaaS 型グループファイル共有での要件をまとめる．3 章で既存の安全なファイル共有方法について述べ，4 章で我々が提案している“単一の鍵で多重帰属できるグループファイル共有のためのプロトコル”について述べる．5 章でシステムの設計と各機能での動作手順，DB 設計について述べ，6 章で実際に作成したシステムについて述べる．7 章でサービス提供者及びグループに所属しない利用者はファイルへアクセス出来ないことを確かめ，8 章でまとめと今後の課題を述べる．

2. SaaS 型グループファイル共有

グループファイル共有は，ファイルをグループに所属するメンバ間で共有する機能を持つ．グループは例えば，営業部，開発部というように複数存在し，2 つ以上のグループに所属するメンバもいる．

†1　名古屋工業大学
Nagoya Institute of Technology
†2　岐阜大学
Gifu University
†3　(株)豊通シスコム
Toyotsu Syscom Corp.

2.1 SaaS 型とオンプレミス型のグループファイル共有の違い

オンプレミス型のグループファイル共有ではファイルを自組織で管理する．このとき非正規ユーザは組織外にのみ存在するため，図 1 のように組織内ネットワークの境界で防御することで非正規ユーザのファイルへの不正アクセスを防ぐことができる．しかし，自組織で管理を行うためアクセス権限の管理や障害対策／対応にコストがかかる．

SaaS 型のファイル共有サービスではファイルの管理をサービス提供者が行うため，サービス利用組織の管理コストはオンプレミス型に比べて少ない．このとき，サービス提供者はサービス利用組織に所属していない，すなわち，ファイルへのアクセス権を持たない非正規ユーザである．図 2 のように，外部の非正規ユーザは境界で防御できるが，サービス提供者はファイルに対し不正なアクセスが可能である．

ここでは，サービス提供者がファイルへアクセスできないように，サービス利用組織側でファイルを暗号化してサーバに保存する SaaS 型グループファイル共有の実現方法について検討する．

2.2 SaaS 型グループファイル共有のためのファイル暗号化モデルの定義

SaaS 型グループファイル共有のモデルを図 3 に示す．サービス利用組織には複数のグループがあり，“一般ユーザ”がこれに所属している．一般ユーザはサービスを利用して，ファイルをクライアント側で暗号化し“ファイル保管サーバ”へアップロードする．同じグループに所属するユーザはファイルをダウンロードし復号できる．グループメンバを管理するユーザが“グループ管理者”であり，全てのグループとユーザを管理するのが“管理者”である．管理者は一般ユーザの中からグループ管理者を任命し，グループ管理者は特定のグループに対する一般ユーザの追加と離脱を管理する．図 3 のモデルは，以下の主体で構成される．

- 一般ユーザ
  - グループファイル共有サービスを利用するユーザ
  - 0 以上のグループに所属する
  - 所属するグループでファイルを共有できる．ファイルを暗号化してからアップロードし，ダウンロードしてから復号する
- 管理者
  - グループウェア利用組織の全ての一般ユーザ・グループ管理者・グループを管理するユーザ
  - サービス利用組織ごとに 1 人以上いる
  - グループの作成・削除を管理する
  - 全てのグループに対し，メンバの追加／離脱を管理する
  - 全ての鍵の更新ができる

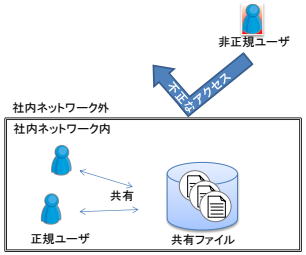

図 1　オンプレミス型ファイル共有

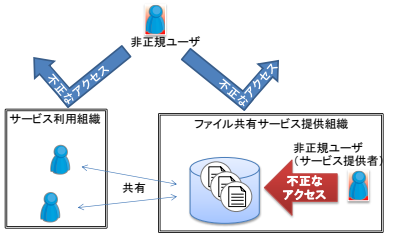

図 2　SaaS 型ファイル共有

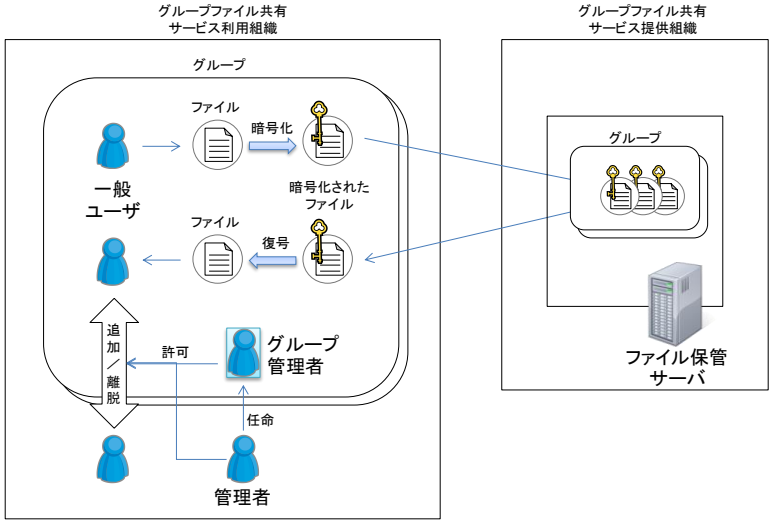

図 3　SaaS 型グループファイル共有システムのためのファイル暗号化モデル

  - 一般ユーザをグループ管理者に任命できる
- グループ管理者
  - 特定のグループを管理するユーザ
  - 管理するグループに対し，メンバの追加／離脱を管理する
  - 管理するグループに所属するユーザの鍵の更新ができる

- グループ管理者は各グループに 0 人以上存在する

- ファイル保管サーバ
  - グループファイル共有サービスを提供し，全ての暗号化されたファイル，各グループに所属する一般ユーザの情報やグループの情報を保持する

一般ユーザ，グループ管理者，管理者はファイル共有サービス利用組織に，ファイル保管サーバは SaaS 型グループファイルサービス提供組織に属している．

2.3 機能

グループファイル共有システムに必要な機能は次のようになる．

［ファイル共有］

ファイルをファイル保管サーバにアップロード，ダウンロードしグループ内でファイルを共有する機能．ユーザは所属するグループでアップロード・ダウンロードできる．ファイルの暗号化と復号は一般ユーザが行うため，サービス提供者はファイルへアクセスできない．

［グループ作成／削除］

ファイルを共有するグループを作成／削除する．グループを削除すると，そのグループにアップロードされたファイルは全て消去される．

［グループへの追加／離脱］

グループへのメンバの追加／離脱を行う．グループに追加されたメンバはグループ内のファイルを復号可能となり，ファイルの共有ができる．一方，離脱したメンバはファイルを復号できないため，グループ内のファイルを閲覧できない．

［鍵更新］

ファイルの暗号化・復号に関する鍵を更新する．セキュリティの観点から鍵は定期的に更新されるべきである．ユーザの業務に影響しないよう，鍵更新中にユーザがシステムを使っていても支障がでない程度の時間で完了することが望ましい．

2.4 暗号化する SaaS 型ファイル共有の要件

2.1 節より，SaaS 型グループファイル共有ではファイル保管サーバでサービス提供者がファイルにアクセスできないことが，ファイルを暗号化して保管する SaaS 型グループファイル共有の要件である．これを実現するために暗号方式を取り入れると，暗号処理に必要なコストが問題となる場合がある．具体的には，クライアント側で暗号化または復号に必要な計算コストが大きい場合，計算資源に乏しい端末でサービスを利用できない可能性がある．暗号方式を取り入れる場合，セキュリティの観点からサービス利用者が持つ鍵は定期的に更新されるべきである．鍵の更新に必要なコストが大きい場合，鍵更新中のシステム利用ができない時間が増える可能性がある．よって，暗号化，復号，鍵更新などに必要な計算コストが小さいことも，暗号化する SaaS 型ファイル共有の要件と言える．以上より，ファイルを暗号化して共有する SaaS 型ファイル共有では，以下の要件が挙げられる．

【要件 1】 ファイル保管サーバではファイルにアクセスできない

【要件 2】 暗号化，復号，鍵管理のサービス利用者側のコストが少ない

3. 既存の安全なファイル共有方法

グループファイル共有において要件 1 を満たすファイル共有方法として以下の実現方法が考えられる．

- 公開鍵暗号方式を用いた方法
- 属性ベース暗号を用いた方法

公開鍵暗号方式を用いた方法は，グループごとに公開鍵／秘密鍵ペアを作成し，グループに所属するメンバが秘密鍵を共有することでファイル共有を実現する．具体的には，ファイルをアップロードするときに共通鍵を作成し，その鍵でファイルを暗号化する．暗号化に利用した共通鍵は，グループの公開鍵により暗号化する．これにより，グループに所属するメンバ以外はグループの秘密鍵を持たず，共通鍵を復号できないため，グループファイル共有が実現できる．しかし，グループからメンバが離脱する場合，グループの公開鍵／秘密鍵ペアを再生成し新しい秘密鍵をメンバに配布するとともに，共通鍵を新しい公開鍵で再暗号化する手続きが必要になる．また，メンバが管理する鍵の数が所属するグループの数に比例して増え，利用者にとって負担になる．よって，要件 2 を満たさない．

次に，属性ベース暗号を用いた方法を検討する．属性ベース暗号[5]とは，暗号文や秘密鍵に復号条件を埋め込み，指定された属性を持つ利用者が暗号文を復号できる方式である．例えば，{“名古屋工業大学”かつ“大学院生”}という条件を指定された暗号文は，名古屋工業大学の大学院生は復号できるが，“名古屋工業大学の教員”や“岐阜大学の大学院生”は復号できない．このとき，“名古屋工業大学”の属性を持つ利用者と“大学院生”の属性を持つ利用者が結託しても暗号文を復号できない．文献[5]の方式は，システムのマスターキーに閾値秘密分散[6]を適用し，そのシェアと属性を一対一対応させている．指定する属性に対応するシェアが含まれる公開パラメータで暗号化し，利用者の鍵が指定された属性を満たしていれば復号できる．

文献[7]では，復号条件に論理積を使用できるようになり，復号条件の自由度が高まった．すべての属性に対して「許可する」「許可しない」のどちらかを指定するため，秘密鍵と暗号文のサイズや暗号化／復号コストがシステム全体の属性数に依存する．秘密鍵や暗号文のサイズや計算コストを軽減するために，「Don't care（どちらでも良い）」を指定する方式が文献[8]で提案されている．この方式では，特定の属性を持つ利用者にだけ見せたいファイル（部外秘のファイルやプロジェクト関係者以外には見せたくないファイル）を暗号化するには，秘密鍵のサイズや，暗号化・復号の計算量がシステム全体の属性数に比例する．よってこれらは要件 2 を満たさない場合がある．

4. 単一の鍵で多重帰属できるグループファイル共有プロトコル

本章では，我々が提案している単一の鍵で多重帰属できるグループファイル共有プロトコル[3]について述べる．ここでは，メンバは常にオンライン状態であるとする．

4.1 ファイル復号鍵を暗号化／復号する鍵

我々の提案しているプロトコルでは，共通鍵暗号方式に基づいて，ファイルを“ファイル復号鍵”で暗号化し，ファイル復号鍵を“グループ公開鍵”で暗号化する．ファイル復号鍵の復号に必要な“グループ秘密鍵”は，グループ秘密鍵をルートとする二分木構造でグループ秘密鍵を分散管理する（図 4）．ファイル保管サーバが管理する分散情報を“サーバ部分復号鍵”，メンバが管理する鍵を“メンバ部分復号鍵”と呼ぶ．この二分木構造では，ある階層のメンバ部分復号鍵に(2,2)閾値秘密分散を適用した分散情報が，1 つ下の階層のメンバ部分復号鍵とサーバ部分復号鍵になる．ファイルの復号はファイル保管サーバとメンバの二者が協力し，(2,2)閾値復号に基づいてファイル復号鍵を復号する．

4.2 ファイル共有に関するプロトコルとグループ管理に関するプロトコル

我々の方式はファイル共有，グループ構築，メンバ追加，メンバ離脱，鍵更新の 5 つのプロトコルから構成される．

［ファイル共有プロトコル］

図 5 に示すようなファイル暗号化とファイル復号を行う．まず，ファイル暗号化の手順は次のようになる．

Step1. クライアントにおいてファイルごとに対するファイル復号鍵を生成し，ファイルを暗号化する．

Step2. ファイル復号鍵をグループ公開鍵で暗号化し，ファイルと共にファイル保管サーバに送信する．

Step3. ファイル保管サーバでは暗号化したファイルと暗号化したファイル復号鍵をセットで保管する．

ファイル復号の手順は次のようになる．

Step1. ファイル保管サーバは暗号化されたファイル復号鍵をサーバ部分復号

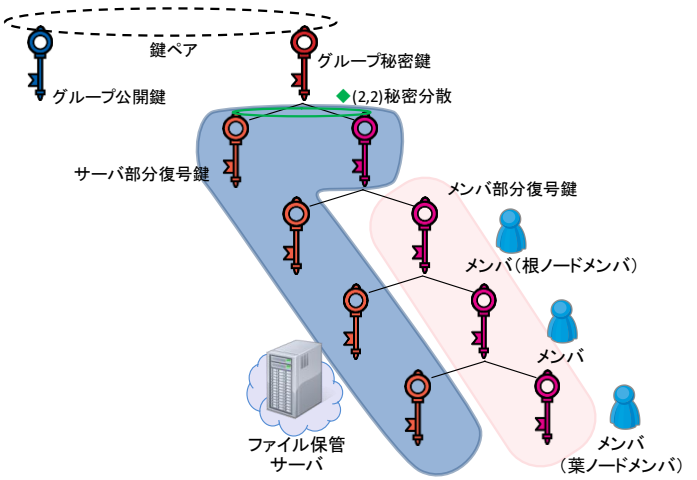

図 4　グループ秘密鍵をルートとする二分木構造

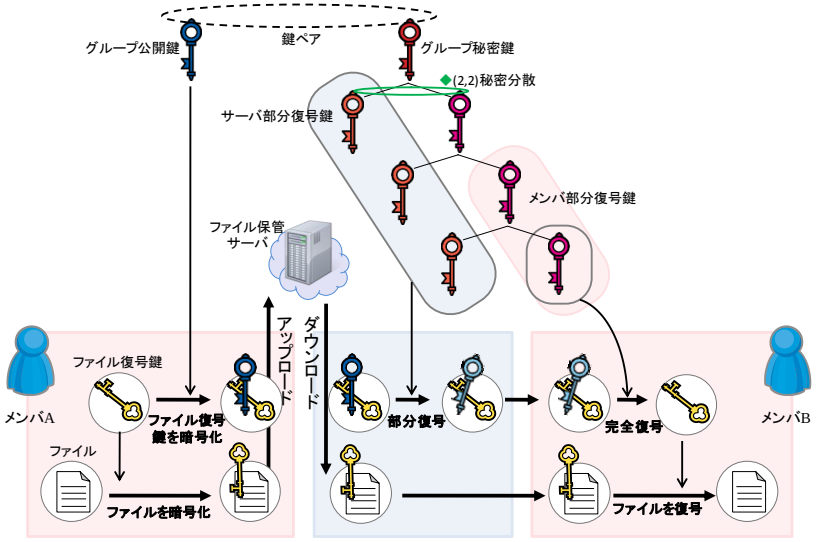

図 5　ファイル共有プロトコル

鍵で“部分復号”し，暗号化されたファイルと共にメンバに送信する．

Step2. メンバがメンバ部分復号鍵を用いてファイル復号鍵を“完全復号”する．

Step3. 復号されたファイル復号鍵でファイルを復号する．

［グループ構築プロトコル］

図 6 に示すグループ秘密鍵の二分木構造の構築と，グループメンバとファイル保管サーバにメンバ部分復号鍵とサーバ部分復号鍵の配布を行うことで，新しいグループを構築する．

Step1. ファイル保管サーバとメンバで分散情報生成プロトコル[9]を実行し，サーバ部分復号鍵とメンバ部分復号鍵を生成する．

Step2. ファイル保管サーバとメンバが情報を交換し，グループ秘密鍵に対応するグループ公開鍵を生成し，公開する．情報を交換する過程で，お互いの部分復号鍵が相手や第三者に知られることはない．

Step3. メンバ部分復号鍵に(2,2)閾値分散を適用し，一方をファイル保管サーバに，他方を次に追加するメンバに配布する．分散情報を受け取ったメンバは，メンバ部分復号鍵に単一化[3]する．

Step4. Step3.を全てのメンバにメンバ部分復号鍵を配布するまで繰り返す．

［メンバ追加プロトコル］

図 4 の二分木構造にノードを追加することで実現する．追加するグループの葉ノード（最後に追加されたメンバ）のメンバ部分復号鍵を(2,2)閾値秘密分散し分散情報を 2 つ生成する．一方をファイル保管サーバに，他方を追加するメンバに配布し，追加するメンバは自身のメンバ部分復号鍵に単一化することで，メンバ部分復号鍵を変更せずに追加されたグループでファイルを共有できるようにする．

［メンバ離脱プロトコル］

図 6 に示すようにグループ秘密鍵の二分木構造から離脱させることで実現する．離脱メンバの親ノードのメンバの部分復号鍵を(2,2)閾値秘密分散し，2 つの分散情報を得る．一方をファイル保管サーバに，他方を離脱メンバの子ノードのメンバに配布し，子ノードのメンバはメンバ部分復号鍵に単一化する．この手続き後は離脱したメンバのメンバ部分復号鍵では離脱したグループではファイル復号鍵を復号できないため，ファイルを閲覧できない．離脱メンバの子ノードメンバが単一化をすることにより，離脱メンバの子孫ノードとなるメンバはメンバ部分復号鍵を変更せずメンバ離脱を実現できるため，メンバ離脱時にグループ公開鍵・秘密鍵ペアの再生成，秘密鍵の再配布が必要ない．

［鍵更新プロトコル］

根ノードとなるメンバとファイル保管サーバはプロアクティブ秘密分散[10]によりそれぞれの部分復号鍵を更新し，更新に使った情報の一部を子ノードに伝播させながら，他のメンバの部分復号鍵と残りのサーバ部分復号鍵を更新する．

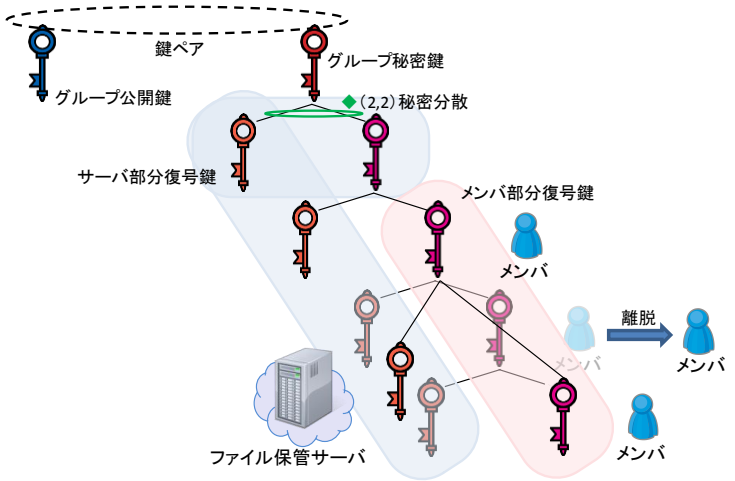

図 6 メンバ離脱

表 1 一般ユーザ，グループ管理者，管理者が利用できる機能

主体	機能
一般ユーザ	ログイン，ログアウト，所属するグループ一覧表示，グループで共有するファイル一覧の表示，ファイルのアップロード，ダウンロード
グループ管理者	一般ユーザが持つ機能，管理するグループへのメンバの追加と離脱，管理するグループの鍵更新の開始
管理者	一般ユーザが持つ機能，グループの作成と削除，グループ管理者権限の付与と剥奪，ユーザアカウントの作成，鍵更新の開始，各グループに対するグループメンバの追加と離脱

【提案方式が要件 1，要件 2 を満たすことの確認】

提案方式では，利用者がファイルを暗号化してファイル保管サーバに保管しているため，サービス提供者はファイルを閲覧できないため，要件 1 を満たす．

提案方式では複数グループに所属するメンバが 1 つの鍵で，所属するすべてのグループのファイルを復号できるため，利用者の鍵管理の負担は少ない．また，グループメンバの変更時にメンバの鍵を更新する必要がない．

我々は既に，これらのプロトコルが実利用可能なものであることをシミュレーショ

ンにより確認しており[4]，要件 2 を満たすといえる．以降では，提案プロトコルを用いたグループファイル共有システムの設計と実装について述べる．

5. システム設計

単一の鍵で多重帰属できるグループファイル共有プロトコルを利用したファイル共有システムを実装する．本システムは“一般ユーザ”，“グループ管理者”，“管理者”，“ファイル保管サーバ”の 4 つの主体で構成される．一般ユーザ，グループ管理者，管理者が利用できる機能を表 1 に示す．なお，ファイル保管サーバはサービス利用者ではないため，利用できる機能はない．

なお鍵更新はファイル保管サーバが定期的に実行するが，グループ管理者と管理者は手動で開始することもできるとする．このときグループ管理者が更新できるのは管理するグループのみである．

システムの構成を図 7 に示す．クライアントソフトウェア側の暗号処理部は暗号化・復号を行う部分である．これは必要な計算量が大きいためスクリプト言語ではなく，コンパイラ型言語で実装する．よって，サービス利用者はコンパイラ型言語で作られた実行ファイルをクライアントにインストールして使う形態とする．

5.1 動作手順

主な機能の動作手順を示す．

［ログイン］

ID とパスワードによりユーザ認証を行う機能である．

Step1. 画面管理部でユーザが ID とパスワードを入力し，ID とパスワードのハッシュ値を通信部経由でファイル保管サーバに送る．

Step2. ユーザ情報管理部でクライアントから送られたハッシュ値とファイル保管サーバで保存されているハッシュ値が合致することを検証する．検証が成功した場合，メンバ部分復号鍵をクライアントに送る．メンバ部分復号鍵はクライアントのパスワードで暗号化されているため，ファイル保管サーバはメンバ部分復号鍵を知ることはできない．

Step3. クライアントの暗号処理部でメンバ部分復号鍵を復号する．

Step4. ファイル保管サーバ側で，ユーザ情報管理部がユーザ情報を，グループ情報管理部がユーザの所属するグループ情報を，ファイル情報管理部がグループで共有しているファイル情報を DB から読み出し，クライアントへ送る．

［ファイル共有］

ファイル共有は一般ユーザがファイルをファイル保管サーバにアップロード／ダウンロードする機能である．まずアップロードの手順を示す．

Step1. クライアントがファイル情報管理部でファイルを読み込む．

Step2. 暗号処理部で共通鍵であるファイル復号鍵を生成し，ファイルを暗号化する．

Step3. 暗号処理部で共有するグループのグループ公開鍵を使いファイル復号鍵を暗号化する．

Step4. 暗号化されたファイルと暗号化されたファイル復号鍵を通信部経由でファイル保管サーバへ送る．

Step5. ファイル保管サーバはファイルに一意なファイル ID を発行し，DB にファイルとファイル復号鍵を保存する．

ダウンロードでは以下の手順を踏む．

Step1. クライアントはユーザの選択したファイルの ID をダウンロード要求と共にファイル保管サーバへ送る．

Step2. ファイル保管サーバはファイル情報管理部がファイル ID から暗号化されたファイル及び暗号化されたファイル復号鍵を読み込む．

Step3. 暗号処理部がサーバ部分復号鍵を用いてファイル復号鍵を部分復号する．

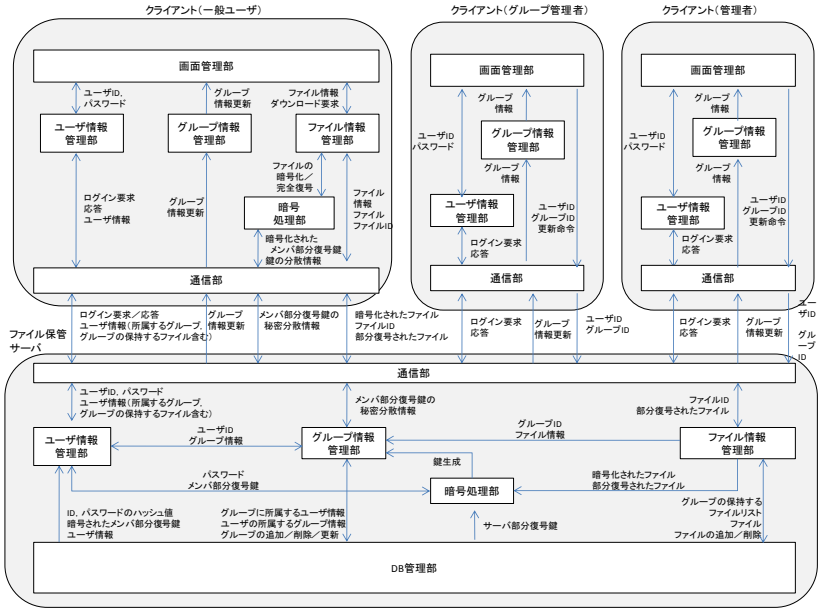

図 7　システム構成

Step4. 部分復号されたファイル復号鍵，暗号化されたファイルをクライアントに送る．

Step5. クライアントは部分復号されたファイル復号鍵をメンバ部分復号鍵で完全復号する．

Step6. Step5.で復号したファイル復号鍵でファイルを復号する．

［メンバ追加］

新たなメンバをグループへ追加する機能である．以下に手順を示す．

Step1. グループ管理者または管理者であるユーザが画面管理部で追加するユーザ，追加先のグループを選択する．

Step2. クライアントは選択されたグループとユーザのグループ ID とユーザ ID を追加要求と共に通信部経由でファイル保管サーバに送る．

Step3. ファイル保管サーバは葉ノードのクライアンにメンバ部分復号鍵分散要求を出す．

Step4. 葉ノードメンバはメンバ部分復号鍵を(2,2)閾値分散し，生成した 2 つの分散情報をファイル保管サーバに送る．

Step5. ファイル保管サーバは 2 つの分散情報を受け取り，一方を新しいサーバ部分復号鍵として DB に格納する．他方を追加するメンバに送る．

Step6. 追加されるメンバは，暗号処理部で受け取った分散情報を元から所持していたメンバ部分復号鍵に単一化する．

Step7. ファイル保管サーバはグループメンバに関する情報を更新する．

このプロトコルは，追加されるメンバがメンバ部分復号鍵を持っていることを前提としている．メンバ部分復号鍵はユーザアカウントを作成する際に生成されるためグループに所属していないメンバでもメンバ部分復号鍵を持っている．

［メンバ離脱］

メンバをグループから離脱させる．離脱したメンバは離脱したグループの部分復号されたファイル復号鍵を手に入れても復号できない．以下に手順を述べる．

Step1. グループ管理者または管理者であるユーザが画面管理部に離脱するメンバ，離脱させるグループを選択する．

Step2. クライアントはユーザ ID とグループ ID をファイル保管サーバへ送る．

Step3. ファイル保管サーバは，離脱メンバの親ノードのクライアントに対し通信部を通してメンバ部分復号鍵のメンバ部分復号鍵分散要求を送る．

Step4. 親ノードのクライアントは暗号処理部でメンバ部分復号鍵に(2,2)閾値秘密分散を行い，2 つの分散情報を得る．

Step5. 得られた 2 つの分散情報をファイル保管サーバに送る．

Step6. ファイル保管サーバは分散情報の一方をサーバ部分復号鍵として DB に格納し，他方を離脱メンバの子ノードに送る．

Step7. 子ノードのクライアントは暗号処理部で，与えられた分散情報を自身のメンバ部分復号鍵に単一化する．

Step8. ファイル保管サーバはグループメンバに関する情報を更新する．

5.2 DB 設計

本システムには“ファイル”，“グループ”，“ユーザ”の 3 種類のデータ群がある．ファイルにはファイル名・暗号化済みファイル復号鍵，グループにはグループ名・グループ公開鍵，ユーザにはユーザ名・パスワードのハッシュ値などの情報がある．これらの値をデータベースのファイルテーブル，グループテーブル，ユーザテーブルに保存し，主キーをファイル ID，ユーザ ID，グループ ID とする．

ファイルはグループに所属しているため，ファイルテーブルは外部キー（Foreign Key）としてグループ ID を属性に持つ．同様にアップロード者のユーザ ID を外部キーに持つ．ファイル本体に関しては，1 つのデータに制限がある DB や，ファイルシステムより検索／読み込み／書き込みの遅い DB の場合，ファイル本体はファイルシステム上に保管し，ファイルテーブルにはパスを保管する．しかし上記を満たさない DB の場合，高速な検索読み書きのためにファイル本体はデータベース内に格納する．

ユーザはどのグループに所属しているかの情報を保持する必要があり，表 2 のようなグループ集合テーブルを使って表現する．グループ集合テーブルは主キーがグループ集合 ID であり，主キー以外の属性はグループ ID である．グループ ID 属性の値は所属する／しないの 2 値である．ユーザはユーザテーブルでグループ集合 ID を持つ．例えば表 2 のグループ集合テーブルでは，グループ集合 ID が 1 のユーザはどのグループにも所属せず，グループ集合 ID が 2 のユーザはグループ ID が 3 のグループに所属する．

次に，グループ ID から所属するユーザ ID のリストを抽出する方法を述べる．グループに所属するメンバはグループ秘密鍵の二分木構造を維持したまま格納する必要がある．そこで，グループ名などが格納されたグループテーブルとは別に，各グループに 1 つ，グループ ID を名前に持つテーブルを作成する．主キーはユーザ ID であり，属性にサーバ部分復号鍵を持つ．サーバ部分復号鍵はユーザ ID と対応したものであり，行はメンバが追加された順となるため，鍵の木構造を保存することができる．

暗号化／復号に使う公開情報は共通情報テーブルに格納する．クライアント側の

表 2　グループ集合テーブルの例

グループ集合 ID	1	3
1	所属しない	所属しない
2	所属しない	所属する

DB ではクライアントアプリケーションの起動時間を短縮するため，頻繁に利用されるテーブルのキャッシュをデータベースに保存する．保持するテーブルは暗号化演算に用いる公開情報を収めた共通情報テーブルと，所属するグループの情報を収めたグループテーブル，所属するグループで共有するファイル，ログイン中のユーザ情報を収めたユーザテーブルである．これらから ER 図は図 8 のようになる．

6. 実装

ファイル保管サーバ，クライアントの画面管理部は Adobe Flex3.0[11]を使い，それ以外の部分は Java 言語を使い実装した．ファイル保管サーバ側の DB は MySQL 5.1[12] ，クライアント側は SQLite3.7.4[13]を使用した．ファイル保管サーバの OS は CentOS5.0 で，クライアントは Windows7 である．

6.1 画面設計と Flex でのユーザインターフェース

Flex とは Adobe Systems が開発したモバイル，Web，デスクトップアプリケーション構築のためのフレームワークである．デザインに MXML（Macromedia Flex Markup Language），プログラムには ActionScript を利用し，Flash Player 上で動作する swf ファイルを生成する．インターフェース実装の容易さやドラッグアンドドロップに代表されるユーザ親和性の高さから画面管理部に Adobe Flex3.0 を使用する．

Flex は Flex3.0 から AIR 上で実行可能となった．AIR とは Adobe Systems の開発した，デスクトップリッチインターネットアプリケーションを複数の OS 上で開発・実行できるランタイムライブラリである．本システムはクライアントでコンパイラ型言語により作られた実行ファイルを動作させる必要がある．そのため今回 Flex はクライアントで動作する AIR 上で実行させ，localhost に対するソケット通信を通してクライアント内の実行ファイルと連携する．このとき，Java 言語で作られた実行ファイルから swf ファイルを実行することはできないため，Flex プログラムは swf ファイルではなく exe 形式で出力する．

本システムの画面を図 9，図 10，図 11 に示す．図 9 は一般ユーザのメイン画面である．中央のタブウィンドウ①がユーザの所属するグループを表している．リスト②が所属するメンバ一覧であり，表③が共有するファイル一覧である．表③へ OS のファイルシステムからファイルをドラッグアンドドロップすることでファイルをアップロードすることができる．また，表③中のアクセスしたいファイルをクリックすることでファイルのダウンロードを行うことができる．タブ④を切り替えることで他のグループに表示を切り替えることができる．ボタン⑤からログアウトすることができ，ログイン画面へ遷移する．

ファイル管理者の場合，ログアウトボタン⑤の横にグループ管理者の管理画面への画面遷移ボタンが出現する．グループ管理者の管理画面を図 10 に示す．表①が全ユー

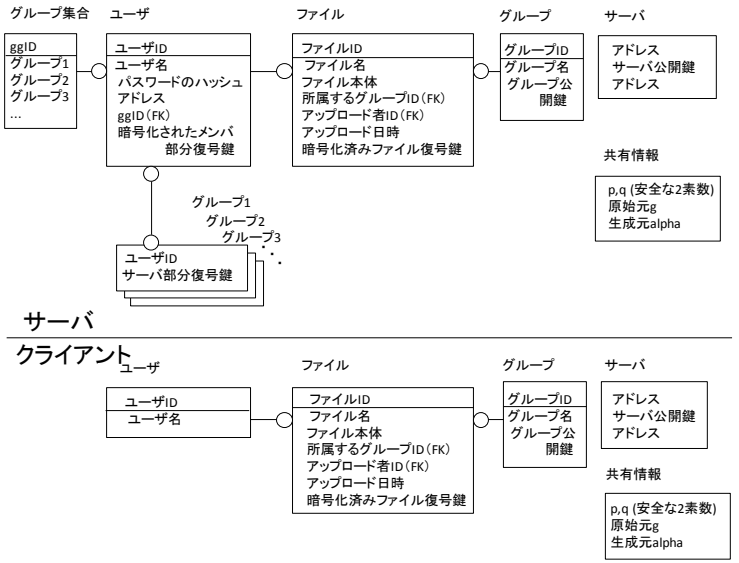

図 8　ER 図

図 9　一般ユーザのファイル共有画面

ザ一覧，表②が管理するグループの一覧，表③が選択中のグループに所属するメンバの一覧である．グループメンバ追加は，ボタン④もしくは表①から表③へのドラッグアンドドロップで行うことができる．表③で削除するユーザ選択した状態でボタン⑤を押し，確認ダイアログで「はい」を選択することでメンバをグループから離脱させることができる．メンバ更新は，ファイル共有画面またはグループ管理画面がリアルタイムに反映される．ボタン⑥で選択したグループの鍵更新，ボタン⑦でファイル一覧へ戻る，ボタン⑧でログアウトしログイン画面へ遷移する．

次に，管理者のメイン画面を図 11 に示す．図 10 のグループ管理者と共通する部分が多いため差分のみ述べる．表①のグループ一覧に全てのグループが表示される．ボタン②を押すとグループ名を入力するダイアログが表示され，入力すると新しいグループが作成され表①に追加されるボタン③を押すと表②で選択中のグループを消すことを確認するダイアログが出る．これで「はい」を選ぶとグループが消され，表①からもそのグループは削除される．グループ作成／削除はメンバの追加／削除と同様，グループに所属するメンバがログインしていた場合にリアルタイムで更新される．ボタン④を押すとユーザ情報入力ダイアログが現れ，ここでユーザ名などを入力することで，どこのグループにも未所属の新規ユーザアカウントが作成できる．表⑤のグループメンバ一覧では，メンバをダブルクリックすることでメンバをグループ管理者に任命することができる．任命されたメンバは表示される色が代わり，再びダブルクリックすることでグループ管理者権限を剥奪できる．最後に，管理者がボタン⑥を押すと全てのグループの鍵が更新される．

6.2 DB：MySQL と SQLite

MySQL はオープンソースのデータベース管理システムである．Java 言語から MySQL に接続するには JDBC[15]を利用する．本システムではファイル保管サーバでのデータ格納に利用している．

MySQL ではバイナリファイルを保存する Blob というデータ型があるが，これの上限は 900KB であるため，4.2 節で述べたとおり，本システムではファイル本体はファイル保管サーバ側のファイルシステム内に保存し，ファイルテーブル内にはファイル本体へのパスを格納する．

SQLite は軽量のデータベースであり，1 つのファイルのみで構成される扱いやすさからアプリケーションに組み込んで利用されるデータベース管理システムである．本システムでもクライアント側に組み込み，JDBC を通して Java 言語からアクセスする．パスワード設定ができず，全てのアプリケーションが利用できるため，平文のパスワードやメンバ部分復号鍵を保存してはならないことに留意する．

7. 評価

2.3 節で定義したグループファイル共有システムに必要な機能を実現した上で，グループファイル共有サービス提供組織はサービス利用組織のファイルを閲覧できないことが本プロトコルの目的である．プロトコルの目的に沿った設計，実装ができたか検証する．

図 10　グループ管理者の管理画面

図 11　管理者の管理画面

それぞれの主体が持つ情報を表 3 に示す．ファイル保管サーバは暗号化されたメンバ部分復号鍵を持つが，パスワードを知らないためメンバ部分復号鍵を復号できない．そのため，ファイル復号鍵を完全復号できず，ファイルを復号できない．よって要件 1 を満たす．

表 3　各主体が取得可能な情報

情報	サーバ	クライアント
パスワード	×	○
メンバ部分復号鍵	×	○
暗号化されたメンバ部分復号鍵	○	○
サーバ部分復号鍵	○	×
暗号化されたファイル復号鍵	○	○
部分復号されたファイル復号鍵	○	○
完全復号されたファイル復号鍵	×	○
暗号化されたファイル	○	○
復号されたファイル	×	○

要件 2 については次のとおりである.

A) 実際にシステムを利用し，ファイル共有，グループ構築・削除，メンバ追加・離脱，鍵更新はシステム利用に現実的な時間で処理を終えることを確認した.

B) グループメンバの追加・離脱では，グループ公開鍵，秘密鍵ペアの再生成，新しいグループ秘密鍵の再配布が不要であるため，一部のメンバがプロトコルに参加すれば実行可能である.

よって，計算やグループメンバの変更に必要なコストが低いため，要件 2 を満たす.
以上から,提案プロトコルを利用したグループファイル共有システムは SaaS 型グループファイル共有に求められる要件を満たすことが確認できた.

8. おわりに

本論文では単一の鍵で多重帰属できるグループファイル共有プロトコルをグループファイル共有システムへ実装した．システムの機能設計を行い，使用する鍵を管理するための DB を設計した．各主体の操作画面について説明し，グループに所属しない人はファイルにアクセスできないことを確認した.

今後の課題として，全てのメンバがオフライン状態でもメンバの追加，削除ができるプロトコルの開発や，各動作の時間計測やアンケートを使った評価などが挙げられる.

参考文献

1) 株式会社 ITR：クラウド時代のコラボレーション／ツールの方向性，〈http://www.itr.co.jp/PDFPUB/ITR_WP_C10090023.pdf〉（参照 2011-01-12）.
2) 国内クラウドサービス市場　2010 年の実績と 2011 年～2015 年の予測，IDC Japan，〈http://www.idcjapan.co.jp/Report/SaaS/j11290102.html〉（参照 2011-05-06）.
3) 内田真理子，福田洋治，毛利公美，白石善明：多重帰属の鍵管理が容易な(2,2)閾値秘密分散を用いたグループファイル共有，電子情報通信学会技術研究報告, vol. 108，no. 473，pp. 71-78，2009
4) 長澤悠貴，白石善明，毛利公美，福田洋治，野口亮司：単一の鍵で多重帰属できるグループファイル共有システムの評価，情報処理学会第 72 回全国大会講演論文集，第 3 分冊，pp.667-668，2010
5) A. Sahai and B. Waters. Fuzzy identity based encryption. In Advances in Cryptology – Eurocrypt, volume 3494 of LNCS, pp 457–473, 2005.
6) A. Shamir : How to share a secret, Communications of the ACM, vol.22,no.11, pp.612-613, Nov. 1979.
7) L. Cheung and C. Newport : Provably Secure Ciphertext Policy ABE,ACM Conference on Computer and Communications Security (ACM CSS'07), 2007
8) 國武大記，満保雅浩，岡本栄司：属性ベース暗号の一改良，電子情報通信学会技術研究報告，情報セキュリティ 110(44)，pp 33-39，2010
9) T. Pedersen : A threshold cryptosystem without a trusted party, Proc. of Eurocrypt '91, LNCS No.547, 1991
10) A. Herzberg, S. Jarecki, H. Krawczyk and M. Yung : Proactive SecretSharing or: How to Cope with Perpetual extended abstract, IBM T.J. Watson Research Center, 1995.
11) Adobe Flex,〈http://www.adobe.com/jp/products/flex/?promoid=BPBSD〉（参照 2011-05-10）
12) MySQL 4.1 リファレンスマニュアル，〈http://dev.mysql.com/doc/refman/4.1/ja/legal-names.html〉（参照 2011-05-10）
13) SQLite,〈http://www.sqlite.org/〉(参照 2011-05-20)
14) Adobe, Adobe Flash Builder 4.5,〈http://www.adobe.com/jp/products/flash-builder.html〉(参照 2011-05-20)
15) Java SE Technologies – Database，Oracle,〈http://www.oracle.com/technetwork/java/javase/jdbc/index.html〉（参照 2011-05-10）