マルチメディア通信と分散処理　85－32
（1997．11．7）

セキュアデジタルコンテンツ VOD システム

山口智久　峯村治実　大野次彦
三菱電機株式会社　情報技術総合研究所

セキュアデジタルコンテンツ配布システムは、著作権保護や課金機能を実現して、デジタルコンテンツのセキュアな配布サービスを行うことを目的としたシステムである。この目的を達成する手段として、暗号化されたコンテンツの配信や利用履歴による管理等があり、我々はインターネット・イントラネット上でセキュアな動画配信システムを実現することを目的とし、動画の不正利用（盗聴・不正コピー）を防止し、クライアントでの利用履歴を収集・管理できる動画配信（VOD）システムの検討を行っている。今回はこの中で、主に企業内 LAN でのビデオサーバークライアントを対象として、暗号化動画コンテンツの配布方式の検討を行い、検討した結果に基づき、試作システムを構築し、評価を行った。

Secure Digital Content VOD System

Tomohisa YAMAGUCHI　Harumi MINEMURA　Tsugihiko OHNO
Information Technology R & D Center, Mitsubishi Electric Corporation

Secure digital content delivery systems are to realize copyright protection and charging mechanism, and it aims at secure delivery service of digital contents. Cryptic content delivery and history management are means to accomplish this purpose. We are investigating a Video-On-Demand system that can prevent illegal usage of video data and manage user history data to achieve a secure video delivery system on the Internet or Intranet. Mainly targeting client-server systems in enterprise LANs, we have implemented and evaluated a prototype system based on the investigation into the delivery method of cryptic contents.

1. はじめに

セキュアデジタルコンテンツ配布システムは、コンテンツ製作者、コンテンツ配布者、コンテンツ利用者間でのセキュアなコンテンツ配布およびコンテンツ利用を実現するシステムである。今回はこの中で、コンテンツ利用者での処理部分である、主に企業内LANでのビデオサーバークライアントを対象として、暗号化動画コンテンツの配布方式の検討を行った。

2. 暗号化動画配信の課題と解決方法の検討

暗号化動画配信における課題としては以下のものがあげられる。

1 クライアントでのリアルタイム暗号解除
2 コンテンツおよび暗号鍵の管理
3 部分暗号化

これらの課題と解決方法について検討した結果を以下に示す。

2．1．クライアントでのリアルタイム暗号解除

サーバから配信された暗号化動画データの暗号解除をクライアントで行う方法として、いったん中間ファイルを生成する方法とリアルタイムに暗号解除を行う方法がある。中間ファイルを生成する方式では、全データをダウンロードするまでの時間がかかる、中間ファイルがディスク上に存在することになり、容易にコピーできてしまうという問題がある。一方リアルタイム暗号解除方式ではサーバからの暗号化データをリアルタイムに暗号解除し、これをビューアで直接再生するため、ダウンロードによる遅延はなく、また中間ファイルからコピーされることもない。図1にリアルタイム暗号解除方式を示す。通常 Windows95 のような OS で MPEG 等の動画ファイルを再生する場合、ビューアからのデータ要求は MPEG デコーダ、ファイルシステムマネージャを経由し、ファイルシステムドライバに渡され、ファイルシステムドライバはこれ

に従いデータをネットワークから取り出し、データの要求元に返すようになっている。

リアルタイム暗号解除方式では、リアルタイム暗号解除部で、ファイルシステムマネージャからのデータ要求を受け取り、各パラメータの修正を行い、これをファイルシステムドライバへのデータ要求として送る。ファイルシステムドライバはこのデータ要求に従い、データをリアルタイム暗号解除部に渡す。リアルタイム暗号解除部では、このデータの暗号解除を行うことにより、リアルタイム暗号解除を行う。

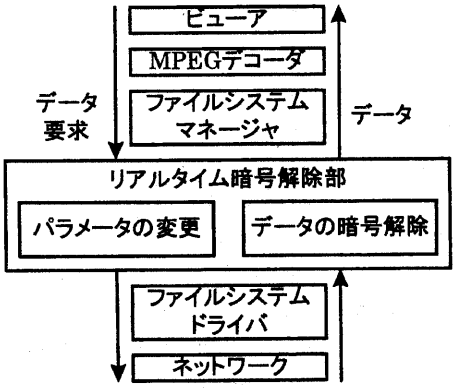


図1 リアルタイム暗号解除

2. 2. コンテンツおよび暗号鍵の管理

動画などのマルチメディアコンテンツを管理する場合、ファイル名だけで管理したのでは、マルチメディアデータ特有の属性を管理することができないため、コンテンツにタイトル、キーワード、暗号化フラグ等の属性を付加しデータベース化することによって管理を行うことにした。一方暗号鍵の管理では暗号鍵をコンテンツの一属性として持たせることにした。図2に今回検討した暗号鍵のサーバ－クライアント間でのキー交換方式を示す。本方式ではコンテンツおよび暗号鍵の管理をビデオサーバとは別のコントロールサーバによって行う。図2に示すように、まずクライアントは、キー交換を行うためにユーザ認証を行う必要があり、コントロールサーバにログインする。次にクライアントは公開鍵／秘密鍵を生成し、公開鍵をコントロールサーバに送る。コントロールサーバではユーザ認証に成功している場合に限り、コンテンツを暗号化した共通鍵を送られてきた公開鍵で暗号化し、クライアントに暗号化共通鍵を送る。クライアントは送られてきた暗号化共通鍵を、先ほど生成した秘密鍵で暗号解除する。このようにキー交換を行うことにより、コンテンツを暗号化した暗号鍵（共通鍵）が盗聴されることを防ぐことができる。

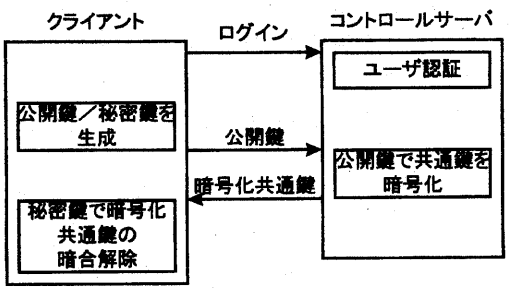


図2 キー交換

2. 3. 部分暗号化

ユーザに動画の内容を確認してもらい、購買意欲／視聴意欲を掻き立てる方法の1つとして、暗号化された動画の最初の数秒間のみを再生する方法の検討を行った。まず図3に示すようにファイルの先頭数秒間は暗号化せず、残りの部分を暗号化する部分暗号化方式の検討を行った。

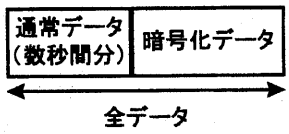


図3 部分暗号化方式1

この方式ではデータを2重持ちしなくてよいという利点があるが、デコーダの処理によっては動作がおかしくなるので、この方式には問題があり、図4に示す方式の検討を行った。

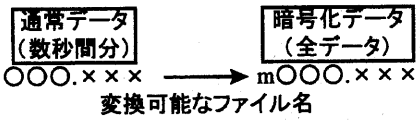


図4 部分暗号化方式2

図4のような方式を使用することによって先頭の数秒間を見ることは可能である。データを2重持ちする必要はあるが、コンテンツの管理も通常データのファイル名が分かれば、暗号化データのファイル名も分かるので管理は複雑化しない(ただし通常データと暗号化データは同一または決められたディレクトリに格納しなければならない)。

3. 暗号化動画配信実現方式の詳細

3. 1. システム構成

図5に本システムのシステム構成を示す。

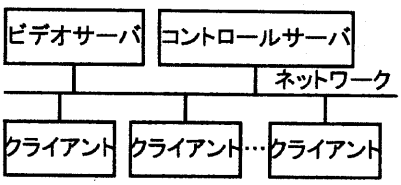


図5 暗号化動画配信システムのシステム構成

以下に各構成要素の説明を行う。

・ビデオサーバ

ビデオサーバにはコンテンツの実ファイルが格納されており、動画の配信を行う。

・コントロールサーバ

コントロールサーバは各種管理情報を保持し、システム全体の配信、運用管理、キー管理を行う。

・クライアント

クライアントはユーザインターフェースとコンテンツの再生を受けもつ。

3. 2. クライアントのモジュール構成

図6にクライアントのモジュール構成を示す。

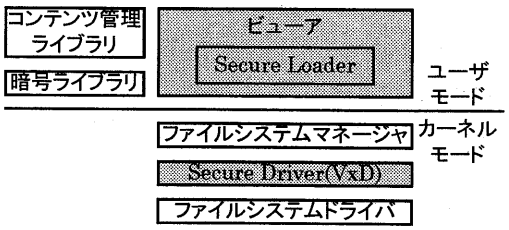


図6 クライアントのモジュール構成

今回、図6の網掛け部分の検討および作成を行った。以下にビューア、Secure Loader、Secure Driverの説明を行う。

・ビューア

ビューアは動画の表示やユーザとのインターフェースを受け持つ、32bitのWindowsアプリケーションである。

・Secure Loader

Secure Loaderはビューア内の関数であり、Secure Driverや暗号ライブラリの呼び出しを行う。

・Secure Driver

Secure Driverはファイル I/Oをフックし、暗号解除をSecure Loaderに依頼するVxD(仮想デバイスドライバ)である。

図7に各モジュール間の制御の流れを示す。

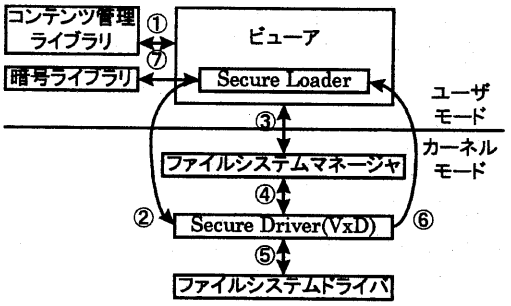


図7 モジュール間の制御の流れ

① ログイン処理およびコンテンツ情報の獲得
② Secure Driverのロードと初期化
③ リード要求
④ フック処理
⑤ リード要求
⑥ 暗号解除要求
⑦ 暗号解除処理

3. 3. 動画配信手順

図8にコンテンツ配信時の各構成要素間の制御の流れを示す。

① クライアントーコントロールサーバ間でログイン処理を行う。
② クライアントがコントロールサーバからコンテンツ情報を持ってくる。
③ クライアントーコントロールサーバ間でキー交換を行う。

④ ビデオサーバからクライアントに動画データが送られる。

⑤ クライアントでこのデータをリアルタイムに暗号解除して動画の再生を行う。

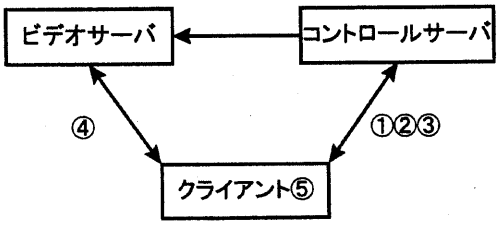


図8 コンテンツ配信時の制御の流れ

4. 試作システムによる評価

4. 1. 性能評価

以下の2種類の測定を行った。

・暗号化サイズ変更による CPU 使用率の測定

・VxD によるフックおよび暗号解除のオーバーヘッドの測定

4. 1. 1. 暗号化サイズ変更による CPU 使用率の測定

本測定は暗号化サイズとアプリケーションからのリードサイズの関係より、最適なバッファサイズを求めることを目的としている。

バッファリングの動作を簡単に説明する。ここでは説明を単純にするために暗号データの開始位置のオフセットとリード要求の開始位置のオフセットが同じ場合を考える。

バッファリングを行う場合には以下の3つの場合が考えられる。

・暗号化サイズ ＜ リードサイズ

・暗号化サイズ ＝ リードサイズ

・暗号化サイズ ＜ リードサイズ

○暗号化サイズ ＜ リードサイズの場合

図9にこの様子を示す。

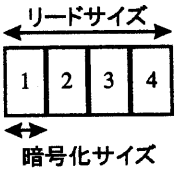


図9 暗号化サイズ ＜ リードサイズの場合

図9の例の場合、動作は以下のようになる。

1 アプリケーションからのリード要求があったら、リードサイズ分のデータをバッファにリードする。

2 バッファ内のデータを4回に分けて暗号解除を行う。

3 バッファ内のデータをリード要求元に渡す。

以上を繰り返す。

このようにバッファのサイズは小さくてもよいが、ディスクからのリードと暗号解除ルーチン呼び出しの回数が多くなる。

○暗号化サイズ ＝ リードサイズの場合

図10にこの様子を示す。

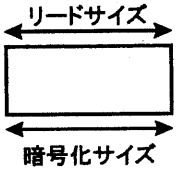


図10 暗号化サイズ ＝ リードサイズの場合

動作は以下のようになる。

1 リードサイズ分のデータをバッファにリードする。

2 バッファ内の全データを1回で暗号解除する。

3 バッファ内のデータをリード要求元に渡す。

○暗号化サイズ ＞ リードサイズの場合

図11にこの様子を示す。

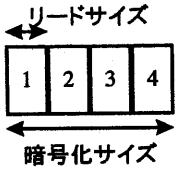


図11 暗号化サイズ ＞ リードサイズの場合

動作は以下のようになる。

1 データ1のリード要求が来たら暗号化サイズ分のデータをバッファにリードする。

2 バッファ内データの暗号解除を行う。

3 バッファ内のデータ1のデータをリード要求元に返す。

4 データ2のリード要求が来たらバッファ内のデ

ータ2のデータをリード要求元に返す。
以上を繰り返す。
このようにディスクからのリードと暗号解除ルーチン呼び出しの回数が少なくなるが、バッファのサイズは大きくなる。
図12に暗号化サイズを変更した場合の CPU 使用率を示す。
測定環境は以下の通りである。なお測定に用いた ActiveMovie(S/W MPEG1 デコーダ)のリード要求サイズは 8Kbyte である。

測定環境

CPU: PentiumPro 200MHz
メモリ: 64Mbyte
ビデオカード: Diamond SpeedStar 64
デスクトップ領域: 1024×768
カラーパレット: 256 色
OS: Windows95

コンテンツ(MPEG1 ファイル)の情報

解像度:352×240
ビットレート:1.15Mbps
フレームレート:29.97fps

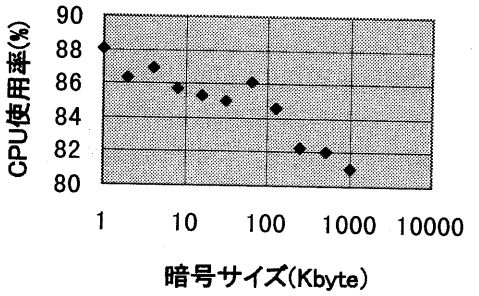


図12 暗号サイズに対する CPU 使用率の変化

図12から分かるように、多少の誤差はあるものの暗号サイズは大きい方が CPU 使用率は小さくなることが分かる。しかしデータに現れてこないが、暗号サイズが 512Kbyte 以上になると映像が一瞬フリーズするようになる(おそらく1回あたりの暗号解除に時間がかかるため)。
以上をまとめると
・暗号サイズは大きい方が CPU 使用率は小さい
・暗号サイズが大きいとバッファのサイズも大きくなる
・暗号サイズが 512Kbyte 以上になると暗号解除の時間が映像品質に影響を及ぼす。

よって今回使用した環境では暗号サイズは ActiveMovie からのリード要求サイズである 8Kbyte より大きく(バッファが効くように)、また映像の悪影響がおよぶ512Kbyteよりも小さいバッファサイズである 16～256Kbyte 程度が適当と思われる。以下、本システムでは 64Kbyte にして各種測定評価を行った。

4.1.2. VxD によるフックおよび暗号解除のオーバーヘッドの測定

ActiveMovie による MPEG1 ファイルの再生と、ActiveMovie＋Secure Driver による非暗号化 MPEG1 ファイルおよび暗号化 MPEG1 ファイル再生時の CPU 使用率をシステムモニタを使用して測定した。測定環境は PentiumPro-200(PC-1)と MMXPentium-200(PC-2)の PC を使用した。結果を以下に示す。

測定環境

表1 測定環境

	PC-1	PC-2
CPU	PentiumPro 200MHz	MMX Pentium 200MHz
メモリ	64 Mbyte	
デスクトップ領域	1280×1024	
カラーパレット	256色	
OS	Windows95	

MPEG1 ファイルの情報

解像度:352×240
ビットレート:1.15Mbps
フレームレート:29.97fps

表2 CPU 使用率

	PC-1	PC-2
VxDフックにかかるCPU使用率(%)	0.3	0.8
メモリ間コピーおよび暗号解除処理にかかるCPU使用率(%)	5.6	5.3
合計	5.9	6.1

表2よりVxDフックにかかるCPU使用率は、ほとんど無視しても構わない程度であった(0.3%/0.8%)。またメモリ間コピーと暗号解除処理によるCPU使用率は5.6%/5.3%でActiveMovieのみによる通常のMPEG1ファイル再生時に比べると5.9%/6.1%高い値になった。測定ではデスクトップのカラーパレットが256色で測定したが、HighColor(65536色)で再生するとCPU使用率が上がり、PC-2で約5%程度、PC-1では10%以上上昇しSecure Driver使用時には100%に達してしまった。このことからPC-2(MMX Pentium)上で1ストリームを再生するなら問題はないと言えるがPC-1(PentiumPro)では多少問題がある。ただしこの問題の原因はMPEGのデコードによるものであって、VxDによるフックと暗号解除にかかるCPU使用率は問題ない範囲である。

4.2. 試作システムによる各機能の検証

図13に示す試作システムで以下の各機能の確認を行った。

○再生機能

暗号化動画を見る権利を持つユーザの場合

(a) 暗号化動画配信→正常に再生された。

(b) 通常動画配信→正常に再生された。

暗号化動画を見る権利を持たないユーザの場合

(a) 暗号化動画配信→先頭部分のみ再生された。

(b) 通常動画配信→正常に再生された。

○登録機能

暗号化動画登録の場合

(a) 暗号化ファイル(全データ)→正常に登録された。

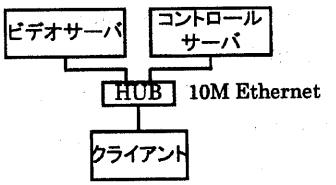
(b) 通常ファイル(一部分データ)→正常に登録された。

(c) 暗号鍵→正常に登録された。

通常動画登録の場合

(a) 通常ファイル(全データ)→正常に登録された。

これにより今回検討・試作した各種機能が正常に動作していることが確認できた。



	ビデオサーバ/コントロールサーバ	クライアント
CPU	Pentium100MHz ×4	Pentium 133MHz
メモリ	128/64 Mbyte	32 Mbyte
OS	WindowsNT4.0 Server	Windows95

図13 試作システムのH/W構成

5. おわりに

今回セキュアデジタルコンテンツ配布システムの中で、コンテンツ利用者部分である、主に企業内LANにおけるビデオサーバークライアントを対象として動画コンテンツ配信方式の機能検討を行い、検討した結果に基づき、試作システムを構築し評価を行った。本システムではVOD機能を実現するために、フックを用いたリアルタイム暗号解除を用いた。また、本システムでは不正利用防止の方式として、コンテンツの暗号化、ユーザ認証、動的な暗号解除、キー交換などを用いた。試作システムの評価では、暗号化サイズ変更によるCPU使用率の変化、フックおよび暗号解除によるオーバーヘッドの測定を行い、暗号化サイズでは最適なサイズを決定し、フックおよび暗号解除のオーバーヘッドはほぼ無視できる程度であることが確認できた。

参考文献

[1] 松井:ブロック暗号アルゴリズムMISTY:信学技報 ISEC-11 1996

[2] 安井 隆一他:ECの技術動向:デジタルコンテンツ作成流通技術、Journal of IPSJ Vol.38 No.9 Sep. 1997 P785-791