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

2008－DPS－134 (21)
2008－CSEC－ 40 (21)
2008／3／6

Thin Clientシステムにおけるリムーバブルメディアのアクセス制御

岩永 真幸† 毛利 公一††
† 立命館大学大学院理工学研究科 †† 立命館大学情報理工学部

近年，情報漏洩事件が問題となっている．漏洩の主な原因は，内部の人間の持ち出し，端末などの盗難や紛失である．これらを防ぐ手段として、情報を本体内に記録しないThin Clientシステムが注目されている．しかし，業務の効率の面からリムーバブルメディアが利用可能なThinClient端末が普及している．そのため，リムーバブルメディアが利用可能な端末において，重要な情報がリムーバブルメディアを介して持ち出されることを防止する必要がある．そこで，我々は実際にThin Clientシステムを導入した際の情報漏洩のシナリオを示し，Thin Clientシステムの仕組みから，OSでリムーバブルメディアへのアクセス制御を行うことで情報漏洩を防止する手法を提案する．

Access Control of Removable Media for Thin Client System

MASAYUKI IWANAGA† KOICHI MOURI††
†Graduate School of Science and Engineering, Ritsumeikan University
††College of Information Science and Engineering, Ritsumeikan University

Recently, leakage of privacy information becomes a serious problem. As a cause of the leakage, there is a case that a user who has authority takes out privacy information or a computer is stolen. Thin Client systems that don't keep information in the terminal prevent these troubles. Thin Client terminals that can use removable media are widely used from viewpoint of efficiency of the business. Therefore, it is necessary to prevent privacy information from being taken out through removable media in such type of terminal. We show the scenarios of the leakage of privacy infomation from thin client system. And we propose the technique for preventing them by doing the access control to removable media with OS.

1 はじめに

近年，情報の漏洩事件が頻繁に発生している．情報の漏洩事件の原因は，端末の盗難，内部の人間による誤操作や管理ミス，正当なアクセス権限を持つ者による情報の持ち出し，外部からの侵入者による情報の不正な持ち出し，ウイルスやワームによる漏洩などである．現在発生している情報漏洩事件の多くは，内部の人間による誤操作，端末の盗難，正当なアクセス権限を持つ者による情報の持ち出しである．従来のセキュリティ技術は，外部からの攻撃からデータを保護するものであるため，これらの原因による漏洩を防止することができない．

このような背景により，セキュリティの観点から，近年Thin Clientシステムが注目されている．Thin Clientシステムは，アプリケーションやファイルをサーバ側で保持しているシステムである．そのため，クライアント側にデータが存在せず，端末の盗難，正当なアクセス権限を持つ者が情報を持ち出すことによる情報の漏洩を防止することが可能である．また，Thin Clientシステムは，セキュリティの対策が統一することが可能である．管理者は，個々の端末にセキュリティ対策を行う必要がなく，サーバにセキュリティ対策を行うだけで情報の漏洩を防止することが可能である．

このような利点から，Thin Clientシステムは注目され，企業や学校などで導入されている．Thin Clientシステムは通常の計算機とほとんど変わらない動作が可能であるが，リムーバブルメディアが利用できない点が，通常の計算機との大きな違いである．ただし，近年リムーバブルメディアが利用可能なThin Client端末が利用され始めている．これは，CD-RやUSBメモリなどのリムーバブルメディアが普及し，リムーバブルメディアが利用できなければ業務を円滑に行えないためである．しかし，リムーバブルメディアが利用可能になると，クライアント側にデータが存在するようになり，クライアント側にデータが存在しないという，情報漏洩の防止の視点から見たThin Clientシステムの利点が損なわれる．すなわち，ユーザは，リムーバブルメディアを用いて情報を持ち出すことが可能となる．そこで，Thin Client端末において，リムーバブルメディアを用いた情報の漏洩を防止するシステムへの要求が高まっている．

以上の背景から，我々は，Thin Clientシステムで利用されることが多いUSB接続のリムーバブルメディアに着目し，アクセス制御によってリムーバブルメディアからの情報漏洩を防止するシステムを開発している．情報の漏洩を防止するために，リムーバブルメディアの利用を厳しく制限すると，リムーバブルメディアを利用できる利点が損なわれる．そこ

で，リムーバブルメディアの利用を可能としながらも，重要な情報の漏洩を防止することを目的とする．

以下，本論文では，2章でThin Clientシステムの概要について述べ，3章でThin Clientシステムにおけるリムーバブルメディアの利用方法のシナリオと研究の目的について述べる．4章で今回提案する手法について，5章で実装方法について，6章で関連技術について述べ，7章でまとめとする．

2 Thin Clientシステムの概要

2.1 実現方式

Thin Clientシステムは，データをサーバで管理しているが，処理を実行する位置によって，画像転送型とネットワークブート型に分類することが可能である．画像転送型は，クライアントでキーボードやマウスなどの入力機器の操作情報をサーバへ送信する．サーバは操作情報に対応した処理を行い，クライアントへ処理後の画像を送信する．そのためクライアント端末は高性能である必要がなく，HDD，メモリやCPUが必要ない．逆に，負担が集中するサーバが高性能である必要がある．ネットワークブート型は，起動時にサーバからOSとアプリケーションをクライアント側に送信し，処理をクライアント側で行う．そのためクライアント端末は，画像転送型に比べて高性能が要求され，サーバは高性能でなくてよい．

画像転送型は，処理をサーバで行うため，実行中もクライアント側にデータが存在せず，情報漏洩の可能性が低い．このことから，本論文では，画像転送型に注目して研究を行っている．画像転送方式には，RDP(Remote Desktop Protocol)，ICA(Independent Client Architecture)プロトコル，VNC [3]などで使われるRFB(Remote Framebuffer)プロトコル，FreeNX [4]などで使われるNXプロトコルなどが存在する

2.2 リムーバブルメディアのアクセス方式

リムーバブルメディアが利用可能なThin Clientシステムでは，クライアント端末に接続されたリムーバブルメディアのデータもサーバで処理される．Thin Clientシステムでは，クライアント端末にリムーバブルメディアが接続されると，以下の動作を行う．

1. クライアント端末は，サーバにリムーバブルメディアが接続されたことを通知する．
2. クライアント端末は，サーバにリムーバブルメディアに存在するファイルシステム情報を送信する．
3. サーバは，ファイルシステム情報を用いて，リムーバブルメディアをマウントする．

このようにクライアント端末に接続されたリムーバブルメディアをサーバにマウントすることによって，サーバは，通常のリムーバブルメディアと同様に利用することが可能となる．

また，クライアント端末のリムーバブルメディアに対して読み込みが発生すると，以下のような動作を行う．

1. サーバはreadシステムコールを発行し，クライアントに読込みが発生したファイルを特定するための情報を送信する．

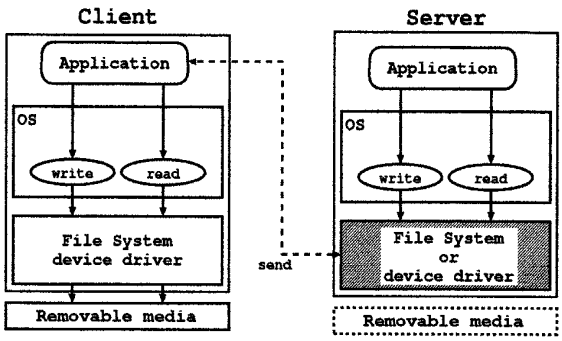

図1 Thin Clientの仕組

2. クライアント端末は，読込みが発生したファイルのデータをサーバに送信する．

サーバではすでにリムーバブルメディアがマウントされているため，通常のreadシステムコールが実行される．次に，クライアント端末に接続されたリムーバブルメディアへファイルを書き込む場合の動作を示す．

1. サーバはwriteシステムコールを発行し，クライアントに書き込むファイルを送信する．
2. クライアント端末は，リムーバブルメディアにデータを書き込む．

以上のように，読込みや書込みが発生すると，リムーバブルメディアがネットワークの先にあるため，データの送受信が必要となる．その仕組みを図1に示す．読込みや書込みが発生すると，readやwriteシステムコールが実行される．実際の，データの送受信処理はデバイスドライバやファイルシステムによって行われている．デバイスドライバを利用しているか，ファイルシステムを利用しているかは，プロトコルによって異なる．

3 Thin Clientでの情報の漏洩シナリオ

企業や学校などでThin Clientシステムが，採用され始めている．そのような状況で，リムーバブルメディアが利用可能であると，情報の漏洩が発生する可能性がある．しかし，情報の漏洩を防止するために，リムーバブルメディアの利用を厳しく制限していまうと，作業が円滑に行えなくなってしまう．理想としては，重要なデータが漏洩することなく，リムーバブルメディアが自由に利用できる環境である．しかし，従来のシステムでは，漏洩を防止するためにリムーバブルメディアの利用を制限する必要があり，この2つを同時に満たすことは不可能である．そこで，リムーバブルメディアが利用可能なThin Clientシステムにおいて，どのような情報漏洩シナリオがあるか，またどのように情報を保護すべきかについて以下に例を示す．

1. 計算機に存在するデータは，すべてが漏洩を防止すべきものではない．情報漏洩を防止すべき重要なデータと，漏洩しても問題にならないデータが存在する．ま

た，漏洩しても問題とならないデータの中には，製品カタログや商品紹介のパワーポイントなど，持ち出して利用する必要のあるデータが存在する．そのため，すべてのデータがリムーバブルメディアへの書込みを禁止されると，業務などに支障をきたす．よって，データの重要度を区別し，データごとにリムーバブルメディアの利用を制御する．データごとにリムーバブルメディアを用いた持ち出しを，許可・禁止することによって，重要なデータを漏洩させることなく，必要なデータは持ち出すことが可能となる．

2. 重要なデータは，利用者が限定されている場合が多い．その場合，持ち出してもよいユーザを限定することができる．すなわち，データの漏洩を防止するために，特定のユーザのみがリムーバブルメディアへの書込みできるように制御する．そこで，データごとにリムーバブルメディアへの書込みをユーザによって制御する．データを持ち出す必要のある社員のみ許可と設定することで，情報の漏洩を防止する．
3. Thin Client システムでは，クライアント端末からサーバにログインするため，どの端末からでも自分の環境を利用することが可能である．しかし，部署や個室などに設置される Thin Client 端末は，利用者が固定されている場合が存在する．このような場合に，端末によってデータのリムーバブルメディアへの書込みを制御する．また，リムーバブルメディアを利用できる端末を限定することで，データを持ち出すユーザを監視することが可能となり，情報漏洩の抑止力や犯人の特定にも利用できる．
4. サーバとクライアントでは，リムーバブルメディアを利用する目的が異なってくる．また，サーバを利用するユーザは，管理者であり，通常のユーザとは権限が異なる．例えば，サーバでは，データのバックアップをとる．重要なデータは，バックアップが必要であるが，リムーバブルメディアへの書込みを自由にしてしまうと情報の漏洩につながる．そこで，サーバでのみ，リムーバブルメディアへの書込みを許可する．サーバとクライアントを区別して制御を行うことで，サーバで必要な処理を制限することなく，クライアント端末から情報が漏洩することを防ぐことができる．

以上のような制御が可能となると，情報が漏洩することを防止することができ，作業に影響を与えずにリムーバブルメディアを利用することが可能となる．本研究では，以上のような制御を可能とすることを目的としている．

4 提案手法

2 章で述べた Thin Client の特徴と 3 章で述べた制御を可能とするために，以下のような手法を用いる．

4.1 ファイル単位のアクセス制御

3 章のシナリオ 1 を実現するためには，データを区別する必要がある．計算機は，データをファイルとして管理している．このことから，ファイルごとにそのファイルの保護方法を記述した保護ポリシを設定することを可能とする．これにより，ファイルが保護の単位となる．重要なファイルに保護ファイルを設定することで，データごとにリムーバブルメディアを用いた情報の漏洩を防止することが可能となる．

4.2 アクセス制御の実現方式

リムーバブルメディアを用いた情報の漏洩は，それへのデータ書込みによって発生する．リムーバブルメディアに存在するデータを読み込んだり，サーバの HDD に書き込むような操作は，ウイルスなどをサーバに持ち込ませ情報漏洩を発生させる可能性がある．しかし，ウイルスによる情報漏洩は，既存の技術で防止することが可能である．よって，リムーバブルメディアへのデータ書き込みを制御することによって情報の漏洩を防止する．

2 章で述べたように，Thin Client システムでは，すべての処理をサーバで行っており，クライアント側のリムーバブルメディアへの書込みもサーバで実行される．また，図 1 に示したように，リムーバブルメディアへの書込みは write システムコールが実行され，データの送受信はファイルシステムやデバイスドライバが行っている．よって，サーバの OS の中で write システムコールを制御することによって，すべてのリムーバブルメディアの書込み制御を行うことが可能となる．write システムコールの中で，リムーバブルメディアへの書込みの可否を決定する．

4.3 コンテキストの利用

3 章のシナリオ 2，3，4 の制御を実現するために，コンテキストとしてユーザや端末を利用する．保護ポリシに，ユーザや端末によってリムーバブルメディアへの書込みの可否を記述可能とする．これにより，ユーザ，端末，サーバ・クライアントによって，書込みの制御を行うことが可能となる．このコンテキストを，write システムコールの中で参照し，保護ポリシと比較する．このような仕組みで，コンテキストを利用したアクセス制御を実現する．

5 実装方法

以上のような手法を実現するために，サーバには，プライバシアウェア OS *Salvia*[1] を利用する．*Salvia* は，4 章で述べた 3 つの手法を用いて情報漏洩を防止している．

5.1 *Salvia* の概要

Salvia は，OS でデータの漏洩を防止する．OS によってデータ漏洩を防止することにより，アプリケーションのセキュリティ機能が不完全であっても情報漏洩を防止することができる．*Salvia* は，ファイルごとに保護ポリシを設定可能としている．保護ポリシは，コンテキストを用いてシステムコールごとに制御条件を定義する．システムコールは多数存在するため，表 1 に示すようにシステムコールは処理内容やデータの伝搬範囲に着目し，自ホスト内での読み書き処理を read と write，他ホストへのデータ送受信処理については send_local と send_remote に分類している．また，*Salvia* で使用するコンテキストには，このうちユーザ，位置，時刻，IP アドレス，プロセスの動作履歴である．ユーザを利用した保護ポリシの例を図 2 に示す．2〜11 行目が，デフォルトのアクセス権限を記述している．この部分は，すべてのアクセス権限が不許可であることを定義している．13〜22 行目が，条件付きアクセス権限の記述である．14〜18 行目でユーザ

```
1  <data_protection_policy>

2  <default_access>
3   <read>deny</read>
4   <write>
5    <write_access update="deny">deny</write_access>
6   </write>
7   <send_local>deny</send_local>
8   <send_remote>
9    <send_remote_access>deny</send_remote_access>
10  </send_remote>
11 </default_access>

12 <data_protection_domain type="none">

13 <ACL>

14  <context>
15  <user>
16   <user_id type="real">1001</user_id>
17  </user>
18  </context>

19  <access>
20   <read> allow</read>
21  </access>

22 </ACL>

23 </data_protection_domain>
24 </data_protection_policy>
```

図2　保護ポリシーの例

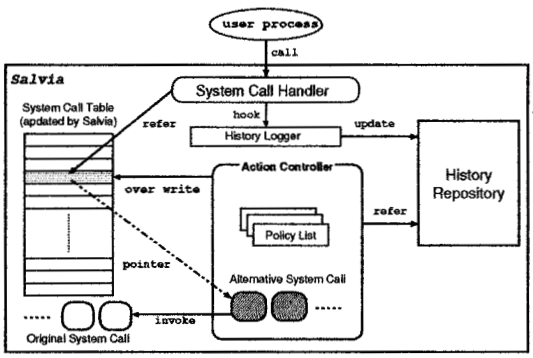

図3　*Salvia* の構造

ID が 1001 のユーザを指定し，19〜21 行目でそのユーザに読込みの許可を設定している．

プロセスは，ファイルをオープンすることにより，ファイルへのアクセスを開始する．そのため，*Salvia* は，保護ポリシが設定されているファイルがオープンされると，プロセスに対してその保護ポリシを適用する．そして，*Salvia* は，そのプロセスが表 1 のシステムコールを実行すると，コンテキストと保護ポリシを比較し，そのアクセスの可否を決定する．

Salvia の構成を図 3 に示す．プロセスが，システムコールを呼び出すと，History Logger によってプロセスの履歴がとられ，History Repository に蓄積される．History Repository には，コンテキストも蓄積されている．そして，呼び出されたシステムコールが，System Call Table を参照し，実行される．システムコールは，Action Controller によって System Call Table を書き換えることによって変更されている．これにより，システムコールが実行された際にアクセス制御を行うことが可能である．書き換えられたシステムコールの中で，そのプロセスに付加されている保護ポリシと History Repository を参照することにより，そのシステムコールの可否を判定する．

5.2　システム構成

Salvia を利用することにり 4 章のファイル単位の制御，アクセス制御，コンテキストの利用が実現されている．しかし，*Salvia* はリムーバブルメディアの制御を行っていないため以下のような機能を追加する必要がある．

5.2.1　書込み制御

現在は，システムコールを read，write，send_local，send_remote システムコールに分類している．本手法では，リムーバブルメディアへの書込みを制御するため，write システムコールの制御を行う．以下に write システムコールの中での制御の流れを示す．

- プロセスに付加しているポリシの参照，コンテキストの取得
- 書込み先のデバイスを確認
- 書込みの可否を判定する

この中で，書込み先のデバイスの確認は，現在の *Salvia* では行っていない．そこで，write システムコールの中で書込み先がリムーバブルメディアであるかを識別する必要がある．

このとき，write システムコールの中だけで識別することは困難である．そのため，OS の中で現在接続されているリムーバブルメディアの情報を管理する．そこで，write システムコールの中で書込み先の i ノードから，書込み先のメジャー番号とマイナー番号を取得することが可能であることを利用する．OS の中で現在接続されているリムーバブルメディアのマイナー番号とメジャー番号を管理し，write システムコールの中で書込み先のメジャー番号とマイナー番号を比較することによって書込み先がリムーバブルメディアであるかを確認する．OS の中で管理するリムーバブルメディアの情報は，マイナー番号とメジャー番号の他に，そのリムーバブルメディアの端末 ID を管理する．これにより，書込み時にリムーバブルメディアが利用されている端末を知ることが可能となり，端末による制御のコンテキストとすることができる．

リムーバブルメディアの情報の取得は，適切なタイミングで行わなければいけない．リムーバブルメディアが利用可能になる時点より遅れて取得したり，情報の取得洩れが発生すると保護すべきデータが書き込まれてしまう．よって，リムーバブルメディアが接続された時に実行されるプログラムである Probe 関数の中で取得する．また，デバイスが切り離された時に実行される Remove 関数の中でデバイス情報を削除する．このような仕組により現在接続中のリムーバブルメディアの情報を漏れなく管理することが可能となる．このような機能を実現するためのシステムの構成を図 4 に示す．

5.2.2　保護ポリシ

以上のような制御システムによって，3 章で述べたシナリオの実現が可能となる．それにともない，保護ポリシも 3

表1　システムコールの分類

分類名	システムコール
read	read, readv, pread64, mmap, mmap2, readahead
write	write, writev, pwrite64, sendfile, sendfile64, mmap, mmap2, munmap
send_local	write, writev, sendfile, sendfile64, mmap, mmap2, socketcall, ipc, mq_timedsend
send_remote	write, writev, sendfile, sendfile64, socketcall

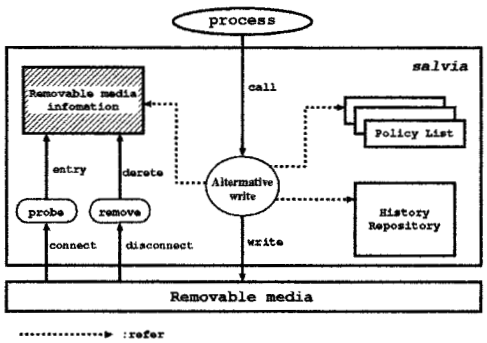

図4　Thin Client 対応 *Salvia* の構造

```
1  <data_protection_policy>

2  <default_access>
3   <read>allow</read>
4   <write>
5    <write_access update="deny">allow</write_access>
6     <write_usb>deny</write_usb>
7   </write>
8   <send_local>deny</send_local>
9   <send_remote>
10    <send_remote_access>deny</send_remote_access>
11  </send_remote>
12 </default_access>

13 <data_protection_domain type="none">

14 <ACL>

15  <context>
16  <user>
17   <user_id type="real">1005</user_id>
18  </user>
19  </context>

20  <access>
21   <write>
22    <write_usb>deny</write_usb>
23   </write>
24  </access>

25 </ACL>

26 </data_protection_domain>
27 </data_protection_policy>
```

図5　シナリオ1, 2 の保護ポリシ記述例

章で述べた制御を記述可能とするための拡張が必要となる．3 章のシナリオ 1 と 2 に対応するためのポリシを図 5 に示す．6 行目の ⟨write_removable_media⟩ のタグを利用して，USB 接続のリムーバブルメディアへの書込みの可否を記述可能とする．これにより，ファイルごとにリムーバブルメディアへの書込みを制御するシナリオ 1 が制御可能となる．シナリオ 2 は，ユーザによる制御である．*Salvia*は，ユーザをコンテキストとして利用可能であるため，15〜24 行目のような記述によってユーザごとにファイルのリムーバブルメディアへの書込みを制御できる．

3 章のシナリオ 3 のような制御をするためのポリシを図 6 に示す．2〜6 行目でリムーバブルメディアへの書込みを禁止し，8〜16 行目で端末 ID が 100 と 102 の端末のみリムーバブルメディアへの書込みを許可している．12，13 行目の ⟨terminal_id⟩ タグによって端末を記述可能とした．

3 章のシナリオ 4 のような制御を行うためのポリシを図 7 に示す．2〜6 行目でリムーバブルメディアへの書込みを禁止し，8〜15 行目で server のみリムーバブルメディアへの書込みを許可している．⟨terminal_type⟩ タグを利用して，server と client のリムーバブルメディアへの書込みの可否を記述可能とした．

6 関連技術

Thin Client 端末で外部記憶装置が利用することが可能にすると，情報漏洩の可能性が発生する．そのため，Thin Client 端末を販売している企業は，さまざまな対策をおこなっている．

富士通の Portshutter [2] は，USB 接続の機器や PC カードの制御が可能である．この機能は，利用可能なデバイスを個々のデバイスごとに設定することが可能である．これにより，会社で支給した USB メモリや PC カードのみ利用可能といったことが可能となるため，会社外の人間が外部記憶装置を利用して情報を持ち出すことを防止することが可能である．しかし，許可されたデバイスを持つ人間は，データを持ち出すことが可能である．内部の人間による情報の漏洩を防止するために，デバイスを利用できるユーザを限定すると，利用できないユーザが業務を円滑に行えないという問題がある．しかし，本手法では，デバイスを利用できるユーザを限定することがない．また，重要なデータはデバイスへ書き込むことができないため，内部の人間であっても情報が漏洩することがない．

いくつかの Thin Client システムで，デバイスの制御を行うことが可能である．これは，ユーザや端末によってデバイスの利用の可否を設定することが可能である．企業などでは，重要なデータにアクセスしてもよいユーザが決まっている場合が多い．よって，重要なデータにアクセスする必要のないユーザのデバイス利用を禁止したり，重要なデータにアクセスする必要のないユーザのみが利用する端末でデバイスの利用を禁止することで情報の漏洩を防ぐことが可能である．このシステムは，制御のためのコンテキストにユーザや端末を用いている点が，我々が提案した手法と同じである．

```
1  <data_protection_policy>

2  <default_access>
3   <write>
4     <write_usb>deny</write_usb>
5   </write>
6  </default_access>

7  <data_protection_domain type="none">

8  <ACL>
9   <access>
10   <write>
11    <write_usb>allow</write_usb>
12     <terminal_id>100</terminal_id>
13     <terminal_id>102</terminal_id>
14   </write>
15  </access>
16 </ACL>

17 </data_protection_domain>
18 </data_protection_policy>
```

図 6　シナリオ 3 の保護ポリシ記述例

```
1  <data_protection_policy>

2  <default_access>
3   <write>
4    <write_usb>deny</write_usb>
5   </write>
6  </default_access>

7  <data_protection_domain type="none">

8  <ACL>
9   <access>
10   <write>
11    <write_usb>allow</write_usb>
12     <terminal_type>server</terminal_type>
13   </write>
14  </access>
15 </ACL>

16 </data_protection_domain>
17 </data_protection_policy>
```

図 7　シナリオ 4 の保護ポリシ記述例

しかし，このシステムは，デバイスを制御の対象としている．そのため，富士通の Portshutter と同じで，内部の人間による情報の漏洩を防ぐことができないといった問題や，デバイスを利用できるユーザを限定してしまうという問題がある．それに対して，我々が提案した手法は，ファイルを制御の対象としているため，このような問題が発生しない．

Thin Client システムにおいても，ユーザのなりすましが問題である．ユーザ ID やパスワードが洩れてしまうと，悪意のある人間に外部記憶装置を使ってデータを持ち出されてしまう．そこで，諮問認証や認証デバイスなどで認証を強化することでなりすましを防ぎ，情報の漏洩を防ぐシステムが，多くの Thin Client システムで実装されている．しかし，このシステムも，内部の人間による情報の漏洩を防止することができない．

7 おわりに

本論文では，Thin Client システムにおけるリムーバブルメディアを用いた情報の漏洩の防止について述べた．Thin Client システムは，データが端末に存在しないために，セキュリティの面から注目されている．しかし，現在利用されている Thin Clinet 端末はリムーバブルメディアが利用できるものが増え，それを用いた情報の漏洩が問題となっている．本論文では，その問題を解決するために Thin Client システムを導入した際のデータ保護シナリオを実現することにより，リムーバブルメディアの利用に過度に制限を加えることなくデータの漏洩を防止し，計算機の利用に支障を与えない手法について述べた．提案する手法は，サーバの OS として *Saliva* を利用し，リムーバブルメディアへのアクセス制御の機能の追加と，保護ポリシの拡張を行うものである．現在，それらの実装を進めている．

参考文献

[1] 鈴来 和久, 一柳 淑美, 毛利 公一, 大久保 英嗣: Privacy-Aware OS Salvia におけるデータアクセス時のコンテキストに基づく適応的データ保護方式, 情報処理学会論文誌:コンピューティングシステム, Vol. 47, No. SIG3, pp. 1-15 (2006).

[2] FMV シンクライアントによるセキュリティ強化，情報漏洩対策の実現, FUJITSU JOURNAL. Vol. 32, NO. 3, pp. 2–7 (2006).

[3] vnc, http://www.realvnc.com/.

[4] FreeNX, http://freenx.berlios.de/.