L-018

パケット 選択破棄による高負荷 Linux での効率的な受信処理

Efficient Receiving Process by Selective Drop of Packets on Linux under High Load

山本 紫乃† 楢崎 修二†
Shino Yamamoto Shuji Narazaki

1 はじめに

ネットワーク技術の発展や利用方法の多様化により、ネットワークシステムに対して通信品質保持や高速化への要求が高まってきている。現在の Linux は、プロセスに対しては優先度を用いて CPU 資源の割当てを行っているのに対し、ネットワークシステムではプロセス優先度を考慮せずに全てのパケットに対して公平な CPU 資源の割当てを行っている。そのため、ネットワーク負荷が高い状態になりパケット破棄が発生するようになると、CPU 資源割当ての不適切さが問題になる。

そこで、本研究では適切な CPU 資源割当てを実現するために、プロセス優先度を考慮したパケットの強制的な破棄手法を提案し、パケット選択破棄の有効性を確かめるための提案の部分的な実装に対して実験を実施して、効果の確認をする。

2 Linux のネットワークシステム

受信処理は、パケットが NIC に到着する時をカーネルが予期できないため、割込みハンドラ経由で動作することが特徴である [4]。具体的な流れを以下に説明する (関数名については図 1 参照)。

1. NIC(Network Interface Card) は、到着したパケットを NIC メモリ上のバッファに格納し、割込み要求を発行する (*driver*_interrupt())。
2. カーネルは実行していた処理を一時中断し、割込み要求に対応する割込みハンドラを起動する。この場合、NIC 用ドライバの割込みハンドラ *driver*_rx_interrupt() が起動する。
3. *driver*_rx_interrupt() はパケットを NIC メモリからカーネルメモリに移動させ、パケット毎に受信処理用のソフト割込みハンドラ net_rx_action() の起動要求を出す。
4. net_rx_action() では、受信したパケットのヘッダを解析し、netif_receive_skb() からヘッダ情報に対応するプロトコルの処理関数を呼び出す。その後も、ヘッダの解析を進め、目的のプロセスにパケットのデータ部分を配送する。

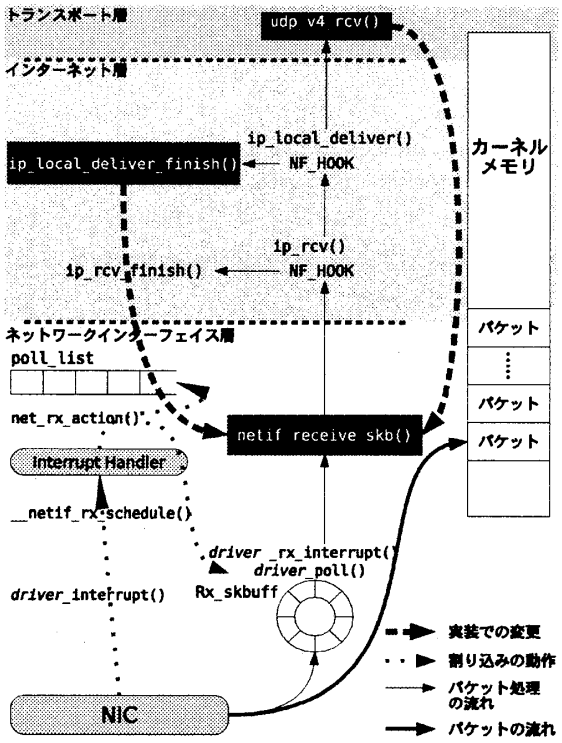

図 1: カーネルの変更点

現在の Linux のネットワークシステムは、パケットの到着順に上記の流れで受信処理を行い、パケットの内容や宛先を考慮して処理を行わないので、全てのパケットに対して公平に CPU 資源を割当てるという問題がある。そのため、高負荷時には、受信処理の途中でのキューへの格納やカーネルメモリへのコピー等で基準の値を越えると無条件にパケットを破棄する。パケットを破棄すると、そのパケットに対してそれまでに割当てられた CPU 資源が無駄になるので、負荷に比例して不適切さが大きくなる。また、カーネルメモリに全てのパケットを格納するため、ネットワークの通信速度が速い場合には、メモリの枯渇の原因となりうる。

3 提案する受信処理方法

前章での問題点を改善するため、本研究は受信プロセスの優先度を用いることによって、Linux のネット

†長崎大学 Nagasaki University

```
if((priority_mode)&&
   (インターネット層のプロトコルが IP である)&&
   (ネットワークからのパケットである)){
     /*IP ヘッダの長さを計算*/
     ihl = (skb->nh.iph->ihl) * 4;
     if(トランスポート層のプロトコルが UDP である。){
       /*TCP ヘッダの先頭番地を計算*/
       uh = (struct udphdr *)(skb->nh.raw + ihl);
       if(priority_mode != uh->送信元ポート番号){
        if(約 50 %の確率で成り立つ){
           /*パケットを破棄する*/
           kfree_skb(skb);
           ret = NET_RX_DROP;
           goto out;
         }
       }
     }
   }
```

図 2: 実装した選択破棄のアルゴリズム

ワークシステムの受信処理において適切な資源割当てを実現する手法を提案する。本手法は、ネットワーク高負荷時の重要なパケットの破棄を避けるため、プロセス優先度を考慮して、優先度の低いパケットを強制的に破棄する。これにより、パケット処理にかかる CPU 処理時間の短縮とメモリの節約を行い、優先度の高いプロセス宛のパケットの再送を防ぐので、プロセスに対して早くデータを渡せるようになる。

本手法では、トランスポート層のヘッダ部分にあるポート番号を利用するが、トランスポート層の処理 (udp_v4_rcv()) で初めてポート番号を取得出来る。しかし、その段階ではパケット処理に CPU 時間を大量に消費してしまっているので、パケット選択破棄ではあまり効果が望めない。そこで、下位層の中 (netif_receive_skb()) に宛先のポート番号を求める処理を前倒しして、パケット選択破棄を行うことにした (図 1 を参照)。

3.1 実装と評価

今回は、一般的なフレームワークの前段階として、ユーザで指定したプロセスを優先度が高いプロセスとし、それ以外のプロセスへのパケットを確定的に選択破棄するものを実装した。実装では、対象パケットをパケットの再送の機能がなく、パケットが破棄されても影響が小さいと考えられる UDP に限定して実装することにした。これにより、ポート番号の位置を特定する計算も一通りで済み、追加するコードも最小限に抑えられる (詳細は図 2 参照)。

カーネル 2.6.10 で NAPI[1] に対応したドライバに変更を加えて前節の実装を行い、1 台のクライアントマシンと 3 台のサーバマシンの全てを Gigabit Ethernet で接続したネットワークを構成し、動画転送を行う実験を行った。パケットの強制的な破棄を行わない時との両方で実験した結果を図 3 と表 1 とに示す。結果から、パケットの受信数の安定性と増加が確認出来るので、パケットの処理に対する CPU 資源への負荷の減少やメモリの効率化につながっていると考えられる。

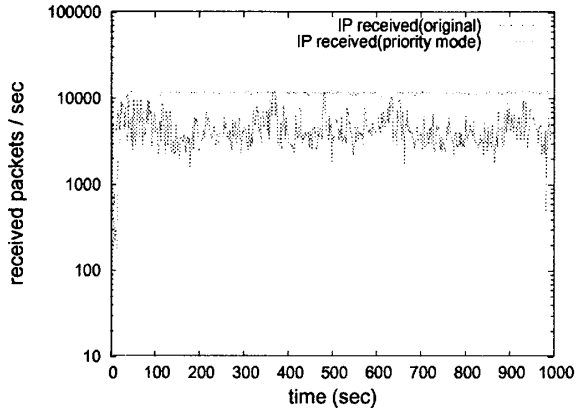

図 3: IP 層でのパケットの受信数

表 1: NIC と IP 層での受信数の総和

	NIC_received	IP_received
original	4458160	445928 (10.00 %)
priority mode	11047289	11044044 (99.96 %)

4 まとめ

本研究では、パケットの宛先となるプロセス優先度を受信処理に反映させることで、ネットワーク高負荷時での効率的なパケット受信処理を行うことを提案した。今回は部分的な実装に対して実験を行った。実験結果からは、一定の効果を確認することが出来た。

今後の課題として、今回の実験結果の詳しい分析をふまえた改良と、動的に現状に対応する強制的なパケットの破棄の実装があげられる。また、他研究 [2, 3] や Net filter との比較を行う予定である。

参考文献

[1] Vincent Guffens. Path of a packet in the linux kernel, Apr 2003. http://www.auto.ucl.ac.be/ guffens/doc/path_packet.pdf.

[2] Nichols K, Blake S, Baker F, and Black D. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. IEFT RFC 2474, Dec 1998.

[3] 尾崎亮太, 中山泰一. Linux におけるプロセス優先度に基づく受信処理の実現. 情報処理学会論文誌, Vol. 45, No. 3, pp. 785–793, Mar 2004.

[4] Miguel Rio, Mathieu Goutell, Tom Kelly, Richard Hughes-Jones, Jean-Philippe Martin_Flatin, and Yee-Ting Li. A map of the networking code in linux kernel 2.4.20, Mar 2004. http: //www.labunix.uqam.ca/ jpmf/papers/tr-datatag-2004-1.pdf.