6T-8

# 拡張可能なネットワーク処理を行うオペレーティングシステムアーキテクチャ

藤田 祥† 奥村 貴史†† 江崎 浩†

† 東京大学 情報理工学系研究科

†† ピッツバーグ大学計算機科学科

## 1 はじめに

近年インターネットがあるゆる情報の通信インフラとして注目されている。これまで専用通信インフラを利用してきた様々なアプリケーションがTCP/IPを用いたものに置き換わってきた。VoIPや動画の配信や、並列分散処理などである。WebやFTPなどバルク通信を行うアプリケーションと異なり、これらのアプリケーションは実時間性や低レイテンシなどネットワークに新しい特性を要求する。今日のオペレーティングシステムが提供するネットワーク処理はこれらの新しいアプリケーションの要求に応えるに至っていない。

本研究ではカーネル内にネットワーク処理を行う仮想マシン、仮想ネットワークプロセッサを設置し、ユーザプロセスがプログラムを読み込ませることによって高速、柔軟にパケット処理を行うアーキテクチャを提案する。想定している仮想ネットワークプロセッサのプログラムとしては、

- アプリケーション的に意味のあるサイズのデータが揃うまで、データのアプリケーション送信（コピー）を遅らせる
- 暗号データの復号を行い、認証に失敗した場合はアプリケーションに渡す前にデータを捨てる
- カーネル内でアプリケーションレイヤードマルチキャストをサポートする。すなわちデータの中を解析し、アプリケーションを経由しないで次のホストにデータをフォワードする

などである。

以下ではまず現在のオペーレティングシステムで提供されているネットワーク処理の概要とその問題点について述べ、提案するカーネル内仮想ネットワークプロセッサについて述べる。最後にまとめと展望について述べる。

Operating System Architecture for Extensible Network Processing

† Sho Fujita (fujisho@wide.ad.jp)

† Hiroshi Esaki (hiroshi@wide.ad.jp)

Graduate School of Information Science and Technology, Tokyo University (†)

†† Takashi Okumura (taka@wide.ad.jp)

Department of Computer Science, University of Pittsburgh (††)

## 2 ネットワーク処理の問題点

まずUNIX系オペレーティングシステムのネットワーク処理の概要について述べる。次の例はUDPを用いた受信処理の概要である。

- パケットがネットワークインターフェースに到着し、ハードウェア割り込みが起こる。適切なドライバ呼び出され処理を行い、パケットをIP層のキューに入れる。その後ソフトウェア割り込みを起こして、リターンする
- ソフトウェア割り込みのコンテキストでIPキューからパケットを取り出し、IPヘッダを処理する。IPの処理が終わったらUDPの処理を行う。ポート番号からどのソケットに対するパケットか探索を行う。その後ソケットバッファにパケットを入れてリターンする
- その後プロセスはrecvシステムコールを呼び出し、ソケットバッファから自分のバッファにデータをコピーする

プロトコル処理がきれいにレイヤ毎に分かれているのが分かる。すなわちイーサーネットの処理が完全に終わるまでIPの処理は行われず、IPの処理が完全に終わるまでTCPの処理は行われない。

TCP層より上の処理に関しては

- 宛先プロセスがスケジュールされる
- 宛先プロセスがrecv()を呼び出しデータをプロセスのメモリ空間にコピーする
- コピーしたデータを処理する

いう手順を追って処理が行われる。カーネル・ユーザプロセス間を跨ぐ処理のコストは特に大きく、ネットワークが高速化し処理しなけらばいけないパケットが多くなってくると無視出来ないオーバーヘッドとなってくる。(図1)

## 3 カーネル内仮想ネットワークプロセッサ

前節で述べたオーバヘッドを軽減するために、カーネル内に拡張可能な仮想ネットワークプロセッサを設

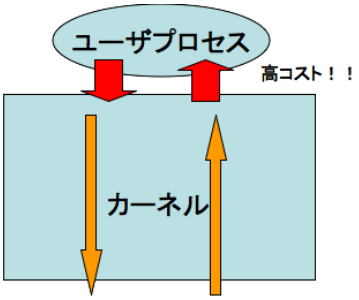


図 1: ユーザ・カーネル間通信

置することを提案する。カーネル内に仮想マシンは特別新しいものではなく、代表的なパケットフィルタの [3] はパケットを受理する条件を表したプログラム、例えば以下のようなものをカーネル内の BPF インターフェースに読み込む。

```
$tcpdump -d ip host 192.168.0.1
(000) ldh      [12]
(001) jeq      #0x800           jt 2    jf 7
(002) ld       [26]
(003) jeq      #0xc0a80001      jt 6    jf 4
(004) ld       [30]
(005) jeq      #0xc0a80001      jt 6    jf 7
(006) ret      #96
(007) ret      #0
```

これにより、一旦全てのパケットをプロセスに渡しフィルタリングを行っていた以前のパケットフィルタから大幅に処理性能を向上させることに成功した。

本研究の仮想ネットワークプロセッサはこれをフィルタリング以外のネットワーク処理に関して拡張したものである。

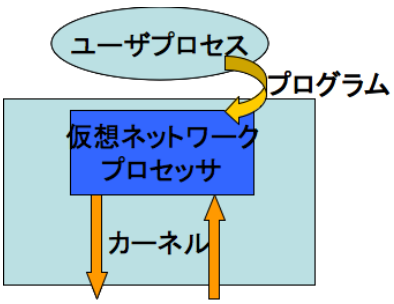


図 2: 仮想ネットワークプロセッサの利用

図 2 のように、ユーザプロセスは受信した情報を解析しなんらかの処理、例えばデータの改変、返信処理などを行うプログラムをカーネル内の仮想ネットワークプロセッサに読み込む。どのプロセスが実行している場合もカーネルイメージは同じメモリ領域にマップされているので前説で指摘したスケジュールのコスト、つまり宛先プロセスが次にスケジュールされるまで待つコスト、メモリコピーのコストをなしに、受け取ったデータの処理が出来る。

## 4 関連研究

[1] ではメモリコピーを行わずにカーネル、ユーザプロセス間でデータ通信を行う。本研究と異なり、データ処理時に宛先プロセスがスケジュールされる必要は依然としてある。[4] ではカーネルを介さず、ユーザプロセスから直接ネットワークインターフェースへのアクセスさせる。コピーやスケジュールのコストを削減することが出来るが、別なハードウェアを必要とする。またカーネルからはどのような処理が行われているか分からないので協調処理などが難しい。[2] は拡張可能なマイクロカーネルで、マイクロカーネルにモジュールを埋め込み自由にカーネルインターフェースを拡張出来る。仮想マシンを用いたアプローチと比較し互換性、セキュリティの確保に問題が残っている。

## 5 まとめと展望

本稿ではカーネルのメモリ空間にユーザプロセスが安全にプログラムを読み込ませ、実行させる仕組みである仮想ネットワークプロセッサを提案した。これにより現在のオペレーティングシステムの通信処理に見られるオーバヘッドが軽減することが期待出来る。今後実際に実装を行い並列分散処理などのアプリケーションで処理性能が向上することを確認したいと考えている。

## 参考文献

[1] Fbufs: A high-bandwidth cross-domain transfer facility. In *SOSP*, 1993.

[2] Susan Eggers Chris Maeda Dylan McNamee Przemyslaw Pardyak Stefan Savage Emin Gün Sirer Brian N. Bershad, Craig Chambers. SPIN An Extensible Microkernel for Application-specific Operating System Services. In *Operating Systems Review*, 1995.

[3] Van Jacbson Steven McCanne. The BSD Packet Filter: A New Architecture for User-level Packet Capture. In *USENIX Winter*, 1993.

[4] Vineet Buch Werner Vogels Thorsten von Eicken, Anindya Basu. U-Net: A User-Level Network Interface for Parallel and Distributed computing. In *SOSP*, 1995.