ケーパビリティに基づくアクセス制御のための ケーパビリティ管理機構

松井慧悟† 新城靖†,‡ 杉本卓哉† 佐藤聡† 中井央† 板野肯三†,‡

筑波大学† 科学技術振興機構‡

1 はじめに

現在われわれは協調作業における ACL(Access Control List) に基づくアクセス制御の問題点を解決し、一時的なアクセス権の委譲を容易にするために、ケーパビリティに基づくアクセス制御の研究を行っている [2, 4]。ケーパビリティとはオブジェクトの識別子とオブジェクトに対して行える操作の集合を組にしたものである。ケーパビリティの性質として他人に渡すことが可能な点と、弱いケーパビリティを作成可能な点があげられる。**弱いケーパビリティ**とは、あるケーパビリティからそれより制限された操作の集合を持つケーパビリティである。

ケーパビリティに基づくアクセス制御の利用が広がるにつれて、ユーザは、多数のケーパビリティを保持するようになり、結果としてケーパビリティの管理が困難になる。この問題点を解決するために、われわれはケーパビリティの管理を容易にする機構の研究を行っている。この論文では、ケーパビリティの形式と、そのケーパビリティを保存するプログラムである**ケーパビリティバスケット**について述べる。

2 ケーパビリティの形式

本研究では、以下の要件をみたすケパビリティ管理機構を実現する。

- 使用目的の異なるケーパビリティを統一的に扱う。
- 弱いケーパビリティを作成可能にする。
- 改ざんを防ぐ。

2.1 ケーパビリティの構成要素

上で述べた要件を満たすためのケーパビリティの構成要素を表 1 に示す。ケーパビリティは本体、変遷、検査値から構成される。本体は、すべてのケーパビリティ共通の要素である基本部分と、型に応じた拡張部分にわかれる。型は、3 章で述べるケーパビリティバスケットにおいて、型ごとの検索やブラウズのために使われる。変遷とは、あるケーパビリティから弱いケーパビリティを作成した際に差分をとったもの

```
text
@CID:33981
OID:spamFilter-who@@example.com-123765
Rigths:bypass
Not After:2007-12-28T20:00:00
Type:email
Extensions:addr=who@@example.com
@

text
@d4 1
a4 1
Not After:2007-12-29T00:00:00
@
```

図 1 ケーパビリティ本体と変遷の例

である。本体と変遷を表現する方法としては、RCS(Revision Control System) で用いられている形式を用いる (図 1)。図 1 の前半部分は、ケーパビリティの本体であり、後半部分は変遷である。この例では、有効期限 (Not After) が短くなっている。検査値はケーパビリティの改ざん防止に用いる乱数、または、一方向関数の結果である。検査値を用いた改ざん防止の方法については 2.2 節で述べる。

2.2 ケーパビリティの改ざん防止

ケーパビリティの改ざんを防ぐための方法として、ケーパビリティを作成する際に作成者のみが知る乱数を生成し、それを検査値とする。弱いケーパビリティを考慮しなければ、提示されたケーパビリティの本体と検査値を、保存されたものと比較することで改ざんされていないことがわかる。以下に、弱いケーパビリティの作成手順、および、検査手順を示す。

弱いケーパビリティの作成手順

1. 本体の値を新しい値に書き換える。
2. 元の値と新しい値の差分をもとめ、変遷に追加する。
3. 新しい本体の値、変遷、および、古い検査値を入力として一方向関数で新しい検査値を作成する。

表 1 ケーパビリティの構成要素

本体	基本	CID	ケーパビリティの識別子
		OID	操作対象のオブジェクトの識別子
		権利	許可される手続きの集合
		属性	有効期限、使用回数、登録時刻など
		型	使用目的の識別子
	拡張		型ごとの特有の情報
変遷			弱いケーパビリティを作成時の変更履歴
検査値			改ざんを防ぐための乱数、または、一方向関数の結果

Capability Management Mechanism for Capability-Based Access Control

Keigo Matsui†, Yasushi SHINJO†,‡, Takuya Sugimoto†, Akira SATO†, Hisashi NAKAI†, Kouzo ITANO†,‡

† University of Tsukuba

‡ Japan Science and Technology Agency

弱いケーパビリティの検査手順

1. 提示されたケーパビリティの本体と変遷から最初に作成されたケーパビリティの本体を復元する。復元された本体と保存してある本体が同じであることを確認する。
2. 変遷を用いて、弱いケーパビリティを作成した手順を再現する。各段階で、権利が前段階の部分集合になっているか、有効期間が長くなっていないか、使用回数が増やされていないかを確認する。
3. 復元された弱いケーパビリティの検査値と提示されたケーパビリティの検査値が同じであることを確認する。
4. 有効期限や利用回数を確認する。

3 ケーパビリティバスケット

ケーパビリティバスケットは個人用のコンピュータで動作し、その個人が所有するケーパビリティを安全に管理するプログラムである。ケーパビリティバスケットは、2 つのインタフェースを持っている。

API メールリーダなどのアプリケーションが利用する。この API は SOAP によりアクセス可能とする。

ユーザインタフェース ユーザに対して保存されているケーパビリティをグラフィカルに表示する。

3.1 API

表 2 に、ケーパビリティバスケットが提供する API を示す。この API の利用方法を電子メールのスパムフィルタをバイパスできるケーパビリティを例として述べる。

はじめに、ユーザがメールリーダにおいてケーパビリティを作成する操作を行う。するとメールリーダは create 手続きを呼び出してケーパビリティを作成する。

次に、ケーパビリティを配布したい時、ユーザがメールリーダにおいてケーパビリティを取り出す操作を行う。すると、メールリーダは get 手続きを呼び出してバスケットからケーパビリティを取り出す。ユーザは、取り出されたケーパビリティから他人に配布するものを選択する。メールリーダは選択されたケーパビリティをメールに付加し、送信する。

そのメールを受信したユーザは、メールリーダにおいてケーパビリティを保存する操作を行う。するとメールリーダは、put 手続きを呼び出してバスケットへケーパビリティを保存する。

所持するケーパビリティを用いて電子メールのスパムフィルタをバイパスしたい場合は、ユーザがメールリーダにおいてケーパビリティを取り出す操作を行う。すると、メールリーダは get 手続きを呼び出してバスケットからケーパビリティを取り出す。ユーザは、取り出されたケーパビリティから希望する手続きが行えるケーパビリティを選択する。すると、メールリーダは選択されたケーパビリティをメールに付加し、送信する。

表 2 ケーパビリティバスケットの API

create	ケーパビリティを作成する
get	バスケットからケーパビリティを取り出す
put	バスケットへケーパビリティを登録する
check	ケーパビリティの検査を行う
used	ケーパビリティの使用回数を増やす

そのメールを受信すると、メールリーダは check 手続きを呼び出してケーパビリティの検査を行い、ケーパビリティの正当性が確認できたら used 手続きを呼び出してケーパビリティの使用回数を増やす。そして、メールリーダはスパムフィルタをバイパスさせてメールを受信箱に移す。

3.2 ユーザインタフェース

ケーパビリティバスケットは以下の機能のユーザインタフェースを提供する。

ブラウズ 保存されているケーパビリティを閲覧する。

メタ情報の付加 本体以外に情報についてのユーザのメモを付加する。

検索 保存されているケーパビリティを本体の情報やメタ情報を使って検索する。

編集 保存されているケーパビリティの情報を編集する。

4 関連研究

Amoeba[3] はオブジェクトの保護のためにケーパビリティを用いている分散型オペレーティングシステムである。Amoeba はオペレーティングシステムという閉じた環境でケーパビリティを扱う。本研究では、インターネットという開いた環境でケーパビリティを扱う。

Keychain Access[1] は Mac OS に付属のパスワード管理のためのユーティリティである。Keychain Access はパスワードや証明書など管理を行う。本研究ではケーパビリティを対象としている。

5 おわりに

本論文では、ケーパビリティの形式とそのケーパビリティを保存するしくみであるケーパビリティバスケットについて述べた。ケーパビリティバスケットは、アプリケーションのための API とユーザのためのユーザインタフェースを提供する。今後は、本論文で提案したケーパビリティバスケットを実装し、評価を行う。

参考文献

[1] Apple Developer Connection. Keychain Manager Reference, 7 2005.

[2] M.Mabuchi, Y.Shinjo, A.Sato, and K.Kato. An Access Control Model for Web-Services that Supports Delegation and Creation of Authority. *The Seventh International Conference on Networking*, 2008. (Accepted for publication).

[3] Sape J. Mullender, Guido van Rossum, Andrew S. Tanenbaum, Robbert van Renesse, and Hans van Staveren. Amoeba: A distributed operating system for the 1990s. *Computer*, Vol. 23, No. 5, pp. 44–53, 1990.

[4] 松井慧悟, 新城靖, 佐藤聡, 板野肯三, 馬渕充啓, 加藤和彦. Web ページに対するケーパビリティを用いたアクセス制御のプロキシによる実現. 情報処理学会, システムソフトウェアとオペレーティングシステム研究会, Vol. 2007-OS-105, pp. 95–102, 2007.