Capability を主軸とするマイクロカーネル
- Caprese：Capability-based Security を採用したマイクロカーネル -

1　背景

Linux に代表される従来の OS は、特定の操作を行うために特権を必要としたり、ハードウェアを操作するようなデバイスドライバがカーネル空間で動作するようになっていたりと、本来不要な権限を保持した状態で実行される環境になっている。

不要な権限を保持しているとそのプログラムにバグがあった場合の影響範囲が広がってしまうため慎重にプログラムを組む必要があるが、開発者は必ずしもカーネルやセキュリティに精通しているわけではなく、そのため脆弱なプログラムを提供してしまうことがしばしば発生する。

2　目的

このような状況を改善するためには、権限を細かく分割し、安全かつ容易に分散できる仕組みが必要である。本プロジェクトではこの仕組みを実現するカーネルを新たに開発する。このカーネルは権限をオブジェクトとして表現し、プロセス間で送受信することで権限の分散を実現する。

3　開発の内容

本プロジェクトでは、Capability-based Security を用いたマイクロカーネル “Caprese” と、そのうえで動作する OS のサンプル、これらの上で動作する OS 非依存で汎用的な C/C++ 標準ライブラリを実装した。対象とする CPU 命令セットは 64bit の RISC-V であり、仮想環境である QEMU と実機である VisionFive2 に向けて動作する。

3.1　Caprese について

まずは Capability-based Security の一般的な概念について説明する。Capability-based Security では、図 1 のように、あるリソースへの参照とアクセス権を一体とした Capability という概念を用いてセキュリティを実現する。Capability は伝達可能かつ偽造不可能な参照であり、このセキュリティを実現したシステムではそのリソースへアクセスする唯一の手段となるという特徴を持つ。

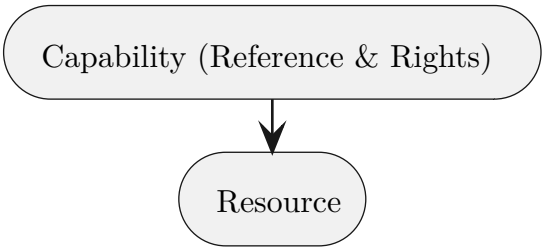

図 1: Capability の概要図

次に Caprese における Capability について説明する。図 2 のように、Caprese が Capability で保護するリソースはカーネル内に存在し、ユーザアプリケーション側から直接操作することはできない。カーネルへのシステムコールがそのリソースを操作する唯一の手段となる。

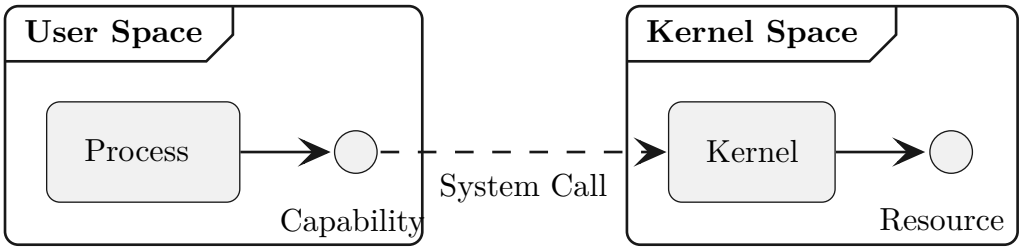

図 2: Caprese における Capability

また、Capability はプロセス間で委譲あるいは転送することができる。どちらの操作を行っても、Capability を持っていたプロセスはその権限を失い、委譲あるいは転送先のプロセスがその権限を得る。委譲した場合は委譲元のプロセスはその権限を取り戻して委譲先の権限を無効化することができるが、転送した場合は取り戻すことはできない。Caprese では、この Capability を物理メモリなどのハードウェアリソースやプロセスなどのカーネルオブジェクトの保護に適用した。これにより、OS 機能やデバイスドライバなどの従来は特権が必要であったプログラムをユーザアプリケーションと同じレイヤで動作させつつ、より容易に最小権限に近づけることができる。

Caprese が提供する Capability には次のようなものがある。

- Memory Capability
  物理メモリの範囲を表す。この Capability は領域の一部をほかの Capability に変換することが可能であり、変換した Capability の種類に応じてその領域に対する操作が可能になる。
- Task Capability
  特定のプロセスに対する操作を提供する。この Capability を利用することで

ユーザアプリケーション側でプロセスを管理する機構を実装したり、別のプロセスへ Capability を委譲/転送したりすることができる。

- Endpoint Capability
  プロセス間通信におけるチャネルに対する操作を提供する。
- Page Table Capability
  プロセスの仮想アドレス空間を構成するページテーブルに対する捜査を提供する。この Capability と後述する Virt Page Capability を利用することで、プロセスの仮想アドレス空間をユーザアプリケーション側から構成することができる。
- Virt Page Capability
  特定のページを表す。この Capability と前述の Page Table Capability を利用することで、プロセスの仮想アドレス空間をユーザアプリケーション側から構成することができる。
- Cap Space Capability
  Capability を管理するための領域を表す。この Capability はカーネルにメモリ領域を割り当てるために使用される。これにより、プロセスが持てる Capability の上限数を増やすことができる。
- ID Capability
  システム全体で一意となる値を表す。この Capability はユーザアプリケーションが提供するリソースに対して一意な ID を割り当てるために使用される。

3.2 OS のサンプルについて

この OS は、Caprese による Capability の機構が実際に OS やドライバ開発に適合することを示すために開発した。OS 機能としては、次のようなものを実装した。

- メモリ管理プロセス
- アプリケーション管理プロセス
- ドライバ管理プロセス
- ファイルシステム
- シェル

3.3 OS 非依存の C/C++ 標準ライブラリについて

Caprese はファイルシステムなどの OS として必要とされる機能を持たず、上に乗る OS のインタフェースを POSIX などの既存のものに制限しない。そのため、OS 機能

に依存する部分をスタブとして提供するような C/C++ 標準ライブラリがあれば、どのようなインタフェースをもつ OS を開発してもその OS に向けた実装が容易になると考え、本ライブラリの開発に至った。これにより OS 開発者は OS 機能の実装のみに集中できるため、OS 開発の生産性向上に役立つと考えられる。

4　従来の技術との相違

従来の OS では、OS 機能やデバイスドライバが特権を持ち、本来不要な権限を保持した状態で実行されていた。Caprese は、従来では操作するために特権が必要であったリソースを Capability として切り出すことで権限を細分化し、各機能に必要最小限の権限を安全かつ容易に付与することを可能にした。

5　期待される効果

Caprese は、Capability-based Security をハードウェアリソースやカーネルオブジェクトに適用することで、上に乗る OS をより簡単に最小権限に近付けることができる。これにより、独自の OS やデバイスドライバを開発する必要がある分野において、セキュリティを重視したシステムを容易に開発することができるようになることが期待される。

6　普及の見通し

本プロジェクトは既にオープンソースとなっているが、ドキュメントを整備することで開発者を増やし、コミュニティを形成し、普及を目指していく。

7　クリエータ名（所属）

- 今村 翔太（慶應義塾大学 環境情報学部 環境情報学科）

（参考）関連 URL

- ソースコード：`https://github.com/caprese-project`