ファイルシステム組み込み暗号ソフト

5Q－8

宮崎 博　　鮫島 吉喜
日立ソフトウェアエンジニアリング(株)

1. はじめに

携帯端末の普及により、機密データを記憶装置内に保存した端末を持ち運ぶことが多くなると予想される。そのため、端末の盗難、紛失などによるデータ漏洩を防ぐ仕組みが必要となる。また、一台の端末を複数ユーザで共用する場合、端末内に保存したデータを他ユーザからの閲覧や改ざんから守る必要もある。

本論文では、セキュリティ機能を持たない既存PC用OSのファイルシステムに以下の機能を組み込んだものについて報告する。

- アプリケーションに透過なファイル暗号処理
- ユーザ及びグループごとのアクセス制御

2. アプリケーション透過暗号処理

ファイルの暗号処理は、ユーザが操作するアプリケーションからのファイルI/O要求時に同時に行っている。記憶装置への書き込み要求時には保存データを暗号化してから書き込み、読み出し要求時には読み出した暗号化データを復号してアプリケーションに渡している。暗号化の指定はユーザがディレクトリごとに行い（以下、暗号ディレクトリと呼び、指定を行ったユーザは暗号ディレクトリの所有者と呼ぶ）、そのディレクトリ以下のすべてのファイルが暗号化される。

暗号アルゴリズムにはブロック暗号を採用したが、アプリケーションからのランダムアクセスを考慮して、図1のようにファイルを暗号化している。暗号化する際はファイルを一定長ごとのレコードに区切り、それぞれのレコードごとに初期化ベクタを用意してCBCモードで暗号化している。レコードの末端のブロックまで暗号化したら、そこで暗号化を終えている。これにより、ファイル内の任意の位置への書き込みがあっても、毎回ファイルの終端まで再暗号化を行う必要がなくなりオーバヘッドが少なくなる。

また、ファイル終端の暗号化の際にはパディングをせずにCFBモードで1バイトずつ暗号化し、ファイルサイズが暗号前と変わらないようにしている。

各レコードごとに使用する初期化ベクタの生成は、ファイルごとに異なる初期化ベクタの種とレコードインデックスを入力とする計算によって行う。暗号鍵はファイル内のすべてのレコードに共通なものを使っている。

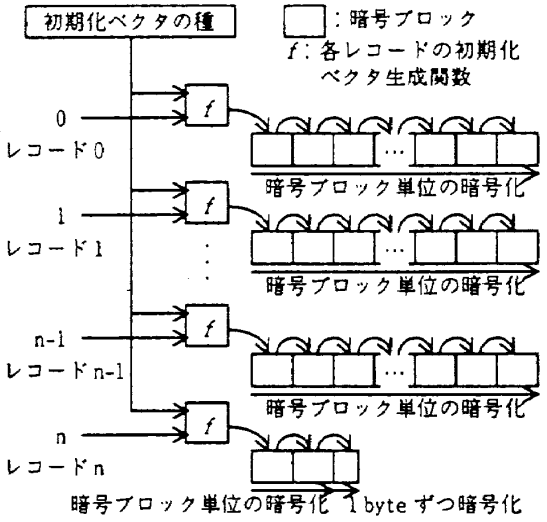

図1 ファイルの暗号方式

3. アクセス制御

本論文でターゲットとしたOSのファイルシステムではファイルの属性として読み取り専用の指定が可能であるが、誰でもその属性を変更できてしまう。そこで、表1に示すアクセス権を定義し、暗号ディレクトリの所有者だけがそのディレクトリへのアクセス権を設定できるようにした。また、他のユーザやグループに対するアクセス権も設定可能とした。

Cryptographic Software Plugged into File System
Hiroshi Miyazaki, Yoshiki Sameshima
Hitachi Software Engineering Co., Ltd.

なお、PC操作者のユーザ名やその所属グループ名を判別するため、OS起動後にユーザ名とパスワードをユーザに入力させてユーザ認証を行っている。

暗号ディレクトリのアクセス権変更は、その所有者であるユーザが認証された時だけ行うことができる。他ユーザからアクセス権を与えられたユーザは、そのアクセス権を変更できない。

表 1 アクセス権の種類

種類	ユーザ認証前	ユーザ認証後
Read	暗号化ファイルの読込み可(復号なし)	暗号化ファイルの読込み可(復号あり)
Write	暗号化ファイルへの書込み不可、暗号化ファイル削除不可	暗号化ファイルへの書込み可(暗号化あり)、暗号化ファイル削除可

4.暗号鍵と設定情報の管理方法

ファイルの暗号鍵はファイルごとに異なるものを使用する。暗号化ファイルの名前とそのファイルの暗号鍵、初期化ベクタの組は、暗号ディレクトリごとに用意する鍵ファイルに保存する。

鍵ファイルに保存するファイル暗号鍵は、鍵ファイルごとに用意するディレクトリ鍵で暗号化する。ディレクトリ鍵は、暗号ディレクトリの所有者やディレクトリへのアクセス権を与えられたユーザやグループの暗号鍵で暗号化して鍵ファイル中に保存する。これにより、ディレクトリへのアクセス権を持つユーザやグループが増えても、鍵ファイルのサイズはそれらの暗号鍵で暗号化されたディレクトリ鍵の分しか増えずにすむ。ユーザやグループに与えられるアクセス権もディレクトリ鍵と一緒に暗号化しておく。図2に鍵ファイル内に保存する各鍵の関係を示す。

上記のディレクトリ鍵やアクセス権、ファイル鍵などの情報は、暗号化ファイルがオープンされた時にそのファイル名を使って鍵ファイルから検索されて読み出される。ここで、ファイルアクセスを行ったユーザやその所属するグループに対応するディレクトリ鍵の情報がなかったり、要求したファイルI/Oに対応するアクセス権が割り当てられていなければ、そのファイルI/Oは失敗する。

なお、ユーザやグループの暗号鍵は、別に用意するファイル内に暗号化して保存している。

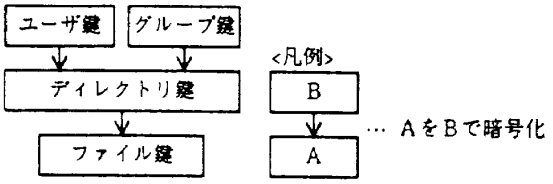

図 2 鍵ファイル内に保存する各鍵の関係

5.性能評価

暗号アルゴリズムに鍵長 128 ビットの SAFER [1]、初期化ベクタ生成関数に MD5[1]を使い、暗号レコードサイズを512バイトとして速度評価を行った。速度評価の際はOSのキャッシュ機能を無効にし、1メガバイト長のファイルのランダム位置からランダム長のデータを読み書きした。

暗号処理・アクセス制御組み込み時の入出力速度は、ファイルのレコード化によるオーバヘッド削減により、非組み込み時に比べて暗号処理時間の分、遅くなっただけである。アプリケーションプログラムでファイルを編集している際の入出力長は短いので、実用上、暗号処理によるオーバヘッドは問題のない程度である。

6.まとめ

セキュリティ機能のないPC用OSのファイルシステムに、ファイルの暗号化とアクセス制御機能を組み込んだ。ユーザが指定したディレクトリ下のファイルの暗号処理をそのファイルへのI/O発生時に行うので、ユーザがファイルの暗号化・復号化をそのたびに指示せずにすむ。また、アクセス権をユーザやグループごとに指定できるため、不正ユーザによる暗号化ファイルの閲覧だけでなく、改ざんや削除を防ぐことができる。

謝辞

本研究を進めるに当たり、適切な助言を頂いた横浜国立大学大学院 工学研究科 松本勉 助教授に感謝致します。

参考文献

[1] Bruce Schneier : Applied Cryptography : John Wiley & Sons (1996)