L-008

コンテキスト依存暗号化ファイルシステム
Context-based cryptographic file system

清水 亮博*
shimizu.akihiro@lab.ntt.co.jp

1 はじめに

携帯型端末の高度化や普及に伴い、顧客情報など重要情報を持ち歩くことが可能になっている。それに伴って端末の盗難や置き忘れによる重要情報の流出が社会問題化している。

重要情報の流出を防ぐために暗号化ファイルシステムなどの手法が実用で使われているが、暗号化を解除している状態での盗難に対応できないなどの問題がある。

本稿ではこの問題を解決するためにコンテキスト依存アクセス制御を提案し、その一方式として暗号化ファイルシステムを用いた方法について論ずる。

2 従来の対策とその問題点

現在情報流出を防止する手段としては、ファイルやファイルシステムの暗号化が使われている。しかしこの方式では暗号を解いた状態で盗難にあうと無力である。その対策として頻繁にパスワード等を入力させると、使いづらい物となってしまう。

また暗号化されたファイルでも流出すると解かれてしまう恐れがある。よって、盗難時の情報漏洩対策としては暗号で守るだけではなくファイル自体を消去したい。

別の既存手法としては、可搬型の Thin クライアントを用いてファイルを一切外に持ち出さない方法がある。しかしこの手法はネットワークに繋がっていないと利用できない上、帯域を十分に確保できない環境では使いづらいという問題がある。

3 コンテキスト依存アクセス制御システム

上記の問題を解決するため、例えば GPS や IC タグ等の位置観測手段にて現在地を把握し、その位置に応じてアクセス制御を変更する方法 [1, 2] が知られている。ただこれらの手法では GPS 衛星の電波を受信できなかったり、位置を知らせる IC タグが無い所では使えない。

これらの手法では「その時」の状況に応じてアクセス制御を定めているため、そのような問題が生じる。しかし直前の状況では位置が測位されており、問題のない場所であることが判っているかも知れない。

このようにその時の状況のみに頼る従来の手法ではなく、様々な状況の変化、すなわちコンテキストに伴ってアクセス制御を変化させるコンテキスト依存アクセス制御システムを考案した。

なお、ここで扱うアクセス制御とは、ファイル自体の削除をも含む。盗難にあった場合など、アクセス制御に頼らずファイルを消去すべき状況もあるためである。

4 コンテキストとは

ここで言うコンテキストとは、状況の変化の列である。状況とは端末が検知可能な内部・外部状態であるとする。またそのコンテキストに対する動作をポリシーとしてユーザが指定する。以下ではコンテキストの例と、対応するポリシーの例を示す。

4.1 論理的なコンテキスト

4.1.1 認証の失敗

端末が盗難された場合、入力されるのは本来と異なる認証データのみである可能性が高い。従って何度も認証を失敗した場合、正規のユーザでないと推測されるので、アクセスを禁止するポリシーにすべきである。特に存在しないユーザ名を一定回数以上連続して入力された場合には正規ユーザではない可能性が非常に高いので、アクセスの全面禁止およびファイル削除を行うようなポリシーが考えられる。

4.1.2 認証の成功

あるコンテキストにおいてアクセス制限がかかっているファイルでも、その場に正規ユーザが居ればファイルにアクセスさせなければならない。そのような場合に認証を行うことでアクセスを許可するようなポリシーが必要である。なお信頼できない環境などの状況によっては、通常の認証に加えて指紋認証やワンタイムパスワードなどによるより高度な認証をユーザに求めるといったポリシーが必要である。

4.1.3 上司の許可

重要情報によっては、アクセスに上司の許可が必要な場合がある。このようなポリシーを実現するために、アクセス制御システムには上司の許可を取得する手段や、取得していることを示す手段が必要である。

4.1.4 アクセス権

例えば認証されたユーザが要求しているファイルは、本来そのユーザがアクセスしてはならないポリシーの場合、そのユーザにアクセス権を渡さない手段が必要である。

4.1.5 ネットワークに関連するコンテキスト

ネットワークにつながっているか、つながっているならその場所は自席/社内/社外か、等のコンテキストにより、アクセス制御を変更したり追加のユーザ認証を

* NTT 情報流通プラットフォーム研究所

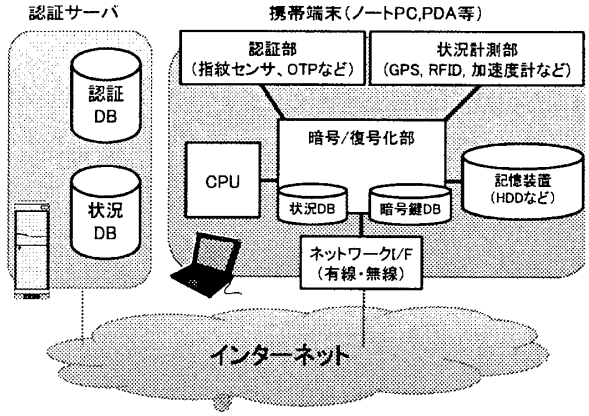

図 1　システム概略

求めたりするポリシーが考えられる。

4.1.6　時間に関連するコンテキスト

不自然な時間にログインしていないか、出張日程によればアクセスしない時間ではないか、ログインしている時間が長すぎないかなどのコンテキストによりアクセス制御や認証要求を行うポリシーが考えられる。

4.2　物理的なコンテキスト

4.2.1　位置情報

GPS や RFID、加速度計の積分による移動量の推計などにより、本来あるべき所にあるかどうかを判断し、必要な認証を要求したりアクセス制限をするポリシーが考えられる。

4.2.2　衝撃

加速度計によって強い衝撃が何度も観測された場合、物理的な不正アクセスや盗難にあっているという可能性がある。よってこのような場合にも必要な認証を要求し、得られなければファイルの削除を行うポリシーが考えられる。

4.2.3　端末本体の分解

通常の利用において端末本体を分解して開ける状況は修理の場合のみである。その場合であれば、重要情報はすべて消去されているか、消去されても構わない。

それ以外の状況としてありうるのは、盗難にあって内部の記憶装置等を抜き出そうとする状況である。このような場合には即座にファイル除去を行うポリシーが考えられる。

4.2.4　盗難

携帯端末が盗難にあえば、盗難された端末のファイルを消去したい。しかし携帯端末単独で盗難にあった事を検知するのは難しい場合がある。このような場合、盗難された端末がネットワークに接続した時点で外部から盗難情報を得て、ファイル消去を行えるポリシーが望ましい。

5　提案システムの構成とその動作

以下ではコンテキスト依存アクセス制御システムを、暗号化ファイルシステムを用いて実現する方法について述べる。

このシステムではアクセス制御は暗号鍵の有無、および暗号化ファイルの消去により実現される。以下の説明では暗号方式として対象鍵を用いる。非対象鍵でも実現可能ではあるが、システムが複雑になるためここでは対象鍵しか扱わない。

5.1　システム構成

システムの概略は図 1 の通り、携帯端末と認証サーバから構成される。

5.1.1　携帯端末

認証部

ユーザ ID とパスワード、OTP, 指紋センサ等によりユーザ認証を行う。

状況計測部

GPS, RFID, 加速度計、本体オープン検知スイッチなどの状況を計測する。

暗号/復号化部

CPU が暗号化されたファイルを読み書きする場合、暗号化部にて暗号化・復号化を行う。なお内部に状況 DB と暗号鍵 DB を持っている。

状況 DB

端末が検知した状況を保存しておき、端末単独でコンテキストの判断を行うときや、認証サーバに現状のコンテキストを送る際に用いる。

暗号鍵 DB

暗号鍵は、暗号化を行う単位（ファイル一つ、サブディレクトリの中、ファイルシステム全体）それぞれに対応しており、アクセス制御はこの暗号鍵の有無で行う。

記憶装置

アクセス制御されるファイルは暗号化されてここに保存される。現在ではハードディスクあるいはフラッシュメモリが用いられる。

ネットワーク I/F

Ethernet や、802.11a/b/g/n, 携帯電話など、Internet につながる様々な物がある。なおここでのリンクの状態や、DHCP による取得アドレス等の layer3 の状態、認証サーバと通信の可否等の layer7 の状態も重要な状況であり、状況 DB に保存される。

5.1.2　認証サーバ

状況 DB

管理下にある各端末毎に状況を蓄積している。

認証 DB

ユーザの認証に必要な情報、および暗号鍵を蓄積している。

5.2　システムの動作

以下では、新たな状況によりコンテキストが変更される場合に、どのように各部が動作するかを示す。

5.2.1　認証サーバと通信できない場合

携帯端末は常にネットワークとアクセス可能とは限らない。よって認証サーバと通信できない場合でも動作しなければならない。図 2 に概略を示す。

状況の変化がおこると、これまでの状況と合わせてコンテキストの判断を行う。その結果、明らかに盗難にあっている場合などの緊急事態には、暗号鍵に留まらず暗号化されたファイル自体を削除する。

緊急事態でない場合、アクセス制御を厳しくするコンテキストでなければ、そのまま終了する。アクセス制御を厳しくする場合は、必要な認証をユーザに求める。十分な認証を得られなかった場合、暗号鍵を消去してファイルにアクセスできないようにする。

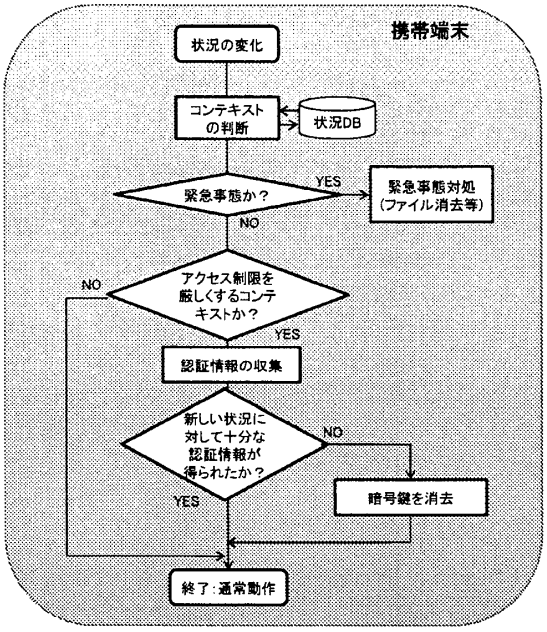

図 2　認証サーバと通信できない場合

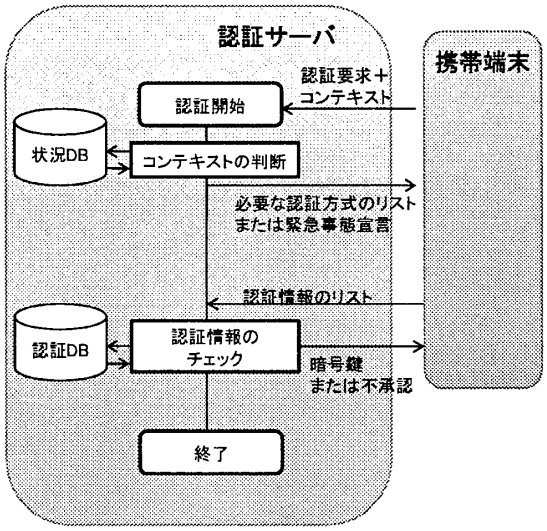

図 3　認証サーバの動作

5.2.2　認証サーバと通信可能な場合

認証サーバと通信可能な場合は、認証サーバにてコンテキストを判断する。

なお以下で必要な通信は、すべて TLS[3] 等で暗号化されているものとする。

図 4 において、携帯端末は状況の変化を検知したため、コンテキストの判断を行う。緊急事態でなければ、暗号鍵を消去し、認証サーバに認証要求とコンテキストを送る。

図 3 において認証サーバは端末より認証要求とコンテキストを受け取ったので、認証を開始する。まず状況 DB にあるコンテキストと、送られてきたコンテキストから状況を判断し、その結果として必要な認証方式のリスト、あるいは緊急事態宣言を送る。

携帯端末は緊急事態宣言を受けると暗号鍵とファイルを消去する。認証方式のリストを受け取った場合は必要な認証情報をユーザに対して求め、その結果を認証サーバに送り返す。

認証サーバは認証の結果を受け取るとその情報をチェックし、正しければ認証 DB に記録されている該当の暗号鍵を端末に送る。正しくなければ不承認を送る。

端末は送られてきた暗号鍵を暗号鍵 DB に保存する。不承認の場合は再度認証要求を行う。同一の端末で不承認が繰り返された時は、緊急事態宣言を送ってファイルや暗号鍵の消去を行う。

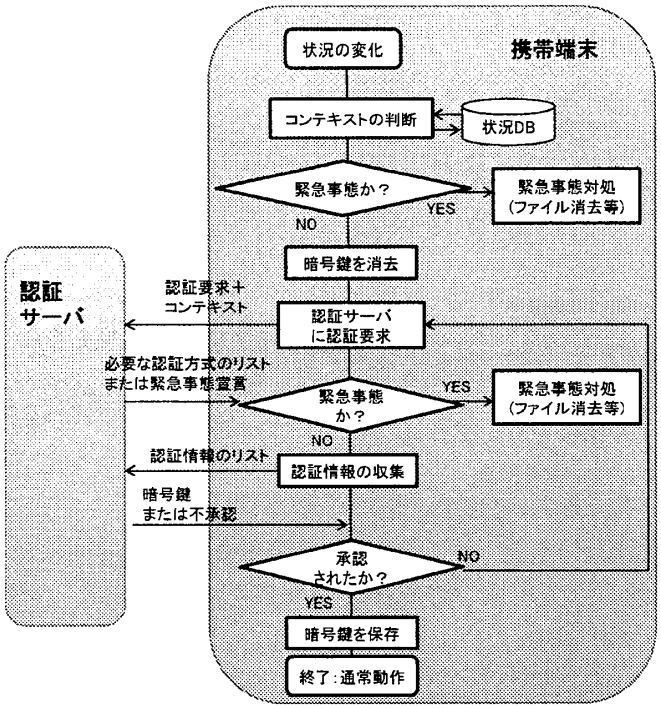

図 4　認証サーバと通信できる場合

6　本システムによる効果

本節では提案システムの一実現例を示す為に、第 4 節に挙げた各コンテキストとポリシーを実現する方法を示す。

6.1　論理的なコンテキスト

6.1.1　認証の失敗

不正ユーザが何度も認証を失敗した場合、その失敗自体が一つの状況なので、コンテキストに連続して認証失敗があった場合にはアクセス制御の全面禁止、すなわち暗号鍵の全消去を行う。また存在しないユーザによるログイン試行は盗難されている可能性が高いので、暗号鍵の全消去だけでなくファイル削除も行うべきであろう。

6.1.2　認証の成功

正規ユーザに認証制限がされているファイルにアクセスさせるためには、ユーザがログインしたいという状況を発生させれば、5.2 節にて示されている方法でアクセス制御を変更する事ができる。

6.1.3　上司の許可

上司のアクセス許可は該当のファイルに対応する上司の許可を表す鍵で表現する。この鍵はあらかじめ

サーバからダウンロードしておいたり、必要になった時に取りに行くことができる。

6.1.4 アクセス権

ユーザが要求したファイルのアクセス許可があるかどうかは、暗号鍵の有無で表現される。あるユーザがアクセス権を持たないファイルにアクセスしようとしても、対応する暗号鍵がないためアクセスできない。追加の認証があればアクセス可能な場合は 6.1.2 にある通り。

6.1.5 ネットワーク的なコンテキスト

ネットワークへの接続の有無はネットワーク I/F が検知する。利用場所は IP アドレスや MAC アドレス、802.1x 認証、認証サーバへのホップ数などで検知する事ができる。これらの状況により、アクセス制御を行ったり、追加のユーザ認証を求めることができる。

6.1.6 時間的なコンテキスト

時間によるアクセス制御は、認証サーバの状況にあらかじめ行動パターンを登録しておき、それからどの程度離れているかでアクセス制御を行うことができる。

同じユーザが長時間ログインし続けている場合、端末を操作しているのが正規ユーザであるかどうかを確かめる必要があるので、長時間利用という状況を発生し、ユーザ認証を再度求めるという対応を行う。

6.2 物理的なコンテキスト

6.2.1 位置情報

GPS や RFID、加速度計の積分による移動量の推計などにより、現在地という状況を発生させ、それが本来あるべき所にあるかどうかでアクセス制御をおこなえる。

6.2.2 衝撃

加速度計によって強い衝撃が何度も観測された場合、衝撃という状況が発生するので、衝撃が多数含まれるコンテキストでは暗号鍵の消去、場合によってはファイル消去を行えばよい。

6.2.3 端末本体

端末をあけた場合、本体オープン検知スイッチを設けておけば、端末オープンという状況が観測できる。この状況が発生した場合には、いかなるコンテキストにおいても暗号鍵の消去とファイル消去を行う。

6.2.4 盗難

盗難を検知するのは困難である。しかし認証サーバの状況 DB に盗難端末情報を記録しておけば、盗難端末が 1 回でも Internet につなげれられば、認証サーバとの通信が発生し緊急事態宣言を受け取るので、ファイル消去が行える。

7 本システムに対する攻撃方法に関する一考察

以下では、本来アクセス権を持たないユーザがアクセス権を奪取する攻撃を考える。それ以外に本来アクセスが許されるユーザがアクセスできなくするという攻撃もあるが、本システムは重要情報の流出を防ぐことが主眼なため、ここでは考慮しない。

7.1 ソフトウェア的な方法

ファイルアクセスは本システムによって守られても、使うアプリケーションに脆弱性があればアプリケーションを攻撃してそこから内容を取り出す事が可能である場合がある。また大抵の場合アプリケーション内部では暗号化されないデータが用いられているため、メモリを直接読んだり仮想記憶の swap 上のデータを読むという方法もある。

この場合、何らかの方法でマルウェア (ウィルスやルートキット、トロイの木馬等) を対象の端末に導入する必要がある。これは従来手法であるアンチウィルスソフト等で防ぐ方法や、OS のアクセス制御で一般ユーザによるアプリケーションの変更を防ぐ方法がある。

7.2 ハードウェア的な方法

暗号化部がハードウェアの場合プローブを刺して信号を直接読み取る方法がある。これを防ぐためには、耐タンパ機能を持つ LSI に暗号化部をすべて載せてしまい、暗号化されていない鍵情報などを一切 LSI 外に出さないという方法がある。

8 今後の課題

現状で分かっている課題は以下の通りである。

8.1 実装例の製作

本方式の有効性を評価するためには、実装する必要がある。本件で最も困難な部分の一つは暗号化ファイルシステムとその鍵のコントロールである。これに対してはアプリケーションレベルで動作する暗号化ファイルシステムを利用して、カーネルプログラミングを不要にする事で実装の難易度を下げる予定である。

8.2 評価方法

現在わかっている評価方法としては、使用感に影響するコンテキストの変化に対して十分速いポリシーの適用が可能かどうかというものがある。これ以外にも評価すべき項目は存在するであろうから、実装を通じて評価方法を確立したい。

9 おわりに

本稿では、携帯端末における情報漏洩防止のためのコンテキスト依存アクセス制御システムについて論じた。今後は評価のために実装を行い、有用性を実証する。

謝辞

本研究を遂行する上で、NTT 情報流通プラットフォーム研究所の高橋健司氏、下村道夫氏には多大なご指導を頂きました。厚く御礼申し上げます。

参考文献

[1] 磯川弘実, 木田雄一：モバイル端末管理装置, 公開特許公報, (株) 日立製作所 (2002).
[2] 広重一仁：電子ファイルのアクセス制御システム及びアクセス制御方法, 公開特許公報, 日本テレコム (株) (2004).
[3] Dierksa, T. and Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.1, RFC 4346, IETF (2006).