Vol.2013-CSEC-62 No.18
Vol.2013-SPT-6 No.18
2013/7/18

ライブラリの置き換えによるVM外部への安全なログ転送方式の評価

佐藤 将也[1,a)]　山内 利宏[1]

概要：ログは，計算機の動作を把握するための重要な情報である．しかし，攻撃や問題の発生により，ログの改ざんや消失が起こる可能性がある．この問題への既存の対処の多くは，対処する以前にログを改ざんされる恐れがある．また，VM 上の OS の種類に応じた対処や性能低下が問題となる．これらの問題への対処として，ライブラリの置き換えによる VM 外部への安全なログ転送方式を提案した．提案方式では，ログ発行時に VMM にログの転送を依頼するように，VM 上のライブラリを置き換える．VMM は VM からログを取得し，ログの取得元とは異なる VM で保存する．これにより，多種の OS へ容易に適用でき，性能低下の小さい方式を実現した．本稿では，提案方式の評価について述べる．提案方式の有効性を示すために，ログの改ざんを防止できるか検証した．また，多種の OS への適用の容易さを評価した．性能評価では，AP の性能への影響と複数台の VM を走行させた場合の性能の変化を評価した．

1. はじめに

計算機の動作を把握するためには，計算機上のログが重要である．コンピュータセキュリティにおけるログの重要さについては，文献 [1] でも指摘されている．しかし，攻撃や問題の発生により，ログの改ざんや消失が起こる可能性がある．例えば，攻撃者は，攻撃の痕跡を消すために，ログを改ざんする [2], [3], [4].

これらの問題へ対処するために，ログを保護する手法が研究されている [5], [6], [7], [8], [9]．これらの研究では，ファイルへ書き出されたログやログ収集プログラムへの攻撃によるログの改ざんを防止できる．しかし，カーネル内におけるログの改ざんには対処できないものが多い．我々の既存研究 [9] は，これらの攻撃へ対処できる．しかし，多種の OS への対応や性能低下に関する問題がある．

また，クラウド環境の普及に伴い，仮想化環境におけるセキュリティが重要な課題となっている [10], [11]．同様に，クラウドアプリケーションにおけるログ管理やデジタルフォレンジックについて研究されている [12], [13]．しかし，仮想化環境において，仮想計算機上のログを保護する研究は少ない．

そこで，ライブラリの置き換えにより VM 外部へログを転送する手法を提案した [14]．提案方式では，ログを発行する関数内に特定の命令を実行するように，VM 上のライブラリを置き換えておく．VMM は，この命令を契機にログを VM から取得し，ログ保存用の VM に転送する．これにより，ログがカーネルへ到達する前に，VM 外部へログを転送できる．これにより，カーネル内やログ収集プログラム内でログを改ざんする攻撃やログファイルを改ざんする攻撃に対処できる．提案方式では，ライブラリへの修正を最小限に抑えることにより，多種の OS への対応が容易となる．また，既存方式 [9] と異なり，ログを発行する関数の呼び出し時のみ，VMM へ処理を移行する．このため，性能低下は最小限に抑えられる．

本稿では，提案方式の評価について述べる．評価では，提案方式を導入した環境において，ログを改ざんする攻撃を提案方式により防止できるか検証する．また，多種の OS への適用の容易さを確認するために，Linux と FreeBSD に提案方式を導入し，導入に必要な追加コードの量を比較する．さらに，提案方式の導入による性能低下を評価する．評価では，Web サーバの性能を測定する．また，複数 VM が同時に走行する場合の AP の性能も評価する．

2. 研究背景

2.1 syslog を用いたログの保存

Linux では，ログの保存に syslog が広く用いられている．Linux 上において，AP から syslog を用いてログを保存する際の手順は以下の通りである．

(1) AP 上でログを生成
(2) AP は，syslog 関数を用い，プロセス間通信により，

[1] 岡山大学 大学院自然科学研究科
[a)] m-sato@swlab.cs.okayama-u.ac.jp

syslog デーモンへログを送信

(3) syslog デーモンは，ログを受信し，ファイルへ書き出し
syslog デーモンは，AP から受信したログを，ログの書き出しポリシに従い，ログファイルへ書き出す．syslog デーモンには，受信したログを暗号化した通信路でリモート計算機に転送する機能を持つ rsyslog や syslog-ng もある．

2.2 想定される攻撃

2.1 節で示したログの保存手順から，ログを改ざんする契機は，以下のように分析できる．プロセス間通信やファイルへの書き出しでは，必ずカーネルを経由する．このため，カーネル内でログを改ざんされる可能性が高い．

（契機 1） AP がログを生成し，syslog デーモンへログを送信するまでの間

（契機 2） AP が syslog デーモンへログを送信し，syslog デーモンがログを受信するまでの間

（契機 3） syslog デーモンが AP からのログを受信し，ファイルへの書き出し処理を開始するまでの間

（契機 4） ファイルへ書き出す処理を行っている間

（契機 5） ファイルへ書き出された後

（契機 1）でログを改ざんするには，ログの生成元の AP を攻撃する．このためには，攻撃対象の AP を事前に攻撃者が特定しておく必要がある．また，AP へ攻撃するためには，攻撃者は，カーネルへのマルウェアの挿入または管理者権限の取得に成功していると推測できる．（契機 2）でログを改ざんするには，カーネルへマルウェアを挿入しておく必要がある．カーネル内でソケット通信を監視することで，AP から syslog デーモンへのログの転送を検知し，改ざんできる．この攻撃を行うマルウェアとして，adore-ng[2] がある．（契機 3）でログを改ざんするには，syslog デーモンの置き換え，またはログの書き出しポリシの改ざんが有効である．syslog デーモンを置き換えるマルウェアとして，tuxkit[3] がある．（契機 4）では，（契機 2）と同様の手法によりログを改ざんできる．カーネル内でファイルアクセスを監視することにより，ログを改ざんできる．（契機 5）でログを改ざんするには，ログファイルへのアクセス権限が必要となる．ログファイルを改ざんする機能を持つマルウェアとして，LastDoor[4] がある．

以上のように，ログの保存経路の様々な箇所でログが改ざんされる恐れがある．特に，（契機 2）～（契機 4）は，ログファイルへのログの書き出し処理で共通であるため，攻撃者は，syslog を用いたログの書き出しを容易に監視し，ログを改ざんできる．

2.3 既存のログ保護手法

（契機 1）や（契機 3）におけるログの改ざんを防止するには，AP の利用するメモリがカーネルや他の AP から書き換えられないようにする手法 [16], [17] がある．また，（契機 3）以降の契機におけるログの改ざんを防止する手法として，走行中の完全性を保証した syslog デーモンによりリモート計算機へログを転送する手法 [7] がある．

（契機 2）や（契機 4）におけるログの改ざんを防止する手法として，走行中のカーネルの完全性を保証する手法 [18] がある．この手法は，利用者が許可していない不正なコードのカーネル空間での実行を防止する．

(契機 5) におけるログの改ざんを防止する手法 [5], [6], [8] が提案されている．また，ファイルへのアクセス権限の不正な変更を防止する手法 [19] が提案されている．これらの手法により，(契機 5) におけるログの改ざんを防止できる．

2.4 既存手法の問題

（契機 1）～（契機 5）のいずれかでログを改ざんされると，最終的に得られるログは改ざんされたことになる．このため，ログの改ざんを防止するには，（契機 1）～（契機 5）のすべての契機においてログの改ざんを防止する必要がある．または，早期に計算機外部へログを転送することで，ログの生成元の計算機への攻撃によるログの改ざんを防止する必要がある．

既存手法を複数組み合わせることで，（契機 1）～（契機 5）におけるログの改ざんを防止できる．しかし，複数の手法を組み合わせると，実装や運用管理のコストが高い．また，OS のバージョンアップや多種の OS への対応が困難になる．さらに，複数の手法を組み合わせると，ログの転送における処理手順が増加する．これに伴う性能低下が予測される．

3. ライブラリの置き換えによる VM 外部への安全なログ転送方式

3.1 目的と課題

2.4 節の問題を解決する．この目的を達成するための課題を以下に示す．

（課題 1） できるだけ早期にログを保護し，計算機外部へ転送すること

（課題 2） OS の種類やバージョンの違いへの容易な対応

（課題 3） ログ転送のオーバヘッドの抑制

（課題 1）により，攻撃者がログを改ざんする契機を少なくする．（課題 2）により，多種の OS への適用やバージョンの違いへの容易な対応を実現する．（課題 3）により，より実用的な機構を実現する．

3.2 全体像

図 1 に提案方式の全体像を示す．提案方式では，仮想化技術を用い，VM 上のログを VMM により取得する．（課題 1 ）へ対処するためには，計算機外部へログを転送する必要がある．しかし，通常のプロセス間通信では，通信部分をカーネルが提供するため，カーネル内にマルウェアを

挿入された場合に対処が困難である．そこで，VMM を用い，VM 上におけるログの送信を検知し，ログがカーネルへ到達する前にログを取得する．

提案方式では，VM 上のライブラリをあらかじめ置き換える．保護対象 VM 上の AP は，置き換えたライブラリを用い，以下の手順によりログを VM 外部へ転送する．

(1) AP が VMM に対してログの転送を依頼
(2) VMM は依頼を受信すると，AP のログを VMM 内のバッファにコピー
(3) VMM は，ログ保存 VM 上のログ保存 AP へ通知し，ログ保存 AP は通知を受信すると，VMM に対してログのコピーを要求
(4) ログ保存 AP は，VMM からログを受信し，ファイルへ書き出し

提案方式では，保護対象 VM 上の AP は，VMM に対してログの転送を依頼する．VMM は，依頼を検知すると，保護対象 VM 上の AP からログをコピーし，VMM 内のバッファへ格納する．コピーされたログは，ログ保存 VM へ転送され，ログ保存 AP がファイルへ書き出す．

提案方式の実現では，AP からログの転送を依頼するために，`libc` ライブラリを修正し，syslog 関数の発行時に，ログの転送を依頼する処理を追加する．この処理では，VM exit を発生させる命令を実行する．この命令が契機となり，syslog 関数により syslog デーモンへログが転送される直前に，VMM へのログの転送依頼が発行される．提案方式では，修正したライブラリを利用している VM のみ，ログの転送を依頼できる．例えば，図 1 では，保護対象 VM1 はログの転送を依頼できる．一方，保護対象 VM2 は，ログの転送を依頼できない．

提案方式では，ライブラリを置き換えることでログを VM 外部へ転送する．このため，ログは，カーネルへ到達する前に VM 外部に転送され，異なる VM 上で保存される．これにより（課題 1）へ対処する．できるだけ早期にログを保護するには，ログの生成処理を検知し，保護するべきである．しかし，その場合，AP ごとに対処する必要があり，（課題 2）へ対処できない．提案方式では，ライブラリを置き換えることで，ライブラリを利用したログの転送処理すべてに対応できるため，（課題 2）へ対処している．処理手順をできるだけ簡略化し，ログの VM 外部への転送にのみ対処することで，（課題 3）へ対処している．

4. 実現方式

4.1 実装に用いたソフトウェア環境

VMM として，Xen[15] を用いた．VM 上では，ゲスト OS として Linux と FreeBSD を用いた．本稿で述べる実現では，Linux と FreeBSD のライブラリを改変することで提案方式を導入した．各 VM は，Intel VT-x の機能により，完全仮想化している．4.2 節と 4.3 節で述べる実現方式では，Intel VT-x または AMD-V により完全仮想化した環境を前提としている．提案方式は，準仮想化環境においても，実現可能である．ただし，この場合，4.2 節と 4.3 節で述べる方法は利用できない．準仮想化環境で提案方式を実現するには，ログの転送依頼の際，VM から VMM を明示的に呼び出す処理を追加する必要がある．本稿では，準仮想化環境での実現については述べない．

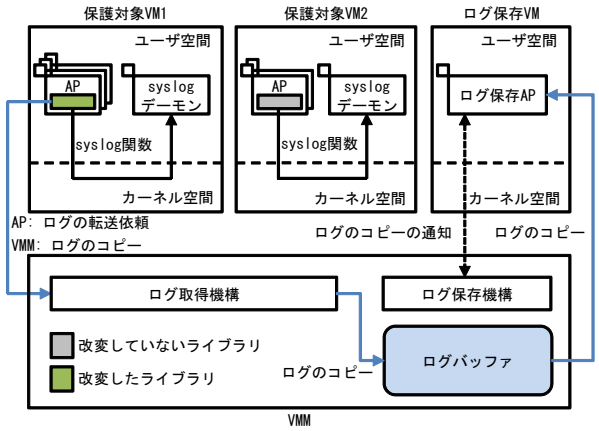

図 1 提案方式の全体像

4.2 AP から VMM へのログの転送依頼

AP から VMM へのログの転送依頼は，CPUID 命令をライブラリ内に埋め込むことで実現する．CPUID 命令は，CPU の情報を取得するための命令であり，CPU 命令の実行により VM や CPU の状態が変化することはない．

VT-x を用いた VM 上では，CPU の状態に関わらず VM exit を起こす命令が 16 種類ある．この中から，VM の状態に影響を与えない CPUID 命令を VMM へのログの転送依頼の手段として用いる．

本稿における実現では，FreeBSD の libc 内の syslog 関数に CPUID 命令を実行する処理を追加した．syslog 関数内では，printf 関数のように，フォーマット文字め，文字列をログに整形する処理の直後に CPUID 命令を実行する処理を追加した．

図 2 に CPUID 命令を用いた VMM へのログの転送依頼の処理の流れを示し，以下で説明する．

(1) AP は，rax レジスタに 0xffff を格納
(2) AP は，rbx レジスタにログを格納したバッファの先頭アドレスを格納
(3) AP は，rcx レジスタにログの長さを格納
(4) AP は，CPUID 命令により VMM へログの転送を依頼

CPUID 命令は，rax レジスタに格納された値に応じた CPU の情報をレジスタに格納する．CPUID 命令の際に rax レジスタに格納される値は，0x0 から 0xd，0x40000000 から 0x4fffffff，および 0x80000000 から 0x80000008 である．このため，未使用の 0xffff を VMM へのログの転送依

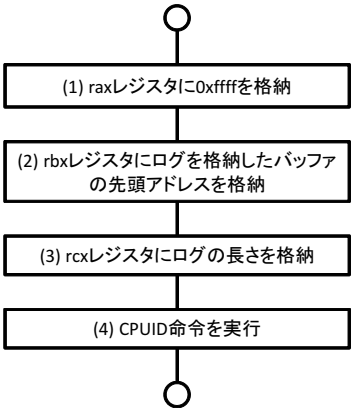

図 2 CPUID 命令を用いた VMM へのログの転送依頼の処理の流れ

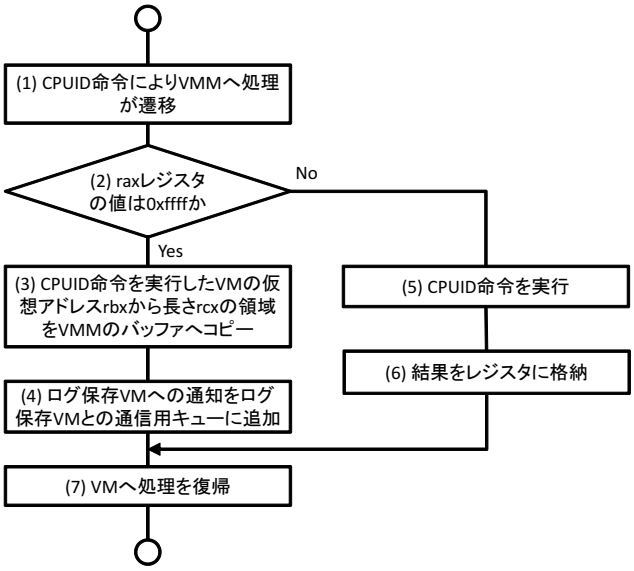

図 3 CPUID 命令による転送依頼の実行後の VMM によるログのコピーの処理の流れ

頼を示す値として用いた．

4.3 VMM による AP 上のログのコピー

図 3 に AP 上のログをコピーする処理の流れを示し，以下で説明する．

(1) CPUID 命令により VMM へ処理が遷移
(2) rax レジスタの値が 0xffff でない場合は (5) へ移動
(3) CPUID 命令を実行した VM の仮想アドレス rbx から長さ rcx の領域を VMM のバッファへコピー
(4) ログ保存 VM への通知をログ保存 VM との通信用キューに追加し，(7) え移動
(5) CPUID 命令を実行
(6) 結果をレジスタに格納
(7) VM へ処理を復帰

rax レジスタの値が 0xffff でない場合は，VMM は，通常の CPUID 命令だと判断し，CPUID 命令を実行後，VM へ処理を復帰する．rax レジスタの値が 0xffff の場合は，CPUID 命令を実行した VM のメモリ領域から該当する領域を VMM 内のバッファへコピーする．その後，ログ保存 VM への通知をログ保存 VM との通信キューに追加し，VM へ処理を復帰する．

5. 考察

5.1 ログの保存経路における安全性

提案方式は，2.2 節で述べた契機のうち，(契機 2)，(契機 3)，(契機 4)，および (契機 5) におけるログの改ざんを防止できる．(契機 1) におけるログの改ざんは，ハイパーバイザを用いたメモリ保護手法 [16] や TrustVisor[17] を併用することで防止できる．この文献 [16] の手法は，カーネル空間からのユーザ空間のメモリの不正な書き換えを防止する．また，TrustVisor[17] は，指定したコードの完全性を保証する．この手法を用いることで，(契機 1) の攻撃を防止できる．しかし，ログは，(契機 1) だけでなく (契機 2) 以降で改ざんされる恐れがある．このため，VM を用いる環境でログを改ざんから保護するには，(契機 1) 〜 (契機 5) のいずれかで，計算機外部へログを転送する手法が必要である．(契機 1) でログを VM 外部へ転送するのが望ましいが，そのためには，AP ごとに実装を変更する必要があり，修正の工数が非常に大きい．提案方式では，修正の工数を最小限に抑えながら，(契機 2) 以降におけるログの改ざんを防止できる．

5.2 提案方式の限界

ライブラリのファイルを置き換えられた場合には，提案方式では，対処できない．また，AP の走行中に，ライブラリの呼び出し箇所を改ざんされた場合には，ライブラリ自体が呼び出されなくなり，ログを転送できない．同様に，メモリ上にロードされたライブラリを改ざんされた場合にも，対処できない．カーネルレベルマルウェアは，これらの攻撃を行える．

AP やライブラリへの攻撃へ対処するには，文献 [16] と [18] の手法を併用するなどして，ライブラリ，AP，およびカーネルの走行中の完全性を保証する必要がある．また，ライブラリのファイルへの攻撃へ対処するには，Tripwire[20] などを用い，ライブラリのファイルの完全性を検証する必要がある．

6. 評価

6.1 目的と環境

以下の項目について評価する．

(1) ログの改ざんと消失の防止
(2) 多種の OS への適用
(3) 性能への影響の評価

ログの改ざん攻撃を行い，提案方式により改ざんを防止できるかを評価した．また，多種の OS への移植の容易さを評価した．最後に，提案方式の導入による AP の性能への

表 1　評価に用いたソフトウェア

VMM	Xen 4.2.0
OS (ログ保存 VM)	Debian (Linux 3.5.0 64-bit)
OS (保護対象 VM)	FreeBSD 9.0.0 64-bit
	Debian (Linux 2.6.32 64-bit)
Web サーバ	thttpd 2.25b
ベンチマークソフト	ApacheBench 2.3
	LMbench version 3

影響，および複数 VM が同時に走行した場合の AP の性能への影響を評価した．

表 1 に評価に用いたソフトウェアを示す．提案方式のプロトタイプは，Xen[15] を用いて作成した．Web サーバの性能評価では，Web サーバとして thttpd を用い，ベンチマークソフトとして ApacheBench を用いた．

評価に用いた計算機は，CPU は Core i7 2600（3.4 GHz, 4 コア），メモリは 16 GB 搭載している．評価で用いた各 VM には，仮想 CPU1 コアと 1 GB のメモリを割り当てている．また，Web サーバの性能測定では，提案方式を適用した VMM 上において，VM 上で Web サーバを動作させ，スループットを測定した．クライアント計算機は，提案方式を適用した計算機とは異なる計算機である．クライアント計算機は，CPU は Pentium 4（3.0 GHz，1 コア），メモリを 1 GB 搭載しており，通信路の帯域幅は 1Gbps である．

6.2　ログの改ざんと消失の防止

6.2.1　カーネルレベルマルウェアによるログの改ざん

カーネル空間におけるログの改ざんへ対処できることを確認するために，adore-ng[2] を用いた．adore-ng は，カーネル空間で動作するマルウェアであり，syslog デーモンへ送信されるログをカーネル空間で検知し，特定の文字列が含まれる場合には，そのログを削除する機能を持つ．実験の結果，adore-ng を挿入した VM 上のログは改ざんされている一方で，提案方式で取得し，VM 外部へ保存したログは改ざんされていなかった．また，VM 上のログを提案方式で取得したログと比較することで，改ざん箇所を特定できた．

6.2.2　syslog デーモンへの攻撃によるログの改ざん

syslog デーモンへの攻撃を防止できるか確認するために，syslog デーモンのポリシファイルを改ざんし，ログがファイルへ書き出されないような環境で実験した．この環境では，VM 上では，AP が syslog 関数によりログを発行しても，ファイルには書き出されない．一方，提案方式では，ログを取得できた．この実験より，提案方式では，ポリシファイルへの攻撃によりログが消失する問題へ対処できることを確認した．このことから，syslog デーモンを置き換えることでログを改ざんする tuxkit[3] のような攻撃も防止できることが分かる．

```
void
  __vsyslog_chk(int pri, int flag, const char *fmt, va_list ap)
  {
            int saved_errno = errno;
            char failbuf[3 * sizeof (pid_t) + sizeof "out of memory []"];

+           reg_t regs;
+           regs.rax = 0xffff;
+
  #define     INTERNALLOG   LOG_ERR|LOG_CONS|LOG_PERROR|LOG_PID
            /* Check for invalid bits. */
            if (pri & ~(LOG_PRIMASK|LOG_FACMASK)) {
***************
*** 278,283 ****
--- 297,308 ----
            if (LogType == SOCK_STREAM)
              ++bufsize;

+           regs.rbx = (unsigned long)buf;
+           regs.rcx = bufsize;
+
+           cpuid_logxfer(regs.rax, &regs);
+           regs.rax = regs.rbx = regs.rcx = 0;
+
            if (!connected || __send(LogFile, buf, bufsize, send_flags) < 0)
              {
                if (connected)
```

図 4　提案方式を適用していない libc と適用した libc のソースコードの比較

6.2.3　syslog デーモンの停止によるログの消失

syslog デーモンの停止によりログを保存できない環境において，提案方式がログを取得できるか確認した．tuxkit や adore-ng などのマルウェアは，マルウェアの挿入手順がログに記録されるのを防ぐために，マルウェアのインストール時に syslog デーモンを停止させる．syslog デーモンが停止した環境では，AP の呼び出した syslog 関数はすべて失敗し，ログはファイルへ書き出されない．一方，提案方式では，ログの送信前に VMM へ転送依頼を発行するため，ログを取得できる．ただし，これは，ライブラリの修正方法に依存する．ライブラリによっては，syslog 関数内において，syslog デーモンに接続できない場合には，即座にログの送信処理を中止するものがある．このため，VMM へログの転送を依頼する処理を確実に呼び出すためには，ログの送信処理が中止されるような箇所よりも前に，転送を依頼する処理を挿入する必要がある．

6.2.4　ログファイルの改ざん

保護対象 VM 上のログファイルに書き出されたログを改ざんした．ログファイルを改ざんする機能は，LastDoor[4] など，多くのマルウェアが持つ．提案方式では，取得したログをログ保存 VM 上のファイルへ保存する．このため，保護対象 VM 上におけるログの改ざんの影響を受けない．

6.3　多種の OS への適用

提案方式を多種の OS に適用できるかを確認するために，Linux と FreeBSD を対象として実験した．実験では，Linux と FreeBSD の両方において，`libc` のソースコードに 20 行追加することで，提案方式を適用できた．追加したソースコードの一部を**図 4** に示す．

提案方式の導入では，`cpuid_logxfer` 関数を `send` システムコールの発行の直前に挿入した．図 4 に示していない追加箇所は，`regs` 構造体の定義と `cpuid_logxfer` 関数の定義である．これらの追加コードは，(1) レジスタに適切

表 2 提案方式の導入による syslog 関数のオーバヘッド

	処理時間 (μs)	オーバヘッド (μs (%))
Xen	31.47	−
提案方式	33.38	1.91 (6.08%)

表 3 Web サーバのスループット

ファイルサイズ	VMM	スループット（要求数／秒）	相対性能
1 KB	Xen	1,433.30	1
	提案方式	1,298.58	0.91
10 KB	Xen	671.23	1
	提案方式	668.34	1.00
1,000 KB	Xen	11.41	1
	提案方式	11.41	1.00

な値を格納し，(2) `cpuid` 命令を実行している．20 行の追加コードにより適用できることから，提案方式の導入は十分容易であるといえる

6.4 性能への影響の評価

6.4.1 測定対象と測定環境

提案方式の導入による AP の性能への影響を評価するために，以下の項目について，提案方式を導入していない場合と導入した場合における性能を測定し，比較する．

(1) syslog 関数
(2) Web サーバ
(3) 複数 VM が同時に走行する場合の AP の性能

（1）と（2）の項目は，文献 [14] において既に評価している．しかし，文献 [14] における評価では，仮想 CPU を特定の物理 CPU コア上に固定せずに性能を測定している．このため，ログ保存 VM と保護対象 VM が同じ物理 CPU コア上で動作した場合に，著しく性能が低下する恐れがある．このため，本評価では，ログ保存 VM と保護対象 VM を別々の物理 CPU コア上で動作するように設定し，再度測定する．

（3）の項目では，syslog 関数を発行するプロセスが動作する VM を作成し，この VM が同一計算機上で並列に複数動作した場合における Web サーバの性能を測定する．これにより，提案方式を導入した場合において，他の VM からのログの転送依頼が AP の性能へどの程度影響するのか評価する．

6.4.2 syslog 関数

表 2 に提案方式を導入した場合の syslog 関数の性能低下を示す．提案方式では，syslog 関数の呼び出しにおける性能低下は，1.91 マイクロ秒（6.08%）である．頻繁に呼び出される関数においては，6.08%の性能低下は大きい．しかし，syslog 関数は，頻繁に呼び出される関数ではない．このため，提案方式の導入による性能低下は，十分小さいといえる．

また，システムコールの性能の変化を LMbench を用いて評価した．LMbench は，ファイルの作成と削除，プロセス生成，システムコール発行による性能低下などを測定する．LMbench による測定では，性能低下は観測できなかった．このことから，提案方式は，syslog 関数の性能にしか影響しないことが分かる．

6.4.3 Web サーバ

提案方式を導入した場合における性能低下を測定した．クライアント計算機上の ApacheBench は，1 KB, 10 KB, および 1,000 KB のファイルを要求する．ApacheBench による評価では，ファイルを要求する際の並列度を 1，要求数を 1,000 に設定し，評価した．

表 3 に各環境におけるスループットの測定結果と比較結果を示す．提案方式の導入による性能低下は，要求するファイルサイズが 1 KB のときは，約 9%であった．要求するファイルサイズが 10 KB のときと 1,000 KB のときは，大きな性能低下は観測できなかった．この評価結果より，提案方式の導入による AP の性能への影響は，十分小さいといえる．また，提案方式の導入による性能低下は一定であり，syslog 関数の呼び出しの際にしか発生しない．このことから，AP による処理時間が増加するほど，提案方式の導入による AP の性能への影響は小さくなるといえる．

6.4.4 複数 VM が同時に走行する場合の AP の性能

複数 VM が同時に走行する場合の AP の性能への影響を評価するために，単一計算機上で複数 VM が走行する環境において，Web サーバの性能を測定した．これらの VM 上では，syslog 関数を毎秒 1 回呼び出すプロセスが走行する．本評価で用いた計算機には，4 つのコアを持つ CPU が搭載されている．各 VM のコアの割り当てでは，測定対象の VM の性能のばらつきを抑えるために，ログ保存 VM をコア 0 に，Web サーバを走行させる VM をコア 1 に，その他の VM をコア 2 とコア 3 に配置した．また，VM 数の増加による性能への影響を明らかにするために，コア 2 とコア 3 上の合計 VM 数を 2, 4, 6, 8, 10, 12 の順に増加させ，各環境において性能を測定した．コア 2 とコア 3 の VM 数は等しくなるように VM を配置した．

表 4 に各環境における測定結果を示す．また，**図 5** に各環境における性能の変化の様子を示す．同時に走行する VM 数が増加すると，Web サーバの性能も低下している．ファイルサイズが 10 KB 以上の場合，提案方式の導入による性能低下は 10%以下である．特に，ファイルサイズが 1,000 KB の場合，性能低下はほとんどない．また，提案方式を導入した場合において，同時に走行する VM 数の増加による Web サーバの性能低下の度合いは，十分小さいといえる．同時に走行する VM 数が変化した場合においても，性能低下の度合いはほぼ同じである．このため，提案方式は，複数 VM が同時に走行する環境においても，十分有用であるといえる．

表 4　複数 VM を動作させた場合の Web サーバのスループット（要求数／秒）

ファイルサイズ	VMM	VM 数					
		2	4	6	8	10	12
1 KB	Xen	1329.27	1295.61	1225.22	1171.51	1231.72	1172.15
	提案方式	1150.54	1057.95	1017.53	987.24	1015.69	946.61
	相対性能	0.87	0.82	0.83	0.84	0.82	0.81
10 KB	Xen	658.15	639.76	627.9	628.56	609.45	615.64
	提案方式	626.12	612.93	559.02	582.24	578.89	589.58
	相対性能	0.95	0.95	0.89	0.93	0.95	0.96
1,000 KB	Xen	11.41	11.4	11.39	11.38	11.39	11.39
	提案方式	11.41	11.4	11.39	11.37	11.39	11.06
	相対性能	1	1	1	1.00	1	0.97

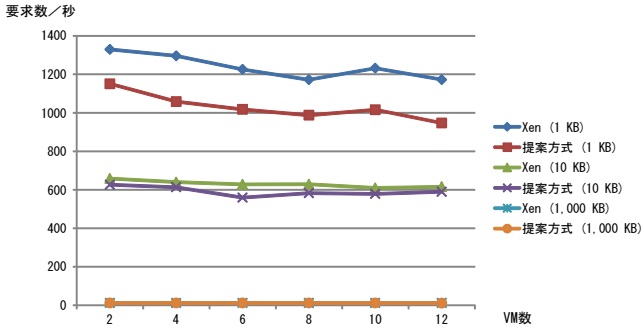

図 5　複数 VM を動作させた場合の Web サーバの性能

7. 関連研究

カーネルレベルマルウェアの挿入を防止する手法として，走行中のカーネルの完全性を検証する手法 [18] がある．この手法は，利用者が許可したコードのみをカーネル空間で実行可能にし，不正なコードの実行を防止する．しかし，利用者がどのコードの実行を許可するかの判断が難しい．syslog デーモンへの攻撃によるログの改ざんを防止する手法として，syslog デーモンの完全性を検証する手法 [7] がある．この手法は，トラステッドブートと Late Launch により syslog デーモンの完全性を検証し，ログをリモート計算機へ転送する．また，ログファイルの改ざんを防止する手法 [8] がある．この手法は，ファイルシステム内に複数のバックアップを作成することで，改ざんされたファイルを復元できる．

これらの手法は，ログの送信元の AP を直接攻撃された場合には，ログの改ざんを防止できない．文献 [16] の手法では，VM によるメモリアクセスを VMM が制御できる点に着目し，VM 上の特定の AP が利用するメモリ領域とその他の AP やカーネルが利用するメモリ領域を分割する．特定の AP が利用するメモリ領域は，カーネルを含む他のプログラムによる読み書きが禁止されている．ただし，ログは，AP から送信された後，カーネルや syslog デーモンを経由してファイルへ書き出されるため，ログの改ざんを完全に防止するためには，ログの保存経路のすべてにおいて，改ざんへ対処する必要がある．一方，提案方式を用いた場合，できるだけ早期に VM 外部へログを転送できる．このため，改ざんされる危険性を低減できる．また，提案方式は，より単純な処理で VM 外部へログを転送できるため，バグの混入が起こりにくく，適用が容易である．

8. おわりに

ライブラリの置き換えにより VM 外部へ安全にログを転送する方式の評価を述べた．評価では，マルウェアによるログの改ざん実験を行い，提案方式により，多くの攻撃に対処できることを示した．また，多種の OS への適用の容易さの評価では，Linux と FreeBSD に，ライブラリへ 20 行のコードを追加するだけで対応できることを示した．性能評価では，Web サーバの性能への影響を測定した．測定結果より，最悪の場合に 9%程度の性能低下が起きることを示した．また，複数 VM が同時に走行する場合において，Web サーバの性能を測定し，同時に走行する VM 数が増加しても，提案方式の導入による性能低下の度合いは一定であり，十分有用であることを示した．

参考文献

[1] Kent, K., Souppaya, M.: Guide to Computer Security Log Management, Special Publication 800-92 (2006).
[2] A new adore root kit (online), available from 〈http://lwn.net/Articles/75990/〉 (accessed 2013-06-17).
[3] Analysis of a Rootkit: Tuxkit (online), available from 〈http://www.ossec.net/doc/rootcheck/analysis-tuxkit.html〉 (accessed 2013-06-17).
[4] Symantec: Backdoor.lastdoor (online), available from 〈http://www.symantec.com/security_response/writeup.jsp?docid=2002-090517-3251-99〉 (accessed 2013-06-17).
[5] Zhao, S., Chen, K. and Zheng, W.: Secure Logging for Auditable File System Using Separate Virtual Machines, *Proc. 2009 IEEE International Symposium on Parallel and Distributed Processing with Applications*, pp.153–160 (2009).
[6] Ashino, Y. and Sasaki, R.: Proposal of Digital Forensic System Using Security Device and Hysteresis Signature, *Proc. 3rd International Conference on International Information Hiding and Multimedia Signal Pro-*

cessing (*IIH-MSP 2007*), Vol.2, pp.3–7 (2007).

[7] Boeck, B., Huemer, D., Tjoa, A.M.: Towards More Trustable Log Files For Digital Forensics by Means of "Trusted Computing," *24th IEEE International Conference on Advanced Information Networking and Applications* (*AINA*), pp.1020–1027 (2010).

[8] Takada, T., Koike, H.: NIGELOG: Protecting Logging Information by Hiding Multiple Backups in Directories, *Proc. 10th International Workshop on Database and Expert Systems Applications*, pp.874–878 (1999).

[9] Sato, M. and Yamauchi, T.: VMM-Based Log-Tampering and Loss Detection Scheme, *Journal of Internet Technology*, Vol.13, No.4, pp.655–666 (2012).

[10] Subashini, S., Kavitha, V.: A Survey on Security Issues in Service Delivery Models of Cloud Computing, *Journal of Network and Computer Applications*, Vol.34, No.1, (2011).

[11] Grobauer, B., Walloschek, T., Stocker, E.: Understanding Cloud Computing Vulnerabilities, *2011 IEEE Symposium on Security and Privacy*, Vol.9, No.2, pp.50–57 (2011).

[12] Marty, R.: Cloud Application Logging for Forensics, *Proc. 2011 ACM Symposium on Applied Computing* (*SAC '11*), pp.178–184 (2011).

[13] Birk, D., Wegener, C.: Technical Issues of Forensic Investigations in Cloud Computing Environments, *Proc. 2011 IEEE Sixth International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE)*, pp.1–10 (2011).

[14] 佐藤将也，山内利宏：ライブラリの置き換えによる VM 外部への安全なログ転送方式の提案，情報処理学会研究報告，Vol.2012-CSEC-59, No.6, pp.1–8 (2012).

[15] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the Art of Virtualization, *Proc. 19th ACM Symposium on Operating Systems Principles*, Vol.37, No.5, pp.164–177 (2003).

[16] Dewan, P., Durham, D., Khosravi, H., Long, M., Nagabhushan, G.: A Hypervisor-Based System for Protecting Software Runtime Memory and Persistent Storage, *Proc. 2008 Spring Simulation Multiconference* (*SpringSim '08*), pp.828–835 (2008).

[17] McCune, J.M., Yanlin L., Ning Q., Zongwei Z., Datta, A., Gligor, V., Perrig, A.: TrustVisor: Efficient TCB Reduction and Attestation, *Proc. 2010 IEEE Symposium on Security and Privacy*, pp.143–158 (2010).

[18] Seshadri, A., Luk, M., Qu, N. and Perrig, A.: SecVisor: A Tiny Hypervisor to Provide Lifetime Kernel Code Integrity for Commodity OSes, *Proc. 21st ACM SIGOPS Symposium on Operating Systems Principles*, pp.335–350 (2007).

[19] Apvrille, A., Gordon, D., Hallyn, S., Pourzandi, M. and Roy, V.: DigSig: Runtime Authentication of Binaries at Kernel Level, *Proc. 18th USENIX Conference on System Administration*, pp.59–66 (2004).

[20] Gene H.K. and Eugene H.S.: The Design and Implementation of Tripwire: A File System Integrity Checker, *Proc. 2nd ACM Conference on Computer and communications security* (*CCS '94*), pp.18–29 (1994).