

Authorization Frameworks and Mechanisms -WG

Markus Lorch (Editor), Virginia Tech
Category Informational Bob Cowles (Co-Editor), Stanford Linear Accelerator Center
Authorization Frameworks and Mechanisms -WG Rich Baker, Brookhaven National Laboratory
Leon Gommans, University of Amsterdam
Paul Madsen, Entrust
Andrew McNab, University of Manchester
Lavanya Ramakrishnan, CNIDR/MCNC
Krishna Sankar, Cisco Systems Inc.
Dane Skow, Fermi National Accelerator Laboratory
Mary R. Thompson, Lawrence Berkeley National Laboratory

作成日： 2003/02/17

改訂日： 2004/11/23

概念的グリッド認可フレームワークおよび分類

本覚書の位置付け

本覚書は、グリッドのセキュリティと認可に関する情報をグリッド・コミュニティに提供するものである。本文書の配布に制限はない。

著作権に関する注意

グローバル・グリッド・フォーラム（2004）がすべての著作権を有する。

概要

本文書は分散システムにおける認可に関するトピックについて特にグリッド環境のニーズや要件に特化して紹介するものである。ここでは概念的なグリッド認可フレームワークを規定し、既存および提案中の認可メカニズムを本フレームワークに従って分類する。このフレームワークは、今後のAPI設計や標準化作業の基本となることを意図している。

目次

概要

1. はじめに
2. 認可フレームワークの概念
 - 2.1 認可に関連する基本的なエンティティ
 - 2.2 認可シーケンス

- 2.2.1 認可プッシュ・シーケンス
- 2.2.2 認可プル・シーケンス
- 2.2.3 認可エージェント・シーケンス
- 2.2.4 ハイブリッド認可シーケンス・モデル
- 2.3 ドメインの考慮
- 2.4 契約関係および信用関係
- 2.5 認可ポリシーおよび認可属性
- 3. 認可アーキテクチャ
 - 3.1 概要
 - 3.2 認可機能
 - 3.3 認可情報の流れ
 - 3.3.1 属性情報の流れ
 - 3.3.2 認可要求および応答
 - 3.3.3 ポリシーの流れ
 - 3.4 ポリシー・サブシステム
 - 3.4.1 ポリシー表現
 - 3.4.2 ポリシー・リポジトリ
 - 3.4.3 ポリシー交換
 - 3.4.4 ポリシー処理
- 4. フレームワーク・コンポーネント
 - 4.1 信用管理
 - 4.1.1 信用機関
 - 4.1.2 信用関係の定義
 - 4.2 特権管理
 - 4.2.1 属性機関
 - 4.2.2 特権割り当て
 - 4.2.3 属性管理
 - 4.3 ポリシー管理
 - 4.4 認可コンテキスト
 - 4.5 認可サーバ

- 4.6 エンフォースメントメカニズム
 - 4.6.1アプリケーション依存エンフォースメントメカニズム
 - 4.6.2アプリケーション独立エンフォースメントメカニズム
- 5. 既存認可メカニズム、モジュール、システムの分類
 - 5.1 Akenti認可サービス
 - 5.1.1モデルおよびアーキテクチャ概要
 - 5.1.2認可表明機能およびポリシー表明機能
 - 5.1.3認可情報の流れ
 - 5.1.4信用管理
 - 5.1.5エンフォースメントメカニズム
 - 5.2 Cardea
 - 5.2.1認可情報
 - 5.2.2認可判断の起動と実施
 - 5.3 CAS
 - 5.4 PRIMA
 - 5.4.1認可シーケンス
 - 5.4.2エンフォースメントメカニズム
 - 5.4.3判断機能
 - 5.4.4属性アサーション機能とポリシーアサーション機能
 - 5.5 PERMIS認可インフラストラクチャ
 - 5.5.1認可フレームワーク
 - 5.5.2認可情報の流れ
 - 5.5.3ポリシー問題
 - 5.5.4信用管理
 - 5.6 EU DataGridセキュリティ・アーキテクチャ
 - 5.6.1 VOMS属性権限
 - 5.6.2認可判断機能
- 6. 関連標準
 - 6.1 ポリシー仕様
 - 6.2 認可コンポーネント間の通信

6.3 関連認可フレームワーク

7. セキュリティ上の注意点

著者の連絡先

謝辞

用語

知的所有権の記述

著作権表示全文

参考文献

付録A: 2つのドメイン認可モデルの分類

1. はじめに

本文書はグリッド・コミュニティに情報を提供するものである。本文書では認可の基本的な概念とモデルを紹介し、グリッドの概念的な認可フレームワークを規定し、既存および提案中の認可メカニズムをこのフレームワークに沿って分類する。

我々はここで一般的に前提としているものとベスト・プラクティスを捕らえようと試みているのであって、読者諸氏におかれては、本文書は標準を定義するものでもなく、標準を定めるものではないという点を念頭におかれたい。

認可とはどちらかという多くの意味を暗示するあいまいな用語であると考えられている。これは、認可が発行、転送、表示、検証、委譲、無効化などといった要素を含んでいるためである。たとえば認可は、

- ・ ユーザーの特性や特権を規定する属性集合(特権管理)
- ・ 資源へのアクセスを許可する判断の基礎としてのポリシー集合(アクセス制御)
- ・ 資源へのアクセス権限を表明したデジタル署名付の文書

と表現されることがある。

認可には数多くの処理とエンティティが関係している。本文書では、さまざまな認可の概念とそれらの関係を取り扱い、任意のタイプの認可システムを分類できるような一般的で抽象的なフレームワークを規定する。抽象的なエンティティや機能を定義し、認可要求や判断のための基本的な通信シーケンスを示す。本文書の最後では、既存の認可システムを本フレームワークで紹介した概念にマッピングし、関連する標準やプロトコルの概要を述べる。

本文書では、特に認可に関係がないと思われる概念について触れる場合は、他の文書を参照する。たとえば、信憑性、完全性、機密保持などといったセキュリティに関連する概念は本文書で触れるが、ごく簡単に説明するにとどめる。

本文書の以後の部分は次のように構成されている。第2節では、基本的なエンティティ認可シーケンス、および関連するトピックを列挙することで認可フレームワークの概念を紹介する。第3節では、認可アーキテクチャ全体を規定し、第4節では一般的な認可フレームワーク・コンポーネントについて詳説する。第5節ではグリッド環境用に開発されている、あるいは使用されている既存の認可メカニズム、モジュール、システムを分類し、第6節では結びとして関連する認可標準をまとめている。付録Aには、2.3節で紹介する2つのドメイン認可スキーマの詳細な分類が記載されている。

2. 認可フレームワークの概念

本節では、認可に関連するさまざまな基本的概念を説明し、それらをフレームワークにまとめる。このフ

フレームワークはグリッド・コンピューティング環境で現在あるいは今後想定される利用ケースに合わせており、IRTF AAAアーキテクチャ研究グループのRFC2904 [RFC2904]に提示されている認可フレームワークとRFC2903 [RFC2903]に提示されている汎用AAAアーキテクチャ、およびISO勧告に記載されているアクセス制御フレームワーク、ISO/IEC 10181-3:1996情報テクノロジー - オープン・システム相互接続 - オープン・システムのためのセキュリティ・フレームワーク: アクセス制御フレームワーク [ISO10181]に基づいている。これらの関連する文書で紹介されている概念や用語は、グリッド・コンピューティングの領域に統合・適用されて、グリッド・コンピューティング環境の要件に特に応えるために必要な機能に注力したグリッド固有の認可フレームワークを作り上げている。

本文書では、認可という用語は次のいずれかの意味を持つという事実を考慮する必要がある。

- 1) 権利の証明を発行する処理
- 2) 権利の証明(またはその証明への参照)そのもの自体(すなわち認可トークン)
- 3) 権利の証明をチェックすることで認可判断する処理。たとえばユーザー属性とアクセス制御ポリシーを比較するなど。

認可判断は以下のようにさまざまな場所で行われる。

- ・ サービス・ポイントの入り口(この場合認可はアクセス制御の意味となる可能性がある)
- ・ サービス・ポイント外にある(セントラル)ポイント

こうした混乱を避けるには、認可という用語を使用する際には常にコンテキストを参照することである。

2.1 認可に関連する基本的なエンティティ

原理的には、認可判断は機関が提供する認可情報に基づいて行われる。この機関は認可主体(たとえば認可の発行対象となるユーザーや組織のメンバー)または認可を催促した要求の対象となっている資源(たとえば資源の所有者または管理者)またはその両者と直接関係があるかまたは権限委譲関係がなければならない。この関係は何らかの暗号化手法(すなわち非対称あるいは対称のキー・メカニズム)に基づいた信用メカニズムを使用して実装するか、まったくオフライン(すなわち他の信用された配布メカニズム)で実装されることになる。

これを考慮すると、認可には次の3つの基本的なハイレベルのエンティティの定義が関連していることがわかる。

主体: ある種の権限を行使するために電子認可を要求、受け取り、所有、転送、提示、権限委譲できるエンティティ(たとえば人間やプロセス)。非公式には、主体とはサービスまたは資源の任意のユーザーである。主体は個々のユーザーとして識別される場合もあれば、あるユーザー・グループのメンバー

として識別される場合もある。また主体はユーザーの代わりに行動するプロセスの場合もあり、そのユーザーから権限委譲されたアクセス権限を保有している。主体はその認可がどのように使用されるかを決定するためのポリシーの集合を定義する場合もある。

資源： サービスを提供またはホスティングするシステムのコンポーネントで、その資源に対する権限を持ったエンティティが定義した規則またはポリシーの集合に基づいてサービスへのアクセスを実施することができる。グリッド環境における典型的な資源は、計算サイクルやデータ記憶域が提供するサービス群を介して計算サイクルやデータ記憶域を提供するコンピュータである。資源へのアクセスは資源自体が実施するかまたは資源と要求者の間に立つ何らかのエンティティ(ポリシー実施点、ゲートウェイ)によって実施されるので、資源が認可されていない方法でアクセスされないように保護している。

機関： 電子的な手段による証明の発行、検証、無効を行うことができ、その権限を持った管理エンティティ。発行された電子的な手段の名前付主体(ホルダーとも言う)は特定の権限を行使したり特定の属性をアサーションしたりする権限を与えられている。権限は電子証明書の中に暗示的あるいは明示的に提示されている。ポリシー集合は、機関が確立した契約関係に基づいて、認可がどのように発行、検証されるのかなどを決定する。

一般的に利用されている機関には現在3つの一般的なタイプがある。属性機関は特定の主体が1つ以上の属性と値のペアを持つことを示す属性証明書を発行する。ポリシー機関は資源および資源が提供するサービスに関する認可ポリシーを発行する。認可ポリシーには、特定の主体があるサービスに対して特定の権限を所有していることが示されている。アイデンティティ機関(たとえば公開鍵インフラストラクチャ(PKI)の認証局(CA))は、暗号トークンを主体識別子にマッピングすることを示した証明書を発行する。アイデンティティ機関は認可ではなく認証を有効化するものであるので、本文書の対象外とする。ただし、証明プロセスの結果が認証プロセスへ入力するものとして使用され、主体のアイデンティティが認可判断対象の主体とは別の認可属性となる、という点を覚えておくことは重要である。

認可は以下の3つの別個のプロセスに分けられるのが普通である。

- 1) 人間または組織によってハイレベルの認可ポリシーが定義される
- 2) ハイレベルのポリシーをコンピュータが解釈可能なデジタル表現に実装する
- 3) デジタル表現されたポリシー内容を確認し、ある主体に対して特定の認可を発行するか特定の行動をとるかを決定する

これら3つのエンティティはそれぞれ認可を制御するポリシー集合を実装できる。ポリシー処理機能はプログラムのロジックの中にハードコードされて実装される場合もあれば、柔軟なポリシー言語によって

実装される場合もある。このレベルではこれ以上細かな前提は設けない。

機関になり代わって認可の判断を行い、行使可能なポリシーを確認するコンポーネントを認可サーバと呼ぶ場合がある。

ただし、認可サーバという用語はどちらかというあいまいな意味を持つ用語と考えられている。通常、認可サーバは以下のいずれか(あるいはその組み合わせ)を作成または実行する。

- a. 認可判断。認可判断サーバという用語が使用される場合もある。認可判断は通常ポリシーを評価した結果である。
- b. 認可照合。何らかの形式で表示し、返されたあるエンティティの権限の照合。この権限は別の認可サーバによってなされる別の認可判断の基礎を形成したり、事前になされた(キャッシュされた)認可判断の結果を表現している場合もある。
- c. 他の認可サーバへの認可判断の委譲または委任。

認可サーバという用語を使用する際は注意が必要であり、認可サーバの特定の機能については具体的に述べるべきである。

2.2 認可シーケンス

RFC2904の図1では基本的なエンティティがいくつか認知されていて以下のように呼ばれている。

1. ユーザー
2. ユーザーのホーム組織
3. サービス・プロバイダー
4. AAA(認証、認可、アカウントिंग)サーバ

これらの用語は前述で定義されたエンティティに対して以下のように概念的にマップできる。

1. ユーザー 主体
2. サービス・プロバイダー 資源
3. AAAサーバ 認可機関の代わりとして代理を務めて行動するサーバ (AAAサーバは、本議論の目的のために認証問題および認可問題も提供するが、認可機能だけがマッピングされる。)

上記のマッピングを考慮すると、RFC2904の第3章で定義されている認可シーケンスはここで述べる3つの汎用的なエンティティ間のシーケンスとして認識することができるようになる。

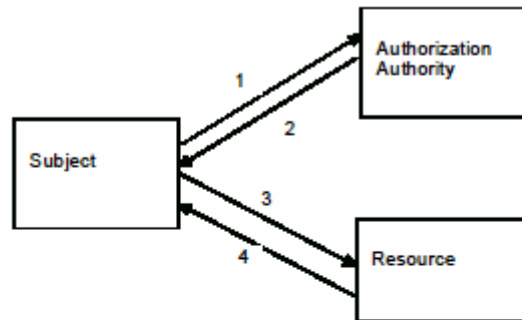


Figure 2.1 Authorization Push Sequence

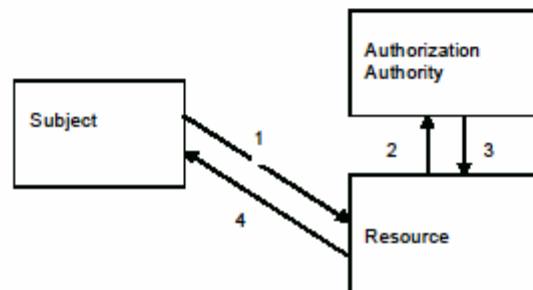


Figure 2.2 Authorization Pull Sequence

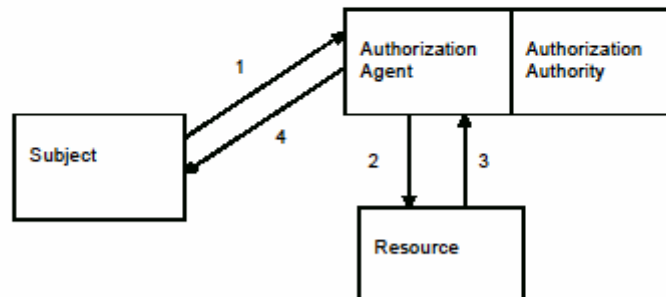


Figure 2.3 Authorization Agent Sequence

2.2.1 認可プッシュ・シーケンス

プッシュ・シーケンスでは、主体はまず(たとえば、認可サーバを経由するなどして)機関からの認可を要求する。機関は主体からの要求を認める場合もあれば認めない場合もある。次に、機関はなんらかのメッセージやセキュアなメッセージ(トークンまたは証明書)を発行したり返したりする場合もあり、それらのメッセージが権利の証明の役目を果たす(認可アサーション)。このようなアサーションには通常、関連し

た有効期間ウィンドウがある。次にアサーションは、主体が資源とコンタクトすることで特定のサービスを要求する際に使用される場合がある。資源はその認可証明を受理したり拒否したりでき、要求した主体にその結果を報告する。このようなシーケンスの例はKerberos [KERBEROS]やKeynote [RFC2704]などのチケット・システムに多く見られる。

2.2.2 認可プル・シーケンス

プル・シーケンスでは、主体は要求を使って資源に直接コンタクトする。サービス要求を許可したり拒否したりするためには、資源は認可機関にコンタクトしなければならない。認可機関は認可判断を下し、認可結果を含むメッセージを返す。次に資源は結果メッセージを返すことで主体に対してサービスを許可または拒否する。このようなシステムの例は、RSVP [RSVP]やRADIUS [RADIUS]プロトコルを使用しているシステムがあるネットワークに見られ、要求は通常「帯域内」で伝送される。グリッド環境においては、このシーケンスはPERMIS [PERMIS]およびAkenti [AKENTI]認可システムによって実装されている。

2.2.3 認可エージェント・シーケンス

主体はエージェント・シーケンスを使用して、サービス認可を取得する要求を出してより上位のレベルのエージェントにコンタクトする。このエージェントは認可機関によって確立された規則に基づいて認可判断を下し、認可に成功すれば資源とコンタクトをとって特定の状態とし、サービスを有効にする。成功したというフィードバックをサービスから受け取ると、エージェントは要求元の主体に成功を報告する。このモデルでは、グリッド・システムから特定のQoSを要求する際(たとえば、スケジューラーによる資源予約)、グリッド・ユーザーとの係わりが生じる。そして主体は資源と直接やり取りをしてサービスにアクセスする。

2.2.4 ハイブリッド認可シーケンス・モデル

2.2に挙げた3つのシーケンスは基礎的なシーケンスである。全ての認可の状況を網羅しているわけではない。ある特定のシーケンスを調べているときに、ここで説明しているフレームワークが上記の3つのシーケンスに完全には合致しない状況になるときもあるであろう。たとえば、プル・モデルとプッシュ・モデルの組み合わせで、主体が事前に管理ドメイン機関から特定のアクセスを要求してその認可を資源へのアクセス要求に対して提供したにもかかわらず(プッシュ)、資源がローカルの管理ドメインにある機関に問い合わせ、そのアクセスがローカルのポリシーに準拠していることを確認する(プル)といった具

合である。こうしたハイブリッドのシーケンスは3つの基本シーケンスの部分に分解できる。

2.3 ドメインの考慮

管理ドメインは機関により定義される。分散型の認可シナリオでは主体と資源の2つの管理ドメインがある場合がほとんどである。グリッド環境では、アイデンティティ、主体属性、資源ポリシー、コミュニティ・ポリシー機関用に別々のドメインがあるというシナリオがほとんどである。単純なグリッドの使用ケースでは、主体はそのホーム・ドメインである1つの管理ドメイン中にあり、資源は別のドメイン(資源のホーム・ドメイン)中にある。もっと高度なシナリオでは、コミュニティあるいは仮想組織(VO)ドメインが存在する。VOドメインはVOの全てのメンバーの特権管理を遂行する機関を提供することができる。典型的なグリッドのシナリオは、主体が複数のドメインのサービスを利用する必要があるというものである。これは、その主体に代わって、ある1つのドメイン中の資源が別のドメイン中の資源を使用することで達成する場合もある。グリッド・サービス・プロバイダーは複数のVOやホーム・ドメイン中のユーザーに対して資源を提供することができる。複数ドメインからなるシナリオに関する詳細な議論については、2つのドメイン認可シナリオの分類について述べた付録Aを参照されたい。

2.4 契約関係および信用関係

認可の承諾と発行を可能にするために、異なる主体、機関、資源のドメイン間の契約関係(ほとんどの場合は法的契約に基づいた合意事項が含まれる)が必要となる場合がしばしばある。こうした契約関係を確立するのはオンラインでもオフラインでもよい。契約関係は、一旦成立すれば信用関係を構築する基礎として使用することができ、この信用関係は異なるドメインのエンティティがどのようにしてお互いの認可を認め合うかを決定する。この信用関係は契約関係と平行している場合もあれば直交している場合もある。たとえば契約は、関連するすべての当事者が独立した第三者を信用しなければならないと規定する場合がある。この関係を示すマッピングが差異を説明するのに役立つ場合がある。このことはRFC2904の第2章に記載されている。

2.5 認可ポリシーおよび認可属性

ポリシー、属性、アイデンティティ、環境パラメータ(たとえば時間)などといった認可情報は、認可判断の際に利用され組み合わせられる。各エンティティはポリシーを使用して要求や応答をどのように処理すべきか決定する。条件やアクションの概念を使用するポリシーがほとんどで、その条件やアクションは実際の要求、要求側主体のアイデンティティ、その主体が保有している属性に従って評価しなければなら

ない。条件やアクションによりメッセージの交換が発生する場合もあり、ポリシーがある程度の柔軟性を持っているのであれば、そのようなメッセージ交換は必ずしも予期できるとは限らない。ポリシーは文字列(s式)で表現することも可能で、他のポリシーと比較される。もしある文字列(要求)がもう一方の文字列(ポリシー)よりも具体的であれば、その要求は認可される。ポリシーは認可メカニズムの中に隠蔽することも可能であるので、あるシーケンスが特定のやり取りの中では起こりえないようにすることもできる。一般的なセキュリティ・コンテキストでは、ポリシーという用語はメッセージ・セキュリティ用の標準、ユーザー識別、文書暗号化要求などといった認可ドメインの外部にあるものを対象としている。本文書でポリシーという用語が別途説明もなく使われる場合、資源アクセスのための規則を定義する認可ポリシーを意味する。

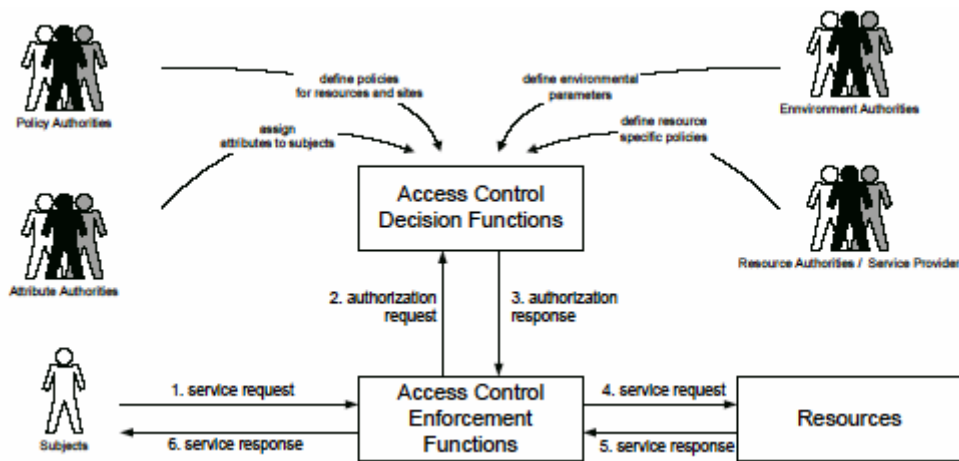
認可属性は、ある資源に対するアクションが許可されたエンティティを暗示的あるいは明示的に定義するエンティティに結び付けられている属性に関する記述である。本文書における属性は、特に断りがない限り、常に認可属性のことを指す。属性は記述属性および特権属性へグループ化することができる。記述属性はエンティティの特性を関連付けるものであるが、特権属性は資源に対するエンティティの適用可能なアクセス権限を直接定義する。記述属性は特性が結び付けられているエンティティに対して特性を伝達するのに使用され、通常は資源、要求、アクションにとらわれず、ポリシーに照らし合わせてアクセス権を与えなければならない。記述属性は、たとえば主体属性や資源属性といったように関係しているエンティティにちなんで名前が付けられることが多い。

3. 認可アーキテクチャ

3.1 概要

認可アーキテクチャはエンティティや機能コンポーネントの集合で構成されており、このエンティティや機能コンポーネントを利用することで、認可条件を定義している属性、パラメータ、ポリシーに基づいて認可判断がなされて実施される。関連する基本的なエンティティについては第2章で述べた。本節では全体的なアーキテクチャをより詳細に説明し、エンティティや機能コンポーネント間の情報交換に焦点をあてる。第2章で紹介したプル・シナリオに基づいた認可システムの概要を図3.1に示す。また同図には認可パラメータ、属性、ポリシーを発行し決定する際に関連するさまざまな機関も示されている。

図3.1プル・シーケンスに基づいた認可アーキテクチャ



プッシュ・シナリオやエージェント・シナリオについても同様の図を描くことができる。こうしたシナリオでは、主体は判断機能に対してサービス要求を送信する。プッシュ・シナリオでは判断機能はサービスへのアクセスを許可するトークンを主体に与える。主体はサービスを要求する際、そのトークンを実施機能へ渡す。エージェントの場合は判断機能に実施コンポーネントがあって、資源の状態を修正するコマンドを発行することで資源を制御する。すなわち判断機能はユーザーの代わりとしてエージェントのように動作することになる。資源はトークン(プッシュの場合)または要求の元(エージェントの場合)の正当性をチェックする思考機能を持っている。プッシュ・シーケンスとエージェント・シーケンスにおいては、主体自体に属性が取り出される機関への参照も含まれており、主体がある程度の自己プロビジョニングを実行できるようになっている。

3.2 認可機能

ISO-101813には2種類のアクセス制御機能が定義されている。

アクセス制御判断機能(ADF: Access Control Decision Function): サービスに対する主体のアクセスに関する認可判断を行う。RFC2904で定義されているポリシー判断ポイント(PDP: Policy Decision Point)と等価である。通常は認可サーバの一部となっていて、資源やアプリケーションからは独立している。ただし、アクセス制御実施機能と同じ場所に設置される場合もある。

アクセス制御実施機能(AEF: Access Control Enforcement Function): 資源やサービスへのアクセスを調停する。RFC2904で定義されているポリシー実施ポイント(PEP: Policy Enforcement Point)と等価である。この機能は、保護している資源の中に統合されているかまたは前に置かれているかのいずれかの場合がほとんどである。

ADF、AEF、主体、資源はさまざまな組み合わせで1つ以上の管理ドメイン内に組み込まれることがある

(2.3節のドメインの考慮を参照方)。その組み合わせについて、2つの管理ドメインの場合を本文書の付録A「2つのドメイン認可モデルの分類」で説明している。取り決めは通常、管理ドメイン間の関係や役割を決定する。管理ドメイン間で信用されて信頼性の高い認可判断の通信を実装するには、具体的なインタフェースやメッセージが必要となる。

3.3 認可情報の流れ

主体、資源、およびさまざまな属性機関間で受け渡す必要のある情報には、属性、ポリシーと認可問い合わせ、応答、の3種類がある。

図3.1に認可システムの情報の流れを単純化した絵を示す。属性、パラメータ、ポリシーはそれぞれに対応する機関によって発行され、認可サーバが利用できるようになっている。認可サーバは、実施機能からの要求に対して、この情報を使用して認可判断を行う。

3.3.1 属性情報の流れ

RFC3281 [RFC3281] (認可のためのインターネット属性証明プロファイル)で指摘されている通り、認可属性がADFに到達するまでのパスは何通りかある。主体は属性機関から属性を取得し、機関はその属性を関連するADFに直接渡すか、あるいは属性リポジトリに置く。ADFが認可判断をする際に属性を必要とするときは、その属性を元の要求の一部としてまたは交渉フェーズ中に主体から取得するか、あるいはローカルリポジトリまたは属性機関、主体、仮想組織に関連したりリポジトリからプルする。

属性は、その属性を保有しているエンティティ(保有者 / 受信者)や発行機関に確実に結び付けられていなければならない。属性は、完全性、発行者の信頼度、発行者の非否認を実現するために保護されていなければならない。これは、デジタル署名されたコンテナに属性を入れる(たとえばX.509属性証明書または署名つきSAML [SAML]属性アサーションなどを使って)か、または認証されたエンティティと信用されたエンティティ間のセキュアなチャンネル上で属性を発行し信用されたセキュアなリポジトリだけに属性を保存するかの、いずれかで実現する。

3.3.2 認可要求および応答

認可要求および認可応答は、セキュアな結び付けをしなければならないという点において属性と似ている。認可要求は主体とそのサービス要求にセキュアに結び付いていなければならない。認可応答は要求にセキュアに結び付いていなければならない。必要があれば元々の応答者にもセキュアに結び付いていなければならない。プル・シーケンス・モデルおよびエージェント・シーケンス・モデルでは、実施

機能と判断機能間のセキュアなチャンネル上の要求 / 応答プロトコル(たとえばSAML-P)か、デジタル署名されたコンテナ(たとえばX.509属性証明書またはSAML認可アサーション)によってこの機能が提供される。プッシュ・モデルが展開されている場合は、実施機能と判断機能との間に直接の接続がないため、セキュアなコンテナを使用する必要がある。実施機能と判断機能が同じ場所に置かれている場合、プログラミング・インタフェース([AZN-API]、[GAA-API]、[PERMIS])を代わりに使用することができる。

3.3.3 ポリシーの流れ

ポリシーは通常リポジトリに保存されるか、またはポリシー機関が判断機能へ直接供給する。ポリシーの転送は、発行者の機関をセキュアに確立してポリシーの完全性を保護するため、属性の転送と同様の要件が伴う。ここでも、ポリシーを転送したり取り出したりするときは、セキュアなコンテナが信用のあるエンドポイント間のセキュアな接続が必要となる。

3.4 ポリシー・サブシステム

ポリシーの評価は認可判断プロセスの核心部分である。ポリシーの表現、保存、取り出し、評価を受け持つコンポーネントはポリシー・サブシステムとみなすことができる。

3.4.1 ポリシー表現

ポリシー表現は通常ポリシー言語で表現されるが、この言語にはさまざまなポリシーに関する事項を表現するための語彙が含まれている。ポリシー言語が拡張可能な場合は、根本的な変更をすることなく、ドメイン固有の語彙や特性をあとで組み入れられる。XMLやS式 [McCarty 1996]などといったさまざまな言語プリミティブをポリシー言語の基礎として使用できる。資源に対するアクセス・ポリシーは、通常は資源の所有者から認可を受けているポリシー機関が記述する。このルート機関(SoA)定義はADF中に静的に構成されるのが一般的であるが、最上位の機関の制御下にある特権管理ポリシーで別途定義されることもある。

資源の現在の状態に関するADFが評価できないポリシーの条件があると、認可判断ができない場合もある。この場合、その条件はAEFに戻されて評価され、AEFはその条件を表現しているポリシー言語を理解できなければならない。

3.4.2 ポリシー・リポジトリ

記述されたポリシーは、ADFが使用できるよう保存しなければならない。ポリシーは、効率性と安全性の

ためしばしばADFと一緒に配置される。この場合、ADFはポリシー評価コンポーネントを直接使用するのでポリシー交換は不必要である。ただし一部の分散環境では、ポリシーを配布してポリシー発行者のドメイン中かまたは共通のVOドメイン中に保存することが望ましい。この場合、ポリシーの要素を必要に応じてADFに転送する場合がある。

3.4.3 ポリシー交換

ポリシーを交換するには、メッセージとプロトコルからなる素地が必要となる。属性交換および認可交換の場合と同様に、ポリシーは署名されたメッセージ内に含まれているか、またはセキュアなポジトリからのセキュアな接続転送によって、発行者へセキュアに結び付けられていなければならない。関係しているエンドポイントは共通のポリシー表現言語の仕様に合意しなければならず、これはポリシー言語を標準的に使用しようという動機付けになる。またポリシー交換にはポリシーに関するメタデータの交換も含まれる。たとえば、作成時刻、ポリシーの正当性、ポリシー発行者、信用アンカーなどがメタデータである。

3.4.4 ポリシー処理

ポリシー処理エンジンはさまざまなシステム固有なものであると思われる。人工知能やニューラルネットのパラダイムに基づいたシステムを含む既存のポリシー・システムは、ポリシー表現とメカニズムを理解している限り、効率的に処理する。

4. フレームワーク・コンポーネント

4.1 信用管理

4.1.1 信用機関

一般的な認可アーキテクチャにおけるポリシーと属性に関するアサーションを発行する機関の4つのタイプを図3.1に示す。信用管理はこれらの機関を定義し、それらの機関が信用の元で行うことを規定する。機関は管理ロール下で機能する個人または人のグループであるが、アーキテクチャのコンテキストでは機関はファイルの所有者、セキュアなサーバ、または公開 / 秘密キーのペアなどといった計算上のオブジェクトで表現される。

ポリシー機関と資源機関はともに資源に関するポリシーを発行するが、ポリシー機関はより上位のレベルで動作し、サイト全体またはVO全体に対してアクセス制御ポリシーを発行する場合もある。ポリシー機関は信用の根幹機関(ルート機関、SOAと呼ばれることもある)であり、ドメインの信用関係を定義する

責務を持っている。属性機関は属性を主体に割り当てる機関で、主体のドメインまたはVOに所属する。環境機関は、ディスクの使用状況、マシンの負荷や、接続の安全性、クライアントやサーバのインターネット・プロトコル(IP)アドレスなどといった分散環境資源環境に関する事項を定義する。図3.1には示されていない機関として、全てのエンティティのアイデンティティを確立する機関がある。

従来のシステムにおいては、機関はただの信用されたファイルでも構わない。アクセス制御情報が正しい場所であれば、ADFによって信用される。たとえば、エンティティは/etc/passwordファイル中にあるエントリによって定義される。またKerberos認証サーバなどのサーバやADFがセキュアな接続をしているNISによって信頼される。PKIをベースとしたシステムでは、機関は公開 / 秘密鍵のペアで表現されることがほとんどで、署名された文書またはセキュアな接続で自身のアサーションを提示する。PKIシステムのベースは1つ以上のCAに参画している全てのエンティティによって受理されてアイデンティティを検証することである。

4.1.2 信用関係の定義

VOや資源ドメインがさまざまな機関の代わりに務めるための方法を理解したら、それぞれの目的に応じてどれを信用してよいかを定義する必要がある。誰が信用関係を定義するかは、使用されているリスク管理戦略によって決定される。たとえば、資源は、どの機関を信用するかを独自に決定できることを選択することもできるし、VOポリシー機関の決定を受け入れることもできる。ユーザーの属性を定義している属性機関に対してユーザーがポインターを提供するものの、資源がそれを受け入れるかどうかはわからないというモデルもある。AEFは、認可判断に際してどのADFを信用すれば良いのかを知る必要がある。認可判断機能は同じ場所に配置されているか、または永続的でセキュアな接続を所有するAEFが自分の署名鍵を信用した場合は、署名された認可判断のアサーションを信用するという場合もある。信用管理のカテゴリーに入るもう一つのポリシー項目は、誰が権限委譲されたエンティティの権利の全てまたは一部を所有する代理人を作成するのか、そして誰によって権限を他のエンティティに権限委譲するかについてのポリシーである。

4.2 特権管理

特権という用語は認可属性や認可特権の定義、獲得、権限委譲、管理を指すのが一般的である。本文書で言う特権管理では、特権属性で表現された特権と、その特権にマッチしたポリシーが存在するという仮定で属性を取り扱う。特権を明示的に制御する必要のあるポリシーについて本節で述べる。

特権は属性の1つのタイプと考えることができ、ここでいう属性とは、何らかの資源に対して許可された

動作を暗示的または明示的に定義する、主体に関連するあらゆる特性である。資源に対して何らかのアクセスを明示的に許可している属性は特権属性と呼ばれる。ロール(ロールをベースとしたアクセス制御用)、クリアランス・レベル(強制アクセス制御用)、グループ・メンバーシップなどといった記述属性は、認可サーバがアクセス・ポリシーを解釈してユーザー固有の動作を付与し、その結果として暗示的にアクセス権を付与するのに使用できる。記述属性は特権の割り当てに間接のレイヤーを付加する(4.2.2、特権割り当てを参照)。通常、特権属性は資源へアクセスする前に主体が獲得し、要求と一緒にプッシュされる。またプッシュ・モデルの場合と同様に要求と一緒に記述属性を提示することもできるが、記述属性は資源ポリシーの一部として、あるいは認可サーバがプル・シーケンスで問い合わせできる属性リポジトリに保存されるのが一般的である。

属性は認可プロセスに関与する任意のエンティティに結び付けることができるが、最も一般的な使用方法は属性を主体に結び付ける使い方である(主体属性)。主体を資源に結び付ける(すなわち資源のクリアランス・レベル)例もある。

特権管理では特権と記述属性の両方の定義、割り当て、記録、提示、権限委譲、取り消しを扱う。特権と記述属性の管理には、特権の収集、特権の使用、特権の削除の3つの個別のフェーズがある。特権属性には、特権を付与 / 削除する機関、特権を要求 / 発行する主体の2つの主要なアクターがある。記述属性には、第3のアクターであるポリシー機関があり、アクセス制御ポリシー中の認可セマンティクスと記述属性を関連付ける際にこれが必要となる。

4.2.1 属性機関

エンティティとホーム・ドメインを関連づけるために、機関がエンティティ(通常は主体または資源)に対して属性を付与する。この機関は、属性に対する要求が自分の定義可能な範囲内のエンティティから来たものなのかを判断する手段をもつ。ルート機関、その元の代理人、属性を受け入れるドメインは機関の有効範囲について共通の情報を持っていなければならない。このことは特権管理ポリシーに記されていないなければならない。この他にもどの機関がどの属性にどの値をつけてどのエンティティに付与することができるのかなども定義する。

属性機関は実組織や仮想組織などといった複数の機関の代理として実行することができる。属性機関が特定の資源に対して特権を付与することができるようにするには、資源所有者(資源のルート機関)によってその属性機関が機関から(おそらく仲介人経由で)権限委譲を受ける。この機関の権限委譲により、資源ドメインで実行されている認可サーバは、アクセスしようとしているエンティティが所有し提示している特権と適用可能なアクセス制御ポリシーの組み合わせに基づいてアクセス許可の判断をできる

ようになる。

与えられた属性の適切なルート機関を決定するためには、特権管理を問い合わせる手段が必要となる。プッシュ・シーケンスでは、主体は要求の前に属性を要求するが、プル・シーケンスでは認可機能が主体の属性を探してアクセス制約を充足させる。属性を収集している間に、主体は期待している最終結果を知る可能性があるが、その結果(たとえば資源Aから資源Bへのファイル転送)を達成するのに必要な特定の属性や特権を必ずしも知るわけではない。必要な属性や特権を見つけ出すことがこのステップに必要な前提条件である。

ルート機関はその権限をさまざまなエージェントに権限委譲することができる。一般的に、この権限委譲により制限付きの機関がいずれかのポリシーに記載され、権限委譲が付与機関によって無効にされ、関連するルート機関への問い合わせが権限委譲に認識されている、というツリー構造が作られる。

4.2.2 特権割り当て

特権の割り当てでは、どのアクセス権が誰に与えられているかを定義するプロセスが記載されている。特権は、主体への直接アクセス権限を規定したポリシー・コンポーネントを発行することで割り当てられる。この自由裁量的なアプローチは、特権を規定したポリシー・コンポーネントを資源アクセス制御ポリシーに組み入れるか、または主体に結び付けられている特権仕様を含んだ特権属性を発行することで実現できる。後者の方法は、主体数が管理可能な範囲にある小さな環境や特定用途の環境で特に用いられている。いずれの方法にしる、特権を定義しているポリシー規則は通常3要素の組(主体、資源、動作)で構成される。

より拡張性のある代替方法としては、ポリシーを使い、アクセス権限を記述主体属性(すなわち、ロール、グループ・メンバーシップ)に結び付け、主体にそれぞれの属性を発行するという方法である。このような特権を記述したポリシー規則は通常3要素の組(属性、資源、動作)で構成され、その属性は2要素の組(主体、属性)で構成される。この方法のことをロールに基づいた割り当てと呼び、前述の自由裁量的なアプローチに比べて柔軟な方法である。ロールに基づいた割り当ては、アクセス権限の資源固有な定義への(ポリシー機関による)特権の割り当てと、資源にとらわれない主体への(属性機関による)属性の割り当てとを区別するので、こうしたタスクを別の機関に配布することを許可する。さらに、主体をロール別にグループ化することで、許可されている主体への権限の直接割り当てよりも拡張性のある管理が可能になる。階層的なロール・スキーマは、特権をあまり付与されていないロールからより多くの特権が付与されているロールへアクセス権限を継承することを許可することで、この考え方をさらに拡張したものである。

また特権の割り当てには、個人や個人のグループに対して特権の一時停止や削除するためのプロセスも含まれている。

4.2.3 属性管理

属性アサーションは記述属性や特権属性を表明する権利の証明である。したがって、属性アサーションには所有者(主体)、発行機関、有効性の範囲、最低1組の属性/値のペア、有効期限などの共通した特性がある。属性アサーションが有している属性のリストは、その属性アサーションが所有している有効な属性アサーションのリストから生成される。

属性の所有により、主体はその属性に対するルート機関が所有しているドメイン内において、そのルート機関として動作することができる。機関は、自分が所有している代理人や他の主体に(その機関の特権管理ポリシーによる制限を受ける)属性を権限委譲することができる。したがって、主体は適切なツールにアクセスして権限委譲された属性を権限委譲したり無効にしたりする必要がある。主体は、権限委譲のために追加のポリシーを定義したり、権限委譲属性を作成したり、あるいはその両方を行う。権限委譲の方法は属性機関によって規定されなければならない。

4.2.3.1 属性スキーマ

属性は、通常複数の管理ドメイン中にある相手を信頼することで認識されなければならない(2.3ドメインの考慮を参照方)。したがって、属性の文法に関する共通のスキーマが必要となる。さらに、機関と信頼している相手は属性のセマンティクスについて合意して、認可インフラストラクチャ内に存在するさまざまな資源サーバおよび認可サーバによる属性の統一した取り扱いや評価を遂行しなければならない。

4.2.3.2 属性リポジトリ

属性アサーションは、それが使用されるまで保存しておかなければならない。この保存場所を属性リポジトリと言う。このリポジトリは主体や共有設備のローカルな制御下に置かれる。主体は、属性のプッシュ・シーケンスを使用するときその属性リポジトリにアクセスする必要がある。プライバシーの考慮の仕方によっては、主体は属性開放ポリシーを定義して、どのエンティティがその属性のどの主体にアクセス可能かを制御できなければならない。

4.2.3.3 属性アサーション

属性機関は属性を発行する方法を選択できる。属性機関とアイデンティティ機関が同じエンティティで

ある場合、属性は(たとえばアイデンティティ証明書などといった) アイデンティティ・トークンに直接埋め込むことができるが、これによりアイデンティティの寿命と属性の寿命が一緒に結び付けられることになる。属性とアイデンティティに別々のトークンを発行することで問題を切り離せるアプローチの方が良い場合が多い。別々のトークンを使用すると、通常、属性アサーションは属性(の集合)、所有者(通常は主体)、発行者、有効期間、その他アサーションの有効性に関するその他の条件、デジタル署名で構成される。信用されているチャンネル上や記憶場所の内部だけで使用されるアサーションでは、その発行者は明示されておらずデジタル署名はオプションである。

主体属性を結び付ける方法を参照する属性アサーションは、特定の資源も参照する。このメカニズムを使用すると、属性機関は、所有者が属性を使用することのできる資源がどれであるかを制御できる。プライバシー要件によっては、属性をエンティティ名以外のオブジェクト(たとえば公開鍵)に結び付けることもでき、属性を暗号化することもできる。

同じプロトコルを用いてADFに対して属性をアサーションすることもできる。このプロトコルは署名された属性アサーションを単に提示するだけか、またはプロトコルが定義した1つのメッセージ形式にユーザーの属性をすべて統合する。後者の場合、発行者や有効性などといった個々のアサーションのすべての情報は、認可サーバが属性を信用するために必要となるのが普通である。属性の適当な集合をアサーションするためのツールは、その属性のプッシュ・シーケンスに従う場合、全ての主体とその権限委譲者に必要となる。

4.3 ポリシー管理

ポリシーという用語は非常に広い意味で使われており、本文書の文脈ではアクセス制御ポリシーの意味に限定しておかなければならない。セキュリティ・ポリシーはメッセージ・セキュリティ、ユーザー識別、文書の暗号化要件などといった認可に関する領域の外部にある事物を対象とする。本文書で対象とするポリシーは資源アクセスに関する情報という意味に限定する。たとえば、どのユーザーにどの資源に対してどんな条件下でどの動作を許可するのか、などである。ポリシー文を程よくまとめるには、資源、動作、主体(ユーザー)の集合毎に参照できるようにする必要がある。

信用管理の節で述べた通り、ポリシーはポリシー機関が発行する。ポリシーの作成には人間のエンティティが関与することがほとんどで、資源を使用する前に作成される。ADFはポリシー機関に対してリアルタイムで問い合わせをすることができるが、ポリシーは通常どこかのリポジトリに保存されている。これはアクセス制御リスト(ACL)という形をとる場合もあれば、データベース、署名されたアサーションの集合、などといった形になる場合もある。ポリシー・リポジトリは、静的な資源に関するポリシーとしてごく自然に

考えられるソリューションである。動的に作成されるオブジェクトのポリシーを作成するのはかなり難しい。静的なポリシー1つでオブジェクト・クラス全体へのアクセスを制御してそのクラス内でオブジェクトを作成するだけにするのが適切な場合もあれば、オブジェクトの作成者がそのオブジェクトのアクセス・ポリシーを同時に作成したいという場合もある。この場合、オブジェクトの作成者は信用されたりポジトリへの書き込み権限を必要とする。

ポリシー管理で取り組む課題としては、各資源のポリシーを作成、修正、削除できるのは誰か、ポリシーはどれくらいの期間で無効にできるのか、ADFはどこでポリシーを見つけるのか、すなわちADFは誰を/何を信用するのか、などがある。また、分散ポリシー管理による課題として、アクセス判断をする際にADFがいかに関連するすべてのポリシーを見つけるかが挙げられる。この問題に対する解の1つとして、ユーザーの権限なしの状態から始め、ポリシーを追加することに権限を蓄積していくという方法がある。しかしこの方法では記述できるポリシーの種類、特にある個人がアクセス権を所有しているグループのメンバーであるにもかかわらず、その個人へのアクセスを明示的に拒否するような種類のポリシーに制限が生じる。

ポリシー管理が取り組まなければならない課題の1つに、資源の所有者やポリシーを追加しようとしている人に対して、どのようにして現在のポリシーを明確に表示するかというものがある。現在のウェブサイトへのアクセス規則は、構成ファイル中の複数箇所にポリシーが存在したり、階層構造ディレクトリ中のさまざまな場所のファイル中に存在する場合もあり、特定の資源に対するアクセス・ポリシーを把握するのがいかに大変かを示している。

4.4 認可コンテキスト

認可コンテキストは認可要求のプロパティで構成されるが、このプロパティは認可属性によって提供されるわけではなく、(指定された、あるいは特定の資源やサイト用の)認可ポリシーに含まれているわけでもないが、認可サーバが下す判断に関連している。

認可コンテキストにはサービス要求の時刻、場所、転送手段、認証に関する情報が含まれ、こうした情報の品質と信頼性の表示も含まれる場合もある。

たとえば、受け取ったサービス要求の時刻に関する文にはその時刻がどれくらい正確に制定されたか(たとえば、ローカルのハードウェア・クロックなのかネットワーク時刻サーバと調整したのかなど)や、その文がどの程度信頼できるか(たとえば、ネットワーク時刻サーバを使用して時刻が制定された場合、それはセキュアなプロトコルで行われたか)なども含まれている。

認可信任状の多くは認証信任状に関連付けられているので、認可コンテキストにもサービスを要求した

認証されたアイデンティティとその認証の強さに関する何らかの文が、数値的特徴(たとえばキー長)、プロトコル(たとえばPKI対Kerberos)、管理ポリシー(たとえば証明局の実際の文に合致した条件)などといった形で含まれている。

同様の情報はサービス要求を配信するのに使用される転送チャンネルにも、たとえばプロトコルやキー長などといった形で適用される。認可コンテキスト中の品質情報および信頼性情報はすべて認可ポリシー中に記載されている要件によって直接参照され、このテキストの特徴をあいまいなシンボル(たとえばその認証はPKI証明書とある一定の長さのキーによってなされなければならない)として取り扱う。ただし、品質や信頼性の指標などといった要件をコンテキストに課す一般化した方法によっても益を得ることができ、それによってこのコンテキストが可能性のあるすべての充足プロパティを数値的に評価できるようになる。

4.5 認可サーバ

図3.1に示した通り、認可サーバは認可要求を評価し、関連する属性、ポリシー、環境パラメータなどを考慮して応答するエンティティである。実際のポリシー文と認可アルゴリズムはアプリケーションに大きく依存することもあるが、一般的なプロパティをまとめることもでき、プロパティが暗示することをポリシー言語で表したものとアルゴリズムを推察することができる。

認可アルゴリズムとして以下の一部または全てを入力とする。

- ・ 要求の性質(たとえば、ファイルの読み込み、ジョブのサブミット、センサーの読み取り)。有効な要求タイプは全てポリシー言語で表現できなければならない。
- ・ 要求者の属性(権限委譲された属性も含む)。認可判断で使用される属性は全てポリシー言語で表現できなければならない。
- ・ 要求を満たすのに必要な資源の属性。どの資源が要求されているのかを把握するのは認可サーバの機能ではないので、認可アルゴリズムが資源属性を必要とする場合、必要な情報は認可要求とともに提供されなければならない。
- ・ 認可コンテキスト(4.4節参照)。
- ・ システム負荷、ネットワーク負荷、ファイル・システム上の利用可能な空間、使用履歴、ユーザー定員数などといった環境要因。環境要因が認可判断において考慮されるのであれば、迅速性と情報の正確性の両方の観点に関連した時間軸について必要な情報を取得する適切な手段が存在しなければならない。環境要因を考慮することはアプリケーションに大きく依存しており、情報の正確性と迅速な判断の間のトレードオフ/最適化の問題を含んでいる。

認可サーバが暗示する主要な点は、静的な環境パラメータと動的な環境パラメータに依存関係があり、こうしたパラメータをポリシー言語で表現できるということである。

・ ポリシー文。それぞれの要求のタイプに対して、ポリシー文は適切な認可応答を発行する基準を規定しなければならない。一般的に、認可アルゴリズムは要求と任意の属性 / 環境要因をその要求タイプのポリシー文と比較する。

認可アルゴリズムの出力とは、認可応答をさす。ALLOW/DENYという2値の応答で十分であるアプリケーションがほとんどであるが、アプリケーション固有の言語を作って応答の形式を指定する場合もある。たとえば、これには応答に結び付けられた条件(すなわち有効時間)、優先度レベル、拒否とその理由などが含まれる。認可応答を詳細化することの利点と欠点を注意深くはかりにかけなければならない。たとえば、応答に指定されている条件がエンフォースメントメカニズム中の別の意思決定ロジックを必要とし、否定的な応答時につけられた理由を見て、攻撃者がセキュリティ・データを収集してアクセス・ポリシーを再構築できてしまうかもしれない。

4.6 エンフォースメントメカニズム

アクセス権のエンフォースメントとは、権限エンティティによって実行を許された主体の代わりに、資源に対して行える操作を制限することである。

従来のエンフォースメントシナリオのほとんどでは、エンフォースメントメカニズムはアプリケーションに対するユーザーのアクセスだけに焦点を当てていた。アプリケーションやサービスがその基礎をなす資源に(たとえばOS経由で)アクセスすることは重要なことではない。なぜなら、通常、アプリケーションは信用されたソフトウェア・コンポーネントであり、アプリケーションのシステムへのアクセスはアプリケーション導入時に静的に構築されているからである。グリッドのコンテキストではこれとは違ったシナリオに直面する。グリッド・アプリケーションやグリッド・サービスはユーザーが計算資源に対して求めるソフトウェア・コンポーネントによって提供されるので、必ずしも資源の所有者に信用されている必要はない。資源はこうしたサービスに対してホスティング環境として動作し、性能基準を満たすためにいろいろな資源間を渡り歩く一時的なものであることが多い。したがって、サービスの特徴によってはサービス・アプリケーションとその基礎をなす資源(ホスティング環境)の間に固定的な信用関係を構築することが必ずしも可能ではない。むしろ、主体がサービスにアクセスすることを制御するだけでなく、サービスが現在のそのサービスのホスティング環境へアクセスすることも制御することが重要である。

エンフォースメント機能は許可された操作の集合をサービス要求の一部として受け取るか(プッシュのシナリオ)、またはADFに問い合わせるか(プルのシナリオ)のいずれかである。ADFとAEFがお互いに離れ

たところにある場合、認可要求 / 応答プロトコル(たとえばSAML-Pなど)を使用することができる。両者が同じ場所にある場合、利用可能なプログラミング・インタフェースはいくつかある(AZN-API、GAA-API、PERMISなど)。

エンフォースメントメカニズムはその特徴によって、アプリケーション依存メカニズムとアプリケーション独立メカニズムの2つの異なるグループに分けられる。

4.6.1 アプリケーション依存エンフォースメントメカニズム

アプリケーション依存メカニズムはアプリケーションやサービスと直接統合されていることが多く、アプリケーションがその基礎をなすOSにアクセスしようとする前にエンフォースメント機能を遂行する。このアプローチにより、特定のアプリケーションに特化した非常にきめ細かいアクセス制御ポリシーを実施することができる。さらに、組み込まれたエンフォースメント機能は非常に効率が良く、サービスは他に悪影響を与えることなく退化でき、認可がないためにタスクを正常に完了できない場合にはユーザーに対して有用な情報を提供することができる。エンフォースメント機能をアプリケーションに統合することによる欠点が2つある。1つはサイトがこのアプリケーションのコードを信用する必要があること、もう1つは同じセキュリティ・コードがその基礎をなすミドルウェアやOS内に重複して存在する可能性があることである。サイトがこのアプリケーションのコードを信用する必要があることは、ユーザーが取り込んだサービスに対して障壁となる。また、既存のレガシー・アプリケーションが認可APIや認可プロトコルを使用して資源に対するアクセスを実施するように改造するのも大変困難である。

したがって、アプリケーション依存エンフォースメントメカニズムは新しくサービスを実装する場合やサービスが申し分のないものである場合に向いている。たとえば、CAS [CAS02]、Akenti [AKENTI]、PERMIS [PERMIS]などはアプリケーション依存エンフォースメントメカニズムを活用している。

4.6.2 アプリケーション独立エンフォースメントメカニズム

アプリケーション独立エンフォースメントメカニズムはサービスやアプリケーションから分離されていて、非常に制約の強い実行環境下でサービスを実行するというアプローチを採用している。これにより、信用されていないサービスを実行することが可能で、コードの移植やユーザーが提供した実行可能モジュールの取り外しをサポートしている。アプリケーションのセマンティクスが通常はわからないため、実施できる操作の決めの細かさやシステムにかかる性能オーバーヘッドなどの点で欠点がある。このようなメカニズムをOSに強く統合する必要があると移植性の点で問題となることが多い。制約された実行環境を実装するには2つの方法が良く知られている。

*OSのカーネルによって実行されるセキュリティ機能を利用して資源へのアクセスを制限する。たとえば、ファイルのアクセス許可、ファイル・システムのアクセス制御リスト、ネットワーク・ファイヤーウォール規則、クォータの設定など。このアプローチでは、性能に影響を与えることなくアクセス規則を実施する。このアプローチの実装も、標準化されているかあるいは広く採用されているインタフェース(たとえばPOSIX標準など)が使用されていれば、かなりの数のOSに移植可能である。このアプローチはOSがサポートしているセキュリティ機能に変換できる規則にしか実施できない。たとえばこのアプローチはPRIMA[PRIMA03]システムに引き継がれている。

*アプリケーションによる資源アクセスがOSに渡される前にそのアクセスを遮断して評価する。このアプローチは「サンドボックス方式」と呼ばれることもあり、きめの細かいアクセス制御を自由裁量で実施することができるが、システム・コールを遮断するため性能に大きな影響を与える危険性をはらんでいる。また、低レベルのOSインタフェースを使用してシステム・コールを遮断しているため移植性に制限があるという欠点もある。仮想実行環境[VXE]やJanus [GOL96]は典型的なサンドボックス方式のシステムで、システム・コールの遮断を実行している。SlashGrid [SLASHGRID]も仮想ファイル・システム・レイヤーを介してアプリケーションとOSの間に存在するシステムである。

5. 既存認可メカニズム、モジュール、システムの分類

グリッド・アプリケーションやその他のアプリケーションが認可に関する課題に対処するために使用している認可システムや認可メカニズムは、現在さまざまなものが存在している。本節では既存の認可システムの選択について、これまでの節で述べてきた認可フレームワーク・メカニズムとの関係の観点で説明する。

5.1 Akenti認可サービス

5.1.1 モデルおよびアーキテクチャ概要

Akenti [AKENTI]はアクセス制御判断機能を提供しており、プッシュ・モデルでもプル・モデルでも使用できる。Akentiは、最も根本的なレベルで要求者のアイデンティティと資源の名前を取り込み、ユーザーのアクセス権を署名付き機能証明書、すなわち認可アサーションとして返す。プル・モデルでは、ユーザーは資源に対して認証された接続を行い、次にAkentiを呼び出し、ユーザーと資源名を渡す。プッシュ・モデルでは、要求者はセキュアでない可能性のある接続上で名前と資源名を使ってAkentiを呼び出す。Akentiは要求者の名前、資源の名前、アクセス権限が含まれている署名付き認可アサーションを返す。要求者は認証された接続上でそのアサーションを資源に渡し、資源はアサーション中にあ

る名前が認証された名前と一致しているかどうかをチェックし、アサーションの署名を検証する。Akentiは、すべてのプリンシパルがX.509公開鍵証明書によって識別できることを求めている。AkentiのADFインタフェースは属性アサーションと要求された権限に渡され、属性検索の範囲を制限することができる。

5.1.2 属性アサーション機能およびポリシーアサーション機能

Akentiは属性アサーション機能とポリシーアサーション機能の両者を提供している。これによりステークホルダーは署名付きXML証明書を発行できる。この証明書はポリシー証明書及び使用条件という形で、属性アサーションとポリシーアサーションを含んでおり、分散して保存され、Akenti ADFが見つかる十分な情報が含まれている。属性アサーションとポリシーアサーションは、ユニークな証明書IDとバージョン、署名アルゴリズム、証明書のタイプ、有効期限の開始および終了時刻、発行者のアイデンティティを含み、署名されたXML形式の証明書として関係者間で保存され、受け渡される。これは、ポリシーがADFから離れた場所に保存されていて安全に交換される必要があるシステムの例である。

5.1.3 認可情報の流れ

ステークホルダーは、HTTP、LDAP、ファイル・システム要求などでアクセスできる属性リポジトリやポリシー・リポジトリ中に作成した証明書を保存する。要求者のAkenti ADFとの間の通信はSOAPプロトコルで行われ、簡単な問い合わせメッセージと返信メッセージがやり取りされる。AEFはAkenti ADFライブラリにリンクできるか、またはSOAPメッセージを介してAkentiサーバと通信できる。

5.1.4 信用管理

Akentiは全ての信用関係が認可ポリシーの一部として明示的に提示されていることを要求する。また、全てのプリンシパルがX.509公開鍵証明書[RFC3280]で識別でき、公開鍵を使用して全てのポリシーアサーションや属性アサーションをアクセス判断の際に検証することを要求する。資源ドメインのルート・ポリシーにはX.509証明書、信用されたすべてのCAの発行ディレクトリ、そのドメインのステークホルダーのリストが含まれている。ステークホルダーは、資源に対する使用条件を発行することのできる唯一のエンティティである。使用条件には、資源にアクセスするのに必要な、要求されている属性と値、および誰がその値で属性を発行したかが提示されている。使用条件は属性アサーションが使用されるときはAkentiは条件に合った関係者が発行したものであるかを常にチェックし、署名を検証してアサーションの完全性を保証する。

5.1.5 エンフォースメントメカニズム

Akentiは基本的には判断機能であり、エンフォースメントは資源管理者に委ねている。ただしAEFが実行時の条件を評価するのを支援するAPIを提供している。Akentiが返した認可アサーションには時間制限や負荷要因制限などといった実行時条件が含まれていて、AFEはこれら进行评估しなければならぬ。AEFが個々の条件を評価する機能を提供する場合、評価フレームワークAPIを使用してこれらの条件を統合することができる。

5.2 Cardea

Cardea [CARDEA]は分散認可システムであり、NASA情報パワーグリッドの一部として開発され、固有のローカルのアイデンティティを考慮するのではなく、認可に関連する資源や要求者の特徴集合に従って認可要求を動的に評価するものである。管理ドメイン下でアクセスされる資源は、要求者と資源の特徴がXACML文法で規定されたローカルのアクセス制御ポリシーで保護されている。認可判断を完了するのに必要な情報は、判断プロセスの間に評価・収集される。この情報は、要求者、エージェント、ポリシー実施ポイント(PEP)、SAMLポリシー判断ポイント(PDP)のいずれかによって適切に組み合わせられ、XACML PDPに提示されて評価される。SAML PDPIは情報を取得後、アクセス要求に対する最終的な認可判断及び関連する詳細情報を要求者に返す。

5.2.1 認可情報

要求された資源、要求された動作、現在の環境などといった、主体のあらゆる特徴が認可判断で考慮される。Cardeaで採用されたモデルはこれらの属性をSAMLアサーションとして表現し、コンポーネント間で受け渡される。各コンポーネントは必要とあれば別のネイティブ内部形式へ変換するなど、あらゆる用途でこのアサーションデータを自由に使用できる。ただし、このデータをコンポーネント間で通信する場合、そのデータの出所や保証人が誰であるかにかかわらず、全ての特徴がこの形式で表現される。

5.2.2 認可判断の起動と実施

CardeaはXACMLモデルを活用して認可を評価し、SAMLを活用して評価プロセス中に使用するアサーションデータを取得する。Cardeaは、初期リクエストを受け取るSAML PDPが最終的な認可判断の詳細をPEPに提供することを前提としている。SAML PDPIは最初の要求内容に応じてその要求を評価する適切なXACML PDPを決定する。そして、エンティティ間の通信の流れがこれらの関連する標準によ

って規定される。

5.3 CAS

通信認可サービス(CAS) [CAS02、CAS03]は、サイトポリシーとVOポリシー間の課題を分離する。特に、サイトはそのポリシー空間の部分集合の管理をVOに権限委譲することができる。CASは、図4に示す通り、「プッシュ・モデル」の認可サービスとして機能する。本節では、CASが通常の操作でどのように使用されるのかについての概要を説明する。

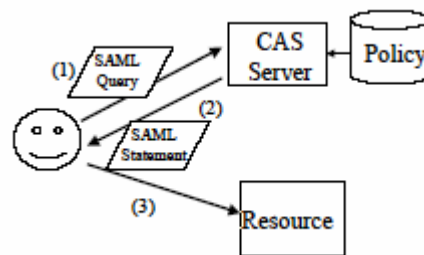


図4: CASのアーキテクチャ。処理ステップは以下に記載。

図中の処理ステップは以下の通り。

1. 左に示されているクライアントが署名されたSAML AuthorizationDecisionQuery要求を右側にあるCASサーバに送信し、アクセスしたい資源と起動したい動作を示す。
2. CASサーバはユーザーのアイデンティティを確立する。このアイデンティティを使用してCASサーバはVOのポリシーによって確立された権限を決定する。次にCASサーバはAuthorizationDecisionStatementを含んだ署名付きSAMLアサーションを返す。このアサーションには主体としてユーザーのアイデンティティとユーザーが要求した動作の部分集合が含まれている。
3. ユーザーはSAMLアサーションと認証された実施要求を資源に提示する。資源はSAMLアサーションを使用し、CASサーバに対してどれだけの機関が権限委譲されたかを示すローカルポリシーに従って要求を認可する。ユーザーはこのアサーションを使用して複数の資源に対して複数の要求を作成することができる。

クライアントが任意の資源に対して所有している権限の全集合、資源の全集合、時にはCASサーバが権限を有している全ての資源をも含んだアサーションを要求するのは特別なことではない。返されたSAML文は通常数時間は有効であるため、複数の権限があるアサーションを使用するとユーザーはCASサーバにもう一度コンタクトすることなく複数の異なる動作を行うことができるが、これは推測できな

い場合がある。

5.4 PRIMA

PRIMA [PRIMA02、PRIMA03]は特権管理とアクセス制御用のシステムである。PRIMAはエンドユーザーや管理者がグリッド・ミドルウェア・メカニズムを通じ、権限のある資源に対する特権を管理するツールを提供する。PRIMAはX.509属性証明書を活用して特権とポリシー文を携行する。アクセス制御判断機能は主体の属性(特権)と資源のポリシーを組み合わせたものに基づいて要求を許可し、低レベルのアクセス制御実施機能と判断資格を供給する。このエンフォースメントメカニズムはローカルのユーザー・アカウントを動的に割り当て、構成し、POSIXファイル・システムのアクセス制御リストとSlashgridプロジェクトのXMLを基本としたグリッド・アクセス制御リスト(GACLs)を活用してきめの細かいアクセス権の割り当ておよび管理を行う。

5.4.1 認可シーケンス

Primaはハイブリッド型の認可シーケンスを使用する。主体が資源からのサービスを要求すると、実施機能(AEF)が認可判断機能(ADF)に対して大まかなアクセス判断(yesかno)とサービスが実行される環境へのハンドルを問い合わせる。これはプル・シナリオで説明したフローに従う。ただしADFは、サービスが要求する必要最小限のきめ細かな権限でサービス用の実行環境(実施機能)も構成する。この2つ目のステップでは判断資格がAEFに供給され、プッシュ・シーケンスが規定する情報フローに従う。

5.4.2 エンフォースメントメカニズム

PRIMAの実施はアプリケーションが実行される環境の制御に基づいている。PRIMA AEFは資源管理者、資源所有者、グループ・リーダー、プロジェクト・リーダーなどといった権限エンティティが発行した主体特権に基づいて、ローカルのユーザー・アカウントをオンデマンドで作成、構成、管理する。PRIMA AEFは動的に修正されたファイル・アクセス制御リストとホスト・ベースのファイヤーウォール規則を利用してサービスを制限する。実行環境を通じた実施により、PRIMAはOSに既に存在しているセキュリティ・コードを複製することなく、信用されていないレガシー・アプリケーションをセキュアに実行することができる。

5.4.3 判断機能

PRIMAのADFは主体属性(特権)とアクセス制御および特権管理ポリシーに基づいてアクセス判断を下

す。主体は、要求に対して属性を選択的に提供することでどの属性がADFによって考慮されるかを規定することができる。PRIMAのアプローチは付加的な機能ベースのセキュリティ・モデルを構成し、見当たらない属性や故意に除外されている属性へのアクセス許可を少なくなるようにしている。ADFは資源と同じ場所に存在し、ゲートキーパーの権限モジュールとXACML権限要求および応答を介して通信する。主体特権は動的に作成されたポリシーという形でADFに供給されるが、このポリシーは要求に固有なものであり、主体が提示している全ての特権を含んでいる。

5.4.4 属性アサーション機能とポリシーアサーション機能

PRIMAは権限エンティティが個々の特権を主体(所有者)に与える主体属性作成ツールを提供する。この特権はXACML規則構成物としてエンコードされ、発行者によって署名付きX.509属性証明書に組み入れられる。

管理者が資源アクセス・ポリシーおよび特権管理ポリシーをXACMLでの作成を支援するポリシー作成ツールは現在開発中である。ポリシーは属性証明書の中に組み入れられていて、その証明書が成立するPRIMAポリシー判断点へセキュアに転送できる。2つのタイプのポリシーが使用される。資源アクセス制御ポリシーは主体属性を介して主体に対して与えられた特権を拘束あるいは増大させるのに使用され、特権管理ポリシーは主体特権に対して権限を持つのは誰か、その特権をどのように権限委譲するかを定義する。

5.5 PERMIS認可インフラストラクチャ

PERMIS [PERMIS]は認可インフラストラクチャをベースとした属性で、以下のコンポーネントで構成される。

- i) XMLで記述され、デジタル署名付きX.509属性証明書でセキュアな状態になっている権限ポリシー
- ii) X.509標準に準拠した属性証明書であるユーザー権限トークン
- iii) 属性証明書(ポリシー・トークンと権限トークン)を保存するのに使用されるLDAPディレクトリ・サーバの1つ
- iv) ADF
- v) OpenGroups AZN APIを非常に大まかに簡素化した、アプリケーション依存AEFが呼び出すADFへのAPI
- vi) 属性証明書とポリシーを作成するさまざまなツール

5.5.1 認可フレームワーク

現在PERMISは認可プル・モデルで動作する。ユーザーは資源(より正確には資源のAEFコンポーネント)にコンタクトし、次に資源はPERMIS ADFにコンタクトする。PERMIS ADFは、(ユーザーの属性証明書から取得した)ユーザーの属性と資源のポリシーに基づいて判断を下し、その結果をAEFに返す。AEFはこの判断を資源になり代わって実施する。PERMISの意思決定は2つのステージからなる。最初のステージは通常ユーザーのログオン時に発生し、ポリシーに従ってユーザーの属性証明書を評価し、信用できない証明書を拒否する。有効な属性は保存されてJavaオブジェクト中のAEFに返される。2番目のステージはユーザーが何らかの動作を実行しようとしたときに発生し、ユーザーの有効な属性とポリシーに基づいて許可または拒否の判断を下す。認可要求および応答はJava APIのパラメータとして渡される。認可判断は単純にブール値である。

5.5.2 認可情報の流れ

属性の獲得は通常はX.509属性証明書を作成する特権割付ツールが実行し、LDAPサーバ上の属性証明書の所有者のエントリ中にその証明書を保存する。属性はプッシュ・モデルまたはプル・モデルのいずれかを介して適用できる。属性プル・モデルでは、PERMIS ADFがコンタクトしているLDAPサーバのURL上で構成され、アクセス要求を作成したユーザーが見つかることができた全てのX.509 ACを取り出す。属性プッシュ・モデルでは、AEFが使用される属性証明書をPERMIS ADFに渡す。この証明書をAEFがどのように受け取るかはアプリケーションに依存した問題である。ユーザーはこの証明書とアクセス要求を資源に送信するか、またはShibbolethタイプのサービスがリモートの属性サーバから証明書を取ってくる。

5.5.3 ポリシー問題

PERMISのポリシーはXMLで記述され、サブ・ポリシーの集合で構成される。サブ・ポリシーの全詳細については論文または我々のウェブサイト(<http://sec.isi.salford.ac.uk>)から入手可能である。PERMISのポリシーは階層的なRBACをサポートしており、ユーザーはロール(または属性)を与えられ、ロール/属性はアクセス権を付与される。上位のロール/属性は下位のロール/属性の特権を継承する。PERMISのポリシーにはGT、LT、EQなどといった条件文をANDやORで組み合わせて自由に含めることができる。

PERMISのポリシーはデジタル署名付き属性証明書として作成者である資源のルート機関(SoA)によってLDAPディレクトリ中に保存される。PERMIS ADFのJavaオブジェクトはコンストラクト時にこのLDAPデ

ィレクトリからポリシーを取り出す。PERMIS ADFは全ての判断を内部で下し、単純なブール値である許可 / 拒否という応答が返されるだけ(すなわち、ポリシー条件文はAEFには返されない)なので、外部処理エージェントはこのPERMISのポリシーを見ることはできない。

5.5.4 信用管理

PERMIS ADFは資源を信用し、資源AEFはPERMISのJavaオブジェクトがAEFによってコンストラクとされたときにPERMISに対して信用の権限ルートの名前(資源SoA名)を提供する。またAEFは使用するポリシー固有のOIDとコンタクト先のLDAPディレクトリのURLもPERMISに提供する。次にPERMIS ADFは構成されたLDAPディレクトリにコンタクトし、資源SoAのエントリからポリシー属性証明書を取り出す。PERMISはそのポリシーがSoAによって署名されたもので、正しいOIDを持っていることをチェックしてポリシーが信用できるか判断する。ポリシーには、ユーザーに属性証明書を発行した信用されたりリモートのSOAの名前が含まれている。このようにして、資源SoAはユーザーに属性証明書を発行するものとして信用されたりリモートのSoA集合にその権限を権限委譲する。属性証明書は信用されたSoAの1つによって署名されていなければならない、ロール割り当てサブ / ポリシーに準拠していなければならない。そうでない場合はPERMIS ADFがその証明書を廃棄する。リモートのSoAからその下位のSoAへの動的な権限委譲は現在サポートされていない。下位のSoAが属性証明書をユーザーに発行しても、その証明書は信用されない。

属性証明書にはその所有者(主体)の識別名が含まれている。PERMISでは、認証の信用のルートはアプリケーションの責任であり、PERMISはアプリケーションを信用してユーザーを適切に認証し、属性証明書上のデジタル署名を検証する。ユーザーはアプリケーションに対して自分自身を認証して、与えられたDNによって識別されていることを証明する。それによりPERMISはこのDNを含んでいる属性証明書がそのユーザーに属しているということを信用することができる。

5.6 EU DataGridセキュリティ・アーキテクチャ

EDGセキュリティ・アーキテクチャ [EDG-SEC]は、属性を管理する仮想組織メンバーシップ・サービス (VOMS)と、資源が利用できる複数の認可判断機能という2つの認可コンポーネントに基づいている。VOMSサーバは資源からの認可プル要求にも、資源を使用したい主体からの属性アサーション要求にも応答する。その結果、このアーキテクチャは前述したプッシュ・モデルとプル・モデルの両方をサポートしている。

5.6.1 VOMS属性権限

プル・モデルでは、VOMSサーバはHTTPSを使用した各資源によって定期的にコンタクトされ、規定された基準(たとえば、ある特定のグループの全てのメンバー)に合致したメンバーは証明書主体名一覧リストにのる。VOMS HTTPSサーバは接続の開始時に独自の証明書で自分自身を識別するので、この情報について追加の署名は使用されない。

プッシュ・モデルでは、資源を使用したい主体がVOMSサーバにコンタクトする。主体のホーム・ロケーションにあるクライアント・ツールがGSIプロキシを使用してその主体をVOMSに対して識別し、以後のセッションで必要とされる属性アサーションの集合を要求する。VOMSサーバは、主体に要求された属性を含む名前/値のペアの署名付き文字ブロックを発行する。文字ブロックは主体名とVOMS証明書名、VO名、アサーションの有効期限の上限と下限で始まっていて、1つ以上のグループ、ロール、機能の3要素の組がそれに続く。特定のロールや機能がアサーションされていない場合はNULL値が続く。この署名されたアサーションは次に、主体の新しいGSIプロキシに主体のクライアント・ソフトウェアによって生成された拡張として含まれる。

5.6.2 認可判断機能

認可判断機能は、アプリケーションのネイティブ認可エンフォースメント機能がリンクするライブラリによって提供される。GACL、LCAS、Java認可マネージャという3つの補助システムが提供される。

GACLはグリッドのアクセス制御言語ポリシー文を処理するためのライブラリで、XMLで記述されている。このポリシー文は(書き込みなどの)権限を(特定の主体名やVOMSグループのメンバーシップなどといった)1つ以上の基準を満たす主体に対して許可する。GACLのC/C++ APIは認可判断機能を提供し、提案された動作(たとえば書き込み)が与えられるとyes/noの回答、主体が所有している信任状、問い合わせ中のオブジェクトに関連しているポリシーを返す。ローカルセンター認可サービス(LCAS)は、大きなサイズのファイルの転送やジョブのサブミットなどといった負荷の重い要求に対して適している、より柔軟でカスタマイズが可能なフレームワークを認可判断機能に対して提供する。判断自体はフレームワークに対するプラグインによって実行される。プラグインは(Globusグリッド・マップファイルなどの)主体名のリストまたはVOMS属性名のリストを使用するか、あるいはGACLで記述されたポリシー文を使用する。Java認可管理システムは、JavaベースのSOAPサービス用の認可判断機能を提供する。XMLポリシーは、証明書の名前がVOMS属性に基づいたサブジェクトに対して内部属性を許可する。

6. 関連標準

GGF、IETF、OASISのさまざまなグループが認可フレームワークのさまざまな要素の標準化に関わっている。本節ではこうした標準化関連活動の概要を述べる。6.1節ではポリシー機関(図3.1)が作成したポリシーをデジタル・エンコーディングするための言語について説明する。6.2節では、ADFやAEFなどといった認可アーキテクチャのコンポーネント間の通信を円滑化するための通信プロトコル、メッセージ形式、関連標準について触れる。最後に6.3節では、認可フレームワークの定義と関連する他のアプリケーション領域に焦点を置いた話題について簡単に述べる。

6.1 ポリシー仕様

拡張可能アクセス制御マークアップ言語(XACML) [XACML]は、インターネット上の情報へのアクセス用のポリシーを表現するために開発されているXML仕様である。このスキーマでは、認可ポリシーを記述したり認可に対する要求が行われるコンテキストを記述したりするのに必要な要素を定義する。ポリシーが適用される対象は、ある資源に対して動作集合を要求している主体である。3つの要素はいずれも属性指名者によって規定されるので、ポリシーをさまざまなスケールで適用できる。規則は、動作が実行されたときに満たさなければならない義務を含んだポリシーと規則結合アルゴリズムで結合される。

拡張可能権限マークアップ言語(XRML)はデジタル資源を使用するための権限と条件を記述する言語である[XRML]。キーとなるトップレベルの要素はXrMLで定義されているライセンスである。このライセンスには、ユーザーがなんらかのオプション条件を適用した上でデジタル資源に対して有する権利を定義する。資源のコア・タイプ、権限、条件および標準の拡張タイプを定義した文書がある。

RFC2704 - キーノート信用管理システム [RFC2704]は、セキュリティ・ポリシー、信用状、関係などを規定し、解釈するための統一したアプローチで信用管理を定義する。この管理システムは「フィールド」と値からなる簡単な言語を定義して認可ポリシーと信用状の両方を表現する。

6.2 認可コンポーネント間の通信

セキュリティアサーションマークアップ言語(SAML) [SAML]は、認証情報と認可情報を交換するための言語とプロトコルを定義する。この言語が主として目的としていることは、許可管理データが標準化された方式でドメイン間あるいはさまざまなシステムで共有できるようなメカニズムを提供することである。SAMLは問い合わせと返信用およびセキュリティ関連のアサーションのためのスキーマを提供する。アサーション文は、認証文、属成文、権限文のいずれかである。

Webサービス・セキュリティ [WSSEC]は、分散システムにおけるすべてのセキュリティ問題についてア

アプリケーション用のプロトコル、メッセージ形式、ポリシー言語を定義するための標準的なXMLの語彙を提供するWebサービスのコミュニティの試みである。WebサービスはSOAPメッセージを使用して通信するため、各セキュリティ仕様は、SOAPメッセージのヘッダーにより転送手段にとらわれない方法で伝えられるメタデータを定義するXMLスキーマで表現される。WSセキュリティ・フレームワークはセキュリティ機能の7つの仕様を定義する。ここでの話題に直接関係する仕様はWS認可である。WS認可は、Webサービス用のアクセスポリシーをどのように規定して管理するかについて記述する。特に、WS認可は申し立てがセキュリティトークン内でどのように規定されるのか、申し立てがエンドポイントでどのように解釈されるのかについて記述することになっているが、まだ公開されていない。WSセキュリティ・ポリシーはセキュリティに関連するポリシーアサーションを定義する。メッセージやセキュリティ・ヘッダーのプロパティの一部の完全性、秘密性、年齢についてのアサーションを定義する。

Webサービスのセキュリティ・スキーマ要素をサービス・インタフェースに組み入れて、その結果としてセキュリティ要件やその要件を実施するために必要なトークンを通信するクライアントやサービスに共通の方法を提供することはオープン・グリッド・サービス・インフラストラクチャの開発者の意図するところである。この時点で、複数のPEPが使用できる認可サーバを記述することが可能になるかもしれない。そのようなサーバが使用するポリシーがXACMLポリシー文のような標準的な方法で形式化されていたら、資源サイトは標準的な認可サーバを使用してポリシーをカスタマイズするだけで要件を満たすことができるようになるかもしれない。

6.3 関連認可フレームワーク

RFC2904 AAA認可フレームワーク [RFC2904]は、ユーザー、ユーザーとの合意に基づきサービスに対するユーザーの要求許可をチェックするホーム組織(UHO)、UHOとの合意に基づいて個々のユーザーを知ることなく(大まかに)アクセスを許可するAAAサーバを含んだサービス・プロバイダー、およびサービスを提供するサービス設備の4つの要素からなる。アクセス・ポリシーの評価と実施のために、RFC2904はポリシー判断ポイント(PDP)とポリシー実施ポイント(PEP)を導入している。PDPは、PDPの管理ドメインに対して権利を有する関係者によって定義されたポリシーに基づいてアクセス制御判断を下す。PEPは通常はサービス設備に設置されている対応するエンティティで、PDPが下したアクセス判断を実施することができる。

ISO 10181-3 [ISO 10181-3] 認可フレームワークは、アクセス要求に参画している4つのコンポーネントを定義する。イニシエータ、ターゲット、アクセス制御実施機能(AEF)、アクセス制御判断機能(ADF)の4つである。インターネットの認可用属性証明書プロファイル(RFC 3281) [RFC3281]は、任意の属性(た

例えば、ロール・メンバーシップ情報、ポリシー文、アカウント情報などをアイデンティティに結び付ける手段を提供する。認可API - 認可フレームワーク用の汎用アプリケーション・インタフェースは、RFC2904に定義されているように、AEFとADFの間のインタフェース用の標準的なAPIを定義する。IETFが提案した、標準化された汎用認可およびアクセス制御API (GAA API)は資源ゲートウェイおよび認可モジュールやサーバと、ステークホルダーがアクセス要件を認可サーバに対して表現するためのポリシー言語との間の簡単なインタフェースの定義である。

7. セキュリティ上の注意点

セキュリティ問題に関する議論は本文書の本質的な部分である。本文書の読者には、認可システムの設計、実装、運用における誤りがあらゆるシステムのセキュリティを危険にさらす可能性があるということを知っておいていただきたい。グリッド環境などの分散環境のセキュアな認可インフラストラクチャを構築するのは複雑な作業である。セキュリティ技術(たとえば、ポリシー言語、認可サーバ、暗号化メカニズムなど)だけではセキュアなシステムはできない。

コンピュータ・システムを設計、実装する際に忘れがちな点は人的要因である。ユーザーはセキュリティを障害物としてみることが多い。特にセキュリティ・メカニズムによりタスクが実行できなかったり作業を効率よく済ませることができなかった場合はなおさらである。そのような場合ユーザーは、資源だけでなく自分たちを保護するために用意されているセキュリティ・メカニズムをバイパスしようとするので、システムのセキュリティにとって大きな脅威となる場合がある。情報漏洩はこのコンテキストにおいて重要なトピックである。もし否定的な認可判断のためにシステムへのアクセスが拒否された場合、サービスの要求者に対してどの情報を公表するかについて注意深く評価しなければならない。システムのその他のコンポーネントにおいては、要求したタスクが何故失敗したかについて全詳細がユーザーに提供される。しかし認可コンポーネントにおいては、悪意を持ったユーザーや攻撃者の場合を考慮する必要があり、アクセスが何故失敗したかに関する詳細をこうした攻撃者に提供すると、(たとえば社会工学などによって)この情報を使用した認可システムを回避することができるようになってしまう。

著者の連絡先

Rich Baker

Building 510m

Brookhaven National Laboratory

PO Box 5000

Upton, NY 11973, USA

email: rbaker@bnl.gov

Andrew McNab

Department of Physics and Astronomy

University of Manchester

MANCHESTER

M13 9PL

England

email: mcnab@hep.man.ac.uk

Bob Cowles (Co-Editor)

Stanford Linear Accelerator Center

2575 Sand Hill Rd., MS 97

Menlo Park, CA 94025, USA

email: bob.cowles@stanford.edu

Lavanya Ramakrishnan

Center for Networked Information Discovery and Retrieval / MCNC

PO Box 12889

Research Triangle Park, NC 27709, USA

email: lavanya@cnidr.org

Leon Gommans

Advanced Internet Research Group

Informatics Institute
University of Amsterdam
Kruisln 403
1098 SJ Amsterdam
The Netherlands
email: lgommans@science.uva.nl

Krishna Sankar
Cisco Systems Inc
170, W. Tasman Drive,
San Jose, CA-95134
email: ksankar@Cisco.com

Markus Lorch (Editor)
Department of Computer Science
Virginia Tech (m/c 106)
Blacksburg, VA 24061, USA
email: mlorch@vt.edu

Dane Skow
Fermi National Accelerator Lab
MS 369
P.O. Box 500
Batavia, IL 60510-0500
email: dane@fnal.gov

Paul Madsen
Entrust
1000 Innovation Drive,
Ottawa, K2K-3E6

Canada

Email: p.madsen@entrust.com

Mary R. Thompson

Lawrence Berkeley National Laboratory

MS50B-2239

1 Cyclotron Rd.

Berkeley, CA 94720, USA

email: MRThompson@lbl.gov

謝辞

コミュニティのメンバー、特にCarlisle Adams、Von Welch、David Chadwick、Rebekah Leproの各氏による貢献に感謝の意を表す。

用語

グリッド認可で使用される用語に関する総合用語集はGGFの認可フレームワークWGおよびメカニズムWGが作成中である。読者の方は最新版の用語集については<http://www.ggf.org>のGGFドラフト版リポジトリを参照されたい。

知的所有権の記述

GGFは、本文書に記載されているテクノロジーの実装や使用に関する主張されるいかなる知的所有権およびその他の権利の正当性や有効範囲、もしくは、その権利下のいかなるライセンスもが利用できる可能性の範囲に関して判断を述べることを控える。公示に使用できる権利の主張のコピーといかなるライセンスの保証、もしくは、一般的なライセンス取得の試みの結果、もしくは、本仕様の実装者または使用者による所有権の使用の認可、GGF事務局から取得できる。

GGFは、本勧告の実践に必要となる可能性のあるテクノロジーを対象としたあらゆる著作権、特許、特許出願、その他の所有権に目を向けるよう勧める。これらに関する情報はGGF事務局長までお寄せいただきたい。

特定の主張を本文書に含めるとその主張が有効であり、一覧にしてある主張が網羅的であることを暗示するため、IPRに関する明示的な文章は本文書に含まれていない。文書が最終版のGFDであると公

開されたら、IPR情報を効果的に更新するメカニズムはない。その代わりに著者はGGF事務局長に対して明示的な文章または関連する主張を提供しなければならない。

著作権表示全文

Copyright © Global Grid Forum (2003). All Rights Reserved.

本文書およびその翻訳版を複製し他者へ提供しても構わない。本文書の内容へのコメントや説明および内容の実装に関する援助に関する派生的な成果物のすべてまたは一部を準備、複製、公開、配布することを、上記の著作権表示および本段落をそうした複製物や派生的な成果物に含めることを条件に、何らの制限なく行うことができる。

ただし、著作権表示やGGFまたはその他の組織への言及を削除するなどして本文書自体を修正することは、グリッド勧告書の発展を目的として必要な場合を除いて、行ってはならない。修正を必要とする場合は、GGFの文書作成プロセスに定義されている著作権手続きに従い、要求に応じて英語以外の言語に翻訳しなければならない。

上記において許可された制限付き認可は永続的なもので、GGFまたはその継承者や任命者によって無効にされることはない。

本文書および本文書に含まれている情報は「現状のまま」で提供されるものであり、GLOBAL GRID FORUMは、明示的か黙示的かを問わず、当該文書に含まれている情報を使用することがいかなる権利、商品性の保証、特定の目的への適合をも侵害しないことなどを保証しない。

参考文献

[AKENTI]

M. Thompson, A. Essiari, S. Mudumbai, "Certificate-based Authorization Policy in a PKI Environment", ACM Transactions on Information and System Security (TISSEC), Volume 6, Issue 4 (Nov. 2003) pp 566-588

[AZN-API]

"Authorization API Generic Application Interface for Authorization Frameworks", Open Group Technical Standard C908. <http://www.opengroup.org/publications/catalog/c908.htm>

[CARDEA]

Lepro, R., "Cardea: Dynamic Access Control in Distributed Systems", NASA Technical Report NAS-03-020, November 2003

[CASR2]

CAS AlphaR2 Web site, <http://www.globus.org/Security/cas/alpha-r2/>, September 2002.

[CAS02]

Pearlman, L., Welch, V., Foster, I., Kesselman, C. and Tuecke, S., A Community Authorization Service for Group Collaboration. IEEE 3rd International Workshop on Policies for Distributed Systems and Networks, 2002.

[CAS03]

Pearlman, L., Welch, V., Foster, I., Kesselman, C. and Tuecke, S., The Community Authorization Service: Status and Futures. Computing in High Energy Physics (CHEP03), 2003.

[Datagrid]

<http://datagrid.in2p3.fr/cgi-bin/cvsweb.cgi/Auth/VO/>

<http://www.gridpp.ac.uk/gridmapdir/>

http://datagrid.in2p3.fr/cgi-bin/cvsweb.cgi/fabric_mgt/edg-lcfg/

<http://cvs.infn.it/cgi-bin/cvsweb.cgi/Auth/voms/>

<http://www.gridpp.ac.uk/gacl/>

[EDG-SEC]

R. Alfieri et al. (EDG Security Co-ordination Group), "Managing Dynamic User Communities in a Grid of Autonomous Resources", Proceedings of Computing in High Energy and Nuclear Physics (2003).

[GAA-API]

<http://www.w.isi.edu/gost/info/gaaapi/>

[GOL96]

I. Goldberg, D. Wagner, R. Thomans, and E. Brewer “A secure environment for untrusted helper applications”, Proceedings of the Sixth USENIX UNIX Security Symposium, July 1996

[McCarthy1996]

McCarthy, J. et al, “LISP 1.5 Programmer s Manual”, MIT Press, Dec. 1969

[ISO-10183]

Information technology - Open Systems Interconnection Security Frameworks for Open Systems: Access Control Framework, ITU-T Recommendation X.812

[KERBEROS]

B. C. Neumann and T. Ts'o, “Kerberos: an Authentication Service for Computer Networks, IEEE Communications, 32(9):33-38, Sept 1994

[RADIUS]

C. Rigney, S. Willens, A. Rubens, W. Simpson. “Remote Authentication Dial In User Service (RADIUS)”. IETF RFC2865 June 2000. <http://www.ietf.org/rfc/rfc2865.txt>

[RSVP]

R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, S. Jamin “Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification”.. IETF RFC2205 September 1997. <http://www.ietf.org/rfc/rfc2205.txt>

[PERMIS] D.W. Chadwick, O. Otenko “The PERMIS X.509 Role Based Privilege Management Infrastructure” in proc. of the 7th ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES (SACMAT 2002), June 2002.

[PRIMA02]

M. Lorch, D. Kafura, “Supporting Secure Ad-hoc User Collaboration in Grid Environments”, 3rd Int.

Workshop on Grid Computing, Baltimore, Nov. 18th, 2002

[PRIMA03]

Markus Lorch, David Adams, Dennis Kafura, Madhu Koeneni, Anand Rathi, Sumit Shah, "The PRIMA System for Privilege Management, Authorization and Enforcement in Grid Environments", 4th Int. Workshop on Grid Computing - Grid 2003, 17 November 2003, Phoenix, AR, USA

[RFC2704]

M. Blaze, et. al. "The KeyNote Trust Management System", IETF RFC2704
<http://www.ietf.org/rfc/rfc2704.txt?number=2704>

[RFC2903]

C. de Laat, "Generic AAA Architecture" IETF RFC2903 <http://www.ietf.org/rfc/rfc2903.txt>

[RFC2904]

J. Vollbrecht, et. al. "AAA Authorization Framework" IETF RFC2904
<http://www.ietf.org/rfc/rfc2904.txt?number=2904>

[RFC3280]

R. Housley, W. Polk, W. Ford, D. Solo. "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List" IETF RFC, April 2002, <http://www.ietf.org/rfc/rfc3280>

[RFC3281]

S. Farrel, R. Housley, "An Internet Attribute Certificate Profile for Authorization." IETF RFC3218
<http://www.ietf.org/rfc/rfc3281.txt>

[SAML]

Assertions and protocol for the OASIS Security Assertion Markup Language (SAML) Committee Specification 01, 31 May 2002,
<http://www.oasis-open.org/committees/security/docs/cs-sstc-core-01.pdf>

[Shibboleth]

Internet2 Shibboleth Project, "Shibboleth Architecture" <http://shibboleth.internet2.edu/docs/draft-internet2-shibboleth-arch-v05.pdf>, 2 May 2002

[SLASHGRID]

The Slashgrid Project, <http://www.gridpp.ac.uk/authz/slashgrid/>, visited 2004-03-03

[VXE]

Virtual Executing Environment, <http://www.intes.odessa.ua/vxe>, visited 2004-04-04

[WSSEC]

Security in a Web Services World: A Proposed Architecture and Roadmap

<http://msdn.microsoft.com/library/en-us/dnwssecur/html/securitywhitepaper.asp>. Version 1.0.

April 7, 2002

[XACML]

OASIS eXtensible Access Control Markup Language (XACML) Committee Specification 1.0 (Revision 1), 12 December 2002

<http://www.oasis-open.org/committees/xacml/docs/s-xacml-specification-1.0-1.doc>

[XRML]

http://www.xrml.org/get_XrML.asp

付録A: 2つのドメイン認可モデルの分類

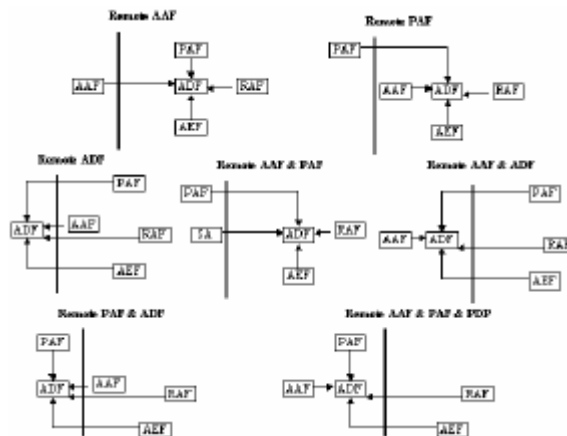
2つのドメイン認可

ドメインは機関が管理しているポリシーに従って制御されているメンバーシップを持つエンティティ(主体または資源など)の集合である。

2つのドメイン認可モデルは、あるドメインの機関が認可を必要とする資源集合を管理しているのに、認可アーキテクチャ中の1つ以上のコンポーネントが別のドメインに存在している時に生じる。こうした状況はグリッド・コンピューティングのシナリオだけでなくB2Bでも見られ、特に複数の物理的組織からの資源が仮想組織を形成するようにプールされているときによく見られる。

2つのドメイン認可モデルの分類

ここでは、(AEFは保護している資源と同じドメインに配置されていて、資源機関は権限を持っている資源と同じドメインに配置されているという制約下で)認可フレームワーク(属性アサーション機能(AAF)、アクセス制御判断機能(AEF)、ポリシーアサーション機能(PAF)、アクセス制御実施機能(AEF)、資源アサーション機能(RAF))の異なるコンポーネントがドメインの境界を越えて配布できるような異なる組み合わせを考慮する。5つのコンポーネントのうち2つを固定することで、残りの3つのコンポーネント(AAF、PAF、ADF)が外部ドメインに移動されるような異なる組み合わせを考えることができる。したがって、コンポーネントを外部ドメインに移動するような組み合わせが7通りあり、それぞれリモートAAF、リモートPAF、リモートADF、リモートAAF&PAF、リモートAAF&ADF、リモートPAF&ADF、リモートAAF&PAF&ADFというレベルが付けられている。



図A-1: 2つのドメイン認可モデル

前頁の図は、コンポーネントの構成を示しているものであって情報の流れを示しているものではないという点に注意されたい。たとえば、主体属性の共有を含むあらゆる構成について、その主体属性は資源にアクセスするための実際の要求に含まれている場合もあれば、アクセス要求を受け取った後に主体からAEF(あるいはその他のコンポーネント)によって要求される場合もあるし、主体ドメイン中の属性機関からのAEF(あるいはその他のコンポーネント)によって要求される場合もあり、何らかのオンライン・リポジトリ形式(LDAPやウォレットなど)で取り出される場合もある。資源属性、環境属性、判断のベースとなるポリシーを取り出すために同様のオプションが利用可能である。また、このデータが取り出される順番は環境によって異なり、要求によって異なることもある。したがって、2つのドメイン認可アーキテクチャにおける任意のコンポーネントの組み合わせに対してさまざまな情報の流れを実装することができる。

これら7種類の構成それぞれに関連した利点と課題を以下の節で簡単に述べる。

リモートAFF

この構成では、主体のドメインにある主体機関が、主体の関連属性を他のドメインにある資源の資源ドメインに対して利用可能にすることで、その資源に対する主体のアクセスを容易にする。ロール、署名機関、セキュリティ・クリアランスなどといった主体属性は主体ドメインで管理され、資源ドメインに通知される。このモデルはX.509 [RFC3281]、SAML [SAML]、Shibboleth [Shibboleth]などといった標準で定義されている。

このアプローチを成功させるには3つの障害物に対処しなければならない。1つ目は主体ドメインと全ての資源ドメインが主体属性のエンコーディングの文法とセマンティクスに合意しなければならないということ、2番目は属性の通知メカニズムについて合意する必要があること、3つ目は機密事項である可能性のある属性情報をネットワーク上で交換することに内在するプライバシーの問題である。

主体属性の文法やセマンティクスに関するドメインの合意に(国または国際的な標準化団体あるいは業界レベルで)成功した例はまだ限られている。

リモートPAF

この構成では、他のドメインの機関がポリシー(おそらく唯一のポリシー)を作成し、その上でADFが動作する。このような構成は、本社が作成した企業ポリシーが支店のローカルな資源についての認可判断を指示し、影響を与えるような大企業の問題環境に向いている。しかしこのモデルは、特定の業界分野内のローカルな環境に法的あるいは規制的ポリシーが課せられるような場合にも適している。この例として、

個々の医療保険組織が従う米国の医療保険の相互運用性確保及び説明責任に関する法律(HIPAA)に基づいてポリシーを定義した米国保健福祉省がある。このモデルは実際には認可判断に3つの関係するドメインがあることを暗示している。すなわちポリシーが定義されるドメイン、資源が保有されているドメイン、主体がメンバーになっているドメインの3つである。

リモートADF

この構成では、ADFはおそらくWebサービスとして実装されて他のドメインにアウトソースされる。資源ドメインは関連情報(ポリシー、属性、アクセス要求など)をすべてこのADFに送信し、ADFが返す認可判断を受け取る。資源ドメインが厄介な作業(関連する全ての規則は何かを知るのに必要なポリシーを理解し、その入力を全て探し、それぞれの認可、完全性、賞味期限を検証する)を全て遂行して単純なタスク(判断に際して到着したポリシーや入力を処理する)だけをアウトソースしているように見えるかもしれないが、このモデルにはおそらく何らかの長所があると思われる。アウトソースされたADFは多くのドメインにおいて認識される判断機関であるので、その判断は法的値あるいは同様の値を所有している(たとえば、監査やアーカイブを目的として不完全な認可判断のリスクを提言する)。

リモートAAF&PAF

この構成では、主体属性とポリシーがともに保護された資源の外部のドメインからくる。このモデルはデジタル権限管理(DRM)シナリオに適している。このシナリオでは、主体はその属性と何らかの形式のポリシー(おそらく「チケット」とともにやってきて、主体の要求が資源にアクセスするのをサポートする。またこのモデルは、主体が固有のプライバシー / ポリシーや私的選択をサブミットして資源ドメインのADFがそれを考慮しなければならないような、プライバシー問題に対処しようとしている環境にも適合する。

リモートAAF&ADF

この構成では、主体ドメインのADFが資源ドメインで作成されたポリシーに従って認可判断を下す。このモデル(「これが要求と資源とポリシー。そちらの主体がアクセスできるか否かを教えてほしい」)は、一般的に使用されているようには思えない。適用が考えられる場合の1つとして事前チェックのタイプがある。主体は自分のADFをチェックして、要求(およびその主体属性)を実際に資源ドメインに送信する前にどの要求が許可されるかを調べるというものである。この事前チェックにより、今行おうとしているアクセスにとってこの属性が必要でしかも十分であることを主体がある程度確信できるまで機密事項を扱う

属性が資源ドメインに送られないため、帯域をいくらか節約することができ、プライバシーを保護することが可能になる。

リモートPAF&ADF

この構成では、アウトソースされたADFが自分のポリシー下で動作して、資源ドメインに知られ管理されている主体に関する認可判断を下す。このモデルは、ポリシーが法律上規定されているものであり、しかもADFが監査または整合性テスト・センターの役を果たしているような場合に適用される。

リモートAAF&PAF&ADF

この構成では、ADFは自分のポリシーに従って主体ドメインの中で動作する。ADFは、資源ドメイン中のAEFが動作できるように認可判断証明を発行する。図に示した通り、資源ドメインが自分のADFに対して自分のポリシーに従って動作するようにほとんど強制しているため、これは単純化しすぎている。このモデルは、AAFとPAFが別のドメインにあるのでリモートAAF構成に似ており、AAF構成の場合と同様に(他に第三者が関与していない)「単純な」双方向のB2B関係に適しているように直感的には見える。しかしリモートAAF構成と異なるのは、このモデルがADFとPAFを用いて主体属性が「移動する」必要を「いたるところで」事前に除去することで、ドメイン境界を越える主体属性に関わるプライバシーの問題を回避している。

情報共有

前節で述べたさまざまな構成と、各構成のドメイン境界を越えて共有する必要がある情報のタイプを下の表に示した。

	主体属性	資源属性	認可判断	ポリシー
リモートAAF				
リモートPAF	*	*		
リモートADF				
リモートAAF&PAF		*		
リモートAAF&ADF	*			
リモートPAF&ADF				
リモートAAF&PAF&ADF				

リモートPAF、リモートAAF&PAF、リモートAAF & ADFにある記号「*」は、これらの構成についてはドメイン境界を越えて共有しなければならない特定の属性はないが、関連するポリシーの定義は属性分類に依存しているということを示している。たとえば、「ジョーは役員である」という形式のアサーションが境界を越える必要がない場合でも、関連する認可ポリシーを定義するために主体が社員か役員かをリモートPAFは知る必要がある。この要件は、PAFがAAFかRAFとは別になっているような構成の特徴である。

上記の表を見てわかる通り、リモートADF、リモートAAF&PAF、リモートPAF&ADFはいずれも主体属性共有に関係しているので、リモートAAFで判明したのと同じプライバシー上の問題に直面する。

また、リモートPAF、リモートADF、リモートAAF&PAF、リモートAAF&ADFはいずれもPAFとADFの分離に関係しているので、ポリシーがドメイン境界を(リモートPAFについては他のドメインから反対の方向へ)越える必要がある。現段階では、そのような情報を伝達するための標準的なプロトコルやデータ構造はほとんどない。X.509属性証明書の作業は特権ポリシーコンストラクトを属性証明書中で転送する方法を規定したが、これも広く採用、展開されてはいない。また、SAML「文」タイプを拡張して「ポリシーステートメント」を定義し、それをSAMLアサーションの中で他の種類の文と一緒に転送しようという議論もあるが、まだこの作業は始まっていない。

今回の議論では環境属性については説明していない。判断するADFはこのタイプの入力(たとえば時刻、現在の株価など)を必要とする。このデータはADFが動作しているドメインからくるか、または他の入力と一緒にドメイン境界を越える必要がある。環境データの出所はアクセス判断を決定する個別の要求や特定のポリシーに依存しているため、このレベルの詳細は上図には示していない。あるモデルにおいて環境データが常にこのドメインやあのドメインからくるとはつきりということは可能ではない。