4E-3

コールスタックを用いた鍵管理方式

砂田　英之†

三菱電機株式会社　情報技術総合研究所†

１．はじめに

近年、多くの企業では情報漏洩の対策として、セキュリティ施策の設定・運用を行っており、機密情報を扱う情報システムにおいては、アクセス制御や情報の暗号化といった施策が実施されている[1]。

情報システム（アプリケーション）が機密情報を暗号化するのに用いる鍵の管理にはパスワードによるアクセス制御により保護する場合が多い。しかし、パスワード情報をアプリケーションから利用できるようにファイル等に記録して運用することは漏洩の危険性が伴う。

この課題に対して、アプリケーションに埋め込んだ乱数とパスワードから共通鍵を生成して暗号化に用いるという方式[2]が提起されているが、埋め込み情報が危殆化する問題や、管理者／開発者による内部犯行による漏洩に対処できないといった課題があった。

本論文では、これらの課題を解決するために、開発者が意識的に埋め込む情報ではなく、無意識に埋め込むアプリケーションのコールスタック情報を用いて鍵を管理する方式について提案する。

２．従来技術

(1)埋め込み情報を用いた共通鍵生成方式

文献[2]では、鍵管理の方式として、アプリケーションに埋め込んだ乱数と管理者の入力するパスワード情報から共通鍵を生成する方式が示されている。

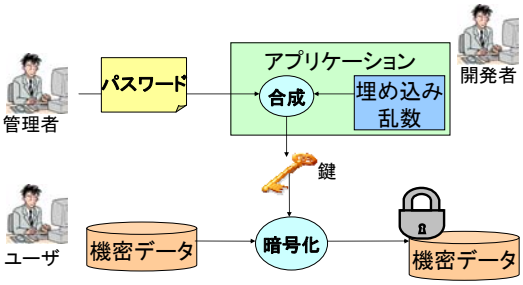

図１　システム構成（従来技術）

Key Management Method to use Call Stack Information

†Information Technology R&D Center, Mitsubishi Electric Corporation

乱数とパスワード情報とを合成して暗号用の共通鍵を生成するため、パスワード情報が漏洩しても、鍵情報そのものは漏洩することがない。

(2)課題

情報漏洩の要因としては、紛失・盗難、ウイルス、誤操作などの他に、内部関係者による犯行が主要な原因となっている[3]。

従来の方式では、アプリケーションに埋め込んだ情報は開発者が知りえる情報であり、情報システムの試験、運用、保守時の管理ミスによりパスワードが開発者に漏洩した場合、共通鍵を生成し機密情報にアクセスが可能となる。

また、埋め込み情報は危殆化する恐れがあり、乱数が漏洩した場合には、アプリケーションのモジュール自体を交換する必要があり、運用面での課題が残っている。

３．解決策

(1)コールスタックを用いた鍵管理方式

アプリケーションで暗号化のために用いる鍵を管理する方法として、図 2 の構成によるコールスタック情報を用いてキーストア用のパスワードを生成し、鍵を管理する方式が考えられる。

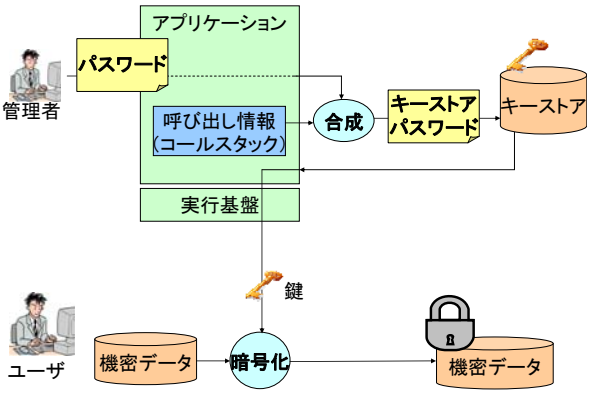

図２　システム構成（解決策）

コールスタック情報は、アプリケーションの呼び出し関係の他に、アプリケーションの実行基盤の情報が必要なため、開発者にも簡単に入手できる情報ではない。

(2)実現方式

コールスタックを用いた鍵管理方式の機能構成を図3に示す。

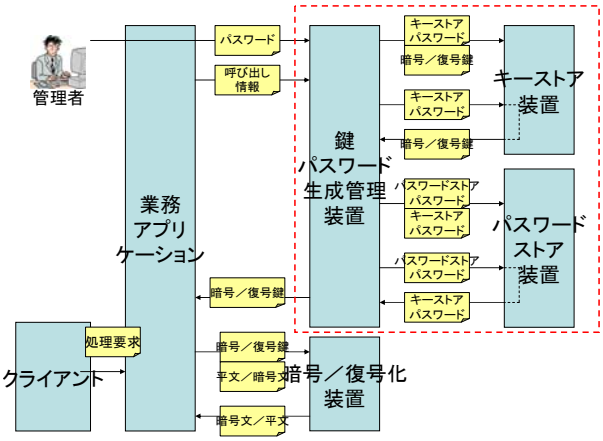

図3　機能構成

アプリケーション、鍵パスワード生成管理装置、キーストア、パスワードストア、暗号／復号装置から構成される。

キーストアはICカードなどパスワードによりアクセス制御を行う鍵用のストレージ、同様にパスワードストアはパスワード用のストレージを指す。

アプリケーションは、初期処理時に管理者（定義ファイル等を含む）から渡されるパスワード情報を用いて鍵取得要求を行う（この際、呼び出し情報としてコールスタックの情報も一緒に渡される）。

鍵パスワード生成管理装置は、アプリケーションのコールスタックとパスワード情報を連結した情報から、新たなパスワード情報（キーストアパスワード）を生成し、本パスワードを用いて、鍵の管理を行う。

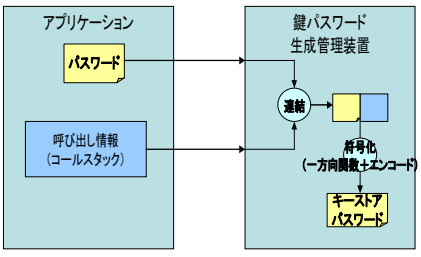

図4　キーストアパスワード生成方式

アプリケーションの初回起動時には、本キーストアパスワードに対応する鍵が登録されていないので、鍵の生成・登録を実施する。以降の起動時には、キーストアパスワードに対応した鍵を取得することができる。

また、鍵情報を更新する場合、鍵情報を取得する場合とはコールスタックの異なる呼び出しとなるため、鍵を保有するアプリケーションがキーストアパスワードを取得できるように、鍵情報を符号化した値をパスワード（パスワードストアパスワード）として、キーストアパスワードを管理する。

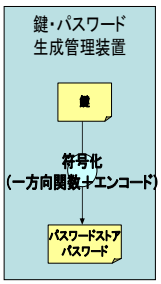

図5　パスワードストアパスワード生成方式

これにより、鍵を取得できるアプリケーションはキーストアへのアクセスおよび鍵の更新が可能になる。

4．評価

鍵管理において、パスワードの漏洩が課題となっていた。従来技術では、アプリケーションに埋め込んだ乱数から共通鍵を生成して暗号化に用いるという方式が提起されているが、埋め込み情報が危殆化する問題や、管理者／開発者の内部犯行による漏洩に対処できないといった問題があった。

本方式を用いると、アプリケーションに無意識に埋め込んだ情報を利用しているため、ソースおよび管理者のパスワードを入手しても、パスワード解析を行うことなく、キーストアパスワードを入手することができなくなり、ソースおよび管理者パスワードの漏洩に起因する管理者／開発者の内部犯行を防ぐことが可能となった。

5．今後の課題

本論文にて管理者／開発者にも漏洩を困難にするコールスタックを用いた鍵管理の方式を示した。今後は、更新時の処理手順の簡便化や、システム試験時の運用について検討を進める予定である。

参考文献

[1] 現状での一般的な情報漏えい対策 (IPA, 2007/10/1)
[2] 共通鍵生成方法(株式会社シーフォーテクノロジー, 特開 2003-304237)
[3] 情報漏えいインシデント対応方策に関する調査報告書(IPA, 2007 年 5 月)