BlackBerry デバイスでの Ultra LightJ のセキュリティ

目次

はじめに

このドキュメントでは、BlackBerry デバイスで発生するセキュリティ関連の問題について紹介すると同時に、Ultra LightJ アプリケーションの開発者向けに、実際のセキュリティ・ニーズに基づいて問題を解決する方法について説明します。データベースの配置先については、Persistent Store (Persistent Object Store)、オンボード・フラッシュ・メモリ (file:///store/ という指定でアクセス。以下、「フラッシュ・ファイル・ストア」)、およびメディア・カードを取り上げます。
このドキュメントは、BlackBerry Device OS 4.2 ～ 5.0 上で SQL Anywhere 11.0.1 リリースを使用する場合に基づいています。ただし、執筆時点での Ultra LightJ の既存バージョン全般に適用されます。暗号化関連の知識や開発者としての経験などは特に問いません。ただし、記載されているコード・サンプルを利用するには、Ultra LightJ API および BlackBerry の開発環境に習熟している必要があります。

基本的なセキュリティ

最も基本的なレベルのセキュリティでは、同僚の給与明細書が開いたまま放置されていれば覗かずにいられない「覗き屋」的な人物による情報の盗用を防止します。同時に、データ自体の価値についても考慮します。情報を狙う人物にとって価値の低いデータであれば、講じるセキュリティ対策は最低限のもので済みます。ただし、子供が遊び半分に BlackBerry デバイスを操作した場合など、意図しない誤操作でデータが破損するような事態は避ける必要があります。さらに、Ultra LightJ データベースと Mobile Link サーバを同期する環境では、不正なデータが統合データベースにアップロードされるリスクも考慮する必要があります。

不正なデータに備えて、BlackBerry デバイスはデバイス・パスワードで保護する必要があります。同時に、Ultra LightJ データベースもデフォルト以外のパスワードで保護する必要があります。データベース・パスワードを設定することで、他の Ultra LightJ アプリケーションによるデータベースへの偶発的なアクセスを防止できます。BlackBerry の Persistent Store に配置されたデータベースは、タイプチェック・メカニズムにより、非 Ultra LightJ アプリケーションから保護されます。ただし、フラッシュ・ファイル・ストアやメディア・カードに配置されたファイルは他のアプリケーションによる閲覧が可能な状態です。

機密データの保護

さらに一段階上のレベルのセキュリティでは、マルウェアや場当たり的な盗用 (デバイスやメディア・カードを盗んでから、何か価値のある情報がないか確認するような行為) からデータを保護します。BlackBerry デバイスの基本的なセキュリティ対策 (デバイス・パスワード) は非常に有効であり、多くの場合、Persistent Store への不正なアクセスを防ぐ効果があります (ただし、OS を再インストールする方法や、パスワード攻撃を繰り返す方法を採れば、Persistent Store のセキュリティを破ることはできます)。これに対して、基本レベルのセキュリティでは、フラッシュ・ファイル・ストアやメディア・カードに保存されているデータは完全に無防備な状態でセキュリティ・リスクに晒されています。データの価値は、盗用に要する労力の量に応じて決まります。たとえば、クレジット・カード番号や、なりすまし犯罪に利用できる情報には、高い価値があると言えます。また、法的な個人情報保護の要件 (個人の医療記録など) や、顧客情報や企業情報の漏えいが発生した場合に企業が受ける損害の度合いについても考慮する必要があります。
Ultra LightJ アプリケーションには、この種の機密データ保護に対応するオプションがいくつか用意されています。

1. 機密データ限定の暗号化

クレジット・カード番号、社会保障番号、運転免許証番号などについては、データベースに格納する前に直接、暗号化する方法が考えられます。このアプローチでは CPU に対する負荷は最小限に抑制されますが、プログラム的には問題の種になります。たとえば、暗号化した値は順序付けに基づくインデックスには使用できません (ルックアップ自体は暗号化された値でも実行できます)。カラム値に対するアクセスは、下記で説明する Encryptor クラスとよく似た、シンプルなアクセサ・クラス (コードを含まな

い) を介して行われます。対応として厄介な部分は、アクセサ・クラスを介してすべての値に確実にアクセスする処理だけです。

2. データベース全体の暗号化

前述の方法よりも簡単で包括的なアプローチとしては、Ultra LightJ EncryptionControl インタフェースを実装するクラスを定義する方法が挙げられます (下記のサンプルを参照)。このクラスはデータベースを作成する Ultra LightJ 構成オブジェクトに渡されて、以降のデータベースへの接続全般で使用されます。Ultra LightJ では、このクラスに基づいて、あらゆるデータベース・ページの暗号化および復号化が行われます。

```
/** データベースの暗号化／復号化を実装するクラス
*/

/** ページ番号に基づいた初期化ベクトルの生成
* @param page_no ページ番号
* @return ページの初期化ベクトル
*/

/** データベースに格納されているページの復号化
* @param page_no 復号化対象のページ番号
* @param src データベースから読み込まれた暗号化ソース・ページ
* @param tgt 非暗号化ページ (メソッドによって設定される)
*/

/** データベースに格納されているページの暗号化
* @param page_no 暗号化対象のページ番号
* @param src 非暗号化ソース
* @param tgt データベースに書き込まれる暗号化ターゲット・ページ (メソッドによって設定される)
*/

/** パスワードによる暗号化制御の初期化
* @param password パスワード
*/

/** 暗号化エラーのエラー・クラス
*/

/**
* 例外に関連するエラー・コードの取得
* @return 例外に関連するエラー・コード (クラス冒頭のリストから取得)
*
*/

/** 例外の原因になった例外の取得 (存在する場合)
     * @return null (原因の例外が存在しない場合)、例外の原因になった例外 (原因
の例外が存在する場合)
*/
/** SQL 文字列に含まれるエラーのオフセットの取得
     * @return (-1) エラー・メッセージに関連する SQL 文字列が存在しない場合
     * エラーが発生した文字列に含まれる (base 0) オフセット (エラー・メッセージに関連
する SQL 文字列が存在する場合)
*/
```

利点

紹介した 2 つの方法はともに適切なセキュリティを提供すると同時に、柔軟な対応を可能にします。作成したデータベースは、フラッシュ・ファイル・ストアへの配布やコピーを目的として、BlackBerry の USB 大容量メディア・モードにより、データ・セキュリティを確保しつつ、メディア・カードに事前ロードできます。このオプションを使用すると、アプリケーションでバックグラウンド同期の実行も可能になります。

BlackBerry の暗号化モード (フラッシュ・ファイル・ストアおよびメディア・カード用) の使用

最後に紹介するアプローチ (対象はフラッシュ・ファイル・ストアおよびメディア・カードにデータベースを配置する事例のみ) では、BlackBerry デバイスの [Options] > [Memory] > [Encryption Mode] (または同等の IT ポリシー) 設定に基づいて、フラッシュ・ファイル・ストアまたはメディア・カードに配置するデータベースのセキュリティ・レベルを指定します。このアプローチでは、アプリケーションでデータベースへのアクセスが必要になるたびに、デバイスをアンロックする必要があります。この設定の値の一部については、設定を使用する BlackBerry 上にデータベースを作成する必要があります。また、このセキュリティが有効な場合、メディア・カードを配布する方法や、USB マス・ストレージ・モードを使用する方法でもデータベースはインポートできなくなります。セキュリティ・パスワードによる暗号化モードは移植性 (BlackBerry デバイス間) に優れている一方で、アルゴリズムは公開されておらず、互換性のある暗号化ファイルを作成するデスクトップ・ツールも存在しません (Media Manager で扱えるのはメディア・ファイルだけであり、USB マス・ストレージ・モードはデバイス間でやり取りされるファイルの暗号化／復号化には対応していません)。万能なセキュリティ・モードを実現するには、persistentContentStateChanged() メソッドで PersistentContentListener をアプリケーションに実装し、デバイスがロックされたときにデータベースへの接続をすべて切断する必要があります (リスナに対する強参照により、メモリ・リークやその他の問題がアプリケーションで発生し、適切に解決されない可能性があります)。ロックが完了し、オペレーティング・システムによってデバイス・キーが解除された後では、データベース・アクセス (Mobile Link サーバとのバックグラウンド同期を含む) は、デバイスがアンロックされるまで実行できません。

暗号化は簡単にオフに切り替え可能

注意すべき点としては、暗号化を設定せずにデータベースを作成した場合、Encryption Mode (または同等の IT ポリシー) 設定に変更しても、データベースのセキュリティ・レベルには影響がないということが挙げられます。何らかの暗号化を設定してデータベースを作成して、(カードはデバイスに装着されている状態で) Encryption Mode を None に変更した場合、そのデータベースはあらゆる BlackBerry データベースで閲覧できるようになります (暗号化ラッパでラップされていても同様です)。

重要データと確信的な盗用

情報窃盗犯の中でも優れた技術を持つ人物は、デバイスがロックされていても、その内部に侵入して、メモリに格納されている暗号化キーを見つけ出すことができます。この種の高度な攻撃は、コールド・ブート攻撃 (http://en.wikipedia.org/wiki/Cold_boot_attack) として知られています。具体的には、メモリへの給電を維持したままデバイスに侵入し、デバイスの暗号化キーを入手してから、Persistent Store やその他の形態のメモリに格納されている情報全般にアクセスします。BlackBerry OS では、デバイスがロックされた時点やデスクトップ PC に接続された時点でデバイス暗号化キーへのアクセスを無効にすることにより、アプリケーションを保護しています。

Ultra LightJ アプリケーションでは、データベースへの接続をすべて切断して、データベースへの接続時に使用される構成オブジェクトへの参照をすべて破棄する MemoryCleanerListener を実装することにより、アプリケーション自体を保護しています。このメモリ・クリーナは、デバイスのロック後、指定された時間が経過してから実行するよう構成できます。次のような方法を採ることで、セキュリティで完全に保護されたソリューションを実現できます。具体的には、デバイスのアンロック後、ユーザがデータベース・パスワードを再入力す

る必要がありません。

- アプリケーションで Ultra LightJ データベースの暗号化用に EncryptionControl を実装する
- 初回実行時に、アプリケーションからユーザに対してデータベース・パスワードの入力を要求する
- 入力されたパスワードは PersistentContent.encode() によってセキュリティ保護された上で Persistent Store に格納される
- 接続時に、アプリケーションでは、上記のパスワードと EncryptionControl に基づいて、データベースへの接続が確立される (構成オブジェクトやパスワードに対する不要な参照は保持されない)
- MemoryCleanerListener が呼び出された時点で、アプリケーションでは、データベースへの接続はすべて切断され、構成、EncryptionControl オブジェクト、およびパスワードのコピー (Persistent Store に格納されているもの以外) はすべて破棄される
- ユーザがデバイスをアンロックした時点で、アプリケーションは Persistent Store からパスワードを取得して、データベースに再接続できるようになる

MemoryCleanerListener と PersistentContentListener を実装して、デバイスをロックする時間帯と完全にセキュリティ保護する時間帯の時間間隔を定義することで、バックグラウンド同期に最適な時間帯が明確になります。

前述の方法でセキュリティを実現している場合、デバイスの [Options] > [Memory] > [Encryption Mode] (および関連 IT ポリシー) がオフになっていることと、前述の機能全般に対応するデータベースがデスクトップに作成済みであることを確認してください。

注意すべき点として、RIM のソリューションにも 1 つ弱点があります。アプリケーションでは「チケット」、つまりデバイス・キーが (アプリケーションで解除されない限り) メモリ内に保持されているということです。問題のあるアプリケーションが 1 つ存在するだけで、デバイスは脆弱な状態に陥る可能性があります。ユーザの視点では、ロックされている画面上の錠前のアイコンに注意を払う必要があります。錠前に鍵がかかるのは、キーがクリーンアップされたときだけです。

Persistent Store のセキュリティ・オプション

デバイスの [Options] > [Security Options] > [General Settings] > [Content Protection] (および関連 IT ポリシー設定) は、デバイスの Persistent and Runtime Store に対してのみ影響します。**コンテンツの保護は自動的に適用されるわけではありません**が、アプリケーションでデータ関連の処理が適正に行われていることが前提になります。現時点では、この設定のサポートは Ultra LightJ には用意されていません。

デバイスの [Options] > [Memory] > [Encryption Mode] は、[Options] > [Security Options] > [General Settings] > [Content Protection] の設定からは独立していることに注意してください。Encryption Mode では、フラッシュ・ファイル・ストアとメディア・カードのセキュリティを管理します。

IT ポリシーとユーザ設定の比較

今回の記事で言及したデバイスのオプション全般は、デバイスとのペアを構成する BES (BlackBerry Enterprise Server) の IT ポリシーによって上書きされる場合があります。具体的には、最も厳格な設定 (ユーザまたは IT) が優先されます。したがって、対応する IT Policy の設定が厳格な場合、それよりも緩やかな Encryption Mode をユーザ側で選択することはできません。

Ultra LightJ 12 についての補足

特に BlackBerry のような処理速度が低い CPU 環境でのパフォーマンス向上を目的として、Ultra LightJ での EncryptionControl の処理方法は改良されています。

- Ultra LightJ では、データおよびシステムの重要なページのみが暗号化されるようになりました (ただし、ファイルとして保管された blob は対象外。下記の注を参照)。
- インタフェースの EncryptionControl.decrypt() メソッドで、復号化に必要なバイト数を指定

するパラメータが取得されるようになりました。

実装が問題なく機能するよう、特に decrypt() メソッドに注意しながら、EncryptionControl インタフェース関連の最新ドキュメントをよくお読みください。

法的注意