3.互換性、課題と対策
～XML暗号化ツール検証報告～

XMLコンソーシアムWeek　2010年03月16日
セキュリティ部会
大沼啓希（日本アイ・ビー・エム株式会社）

ご説明内容

- Webサービス実証部会の皆様へ
- XML暗号化検証報告作成の経緯
- 暗号化ツール検証報告書 (目次)
- 暗号化ツール検証報告書 (1章)
- 検証プログラムのフロー (2章)
- 暗号化ツール検証報告書 (3章検証結果)
- 課題と対処
- 終わりに

Webサービス実証部会の皆様へ

- 有難うございました(2009年5月12日コンソーシアムWeek)

- 詳細な報告書は、
  - 2010年3月16日公開
  - http://www.xmlconsortium.org/public_doc/securitytool/


暗号化ツール検証報告書

2010 年 01 月 13 日
XML コンソーシアム
セキュリティ部会 Web サービス実証部会

XML暗号化検証報告作成の経緯

- 2009/06/04 セキュリティ部会2009年度第1回ミーティング
  - 再現テストの作業開始
  - ストーリー確認及びテスト結果の画面キャプチャー作業完了(2009/8/27)
- 2009/09/27 ドラフト前版完成
- 2009/10/07 ドラフト第一版完成
- 2009/11/02 セキュリティ部会2009年度第6回ミーティング
  - 部会レビュー
- 2009/11/12 検証報告書を実証部会と連名に
- 2009/12/18 最終部会レビュー
- 2010/01/13 最終版（運営委員会・理事会用）完成
- 2010/01/15 運営委員会レビュー
- 2010/02/20 理事会審議終了
- 2010/0x/yy 報告書公開

本日公開

http://www.xmlconsortium.org/public_doc/securitytool/

暗号化ツール検証報告書

2010年1月

XMLコンソーシアム

セキュリティー部会　Webサービス実証部会

1. 初めに

XMLコンソーシアム セキュリティ部会では、2008年のMOF2008 (Manufacturing Open Forum 2008)の実証デモシステムにおけるセキュリティについての検討を実施し、その報告書を作成した。

その検討を深める意味で、実際のXMLインスタンスを使った複数の暗号化パターンによる暗号化・復号の検証、複数の暗号化ツールにおける暗号化・復号の検証、複数のツール間での復号互換性の確認を実施し、2009年5月12日のXMLコンソーシアムWeekにて検証結果の概要を報告した。

また、検証に必要となるコーディングやテスト作業においては、Webサービス実証部会の協力を得て、共同して検証活動を実施した。

本報告書は、その検証結果の詳細報告書である。

1.1. 検証の目的

2008年の活動では、Apache XML Securityのサンプル・プログラムを利用して、暗号化・復号のテストを実施した。その中で単一のXMLデータ構造のテストしかしておらず、実際に暗号化を実施する上での*パターンとしては不足している事が課題として残った。*

更に暗号化・復号ツールとして、Apache XML Securityだけではなく、*他の暗号化ライブラリーによる複数のツールでの検証とその互換性検証も追加の課題として残った。*本検証では、上記課題を解決することを目的としている。

1.2. 検証期間と参加者

検証作業は2009年2月から開始し、4月に完了した。
1月 9日－第6回部会にて、暗号化ツールの継続検証を決定
2月20日－第7回部会にて、Webサービス実証部会との協業を決定
3月18日－第8回部会にて、暗号化の要件及びその対象と方式を決定
4月16日－第9回部会にて、検証結果確認と報告内容を決定
5月12日－XMLコンソーシアムWeekにて検証結果を報告

検証メンバーは、以下の通り。(順不同)

- 荒本道隆 (アドソル日進)
- 上村準也 (キヤノンソフト情報システム)
- 大沼啓希 (日本アイ・ビー・エム株式会社)
- 松永豊 (東京エレクトロン デバイス株式会社)

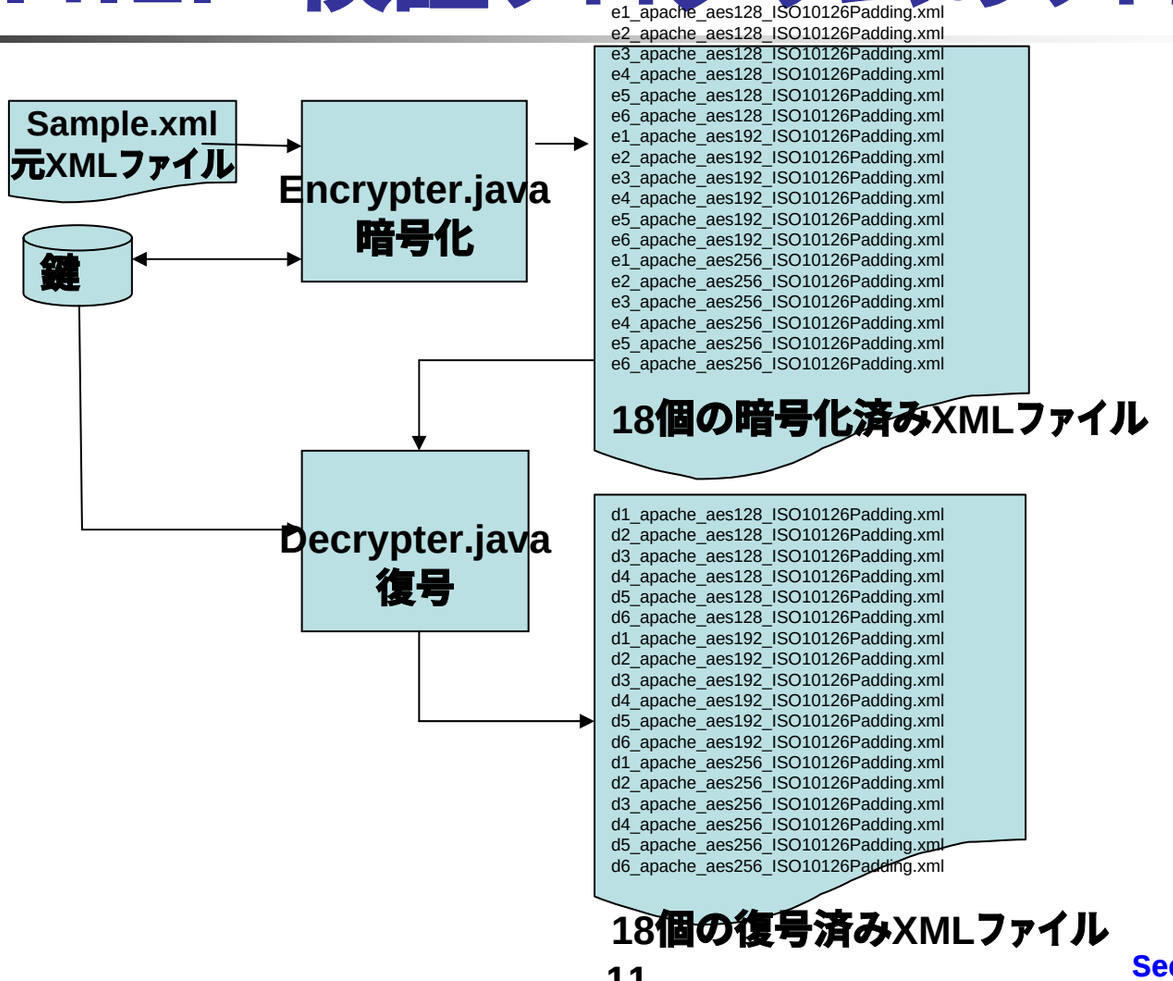
1.3. 検証の範囲

「初めに」で説明しているように今回の検証では、「使用するXMLインスタンス」の*6つのパターンの対象要素や構造の暗号化・復号と、それらを各々以下の3つのライブラリーでの稼動検証及び各ライブラリー間での相互復号検証を実施した。*

(1) **Apache XML Security**
(2) **IBM Java SDK 6.0 (以下IBM Java6と略す)のJSR-106**
(3) **Microsoft .NET Framework (以下.NETと略す)**

検証には暗号方式として**AES+TripleDES**を使用した。
AESの鍵の長さは**3種類(128,192,256Bit)**で検証した。
ブロック暗号を使用している**AES**のパディング方式は**W3C**の仕様で決まっているが、今回相互復号の検証の為にそれが変更可能であれば変更して検証を行った。
これらの検証だけでも、「検証結果」で説明する**3**つの課題(**AES**の鍵長、パディング方式の相違、**KeyName**)が浮かび上がり、それらの対応策を検討し適用することができた。

2.4.1. 検証プログラムのフロー



検証結果

3.1.1. 初期状態

初期状態では、以下の結果が得られた。鍵長が192及び256ビットの場合、Apache XML SecurityとJSR-106では暗号化が Illegal key size or default parametersとなり、同様に復号でも、Apache XML SecurityとJSR-106で同じエラーとなった。また、JSR-106では、他の方法で暗号化されたXMLファイルは復号できなかった。

		復号		
暗号化	AES鍵長	Apache	JSR-106	.NET
Apache	128	○	×	○
Apache	192	NA	NA	NA
Apache	256	NA	NA	NA
JSR-106	128	×	○	×
JSR-106	192	NA	NA	NA
JSR-106	256	NA	NA	NA
.NET	128	○	×	○
.NET	192	NG	NG	○
.NET	256	NG	NG	○

検証結果

3.1.2. 最大暗号化強度の制限を解除

Apache XML SecurityとJSR-106でも鍵長が192及び256ビットの場合に暗号化できるように、以下のサイトから最大暗号化強度の制限を解除してあるポリシーファイルをダウンロードし、それに置換した所、以下の結果が得られた。

		復号		
暗号化	AES鍵長	Apache	JSR-106	.NET
Apache	128	○	×	○
Apache	192	○	×	○
Apache	256	○	×	○
JSR-106	128	×	○	×
JSR-106	192	×	○	×
JSR-106	256	×	○	×
.NET	128	○	×	○
.NET	192	○	×	○
.NET	256	○	×	○

検証結果

3.1.3. PaddingのISO10126からPKCS5への変更（Apache,.NET）

Apache XML SecurityとMicrosoft .NET Frameworkに対して、Padding方式をISO10126とPKCS5の両方を使用できるようにしたところ、3種類の鍵長において以下の結果を得た。これにより、IBM Java6のJSR-106は、PKCS5のパディング方式しか扱えないことが判明した。

		復号		
暗号化	Padding	Apache	JSR-106	.NET
Apache	ISO10126	○	×	○
Apache	PKCS5	○	○	○
JSR-106	PKCS5	○	○	○
.NET	ISO10126	○	×	○
.NET	PKCS5	○	○	○

課題と対処

3.2. AES鍵長の課題

Sun Java6とIBM SDK 6は、どちらもAESで使用できるビット数が128までである。XML暗号のW3C勧告によればAES-256は必須（REQUIRED）なので、Sun Java6とIBM Java6にそれぞれJCE（Java Cryptography Extension）をインストールして最大暗号化強度の制限を解除し、AES-256を使用可能にする必要がある。.NETではデフォルトでAES-256が使用可能になっている。

課題と対処

3.3. パディング方式の相違の課題

例えば、暗号化に際して 1, 2, 3, 4 バイトのパディングが必要になった場合、ISO10126 形式では以下の様に、元のデータの最後にダミーのバイトを付加する。

データ	データ	データ	データ	0x01
データ	データ	データ	0x??	0x02
データ	データ	0x??	0x??	0x03
データ	0x??	0x??	0x??	0x04

PKCS5 形式でのパディングで以下の様になる。

データ	データ	データ	データ	0x01
データ	データ	データ	0x02	0x02
データ	データ	0x03	0x03	0x03
データ	0x04	0x04	0x04	0x04

課題と対処

3.3. パディング方式の相違の課題

パディング部分が 1 バイトの場合、ISO10126 と PKCS5 の2つの形式でパディングを含めた結果が同じとなり、パディング形式による違いが無くなるため、正しく復号できる。

パディング部分が 2 バイト以上の場合は、ISO10126形式のものをPKCS5として見ると、最後のバイト以外が異なっているために正しくないと判断され、復号できない。

ただし、PKCS5形式のものをISO10126として見ると、最後のバイト以外は無視するために正しいとみなされるので、IBM Java6 の JSR-106で暗号化した出力はApache XML SecurityとMicrosoft .NET Frameworkで復号できる。

課題と対処

3.4. KeyNameの課題

暗号化したXMLに
「KeyInfo/EncryptedKey/KeyInfo/KeyName」という要素は必須ではないが、今回の検証では.NETでの復号のコードを簡略化するために、KeyName 要素を付けた形式で行った。

4. 終わりに

この検証は、ＸＭＬインスタンスの6種の暗号化パターンと複数の暗号化ツールの利用が当初のゴールであった。容易にこれらの検証が完了できので、相互復号検証を加えたところ、3つの課題が浮かび上がった。 それらは、比較的短期間にこれらの課題を全て解決することができた。

しかしながら、全ての利用パターンや環境を検証してはいない。 例えば、

暗号化済みのXMLインスタンスの再暗号化、複数キーによる別要素・別構造の暗号化やその復号処理順序、複数OS環境下での複数暗号化ライブラリーの利用、複数の鍵の受け渡し（交換）方法や鍵の管理、SOAPへの適用(WS-Security)、処理性能やコーディング量の比較・・・等のようなカバーしていない課題がまだ積み残されている。