2A-01

KVMを利用した機密情報の拡散追跡機能における高速化の評価

森山 英明† 山内 利宏‡ 佐藤 将也‡ 谷口 秀夫‡

†有明工業高等専門学校 創造工学科 ‡岡山大学 大学院自然科学研究科

1. はじめに

計算機内で管理されている機密情報は，外部に漏えいすることで，企業や個人にとって大きな損失となる．機密情報を保持したファイルへの誤操作や，外部からの機密情報へのアクセスを検知するために，仮想計算機モニタ(VMM)を利用した機密情報の拡散追跡機能が提案され，実現されている．この機能では，機密情報を操作するシステムコールをフックして情報を取得することで，機密情報の拡散経路を利用者に通知することを可能とする．一方，システムコール発行時に検知処理が発生するため，システムの応答性に関する影響が問題となる．このため，これまでに高速化の手法を提案している．

本稿では，KVM 上に実現されている機密情報の拡散追跡機能について，高速化を適用した際の詳細な評価について述べる．

2. KVMにおける機密情報の拡散追跡機能

計算機内の機密情報の利用状況を把握するために，仮想計算機モニタ(VMM: Virtual Machine Monitor)における機密情報の拡散追跡機能を提案している．また，KVM(Kernel-based Virtual Machine) 上に機能を実現し，評価結果を報告した[1]．機密情報の拡散追跡機能では，機密情報を有する可能性のあるファイルとプロセス(以降，管理対象ファイルと管理対象プロセス)を拡散情報として記録し，管理する．この機能を VMM 上に実装することにより，オペレーティングシステム(OS)よりも攻撃が困難である VMM で機密情報を管理できる．また，ゲスト OS のソースコードを改変することなく VMM の改変のみで機能を提供できる利点がある．

KVM における機密情報の拡散追跡機能では，仮想計算機上で発行されるシステムコールをフックするために，ハードウェアブレークポイントを用いてシステムコール発行前 (SYSCALL 命令) と終了直前 (SYSRET 命令) でデバッグ例外を発生させる．これにより，処理をゲスト OS から VMM へ移行させる．VMM 側では，デバッグアドレスレジスタによるデバッグ例外であることを確認し，SYSCALL 命令と SYSRET 命令のどちらの実行前に発生したものか確認する．次に，システムコール番号から，機密情報の拡散に関係するシステムコールであるか否かを判定する．機密情報の拡散に関係するシステムコールの場合は，SYSCALL 命令実行前の拡散追跡処理として，プロセスが発行したシステムコール番号，ページテーブル情報，扱うファイルのファイルディスクリプタの値などを取得する．SYSRET 命令実行前の拡散追跡処理では，システムコール処理の成否，システムコールを発行したプロセスが扱うファイルの情報などを取得する．もし，機密情報の拡散に関係しないシステムコールの場合は，拡散追跡にともなう処理を行わず，システムコール処理を続行する．

機密情報の拡散追跡処理では，仮想計算機上で発行されるすべてのシステムコールに対してフックを行う．このため，機密情報の拡散追跡機能による処理時間がオーバヘッドとなり，オーバヘッドの削減が課題となる．オーバヘッドを削減するため，処理の分析，機能の改良，および簡単な評価を行った[2]．処理の詳細な分析より，管理対象プロセスと管理対象ファイルの表示処理によるオーバヘッドが大きいことが明らかになったため，これらの処理に対して改良を行った．以降では，改良した機密情報の拡散追跡機能を導入した環境における評価結果を述べ，高速化の効果を明らかにする．

3. 評価

3.1 評価の概要

本評価の測定環境を表 1 に示す．本評価では，以下の 3 つの環境でそれぞれ測定し，結果を比較することで，機密情報の拡散追跡機能導入時のオーバヘッドと高速化の効果を明確化する．

(A) 拡散追跡機能を導入していない環境
(B) 改良前の拡散追跡機能を導入した環境
(C) 改良後の拡散追跡機能を導入した環境

また，機密情報の拡散追跡機能のオーバヘッドを測定し，高速化の効果を明確化するために，以下の 2 つの処理における実行時間を測定する．

(1) システムコールを用いた測定:
1 つのファイルに対して，read システムコー

Evaluation of Improved Function for Tracing Diffusion of Classified Information on KVM
† National Institute of Technology, Ariake College
‡ Graduate School of Natural Science and Technology, Okayama University

表 1 測定環境

測定用計算機	
CPU	Intex Xeon Processor E5-2609
コア数	8個
メモリ	64GB
OS	Fedora18 (Linux Kernel 3.6.10)(64bit)
VMM	KVM-kmod-3.6
仮想計算機	
コア数(vCPU)	1個
メモリ	1GB
OS	Fedora18 (Linux Kernel 3.6.10)(64bit)

ルによりデータを取得し，取得したデータを write システムコールで別のファイルに書き込みを行う．この処理を，管理対象ファイルと管理対象ではないファイルに対してそれぞれ行い，処理時間を測定して比較する．

(2) マイクロベンチマークを用いた測定:

マイクロベンチマークとして LMbench を利用する．LMbench は，計算機の基本性能を測定するベンチマークツールである．VMM における拡散追跡機能が仮想計算機の性能に及ぼす影響を評価するために，仮想計算機上で動作させ，基本的な OS 機能のレイテンシを測定する．

3.2 システムコール実行時間

システムコールの処理時間を測定した結果を表 2 に示す．改良前の拡散追跡機能を導入した環境において，管理対象ファイルを対象とした場合の処理時間は，対象としない場合の処理時間と比較すると，read システムコールで 72.8 μs，write システムコールで 23.0μs 増加した．これは，管理対象ファイルやプロセスの情報を取得し，ログとして出力する処理によるものである．次に，改良前と改良後の環境における処理時間を比較すると，管理対象ファイルを対象としたシステムコールの処理において，改良後の環境では read システムコールで 83.0μs，write システムコールで 1.60μs 処理時間が減少した．このことから，改良による高速化の効果を確認することができ，特に管理対象ファイルを対象とした read システムコールの処理で効果が高いことがわかる．

3.3 マイクロベンチマーク実行時間

LMbench により，OS 機能のレイテンシを測定した結果を表 3 に示す．表 3 において，exec proc で約 20μs，sh proc で約 70μs の処理時間の減少を確認できる．また，fork proc では，約 30μs の処理時間の増加を確認できる．これらの差異は，各処理の全体の時間と比較すると小さい．この結果より，LMbench の各処理は管理対象ファイルを取り扱わないため，改良による影響が小さいと考えられる．

表 3 LMbench を用いた測定結果

	改良前の機密情報の拡散追跡機能(μs)	改良後の機密情報の拡散追跡機能(μs)
null call	7.47	7.50
null I/O	19.10	11.00
stat	8.61	8.68
open clos	17.50	17.50
slct TCP	11.40	11.40
sig inst	7.64	7.66
si hndl	11.70	11.80
fork proc	914.00	946.00
exec proc	2810.00	2791.00
sh proc	7407.00	7332.00

4. おわりに

KVM における機密情報の拡散追跡機能に関して，ボトルネックとなる箇所に改良を加え，改良の前後で測定を行い，高速化の効果を明確化した．今後は，複数の管理対象ファイルとプロセスを扱う際の評価を行う．

謝辞 実験に協力していただいた有明工業高等専門学校の荒木涼氏，梅村俊英氏，古賀友渉氏，田中祐太氏に感謝します．本研究の一部は JSPS 科研費 16H02829（基盤研究(B)）の助成を受けたものです．

参考文献

[1] Fujii, S., Sato, M., Yamauchi, T., and Taniguchi, H.: Evaluation and Design of Function for Tracing Diffusion of Classified Information for File Operations with KVM, The Journal of Supercomputing, Vol.72, Issue 5, pp.1841-1861 (2016).

[2] Moriyama, H., Yamauchi, T., Sato, M., and Taniguchi, H.: Performance Improvement and Evaluation of Function for Tracing Diffusion of Classified Information on KVM, 4th International Workshop on Information and Communication Security (WICS 2017), pp.463-468, (11, 2017).

表 2 システムコールの処理時間(μs)

拡散追跡機能を導入していない環境		拡散追跡機能を導入した環境							
		改良前				改良後			
		管理対象ファイルを対象としない処理		管理対象ファイルを対象とした処理		管理対象ファイルを対象としない処理		管理対象ファイルを対象とした処理	
read(μs)	write(μs)	read(μs)	write(μs)	read(μs)	write(μs)	read(μs)	write(μs)	read(μs)	write(μs)
218.60	88.60	473.80	160.60	546.60	183.60	460.60	149.00	463.60	182.00