マルチメディア通信と分散処理　97－30
コンピュータセキュリティ　8－30
（２０００．３．２２）

ホームページ改竄パトロール方式

可部 孝二†　曽我正和‡　西垣正勝†　田窪昭夫††

† 静岡大学情報学部情報科学科 〒 432-8011 浜松市城北 3 - 5 -1
‡ 岩手県立大学ソフトウェア情報学部 〒 020-0173 岩手県滝沢村滝沢字巣子 152 - 52
†† 三菱電機情報システム情報システム製作所 〒 247-8520 鎌倉市上町屋 325
E-mail: nisigaki@cs.inf.shizuoka.ac.jp

あらまし

ホームページの改竄の有無を情報発信者 (WWW サーバ) 側が定期的にチェックする「ホームページ改竄パトロール」方式を提案する。サーバが改竄の有無をチェックすることにより、ホームページの改竄を発見した際の対応を迅速に、かつ、柔軟にとることができる。本稿では、パトロール方式を実装するに当たっての仕様を確定し、プロトタイプの実装までを行った。

キーワード：ホームページ改竄、改竄パトロール、ディジタル署名、ハッシュ関数

Home Page Patrol

Koji KABE†　Masakazu SOGA‡　Masakatsu NISHIGAKI†　Akio TAKUBO††

†Faculty of Information, Shizuoka University,
3-5-1 Johoku,Hamamatsu,432-8011,Japan,
‡Faculty of Software and Information Science,Iwate Prefectural University,
Sugo 152-52,Takizawa,Iwate,020-0173,Japan,
††Mitsubishi Electric Corp.,325 Kamimachiya,Kamakura,247-8520,Japan
E-mail: nisigaki@cs.inf.shizuoka.ac.jp

Abstract

This paper proposes a "home page patrol system". In this system, the WWW server patrols all the home pages within itself, and checks their integrity. By watching reports from the server, the home page authors are able to immediately respond and make the necessary corrections if any tampering or fraud has occurred.

Keywords: Home page tampering, patrol, digital signature, one-way hash function

1　はじめに

近年、インターネットの普及にともない、オンライン販売や個人情報のやりとりが活発に行われるようになった。このようなディジタル社会においては、「情報」が持つ価値は非常に高く、それ故にその信憑性が問われることになる。しかし、現在の計算機ネットワーク環境におけるセキュリティは完璧とは言えず、ホームページの改竄による被害は増加・深刻化している。

ホームページの改竄を完全に防止することは難しい。しかし、ディジタル署名 [1] や電子透かし [2] によりホームページの改竄を検知することは可能である [3]。ここで、従来の研究では、閲覧者側が改竄の有無の検出を行うというスタンスがとられていた。これはホームページを改竄されて困るのは閲覧者で

あるとの観点から派生したものであり、確かにこの方法には、閲覧者が容易にホームページの信頼性を判定することができるという利点がある。

しかし実際には、ホームページが改竄された場合に、ホームページ作成者が最大の被害者となるケースも数多く存在する。例えば、ホームページ上に無断で誹謗中傷する記事を上書きされて名誉を毀損されるケースや、商品の金額情報などが改竄されて商売を妨害されるケースがこれにあたる。また、閲覧者側に改竄チェックを任せた場合には、ホームページ作成者側に改竄の事実が通知されず、作成者は自身のホームページの改竄を知らぬままに放置されるという問題も発生し得る。

以上を考慮し、本稿ではWWWサーバが自己の管理している全ホームページデータの改竄チェックを行なう「改竄パトロール」方式を提案する。サーバ側でチェックを行うことにより、改竄が検知された際に早急な対応が可能となる。また、本方式は軽微の変更により閲覧者側のPCにおいても動作するアルゴリズムとなっている。閲覧者側のPCに本システムをインストールすることにより、作成者 - 閲覧者でのダブルチェックが可能なホームページ改竄チェック方式へと発展させることも容易である。

本稿では仕様を確定し、プロトタイプの実装までを行う。

2 ホームページ改竄パトロール方式

本稿で提案するパトロール方式においては、WWWサーバが定期的に自己の管理している全ホームページデータの改竄チェックを行う。一般にWWWサーバは複数のホームページ作成者のホームページデータを管理するが、各作成者のホームページが各々、全てチェックされる。

本方式の基盤はディジタル署名[1]と一方向性ハッシュ関数[4]にある。図1にパトロール方式の手順を示し、次節よりその説明を行う。なお、ここで次の前提を置く。

(1) ホームページ作成者とサーバは同一のハッシュ関数を持つ。
(2) サーバは、各ホームページ作成者ごとに公開鍵と秘密鍵のペアを作成する。
(3) サーバが作成した秘密鍵は安全にホームページ作成者に配送される。
(4) 公開鍵は、サーバがその特権領域(スーパーユーザでのみアクセス可能な領域)に保持する。
(5) サーバ内の特権領域にクラッカーが侵入することは難しいとする。

前提(1)は、ハッシュ関数は、各サーバとそのサーバにホームページを登録している作成者の間でのみ共有化されていれば良いということを意味する。また、前提(2)は、各サーバが独自に、そのサーバにホームページを登録している作成者の公開鍵と秘密鍵のペアを作成することが可能であることを示す。したがって、本方式は分散型のインターネット環境に適した構造となっている。なお、4章には、WWWサーバ内の特権領域にクラッカーの侵入を許した場合に対する対処法についても示す。

2.1 ホームページのアップ

ホームページ作成者がホームページデータをサーバにアップロードするまでの流れを説明する。図1の左側がこれを表す。

1. ホームページ作成者はローカルマシンにおいてホームページデータを作成する。
2. ホームページ作成者は、全ホームページデータのハッシュ値を計算し、「ハッシュインデックスファイル」を作成する。
3. ホームページ作成者は、ハッシュインデックスファイルのハッシュ値を更に計算し、「エッセンシャルハッシュ」を作成する。
4. ホームページ作成者は、エッセンシャルハッシュをホームページ作成者の秘密鍵で暗号化することにより、ディジタル署名する。
5. ホームページ作成者は、全ホームページデータと署名データをサーバにアップロードする。
6. サーバは作成者ごとに最新アップロード日時情報を管理する。

図2(b)に「ハッシュインデックスファイル」の書式を示す。ハッシュインデックスファイルは各ホームページデータのハッシュ値とファイル名および作成日時を一覧表にしたものである。ファイル名にはホームページデータ用ルートディレクトリを基点としたパスの情報も含む。各ホームページ作成者の全てのホームページデータが一つのハッシュインデックスファイルにまとめられる。なお、図2(b)は図

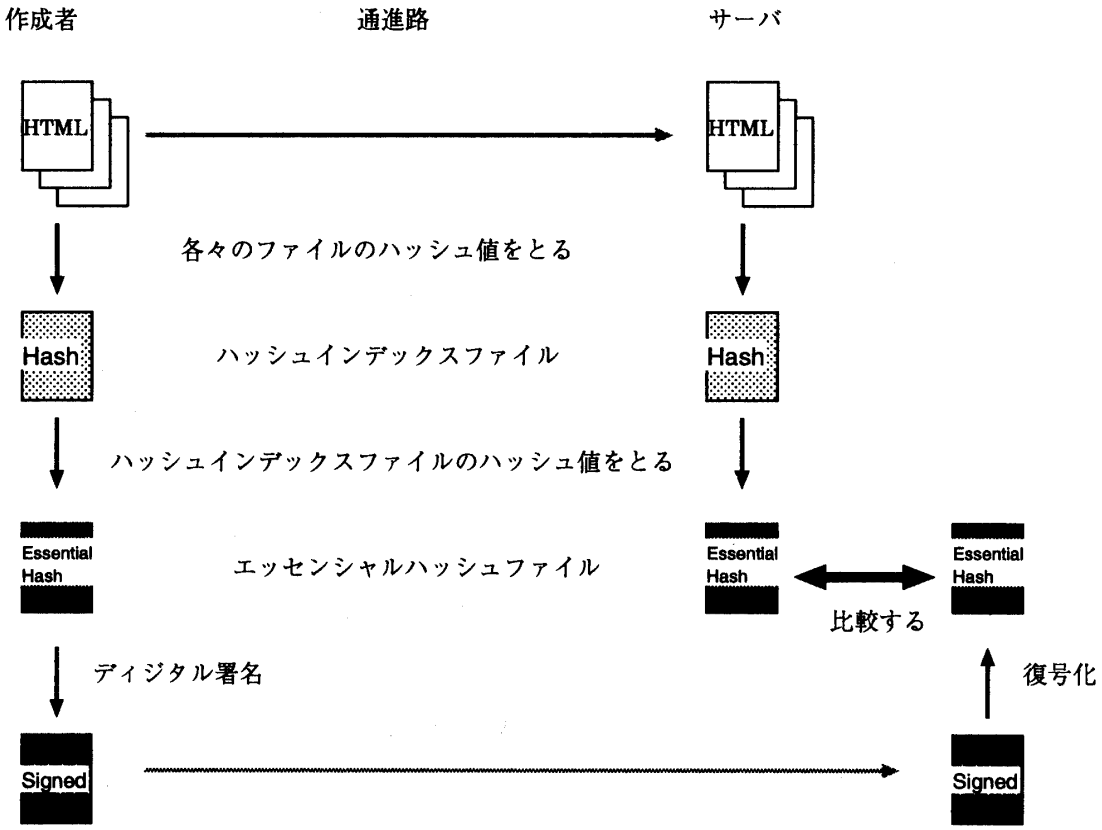

図 1: パトロール方式の手順

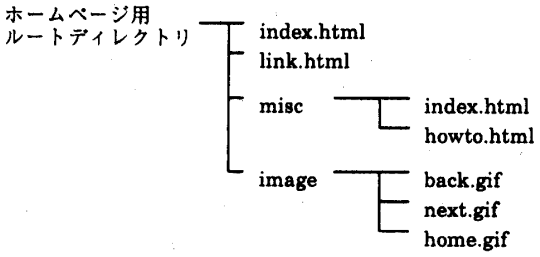

図 2(a): ホームページデータの木構造

ハッシュ値	ファイル名	作成日時
7283a355c8bafafe6cdb79bc89d80419	index.html	2000/2/19/19:32
195af16550f42de2d7f436a6e2b3de69	link.html	2000/2/10/10:25
371a6865442036fdb742b83d3cf39bf3	misc/index.html	2000/1/30/11:30
0ebb691bf8e4c0815b3f7f8e1d23dee7	misc/howto.html	2000/2/9/23:49
a26f98b43e211c14b8a781bf605e35a0	image/back.gif	2000/2/2/01:03
e855afb0ad377f6bfecc4013082012c2	image/next.gif	2000/2/2/01:06
edb7bf0087ecd6416d9dbe2e21113d74	image/home.gif	2000/1/30/01:09

図 2(b): インデックスハッシュの書式

2(a) の構成のホームページに対してハッシュインデックスファイルを作成した例である。

図 3 に「エッセンシャルハッシュ」の書式を示す。エッセンシャルハッシュはハッシュインデックスファイルを一つのホームページデータとみなし、そのハッシュ値を計算したものである。更に、ファイル名、エッセンシャルハッシュ作成日時、作成者のユーザ名が併記される。ホームページを如何なる構成で作成したとしても、そこから得られるハッシュインデックスファイルは一つである。したがって、エッセンシャルハッシュは必ず一行のデータとなる。なお、図 3 はハッシュインデックスファイルが index.hash という名前でホームページ作成者用のユーザホームディレクトリに置かれていた場合のエッセンシャルハッシュを示した例である。

エッセンシャルハッシュをディジタル署名する際に必要となるホームページ作成者用の秘密鍵は、作成者がサーバにホームページデータの掲載を契約し

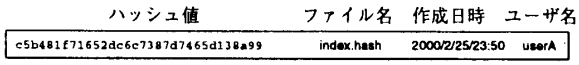

図 3: エッセンシャルハッシュの書式

た際にサーバから送られている。この秘密鍵はパスワード等により (共通鍵暗号方式により) 更に暗号化された上でローカルマシンに保管される。本稿においては、ローカルマシンは常時ネットワークに接続されないため、ローカルマシンがネットワーク経由で攻撃され、秘密鍵が盗まれる危険性は低いとする。したがって、第三者が署名データを偽造することはできない。なお将来的には、秘密鍵はホームページ作成者がICカード等に保管することにより、そのセキュリティをより高めることができる。

全ホームページデータおよび署名データは、サーバ上の通常の領域 (ホームページ作成者用のユーザディレクトリ) に格納される。一方、アップロード日時データはサーバ内の特権領域に保管される。

2.2 改竄パトロール

サーバが行うパトロールの流れを示す。図 1 の右側がこれを表す。サーバは自己が管理する全てのホームページ作成者ごとに、下記手順によるホームページデータの改竄チェックを行う。パトロールは定期的に行われる。

1. サーバは、作成者の署名データを公開鍵で復号し、作成者から送られたエッセンシャルハッシュ値とエッセンシャルハッシュ作成日時データを得る。
2. サーバは、サーバにアップロードされている作成者の全ホームページデータから、ハッシュインデックスファイルを実際に計算する。
3. サーバは、ハッシュインデックスファイルのハッシュ値を更に計算することにより、実際のエッセンシャルハッシュを得る。
4. サーバは、1. で得られたエッセンシャルハッシュ値と 3. で得られたエッセンシャルハッシュ値とを比較する。
5. サーバは、1. で得られたエッセンシャルハッシュ作成日時が、サーバが管理しているアップロード日時よりも古いことを確認する。

サーバは自己の管理する各ホームページ作成者の公開鍵を保持しているので、署名データを復号することができる。公開鍵は特権領域に保管されている。

万一、クラッカーがサーバに侵入し、ホームページデータや署名データを改竄したとしても、秘密鍵を持たないクラッカーは改竄したホームページデータに対応する正しい署名データを作成することができず、改竄が発見される。

クラッカーがある時点における全ホームページデータと署名データをコピーしておき、当該ホームページが正規作成者により更新された後に、その全ホームページデータと署名データを書き戻すことにより、古いバージョンのホームページへと変更してしまうという攻撃が考えられる。この場合、上記 4. のチェックは通過してしまうが、5. のチェックで改竄を検出できる。

3 実装

本方式は任意のディジタル署名技術および任意のハッシュ関数を用いて良い。本稿では、RSA 公開鍵暗号 [5] と MD5[6] を採用し、システムのプロトタイプを作成した。(ハッシュ値に RSA 暗号で署名する技術は、RSA 署名 [7] と呼ばれている。)

RSA 暗号の暗号化プログラム、鍵生成プログラム、MD5 プログラムは名古屋工業大学岩田研究室にて公開されているライブラリ [8] を用いた。これらを利用し、下記各種コマンドを作成した。

- **f_hash** *file*

  引数 *file* に指定されたファイルのデータを MD5 にてハッシュ化する。生成されるハッシュ値は必ず 128bit となる。

- **d_hash** *directory hash_index_file*

  ホームページデータ用のルートディレクトリを引数 *directory* に指定することにより、その階層以下の全てのディレクトリに格納されているホームページデータ (HTML ファイル、JPG ファイル等々) を順次、**f_hash** コマンドに引き渡す。**f_hash** により計算されたハッシュ値とファイル名を一覧表形式にまとめ、ハッシュインデックスファイル *hash_index_file* を生成する。

- **e_hash** *hash_index_file e_hash_file*

hash_index_file を f_hash コマンドに引き渡し、エッセンシャルハッシュ値を計算する。ハッシュ値に加え、エッセンシャルハッシュ作成日時、作成者のユーザ名を *e_hash_file* として出力する。
なお、RSA 暗号方式で暗号化を行う場合、メッセージ長は公開鍵よりも小さくなければいけないが、f_hash の出力は 128bit であるため支障は無い。

- RSA_encrypt *prv_key* *e_hash_file* *sig_file*

  e_hash_file データを秘密鍵 *prv_key* により暗号化することにより、ディジタル署名し、署名データ *sig_file* を出力する。

- RSA_decrypt *pub_key* *sig_file* *e_hash_file*

  sig_file データを公開鍵 *pub_key* により復号することにより、エッセンシャルハッシュデータを得る。ハッシュデータは *e_hash_file* に出力される。(なお、RSA 暗号方式の性質上、実際には、RSA_encrypt と RSA_decrypt は同一プログラムである。)

更に、上記コマンドをシェルスクリプト言語にて以下のように結合し、本システムを実装した。

- ホームページ作成者用プログラム
  signature_maker
  1. d_hash *directory* *hash_index_file* により、ハッシュインデックスファイルを作成する。
  2. e_hash *hash_index_file* *e_hash_file* により、エッセンシャルハッシュを作成する。
  3. RSA_encrypt *prv_key* *e_hash_file* *sig_file* により、エッセンシャルハッシュにディジタル署名する。

- サーバ用プログラム
  HP_checker
  1. RSA_decrypt *pub_key* *sig_file* *e_hash_file* により、作成者から送られたエッセンシャルハッシュ値とエッセンシャルハッシュ作成日時データを得る。
  2. サーバにアップロードされている作成者の全ホームページデータから、
     d_hash *directory* *hash_index_file* により、ハッシュインデックスファイルを実際に計算する。
  3. e_hash *hash_index_file* *e_hash_file* により、実際のエッセンシャルハッシュ値を得る。
  4. 1. で得られたエッセンシャルハッシュ値と 3. で得られたエッセンシャルハッシュ値とを比較する。
  5. 1. で得られたエッセンシャルハッシュ作成日時が、サーバが管理しているアップロード日時よりも古いことを確認する。

ホームページ作成者は、ホームページデータをサーバにアップロードする前に、プログラム signature_maker により署名データを作成する。サーバは cron プログラム等を利用し、プログラム HP_checker を定期的に実行する。

4 認証局を利用した改竄パトロール

公の認証局を仮定することにより、本方式を更に発展させることが可能である。認証局は各ホームページ作成者の公開鍵と、最新アップロード日時情報を保管する。まず、WWW サーバが各ホームページ作成者ごとに公開鍵と秘密鍵のペアを作成した際に、秘密鍵が作成者に送られると同時に、公開鍵が認証局に送られ、登録される。そして、2.1 節の手順 6 において、WWW サーバは作成者ごとの最新アップロード日時情報を認証局にも送信する。

4.1 閲覧者における改竄チェック

閲覧者は認証局からホームページ作成者の公開鍵と最新アップロード日時情報を取得することができるので、閲覧者のローカルマシンにプログラム HP_checker をインストールすれば、閲覧者側での改竄チェックが可能となる。これにより、作成者-閲覧者でのダブルチェックが可能なホームページ改竄チェック方式が実現する。

4.2 認証局による改竄チェック

前章までは、WWW サーバ内の特権領域 (スーパーユーザ権限でのみアクセス可能な領域) にクラッカーが侵入することは難しいという前提を置いてい

た。しかし、クラッカーがWWWサーバのスーパーユーザ権限を手にいれることも十分に考えられる。

もし、クラッカーがWWWサーバの特権領域に侵入した場合には、プログラムHP_checkerの定期的実行を停止させたり、WWWサーバ内の公開鍵や最新アップロード日時情報を不正に書き換えることにより、WWWサーバの行う改竄チェックパトロールを無効化するという攻撃を行うだろう。

だが、この攻撃に対しては、認証局が定期的にWWWサーバにおけるHP_chekerの稼働状況を確認したり、WWWサーバ内の公開鍵や最新アップロード日時情報が認証局に登録されているものと等しいかどうかをチェックすることにより、対処することができる。

5 まとめ

ホームページの改竄の有無をWWWサーバが定期的にチェックするホームページの改竄パトロールシステムを提案し、そのプロトタイプを実装した。本システムを実用化するに当たっては、以下の課題を解決しなければならないと思われる。

- 頻繁にパトロールを行うほど、改竄検知率は向上するが、その分、サーバに負荷がかかることになる。最適なパトロール間隔を導出する必要がある。
- パトロールしている間はホームページのアップロードを禁止しなければならない。
- 掲示版等のCGIを使い、サーバ上のホームページデータを直接書き換えるホームページに対しては、改竄チェックができない。
- ホームページ作成者は、データの一部を書き換えただけの場合にも、ホームページ全体のエッセンシャルハッシュを求め直す必要がある。

謝辞

本研究は、一部、(財)テレコム先端技術研究支援センターの助成を受けた。ここに謝意を表する。

参考文献

[1] Goldwasser,S.,Micali,S. and Rivest,R: A Digital Signature Scheme against Adaptive Chosen Message Attack,SIAM Journal on Computing, 17,2,pp.281-308(1998).

[2] J.Zhao and E.Koch: Embedding robust labels into images for copyright protection, Proceedings of ICIPR (1995).

[3] インターネットマークス, http://www.tao.go.jp/prs10102.htm .

[4] Menezes,A.J.,van Oorschot,P. and Vanstone, S.A.: Handbook of Applied Cryptography, CRC Press (1996).

[5] Rivest, R., Shamir, A. and Adleman,L.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems,Communications of the ACM, Vol.21, No.2, pp.120-126(1978).

[6] The MD5 Message-Digest Algorithm http://info.inernet.isi.edu/in-notes/rfc/files /rfc1321.txt .

[7] Ballare,M. and Rogaway,P.: The Exact Security of Digital Signatures - How to Sign with RSA and Rabin, Proc. of Eurocrypt'96,LNCS 1070, Springer-Verlag, pp.399-416(1996).

[8] 暗号ライブラリ 名古屋工業大学岩田研究室, http://mars.elcom.nitech.ac.jp/Research/MM/security/aicrypto.html .