# 制御構造と論理言語を基にしたアクセス制御のポリシー記述言語の構築

田原　聖悟†　　長谷部　浩二‡　　加藤　和彦‡
†筑波大学第三学群情報学類　　‡筑波大学大学院システム情報工学研究科

## 1　はじめに

アクセス制御は，システムのリソースに対してアクセスを行う能動的な主体が，そのリソースに対する可能な操作を管理する技術であり，コンピュータセキュリティを保つための重要な技術の一つである．アクセス制御のポリシーを記述する言語には，可読性や目的とするポリシーを記述するために必要な表現力が求められる．また，その言語によって記述されたポリシーは，解釈の多様性や暗黙の了解のような曖昧さを排したものでなければならない．これらの条件を満たすために，近年数理論理学の言語を基にしたアクセス制御のポリシー記述言語の提案が数多くなされている．こうした研究における基本的なアイデアは，まずポリシーを論理式により記述し，また，そのポリシーを論理推論の公理とみなすことで，アクセスの許可・禁止の判定を論理推論により実現するというものである．

しかし，論理式によって記述されたポリシーでは，ポリシーを場合分けにより記述する際に可読性が低く直観的に理解しにくくなる．そのため，場合分けによる記述の可読性を高くする工夫として，否定を用いた例外的表現（例えば，「全てのユーザはファイル1にアクセスできるが，ロール1のユーザはこのファイルにアクセスできない」など）を行う方法についての研究がBenferhatら[3]によってなされている．一般に，否定を用いた例外的表現を許すことにより，次の二つの問題が生じる．第一の問題は，論理体系から矛盾が導出されてしまい，その結果このような表現を公理とするアクセス可否の判定が正しく行われないというものである．そのため，この論文においてBenferhatら[3]は，ポリシーを記述する論理式に優先順位をつけることで矛盾を解消する方法を提案している．しかしながら，ポリシーの条件が複雑な場合に優先順位をつけながら記述することは難しく，可読性も低くなる．また，第二の問題として，ポリシー内での条件文の前件で否定表現を許すと，正しくアクセスの可否を判定するためには，全ての個体に対して言語に導入した全ての述語に関する真偽を明示的に記述しなければならないことも挙げられる．

これらの問題に対し，本研究では，論理言語に制御構造の文法を加えたポリシー記述言語を提案する．この言語では，Benferhatら[3]のアイデアに基づき，否定表現を伴う論理言語によりポリシーを記述する．さらに，優先順位の概念をより明示的に表現し，可読性を高めるために，これを制御構造によって記述する．以上により，複雑な条件を伴うポリシーを既存の言語に比べより簡潔に表現することを目指す．

## 2　関連研究の概要とその課題

Abadiらによる研究[1]以来，論理言語を基にしたポリシー記述言語やそれらを用いたアクセスの可否判定のアルゴリズムが数多く提案されている．例えば，Halpernら[2]は一階述語論理を基に，ACL(Access Control List)をRBAC(Role-based access control)におけるロールでまとめて記述する方法を提案している．しかし，複数のロールが重なり合う時，それらを場合分けしてポリシーを記述しようとすると記述量が増えるとともに，可読性が低くなるという問題が生じる．

こうした場合分けに伴うポリシーの記述量に関する問題に対処するため，Benferhatら[3]は，ポリシーに優先順位をつけることにより例外的表現を用いる提案をしている．複数の場合に共通して同じポリシーを適用するとき，それらをまとめて記述し，その記述にそぐわない部分を例外的に記述するというものである．しかし，アクセスの可否を問うクエリに対して，論理推論による証明可能性の判定を行う際には，その証明において公理となるポリシーの中に矛盾が含まれるため，優先順位無しのポリシーに直してから証明を行う．しかしながら，優先順位をどのようにつけ，またどのようにそれを処理すべきかということは，例外やロールの階層については考えられているが，アクセスの可否以外のポリシーの記述については十分な研究がなされていない．

## 3　提案手法

本研究では，こうした複雑な場合分けを伴うポリシーを記述するために，以下のような言語を提案する．ま

A Policy Specification Language for Access Control Based on Control Flow and Logical Language
†Shogo TAHARA　‡Koji HASEBE　‡Kazuhiko KATO
†College of Information Sciences, the Third Cluster of Colleges, University of Tsukuba
‡Graduate School of Systems and Information Engineering, University of Tsukuba

ず，アクセスの可否に直接関わる記述に関しては一階述語論理の言語を用いる．特に，否定を伴う表現を許すことで，Benferhat ら [3] と同様例外的な記述も許す．さらに，優先順位の概念を言語レベルで明確に表現するため，本研究ではこうした優先順位の概念を制御構造により表現する．より具体的には以下の通りである．

論理言語によるポリシー記述言語に，if-then-else 文，switch-case 文という二つの制御構造を導入することで，ポリシーにプログラムのようなフローを持たせる．

### 制御構造を導入する利点

制御構造を導入することによる利点は以下の二点である．まず第一に，可読性の向上がある．制御構造の構文を導入することで，ポリシーの記述に例外的表現を用いる場合であっても，優先順位が自然とつくことで可読性が高まる．例えば，「医者はファイル f1 に read 可能である．ただし，21 時以降は除く．」という文を我々の言語で書くと以下のようになる．

```
if(doctor(x)){may_access(x,f1,read);
 if(time>=21){¬may_access(x,f1,read);}
}
```

このように，例外的表現に制御構造を用いて記述すると，例外となる事実は必ず例外でない事実よりもフロー上で後ろにある．そのため，フローを追うごとに優先度を高くするだけで，例外的表現に伴うポリシーの記述の矛盾を解決することができる．

また第二の利点は，先に指摘した全ての否定文の明示的記述に関して，このような否定文を省略できるという点である．一般に，ポリシーの条件の前件に否定が含まれる場合は，論理言語のみによる記述においては，すべてのロールなどの述語について肯定か否定かをポリシー内で明示的に記述する必要があった．しかし，こうした否定文を記述する代わりに，制御構造を用いることにより直観的な解釈に合致した方法でポリシーを記述することができる．例えば，「医者はファイル 1 に read 可能である．医者以外の人はファイル 2 に read 可能である．」という文を我々の言語で書くと以下のようになる．

```
if(doctor(x)){may_access(x,f1,read);}
else{may_access(x,f2,read);}
```

この制御構造では，条件文において，doctor(x) が成り立たない場合はすべて else 文に移るものと解釈する．このように，論理言語の含意とは異なり，あくまで制御フローとして扱うため，¬doctor(x) であるということを明示的に表現する必要はない．

### 提案する言語の表現力

制御構造を導入する第二の利点より，新たな述語を導入した場合，その述語の肯定と否定の全てをポリシー内に記述する必要はなく，肯定のみの記述で十分である．そこで本研究で提案する言語では，アクセスの許可・禁止に直接関わる文 (may_access が現れる文) 以外に関しても記述可能であるとすることで，ポリシーの記述を大幅に増やすことなく，表現可能な範囲を広げる．その具体例は以下の通りである．

**ロールの階層:**ユーザに与えられるロール同士を階層的に結びつけることができる．これにより，ポリシーの記述量を減らすことができる．例えば，「医者はロールの階層で看護師の上にいる.」という文は以下のように記述する．

```
doctor(x)⇒nurse(x)
```

**ワークフロー:**複数のユーザが共同作業する際に必要な手続き的処理手順を指す．これにより，手続きの前後でポリシーを変えたい場合に簡潔に記述できる．さらに，許可証のようなアクセス権をわかりやすく記述するための方法も記述できる．例えば，「Alice が作業 a を終えていないと，Bob は作業 b をすることができない.」という文は以下のように記述する．

```
if(¬finish(Alice,a)){¬do(Bob,b);}
```

## 4 まとめ

本研究では，制御構造と論理言語を基にしたアクセス制御のポリシー記述言語の提案をした．このポリシー記述言語により，ポリシーやコンテクストがアクセスの許可・禁止以外の条件を伴い複雑となる場合の記述も可能とした．

## 参考文献

[1] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin. A calculus for access control in distributed systems. ACM Transactions on Programming Languages and Systems, 15(4):706-734, 1993.

[2] J. Y. Halpern, and V. Weissman. Using first-order logic to reason about policies. SACMAT'03, 187-201, 2003.

[3] S. Benferhat, R. E. Baida, and F. Cuppens. A stratification-based approach for handling conflicts in access control. SACMAT'03, 189-195, 2003.