

セキュリティ対策研究開発等事業

セキュアなインターネットサーバー構築に関する調査

啓発資料

「セキュア・プログラミング講座2」



平成15年2月

情報処理振興事業協会

セキュリティセンター

はじめに

サーバにおける開発において、Java 言語での開発が非常に多くなってきた。これは、Java 言語がオブジェクト指向言語であることと、Java 言語を前提とした多くの良質のサーバサイド開発プラットフォームやフレームワークなどが提供されたことによる。その中でも中核をなす Java 2 Enterprise Edition の存在が大きく寄与している。Java 2 Enterprise Edition の登場は、エンタープライズサーバサイドプログラミングにおける開発設計を劇的に変化させたといっても過言ではない。Java 2 Enterprise Edition が持つエンタープライズサーバサイドプログラミングに必要な膨大なクラス群の存在や Java 2 Standard Edition のベースとして開発されたことは、これまで提唱され続けてきたオブジェクト指向の開発を現実のものとした。

その一方で、Java や Java2 Enterprise Edition を利用しないエンタープライズサーバサイドプログラムは仕様が非常に大きくなり、不十分な設計で開発そのものの規模も非常に大きく、機能実装の開発そのものだけに注力されることが多くなってきている。このような状況下で、既存の企業情報システムを統合しつつ多種多様な言語や簡易言語などを組み合わせることで開発することになる。このような開発では使用される多くの言語のアーキテクチャの違いは相互接続や相互利用が非常に困難である。その結果、多くの不具合が発生しやすくなり、エンタープライズサーバサイドプログラムのセキュリティが手薄になり脆弱性を潜在的に持ちやすくなる。これは、使用されている言語自身に良く設計されたセキュリティモデルが存在しないためでもある。さらに、エンタープライズサーバサイドプログラミングにおいて良く設計されたセキュリティモデルが使われることは稀であった。

Java 2 Enterprise Edition には必要なセキュリティアーキテクチャが実装されている。また、良く設計されたセキュリティモデルを使って品質の良いエンタープライズサーバサイドプログラミングが開発され始めてきた。その例として Java 2 Enterprise Edition のセキュリティモデルのひとつである「宣言によるセキュリティ」の使用により、開発者は、アプリケーション毎のセキュリティの機構固有の詳細な実装をする必要がなくなる。これは、困難なセキュリティに関する実装から開発者を解放することを意味する。さらに、「宣言によるセキュリティ」を利用することで、Java 2 Enterprise Edition において、プログラムの変更なしで、環境に応じてアプリケーションのセキュリティを設定できるという非常に高い移植性を得ることができる。

Java 2 Enterprise Edition のこのようなセキュリティモデルはアクセスコントロールなどの非常に抽象度が高い概念が導入されている。これを十分に使いこなすためには、Java 2

Enterprise Edition のセキュリティについて、Bean 開発者、アプリケーションアセンブラ、アプリケーションデプロイアの立場から深く理解する必要がある。そこで、本講座では Java2 Enterprise Edition のリファレンス実装の J2EE Standard Developer Kit を実際に使用しながら Java 2 Enterprise Edition のセキュリティの理解を深めていくこととする。

本文中の製品名は、一般に各社の登録商標、商標、または商品名である。
本文では、TM、©、®マークは省略している。

1. 本書の概要	1
1.1. 調査概要	1
1.2. 調査期間	1
1.3. 調査対象	1
1.4. 調査方法	1
1.5. 想定される読者	1
1.6. 前提となる環境	2
2. J2EE セキュアプログラミング概要	3
3. J2EE セキュリティアーキテクチャによるセキュリティ実装からの解放	5
3.1. J2EE セキュリティアーキテクチャの目的	5
4. セキュリティロール	10
4.1. コンポーネントに対するアクセス権の定義	10
4.2. セキュリティロール参照と宣言のリンク	16
4.3. J2EE のユーザとグループへのロールのマッピング	25
5. Web 層のセキュリティ	31
5.1. Web 層のセキュリティアーキテクチャ	31
5.2. Web 層のリソースの保護	32
5.3. Web リソースへのアクセス制御	34
5.4. Web リソースのユーザ認証	40
5.5. HTTP 基本認証	43
5.6. フォームベース認証	44
5.7. クライアント証明書認証	46
5.8. SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張	48
5.9. J2EE SDK の deploytool による Web リソースの認証機構の設定	54
5.10. Web 層のプログラムによるセキュリティの使用	57
5.11. 保護されていない Web リソース	60
6. EJB 層のセキュリティ	63
6.1. EJB 層のセキュリティ	63
6.2. EJB のメソッドのアクセス権の宣言	65
6.3. EJB 層のプログラムによるセキュリティ	74
6.4. 保護されていない EJB 層のリソース	78
6.5. EJB 層の認証	79
7. アプリケーションクライアント層のセキュリティ	82
7.1. アプリケーションクライアント層のセキュリティ	82
7.2. アプリケーションクライアントのコールバックハンドラの指定	84
8. EIS 層のセキュリティ	88
8.1. EIS 層のセキュリティ	88
8.2. EIS 認証の設定	91
8.3. EIS コンテナ管理による認証	96
8.4. EIS コンポーネント管理による認証	97
8.5. リソースアダプタのセキュリティの設定	99
9. セキュリティアイデンティティの伝達	106
9.1. セキュリティアイデンティティの伝達	106
9.2. コンポーネントの伝達されたセキュリティアイデンティティの設定	108
9.3. クライアント認証の設定	113

9.4.	コンテナ間の信頼	117
10.	J2EE のユーザ、レルム、およびグループ	119
10.1.	J2EE のユーザ、レルム、およびグループ	119
10.2.	J2EE のユーザとグループの管理	121
11.	サーバ証明書の設定	134
11.1.	X.509 デジタル証明書	134
11.2.	J2EE サーバ証明書の設定	134
12.	監査	138
12.1.	監査	138
13.	チェックリスト	142

収録記事のご紹介

この「セキュアプログラミング講座2」には、次のような記事が収録されている。

1. 本書の概要

2. J2EE セキュアプログラミング概要

3. J2EE セキュリティアーキテクチャによるセキュリティ実装からの解放

J2EE セキュリティアーキテクチャを使用することで、セキュリティに関する実装から解放されることについて述べる

4. セキュリティロール

アプリケーションコンポーネントプロバイダまたはアプリケーションアセンブラが定義する論理的なユーザとグループ化とその使用と利点について述べる。

5. Web 層のセキュリティ

Web 層のリソースを保護するために使用する、宣言によるセキュリティ機構とプログラムによるセキュリティ機構について述べる。さらに、Web 層のユーザ認証について述べる。

6. EJB 層のセキュリティ

EJB 層のリソースを保護するために使用する、宣言によるセキュリティ機構とプログラムによるセキュリティ機構について述べる。さらに、EJB 層における認証についても述べる。

7. アプリケーションクライアント層のセキュリティ

アプリケーションクライアント層のセキュリティ機構について述べる。

8. EIS 層のセキュリティ

EIS リソースにセキュアにアクセスするセキュリティ機構について述べる。

9. セキュリティアイデンティティの伝達

コンポーネント間のセキュリティアイデンティティの伝達について述べる。

10. J2EE のユーザ、レルム、およびグループ

J2EE セキュリティアーキテクチャにおけるユーザ、レルム、グループの概念について述べる。

11. サーバ証明書の設定

J2EE サーバで使用する X.509 証明書とその役割と設定について述べる。

12. 監査

J2EE セキュリティアーキテクチャにおける監査と監査機構について述べる。

13. チェックリスト

1. 本書の概要

1.1. 調査概要

J2EE サーバアプリケーションプログラミングモデルにおいて、開発者は J2EE サーバアプリケーションでこれらアプリケーションのセキュリティ機構固有の詳細な実装をする必要はない。J2EE は、アプリケーションの移植性を強化するために、アプリケーションの移植性を強化するようにこのようなモデルを提供しているため、様々なセキュリティ環境に配備することが可能である。このセキュリティプログラミングモデルを本講座で探求していく。

1.2. 調査期間

調査期間は、2002 年 12 月から 2003 年 2 月である。

1.3. 調査対象

本講座は、Sun Microsystems 社が公開している J2EE SDK 環境上で J2EE を関するセキュアプログラミングを対象として調査、分析、解析を実施して作成したものである。

1.4. 調査方法

本講座は、Sun Microsystems 社が公開している J2EE と J2EE SDK に関する資料、文献やインターネットや書籍の情報などで情報収集を行い、J2EE に関するセキュリティの調査、設計、分析、解析を行う。必要に応じてサンプルプログラムを作成し検証する。その結果を J2EE でのセキュアプログラミングに必要なセキュリティベストプラクティスを啓発資料としてまとめる。

1.5. 想定される読者

本講座は、J2EE のリファレンス実装である J2EE SDK を前提として、J2EE を使用する際のセキュアなプログラムの実装および運用などの指針をまとめたものである。本ガイドラインは次に挙げるスキルを有している読者を想定して作成されている。

- Java 2 SDK がインストールできる
- Java 2 EE SDK がインストールできる

- 基本的な Java 言語プログラミングを理解してプログラミングができる
- 基本的な Java Servlet を理解してプログラミングができる
- 基本的な Java Server Page を理解してプログラミングができる

すなわち、Java を使用してサーバプログラミングを経験し運用管理している読者を想定する。ただし、J2EE に関しては非常に多岐な分野を熟知している必要があるので、本報告書に記載されている参考文献を調べながら読んでいただきたい。

1.6. 前提となる環境

J2EE SDK がインストールされたプラットフォームを前提とする。さらに、J2EE SDK に付属しているサンプルプログラムを利用する。

2. J2EE セキュアプログラミング概要

J2EE プラットフォームは、アプリケーションコンポーネントの開発とアSEMBルを行う側と運用環境におけるアプリケーションを設定する側との間の宣言規約を定義している。アプリケーションのセキュリティにおいては、アプリケーションプロバイダは、アプリケーションのセキュリティ要件を宣言することを要求される。そのセキュリティ要件は、アプリケーション構成時に要件を満たすものでなければならない。アプリケーションで使用する「宣言によるセキュリティ」機構は「配備記述子」と呼ばれるドキュメントの宣言構文で表現される。アプリケーションデプロイヤーはコンテナ固有のツールを使用して、配備記述子に基づくアプリケーションのセキュリティ要件を、J2EE コンテナが実装しているセキュリティ機構にマップする。J2EE SDK は、この機能を `deploytool` で提供する。

「宣言によるセキュリティ」に対して、「プログラムによるセキュリティ」とは、セキュリティ性を備えたアプリケーションによるセキュリティ判断のことを示す。プログラムによるセキュリティは、宣言によるセキュリティだけではアプリケーションのセキュリティモデルを表現するのに不十分な場合に役立つ。たとえば、アプリケーションは、時間帯、呼び出しパラメータ、EJB や Web コンポーネントの内部状態などに基づいて、承認の判断を行うことがある。アプリケーションによっては、データベースに格納されたユーザ情報に基づいてアクセスを限定することもある。

J2EE アプリケーションは、異なるコンテナに配備可能なコンポーネントで構成されている。これらのコンポーネントは、多層エンタープライズアプリケーションを構築するために使用される。J2EE セキュリティアーキテクチャの目的は、各層をセキュリティ保護することにより、エンド・ツー・エンドのセキュリティを達成することである。

各層には、保護されているリソースと保護されていないリソースが含まれている。承認されたユーザのみがアクセスできるようリソースを保護する必要がでてくる場合が頻出する。「承認」は、保護されているリソースへの制御されたアクセスを提供する。承認は、識別と認証に基づいている。「識別」とはシステムによってエンティティの認識を可能にするプロセスである。「認証」とはユーザ、デバイス、またはコンピュータシステム内において他のエンティティのアイデンティティを確認するプロセスであり、システムリソースへアクセスするための前提条件となるプロセスである。

?承認は、保護されていないリソースへのアクセスには必要とされない。承認は認証に基づいて構築されており、保護されていないリソースにアクセスする場合は、認証も必要な

いからである。認証を使用しないリソースへのアクセスは、「非認証アクセス」または「匿名アクセス」と呼称される。

3. J2EE セキュリティアーキテクチャによるセキュリティ実装からの解放

この章では、J2EE セキュリティアーキテクチャを使用することで、セキュリティに関する実装からアプリケーション開発者すなわち Bean 開発者が解放されることを述べる。

3.1. J2EE セキュリティアーキテクチャの目的

J2EE セキュリティアーキテクチャの主たる目標の一つは、セキュリティ機構やセキュリティ実装からプログラム開発者を可能な限り解放し、アプリケーションを多種多様な環境に安全かつ容易に配備できることである。この目標を達成するには、アプリケーションのセキュリティ仕様要件をアプリケーションの外側に明確に定義することである。

3.1.1. 従来のエンタープライズサーバアプリケーションプログラミングモデルの問題

従来のエンタープライズサーバアプリケーションプログラミングモデルにおいて、開発者は、ビジネスロジックを設計、実装するだけでなくセキュリティロジックをも設計、実装する。開発者がユーザやグループを定義し、アプリケーションの権限を定義したユーザやグループに割り当てる。さらに、これらのユーザやグループを効率よく格納しアクセスする機構を提供し、実行時に承認検査を行うロジックを実装しなければならない。このような複雑なセキュリティ機能の実装は面倒な作業である。また、低水準のシステム API やセキュリティに関する専門的な知識が必要となる。ビジネスロジックを設計実装するプログラマは、一般的にセキュリティに詳しくない。それにも関わらず、オペレーティングシステムの低水準のセキュリティ API を使用し、セキュリティに関するコードを実装しなければならない。出来上がるこのようなビジネスサーバアプリケーションは巨大になる傾向があり、従って、セキュリティホールが存在する可能性が高くなる。

3.1.2. J2EE セキュリティアーキテクチャによる難解なセキュリティ機構からの解放

J2EE セキュリティアーキテクチャは、エンタープライズサーバのアプリケーション開発者から難解なセキュリティ機構から解放し、アプリケーションそのものの開発に専念することが可能である。言い換えると、J2EE セキュリティアーキテクチャは、ビジネスロジックの専門家である Bean の開発者をセキュリティの設計と実装から可能な限り解放し、セキュリティの専門家である EJB コンテナベンダーに委譲することが可能である。もちろん、J2EE セキュリティアーキテクチャ機構を理解しセキュリティをプログラムで実装することができるが、J2EE セキュリティプラットフォームで提供される API を利用することを

推奨する。この提供される API を利用することにより、プログラムの可用性や再利用性が非常に高まる。J2EE セキュリティアーキテクチャでは、アプリケーションデプロイヤーは、配備時にセキュリティ要件を決定することが可能であるので、アプリケーションの運用される環境が変化してもプログラムの変更をほとんど必要としないでセキュリティ要件を変更することが可能である。

3.1.3. J2EE セキュリティアーキテクチャにおける目標

J2EE セキュリティアーキテクチャでは、セキュリティに関して次のことを目標とする。

- アプリケーションのセキュリティを設計、実装するタスクを EJB コンテナに委譲することで、Bean の開発者をこれらの作業から解放する。
- アプリケーションアセンブラやデプロイヤーがセキュリティポリシーを宣言によって設定することを推奨し、Bean クラスでのセキュリティポリシーのハードコーディングを防止する。これを宣言によるセキュリティの実装と呼称する。これに対して、プログラムによるセキュリティの実装がある。
- セキュリティ機構に依存することなく、複数の EJB サーバへの移植性がある EJB アプリケーションを作成できるようにする。

3.1.4. J2EE セキュリティアーキテクチャでの役割と責任

J2EE セキュリティアーキテクチャの設計と実装において、役割と責任が明確に分離されて定義されている。その役割と責任に関しては次のとおりである。

- **Bean 開発者**
 - Bean 開発者は次の項目に対して責任がある。
 - 宣言によるセキュリティの実装を使用する場合、Bean の開発者はビジネスロジックの作成に専念する。セキュリティの管理と実装はアプリケーションアセンブラ、デプロイヤー、管理者、EJB コンテナに任せる。
 - プログラムによるセキュリティを使用する場合、<security-role-ref>要素を用いて、EJB コードで使用したすべてのセキュリティロール名を配備記述子で宣言しなければならない。J2EE セキュリティアーキテクチャでは、Bean の開発者が EJB のビジネスメソッドにセキュリティメカニズムを宣言に従って、またはプログラムで実装することを推奨しない。セキュリティを実装する責任は、EJB コンテナ、

アプリケーションアセンブラ、アプリケーションデプロイヤにあることが推奨される。

- Bean 開発者は、配備記述子を使用して、EJB の適切なセキュリティポリシーに関するセキュリティ関連情報をデプロイヤに伝えることが推奨されている。

- アプリケーション開発者

アプリケーション開発者は次の項目に対して責任がある。

- アプリケーションロールを指定する
- アプリケーションコンポーネントすなわちサーブレット、JSP、EJB コンポーネントに対するロールベースのアクセス制御を定義する。
- プログラムによるセキュリティ実装を使用する場合は、ユーザロールを確認し、これらのロールに基づき機能へのアクセスの承認する。プログラムによるセキュリティの実装の管理では、コンテナでセキュリティを管理すなわち宣言によるセキュリティの実装の管理をする代わりに、アプリケーション内にセキュリティに関するコードがハードコードされるため推奨できない。

- アプリケーションアセンブラ

アプリケーションアセンブラまたはアプリケーションコンポーネントプロバイダは、コンポーネントに組み込まれた次のようなセキュリティ関連事項をすべて確認する必要がある。

- コンポーネントがプログラムによるセキュリティ実装を使用している場合、isCallInRole メソッドまたは isUserInRole メソッドを呼び出すときに使用されるロール名
- コンポーネントがアクセスするすべての外部リソースへの参照
- コンポーネントが行うすべての内部コンポーネント呼び出しへの参照

- アプリケーションデプロイヤ

アプリケーションデプロイヤは、アプリケーションアセンブラが提供したすべてのコンポーネントのセキュリティビューを受け取り、それを使用して次のようにアプリケーションにおける特定のシステム環境を保護する。

- ユーザまたはグループあるいは、その両方をセキュリティロールに割り当てる
- コンポーネントメソッドへのアクセスに必要な権限を修正して、特定の配備シナリオの要件に適合させる

- システム管理者

システム管理者は、ユーザアカウントやグループの作成、システム監査の実施、J2EE

サーバを実行するシステムの適切な保守などの、システム管理作業に責任を持つ。アプリケーションデプロイヤーとシステム管理者は、セキュリティ ID の作成、管理、システム上のエンティティへのマッピングを行うために、協力して作業を進める。システム管理者には、セキュリティ監査の記録を調査し管理する責任がある。

- EJB コンテナプロバイダ

EJB コンテナベンダーまたはアプリケーションサーバベンダーは、EJB をサポートする J2EE 認定のアプリケーションサーバを提供する。また、アプリケーションサーバの提供に加えて、関係者が J2EE アプリケーションの作成、アSEMBル、配備を行うための配備ツールを提供する。コンテナは、実行時に前述の EJB セキュリティを適用しセキュリティ監査追跡機能を提供する必要がある。この機能では、すべての `java.security` インタフェースのすべての例外と EJB サーバ、コンテナ、ホームインタフェース、コンポーネントインタフェースへのアクセス拒否をログに記録するものでなければならない。

J2EE プラットフォームにはこれらの役割と責任が存在して、それぞれ独立している。これらの役割は、抽象的な概念であり、現実の J2EE アプリケーション開発においては、1 人の担当者が複数の役割を兼任したり複数の担当者が 1 つの役割を受け持つこともある。

J2EE セキュリティアーキテクチャの理解において、これらの役割を理解することは非常に重要である。このような役割と責任を明確に分離定義することにより、それぞれの担当責務に専念できる。従って、プログラムの質の向上に繋がり、セキュリティ向上に寄与する。

3.1.5. J2EE セキュリティアーキテクチャにおけるセキュリティ要件に関する注意事項

J2EE アーキテクチャの開発において、Bean のプロバイダやアプリケーションアセンブラやアプリケーションデプロイヤーが一箇所に集まり開発しなくてもよいし、そのような開発態勢を有する可能性が高い。このことが、セキュリティ実装を難しくすることがある。このような分散開発環境では、Bean 開発者は、アプリケーションアセンブラが組み立てた J2EE アプリケーションのセキュリティ要件や J2EE アプリケーションが最終的に配備される運用環境を予測できない。アプリケーションアセンブラは、J2EE アプリケーションが最終的に配備される運用環境やセキュリティ要件も完全に把握していない可能性がある。ほとんどの場合、アプリケーションアセンブラは、ソースコードにアクセスできない。従って、Bean 開発者が Bean の実装クラスに何らかのセキュリティ要件をプログラムによるセキュリティの実装によりハードコードしていたとしても気付くことはない。アプリケーションデプロイヤーは、セキュリティ要件を J2EE アプリケーションサーバが存在するオペレーティ

ングシステム環境にマップしなければならないため、アプリケーションアセンブラが指定したアプリケーションのセキュリティ要件と Bean 開発者がハードコードしている可能性のあるセキュリティロジックを最終的にすべて把握しなければならない。

J2EE セキュリティアーキテクチャの課題は、プログラムによるセキュリティの実装により Bean クラスに実装されたセキュリティ要件を、EJB コンポーネントを使用して J2EE アプリケーションを作成するアプリケーションアセンブラに如何に伝えるかである。また、アプリケーションアセンブラは、J2EE アプリケーションのセキュリティ要件をデプロイヤーに伝えなければならない。従って、アプリケーションデプロイヤーはアプリケーションの配備の際にはセキュリティ要件を満たさなければならない。

3.1.6. まとめ

J2EE セキュリティプラットフォームはアプリケーション開発者から難解なセキュリティ機構から解放し、アプリケーションの開発に専念することが可能である。J2EE セキュリティプラットフォームでは、アプリケーションデプロイヤーは、配備時にセキュリティ要件を決定することが可能であるので、アプリケーションの運用される環境が変化してもプログラムの変更をほとんど必要としないでセキュリティ要件を変更することが可能である。

3.1.7. 関連記事

- 4.2.セキュリティロール参照と宣言のリンク
- 4.3. J2EE のユーザとグループへのロールのマッピング
- 6.1. EJB 層のセキュリティ
- 6.2. EJB のメソッドのアクセス権の宣言

3.1.8. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun™ ONE Application Server 開発者ガイド, 第3章 J2EE アプリケーションセキュ

リティ

<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>

4. セキュリティロール

アプリケーションコンポーネントプロバイダまたはアプリケーションアセンブラが定義する論理的なユーザのグループ化とその使用と利点について述べる。

4.1. コンポーネントに対するアクセス権の定義

EJB や Web コンポーネントを設計する際、コンポーネントのメソッドにアクセスするユーザやグループについてセキュリティの観点から常に考慮する必要がある。さらに、ユーザやグループに与えられるアクセス権限は、適切かつ最小に定義する必要がある。J2EE セキュリティアーキテクチャにおいて、このコンポーネントに対するアクセス権限が与えられるユーザの集合をセキュリティロールと呼称する。

4.1.1. セキュリティロールとは

セキュリティロールは、ユーザの抽象的かつ論理的なグループである。セキュリティロールは、J2EE アプリケーションコンポーネントプロバイダや J2EE アプリケーションをアセンブル担当者によって、少なくとも1つ以上配備記述子<security-role>タグにより定義される。その後、J2EE アプリケーションを配備する際、アプリケーションデプロイヤはセキュリティロールを J2EE 運用環境のセキュリティアイデンティティへマップする。これにより、アプリケーションの機能を定義することになる。

また、セキュリティロールは、宣言によるセキュリティにもプログラムによるセキュリティに使用される。それらの方法については、「5.10.1. Web 層のプログラムによるセキュリティ」、「6.3.1. EJB 層のプログラムによる」で述べる。この宣言によるセキュリティを利用することにより、セキュリティポリシーが EJB コードから独立することになる。このような J2EE セキュリティアーキテクチャは実行環境に応じて配備時に適宜調整できる。すなわち、EJB コンポーネントの柔軟性と移植性が高い。宣言によるセキュリティを使用することを推奨する。

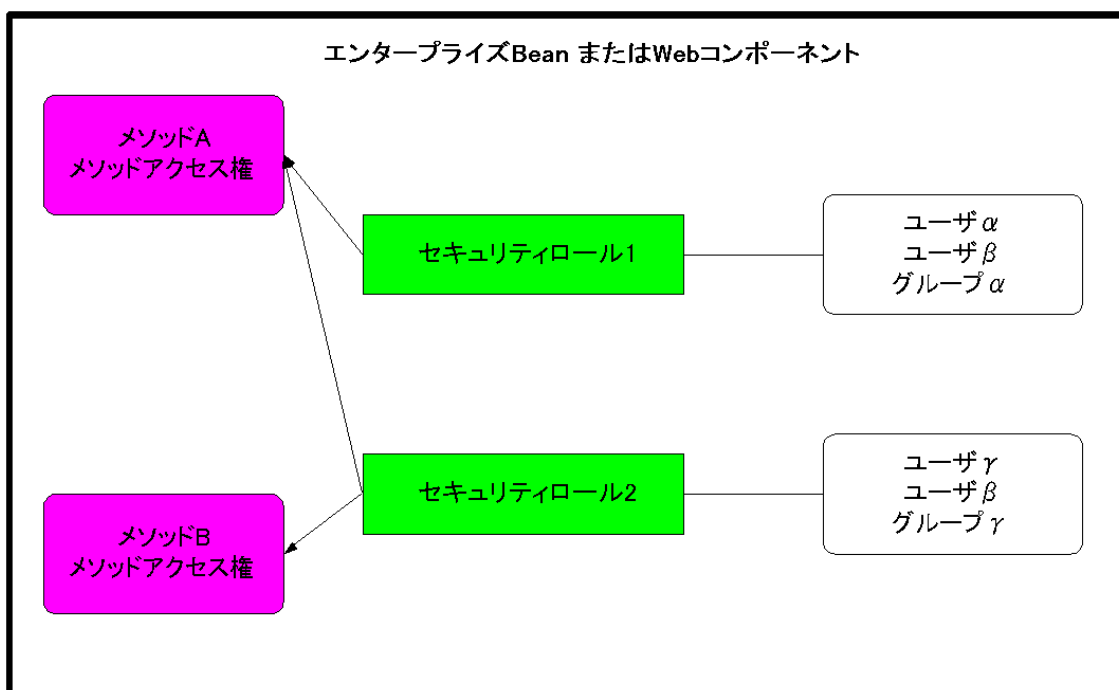
4.1.2. セキュリティロールを利用しない欠点と使用する利点

抽象的かつ論理的なユーザの集合であるセキュリティロールを利用しない場合、アプリ

ケーションは個別のユーザを対象にしてアクセス権の管理を行うことになる。このためには具体的なユーザ名を含んだプログラムの実装を必要とする。そのため、アプリケーションを利用するユーザの構成が変わった場合にはソースコードの修正が必要になるなど、アプリケーションのメンテナンス性が低下することになる。これは、セキュリティ上の脆弱性を引き起こす可能性がある。セキュリティロールを使用することにより宣言によるセキュリティの実装の手法が使える、その利点を享受できる。従って、セキュリティロールを利用すべきである。

4.1.3. セキュリティロールとユーザ、グループの違いについて

J2EE のユーザとグループはユーザカテゴリを表す。しかし、ユーザカテゴリとロールとは異なるスコープを持つ。J2EE のユーザとグループは、J2EE サーバ全体に指定されて J2EE プラットフォーム全体で一意的に識別される。一方、ロールは、J2EE サーバの特定のアプリケーションのみを対象とする。また、J2EE のユーザとグループは J2EE プラットフォームが存在する OS のユーザとグループとは関係なく J2EE プラットフォーム内で独立している。



セキュリティロールとユーザ、グループの関係

4.1.4. セキュリティロールを定義するうえでの注意

ロールは、J2EE アプリケーション固有の抽象概念であることを十分理解する必要がある。J2EE セキュリティロールにおいて、ユーザ、ユーザカテゴリ、グループ、セキュリティロールなどの複数の抽象的な概念が存在する。それぞれの概念は異なるスコープを持ち、異なる担当者によって管理運用されることに要注意すべきである。すなわち、グループはユーザのカテゴリを表すが、グループの範囲は、ロールの範囲と異なる。ロールは、J2EE アプリケーション固有の抽象概念であるが、グループは、現在のレルムに基づいた環境固有のユーザの集合である。グループのメンバーシップは、基になるレルムの実装によって決定される。レルム、グループ、ユーザカテゴリについては、「10. J2EE のユーザ、レルム、およびグループ」を参照すること。

セキュリティロールは、J2EE アプリケーションコンポーネントプロバイダや J2EE アプリケーションをアSEMBL担当者によって定義される。また、配備時にアプリケーションデプロイヤによって配備時の環境に応じて修正されることがある。

セキュリティロールのスコープは、EJB JAR ファイル内のすべての EJB に適用されることを再度、強調しておく。

セキュリティロールは、J2EE アプリケーションの動作に密接に関連する。従って、セキュリティロールに付与される名前は、対象となる J2EE アプリケーションに関するユーザ情報や顧客情報や案件名などを的確に表し、省略されないものを使用すべきである。さらに、セキュリティロールの定義において、安全や管理面から最小限の定義にとどめるべきである。不要なセキュリティロールを作成しないことである。さらに、セキュリティロールに不必要にユーザやグループを含めないべきである。

4.1.5. セキュリティロールに関する例

たとえば、給与計算を行う J2EE アプリケーションにおいて、`employee` と `administrator` という2つのセキュリティロールが指定されると仮定する。給与更新作業を実行できるのは `administration` セキュリティロールをもつ主体のみである。一方、給与読み取り作業はどちらのセキュリティロールでも実効できるようにセキュリティロールを使用する。

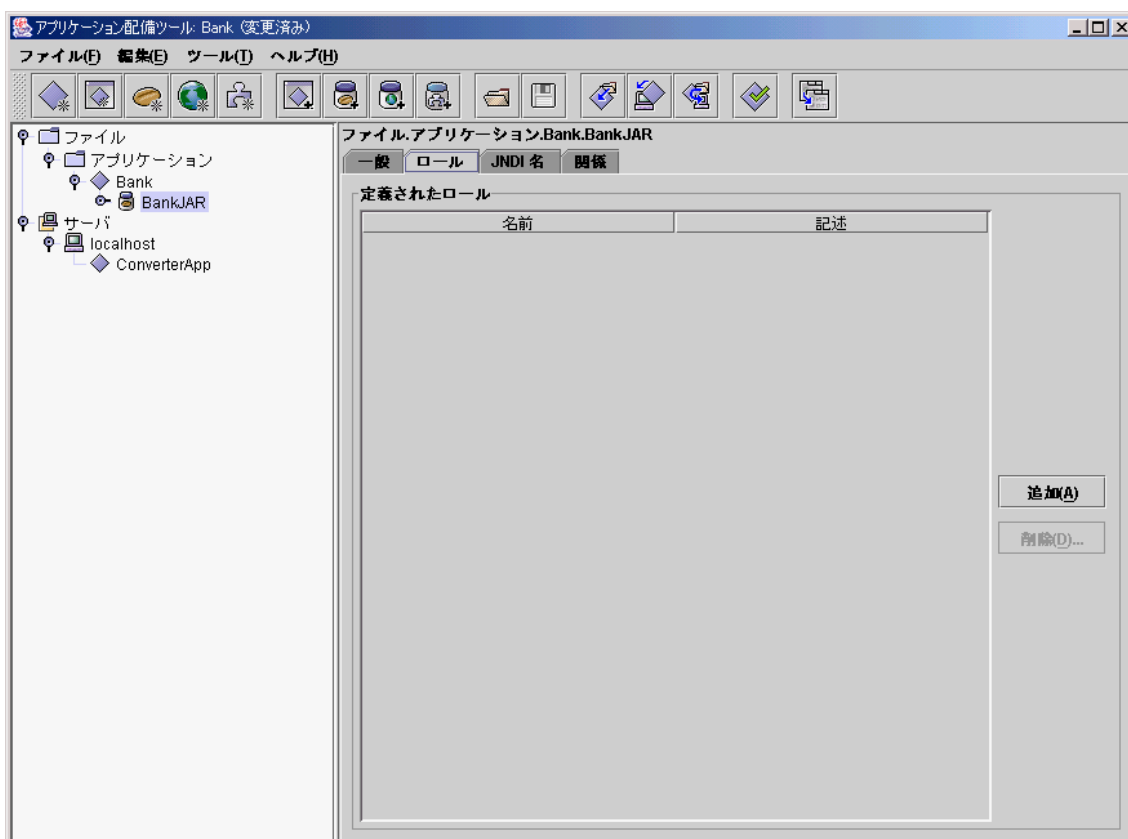
給与計算 J2EE アプリケーションの配備時に、デプロイヤはユーザである管理者および従業員の主体(プリンシパル)または、グループの集合とそれぞれのセキュリティロールとのマッピングを提供することができる。

給与更新メソッドが実行される時に、Enterprise Bean のコンテナは、Web サーバから送られた主体またはグループがそのメソッドを実行できるロールにマッピングされているかをチェックできる。また、メソッド自体でセキュリティ API の1つを使用して、このチェックを実行することもできる。

4.1.6. セキュリティロールの作成作業

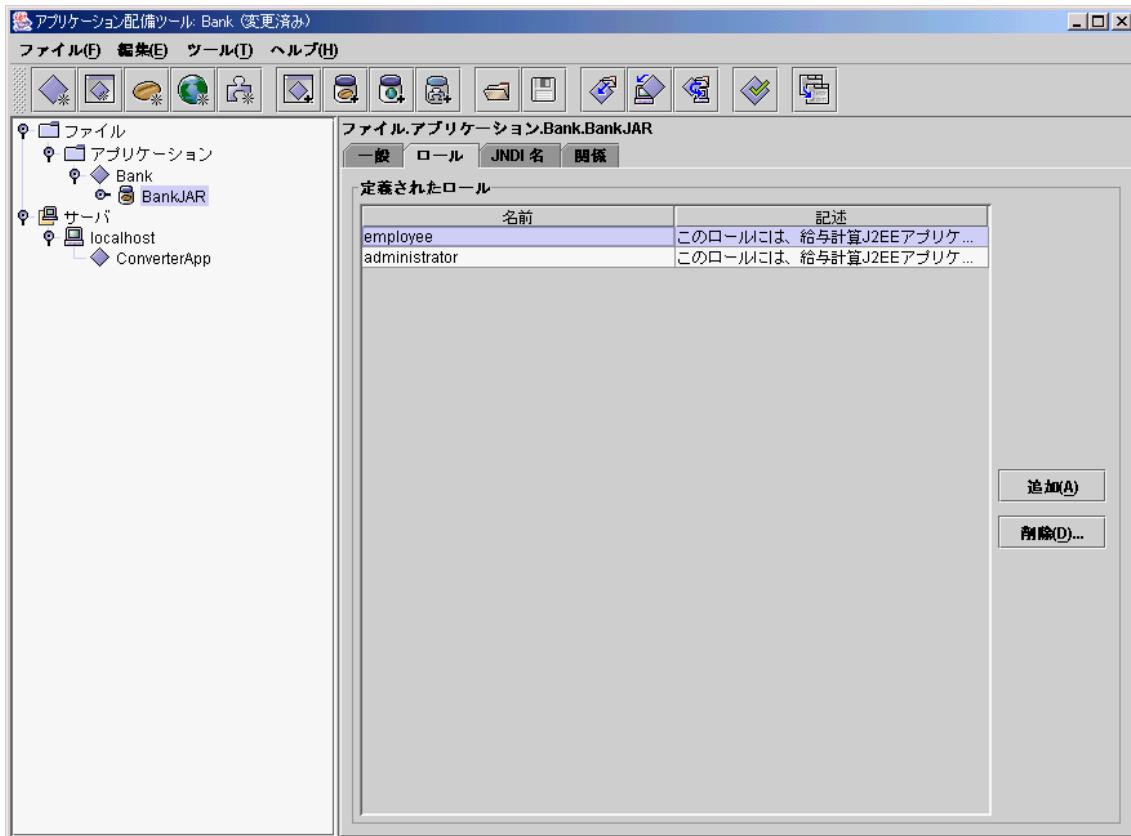
J2EE アプリケーションのセキュリティロールを作成するには、セキュリティロールをアプリケーションの EJB JAR ファイルかアプリケーションを含む WAR ファイルに対して定義する。次に典型的な定義の手順を示す。この手順により配備ツールでセキュリティロールを作成することができる。

1. EJB の EJB JAR ファイル、または Web コンポーネントの WAR ファイルを選択する。ここでは、BankJAR を選択する。



2. 「ロール」パネルで「追加」をクリックする。

3. テーブルの「名前」フィールドと「記述」フィールドへ値を入力する。



この作業の結果、BankJAR の配備記述子は次のように自動的に設定される。その抜粋を示す。配備記述子には、employee と administrator の2つの role-name の要素が生成されている。配備記述子のすべてを閲覧したい場合は、BankEJB にフォーカスして右クリックで「記述子ビューア」を選択する。

```

<assembly-description>
  <security-role>
    <description>
      このロールには、給与計算 J2EE アプリケーションの給与読み取り
      作業にアクセスできる社員が含まれる。
    </description>
    <role-name>employee</role-name>
  </security-role>
  <security-role>
    <description>
      このロールには、給与計算 J2EE アプリケーションの給与読み取り
      作業や給与更新作業にアクセスできる社員が含まれる。
    </description>
  </security-role>

```

```
        <role-name>administrator</role-name>
    </security-role>
...
</assembly-description>
```

このサンプルにおいては、2つのサブ要素を持つ<security-role>要素で構成される。サブ要素は、特定のロール名を指定する<role-name>要素とセキュリティロールを説明するオプションの<description>要素で構成される。Bean 開発者とアプリケーションアセンブラは、<description>要素を使用してセキュリティ情報をデプロイヤーに伝えられなければならない。

4.1.7. まとめ

J2EE プラットフォームにおいては、宣言によるセキュリティを用いてセキュリティロールを定義してコンポーネントのメソッドへのアクセス権を管理し、柔軟性で移植性の高いアプリケーションを構築することができる。

4.1.8. 関連記事

- 3. J2EE セキュリティアーキテクチャによるセキュリティ
- 5.10. Web 層のプログラムによるセキュリティの使用
- 6.3. EJB 層のプログラムによるセキュリティ
- 10. J2EE のユーザ、レルム、およびグループ

4.1.9. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- J2EE アプリケーションのプログラミング
http://jp.sun.com/products/software/tools/jde/documentation/j2eeapps_ja.pdf
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>

- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

4.2. セキュリティロール参照と宣言のリンク

EJB や Web コンポーネントからセキュリティロールを参照する場合、セキュリティロール参照を宣言し、より抽象化した参照を利用することができる。これにより、アプリケーション内でのセキュリティロール参照が抽象化され、移植性を高めることができる。

4.2.1. セキュリティロール参照

セキュリティロール参照により、EJB または Web コンポーネントは、既に J2EE アプリケーションで既に定義されているセキュリティロールを参照することができる。セキュリティロールは、J2EE アプリケーションに固有のユーザの論理グループでもあり、顧客情報や案件名といった一般的な特徴により分類され命名される。J2EE アプリケーションが配備されると、ロールは運用環境のセキュリティアイデンティティへマップされる。セキュリティアイデンティティとは、認証によりユーザへ割り当てられたアイデンティティである主体(Principal)やグループことを示す。これに基づき、特定のセキュリティロールを持つユーザは、J2EE アプリケーションへの関連付けられたアクセス権を持つ。セキュリティロールと呼ばれているものの実体は、セキュリティロールのロールからセキュリティアイデンティティへのリンクである。

J2EE アプリケーションのアセンブル工程において、アプリケーションアセンブラは、アプリケーション用のセキュリティロールを作成し、そのロールを利用可能なセキュリティ機構に関連付ける。アプリケーションアセンブラは、個々の EJB、サーブレット、JSP などのセキュリティロール参照を、J2EE アプリケーションのために定義されたロールとリンクすることで解決することができる。

セキュリティロール参照により、EJB または Web コンポーネントは、既存のセキュリティロールすなわち、「4.1.6. セキュリティロールの作成作業」で作成されたセキュリティロールを参照することができるようになる。

セキュリティロール参照では、Web コンポーネントまたは EJB から呼び出されるセキュリティロール名と、アプリケーションに定義されたセキュリティロール名のマッピングを定義する。プログラムによるセキュリティにおいては、Web コンポーネントからのロー

ル名の呼び出しでは `isUserInRole(String name)` を使用し、EJB からのロール名の呼び出しでは `isCallerInRole(String name)` によって行う。これらにメソッドについては、「5.10. Web 層のプログラムによるセキュリティの使用」および「6.3. EJB 層のプログラムによるセキュリティ」を参照すること。

4.2.2. セキュリティロールとセキュリティロール参照のリンクの例

Bean の開発者が指定した `supervisor` というロール名は、EJB のコードで使用されている抽象的なセキュリティロール名にすぎない。従って、J2EE アプリケーションで指定されているセキュリティロールにリンクしなければならない。J2EE アプリケーションでは、アプリケーションアセンブラが管理のためのロールとして `manager` というセキュリティ ID を指定していることもあり得る。このため、Bean 開発者は、`<security-role-ref>` 要素の `<role-link>` 要素を使用し、EJB のコードで使用している `supervisor` というセキュリティロールを、アプリケーションで指定されている `manager` というセキュリティロールにマップする。次のコードは `<role-link>` 要素の使い方を示している。このコードの `<security-role-ref>` セクションは、EJB のコードのロール名 `supervisor` を J2EE アプリケーションのセキュリティロール名の `manager` にマップしている。`<role-link>` 要素で指定される J2EE アプリケーションのセキュリティロール名 `manager` は、配備記述子の `<assembly-descriptor>` セクションで、`<security-role>` 要素の中の `<role-name>` 要素に宣言しなければならない。

```
<enterprise-beans>
.
.
.
<entity>
  <ejb-name>AddressEJB</ejb-name>
  <ejb-class>com.j2eebootcamp.ejbbook.AddressBean</ejb-class>
  .
  .
  .
  <security-role-ref>
    <description>
      このロールは、クラスのスケジュールの追加、削除、変更の権限を持つ
      ユーザにのみ割り当てられる
    </description>
    <role-name>supervisor</role-name>
    <role-link>manager</role-link>
  </security-role-ref>
</entity>
.
.
```

```

.
<assembly-descriptor>
  <security-role>
    <description>
      グループの管理者の権利を持ち、スケジュールを追加、削除、変更できる
    </description>
    <role-name>manager</role-name>
  </security-role>
  <security-role>
    <decription>登録された受講生</description>
    <role-name>student</role-name>
  </security-role>
  .
  .
  .
</assembly-descriptor>
.
.
.
</enterprise-beans>

```

4.2.3. セキュリティロールとセキュリティロール参照の運用についての注意

Bean 開発者が宣言したセキュリティロール名が、アプリケーションアセンブラが割り当てたロール名と偶然一致したとしても、<security-role-ref>要素には該当するロール名を含んだ<role-link>要素を指定しなければならない。

EJB のセキュリティをプログラムで実装する場合でも、アプリケーションアセンブラは、<security-identity>要素と<method-permission>要素を使用して、アプリケーションのセキュリティ要件を規定するセキュリティビューを配備記述子で定義する。また、Bean の開発者が EJB のコードで指定したセキュリティロールを、<security-role-ref>要素で指定されたアプリケーションのセキュリティロールにリンクする場合もある。

EJB のセキュリティを宣言によって実装する場合でもプログラムで実装する場合でも、アプリケーションデプロイヤーはアプリケーションを配備する前に、すべてのセキュリティ問題を解決し、配備記述子で指定された論理的なセキュリティロールをマップすることでセキュリティビューを実装しなければならない。

4.2.4. セキュリティロール参照を宣言せずにロール名を参照する欠点

セキュリティロール参照を宣言せずに、アプリケーションに定義されたロール名を Web コンポーネントや EJB で直接利用することは可能である。しかし、セキュリティロール参照を宣言した場合に比べて構造的な柔軟性が失われ、メンテナンス時にプログラムを直接

修正する可能性が増えることになる。極力、セキュリティロール参照を宣言すべきである。

4.2.5. セキュリティロール参照とセキュリティロールにマップに関する注意

セキュリティと管理からセキュリティロール参照とセキュリティロールのマッピングは最小限かつ適切に行うべきである。不要なマッピングは行わない。そのために、アプリケーションのビジネスロジックを十分に吟味して、最小限の必要かつ適切なセキュリティポリシーを作成しセキュリティロールとセキュリティ参照を作成し、マッピングの作業を行う。セキュリティロールとセキュリティロール参照のリンクが増えれば、JVM に対して特権得るために Java の低レベルの API の `doPrivileged` メソッドを呼び出すことが増える。これは、パフォーマンスに影響を与えることになる。従って、パフォーマンスの面からもマッピングは必要最小限に抑えるべきである。

4.2.6. セキュリティロールと最小特権

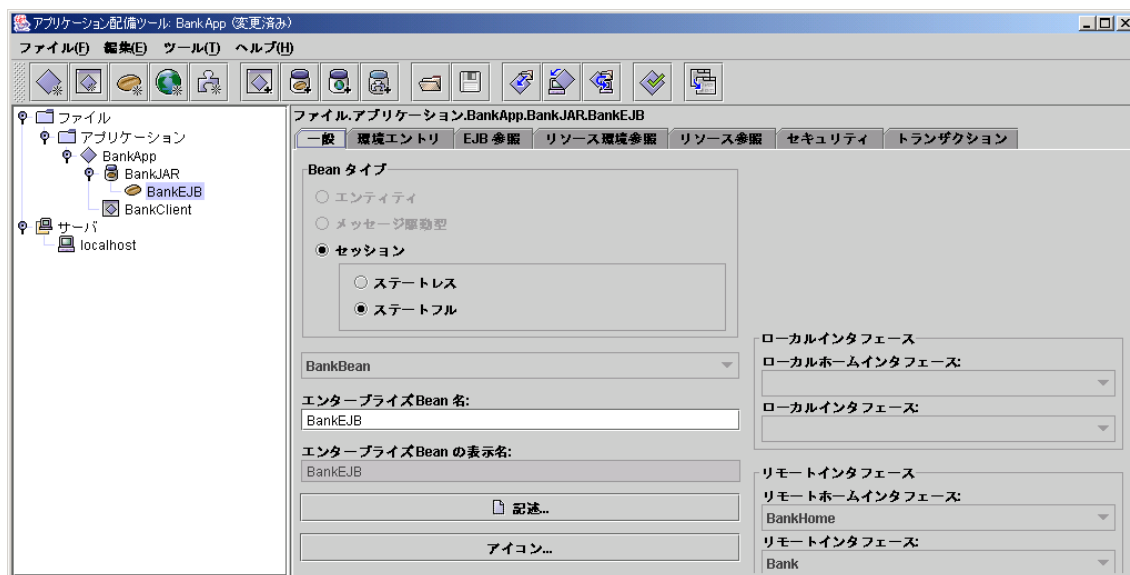
通常、J2EE サーバアプリケーションは、サーバのファイルシステムから読み取りまたは書き込みを行うための特権、データベースやアクセスする特権、アプリケーションサーバの機能を利用する特権、ソケットを使用して通信するための特権を持って実行される。これらの特権は、サーバアプリケーション実行時のセキュリティアイデンティティによってある程度制限される。J2EE サーバアプリケーションを構成する Web コンテナや EJB や配備記述子にはセキュリティ上の脆弱性をまったく否定することはできない。従って、セキュリティアイデンティティに関連する特権を J2EE サーバアプリケーションの処理に必要な最小限に制限することが重要である。この制限は、最小特権(least privilege)と呼ばれる良く知られているセキュリティ上の概念である。たいていの J2EE サーバアプリケーションと J2EE アプリケーションサーバの実行環境では、基本となるオペレーティングシステムが設定されているセキュリティに関する制限以外には制限はない。しかし、J2EE アプリケーションは J2EE アプリケーションサーバの JVM 上で稼動しているので、この JVM の sandbox のセキュリティポリシーに従い最小特権を強制されている。

4.2.7. セキュリティロール参照をセキュリティロールにマップする作業手順

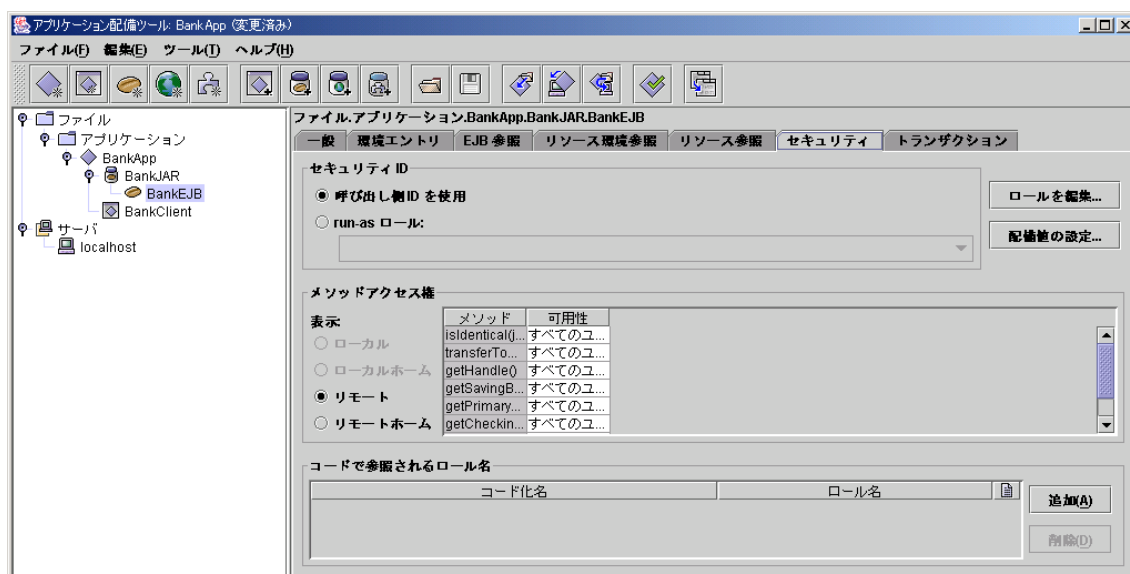
本講座では、J2EE SDK の `deploytool` を使用してセキュリティロール参照をセキュリティロールにマップする作業を行う。

たとえば、`cust` というセキュリティロール参照を、`bankCustomer` というロール名を持つセキュリティロールにマップするには、次の手順で行う。

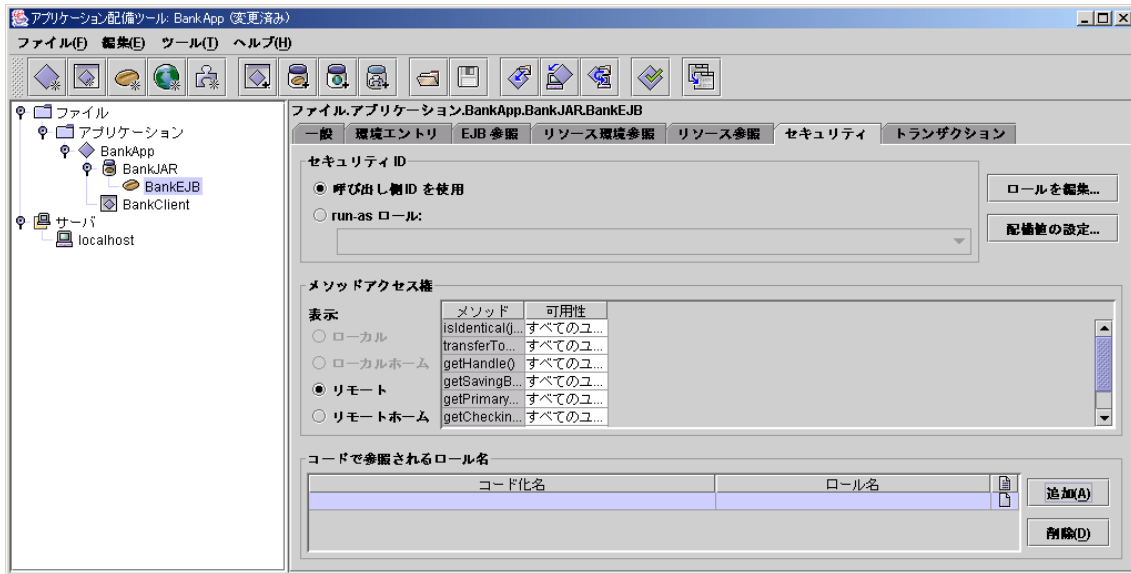
1. Web コンポーネントか EJB を選択する。ここでは、「BankEJB」を選択する。



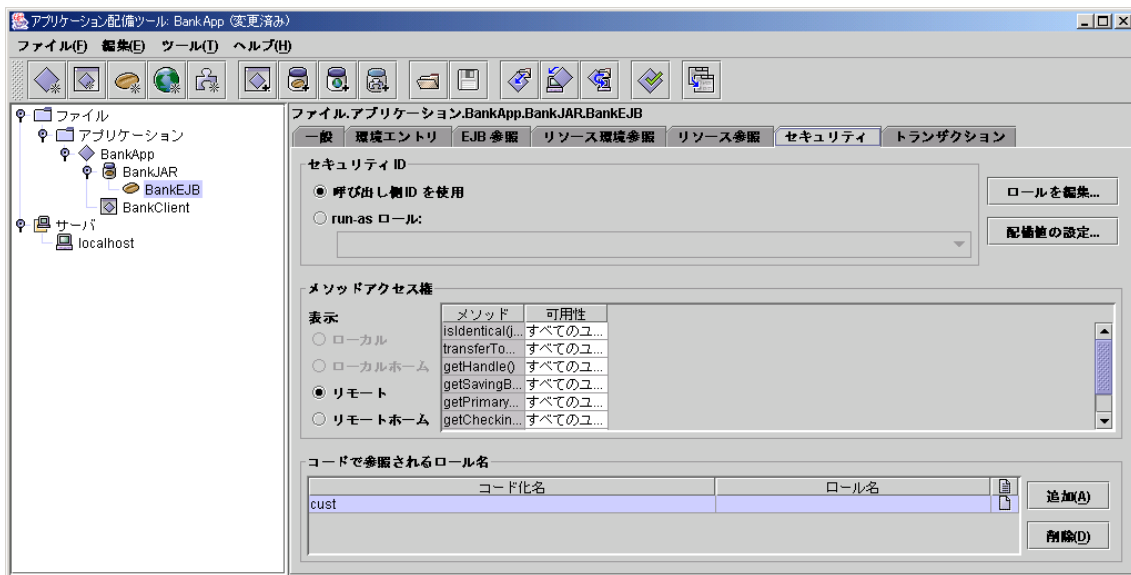
2. 「セキュリティ」タブを選択する。



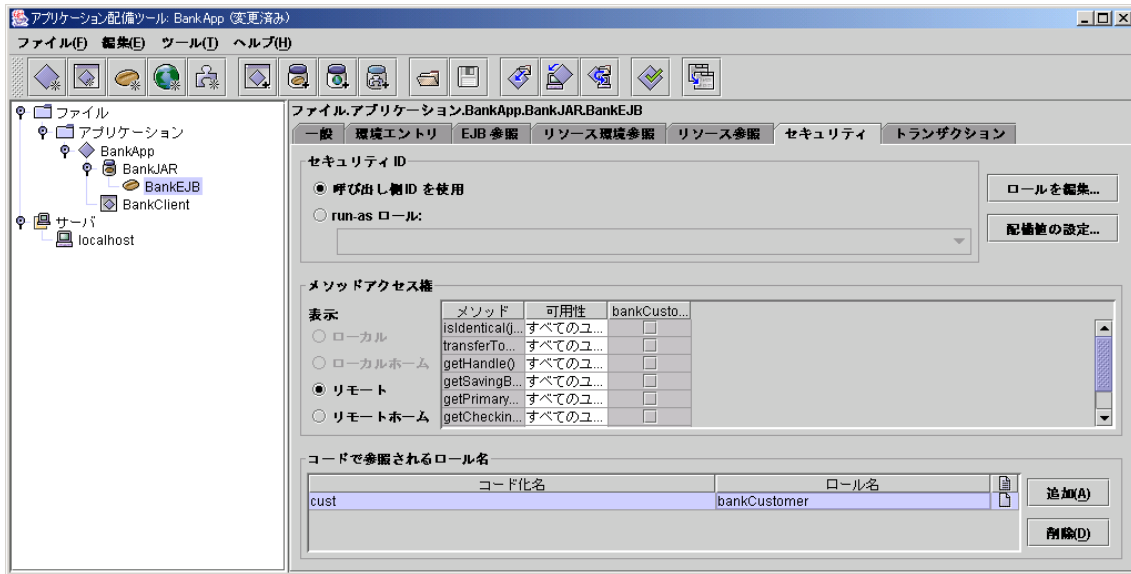
3. 「コードで参照されるロール名」パネルに cust のエントリが表示されない場合は、「追加」ボタンをクリックする。



4. 「コード化名」列に、セキュリティロール参照名 cust を入力する。



5. 「ロール名」列のドロップダウンメニューから、コード化された名前をマップするセキュリティロール名 bankCustomer を選択する。セキュリティロール参照にマップしたいセキュリティロール名が「ロール名」列にない場合、「ロールを編集」をクリックしてロールを追加する。詳細は、「4.1.1. セキュリティロール」を参照すること。



6. 「記述」アイコンをクリックし、cust ロール参照へ説明を追加する。
7. 「記述」ダイアログボックスに説明を入力する。



8. 「了解」をクリックして入力内容を確定、もしくは「取消し」をクリックして入力内容をキャンセルする。

この作業の結果、BankJAR の配備記述子は次のように自動的に設定される。その抜粋を示す。配備記述子のすべてを閲覧したい場合は、BankEJB にフォーカスして右クリックで「記述子ビューア」選択する。

```
<ejb-jar>
  <display-name>BankJAR</display-name>

  ...

  <security-role-ref>
    <description>銀行の顧客(預金者)</description>
    <role-name>bankCustomer</role-name>
    <role-link>bankCustomer</role-link>
  </security-role-ref>
  <security-identity>
    <description></description>
    <use-caller-identity></use-caller-identity>
  </security-identity>

  ...
</enterprise-beans>
<assembly-descriptor>
  <security-role>
    <description>銀行の顧客(預金者)</description>
    <role-name>bankCustomer</role-name>
  </security-role>

  ...
</ejb-jar>
```

この例では、EJB のセキュリティをプログラムで実装する場合には、`isUserInRole("bankCustomer")` および `isUserInRole("cust")` の両方が「メソッドアクセス権」パネルに示されたメソッドに対して TRUE を返すことになる。

コード化名はロール名にリンクされるため、あとになってもコード化名を変更することなく、ロール名を変更することができる。たとえば、ロール名を `bankCustomer` から別の名前へ変更する場合、コード内の `cust` の名前を変更する必要はない。しかし、コード化名の `cust` と新しいロール名の再リンクを行う必要はある。

4.2.8. まとめ

Bean 開発者は、配備記述子 `<security-role-ref>` 要素の中の `<role-name>` 要素で抽象的なセキュリティロール名を宣言しなければならない。さらに、セキュリティロール参照をセキュリティロールにマップして使用すること。

4.2.9. 関連記事

- 5.10. Web 層のプログラムによるセキュリティの使用
- 6.3. EJB 層のプログラムによるセキュリティ
- 10. J2EE のユーザ、レルム、およびグループ

4.2.10. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第 2 版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- Java セキュリティ, スコット・オーク著, オライリー・ジャパン
- Java2 セキュリティプログラミング, ジェミー・ジョウォルスキー 他著, 株式会社ピアソン・エデュケーション
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf

- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>

4.3. J2EE のユーザとグループへのロールのマッピング

J2EE のユーザとグループへのロールのマッピングについて述べる。さらに、J2EE ユーザとグループとロールとの関係を述べる。

4.3.1. J2EE のユーザとグループへのロールのマッピングの機構

ユーザすなわちプリンシパルをロールにマップする作業はアプリケーションデプロイヤが担当する。ユーザをロールにマップする正確な機構は、EJB の仕様には、少なくとも EJB2.0 では定義されていない。配備記述子にはタグが定義されていない。従って、各 J2EE アプリケーションサーバがユーザをロールに対応付ける独自の機構を定義している。J2EE SDK では、EAR(Enterprise Application Resource)ファイルに格納される独自仕様の XML ファイル `sun-j2ee-ri.xml` が定義されている。このファイルは `deploytool` を使って管理され、EJB コンポーネントとともに配備される。このマップは、J2EE アプリケーションごとに行われる。複数の独立した EJB JAR ファイルが同じセキュリティロール名を使用する場合は、それぞれの割り当てを別個に行うことができる。

また、J2EE アプリケーションを開発する際、事前にすべてのユーザのロールについて把握すべきであるが、実際には、具体的なユーザが誰なのかはおそらく分からないことがある。このような事象は、J2EE セキュリティアーキテクチャの宣言によるセキュリティアーキテクチャで対処することができる。コンポーネントが配備されたあと、J2EE サーバの管理者またはデプロイヤがデフォルトレルムの J2EE ユーザもしくはグループにセキュリティロールをマップすることにより対処する。すなわち、セキュリティロールの設定を開発プロセスのもっと後の段階にまで延期できる。たとえば、アプリケーションのアセンブル担当者は、モジュールをアセンブルしてアプリケーションを作成する前に、モジュールレベルの配備記述子を操作できる。

グループは、プロファイルのような共通の特性によって分類されたユーザのカテゴリである。レルムは、同一のセキュリティポリシーが適用される保護領域である。例えば、J2EE SDK においては、2つのレルムが定義されている。

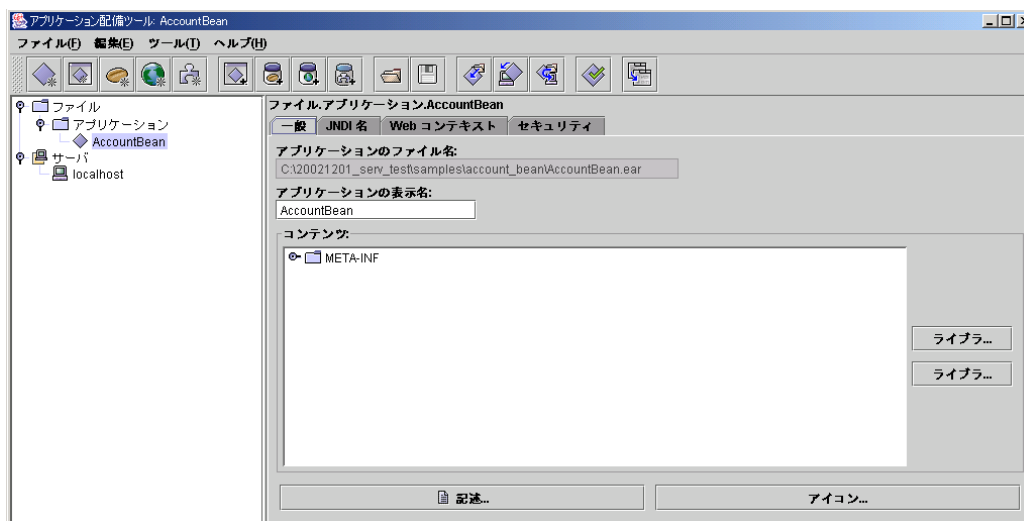
- デフォルトレルム
パスワードで識別されるユーザで構成される。
- 証明書レルム
X.509 デジタル証明書で識別されるユーザで構成される。証明書は、Web ブラウザクライアントの認証のみ使用される。

通常、J2EE SDK の `deploytool` を利用して定義したユーザは、デフォルトレルムに含まれる。レルムについては、「10.J2EE のユーザ、レルム、およびグループ」を参照すること。

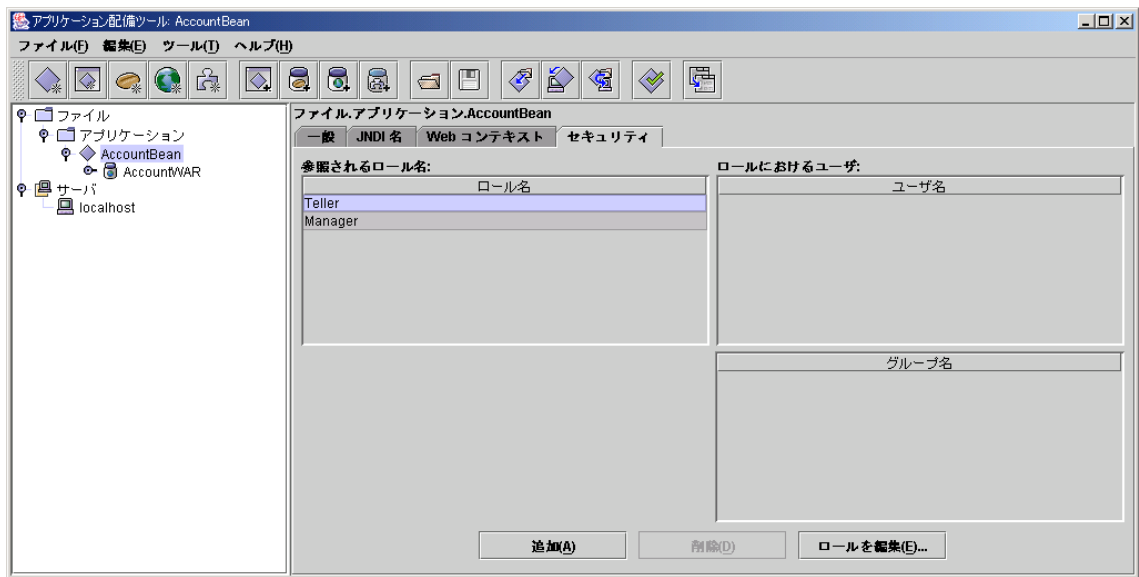
4.3.2. デフォルトレルムの J2EE ユーザまたはグループにセキュリティロールにマップする作業手順

この節では、Account Bean という Bean を例題として、管理者はユーザ Sally に Manager ロールを割り当て、ユーザの Bob、Ted、および Clara に Teller ロールを割り当てる。管理者は、J2EE SDK の `deploytool` を使用して次の手順で、J2EE ユーザとグループにロールをマップすることができる。

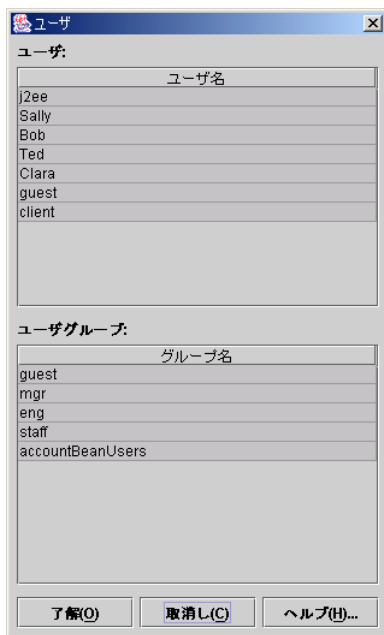
1. J2EE アプリケーションを選択する。



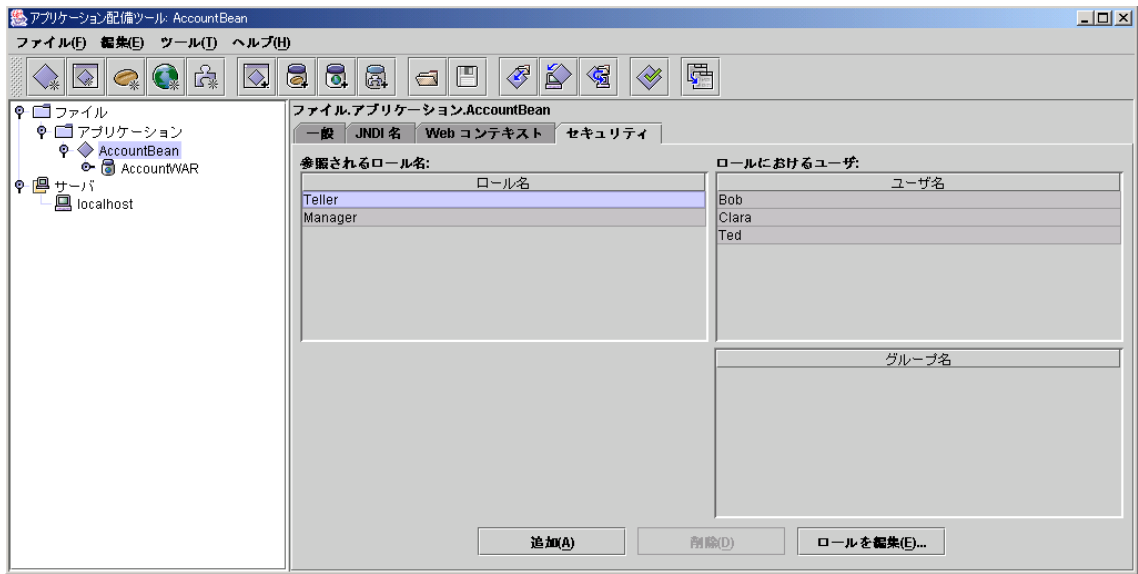
2. 「セキュリティ」タブで「ロール名」一覧から適切なロールを選択する。



3. 「追加」をクリックする。



4. 「ユーザ」ダイアログボックスで、ロールに所属する必要のあるユーザとグループを選択する。deploytool でユーザとグループを作成する方法については、「10. J2EE のユーザ、レルム、およびグループ」を参照すること。



この作業により、次の配備記述子内にデフォルトレルムの J2EE ユーザまたはグループにセキュリティロールへのマップが定義されている。ここでは、上の操作で定義された要素が生成されている。

```
<application>
  <display-name>AccountBean</display-name>
  <description>アプリケーションの記述</description>
  <module>
    <web>
      <web-uri>war-ic.war</web-uri>
      <context-root></context-root>
    </web>
  </module>
  <security-role>
    <role-name>Teller</role-name>
  </security-role>
  <security-role>
    <role-name>Manager</role-name>
  </security-role>
</application>
```

4.3.3. まとめ

ユーザすなわちプリンシパルをロールにマップする作業はアプリケーションデプロイヤが担当する。ユーザをロールにマップする正確な機構は、EJB の仕様には、少なくとも EJB2.0 では定義されていない。

セキュリティロールの設定は開発プロセスの後の段階にまで延期できることに注目すべきである。

4.3.4. 関連記事

- 3.1. J2EE セキュリティ
- 10. J2EE のユーザ、レルム、およびグループ

4.3.5. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社

- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第 2 版,
Rich Grenn 他 著、株式会社ピアソン・エデュケーション
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1_3-doc-tutorial-j.pdf
- J2EE アプリケーションのプログラミング
http://jp.sun.com/products/software/tools/jde/documentation/j2eeapps_ja.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

5. Web 層のセキュリティ

この章では、Web 層のセキュリティアーキテクチャと Web 層のセキュリティにおけるリソースの保護とユーザ認証について述べる。

5.1. Web 層のセキュリティアーキテクチャ

この節では、Web 層のセキュリティアーキテクチャについて述べる。

5.1.1. Web 層のセキュリティアーキテクチャ

J2EE の Web 層におけるセキュリティアーキテクチャは、EJB のセキュリティアーキテクチャと同じモデルが使用される。セキュリティは宣言によるセキュリティと Web ページにおいてプログラムによるセキュリティにより実装される。EJB セキュリティアーキテクチャ同様に宣言によるセキュリティを使用することが推奨される。ユーザ権限認証は、EJB のセキュリティと同じようにロールとプリンシパルまたはユーザを使って適用される。Web 層におけるセキュリティモデルの主要な概念は次のとおりである。

- シングルサインオン
クライアントは一度認証を受けるだけで、同一のレルム内のすべての Web ページにアクセスできる。セキュリティレルムは Web サーバによって定義され、各 Web アプリケーションが属するレルムはデプロイヤーが決定する。レルムごとに異なる認証メカニズムすなわち実質的には異なるユーザ名の集合を使うことができる。
- 複数のアプリケーションに適用される
認証されたクライアントが、アプリケーションごとにログインすることなく、さまざまな Web アプリケーションから Web ページを利用できるようにする必要がある。
- セッションに関連付けられる
ユーザ認証情報はサーブレットセッションに関連付けられていなければならない。それにより、各サーブレットまたは JSP は、プログラムによるユーザ権限認証が必要になったときユーザ認証情報にアクセスすることができる。

5.1.2. まとめ

J2EE の Web 層におけるセキュリティアーキテクチャは、EJB のセキュリティアーキテクチャと同じモデルが使用される。宣言によるセキュリティを使用することが推奨される。

5.1.3. 関連記事

- 6. EJB 層のセキュリティ
- 7. アプリケーションクライアント層のセキュリティ

5.1.4. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

5.2. Web 層のリソースの保護

セキュリティ制約の指定により、Web リソースすなわち Web コンポーネント、HTML ドキュメント、画像ファイル、圧縮されたアーカイブなどを保護することが可能である。

5.2.1. セキュリティ制約

セキュリティ制約は、誰が Web リソースコレクションへのアクセスを承認されているかを判断する。Web リソースコレクションとは、保護すべき一連のリソースを記述した URL パターンと HTTP メソッドの集合である。セキュリティ制約は、配備ツールを使用して簡単に定義することができる。配備ツールを使用したセキュリティ制約については、「5.3. Web リソースへのアクセス制御」で説明する。

5.2.2. Web リソースへのアクセス

クライアントは一般的に、J2EE アプリケーションのコンテナを使用して、Web 層または EJB 層内のエンタープライズリソースと通信を行う。Web 層または EJB 層内のリソースは、保護されている場合も保護されていない場合もある。保護されているリソースでは、匿名でないアイデンティティのある部分集合に対してアクセス制限する認証規則が配備記述子に定義されている。保護されたリソースにアクセスする場合、ユーザは匿名でないことを示す資格を提示し、アイデンティティをリソースの認証ポリシーに対して評価可能にする必要がある。

たとえば、認証されていないクライアントユーザが保護されている Web リソースにアクセスを試みると、Web コンテナは、Web コンテナによる認証のためのプロンプトをクライアントユーザに表示する。ユーザのアイデンティティが Web コンテナに証明され、リソースへのアクセス権が与えられているアイデンティティの 1 つであること示されるまで、クライアントユーザからのアクセス要求は、Web コンテナに受け入れられない。保護された Web リソースへの最初のアクセスで実行される呼び出し側の認証は、遅延認証と呼ばれる。

5.2.3. まとめ

Web 層で保護されている Web リソースは認証されたユーザしかアクセスできない。このアクセス制限は配備記述子で定義される。

5.2.4. 関連記事

- 6. EJB 層のセキュリティ
- 7. アプリケーションクライアント層のセキュリティ

5.2.5. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1_3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

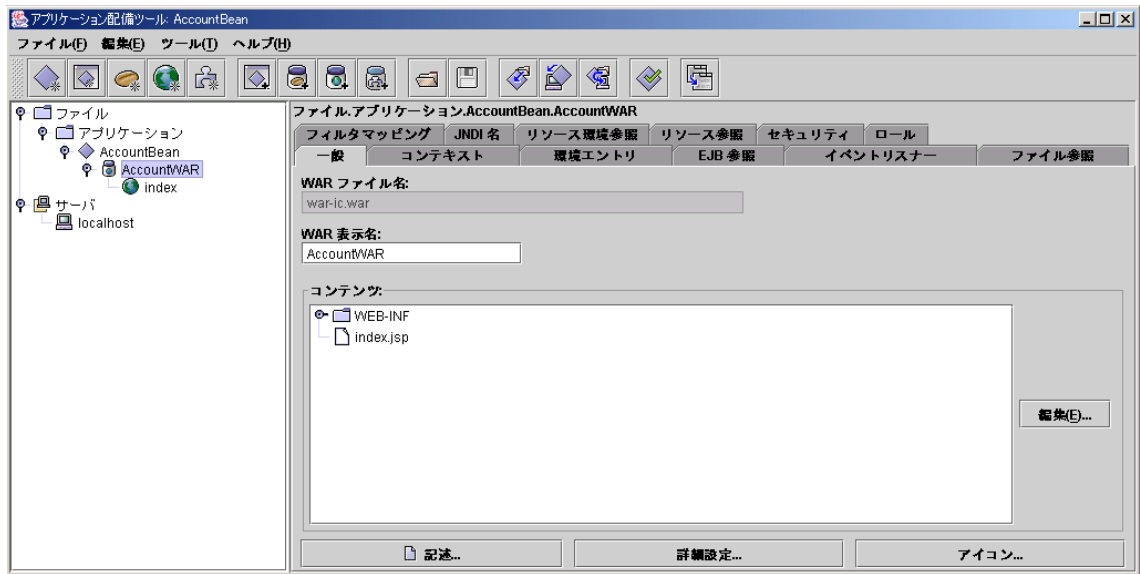
5.3. Web リソースへのアクセス制御

この節では実際に配備ツールを使用して Web リソースへのアクセス制御を設定する。

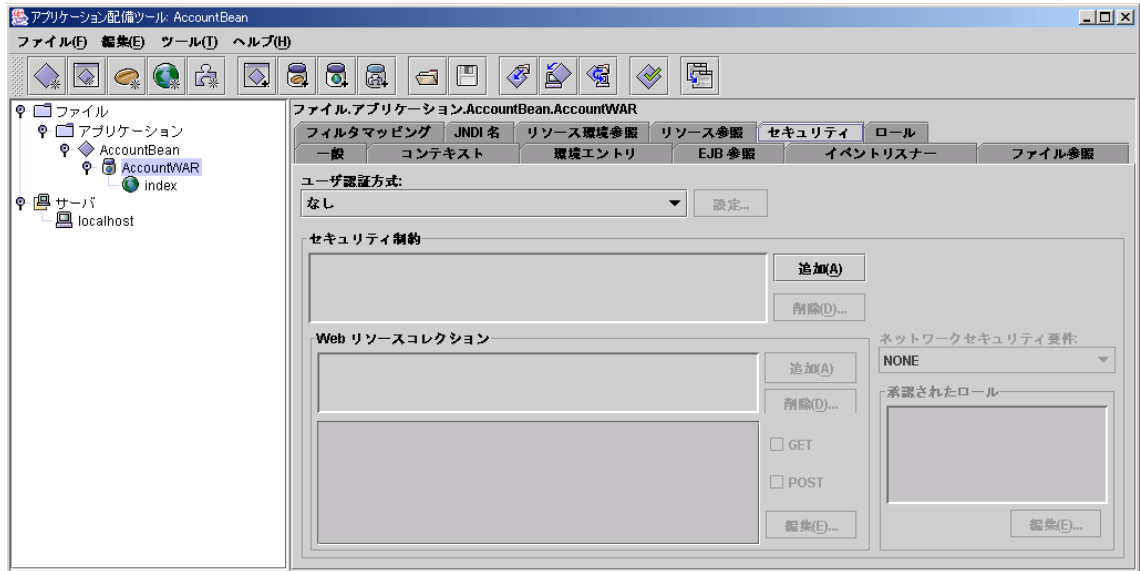
5.3.1. Web リソースへのアクセス制御の作業手順

保護される Web リソースを決定すれば、Web リソースへのアクセスを制御するためセキュリティ制約を指定するには、J2EE SDK の `deploytool` を使用し次の手順を行う。

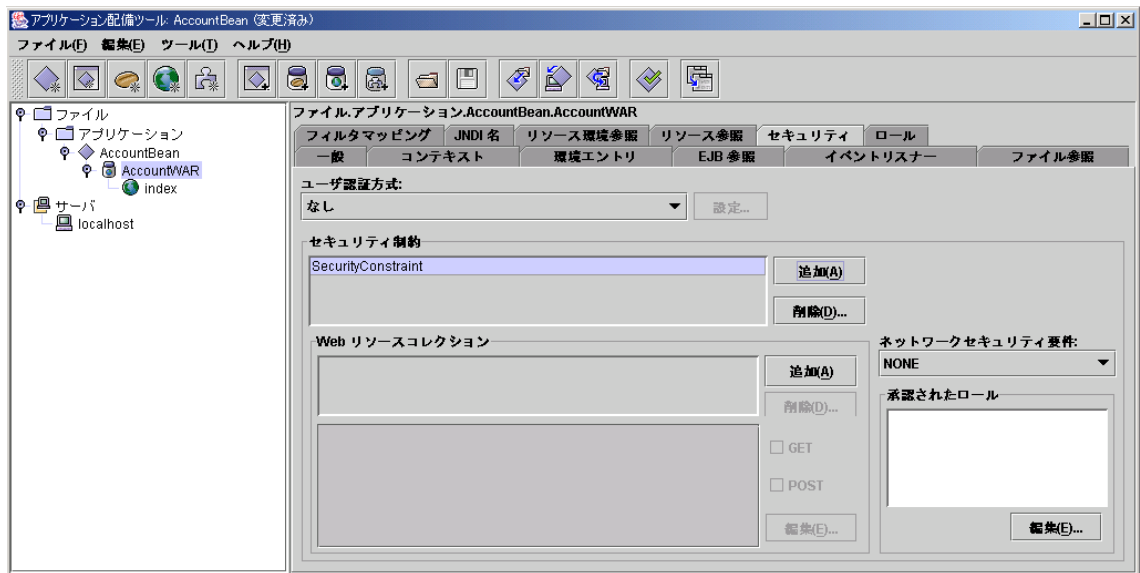
1. Web リソースを含む WAR ファイルを選択する。



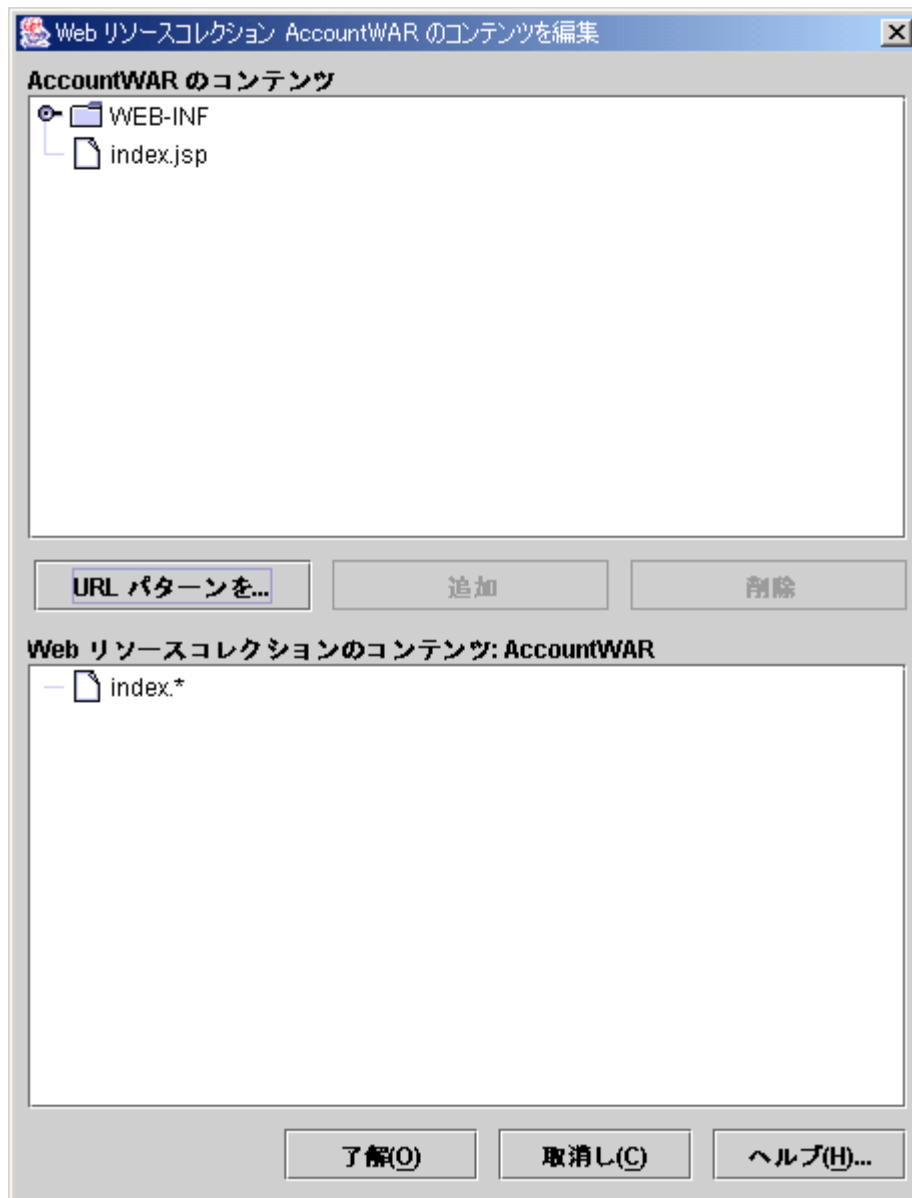
2. 「セキュリティ」タブを選択する。



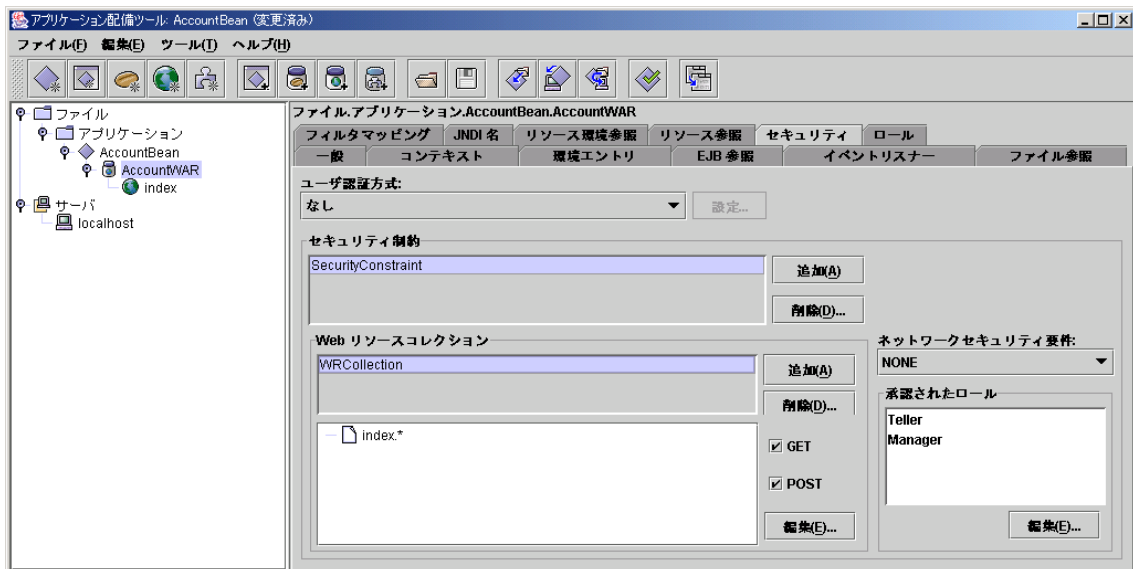
3. 画面の「セキュリティ制約」セクションの「追加」ボタンをクリックする。



4. 「Web リソースコレクション」フィールドでの横にある「追加」ボタンをクリックして、Web リソースコレクションをセキュリティ制約に追加する。Web リソースコレクションは、URL パターンと HTTP メソッドの対で保護される必要のあるリソースを参照する。



- 「承認されたロール」フィールドの「編集」ボタンをクリックして、セキュリティ制約にロールを追加する。これにより、Web リソースコレクションへのアクセスを許可するロールを指定する。



この一連の操作により、特定の Web リソースコレクションは、指定されたロールしかアクセスできなくなる。

アクセス制御のための新たなコードを作成することなしに、この作業すなわち宣言によるセキュリティを使用することで、次の配備記述子内に Web リソースへのアクセス制御が定義される。ここでは、Teller, Manager が指定された Web リソースがアクセス可能になる。

Web リソースへのアクセスを制御するために、アプリケーションコンポーネントプロバイダまたはアプリケーションアセンブラは、Web 配備記述子に<security-constraint>要素と<auth-constraint>サブ要素を指定する。また、保護される Web リソース<web-resource-collection>要素で定義される。次のコード例では、Web コンポーネント配備記述子の保護リソースの定義を行っている。

```
<web-app>
  <display-name>AccountWAR</display-name>
  <servlet>
    <servlet-name>index</servlet-name>
    <display-name>index</display-name>
    <jsp-file>/index.jsp</jsp-file>
  </servlet>
  <security-constraint>
    <web-resource-collection>
      <web-resource-name>WRCollection</web-resource-name>
      <url-pattern>/index.*</url-pattern>
      <http-method>POST</http-method>
      <http-method>GET</http-method>
    </web-resource-collection>
    <auth-constraint>
      <role-name>Teller</role-name>
      <role-name>Manager</role-name>
    </auth-constraint>
  </security-constraint>
  <security-role>
    <role-name>Manager</role-name>
  </security-role>
  <security-role>
    <role-name>Teller</role-name>
  </security-role>
</web-app>
```

5.3.2. まとめ

配備ツールを使用して簡単に、宣言によるセキュリティによりコードを新たに実装することなく、セキュリティ制約の指定を行い、Web リソースへのアクセスを制限を行い保護することができる。

5.3.3. 関連記事

- 6.1.1. EJB 層のセキュリティ
- 6.2.3. メソッドアクセス権の宣言

5.3.4. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

5.4. Web リソースのユーザ認証

ユーザが保護された Web リソースへアクセスを試みると、Web コンテナは Web リソース用に構成された認証機構を起動する。その認証機構について述べる。

5.4.1. 宣言によるセキュリティによる Web リソースのユーザ認証機構

認証機構は、Web コンポーネント配備記述子の `login-config` 要素を使用して、HTTP 基本認証機構、フォームベース認証機構、クライアント証明書認証を行うことができる。認証のためのプログラムを作成することなしに、Web コンポーネント配備記述子における宣言だけで、J2EE プラットフォームのこれら認証機構が使用できる。これは、Web コンポーネントを設計者および実装者は Web コンポーネントに認証機構を予め実装しなくも、後の配備時移行に、J2EE プラットフォームの運用環境に最適な認証機構を Web コンポーネント配備記述子で宣言することにより使用できることに注目すべきである。

ユーザが保護された Web リソースへアクセスを試みると、Web コンテナは Web リソー

ス用に構成された認証機構を起動する。次のような、Web リソースの認証機構を構成することが可能である。

- HTTP 基本認証
- フォームベース認証
- クライアント証明書認証

認証機構は、Web コンポーネント配備記述子の `login-config` 要素を使用して構成される。これらの認証機構そのものを実装することなく、Web コンポーネント配備記述子の宣言により使用できる。従って、認証機構そのものを実装を行うことによるプログラムにおけるセキュリティ上の脆弱性をなくすることができる。

J2EE Web コンテナにおける HTTP 基本認証機構の宣言の構成は次の通り。

```
<web-app>
  <login-config>
    <auth-method>BASIC</auth-method>
    <realm-name>sample</realm-name>
  </login-config>
</web-app>
```

J2EE Web コンテナにおけるフォームベース認証機構の宣言の構成は次の通り。

```
<web-app>
  <login-config>
    <auth-method>FORM</auth-method>
    <form-login-config>
      <form-login-page>login.jsp</form-login-page>
      <form-error-page>error.jsp</form-error-page>
    </form-login-config>
  </login-config>
</web-app>
```

J2EE Web コンテナにおけるクライアント証明書認証の宣言の構成は次の通り。

```
<web-app>
  <login-config>
    <auth-method>CLIENT-CERT</auth-method>
  </login-config>
</web-app>
```

各認証機構は、「5.5. HTTP 基本認証」、「5.6. フォームベース認証」、「5.7.クライアント証明書認証」において詳細に述べる。

5.4.2. まとめ

Web リソースのユーザ認証のためのプログラムを新たに作成することなしに、Web コンポーネント配備記述子による宣言すなわち宣言によるセキュリティだけで、J2EE プラットフォームの認証機構が使用できる。

5.4.3. 関連文献

- 5.5. HTTP 基本認証
- 5.6. フォームベース認証
- 5.7.クライアント証明書認証

5.4.4. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド

<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>

- Sun ONE Application Server 7 Web アプリケーション開発者ガイド

<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

5.5. HTTP 基本認証

最も基本的な認証機構である、HTTP 基本認証について述べる。

5.5.1. HTTP 基本認証

HTTP 基本認証を指定した場合、Web サーバは Web クライアントから取得したユーザ名とパスワードを使用してプリンシパルを認証する。最も基本的な認証である。この認証を使用すると、Web クライアントのブラウザから認証のための入力を促すダイアログボックスが表示されて、ユーザ名とパスワードの入力を促される。この認証において認証が失敗した場合、認証のための入力を促すダイアログボックスが再度、表示される。また、HTTP 基本認証では、サーバの認証を行わないことを注意する。

5.5.2. HTTP 基本認証においては認証情報は保護されない

HTTP 基本認証では、認証のための入力を促すダイアログボックスから入力されたユーザ名とパスワードは、base64 エンコード方式でサーバに送られる。従って、第3者が Web クライアントとサーバのネットワークを盗聴することで、容易に、base64 でエンコードされたユーザ名とパスワードをデコードして入手することができる。ユーザ名とパスワードを保護したいのならば、クライアント証明書認証または、SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張を使用すべきである。

5.5.3. HTTP 基本認証のユーザインタフェース

HTTP 基本認証では、認証のための入力を促すダイアログボックスは、Web クライアントの OS やブラウザに依存する。そのダイアログボックスは、挙動は Web クライアントの OS やブラウザに依存する。認証が失敗した場合、ダイアログボックス再度、表示されるだけなので、エラー処理に関しては、Web クライアントの OS やブラウザに依存することになる。

5.5.4. まとめ

HTTP 基本認証は最も基本的な認証機構である。ユーザ名やパスワードは平文で送信されることに注意する。

5.5.5. 関連記事

- 5.4.1. 宣言によるセキュリティによる Web リソースのユーザ認証
- 5.6.1. フォームベース認証
- 5.7.1. クライアント証明書認証

5.5.6. 参考文献

- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-13-doc-tutorial-j.pdf
- Enterprise JavaBeans Specification, Version2.0
ftp://ftp.java.sun.com/pub/ejb/947q9tbb/ejb-2_0-fr2-spec.pdf
- セキュリティ管理者ガイド Sun™ONE Application ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版,
Rich Grenn 他 著, 株式会社ピアソン・エデュケーション

5.6. フォームベース認証

サーブレット仕様で規定されている認証形式について述べる。HTTP 基本認証との違いについても述べる。

5.6.1. フォームベース認証

フォームベース認証を指定した場合、HTTP ブラウザに従ってユーザへ表示される、口

ログイン画面とエラーページをカスタマイズすることが可能である。フォームベース認証は、基本認証より、Web クライアントの OS やブラウザに依存することなく多種多様なページのカスタマイズが可能である。たとえば、ログイン画面やエラーページなどである。

5.6.2. フォームベース認証においては認証情報は保護されない

フォームベース認証も HTTP 基本認証も、クライアントで入力された認証に関するデータは、特別にセキュリティ保護されていない。フォームベース認証では、ユーザのダイアログボックスの内容は平文で送信され、受信側のサーバも認証されない。フォームベース認証は、ユーザ名とパスワードをインターネット経由でテキストとして送信する。このテキストは、uuencode 形式に符号化されているが、暗号化はされていない。base64 エンコーディングを使用する基本認証では、第3者が送信内容を盗聴を行い、ユーザ名とパスワードは簡単に復号化することができる。従って、セキュリティ上の脆弱な点となる。すべての接続が SSL(Secure Socket Layer) で接続されていないと、ユーザ名とパスワードを盗聴されることになる。

5.6.3. まとめ

フォームベース認証は基本的な認証機構である。認証に使用する独自のログインページとエラーページが定義できる。ユーザ名やパスワードは平文で送信されることに注意する。

5.6.4. 関連記事

- 5.4.1. 宣言によるセキュリティによる Web リソースのユーザ認証
- 5.5.1. HTTP 基本認証
- 5.7.1. クライアント証明書認証

5.6.5. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー

- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

5.7. クライアント証明書認証

クライアント証明書認証は、基本認証やフォームベース認証に比べ、より安全性の高い方法である。この認証では、HTTPS(HTTP over SSL)を使用する。

5.7.1. クライアント証明書認証

X.509 デジタル証明書と HTTPS(HTTP over SSL)を使用する認証機構である。ユーザが本当に本人かどうかの検証に使用できるデジタル証明書すなわち X.509 デジタル証明書を使用することで前出の認証機構より高い安全性を備えている。クライアント証明書認証は、SSL を実装されている Web ブラウザを使用することになる。

5.7.2. HTTPS

HTTPS では、サーバや任意のクライアントが互いに公開鍵システムを使用しサーバとクライアントの相互認証、サーバ認証、クライアント認証を行う。SSL(Secure Sockets Layer)は、データの暗号化、サーバ認証、メッセージの完全性、オプションでの TCP/IP 接続のクライアントの認証を提供する。公開鍵証明書は、実世界での印鑑登録証明書に相当すると考えられる。公開鍵証明書は、認証局(Certification Authority:CA)と呼ばれる信頼性のある組織により発行されて、その保有者に ID を含む X.509 デジタル証明書を与える。クライアント証明書認証を指定した場合、Web サーバは、クライアントの X.509 デジタル証明書を使用して、クライアントを認証する。また、クライアントは同様に、サーバの X.509 デジタル証明書を使用してサーバの認証をおこなう。すなわち、クライアントとサーバの相互認証を行うことである。

5.7.3. X.509 デジタル証明書

X.509 デジタル 証明書とは、X.509 デジタル公開鍵インフラストラクチャ(Public Key Infrastructure: PKI)で定義された標準に準拠する公開鍵証明書のことである。

クライアント証明書認証の場合、クライアントに X.509 デジタル証明書をインストールしなければならない。クライアントユーザは X.509 デジタル証明書を自らの責任で管理運用する必要がある。さらに、サーバおよびクライアント証明書すべてを発行、管理する認証局(Certification Authority:CA)や登録局(Registration Authority)を稼働させ運用管理を行い、X.509 デジタル証明書を配布する必要がある。

5.7.4. クライアントにおける X.509 デジタル証明書の運用管理コスト

クライアント証明書認証の場合、これまで述べてきたとおり、クライアントユーザが X.509 証明書の管理を行わなければならない。この運用管理コストは非常に高いことがある。すなわち、一般的なレベルのユーザにとっては X.509 デジタル証明書の運用管理が困難になる。従って、近年、クライアント証明書認証よりも「5.8.4.SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張」を使い、アプリケーションレベルである暗号システムたとえば乱数表を用いて認証を行うシステムが多くなってきている。

5.7.5. まとめ

クライアント証明書認証は、非常に安全な認証手段である。その反面、X.509 デジタル証明書の運用管理などで、他の認証手段より運用管理コストがかかる。

5.7.6. 関連記事

- 5.4.1. 宣言によるセキュリティによる Web リソースのユーザ認証
- 5.5.1. HTTP 基本認証
- 5.6.1. フォームベース認証

5.7.7. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション

- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- PKI 公開鍵基盤、トム・オースティン著、日経 B P
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

5.8. SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張

5.8.1. SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張

HTTP 基本認証、およびフォームベース認証の両方では、アカウントとパスワードは機密保護機構がないため認証情報は保護されない。認証情報が保護されていないことによる脆弱性は非常に危険である。従って、SSL によって保護されているセッション上で、これらの認証プロトコルを実行することによりアカウントとパスワードなどの情報は保護される。このようなセッションは、クライアント認証データなどを含むメッセージ内容のすべてが機密保持されて保護されることが保証される。

5.8.2. SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張における利点と欠点

SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張は、実際に最もよく使われる認証機構である。理由は、「5.7.4. クライアントにおける X.509 デジタル証明書の運用管理コスト」で述べたように、クライアントユーザの X.509 デジタルの証明書の管理運用が面倒だということである。SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張は、その管理運用コストがかからない。ただし、サーバ側からみるとクライアントユーザの認証に関しては、クライアント証明書認証よりセキュリティ上の強度は低い。従って、強度を上げるためにクライアントユーザがアプリケーションレベルで

暗号システムたとえば乱数表などを使用し、さらなる認証を行うことがある。

5.8.3. SSL 使用によるシステムへの負荷とその対策

SSL を使用することで、HTTP 基本認証、およびフォームベース認証のみを使用した場合よりサーバシステムとクライアントシステムに負荷がかかる。特に多数のクライアントシステムにサーバシステムが対応しなければならない時、サーバシステムにパフォーマンス的な余裕が必要である。対策としては SSL アクセラレータのようなハードウェアをクライアントシステムとサーバシステムとのネットワークに挿入して、SSL アクセラレータに SSL の処理を任せるといことが考えられる。

5.8.4. SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張

SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張では SSL を使用するがクライアント認証を行わない。次のコード例では、`transport-guarantee` 要素を使用して、SSL 上で HTTP 基本認証を構成する方法を示す。

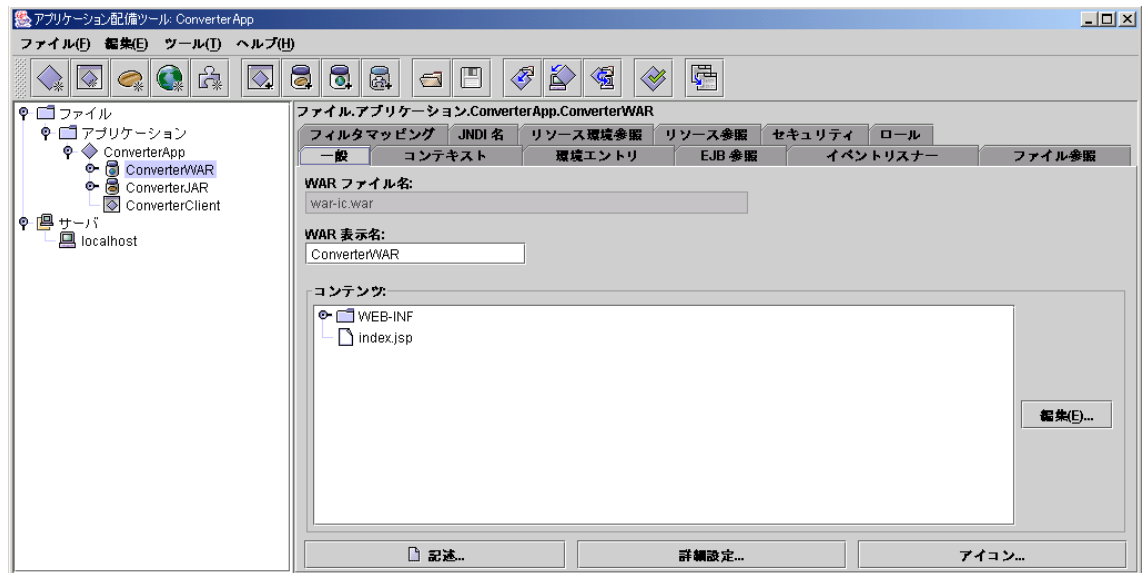
```
<web-app>
  <security-constraint>
    ...
    <user-data-constraint>
      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
  </security-constraint>
</web-app>
```

SSL 上のフォームベース認証も同様に構成される。

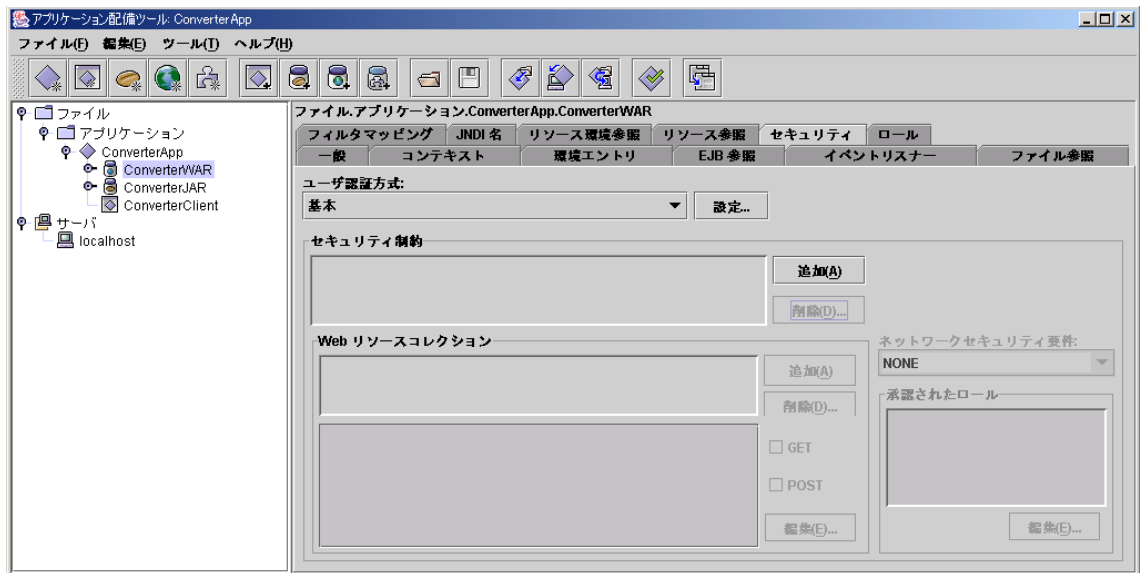
5.8.5. SSL 使用の HTTP 基本認証とフォームベース認証の設定

SSL を介した HTTP 基本認証またはフォームベース認証を J2EE SDK の `deploytool` を使用して次の手順により設定する。

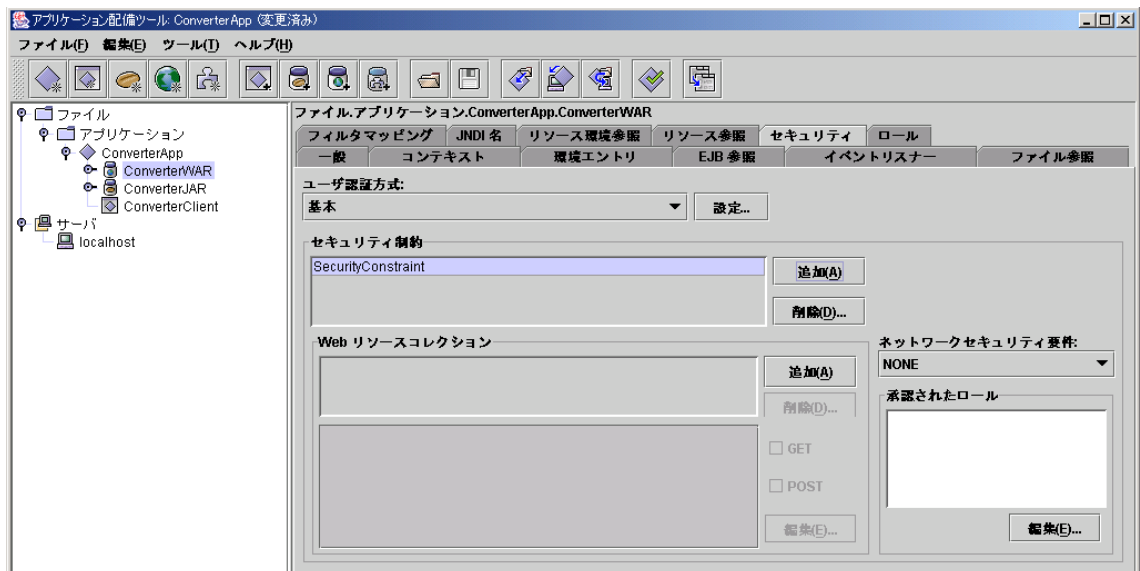
1. Web コンポーネントを選択する。Web コンポーネントインスペクタが表示される。



- 「セキュリティ」タブで、「ユーザ認証方式」プルダウンメニューに「基本」または「フォームベース」の認証が選択されていることを確認する。

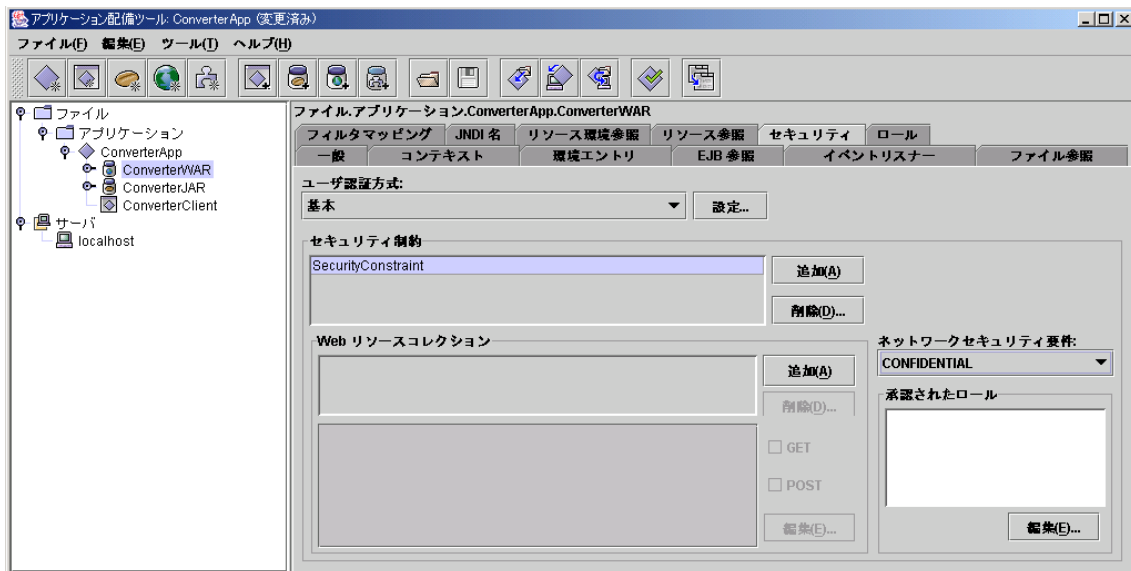


- 「セキュリティ制約」セクションで「追加」ボタンをクリックする。



- 追加されたセキュリティ制約をクリックする。

- 「ネットワークセキュリティ要件」プルダウンメニューから「CONFIDENTIAL」を選択する。



この設定作業により、次のコード例では、`transport-guarantee` 要素を使用して、SSL 上で HTTP 基本認証を構成する方法を示す。

```

<web-app>
  <security-constraint>
    ...
    <user-data-constraint>
      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
    ...
  </security-constraint>
</web-app>

```

SSL 上のフォームベース認証も同様に構成される。

5.8.6. まとめ

SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張により、認証情報が暗号化されて機密が保護される。また、この SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張された認証機構が最も使用される。

5.8.7. 関連記事

- 5.2.2. セキュリティ制約
- 5.3.1. Web リソースへのアクセス制御
- 5.4.1. 宣言によるセキュリティによる Web リソースのユーザ認証
- 5.5.1. HTTP 基本認証
- 5.6.1. フォームベース認証
- 5.7.1. クライアント証明書認証

5.8.8. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版,
Rich Grenn 他 著、株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1_3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

5.9. J2EE SDK の deploytool による Web リソースの認証機構の設定

この節では、J2EE SDK に付属している GUI ベースの配備ツールを使用して Web リソースの認証機構の設定する。

5.9.1. J2EE SDK の deploytool 使用による Web リソースの認証機構の設定の利点

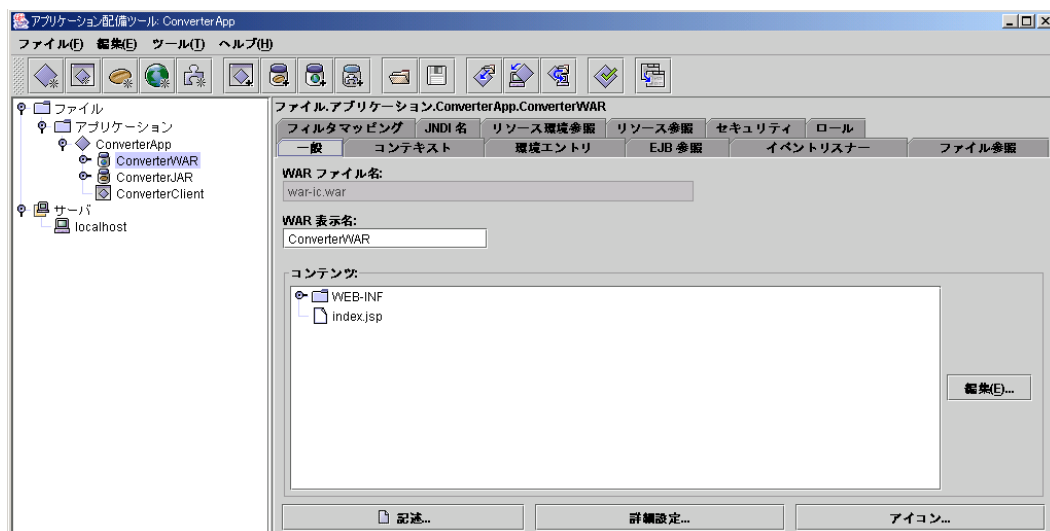
J2EE SDK に付属している GUI ベースの配備ツールである **deploytool** を使用して Web リソースの認証機構すなわち基本認証、フォームベース認証、クライアント証明書認証を設定する。**deploytool** を使用して設定するだけで各認証機構を機能させることができる。すなわち、宣言によるセキュリティにより各認証機構そのものの実装を行うことなしに、各認証機構が使用可能となる。

各環境で使用されるアプリケーションサーバにも、GUI ベースの配備ツールがあり、それを使用して同様のことができる。もちろん、該当する配備記述子を直接編集して認証機構の設定を行ってもよい。その場合、XML 形式のテキストを編集することになるので十分に注意をする必要がある。

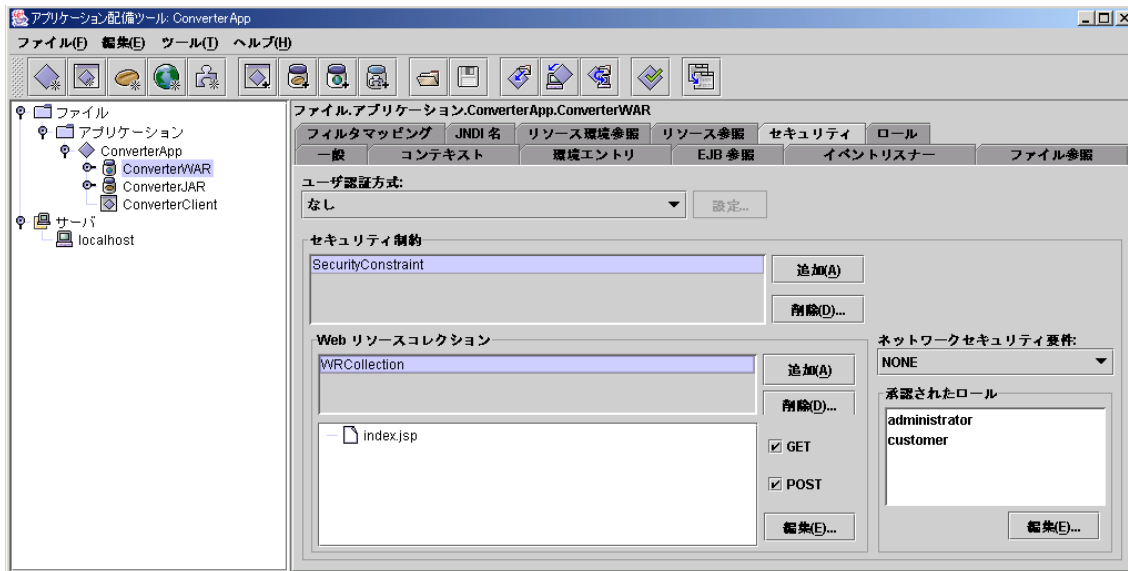
5.9.2. J2EE SDK の deploytool 使用による Web リソースの認証機構の設定例

ここで、Web リソースの認証機構の設定を行う。WAR ファイル内の Web リソースが使用する認証機構は、次の手順により設定する。

1. Web リソースを含む WAR ファイルを選択する。



2. 「セキュリティ」タブを選択する。

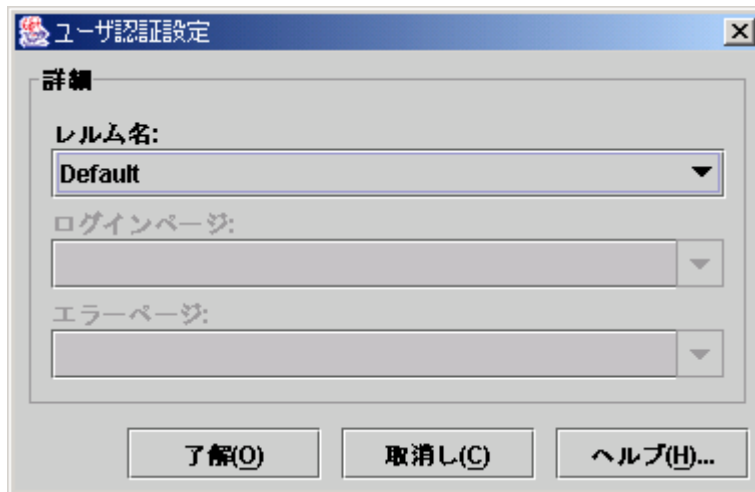


3. 「ユーザ認証方式」をプルダウンメニューより、次の認証機構のいずれかを選択する。

- なし
 - 基本認証
 - クライアント証明書認証
 - フォームベース認証
- a. フォームベース認証を選択した場合、「設定」を選択し、「ユーザ認証設定」ダイアログの「レルム名」, 「ログインページ」, および「エラーページ」の各フィールドに入力する必要がある。「エラーページ」はログインの失敗時に表示される内容である。



- b. 基本認証を選択した場合、「設定」を選択し、「ユーザ認証設定」ダイアログの「レルム名」フィールドで Default を選択する。



5.9.3. まとめ

J2EE SDK に付属している GUI ベースの配備ツールである `deploytool` を使用して Web リソースの認証機構すなわち基本認証、フォームベース認証、クライアント証明書認証を設定するだけで各認証機構を機能させることができる。各認証機構そのものの実装を行うことなしに、各認証機構が使用可能となる。

5.9.4. 関連記事

- 5.2.2.Web リソース
- 5.3.1.Web リソースへのアクセス制御
- 5.4.1.宣言によるセキュリティによる Web リソースのユーザ認証
- 5.5.1. HTTP 基本認証
- 5.6.1.フォームベース認証
- 5.7.1. クライアント証明書認証

5.9.5. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション

- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版,
Rich Grenn 他 著、株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1_3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

5.10. Web 層のプログラムによるセキュリティの使用

プログラムによるセキュリティは、宣言によるセキュリティだけでは Web アプリケーションのセキュリティモデルを十分に表せない時に、より複雑かつ高度な安全性を必要とする Web アプリケーションによって使用されることがある。

5.10.1. Web 層のプログラムによるセキュリティの必要性

プログラムによるセキュリティは、宣言によるセキュリティだけでは、Web アプリケーションが要求するセキュリティモデルを表現することができない複雑かつ高度なセキュリティを実装する場合に使用する。プログラムによるセキュリティにより、動的なセキュリティポリシーを実装することができる。例えば、Web クライアントに対してある特定のデータに対してアクセスを許すような場合を仮定する。この場合、Web アプリケーションがそのリクエストを行ったユーザを認識し、そのユーザに対して要求しているデータへのアクセスを承認するかしないかを決定する必要がある。例えば、ユーザ Alice は自分のアカウントデータへのアクセスは許されても、Bob のデータへのアクセスが許されてはならない。プログラムによるセキュリティは、ランタイムにおいて適切なルールを決定しなければならないようなルールやセキュリティ情報とデータが密接に関連している場合において有効なセキュリティモデルである。

5.10.2. プログラムによるセキュリティで 사용되는 API

セキュリティに対応する Web アプリケーションは、プログラムによるセキュリティを使用する際、`HttpServletRequest` インタフェースの次のメソッドを使用して、認証されたクライアントのセキュリティ情報にアクセスする。

- **`public String HttpServletRequest.getRemoteUser()`**
HTTP サーブレットの要求を作成したユーザが認証されている場合はそのログイン名すなわちプリンシパル名を返す。認証されていない場合は `null` を返す。ユーザ名が以後の各要求とともに送信されるかどうかは、ブラウザと認証のタイプによって異なる。返される値は、CGI 変数 `REMOTE_USER` の値と同じ。
- **`public boolean HttpServletRequest.isUserInRole(String role)`**
認証されたユーザが指定された論理的な「ロール」に含まれているかどうかを示す論理値を返す。ロールおよびロールのメンバーシップは、配備記述子を使用して定義できる。パラメータとして渡されたロールにクライアントが属している場合、`true` を返す。ユーザが認証されていなかった場合、このメソッドは `false` を返す。
- **`public java.security.Principal HttpServletRequest.getUserPrincipal()`**
現在の認証済みユーザの名前が格納された `java.security.Principal` オブジェクトを返す。ユーザが認証されていなかった場合、このメソッドは `null` を返す。`Principal` オブジェクトは、要求を発行したユーザに属するアイデンティティを表すためのものであり、ログイン名から X.509 デジタル証明書まで含まれる。
- **`public String HttpServletRequest.getAuthType()`**
サーブレットの保護に使用される認証タイプを返す。戻り値としては、基本認証を示す `BASIC`、SSL ベースの認証を示す `SSL`、それ以外を示す `null` がある。

クライアントが認証したユーザ名すなわち要求を出したユーザ名の決定には、`getRemoteUser` メソッドを使用できる。ユーザが特定のセキュリティロールを持つかどうかの確認は、`isUserInRole` を使用して行う。`getUserPrincipal` メソッドは、`java.security.Principal` オブジェクトを返す。

これらの API により、Servlet はリモートユーザの論理ロールに基づいたビジネスロジックを決定できるようになる。またこれらの API を使用することにより、Servlet は現在の

ユーザの主体名すなわちプリンシパル名を決定することができる。

実際の Web コンテナでは、これらのメソッドが実装されている場合もあれば、されていない場合もあるという点に、注意しておくほうがよい。これらのメソッドの目的は、より高度な認証方式を実装可能にすることにある。これらのメソッドは、単に基底にある Web コンテナの実装によって集められているかもしれない情報へのアクセスを提供するにすぎないからである。

5.10.3. プログラムによるセキュリティ設定は最小限にする

プログラミングにおいてセキュリティをきめ細かに設定することはできる。しかし、プログラムやコンポーネントを変更することなく、コンポーネントの配備時に、事後的にセキュリティの設定が可能となっている宣言によるセキュリティをできる限り活用すべきである。プログラムによるセキュリティを設定する通常の手続き的な手法に対して、宣言によるセキュリティにおいて、配備時の宣言を通じてコンポーネントにセキュリティ属性を設定する宣言的手法は、J2EE のすべての層で一貫して適用されていて、J2EE のセキュリティの大きな特徴を形成している。もちろん、Web アプリケーションのセキュリティ要件によっては、宣言によるセキュリティとプログラムによるセキュリティを適切に組み合わせて使用することも可能である。

5.10.4. プログラムによるセキュリティのサンプルコード

データベースにアクセスするサーブレットプログラムを考える。このサーブレットでは、Administrator グループと Administrator ロールに属するユーザに対してのみデータの更新を承認するものとする。データの更新が可能か否かを確認するために、次のメソッドをサーブレットプログラムに追加する。

```
public void onlyAdministrators(){
    out.println("onlyAdministrators was called by: " + request.getUserPrincipal());

    if (request.isUserInRole("Administrator")) {
        out.println("Only admins are allowed in this method.");
        //特権を使った処理をここで行う
    } else {
        out.println("User: " + request.getUserPrincipal() + " was not in the
            Administrator role.");
    }
}
```

5.10.5. まとめ

プログラムによるセキュリティは、宣言によるセキュリティだけではアプリケーションのセキュリティ要件を十分に満たせない時に、さらに複雑かつ高度な安全性を必要とするアプリケーションによって使用される。ただし、最小限になるように設計実装をおこなう。

5.10.6. 関連記事

- 4.2.6. セキュリティロールと最小特権
 - 6.1.3. プログラムによるセキュリティの利用とその利点と欠点

5.10.7. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-13-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

5.11. 保護されていない Web リソース

多くのアプリケーションでは、Web コンポーネントは保護されておらず、どの呼び出し

でも認証なしでアクセスすることができる Web リソースについて述べる。

5.11.1. 保護されていない Web リソース

多くのアプリケーションでは、Web コンポーネントは保護されておらず、デフォルトでは、どの呼び出しでも認証なしでアクセスすることができる Web リソースが多数存在する。Web 層では、単に認証機構を設定しないことによりクライアントからの無制限のアクセスを提供する。

5.11.2. 保護されていない Web リソースにアクセスするためのロールの危険性

すべてのユーザには匿名のロールが割り当てられている。デフォルトでは、匿名のロール値は ANYONE であり、配備記述子で設定できる。ANYONE は認証を必要とされない。この匿名の ANYONE により保護されていない Web リソースにアクセスすることになる。

従って、このようなデフォルトの状態はセキュリティ上非常に危険である。適切な設定によりデフォルトの状態で使用しないようにすべきである。

5.11.3. まとめ

保護される Web リソースを明確に定義し、これまで述べてきた手法で保護することが重要である。適切な設定によりデフォルトの状態で使用しないようにすべきである。

5.11.4. 関連記事

- 6.4. 保護されていない EJB 層のリソース

5.11.5. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社

- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1_3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

6. EJB 層のセキュリティ

この章では、EJB 層のリソースを保護するために使用する、宣言によるセキュリティ機構とプログラムによるセキュリティ機構について述べる。

6.1. EJB 層のセキュリティ

保護されるリソースには、アプリケーションのクライアントから呼び出される EJB のメソッド、Web コンポーネント、およびその他の EJB が含まれる。それらリソースの保護について述べる。

6.1.1. EJB 層のセキュリティの設定

EJB 層のリソースを保護するには、次の機構を使用する。

- メソッドアクセス権の宣言
- J2EE のユーザとグループへのロールマッピング

これらの機構の使用するには、次の 2 つの手法をすることにより、EJB 層のリソースを保護しセキュリティを設定することができる。

- 宣言によるセキュリティ
- プログラムによるセキュリティ

一般的には、宣言によるセキュリティでセキュリティ確保することが推奨される。EJB リソースの宣言によるセキュリティでセキュリティを確保することを推奨する。しかし、よりきめ細かな複雑なセキュリティを確保したい場合、プログラムによりセキュリティを併用することがある。また、宣言によるセキュリティを可能な限り使用し、宣言によるセキュリティで実現できないセキュリティ要件を、プログラムによるセキュリティにより補完する。

6.1.2. 宣言によるセキュリティの利用とその利点と欠点

J2EE アプリケーションで定義されたロールに基づくユーザ権限認証を定義するために、宣言によるセキュリティを推奨する。EJB の各メソッドを、すべてのプリンシパルまたは

特定のロールのリストに対して、プログラムのハードコードを行わずにユーザ権限認証を行うことができる。この宣言によるセキュリティにより、EJB 層のコードと Web 層のコードの開発が実行時の認証方式と分離されます。また、宣言によるセキュリティは、Bean 開発者とアプリケーションアセンブラおよびデプロイアの役割の分離を促進する。これは、宣言によるセキュリティを利用した EJB の独立性も高める。

一方、EJB セキュリティモデルは、その単純さを非難されることがある。例えば、EJB のインスタンスすなわちクラスだけに基づいてセキュリティを設定することはできない。このため、EJB 開発者は、枯れた実績のあるアプローチである Web レベルで提供されるセキュリティに頼る傾向がある。詳細はに関しては、「6.5.1.Web コンテナの認証機構の」を参照すること。

6.1.3. プログラムによるセキュリティの利用とその利点と欠点

理想的なセキュリティの実現は、宣言によるセキュリティだけを使用すべきである。宣言によるセキュリティではアプリケーションのセキュリティ要件を満たせない状況がある。その際には、プログラムによるセキュリティにより EJB クラスにそのセキュリティ要件を実装しなければならない。しかし、プログラムによるセキュリティを使用することにより EJB クラスの移植性が低くなり、アプリケーションアセンブラがさまざまなソースからの Bean を結合する方法が制限されてしまう。

さらに詳細に述べると、セキュリティポリシーは、できる限り宣言によるセキュリティすなわち配備記述子によって指定されなければならない。そして、コンテナは、実行時にセキュリティポリシーを管理しなければならない。ただし、EJB 開発者が実行時にプログラムによるセキュリティを利用してプログラムをハードコードすることによりセキュリティコンテキストにアクセスしなければならない場合がある。その場合には、EJBContext オブジェクト内の適切なメソッドを呼び出せば、それが可能である。セキュリティ上の特殊な必要条件があるアプリケーションでは、このようにする必要がある。たとえば、あるアプリケーションでは、時間帯、呼び出し側の物理的な位置、呼び出しのパラメータ、EJB の内部状態に基づいて承認の判断を行い、別のアプリケーションでは、データベースに格納されたユーザ情報に基づいてアクセスを制限することがある。

6.1.4. まとめ

一般的には、宣言によるセキュリティでセキュリティ確保することを推奨する。しかし、宣言によるセキュリティでセキュリティ要件を満たせない場合はプログラムによるセキュリティも利用する。

6.1.5. 関連記事

- 5. Web 層のセキュリティ
- 7. アプリケーションクライアント層のセキュリティ

6.1.6. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

6.2. EJB のメソッドのアクセス権の宣言

J2EE SDK に付属している `deploytool` を使用して EJB のメソッドのアクセス権を設定について述べる。

6.2.1. EJB のメソッドのアクセス権の宣言

ルールを定義したあと、EJB のメソッドアクセス権を定義することができる。メソッドアクセス権は各ルールが呼び出せるメソッドを示す。メソッドのアクセス権は、セキュリ

ティロールから EJB のホームインフェースとコンポーネントインフェースされにこれらのスーパーインタフェースを含むインタフェースのメソッドの二項関係として、`<method-permission>`要素を使って定義する。この要素には、任意の数のセキュリティロールのリストと任意の数のメソッドのリストを含むことができる。リストの各セキュリティロールは、`<role-name>`要素によって識別され、メソッドは、`<method-name>`要素で定義される。`<method-permission>`要素には、オプションの`<description>`要素を含むことができる。1つのセキュリティロールまたはメソッドを、複数の`<method-permission>`要素に指定することができる。

6.2.2. メソッドアクセス権でのメソッドの指定方法

`<method-permission>`要素では、次の3つの主な方法でメソッドを指定できる。

- Bean クラスのすべてのメソッドに同じ権限を指定するには、メソッド名とアスタリスク(*)を使用する。

```
<method>
  <ejb-name>ScheduleEJB</ejb-name>
  <method-name>*</method-name>
</method>
```

- 1つのメソッドやすべてのオーバーロードメソッドに同じ権限を指定するには、メソッド名を使用する。

```
<method>
  <ejb-name>StudentName</ejb-name>
  <method-name>getScheduleList</method-name>
</method>
```

- 特定のオーバーロードメソッドを指定するには、そのオーバーロードメソッドのパラメータを指定する。

```
<method>
  <ejb-name>StudentName</ejb-name>
  <method-name>getScheduleList</method-name>
  <method-params>
    <method-param>String</method-param>
  </method-params>
</method>
<method>
  <ejb-name>StudentName</ejb-name>
  <method-name>getScheduleList</method-name>
  <method-params>
    <method-param>String</method-param>
```

```

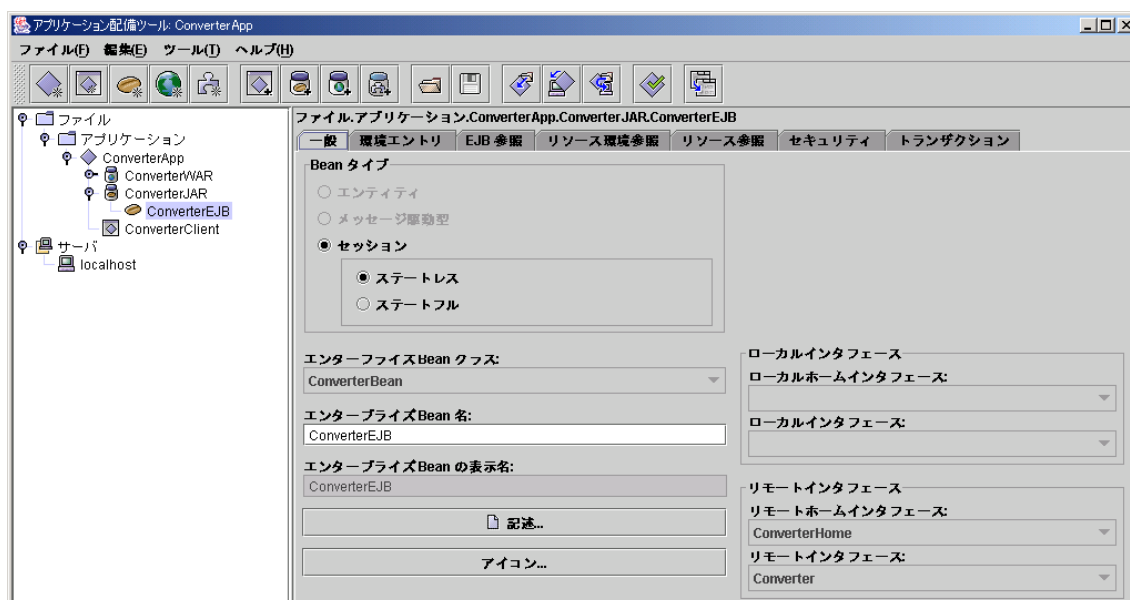
    <method-param>String</method-param>
  </method-params>
</method>
<method>
  <ejb-name>StudentName</ejb-name>
  <method-name>getScheduleList</method-name>
  <method-params>
    <method-param>String</method-param>
    <method-param>StartDate</method-param>
  </method-params>
</method>

```

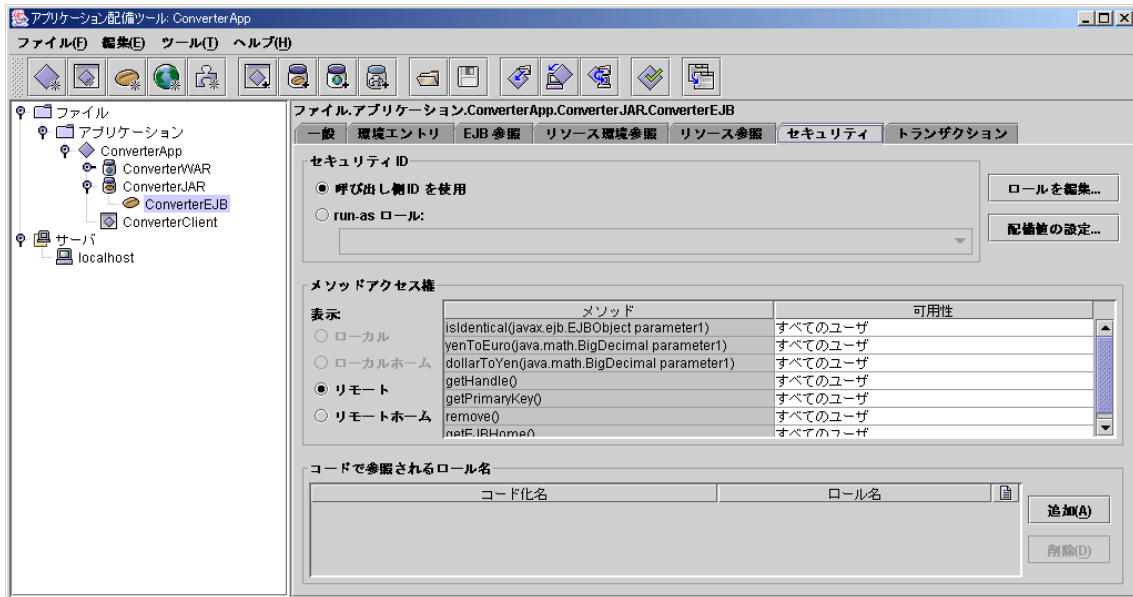
6.2.3. メソッドアクセス権の宣言

この節では、J2EE SDK に付属している `deploytool` を使用して EJB のメソッドのアクセス権を設定する。すなわち宣言によるセキュリティにより EJB のメソッドのアクセス権を設定する。各環境で使用される他のアプリケーションサーバにも、GUI ベースの配備ツールがありそれを使用することができる。もちろん、該当する配備記述子を直接編集して行ってもよい。その場合、XML 形式のテキストを編集することになるので十分に注意をする必要がある。ロールをメソッドへマップすることによりメソッドアクセス権を設定するには、`deploytool` で次の手順を行う。

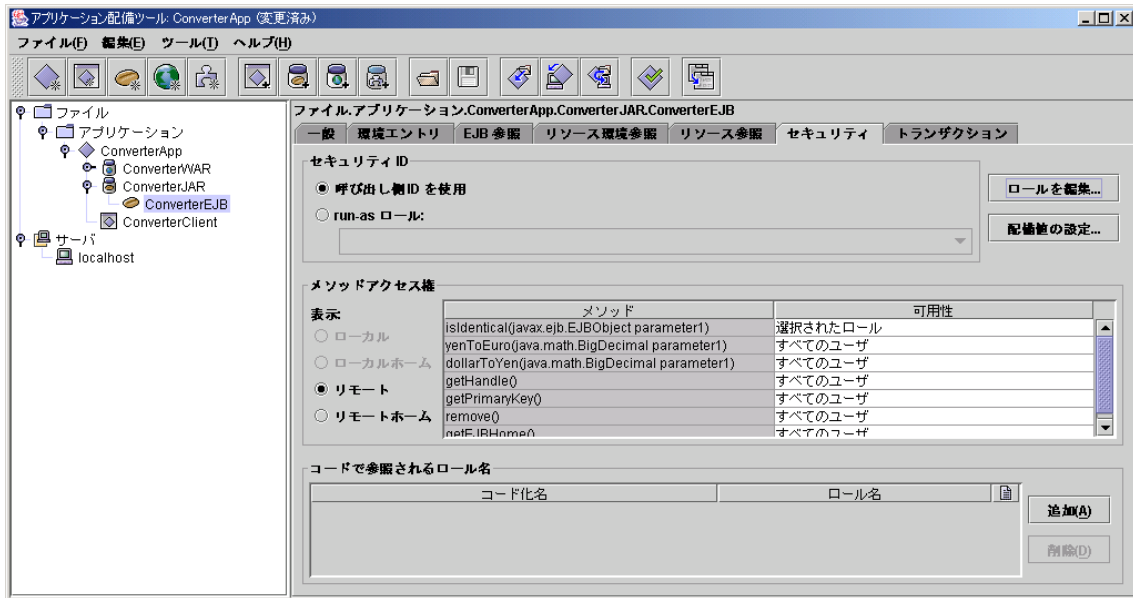
1. EJB を選択する。



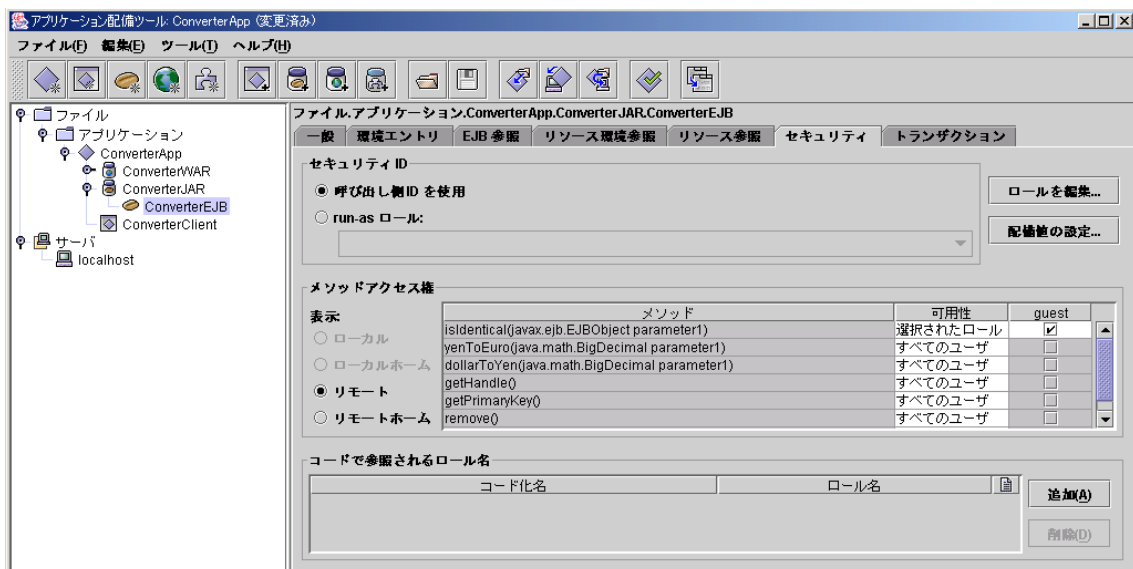
2. 「セキュリティ」タブを選択する。



- 「メソッドアクセス権」テーブルで、「可用性」列から「選択されたロール」を選択する。



- ロールがメソッド呼び出しを許可されている場合は、そのロール名の列でチェックボックスを選択する。



この作業の結果、ConverterEJB の配備記述子は次のように自動的に設定される。その抜粋を示す。配備記述子のすべてを閲覧したい場合は、ConverterEJB にフォーカスして

右クリックで「記述子ビューア」を選択する。

```
<ejb-jar>
  <display-name>ConverterJAR</display-name>
  <enterprise-beans>
    <session>
      ...
      <security-identity>
        <description></description>
        <use-caller-identity></use-caller-identity>
      </security-identity>
    </session>
  </enterprise-beans>
  <assembly-descriptor>
    <security-role>
      <role-name>customer</role-name>
    </security-role>
    ...
    <method-permission>
      <role-name>customer</role-name>
      <method>
        <ejb-name>ConverterEJB</ejb-name>
        <method-intf>Remote</method-intf>
        <method-name>isIdentical</method-name>
        <method-params>
          <method-param>javax.ejb.EJBObject</method-param>
        </method-params>
      </method>
    </method-permission>
  </assembly-descriptor>
</ejb-jar>
```

6.2.4. メソッドアクセス権の例

各アプリケーションおよびアプリケーション内の各コンポーネントによる、それぞれの認証要件の適用方法を理解するために、次の例について考察する。

1つ目のアプリケーションは、それぞれが1つのメソッドを持つ2つのEJBである、EJB 1 と EJB 2 から構成されているとする。各メソッドはロール名 MANAGER を使用して isCallerInRole メソッドを呼び出す。配備記述子には、各 EJB 内の isCallerInRole メソッドの呼び出し用である <security-role-ref> 要素が含まれている。EJB 1 の <security-role-ref> は、MANAGER をロール bad-managers にリンクする。配備記述子では、2つの <method-permission> 要素を定義する。1つはロール employees が EJB 1 のすべてのメソッドにアクセスできることを確立し、もう1つは、EJB 2 に対して同様の定義を行う。配備記述子には、employees, good-managers, bad-manager の3つの <security-role> 要素がある。デプロイはユーザ 1 にロール employees と good-managers を割り当てる。

2つ目のアプリケーションでは、EJB の EJB 3 によりコンテナ内に配備される。EJB 3 は同様に、ロール名 MANAGER を使用して isCallerInRole メソッドを呼び出す。この2つ目のアプリケーションの配備記述子には、MANAGER をロール good-managers にリンクする <security-ref> 要素が含まれている。同様に配備記述子は、ロール employees が EJB 3 のすべてのメソッドにアクセスできることを確立する <method-permission> 要素を定義する。配備記述子には、employee と good-managers の2つのロール要素が存在する。デプロイはユーザ 2 にロール employees と good-managers を割り当てる。

次の図において、ロールとメソッド間の関係としてメソッドアクセス権の構成を示す。また、呼び出し側のセキュリティ属性のロールへのマッピングおよびアプリケーションに埋め込まれた特権名とロール間のリンクも示す。

Application1

Application2

Security Roles Allowed to Access Method

	Employees	Good Mgrs.	Bad Mgrs.
EJB1 Method	×		
EJB2 Method	×		

	Employees	Good Mgrs.
EJB3 Method	×	

Security Attribute to Security Role Mapping

	Employees	Good Mgrs.	Bad Mgrs.
User1	×	×	
User2	×		×

	Employees	Good Mgrs.
User2	×	×

Application Role to Security Role Linkages

	Employees	Good Mgrs.	Bad Mgrs.
MANAGER		×	
MANAGER			×

	Employees	Good Mgrs.
EJB3メソッド	×	

次の表においては、異なるユーザがこれらの EJB に対してメソッド呼び出しを実行するときには発生する許可決定の一覧を示す。たとえば、ユーザ1が EJB 2 のメソッドに対してメソッド呼び出しを起動すると、<method-permission>要素にセキュリティロール employees と good-managers が指定されていて、デプロイヤがユーザ1にセキュリティロール employees を割り当てているため、コンテナは呼び出しをディスパッチする。しかし、EJB 2 の<security-role-ref>要素は MANAGER をセキュリティロール bad-managers にリンクし、これはユーザ1に当てはまらないため、isCallerInRole("MANAGER")メソッドは false を返す。ユーザ1が EJB 3 のメソッドを呼び出す場合、呼び出し側はディスパッチされない。これは、ユーザ1がどのセキュリティロールにも割り当てられないためである。

呼び出し	呼び出しがディスパッチされたか	IsCallerInRole の戻り値
ユーザ1 - EJB 1	yes	true

ユーザ 1 - EJB 2	yes	fales
ユーザ 1 - EJB 3	no	呼び出されず
ユーザ 2 - EJB1	yes	false
ユーザ 2 - EJB2	yes	true
ユーザ 2 - EJB3	yes	true

承認決定表

6.2.5. まとめ

各環境で使用されるアプリケーションサーバにおいて、GUI ベースの配備ツールを使用し、セキュリティに関するコードを変更することなく EJB のメソッドへのアクセス権を変更設定することができる。これは、宣言によるセキュリティの機構を利用しているからである。

6.2.6. 関連記事

- 6.1.1. EJB 層のセキュリティ

6.2.7. 参考文献

- J2EE チュートリアル, S・ポドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド

<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>

- Sun ONE Application Server 7 Web アプリケーション開発者ガイド

<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

6.3. EJB 層のプログラムによるセキュリティ

プログラムによるセキュリティは、宣言によるセキュリティだけではアプリケーションのセキュリティ要件を十分に表せない時に、その安全性を備えたいアプリケーションによって使用される。

6.3.1. EJB 層のプログラムによるセキュリティに関する API

EJB 層のプログラムによるセキュリティのために、`javax.ejb.EJBContext` インフェースに次の 2 つのメソッドが定義されている。

EJB 層のプログラムによるセキュリティは、`getCallerPrincipal` メソッドと `isCallerInRole` メソッドから構成される。EJB の呼び出し元の判定には `getCallerPrincipal` メソッドを使用し、呼び出し元のロールの判定には `isCallerInRole` メソッドを使用する。

- `java.security.Principal getCallerPrincipal()`

メソッドを呼び出しているプリンシパルのオブジェクトを返す。`Principal` クラスは、プリンシパルの名前を返す `getName` メソッドを定義している。`getCallerPrincipal` メソッドが `null` を返すことはない。また、プリンシパルはユーザと同一である。次のコード例では、EJB の `getUser` メソッドが、それを呼び出した J2EE ユーザの名前を返す。

```
public String getUser() {
    Return context.getCallerPrincipal().getName();
}
```

- `boolean isCallerInRole(String roleName)`

このメソッドは、メソッドの呼び出し側が、パラメータとして渡されたロールに属している場合に `true` を返す。次にコード例を示す。

```
Boolean result = context.isCallerInRole("Customer");
```

これら 2 つのメソッドを利用して、Bean 開発者はクライアントセキュリティ識別を行い、EJB の機能を有効または無効にすることができる。

6.3.2. プログラムによるセキュリティで使用されるメソッドの移植性について

プログラムによるセキュリティにおいて使用される 2 つのメソッドの移植性について考察する。

`isCallerInRole` メソッドは移植性があると考えられる。Bean 開発者はコードで使用するロール名を定義し、デプロイまたはアプリケーションアセンブラがこのロール参照を実際のロールに対応づけるからである。このため Bean 開発者はアプリケーションに定義される実際のロール名を知らなくてもコードを書くことができる。

一方、`getCallerPrincipal` メソッドは移植性がないと考えられる。使用されるプリンシパル名は、対象となる J2EE アプリケーションサーバが使用する認証機構に依存するからである。実際には、プリンシパル名が Java コードで文字列リテラルとしてハードコードされていないければ、`getCallerPrincipal` メソッドを移植性があるような形で使用可能である。

6.3.3. 宣言によるセキュリティとプログラムによるセキュリティの比較

宣言によるセキュリティの実装では、非常に高いレベルの柔軟性と移植性が得られる。従って、可能な限り EJB セキュリティは宣言によって実装することを推奨する。プログラムによるセキュリティで実装するのは、宣言によるセキュリティの実装ではセキュリティ要件を満たせない場合である。次の表で、これらの実装の長所と短所の比較を示す。

項目	宣言によるセキュリティ	プログラムによるセキュリティ
柔軟性	柔軟なセキュリティモデル	あまり柔軟ではないセキュリティモデル、セキュリティロールとロジックはハードコーディングされる
移植性	配備の際に開発者が変更できる	セキュリティロジックは Bean の開発者しか変更できない
脆弱性	セキュリティホールが少ない	セキュリティホールを作る可能性が高い
制御	複雑なセキュリティ制御を実装できず、メソッドレベルのセキュリティしか適用できない	複雑なセキュリティ制御が可能で、デプロイヤが変更できる
セキュリティポリシー	セキュリティポリシーはメソッドレベルやクラスレベルでのみ適用できる	セキュリティポリシーはカスタマイズ可能であり、コード行レベルで適用できる

6.3.4. まとめ

プログラムによるセキュリティで複雑なセキュリティ要件を実装することはできる。しかし、宣言によるセキュリティにより、プログラムやコンポーネントを変更することなくコンポーネントの配備時に事後的にセキュリティの設定が可能となっていることを活用すべきである。プログラムによるセキュリティによって実装する通常の手続き的な手法に対して、配備時の宣言を通じてコンポーネントにセキュリティ属性を設定する宣言によるセキュリティの手法は、J2EE のすべての層で一貫して適用されていて、J2EE のセキュリティの大きな特徴を形成している。

6.3.5. 関連記事

- 5.10.1. Web 層のプログラムによるセキュリティ

6.3.6. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-13-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

6.4. 保護されていない EJB 層のリソース

アプリケーションの中には、保護されていないすなわちアクセスするのに認証を必要としない EJB を持つものがある。アプリケーションアセンブラは、EJB 層内で、EJB 内のメソッドが呼び出し側のアイデンティティには関係なくコンテナに認証されることを示すために、<method-permission>要素内の<unchecked>要素を使用する。

6.4.1. ANYONE ロールの危険性

デフォルトでは、J2EE SDK は EJB メソッドにデフォルトで ANYONE ロールを割り当てている。例えば、匿名で、認証されない guest ユーザは、ANYONE ロールに所属する。このため、ロールのマッピングを行わない場合、どのユーザも EJB のメソッドを呼び出すことができる。この設定は、配備記述子で記述される。従って、このようなデフォルトの状態はセキュリティ上非常に危険である。適切な設定によりデフォルトの状態で使用しないようにすべきである。

6.4.2. 保護されない EJB 層のリソースの設定

次に示す配備記述子による宣言で、匿名で認証されていないユーザ、たとえばロール ANYONE が、特定の EJB リソースにアクセスできる。この場合、Catalogue という名前の EJB 内の browseSpecials メソッドが保護されず、どのユーザでも呼び出すことが可能である。

```
<method-permission>
  <unchecked></unchecked>
  <method>
    <ejb-name>Catalogue</ejb-name>
    <method-name>browseSpecials</method-name>
  </method>
  ...
</method-permission>
```

メソッドアクセス権が存在する場合、常にそのパーミッションが適用される。たとえば、メソッドアクセス権で、updateEmployeeInfo メソッドには employee ロールのみがアクセスできるように設定されている場合、employee ロールを持たなければこのメソッドにアクセスができない。employee ロールがどのユーザまたはグループにもマッピングされていない場合は、誰も updateEmployeeInfo メソッドを呼び出すことはできない。

6.4.3. まとめ

セキュリティポリシーから保護される EJB と保護されない EJB を明確に定義しなければならない。可能な限り保護されない EJB を定義しないことを推奨する。

6.4.4. 関連記事

- 5.11. 保護されていない Web リソース

6.4.5. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

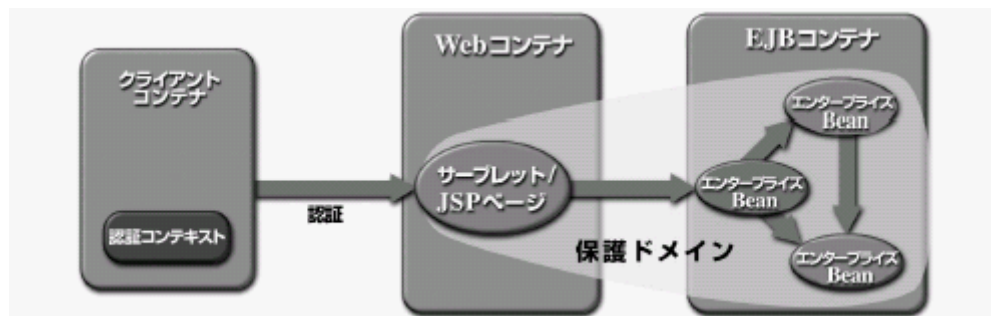
6.5. EJB 層の認証

Web コンテナの認証機構で認証されたセキュリティアイデンティティを基にして EJB へのアクセスを制御について述べる。

6.5.1. Web コンテナの認証機構の利用

セキュリティの意識の高まりとともにネットワークシステム全体の最外縁にファイアウォールを設置されることが多くなった。従って、特定のポート番号を使ったプロトコルでクライアント J2EE サーバが通信することは非常に困難である。従って、一般的に安全に開かれているポート番号を使用する HTTP プロトコルや HTTPS プロトコルを使って Web コンテナの認証機構を使うことが多くなってきた。さらに、Web コンテナの認証機構で認証されたセキュリティアイデンティティを基にして EJB へのアクセスを制御を行う。

J2EE 1.3 および EJB2.0 以前の J2EE プラットフォームでは、EJB コンテナは特定の認証機構を実装する必要はなかった。さらに、多くの環境では、ネットワークファイアウォールなどの技術により、クライアントコンテナと EJB 間の直接の相互作用(RMI 経由)は避けられていた。その結果、保護された Web コンポーネントを介して EJB にアクセスするユーザのアイデンティティを保証する Web コンテナの認証機構およびネットワークのアクセス可能性に EJB コンテナが依存することが一般的である。これは、「6.1.2. 宣言によるセキュリティの利用とその利点と欠点」でも述べた。次の図が示すように、このような構成では、Web コンテナを使用して、Web コンポーネントおよび Web コンポーネントが呼び出す EJB 用の保護ドメインの境界を適用する。



6.5.2. まとめ

EJB 層の認証には、Web コンテナを使用して、Web コンポーネントおよび Web コンポーネントが呼び出す EJB 用の保護ドメインの境界を適用する。

6.5.3. 関連記事

- 5.4. Web リソースのユーザ認証

- 9. セキュリティアイデンティティの伝達

6.5.4. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

7. アプリケーションクライアント層のセキュリティ

この章では、アプリケーションクライアント層のセキュリティ機構について述べる。

7.1. アプリケーションクライアント層のセキュリティ

アプリケーションクライアントの認証と承認のモジュールを作成する場合、JAAS(Java Authentication and Authorization Service)を使用するが、単に JAAS の API ドキュメントを参照して作成するのは非常に困難である。認証に関する暗号システムの深い理解が必要であり、それを前提としなければセキュアな認証モジュールの作成は非常に困難である。たとえば、X.509 デジタル証明書を使った相互認証モジュールにおいて、X.509 デジタル証明書を使った認証技術に関して深い理解が必要であるし、X.509 デジタル証明書の管理に関しても深い理解が必要である。

7.1.1. アプリケーションクライアント層のセキュリティの実装方法

J2EE アプリケーションクライアントの認証に必要な条件は、他の J2EE コンポーネントの必要条件と同じである。EJB 層か Web 層のいずれかの層の保護されていたリソースへのアクセスにはユーザ認証を要求するが、保護されていないリソースへのアクセスにはユーザ認証は行わない。

アプリケーションのクライアントは、認証のために JAAS(Java Authentication and Authorization Service)を使用できる。JAAS は、標準の PAM(Pluggable Authentication Module)フレームワークの Java バージョンを実装している。PAM フレームワークは、基本となる認証技術からアプリケーションを独立させるものである。アプリケーション自体へのどんな修正も行わずに、新しい、あるいは更新された認証技術をアプリケーションに接続することができる。アプリケーションは、LogicContext クラスをインスタンス化することにより認証手続きを可能にする。次に LogicContext オブジェクトは、認証を実行するために使用される認証技術やログインモジュールを判定するために、設定を参照する。

一般的なログインモジュールは、ユーザ名およびパスワードを表示し確認することができる。音声や指紋や光彩のパターンを読み取り、確認するログインモジュールもある。

場合により認証情報を得るためにユーザと通信する必要がある。ログインモジュールはこの目的のために、`javax.security.auth.callback.CallbackHandler` を使用する。アプリケー

ションは、`CallbackHandler` インタフェースを実装し、ログインコンテキストへ渡す。ログインコンテキストは、ベースとなるログインモジュールに直接それを転送する。ログインモジュールは、コールバックハンドラを使用して、ユーザからパスワードやスマートカードの暗証番号などの情報を収集したり、状態情報をなどをユーザに提供する。アプリケーションがコールバックハンドラを指定するのを可能にすることにより、ベースとなるログインモジュールは、アプリケーションがユーザがやり取りするのとは別に、独立した状態であることができる。

たとえば、GUI アプリケーションのコールバックハンドラの実装は、ユーザ入力を求めるためにウィンドウを表示する可能性がある。あるいは、コマンドラインツールのコールバックハンドラの実装では、単にユーザがコマンドラインから入力させるという可能性がある。

ログインモジュールは、適当なコールバックの配列をコールバックハンドラの `handle` メソッドに渡す。たとえば、ユーザ名には `NameCallback` を、パスワードには、`PasswordCallback` を渡す。コールバックハンドラは要求されたユーザとのやり取りを行い、コールバックに適切な値を設定する。たとえば、`NameCallback` を処理するために、`CallbackHandler` は名前の入力を要求し、ユーザから値を取得し、そして `NameCallback` の `setName` メソッド呼び出して名前を格納する。

7.1.2. まとめ

アプリケーションクライアント層のセキュリティの実装において、JAAS を使用することを推奨する。

7.1.3. 関連記事

- 5.4.1. 宣言によるセキュリティによる Web リソースのユーザ認証
- 6.1.1. EJB 層のセキュリティの設定

7.1.4. 参考文献

- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション

7.2. アプリケーションクライアントのコールバックハンドラの指定

アプリケーションクライアントのログインモジュールを実装する際は、コールバックハンドラを使用してユーザから認証情報を取得することについて述べる。

7.2.1. ログインモジュールにおける認証情報の取得

ログインモジュールがユーザとの通信を介して認証情報を取得する場合、ログインモジュールは、`javax.security.auth.callback.CallbackHandler` を使用してこれを実行する。アプリケーションは、`CallbackHandler` インタフェースを実装し、これを `LoginContext` に渡す。`LoginContext` はこれを基盤となるログインモジュールに直接転送する。ログインモジュールは `CallbackHandler` を使って、ユーザからの入力 (パスワード、スマートカードなどの暗証番号など) を収集したり、ユーザに情報 (状態情報など) を提供したりする。アプリケーションに `CallbackHandler` の指定を許可することにより、基盤となるログインモジュールは、アプリケーションとユーザ間の通信方法に依存せずに動作するようになる。たとえば、GUI アプリケーション用の `CallbackHandler` 実装は、ウィンドウを表示してユーザの入力を促す。一方、非 GUI ツール用の `CallbackHandler` は、コマンド行から直接入力するようユーザに求める。

7.2.2. コールバックハンドラ

`CallbackHandler` は、1 つのメソッドを持った JAAS インタフェースである。

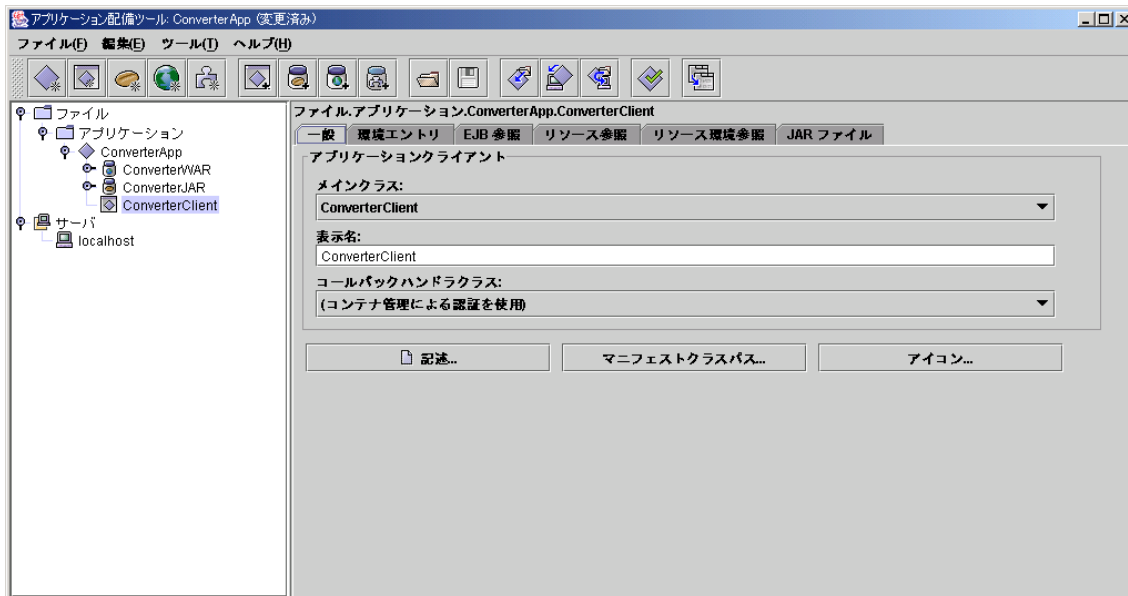
```
void handle(Callback[] callbacks)
    throws java.io.IOException, UnsupportedCallbackException;
```

ログインモジュールは `CallbackHandler handle` メソッドに適切な `Callback` からなる配列 (たとえばユーザ名の場合 `NameCallback`、パスワードの場合 `PasswordCallback`) を渡し、要求によりユーザと通信し、`Callback` 内に適切な値を設定する。たとえば、`NameCallback` を処理する場合、`CallbackHandler` はユーザから名前を取得し、`NameCallback` の `setName` メソッドを呼び出してその名前を格納する。

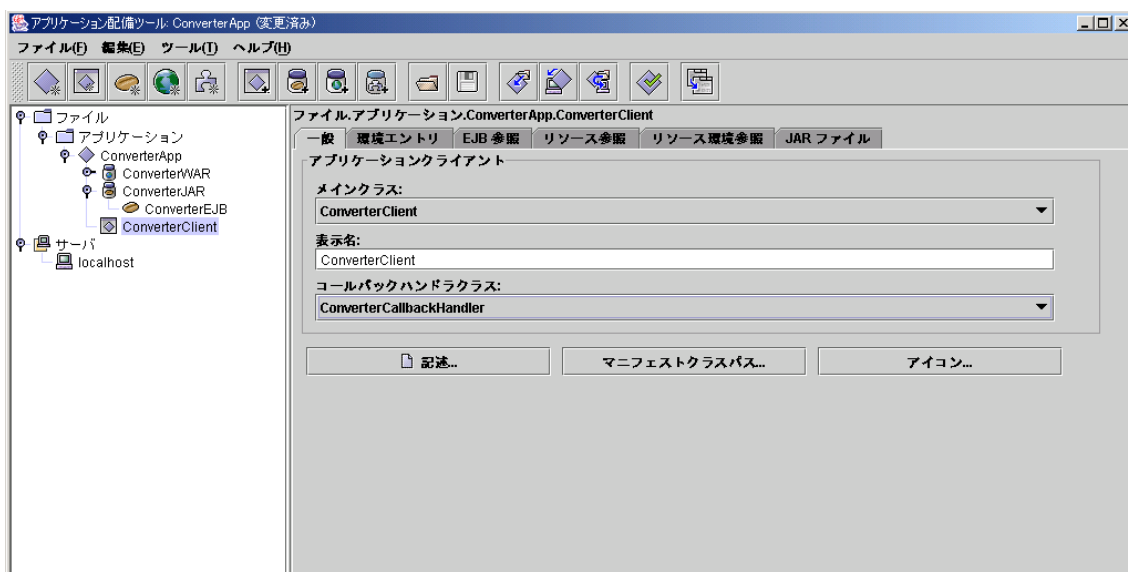
7.2.3. アプリケーションクライアントのコールバックハンドラの指定

アプリケーションクライアントのコールバックハンドラを指定するには、J2EE SDK deploytool を使用して次の手順を行う。

1. アプリケーションクライアント JAR を選択する。
2. 「一般」タブを選択する。



3. 「コールバックハンドラクラス」メニューから、ユーザ認証データを集めるインタフェースとして使用される、CallbackHandler クラスを選択する。



この作業により、次の配備記述子内にアプリケーションクライアントのコールバックハンドラが定義される。ここでは、`ConverterCallbackHandler` という名前の `callback-handler` の要素が生成されている。配備記述子のすべてを閲覧したい場合は、`ConverterClient` にフォーカスして右クリックで「記述子ビューア」を選択する。

```
<application-client>
  <display-name>ConverterClient</display-name>
  <ejb-ref>
    <ejb-ref-name>ejb/SimpleConverter</ejb-ref-name>
    <ejb-ref-type>Session</ejb-ref-type>
    <home>ConverterHome</home>
    <remote>Converter</remote>
  </ejb-ref>
  <callback-handler>ConverterCallbackHandler</callback-handler>
</application-client>
```

7.2.4. まとめ

アプリケーションクライアントのログインモジュールは、コールバックハンドラを使用してユーザから認証情報を取得する。

7.2.5. 関連記事

- 7.1.1. アプリケーションクライアント層のセキュリティ

7.2.6. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1

[3-doc-tutorial-j.pdf](#)

- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

8. EIS 層のセキュリティ

この章では、Enterprise Information System リソースにセキュアにアクセスすることについて述べる。

8.1. EIS 層のセキュリティ

この節では、ESI 層について簡単に述べて、EIS 層のセキュリティの概要を述べる。さらに EIS 層のセキュリティに脆弱性があると非常に危険であることを述べる。

8.1.1. EIS サインオンの種類とその選択

企業の既存の情報システムを構成するアプリケーション群を EIS;Enterprise Information System と呼称する。このアプリケーション群により、企業の情報インフラストラクチャが提供される。EIS は、クライアントに対して、明確に定義された一連のサービスを提供する。これらのサービスは、ローカルインタフェースやリモートインタフェースまたはその両方のインタフェースとして提供される。EIS の例として、ERP システム、メインフレームトランザクション処理システム、レガシーデータベースなどがある。

EIS 層では、アプリケーションコンポーネントが EIS リソースへの接続を要求する。この接続の一部として、EIS はリソースへの認証を要求する場合がある。アプリケーションコンポーネントのプロバイダには、EIS 認証の設計の際に 2 つの選択肢がある。

- コンテナ管理による認証を使用する。アプリケーションコンポーネントは、コンテナに EIS への認証の設定および管理を行わせる。コンテナは、EIS インスタンスへ接続を確立するためにユーザ名およびパスワードを決定する。
- コンポーネント管理による認証を使用する。アプリケーションコンポーネントのコードに EIS への認証処理を実行するコードを開発し含めることにより、EIS への認証を管理する。

コンポーネントプロバイダは、配備ツールを使用して上記 2 つのタイプの認証を選択できる。J2EE アプリケーションサーバが接続の対象となる EIS リソースに対するコンテナをサポートしているならば、コンテナ管理による認証を推奨する。

8.1.2. Java クライアントの J2EE アプリケーションの EIS 層に直接接続の危険性

EIS には機密情報たとえば、社員の人事情報やミッションクリティカルな情報たとえば、コールセンターの顧客情報が格納されていることがほとんどである。権限のあるユーザ以外はこれらの情報にアクセスできないようにすることが非常に重要である。

一般的に、Java クライアントは、J2EE アプリケーションの EIS 層に直接接続することはできない。それをさせてはならない。EIS クライアントでは、リモートの EIS リソース上にあるデータを操作するために JDBC API などの有効なインターフェースを必要とする。実装した EIS クライアントが不安定だったり、ハッキングされたり、最初から構築された悪質なクライアントがこのインターフェースを間違えて使用すると、データベースなどのデータに欠陥を生じる可能性がある。

8.1.3. まとめ

EIS 層では、アプリケーションコンポーネントが EIS リソースへの接続を要求する。この接続の一部として、EIS はリソースへの認証を要求する場合がある。アプリケーションコンポーネントのプロバイダには、EIS 認証の設計の際に 2 つの選択肢を選択できる。EIS リソースに対するコンテナをサポートしているならば、コンテナ管理による認証を推奨する。

8.1.4. 関連記事

- 8.2.1. EIS 認証の設定
- 8.3.1. EIS コンテナ管理による認証
- 8.4.1. EIS コンポーネント管理による認証
- 8.5.2. リソースアダプタのセキュリティの設定

8.1.5. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第 2 版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
-

- プロフェッショナル Java サーバプログラミング J2EE の設計と開発、Subrahmanyam Allamaraju 他著、インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

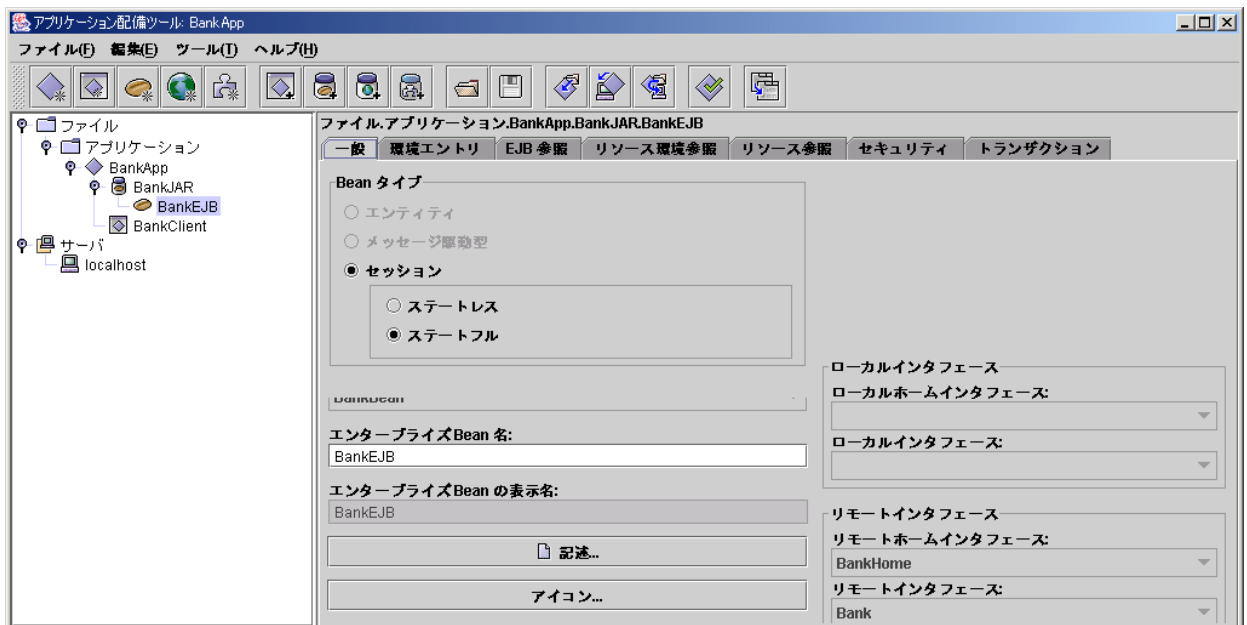
8.2. EIS 認証の設定

この節では、EIS 層に正しくアクセスするための認証の設定を J2EE SDK の deploytool を使い設定することについて述べる。

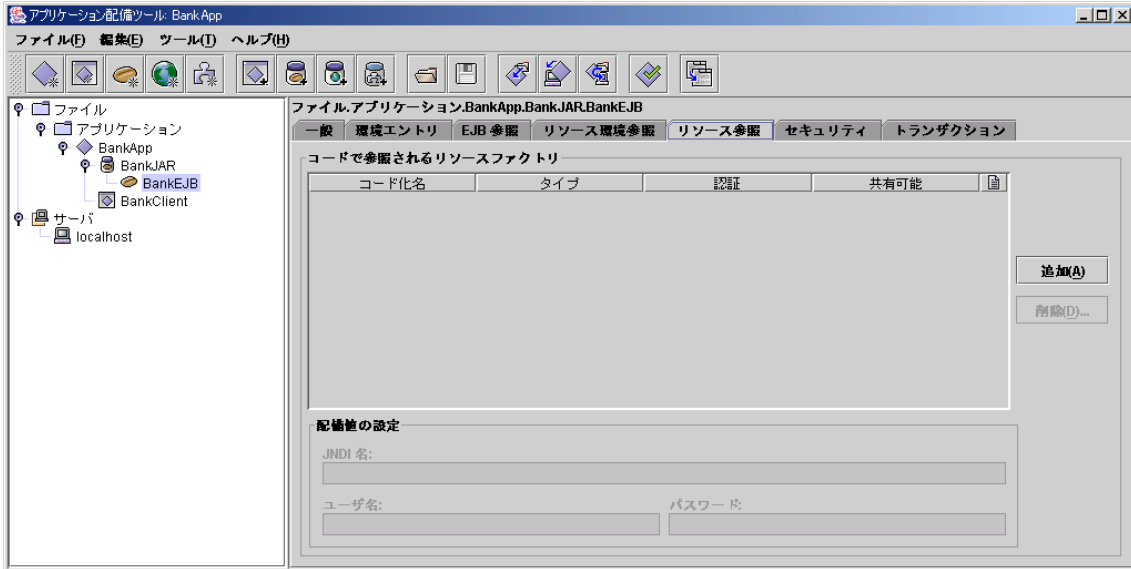
8.2.1. EIS 認証の設定

EIS 認証のタイプを設定するには、J2EE SDK の deploytool を使用して次の手順を行う。

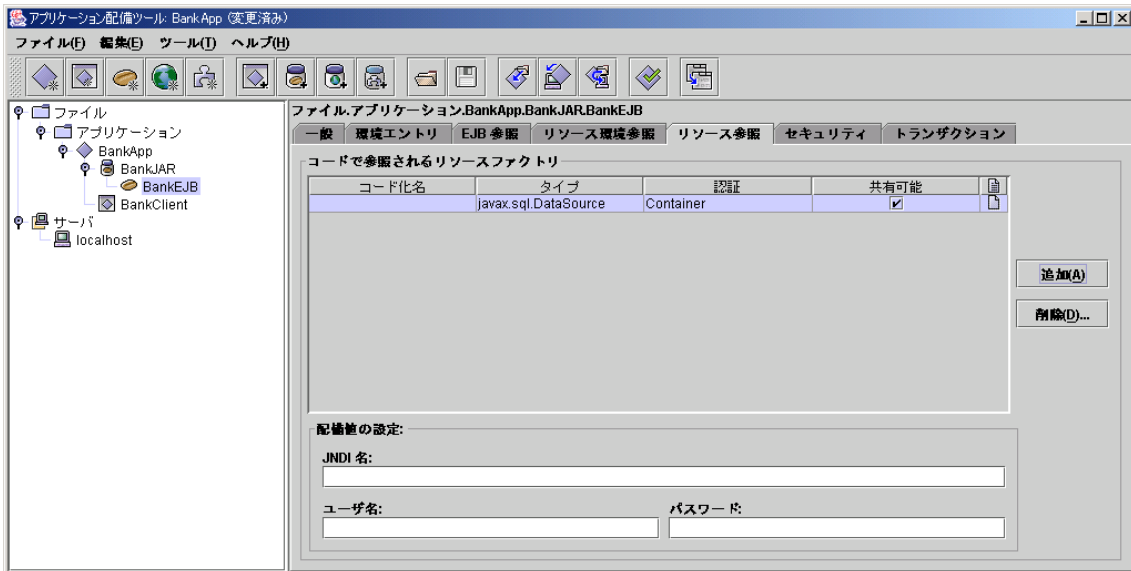
1. コンポーネントを選択する。



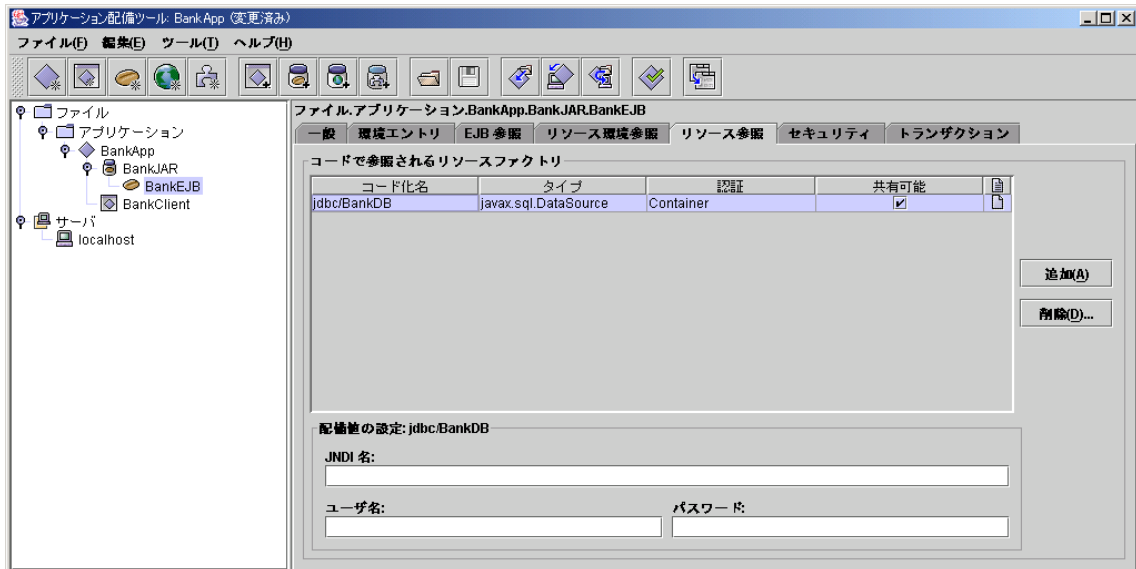
2. 「リソース参照」タブを選択する。



3. 「追加」をクリックする。



4. 「認証」コンボボックスで、コンテナ管理による認証には「Container」を、コンポーネント管理による認証には「Application」を選択する。



この作業により、次の配備記述子内に EIS 認証が定義される。ここでは、jdbc/BankDB というリソースへの <resource-ref>の要素が生成されている。

```
<ejb-jar>
  <display-name>BankJAR</display-name>
  <enterprise-beans>
    <session>
      <display-name>BankEJB</display-name>
      <ejb-name>BankEJB</ejb-name>
      <home>BankHome</home>
      <remote>Bank</remote>
      <ejb-class>BankBean</ejb-class>
      <session-type>Stateful</session-type>
      <transaction-type>Container</transaction-type>
      <security-identity>
        <description></description>
        <use-caller-identity></use-caller-identity>
      </security-identity>
      <resource-ref>
        <res-ref-name>jdbc/BankDB</res-ref-name>
        <res-type>javax.sql.DataSource</res-type>
        <res-auth>Application</res-auth>
        <res-sharing-scope>Shareable</res-sharing-scope>
      </resource-ref>
    </session>
  </enterprise-beans>
  ...
</ejb-jar>
```

8.2.2. EIS 認証にコンポーネント管理を選択したときの注意

EIS 認証にコンポーネント管理を選択すなわち `deploytool` で認証に「Application」を選択した時、アプリケーションコンポーネントのコードに EIS への認証処理をするコードを実装することになる。従って、実装には十分に注意して認証で使用するユーザ名とパスワードの管理や扱うプログラムの実装に対して十分に注意を払わなければならない。

8.2.3. まとめ

EIS 層に正しくアクセスするための認証の設定を J2EE SDK の `deploytool` を使い設定できる。コンテナ管理による認証には「Container」を、コンポーネント管理による認証には「Application」を選択することができる。

8.2.4. 関連記事

- 8.1. EIS 層のセキュリティ
- 8.3.1. EIS コンテナ管理による認証
- 8.4.1. EIS コンポーネント管理による認証
- 8.5.2. リソースアダプタのセキュリティの設定

8.2.5. 参考文献

-
- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>

- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

8.3. EIS コンテナ管理による認証

`javax.resource.cci.ConnectionFactory.getConnection` メソッドは、EIS インスタンスへの接続を取得する。コンポーネントがコンテナに EIS サインオンを管理させる場合は、この `getConnection` 形式を使用する必要がある。この場合は、コンポーネントはどのセキュリティ情報も渡さない。

8.3.1. EIS コンテナ管理による認証

コンテナ管理による認証では、アプリケーションコンポーネントは、リソースに認証するために `getConnection` メソッドに対してセキュリティ情報を渡す必要がない。次の例に示されるように、セキュリティ情報はコンテナにより提供される。

```
// Business method in an application component
Context initctx = new InitialContext();

// perform JNDI lookup to obtain a connection factory
javax.resource.cci.ConnectionFactory cxf =
    (javax.resource.cci.ConnectionFactory)initctx.lookup(
        "java:comp/env/eis/MainframeCxfactory");

// Invoke factory to obtain a connection. The security
// information is not passed in the getConnection method
javax.resource.cci.Connection cx = cxf.getConnection();
...
```

8.3.2. まとめ

`javax.resource.cci.ConnectionFactory.getConnection` メソッドは、EIS インスタンスへの接続を取得する。コンポーネントがコンテナに EIS 認証を管理させる場合は、この `getConnection` 形式を使用する必要がある。この場合は、コンポーネントはどのセキュリティ情報も渡さない。

8.3.3. 関連記事

- 8.1. EIS 層のセキュリティ
- 8.2.1. EIS 認証の設定

- 8.4.1. EIS コンポーネント管理による認証
- 8.5.2. リソースアダプタのセキュリティの設定

8.3.4. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

8.4. EIS コンポーネント管理による認証

`javax.resouce.cci.ConecionFactory.getConnection` メソッドは、`javax.resouce.cci.ConnectionSpec` を引数として、EIS インスタンスへの接続を取得する。コンポーネントは、任意のリソースアダプタ固有のセキュリティ情報および接続パラメータを渡す必要がある場合、`getConnection` 形式を `javax.resouce.cci.ConnectionSpec` パラメータと共に使用する必要がある。コンポーネント管理サインオンの場合、アプリケーションコンポーネントはセキュリティ情報（ユーザ名やパスワードなど）を `ConnectionSpec` インスタンスを通じて渡す。

`getConnection` メソッドを通じて渡されるプロパティは、クライアント固有（例: ユーザ

名、パスワード、言語) であり、ターゲットの EIS インスタンス (例: ポート番号、サーバ名) の構成に関連しないものにする必要があることに注意する。**ManagedConnectionFactory** インスタンスは、EIS インスタンスへの接続を作成するために必要な完全なプロパティセットにより設定される。

8.4.1. EIS コンポーネント管理による認証

コンポーネント管理による認証では、アプリケーションコンポーネントはリソースへ認証するのに必要なセキュリティ情報を、`getConnection()` メソッドへ渡す必要がある。たとえば次の例に示されるように、セキュリティ情報はユーザ名やパスワードの場合がある。

```
// Method in an application component
Context initctx = new InitialContext();

// perform JNDI lookup to obtain a connection factory
javax.resource.cci.ConnectionFactory cxf =
(javax.resource.cci.ConnectionFactory)initctx.lookup(
"java:comp/env/eis/MainframeCxFactory");

// Invoke factory to obtain a connection
com.myeis.ConnectionSpecImpl properties = //..

// get a new ConnectionSpec
properties.setUsername("...");
properties.setPassword("...");
javax.resource.cci.Connection cx =
cxf.getConnection(properties);
...
```

8.4.2. まとめ

コンポーネント管理によるサインオンでは、アプリケーションコンポーネントはリソースへサインオンするのに必要なセキュリティ情報を、`getConnection()` メソッドへ渡す必要がある。

8.4.3. 関連記事

- 8.1. EIS 層のセキュリティ
- 8.2.1. EIS 認証の設定

- 8.3.1. EIS コンテナ管理による認証
- 8.5.2. リソースアダプタのセキュリティの設定

8.4.4. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

8.5. リソースアダプタのセキュリティの設定

リソースアダプタのセキュリティ設定において、認証機構について述べる。

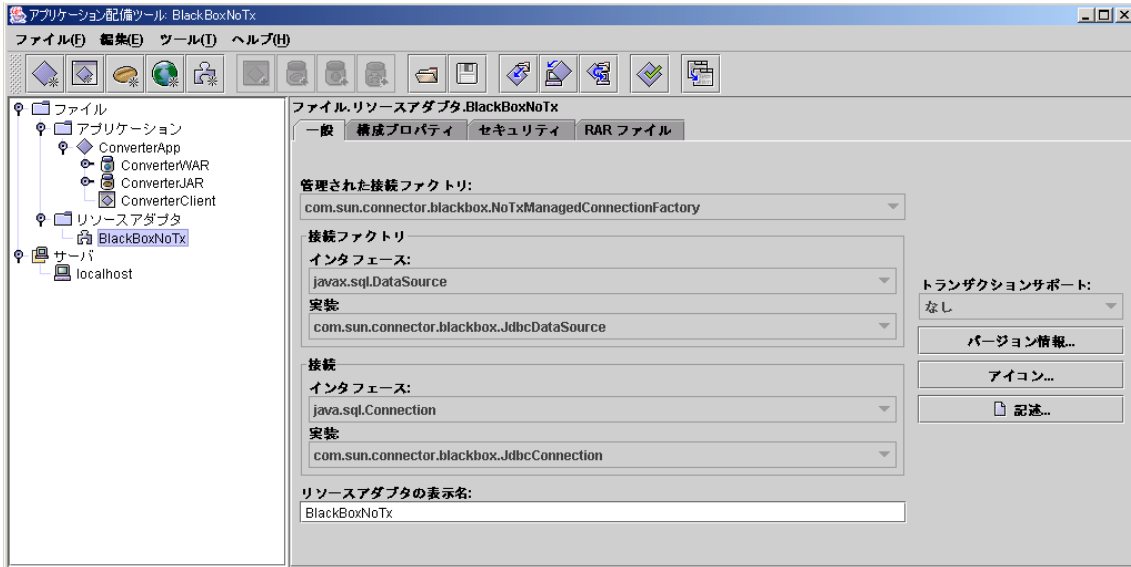
8.5.1. リソースアダプタのセキュリティ設定の必要性

リソースアダプタのセキュリティ設定において、認証なしで不特定のアクセスを許すことは、あらゆる攻撃をかけられる可能性がある。特に、データベースの認証機構も深く理解して、リソースアダプタのセキュリティ設定を行うべきである。

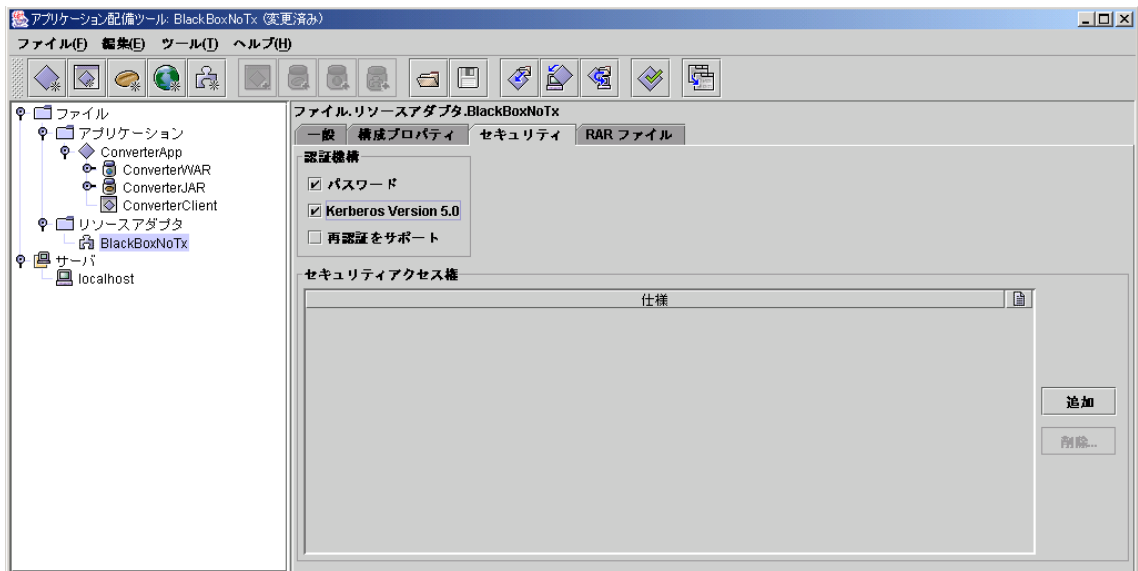
8.5.2. リソースアダプタのセキュリティの設定の作業手順

認証の設定に加え、リソースアダプタのセキュリティ要件の設定も行う必要がある。リソースアダプタにセキュリティを加えるには、J2EE SDK の `deploytool` を使い次の手順を行う。

1. リソースアダプタ RAR (Resource Adapter Archive) を選択する。



2. 「セキュリティ」タブを選択する。「認証機構」パネルで、このリソースアダプタがサポートする認証機構を選択する。

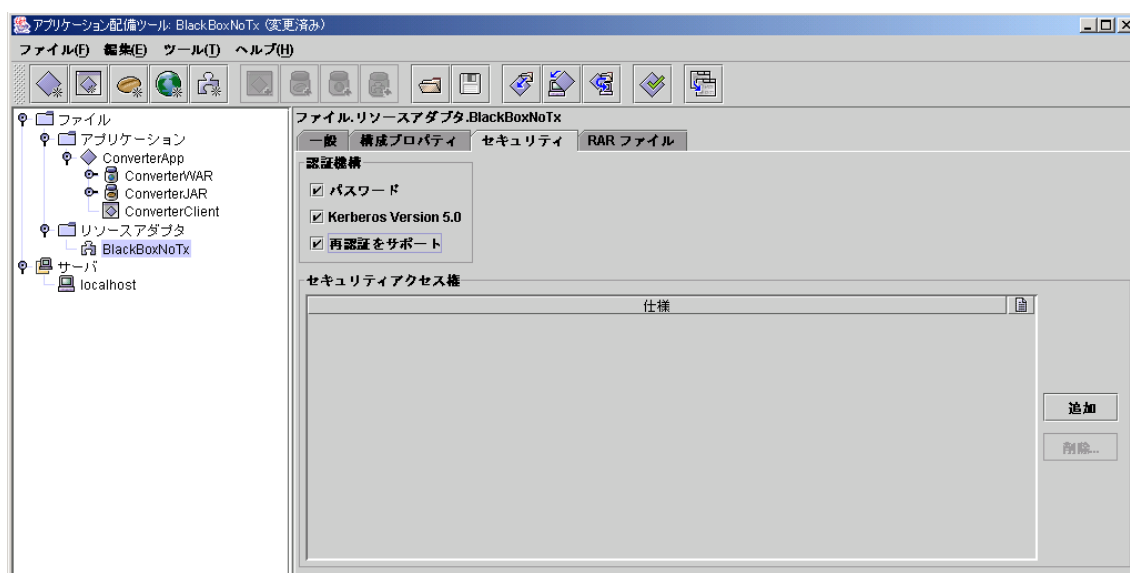


- パスワード: EIS への接続には、ユーザ名とパスワードが必要である。
- Kerberos Version 5.0 : リソースアダプタは、Kerberos 認証機構をサポートする。詳細は、RFC-1510 の『The Kerberos Network Authentication Service

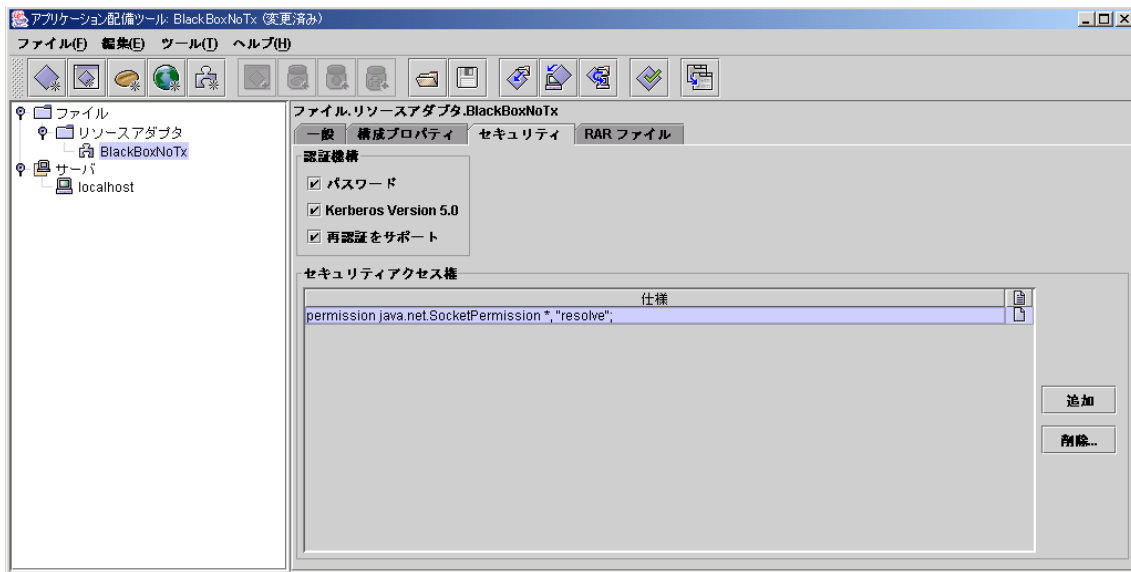
(V5)』を参照すること。この仕様は、<http://www.ietf.org/rfc/rfc1510.txt> で参照できる。

機構を選択しないことも、複数の機構を選択することもできる。機構を選択しない場合は、セキュリティ認証はサポートされない。

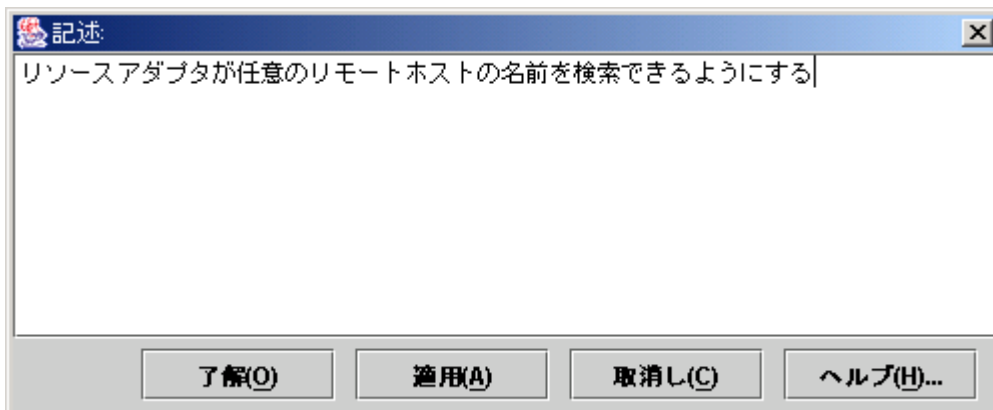
3. リソースアダプタが、既存の物理的な接続での再認証の実行をサポートする場合は、「再認証をサポート」を選択する。再認証は、接続を確立するために使われたものとは異なるセキュリティコンテキストを用いて、アプリケーションサーバが `getConnection()` メソッドを呼び出すときに実行される。



4. 運用環境上のシステムリソースにアクセスが必要なリソースアダプタにセキュリティアクセス権を追加するには、「セキュリティアクセス権」パネルの「追加」ボタンをクリックする。デフォルトの設定に含まれていないアクセス権だけを指定する。デフォルトのアクセス権は、『J2EE Connector Architecture Specification 1.0』の「Section 11.2」の「Table 2」に一覧表示されている。



5. 各セキュリティアクセス権に対して、「記述」アイコンの列をクリックして、アクセス権に対する説明を入力する。



この作業の結果、**BlackBoxNoTx** の配備記述子は次のように自動的に設定される。その抜粋を示す。配備記述子のすべてを閲覧したい場合は、**BlackBoxNoTx** にフォーカスして右クリックで「記述子ビューア」を選択する。

```

<connector>
  <display-name>BlackBoxNoTx</display-name>
  <vendor-name>Java Software</vendor-name>
  <spec-version>1.0</spec-version>
  <eis-type>JDBC Database</eis-type>
  <version>1.0</version>
  <resourceadapter>
    ...
    <authentication-mechanism>
      <authentication-mechanism-type>BasicPassword</authentication-mechanism-type>
  </authentication-mechanism>
  <credential-interface>javax.resource.security.PasswordCredential</credential-interface>
  </authentication-mechanism>
  <authentication-mechanism>
    <authentication-mechanism-type>Kerbv5</authentication-mechanism-type>
  </authentication-mechanism>
  <credential-interface>javax.resource.spi.security.GenericCredential</credential-interface>
  </authentication-mechanism>
  <reauthentication-support>true</reauthentication-support>
  <security-permission>
    <description>
      リソースアダプタが任意のリモートホストの名前を検索できるようにする<
    /description>
    <security-permission-spec>permission          java.net.SocketPermission
  "resolve";</security-permission-spec>
  </security-permission>
  </resourceadapter>
</connector>

```

セキュリティアクセス権を削除するには、テーブルからアクセス権を選択し、「削除」をクリックする。

8.5.3. まとめ

リソースアダプタのセキュリティ設定において、認証機構は必ず設定すべきである。認証なしで不特定のアクセスを許すことは、あらゆる攻撃をかけられる可能性がある。特に、データベースの認証機構も深く理解して、リソースアダプタのセキュリティ設定を行うべきである。

8.5.4. 関連記事

- 8.2.1. EIS 認証の設定
- 8.3.1. EIS コンテナ管理による認証
- 8.4.1. EIS コンポーネント管理による認証
- 8.5.2. リソースアダプタのセキュリティの設定

8.5.5. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版,
Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発,
Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

9. セキュリティアイデンティティの伝達

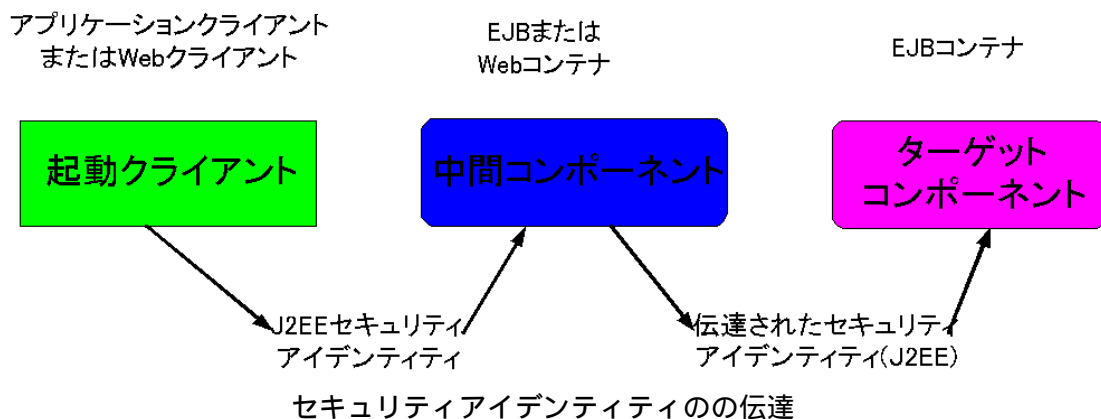
この章ではコンポーネント間のセキュリティアイデンティティの伝達について述べる。

9.1. セキュリティアイデンティティの伝達

セキュリティアイデンティティの伝達の方法には2通り存在する。セキュリティ要件からどちらを選択し使用すれば良いかを述べる。

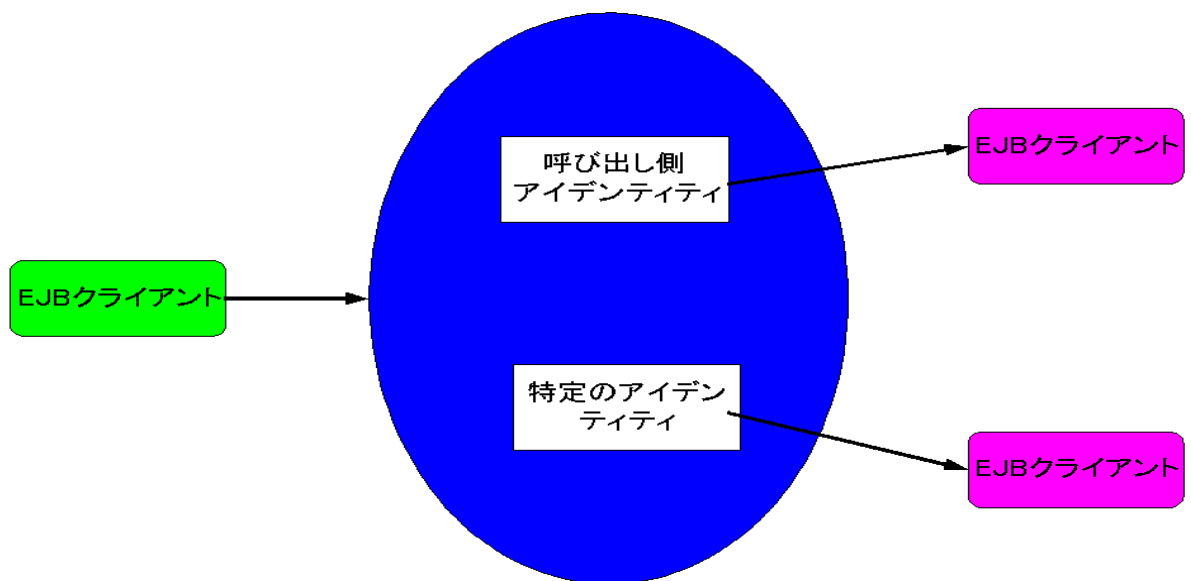
9.1.1. セキュリティアイデンティティの伝達

EJB または Web コンポーネントを配備するときに、そのコンポーネントの内部から呼び出された EJB に伝達されるセキュリティアイデンティティを指定することができる。次の図を参照。



次の伝達の方法から、1つを選択することができる。

- 呼び出し元の間コンポーネントのアイデンティティが、ターゲット EJB へ伝達される。この手法は、ターゲットコンテナが中間コンテナを信頼している場合に使用される。
- 特定のアイデンティティがターゲット EJB へ伝達される。この手法は、ターゲットコンテナが特定のアイデンティティを介してアクセスすることが予期される場合に使用される。



セキュリティアイデンティティの設定により、どのアイデンティティがターゲットとなる EJB コンテナにアクセスするかを十分に吟味しなければならない。特にアプリケーションクライアントと Web クライアントのアイデンティティを注意深く識別する必要がある。設計上可能な限り、呼び出し元の間コンポーネントのアイデンティティが、ターゲット EJB へ伝達する方法を推奨する。

9.1.2. まとめ

セキュリティアイデンティティの伝達の設定において、セキュリティ要件において可能な限り、呼び出し元の間コンポーネントのアイデンティティが、ターゲット EJB へ伝達する方法を選択することを推奨する。

9.1.3. 関連記事

- 9.2.1. コンポーネントの伝達されたセキュリティアイデンティティの設定
- 9.3.1. クライアント認証の設定
- 10.1.1. J2EE のユーザ

9.1.4. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版,
Rich Grenn 他 著, 株式会社ピアソン・エデュケーション

- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

9.2. コンポーネントの伝達されたセキュリティアイデンティティの設定

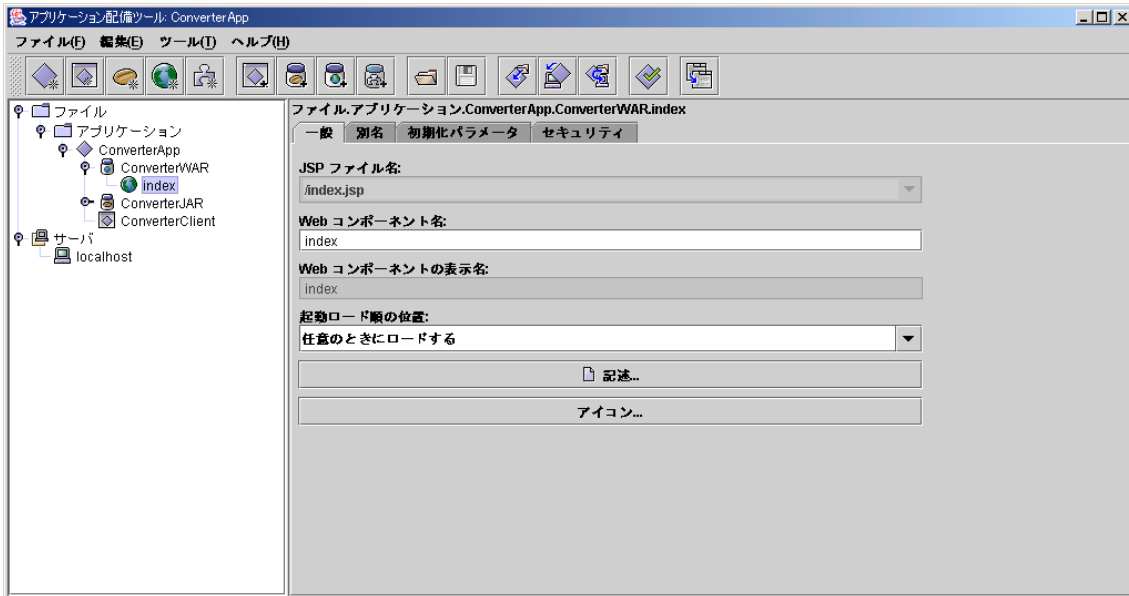
J2EE SDK に付属している `deploytool` を使用しコンポーネントから伝達されるセキュリティアイデンティティのタイプを選択について述べる。

9.2.1. コンポーネントの伝達されたセキュリティアイデンティティの設定

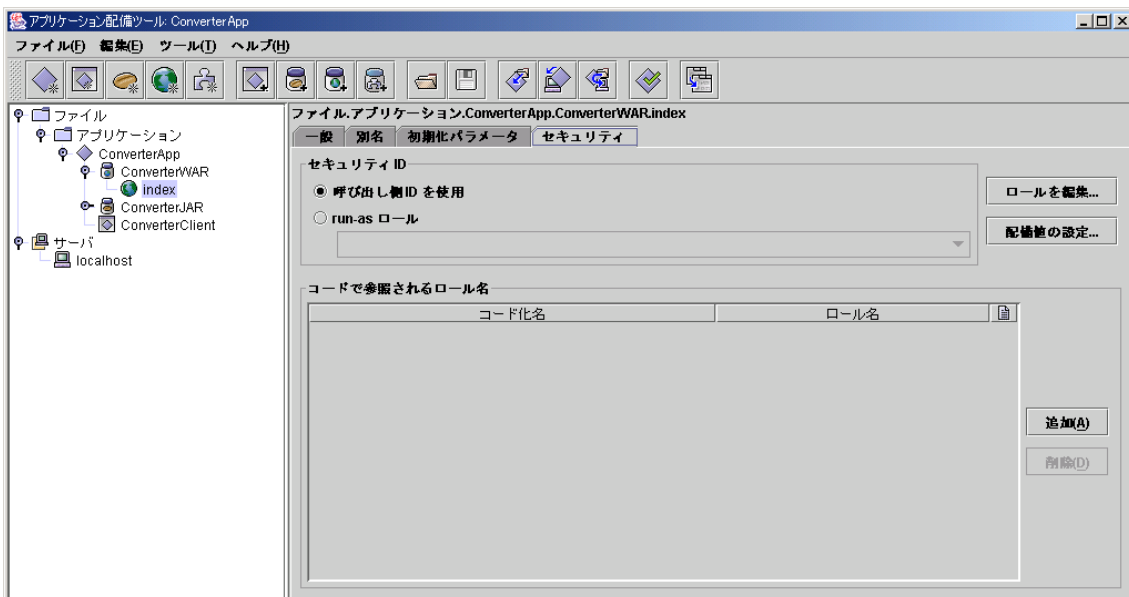
J2EE SDK に付属している `deploytool` を使用して、EJB または Web コンポーネントから伝達されるセキュリティアイデンティティのタイプを選択することができる。

コンポーネントが実行中の呼び出し元のアイデンティティを伝達するために、EJB または Web コンポーネントを設定するには、次の手順を行う。

1. コンポーネントを選択する。



2. 「セキュリティ」タブを選択する。
3. 「セキュリティ ID」パネルでラジオボタンの「呼び出し側 ID を使用」を選択する。



この作業の結果、ConverterJAR の配備記述子は次のように自動的に設定される。その

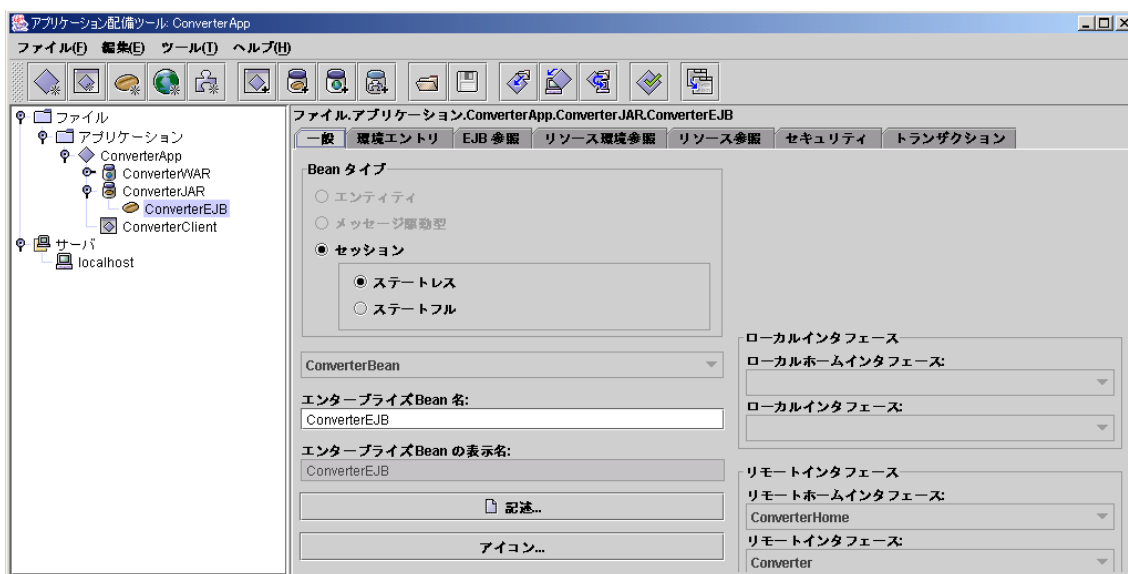
抜粋を示す。配備記述子のすべてを閲覧したい場合は、ConverterJAR にフォーカスして右クリックで「記述子ビューア」を選択する。

```
<ejb-jar>
  <display-name>ConverterJAR</display-name>
  <enterprise-beans>
    <session>
      ...
      <security-identity>
        <description></description>
        <use-caller-identity></use-caller-identity>
      </security-identity>
    </session>
  </enterprise-beans>
  ...
</ejb-jar>
```

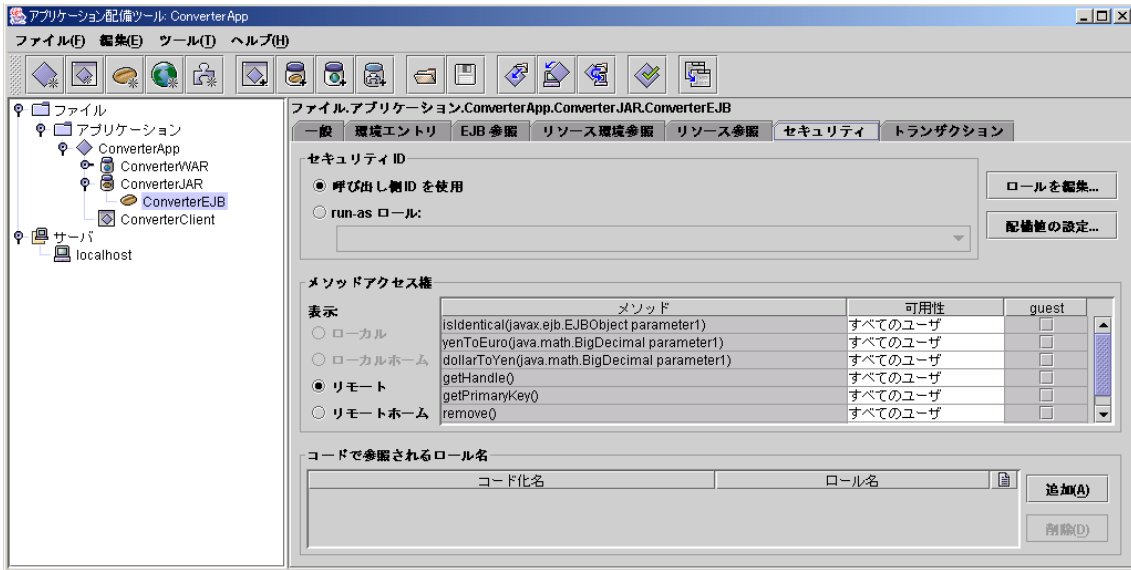
コンポーネントが実行中の呼び出し元のアイデンティティを伝達するために、EJB または Web コンポーネントを設定では配備記述子の<security-identity>要素の中で、<user-caller-indetity>オプション要素で定義されている。

コンポーネントが実行中でないセキュリティアイデンティティを伝達するコンポーネントを設定するには、次の手順を行う。

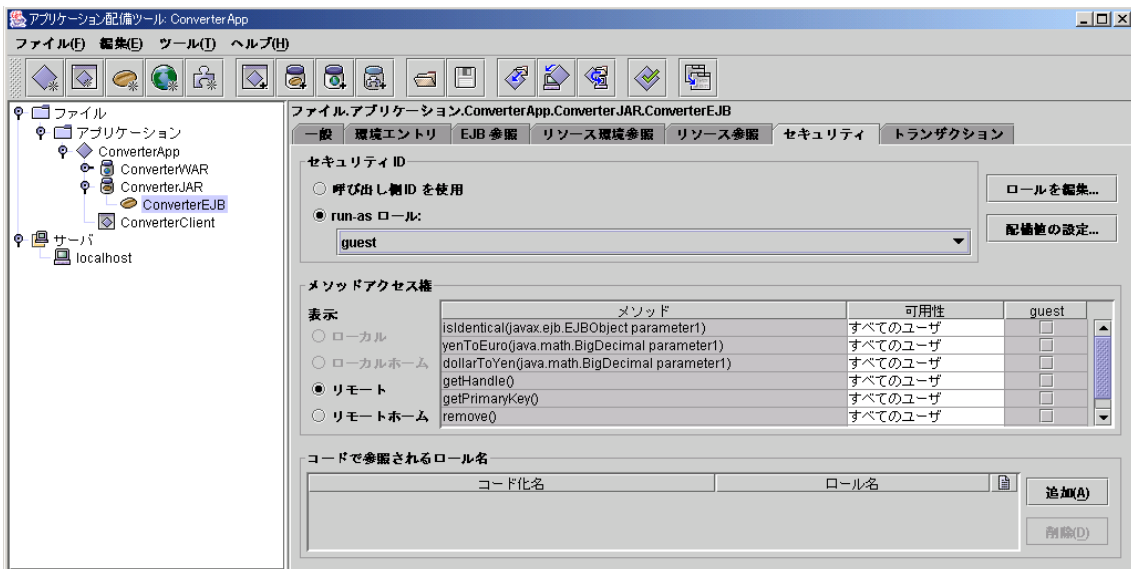
1. コンポーネントを選択する。



2. 「セキュリティ」タブを選択する。

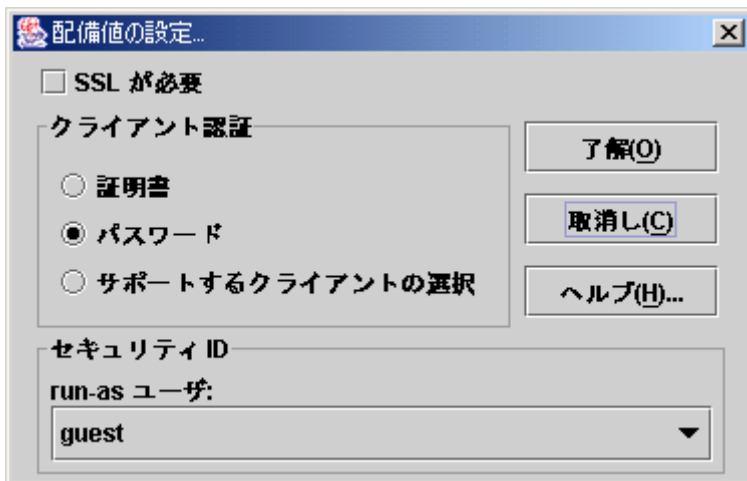


3. 「セキュリティ ID」パネルで「run-as ロール」オプションを選択する。



4. ドロップダウンメニューから実行するロールを選択する。

5. ロールを選択したあと、そのロールからユーザを選択する。これを行うには、「配備値の設定」を選択する。



6. 「run-as ユーザ」から、クライアントが EJB のメソッドを呼び出す際に使うユーザ名を選択する。
7. 「了解」をクリックする。

この作業の結果、ConverterJAR の配備記述子は次のように自動的に設定される。その抜粋を示す。配備記述子のすべてを閲覧したい場合は、ConverterJAR にフォーカスして右クリックで「記述子ビューア」を選択する。

```
<ejb-jar>
<display-name>ConverterJAR</display-name>
<enterprise-beans>
  <session>
    ...
    <security-identity>
      <description></description>
      <run-as>
        <description></description>
        <role-name>guest</role-name>
      </run-as>
    </security-identity>
  </session>
</enterprise-beans>
<assembly-descriptor>
  <security-role>
    <role-name>guest</role-name>
  </security-role>
  ...
</assembly-descriptor>
</ejb-jar>
```

コンポーネントが実行中でないセキュリティアイデンティティを伝達するコンポーネン

トを設定では、配備記述子の<security-identity>要素の中で、<run-as>オプション要素で定義されている。

9.2.2. まとめ

J2EE SDK の `deploytool` ツールを使用してコンポーネントの伝達されたセキュリティアイデンティティの設定を行うことができる。直接、配備記述子を編集してもかまわないが、その際、配備記述子のタグの意味を十分理解して記述子ないとセキュリティ上の脆弱性を作ることになる。

9.2.3. 関連記事

- 9.1.1. セキュリティアイデンティティの伝達
- 9.3.1. クライアント認証の設定
- 9.4.1. コンテナ間の信頼

9.2.4. 参考文献

- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Enterprise JavaBeans Specification, Version2.0
ftp://ftp.java.sun.com/pub/ejb/947q9tbb/ejb-2_0-fr2-spec.pdf
- J2EE チュートリアル, S・ポドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版,
Rich Grenn 他 著, 株式会社ピアソン・エデュケーション

9.3. クライアント認証の設定

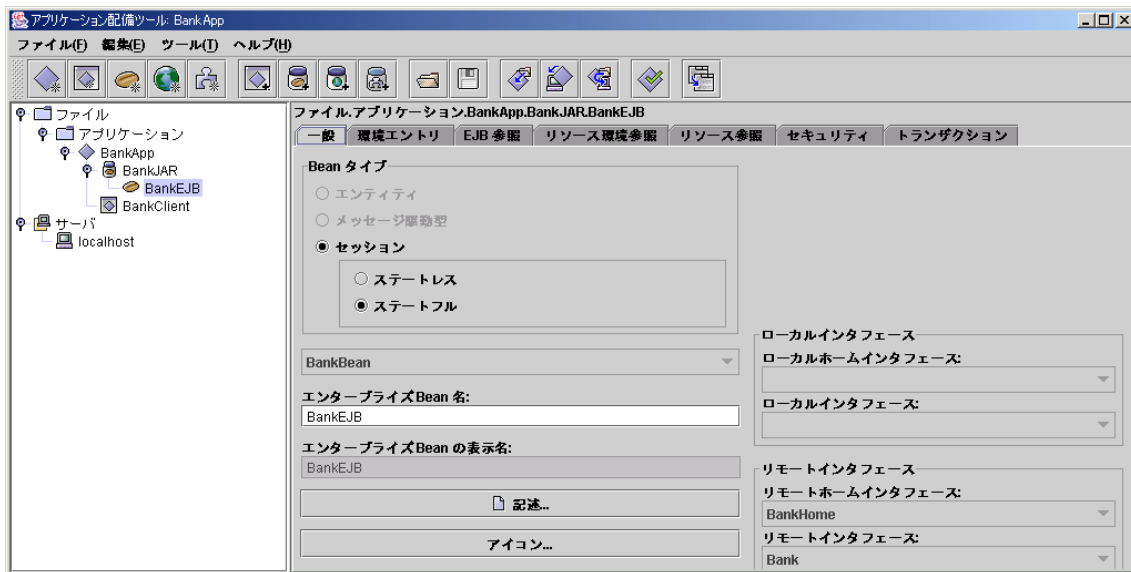
クライアント認証の設定を J2EE SDK に付属している `deploytool` を使用して設定することについて述べる。

9.3.1. クライアント認証の設定

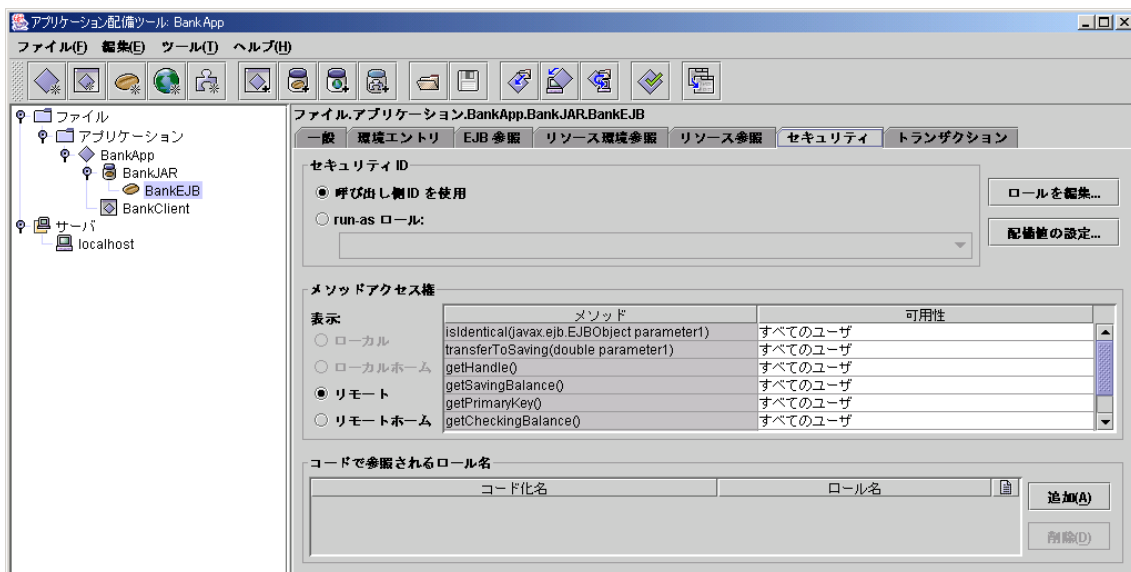
アプリケーションクライアントコンテナのアプリケーションコンポーネントが、Bean の保護されたメソッドにアクセスする場合は、クライアント認証を使用する。

クライアント認証を設定するには、配備ツールを使用して次の手順を行う。

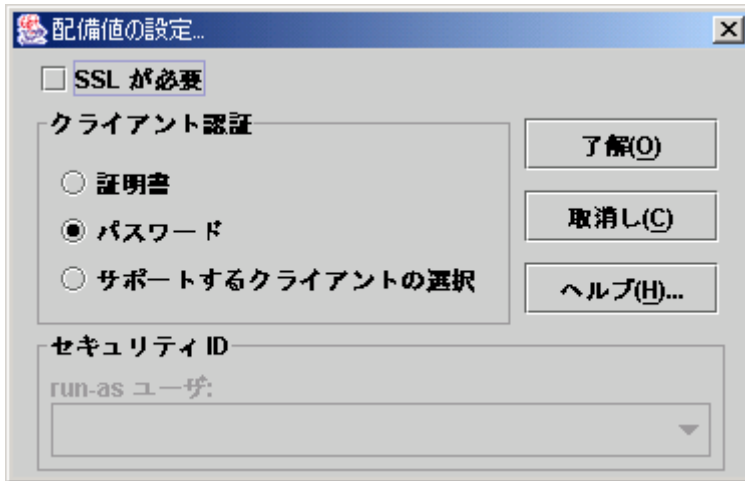
1. 設定する EJB を選択する。



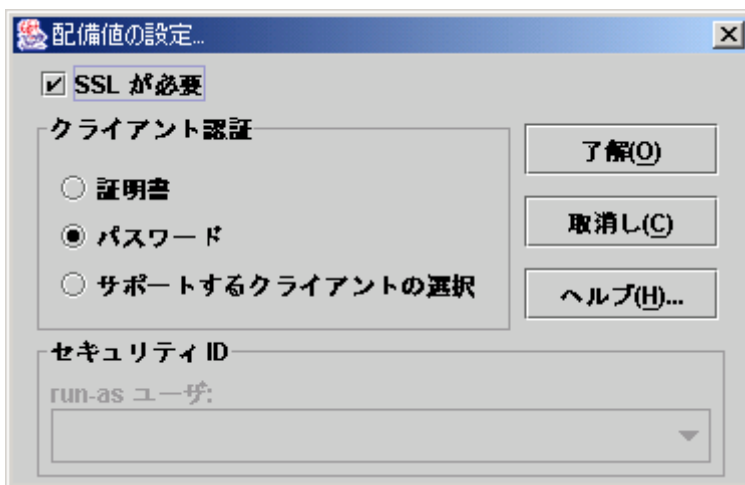
2. 「セキュリティ」タブを選択する。



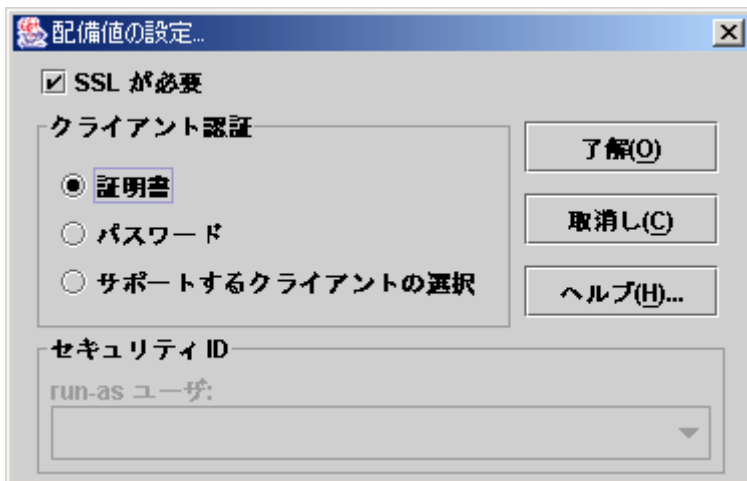
3. 「配備値の設定」を選択して、「配備値の設定」ダイアログボックスを表示する。



4. 「SSL が必要」チェックボックスを選択して、SSL を有効にする。



5. 「クライアント認証」パネルで、認証方法として「証明書」を選択する。これにより、クライアント自身がサーバに認証される。



6. 「了解」をクリックする。

9.3.2. まとめ

?クライアント認証を設定する場合、X.509 デジタル証明書の取得、設定、管理、運用に十分な注意をすべきである。特に X.509 デジタルの証明書の期限など十分に気をつける必要がある。

9.3.3. 関連記事

- 9.1.1. セキュリティアイデンティティの伝達
- 9.2.1. コンポーネントの伝達されたセキュリティアイデンティティの設定
- 9.4.1. コンテナ間の信頼

9.3.4. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
-

- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1_3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

9.4. コンテナ間の信頼

コンテナ間で伝達されるセキュリティアイデンティティの扱いについて述べる。

9.4.1. コンテナ間の信頼

EJB が呼び出し元のアイデンティティまたは指定されたアイデンティティを使用して、ターゲット Bean を呼び出すように設計されている場合、ターゲット Bean は伝達されたアイデンティティのみを受信する。ターゲット Bean は、認証データを受信しない。ターゲットのコンテナが伝達されたセキュリティアイデンティティを認証する方法はない。しかしセキュリティアイデンティティは、たとえばメソッドアクセス権や `isCallerInRole()` メソッドを用いた認証検査で使用されるので、セキュリティアイデンティティが本物であることは非常に重要である。伝達されたアイデンティティを認証するための認証データがないので、ターゲットは、呼び出し元のコンテナが認証されたセキュリティアイデンティティを伝達したということを信頼しなければならない。デフォルトでは、J2EE SDK サーバは異なるコンテナから伝達されるアイデンティティを信頼するように設定される。従って、信頼関係を設定するための特別な手順はない。

アプリケーションアセンブラは、配備記述子の中のコンポーネントアイデンティティ選択ポリシーを記述する必要がある。アプリケーションアセンブラからアイデンティティ選択ポリシーの特定の記述表現がない場合、デプロイヤは、コンポーネントは呼び出し側のアイデンティティを使用して他のコンポーネントを呼び出すことを前提とする。

9.4.2. まとめ

J2EE SDK サーバは異なるコンテナから伝達されるアイデンティティを信頼するように設定される。従って、信頼関係を設定するための特別な手順はない。

9.4.3. 関連記事

- 9.1.1. セキュリティアイデンティティの伝達
- 9.2.1. コンポーネントの伝達されたセキュリティアイデンティティの設定
- 9.3.1. クライアント認証の設定

9.4.4. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-13-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

10. J2EE のユーザ、レルム、およびグループ

この章では、J2EE におけるユーザ、レルム、グループについて述べる。

10.1. J2EE のユーザ、レルム、およびグループ

J2EE の環境において、ユーザ、レルム、グループを理解することは非常に大切である。この節では、J2EE のユーザとグループの概念とオペレーティングシステムのユーザとグループの違いについて述べる。さらに、J2EE セキュリティの固有の概念であるレルムについて述べる。

10.1.1. J2EE のユーザ

J2EE のユーザは、オペレーティングシステムのユーザに似ている。一般的には、これら両方の種類のユーザは個々人を表す。しかし、J2EE のユーザとオペレーティングシステムのユーザは同一ではない。まったく異なるものだということを認識すべきである。J2EE 認証サービスは、オペレーティングシステムにログオンするときのユーザ名やパスワードの情報を持っていない。J2EE 認証サービスは、オペレーティングシステムのセキュリティ機構に接続されない。このことから J2EE のユーザとオペレーティングシステムのユーザがまったく異なりそれぞれ独立していることを理解しなければならない。

10.1.2. レルム

J2EE 認証サービスとオペレーティングシステムの認証サービスにおいて、異なるレルムに属するユーザを管理する。レルムとは、同じ認証機構により制御されるユーザの集合である。一方、レルムは、J2EE 仕様でセキュリティポリシードメインまたはセキュリティドメインとも呼ばれる。レルムは、共通のセキュリティポリシーがセキュリティサービスのセキュリティ管理者により定義および適用される論理的範囲のことである。多くの J2EE コンテナの実装では、J2EE 認証サービスは、証明書とデフォルトという2つのレルムによりユーザを管理する。レルムの特徴には、ユーザ、グループまたは主体の集合が1つ定義される。そして、ユーザ、グループまたは主体を認証するため明確に定義された認証プロトコルが1つまたは複数使用されることがある。さらに、セキュリティポリシーの設定を簡素化するためにグループが定義される場合がある。証明書は、Web ブラウザクライアントを認証するために HTTPS プロトコルと共に使用される。証明書レルムのユーザのアイデンティティを確認するために、認証サービスは X.509 デジタル 証明書を確認する。段

階的な手順については、「11.2.1. サーバ証明書の設定」を参照すること。X.509 デジタル 証明書の共通名称フィールドは、プリンシパル名として使用され、デフォルトレルムでは、ユーザ名がプリンシパル名として使用される。ほとんどの場合、J2EE 認証サービスはデフォルトレルムの検証によりユーザアイデンティティを確認する。このレルムは、HTTPS プロトコルおよび証明書を使用する Web ブラウザクライアントを除いた、すべてのクライアントの認証に使用される。デフォルトレルムの J2EE のユーザは、J2EE グループに属することができるが証明書レルムのユーザはできない。

10.1.3. J2EE のグループ

J2EE のグループは、肩書きや顧客のプロファイルのような共通の特性により分類されたユーザのカテゴリである。たとえば、電子商取引アプリケーションのほとんどの顧客は、CUSTOMER グループに属する。しかし、大口顧客は PREFERRED グループに属する。ユーザをグループへ分類すると、膨大な数のユーザアクセスの制御がより簡単になる。「6. EJB 層のセキュリティ」では、EJB へのアクセスをどのように制御するかを説明している。

10.1.4. J2EE グループのレルムと J2EE アプリケーションロールの混同による不具合

メソッドのアクセス制限とロールマッピングを定義するときは、グループのレルムと J2EE アプリケーションロールが混同されがちである。このような混乱は、予期せぬアクセスや、動作不可能なアプリケーション設定につながりかねないので注意すること。

10.1.5. まとめ

レルムは、J2EE 仕様でセキュリティポリシードメインまたはセキュリティドメインとも呼ばれる。レルムは、共通のセキュリティポリシーがセキュリティサービスのセキュリティ管理者により定義および適用される論理的範囲のことである。

レルムの特徴には、ユーザ、グループまたは主体の集合が1つ定義される。そして、ユーザ、グループまたはプリンシパルを認証するために明確に定義された認証プロトコルが1つまたは複数使用されることがある。さらに、セキュリティポリシーの設定を簡素化するためにグループが定義される場合がある。

10.1.6. 関連記事

- 10.2.1. J2EE のユーザとグループの管理

10.1.7. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

10.2. J2EE のユーザとグループの管理

この節では J2EE ユーザとグループの管理を J2EE SDK に付属している `deploytool` で行うことについて述べる。

10.2.1. J2EE のユーザとグループの管理

J2EE SDK に付属している GUI ベースの `deploytool` または、CUI ベースの `realmtool` を使用して次のことが行える。

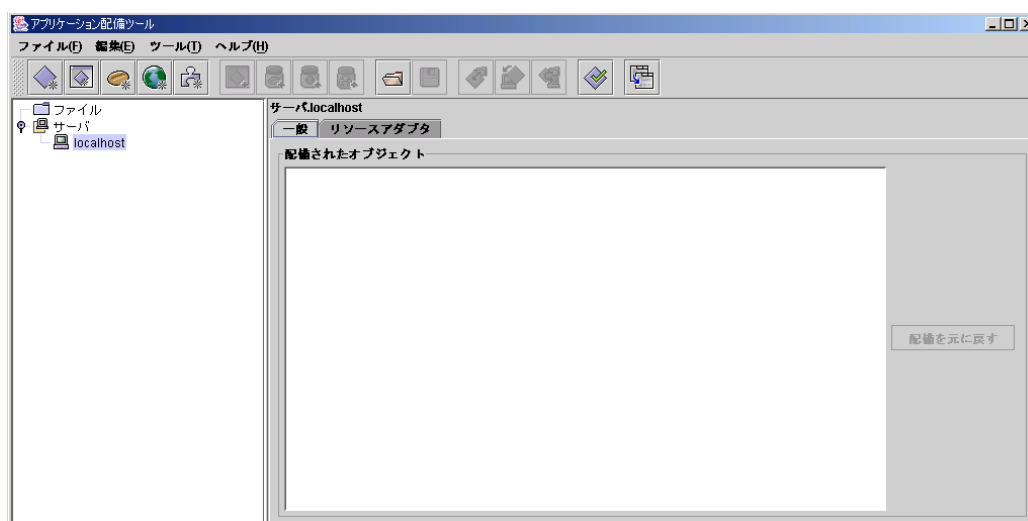
- デフォルトレルムのすべてのユーザを表示する。
- ユーザをデフォルトレルムに追加する。
- ユーザを証明書レルムに追加する。

- ユーザを削除する。
- グループをデフォルトレルムに追加する(グループは証明書レルムには追加できない)。
- グループをデフォルトレルムから削除する。

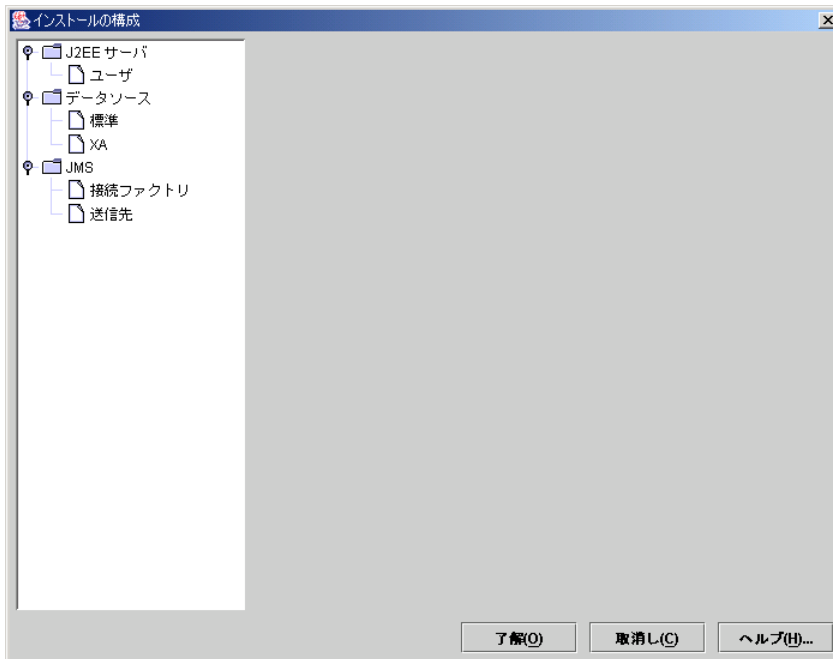
10.2.2. deploytool を使用したユーザとグループの管理

deploytool を使用したデフォルトレルムまたは証明書レルムのすべてのユーザを表示するには、次の手順を行う。

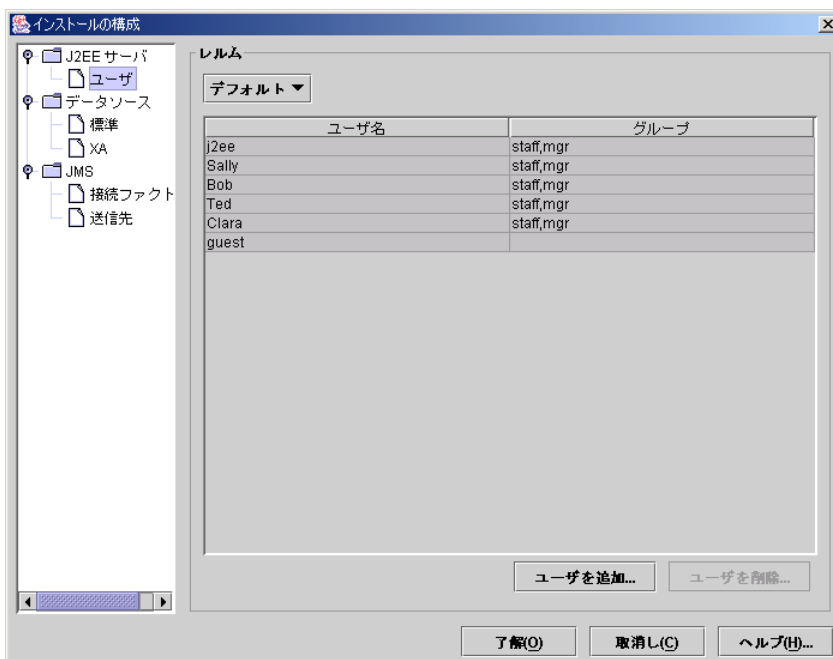
1. ユーザ、グループ、またはその両方を追加するサーバを選択する。



2. 「ツール」 「サーバ構成」を選択して、「インストールの構成」画面を表示する。



3. ツリー表示で、「J2EE サーバ」の下の「ユーザ」を選択する。



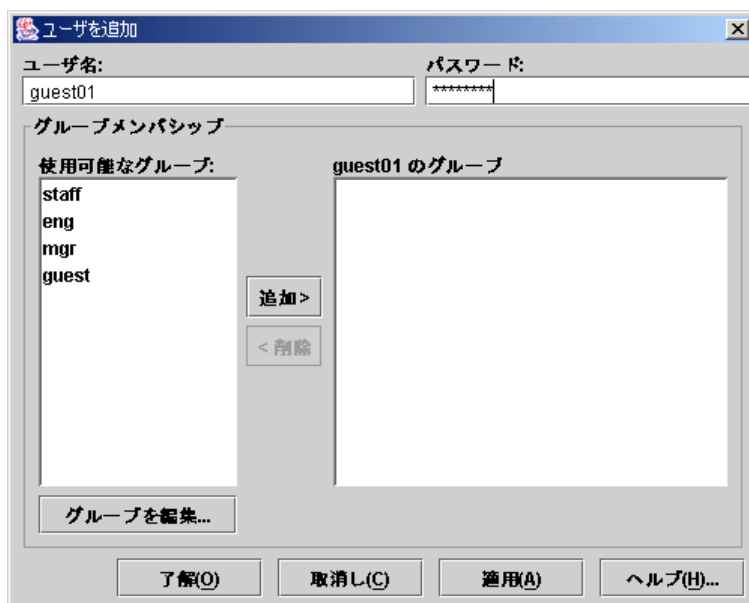
4. レルム(「デフォルト」または「証明書」)を選択する。

ユーザをデフォルトレルムに追加するには、次の手順を行う。

1. 「ユーザを追加」をクリックする。



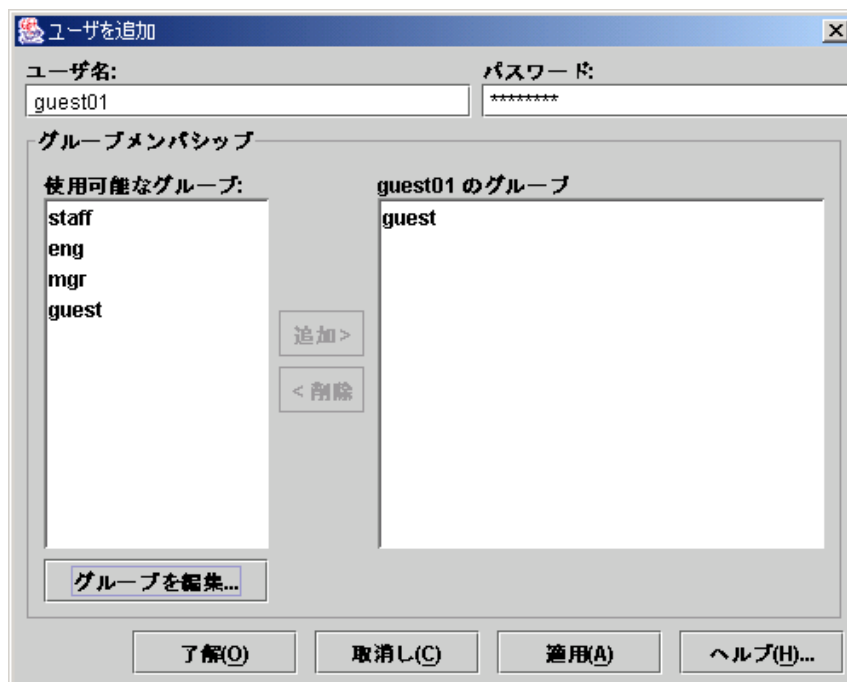
2. ユーザ名とパスワードをそれぞれのフィールドに入力する。



3. 「グループメンバシップ」パネルで、ユーザを追加したいグループを「使用可能なグループ」から選択する。複数のグループを選択するには、この手順を繰り返す。



5. 「追加」をクリックして、選択したグループを対象のユーザのグループの欄に移動する。



5. 完了したら「了解」をクリックする。



新しいグループをデフォルトレルムに追加するには、次の手順を行う。

1. 「グループを編集」をクリックする。



2. 「グループ」ウィンドウで、「追加」をクリックする。



3. 追加された空白行を選択して、グループの名前を入力する。



4. 完了したら「了解」をクリックする。



グループをデフォルトレルムから削除するには、次の手順を行う。

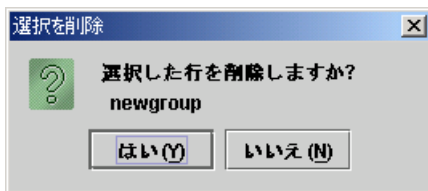
1. 「グループを編集」をクリックする。



2. 「グループ」ウィンドウから、削除するグループを選択する。



3. 「削除」をクリックする。



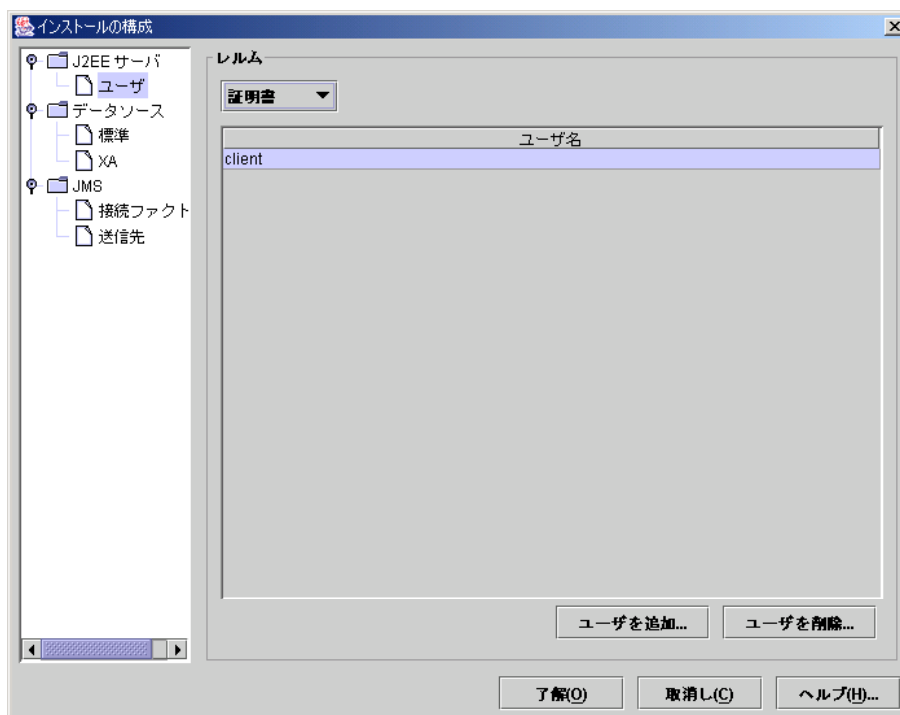
4. 確認ダイアログが表示されたら「はい」をクリックする。

5. 完了したら「了解」をクリックする。



新しいユーザを証明書レルムに追加するには、次の手順を行う。

1. 「証明書」レルムを選択する。

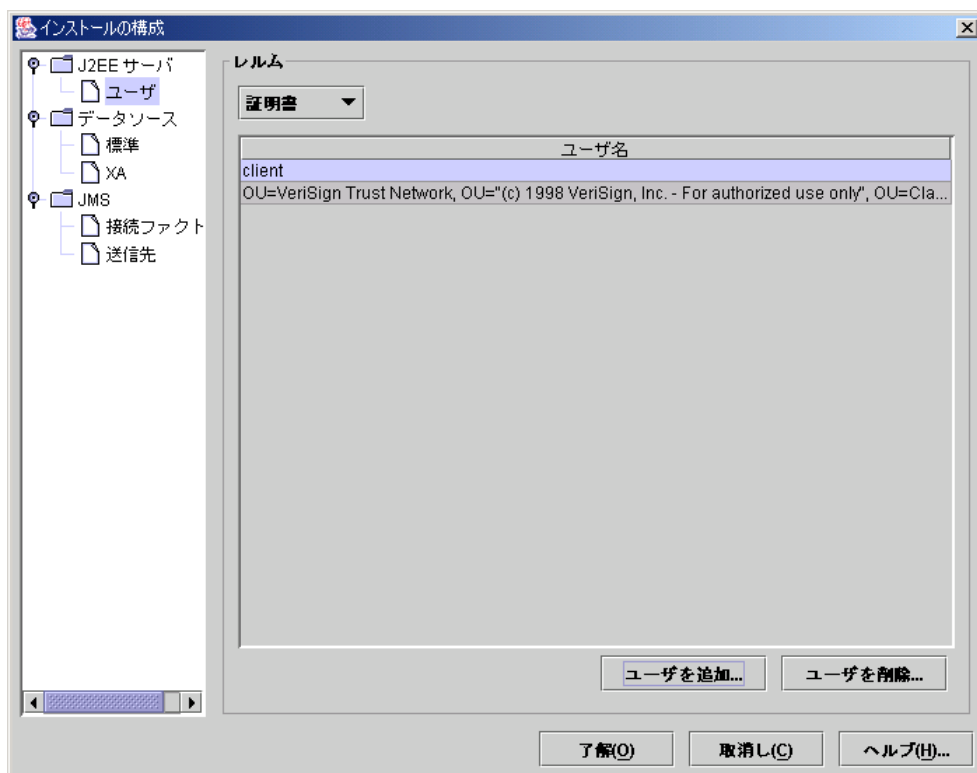


2. 「ユーザを追加」をクリックする。



3. 証明書のあるディレクトリを選択する。

4. 証明書のファイル名を選択する。
5. 完了したら「了解」をクリックする。



10.2.3. realmtool を使用したユーザとグループの管理

J2EE SDK には、GUI ベースの `deploytool` 以外にほぼ、同等の機能を有している CUI ベースの `realmtool` が提供されている。

J2EE SDK の `deploytool` は GUI ベースのツールなので大勢のユーザとグループの追加修正削除作業に向いていない。J2EE SDK には CUI ベースの `realmtool` がある。これを利用すれば大勢のユーザとグループの追加修正削除作業を効率的に行うことができる。

`realmtool` のオプションは次のとおりである。

```
-show
-list          realm-name
-add           username password group[, group]
-addGroup     group
-import       certificate -file -alias alias
-remove       realm -name username
-removeGroup  group
```

10.2.4. deploytool と realmtool の比較

`deploytool` と `realmtool` は機能的にはほぼ同等であるが、`realmtool` は、デフォルトレアルムのグループを一覧表示する機能がない。`deploytool` は J2EE サーバが動作していないと使えないが、`realmtool` は J2EE サーバが動作していなくても動作する。`deploytool` でアプリケーションに変更を認識させるには、J2EE サーバを停止して再起動する必要がある。どちらのツールもユーザのパスワードを変更する機能はサポートされていない。パスワードの変更などは、そのユーザを削除して再作成して、新しいパスワードを定義することになる。

10.2.5. まとめ

J2EE SDK にはユーザとグループの管理をおこなうために GUI ベースの `deploytool` と CUI ベースの `realmtool` が提供されている。

10.2.6. 関連記事

- 4. セキュリティロール
- 10.1.1. J2EE のユーザ

10.2.7. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-13-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

11. サーバ証明書の設定

この章では、J2EE サーバにおけるサーバ証明書の設定について述べる。

11.1. X.509 デジタル証明書

X.509 デジタル証明書について述べる。

11.1.1. X.509 デジタル証明書とは

デジタル証明書は、ITU の X.509 国際規格によって規程された。公開鍵やユーザ、プログラム、会社、個人など公開鍵を持たせられるあらゆるものすなわちエンティティに関する情報を表すための書式を定義している。デジタル証明書は、サーバが受信者の公開鍵を使ってデータを暗号化できるように、しばしばデータ要求とともにサーバに送られる。暗号化されたデータは公開鍵システムのいうところの秘密鍵により復号化される。デジタル証明書は、その正当性を証明する認証局(CA:Certificate Authority)により署名されていなければならない。署名付のデジタル証明書には、認証局の秘密鍵を使い暗号化された証明書のメッセージダイジェストが含まれる。証明書の受信者は、認証局の公開鍵を使ってメッセージダイジェストを復号化し、証明書の本体部分が破損または改竄されたりしていないかを確認できる。デジタル証明書は、認証、機密性、否認防止の確保に利用できる。この節では、SSL においてサーバの認証に使用される。

11.2. J2EE サーバ証明書の設定

この節では J2EE サーバの HTTP over SSL で使用されるサーバ証明書の設定について述べる。

11.2.1. サーバ証明書の設定

証明書は、Web クライアントを認証するために、HTTPS プロトコルと共に使用される。サーバ証明書がインストールされていない場合、J2EE サーバの HTTPS サービスは起動しない。J2EE サーバ証明書を設定するには、J2EE SDK に付属している keytool ユーティリティで次の手順を行う。

1. 鍵ペアと自己署名付き証明書を生成する。

keytool ユーティリティを使用して証明書を作成することができる。J2EE SDK の keytool ユーティリティは、J2SE ソフトウェアの keytool をラップしている。しかし J2EE SDK バージョンには、RSA アルゴリズムを実装する Java Cryptographic Extension プロバイダがプログラムによって追加されている。このプロバイダを使用すると、RSA 署名付き証明書をインポートすることができる。

証明書を生成するには次のように keytool を実行する。<certificatealias> は証明書の別名で、<keystore-filename> はキーストアファイル名を表す。

```
keytool -genkey -keyalg RSA -alias <certificate-alias>
        -keystore <keystore-filename>
```

2. keytool ユーティリティは、次の情報を表示する。

- a. キーストアパスワード: パスワードを入力する (J2EE SDK キーストアのデフォルトパスワードと同じように、「changeit」を使用するのも良い)。
- b. 姓および名前: サーバの有効な名前を入力する。この有効名は、ホスト名およびドメイン名を含んでいる。
- c. 組織単位: 適切な値を入力する。
- d. 組織: 適切な値を入力する。
- e. 都市または地域: 適切な値を入力する。
- f. 州または県: 名前を省略せずに入力する。
- g. 2 文字の国コード: USA の国コードは「US」である。
- h. 別名の鍵パスワード: ここではパスワードを入力してはいけない。
リターンキーを押す。

3. 証明書のインポート

証明書が VeriSign 以外の CA により署名される場合、CA 証明書をインポートしなければならない。VeriSign を使用する場合は、この手順は省略することができる。証明書が VeriSign Test CA により署名される場合は、インポートする必要がある。証明書をインポートするには、次の手順を実行する。

- a. CA に CA 証明書を要求する。その証明書をファイルに格納する。
- b. Java 2 Platform, Standard Edition に CA 証明書をインストールするには、次のように keytool ユーティリティを実行する。\$JAVA_HOME/jre/lib/security/cacerts

ファイルを修正するための権限が必要である。

```
keytool -import -trustcacerts -alias <ca-cert-alias>
-file <ca-cert-filename>
```

4. CA によるデジタル署名付き証明書が必要な場合、次の手順を行う。

a. Certificate Signing Request (CSR) を生成する。

```
keytool -certreq -sigalg MD5withRSA -alias <cert-alias>
-file <csr-filename>
```

b. 署名のために<csr-filename> のコンテンツを送信する。VeriSign CA を使用する場合は、<http://digitalid.verisign.com/> へ行く。VeriSign から、署名付き証明書が電子メールで送信される。この証明書を、ファイルに格納する。

c. 電子メールで受け取った署名付き証明書をサーバへインポートする。

```
keytool -import -alias <cert-alias> -file <signed-cert-file>
```

11.2.2. まとめ

X.509 デジタル証明書の作成、管理、運用を J2EE SDK に付属している keytool を使用する。

11.2.3. 関連記事

- 5.7. クライアント証明書認証
- 5.8. SSL の使用による HTTP 基本認証とフォームベース認証の機密性の拡張

11.2.4. 参考文献

- J2EE チュートリアル, S・ポドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー

- プロフェッショナル EJB, Rahim Adatia 他著,インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発、Subrahmanyam Allamaraju 他著、インプレス社
- Java セキュリティ,Scott Oaks 著,株式会社オライリー・ジャパン
- X.500 ディレクトリ入門,大山実 他著,東京電機大学出版局
- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1-3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 セキュリティ管理者ガイド
<ftp://docs-pdf.sun.com/816-6482/816-6482.pdf>
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

12. 監査

J2EE サーバすなわち Java アプリケーションサーバは、サーバシステムである。従って、安全かつ安定して動作させるためにはログを記録し監査する必要がある。この章では、J2EE SDK におけるログ記録機構の概念とそのログの監査について述べる。

12.1. 監査

この節では、J2EE における監査機構について述べる。また J2EE SDK においてログを記録し監査することを考察する。さらに、他のベンダーの EJB サーバの EJB セキュリティ監査機能についても述べる。

12.1.1. 監査

一般的に、監査とは、ユーザまたはシステムが動作に対する責任を保持できるように、セキュリティに関連するイベントの記録を採ることである。監査の価値は、セキュリティ機構がアクセスをあるシステムに限定するかどうかを決定することだけではない。セキュリティが侵害された場合は通常、誰がアクセスを許可されていないかよりも、誰がアクセスを許可されていることを知ることが大切である。誰がシステムにアクセスしたかを知ることにより、セキュリティの侵害に対する責任を明確にすることができる。さらに、セキュリティのシステムへの影響を評価するために監査を使用する際に、何が監査されて何が監査されていないかを正確に把握する必要がある。

J2EE において、デプロイヤは、エンタープライズコンテナに対して適用されるセキュリティ機構を構成する必要がある。構成された各セキュリティ機構は、コンテナがコンポーネント間の相互作用に適用を試みる制約と考えることができる。デプロイヤまたはシステム管理者は、プラットフォームに対して確立されたセキュリティ制約を確認し、コンテナが次のどれか1つを監査するように、各制約と監査動作を関連付けることができる。

- 制約を満たしていた評価のすべて
- 制約を満たしていなかった評価のすべて
- 結果に依存しない評価のすべて
- 評価なし

給与計算を行う J2EE アプリケーションにおいて給与データベースに対するアクセスロ

ログをの記録などが例として考えられる。このログを検証することにより、承認されたユーザだけが給与データベースを更新できるように正しく設定されているかを確認することができる。もし、ログからそれを確認できない場合、デプロイヤーは、エンタープライズコンテナに対して適用されるセキュリティ機構を構成になんらかの間違があることが推測される。

プラットフォームによって提供される監査構成、または制約へのすべての変更(配備またはその後の管理の結果)も慎重に監査する必要がある。攻撃者が有罪を立証されうような記録をシステムから削除したり、その内容を改竄したりして責任を回避できないように監査記録は保存および保護される必要があるのはいうまでもない。

12.1.2. 収集されたログからの監査

J2EE プログラミングモデルでは、監査責任を開発者や統合者からアプリケーションの配備と管理の担当者に移行する。従って、現在の J2EE 仕様では必須ではないが、コンテナが適用するセキュリティポリシーの評価を容易にする監査機能を J2EE コンテナで提供することを推奨する。実際には、多くのアプリケーションサーバは該当する機能を実装している。たとえば、J2EE SDK においては、次の手順で監査に必要なログを出力させることができる。

12.1.3. J2EE SDK におけるログ機構と監査機構

J2EE SDK サーバ情報を記録するログファイルには監査(audit.log)、エラー(error.log)、出力(output.log)の3種類がある。これらのログファイルは、%J2EE_HOME%logs/CopmputerName/j2ee/j2ee ディレクトリに置かれる。エラーファイルには非常に監査を行う際に、非常に役立つ情報が記録される。また、J2EE サーバは、配備や実行時の誤動作に関する情報を含むファイルを生成することがある。通常、これらのファイルは temp ディレクトリに保存される。この特殊なエラーが発生すると、deploytool は該当するファイルの名前と場所を表示する。ログファイルには、バグや実装に関連する誤った設定を突き止めるのに役立つ重要な情報が含まれているため、必ず確認すべきである。さらに、J2EE SDK のサーバ起動において、-verbose オプションを指定する。

```
%j2ee -verbose
```

このオプションの指定ですべてのログ出力を j2ee を起動したシェルにリダイレクトする。このログ出力からサーバの様々な稼動状況を知ることができる。

12.1.4. 他のベンダーの EJB サーバの EJB セキュリティ監査に関する実装

J2EE SDK サーバのログ機構について説明したが、EJB サーバベンダーは、その EJB サーバ固有のログ機構や監査機構を実装していることが多い。たとえば、ある EJB サーバのベンダーは、セキュリティクリティカルなイベントを監査する手段を提供する。セキュリティクリティカルなイベントには、`java.security.Exception` 例外の生成、認証試行の成功および失敗、EJB アクセス試行の失敗のロギングなどがある。たとえばあるベンダーのアプリケーションサーバでは、ベンダー独自の `AuditProvider` 監査クラスを実装することができる。この場合、そのアプリケーションサーバのプロパティファイルの `AuditProvider` に関する特殊なプロパティを、`AuditProvider` クラス名に設定する。この設定により認証試行が発生したり、認証要求が行われたり、無効なユーザ証明書がサーバに伝播されたりすると、このアプリケーションサーバは、`AuditProvider` クラスでメソッドを呼び出す。

12.1.5. まとめ

J2EE プログラミングモデルでは、監査責任を開発者や統合者からアプリケーションの配備と管理の担当者に移行する。従って、現在の J2EE 仕様では必須ではないが、コンテナが適用するセキュリティポリシーの評価を容易にする監査機能を J2EE コンテナで提供することを推奨する。実際には、多くのアプリケーションサーバは該当する機能を実装している。

12.1.6. 関連記事

- 2. J2EE セキュアプログラミング概要

12.1.7. 参考文献

- J2EE チュートリアル, S・ボドフ 他 著, 株式会社ピアソン・エデュケーション
- Java2 Platform, Enterprise Edition アプリケーション設計ガイド第2版, Rich Grenn 他 著, 株式会社ピアソン・エデュケーション
- 標準 J2EE テクノロジー 2, Martin Bond 他 著, 株式会社翔泳社
- EJB コンポーネント開発完全ガイド, Pranvin V. Tulachan 著, 株式会社アスキー
- プロフェッショナル EJB, Rahim Adatia 他著, インプレス社
- プロフェッショナル Java サーバプログラミング J2EE の設計と開発, Subrahmanyam Allamaraju 他著, インプレス社

- J2EE チュートリアル
http://sdc.sun.co.jp/NASApp/sdcpersonal/private/jdc/download/j2ee_tutorial/j2ee-1_3-doc-tutorial-j.pdf
- Sun ONE Application Server 7 開発者ガイド
<ftp://docs-pdf.sun.com/817-0602/817-0602.pdf>
- Sun ONE Application Server 7 Enterprise JavaBeans 開発者ガイド
<ftp://docs-pdf.sun.com/817-0605/817-0605.pdf>
- Sun ONE Application Server 7 Web アプリケーション開発者ガイド
<ftp://docs-pdf.sun.com/817-0604/817-0604.pdf>

● 13. チェックリスト

本資料で取り上げたセキュリティ脆弱性と対策に関するチェックリストを示す。各項目の末尾の「 n.n」は、関連する記事の番号を示す。

- 1) アプリケーションのセキュリティを設計、実装するタスクを EJB コンテナに委譲することで、Bean の開発者をビジネスロジックに専念させている。 3.1
- 2) セキュリティポリシーを宣言によって設定し、Bean クラスでセキュリティポリシーをハードコーディングすることを避けている。 3.1
- 3) セキュリティポリシーを宣言によって設定し、作成した EJB アプリケーションは複数の EJB サーバへの移植性を確保できている。 3.1
- 4) セキュリティロールを利用して、コンポーネントに対する適切なアクセス権を定義している。 4.1
- 5) エンタープライズ Bean や Web コンポーネントからセキュリティロールを参照する場合、セキュリティロール参照を宣言している。 4.2
- 6) セキュリティ制約の指定により、Web リソースを保護している。 5.1
- 7) Web リソースに対して、目的に応じた適切な認証機構を指定している。 5.3
- 8) HTTP 基本認証を使用する場合、SSL を併用してユーザ名とパスワードを保護している。 5.5, 5.8
- 9) フォームベース認証を使用する場合、SSL を併用してユーザ名とパスワードを保護している。 5.6, 5.8
- 10) クライアント証明書認証の使用に際して、HTTP 基本認証およびフォームベース認証よりも運用コストがかかることを考慮している。 5.7
- 11) SSL を使用して、かつ多数のクライアントシステムに対応しなければならない場合、SSL アクセラレータのようなハードウェアを導入するなどして、SSL 使用によるシステムへの負荷を分散している 5.8
- 12) Web リソースの認証機構の設定には、J2EE SDK 付属の配備ツールなどを使用して省力化をはかる。 5.9
- 13) Web 層のセキュリティに関して、宣言によるセキュリティだけでアプリケーションのセキュリティモデルを十分に表せない場合、プログラミングによるセキュリティを併用する。 5.10
- 14) プログラミングによるセキュリティの使用は、最小限におさえている。 5.10
- 15) エンタープライズ Bean のメソッドアクセス権を定義して、意図しないロールからのメソッド呼び出しを制限する。 6.2

- 16) EJB 層のセキュリティに関して、宣言によるセキュリティだけでアプリケーションのセキュリティモデルを十分に表せない場合、プログラミングによるセキュリティを併用する。 6.3
- 17) アプリケーションクライアント層でのセキュリティには JAAS を使用する。 7.1
- 18) EIS への認証処理で使用するユーザ名とパスワードの管理や、それらを扱うプログラムの実装には十分な注意を払う。 7.1
- 19) EIS への認証処理をコンポーネント管理で行う場合、`getConnection` メソッドを通じて受け渡すプロパティ(ユーザ名、パスワード等)はクライアント固有とする。ターゲットの EIS インスタンスの構成には関連しない。 8.3
- 20) リソースアダプタを使用する場合、認証機構を設定している。 8.4
- 21) セキュリティ要件において可能な限り、呼び出し元の間コンポーネントのアイデンティティが、ターゲット EJB へ伝達される方法を選択している。 9.1
- 22) クライアント認証を設定する場合、X509 証明書の取得、設定、管理、運用には十分な注意を払っている。 9.3
- 23) Java アプリケーションサーバが安全かつ安定して動作していることを確認するため、J2EE SDK におけるログ記録機構を使用している。 12.1