2004－ＱＡＩ－12　（４）
2004／7／12

# オンラインストレージ上のデータ保護機能のプロキシ方式による実現

見上 英里† 来住 伸子† 砂原 秀樹‡ 小川 貴英†

†津田塾大学大学院 理学研究科 〒187-8577 小平市津田町 2-1-1
‡奈良先端科学技術大学院大学 情報科学研究科 〒630-0192 生駒市高山町 8916-5

E-mail: †{m02mikam, kishi, ogawa}@tsuda.ac.jp, ‡suna@wide.ad.jp

**あらまし** 近年、ネットワーク上のどこからでもデータにアクセスできる、オンラインストレージと呼ばれるサービスが一般に普及しはじめている。しかし、これらのサービスではサーバが信頼できることを前提としており、通信路上のデータを暗号化して保護することは可能であるが、サーバ上にあるデータ自体を保護することは考慮していない。本研究では、WebDAV を用いたオンラインストレージにプロキシ方式でデータ暗号化機能を追加するシステムを試作した。システムを導入することで、既存のサーバとクライアントを今までどおりに使用しながら、データを送受信時に暗号化・復号化できることを確認した。また、システム導入によるクライアントの応答時間への影響も評価した。

**キーワード** オンラインストレージ、データ保護、WebDAV、暗号化

# The prototype of data encrypting proxy for online-storage

Eri MIKAMI† Nobuko KISHI† Hideki SUNAHARA‡ Takahide OGAWA†

† Graduate School of Science, Tsuda College 2-1-1 Tsudamachi Kodaira-shi 187-8577, Japan
‡ Graduate School of Information Science, Nara Institute of Science and Technology 8916-5 Takayamacho Ikoma-shi 630-0192, Japan

E-mail: †{m02mikam, kishi, ogawa}@tsuda.ac.jp, ‡suna@wide.ad.jp

**Abstract** Recently, online-storage services which allow access to users' data on the Internet become popular services. However, these services do not protect data against intrusion and attack. In order to protect data on the server, data should be encrypted by user side. In this paper, we implemented a prototype of data encrypting systems as a proxy for WebDAV services. This prototype can be used with existing WebDAV clients and WebDAV servers. We also evaluate the execution time of file transfer operation.

**Keyword** Online-storage, Data protection, WebDAV, Encryption

## 1. 研究の背景と目的

複数の異なるコンピュータからネットワークを通じてデータを共有する方法として、オンラインストレージが普及しはじめている。現状のオンラインストレージサービスでは、データは送信時の形式のままで保存されるため、どんなにセキュリティを確保しても、サーバに対する不正侵入などによるデータ漏洩の危険性がある。そこで、本研究ではオンラインストレージサーバ上に置かれるデータを暗号化するシステムを提案し、試作した。

現在、インターネット上の個人向けサービスでは、ユーザがサービス提供者を信頼して個人情報を預け、サービス提供者がそれらの個人情報を利用してユーザにサービスを行う形態が多い。そのためサーバ内部でのデータの保護についてはあまり問題にされてこなかった。

サーバが不正侵入されたときなどへの対策として、データをオンラインストレージに保存する時にはユーザ側でデータを暗号化するという対策を行う方法がある。現状でも、ユーザ自身が送信データをあらかじめ手動で暗号化する方法もあるが、毎回の暗号化・復号化に手間がかかる。

本研究では，手軽なオンラインストレージとして広く普及している WebDAV を用いたオンラインストレージシステムを対象とし，既存の WebDAV サーバ、クライアントを利用して暗号化したデータを保存するシステムを構築することを目指した。多数存在する既存のサーバ、クライアントをできるだけそのまま使用することを考え、WebDAV 上でのデータ転送時に暗号化・復号化を行うプロキシ方式のシステムを構築し、その評価を行った。

## 2. システムの設計方針

本研究では、一般ユーザが簡単に導入できる、データの自動暗号化システムを提案する。

本システムでは、クライアントが動作するマシン上で動くプロキシ方式を採用した。このことにより、ユーザが使用していた既存のクライアントを、本システム導入後もそのまま使用することができ、オンラインストレージサーバやネットワークの構成に手を加える必要がない。

なお、本システムはオンラインストレージ上に保存されるデータの安全性を高めることを目的としているため、クライアントが動作しているマシン内部の間の通信は安全であると仮定している。

### 2.1. システムの要件

本システムの要件とその実現方法をまとめると以下の通りである。

- 許可されたユーザ以外はデータを読むことができない
  - サーバに置くデータはすべて暗号化する
  - 暗号化には公開鍵暗号方式を使用する
- ユーザが簡単に導入・使用できる
  - クライアントマシン上で動作する
  - 既存のクライアントをそのまま使用する
  - 暗号化はバックグラウンドで行う
- サーバ管理者、ネットワーク管理者が何がしかの作業を行う必要はない
  - サーバは既存のオンラインストレージをそのまま使用する

### 2.2. システムの構成と動作

本システムは、クライアントマシン上で動作するWebDAVプロキシであり、WebDAVクライアントからはローカルホスト上のWebDAVサーバとして動作する。図1に本システムの構成を示す。

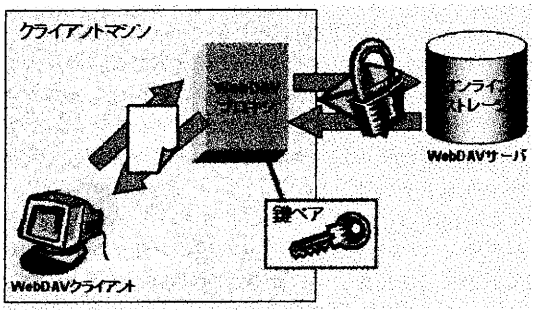


**図 1　システムの構成**

ユーザは、あらかじめWebDAVクライアントからプロキシをWebDAVサーバとして登録しておく。

WebDAVプロキシはクライアントからはローカルホスト上のWebDAVサーバとして扱われるため、Hostヘッダが実際と異なる。これを修正するために、すべてのリクエストとレスポンスのHostヘッダを変更する。

プロキシは、クライアントから「localhostのWebDAVサーバ」に対して送られたリクエストを受信すると、それを必要に応じて改変し、サーバに送信する。同様にサーバからのレスポンスも必要に応じて改変を行った上でクライアントに送信する。

本研究での目的を達成するため、WebDAVプロトコルのリクエストメソッドのうち、サーバにデータを保存するメソッドのPUT、サーバからデータを取得するGETの2つに対しての処理を行った。以下で、それぞれの場合のプロキシでのデータの具体的な改変の内容を述べる。

- **暗号化 - PUT**

サーバにデータを保存する際にプロキシでデータ本体の暗号化を行う。図2にデータ保存時の本システムの動作を示す。

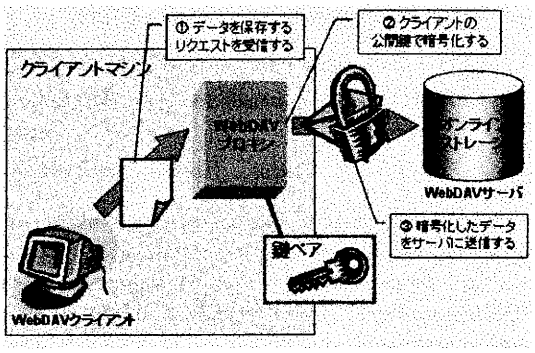


**図 2　データ保存時の動作（暗号化）**

プロキシは、クライアントからデータを保存するメソッドであるPUTを受信したとき、データのボディ部をユーザの公開鍵で暗号化したものに差し替える。暗号化後のデータに合わせてContent-TypeとContent-Lengthを変更してサーバに送信する。

- **復号化 - GET**

サーバからデータを取得する際に、プロキシで暗号化されたデータを復号化する。図3にデータ取得時の本システムの動作を示す。

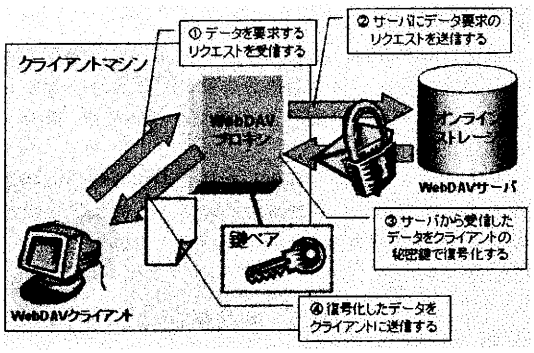


**図 3 データ取得時の動作（復号化）**

プロキシは、クライアントからデータ取得のメソッドである GET のリクエストを受け取ったならば、そのリクエストと対になるレスポンスのボディ部をユーザの秘密鍵で復号化したものと差し替え、PUT の場合と同様、Content-Type と Content-Length を変更してクライアントに送信する。

### 2.3. 鍵の管理

本システムでは公開鍵暗号方式を使用する。今回の試作では暗号方式として PGP を採用した。鍵は以下のように取り扱う。

- 使用する鍵ペア

本システムで鍵ペアを使用するのはシステム内部のみであり、通常の公開鍵ペアとは用途が異なるため、現在すでに鍵ペアを所有している場合でもユーザごとに別途専用の鍵ペアを生成する必要がある。一人のユーザが 2 台目以降のマシンに本システムを導入する場合には、最初のマシンで生成した鍵ペアを指定することで異なる場所からのデータの共有が可能になる。

- 鍵の管理方法

鍵は USB メモリなどの容易に持ち運び可能なメディアに保存するのが好ましい。アクセス手段と鍵を分けることにより、複数のユーザで共有しているマシンからでも暗号化したデータを参照することができる。

## 3. システムの実装

今回、WebDAV プロキシの実装は Perl5 で行った。暗号化部分では PGP を使い、プログラム中から PGP コマンドを呼び出している。

本システムが動作するためには、Perl と PGP が動作する環境と、コマンドラインモードで使用できる PGP のインストールが必要である。

実験に使用する WebDAV サーバには Apache 2.0.47[6][7]を, WebDAV クライアントには Windows XP の Web フォルダというシェアが高い組み合わせを選定した。また、Perl と PGP が動作する環境として Cygwin を、使用する PGP として PGP 6.5.8ckt 日本語版 r3[8]を使用した。

本システムは各マシンへの導入時にシステムで使用する鍵ペアとオンラインストレージのサーバとポート番号を指定する必要がある。現状では、直接プログラム内部に書き込みを行うこととなるが、将来的には簡単に設定できるような GUI を作成する予定である。

本システムはバックグラウンドで動作するため、本システムを起動しておけば、実際の使用時の操作は今までと変わらない。

## 4. 評価

本システムの動作性能を評価するための指標のひとつとして、応答時間の計測を行った。

### 4.1. 評価環境

本システムを導入した場合と導入しない場合（以下、直接接続と呼ぶ。）の応答時間をそれぞれ計測し、比較を行った。応答時間の計測には JScript で記述した WSH[10]のスクリプトを使用した。

暗号化・復号化には、2048bit の長さの鍵を用いて計測した。この長さの鍵を使用したのは、今回使用した PGP でデフォルトとして指定されているためである。

評価環境を図 4 に示す。オンラインストレージのサーバはインターネット上にあるのが普通であるが、今回は LAN 内での実験とした。サーバとプロキシは 10Base-T の HUB を介して接続した。

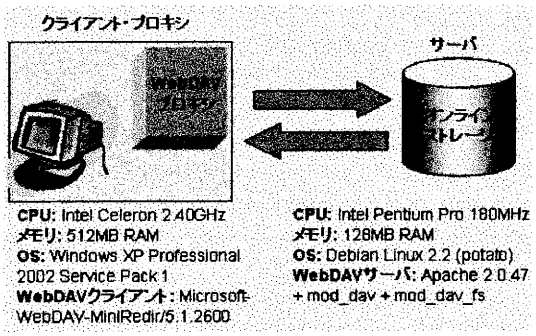


**図 4 評価環境**

### 4.2. 使用したデータ

データは暗号化によるデータサイズの変化を抑えるため、ランダムなバイナリファイルを使用し、クライアント、サーバのキャッシュの利用を避けるために毎回異なるものを用いた。

今回暗号化のために使用する PGP では暗号化処理の前にデータの圧縮を行うため、データサイズが小さくなることにより応答時間が変わる可能性がある。今回はサイズの変化を避けるため、ランダムなデータを

用いた。表 1 に示すとおり、データの圧縮は起きていない。

**表 1　データサイズ変化**

| 元データサイズ (KB) | 圧縮後サイズ/元サイズ(%) |
|---|---|
| 16 | 103.8 |
| 32 | 102.0 |
| 64 | 101.1 |
| 128 | 100.6 |
| 256 | 100.4 |
| 512 | 100.3 |
| 1024 | 100.3 |
| 2048 | 100.2 |

### 4.3. 測定方法

応答時間はクライアントからサーバへ、あるいはサーバからクライアントへのデータのコピーが行われた場合で計測する。

実際に計測に用いたスクリプトの一部を以下に示す。

```
……

/* 実行前の時間を測る */
oldtime = (new Date()).getTime();

/* ファイルのコピー */
fs.CopyFile(name, to, true);

/* 実行後の時間を測る */
newtime=(new Date()).getTime();

/* 差分を計算する */
dt = newtime - oldtime;

……
```

ここで name はコピー元ファイルのフルパス付きの名前、to はコピー先フォルダのフルパスであり、dt が応答時間となる。応答時間の中には、実際に PUT、GET が行われる前後に発行される一連の処理にかかる時間も含まれる。

クライアントの応答時間は、各データサイズで 10 回計測を行って、最大値と最小値を捨てた 8 回の平均値を算出した。最大値と最小値を平均の対象からはずしたのは、ネットワークの状態によるノイズを押さえるためである。

### 4.4. PGP の影響

参考値として、コマンドラインでの PGP の暗号化・復号化の実行速度も計測した。計測には time コマンドを使用し、本システム内部で呼び出されるコマンドと同じ引数と鍵ペアを用いた。

PGP の実行時間は各データサイズで 10 回の計測を行い、10 回の平均値を算出した。

### 4.5. 評価結果

表 2 に計測を行った結果を示す。現在の暗号化プロキシには暗号化・復号化部分の設計に不具合があり、2MB 以上のデータを扱うことができないため、計測しなかった。

表 3 は本システムを導入した場合の応答時間の増加率を調べるために、直接接続の時間と、直接接続の時間に PGP の実行時間を加えたものと、それぞれ比較した結果である。

**表 2　計測結果**

| 種類 | データサイズ (KB) | 応答・実行時間（ミリ秒） | | |
|---|---|---|---|---|
| | | ① 暗号化プロキシ | ②（うち PGP 実行） | ③直接接続 |
| PUT | 16 | 362 | 203 | 63 |
| | 32 | 384 | 199 | 78 |
| | 64 | 466 | 214 | 117 |
| | 128 | 619 | 261 | 190 |
| | 256 | 984 | 290 | 389 |
| | 512 | 1634 | 387 | 710 |
| | 1,024 | 3108 | 638 | 1168 |
| | 2,048 | - | 912 | 3244 |
| GET | 16 | 299 | 211 | 61 |
| | 32 | 310 | 211 | 66 |
| | 64 | 402 | 223 | 80 |
| | 128 | 635 | 251 | 211 |
| | 256 | 849 | 267 | 504 |
| | 512 | 1244 | 302 | 795 |
| | 1,024 | 2713 | 438 | 1504 |
| | 2,048 | - | 524 | 4513 |

**表 3　応答時間増加率**

| 種類 | データサイズ (KB) | プロキシ使用/直接接続 ①/③ | プロキシ使用/(直接接続＋PGP) ①/(②+③) |
|---|---|---|---|
| PUT | 16 | 5.75 | 1.36 |
| | 32 | 4.92 | 1.39 |
| | 64 | 3.98 | 1.41 |
| | 128 | 3.26 | 1.37 |
| | 256 | 2.53 | 1.45 |
| | 512 | 2.30 | 1.49 |
| | 1,024 | 2.66 | 1.72 |
| | 2,048 | - | - |
| GET | 16 | 4.90 | 1.10 |
| | 32 | 4.70 | 1.12 |
| | 64 | 5.03 | 1.33 |
| | 128 | 3.01 | 1.37 |
| | 256 | 1.68 | 1.10 |
| | 512 | 1.56 | 1.13 |
| | 1,024 | 1.80 | 1.40 |
| | 2,048 | - | - |

図 5、図 6 はそれぞれ表 2 の PUT の応答時間、GET の応答時間をグラフにしたものである。

グラフからもわかるとおり、今回の試作システムで

実験に用いた 16KB から 1024KB のデータでは、直接接続の時間に対してシステムを導入した場合の応答時間の比は、徐々に小さくなっており、データサイズが 1024KB のときで暗号化時に約 2.7 倍、復号化時に約 1.8 倍になる。

小さいデータでは、16KB のときに暗号化時に約 5.8 倍、復号化時に約 4.9 倍の時間がかかっているが、これは PGP 使用による増加分であり、このことを考慮して直接接続と PGP 実行の時間を足したものと比較した場合、それぞれ約 1.4 倍、約 1.1 倍に抑えられている。

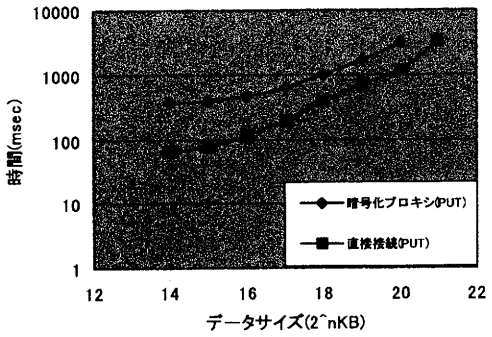


**図 5　データ保存時(PUT)の応答時間**

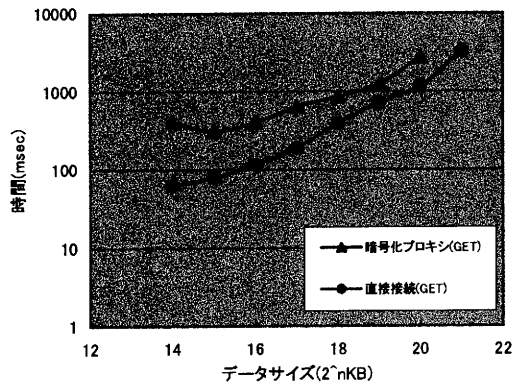


**図 6　データ取得時(GET)の応答時間**

## 5. 結論

本研究では、オンラインストレージサーバにデータを暗号化して保存するシステムを提案・試作し、応答時間の計測を行った。

オンラインストレージサービスでは、サーバのデータを利用するのはデータを保存したユーザまたはユーザグループのみであり、サービス提供者や第三者がデータの中身を参照する必要がない。この点に着眼し、ユーザの操作性を損なうことなく、データの機密保護を行うシステムを作成した。

プロキシとして実現したため、応答時間の増加はあるが、PGP で暗号化を行ったあと直接接続した場合と比較すると、平均して増加時間は約 1.5 倍程度であった。手動で暗号化を行う時間を考慮すると、このシステムを使用したほうが、暗号化したデータをより早くサーバに送信することができると考えている。

今後の課題としては、応答時間の短縮と導入の簡便化、操作性の向上が挙げられる。これらを実現することで、データの保護に対する関心の薄いユーザにも使用されることが期待できる。

応答時間の短縮方法としては、まず、現在暗号化・復号化をデータ全体を取得した後に行っているのを、受信した分からブロック単位で処理するように変更することが挙げられる。また、現在プロキシはサーバからのレスポンスをクライアントに伝えるだけであるが、サーバからのレスポンスを待たずに、プロキシから擬似的なレスポンスをクライアントに返却することも考えられる。

導入の簡便化方法としては、現在システムに必要な Cygwin などの環境と PGP などソフトウェアの、事前の導入を可能な限り不要とすることが挙げられる。

操作性の向上として、本システムの導入、使用する鍵ペアとオンラインストレージの指定などの操作を GUI で行えるようにすることが挙げられる。

## 文　献


[1] J. Slein, F. Vitali, E. Whitehead, D. Durand, Requirements for a Distributed Authoring and Versioning Protocol for the World Wide Web, http://www.ietf.org/rfc/rfc2291.txt, February 1998

[2] Y. Goland, E. Whitehead, A. Faizi, S. Carter, D. Jensen, HTTP Extensions for Distributed Authoring – WEBDAV, http://www.ietf.org/rfc/rfc2518.txt, February 1999

[3] T. Berners-Lee, R. Fielding, H. Frystyk, Hypertext Transfer Protocol -- HTTP/1.0, http://www.ietf.org/rfc/rfc1945.txt, May 1996

[4] J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, Hypertext Transfer Protocol -- HTTP/1.1, R. Fielding, http://www.ietf.org/rfc/rfc2068.txt, January 1997

[5] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L.

Masinter, P. Leach, T. Berners-Lee, Hypertext Transfer Protocol -- HTTP/1.1, http://www.ietf.org/rfc/rfc2616.txt, June 1999

[6] JAPAN APACHE USERS GROUP, http://www.apache.jp/

[7] The Apache Software Foundation, http://www.apache.org/

[8] Hizuya Atsuzaki, PGP 6.5.8 ckt for Windows Japanese Version, http://www.hizlab.net/pgp/

[9] Jason M. Hinkle, Crypt::PGPSimple, http://search.cpan.org/~jhinkle/Crypt-PGPSimple-0.13/PGPSimple.pm

[10] MSDN ライブラリ, http://www.microsoft.com/japan/msdn/library/default.asp