暗号化ファイル共有システムの実装と基本的評価

5U－6

室田 真男　新保 淳　高橋 俊成

(株)東芝　研究開発センター

1　はじめに

著者らは、前回の全国大会において、"暗号化ファイル共有システム"の概要とそのセキュリティ機構について発表した[1][2]。本システムは、一つのファイルを複数の利用者がロックをかけることなく同時編集可能、暗号技術の導入により通信路およびファイルサーバの管理者に対するファイルの内容保護の実現、という特徴をもつ。

今回は、システムの基本部分の実装を行い、その概要と基本的評価を報告する。

2　特徴

本システムは、クライアント・サーバ型で実現されている。特徴を以下に示す。

高セキュリティ データ内容の暗号化とユーザ認証を行う。暗号化はクライアントが行い、サーバは復号することなくクライアントからのデータを保存する。暗号方式はDES(Data Encryption Standard)の8ビットCFBモードを用いる。認証は、電子署名を用いたユーザ認証およびコマンド単位のメッセージ認証を行う。

高速書き込み可能 クライアントからサーバへ送るのは編集差分データのみであるため高速書き込みが可能。

同時アクセス可能 クライアントは、ファイルにロックをかけることなく編集可能。クライアントからのデータは差分情報であるので、サーバは複数ユーザからのリクエストを一つのファイルにマージすることができる。

3　データ構造とプロトコル

共有ファイルのデータ構造を図1に示す。

ファイル内容部のデータは、ファイルごとに固有の鍵(ファイル鍵WK)で暗号化される。ファイル内容にアクセス可能なユーザをメンバと呼ぶ。メンバのみがファイル鍵にアクセス可能とするために、公開鍵暗号による鍵配送方式を用いる。

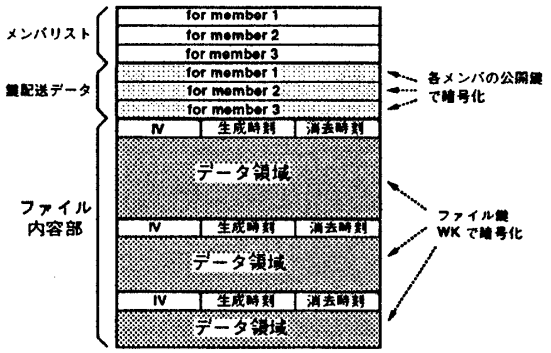

図 1: 共有ファイルの暗号化に関わるデータ構造

ファイル鍵を各メンバの公開鍵で暗号化しデータのヘッダ部に記録する(鍵配送データ)。

ファイル内容部は複数のブロックからなり、マージ処理と履歴管理に適したデータ構造をしている。

クライアント・サーバ間の通信は、SunRPC(Remote Procedure Call)、トランスポートプロトコルはTCP、ネットワーク上のデータは、XDR(eXternal Data Representation)を用いている。

実装した基本プロトコルを以下に示す。

CHECKIN クライアントがサーバに対し共有ファイルの参照開始を宣言する。サーバは、リクエストの電子署名を検査することによりユーザ認証を行う。ファイルアクセス許可情報をチェックした後、CHECKOUTするまでクライアントと共有する認証鍵を作成し、ファイルヘッダと共にクライアントに返す(認証鍵はクライアントの公開鍵で暗号化する)。クライアントは、認証鍵とファイル鍵を取得する。以降のリクエストコマンドには認証鍵を用いたメッセージ認証コードが付けられる。

READ ファイルを読み込み、データを復号化。

WRITE ファイルに書き込む。送られるデータは、挿入と削除で表される編集差分のみ。挿入は、挿入地点と(暗号化)データ、削除は、削除範囲のみを送る。

CHECKOUT ファイルの参照終了を宣言する。

Implementation and Fundamental Evaluation of Privacy Enhanced File Sharing System

Masao MUROTA, Atsushi SHIMBO, Toshinari TAKAHASHI
Toshiba Corporation, Communication and Information Systems Research Labs.

4 評価

本システムの実装を行った。サーバは Sun SPARC Station 20(SS20)、クライアントは Sun SPARC Station 2(SS2) 上に構築した。なお、DES は GNU ソフトウェアの Eric DES、公開鍵暗号方式は RSA 方式を用いた。

CHECKIN と READ に関する評価を示す。処理時間は、計算機の内部時計 (time() システムコール) により計測し、誤差を小さくするため、複数回の平均をとっている。

CHECKIN について、メンバー数を変化させた時の処理時間を図 2に示す。CHECKIN 処理の流れは、(1) リクエストへ電子署名を付ける (図 2では signature)、(2) リクエストを送ってレスポンスを受け取る (checkin)、(3) 認証鍵を復号 (authkey)、(4) ファイル鍵を復号 (filekey)、と表される。

(1)(3)(4) は RSA 暗号の復号処理であり、それぞれ約 1.2 秒かかっている。メンバー数が増えるとヘッダ情報が増えるので、(2) が若干増えているが、RSA の処理時間に比べると無視できるほどである。全体としては約 4 秒かかっており、公開鍵暗号方式のオーバヘッドをもう少し軽減したい。

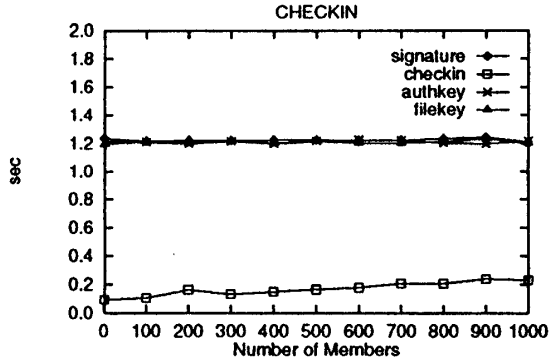

図 2: メンバー数による CHECKIN の処理時間

READ について、データサイズを変化させた時 (データブロック数は 1 つ) の処理時間、データサイズを 12Kbyte としてブロック数を変化させた時の処理時間を、それぞれ、図 3、4に示す。READ 処理の流れは、(1) メッセージ認証コード作成 (read_mac)、(2) リクエストを送ってレスポンスを受け取る (read)、(3) レスポンスの認証 (readres_mac)、(4) データの復号 (decrypt)、と表される。

図 3より、データ量にほぼ比例して復号時間がかかっていることがわかる。復号化速度は約 18.4(Kbyte/s) であった。他の処理時間は充分に小さい。データ量に応じて処理時間が増えるが、十分利用できる利用感が得られた。

図 4より、ブロック数にほぼ比例して (2)(3) の処理時間が増えていることがわかる。これは、データブロックのメモリーアロケーションの時間である。実際の利用上で、どの程度までブロック数が増えるかわからないが、ブロック数が増

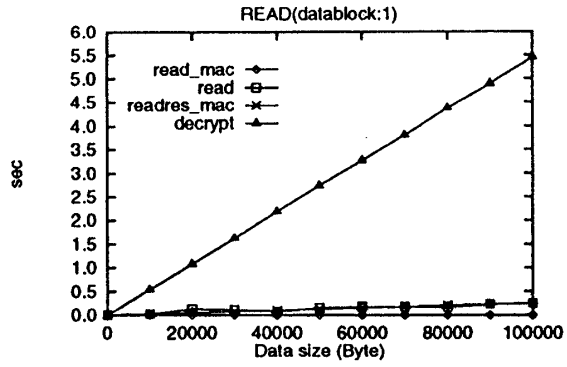

図 3: データサイズによる READ の処理時間

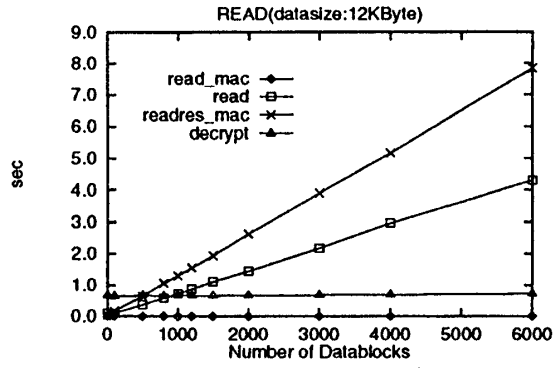

図 4: データブロック数による READ の処理時間

えると無視できないオーバヘッドである。なお、復号化の処理時間はブロック数には依存しない。

今回は割愛したが、WRITE は READ の特性と似た傾向を示し、CHECKOUT はほとんど時間がかからない。

5 おわりに

"暗号化ファイル共有システム" の基本部分の実装を行い、その評価を行った。公開鍵暗号方式部分やデータブロック数による処理時間の増加などが多少大きいが、まずまずの利用感が得られる。今後は実装を進め、実際に使いながらの評価を進めて行く予定である。

参考文献

[1] 高橋, 新保, 室田, "暗号化ファイル共有システムの概要", 情報処理学会第 49 回大会、1D-3、1994.

[2] 新保, 室田, 高橋, "暗号化ファイル共有システムのセキュリティ機構", 情報処理学会第 49 回大会、1D-2、1994.