Vol.2010-CSEC-51 No.5
2010/12/10

擬似シンクライアント端末のアップデートにおけるプロセス単位書込み制御方式の提案

中山晃治† 桐畑康裕† 鮫島吉喜†

シンクライアントシステムを低コストに構築するため，既存 PC のディスク書込みをソフトウェアで禁止した擬似シンクライアント端末が考案されている．擬似シンクライアント端末は，ソフトウェアのアップデートの際，ディスクへの書込みを許可したモードで端末を再起動する必要があり，利便性が悪い問題がある．この問題を解決するため，アップデートプロセスのみ，再起動なしでディスク書込みを可能にする，プロセス単位書込み制御方式を提案する．本方式を適用し，OS のパッチとウィルスパターンファイルのアップデートに関する利便性とセキュリティを評価した結果，安全で利便性の良い擬似シンクライアント端末を実現できることが確認できた．

A Process-based Write Control for Software Update on the Pseudo Thin Client PC

Koji Nakayama† Yasuhiro Kirihata† Yoshiki Sameshima†

To reduce the initial cost of a thin client system, a pseudo thin client terminal is developed in which a write control mechanism prohibits writing the data to the local disk. The terminal has a usability problem, because in the software update a reboot process is required to switch the mode which allows for writing the data to the local disk. To address this issue, we propose a process-based write control method in which only the update processes are allowed to write to the local disk without reboot process. We evaluate the security and usability of the client terminal with respect to the updating of OS patches and virus pattern files. This evaluation shows that this method improves the usability of the software update with keeping the security of the client terminal.

1. はじめに

2005 年より施行の個人情報保護法や，相次ぐセキュリティインシデントの影響により，シンクライアントシステムが注目されている[1]．これは，たとえ，暗号化してあるデータであっても，紛失すると企業イメージがダウンすることと，紛失時の対応にかかるコストを低減したいという企業のニーズに，紛失事故自体を防止するシンクライアントシステムがマッチしたことによる[2][3]．一方で，大学などの教育機関においても，運用管理コストを抑える目的でシンクライアントシステムが導入されており，多くの分野でシンクライアント市場は拡大することが予想されている[4][5]．こうした背景の中，より低コストにシンクライアントシステムを実現するため，二次記憶装置を持つ既存の端末を，ソフトウェアで擬似的にディスクレス化したクライアント端末，いわゆる擬似シンクライアント端末が考案されている[6][7][8]．擬似シンクライアント端末では，二次記憶装置上のボリュームに対するフィルタドライバを組込み，二次記憶装置への書込みを制御してメモリにキャッシュする．この書込み制御により，シャットダウン後にデータが揮発し，端末上にデータが残らなくなる．擬似シンクライアント端末を利用したシンクライアントシステムでは，新たな端末の購入コストを削減できることに加え，あらかじめ端末にアプリケーションをインストールして利用することで，アプリケーション実行にかかるサーバリソースや，画面転送にかかるネットワークの増強が不要となり，低コストにシステムを構築・運用することが可能となる．

しかし，擬似シンクライアント端末は，ローカル端末上に OS やアプリケーションがインストールされているため，ソフトウェアのアップデートが必要となる．特に，Windows では，Microsoft Update は 1 回／月，ウィルス対策ソフトパターンファイルはほぼ毎日更新される．アップデートを行うためには，一時的に書込み制御機能をオフにして，ディスク書込みを行う必要があるが，ボリュームフィルタを利用している場合，ボリュームフィルタより上位のファイルシステムを制御できないため，アップデートごとに再起動が必要であり，利便性が著しく悪くなる問題がある．

本論文では，この問題を解決するため，アップデート処理に再起動が不要なプロセス単位書込み制御方式を提案する．プロセス単位書込み制御方式は，ファイルシステムとボリューム，及び，レジストリのフィルタドライバで，プロセスごとの書込みを制御し，アップデートに関する書込みのみを二次記憶装置へ保存する方式である．この方式により，ユーザによる書込みを禁止しつつ，OS のパッチやウィルス対策ソフ

† 株式会社日立ソリューションズ
Hitachi Solutions, Ltd.
商品名称等に関する表示
本論文に記載されている会社名，製品名は各社の登録商標もしくは商標です．

トパターンファイル等の頻繁に発生するアップデートごとに再起動を行う必要が無くなり，利便性が向上する．以下，擬似シンクライアントの概要を説明し，利便性を向上させる上での課題を明確にする．次に，プロセス単位書込み制御方式を提案し，利便性の面とセキュリティ面での評価を行う．

2. 擬似シンクライアントシステムの概要と問題

2.1 擬似シンクライアントシステムの概要

図 1 に示す擬似シンクライアントシステムは，ファイルサーバと擬似シンクライアント端末とそれらをつなぐネットワークで構成される．擬似シンクライアント端末は，書込み制御ドライバによるローカルへのファイル保存禁止機能，および，持出し制御ドライバによる USB 等の外部記憶媒体への書き出し禁止機能を持つ．

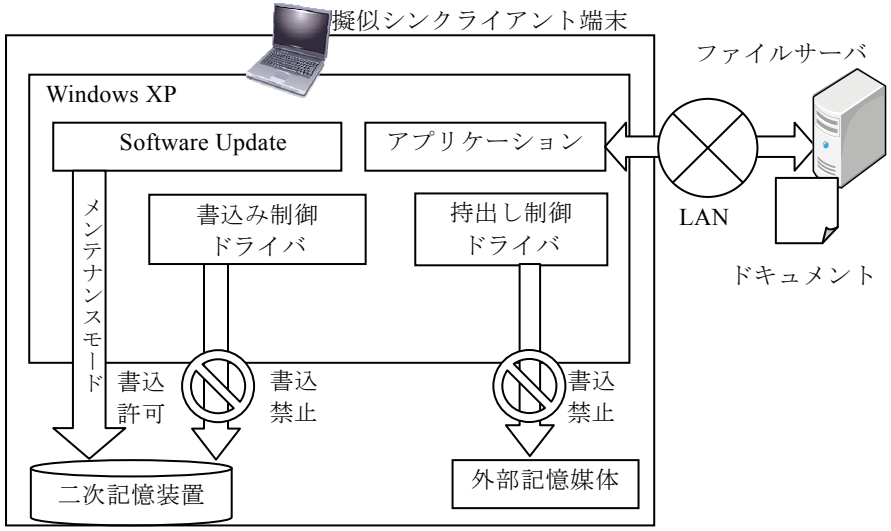

図 1　擬似シンクライアントシステムの構成図

この擬似シンクライアント端末は，メンテナンスモードと書込禁止モードの二つのモードを持つ．メンテナンスモードは，OS のパッチやウィルス対策ソフトのパターンファイル更新やアプリケーションのインストール等，二次記憶装置への書込みが必要な場合に実行されるモードである．書込み禁止モードは，ユーザによるローカル書込みを禁止したモードである．この構成により，ユーザが擬似シンクライアント端末上で作成したドキュメントは，ローカル上に保存できなくなるため，結果としてファイルサーバに集約されることになる．

2.2 擬似シンクライアント化のための書込み制御技術

図 2 は，擬似シンクライアント端末における書込み制御ドライバの構成を示したものである．擬似シンクライアント化は，システムボリュームをフィルタする「書込み禁止ボリュームフィルタ」により実現する．システムボリュームとは，OS がインストールされているボリュームである．書込み禁止ボリュームフィルタは，システムボリュームにアタッチし，OS 上で動作する全てのプロセスから発行されるシステムボリュームへの I/O をフックする．書込み禁止ボリュームフィルタは，ファイルシステムからの書込みデータをメモリ上にキャッシュし，二次記憶装置の書込み先セクタ情報を保持しておく．ファイルシステムから読込み要求があった場合は，書込み時に取得したセクタ情報を元にキャッシュの有無を判断し，キャッシュがない場合にはシステムボリューム，キャッシュがある場合はメモリからデータを読込んでファイルシステムにデータを返す．これにより，ファイルシステムより上位からのシステムボリュームへの書込みを全てメモリ上にリダイレクトすることが可能となり，擬似的にシンクライアント端末を実現することが可能となる．

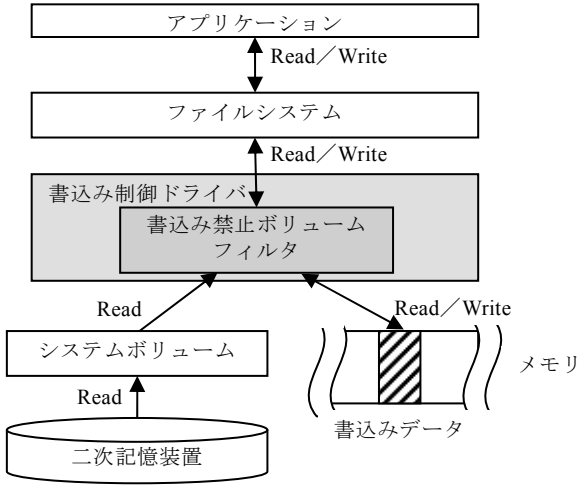

図 2　書込み制御ドライバの構成

2.3 擬似シンクライアント端末におけるアップデートの問題

擬似シンクライアント端末では，OS のパッチやウィルス対策ソフトパターンファイル等のアップデート時にメンテナンスモードで端末を再起動しなければならないことによる，利便性の悪さが問題である．例えば，Microsoft Update のサービスパックやソフトウェアのバージョンアップ等は，発生頻度が低いため，それ程利便性は損なわれないが，月例の Microsoft Update やウィルス対策ソフトパターンファイル更新は，発生頻度が高く，アップデートごとにメンテナンスモードで再起動を行うと，著しく利便性が悪くなる．従って，擬似シンクライアント端末では，それら発生頻度が高いアップデートへの対策が求められている．

再起動が必要な理由は，OS の起動中にメンテナンスモードと書込み禁止モードを自由に切り替えることができないという制限事項に起因する．モード切替えには，ファイルシステムがキャッシュしているデータを二次記憶装置にフラッシュし，データの整合性を保つ必要がある．これには，書込み制御ドライバ以外の OS のモジュールやアプリケーション等がオープンしている全てのファイルをクローズしてファイルへの書込みを停止させ，ファイルシステムがキャッシュしているファイルのデータを二次記憶装置に一度フラッシュしなければならない．しかし，書込み制御ドライバ以外の OS のモジュールやアプリケーションがオープンしているファイルを，書込み制御ドライバがクローズすると，それらの動作が不安定になり，最悪の場合，システムがクラッシュする．従って，現状ではアップデートのためにメンテナンスモードでの再起動が必須である．

更に，擬似シンクライアント端末は，擬似的にディスクレス状態を実現することでシンクライアント端末の代わりを果たすセキュリティ製品であるため，悪意のあるユーザによるローカル保存を禁止し，アップデートに関する書込みのみを適用するというセキュリティ要件を満たす必要がある．以上により，擬似シンクライアント端末の利便性を向上させるためには，書込み禁止モードのまま，モード切替え無しに月例の Microsoft Update や毎日のウィルス対策ソフトパターンファイル等の頻繁に発生する更新データのみを書込む技術の実現が求められている．

3. プロセス単位書込み制御方式の実現方法

アップデートの問題を解決するために，ファイルシステムとボリューム，及び，レジストリのフィルタドライバを組み合わせて，プロセスごとの書込みを制御するプロセス単位書込み制御方式を提案する．

3.1 プロセス単位書込み制御方式のドライバ構成

プロセス単位書込み制御方式では，ファイル I/O 制御ドライバとレジストリ I/O 制御ドライバで，I/O の発行元のプロセス判定を行い，書込み制御ドライバで，OS のパッチやウィルス対策ソフトパターンファイル更新に関わる，書込みを許可されたプロセスが発行する I/O のみのキャッシュデータを作成する．その後，端末のシャットダウン時に，そのキャッシュデータを二次記憶装置へ書込む．これにより，メンテナンスモードを使用することなく，OS のパッチやウィルス対策ソフトパターンファイル更新が可能となる．図 3 はプロセス単位書込み制御方式のドライバ構成である．

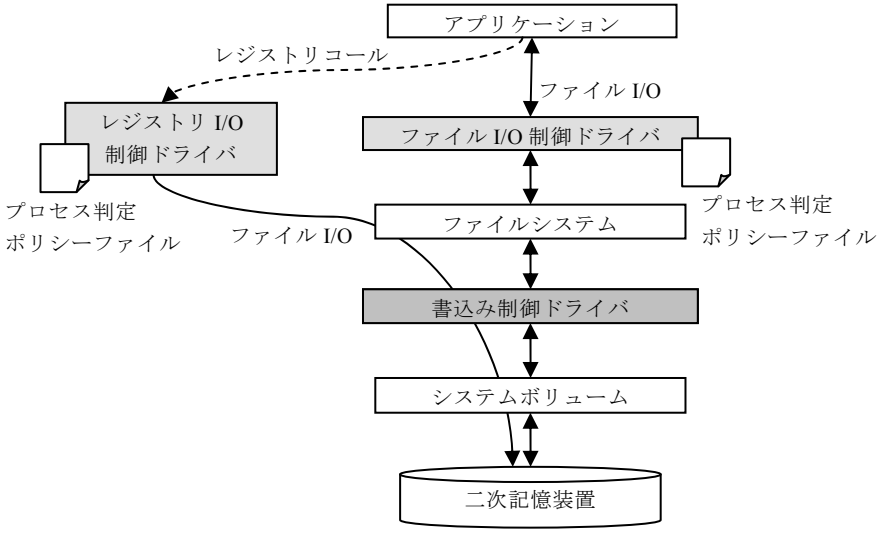

図 3　プロセス単位書込み制御方式のドライバ構成

3.2 書込み制御ドライバ

図 4 はファイルシステムより下位のモジュール構成図である．書込み制御ドライバ内には，システムボリュームへの I/O をフィルタする「書込み禁止ボリュームフィルタ」と「書込み許可ボリュームフィルタ」の二つの異なるボリュームフィルタが存在する．書込み禁止ボリュームフィルタは，システムボリュームにアタッチし，OS 上で動作する全てのプロセスから発行されるシステムボリュームへの I/O をフックする．書込み許可ボリュームフィルタは，後述するファイル I/O 制御ドライバで二重化（以後，ミラーリングと呼ぶ）された書込み許可プロセスの書込みデータをキャッシュするためのボリュームであり，シャットダウン時に二次記憶装置へ書込みを行う．両ボリュームフィルタは共に，ファイルシステムからの書込みデータをメモリ上にキャッシュし，書込み先のセクタ情報を保持する．ファイルシステムから読込み要求があっ

た場合は，書込み時に取得したセクタ情報を元にキャッシュの有無を判断し，メモリにキャッシュがない場合にはシステムボリューム，メモリにキャッシュがある場合はメモリからデータを読込んでファイルシステムにデータを返す．OS を起動した時点では，書込み禁止ボリュームフィルタと書込み許可ボリュームフィルタは，システムボリュームの同じ MFT(Master File Table)領域を読込むためファイル・フォルダ構成は同じである．MFT とは，ボリューム内のファイル・フォルダの構成情報を保持している関連データベースである[9][10].

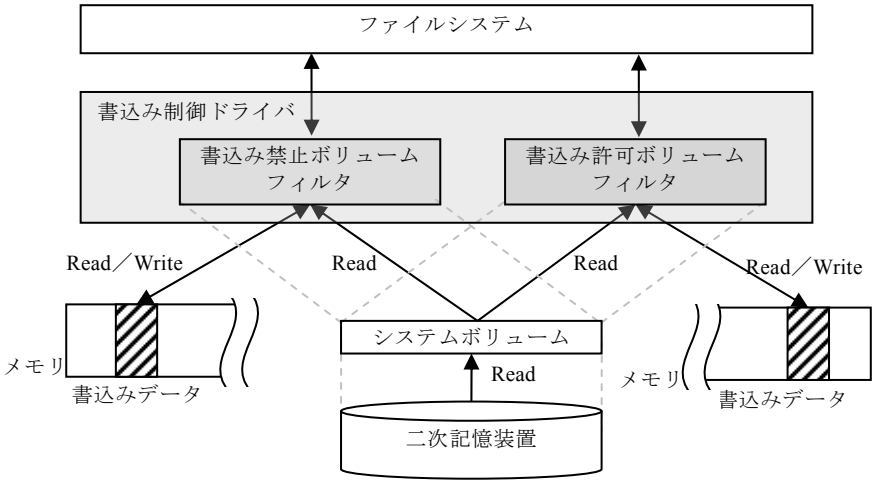

図 4　ファイルシステム下位のモジュール構成

3.3 ファイルI/O制御ドライバ

ファイル I/O 制御ドライバは，ファイル I/O の発行元プロセス名からアップデートに関わるプロセスか否かを判定し，書込み制御ドライバ内の書込み禁止ボリュームと，書込み許可ボリュームに I/O を振り分ける機能を持つファイルシステムのフィルタドライバである．ファイル I/O 制御ドライバの振り分けにより，書込み許可ボリュームは書込み許可プロセスによる書込みデータのみを受け取ることができるようになる．振り分け機能の動作は，書込み許可プロセスからの書込みは I/O をミラーリングし，書込み許可ボリュームと書込み禁止ボリュームの両方へ書込む．書込み禁止プロセスからの書込みは書込み禁止ボリュームのみに書込みを行う．両方のボリュームに書込みを行っている理由は，アプリケーションからの読込み要求に対するデータは，書込み禁止フィルタが二次記憶装置から読込んだデータ，または，メモリにキャッシュしたデータが返されるためである．アプリケーションの読込み要求に対して書込み禁止ボリュームに送られる理由は，書込み禁止ボリュームがシステムボリュームにアタッチしているため，ファイルシステムより上位レイヤでは，書込み禁止ボリュームがシステムボリュームに見えるためである．

3.4 レジストリI/O制御ドライバ

レジストリ I/O 制御ドライバは，OS 起動からシャットダウンまでにコールされた書込み許可プロセスがコールするレジストリ API を全てレジストリ API コールキャッシュとしてキャッシュしておき，シャットダウン時に一括して書込み許可ボリューム上のレジストリハイブファイルへ書込むドライバである．レジストリハイブファイルとは，レジストリデータが格納されるデータベースファイルである．レジストリハイブファイルは決まったレジストリキーに対応付けられており，アプリケーションはこのキーを引数に，レジストリ API をコールすることで，レジストリを読み書きすることが可能になる．Windows カーネルのモジュールである構成マネージャは，レジストリ API がコールされると，レジストリハイブファイルに対してファイル I/O を発行する．本システムにおいて，シャットダウン後に二次記憶装置に保存されるのは書込み許可ボリューム上のレジストリハイブファイルである．従って，書込み禁止ボリューム上のファイルに関連付けられているキー名とは別のキーを新たに作成し，書込み許可ボリューム上のレジストリハイブファイルと関連付けを行う．その後，シャットダウン直前に，キャッシュしておいたレジストリ API コールキャッシュを元に，新しく作成したレジストリキーにレジストリ API コールを行うことで，書込み許可ボリューム上のレジストリハイブファイルに書込みを行い，書込み許可ボリュームがキャッシュしたデータを，システムボリュームに書込むことでレジストリ書込みが完了する．

3.5 プロセス判定ポリシーファイル

プロセス判定ポリシーファイルは，プロセス判定における書込みポリシーを記述したファイルであり，ファイル I/O 制御ドライバとレジストリ I/O 制御ドライバが使用する．以下，表 1 に，Microsoft Update とウィルス対策ソフトのパターンファイル更新データのみを書込むための書込みポリシーを示す．

表 1　プロセス判定ポリシー

項番	プロセス	書込み元プロセス	子プロセス
1	システム関連	○	×
2	Microsoft Update， ウィルス対策ソフト関連	○	○
3	Microsoft Office，Explorer，その他	×	×

○：書込み許可　×：書込み禁止

4. 評価

4.1 利便性に関する評価

プロセス単位書込み制御方式により，モード切替えによる再起動を行わず，アップデートが成功することを確認するため，月例の Microsoft Update とウィルス対策ソフトパターンファイルのアップデートを検証する．また，アップデートと同時に，Microsoft Office で作成したドキュメントをローカル保存して，再起動後にローカルから消去されているかを検証し，不要な書込みが発生していないことを確認する．アップデートの検証は，擬似シンクライアントの書込み制御を適用せずにアップデートを行ったローカルファイル毎のハッシュ値と，プロセス判定方式を適用した擬似シンクライアントを使用してアップデートを行ったローカルファイル毎のハッシュ値を比較することで判定する．ハッシュ値を比較するために，仮想マシンを使用してパッチ状態が同じ OS イメージを二つ用意し，ひとつにプロセス単位書込み制御方式のドライバをインストール, もう片方は何もインストールしない状態の仮想マシンを作成する．その後，各々の仮想マシンで Microsoft Update とウィルス対策ソフトパターンファイル更新を行い，再起動後にファイルハッシュ値の比較を行う．表 2 に評価に使用したテスト環境を示す．

表 2　テスト環境

項番	項目	内容
1	OS	Windows XP
2	メモリ/二次記憶装置	4GB/80GB
3	インストール済み アプリケーション	Microsoft Office 2003 TRENDMICRO ウィルスバスター 7.3

ハッシュ値比較を利用してアップデート状態を評価した結果，月例の Microsoft Update とウィルスバスターのパターンファイル更新で，システムファイルとウィルス対策ソフトパターンファイルのファイルハッシュ値が一致した．システムファイル以外のものでは，ハッシュ値が異なるファイルが存在したが，これに関しては，ファイル内に時間やトランザクションデータ等のタイミングで異なるデータを持つファイルである．システムファイルとウィルス対策ソフトパターンファイルのファイルハッシュ値が一致したことから，月例の Microsoft Update と，毎日のウィルスバスターのパターンファイル更新はメンテナンスモードに切替えることなくアップデート可能であることが確認できた．また，Microsoft Office で作成・保存したドキュメントを確認した結果，再起動後にローカル上から消去されていた．従って，書込み禁止モード時において, Microsoft Update やウィルスバスター等の書込み許可プロセスからの書込みは可能であるが，Microsoft Office 等の書込み禁止プロセスからの書込みはできないことが確認できた．以上により，頻繁にメンテナンスモードでアップデートする必要がなくなり，アップデート時の利便性が向上したといえる．

4.2 セキュリティに関する評価

擬似シンクライアント端末は，セキュリティ製品であるため，悪意のあるユーザが意図的に端末内へデータを保存する場合の対策が必要である．この点に関して，Microsoft 社の STRIDE 脅威モデルに基づき，なりすまし（S : Spoofing），改竄（T : Tampering），否認（R : Repudiation），情報漏洩（I : Information disclosure），Dos 攻撃（D : Denial of service），特権昇格（E : Elevation of privilege）の観点で評価を行った結果，四つの脅威が確認された[11]．各脅威への対策を以下に示す．

(1) 書込み許可プロセスへのなりすまし

ユーザが, プロセス判定ポリシーファイルを盗み見て書込み許可プロセスを特定し，書込み許可プロセスの実行ファイルを悪意の実行ファイルに置き換えて起動させる可能性がある．この場合，悪意のある書込み許可プロセスが起動され，ローカルファイルへの書込み・作成・保存，場合によってはプロセス判定ポリシーファイル自体を改竄される．そのため，不正なプログラムの起動を監視・制限する機能が必要である．例えば，実行ファイルのハッシュ値を予め保存しておき，プロセス起動時に使用される実行ファイルのハッシュ値と保存していたハッシュ値を比較し，不正な実行ファイルか否かをチェックする方法で可能である．同様に，プロセス判定ポリシーファイル自体の改竄に対しても，端末の起動時にファイルのハッシュ値を保存しておき，ファイルの読込み時にハッシュ値を比較して，改竄を検出することで対策可能である．

(2) 書込み許可フィルタへのダイレクト I/O

ユーザがドライバへ直接書込み命令を実行するアプリケーションを作成してデータ書込みを行う可能性が考えられる．不要アプリケーションの動作を監視・制御するサービスプログラムを動作させて監視することで対応できる．

(3) ドライバのアンインストール

ユーザが意図的に書込み制御ドライバをアンインストールすると擬似シンクライアント化機能が無効化され，情報を端末に保存して持ち出すことが可能となってしまう．この問題の対策として，アカウント権限を使った制御により，対策可能である．書込み制御ドライバは管理者が管理者アカウントでインストールを行い，端末を利用するユーザには，インストール・アンインストール操作が禁止されたアカウントをユーザに利用させることで対策可能である．

(4) **インストールプロセスの利用**

msiexec プロセスは，ユーザがアプリケーションをインストールする場合にも，Microsoft Update 時にも起動するプロセスである．そのため，msiexec プロセスを書込み許可にすると，ユーザによるアプリケーションインストールを許可することになってしまう．ユーザによるアプリケーションインストールを禁止し，アップデートだけを成功させるための対策として，explorer プロセス下で起動する msiexe.exe ファイルのオープン要求を禁止し，services プロセス下で起動する msiexec プロセスの要求は許可とすることで実現することが可能である．

5. 関連研究

ある時点でのボリュームの状態をミラーリングし，二つのボリュームを利用する方法は，一般に，ボリュームのスナップショットとして知られており，Windows では，ボリュームシャドウコピーサービスとして提供されている[12]．また，Linux では，スナップショット機能を実装した ext3 や ZFS 等のファイルシステムが存在するが，これらは，いずれも，システムやデータ復旧用のファイルバックアップに利用されている[13][14]．しかし，これらの技術を利用しただけでは，アップデートの書込みとそれ以外の書込みを振分ける機能は実現できない．振分けにはプロセス判定が必要であるが，ファイルシステムより上位でなければプロセス判定はできないため，アップデートを行うには書込み許可モードでの再起動が必要である．プロセス単位書込み制御方式は，起動時に書込み許可ボリュームフィルタを作成し，システムボリュームをミラーリングしている点では，前述のスナップショットと同じ技術を利用しているが，上位のファイルシステムフィルタドライバで，プロセス判定を行い，プロセスごとの I/O を振分ける技術を新たに追加したことで，再起動不要でかつアップデートに関する書込みのみをディスクに適用する擬似シンクライアント端末を実現している．

6. まとめ

擬似シンクライアント端末の利便性を向上させるために，Microsoft Update とウィルス対策ソフトパターンファイル更新を書込み禁止モードで実行する，プロセス単位書込み制御方式を提案した．プロセス単位書込み制御方式は，ファイル I/O とレジストリ I/O の発行元プロセスを判定し，更新データに関わるプロセスが発行する I/O のみのキャッシュを作成しておく．その後，端末のシャットダウン時に，キャッシュしておいたデータを二次記憶装置へ書込む方式である．評価では，月例の Microsoft Update と TRENDMICRO ウィルスバスターのパターンファイル更新において，ハッシュ値比較によるアップデートの検証を行ったところ，メンテナンスモードでの再起動なしでアップデートが可能なことが確認できた．また，ユーザによるローカル保存を禁止するというシンクライアント端末としてのセキュリティ要件を評価し，要件達成の見通しを得た．以上から，本方式を利用することにより，頻繁なメンテナンスモードでの作業が不要となり，擬似シンクライアント端末における利便性が向上した．

7. 参考文献

[1] IDC Japan，国内クライアント仮想化関連市場規模予測を発表，
http://www.idcjapan.co.jp/Press/Current/20100607Apr.html
[2] (株)富士キメラ総研，2008 ネットワークセキュリティビジネス調査総覧
[3] (株)日経 BP，データセンター利用実態調査 2008 概要
[4] 江藤 博文，田中 芳雄，松原 義継，只木 進一，渡辺 健次，渡辺 義明，「演習用 Windows 端末群のディスクレスによる安定運用」, 情報処理学会論文誌，Vol45, No.1，pp.2-11 (2004)
[5] 奥村 勝，藤村 丞，「1000 台規模のディスクレス PC システムの構築と運用」，
情報処理学会研究報告，2008-DSM-48, pp.61-66 (2008),http://ci.nii.ac.jp/naid/110006820501
[6] Microsoft, Windows XP Embedded シンクライアントの構成,
http://www.microsoft.com/windowsembedded/ja-jp/developercenter/whitepaper/xpe/thinclient.mspx
[7] Microsoft, Windows Embedded,
http://www.microsoft.com/windowsembedded/ja-jp/default.mspx
[8] 日立ソリューションズ，在宅勤務サービス,
http://hitachisoft.jp/products/so/ws.html
[9] Microsoft, How NTFS Works,
http://technet.microsoft.com/en-us/library/cc781134(WS.10).aspx
[10] Microsoft, NTFS System Files,
http://support.microsoft.com/kb/103657/en-us
[11] Frank Swiderski, Window Snyder，渡辺 洋子 訳，脅威モデル–セキュアなアプリケーション構築，日経 BP ソフトプレス（2005）
[12] Microsoft, Volume Shadow Copy Service Technical Reference,
http://technet.microsoft.com/ja-jp/library/cc738819(WS.10).aspx
[13] Seungjun Shim, Woojoong Lee and Chanik Park, 「An Efficient Snapshot Tecnique for Ext3 File System in Linux 2.6」, http://www.linuxfordevices.com/files/rtlws-2005/SeungjunShim.pdf
[14] Open Soralis Community, ZFS, http://opensolaris.org/os/community/zfs