■3 群（コンピュータネットワーク）－ 7 編（コンピュータネットワークセキュリティ）

2 章 アクセス制御

（執筆者：工藤道治）［2008 年 7 月 受領］

■概要■

アクセス制御とは，情報資源の不正利用を防止するために，決められた規則に従って資源へのアクセスを制限することである[1,2]. 正当なユーザにのみ情報のアクセスを許可することにより，システムやデータの機密性と完全性を保持する．本章では，アクセス制御固有の技術について解説する．

■3 群 － 7 編 － 2 章

2-1 概　要

（執筆者：工藤道治）［2008 年 7 月 受領］

アクセス制御は，情報資源に対するアクセスがすべて正当なアクセスであることを保証することであり，システムに対する不正操作の脅威（データの不正利用，漏えい，改ざん，データの破壊など）に対抗するために用いられる技術である．**図 2･1** に，一般的なアクセス制御の概念を示す．

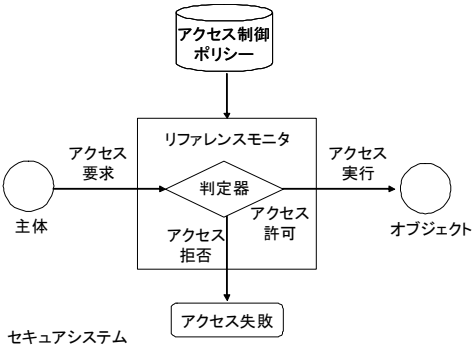

図 2・1　アクセス制御の枠組み

ユーザやシステムによって起動されたプロセス（主体）は，アクセス対象である計算機資源（オブジェクト）に対してアクセスを行う．リファレンスモニタ [1] は，すべてのアクセスが正当なアクセスであることを保証する機能要素であり，資源へのアクセスを監視しアクセス制御上の決定を調停する役割をもつ．具体的には，アクセス制御ポリシー，主体の種類，アクセス対象となるオブジェクトの種類などの情報に基づいて，アクセス許可か拒否のどちらかを決定する．

アクセス許可の場合はアクセスが実行され，アクセス拒否の場合はアクセス失敗となる．典型的なリファレンスモニタは，オペレーティングシステムの構成要素であり，その動作の完全性が検証された TCB（trusted computing base）の一部である．一般には，ハードウェア，ミドルウェア，アプリケーションなど，情報システムの様々なレベルでアクセス制御の機能が使われている．

2-1-1 構成要素

(1)　主　体

主体とは，システムのユーザやユーザに起動されたプロセスなど，システムの中の能動的な実体を意味する．以後では，「ユーザ」によって人間を表し，「主体」によってシステム内のユーザプロセスと人間の両方を表す．主体は，そのプロセスを生成したユーザの名前，ユーザ識別子，プロセス番号，接続点の IP アドレスなどの属性によって識別される．「開始者」，「要求者」，「プリンシパル」と呼ぶこともある．

(2) オブジェクト

オブジェクトとは，アクセスを受ける受動的な実体を意味する．ファイル，ディレクトリ，メモリ空間，データベースなど，ほとんどすべての計算機資源が該当する．また，アクセス制御に使われる判定規則そのものもオブジェクトの一つである．オブジェクトは，ファイルシステムのパス，URI，オブジェクト ID などの属性によって識別される．「客体」，「対象」，「資源」と呼ぶこともある．

(3) 権　限

権限とは，主体がオブジェクトに対して実行できる操作を意味する．例えば，オブジェクトがファイルの場合，読込み，書込み，実行などの権限がある．一般に，オブジェクトに対して実行できる操作の種類はそのタイプごとに異なりあらかじめ決まっている．権限は，それを更に制限する目的で使われる．「アクセス」，「アクセスモード」，「アクセス要求」，「権限」，「作用」，「特権」と呼ぶこともある．

(4) アクセス制御ポリシー

アクセス制御ポリシーとは，アクセスを許可する条件を定義した規則の集まりを意味する．一般に，アクセス制御ポリシーは，【主体，オブジェクト，権限】の三つ組みから構成され，「“主体”が“オブジェクト”に対して“権限”をもつ」ことを表明する [3]．個々の規則に条件式を追加して制限を強くしたり，「権限をもたない」という否定の表明も用いられる．「アクセス制御判定規則」，「アクセス制御情報」と呼ぶこともある．

2-1-2 アクセス制御用語

各標準での用語定義と本章での用法を以下にまとめる．

セキュリティポリシー

組織において機密情報の管理，保護，配布方法を規制する法律，規則，実践などの文書 [1]．

アクセス

主体とオブジェクト間で情報の流れが発生するような相互作用 [1]．

認　可

ユーザやプログラムなどにアクセス権を与えること [1]．アクセス制御と認可は，しばしば明確な区別なしに交換可能な用語として使われることがある．本章では，どのような条件のときにアクセス権を与えるべきかを決定するモデル・メカニズムを意味する用語として「認可」を用いる．

アクセス制御ポリシーモデル

アクセス制御ポリシーと同様の意味をもつが，システムとして実装されるより形式的・数学的なモデル．HRU モデルや Bell-LaPadula モデルなど．

アクセス制御スキーム

アクセス制御ポリシーモデルをシステム上で実装する場合に用いられる複数の実装手法 [4]．

アクセス制御メカニズム

アクセス制御を実現するためのハードウェアやソフトウェア，あるいはその組合せ [1].

2-1-3 国際標準

(1) ISO アクセス制御フレームワーク

ISO/IEC 10181-3 [4]は，オープンシステムのためのアクセス制御フレームワークを定めた国際標準である．フレームワークに基づいた汎用的なアクセス制御メカニズムをつくることにより，その上で複数の異なるアクセス制御ポリシーモデルの実装が可能になる．**図 2･2** はアクセス制御フレームワークを示す.

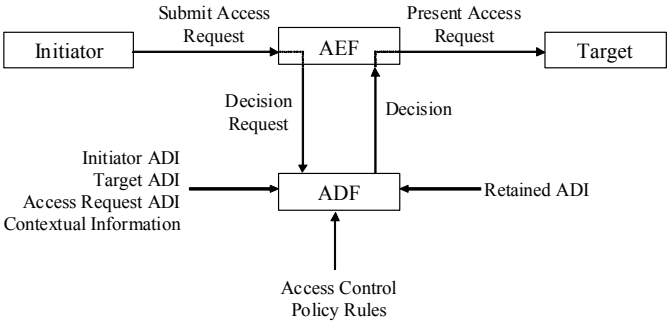

図 2・2 ISO/IEC 10181-3 **アクセス制御フレームワーク**

基本構成要素は，Initiator（開始者），AEF（アクセス制御執行機能），ADF（アクセス制御決定機能），Target（対象）である．ここで Initiator と Target は図 2･1 の主体とオブジェクトにそれぞれ該当する．AEF は情報資源を管理するシステム実体（Web サーバやデータベースなど）であり，ADF はアクセス許可・拒否の判定のみ行うシステム実体である．図 2･1 で示したリファレンスモニタは，ここでは AEF＋ADF に相当する．二つの役割に分離すると，ネットワーク上に分散した AEF が一つの ADF を共有することで，アクセス制御ポリシーの集中管理と判定を行うシステムが実現できる.

ADF への入力は，個々の決定要求から決められるアクセス制御決定情報（ADI），コンテキスト情報，アクセス制御ポリシー規則，保持アクセス制御決定情報（Retained ADI）である. ADI の例としては, ユーザの識別子やセキュリティクラスである. コンテキスト情報は，開始者の現在位置や認証強度，Retained ADI は，状態依存のアクセス制御を行う場合に必要となる情報を示す. ADF は上記の入力値からアクセス許可･拒否を決定し, AEF に回答する.

本標準は，代表的なアクセス制御スキーム（ACL スキーム，ケイパビリティスキーム，多階層セキュリティ）を本フレームワークに適用した場合のプロファイルについても記述している．アクセス制御ポリシーの表現形式やそのセマンティクス，構成要素間のセキュリティプロトコル，アクセス制御システムの管理方式などは本標準の範囲外である.

(2) OMG CORBA セキュリティ

CORBA（common object request broker architecture）は，OMG（object management group）が策定したオープンシステムのためのオブジェクト指向アーキテクチャである．OMG は CORBA の Security 仕様を定めた CORBA Security[5]の中でアクセス制御に関する仕様を決めている．

CORBA でのアクセス制御とは，オブジェクトが別のオブジェクトのあるメソッドを呼び出すことができるか否かを検証するプロセスである．**図 2･3** に示す CORBA のアクセス制御モデルは，オブジェクト呼出し層（オブジェクト呼出し時に必ず検証される）とアプリケーション層（アプリケーション独自の検証）の二層のアクセス制御から構成されている．前者のアクセス制御（client-side invocation access decision など）は，ORB（object request broker）と呼ばれる CORBA 標準のオブジェクトハンドラーによってオブジェクト呼出し時にアクセス制御が強制される．後者のアクセス制御（client application access decision など）は Client や Target Object などの個々のオブジェクトが任意に設定するアクセス制御モデルである．

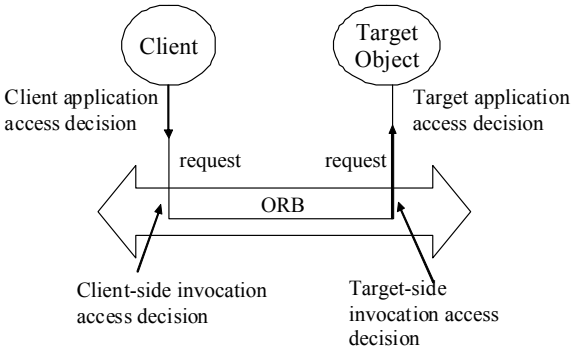

図 2･3 CORBA アクセス制御モデル

以下に，ORB によるアクセス制御について説明する．CORBA のアクセス制御ポリシーは，Domain Access Policy オブジェクトと RequiredRights インタフェースから構成される．前者は，オブジェクトにアクセスできる主体と権限を定義するオブジェクトである．後者は，オブジェクトを呼び出すときに必要な権限の集合を定義するインタフェース（オブジェクトをグルーピングする単位）である．アクセス判定のセマンティクスは次のように与えられる．ドメインアクセスポリシーを使って，呼出し元であるクライアントオブジェクトのユーザに許可された権限を取り出す．ターゲットオブジェクトに設定されている必要権限インタフェースから必要な権限を取り出す．二つの権限の集合がある指定されたクライテリア上で互いに適合しているかどうかを判定し，アクセスの許可・拒否を決定する．

(3) OASIS XACML

XACML（eXtensible Access Control Markup Language）[6]は，OASIS（Organization for Advanced Structured Information Systems）において 2003 年に策定されたアクセス制御ポリシー記述言語の国際標準である．本標準の特長は，汎用的かつ拡張可能なアクセス制御ポリシーのセマン

ティクスを定義していること，XML 言語による構文を定義していること，アプリケーション特有の拡張認可ポリシーに使われる豊富なデータ型と関数を標準で定義していることである．

XACML 言語では，アクセス制御ポリシーを Rule，Policy，PolicySet の三つの構成要素を組み合わせて記述する．Rule 要素は，アクセス制御ポリシーの最小構成単位である．Policy 要素は，複数の Rule 要素を結合し，一つのまとまったポリシーを表現する単位である．PolicySet 要素は，複数の Policy 要素または PolicySet 要素を結合する単位である．**図 2･4** に XACML 言語で記述したアクセス制御ポリシーの例を示す（一部記述の省略あり）．

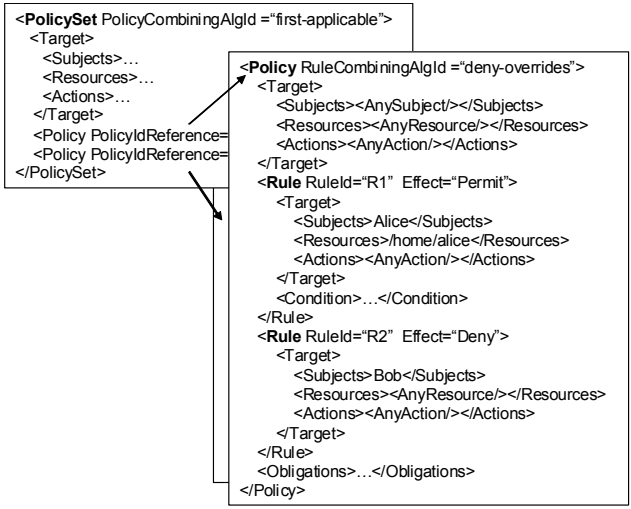

図 2・4　XACML アクセス制御ポリシーの記述例

Rule 要素は規則の適用範囲を表す Target 要素，アクセス許可・拒否を表す Effect 属性などから構成される．図 2･4 の 1 番目の Rule 要素は，“Alice”が“/home/alice”に対してすべての権限をもつことを表す．Condition 要素には拡張認可モデルのためのより粒度の細かい条件式を記述できる．2 番目の Rule 要素は，“Bob”からのすべてのアクセスを拒否するという規則を表す．図 2･4 の Policy 要素は，二つの Rule を“deny-overrides”（否定規則優先）アルゴリズムによって結合することを示している．Obligations 要素は，アクセス判定に付随する任意の必須処理を記述することができる．PolicySet 要素は，二つの Policy を“first-applicable”（記述順序優先）アルゴリズムによって結合することを示している．PolicySet 要素を使うことによって，個々の組織で管理するポリシーと組織全体で執行すべきポリシーとを組み合わせるような，柔軟なアクセス制御ポリシーの記述が可能になる．

■3 群 － 7 編 － 2 章

2-2 強制アクセス制御

（執筆者：工藤道治）［2008 年 7 月 受領］

2-2-1 強制アクセス制御とは

強制アクセス制御（MAC）は，主体やオブジェクトに割り当てられたセキュリティレベルと，形式的に定義された認可モデルとに基づいてオブジェクトへのアクセスを制限するアクセス制御の方式である [7]．強制アクセス制御は情報フローの厳密な制御を実現するために考え出された仕組みであり，管理者によって決められたセキュリティポリシーに従ってすべてのアクセスが決定される．そのため，ユーザ自身が生成したリソースであっても自由にアクセスできるとは限らない．当初から主に軍関連システムとして利用されてきたが，近年ではよりセキュアなオフィス用オペレーティングシステムとしても使われるようになってきた．

(1) 情報のラベル付け

政府機関などで重要な情報（特に軍事・諜報データ）を扱う場合，情報にラベル付けを行って管理する．例えば米国防総省では，文書のもつ情報の重要度を表すセキュリティレベルと関連分野を表すコード名とを組み合わせてセキュリティカテゴリ（またはコンパートメント）を生成し，それを個々の情報にラベル付けする．セキュリティレベル（または秘密区分）は，情報の機密性の度合いに応じて全順序が与えられる．例えば，区分外，機密，秘，極秘などの区分が用いられ，後ろの区分がより機密性が高いことを意味する．コンパートメントは，機密情報の分野や関連組織，地域などを表すものであり，例えば，{核，ミサイル}，{陸軍，海軍} といったものである．各々の項目はコード名と呼ばれ，互いの順序関係は与えられない．一つの情報には，{ミサイル，海軍} のように複数のコード名を与えることができ，それにより一つのコンパートメントが形成される．文書を扱う職員に割り当てられるセキュリティカテゴリは，使用許可と呼ばれる．情報のアクセス制御ポリシーは単純で，どの職員も自分に割り当てられた機密区分と同等かそれ以下の秘密区分の文書を読むことができる．例えば，「秘」を扱える職員は「機密」文書を読むことができるが「極秘」文書は許されない．更に，職員は自分に割り当てられたコンパートメントに適合する文書しか読むことができない．このようなラベル付けにより，セキュリティレベルの高い情報が間違ってセキュリティレベルの低い職員の手に渡ることなく，かつ担当外の情報を間違って取り扱うことを防止することができる．このようなアクセス制御ポリシーを，多階層セキュリティポリシーという．

2-2-2 Bell-LaPadula モデル

Bell と LaPadula は，情報のラベル付けを形式化した Bell-LaPadula（BLP）モデルを考案した [8]．BLP モデルは多階層セキュリティモデルとも呼ばれ，その後の米国政府におけるコンピュータセキュリティ標準の基礎となった．BLP モデルを実装するシステムを多階層セキュアシステムと呼び，SCOMP，System V/MLS，CMW，SELinux などが知られている．

BLP モデルの中心は，以下に示す二つのセキュリティ特性である．

- 単純セキュリティ特性：どの主体も，そのセキュリティレベルが dominate するオブジェクトに対してのみ読込みが許可される．“No read-up” とも呼ばれる．

- *-特性 : どの主体も，オブジェクトのセキュリティレベルが主体のセキュリティレベルを dominate する場合にのみ情報の書込みが許可される．“No write-down”とも呼ばれる．

セキュリティレベルが Dominate するとは, 主体のセキュリティレベルがオブジェクトのそれと同じか高位，かつオブジェクトのコンパートメントが主体のコンパートメントの部分集合になっていることを指す．単純セキュリティ特性は,「上位の者に対して情報が流れる」ことを，*-特性は「上位の者が下位に情報を流せない」ことを意味する．

ここで，あるユーザ s がファイル o に読込みアクセスをしているとする．s と o のセキュリティレベルは共に機密だと仮定する（単純セキュリティ特性は満たされている）．ここで，s が区分外の文書しか印刷できないプリンタに印刷をしようと試みる（印刷処理は書込み操作と同一に扱うことができる）．もし書込みを許可すれば，機密情報が区分外レベルの環境に出力されてしまい機密情報が漏えいしてしまう．印刷要求を拒否することは，*-条件を満足させることと同じである．ところで, *-条件を満たすためにはもう一つ別な方法が存在する．書き込む対象のセキュリティレベルを上げてから書込みを許可する方法である．プリンタの場合は機密レベルを上げることは現実的ではないが，ファイルに書込みをする場合などは可能である．システム動作中にセキュリティレベルを変更することは当初のモデルでは想定されていなかったが，後に平穏条件をモデルに導入しそれを可能とした．強い平穏条件では，システム動作中のセキュリティラベルの変更はできないが，弱い平穏条件では，セキュリティポリシーに反しない範囲でラベルの変更を許すのである．

BLP モデルでは，システムの初期状態が安全な状態であり，システムの状態遷移関数が単純セキュリティ特性と*-特性を満たすように制限されているならば，システムは常に安全な状態を保持することを証明した．これにより，BLP モデルの下でデータの機密性が保護されることが保証される．BLP モデルとその拡張モデルに関して，不干渉条件，非演繹性，制約性など多くの関連研究が行われている．

2-2-3 情報フロー制御

情報フロー制御 [9] は，多階層セキュリティポリシーを表現する束モデルを定義し，それによって情報の流れを制御する．情報フロー制御は BLP モデルと実質的に等価であることが知られている．

多階層セキュリティポリシーは，束＜SL, ≦＞によって構成される．SLはセキュリティラベルの集合, ≦はSL上の半順序集合を定義する．束モデルでは，上限演算子⊕と下限演算子⊗を定義し，それぞれ最も高位と最も低位のセキュリティラベルを計算する演算子とする．米国政府の多階層セキュリティポリシーの場合，セキュリティラベルは (L,C)として表現できる．Lはセキュリティレベル，Cはコンパートメントである．ここで，BLPモデルにおけるdominateは，“(L, C)≦(L’, C’) if and only if L≦L’ and C⊆C’”として定義できる．上限演算子⊕は“(L, C) ⊕ (L’, C’)＝(max(L∪L’), C∪C’)” として定義できる．

例えば，L＝{秘，極秘}かつ秘≦極秘，C＝{核，暗号}の場合，多階層セキュリティポリシーは**図 2･5** に示す束構造により表現できる．

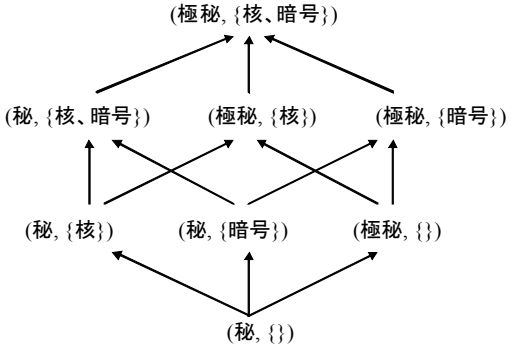

図 2・5　多階層セキュリティポリシーの束表現

単純セキュリティ特性により，(秘, {核})のラベルが割り当てられたユーザは，(秘，{核，暗号})や(極秘，{核})のラベルがついたオブジェクトの読込みができない．

2-2-4 隠れチャネル

隠れチャネルとは，通信用に設計されたチャネルではない計算機資源を悪用し，情報を高位から低位に送信する手法を意味する [10]．隠れチャネルは，多階層セキュアシステムにおいて高位のプロセスと低位のプロセスが何らかの資源を共有している場合に問題となる．例えば，両プロセスでディスクドライブを共有していると，高位プロセスがディスクドライブを意図的に満杯にすることで，低位のプロセスに何らかの信号を発することができる．ほかにも，CPU 使用率を意図的に高負荷にする，ファイルを意図的にロック／アンロックする，空でないディレクトリを削除するなど，様々な隠れチャネルが存在する．この問題を解決するためには，隠れチャネルに対するセキュリティ要求仕様を形式的に記述し，それを満たすシステムを構築する必要がある．具体的な対処方法としては，1) 資源を共有しない，2) 隠れチャネルにノイズを挿入したり遅延を設けることでバンド幅を制限する，3) 隠れチャネルの監査を行うことで利用を抑止する，などが有効である [11]．隠れチャネルのバンド幅を制限すると，一般にシステム全体の実行効率を低下させてしまう．実際には，バンド幅を毎秒 1 ビット程度にできるならば隠れチャネルは容認できるとしている．

■3 群 － 7 編 － 2 章

2-3 任意アクセス制御

（執筆者：工藤道治）［2008 年 7 月 受領］

2-3-1 任意アクセス制御とは

任意アクセス制御（DAC）は，主体の識別子やグループ名などに基づいてオブジェクトへのアクセスを制限する方式であり，オブジェクトに誰がアクセスできるかは，オブジェクトの所有者が任意に決定することのできるアクセス制御の方式である[7]．任意アクセス制御の特徴の一つは，オブジェクトの所有者にアクセス制御ポリシーの決定権を移譲することで，中央集権的ではなく柔軟にシステムが運用できることである．本章では，任意アクセス制御の代表的なモデルについて解説する．

2-3-2 アクセス行列モデル

アクセス行列モデルは，アクセス制御ポリシーを行列によって表現する任意アクセス制御のモデルである[12]．アクセス行列の行は主体を，列はオブジェクトに相当し，各々のセルは該当する主体がオブジェクトに対して実行できる権限の種類を表す．

	file1	file2	service1	service2
Alice	読込み		利用	
Bob		読込み 変更	利用	利用
Carol	所有 読込み 書込み		起動 終了	終了

図 2・6　アクセス行列の例

図 2･6 に示したアクセス行列の例では，主体である Alice は，オブジェクトである file1 に対して読込み権限と service1 に対して利用権限が許可されている．file2 のセルには何も記入されていないが，空のセルは権限を一つももたないことを意味する．行としてユーザの名前を使っているが，ほかの属性（所属するグループ）でもよい．アクセス行列を使うと，オブジェクトの所有という概念を「所有」権限によって容易に表現できる．図 2・6 では，Carol は file1 に対する「所有」権限をもっているため，file1 に対するアクセス制御ポリシーを自由に設定できる．アクセス行列をシステム上に実装するときは，以下に説明するアクセス制御リスト，ケイパビリティ，認可表などのスキームが使われる．

(1)　アクセス制御リスト

アクセス制御リスト（ACL）は，オブジェクト単位にアクセス行列を切り離し，列に属する各々のセルの情報をリストとして管理する方式である．図 2･6 の場合，file1 を表す列を取り出し，Alice がもつ権限と Carol がもつ権限とをリストによって結合しそれを file1 の属性として保管する．これにより，アクセス権をもたないユーザの情報を保持する必要がなくなり（Bob），管理すべきデータのサイズを小さくできる．UNIX や DCE などのファイルシステムのアクセス制御に見られる．

ACL スキームは，対象となるデータが頻繁に生成・削除され，ユーザの登録変更の頻度が少ない場合には，管理の観点から有利となる．一方，アクセスごとに ACL 構造体のチェックが必要であること，ユーザ情報の更新があった場合，全 ACL に更新を反映させることが容易でないことが欠点である．

(2) ケイパビリティ

ケイパビリティは，ACL とは反対に，行に属する各々のセルをリストとして管理する方式である．図 2･6 の場合，Alice を表す行を取り出し，file1 に対する権限と service1 に対する権限とをリストによって結合し，それを Alice の属性として保管する．ACL と同様，アクセス権のないオブジェクトの情報を保持する必要がなくなり（file2 と service2），管理すべきデータのサイズを小さくできる．ケイパビリティの利点と欠点は，ACL のそれと逆になる．ケイパビリティは，ACL スキームと組み合わされて使われることが多い．

(3) 認可表

認可表は，アクセス行列の空でないセルのみを表形式にしたものであり，主体，権限，オブジェクトの三つの列をもつ．図 2･6 の場合，「Alice, 読込み，file1」，「Alice, 利用，service1」などが認可表の行に相当する．認可表スキームは，主に関係データベースにおけるアクセス制御ポリシーの表現方式として使われる．

2-3-3 HRU モデル

アクセス行列モデルは，任意アクセス制御において柔軟なアクセス制御ポリシーを表現できる有用なモデルだが，一方でその表現力の高さが問題となる場合がある．例えば file1 は機密データであり，Bob はその読出し権限をもつことは許されないと仮定する．図 2･6 の状態ではその制約は満たされているが，この後も Bob が読出し権限をもつことはあり得ないと保証できるだろうか．Harrison, Ruzzo, Ullman はアクセス行列モデルを定式化し，安全性にかかわる性質を明らかにした [13]．ここで安全性とは，ユーザがオブジェクトに対してある種のアクセス権限を手に入れることができないという性質を表す．彼らの定式化したモデルを HRU モデルと呼ぶ．

HRU モデルでは，ユーザの集合 S，オブジェクトの集合 O，権限の集合 R，アクセス行列 A，コマンドの集合 C から構成される保護システムをモデルの構成要素とする．コマンドは条件部と実行部から構成される．条件部は，アクセス行列上でアクセス権限の有無をチェックし，実行部は，条件部が真になったときに実行される基本操作（アクセス行列の変更）の並びである．アクセス行列の初期状態 Q_0 において，あるユーザ $s \in S$ があるオブジェクト $o \in O$ に対してある権限 $r \in R$ をもっていなかったとき，あるコマンド $\alpha \in C$ を実行すると s が o に対して r をもつような状態 Q が存在するとき，Q_0 は権限 r に対して「安全でない」と定義する．この安全性問題は一般には決定不能であることが示されている．一方，汎用的な HRU モデルに制限を与えることによって安全性問題が決定可能になることが示されている．例えば，コマンドの実行部が一つの基本操作からだけ構成される場合である．アクセス行列モデルの定式化に関しては，多くの関連研究が知られている [14, 15, 16]．

2-3-4 論理型アクセス制御

アクセス制御のモデルを論理型言語の枠組みで定式化したものを，論理型アクセス制御ポ

リシーモデルと呼ぶ．論理型ポリシーモデルでは，アクセス制御規則を論理式の集合で表現する．アクセス要求は論理式の集合に対する問合せに相当し，推論規則に従って問合せを処理し，最終的にアクセス許可あるいは拒否を出力する．代表的な論理型アクセス制御ポリシーモデルについて解説する．

(1) Woo-Lam モデル

分散環境においては，複数のドメインが独立にアクセス制御ポリシーを策定・執行することが多い．個々のアクセス制御ポリシーを結合したとき，想定しなかった不整合が発見されるかもしれない．そこで，幅広いポリシーの表現と，複数ポリシーの合成セマンティクスを厳密に定義するために，論理型言語に基づく Woo-Lam モデルが考案された [17]．Woo-Lam モデルでは，アクセス制御ポリシーは「f : f' / g」と表現され，「g で表現される認可は，f で記述される認可条件が成立し，かつ f'に矛盾する認可が存在しない場合に成立する」を意味する．ここで f と f'は論理式，g は論理積だけで構成された論理式である．認可の例外条件，遷移的な認可，否定を含んだ認可など，非常に幅広いポリシー表現力をもつ．また，次項で説明する多階層セキュリティポリシーも同じ枠組みで扱うことができる．Woo-Lam モデルは，論理型言語に基づくアクセス制御モデルの構築可能性を示したが，アクセス判定の効率に問題があることが知られている [18]．

(2) ASL 認可モデル記述言語

ASL（Authorization Specification Language）は，Jajodia らにより考案された論理型言語に基づくアクセス制御認可モデルとその記述言語である [19]．ASL 言語は，アクセス制御ポリシーの表現力を Woo-Lam モデルより制限することで，アクセス判定を効率的に行うことができる．ここでは，ASL 言語の基本述語について簡単に説明する．

- cando 述語：セキュリティ管理者がアクセス許可，拒否を明示的に記述する述語．“cando(file1, ‘Alice’, +r)”は，Alice が file1 を読込み可であることを意味するファクト．“+r”は読込み許可，“-r”は読込み不可を表す．
- dercando 述語：cando 述語で指定されたアクセス制御規則を，サブジェクト階層やオブジェクト階層上で伝播させるために使われる述語．例えばあるグループに対して記述された規則をそのメンバー全員に適用させるときに使われる．“dercando(O, S, +A) :- cando(O, S’, +A) & in(S, S’)”は，「グループ S’に対して O に A という権限が与えられているならば，S’の下位に位置するグループ S に対しても同様の権限を認める」を意味する規則．
- do 述語：dercando 述語で表現された規則の中で矛盾が発生した場合（アクセス許可と拒否が同時に成立），その矛盾解消の方式を表現する述語．“do(O,S,+A) :- dercando(O, S, +A) & not(dercando(O, S, -A))”は，「アクセス許可の規則が導かれるならば，否定の規則が導かれない限り，アクセスを許可する」を意味する規則．
- in 述語：主体とオブジェクトの階層を表現するための述語．“in(‘Alice’, group1)”は，Alice が group1 に所属している事実を意味するファクト．

ASL 言語は層状論理プログラムの性質をもつため，否定を含む論理式を効率的に計算できる．また，論理式の集合を具体化（materialization）する技術を適用することにより，アクセス判定を非常に高速に決定できる [18]．

■3 群 － 7 編 － 2 章

2-4 アクセス制御ポリシーモデル

（執筆者：工藤道治）［2008 年 7 月 受領］

本節では，任意アクセス制御，強制アクセス制御以外の典型的なアクセス制御ポリシーモデルについて解説する．

2-4-1 ロールに基づくアクセス制御

一般に，規模の大きな組織ではその中で特定の役割を担うロール（職務）の集合から構成されている．それぞれのロールは異なる責任をもち，行うべき仕事をあらかじめ定義することができる．例えば，医師，看護士，臨床医，薬剤師などは病院におけるロールであり，医師は薬を処方する権限を，看護士は患者の容態を記録する権限をもつ．このようにロールの概念を積極的にアクセス制御の枠組みに取り入れたものがロールに基づくアクセス制御（RBAC）というアクセス制御の方式である [20]．RBAC では，ユーザとアクセス権は必ずロールを介して結びつけられる．任意アクセス制御とは異なり，ユーザがロールに対するアクセス権を自由に設定することはできない．

ロールと類似の概念としてグループがある．厳密には，グループは主体の一覧を意味し，ロールは主体に一定期間割り当てられるアクセス権限の集まりを意味する．一般に，グループは所属組織や給与レベルなどで静的に決められるが，ロールは業務の状況に応じて動的に割り当てられる．例えば，企業における部長は「部長グループ」に所属し人事異動以外では変更されない．一方「部長ロール」は部長職に対して許されている権限の集まりであり，部長の不在時（休暇や病気）には代行者に一定期間割り当てられる．

RBAC のモデルでは，個々のロールには実行可能なトランザクションの集合と割当て可能なユーザの集合とが関連付けられる．トランザクションとは，データに対するある種の変換手続きであり，オブジェクトと権限の組によって表現される．また，ロールは階層構造をもつことができる．個々のユーザは，ロール割当て機能を介して動的にロールを選択するかあるいは割り当てられる．アクティブロールは，実行時にユーザに割り当てられるロールを意味する．**図 2･7** は，RBAC のアクセス制御ポリシーの例である．

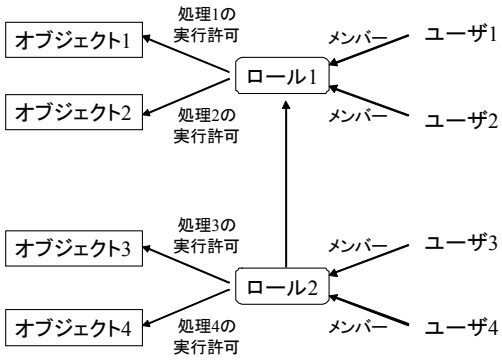

図 2･7 RBAC のアクセス制御ポリシーの例

ユーザ 1 はロール 1（例えば医局のインターン）のメンバーであり，オブジェクト 1（例えば医療カルテ）に対して処理 1（例えば閲覧）を，オブジェクト 2 に対して処理 2 を許可されている．一方，ロール 2（例えば医師）に属するユーザ 3 は，ロール 2 に対して許可された処理（例えば医療カルテへの書込み）と同時に，ロール 1 に許可された処理も実行できる．例えば，ユーザ 3 がインターン生を教育する状況で病院の医療カルテにアクセスする場合，閲覧操作のみが許可されるインターンロールを選択することにより，カルテに対する変更操作を禁止することができる．

2-4-2 Clark-Wilson モデル

多くの企業では，多階層セキュリティモデルに代表される機密データの保護よりも，データの完全性を保証することにより重点を置いている．例えば，銀行の預金システムにおいて預金残高の完全性（改ざんされていない，収支が合っているなど）を保証することは，おそらく預金残高の機密性を保つことよりも重要である．そこで，Clark と Wilson は，データの完全性の問題を解決するモデルを考案した [21]．

Clark-Wilson モデルは無制約データ項目（unconstrained data item: UDI），制約データ項目（constrained data items: CDI），変換プロシージャ（transformation procedures: TP），整合性検査プロシージャ（integrity verification procedures: IVP）から構成される．UDI は完全性が検証されていない状態にあるデータ，CDI は完全性が検証されたデータを意味する．IVP は CDI の完全性を検証する機能，TP は UDI や CDI から別の CDI を生成する変換機能である．処理の流れとしては，UDI がシステムに入力されるとき，CDI に変換される．IVP が CDI の妥当性を検査する．TP は，CDI に対して行う業務トランザクションであり，CDI の完全性を維持することを保証する．**図 2･8** に Clark-Wilson モデルの構成図を示す．

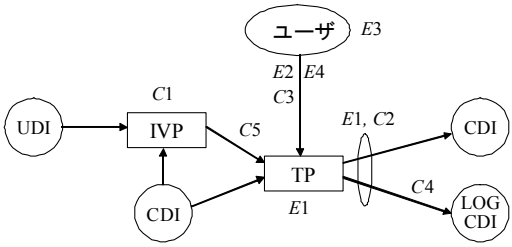

図 2・8　Clark-Wilson モデル

図に示された記号（*C*1, *E*1 など）は，Clark-Wilson モデルで満たすべき制約と執行条件であり，次のように定義されている．

- *C*1: すべての CDI についてその完全性を保証する IVP が存在しなければならない．
- *C*2: すべての TP は，CDI の変更に対してその完全性を維持しなければならない．
- *C*3: *E*2 のアクセス権限は，責務の分離の原則に従わなければならない．

- *C*4: すべての TP は, 監査のために必要なすべての情報を書換え不可の CDI に書き込まなければならない.
- *C*5: UDI を CDI に変換する TP の正当性は保証されなければならない.
- *E*1: システムは，*C*2 で示された関係を管理しかつそれを執行する.
- *E*2: システムは，適切なアクセス制御ポリシーを執行しなければならない.
- *E*3: システムは，各々のユーザを認証しなければならない.
- *E*4: アクセス制御ポリシーの変更はセキュリティ管理者だけが行える.

*E*2 によって TP に対するアクセス制御が行われる．アクセス制御ポリシーは,【主体，TP, CDI】の組に対して定義する．これにより，CDI の完全性を維持することが保証される.

2-4-3 チャイニーズウォールポリシーモデル

過去のアクセス履歴によってアクセスできるデータを動的に決定しなければならないことがある．例えば，コンサルタント会社において，利害の対立する複数の会社の機密データを一人のコンサルタントが同時にアクセスすると，その人間を通して競合会社間で機密情報の漏えいが起こる可能性がある．この問題を防ぐには，あるデータにアクセスすると，それ以降はそのデータと競合するほかのデータへのアクセスを拒否する（隔離する）ことから，チャイニーズウォールポリシーモデルと呼ばれる [22].

チャイニーズウォールポリシーモデルの中心は，次の示すセキュリティ特性である.

- 単純セキュリティ規則 : 主体 s がオブジェクト o を読み込むことができる条件は，s が同じ会社のデータセットに既にアクセスしているか，または利害の対立しないほかのクラスのデータにアクセスする場合.
- *-条件 : 主体 s がオブジェクト o に書き込める条件は，単純セキュリティ規則でアクセスが許可されている場合．ただし，もし書き込むデータが元の会社の機密情報などを含んでいる場合は，書き込まれたデータは誰も読み込むことはできない.

単純セキュリティ規則は機密情報の直接的な漏えいを防止する．一方，*-条件は二者以上の主体の結託による間接的な情報漏えいを防止する．**図 2･9** の例で説明する.

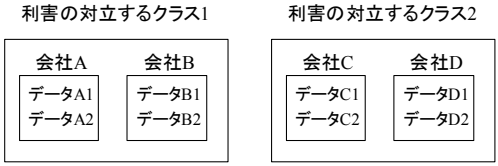

図 2･9　チャイニーズウォールポリシーモデルの例

会社 A と C にアクセスが許されているユーザが, 会社 A のデータ A1 を読み込んでから会社 C のデータ C1 に書込みを行い，会社 C と B にアクセスが許されている別のユーザが，会社 C のデータ C1 を読み込み，会社 B のデータ B1 に書込みを行う．競合会社 B にデータ A1 の情報フローが発生してしまうが，*-条件がこのような間接的な漏えいを防止する.

■3 群 － 7 編 － 2 章

2-5 認　証

（執筆者：工藤道治）［2008 年 7 月 受領］

インターネットの普及に伴い，認証（authentication）という用語は様々な場面で使われている．情報システムにアクセスを要求しているユーザが正当かどうかを判定するプロセスをユーザ認証や本人認証という．また，ユーザがアクセスしているサーバが，正当なサーバであることを判定するプロセスをサーバ認証という．情報機器どうしの通信時に，通信相手が正当な機器であることを識別するプロセスを機器認証という．ほかにも，メッセージ認証や時刻認証などの用語があるが，本節ではよく使われるユーザ認証，サーバ認証，機器認証について説明する．

2-5-1 ユーザ認証

情報システムにアクセスを要求している人間（ユーザ）が正当かどうかを判定するプロセスをユーザ認証という．システムがユーザを認証するためには，ユーザ情報をあらかじめシステムに登録しておく必要がある．名前やパスワードなどユーザに固有な情報を決めてシステムやサービスにユーザ登録申請を行う．サービス提供者側で申請内容を審査し，問題がなければ，ユーザ登録データベースにユーザ固有情報を登録し，ユーザ ID を割り当てる．ユーザがシステムやサービスにアクセスするときは，ユーザ ID と登録固有情報をシステムに提示してユーザ認証を受ける．

ユーザ認証に使われる固有情報として，一般に次の三つの認証タイプが知られている．

- 知識による認証
- 所有物による認証
- 身体的特徴（バイオメトリクス）による認証

知識による認証は，パスワードや暗証番号を用いたユーザ認証である．この場合，ユーザしか知らない固有情報を他人に知られないように保護することが重要なセキュリティ要件になる．所有物による認証は，IC カードや印鑑，身分証明書などのユーザの持ち物によるユーザ認証である．この場合，所有物の盗難や複製が容易にできない改ざん防御機能をもつことが重要なセキュリティ要件になる．身体的特徴を使う認証は，指紋や網膜，顔などの生体情報を用いたユーザ認証である．生体情報の特性として，データが生体として唯一性をもつこと，偽造が困難であること，経年変化が少ないことなどが望ましいとされる．三つの認証タイプは，二つ以上の方式を組み合わせて使うことによって，より確かなユーザ認証の実現が可能である．例えば，IC カードと保護パスワードにより，所有物と知識の二つのタイプでユーザ認証を行うことができる．公開鍵インフラストラクチャ（Public Key Infrastructure: PKI）[23] を用いたユーザ認証としては，SSL/TLS が代表的である．

SSL（Secure Socket Layer）は，Web サーバと Web ブラウザ間の HTTP 通信をセキュアにするために Netscape 社によって開発され，その後 IETF が SSL Version 3 をベースに TLS（Transport Layer Security）[24] と呼ばれる標準仕様を公開した．クライアント認証を行うため

には，PKI の認証機関（Certification Authority: CA）にユーザ登録し，公開鍵証明書とプライベート鍵を受け取る．プライベート鍵のデータはユーザ側で安全に保管しておく．サーバがユーザ認証するときは，サーバが送信したチャレンジ（乱数）に対してユーザ側でディジタル署名を生成し，公開鍵証明書と共にサーバに送信する．サーバはディジタル署名が正しく生成されているか検証し，正しければユーザを認証する．このように SSL/TLS はユーザ認証の機能を提供しているが，普及のレベルは限定的である．

2-5-2 サーバ認証

ユーザがアクセスしているサーバが，正当なサーバであることを判定するプロセスをサーバ認証という．前述の SSL/TLS の中で規定されているサーバ認証の方式は世の中で広く普及している．サーバ運営組織が認証機関（CA）に登録申請し，サーバに対する SSL 証明書とプライベート鍵を受け取る．ユーザがサーバにアクセスするときは，サーバから送られる SSL 証明書の内容を検証して問題なければサーバを認証する．サーバの認証手順の一部を以下に示す．

1. 現在の日付がサーバ証明書の有効期間内にあることを確認する．
2. SSL 証明書の発行元認証機関（CA）が信頼できる CA であることを確認する．
3. 発行元 CA の公開鍵を用いて送信サーバのディジタル署名を検証する．
4. SSL 証明書のドメイン名がサーバ自体のドメイン名と一致するか確認する．

以上が正しく検証できたとき，SSL/TSL サーバ認証の観点で正当なサーバであることを確認することができる．しかし，サーバドメインの所有者確認だけで取得できる SSL 証明書を悪用し本物のサーバと酷似したサーバ証明書をつくられてしまうと，一般ユーザには本物と偽物のサーバの区別が難しいという問題があった．これを解決するため，SSL 証明書の取得に際して，より厳格な実在証明（運営組織の法人登記，申請責任者の権限の明確化など）を要求する新しい EV SSL 証明書（Extended Validation SSL Certificates）の規格が 2006 年に制定された [25]．更に，EV SSL 対応ブラウザで EV SSL が導入されたサイトにアクセスすると，アドレスバーが緑色になり，ユーザにも視認しやすい表示方法が採用された．このような管理プロセスの改善とソフトウェアの改良により，サーバ認証をより効果的に行うことが可能になった．一方，サーバのソフトウェア構成や設定ファイルなどの問題によって発生するぜい弱性の有無を検証するためには，次節で述べる機器認証やトラステッド・コンピューティングの技術が必要になる．

2-5-3 機器認証

機器認証は，通信している機器が本当にその機器かどうかを識別し，必要な機能やインテグリティを満たしているかを判定することが目的である．近年のディジタル家電の普及により，家庭内で複数の家電製品間でデータ送受信や制御を行う必要が高まってきた．例えば，著作権のついたディジタルコンテンツの安全な交換や，ネットワーク接続された家電製品の遠隔地アクセスといった利用シナリオにおいて，機器認証は今後重要な役割を果たすと考えられている．新エネルギー産業技術総合開発機構（NEDO）が主催する「ディジタル情報機

器相互運用基盤プロジェクト」[26]では，情報家電における機器認証の技術・運用管理が検討されており，標準仕様書が公開されている．

トラステッドコンピューティンググループ（Trusted Computing Group: TCG）[1]は，信頼できるコンピューティング・プラットフォームを構築するハードウェア・ソフトウェアの業界標準の策定及び普及を目的とした業界団体である．IT機器のインテグリティを測定・検証するための新しいITインフラストラクチャの構築を目指しており，パーソナルコンピュータや携帯電話といったユーザ端末と，サーバやストレージデバイス，ネットワーク基盤といったITインフラストラクチャの両方で標準化が進んでいる．

TCG におけるトラスト（Trust）とは，「コンピュータシステムがユーザの意図したとおりに動作することが期待できること」と定義されている．このトラストモデルでは，システム起動時に何らかの信頼できるプログラムコードの存在が仮定されている．その信頼できるコードが次に実行されるプログラムコードの信頼性を判定し，もし信頼できる場合は推移的にトラストを拡張していく Transitive Trust と呼ばれるモデルである．TCG では，TPM（Trusted Platform Module）と呼ばれるセキュリティチップとそのプログラムコードに対してトラストルート（Root of Trust）を置き，ハードウェアから BIOS，カーネルローダー，オペレーティングシステム，アプリケーションに至るトラストチェーン（Chain of Trust）が構築される．このトラストチェーンと呼ばれる考え方がトラステッドコンピューティングの重要な技術要素の一つとなっている．遠隔地から IT 機器のインテグリティを安全に測定・検証するためのプラットフォーム・トラスト・サービス（Platform Trust Service: PTS）と呼ばれる仕様[27]を公開している．

[1] https://www.trustedcomputinggroup.org

■3 群 － 7 編 － 2 章

2-6 アクセス制御システム

（執筆者：工藤道治）［2008 年 7 月 受領］

2-6-1 OS のアクセス制御

UNIX のファイルシステムは ACL スキームを使った任意アクセス制御を実装している [28]．ユーザは UID と呼ばれる属性で識別され，複数のグループ（GID 属性で識別）に属することができる．UNIX には，オブジェクトの所有者（owner）という概念があり，すべてのファイルやプロセスはあるユーザの所有物である（UID 属性によって識別）．所有者は任意のアクセス制御ポリシーを ACL として設定することができる．

図 2･10 は，ファイル「contract.txt」に設定された ACL の例である．

モードビット	所有者	グループ	ファイル名
-rw-r-----	alice	marketing	contract.txt

図 2・10　UNIX の ACL の例

ファイル「contract.txt」の所有者が「alice」であり，ファイルはグループ「marketing」に属しており，このファイルへの権限は「-rw-r-----」で表されるモードビット（あるいはパーミッションモード）によって決められる．モードビットは，ファイルの所有者，グループ，それ以外の主体の権限をまとめて指定するためのデータ構造である．モードビットの 2 文字目から 4 文字目が owner モードビット，5 文字目から 7 文字目までが group モードビット，8 文字目から 10 文字目までが other モードビットと呼ばれる．権限の種類は，「r」が読込み権限（read），「w」が書込み・変更権限（write），「x」が実行権限（execute）を表現し（ディレクトリの場合は探索権限），三つの権限はこの順番に並べられる．該当する権限が「-」の場合は権限がないことを表す．ファイルに対するアクセス判定は，1) ファイル（オブジェクト）の UID がプロセス（主体）の UID と同じときは owner モードビット，2) ファイルの GID がプロセスの GID のリストのどれかと同じときは group モードビット，3) 前の二つに当てはまらないときは other モードビットを使い，該当する権限があればアクセスが許可される．

図 2･10 の場合は，contract.txt ファイルの所有者である alice はファイルに読み書きができ，marketing グループに属するユーザは読込みだけができる．それ以外のユーザは読むことも書くこともできない．所有者は，chmod コマンドを使って自分の所有するオブジェクトの ACL を自由に変えることができる．例えば，すべてのユーザにファイルの読込み権限を与える場合は，chmod コマンドを使ってモードビットを「-rw-r--r--」に変更すればよい．

DFS [29]は UNIX のアクセス制御を拡張し，よりきめの細かいアクセス制御を可能にしている．例えば，read, write, execute, control, insert, delete の六つの権限を使って，unauthenticated など 20 のエントリに対して ACL を設定できる．

(1)　SELinux

SELinux（Security-enhanced Linux）は，Linux®のセキュリティ機能を改善するためにつくられた研究用システムであり，米国国家安全保障局（NSA）が中心となって開発が進められ

ている[2]. 企業などで使用するオペレーティングシステムは任意アクセス制御を適用したものが一般的だが，強制アクセス制御の機能を追加することで，機密性と完全性の要件を満たしたより安全なIT基盤を実現することを目指している．SELinuxは，型強制（TE），ロールに基づくアクセス制御（RBAC），多階層セキュリティ（MLS）の三種類のアクセス制御機能に対応している．

SELinux のセキュリティアーキテクチャでは，システム上のすべての主体とオブジェクトに対してセキュリティコンテキストと呼ばれるセキュリティラベルが付加される．主体には domain 属性や role 属性が，オブジェクトには type 属性などが割り当てられる．主体やオブジェクトを新しく生成するときは，生成プロセスや親ディレクトリなどに割り当てられたセキュリティコンテキストを使って新しいセキュリティコンテキストを決定する．

主体がオブジェクトにアクセスするとき，主体のもつセキュリティコンテキストとオブジェクトのもつセキュリティコンテキストの組合せでアクセスの許可・拒否を決定する．このとき，組み合わせることのできる domain 属性と type 属性のペアに制限を設けてアクセス制御することを型強制（TE）と呼ぶ．また，role 属性を使って型強制で許可される権限を更に制限してアクセス制御する（RBAC）．なお，多階層セキュリティアクセス制御はオプションである．

セキュリティコンテキストの生成とアクセス制御判定は，セキュリティサーバが行う．システム資源とセキュリティコンテキストの対応を管理するのが，オブジェクトマネージャーである．SELinux の強制アクセス制御の機能は，Linux の従来の任意アクセス制御の機能とは独立しており，強制アクセス制御で許可された場合のみ任意アクセス制御がチェックされる．以上のアクセス制御機能により，root 特権を分散させたり，不正なプログラムを実行することによるダメージを制限することができる．

2-6-2 プログラミング言語のアクセス制御

Java 2 プラットフォーム（Java 2 Platform Standard Edition 1.4, J2SE 1.4）には，JAAS（Java™ Authentication and Authorization service）と呼ばれる認証とアクセス制御のためのフレームワークが組み込まれている．以下に JAAS 1.0[30]のアクセス制御の機能について説明する．

Java ランタイムのクラスローダーは，ロードしたクラスと，そのクラスに与えられたパーミッションの集合とを関連付ける．クラスがどこからロードされたか，誰がそのクラス（パッケージ）に署名を与えたか，誰がそのコードを実行しているか，などの条件を用いてアクセス制御ポリシーを設定できる．システムのデフォルト構成では，Java のホームディレクトリ下の jre/lib/security ディレクトリにある java.policy ファイルに記述する．**図 2･11** は，java.policy ファイルに記述されているポリシーのサンプルである．

[2] http://www.nsa.gov/selinux/

```
grant {
    permission java.util.PropertyPermission "java.version", "read";
};

grant codeBase "file:${java.home}/lib/ext/*" {
    permission java.security.AllPermission;
};
```

図 2・11　JAAS 1.0 のアクセス制御ポリシーの例

一つ目の grant 文は，現在実行中の Java ランタイムのバージョン番号プロパティでは，誰でも（プログラムの実行者やダウンロード元に関係なく）読むことができることを示している．二つめの grant 文は，ダウンロード元の場所を制限して権限を与えている例である．Java のホームディレクトリの下にある lib/ext サブディレクトリに格納されているコードに対して，すべての Permission を与えている．コードの署名者を指定する場合は“SignedBy”タグを，コードの実行者を指定する場合は“Principal”タグをそれぞれ使って記述する．

■3 群 － 7 編 － 2 章

2-7 ファイアウォール

（執筆者：工藤道治）［2008 年 7 月 受領］

ファイアウォールとは，ネットワーク境界を通過しようとするパケットの伝送を制御するハードウェアまたはソフトウェアの総称である．ネットワーク境界とは，ネットワークとネットワークが相互接続されている部分であり，両者のセキュリティレベルが異なる場合，ファイアウォールを用いて適切なセキュリティ制御を行う．ファイアウォールでは，ネットワーク管理者あるいはセキュリティ管理者によって定義されたセキュリティポリシーに従ってパケットの通過を許可あるいは遮断する．この機能の利点は，外部から悪意のある攻撃が内部に侵入する前に防御することができること，また通過したパケットを元にトラフィックをすべて記録することによって監査性が向上することがあげられる．**図 2･12** は，イントラネットとインターネットのネットワーク境界に設置されたファイアウォールを含むネットワーク構成図である．

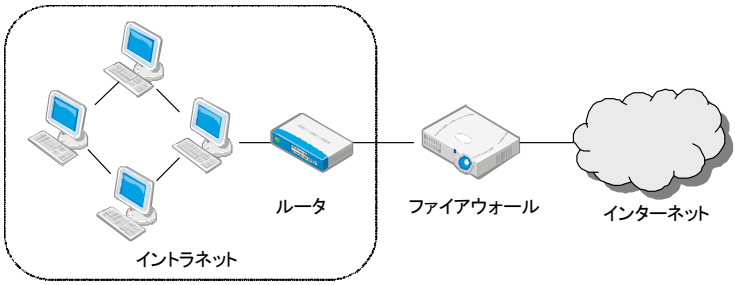

図 2・12　ファイアウォールとネットワーク構成

一般にファイアウォールは，以下に示す二つの基本的なセキュリティ機能を提供する．

- パケットフィルタ機能　―　ネットワーク管理者あるいはセキュリティ管理者が作成したセキュリティポリシーに基づき，データパケットの通過を許可するか遮断するアクセス制御機能を提供する．
- プロキシサーバ　―　ネットワークとネットワークの間の IP フローを制御し，一方のネットワークに設置された IT 機器を保護するために，もう一方のネットワークのアクティビティと適切に切り離してネットワークサービスを提供する．

パケットは，ヘッダとデータから構成され，前者にはパケットサイズ，送信プロトコル・ポート，送信元・送信先の IP アドレスの制御情報などが，後者にはデータのコンテンツが格納される．パケットフィルタ機能は，パケットのヘッダの値を評価することで通過の是非を判断し，通過拒否の場合はパケットを破棄する．一方，プロキシサーバは，パケットのデータを評価し，セキュリティルールに従ってパケットを宛先に転送する．以下の節で，パケットフィルタとプロキシサーバについて説明する．

2-7-1 パケットフィルタ

パケットフィルタとは，パケットヘッダの情報を利用してパケットの通過を許可するか，遮断するかを判断するファイアウォールの基本的なセキュリティ機能である．パケットフィルタには，ステートレス・パケット・フィルタリング（静的パケットフィルタリング）と，ステートフル・パケット・フィルタリングの 2 種類のアプローチがある．ステートレス・パケット・フィルタリングは，実際に確立しているネットワークコネクションに基づいて，パケットを許可するか破棄するかを決定する方法である．例えば，TCP ポート 80（HTTP）へのアクセスは，すべての外部ホストに許可するが，TCP ポート 23（Telnet）へのアクセスは，内部ホストからの接続のみ許可し外部ホストには許可しないといった制御ができる．また，TCP ヘッダの値（例えば ACK フラグ）に特定の値がセットされている場合，外部ホストからの接続を許可する制御も可能である．**図 2･13** は，任意のソース IP アドレスから 192.0.2.1 の目的 IP アドレスの TCP ポート 80（HTTP）及び 443（HTTPS）に対するアクセスを許可するポリシーの例である．

プロトコル	転送プロトコル	ソースIP	ソースポート	宛先IP	宛先ポート	判定
HTTP Inbound	TCP	*.*.*.*	*	192.0.2.1	80	許可
HTTPS Inbound	TCP	*.*.*.*	*	192.0.2.1	443	許可

図 2・13　ファイアウォールのポリシーの例

ステートフル・パケット・フィルタリングは，接続状態を内部に記録することにより，パケットが本物か偽物かを判別することができるより高度なパケットフィルタである．パケットの接続状態を記録するため，ステートテーブルと呼ばれる内部データ構造を保持する．ステートフル・パケット・フィルタリングでは，通過するパケットに対してセキュリティルールとステートテーブルの両方に問合せを行う．例えば，内部ホストが TCP ポート 80（HTTP）にアクセス可能であるというセキュリティルールがあると仮定する．このとき，内部ホストがある外部ホスト www.host.com に接続を試みる場合，初めにステートテーブルに接続が存在しないことを確認し，次にステートテーブルに接続エントリを作成する．その後セキュリティルールによる判定を行い，接続を許可する．www.host.com から応答を受信したときは，ステートテーブルにエントリがあるか確認し，エントリが存在すれば受信パケットの通過を許可する．

制御パケットである ICMP（Internet Control Message Protocol）では，正式なセッションの場合でもステートテーブルに記録されたソース IP とは異なるアドレスから応答がくる可能性があり，正しい ICMP パケットが誤って破棄されてしまう問題が発生する．この場合，ヘッダだけでなくパケットのデータの内容も調べて正しいパケットであることを確認する（ステートフル･インスペクション）．

2-7-2 プロキシサーバ

プロキシサーバは，パケットのデータ部分を調べて，その内容とセキュリティルールに従って動作するファイアウォールソフトウェアである．プロキシサーバと類似の機能を提供す

るものとしては，プロキシサービス，アプリケーションプロキシ，アプリケーション・レベル・ゲートウェイ，アプリケーション・プロキシ・ゲートウェイのように様々な別称があるが，本節ではプロキシサーバという用語を使用する．プロキシサーバの動作例を以下に説明する．

- 内部ホストから外部 Web サイトへのパケットを受信する．このとき，パケットの宛先 IP アドレスはプロキシサーバの IP アドレスであり，ターゲットとなる Web サイトの IP アドレスはパケットのデータ部分に格納される．
- プロキシサーバは，パケットのヘッダとデータをセキュリティルールに従って検証する．検証後，ソース IP と宛先 IP アドレスを変更してパケットを再構成し，ターゲットである Web サイトの IP アドレスに送信する．
- Web サイトからの応答パケットは，上記の手続きと逆の処理が行われる．まずプロキシサーバでパケットヘッダとデータの検証が行われ，問題なければソース IP と宛先 IP アドレスが変更されたパケットが再構成され，送信元の内部ホストに送信される．

プロキシサーバでは，各サービスが使用するポート単位にすべてのパケットをスクリーニングし，セキュリティルールに基づいて通過を許可するか遮断するかを決定する．パケットの通過制御をするという役割は，2-7-1 節のパケットフィルタと似ているが，パケットのデータ全体をスキャンするので，より詳細な制御が可能になる．プロキシサーバの利点の一つとして，外部ホストから内部ネットワークや内部ホストの詳細を隠蔽する能力がある．外部ホストは，すべての通信をプロキシサーバと送受信するため，背後にある内部ホストの検出や攻撃は不可能となる．更に，プロキシサーバはパケットのデータをスキャンし（コンテンツフィルタリング），セキュリティルールに従って安全でないコンテンツを除去できるという利点もある．通常，プロキシサーバは，パケットフィルタやほかの機能と組み合わされて使用される．

2-7-3 パーソナルファイアウォール

パーソナルファイアウォールは，個人や小規模オフィス環境のパーソナルコンピュータにインストールされるファイアウォールソフトウェアの総称である．一般的なファイアウォールは，主に企業向けであり，それらと区別してパーソナルファイアウォールという用語が使われている．パケットフィルタ機能やアプリケーション単位で外部ネットワークへの通信を検出・遮断する機能をもつものが多数ある．単体のソフトウェア製品として提供される以外に，アンチウイルスソフトウェアと組み合わされ，統合セキュリティソフトウェアの一部として提供されることもある．

登録商標

■参考文献

1) National Computer Security Center, “Glossary of Computer Security Terms,” Report NCSC-TG-004, Fort Meade, 1988.
2) ISO 7498-2, “Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture,” 1988.
3) Samarati, P. and Vimercati, S.D.C., “Access Control: Policies, models, and Mechanisms,” Foundations of Security Analysis and Design, Lecture Notes in Computer Science 2171, Springer-Verlag, 2001.
4) ISO/IEC 10181-3, “Information technology – Open Systems Interconnection – Security frameworks for open systems: Access control framework,” 1996.
5) Object Management Group, “Security Service Specification,” In CORBAservices: Common Object Services Specification, Chapter 15, Nov. 1997.
6) Organization for Advanced Structured Information Systems, “eXtensible Access control Markup Language (XACML) Version 1.0,” 2003.
7) Department of Defense, “Trusted Computer Security Evaluation Criteria,” DoD 5200.28-STD, 1985.
8) Bell, D.E. and LaPadula, L., “Secure computer systems,” ESD-TR-73-278, Mitre Corporation; vol.I and vol.II, Nov. 1973, and vol. III, Apr. 1974.
9) Denning, D., “A lattice model of secure information flow,” Commun. ACM, vol.19, no.5, pp.236-243, 1976.
10) Lampson, B., “A note on the confinement problem,” Commun. ACM vol.16, no.10, pp.613-615, 1973.
11) National Computer Security Center, “A guide to understanding covert channel analysis of trusted systems,” NCSC-TG-030, Fort Meade, Nov. 1993.
12) Lampson, B.W., “Protection,” ACM Operating Systems Review, vol.8, no.1, pp.18-24, 1974.
13) Harrison, M.A., Ruzzo, W.L. and Ullman, J.D., “Protection in operating systems,” Communications of the ACM, vol.19, no.8, pp.461-471, 1976.
14) Sandhu, R.S., “The schematic protection model: Its definition and analysis for acyclic attenuating schemes, ” J. Assoc. Comput. Mach., vol.35, no.2, pp.404-432, Apr. 1988.
15) Sandhu, R.S., “The typed access matrix model,” Proc. IEEE Symposium on Security and Privacy, pp.122-136, May. 1992.
16) Soshi, M., “Safety analysis of the dynamic-typed access matrix model,” Proc ESORICS 2000, Lecture Notes in Computer Science, Springer-Verlag, vol.1895, pp.106-121, 2000.
17) Woo, T.Y. and Lam, S.S., “Authorizations in distributed systems: A new approach,” J. Computer Security, IOS Press, 1993.
18) Jajodia, S., Smarati, P., Sapino, M.L. and Subrahmanian, V.S., “Flexible support for multiple access control policies”, ACM Trans. Database Systems, vol. 6, pp.214-260, Jun. 2000.
19) Jajodia, S., Samarati, P., Subrahmanian, V.S., “A logical language for expressing authorizations,” Proc IEEE Symposium on Security and Privacy, pp.31-42, Aug. 1997.
20) Ferraiolo, D. and Kuhn, R., “Role-based access controls,” Proc 15th National Computer Security Conference, NIST, pp.554-563, 1992.
21) Clark, D.D. and Wilson, D.R., “A comparison of commercial and military computer security policies”, Proc IEEE Symposium on Security and Privacy, CA, pp.184-194, May. 1987.
22) Brewer, D.F.C. and Nash, M.J., “The chinese wall security policy,” Proc IEEE Symposium on Security and Privacy, CA, pp.215-228, 1989.
23) Housley, R., Ford, W., Polk, W., and Solo, D., “Internet X.509 Public Key Infrastructure Certificate and CRL Profile,” Network Working Group, RFC2459, IETF, Jan. 1999.
24) Dierks, T. and Allen, C., “The TLS Protocol Version 1.0,” Network Working Group, RFC2246, IETF, Jan. 1999.
25) CA/Browser Forum, “Guidelines For The Issuance and Management of Extended Validation Certificates Version 1.1”, vol.10, Apr. 2008.
26) 新エネルギー産業技術総合開発機構電子情報技術開発部, “デジタル情報機器相互運用基盤プロジェクト基本計画,” 2006.
27) Smith, N., Kazmierczak, G., Hardjono, T., Hanna, S., and Sangster, P., “Platform Trust Services Interface

Specification (IF-PTS) Specification Version 1.0 Revision 1.0”, TCG Infrastructure Working Group, vol.17, Nov. 2006.

28) Nemeth, E., Snyder, G., Seebass, S. and Hein, T.R., “UNIX system administration handbook (3rd edition),” Prentice Hall PTR, 2000.

29) Open Software Foundation, “Introduction to OSF DCE, DCE Release 1.1,” Pearson Education POD, 1995.

30) Gong, L. Ellison, G. and Dageforde, M., “Inside Java 2 platform security: architecture, API design, and implementation,” Addison-Wesley, 2003.