# 5D-4 CGIによるリソース濫用の防止機構

揚妻 匡邦[†]　河野 健二[††,‡]　益田 隆司[††]

[†]電気通信大学大学院電気通信学研究科情報工学専攻
[††]電気通信大学情報工学科　[‡]科学技術振興事業団 さきがけ研究21

## 1　はじめに

現在，インターネット等の広域分散環境は，情報共有の基盤としてますます重要になりつつある．特に，インターネット上のCGIを利用したサービスは，情報検索や情報の交換などの対話的なサービスを柔軟に提供でき，その利用価値は極めて高い．しかしながら，一般ユーザの作成したCGIは信頼性が低く，ウェブサーバの資源を濫用しサーバの性能を低下させる危険性がある．Resource containers[3]という研究では，粒度の細かい資源管理機構を実装し，ウェブサーバの性能低下を抑えている．本稿では，我々が研究・開発を進めている高度なセキュリティ機構を持つOS [1][2]の一環として，CGIが利用できる資源を制限する機構を提案する．またその機構をLinuxのネットワーク資源の制限に適用した結果を報告する．

## 2　従来のOSの問題点

従来のOSにおける資源管理では，プロセスが資源割り当ての対象である．そのため，プロセスを次々と生成すれば，大量の資源を容易に占有できてしまう．例えば，通信用バッファを占有するには，次々とプロセスを生成しながら通信用のソケットを確保すればよい．プロセスごとのファイル記述子の最大数によって，一つのプロセスが占有できるバッファの総量が制限されているにもかかわらず，容易に通信用バッファが占有できてしまう．このように従来の資源管理では，CGIなどの信頼性の低いプログラムでも，際限なく資源が使用できてしまう．

## 3　リソースドメインの提案

本稿では，CGIによるリソースの濫用を防ぐため，リソースドメインと呼ぶ機構を提案する．リソースドメインとは，リソース割り当ての対象であり，リソースドメインごとに資源の使用量を設定することができる．各プロセスは必ずただ1つのリソースドメインに属しており，1つのリソースドメインには複数のプロセスが同時に属する事ができる．リソースドメインを用いると，複数のプロセスが利用する資源の総量を一つのリソースドメインとして課金でき，複数のプロセスによる資源使用量を制限できる．なお，従来のOSにおける資源管理は各プロセスがそれぞれ別のリソースドメインに属しているのと同等である．

リソースドメインを操作するためのシステムコールとして以下のシステムコールを用意した．

- `div_rd()` `div_rd()` を実行するプロセスが属するリソースドメインを分割し，新たなリソースドメインを生成する．分割元のリソースドメインを親リソースドメイン，新たに生成するリソースドメインを子リソースドメインと呼ぶ．
- `fork_rd()` 子プロセスを生成し，その子プロセスを引数で指定したリソースドメインに配属させる．ただし指定できるリソースドメインは親プロセスが属するリソースドメイン，または親プロセスが作成したリソースドメインに限られる．なお，通常の`fork()` システムコールでは，生成した子プロセスは親プロセスと同じリソースドメインに属するようにした．
- `abs_rd()` 不要になったリソースドメインを親リソースドメインに吸収する．
- `get_rd()` リソースの使用情報を得る．

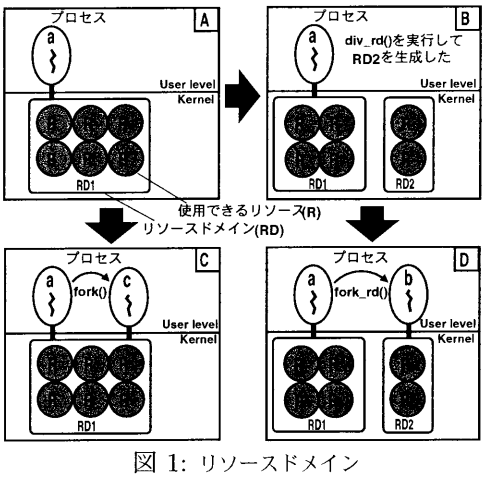


図1: リソースドメイン

図1にリソースドメインの使用例を示す．図1における A → B では，`div_rd()` を使い，親リソースドメイン RD1 を分割してリソースドメイン RD2 を生成している．この時，新しいリソースドメインを生成し

---

**Defending againt resource abuse by CGI**

Masakuni AGETSUMA[†], Kenji KONO[††,‡], Takashi MASUDA[††]

[†]Course in Computer Science and Information Mathematics, Graduate School of Electro-Communications, The University of Electro-Communications, [††]Department of Computer Science, The University of Electro-Communications, [‡]PRESTO, Japan Science and Technology Corporation

ても，使用できる総資源量は変わらない．B → D では fork_rd() を使い，子プロセス b を RD2 に配属させている．プロセス b はすでに作成済みのリソースドメインに配属されただけであり，使用できる総資源量は変わらない．A → C では，fork() を使い，親プロセス a と同じ RD1 に属する子プロセス c を作っている．この場合も，親子で利用できる資源の総量に変わりがない．このようにリソースドメインを導入する事で，プロセス全体で使用できる資源の総量に制限をかけることが容易になる．

## 4 実装

Linux 2.0.38 をベースに，ネットワークバッファと通信ポートという二つの資源に対して，リソースドメインの実装を行った．Linux カーネルを改変し，リソースドメインを表す構造体を追加し，各資源の使用量と使用制限を管理できるようにした．また，プロセス構造体に，所属するリソースドメインを記録するメンバを追加した．さらに 3 節で説明した 4 つのシステムコール div_rd()，fork_rd()，abs_rd()，get_rd() を追加した．また，socket(),bind(), setsockopt(),close(), fork() の 5 つのシステムコールを改変し，その実行前に資源使用量を累計するコードと，資源の使用制限の超過の有無を検査するコードを追加した．

## 5 実験

以下の実験では，PentiumII 400MHz プロセッサ，メモリ 128Mbytes を搭載した PC/AT 互換機を使用し，ネットワークは 100BASE-T スイッチングハブを使用した．

### 5.1 リソースドメインによる資源の制限

リソースドメインによって，複数のプロセス全体が利用できる資源の制限ができる事を確認する実験を行った．クライアントからの接続要求を受け付けるだけの，fork() を使う並行サーバと，サーバがブロックするまで大量のデータを送るクライアントを用意する．クライアントの数を増やしながら，メモリの消費量を測定する．実験結果を図 2 に示す．1 はリソースドメインを実装した Linux 上でサーバを動かした場合であり，2 はオリジナルの Linux 上でサーバを動かした場合である．1 では，クライアント数が 50 前後でメモリの消費量が制限されている事が確認できる．

### 5.2 リソースドメインによるオーバヘッドの測定

リソースドメインを導入したことによる各システムコールのオーバヘッドの計測を行った．表 1 に示した結果は，リソースドメインを実装する上で手を加えたシステムコールをそれぞれ 1000 回実行し，その平均実行時間である．表 1 中の"ORG"はオリジナルの Linux 上での結果であり，"RD"はリソースドメインを実装した Linux 上で実行した結果である．リソースドメインによるオーバヘッドは高々 6.7% 以下である事がわかる．表 2 には，リソースドメインを操作するシステムコールを 1000 回実行した平均実行時間を示す．

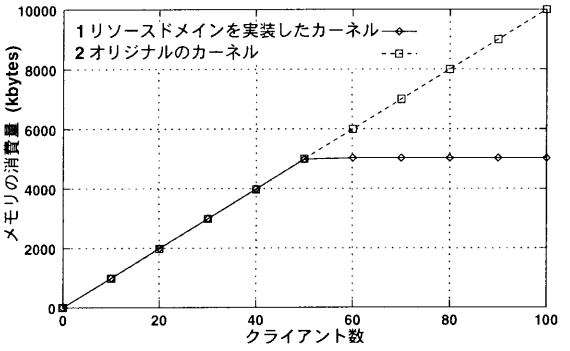

図 2: クライアントの増加に対するメモリの消費量

表 1: システムコールの実行時間の比較

| システムコール | cost($\mu$sec) | | overhead |
|---|---|---|---|
| | RD | ORG | (%) |
| socket() | 18 | 18 | 0 |
| bind() | 12 | 12 | 0 |
| setsockopt() | 11 | 11 | 0 |
| close() | 16 | 15 | 6.7 |
| fork() | 145 | 144 | 0.7 |

表 2: リソースドメインを操作するコスト

| システムコール | cost($\mu$sec) |
|---|---|
| div_rd() | 10 |
| get_rd() | 10 |
| abs_rd() | 10 |
| fork_rd() | 146 |

## 6 まとめ

従来の OS における資源管理では，資源の制限を柔軟に行うことが難しく，CGI などの信頼性の低いプロセスが容易に資源の占有ができてしまう．本稿では，柔軟な資源管理を可能にするため，リソースドメインという機構を提案した．また，リソースドメインの機能をネットワークバッファと通信ポートに適用し，その効果を確認した．

今後，物理メモリ，CPU 時間，I/O 割り当てなどの他の資源に対してもリソースドメインによる資源の制限を適用していく方針である．

## 参考文献

[1] 金子 済, 河野 健二, 益田 隆司 : 悪意ある外部プログラムによるリソース濫用の防止— ファイルキャッシュのケーススタディ —, 情報処理学会研究会報告, 2000-OS-84, pp.205-212, 2000.

[2] 品川高廣, 河野健二, 高橋雅彦, 益田隆司 : 拡張コンポーネントのためのカーネルによる細粒度軽量保護ドメインの実現, 情報処理学会論文誌, 第 40 巻, 第 6 号, pp.2596-2606, 1999.

[3] Gaurav Banga, Peter Druschel, Jeffrey C. Mogul : Resource containers, Third Symposium on Operating Systems Design and Implementation, pp.45-58, 1999.