HITACHI
Inspire the Next

Cosminexus V9 アプリケーションサーバ Web サービスセキュリティ構築ガイド

解説・手引・文法書

3020-3-Y24-50

■ 対象製品

マニュアル「アプリケーションサーバ & BPM/ESB 基盤 概説」の前書きの対象製品の説明を参照してください。

■ 輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。
なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

AMD は，Advanced Micro Devices, Inc.の商標です。
HP-UX は，Hewlett-Packard Development Company, L.P.のオペレーティングシステムの名称です。
IBM，AIX は，世界の多くの国で登録された International Business Machines Corporation の商標です。
Itanium は，アメリカ合衆国およびその他の国における Intel Corporation の商標です。
Linux は，Linus Torvalds 氏の日本およびその他の国における登録商標または商標です。
Microsoft は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Oracle と Java は，Oracle Corporation 及びその子会社，関連会社の米国及びその他の国における登録商標です。
Red Hat は，米国およびその他の国で Red Hat, Inc. の登録商標もしくは商標です。
SOAP（Simple Object Access Protocol）は，分散ネットワーク環境において XML ベースの情報を交換するための通信プロトコルの名称です。
UNIX は，The Open Group の米国ならびに他の国における登録商標です。
Windows は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Windows Server は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
Windows Vista は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
その他記載の会社名，製品名は，それぞれの会社の商標もしくは登録商標です。
This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

■ 発行

2015 年 4 月 3020-3-Y24-50

■ 著作権

変更内容

変更内容(3020-3-Y24-50) uCosminexus Application Server 09-70，uCosminexus Application Server(64) 09-70，uCosminexus Client 09-70，uCosminexus Developer 09-70，uCosminexus Service Architect 09-70，uCosminexus Service Platform 09-70，uCosminexus Service Platform(64) 09-70

追加・変更内容	変更個所
記載内容は変更なし（リンク情報だけを変更した）。	－

uCosminexus Application Server 09-60，uCosminexus Application Server(64) 09-60，uCosminexus Client 09-60，uCosminexus Developer 09-60，uCosminexus Service Architect 09-60，uCosminexus Service Platform 09-60，uCosminexus Service Platform(64) 09-60

追加・変更内容	変更個所
記載内容は変更なし（リンク情報だけを変更した）。	－

単なる誤字・脱字などはお断りなく訂正しました。

はじめに

このマニュアルをお読みになる際の前提情報については，マニュアル「アプリケーションサーバ & BPM/ESB 基盤 概説」のはじめにの説明を参照してください。

目次

1 Web サービスセキュリティの概要

この章では，Web サービスセキュリティとは何か，また，Web サービスセキュリティと SOAP，および XML セキュリティとの関係について説明します。また，Application Server が提供する Web サービスセキュリティ機能についても説明します。

1.1　マニュアルの説明，プレフィクスと名前空間 URI

このマニュアルの対象となる Web サービスセキュリティ機能については，アプリケーションサーバを構成する次のプログラムプロダクトで提供されています。

- Application Server
- Service Platform
- Developer
- Service Architect

なお，XML 署名・暗号処理機能については，マニュアル「XML Security - Core ユーザーズガイド」を参照してください。

このマニュアルで使用している表記について説明します。

文法で使用している記号

このマニュアルで使用している記号を次のように定義します。

記号	意味
＜＞	＜＞で囲まれた部分は状況に応じて変化する内容であることを示します。

プレフィクスと名前空間 URI の対応

このマニュアルで使用するプレフィクスと名前空間 URI の対応を次に示します。特に断りがないかぎり，次のプレフィクスを使用します。

プレフィクス	名前空間 URI
ds	http://www.w3.org/2000/09/xmldsig#
soap	http://schemas.xmlsoap.org/wsdl/soap/
soapenv	http://schemas.xmlsoap.org/soap/envelope/
soapenv12	http://www.w3.org/2003/05/soap-envelope
sp	http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702
wsdl	http://schemas.xmlsoap.org/wsdl/
wsp	http://www.w3.org/ns/ws-policy
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd
xsd	http://www.w3.org/2001/XMLSchema

1.2　Web サービスセキュリティとは

Web サービスセキュリティとは，広義には Web サービスを安全に実現するためのセキュリティ技術全般を指します。現在，代表的な Web サービスセキュリティは，XML 署名および XML 暗号を利用した XML セキュリティです。また，Web サービスの実現には，SOAP を使用するのが一般的です。この節では，Web サービスセキュリティと SOAP，および XML セキュリティとの関係について説明します。

1.2.1　Web サービスセキュリティと SOAP との関係

Web サービスの多くは，SOAP メッセージを送受信することで実現します。SOAP メッセージは XML 形式のデータです。そのため，ネットワークを経由して，そのまま SOAP メッセージを送信した場合，データの内容を改ざんされたり，カード番号などの重要な情報を第三者に盗聴されたりしてしまうおそれがあります。このような改ざんや盗聴を防止するための技術が Web サービスセキュリティです。

Web Services - Security は，Component Container の SOAP 通信基盤または JAX-WS エンジンに組み込んで，Web サービスセキュリティに対応した SOAP メッセージを送受信する場合に使用します。

1.2.2　Web サービスセキュリティと XML セキュリティとの関係

W3C が規定している XML 署名および XML 暗号の仕様を XML セキュリティと呼びます。XML セキュリティを利用すると，SOAP メッセージに XML 署名を付与したり，SOAP メッセージを暗号化したりできます。代表的な Web サービスセキュリティは，OASIS で規定されているもので，XML セキュリティを利用した安全な Web サービスを実現します。

Web Services - Security は，XML 署名・暗号処理機能を利用した Web サービスセキュリティ機能を提供します。XML 署名・暗号処理機能については，マニュアル「XML Security - Core ユーザーズガイド」を参照してください。

1.3　Application Server が提供する Web サービスセキュリティ機能

Application Server では，Web Services - Security を利用して Web サービスセキュリティを実現します。Web Services - Security が提供する機能のうち，SOAP メッセージの完全性や秘匿性などのセキュリティを提供する機能を Web サービスセキュリティ機能と呼びます。

Component Container で Web サービスセキュリティ対応の SOAP アプリケーションまたは Web サービスを開発する場合，Web Services - Security を使用します。SOAP アプリケーションの開発の詳細はマニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を，Web サービスの開発の詳細はマニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

参考

このマニュアルでは，Application Server の SOAP アプリケーション開発支援機能を使用して開発したプログラムを SOAP アプリケーション，Application Server の JAX-WS 機能を使用して開発したプログラムを Web サービスと呼びます。ただし，Web サービスという用語については，SOAP アプリケーションも含めた Web サービス全般を指している場合もあります。

1.3.1　SOAP メッセージの完全性を保証する

Web サービスセキュリティ機能は, SOAP メッセージの完全性を保証します。SOAP メッセージが送受信中に改ざんされないよう，SOAP メッセージの完全性を保証するための方法として，署名を利用する方法があります。

Web サービスセキュリティ機能を使用すると, SOAP メッセージに署名を付与できます。SOAP メッセージに署名が付与されている場合, SOAP メッセージを受信したときに署名を検証することによって, SOAP メッセージが送受信中に改ざんされていないかどうかを調べられます。また，SOAP メッセージに証明書が付与されている場合は，Web サービスセキュリティ機能で証明書も検証できます。

1.3.2　SOAP メッセージの秘匿性を保証する

Web サービスセキュリティ機能は, SOAP メッセージの秘匿性を保証します。SOAP メッセージが送受信中に第三者によって盗聴されないよう，SOAP メッセージの秘匿性を保証するための方法として，SOAP メッセージを暗号化する方法があります。

Web サービスセキュリティ機能を使用すると，必要な部分だけを指定して SOAP メッセージを暗号化できます。Web サービスセキュリティ機能では，暗号化に XML 暗号を使用します。暗号化することによって，SOAP メッセージの送受信中に第三者にメッセージの内容を盗聴されるおそれがなくなります。

1.3.3　SOAP メッセージの認証をサポート

Web サービスセキュリティ機能は，SOAP メッセージの認証および証明書の検証をサポートしています。SOAP メッセージの送信者を特定する必要がある場合は，ユーザー名やパスワードを SOAP メッセージに含めるように設定できます。

2 開発または実行に必要な製品

この章では，Web サービスセキュリティ機能の前提 OS および前提プログラムを，開発時と実行時に分けて説明します。また，開発時と実行時のプログラム構成例をそれぞれ紹介します。

2.1　開発に必要な製品

この節では，Web Services - Security が提供する Web サービスセキュリティ機能を組み込んで SOAP アプリケーションまたは Web サービスを開発する場合の，前提 OS および前提プログラムについて説明します。また，開発時のプログラム構成例についても説明します。

なお，開発に SOAP アプリケーション開発支援機能を使用した場合は，実行環境にも SOAP 通信基盤を利用する必要があります。同様に，開発に Application Server の JAX-WS 機能を使用した場合は，実行環境では JAX-WS エンジンを利用する必要があります。

2.1.1　開発時の前提 OS

Web サービスセキュリティ機能を組み込んだ SOAP アプリケーションまたは Web サービスを開発する場合の前提 OS については，マニュアル「アプリケーションサーバ & BPM/ESB 基盤 概説」を参照してください。

2.1.2　開発時の前提プログラム

Web サービスセキュリティ機能を使用して SOAP アプリケーションまたは Web サービスを開発する場合に，必要となる前提プログラムを次の表に示します。前提プログラムのバージョンについては「リリースノート」でご確認ください。

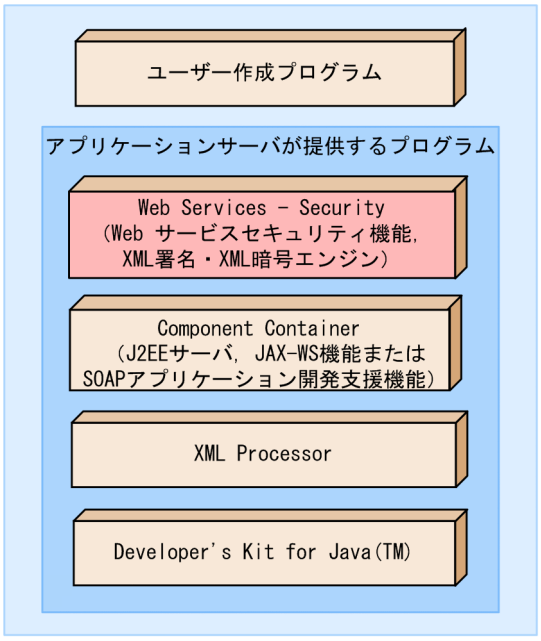
表 2-1　前提プログラム一覧（開発時）

分類	プログラム名
J2EE サーバ，SOAP アプリケーション開発支援機能または JAX-WS 機能	Component Container
Java 2 SDK	Developer's Kit for Java
XML プロセッサ	XML Processor

2.1.3　開発時のプログラム構成例

Web サービスセキュリティ機能を使用して SOAP アプリケーションまたは Web サービスを開発する場合のプログラム構成例を次に示します。

図 2-1　Web サービスセキュリティ機能を組み込む場合のプログラム構成例



2.2　実行に必要な製品

この節では，Web Services - Security が提供する Web サービスセキュリティ機能を使用して開発された SOAP アプリケーションまたは Web サービスを実行する場合の，前提 OS および前提プログラムについて説明します。また，実行時のプログラム構成例についても説明します。

なお，SOAP 通信基盤で実行できるのは，SOAP アプリケーション開発支援機能で開発した SOAP アプリケーションだけです。同様に，JAX-WS エンジンで実行できるのは，Application Server の JAX-WS 機能で開発された Web サービスだけです。

2.2.1　実行時の前提 OS

Web サービスセキュリティ機能を組み込んだ SOAP アプリケーションまたは Web サービスを実行する場合の前提 OS については，マニュアル「アプリケーションサーバ & BPM/ESB 基盤 概説」を参照してください。

2.2.2　実行時の前提プログラム

Web サービスセキュリティ機能を使用して開発された SOAP アプリケーションまたは Web サービスを実行する場合に必要となる前提プログラムを次の表に示します。前提プログラムのバージョンについては「リリースノート」でご確認ください。

表 2-2　前提プログラム一覧（実行時）

分類	プログラム名
Web サーバ	Windows の場合： 　Component Container が動作する Web サーバ※ その他の OS の場合： 　HTTP Server
J2EE サーバ，SOAP 通信基盤または JAX-WS エンジン	Component Container
Java 2 SDK	Developer's Kit for Java
XML プロセッサ	XML Processor

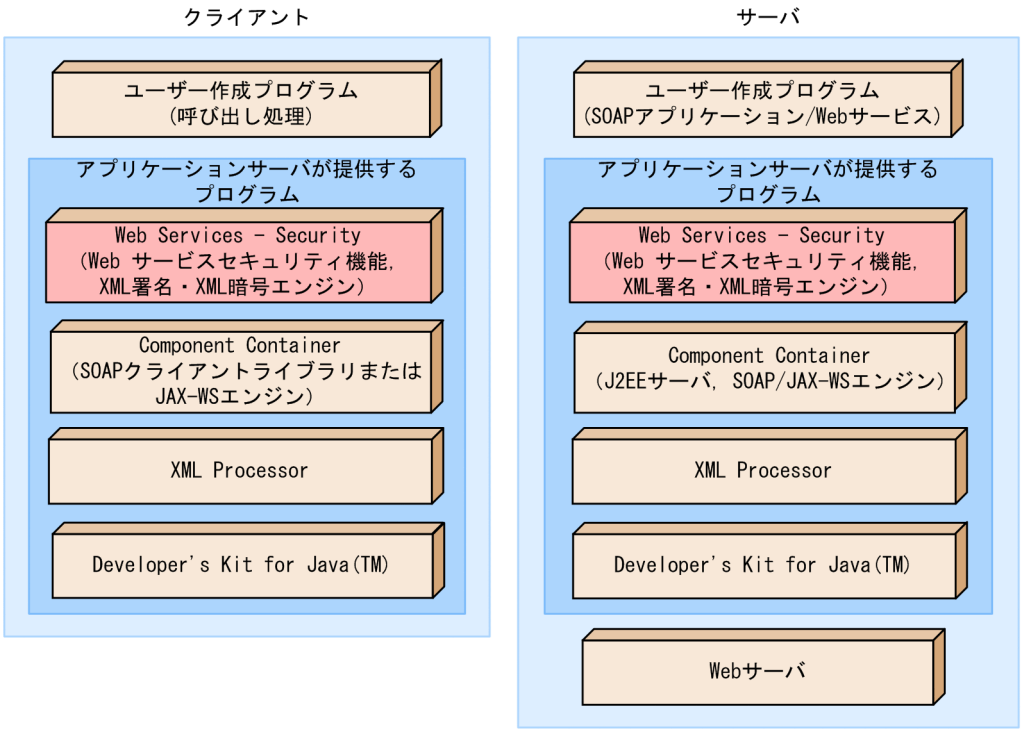
注※

Component Container が動作する Web サーバについての詳細は，マニュアル「アプリケーションサーバ & BPM/ESB 基盤 概説」を参照してください。

2.2.3　実行時のプログラム構成例

Web サービスセキュリティ機能を使用して開発された SOAP アプリケーションまたは Web サービスを実行する場合の，クライアント側とサーバ側のプログラム構成例を次に示します。

図 2-2　Web サービスセキュリティ機能を組み込んだ SOAP アプリケーションまたは Web サービスを実行する場合のプログラム構成例



3 Web サービスセキュリティ機能を使用する

この章では，開発した SOAP アプリケーションまたは Web サービスに対して，Web サービスセキュリティ機能を使用する手順について説明します。

3.1　定義ファイルの設定

Web サービスセキュリティ機能を使用するためには，次に示す二つの定義ファイルに必要な項目を設定する必要があります。

- Web サービスセキュリティ機能定義ファイル（security-config.xml）
- Web サービスセキュリティ方針定義ファイル（policy-config.xml）

これらの定義ファイルは，Web サービス（サーバ側）および Web サービスクライアント（クライアント側）の両方に配置する必要があります。

次に，各定義ファイルの概要を説明します。

3.1.1　Web サービスセキュリティ機能定義ファイル

Web サービスセキュリティ機能定義ファイルは，Web サービスセキュリティ機能の動作を定義するためのファイルです。例えば，SOAP メッセージに署名を付与する場合，署名に用いる証明書を指定します。または，SOAP メッセージを暗号化する場合は，どの暗号化アルゴリズムを用いるのか，などを定義します。

Web サービスセキュリティ機能定義ファイルは，次の要素を持っています。

- BindingConfig
  Web サービスセキュリティの各機能で共通して使用する情報を設定します。
- RequestSenderConfig
  リクエストメッセージ送信時の Web サービスセキュリティ機能を設定します。
- ResponseSenderConfig
  レスポンスメッセージ送信時の Web サービスセキュリティ機能を設定します。
- RequestReceiverConfig
  リクエストメッセージ受信時の Web サービスセキュリティ機能を設定します。
- ResponseReceiverConfig
  レスポンスメッセージ受信時の Web サービスセキュリティ機能を設定します。

クライアントとサーバでは，必要な要素が異なります。

Web サービスセキュリティ機能を使用するために，クライアントおよびサーバで設定する必要がある要素を次の表に示します。

表 3-1　設定が必要な要素（Web サービスセキュリティ機能定義ファイル）

要素名	クライアント	サーバ
BindingConfig	○	○
RequestSenderConfig	○	×
ResponseSenderConfig	×	○
RequestReceiverConfig	×	○
ResponseReceiverConfig	○	×

（凡例）

○：必要

×：不要

Web サービスセキュリティ機能定義ファイルの項目については，「付録 C.1　Web サービスセキュリティ機能定義ファイルの項目」を参照してください。

3.1.2　Web サービスセキュリティ方針定義ファイル

Web サービスセキュリティ方針定義ファイルは，受信した SOAP メッセージを検証するときの方針を定義するためのファイルです。Web サービスセキュリティ機能では，この方針定義ファイルを基に，SOAP メッセージが受信側の意図したものかどうかを検証します。

Web サービスセキュリティ方針定義ファイルは，次の要素を持っています。

- GlobalConfig
  Web サービスセキュリティ機能の方針定義で共通して使用する項目を設定します。
- RequestReceiverConfig
  リクエストメッセージ受信時の方針を設定します。
- ResponseReceiverConfig
  レスポンスメッセージ受信時の方針を設定します。

クライアントとサーバでは，必要な要素が異なります。

Web サービスセキュリティ機能を使用するために，クライアントおよびサーバで設定する必要がある要素を次の表に示します。

表 3-2　設定が必要な要素（Web サービスセキュリティ方針定義ファイル）

要素名	クライアント	サーバ
GlobalConfig	○	○
RequestReceiverConfig	×	○
ResponseReceiverConfig	○	×

（凡例）

○：必要

×：不要

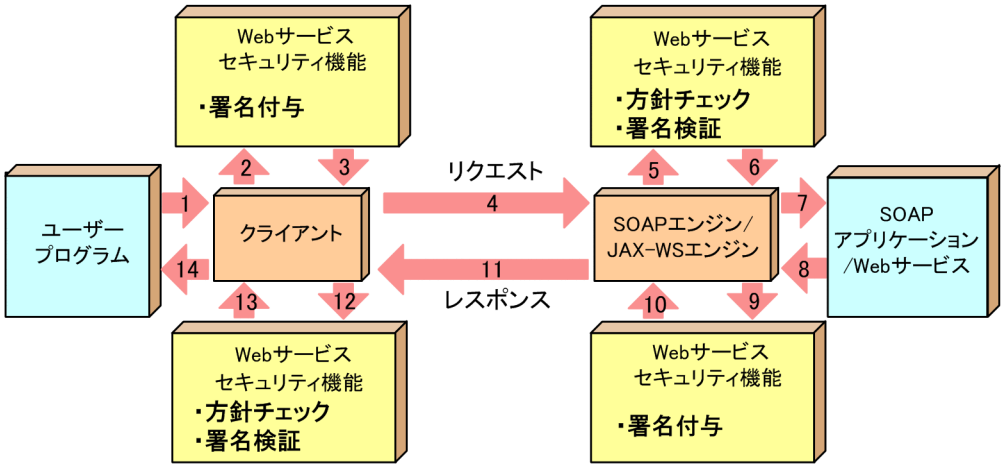
Web サービスセキュリティ方針定義ファイルの項目については，「付録 C.2　Web サービスセキュリティ方針定義ファイルの項目」を参照してください。

3.2　署名付与／検証機能を設定する

署名付与／検証機能は，送信するメッセージに対して，XML 署名を付与したり，受信する SOAP メッセージに付与されている XML 署名を検証したりする機能です。署名付与機能は，メッセージ送信時に，署名検証機能はメッセージ受信時に使用します。

署名付与／検証機能を使用する場合のプログラムの構成を次の図に示します。矢印およびその番号は，送受信するデータの処理の流れを示しています。

図 3-1　Web サービスセキュリティ機能使用時のプログラム構成（署名付与／検証）



署名付与／検証機能を使用する際の，Web サービスセキュリティ機能定義ファイルの設定項目を次に示します。

```
＜送信側＞
SecurityConfig
├── BindingConfig
└── RequestSenderConfig/ResponseSenderConfig
    └── SenderPortConfig
        └── RoleConfig
            ├── BinarySecurityTokenConfig
            └── SignatureConfig
＜受信側＞
SecurityConfig
├── BindingConfig
└── RequestReceiverConfig/ResponseReceiverConfig
    └── ReceiverPortConfig
        └── VerificationConfig
```

署名付与／検証機能を使用する際の，Web サービスセキュリティ方針定義ファイルの設定項目を次に示します。

```
PolicyConfig
└── RequestReceiverConfig/ResponseReceiverConfig
    └── ReceiverPortConfig
        ├── SecurityTokenConfig
        │   └── BinarySecurityTokenConfig
        └── VerificationConfig
```

設定する要素の詳細については，「付録 C　定義ファイルの項目の詳細」を参照してください。

Windows の場合，定義ファイルのサンプルは次のディレクトリに格納されていますので，これらを参考に定義ファイルを作成してください。なお，作成した定義ファイルの格納場所については，「3.9　Web サービスセキュリティ機能の実装手順（SOAP アプリケーション開発支援機能を使用する場合）」を参照してください。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

```
signature/message/client/WEB-INF/classes
signature/message/service/WEB-INF/classes
signature/rpc/client/WEB-INF/classes
signature/rpc/service/WEB-INF/classes
```

また，Windows の場合，コーディングのサンプルは，次のディレクトリに格納されていますので，参考にしてください。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

```
signature/KeyStore
signature/message/client/WEB-INF/classes/messagesampleclient
signature/message/service/WEB-INF/classes/messagesampleservice
signature/rpc/client/WEB-INF/classes/localhost
signature/rpc/service/WEB-INF/classes/localhost
```

署名付与の設定情報について

- 署名付与に必要な情報は，Web サービスセキュリティ機能定義ファイルに指定された設定情報から取得します。
- 署名に用いる秘密鍵は，環境設定ファイルで指定したキーストアファイルを使用します。環境設定ファイルの詳細については，「3.8.2　環境設定ファイルの設定項目」を参照してください。
- 署名に関する仕様のサポート範囲については，「付録 A.2　XML 署名標準仕様のサポート範囲」を参照してください。

署名検証の設定情報について

- 署名検証に必要な情報は，Web サービスセキュリティ機能定義ファイル，および Web サービスセキュリティ方針定義ファイルに指定された設定情報から取得します。
- 証明書検証に用いる証明書は，環境設定ファイルで指定した証明書ファイルを使用します。環境設定ファイルの詳細については，「3.8.2　環境設定ファイルの設定項目」を参照してください。

注意事項

署名付与／検証で使用するキーストアファイルは，読み取り権限を限定するなど，第三者に読み取られないようにご注意ください。

3.2.1　署名を付与する個所をパート名で指定する

署名を付与する個所をパート名という名称で指定します。Web Services - Security で，署名を付与する際に指定できるパート名は，"Body"だけです。"Body"を指定した場合，SOAP メッセージの Body 要素そのものを意味します。

署名個所をパート名で指定する際の，Web サービスセキュリティ機能定義ファイルの設定項目を次に示します。

```
SecurityConfig
└─ RequestSenderConfig/ResponseSenderConfig
   └─ SenderPortConfig
      └─ RoleConfig
         └─ SignatureConfig
            └─ SignatureTarget
```

署名個所をパート名で指定する際の，Web サービスセキュリティ方針定義ファイルの設定項目を次に示します。

```
PolicyConfig
└─ RequestReceiverConfig/ResponseReceiverConfig
   └─ ReceiverPortConfig
      └─ VerificationConfig
         └─ SignatureTarget
```

設定する要素の詳細については，「付録 C　定義ファイルの項目の詳細」を参照してください。

3.2.2　署名を付与する個所を ID 属性で指定する

署名を付与する個所を，ID 属性で指定し，その ID 値が含まれる要素に対して署名します。なお，受信側では ID 属性を検証できません。

署名個所を ID で指定する際の，Web サービスセキュリティ機能定義ファイルの設定項目を次に示します。

```
SecurityConfig
└─ RequestSenderConfig/ResponseSenderConfig
   └─ SenderPortConfig
      └─ RoleConfig
         └─ SignatureConfig
            └─ SignatureTarget
```

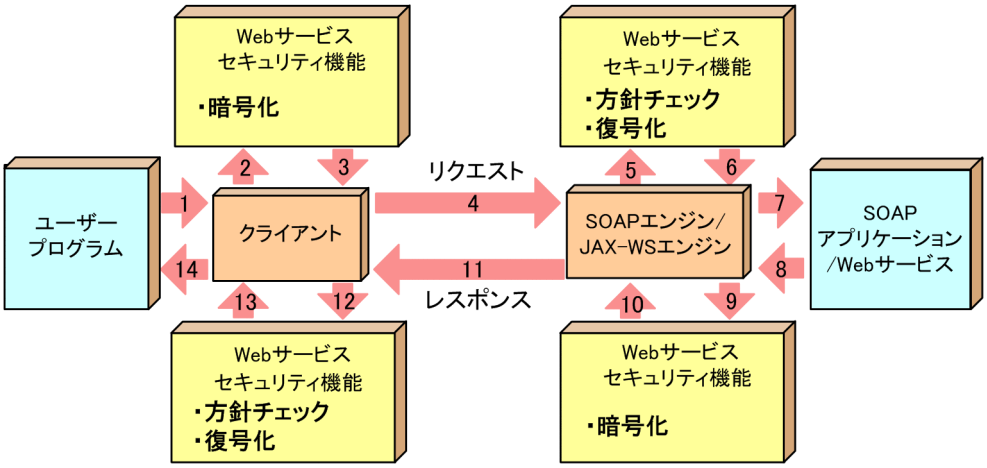
設定する要素の詳細については，「付録 C　定義ファイルの項目の詳細」を参照してください。

3.3　暗号化／復号化機能を設定する

暗号化／復号化機能は，送受信するメッセージを，暗号化／復号化する機能です。暗号化機能はメッセージの送信時に，復号化機能はメッセージ受信時に使用します。

暗号化／復号化機能を使用する場合のプログラムの構成を次の図に示します。矢印およびその番号は，送受信するデータの処理の流れを示しています。

図 3-2　Web サービスセキュリティ機能使用時のプログラム構成（暗号化／復号化）



暗号化／復号化機能を使用する際の，Web サービスセキュリティ機能定義ファイルの設定項目を次に示します。

```
＜送信側＞
SecurityConfig
├── BindingConfig
└── RequestSenderConfig/ResponseSenderConfig
    └── SenderPortConfig
        └── RoleConfig
            └── EncryptionConfig
＜受信側＞
SecurityConfig
├── BindingConfig
└── RequestReceiverConfig/ResponseReceiverConfig
    └── ReceiverPortConfig
        └── DecryptionConfig
```

暗号化／復号化機能を使用する際の，Web サービスセキュリティ方針定義ファイルの設定項目を次に示します。

```
PolicyConfig
└── RequestReceiverConfig/ResponseReceiverConfig
    └── ReceiverPortConfig
        └── DecryptionPolicyConfig
```

設定する要素の詳細については，「付録 C　定義ファイルの項目の詳細」を参照してください。

Windows の場合，定義ファイルのサンプルは，次のディレクトリに格納されていますので，これらを参考に定義ファイルを作成してください。なお，作成した定義ファイルの格納場所については，「3.9　Web サービスセキュリティ機能の実装手順（SOAP アプリケーション開発支援機能を使用する場合）」を参照してください。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

```
encryption/message/client/WEB-INF/classes
encryption/message/service/WEB-INF/classes
encryption/rpc/client/WEB-INF/classes
encryption/rpc/service/WEB-INF/classes
```

また，Windows の場合，コーディングのサンプルは，次のディレクトリに格納されていますので，参考にしてください。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

```
encryption/SecretKey
encryption/message/client/WEB-INF/classes/messagesampleclient
encryption/message/service/WEB-INF/classes/messagesampleservice
encryption/rpc/client/WEB-INF/classes/localhost
encryption/rpc/service/WEB-INF/classes/localhost
```

暗号化の設定情報について

- 暗号化に必要な情報は，Web サービスセキュリティ機能定義ファイルに指定された設定情報から取得します。
- 暗号化に関する仕様のサポート範囲については，「付録 A.3　XML 暗号標準仕様のサポート範囲」を参照してください。

復号化の設定情報について

復号化に必要な情報は，Web サービスセキュリティ機能定義ファイル，および Web サービスセキュリティ方針定義ファイル指定された設定情報から取得します。

暗号化／復号化に用いる共通鍵は，環境設定ファイルで指定した共通鍵ファイルを使用します。環境設定ファイルの詳細については，「3.8.2　環境設定ファイルの設定項目」を参照してください。

注意事項

暗号化／復号化で使用する鍵ファイルは，読み取り権限を限定するなど，第三者に読み取られないようにご注意ください。

3.3.1　暗号化する個所をパート名で指定する

暗号化する個所をパート名という名称で指定します。Web Services - Security で，暗号化する際に指定できるパート名は，"BodyContent"だけです。"BodyContent"を指定した場合，SOAP メッセージの Body 要素に含まれる子要素，テキストなどをすべて指定することを意味します。

暗号個所をパート名で指定する際の，Web サービスセキュリティ機能定義ファイルの設定項目を次に示します。

```
SecurityConfig
└─ RequestSenderConfig/ResponseSenderConfig
   └─ SenderPortConfig
      └─ RoleConfig
         └─ EncryptionConfig
            └─ ContentsEncryption
               └─ EncryptionTarget
```

暗号個所をパート名で指定する際の，Web サービスセキュリティ方針定義ファイルの設定項目を次に示します。

```
PolicyConfig
└─ RequestReceiverConfig/ResponseReceiverConfig
   └─ ReceiverPortConfig
      └─ DecryptionPolicyConfig
         └─ DecryptionTarget
```

設定する要素の詳細については，「付録 C　定義ファイルの項目の詳細」を参照してください。

3.3.2　暗号化する個所を ID 属性で指定する

暗号化する個所を，ID 属性で指定し，その ID 値が含まれる要素に対して暗号化します。なお，受信側では ID 属性を検証できません。

暗号個所を ID で指定する際の，Web サービスセキュリティ機能定義ファイルの設定項目を次に示します。

```
SecurityConfig
└─ RequestSenderConfig/ResponseSenderConfig
   └─ SenderPortConfig
      └─ RoleConfig
         └─ EncryptionConfig
            └─ ContentsEncryption
               └─ EncryptionTarget
```

設定する要素の詳細については，「付録 C　定義ファイルの項目の詳細」を参照してください。

3.4　認証機能を設定する

セキュリティトークンを通信先に受け渡すことで，認証を実現します。

Web Services - Security では，JAAS を使用した認証方式をサポートします。

認証機能を使用する際の，Web サービスセキュリティ機能定義ファイルの設定項目を次に示します。

```
＜送信側＞
SecurityConfig
├── BindingConfig
└── RequestSenderConfig/ResponseSenderConfig
    └── SenderPortConfig
        └── RoleConfig
            └── UsernameTokenConfig
＜受信側＞
SecurityConfig
├── BindingConfig
└── RequestReceiverConfig/ResponseReceiverConfig
    └── ReceiverPortConfig
        └── AuthenticationConfig
```

認証機能を使用する際の，Web サービスセキュリティ方針定義ファイルの設定項目を次に示します。

```
PolicyConfig
├── GlobalConfig
└── RequestReceiverConfig/ResponseReceiverConfig
    └── ReceiverPortConfig
        └── SecurityTokenConfig
            └── UsernameTokenConfig
```

設定する要素の詳細については，「付録 C　定義ファイルの項目の詳細」を参照してください。

Windows の場合，定義ファイルのサンプルは，次のディレクトリに格納されていますので，これらを参考に定義ファイルを作成してください。なお，作成した定義ファイルの格納場所については，「3.9　Web サービスセキュリティ機能の実装手順（SOAP アプリケーション開発支援機能を使用する場合）」を参照してください。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

```
usernameToken/rpc/client/WEB-INF/classes
usernameToken/rpc/service/WEB-INF/classes
```

また，Windows の場合，コーディングのサンプルは，次のディレクトリに格納されていますので，参考にしてください。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

```
usernameToken/rpc/Auth
usernameToken/rpc/client/WEB-INF/classes/localhost
usernameToken/rpc/service/WEB-INF/classes/localhost
```

認証の設定情報について

認証に必要な情報は，Web サービスセキュリティ機能定義ファイルに指定された設定情報から取得します。ただし，一部の情報は Web サービスセキュリティ機能が提供する API を使用して指定できます。API については，「5.　Web サービスセキュリティ機能が提供する API」を参照してください。

認証の SOAP メッセージ形式について

Web サービスセキュリティ機能が生成する SOAP メッセージ内の UsernameToken 要素の Nonce 要素と Created 要素は，パスワードのタイプがダイジェスト形式の場合に付与されます。パスワードタイプがテキスト形式の場合はこれらの要素は付与されません。

パスワードのタイプがダイジェスト形式の SOAP メッセージでダイジェストを求めたい場合は，コーディングのサンプルのソースコード（UsernameLoginModule.java）の login メソッドの実装を参考にしてください。

認証に関する方針チェックについて

Web サービスセキュリティ方針定義ファイルに指定された設定情報を基に，受信した SOAP メッセージが決められた方針に従っているかどうかをチェックします。

注意事項

認証で，パスワードの記述されたファイルなどを使用する場合は，読み取り権限を限定するなど，第三者に読み取られないようにご注意ください。

3.5　メッセージに有効期限を設定する

Web Services - Security では，SOAP メッセージの送信時にタイムスタンプや有効期限を設定したり，受信側で古いメッセージや有効期限を経過したメッセージの受信を拒否したりする機能を提供します。

メッセージに有効期限を設定する際の，Web サービスセキュリティ機能定義ファイルの設定項目を次に示します。

```
SecurityConfig
└─ RequestSenderConfig/ResponseSenderConfig
   └─ SenderPortConfig
      └─ RoleConfig
         └─ TimestampConfig
            └─ Expires
```

メッセージに付与されているタイムスタンプや有効期限を検証する際の，Web サービスセキュリティ方針定義ファイルの設定項目を次に示します。

```
PolicyConfig
├─ GlobalConfig
└─ RequestReceiverConfig/ResponseReceiverConfig
   └─ ReceiverPortConfig
      └─ TimestampConfig
         ├─ Created
         └─ Expires
```

設定する要素の詳細については，「付録 C　定義ファイルの項目の詳細」を参照してください。

Windows の場合，定義ファイルのサンプルは，次のディレクトリに格納されていますので，これらを参考に定義ファイルを作成してください。なお，作成した定義ファイルの格納場所については，「3.9　Web サービスセキュリティ機能の実装手順（SOAP アプリケーション開発支援機能を使用する場合）」を参照してください。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

```
timeStamp/rpc/client/WEB-INF/classes
timeStamp/rpc/service/WEB-INF/classes
```

また，Windows の場合，コーディングのサンプルは，次のディレクトリに格納されていますので，参考にしてください。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

```
timeStamp/rpc/client/WEB-INF/classes/localhost
timeStamp/rpc/service/WEB-INF/classes/localhost
```

3.6　定義ファイルの構文をチェックする

Web サービスセキュリティ機能定義ファイル，および Web サービスセキュリティ方針定義ファイルを設定したあと，定義ファイルの記述内容が正しいかどうかをチェックするために，構文チェック用のコマンドを使用します。

構文チェック用のコマンドの使い方については，「4.2　定義ファイル構文チェックコマンド（CWSSConfCheck）」を参照してください。

注意事項

定義ファイル構文チェックコマンドは，Windows で使用してください。その他の OS では使用できません。

3.7　定義ファイルに関する注意事項

Web サービスセキュリティ機能定義ファイル，および Web サービスセキュリティ方針定義ファイルを設定する上での注意事項について説明します。

(1)　Timestamp 要素を使用する場合

- Web サービスセキュリティ機能定義ファイルの Created 要素，Expires 要素を共に省略した場合は，Timestamp 要素自体が作成されません。
- SOAP メッセージを送信するマシンと受信するマシンに設定されている時刻は，一致していないおそれがあります。Created 要素に設定されている値を基にそのメッセージを古いと見なしたり，Expires 要素に設定されている値を基にそのメッセージを有効期限切れと見なしたりする際，許容範囲を設定できます。

  マシン間の設定時刻の差に関する許容範囲を設定する際の，Web サービスセキュリティ方針定義ファイルの設定項目を次に示します。

```
PolicyConfig
└─ GlobalConfig
   ├─ Max-Clock-Skew
   └─ Fresh-Time-Limit
```

詳細については，「付録 C.2　Web サービスセキュリティ方針定義ファイルの項目」を参照してください。

(2)　Web サービスセキュリティ機能定義ファイルの運用について

- Web サービスセキュリティ機能定義ファイルは，第三者によって改ざんされないようにアクセス権の設定をしてください。

(3)　Web サービスセキュリティ方針定義ファイルの運用について

- Web サービスセキュリティ方針定義ファイルは，第三者によって改ざんされないようにアクセス権の設定をしてください。
- Web サービスセキュリティ方針定義ファイルを SOAP サービスまたは Web サービスの利用者に配布する場合，ファイルが改ざんされないよう対策してください。
- Web サービスセキュリティ方針定義ファイルは，安全な方法で SOAP サービスまたは Web サービスの利用者に配布してください。
- SOAP サービスまたは Web サービスの利用者は，Web サービスセキュリティ方針定義ファイルの内容に従って，Web サービスセキュリティ機能定義ファイルを作成する必要があります。

(4)　セキュリティ機能を組み合わせて使用する場合

署名付与／検証機能，暗号化／復号化機能，認証機能，およびメッセージに有効期限を設定する機能は，組み合わせて使用できます。複数の機能を組み合わせて使用する場合は，Web サービスセキュリティ機能定義ファイルの RoleConfig 要素内に，それぞれの設定項目を併記してください。

次に，設定の順番についての注意事項を示します。

- 送信側の Web サービスセキュリティ機能定義ファイルに設定した機能は，設定した順番で処理されます。各機能に対応するセキュリティ要素が SOAP メッセージのセキュリティヘッダに出現しますが，その順番は，Web サービスセキュリティ機能定義ファイルで設定した順番とは逆順になります。
- Web サービスセキュリティ機能が提供する API を使用して UsernameToken を付与した場合，対応するセキュリティ要素は SOAP メッセージのセキュリティヘッダのいちばん下に出現します。

なお, 受信側の Web サービスセキュリティ機能定義ファイルと, Web サービスセキュリティ方針定義ファイルに設定する順番は，特に意識する必要はありません。

3.8　実行環境に合わせて設定を変更する

Web サービスセキュリティ機能定義ファイルおよび Web サービスセキュリティ方針定義ファイルに記述する内容で，実行環境に依存する部分は，環境設定ファイルで設定します。

環境設定ファイルは，次のディレクトリに格納されています。ファイル名は，「cwsscfg.properties」で固定です。

＜Application Serverのインストールディレクトリ＞/wss/conf の下

次に，環境設定ファイルの記述規則と設定項目について説明します。

3.8.1　環境設定ファイルの記述規則

環境設定ファイルは，J2SE のプロパティ形式に従い，「キー名称=値」の形式で記述します。次に，環境設定ファイルの記述規則について示します。

- 1 行の終わりは必ず改行されていなければなりません。
- 「#」で始まる行はコメントとみなされます。
- 値が存在しない行を設定した場合，空文字として処理されます。
- キー名称と=の間，および=と値の間にスペースを入れてはいけません。
- 値に 2 バイト文字を使用する場合は，Application Server で提供されている Java^{TM}2SDK の native2ascii コマンドで環境設定ファイルを変換してから使用しなければなりません。
- 設定できるキー名称以外のキー名称を設定した場合，その行は無視されます。
- キー名称の大文字と小文字は区別されます。
- 値には半角スペースも設定できます。

  例

  ```
  cwss.binding.KeyLocator.KeyStoreDir=d:/Program Files/HITACHI/Cosminexus/wss/KeyStore
  ```

3.8.2　環境設定ファイルの設定項目

環境設定ファイルで設定できるキーおよび値を次の表に示します。

表 3-3　環境設定ファイルの設定項目

キー名称	指定値	説明	デフォルト値 (値省略時または範囲外)
cwss.binding.KeyLocator.KeyStoreDir	任意のディレクトリ名	署名付与／検証で使用するキーストアファイルを格納するディレクトリ名を指定します。	＜Application Server のインストールディレクトリ＞/wss/KeyStore
cwss.binding.KeyLocator.CertificateDir	任意のディレクトリ名	証明書検証で使用する証明書ファイルを格納するディレクトリ名を指定します。	＜Application Server のインストールディレクトリ＞/wss/Cert

キー名称	指定値	説明	デフォルト値（値省略時または範囲外）
cwss.policy.usernameToken.maxNonceCount	1～100,000	受信済み Nonce 値を保存する最大個数を指定します。	100,000
cwss.policy.usernameToken.maxNonceAge	1,000～86,400,000（最大 24 時間）	受信済み Nonce 値の最大保存時間を現在時刻からの相対ミリ秒で指定します。	300,000（5 分）
cwss.binding.KeyLocator.SecretKeyDir	任意のディレクトリ名	暗号化／復号化で使用する共通鍵ファイルを格納するディレクトリ名を指定します。	＜ Application Server のインストールディレクトリ＞/wss/SecretKey
cwss.engine.receive.unsecured.message	true または false	受信側で Web サービスセキュリティ方針定義ファイルの設定をしている場合, 受信メッセージに SOAP メッセージの Security 要素を含まなかったときの動作を指定します。※	false

注※

受信側で Web サービスセキュリティ方針定義ファイルの設定をしている場合で，false を設定した場合，エラーメッセージ KDCGF0003-E が出力されます。true を設定した場合，Web サービスセキュリティ機能定義ファイルおよび Web サービスセキュリティ方針定義ファイルの設定内容に関係なく，エラーにはなりません。

3.9　Web サービスセキュリティ機能の実装手順 (SOAP アプリケーション開発支援機能を使用する場合)

ここでは，実行環境に必要な定義ファイルやユーザー作成のプログラムを実装し，SOAP アプリケーション開発支援機能を使用して開発した SOAP アプリケーションで Web サービスセキュリティ機能を使用する手順を説明します。

SOAP アプリケーション開発支援機能を利用した SOAP アプリケーションの開発の流れは次のとおりです。

(a) サービスの設計

(b) WSDL の作成

(c) サーバスケルトンの作成

(d) サーバ実装

(e) WAR ファイルの作成

(f) サービスの開始

(g) クライアントスタブの生成

(h) クライアントの実装

SOAP アプリケーション開発の詳細については，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。

また，07-60 以降のバージョンを使用して Web サービスセキュリティ機能を使用する場合は，移行が必要です。移行の詳細については，「付録 B　下位バージョンからの移行手順」を参照してください。

前に示した SOAP アプリケーション開発の流れ（以降，「開発ステップ」と呼びます）を基本とし，Web サービスセキュリティ機能を使用する上で必要なステップを説明します。

3.9.1　サーバ側の実装手順

Web サービスセキュリティ機能をサーバ側に実装する手順について説明します。

(1)　デプロイ定義ファイルにセキュリティ情報を付加する

SOAP アプリケーションの形態が，RPC または EJB の場合

開発ステップ「(c) サーバスケルトンの作成」で，Web サービスセキュリティ機能に関する情報※を，サーバ側 WAR ファイルの WEB-INF の下のデプロイ定義ファイルに手動で追加します。

SOAP アプリケーションの形態が，既存 Java クラスを利用，またはメッセージングの場合

開発ステップ「(e) WAR ファイルの作成」で，Web サービスセキュリティ機能に関する情報※をサーバ側 WAR ファイルの WEB-INF の下のデプロイ定義ファイルに手動で追加します。

注※

「Web サービスセキュリティ機能に関する情報」を次に示します。

```
<handler name="WSSResponseSenderHandler"
type="java:com.cosminexus.wss.handlers.WSSResponseSenderHandler"/>
<handler name="WSSRequestReceiverHandler"
type="java:com.cosminexus.wss.handlers.WSSRequestReceiverHandler"/>

<requestFlow>
 <handler type="WSSRequestReceiverHandler"/>
</requestFlow>
<responseFlow>
 <handler type="WSSResponseSenderHandler"/>
</responseFlow>
```

Windows の場合，サーバ側のデプロイ定義ファイル（server-config.xml）のサンプルは，サンプルディレクトリの下の，機能ごとのサービスに格納されていますので，参考にしてください。次に，格納先の一例を示します。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

usernameToken/rpc/service/WEB-INF

(2)　ユーザー作成のプログラムを実装する

Web サービスセキュリティ機能が提供する API を利用する場合は，開発ステップ「(d) サーバ実装」で，ユーザー作成のプログラムを実装します。詳細については，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。

(3)　WAR ファイルを作成する

開発ステップ「(e) WAR ファイルの作成」で，Web サービスセキュリティ機能が提供する Web サービスセキュリティ機能定義ファイル，および Web サービスセキュリティ方針定義ファイルをサーバ側 WAR ファイルの WEB-INF/classes の下に組み込みます。Web サービスセキュリティ機能定義ファイルと Web サービスセキュリティ方針定義ファイルについては，構文チェック用のコマンドを使用して，構文に誤りがないことを事前に確認してください。構文チェック用のコマンドの使い方については，「4.2　定義ファイル構文チェックコマンド（CWSSConfCheck)」を参照してください。

WAR ファイルへの組み込み方法については，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。

サーバ側の実装に関するファイルの配置は，各アプリケーションごと（WAR ファイルごと）に次のようになります。

```
WEB-INF
 ├server-config.xml       デプロイ定義ファイル
 └classes
   ├security-config.xml  Webサービスセキュリティ機能定義ファイル
   └policy-config.xml    Webサービスセキュリティ方針定義ファイル
```

3.9.2　クライアント側が Web アプリケーションの場合の実装手順

Web サービスセキュリティ機能をクライアント側の Web アプリケーションに実装する手順について説明します。

(1)　デプロイ定義ファイルを組み込む

開発ステップ「(g) クライアントスタブの生成」で，Web サービスセキュリティ機能が提供するクライアント用のデプロイ定義ファイル（client-config.xml）を，クライアント側 WAR ファイルの WEB-INF/classes の下に組み込みます。Windows の場合，Web サービスセキュリティ機能が提供するクライアント用のデプロイ定義ファイルは，サンプルディレクトリの下の，機能ごとのクライアントに格納されています。どのクライアント用のデプロイ定義ファイルも内容は同じです。次に，格納先の一例を示します。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下
　usernameToken/rpc/client/WEB-INF/classes

(2)　ユーザー作成のプログラムを実装する

Web サービスセキュリティ機能が提供する API を利用する場合は，開発ステップ「(h) クライアントの実装」で，ユーザー作成のプログラムを実装します。詳細については，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。

(3)　WAR ファイルを作成する

サンプルファイルを基に作成した Web サービスセキュリティ機能定義ファイル，および Web サービスセキュリティ方針定義ファイルをクライアント側 WAR ファイルの WEB-INF/classes の下に組み込みます。Web サービスセキュリティ機能定義ファイルと Web サービスセキュリティ方針定義ファイルについては，構文チェック用のコマンドを使用して，構文に誤りがないことを事前に確認してください。構文チェック用のコマンドの使い方については，「4.2　定義ファイル構文チェックコマンド (CWSSConfCheck)」を参照してください。

WAR ファイルへの組み込み方法については，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。

クライアント側の実装に関するファイルの配置は，各アプリケーションごと（WAR ファイルごと）に次のようになります。

```
WEB-INF
 └classes
   ├client-config.xml    デプロイ定義ファイル
   ├security-config.xml  Webサービスセキュリティ機能定義ファイル
   └policy-config.xml    Webサービスセキュリティ方針定義ファイル
```

3.9.3　クライアント側がコマンドライン Java アプリケーションの場合の実装手順

サンプルファイルを基に作成した Web サービスセキュリティ機能定義ファイル，Web サービスセキュリティ方針定義ファイル，およびデプロイ定義ファイルを，クライアント実行環境の CLASSPATH に設定したディレクトリの下に格納します。なお，Web サービスセキュリティ機能定義ファイルと Web サービスセキュリティ方針定義ファイルについては，構文チェック用のコマンドを使用して，構文に誤りがないことを事前に確認してください。構文チェック用のコマンドの使い方については，「4.2　定義ファイル構文チェックコマンド（CWSSConfCheck)」を参照してください。

さらに，cjclstartap コマンドを使用して，Web サービスセキュリティ機能を適用したクライアントを実行する場合，Java アプリケーション用オプション定義ファイルに次の値を設定したキーを追加する必要があります。

- add.class.path=<Application Serverのインストールディレクトリ>¥wss¥lib¥cwssec.jar
- add.class.path=<Application Serverのインストールディレクトリ>¥XMLSEC¥lib¥csmxsec.jar

Java アプリケーション用オプション定義ファイルについては，マニュアル「アプリケーションサーバ リファレンス 定義編(サーバ定義)」を参照してください。また，コマンドライン利用時の設定については，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。

3.9.4　JAAS ログインモジュールの実装時の注意

ここでは，JAAS ログインモジュールを実装する際の注意事項について説明します。なお，JAAS ログインモジュールはユーザー側で開発します。一般的な JAAS ログインモジュールの開発方法については，Sun Microsystems,Inc. が公開している JAAS 認証に関するリファレンスガイドを参照してください。

(1)　Callback オブジェクトの生成について

JAAS ログインモジュールで認証情報を取得するためには，Web サービスセキュリティ機能が提供するコールバックハンドラを使用します。次に，JAAS ログインモジュール内で必要な処理を示します。

LoginModule.initialize

コールバックハンドラ（javax.security.auth.callback.CallbackHandler の実装クラス）をクラスメンバ変数に保持します。

LoginModule.login

コールバックハンドラ内で使用する Callback オブジェクト（javax.security.auth.callback.Callback）の配列を生成して，コールバックハンドラの handle メソッドを呼び出します。handle メソッドを呼び出したあと，Callback オブジェクトから認証情報を取り出して認証処理を行います。Callback オブジェクトの配列は，次の表に示す構成で生成してください。

表 3-4　生成する Callback オブジェクトの配列

配列番号	Callback オブジェクト	用途
0	javax.security.auth.callback.NameCallback	ユーザー名の取得
1	javax.security.auth.callback.PasswordCallback	パスワードの取得
2	javax.security.auth.callback.TextInputCallback	パスワード形式の取得
3	javax.security.auth.callback.TextInputCallback	Nonce 値の取得
4	javax.security.auth.callback.TextInputCallback	Created 値の取得

配列要素の取得方法を次に示します。

0：ユーザー名の取得

ユーザー名は javax.security.auth.callback.NameCallback.getName()メソッドで取得します。

1：パスワードの取得

パスワードは javax.security.auth.callback.PasswordCallback.getPassword()メソッドで取得します。パスワード形式がテキスト形式の場合，パスワードテキスト文字配列を返します。パスワード形式がダイジェスト形式の場合，ダイジェスト形式の文字配列を返します。パスワード省略時は null を返します。

2：パスワード形式の取得

パスワード形式は javax.security.auth.callback.TextInputCallback.getText()メソッドで取得します。パスワードの形式は次の文字列で返します。

"PasswordText"：テキスト形式

"PasswordDigest"：ダイジェスト形式

3：Nonce 値の取得

Nonce 値は javax.security.auth.callback.TextInputCallback.getText()メソッドで取得します。

4：Created 値の取得

Created 値は javax.security.auth.callback.TextInputCallback.getText()メソッドで取得します。

Windows の場合，JAAS ログインモジュールのコーディング例については，次のディレクトリに格納されているサンプルを参照してください。

＜Application Serverのインストールディレクトリ＞/wss/samples/c4web/ の下

```
usernameToken/rpc/Auth
usernameToken/rpc/client/WEB-INF/classes/localhost
usernameToken/rpc/service/WEB-INF/classes/localhost
```

(2)　ログイン構成ファイルの配置について

Component Container のユーザー定義ファイル（usrconf.properties）に，次のエントリーを追加してください。

```
java.security.auth.login.config==＜ログイン構成ファイルのフルパス名＞
```

ユーザー定義ファイルについては，マニュアル「アプリケーションサーバ リファレンス 定義編(サーバ定義)」を参照してください。

3.10　Web サービスセキュリティ機能の実装手順（JAX-WS 機能を使用する場合）

ここでは，実行環境に必要な定義ファイルやユーザー作成のプログラムを実装し，JAX-WS 機能を使用して開発した Web サービスで Web サービスセキュリティ機能を使用する手順を説明します。

JAX-WS 機能を利用した Web サービスに Web サービスセキュリティ機能を組み込む場合の開発の流れ（以降，「開発ステップ」と呼びます）は次のとおりです。

(a) Web サービスまたは Web サービスクライアントの実装（3.10.1，3.10.2，3.10.3）

(b) Web サービスセキュリティハンドラの追加（3.10.1(1)，3.10.2(1)，3.10.3(1)）

(c) Web サービスセキュリティ機能定義ファイル，Web サービスセキュリティ方針定義ファイル，ハンドラチェイン設定ファイルの追加（3.10.1(2)，3.10.2(2)，3.10.3(2)）

(d) 認証情報の取得・設定（任意）（3.10.4）

括弧内の番号は，このマニュアルでの参照個所を表しています。JAX-WS 機能を利用した Web サービス開発の詳細および開発ステップ（d）については，マニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

3.10.1　Web サービスの実装手順（WSDL 起点）

WSDL を起点に JAX-WS 機能で Web サービスを開発する場合の流れを次に示します。

1. WSDL ファイルの作成
2. Java ソースの生成
3. Web サービスの実装
4. DD の作成
5. EAR ファイルの作成
6. EAR ファイルのデプロイと開始

Web サービスセキュリティ機能を組み込むためには，開発ステップ 3 で Web サービスセキュリティハンドラを追加し，開発ステップ 5 で各種定義ファイルと設定ファイルを追加する必要があります。

! 注意事項

Web サービスセキュリティハンドラは，必ず単独で使用する必要があります。ほかのハンドラとの併用はできないため，注意してください。

ここでは，開発ステップ 3 と開発ステップ 5 についてだけ説明します。そのほかの開発ステップの詳細は，マニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

(1)　Web サービスセキュリティハンドラを追加する

Application Server では，ハンドラチェイン設定ファイルのテンプレート（ cwsshandler.xml）を提供しています。Web サービスに Web サービスセキュリティ機能を組み込む場合は，テンプレートをコピーして作成したハンドラチェイン設定ファイルを利用して，Web サービスセキュリティハンドラ（ com.cosminexus.wss.handlers.WSSServerHandler）を追加してください。

Web サービス実装時，javax.jws.HandlerChain アノテーションで Web サービス実装クラスをアノテートし，同アノテーションの file 要素に作成したハンドラチェイン設定ファイルを指定します。指定例を次に示します。

```
@javax.jws.WebService
@javax.jws.HandlerChain(file="cwsshandler.xml")
public class AddNumbersImpl{

    public int add( int number1, int number2 ) throws AddNumbersFault{
      ...
    }
}
```

(2)　Web サービスセキュリティ機能定義ファイル，Web サービスセキュリティ方針定義ファイル，ハンドラチェイン設定ファイルを追加する

Web サービスセキュリティ機能定義ファイル，Web サービスセキュリティ方針定義ファイル，およびハンドラチェイン設定ファイルは，すべて EAR ファイルに配置する必要があります。

Web サービスセキュリティ機能定義ファイルおよび Web サービスセキュリティ方針定義ファイルは，組み込みたい Web サービスセキュリティ機能の種類によって編集方法が異なります。編集方法については，目的に応じて「3.2　署名付与／検証機能を設定する」，「3.3　暗号化／復号化機能を設定する」，「3.4　認証機能を設定する」，または「3.5　メッセージに有効期限を設定する」を参照してください。配置方法については，「3.9.1(3)　WAR ファイルを作成する」を参照してください。

ハンドラチェイン設定ファイルは，どの Web サービスセキュリティ機能を使用する場合も同じです。ハンドラチェイン設定ファイルの編集方法については「3.10.1(1)　Web サービスセキュリティハンドラを追加する」を，配置方法についてはマニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

3.10.2　Web サービスの実装手順（SEI 起点）

SEI（Service Endpoint Interface）を起点に JAX-WS 機能で Web サービスを開発する場合の流れを次に示します。

1. Web サービス実装クラスの作成
2. Java ソースの生成
3. DD の作成
4. EAR ファイルの作成
5. EAR ファイルのデプロイと開始

Web サービスセキュリティ機能を組み込むためには，開発ステップ 1 で Web サービスセキュリティハンドラを追加し，開発ステップ 4 で各種定義ファイルと設定ファイルを追加する必要があります。

！ 注意事項

Web サービスセキュリティハンドラは，必ず単独で使用する必要があります。ほかのハンドラとの併用はできないため，注意してください。

(1)　Web サービスセキュリティハンドラを追加する

Application Server では，ハンドラチェイン設定ファイルのテンプレート（cwsshandler.xml）を提供しています。Web サービスセキュリティ機能を組み込む場合は，テンプレートをコピーしてハンドラチェイン設定ファイルを作成してください。

Web サービス実装時，javax.jws.HandlerChain アノテーションで SEI をアノテートし，同アノテーションの file 要素に作成したハンドラチェイン設定ファイルを指定します。指定例は，「3.10.1(1)　Web サービスセキュリティハンドラを追加する」を参照してください。

(2)　Web サービスセキュリティ機能定義ファイル，Web サービスセキュリティ方針定義ファイル，ハンドラチェイン設定ファイルを追加する

Web サービスセキュリティ機能定義ファイル，Web サービスセキュリティ方針定義ファイル，およびハンドラチェイン設定ファイルは，すべて EAR ファイルに配置する必要があります。

Web サービスセキュリティ機能定義ファイルおよび Web サービスセキュリティ方針定義ファイルは，組み込みたい Web サービスセキュリティ機能の種類によって編集方法が異なります。編集方法については，目的に応じて「3.2　署名付与／検証機能を設定する」，「3.3　暗号化／復号化機能を設定する」，「3.4　認証機能を設定する」，または「3.5　メッセージに有効期限を設定する」を参照してください。配置方法については，「3.9.1(3)　WAR ファイルを作成する」を参照してください。

ハンドラチェイン設定ファイルは，どの Web サービスセキュリティ機能を使用する場合も同じです。ハンドラチェイン設定ファイルの編集方法については「3.10.1(1)　Web サービスセキュリティハンドラを追加する」を，配置方法についてはマニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

3.10.3　Web サービスクライアントの実装手順

JAX-WS 機能で Web サービスクライアントを開発する場合の流れを次に示します。

1. WSDL ファイルの入手
2. Java ソースの生成
3. Web サービスクライアントの実装
4. Web サービスの呼び出し

Web サービスセキュリティ機能を組み込むためには，開発ステップ 2 で Web サービスセキュリティハンドラを追加し，開発ステップ 3 で各種定義ファイルと設定ファイルを追加する必要があります。

! 注意事項

Web サービスセキュリティハンドラは，必ず単独で使用する必要があります。ほかのハンドラとの併用はできないため，注意してください。

参考

サービスクラスの生成およびポートの取得には処理コストが掛かるため，一度生成したサービスクラスおよび取得したポートは再利用することを推奨します。再利用の詳細については，マニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

ここでは，開発ステップ 2 と開発ステップ 3 についてだけ説明します。そのほかの開発ステップの詳細は，マニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

(1)　Web サービスセキュリティハンドラを追加する

Application Server では，外部バインディングファイルのテンプレート（cwssbinding.xml）を提供しています。Web サービスクライアントに Web サービスセキュリティ機能を組み込む場合は，テンプレートをコピーして作成した外部バインディングファイルを利用して，Web サービスセキュリティハンドラ（com.cosminexus.wss.handlers.WSSClientHandler）を追加してください。

Web サービスクライアント実装時には，まず外部バインディングファイルの bindings 要素の wsdlLocation 属性に WSDL ファイルを指定します。例を次に示します。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<jaxws:bindings
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:jaxws="http://java.sun.com/xml/ns/jaxws"
    xmlns:javaee="http://java.sun.com/xml/ns/javaee"
    wsdlLocation="./WEB-INF/wsdl/WebService.wsdl">
    <jaxws:bindings node="wsdl:definitions">
        <javaee:handler-chains>
            <javaee:handler-chain>
                <javaee:handler>
                    <javaee:handler-class>com.cosminexus.wss.handlers.WSSClientHandler</
javaee:handler-class>
                </javaee:handler>
            </javaee:handler-chain>
        </javaee:handler-chains>
    </jaxws:bindings>
</jaxws:bindings>
```

編集した外部バインディングファイルを引数に指定して，JAX-WS 機能が提供する cjwsimport コマンドを実行します。これにより，Web サービスクライアントに Web サービスセキュリティハンドラを追加するための二つのハンドラチェイン設定ファイルが生成されます。生成されるハンドラチェイン設定ファイルの名称と用途を次に示します。

表 3-5　cjwsimport コマンドで生成されるハンドラチェイン設定ファイル

ファイル名	用途
サービス名 ＋ "_handler.xml "	サービスクラス用ハンドラチェイン設定ファイル
SEI 名+ "_handler.xml "	SEI 用ハンドラチェイン設定ファイル

生成されたハンドラチェイン設定ファイルは，編集しないでそのまま使用してください。また，cjwsimport コマンド，サービス名，および SEI 名の詳細については，マニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

(2)　Web サービスセキュリティ機能定義ファイル，Web サービスセキュリティ方針定義ファイル，ハンドラチェイン設定ファイルを追加する

Web サービスセキュリティ機能定義ファイル，Web サービスセキュリティ方針定義ファイル，および「3.10.3(1)　Web サービスセキュリティハンドラを追加する」で生成したサービスクラス用ハンドラチェイン設定ファイルおよび SEI 用ハンドラチェイン設定ファイルは，それぞれ適切なディレクトリに配置する必要があります。

Web サービスセキュリティ機能定義ファイルおよび Web サービスセキュリティ方針定義ファイルは，組み込みたい Web サービスセキュリティ機能の種類によって編集方法が異なります。編集方法については，目的に応じて次の個所を参照してください。

- 「3.2　署名付与／検証機能を設定する」
- 「3.3　暗号化／復号化機能を設定する」
- 「3.4　認証機能を設定する」
- 「3.5　メッセージに有効期限を設定する」

配置方法についての参照個所は，次のとおりです。

- Web サービスクライアントが Web アプリケーションの場合の配置方法は，「3.9.2(3)　WAR ファイルを作成する」を参照してください。
- Web サービスクライアントがコマンドライン Java アプリケーションの場合の配置方法は，「3.9.3　クライアント側がコマンドライン Java アプリケーションの場合の実装手順」を参照してください。

生成した二つのハンドラチェイン設定ファイルは編集しないで，そのままサービスクラスおよび SEI のクラスファイルと同じディレクトリに配置してください。

(3)　コマンドラインを利用して Web サービスクライアントを実行する

cjclstartap コマンドを使用して，Web サービスセキュリティ機能を適用したクライアントを実行する場合，Java アプリケーション用オプション定義ファイルに次の値を設定したキーを追加する必要があります。

- add.class.path=<Application Serverのインストールディレクトリ>¥wss¥lib¥cwssec.jar
- add.class.path=<Application Serverのインストールディレクトリ>¥XMLSEC¥lib¥csmxsec.jar

Java アプリケーション用オプション定義ファイルについてはマニュアル「アプリケーションサーバ リファレンス 定義編(サーバ定義)」を，コマンドライン利用時の設定についてはマニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

(4)　サービスクラス生成時の注意事項

Web サービスセキュリティ機能を Web サービスクライアントに適用する場合，複数のスレッドで同時にサービスクラスを生成しないでください。異なるサービスクラスの場合も，複数のスレッドで同時に生成するとエラーが発生するおそれがあります。

3.10.4　認証情報の取得・設定手順

JAX-WS 機能を使用して開発した Web サービスに対して認証機能を設定したい場合，Web サービスセキュリティ機能が提供する API を利用すると，メッセージ内の UsernameToken 要素を操作できます。ここでは，UsernameToken 要素を操作することで，認証情報を取得または設定する方法を説明します。

(1)　Web サービス側での認証情報の取得

Web サービスが受信したメッセージの認証情報は，JAX-WS 機能のサポート範囲内でアクセスできるメッセージコンテキストから取得します。手順を次に示します。

1. Web サービス実装クラスで javax.xml.ws.WebServiceContext クラスのインスタンス変数を宣言します。
2. 手順 1 で宣言したインスタンス変数を javax.annotation.Resource アノテーションでアノテートします。
3. Web サービス実装クラスのメソッド内で，手順 1 で宣言したインスタンス変数の getMessageContext()メソッドを呼び出します。
4. 呼び出した getMessageContext()メソッドの戻り値として，MessageContext インタフェースを取得します。
5. WSSConstants.WSS_RECV_ELEMENTPROXY をキーとして MessageContext インタフェースから java.util.List<WSSElementProxy>クラスのオブジェクトを取得します。

これらの手順に従って，Web サービス実装クラス TestJaxWsImpl のメソッド jaxWsTest()でメッセージコンテキストを取得する例を示します。

```
import javax.annotation.Resource;
import javax.xml.ws.WebServiceContext;
import javax.xml.ws.handler.MessageContext;
import com.cosminexus.wss.element.WSSConstants;
・・・
public class TestJaxWsImpl {
    @Resource WebServiceContext wsContext;
    public String jaxWsTest()
        throws UserDefinedException
    {
        ・・・
        MessageContext msgContext = wsContext.getMessageContext();
        Object obj = msgContext.get(WSSConstants.WSS_RECV_ELEMENTPROXY);
        ・・・
   }
}
```

(2)　Web サービスクライアント側での認証情報の設定

Web サービスクライアントが送信するメッセージに認証情報を設定したい場合，JAX-WS 機能のサポート範囲内のメッセージコンテキストを設定します。手順を次に示します。

1. Web サービスクライアントのサービスクラスから SEI（ポート）を取得します。
2. SEI から getRequestContext()メソッドを呼び出し，戻り値として MessageContext インタフェースを取得します。
3. WSSConstants.WSS_SEND_ELEMENTPROXY をキーとして MessageContext インタフェースに WSSElementProxy クラスを設定します。

これらの手順に従って，Web サービスクライアントの TestClient クラスを利用してメッセージコンテキストを取得する例を示します。

```
import java.util.Map;
import javax.xml.ws.BindingProvider;
import com.cosminexus.wss.element.WSSConstants;
import com.cosminexus.wss.element.WSSElementProxy;
import com.cosminexus.wss.element.WSSElementProxyBuilder;
import com.example.sample.TestJaxWs;
import com.example.sample.TestJaxWsService;
・・・
public class TestClient {
    public static void main( String[] args ) {
        ・・・
        TestJaxWsService service = new TestJaxWsService();
        TestJaxWs port = service.getTestJaxWs();
        ・・・
        WSSElementProxyBuilder proxyBuilder = WSSElementProxyBuilder.newInstance();
        WSSElementProxy proxy = proxyBuilder.createWSSElementProxy();
        ・・・
        Map<String, Object> context = ((BindingProvider) port).getRequestContext();
        context.put(WSSConstants.WSS_SEND_ELEMENTPROXY, proxy);
        ・・・
    }
```

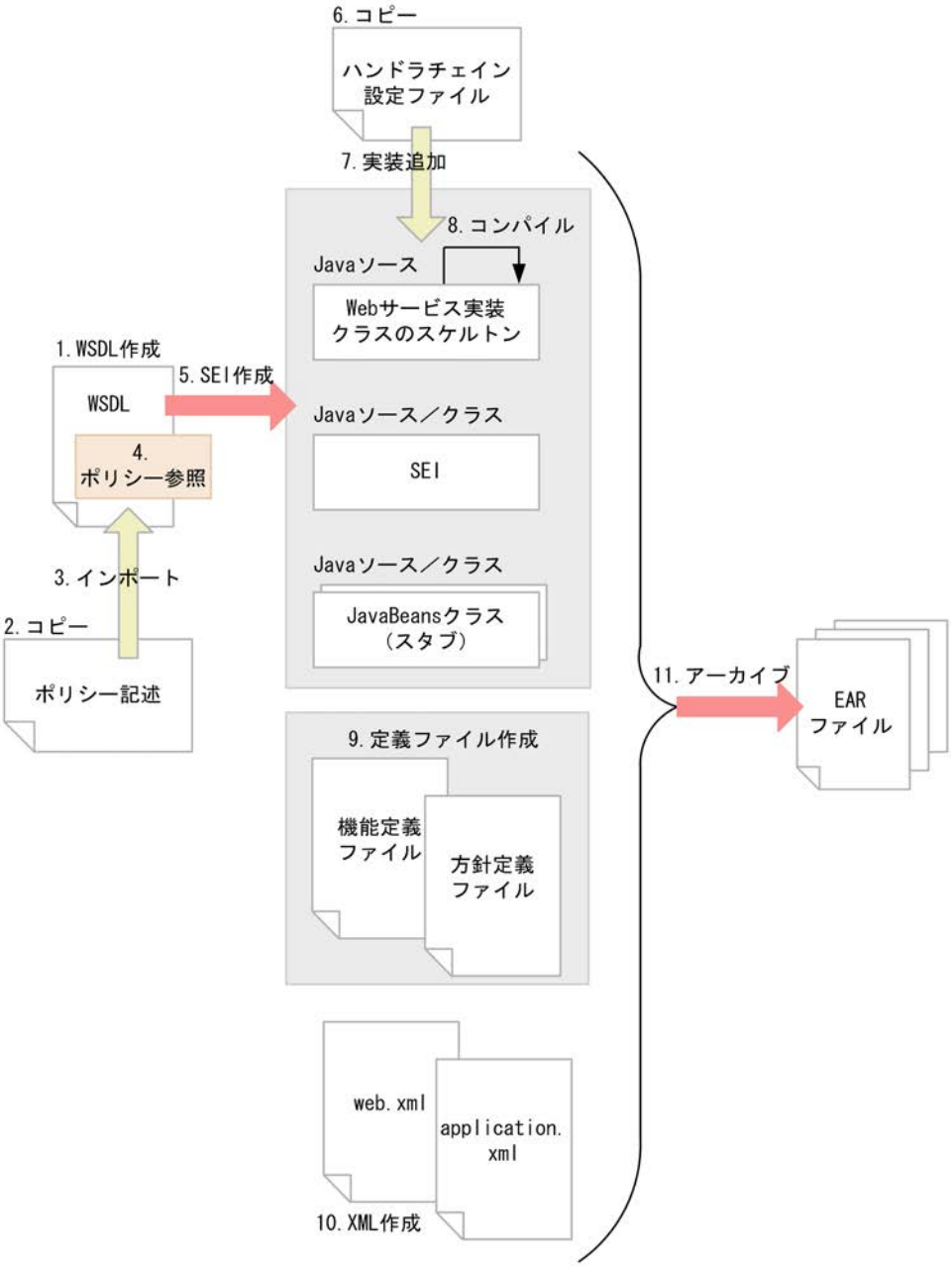
3.11　Web サービスセキュリティ機能の実装手順（ポリシーを使用する場合）

JAX-WS 機能で Web サービスを開発する場合，WSDL には記述できない機能や要件をポリシーとして定めることができます。ここで使用するポリシーでは， WS-SecurityPolicy 仕様に準拠したアサーションを利用する必要があります。

3.11.1　Web サービスの実装手順

ポリシーを使用した Web サービスの開発の流れを次に示します。

図 3-3　ポリシーを使用した Web サービスの開発



1. WSDL ファイルを作成します。※1
2. ポリシー記述をコピーします。
3. WSDL ファイルにポリシー記述をインポートします（詳細は「3.11.1(1)　ポリシー記述をインポートする」を参照)。
4. WSDL ファイルにポリシー参照を記述して，ポリシー記述内のアサーションと WSDL の要素を関連づけます。
5. SEI を作成します。※1
6. ハンドラチェイン設定ファイルをコピーします。※1
7. Web サービス実装クラスを作成します。
8. Web サービス実装クラスをコンパイルします。※1
9. ポリシー記述に対応する Web サービスセキュリティ機能定義ファイルおよび Web サービスセキュリティ方針定義ファイルを作成します（詳細は「3.11.3　定義ファイルの編集」を参照)。
10. web.xml および application.xml を作成します。※1
11. EAR ファイルを作成します。※2

注※1

ポリシーを使用しない場合と共通の手順です。

注※2

ポリシーを使用する場合，EAR ファイルを作成するときには次のことに注意してください。

- Web サービスセキュリティ機能定義ファイル，Web サービスセキュリティ方針定義ファイル，およびハンドラチェイン設定ファイルを EAR ファイルに配置する必要があります。詳細は，「3.10.1　Web サービスの実装手順（WSDL 起点)」を参照してください。
- ポリシー記述を含むメタデータを発行するには，WSDL およびポリシー記述を WAR ファイルの WEB-INF/wsdl ディレクトリに含める必要があります。詳細はマニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

(1)　ポリシー記述をインポートする

Web サービスセキュリティ機能では，次のポリシー記述をサポートしています。

表 3-6　サポートしているポリシー記述と格納先ディレクトリ

ポリシー記述	説明	格納先
SignaturePolicy.wsdl	リクエストメッセージおよびレスポンスメッセージの両方に対して，X.509 証明書を使用した署名付与／検証機能を使用できます。	<Application Server のインストールディレクトリ>/wss/policies/signature/
EncryptionPolicy.wsdl	リクエストメッセージおよびレスポンスメッセージの両方に対して，共通鍵暗号 AES-128 を使用した暗号化／復号化機能を使用できます。	<Application Server のインストールディレクトリ>/wss/policies/encryption/
AuthenticationPolicy.wsdl	ユーザー名およびダイジェスト形式のパスワードを設定した UsernameToken 要素をリクエストメッセージに付与し，通信先にセキュリ	<Application Server のインストールディレクトリ>/wss/policies/authentication/

ポリシー記述	説明	格納先
AuthenticationPolicy.wsdl	ティトークンを渡すことによって，認証機能が使用できます。	<Application Server のインストールディレクトリ>/wss/policies/authentication/

Web サービスセキュリティ機能は，これらのポリシー記述をテンプレートとして提供しています。テンプレートの中から必要なポリシー記述を一つだけ選択してコピーし，wsdl:import 要素を使用して WSDL ファイルにポリシー記述をインポートしてください。インポート例を次に示します。

```
<wsdl:definitions ・・・>
  <wsdl:import
     namespace="http://cosminexus.com/wss/policy/signature"
     location="SignaturePolicy.wsdl"/>
  ・・・
</wsdl:definitions>
```

! 注意事項

ポリシー記述をコピーして使用する場合は，次のことに注意してください。

- コピーできるポリシー記述は一つです。複数のポリシー記述を組み合わせて利用することはできません。
- コピーしたポリシー記述は変更しないでください。変更した場合の動作は保証されません。
- ポリシー記述を含むメタデータを発行したい場合は，ローカルのポリシー記述を相対パスで指定してください。絶対パスで指定すると，ポリシー記述を含むメタデータを発行できません。

次に，Web サービスセキュリティ機能がサポートしているポリシー記述に含まれるアサーションを示します。

表 3-7　アサーション一覧

ポリシー記述	アサーション種別	アサーション名 (wsp:Policy 要素の Name 属性)
SignaturePolicy.wsdl	セキュリティバインディング	http://cosminexus.com/wss/policy/signature#SignaturePolicy
	リクエストメッセージのプロテクション	http://cosminexus.com/wss/policy/signature#SignedInputMessagePolicy
	レスポンスメッセージのプロテクション	http://cosminexus.com/wss/policy/signature#SignedOutputMessagePolicy
EncryptionPolicy.wsdl	セキュリティバインディング	http://cosminexus.com/wss/policy/encryption#EncryptionPolicy
	リクエストメッセージのプロテクション	http://cosminexus.com/wss/policy/encryption#EncryptedInputMessagePolicy
	レスポンスメッセージのプロテクション	http://cosminexus.com/wss/policy/encryption#EncryptedOutputMessagePolicy
AuthenticationPolicy.wsdl	サポーティングトークン	http://cosminexus.com/wss/policy/authentication#AuthenticationPolicy

(2)　ポリシー参照を追加する

ポリシーを利用するには，wsp:PolicyReference 要素を使用して WSDL ファイルにポリシー参照の記述を追加し，ポリシー記述のアサーションと WSDL の要素とを関連づける必要があります。

wsp:PolicyReference 要素は，次の形式で記述します。

```
<wsp:PolicyReference URI="参照するアサーション名"/>
```

アサーション名には，wsp:Policy 要素の Name 属性を指定します。アサーション名（wsp:Policy 要素の Name 属性）の詳細は，表 3-7 を参照してください。

ポリシー記述に含まれるアサーションは，すべて WSDL の要素から参照される必要があります。wsp:PolicyReference 要素を使用した場合に，アサーションを参照する WSDL の要素は次のとおりです。

表 3-8　アサーションを参照する WSDL の要素

WSDL の要素	参照できるアサーションの種別
/wsdl:definitions/wsdl:binding	• セキュリティバインディング • サポーティングトークン
/wsdl:definitions/wsdl:binding/wsdl:operation/wsdl:input	リクエストメッセージのプロテクション
/wsdl:definitions/wsdl:binding/wsdl:operation/wsdl:output	レスポンスメッセージのプロテクション

ポリシー参照の例を次に示します。

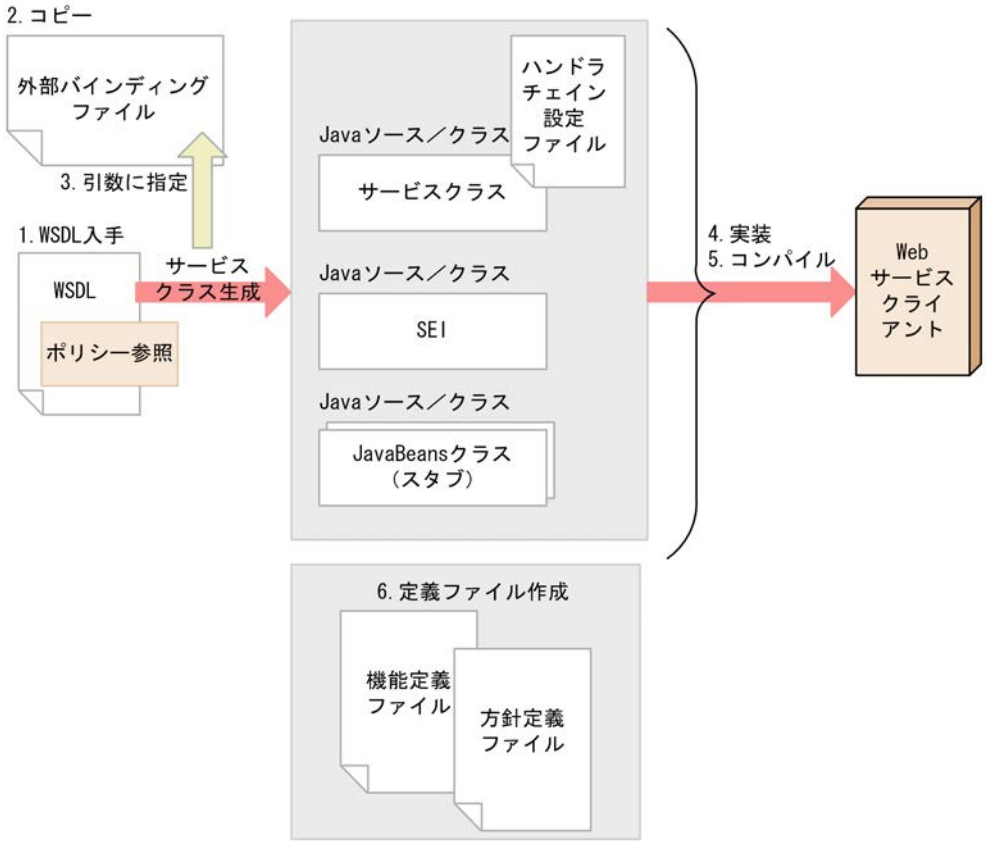
```
<wsdl:definitions
   xmlns:wsp="http://www.w3.org/ns/ws-policy/"
  ・・・>

  <wsdl:import
     namespace="http://cosminexus.com/wss/policy/signature"
     location="SignaturePolicy.wsdl"/>
  <wsdl:types>・・・</wsdl:types>
  <wsdl:message ・・・>・・・</wsdl:message>
  ・・・
  <wsdl:portType ・・・>・・・</wsdl:portType>

  <!-- バインディング -->
  <wsdl:binding ・・・>
    <wsp:PolicyReference URI="http://cosminexus.com/wss/policy/signature#SignaturePolicy"/>
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <!-- オペレーション -->
    <wsdl:operation ・・・>
      <soap:operation/>
      <!-- リクエストメッセージ -->
      <wsdl:input>
        <wsp:PolicyReference URI="http://cosminexus.com/wss/policy/
signature#SignedInputMessagePolicy"/>
        <soap:body use="literal"/>
      </wsdl:input>
      <!-- レスポンスメッセージ -->
      <wsdl:output>
        <wsp:PolicyReference URI="http://cosminexus.com/wss/policy/
signature#SignedOutputMessagePolicy"/>
        <soap:body use="literal"/>
      </wsdl:output>
      ・・・
    </wsdl:operation>
  </wsdl:binding>

  <wsdl:service ・・・>・・・</wsdl:service>
```

```
</wsdl:definitions>
```

3.11.2　Web サービスクライアントの実装手順

ポリシーを使用した Web サービスクライアントの開発の流れを次に示します。

図 3-4　ポリシーを使用した Web サービスクライアントの開発



1. WSDL ファイルを入手，または公開されている WSDL の URL を取得します。※
2. 外部バインディングファイルをコピーして，編集します（詳細は「3.10.3(1)　Web サービスセキュリティハンドラを追加する」を参照）。
3. 外部バインディングファイルを引数に指定して，cjwsimport コマンドを実行し，サービスクラスを生成します。
4. Web サービスクライアントの実装クラスを作成します。※
5. Web サービスクライアントの実装クラスをコンパイルします。※
6. ポリシー記述に対応する Web サービスセキュリティ機能定義ファイルおよび Web サービスセキュリティ方針定義ファイルを作成します（詳細は「3.11.3　定義ファイルの編集」を参照）。

注※

ポリシーを使用しない場合と共通の手順です。

3.11.3　定義ファイルの編集

ポリシー記述を使用する場合は，Application Server が提供する，各ポリシー記述に対応する Web サービスセキュリティ機能定義ファイルおよび Web サービスセキュリティ方針定義ファイルのテンプレートをコピーして使用します。使用するポリシー記述に応じて，次に示す場所からテンプレートをコピーしてください。

表 3-9　ポリシー記述に対応した定義ファイルのテンプレートの格納場所

ポリシー記述	定義ファイルの格納場所
SignaturePolicy.wsdl	• Web サービス側 <Application Server のインストールディレクトリ>/wss/policies/signature/templates/server/ • Web サービスクライアント側 <Application Server のインストールディレクトリ>/wss/policies/signature/templates/client/
EncryptionPolicy.wsdl	• Web サービス側 <Application Server のインストールディレクトリ>/wss/policies/encryption/templates/server/ • Web サービスクライアント側 <Application Server のインストールディレクトリ>/wss/policies/encryption/templates/client/
AuthenticationPolicy.wsdl	• Web サービス側 <Application Server のインストールディレクトリ>/wss/policies/authentication/templates/server/ • Web サービスクライアント側 <Application Server のインストールディレクトリ>/wss/policies/authentication/templates/client/

なお，コピーした Web サービスセキュリティ機能定義ファイルおよび Web サービスセキュリティ方針定義ファイルは，編集する必要があります。編集方法を次に示します。

(1)　署名付与／検証機能を設定する場合

表 3-10　ポリシー記述を使用するために編集が必要な属性（署名付与／検証機能を設定する場合）

定義ファイル	要素	編集が必要な属性
Web サービスセキュリティ機能定義ファイル	KeyStore 要素	• File 属性 • Type 属性 • Password 属性
	Certificate 要素	Alias 属性
	PrivateKey 要素	• Alias 属性 • Password 属性
Web サービスセキュリティ方針定義ファイル	AuthorityCertificateFile 要素	Name 属性

(2)　暗号化／復号化機能を設定する場合

表 3-11　ポリシー記述を使用するために編集が必要な属性（暗号化／復号化機能を設定する場合）

定義ファイル	要素	編集が必要な属性
Web サービスセキュリティ機能定義ファイル	SecretKeyFile 要素	Name 属性

注

暗号化／復号化機能を設定する場合，Web サービスセキュリティ方針定義ファイルで編集が必要な属性はありません。

(3)　認証機能を設定する場合

認証機能を設定する場合に編集が必要なのは，Web サービス側に配置する Web サービスセキュリティ機能定義ファイルの ConfigurationIndex 要素だけです。Web サービスセキュリティ方針定義ファイルおよび Web サービスクライアント側の Web サービスセキュリティ機能定義ファイルは，テンプレートをコピーしたものをそのまま使用してください。

なお，Web サービスクライアントは，認証に必要なユーザー名とパスワード（ダイジェスト形式）を指定するように実装する必要があります。実装に必要な API については，「5.　Web サービスセキュリティ機能が提供する API」を参照してください。

4 Web サービスセキュリティ機能が提供するコマンド

この章では，Web サービスセキュリティ機能が提供するコマンドの使い方について説明します。

4.1　共通鍵生成コマンド（CWSSCreateSecretKey）

共通鍵生成コマンドを使用して，暗号化機能を利用するときに必要な共通鍵を生成します。共通鍵生成コマンドを実行すると，共通鍵生成コマンドのオプションで指定したファイルに共通鍵が出力されます。共通鍵のファイルはバイナリー形式です。共通鍵生成コマンドの所在は次のとおりです。

＜Application Serverのインストールディレクトリ＞/wss/bin/ の下

出力された共通鍵の扱い方については，「付録 C.1　Web サービスセキュリティ機能定義ファイルの項目」の SecretKeyFile 要素を参照してください。

(1)　形式

共通鍵生成コマンドの形式を次に示します。

```
CWSSCreateSecretKey.bat -h | -a <アルゴリズム識別子> -o <出力ファイル名>
```

コマンド名（CWSSCreateSecretKey.bat）のあとに，一つ以上の空白を挿入し，オプションを指定します。オプションと指定する値の間も一つ以上の空白を挿入します。

(2)　オプション

共通鍵生成コマンドで指定するオプションを次に示します。

表 4-1　共通鍵生成コマンドのオプション

オプション	説明
-h	コマンドのオプション説明を表示します。
-a	生成する共通鍵のアルゴリズム識別子を指定します。 指定可能な値を次に示します。 "TRIPLEDES"： Triple DES ブロック暗号の共通鍵を生成します。 "AES128"： AES-128 ブロック暗号の共通鍵を生成します。
-o	生成した共通鍵を出力するファイル名を指定します。

(3)　メッセージ

コマンド実行時のメッセージについては，マニュアル「アプリケーションサーバ メッセージ(構築／運用／開発用)」を参照してください。

(4)　注意事項

- OS が Windows（x86 および x64）の場合だけ実行できます。ほかの OS では実行できません。
- 出力ファイル名に空白が含まれる場合は出力ファイル名を「"」で囲みます。
- 出力ファイル名にディレクトリを含まない場合は，カレントディレクトリに出力されます。
- 出力ファイル名にディレクトリを含めた場合は，既存のディレクトリを指定する必要があります。
- 出力ファイル名には既存のファイルを指定してはいけません。
- 同じオプションを 2 回以上指定すると，最後に指定したオプションの値が有効になります。次の例では，共通鍵は file2 に出力されます。

例

```
CWSSCreateSecretKey.bat -a AES128 -o file1 -o file2
```

4.2　定義ファイル構文チェックコマンド (CWSSConfCheck)

Web サービスセキュリティ機能定義ファイル，および Web サービスセキュリティ方針定義ファイルを作成したあと，設定した XML の構文が正しいかどうかをチェックするために，定義ファイル構文チェックコマンドを使用します。定義ファイル構文チェックコマンドの所在は次のとおりです。

＜Application Serverのインストールディレクトリ＞/wss/bin/ の下

(1)　形式

定義ファイル構文チェックコマンドの形式を次に示します。

```
CWSSConfCheck.bat -h | [-s | -p] -f <チェックする定義ファイル名>
```

コマンド名（CWSSConfCheck.bat）のあとに，一つ以上の空白を挿入し，オプションを指定します。オプションと指定する値の間も一つ以上の空白を挿入します。

(2)　オプション

構文チェックコマンドで指定するオプションを次に示します。

表 4-2　定義ファイル構文チェックコマンドのオプション

オプション	説明
-h	コマンドのオプション説明を表示します。
-s	-f オプションで指定するファイルが，Web サービスセキュリティ機能定義ファイルの場合に指定します。
-p	-f オプションで指定するファイルが，Web サービスセキュリティ方針定義ファイルの場合に指定します。
-f	チェック対象の定義ファイル名を指定します。

(3)　終了コード

構文チェックコマンドの終了コードを次に示します。

- 0：正常終了
- 1：エラー終了

(4)　メッセージ

コマンド実行時のメッセージについては，マニュアル「アプリケーションサーバ メッセージ(構築／運用／開発用)」を参照してください。

(5)　注意事項

- OS が Windows（x86 および x64）の場合だけ実行できます。ほかの OS では実行できません。
- 入力ファイル名に空白が含まれる場合は入力ファイル名を「"」で囲みます。
- 構文チェックで不正な構文があった場合，または内部エラーが発生した場合，エラーメッセージが標準出力に表示されます。

- 同じオプションを 2 回以上指定すると，最後に指定したオプションの値が有効になります。次の例では，file2 に対してだけ構文をチェックします。

  例

  ```
  CWSSConfCheck -s -f file1.xml -f file2.xml
  ```

5 Web サービスセキュリティ機能が提供する API

この章では, Web サービスセキュリティ機能が提供する API について説明します。

5.1　インタフェースおよびクラスの一覧

Web サービスセキュリティ機能を使用する場合，SOAP アプリケーション開発支援機能または JAX-WS 機能のどちらを使用して Web サービスを開発するかによって，使用できる API が異なります。Web サービスセキュリティ機能が提供するインタフェースおよびクラスの一覧を次に示します。

表 5-1　提供するインタフェースおよびクラスの一覧

インタフェース名およびクラス名	説明	サポート状況	
		SOAP アプリケーション開発支援機能	JAX-WS 機能
WSSElementProxyBuilder	セキュリティ項目操作クラスのビルダクラス	×※1	○
WSSElementProxyFactory	セキュリティ項目操作クラスのファクトリクラス	○	×※2
WSSElementProxy	セキュリティ項目操作クラス	○	○
WSSConstants	定数定義インタフェース	×	○
WSSUsernameToken	UsernameToken 要素の操作クラス	○	○
WSSUsernameToken.PasswordType	UsernameToken 要素のパスワード種別を示す列挙定数	○	○
WSSException	例外情報を保持するクラス	○	○

（凡例）

○：使用できます。

×：使用できません。

注※1

使用した場合，コンパイルは正常に終了しますが，実行時にエラーが発生します。

注※2

使用した場合，コンパイル時にエラーが発生します。

なお，これらの API を使用して SOAP 通信基盤上または JAX-WS エンジン上で動作する Web サービスまたは Web サービスクライアントを開発する場合は，クラスパスに＜ Application Server のインストールディレクトリ＞/wss/lib/cwssec.jar を追加してコンパイルする必要があります。

5.2　WSSElementProxyBuilder クラス

セキュリティ項目操作クラスのビルダクラスです。このクラスは，JAX-WS 機能で開発した Web サービスクライアントに対して使用できます。

クラス定義

```
public abstract class WSSElementProxyBuilder
```

パッケージ名

```
com.cosminexus.wss.element
```

WSSElementProxyBuilder クラスのメソッドを次の表に示します。

表 5-2　WSSElementProxyBuilder クラスのメソッド一覧

メソッド	機能概要
newInstance	WSSElementProxyBuilder クラスのインスタンスを生成します。
createWSSElementProxy	セキュリティ項目操作クラスのインスタンスを生成します。

次に，各メソッドの詳細について説明します。

newInstance

クラス名：WSSElementProxyBuilder

機能

WSSElementProxyBuilder クラスのインスタンスを生成します。

構文

```
public static WSSElementProxyBuilder newInstance() throws WSSException;
```

引数

ありません。

戻り値

WSSElementProxyBuilder クラスのインスタンスです。

例外

WSSException

Application Server の JAX-WS 機能による実行環境以外でこのメソッドが呼び出された場合にスローされます。メッセージ KDCGA9000-E の一部に，エラーの詳細として次の内容が表示されます。

WSSElementProxyBuilder class is not available outside of JAX-WS runtime environment.

createWSSElementProxy

クラス名：WSSElementProxyBuilder

機能

空のセキュリティ項目操作クラスのインスタンスを生成します。

このメソッドで取得したセキュリティ項目操作クラスに UsernameToken 要素クラスのインスタンスを設定します。その後，設定したインスタンスを Application Server の JAX-WS 機能のメッセージコンテキストに追加すると，設定内容に従って SOAP メッセージが生成されます。

構文

```
public abstract WSSElementProxy createWSSElementProxy();
```

引数

ありません。

戻り値

セキュリティ項目操作クラスのインスタンスです。

5.3　WSSElementProxyFactory クラス（セキュリティ項目操作クラスの生成）

セキュリティ項目操作クラスのファクトリクラスです。このクラスは，SOAP アプリケーション開発支援機能で開発した SOAP アプリケーションに対して使用できます。

クラス定義

```
public final class WSSElementProxyFactory
```

パッケージ名

```
com.cosminexus.wss.element
```

WSSElementProxyFactory クラスのメソッドを次の表に示します。

表 5-3　WSSElementProxyFactory クラスのメソッド一覧

メソッド	機能概要
newWSSElementProxy（スタブクラスから生成）	スタブクラスから空のセキュリティ項目操作クラスのインスタンスを生成します。
newWSSElementProxy（実装クラスから生成）	空のセキュリティ項目操作クラスのインスタンスを生成します。
getWSSElementProxy（スタブクラスから生成）	スタブクラスから，受信したメッセージのセキュリティ項目操作クラスのインスタンスを生成します。
getWSSElementProxy（実装クラスから生成）	受信したメッセージのセキュリティ項目操作クラスのインスタンスを生成します。

次に，各メソッドの詳細について説明します。

newWSSElementProxy（スタブクラスから生成）

クラス名：WSSElementProxyFactory

機能

スタブクラスから空のセキュリティ項目操作クラスのインスタンスを生成します。このメソッドの引数で指定したクライアントのインタフェースクラスのサービスメソッドを呼び出すと，セキュリティ項目操作クラスに設定した内容でセキュリティ要素を生成します。このメソッドは，呼び出すサービスメソッドの形態が RPC または EJB の場合に使用します。

構文

```
public static WSSElementProxy newWSSElementProxy (
     javax.xml.rpc.Stub a_Stub
) throws WSSException;
```

引数

表 5-4　newWSSElementProxy（スタブクラスから生成）メソッドの引数

仮引数名	名称	in/out	説明
a_Stub	スタブクラス	in	クライアントのインタフェースクラス(スタブクラス)を指定します。

戻り値

セキュリティ項目操作クラスのインスタンスです。

例外

WSSException
　処理中に予測しない例外が発生した場合にスローされます。

注意事項

- メソッドの引数の指定とは異なるクライアントのインタフェースクラスのサービスメソッドを呼び出した場合，セキュリティ項目操作クラスに設定した内容は反映しません。Web サービスセキュリティ機能定義ファイルに設定した内容に従って，セキュリティ項目を設定します。
- メソッドで取得したセキュリティ項目操作クラスが提供するメソッドを利用しないでサービスメソッドを呼び出した場合，Web サービスセキュリティ機能定義ファイルに設定した内容に従って，セキュリティ項目を設定します。
- 送信時に Web サービスセキュリティ機能を利用しない場合は，このメソッドで取得したセキュリティ項目操作クラスが提供するメソッドを設定しても，セキュリティ項目は SOAP メッセージに設定しません。Web サービスセキュリティ機能を利用しない場合を次に示します。
  - Web サービスセキュリティ機能定義ファイルが送信時にデプロイされていない場合
  - Web サービスセキュリティ機能定義ファイル中のリクエストメッセージ送信時の設定が省略されている場合

newWSSElementProxy（実装クラスから生成）

クラス名：WSSElementProxyFactory

機能

空のセキュリティ項目操作クラスのインスタンスを生成します。このメソッドは，スタブまたはメッセージクラスから生成するメソッドとは異なり，SOAP サービスの実装クラスでサービスの応答を送信する際に使用します。このメソッドで取得したセキュリティ項目操作クラスが提供するメソッドを利用すると，サービスの応答をクライアントに送信する際に，セキュリティ項目操作クラスに設定した内容でセキュリティ要素を生成します。

構文

```
public static WSSElementProxy newWSSElementProxy (
) throws WSSException;
```

引数

ありません。

戻り値

セキュリティ項目操作クラスのインスタンスです。

例外

WSSException
　処理中に予測しない例外が発生した場合にスローされます。

注意事項

- メソッドで取得したセキュリティ項目操作クラスが提供するメソッドを利用しない場合，Web サービスセキュリティ機能定義ファイルに設定した内容に従って，レスポンスメッセージ内のセキュリティ項目を設定します。
- メソッドをリクエスト受信処理以外の場所で呼び出した場合は null を返します。
- 送信時に Web サービスセキュリティ機能を利用しない場合は，このメソッドで取得したセキュリティ項目操作クラスが提供するメソッドを利用しても，セキュリティ項目は設定しません。Web サービスセキュリティ機能を利用しない場合を次に示します。
  - Web サービスセキュリティ機能定義ファイルが送信時にデプロイされていない場合
  - Web サービスセキュリティ機能定義ファイル中のリクエストメッセージ送信時の設定が省略されていた場合

getWSSElementProxy（スタブクラスから生成）

クラス名：WSSElementProxyFactory

機能

スタブクラスから，受信したメッセージのセキュリティ項目操作クラスのインスタンスを生成します。このメソッドの引数で指定したクライアントのインタフェースクラスのサービスメソッドの呼び出し後にこのメソッドを呼び出すと，サービスメソッドのレスポンスメッセージに含まれるセキュリティ項目を取得し，セキュリティ項目操作クラスを生成します。その後，このメソッドで取得したセキュリティ項目操作クラスのメソッドを呼び出すことで，セキュリティ項目を取得できます。

構文

```
public static WSSElementProxy[] getWSSElementProxy (
     javax.xml.rpc.Stub a_Stub
) throws WSSException;
```

引数

表 5-5　getWSSElementProxy（スタブクラスから生成）メソッドの引数

仮引数名	名称	in/out	説明
a_Stub	スタブクラス	in	RPC または EJB を利用した SOAP アプリケーションの場合で，SOAP アプリケーション開発支援機能に

仮引数名	名称	in/out	説明
a_Stub	スタブクラス	in	よって作成されるクライアントのインタフェースクラス（スタブクラス）を指定します。

戻り値

セキュリティ項目操作クラスのインスタンス配列です。セキュリティ項目操作クラスが生成できない場合は null を返します。

例外

WSSException

処理中に予測しない例外が発生した場合にスローされます。

注意事項

- このメソッドは，引数に指定したスタブクラスのサービスメソッドの呼び出し直後に必ず呼び出すようにしてください。引数に指定したものと異なるスタブクラスのサービスメソッド呼び出し直後にこのメソッドを呼び出した場合，引数に指定したスタブクラスのサービスメソッドの応答メッセージに含まれるセキュリティ項目は取得されません。
- メソッドで取得するセキュリティ項目は，このメソッドを呼び出す直前に呼び出したサービスメソッドのレスポンスメッセージに含まれるセキュリティ項目です。サービスメソッドを複数回呼び出した場合，最後のサービスメソッド呼び出しのレスポンスメッセージに含まれるセキュリティ項目を取得します。サービスメソッドを一度も呼び出さないで，このメソッドを呼び出した場合は null を返します。
- 戻り値であるセキュリティ項目クラスの配列の順序と，レスポンスメッセージに含まれるセキュリティ項目の順序は必ずしも一致しません。

getWSSElementProxy（実装クラスから生成）

クラス名：WSSElementProxyFactory

機能

受信したメッセージのセキュリティ項目操作クラスのインスタンスを生成します。このメソッドは，スタブまたはメッセージクラスから生成するメソッドとは異なり，SOAP アプリケーションの実装クラスでサービスのリクエストメッセージに含まれるセキュリティ項目を取得する際に使用します。SOAP アプリケーションの実装クラスでこのメソッドを呼び出すと，リクエストメッセージに含まれるセキュリティ項目を取得し，セキュリティ項目操作クラスを生成します。その後，このメソッドで取得したセキュリティ項目操作クラスのメソッドを呼び出すことで，セキュリティ項目を取得できます。

構文

```
public static WSSElementProxy[] getWSSElementProxy (
) throws WSSException;
```

引数

ありません。

戻り値

セキュリティ項目操作クラスのインスタンス配列です。セキュリティ項目操作クラスが生成できない場合は null を返します。

例外

WSSException
　処理中に予測しない例外が発生した場合にスローされます。

注意事項

- このメソッドで取得するセキュリティ項目は，Web サービスセキュリティ機能が処理するセキュリティ項目です。
- 戻り値であるセキュリティ項目クラスの配列の順序と，リクエストメッセージに含まれるセキュリティ項目の順序は必ずしも一致しません。

5.4　WSSElementProxy クラス（セキュリティ項目の操作）

セキュリティ項目の操作クラスです。

クラス定義

```
public final class WSSElementProxy
```

パッケージ名

```
com.cosminexus.wss.element
```

WSSElementProxy クラスのメソッドを次の表に示します。

表 5-6　WSSElementProxy クラスのメソッド一覧

メソッド名称	説明
getWSSUsernameToken	UsernameToken 要素クラスのインスタンスを取得します。
setWSSUsernameToken	UsernameToken 要素クラスのインスタンスをセキュリティ項目操作クラスのインスタンスに設定します。
removeWSSUsernameToken	UsernameToken 要素クラスのインスタンスをセキュリティ項目操作クラスのインスタンスから削除します。
getRole	セキュリティ項目の role 属性を取得します。
setRole	セキュリティ項目の role 属性を設定します。

getWSSUsernameToken

クラス名：WSSElementProxy

機能

UsernameToken 要素クラスのインスタンスを取得します。

構文

```
public WSSUsernameToken[] getWSSUsernameToken (
) throws WSSException;
```

引数

ありません。

戻り値

UsernameToken 要素クラスのインスタンス配列です。

例外

WSSException

処理中に予測しない例外が発生した場合にスローされます。

注意事項

- セキュリティ項目操作クラスのファクトリクラスの，newWSSElementProxy メソッドによって取得した WSSElementProxy クラスのインスタンスの場合，このメソッドの戻り値は null です。ただし，setWSSUsernameToken を呼んだ後，このメソッドを呼び出すと setWSSUsernameToken で指定した値を返します。
- 戻り値の UsernameToken 要素クラスのインスタンス配列の順序と，メッセージ中の UsernameToken 要素の順序とは必ずしも一致しません。

setWSSUsernameToken

クラス名：WSSElementProxy

機能

UsernameToken 要素クラスのインスタンスをセキュリティ項目操作クラスに設定します。

構文

```
public void setWSSUsernameToken (
     WSSUsernameToken a_UsernameToken
) throws WSSException;
```

引数

表 5-7　setWSSUsernameToken メソッドの引数

仮引数名	名称	in/out	説明
a_UsernameToken	UsernameToken 要素クラス	in	セキュリティ項目操作クラスに設定する UsernameToken 要素操作クラスのインスタンスを指定します。

戻り値

ありません。

例外

WSSException
　処理中に予測しない例外が発生した場合にスローされます。

注意事項

引数に null を指定した場合，セキュリティ項目操作クラスに UsernameToken 要素クラスを設定しないで正常終了します。すでに UsernameToken 要素クラスを設定している場合は，元の UsernameToken 要素クラスも更新しません。

removeWSSUsernameToken

クラス名：WSSElementProxy

機能

UsernameToken 要素クラスのインスタンスをセキュリティ項目操作クラスから削除します。

構文

```
public void removeWSSUsernameToken (
) throws WSSException;
```

引数

ありません。

戻り値

ありません。

例外

WSSException

　処理中に予測しない例外が発生した場合にスローされます。

注意事項

UsernameToken 要素クラスのインスタンスがセキュリティ項目操作クラスに存在しない場合，何も処理をしないで正常終了します。

getRole

クラス名：WSSElementProxy

機能

セキュリティ項目の role 属性値を取得します。

構文

```
public java.net.URI getRole (
) throws WSSException;
```

引数

ありません。

戻り値

セキュリティ項目に含まれる role 属性値です。セキュリティ項目に role 属性がない場合は null を返します。

例外

ありません。

setRole

クラス名：WSSElementProxy

機能

セキュリティ項目の role 属性値を設定します。

構文

```
public void setRole (
     java.net.URI a_Role
) throws WSSException;
```

引数

表 5-8　setRole メソッドの引数

仮引数名	名称	in/out	説明
a_Role	role 属性値	in	セキュリティ項目操作クラスに設定する role 属性値を指定します。

戻り値

ありません。

例外

ありません。

5.5　WSSConstants インタフェース

定数定義インタフェースです。このインタフェースは，JAX-WS 機能で開発した Web サービスクライアントおよび Web サービスに対して使用できます。

インタフェース定義

```
public interface WSSConstants
```

パッケージ名

```
com.cosminexus.wss.element
```

(1)　WSS_SEND_ELEMENTPROXY プロパティ

Application Server の JAX-WS 機能のメッセージコンテキストのプロパティです。このプロパティは，送信する SOAP メッセージのセキュリティヘッダに UsernameToken 要素を設定する場合に必要です。このプロパティをキーにして，JAX-WS 機能のメッセージコンテキストに com.cosminexus.wss.element.WSSElementProxy クラスのオブジェクトを設定します。

注意事項

次の場合は，Web サービスセキュリティ機能定義ファイルに設定した内容に従って，セキュリティ項目を設定します。

- WSS_SEND_ELEMENTPROXY プロパティを使用して，セキュリティ項目操作クラス以外のインスタンスをメッセージコンテキストに設定した場合
- メッセージコンテキストに設定したセキュリティ項目操作クラスのインスタンスに，UsernameToken 要素クラスのインスタンスを設定しないで Web サービスを呼び出した場合

Web サービスセキュリティ機能定義ファイル中のリクエストメッセージ送信時の設定が省略されていると，送信時に Web サービスセキュリティ機能は適用されません。この場合，メッセージコンテキストにセキュリティ項目操作クラスが提供するメソッドを設定しても，SOAP メッセージにセキュリティ項目は設定されません。

(2)　WSS_RECV_ELEMENTPROXY プロパティ

Application Server の JAX-WS 機能のメッセージコンテキストのプロパティです。このプロパティは，受信する SOAP メッセージのセキュリティヘッダから UsernameToken 要素を取得する場合に必要です。このプロパティをキーにして，JAX-WS 機能のメッセージコンテキストから java.util.List<WSSElementProxy>クラスのオブジェクトを取得します。

注意事項

- WSS_RECV_ELEMENTPROXY プロパティを使用して取得した値は変更できません。
- WSS_RECV_ELEMENTPROXY プロパティを使用してメッセージコンテキストから取得したセキュリティ項目は，Web サービスセキュリティ機能によって処理されます。
- java.util.List<WSSElementProxy>クラスから取得したセキュリティ項目操作クラスの順序は，リクエストメッセージに含まれるセキュリティ項目の順序と異なる場合があります。
- 受信したリクエストメッセージに UsernameToken 要素が含まれていない場合，WSS_RECV_ELEMENTPROXY プロパティを参照すると null を返します。

5.6　WSSUsernameToken クラス (UsernameToken 要素の操作)

UsernameToken 要素の操作クラスです。

クラス定義

```
public final class WSSUsernameToken extends
     com.cosminexus.wss.element.WSSElementBase
```

パッケージ名

```
com.cosminexus.wss.element
```

WSSUsernameToken クラスのメソッドを次の表に示します。

表 5-9　WSSUsernameToken クラスのメソッド一覧

メソッド名称	説明
コンストラクタ	UsernameToken 要素クラスのコンストラクタです。
getUsername	ユーザー名を取得します。
setUsername	ユーザー名を設定します。
getPassword	パスワードを取得します。
setPassword	パスワードを設定します。
getId	UsernameToken 要素の Id 属性を取得します。
setId	UsernameToken 要素の Id 属性を設定します。
getPasswordType	パスワード種別を取得します。
setPasswordType	パスワード種別を設定します。
getNonce	Nonce 属性を取得します。
getCreated	Created 属性を取得します。

注意事項

- セキュリティ項目操作クラスのインスタンスを，セキュリティ項目操作クラスのファクトリクラスの newWSSElementProxy メソッドによって生成した場合，このクラスのメソッドで指定した内容で UsernameToken 要素を生成します。ただし，Web サービスセキュリティ機能定義ファイルに，呼び出すサービスメソッドに対応する SenderPortConfig 要素，RoleConfig 要素の指定がない場合は UsernameToken 要素を生成しません。
- セキュリティ項目操作クラスのインスタンスを，セキュリティ項目操作クラスのファクトリクラスの getWSSElementProxy メソッドによって生成した場合，このクラスの名称が set で始まるメソッドで値を設定しても，UsernameToken 要素は生成しません。
- Web サービスセキュリティ機能定義ファイルに UsernameToken 要素を定義している場合，このメソッドが生成した UsernameToken 要素とは別に，複数の UsernameToken 要素を生成します。

コンストラクタ

機能

UsernameToken 要素クラスのコンストラクタです。

構文

```
public WSSUsernameToken (
     java.lang.String a_Username
) throws WSSException;
```

引数

表 5-10　コンストラクタの引数

仮引数名	名称	in/out	説明
a_Username	ユーザー名	in	ユーザー名を示す文字列を指定します。

戻り値

UsernameToken 要素クラスのインスタンスです。

例外

WSSException

引数に空文字""または null が指定されました。

注意事項

このクラスが生成される際の UsernameToken 要素の初期値を次に示します。

表 5-11　UsernameToken 要素の初期値

クラスの要素	初期値 1 ※1	初期値 2 ※2
Username	コンストラクタの引数で指定したユーザー名	UsernameToken 要素のユーザー名
Password	null	UsernameToken 要素のパスワード
PasswordType	WSSUsernameToken.PasswordType.TEXT	WSSUsernameToken.PasswordType.TEXT または WSSUsernameToken.PasswordType.DIGEST
Id	null	UsernameToken 要素の Id 属性
Nonce	null	UsernameToken 要素の Nonce 属性
Created	null	UsernameToken 要素の Created 属性

注※1

コンストラクタを使用した場合

注※2

このクラスをセキュリティ項目操作クラスの getWSSUsernameToken メソッドで生成した場合

- 引数に null を設定した場合，WSSException 例外が発生します。
- 引数に空文字""を設定した場合，WSSException 例外が発生します。
- 引数に一つ以上の空白文字を設定した場合，ユーザー名の情報は一つ以上の空白文字に置き換えられます。

getUsername

クラス名：WSSUsernameToken

機能

UsernameToken 要素のユーザー名を取得します。

構文

```
public java.lang.String getUsername (
);
```

引数

ありません。

戻り値

UsernameToken 要素のユーザー名です。

例外

ありません。

setUsername

クラス名：WSSUsernameToken

機能

UsernameToken 要素のユーザー名を設定します。

構文

```
public void setUsername (
     java.lang.String a_Username
) throws WSSException;
```

引数

表 5-12　setUsername メソッドの引数

仮引数名	名称	in/out	説明
a_Username	ユーザー名	in	ユーザー名を示す文字列を指定します。

戻り値

ありません。

例外

WSSException

　引数に空文字""が指定されました。

注意事項

- 引数に null を設定した場合，このクラスが保持するユーザー名の情報は更新しません。
- 引数に一つ以上の空白文字を設定した場合，このクラスが保持するユーザー名の情報は一つ以上の空白文字に置き換えられます。

getPassword

クラス名：WSSUsernameToken

機能

UsernameToken 要素のパスワードを取得します。

構文

```
public char[] getPassword (
);
```

引数

ありません。

戻り値

UsernameToken 要素のパスワードです。UsernameToken 要素にパスワードがない場合は null を返します。

例外

ありません。

注意事項

- セキュリティ項目操作クラスのインスタンスを，セキュリティ項目操作クラスのファクトリクラスの newWSSElementProxy メソッドによって生成した際，UsernameToken 要素内に Password 要素が存在しない場合は null を返します。ただし，すでに setPassword メソッドでパスワードを設定している場合はその値を返します。
- 受信メッセージのセキュリティ項目の，UsernameToken 要素内の Password 要素が空要素の場合は null を返します。
- セキュリティ項目操作クラスのインスタンスを，セキュリティ項目操作クラスのファクトリクラスの getWSSElementProxy メソッドによって生成した際，受信メッセージ内の Password 要素の種別が平文（SOAP メッセージの PasswordText 要素）の場合は平文のパスワードを返します。Password 要素の種別がダイジェスト（SOAP メッセージの PasswordDigest 要素）の場合はダイジェスト値をそのまま返します。

setPassword

クラス名：WSSUsernameToken

機能

UsernameToken 要素のパスワードを設定します。

構文

```
public void setPassword (
     char[] a_Password
);
```

引数

表 5-13　setPassword メソッドの引数

仮引数名	名称	in/out	説明
a_Password	パスワード	in	パスワードを指定します。

戻り値

ありません。

例外

ありません。

注意事項

- 引数に null または空文字""を設定した場合，このクラスに設定したパスワードの情報は更新されません。
- 引数に空白文字を設定した場合，このクラスに設定したパスワードの情報は空白文字に置き換えられます。

getId

クラス名：WSSUsernameToken

機能

UsernameToken 要素の Id 属性を取得します。

構文

```
public java.lang.String getId (
);
```

引数

ありません。

戻り値

UsernameToken 要素の Id 属性値を示す文字列です。UsernameToken 要素に Id 属性がない場合は null を返します。

例外

ありません。

注意事項

- このメソッドで取得するのは，UsernameToken 要素の Id 属性です。
- セキュリティ項目操作クラスのインスタンスを，セキュリティ項目操作クラスのファクトリクラスの newWSSElementProxy メソッドによって生成した場合は null を返します。ただし，すでに setId メソッドで Id を設定している場合はその値を返します。

setId

クラス名：WSSUsernameToken

機能

UsernameToken 要素の Id 属性を設定します。

構文

```
public void setId (
     java.lang.String a_Id
);
```

引数

表 5-14　setId メソッドの引数

仮引数名	名称	in/out	説明
a_Id	Id	in	Id 属性を示す文字列を指定します。

戻り値

ありません。

例外

ありません。

注意事項

このメソッドで設定した Id 属性値は UsernameToken 要素の Id 属性に設定されます。ただし，ほかの要素の Id 属性値と重複しているかどうかはチェックしません。

getPasswordType

クラス名：WSSUsernameToken

機能

UsernameToken 要素のパスワード種別を取得します。

構文

```
public WSSUsernameToken.PasswordType getPasswordType (
);
```

引数

ありません。

戻り値

UsernameToken 要素のパスワード種別です。UsernameToken 要素にパスワードがない場合は，WSSUsernameToken.PasswordType.TEXT を返します。

表 5-15　WSSUsernameToken.PasswordType クラスの列挙値

列挙値	意味
WSSUsernameToken.PasswordType.TEXT	平文形式のパスワードです。 (http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-username-token-profile-1.0#PasswordText 形式)
WSSUsernameToken.PasswordType.DIGEST	ダイジェスト形式のパスワードです。 (http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-username-token-profile-1.0#PasswordDigest 形式)

例外

ありません。

注意事項

セキュリティ項目操作クラスのインスタンスを，セキュリティ項目操作クラスのファクトリクラスの newWSSElementProxy メソッドによって生成した場合，WSSUsernameToken.PasswordType.TEXT を返します。ただし，すでに setPasswordType メソッドでパスワード種別を設定している場合は，その値を返します。

setPasswordType

クラス名：WSSUsernameToken

機能

UsernameToken 要素のパスワード種別を設定します。

構文

```
public void setPasswordType (
     WSSUsernameToken.PasswordType a_PasswordType
);
```

引数

表 5-16　setPasswordType メソッドの引数

仮引数名	名称	in/out	説明
a_PasswordType	パスワード種別	in	パスワード種別を示す，WSSUsernameToken.PasswordType の列挙値を指定します。

戻り値

ありません。

例外

ありません。

getNonce

クラス名：WSSUsernameToken

機能

UsernameToken 要素の Nonce 属性の内容を取得します。

構文

```
public java.lang.String getNonce (
);
```

引数

ありません。

戻り値

UsernameToken 要素の Nonce 属性値です。UsernameToken 要素に Nonce 属性がない場合は null を返します。

例外

ありません。

注意事項

セキュリティ項目操作クラスのインスタンスを，セキュリティ項目操作クラスのファクトリクラスの newWSSElementProxy メソッドによって生成した際，UsernameToken 要素内に Nonce 属性がない場合は null を返します。

getCreated

クラス名：WSSUsernameToken

機能

UsernameToken 要素の Created 属性の内容を取得します。

構文

```
public java.lang.String getCreated (
);
```

引数

ありません。

戻り値

UsernameToken 要素の Created 属性文字列です。UsernameToken 要素に Created 属性がない場合は null を返します。

例外

ありません。

注意事項

セキュリティ項目操作クラスのインスタンスを，セキュリティ項目操作クラスのファクトリクラスの newWSSElementProxy メソッドによって生成した際，UsernameToken 要素内に Created 属性がない場合は null を返します。

5.7　WSSUsernameToken.PasswordType インタフェース（PasswordType 要素の操作）

PasswordType 要素の操作クラスです。

クラス定義

```
public static final class WSSUsernameToken.PassswordType
```

パッケージ名

```
com.cosminexus.wss.element
```

機能

WSSUsernameToken クラスの getPasswordType()，setPasswordType()メソッドの戻り値および引数で指定する，PasswordType 要素を示す列挙定数です。次にパスワード種別を示します。

TEXT

パスワード種別が平文形式（http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText）であることを示します。

DIGEST

パスワード種別がダイジェスト形式（http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-username-token-profile-1.0#PasswordDigest）であることを示します。

5.8　WSSException クラス（例外情報の取得）

Web サービスセキュリティ機能が提供する API で発生する例外クラスです。

クラス定義

```
public final class WSSException extends
     java.lang.Exception
```

パッケージ名

```
com.cosminexus.wss.faults
```

WSSException クラスのメソッドを次の表に示します。

表 5-17　WSSException クラスのメソッド一覧

メソッド名称	説明
getMessage	詳細メッセージを取得します。

getMessage

クラス名：WSSException

機能

例外の詳細メッセージを取得します。

構文

```
public java.lang.String getMessage (
);
```

引数

ありません。

戻り値

詳細メッセージを示す文字列です。

6 障害対策

この章では，Web サービスセキュリティ機能の実行，および運用中の障害対策のために出力される情報について説明します。

6.1　トレースを収集する

Web サービスセキュリティ機能では，障害対策に必要な情報をトレースとして出力します。トレースは障害の発生個所の割り出しや原因の調査などに利用します。Web サービスセキュリティ機能で出力するトレースの種類を次に示します。

表 6-1　Web サービスセキュリティ機能が出力するトレースの種類

トレースの種類	説明
サーバトレース	Web サービスセキュリティ機能のサーバ処理部分が出力します。
クライアントトレース	Web サービスセキュリティ機能のクライアント処理部分が出力します。
コマンドトレース	Web サービスセキュリティ機能のコマンドラインインタフェースが出力します。

6.1.1　トレースの内容

サーバトレース，クライアントトレース，およびコマンドトレースが出力するトレースの内容と出力例を次に示します。

(1)　出力内容

表 6-2　トレースの出力内容

項目	内容
日付	出力時の日付（yyyy/mm/dd 形式）が出力されます。
時刻	出力時の時刻（hh:mm:ss.sss 形式）が出力されます。
アプリケーション名	「wss」が出力されます。
プロセス識別子	プロセス識別子（16 進数）が出力されます。
スレッド識別子	スレッド識別子（16 進数）が出力されます。
メッセージ ID	メッセージ ID が出力されます。メッセージ ID を持たないものは出力されません。
メッセージ種別	OC：メソッドの入口を表します。 OD：メソッドの出口を表します。 EC：例外をキャッチしたことを表します。 ER：エラーメッセージを表示したことを表します。 FB：ほかのプログラムの処理の呼び出しを表します。 FE：呼び出したほかのプログラムの処理の終了を表します。 PB：Web サービスセキュリティ機能が提供する API の開始を表します。 PE：Web サービスセキュリティ機能が提供する API の終了を表します。
メッセージテキスト	実行時の情報を示すメッセージが出力されます。

(2)　出力例

図 6-1　トレースの出力例

```
yyyy/mm/dd hh:mm:ss.sss       pid  tid       message-id  message(LANG=0x0411)
0001 2002/01/17 18:09:54.224  wss  000006CC  J00737FE    0C  enter WSSRequestSenderHandler::init()
0002 2002/01/17 18:09:54.224  wss  000006CC  J00737FE    0D  exit WSSRequestSenderHandler::init()
```

6.1.2　トレースの出力先

トレースの出力先を次に示します。

表 6-3　トレースの出力先

トレースの種類	出力先
サーバトレース	SOAP 通信基盤または JAX-WS エンジンのサーバ側で出力するトレースと同じファイル上に出力されます。 JAX-WS エンジンの場合，JAX-WS 機能の稼働ログにメッセージが，例外ログにスタックトレースが，保守ログに保守情報が出力されます。
クライアントトレース	SOAP 通信基盤または JAX-WS エンジンのクライアント側で出力するトレースと同じファイル上に出力されます。 JAX-WS エンジンの場合，JAX-WS 機能の稼働ログにメッセージが，例外ログにスタックトレースが，保守ログに保守情報が出力されます。
コマンドトレース	コマンドトレースは，＜Application Server のインストールディレクトリ＞¥CC¥client¥logs¥system¥ejbcl¥WS に，次のファイル名で出力されます。 • 共通鍵生成コマンドの場合 CWSSCreateSecretKey-n.log (n:1～2) • 定義ファイル構文チェックコマンドの場合 CWSSConfCheck-n.log (n:1～2)

注

SOAP 通信基盤のトレース出力先とモードについては，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。JAX-WS エンジンのトレース出力先と動作定義ファイルについてはマニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

6.1.3　トレースの重要度

トレースの重要度を変更することで，出力するトレースの情報量を変更できます。出力する情報量を多くすることで，障害発生の要因を特定しやすくなります。ただし，出力する情報量を多くすると，プログラムの処理性能への影響が大きくなります。

トレースの重要度には次のレベルがあります。

- ERROR
- WARN
- INFO
- DEBUG

Web サービスセキュリティ機能で出力するトレースの重要度は，SOAP 通信基盤または JAX-WS 機能の設定に従います。JAX-WS 機能の場合の詳細を次の表に示します。

表 6-4 JAX-WS 機能を利用している場合のトレースの重要度

トレースの種類	出力内容	有効となる重要度
サーバトレース	メッセージ	J2EE サーバ上で動作させた場合の JAX-WS 機能の稼働ログの重要度
	スタックトレース	J2EE サーバ上で動作させた場合の JAX-WS 機能の例外ログの重要度
	保守情報	J2EE サーバ上で動作させた場合の JAX-WS 機能の保守ログの重要度
クライアントトレース	メッセージ	J2EE サーバ上またはコマンドインタフェースで動作させた場合の JAX-WS 機能の稼働ログの重要度
	スタックトレース	J2EE サーバ上またはコマンドインタフェースで動作させた場合の JAX-WS 機能の例外ログの重要度
	保守情報	J2EE サーバ上またはコマンドインタフェースで動作させた場合の JAX-WS 機能の保守ログの重要度

JAX-WS 機能のログの重要度の詳細については，マニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。また，SOAP 通信基盤を利用している場合は，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。

付録

付録 A　標準仕様への対応

Web サービスセキュリティ機能は，次の仕様に従っています。

- WS-Security 仕様
  - 07-60 以降のバージョン
    Web Services Security: SOAP Message Security 1.1
  - 07-60 より前のバージョン
    Web Services Security: SOAP Message Security Working Draft 17
- WS-Policy 1.5 仕様
  - Web Services Policy 1.5 - Framework
  - Web Services Policy 1.5 - Attachment
- WS-SecurityPolicy 1.3 仕様
- XML 署名標準仕様（2002/2/12　W3C 勧告）
- XML 暗号標準仕様（2002/12/10　W3C 勧告）

WS-Security 仕様および WS-SecurityPolicy 1.3 仕様の詳細については，OASIS のホームページを参照してください。また，XML 署名および XML 暗号の標準仕様，および WS-Policy 1.5 仕様の詳細については，W3C のホームページを参照してください。ここでは，各仕様のうち，Web サービスセキュリティ機能がサポートしている範囲を説明します。

注意事項

- 07-60 以降のバージョンでは，Web Services Security: SOAP Message Security 1.1 以外の仕様を実装した他社製品とは接続できない可能性があります。
- 07-60 より前のバージョンでは，Web Services Security: SOAP Message Security Working Draft 17 以外の仕様を実装した他社製品とは接続できない可能性があります。

付録 A.1　WS-Security 仕様のサポート範囲

Web サービスセキュリティ機能がサポートする WS-Security 仕様の範囲を次の表に示します。

表 A–1　WS-Security 仕様のサポート範囲

項番	WS-Security 仕様			推奨レベル	サポートの有無
	大分類	中分類	小分類		
1	SecurityToken	UsernameToken	PasswordText	推奨	○
2			PasswordDigest	推奨	○
3			Nonce	推奨	○
4			Created	任意	○
5		BinarySecurityToken	X.509v3	任意	○
6			X509PKIPathv1	任意	×
7			PKCS7	任意	×

項番	WS-Security 仕様			推奨レベル	サポートの有無
	大分類	中分類	小分類		
8	SecurityToken	BinarySecurityToken	Kerberos5TGT	任意	×
9			Kerberos5ST	任意	×
10		SAML Assertion	−	任意	×
11		XrML	−	任意	×
12		XCBF	−	任意	×
13		EncryptedData Token	−	任意	×
14	TokenReference	Direct Reference	−	推奨	○
15		KeyIdentifiers	−	推奨	○
16		Embedded Reference	−	任意	×
17		KeyNames	−	任意	×
18		ds:KeyInfo	−	推奨	○
19		Encrypted Key Reference	−	任意	×
20	Signature	Algorithms	−	推奨	○
21		Signing Messages	−	必須	○
22		Signing Tokens	−	任意	△※
23		Signature Validation	−	必須	○
24		Signature Confirmation	−	任意	×
25	Encryption	xenc:ReferenceList	−	推奨	×
26		xenc:EncryptedKey	−	推奨	○
27		Encrypted Header	−	任意	×
28		Processing Rules	−	必須	○
29	Security TimeStamp	−	−	推奨	○
30	Error Handling	−	−	推奨	○

（凡例）

○：サポートあり

×：サポートなし

−：該当しない

注※

SecurityToken 要素を直接署名できますが，STR Transform を非サポートのため，SecurityTokenReference 要素を署名できません。

付録 A.2　XML 署名標準仕様のサポート範囲

Web サービスセキュリティ機能がサポートする XML 署名標準仕様の範囲を次の表に示します。

表 A-2　XML 署名標準仕様のサポート範囲

項番	役割	項目	推奨レベル	サポートの有無
1	ダイジェスト値計算	SHA1	必須	○
2	符号化	base64（エンコード）	必須	○
3	署名値計算	HMAC-SHA1	必須	×
4		DSAwithSHA1	必須	○
5		RSAwithSHA1	推奨	○
6	正規化処理	Canonical XML（コメント付き）	推奨	○
7		Canonical XML（コメントなし）	必須	○
8		Exclusive Canonical XML（コメント付き）	任意	○
9		Exclusive Canonical XML（コメントなし）	任意※1	○
10	変換処理	Canonical XML（コメント付き）	推奨	○
11		Canonical XML（コメントなし）	必須	○
12		Exclusive Canonical XML（コメント付き）	任意	○
13		Exclusive Canonical XML（コメントなし）	任意※1	○
14		base64（変換アルゴリズム）	必須	×
15		XSLT	任意	×
16		XPath	推奨	×
17		XPath Filter 2.0	任意	×
18		Enveloped Signature	必須	×
19		STR Dereference※2	推奨	×

（凡例）

○：サポートあり

×：サポートなし

注※1

WS-Security 仕様では，サポートを推奨しています。

注※2

XML 署名標準仕様で規定されている変換処理ではなく，WS-Security 仕様で規定されているものです。

付録 A.3　XML 暗号標準仕様のサポート範囲

Web サービスセキュリティ機能がサポートする XML 暗号標準仕様の範囲を次の表に示します。

表 A-3　XML 暗号標準仕様のサポート範囲

項番	項目	推奨レベル	サポートの有無
1	Triple DES	必須	○
2	AES-128	必須	○
3	AES-192	任意	×
4	AES-256	必須	×
5	RSA-v1.5	必須	×
6	RSA-OAEP	必須	×
7	Diffie-Hellman Key Values	任意	×
8	Diffie-Hellman Key Agreement	任意	×
9	TRIPLEDES 鍵ラッピング	必須	○
10	AES-128 鍵ラッピング（128bit 鍵）	必須	○
11	AES-192 鍵ラッピング	任意	×
12	AES-256 鍵ラッピング（256bit 鍵）	必須	×
13	XML Decryption Transformation	推奨※	×

（凡例）

○：サポートあり

×：サポートなし

注※

WS-Security 仕様で規定されている推奨レベルです。

付録 A.4　WS-Policy 1.5 仕様のサポート範囲

ここでは，Web サービスセキュリティ機能がサポートする WS-Policy 1.5 仕様の範囲を説明します。

(1)　WS-Policy 1.5 - Framework 仕様のサポート範囲

Web サービスセキュリティ機能がサポートする WS-Policy 1.5 - Framework 仕様の範囲を次の表に示します。

表 A-4　WS-Policy 1.5 - Framework 仕様のサポート範囲

該当個所※	大分類	中分類	小分類	要素と属性（XPath 形式）	サポートの有無
4.1	Policy Expression	Normal Form Policy Expression	－	－	○
4.2		Policy Identification	－	/wsp:Policy/@Name	○
				/wsp:Policy/@wsu:Id	×
				/wsp:Policy/@xsd:ID	×
4.3.1		Compact Policy Expression	Optional Policy Assertions	－	×
4.3.2			Policy Assertion Nesting	－	○
4.3.3			Policy Operators	－	×
4.3.4			Policy References	/wsp:PolicyReference	○
				/wsp:PolicyReference/@URI	○
				/wsp:PolicyReference/@Digest	×
				/wsp:PolicyReference/@DigestAlgorithm	×
4.3.5			Policy Inclusion	－	×
4.3.6			Normalization	－	×
4.4		Ignorable Policy Assertions	－	－	×
4.5		Policy Intersection	－	－	×
4.6		Use of IRIs in Policy Expressions	－	－	×

（凡例）

○：サポートあり

×：サポートなし

－：該当する分類，要素，または属性なし

注※

WS-Policy 1.5 - Framework 仕様の該当個所（章節項）を示します。

(2)　WS-Policy 1.5 - Attachment 仕様のサポート範囲

Web サービスセキュリティ機能がサポートする WS-Policy 1.5 - Attachment 仕様の範囲を次の表に示します。

表 A-5　WS-Policy 1.5 - Attachment 仕様のサポート範囲

該当個所※	大分類	中分類	小分類または内容	サポートの有無
3.1	Policy Attachment	Effective Policy	−	○
3.2		Policy Attachment Mechanisms	−	○
3.3		XML Element Attachment	−	×
3.4		External Policy Attachment	−	×
4.1	Attaching Policies Using WSDL 1.1	Calculating Effective Policy in WSDL 1.1	wsdl:required="true"	×
4.1.1			Service Policy Subject	×
4.1.2			Endpoint Policy Subject	○
4.1.3			Operation Policy Subject	×
4.1.4			Message Policy Subject	○
5	WS-Policy Attachment for WSDL 2.0	−	−	×
6	Attaching Policies Using UDDI	−	−	×

（凡例）

○：サポートあり

×：サポートなし

−：該当する分類または内容なし

注※

WS-Policy 1.5 - Attachment 仕様の該当個所（章節項）を示します。

付録 A.5　WS-SecurityPolicy 1.3 仕様のサポート範囲

ここでは，Web サービスセキュリティ機能がサポートする WS-SecurityPolicy 1.3 仕様の範囲を説明します。

(1)　Protection Assertions のサポート範囲

WS-SecurityPolicy 1.3 仕様の中の Protection Assertions のうち，Web サービスセキュリティ機能がサポートする範囲を次の表に示します。

表 A-6　Protection Assertions のサポート範囲

該当個所※	大分類	小分類	アサーション (XPath 形式)	サポートの有無
4.1.1	Integrity Assertions	SignedParts Assertion	/sp:SignedParts	○
			/sp:SignedParts/sp:Body	○
			/sp:SignedParts/sp:header	×

該当個所※	大分類	小分類	アサーション（XPath 形式）	サポートの有無
4.1.1	Integrity Assertions	SignedParts Assertion	/sp:SignedParts/ sp:Attachments	×
4.1.2		SignedElements Assertion	/sp:SignedElements	×
4.2.1	Confidentiality Assertions	EncryptedParts Assertion	/sp:EncryptedParts	○
			/sp:EncryptedParts/sp:Body	○
			/sp:EncryptedParts/ sp:Header	×
			/sp:EncryptedParts/ sp:Attachments	×
4.2.2		EncryptedElements Assertion	/sp:EncryptedElements	×
4.2.3		ContentEncryptedElements Assertion	/ sp:ContentEncryptedElements	×
4.3.1	Required Elements Assertion	RequiredElements Assertion	/sp:RequiredElements	×
4.3.2		RequiredParts Assertion	/sp:RequiredParts	×

（凡例）

○：サポートあり

×：サポートなし

注※

WS-SecurityPolicy 1.3 仕様の該当個所（章節項）を示します。

(2)　Token Assertions のサポート範囲

WS-SecurityPolicy 1.3 仕様の中の Token Assertions のうち，Web サービスセキュリティ機能がサポートする範囲を次の表に示します。

表 A-7　Token Assertions のサポート範囲

該当個所※1	大分類	小分類	アサーション（XPath 形式）	サポートの有無
5.1	Token Inclusion	−	@sp:IncludeToken	○
5.2.1	Token Issuer and Required Claims	Token Issuer	sp:Issuer	×
5.2.2		Token Issuer Name	sp:IssuerName	×
5.2.3		Required Claims	wst:Claims	×
5.3.1	Token Properties	[Derived Keys] Property	sp:RequireDerivedKeys	×
5.3.2		[Explicit Derived Keys] Property	sp:RequireExplicitDerivedKeys	×

該当個所 ※1	大分類	小分類	アサーション (XPath 形式)	サポートの有無
5.3.3	Token Properties	[Implied Derived Keys] Property	sp:RequireImpliedDerivedKeys	×
5.4.1	Token Assertion Types	UsernameToken Assertion	/sp:UsernameToken	○
			/sp:UsernameToken/ @sp:IncludeToken	○※2
			/sp:UsernameToken/sp:Issuer	×
			/sp:UsernameToken/ sp:IssuerName	×
			/sp:UsernameToken/wst:Claims	×
			/sp:UsernameToken/wsp:Policy/ sp:NoPassword	×
			/sp:UsernameToken/wsp:Policy/ sp:HashPassword	○
			/sp13:UsernameToken/ wsp:Policy/sp13:Created	×
			/sp13:UsernameToken/ wsp:Policy/sp13:Nonce	×
			/sp:UsernameToken/wsp:Policy/ sp:RequireDerivedKeys	×
			/sp:UsernameToken/wsp:Policy/ sp:RequireExplicitDerivedKeys	×
			/sp:UsernameToken/wsp:Policy/ sp:RequireImpliedDerivedKeys	×
			/sp:UsernameToken/wsp:Policy/ sp:WssUsernameToken10	○
			/sp:UsernameToken/wsp:Policy/ sp:WssUsernameToken11	×
5.4.2	Token Assertion Types	IssuedToken Assertion	/sp:IssuedToken	×
5.4.3	Token Assertion Types	X509Token Assertion	/sp:X509Token	○
			/sp:X509Token/ @sp:IncludeToken	○※3
			/sp:X509Token/sp:Issuer	×
			/sp:X509Token/sp:IssuerName	×
			/sp:X509Token/wst:Claims	×
			/sp:X509Token/wsp:Policy/ sp:RequireDerivedKeys	×

該当個所 ※1	大分類	小分類	アサーション (XPath 形式)	サポートの有無
5.4.3	Token Assertion Types	X509Token Assertion	/sp:X509Token/wsp:Policy/sp:RequireExplicitDerivedKeys	×
			/sp:X509Token/wsp:Policy/sp:RequireImpliedDerivedKeys	×
			/sp:X509Token/wsp:Policy/sp:RequireKeyIdentifierReference	×
			/sp:X509Token/wsp:Policy/sp:RequireIssuerSerialReference	×
			/sp:X509Token/wsp:Policy/sp:RequireEmbeddedTokenReference	×
			/sp:X509Token/wsp:Policy/sp:RequireThumbprintReference	×
			/sp:X509Token/wsp:Policy/sp:WssX509V3Token10	○
			/sp:X509Token/wsp:Policy/sp:WssX509Pkcs7Token10	×
			/sp:X509Token/wsp:Policy/sp:WssX509PkiPathV1Token10	×
			/sp:X509Token/wsp:Policy/sp:WssX509V1Token11	×
			/sp:X509Token/wsp:Policy/sp:WssX509V3Token11	×
			/sp:X509Token/wsp:Policy/sp:WssX509Pkcs7Token11	×
			/sp:X509Token/wsp:Policy/sp:WssX509PkiPathV1Token11	×
5.4.4	Token Assertion Types	KerberosToken Assertion	/sp:KerberosToken	×
5.4.5	Token Assertion Types	SpnegoContextToken Assertion	/sp:SpnegoContextToken	×
5.4.6	Token Assertion Types	SecurityContextToken Assertion	/sp:SecurityContextToken	○
			/sp:SecurityContextToken/@sp:IncludeToken	○※4
			/sp:SecurityContextToken/sp:Issuer	×
			/sp:SecurityContextToken/sp:IssuerName	○

該当個所 ※1	大分類	小分類	アサーション (XPath 形式)	サポートの有無
5.4.6	Token Assertion Types	SecurityContextToken Assertion	/sp:SecurityContextToken/ wst:Claims	×
			/sp:SecurityContextToken/ wsp:Policy/ sp:RequireDerivedKeys	×
			/sp:SecurityContextToken/ wsp:Policy/ sp:RequireExplicitDerivedKeys	×
			/sp:SecurityContextToken/ wsp:Policy/ sp:RequireImpliedDerivedKeys	×
			/sp:SecurityContextToken/ wsp:Policy/ sp:RequireExternalUriReference	×
			/sp:SecurityContextToken/ wsp:Policy/ sp:SC13SecurityContextToken	×
5.4.7	Token Assertion Types	SecureConversationToken Assertion	/sp:SecureConversationToken	×
5.4.8	Token Assertion Types	SamlToken Assertion	/sp:SamlToken	×
5.4.9	Token Assertion Types	RelToken Assertion	/sp:RelToken	×
5.4.10	Token Assertion Types	HttpsToken Assertion	/sp:HttpsToken	×
5.4.11	Token Assertion Types	KeyValueToken Assertion	/sp:KeyValueToken	×

（凡例）

○：サポートあり

×：サポートなし

－：該当する分類または内容なし

注※1

WS-SecurityPolicy 1.3 仕様の該当個所（章節項）を示します。

注※2

http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/IncludeToken/AlwaysToRecipient だけをサポートしています。

注※3

sp:InitiatorToken をネストした sp:X509Token の場合は，http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/IncludeToken/AlwaysToRecipient だけをサポートしています。

sp:RecipientToken をネストした sp:X509Token の場合は，http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/IncludeToken/AlwaysToInitiator だけをサポートしています。

注※4

http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/IncludeToken/ Never だけをサポートしています。

(3)　Security Binding Assertions のサポート範囲

WS-SecurityPolicy 1.3 仕様の中の Security Binding Assertions のうち，Web サービスセキュリティ機能がサポートする範囲を次の表に示します。

表 A-8　Security Binding Assertions のサポート範囲

該当個所※	分類	アサーション (XPath 形式)	サポートの有無
7.1	AlgorithmSuite Assertion	/sp:AlgorithmSuite	○
		/sp:AlgorithmSuite/wsp:Policy/sp:Basic128	○
		/sp:AlgorithmSuite/wsp:Policy/sp:Basic128 以外	×
7.2	Layout Assertion	/sp:Layout	○
		/sp:Layout/wsp:Policy/sp:Strict	×
		/sp:Layout/wsp:Policy/sp:Lax	○
		/sp:Layout/wsp:Policy/sp:LaxTsFirst	×
		/sp:Layout/wsp:Policy/sp:LaxTsLast	×
7.3	TransportBinding Assertion	/sp:TransportBinding	×
7.4	SymmetricBinding Assertion	/sp:SymmetricBinding	○
		/sp:SymmetricBinding/wsp:Policy/sp:EncryptionToken	×
		/sp:SymmetricBinding/wsp:Policy/sp:SignatureToken	×
		/sp:SymmetricBinding/wsp:Policy/sp:ProtectionToken	○
		/sp:SymmetricBinding/wsp:Policy/sp:AlgorithmSuite	○
		/sp:SymmetricBinding/wsp:Policy/sp:Layout	○
		/sp:SymmetricBinding/wsp:Policy/sp:IncludeTimestamp	×
		/sp:SymmetricBinding/wsp:Policy/sp:EncryptBeforeSigning	×
		/sp:SymmetricBinding/wsp:Policy/sp:EncryptSignature	×
		/sp:SymmetricBinding/wsp:Policy/sp:ProtectTokens	×
		/sp:SymmetricBinding/wsp:Policy/sp:OnlySignEntireHeadersAndBody	×
7.5	AsymmetricBinding Assertion	/sp:AsymmetricBinding	○
		/sp:AsymmetricBinding/wsp:Policy/sp:InitiatorToken	○

該当個所※	分類	アサーション (XPath 形式)	サポートの有無
7.5	AsymmetricBinding Assertion	/sp:AsymmetricBinding/wsp:Policy/sp:InitiatorSignatureToken	×
		/sp:AsymmetricBinding/wsp:Policy/sp:InitiatorEncryptionToken	×
		/sp:AsymmetricBinding/wsp:Policy/sp:RecipientToken	○
		/sp:AsymmetricBinding/wsp:Policy/sp:RecipientSignatureToken	×
		/sp:AsymmetricBinding/wsp:Policy/sp:RecipientEncryptionToken	×
		/sp:AsymmetricBinding/wsp:Policy/sp:AlgorithmSuite	○
		/sp:AsymmetricBinding/wsp:Policy/sp:Layout	○
		/sp:AsymmetricBinding/wsp:Policy/sp:IncludeTimestamp	×
		/sp:AsymmetricBinding/wsp:Policy/sp:EncryptBeforeSigning	×
		/sp:AsymmetricBinding/wsp:Policy/sp:EncryptSignature	×
		/sp:AsymmetricBinding/wsp:Policy/sp:ProtectTokens	×
		/sp:AsymmetricBinding/wsp:Policy/sp:OnlySignEntireHeadersAndBody	○

（凡例）

○：サポートあり

×：サポートなし

注※

WS-SecurityPolicy 1.3 仕様の該当個所（章節）を示します。

(4)　Supporting Tokens のサポート範囲

WS-SecurityPolicy 1.3 仕様の中の Supporting Tokens のうち，Web サービスセキュリティ機能がサポートする範囲を次の表に示します。

表 A-9　Supporting Tokens のサポート範囲

該当個所※	分類	アサーション (XPath 形式)	サポートの有無
8.1	SupportingTokens Assertion	/sp:SupportingTokens	○
		/sp:SupportingTokens/wsp:Policy/sp:AlgorithmSuite	×
		/sp:SupportingTokens/wsp:Policy/sp:SignedParts	×

該当個所※	分類	アサーション (XPath 形式)	サポートの有無
8.1	SupportingTokens Assertion	/sp:SupportingTokens/wsp:Policy/sp:SignedElements	×
		/sp:SupportingTokens/wsp:Policy/sp:EncryptedParts	×
		/sp:SupportingTokens/wsp:Policy/sp:EncryptedElements	×
8.2	SignedSupportingTokens Assertion	/sp:SignedSupportingTokens	×
8.3	EndorsingSupportingTokens Assertion	/sp:EndorsingSupportingTokens	×
8.4	SignedEndorsingSupportingTokens Assertion	/sp:SignedEndorsingSupportingTokens	×
8.5	SignedEncryptedSupportingTokens Assertion	sp:SignedEncryptedSupportingTokens	×
8.6	EncryptedSupportingTokens Assertion	sp:EncryptedSupportingTokens	×
8.7	EndorsingEncryptedSupportingTokens Assertion	sp:EndorsingEncryptedSupportingTokens	×
8.8	SignedEndorsingEncryptedSupportingTokens Assertion	sp:SignedEndorsingEncryptedSupportingTokens	×

（凡例）

○：サポートあり

×：サポートなし

注※

WS-SecurityPolicy 1.3 仕様の該当個所（章節）を示します。

(5)　WSS:SOAP Message Security Options のサポート範囲

WS-SecurityPolicy 1.3 仕様の中の WSS:SOAP Message Security Options は，Web サービスセキュリティ機能ではサポートしていません。

表 A-10　WSS:SOAP Message Security Options のサポート範囲

該当個所※	分類	アサーション (XPath 形式)	サポートの有無
9.1	Wss10 Assertion	/sp:Wss10	×
		/sp:Wss10/wsp:Policy/sp:MustSupportRefKeyIdentifier	×
		/sp:Wss10/wsp:Policy/sp:MustSupportRefIssuerSerial	×
		/sp:Wss10/wsp:Policy/sp:MustSupportRefExternalURI	×
		/sp:Wss10/wsp:Policy/sp:MustSupportRefEmbeddedToken	×
9.2	Wss11 Assertion	/sp:Wss11	×

該当個所※	分類	アサーション (XPath 形式)	サポートの有無
9.2	Wss11 Assertion	/sp:Wss11/wsp:Policy/sp:MustSupportRefKeyIdentifier	×
		/sp:Wss11/wsp:Policy/sp:MustSupportRefIssuerSerial	×
		/sp:Wss11/wsp:Policy/sp:MustSupportRefExternalURI	×
		/sp:Wss11/wsp:Policy/sp:MustSupportRefEmbeddedToken	×
		/sp:Wss11/wsp:Policy/sp:MustSupportRefThumbprint	×
		/sp:Wss11/wsp:Policy/sp:MustSupportRefEncryptedKey	×
		/sp:Wss11/wsp:Policy/sp:RequireSignatureConfirmation	×

（凡例）

×：サポートなし

注※

WS-SecurityPolicy 1.3 仕様の該当個所（章節）を示します。

(6)　WS-Trust Options のサポート範囲

WS-SecurityPolicy 1.3 仕様の中の WS-Trust Options は，Web サービスセキュリティ機能ではサポートしていません。

表 A-11　WS-Trust Options のサポート範囲

該当個所※	分類	アサーション（XPath 形式）	サポートの有無
10.1	Trust13 Assertion	/sp:Trust13	×

（凡例）

×：サポートなし

注※

WS-SecurityPolicy 1.3 仕様の該当個所（章節）を示します。

付録 B　下位バージョンからの移行手順

07-60 より前のバージョンからバージョン 07-60 以降に移行する手順について説明します。07-60 より前のバージョンで作成したユーザープログラムとバージョン 07-60 以降で作成したユーザープログラムは，相互に接続できません。接続した場合は，例外が発生します。なお，07-60 より前のバージョンで作成したユーザープログラムは，再コンパイルしないで，バージョン 07-60 以降で実行できます。

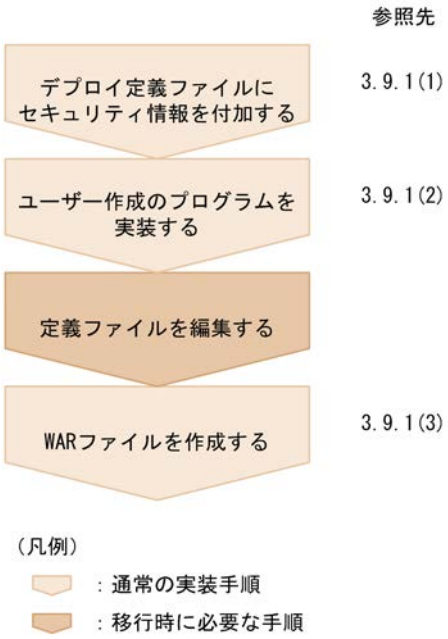
バージョン 07-60 からバージョン 08-00 以降にバージョンアップした場合は，移行による作業はありません。

(1)　サーバ側の移行手順

07-60 より前のバージョンからバージョン 07-60 以降に移行するには，通常の実装手順に加えて，定義ファイルを編集する必要があります。

Web サービスセキュリティ機能をサーバ側で移行する手順を次の図に示します。

図 B-1　サーバ側の移行手順



ここでは，定義ファイルの編集方法について説明します。定義ファイルの編集以外の手順については，「3.9.1　サーバ側の実装手順」を参照してください。

(a)　定義ファイルを編集する

07-60 より前のバージョンからバージョン 07-60 以降に移行するには，次の 2 種類の定義ファイルを編集します。

- 機能定義ファイル（security-config.xml）
- 方針定義ファイル（policy-config.xml）

定義ファイルの編集手順を次に示します。

1. 各定義ファイルのデフォルト名前空間を次のように変更します。

- 機能定義ファイル

http://www.hitachi.co.jp/soft/xml/cosminexus/ws/security/0760/securityconfig

- 方針定義ファイル

http://www.hitachi.co.jp/soft/xml/cosminexus/ws/security/0760/policyconfig

2. 機能定義ファイルおよび方針定義ファイルのプレフィクス wsse，wsu に対応する名前空間を次のように変更します。

- wsse

http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd

- wsu

http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd

3. 各定義ファイルについて，次に示す XML 要素を変更します。

機能定義ファイル

- BinarySecurityTokenConfig 要素
  BinarySecurityTokenConfig 要素の詳細については，「表 C-12　BinarySecurityTokenConfig」を参照してください。
- KeyIdentifier 要素
  KeyIdentifier 要素の詳細については，「表 C-24　KeyIdentifier」を参照してください。
- Password 要素
  Password 要素の詳細については，「表 C-35　Password」を参照してください。

方針定義ファイル

- TokenValidation 要素
  TokenValidation 要素の詳細については，「表 C-62　TokenValidation」を参照してください。
- X509TokenValidation 要素
  X509TokenValidation の詳細については，「表 C-63　X509TokenValidation」を参照してください。

(2)　クライアント側が Web アプリケーションの場合の移行手順

07-60 より前のバージョンからバージョン 07-60 以降に移行するには，通常の実装手順に加えて，定義ファイルを編集する必要があります。

クライアント側が Web アプリケーションの場合に，Web サービスセキュリティ機能を移行する手順を次の図に示します。

図 B-2　クライアント側が Web アプリケーションの場合の移行手順

定義ファイルの編集方法については，「付録 B(1)(a)　定義ファイルを編集する」を参照してください。定義ファイルの編集以外の手順については，「3.9.2　クライアント側が Web アプリケーションの場合の実装手順」を参照してください。

(3)　クライアント側がコマンドライン Java アプリケーションの場合の移行手順

07-60 より前のバージョンからバージョン 07-60 以降に移行するには，通常の実装手順に加えて，定義ファイルを編集する必要があります。

クライアント側がコマンドライン Java アプリケーションの場合に，Web サービスセキュリティ機能を移行する手順を次の図に示します。

図 B-3　クライアント側がコマンドライン Java アプリケーションの場合の移行手順

定義ファイルの編集方法については，「付録 B(1)(a)　定義ファイルを編集する」を参照してください。定義ファイルの編集以外の手順については，「3.9.3　クライアント側がコマンドライン Java アプリケーションの場合の実装手順」を参照してください。

付録 C　定義ファイルの項目の詳細

Web サービスセキュリティ機能を使用するには，次の二つの定義ファイルを設定します。

- Web サービスセキュリティ機能定義ファイル
- Web サービスセキュリティ方針定義ファイル

Web サービスセキュリティの各機能で必要な定義ファイルの項目については，「3.1　定義ファイルの設定」を参照してください。

ここでは，各定義ファイルで設定する要素の指定回数や，注意事項などの詳細を説明します。なお，各定義ファイルの要素の構成は次の図のようになっています。

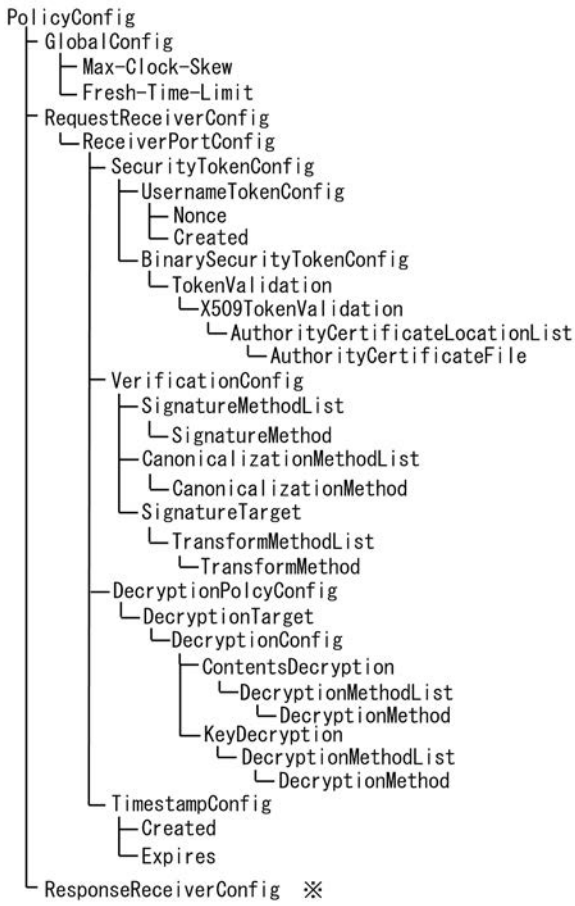
図 C-1　Web サービスセキュリティ機能定義ファイルの要素の構成

```
SecurityConfig
  ├ BindingConfig
  │  └KeyLocator
  │     ├ KeyStore
  │     │  ├ Certificate
  │     │  └ PrivateKey
  │     └ SecretKeyLocationList
  │        └ SecretKeyFile
  ├ RequestSenderConfig
  │  └ SenderPortConfig
  │     └RoleConfig
  │        ├BinarySecurityTokenConfig
  │        ├SignatureConfig
  │        │  ├CanonicalizationMethod
  │        │  │  └ CanonicalizeParam
  │        │  ├SignatureMethod
  │        │  ├SignatureTarget
  │        │  │  └ Transform
  │        │  │     └ TransformPram
  │        │  ├SignatureKey
  │        │  └SignatureKeyInfo
  │        │     └ KeyReferenceConfig
  │        │        ├ DirectReference
  │        │        └ KeyIdentifier
  │        ├EncryptionConfig
  │        │  ├ContentsEncryption
  │        │  │  └ EncryptionTarget
  │        │  │     └ EncryptionMethod
  │        │  └KeyEncryption
  │        │     ├KeyEncryptionKey
  │        │     └EncryptionMethod
  │        ├TimestampConfig
  │        │  └ Expires
  │        └UsernameTokenConfig
  │           ├ Username
  │           └ Password
  ├ ResponseSenderConfig    ※1
  ├ RequestReceiverConfig
  │  └ ReceiverPortConfig
  │     ├AuthenticationConfig
  │     │  └UsernameTokenAuthnConfig
  │     │     ├AuthnMethod
  │     │     └LoginContext
  │     │        └ConfigurationIndex
  │     ├VerificationConfig
  │     │  └VerificationKeyLocationList
  │     │     └VerificationKeyStore
  │     └DecryptionConfig
  │        └DecryptionKeyLocationList
  │           └DecryptionSercretKeyLocationList
  └ ResponseReceiverConfig  ※2
```

注※1
ResponseSenderConfig要素以下の構成は，RequestSenderConfig要素以下の構成と同じです。

注※2
ResponseReceiverConfig要素以下の構成は，RequestReceiverConfig要素以下の構成と同じです。

図 C-2　Web サービスセキュリティ方針定義ファイルの要素の構成



注※
ResponseReceiverConfig要素以下の構成は，RequestReceiverConfig要素以下の構成と同じです。

付録 C.1　Web サービスセキュリティ機能定義ファイルの項目

Web サービスセキュリティ機能定義ファイルは，XML ファイルです。ここでは，Web サービスセキュリティ機能定義ファイルの要素，および要素で指定できる属性とコンテンツ（子要素）について説明します。なお，表中のデータ型は，XML Schema のデータ型を表しています。

(1)　SecurityConfig

Web サービスセキュリティ定義ファイルのルート要素です。コンテンツ（子要素）は，次の表の順番で指定する必要があります。

表 C-1　SecurityConfig

種別	要素	説明	データ型	指定回数
属性	−	−	−	−
コンテ	BindingConfig	Web サービスセキュリティの署名，暗号化機能で使用するトークンの情報を指定	−	0 または 1

種別	要素	説明	データ型	指定回数
ンツ	BindingConfig	します。このコンテンツは署名，暗号化機能使用時は必須です。	-	0または1
	RequestSenderConfig	リクエストメッセージ送信時の設定を指定します。省略時はリクエストメッセージ送信時に Web サービスセキュリティ機能が実行されません。	-	0または1
	ResponseSenderConfig	レスポンスメッセージ送信時の設定を指定します。省略時はレスポンスメッセージ送信時に Web サービスセキュリティ機能が実行されません。	-	0または1
	RequestReceiverConfig	リクエストメッセージ受信時の設定を指定します。省略時はリクエストメッセージ受信時に Web サービスセキュリティ機能が実行されません。	-	0または1
	ResponseReceiverConfig	レスポンスメッセージ受信時の設定を指定します。省略時はレスポンスメッセージ受信時に Web サービスセキュリティ機能が実行されません。	-	0または1

(2)　BindingConfig

Web サービスセキュリティの各機能で共通的に使用する情報を指定します。

表 C-2　BindingConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	KeyLocator	鍵の格納場所の情報を指定します。このコンテンツは署名および暗号化機能を使用する場合は必須です。	-	0または1

(3)　KeyLocator

鍵の格納場所の情報を指定します。コンテンツ（子要素）は，次の表の順番で指定する必要があります。

表 C-3　KeyLocator

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コン	KeyStore	Java のキーストア形式の鍵の格納場所情報を指定します。このコンテンツは署名	-	0以上

種別	要素	説明	データ型	指定回数
テンツ	KeyStore	および暗号化機能を使用する場合は必須です。	-	0以上
	SecretKeyLocationList	共通鍵の格納場所の情報を指定します。このコンテンツは暗号化機能を使用する場合は必須です。	-	0 または 1

(4)　KeyStore

Java のキーストア形式の鍵の格納場所情報を指定します。

表 C-4　KeyStore

種別	要素	説明	データ型	指定回数
属性	Id	セキュリティ定義ファイル中で一意に識別するための ID 値を指定します。	ID	1
	File	キーストアファイルの名称を"ラベル名+拡張子"の形式で指定します。	String	1
	Type	キーストアのタイプを示す文字列を指定します。	String	1
	Password	キーストアのパスワードを指定します。	String	1
コンテンツ	Certificate	キーストア内の X.509 証明書の情報を指定します。このコンテンツは，キーストア内の証明書を利用する場合は必須です。	-	0以上
	PrivateKey	キーストア内の非公開鍵の情報を指定します。このコンテンツは，キーストア内の非公開鍵を利用する場合は必須です。	-	0以上

(5)　Certificate

キーストア内の証明書の情報を指定します。コンテンツはありません。

表 C-5　Certificate

種別	要素	説明	データ型	指定回数
属性	Id	セキュリティ機能定義ファイル中で一意に識別するための ID 値を指定します。	ID	1
	Alias	キーストアファイル中の X.509 証明書のエイリアス名を指定します。	String	1

(6)　PrivateKey

キーストア内の非公開鍵の情報を指定します。コンテンツはありません。

表 C-6　PrivateKey

種別	要素	説明	データ型	指定回数
属性	Id	セキュリティ機能定義ファイル中で一意に識別するための ID 値を指定します。	ID	1
	Alias	キーストアファイル中の秘密鍵のエイリアス名を指定します。	String	1
	Password	キーストアファイル中の秘密鍵のパスワードを指定します。	String	1

(7)　SecretKeyLocationList

共通鍵の格納場所の情報を指定します。

表 C-7　SecretKeyLocationList

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	SecretKeyFile	共通鍵ファイルの情報を指定します。このコンテンツは暗号化機能使用時は必須です。	-	0以上

(8)　SecretKeyFile

共通鍵ファイルの情報を指定します。コンテンツはありません。

表 C-8　SecretKeyFile

種別	要素	説明	データ型	指定回数
属性	Id	セキュリティ定義ファイル中で一意に識別するための ID 値を指定します。	ID	1
	Name	共通鍵作成コマンドで作成した共通鍵ファイル名称を"ラベル名+拡張子"の形式で指定します。	String	1
	KeyType	共通鍵ファイル作成時に指定したアルゴリズム識別子を指定します。	String	1
	KeyName	送信する鍵の識別子を指定します。	String	1

(9)　RequestSenderConfig

リクエストメッセージ送信時の設定を指定します。

表 C-9　RequestSenderConfig

種別	要素	説明	データ型	指定回数
属性	−	−	−	−
コンテンツ	SenderPortConfig	送信時に，Web サービスセキュリティ機能を適用する SOAP サービスエンドポイントの情報を指定します。	−	1以上

(10)　SenderPortConfig

送信時に，Web サービスセキュリティ機能を適用する，SOAP サービスまたは Web サービスのエンドポイントの情報を指定します。ResponseSenderConfig の下に SenderPortConfig を指定する場合は，Name 属性に"*"を指定してください。

表 C-10　SenderPortConfig

種別	要素	説明	データ型	指定回数
属性	Name	Web サービスセキュリティ機能を適用する SOAP サービスまたは Web サービスの URL を指定します。ここで指定したエンドポイントへメッセージを送信する際に，このコンテンツ以下で指定する Web サービスセキュリティ機能の設定が適用されます。"*"を指定すると，エンドポイントの指定に関係なく Web サービスセキュリティ機能の設定が適用されます。	anyURI	1
コンテンツ	RoleConfig	メッセージ送信時に適用するユーザー名，パスワード，署名，暗号化に関する情報を指定します。省略時はメッセージ送信時に Web サービスセキュリティ機能が実行されません。	−	0以上

(11)　RoleConfig

メッセージ送信時に適用するユーザー名，パスワード，署名，暗号化に関する情報を指定します。

表 C-11　RoleConfig

種別	要素	説明	データ型	指定回数
属性	mustUnderstand	送信する SOAP メッセージヘッダの mustUnderstand 属性を"true", "false", "1", "0"のどれかで指定します。この属性省略時は，"true"が仮定されます。"true", "1"を指定した場合は，SOAP メッセージヘッダに mustUnderstand 属性が付加さ	boolean	0 または 1

種別	要素	説明	データ型	指定回数
属性	mustUnderstand	れます。"false"，"0"を指定した場合は，SOAP メッセージヘッダに mustUnderstand 属性は付加されません。	boolean	0 または 1
	role	SOAP メッセージヘッダの role 属性（SOAP1.1 での actor 属性）を指定します。	anyURI	1
	Operation	SOAP サービスまたは Web サービスのサービスメソッド名を指定します。メソッドが複数ある場合は，半角スペースで区切って指定します。この要素を省略した場合は，SOAP サービスまたは Web サービスのすべてのメソッドに対して，このコンテンツ以下で指定する Web サービスセキュリティ機能の設定が適用されます。 この指定は，RequestSenderConfig 以下の RoleConfig に指定した場合だけ有効となります。また，呼び出す SOAP サービスがメッセージング形態の場合は，この指定は無視されます。	NMTOKENS	0 または 1
コンテンツ	BinarySecurityTokenConfig	SOAP メッセージに付与するバイナリセキュリティトークンの情報を指定します。省略時は，SOAP メッセージにバイナリセキュリティトークンは付与されません。	-	0以上
	SignatureConfig	署名に関する情報を指定します。省略時は，SOAP メッセージに署名は付与されません。	-	0以上
	EncryptionConfig	暗号化に関する情報を指定します。省略時は，SOAP メッセージが暗号化されません。	-	0以上
	TimestampConfig	Timestamp 要素に関する情報を指定します。省略時は，SOAP メッセージに Timestamp 要素は付与されません。	-	0 または 1
	UsernameTokenConfig	UsernameToken 要素に関する情報を指定します。省略時は，SOAP メッセージに UsernameToken 要素は付与されません。	-	0以上

(12)　BinarySecurityTokenConfig

SOAP メッセージに付与するバイナリセキュリティトークンの情報を指定します。コンテンツはありません。

表 C-12　BinarySecurityTokenConfig

種別	要素	説明	データ型	指定回数
属性	Id	セキュリティ機能定義ファイル中で一意に識別するための ID 値を指定します。	ID	1
	IdRef	バイナリセキュリティトークンとして使用するリソースの位置（Id）を指定します。Certificate 要素の Id 値を指定します。	IDREF	1
	EmbedId	バイナリセキュリティトークン要素をセキュリティヘッダに埋め込む時に付加する Id 値を指定します。	String	1
	EncodingType	セキュリティトークンを送信する際のエンコード種別を指定します。指定できる値は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary"だけです。省略時は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary"が仮定されます。	anyURI	0 または 1
	ValueType	バイナリセキュリティトークンの種別を指定します。指定できる値は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3"だけです。省略時は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3"が仮定されます。	anyURI	0 または 1

(13)　SignatureConfig

署名に関する情報を指定します。コンテンツ（子要素）は，次の表の順番で指定する必要があります。

表 C-13　SignatureConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	CanonicalizationMethod	正規化アルゴリズムを指定します。	-	1
	SignatureMethod	署名アルゴリズムを指定します。	-	1
	SignatureTarget	署名対象を指定します。	-	1以上
	SignatureKey	署名に使用する鍵の情報を指定します。	-	1

種別	要素	説明	データ型	指定回数
コンテンツ	SignatureKeyInfo	署名に使用する鍵の参照情報を指定します。	-	1

(14)　CanonicalizationMethod

正規化アルゴリズムを指定します。

表 C-14　CanonicalizationMethod

種別	要素	説明	データ型	指定回数
属性	Algorithm	正規化アルゴリズムのアルゴリズム識別子を指定します。	anyURI	1
コンテンツ	CanonicalizeParam	正規化処理で使用するパラメタを指定します。省略時は正規化処理時にパラメタは付与されません。	-	0 または 1

(15)　CanonicalizeParam

正規化処理で使用するパラメタを指定します。コンテンツはありません。

表 C-15　CanonicalizeParam

種別	要素	説明	データ型	指定回数
属性	InclusiveNamespaces	CanonicalizationMethod の Algorithm 属性が Exclusive Canonical XML の場合に，Exclusive XML Canonicalization Version 1.0 で規定されている名前空間プレフィクスを指定します。指定方法は，対象となる名前空間プレフィクス名をスペースで区切って指定します（例 "ns1 ns2 ns3"）。CanonicalizationMethod の Algorithm 属性が Exclusive Canonical XML（Exclusive Canonical XML with Comments または Exclusive Canonical XML omits comments）でない場合はこの指定は無視されます。	NMTOKENS	1

(16)　SignatureMethod

署名アルゴリズムを指定します。コンテンツはありません。

表 C-16　SignatureMethod

種別	要素	説明	データ型	指定回数
属性	Algorithm	署名アルゴリズムのアルゴリズム識別子を指定します。	anyURI	1

(17)　SignatureTarget

署名対象を指定します。属性の指定は，Part または TargetId のどちらかが必須になります。

表 C-17　SignatureTarget

種別	要素	説明	データ型	指定回数
属性	Part	署名対象となる SOAP エンベロープ中のエレメントを指定します。指定できる値は"Body"だけです。	enum	0 または 1
	TargetId	署名対象の Id 値（SOAP メッセージ内の要素にあらかじめ設定済みの wsu:Id の値）を指定します。この指定はメッセージング形態の場合だけ有効です。	String	0 または 1
コンテンツ	Transform	トランスフォームアルゴリズムを指定します（必須）。	-	1以上

(18)　Transform

トランスフォームアルゴリズムを指定します。

表 C-18　Transform

種別	要素	説明	データ型	指定回数
属性	Algorithm	トランスフォームアルゴリズムのアルゴリズム識別子を指定します。	anyURI	1
コンテンツ	TransformParam	トランスフォームアルゴリズムで使用するパラメタを指定します。省略時はトランスフォーム処理時にパラメタは付与されません。	-	0 または 1

(19)　TransformParam

トランスフォームアルゴリズムで使用するパラメタを指定します。コンテンツはありません。

表 C-19　TransformParam

種別	要素	説明	データ型	指定回数
属性	InclusiveNamespaces	Transform の Algorithm 属性が Exclusive Canonical XML の場合に，Exclusive XML Canonicalization Version 1.0 で規定されている名前空間プレフィクスを指定します。指定方法は，対象となる名前空間プレフィクス名をスペースで区切って指定します（例 "ns1 ns2 ns3"）。 Transform の Algorithm 属性が Exclusive Canonical XML（Exclusive Canonical XML with Comments または Exclusive Canonical XML omits comments）でない場合はこの指定は無視されます。	NMTOKENS	1

(20)　SignatureKey

署名に使用する鍵の情報を指定します。コンテンツはありません。

表 C-20　SignatureKey

種別	要素	説明	データ型	指定回数
属性	IdRef	鍵の位置（Binding 要素の PrivateKey 要素で指定する Id 属性）を指定します。	IDREF	1

(21)　SignatureKeyInfo

署名に使用する鍵の参照情報を指定します。

表 C-21　SignatureKeyInfo

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	KeyReferenceConfig	署名に使用する鍵のリファレンス情報（KeyInfo 要素内の SecurityTokenReference 要素として設定される情報）を指定します。	-	1

(22)　KeyReferenceConfig

署名に使用する鍵のリファレンス情報を指定します。コンテンツは，DirectReference または KeyIdentifier のどちらかを指定します。

表 C-22　KeyReferenceConfig

種別	要素	説明	データ型	指定回数
属性	−	−	-	-
コンテンツ	DirectReference	鍵の参照メカニズムを DirectReference にする場合の情報を指定します。	-	0 または 1
	KeyIdentifier	鍵の参照メカニズムを KeyIdentifier にする場合の情報を指定します。	-	0 または 1

(23)　DirectReference

鍵の参照メカニズムを DirectReference にする場合の情報を指定します。コンテンツはありません。URI 属性は，先頭に"#"を必ず付加してください。

表 C-23　DirectReference

種別	要素	説明	データ型	指定回数
属性	URI	セキュリティヘッダに埋め込む鍵（バイナリセキュリティトークン）の位置（URI）を指定します。ここで指定した値がセキュリティヘッダ内の wsse:Reference 要素の URI 属性に設定されます。ここで指定する値は，BinarySecurityTokenConfig 要素の EmbedId の値を参照する値である必要があります。	anyURI	1

(24)　KeyIdentifier

鍵の参照メカニズムを KeyIdentifier にする場合の情報を指定します。コンテンツはありません。

表 C-24　KeyIdentifier

種別	要素	説明	データ型	指定回数
属性	IdRef	鍵として使用するキーストアの非公開鍵の位置（Certificate 要素の Id 値）を指定します。	IDREF	1
	EncodingType	KeyIdentifier 要素のエンコード種別を指定します。 指定できる値は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary"だけです。 省略時は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-	anyURI	0 または 1

種別	要素	説明	データ型	指定回数
属性	EncodingType	security-1.0#Base64Binary"が仮定されます。	anyURI	0または1
	ValueType	KeyIdentifier 要素の種別を指定します。指定できる値は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509SubjectKeyIdentifier"だけです。省略時は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509SubjectKeyIdentifier"が仮定されます。	anyURI	0または1

(25)　EncryptionConfig

暗号化に関する情報を指定します。

表 C–25　EncryptionConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	ContentsEncryption	メッセージ内容の暗号化に関する情報を指定します。	-	1
	KeyEncryption	鍵の暗号化に関する情報を指定します。EncryptionType が"ContentsEncryption"の場合は不要です。	-	1

(26)　ContentsEncryption

メッセージ内容の暗号化に関する情報を指定します。

表 C–26　ContentsEncryption

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	EncryptionTarget	暗号化対象を指定します。	-	1

(27)　EncryptionTarget

暗号化対象を指定します。Part または TargetId のどちらかを指定してください。Part および TargetId の両方を指定した場合は，Part の指定だけが有効になります。また，TargetId を指定する場合は，対象となる SOAP メッセージのエレメントに同じ値の wsu:Id 属性が指定されている必要があります。

表 C-27　EncryptionTarget

種別	要素	説明	データ型	指定回数
属性	Part	暗号化対象となる SOAP エンベロープ中のエレメントを指定します。指定可能な値は"BodyContent"だけです。	enum	0 または 1
	TargetId	暗号化対象の Id 値（SOAP メッセージ内の要素にあらかじめ設定済みの wsu:Id の値）を指定します。この指定はメッセージング形態の SOAP サービスの場合だけ有効です。	String	0 または 1
	EmbedId	暗号化適用後のエレメントに付加する Id 値（wsu:Id 属性として設定される）を指定します。この属性を省略した場合は Id 値は Web サービスセキュリティ機能独自の値を自動的に付加します。	String	0 または 1
コンテンツ	EncryptionMethod	暗号化アルゴリズムを指定します。	-	1

(28)　EncryptionMethod

暗号化アルゴリズムを指定します。コンテンツはありません。

表 C-28　EncryptionMethod

種別	要素	説明	データ型	指定回数
属性	Algorithm	暗号化アルゴリズムのアルゴリズム識別子を指定します。	anyURI	1

(29)　KeyEncryption

鍵の暗号化に関する情報を指定します。

表 C-29　KeyEncryption

種別	要素	説明	データ型	指定回数
属性	－	－	-	-

種別	要素	説明	データ型	指定回数
コンテンツ	EncryptionMethod	暗号化アルゴリズムを指定します。	-	1
	KeyEncryptionKey	鍵の暗号化に使用する鍵の情報を指定します。	-	1

(30)　KeyEncryptionKey

鍵の暗号化に使用する鍵の情報を指定します。コンテンツはありません。

表 C-30　KeyEncryptionKey

種別	要素	説明	データ型	指定回数
属性	IdRef	使用する鍵の位置（SecretKeyFile 要素で指定する Id 属性）を指定します。	IDREF	1

(31)　TimestampConfig

Timestamp 要素に関する情報を指定します。

表 C-31　TimestampConfig

種別	要素	説明	データ型	指定回数
属性	EmbedId	Timestamp 要素をセキュリティヘッダに埋め込む時の Id 値を指定します。省略時は Timestamp 要素をセキュリティヘッダに埋め込む時に Id は付与されません。	String	0 または 1
	Created	Timestamp 要素に Created 要素を付加するかどうかを"true", "false", "1", "0"のどれかで指定します。 "true", "1"を指定した場合 Created 要素が付加されます。 "false", "0"を指定した場合 Created 要素は付加されません。省略時は"false"が仮定されます。	boolean	0 または 1
	Expires	Timestamp 要素に Expires 要素を付加するかどうかを"true", "false", "1", "0"のどれかで指定します。 "true", "1"を指定した場合 Expires 要素が付加されます。 "false", "0"を指定した場合 Expires 要素は付加されません。省略時は"false"が仮定されます。	boolean	0 または 1
コン	Expires	Expires 要素に関する情報を指定します。	-	0 または 1

種別	要素	説明	データ型	指定回数
コンテンツ	Expires	Expires 要素に関する情報を指定します。	-	0 または 1

(32)　Expires

Expires 要素に関する情報を指定します。コンテンツはありません。

表 C-32　Expires

種別	要素	説明	データ型	指定回数
属性	Value	有効期限を設定します。現在時刻からの相対時間をミリ秒単位で指定します。指定できる範囲は 1,000～2,147,483,647 です。範囲外の値を指定した場合，または省略時は 300,000（5 分）が仮定されます。	int	0 または 1

(33)　UsernameTokenConfig

UsernameToken 要素に関する情報を指定します。

表 C-33　UsernameTokenConfig

種別	要素	説明	データ型	指定回数
属性	Id	Web サービスセキュリティ機能定義ファイル中で，UsernameTokenConfig 要素を一意に識別するための ID 値を指定します。	ID	1
	EmbedId	UsernameToken 要素をセキュリティヘッダに埋め込む時の Id 値を指定します。省略時は UsernameToken 要素をセキュリティヘッダに埋め込む時に Id が付与されません。	String	0 または 1
コンテンツ	Username	ユーザー名に関する情報を指定します。	-	1
	Password	パスワードに関する情報を指定します。省略時は UsernameToken 要素にパスワードが付与されません。	-	0 または 1

(34)　Username

ユーザー名に関する情報を指定します。

表 C-34　Username

種別	要素	説明	データ型	指定回数
属性	−	−	-	-
コンテンツ	ユーザー名	認証に使用するユーザー ID 値を指定します。	String	1

(35)　Password

パスワードに関する情報を指定します。

表 C-35　Password

種別	要素	説明	データ型	指定回数
属性	Type	パスワードの形式を指定します。指定できる値は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"（テキスト），または"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordDigest"（ダイジェスト）の 2 種類だけです。省略時は，"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"が仮定されます。	anyURI	0 または 1
コンテンツ	パスワード値	認証に使用するパスワード値を指定します。	String	1

(36)　ResponseSenderConfig

レスポンスメッセージ送信時の設定を指定します。

表 C-36　ResponseSenderConfig

種別	要素	説明	データ型	指定回数
属性	−	−	-	-

種別	要素	説明	データ型	指定回数
コンテンツ	SenderPortConfig	送信時に Web サービスセキュリティ機能を適用する SOAP サービスエンドポイントの情報を指定します。	-	1以上

(37)　RequestReceiverConfig

リクエストメッセージ受信時の設定を指定します。

表 C-37　RequestReceiverConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	ReceiverPortConfig	Web サービスセキュリティ機能を適用する SOAP サービスまたは Web サービスのエンドポイントの情報を指定します。	-	1以上

(38)　ReceiverPortConfig

受信時に Web サービスセキュリティ機能を適用する SOAP サービスまたは Web サービスのエンドポイントの情報を指定します。

表 C-38　ReceiverPortConfig

種別	要素	説明	データ型	指定回数
属性	Name	メッセージを受信する SOAP サービスまたは Web サービスの URL を指定します。	anyURI	1
	My_role	Web サービスセキュリティ機能を適用する SOAP サービスまたは Web サービスのロール名を URI 形式で指定します。受信した SOAP メッセージのセキュリティヘッダ中の role 属性値が, ここで指定したロール名と一致した場合に, PortTypeConfig 以下に指定した動作定義に従って Web サービスセキュリティ機能を実行します。この属性を省略した場合は，受信した SOAP メッセージのセキュリティヘッダ中の role 属性の内容（role 属性がない場合も含む）にかかわらず，PortTypeConfig 以下に指定した動作	anyURI	1

種別	要素	説明	データ型	指定回数
属性	My_role	定義に従って Web サービスセキュリティ機能を実行します。	anyURI	1
コンテンツ	AuthenticationConfig	セキュリティトークンの認証に関する情報を指定します。省略時はセキュリティトークンの認証を行いません。	-	0 または 1
	VerificationConfig	署名検証に関する情報を指定します。省略時は署名検証を行いません。	-	0 または 1
	DecryptionConfig	復号化に関する情報を指定します。省略時は復号化を行いません。	-	0 または 1

(39)　AuthenticationConfig

セキュリティトークンの認証に関する情報を指定します。

表 C-39　AuthenticationConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	UsernameTokenAuthnConfig	UsernameToken の認証に関する情報を指定します。省略時は UsernameToken の認証を行いません。	-	0 または 1

(40)　UsernameTokenAuthnConfig

UsernameToken の認証に関する情報を指定します。LoginContext タグを省略した場合は，UsernameToken の認証をしないで処理を続行します。

表 C-40　UsernameTokenAuthnConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	AuthnMethod	認証方式に関する情報を指定します。	-	1
	LoginContext	JAAS 認証に関する情報を指定します。AuthnMethod に"JAASAuthn"を指定した場合，この要素は必須です。	-	0 または 1

(41)　AuthnMethod

認証方式に関する情報を指定します。コンテンツはありません。

表 C-41　AuthnMethod

種別	要素	説明	データ型	指定回数
属性	Type	認証方式を指定します。指定できる値は"JAASAuthn"だけです。	enum	1

(42)　LoginContext

JAAS 認証に関する情報を指定します。

表 C-42　LoginContext

種別	要素	説明	データ型	指定回数
属性	−	−	−	−
コンテンツ	ConfigurationIndex	ログイン構成のインデックスを指定します。	−	1

(43)　ConfigurationIndex

ログイン構成のインデックスを指定します。

表 C-43　ConfigurationIndex

種別	要素	説明	データ型	指定回数
属性	−	−	−	−
コンテンツ	インデックス値	LoginContext クラスをインスタンス化する際に使用するログイン構成のインデックス(jaas.conf 内のインデックス名)を指定します。	−	1

(44)　VerificationConfig

署名検証に関する情報を指定します。

表 C-44　VerificationConfig

種別	要素	説明	データ型	指定回数
属性	−	−	−	−
コン	VerificationKeyLocationList	署名検証に用いる鍵に関する情報を指定します。	−	1

種別	要素	説明	データ型	指定回数
コンテンツ	VerificationKeyLocationList	署名検証に用いる鍵に関する情報を指定します。	-	1

(45)　VerificationKeyLocationList

署名検証に用いる鍵に関する情報を指定します。受信したセキュリティヘッダ内の署名鍵の参照形式（SecurityTokenReference 要素で示されます）が"wsse:Reference"の場合，この要素の VerificationKeyStore タグで指定された内容は無視されます。

表 C-45　VerificationKeyLocationList

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	VerificationKeyStore	署名検証に用いる鍵の所在に関する情報を指定します。	-	1以上

(46)　VerificationKeyStore

署名検証に用いる鍵の所在に関する情報を指定します。コンテンツはありません。

表 C-46　VerificationKeyStore

種別	要素	説明	データ型	指定回数
属性	IdRef	使用する鍵を含む KeyStore 要素の Id 属性を指定します。	IDREF	1

(47)　DecryptionConfig

復号化に関する情報を指定します。

表 C-47　DecryptionConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	DecryptionKeyLocationList	復号に用いる鍵の所在に関する情報を指定します。	-	1

(48)　DecryptionKeyLocationList

復号に用いる鍵の所在に関する情報を指定します。

表 C-48　DecryptionKeyLocationList

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	DecryptionSecretKeyLocationList	復号に用いる鍵の所在に関する情報を指定します。	-	1以上

(49)　DecryptionSecretKeyLocationList

復号に用いる鍵の所在に関する情報を指定します。コンテンツはありません。

表 C-49　DecryptionSecretKeyLocationList

種別	要素	説明	データ型	指定回数
属性	IdRef	使用する鍵を含む SecretKeyFile 要素の Id 属性の値を指定します。	IDREF	1

(50)　ResponseReceiverConfig

リクエストメッセージ受信時の設定を指定します。

表 C-50　ResponseReceiverConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	ReceiverPortConfig	Web サービスセキュリティ機能を適用する SOAP サービスまたは Web サービスのエンドポイントの情報を指定します。	-	1

付録 C.2　Web サービスセキュリティ方針定義ファイルの項目

Web サービスセキュリティ方針定義ファイルは，XML ファイルです。ここでは，Web サービスセキュリティ方針定義ファイルの要素，および要素で指定できる属性とコンテンツ（子要素）について説明します。なお，表中のデータ型は，XML Schema のデータ型を表しています。

(1)　PolicyConfig

方針定義ファイルのルート要素です。

表 C-51　PolicyConfig

種別	要素	説明	データ型	指定回数
属性	−	−	-	-
コンテンツ	GlobalConfig	Web サービスセキュリティ方針定義で共通的に使用する規則を指定します。省略時は，GlobalConfig 要素内の値はすべてデフォルト値が仮定されます。	-	0 または 1
	RequestReceiverConfig	リクエストメッセージ受信時の設定を指定します。 省略した場合，リクエストメッセージ受信時の方針に従っているかどうかのチェックは実施されません。	-	0 または 1
	ResponseReceiverConfig	レスポンスメッセージ受信時の設定を指定します。 省略した場合，レスポンスメッセージ受信時の方針に従っているかどうかのチェックは実施されません。	-	0 または 1

(2)　GlobalConfig

Web サービスセキュリティ方針定義で共通的に使用する規則を指定します。

表 C-52　GlobalConfig

種別	要素	説明	データ型	指定回数
属性	−	−	-	-
コンテンツ	Max-Clock-Skew	有効期限をチェックする際の時間差に関する情報を指定します。省略時は Max-Clock-Skew 要素内の値はすべてデフォルト値が仮定されます。	-	0 または 1
	Fresh-Time-Limit	UsernameToken 要素，Timestamp 要素内の Created 要素の有効期間に関するチェック情報を指定します。省略時は Fresh-Time-Limit 要素内の値はすべてデフォルト値が仮定されます。	-	0 または 1

(3)　Max-Clock-Skew

有効期限をチェックする際の時間差に関する情報を指定します。コンテンツはありません。

表 C-53　Max-Clock-Skew

種別	要素	説明	データ型	指定回数
属性	Value	UsernameToken 要素および Timestamp 要素の子要素である Created 要素，または Timestamp 要素の子要素である Expires 要	int	1

種別	要素	説明	データ型	指定回数
属性	Value	素に指定された値に基づいて SOAP メッセージの有効期限を確認する場合に，送信側と受信側との時間の差をどこまで許容するかを指定します。指定するときの単位はミリ秒です。 指定できる範囲は，1～2,147,483,647 の間です。範囲外の値を指定した場合，または指定を省略した場合は，0 ミリ秒が仮定されます。	int	1

(4)　Fresh-Time-Limit

UsernameToken 要素，および Timestamp 要素内の Created 要素の有効期間に関する情報を指定します。コンテンツはありません。

表 C-54　Fresh-Time-Limit

種別	要素	説明	データ型	指定回数
属性	Value	UsernameToken 要素および Timestamp 要素の子要素である Created 要素に指定された値を確認する場合に，送信側と受信側との時間の差をどこまで許容するかを指定します。 指定するときの単位はミリ秒です。 指定できる範囲は，1,000～2,147,483,647 の間です。範囲外の値を指定した場合，または指定を省略した場合は，300,000 ミリ秒が仮定されます。	int	1

(5)　RequestReceiverConfig

リクエストメッセージ受信時の設定を指定します。

表 C-55　RequestReceiverConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	ReceiverPortConfig	Web サービスセキュリティ方針定義を適用する SOAP サービスまたは Web サービスのエンドポイントの情報を指定します。	-	1以上

(6)　ReceiverPortConfig

Web サービスセキュリティ方針定義を適用する SOAP サービスまたは Web サービスのエンドポイントの情報を指定します。

表 C-56　ReceiverPortConfig

種別	要素	説明	データ型	指定回数
属性	Name	Web サービスセキュリティ方針定義を適用する SOAP サービスまたは Web サービスの URL を指定します。 アスタリスク"*"を指定した場合，すべての SOAP サービスまたは Web サービスに対して Web サービスセキュリティ方針定義が適用されます。	anyURI	1
	My_role	Web サービスセキュリティ方針定義を適用する SOAP サービスまたは Web サービスのロール名を，URI で指定します。受信した SOAP メッセージのセキュリティヘッダ内で，RoleConfig 要素の role 属性に指定されている値が My_role 属性で指定したロール名と一致した場合に，ReceiverPortConfig 要素で指定した定義に従って Web サービスセキュリティ方針定義が適用されます。	anyURI	1
コンテンツ	SecurityTokenConfig	セキュリティトークンに関するチェック情報を指定します。省略時はセキュリティトークンに関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1
	VerificationConfig	署名に関するチェック情報を指定します。省略時は署名に関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1
	DecryptionPolicyConfig	復号化に関するチェック情報を指定します。省略時は復号化に関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1
	TimestampConfig	タイムスタンプに関するチェック情報を指定します。省略時はタイムスタンプに関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1

(7)　SecurityTokenConfig

セキュリティトークンに関する方針チェックの情報を指定します。

表 C-57　SecurityTokenConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテ	UsernameTokenConfig	UsernameToken 要素に関するチェック情報を指定します。省略時は	-	0 または 1

種別	要素	説明	データ型	指定回数
ンツ	UsernameTokenConfig	UsernameToken に関する方針チェックは実施されません。	-	0 または 1
	BinarySecurityTokenConfig	バイナリセキュリティトークンに関するチェック情報を指定します。省略時はバイナリセキュリティトークンに関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1

(8) UsernameTokenConfig

UsernameToken 要素に関する方針チェックの情報を指定します。

表 C-58　UsernameTokenConfig

種別	要素	説明	データ型	指定回数
属性	Required	セキュリティヘッダ内の UsernameToken 要素の有無をチェックするかどうかを"true"，"false"，"1"，"0"のどれかで指定します。 "true"または"1"を指定した場合，セキュリティヘッダ内に UsernameToken 要素が存在しないときは方針に違反するものと見なし，SOAP Fault をスローします。 "false"または"0"を指定した場合，セキュリティヘッダ内の UsernameToken 要素の有無をチェックせず，UsernameToken 要素が存在しても処理しません。	boolean	1
コンテンツ	Nonce	Nonce 要素に関するチェック情報を指定します。省略時は Nonce 要素に関する方針チェックは実施されません。	-	0 または 1
	Created	UsernameToken 要素内の Created 要素に関するチェック情報を指定します。省略時は Created 要素に関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1

(9) Nonce

Nonce 要素に関する方針チェックの情報を指定します。コンテンツはありません。

表 C-59　Nonce

種別	要素	説明	データ型	指定回数
属性	Required	UsernameToken 要素内の Nonce 要素の有無をチェックするかどうかを"true"，"false"，"1"，"0"のどれかで指定します。 "true"または"1"を指定した場合，UsernameToken 要素内に Nonce 要素が存在しないときは方針に違反するものと見なし，SOAP Fault をスローします。 "false"または"0"を指定した場合，UsernameToken 要素内の Nonce 要素の有無をチェックせず，Nonce 要素が存在しても処理しません。	boolean	1

(10)　Created

UsernameToken 要素内の Created 要素に関する方針チェックの情報を指定します。

表 C-60　Created

種別	要素	説明	データ型	指定回数
属性	Required	UsernameToken 要素内の Created 要素の有無をチェックするかどうかを"true"，"false"，"1"，"0"のどれかで指定します。 "true"または"1"を指定した場合，UsernameToken 要素内に Created 要素が存在しないときは方針に違反するものと見なし，SOAP Fault をスローします。 "false"または"0"を指定した場合，UsernameToken 要素内の Created 要素の有無をチェックせず，Created 要素が存在しても処理しません。	boolean	1

(11)　BinarySecurityTokenConfig

バイナリセキュリティトークンに関する方針チェックの情報を指定します。

表 C-61　BinarySecurityTokenConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	TokenValidation	BinarySecurityToken 要素の検証に関するチェック情報を指定します。	-	1

(12)　TokenValidation

BinarySecurityToken 要素の検証に関する方針チェックの情報を指定します。

表 C-62　TokenValidation

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	X509TokenValidation	X.509 証明書の検証に関するチェック情報を指定します。この要素が指定されている場合，受信した SOAP メッセージに X.509 証明書の BinarySecurityToken 要素（ValueType が"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3"である BinarySecurityToken 要素）	-	0 または 1

種別	要素	説明	データ型	指定回数
コンテンツ	X509TokenValidation	が存在しない場合，方針に違反するものと見なして SOAP Fault をスローします。 省略時は X.509 証明書の検証に関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1

(13)　X509TokenValidation

X.509 証明書の検証に関する方針チェックの情報を指定します。

表 C-63　X509TokenValidation

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	AuthorityCertificateLocationList	X.509 証明書の検証に関するチェック情報を指定します。	-	1

(14)　AuthorityCertificateLocationList

X.509 証明書の検証に関する方針チェックの情報を指定します。

表 C-64　AuthorityCertificateLocationList

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	AuthorityCertificateFile	X.509 証明書の署名検証に使用する証明書ファイルの情報を指定します。	-	1以上

(15)　AuthorityCertificateFile

X.509 証明書の署名検証に使用する証明書ファイルの情報を指定します。コンテンツはありません。

表 C-65　AuthorityCertificateFile

種別	要素	説明	データ型	指定回数
属性	Name	X.509 証明書の署名検証に使用する証明書ファイル名を指定します。	String	1

(16)　VerificationConfig

署名に関するチェック情報を指定します。VerificationConfig 要素の指定があり，署名がされていないメッセージを受信した場合は方針に違反するものと見なして，SOAP Fault をスローします。

表 C-66　VerificationConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	SignatureMethodList	署名アルゴリズムに関するチェック情報を指定します。	-	1
	CanonicalizationMethodList	正規化アルゴリズムに関するチェック情報を指定します。	-	1
	SignatureTarget	署名個所に関するチェック情報を指定します。	-	1

(17)　SignatureMethodList

署名アルゴリズムに関するチェック情報を指定します。受信メッセージ中の署名アルゴリズムが，この要素の SignatureMethod タグで指定した，どのアルゴリズムとも一致しない場合は，方針に違反するものと見なして，SOAP Fault をスローします。

表 C-67　SignatureMethodList

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	SignatureMethod	署名アルゴリズムに関する情報を指定します。省略時は署名アルゴリズムに関する方針に従っているかどうかのチェックは実施されません。	-	0以上

(18)　SignatureMethod

署名アルゴリズムに関する情報を指定します。コンテンツはありません。

表 C-68　SignatureMethod

種別	要素	説明	データ型	指定回数
属性	Algorithm	受信側で処理可能な署名アルゴリズムの URI を指定します。	anyURI	1

(19)　CanonicalizationMethodList

正規化アルゴリズムに関するチェック情報を指定します。受信メッセージ中の正規化アルゴリズムが，この要素の CanonicalizationMethod タグで指定した，どのアルゴリズムとも一致しない場合は，方針に違反するものと見なして，SOAP Fault をスローします。

表 C-69　CanonicalizationMethodList

種別	要素	説明	データ型	指定回数
属性	−	−	-	-
コンテンツ	CanonicalizationMethod	正規化アルゴリズムに関する情報を指定します。省略時は正規化アルゴリズムに関する方針に従っているかどうかのチェックは実施されません。	-	0以上

(20)　CanonicalizationMethod

正規化アルゴリズムに関する情報を指定します。コンテンツはありません。

表 C-70　CanonicalizationMethod

種別	要素	説明	データ型	指定回数
属性	Algorithm	受信側で処理可能な正規化アルゴリズムの URI を指定します。	anyURI	1

(21)　SignatureTarget

署名個所に関するチェック情報を指定します。

表 C-71　SignatureTarget

種別	要素	説明	データ型	指定回数
属性	Part	署名個所となる SOAP エンベロープ中のエレメントを指定します。メッセージの署名個所がこの属性に指定した個所と異なる場合，方針に違反するものと見なして，SOAP Fault をスローします。指定できる値は"Body"だけです。	enum	1

種別	要素	説明	データ型	指定回数
コンテンツ	TransformMethodList	トランスフォームアルゴリズムに関する情報を指定します。省略時はトランスフォームアルゴリズムに関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1

(22)　TransformMethodList

トランスフォームアルゴリズムに関する情報を指定します。受信メッセージ中のトランスフォームアルゴリズムが，この要素の TransformMethod タグで指定した，どのアルゴリズムとも一致しない場合は，方針に違反するものと見なして，SOAP Fault をスローします。

表 C–72　TransformMethodList

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	TransformMethod	トランスフォームアルゴリズムのアルゴリズム識別子を指定します。省略時はトランスフォームアルゴリズムに関する方針に従っているかどうかのチェックは実施されません。	anyURI	0以上

(23)　TransformMethod

トランスフォームアルゴリズムのアルゴリズム識別子を指定します。コンテンツはありません。

表 C–73　TransformMethod

種別	要素	説明	データ型	指定回数
属性	Algorithm	受信側で処理可能なトランスフォームアルゴリズムの URI を指定します。	anyURI	1

(24)　DecryptionPolicyConfig

復号化に関するチェック情報を指定します。DecryptionConfig 要素の指定があり，暗号化がされていないメッセージを受信した場合は，方針に違反するものと見なして，SOAP Fault をスローします。

表 C–74　DecryptionPolicyConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コン	DecryptionTarget	復号化個所に関するチェック情報を指定します。	-	1

種別	要素	説明	データ型	指定回数
テンツ	DecryptionTarget	復号化個所に関するチェック情報を指定します。	-	1

(25)　DecryptionTarget

復号化個所に関するチェック情報を指定します。

表 C-75　DecryptionTarget

種別	要素	説明	データ型	指定回数
属性	Part	復号化する個所となる SOAP エンベロープ中のエレメントを指定します。メッセージの復号化個所がこの属性に指定した個所と異なる場合は，方針に違反するものと見なして，SOAP Fault をスローします。指定できる値は"Body"だけです。	enum	1
	Type	復号化する個所の暗号化タイプを指定します。指定できる値は"Content"だけです。	enum	1
コンテンツ	DecryptionConfig	復号化に関するチェック情報を指定します。	-	1

(26)　DecryptionConfig

復号化に関するチェック情報を指定します。DecryptionConfig 要素の指定があり，暗号化がされていないメッセージを受信した場合は，方針に違反するものと見なして，SOAP Fault をスローします。

表 C-76　DecryptionConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	ContentsDecryption	メッセージ内容の復号化に関するチェック情報を指定します。	-	1
	KeyDecryption	復号に使用する鍵の復号化に関するチェック情報を指定します。EncryptionType が"ContentsEncryption"の場合は不要です。	-	1

(27)　ContentsDecryption

メッセージ内容の復号化に関するチェック情報を指定します。

表 C-77　ContentsDecryption

種別	要素	説明	データ型	指定回数
属性	−	−	-	-
コンテンツ	DecryptionMethodList	復号化個所の暗号アルゴリズムに関するチェック情報を指定します。省略時は復号化個所の暗号アルゴリズムに関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1

(28)　DecryptionMethodList

復号化個所の暗号アルゴリズムに関するチェック情報を指定します。受信メッセージ中の暗号アルゴリズムが，この要素の DecryptionMethod タグで指定した，どのアルゴリズムとも一致しない場合は，方針に違反するものと見なして，SOAP Fault をスローします。

表 C-78　DecryptionMethodList

種別	要素	説明	データ型	指定回数
属性	−	−	-	-
コンテンツ	DecryptionMethod	復号化に用いる暗号アルゴリズムに関する情報を指定します。省略時は復号化に用いる暗号アルゴリズムに関する方針に従っているかどうかのチェックは実施されません。	-	0以上

(29)　DecryptionMethod

復号化に用いる暗号アルゴリズムに関する情報を指定します。コンテンツはありません。

表 C-79　DecryptionMethod

種別	要素	説明	データ型	指定回数
属性	Algorithm	受信側で処理できる暗号アルゴリズムの URI を指定します。	anyURI	1

(30)　KeyDecryption

復号に使用する鍵の復号化に関するチェック情報を指定します。

表 C-80　KeyDecryption

種別	要素	説明	データ型	指定回数
属性	−	−	-	-
コンテンツ	DecryptionMethodList	復号化個所に関するチェック情報を指定します。省略時は復号化個所に関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1

(31)　TimestampConfig

タイムスタンプに関するチェック情報を指定します。

表 C-81　TimestampConfig

種別	要素	説明	データ型	指定回数
属性	−	−	-	-
コンテンツ	Created	タイムスタンプ要素の Created 要素に関するチェック情報を指定します。省略時は Created 要素に関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1
	Expires	タイムスタンプ要素の Expires 要素に関するチェック情報を指定します。省略時は Expires 要素に関する方針に従っているかどうかのチェックは実施されません。	-	0 または 1

(32)　Created

タイムスタンプ要素の Created 要素に関するチェック情報を指定します。コンテンツはありません。

表 C-82　Created

種別	要素	説明	データ型	指定回数
属性	Required	タイムスタンプ要素内の Created 要素の有無をチェックするかどうかを"true", "false", "1", "0"のどれかで指定します。 "true"または"1"を指定した場合タイムスタンプ要素内に Created 要素が存在しない場合は方針に違反するものと見なして，SOAP Fault をスローします。 "false"または"0"を指定した場合，タイムスタンプ要素内の Created 要素の有無をチェックしません。	boolean	1

(33)　Expires

タイムスタンプ要素の Expires 要素に関するチェック情報を指定します。コンテンツはありません。

表 C-83　Expires

種別	要素	説明	データ型	指定回数
属性	Required	タイムスタンプ要素内の Expires 要素の有無をチェックするかどうかを"true", "false", "1", "0"のどれかで指定します。 "true"または"1"を指定した場合, タイムスタンプ要素内に Expires 要素が存在しない場合は方針に違反するものと見なして, SOAP Fault をスローします。 "false"または"0"を指定した場合, タイムスタンプ要素内の Expires 要素の有無をチェックしません。	boolean	1

(34)　ResponseReceiverConfig

レスポンスメッセージ受信時の設定を指定します。

表 C-84　ResponseReceiverConfig

種別	要素	説明	データ型	指定回数
属性	－	－	-	-
コンテンツ	ReceiverPortConfig	Web サービスセキュリティ方針定義を適用する SOAP サービスエンドポイントの情報を指定します。	-	1以上

付録 D　用語解説

マニュアルで使用する用語について

マニュアル「アプリケーションサーバ & BPM/ESB 基盤 用語解説」を参照してください。

索引

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

あ

い

か

き

く

こ

さ

し

せ

て

と

に

は

ひ

ふ

ほ

め

ろ