社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2004－DSM－32 (8)
2004／3／29

WWW Proxy サーバにおけるログ管理方式の提案

篠田和幹† 若山公威† 杉江修‡ 鈴木春洋‡ 岩田彰†

†名古屋工業大学 電気情報工学専攻
〒466-8555 愛知県名古屋市昭和区御器所町
‡株式会社中電シーティーアイ
〒450-0003 愛知県名古屋市中村区南1丁目27番2号

E-mail : deer@mars.elcom.nitech.ac.jp

インターネットの普及により、不正アクセスや、機密情報の漏洩などが問題となっている。WWW Proxy サーバを利用してインターネットへアクセスする場合も同様のことが言える。

そこで本稿では、WWW Proxy サーバに Schneier らのログ管理方式を基にしたアクセス先を隠蔽する方法を提案する。そしてアクセス先の URL を隠蔽することによって生じる問題点について検討する。

A proposal of log management method for a WWW proxy server

Kazumoto Shinoda† Kimitake Wakayama† Shunyo Suzuki‡
Osamu Sugie‡ Akira Iwata†

†Nagoya Institute of Technology
Gokiso-cho, Syouwa-ku, Nagoya, Aichi, 466-8555, Japan
‡ ChudenCTI Co.,Ltd.
1-27-2 Nakamura-ku, Nagoya, Aichi, 450-0003, Japan

By the spread of the Internet, unlawful access, disclosure of confidential information pose a problem. The same thing can be said when accessing to the Internet using a WWW proxy server.

In this paper, we propose the method of concealing URL based on Schneier's log management method to a WWW proxy server. And the problem produced by concealing URL is examined.

1. はじめに

近年インターネットの普及により様々なサービスを利用することが可能である。そのような状況の中で不正アクセスが増加している。この不正アクセスを検出する証拠物としてアクセスログは重要な役割を担っている。そのため、アクセスログは厳重に管理されなければならない。

しかし、一般的に WWW Proxy で用いられているアクセスログはログの管理者だけがアクセスできるようなアクセス制限はなされているが、ログの内容は平文であるため、攻撃者に管理者権限を取得されるとログの閲覧や改竄が可能になってしまう。そのため、ログを安全に保管し、改竄を検出できる仕組みが必要である。

Schneier らは、信頼できる第三者機関(TTP : Trusted Third Party)を設け、ログを暗号化して安全に保管する方式を提案した([1])。この方式は、TTP に保存する値は認証に用いる値の初期値のみであるため、通信の負荷が少ない。また、ハッシュチェーンを使用することで、ログの順序性も保証している。以降この方式を Schneier らの方式と呼ぶ。

Schneier らの方式においても、ログの管理者であれば復号化することにより、ログの内容の閲覧が可能である。ログの管理者であれば、ログの内容を閲覧できることに問題はないがその情報を第三者に公開し

てしまうとプライバシーの侵害など問題になる。

そこで本稿では、Schneierらの方式を改良しアクセス先を隠蔽した方式をWWW Proxyに実装する。また、提案方式における問題点を検討する。

2. Schneierらの方式の概要

2.1. エンティティの定義

Schneierらの方式では次に示す3つのエンティティが存在する。

- U：ログを保管するエンティティ。耐タンパ性がないため、ログを改竄される可能性がある。
- T：耐タンパ性のある信頼できるマシン。このマシンに保存されている値は改竄されないことを保証する。
- V：ログの検証を行うエンティティ。Uが検証者を兼ねる場合もある。

2.2. ログ管理プロトコル

ログエントリの追加の方式を示す前に以降で使用する表記方法を示す。

X,Y ：XとYの連結。

$hash(X)$ ：Xの一方向ハッシュ値。

$E_K(X)$：共通鍵Kを使用しXを共通鍵暗号化。

$MAC_K(X)$：Xのメッセージ認証。鍵Kを用いる。

D_j：j番目のログエントリのデータ。

W_j：D_jのログエントリタイプ。パーミッションマスクの役割を果たし、Vが検証可能かどうかを表す。

A_j：j番目のログエントリの認証用の鍵。

A_{j+1}=hash("Increment Hash", A_j)

K_j： D_jを暗号化する鍵。

K_j=hash(*"Encryption Key"*, W_j, A_j)

Y_j：各ログエントリのハッシュチェーン。

Y_j=hash(Y_{j-1}, $E_{K_j}(D_j)$, W_j)

Z_j：Y_jのMAC値。

$Z_j = MAC_{A_j}(Y_j)$

L_j：j番目のログエントリ。

$L_j = W_j,\ E_{K_j}(D_j),\ Y_j,\ Z_j$

ログエントリ追加の流れを図1に示す。ログエントリを追加において使用済みの鍵は破棄される。図1では、データの暗号化が終わり次第K_jを破棄し、次のログエントリに用いるA_{j+1}を計算し次第A_jの値を破棄することで、U上に以前の鍵の値を保管しないようにする。このように順次鍵を破棄することで攻撃者がログの管理権限を得たとしても以前のログエントリを閲覧することも、改竄することも不可能になる。

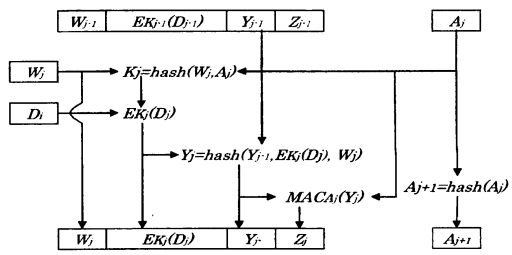

図 1：ログエントリ追加の流れ

これらの鍵は全てA_0の値を基に計算されるため、A_0を攻撃者に知られてしまうと全てのログエントリを読むことが可能になる。そのため、A_0は安全にTに保存できなければならない。

この認証の基になる値A_0をUからTに安全に保存する方式を図2に示す。

3. 提案方式

Schneierらの方式をWWW Proxyに適用するにあたって、ログの管理者の行う作業を定義する。また、この定義を基に、ログエントリのデータを保存の方式を変更する。

	U		T
forms	$K_0, d, d^+, ID_{\log}, C_U, A_0$		
	$X_0 = p, d, C_U, A_0$		
	$M_0 = p, ID_U, PKE_{PK_T}(K_0), E_{K_0}(X_0, SIGN_{SK_U}(X_0))$	→ verifies	M_0
forms	$L_0(D_0 = d, d^+, ID_{\log}, M_0)$		
		forms	$X_1 = p, ID_{\log}, hash(X_0)$ and K_1
verifies	M_1	←	$M_1 = p, ID_T, PKE_{PK_U}(K_1), E_{K_1}(X_1, SIGN_{SK_T}(X_1))$
forms	$L_1(D_1 = M_1)$		

K_0 ：セッション鍵
d ：現在時間
d^+ ：タイムアウトする時間
$ID_{\log}$ ：ログファイルの識別子
C_U ：Uの公開鍵証明書
A_0 ：Aの初期値
p：プロトコルの識別子
E_K ：共通鍵暗号方式
PKE_{PK} ：公開鍵暗号方式
$SIGN_{SK}$ ：電子署名

図 2：A の初期値の受け渡し方法

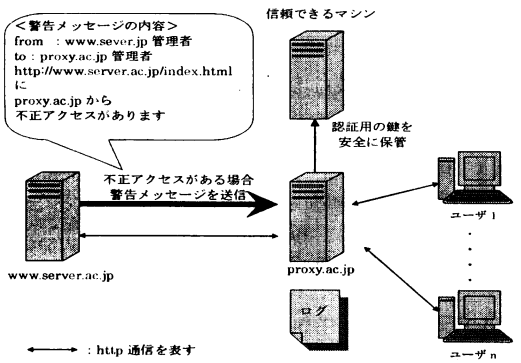

図 3：提案方式の WWW Proxy の構成

3.1. エンティティ

A の値を保存する信頼できるマシン(Schneier らの方式における T)と WWW Proxy として機能するマシン(Schneier らの方式における U) と Proxy を使用するユーザから構成される(図 3)。WWW Proxy の場合、ユーザから新たなアクセスがあればログエントリを追加し、ログファイルの検証はログの管理者が行う。このため、Schneier らの方式におけるパーミッションマスク Wは必要なくなる。

3.2. ログの操作

ログの管理者がログファイルに対して行う操作を以下にまとめる。

- ログの閲覧(部分的に)
- ログの検索
- ログのバックアップ

WWW Proxy のログは、該当するログエントリを検索可能ならば、全てのログの内容を閲覧出来なくてもいいとする。

前提 1：検索時に該当ログエントリを検索できればログエントリすべてを閲覧できなくてよい

また、ログファイル以外からログエントリ検索に用いるアクセス先の URL を入手できなければならない。不正アクセスなどがあった場合には図 3に示すように外部から不正アクセスの通知がくるため、この通知に記述されている URL から入手することが可能である。

前提 2：検索対象にあるアクセス URL をログファイル以外から入手可能

このような前提がある場合には、アクセス先の URL をハッシュ値として保存することができる。このように、ハッシュ値として保存する場合には、リプレイ攻撃により容易にアクセス先の URL を割り出せないようにソルトを追加する。

3.3. 提案方式のログ管理プロトコル

提案方式で新たに使用する表記方法を以下に示す。

D_j : j番目のログエントリに入るデータ。

D_j はさらに以下のように表記する(n : ログの要素数、n=0 の要素をアクセス先の URL とする)。各要素はアクセス先 URL や時刻情報から構成される。

$$D_j = d_{j(0)} + d_{j(1)} + \cdots + d_{j(n)}$$

S_j : アクセス先の URL($d_{j(0)}$)のハッシュ値を計算するときに使用するソルト。

提案方式におけるログエントリの追加の流れを図 4に示す。表記は Schneier らの方式と同様である。

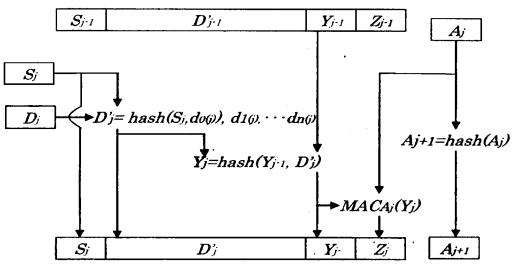

図 4 : 提案方式のログエントリ追加の流れ

3.4. ログエントリの検索

ログファイルから該当するログエントリを検索する場合には、前提 2 に示すようにログファイル以外から検索するアクセス先 URL を入手しなければならない。図 3にあるようにログファイル以外から URL を入手可能であるため、ここではその URL を入手した後の検索の流れを示す。

1. アクセス先 URL の値 R を入手する。
2. ログファイル $L_0 \sim L_f$ (f: 最後のログエントリを表す)を取得する。
3. j 番目のエントリにある S_j の値と R からハッシュ値を求める(j=0..f)。 $hash(S_j, R)$
4. 手順 3 で求めたハッシュ値と L_j 中の $hash(S_j, d_{j(0)})$ と比較し、一致するかを確認する。
5. 以降手順 3,4 を繰り返して $L_0 \sim L_f$ をすべて確認する。

3.5. 実装上の問題点と対策

提案方式の手法ではアクセス先の URL をそのままハッシュ値として保存すると検索できなくなる可能性がある。検索できなくなる場合としては URL としては等しいが、文字列としては異なる場合がある。

- エスケープ符号化
  URL において"%7E"と"~(チルダ)"は等しい。
- ホスト名と IP アドレス
  ホスト名の mars.elcom.nitech.ac.jp と IP アドレスの 133.68.11.4 は同じマシンを表す。

どちらの場合も URL としては同じものを表すが、ハッシュ値をとると異なる値になってしまうため検索が出来なくなってしまう。このような場合はログ作成時に統一して保存することにより、解決することが可能である。検索時にも同じように、統一する必要がある。

また IP とホスト名が 1 対 1 に対応している場合は、単純にログファイル作成時に置換することで対応可能であるが、負荷分散などの理由から一つのホスト名が複数の IP アドレスに対応している場合などは単純に置換するだけでは解決できない。

このような場合にはログエントリ作成時に工夫が必要となる。このような場合には一つのアクセスに対し複数ログエントリを記入することで対応する。

4. 実装と測定

4.1. 実装環境

ここまでに提案した方式をフリーの WWW Proxy である squid2.5 STABLE4 と本研究室で開発した aicrypto2.1(ハッシ

ュ関数：SHA-1，メッセージ認証：HMAC-SHA-1)を使用して実装した。WWW Proxy の実装環境は以下のとおりである。

CPU：Pentium4　1.7GHz,
メモリ：512 MB(DDR),
OS：Vine Linux 2.5

4.2. 測定結果

測定は、通常の平文のログファイルを作成する場合と、提案方式でログファイルを作成する場合の 2 通り行った。2 つの方式で 10,000 行のログファイルを作成し、1 行のログエントリ作成にかかる時間と、作成したログファイルから該当するログエントリの検索にかかる時間を測定した。

表 1にログエントリ作成にかかる平均時間を示す。表からわかるように提案手法は平文のログの場合の約 4.5 倍の時間がかかる。1 秒当たりに処理できるログエントリの行数に変換すると提案手法においては約 14,700 行のログエントリを書き込むことが可能である。1 秒間にこれほど多くのアクセスが集中することは考えにくい。また、名古屋工業大学の WWW Proxy のログエントリの行数が 1 日あたり約 4,300,000 行であることからも、実環境での利用が十分可能であるといえる。

表 1：1 行のログエントリ作成時間

平文のログ(μs)	提案方式のログ(μs)
15.2	68.1

表 2にログファイルの検索にかかる時間を示す。提案方式の測定では、ハッシュチェーンの値と MAC 値の検証をログエントリごとに行う場合と、最後のログエントリだけ行う場合の 2 つ測定を行った。検索にかかる時間は平文であっても比較的時間がかかることがわかる。

最後のログエントリの検証が正しければ以前のログすべてが正しいことが保障されるため、ログの検索を行うだけならば平文の約 6.7 倍の時間で検索が可能である。

ただし、改竄などがあった場合に何処から改竄されているか調べるためにはすべてのログエントリについてハッシュチェーンの値と MAC 値の検証を行わなければならない。

表 2：ログの検索時間

		10,000 行	平文との比
提案方式	全て検証	0.176[s]	11.0
	最後のみ検証	0.107[s]	6.7
平文のログ		0.016[s]	1

5. 考察

測定結果から実環境において実装することが可能であることを示した。ここでは、提案方式の利点と欠点をまとめる。

5.1. 提案方式の特徴

5.1.1. 提案方式の利点

- アクセス先 URL の隠蔽が可能。
  アクセス先の URL をハッシュ値として保存するためログの管理者においてもアクセス先の URL を知ることができない。
- ログエントリすべてを暗号化しないため、検索時間を短縮することが可能である。

5.1.2. 提案方式の欠点

- アクセス先 URL が読めないためログファイルを IDS(Intrusion Detection System：進入検知システム)などに二次利用することができない。

6. おわりに

本稿では Schneier らの方式を改良したアクセス先の URL を隠蔽するログ管理方式を提案し、WWW Proxy に実装した。ログの作成にかかる時間からも実環境で使用できることを示した。

今後は、提案方式の WWW Proxy 以外への適用を検討したい。

参考文献

[1] Bruce Schneier, John Kelsey : "Cryptographic Support for Secure Logs on Untrusted Machines" the Proceedings of the 7th USENIX security Symposium(1998)

[2] 安東 学, 松浦幹太, 馬場 章. "分散環境で保存されるログファイルにおける各ログエントリ間の順序関係保証方法に関する考察", コンピュータセキュリティシンポジウム(CSS)2002論文集, 情報処理学会シンポジウムシリーズ, Vol.2002, No.16, pp.1-6, Oct. 2002.