Vol.2019-OS-146 No.10
2019/5/31

プロファイリングを用いてソフトウェア仮想ネットワークの性能劣化を診断する手法の検討

藤田 祐也[†1]　阿部 洋丈[†1]　李 忠翰[†2]

概要：クラウドシステムにおけるアプリケーションや仮想アプライアンスは VM 上で実現されている。その性能は、物理的なサーバ内部においてソフトウェアにより構築された仮想ネットワークの影響を大きく受けるため、その性能劣化時の原因診断が重要である。一般的な診断手法であるパケットキャプチャでは、ネットワークに関する詳細な情報を得ることができる。しかし、キャプチャ自体のリソースへの負荷やプライバシーの問題が懸念される。仮想ネットワークの性能劣化の原因を診断するためには、パケットキャプチャによらず、ネットワークに関する情報を十分な測定間隔や精度で得ることができ、なおかつ容易に実施できる手法が求められる。そこで本研究では、物理サーバにおいてカーネル関数の呼び出しを観測・集計するプロファイリングに着目した。2 つの物理サーバを用意し、一方において CPU に負荷をかける VM およびもう一方の物理サーバと通信を行う VM をそれぞれ複数実行し、CPU に負荷をかける VM の個数を変化させることによりサーバ全体の CPU への負荷を変化させたときの、VM からサーバの外部へ通信を行うスループットを計測した。この際、VM を実行するサーバにおいてプロファイリングを実施したところ、負荷が増大するにつれて、仮想ネットワークの動作に関連する関数の呼び出し回数と仮想ネットワークのスループットが共に低下した。このことからプロファイリングによる調査によって仮想ネットワークを診断できる可能性を報告する。

A Diagnosis Method for Performance Degradation of Software-based Virtual Network with Profiling

YUYA FUJITA[†1]　HIROTAKE ABE[†1]
CHUNGHAN LEE[†2]

1. はじめに

データベースやストレージなどのアプリケーションを実行するコンピューティング資源ならびに、ルータやファイヤウォール、ロードバランサといったネットワークアプライアンスは、従来専用のハードウェアによって担われてきた（図 1）。一方でクラウドシステムでは、これらのアプリケーションやネットワークアプライアンスは、物理サーバ内部にソフトウェアによって作成された汎用的な仮想サーバによって実現されている（図 2）。仮想サーバを用いる利点には、専用のハードウェアに対する設備投資や管理が不要になるため柔軟に拡張・変更を行える、需要の変動や障害、輻輳に対しても迅速に対応することが可能となる等が挙げられる。

図 1　従来のネットワーク

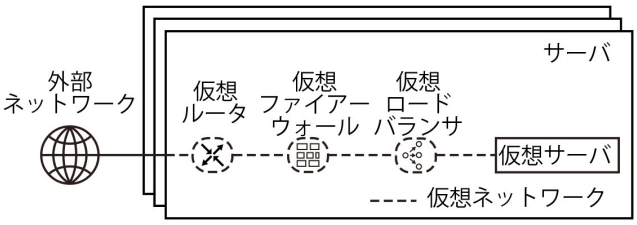

図 2　クラウドにおける仮想ネットワーク

また、仮想サーバを実現している物理サーバ内部にはソフトウェアによる仮想ネットワークが存在する。この仮想ネットワークは仮想サーバと仮想サーバ、あるいは仮想サーバとサーバ外部のネットワークとを接続している。したがって仮想サーバが他の仮想サーバやサーバ外部と通信を行う場合、そのパケットは仮想ネットワークを通じて伝送される。

以上のことから仮想ネットワークの性能を診断することが重要である。スループットやレイテンシといった仮想ネットワークの性能は、仮想サーバが行う通信の品質に影響を及ぼし、アプリケーションや仮想アプライアンスの性能にも影響を与えるためである。

†1 筑波大学　University of Tsukuba
†2 株式会社富士通研究所　Fujitsu Laboratories Ltd.

仮想ネットワークにおけるボトルネックを調査する際には、パケットキャプチャ[1][2]が広く用いられる。パケットキャプチャは特定のネットワークデバイスを通過するパケットを保存ことができるため、このパケットを解析することにより該当するネットワークにおいて生じている問題の調査に有用である。

しかし、パケットキャプチャにはいくつかの問題点が存在する。まず、パケットキャプチャは、パケットを複製しそれをストレージに書き込む処理を必要とするため、キャプチャ自体が CPU やストレージに対する高い負荷となる。何らかの異常がみられるサーバにおいて、パケットキャプチャを実施することはサーバの負荷をさらに高め、仮想サーバやアプリケーションの性能にも影響を与える。加えて、近年の高速なネットワークにおけるパケットキャプチャではネットワークインタフェースを通過するパケットは膨大な量であるため、パケットの一部を取りこぼし、調査に有用な情報を含むパケットが記録されない可能性も指摘されている[10]。さらにパケットキャプチャでは、ユーザが行っている通信のパケットを傍受することになる。収集する対象がヘッダのみであっても、パケットの送信先などの情報を不用意に取得してしまうことになるという問題も存在する。

これらの問題を回避するため、本研究は、サーバ内部の仮想ネットワークにおけるパケットキャプチャによらず、性能劣化の原因を識別する手法の確立を目指す。そこで本研究では物理サーバにおいてカーネル関数の呼び出しを観測・集計するプロファイリングに着目した。プロファイリングはソフトウェア開発においてよく用いられる手法であり、ソフトウェアに含まれる関数の呼び出し回数や実行時間などの情報を計測して性能解析を行うことにより、ボトルネックの調査やデバッグなどに有用な情報を得ることができる。

プロファイリングの結果と仮想ネットワークの性能にどのような関係があるかを調査するため、次のような実験を行った。2 つの物理サーバを用意し、一方において CPU に負荷をかける VM およびもう一方の物理サーバと通信を行う VM をそれぞれ複数実行し、CPU に負荷をかける VM の個数を変化させることによりサーバ全体の CPU への負荷を変化させたときの、VM からサーバの外部へ通信を行うスループットを計測した。

この際、VM を実行するサーバにおいてプロファイリングを実施したところ、様々な負荷によって、仮想ネットワークの動作に関連する関数の呼び出し回数と仮想ネットワークのスループットとが相関係数 0.605 で共に変動した。このことからプロファイリングによる調査によって仮想ネットワークを診断できる可能性を報告する。

2. 関連研究

1 章で述べたように、クラウドシステムにとって仮想ネットワークの性能は重要であるため、様々な提案がなされてきた。

Callegati ら[3][4]や Foresta ら[5]は仮想ネットワークとして、Linux Bridge と Open vSwitch（OvS）とを比較し、高負荷時のスループットやパケットレートについて OvS のほうが高い値が得られるため、OvS を用いることを薦めている。

Lee ら[7][8][9]は、サーバの仮想ネットワークにおいてスループットやレイテンシ等の指標の計測やパケットのキャプチャ結果などを分析し、レイテンシの急激な変動が RTT を増大させる要因であることや、CPU におけるスケジューリングによってパケット送信にスループットが低下する場合があることを指摘した。このようにパケットキャプチャによって得られた結果等を分析することにより、仮想ネットワークの性能診断を行うことができる。しかしパケットキャプチャやモニタリング用のプログラムには、第 1 章で述べた計測自体の負荷という問題が残る[8]。

Suo ら[10]はいくつかの状況における仮想ネットワークの診断のために extended Berkeley Packet Filter（eBPF）によってユーザ定義のプログラムをカーネルにアタッチすることにより様々な指標を計測する vNetTracer を作成した。仮想 VM 間の通信の RTT の内大部分が OvS にて消費されていることを指摘し、各 VM からのパケット入力に対し階層型トークンバケット（HTB）による帯域制限を設けることにより、レイテンシが大幅に改善した、としている。しかし eBPF はネットワークソケットや kprobes 等の限られたトレースポイントよってしか実行できないため、サーバ内部の状態を十分に収集できない可能性がある。

3. プロファイリング

3.1 ソフトウェア開発におけるプロファイリング

ソフトウェア開発において、実行時間やメモリ使用量といった性能を改善するため、プログラムのどこを改良する必要があるかを調査する際、プロファイラを用いてプログラムを実行している際の各種情報を収集し分析する。この情報には関数呼び出しの頻度や関数の実行に要する時間などが含まれる。プロファイリングの結果、長い実行時間を占めている、あるいは呼び出し回数が極めて多いことが明らかになった箇所、すなわちボトルネックとなっている箇所を改善することは、ソフトウェア全体の性能を大きく向上させる。

プロファイラには、割り込みによって一定間隔でサンプリングを行うものや解析対象のプログラムに集計のための命令を埋め込むものなど様々なものがある。その一例として perf[11]や ftrace[12]が挙げられる。perf は実行している

プログラムや OS、ハードウェアにおいて発生する様々なイベントをサンプリングすることが可能である。ftrace は、マシン内部で動作するプロセスで実行されるほぼ全ての関数について、呼び出しのタイムスタンプやその関数を実行した CPU の ID などを取得することが可能である。

3.2 仮想ネットワークの診断への応用

本研究は、プロファイリングを仮想ネットワークの性能調査に応用しようとするものである。プロファイリングによる情報収集は、実行しているプログラムに対して行われるものであり、サーバ内部の仮想ネットワークや VM はソフトウェアによって実現されているものである。したがってこれらに含まれる関数の呼び出し回数や実行時間などの情報を計測して性能解析を行うことにより、ボトルネックの調査に有用な情報を得ることができれば、プロファイリングにより仮想ネットワークの診断が可能となる。

パケットキャプチャにより得られる情報の一つにパケットの送受信が行われた際のタイムスタンプがあり、RTT などを求めることができる。VM が通信を行う際、仮想や仮想ネットワークでは、パケットの送受信に関連する関数が実行される。それらを観察することにより、プロファイリングによっても、RTT などの傾向を推定することができることが期待される。

本研究では、perf により関数の CPU 時間の消費割合を計測する。また、ftrace により各関数の呼び出しのタイムスタンプを用いて、呼び出し回数や頻度を得ている。

4. 実験と考察

サーバ内部の仮想ネットワークの性能調査を行う際に、プロファイリングを行うことによりどのような情報を収集することができるのかを確認するため次のような実験を行った。

4.1 実験環境

図 3 に示すように 10G ビットの物理的なネットワークで接続された 2 台の物理マシン（物理マシン 1, 2）を用いた。物理マシン 1 の内部には複数の VM（VM1, 2, …, 8）を用意する。すべての VM は Open vSwtich による仮想スイッチに接続されており、仮想スイッチはソフトウェアによる仮想ネットワークを通じて物理マシン 1 のネットワークカードに接続されている。

4.2 測定方法

この環境において、VM および物理マシン 2 でスループット計測ツール iperf3 を実行することにより、VM から物理マシン 2 へ TCP パケットを送信する。以後スループットの値としては、VM の側で送出しているパケットのスループットではなく、物理マシン 2 の側において受信したパケットのスループットの値を用いる。また同時に、物理マシン 1 においてプロファイラとして、perf か ftrace のいずれかを実行する。

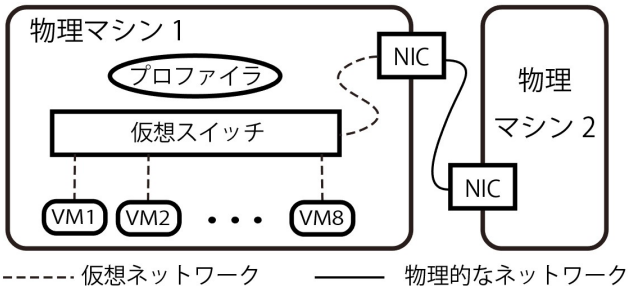

図 3　実験に用いた環境の構成

表 1　実験に用いたマシンの構成

物理マシン		仮想マシン	
CPU	Intel Core i7-4765T (4C8T) (※)	vCPU	2 Cores
RAM	32GB	RAM	2GB
OS	Ubuntu 16.04.5	OS	Ubuntu 16.04.5
NIC	Intel X540-AT2	NIC	virtio

4.3 CPU 時間の消費割合に着目した実験

複数のパターンで VM を動作させ、スループットやプロファイリングの結果がどのように変化するかを調査する。ここで、通信を行う VM は iperf3 により物理マシン 2 と TCP 通信を行う VM のことであり、負荷をかける VM とは yes コマンドを複数実行することにより CPU に負荷をかける VM のことを指す。これらの台数の内訳は表 2 のとおりである。また、いずれのパターンにおいても VM は 8 台起動しており、通信を行う VM でも負荷をかける VM でもない VM はアイドル状態である。

まず、パターン A, B, C により仮想ネットワークに対する負荷が変化した際の、スループットがどのように変化するかを調査した。いずれのパターンにおいても図 4 のように、VM ごとのスループットに大きな差は見られず、合計のスループットは 9.2Gbps 程度であり、全体でのスループットの低下はほぼみられなかった。

※　物理マシン 1 においては、CPU の負荷による影響を大きくすることで悔過をわかりやすくするため、使用するコアを 2 つに制限した。

表 2　実験のパターン

パターン	通信を行う VM	負荷をかける VM
A	2 台	0 台
B	4 台	0 台
C	8 台	0 台
D	2 台	2 台
E	2 台	4 台
F	2 台	6 台

仮想ネットワークに対する負荷が変化しても合計のスループットがあまり変化しないのは、TCP による輻輳制御が働いているためと考えられる。一方で CPU の負荷が増大することにより、スループットが低下するのは、各 VM が CPU による処理を要求するため、パケットを伝送する処理に対して十分に行うことができなくなったためと考えられる。

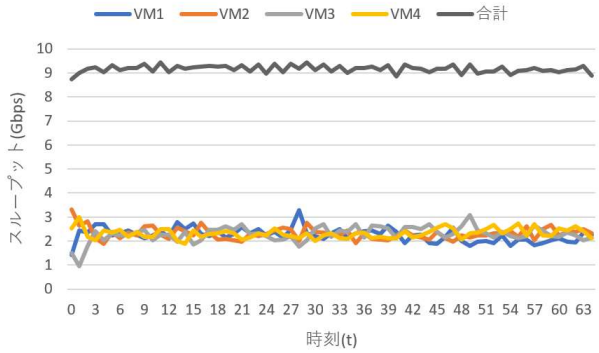

図 4　パターン(B) におけるスループット

次にパターン B, D, E, F により、VM 内で CPU の負荷が高いことにより、物理マシン 1 の CPU に高い負荷がかかっている状態でスループットがどのように変化するかを調査した。結果として、CPU の負荷が増大するにつれ、スループットは振れ幅が大きくなりながら低下していった。

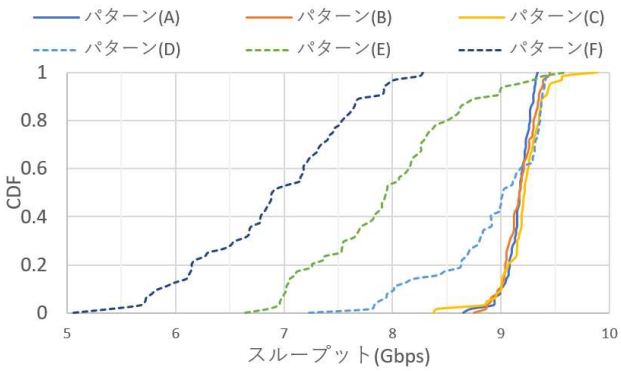

図 5　各パターンにおけるスループットの累積分布関数

これらのパターンについて perf によるプロファイリングの結果を調査したところ、CPU をエミュレーションする処理（vcpu_enter_guest()）、アイドル状態にある VM の停止した CPU を再開するためポーリングを行う処理[13]（kvm_vcpu_block()）、ソフトウェア割り込みに伴いパケットの送信を行う処理(handle_tx())、同じく受信を行う処理(handle_rx())の 4 つによって、物理マシン 1 の CPU 時間の約 90%が消費されていることが分かった（図 6）。

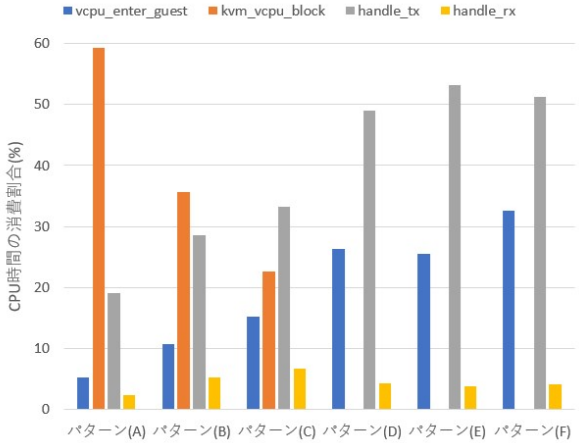

図 6　各パターンにおける関数による CPU 時間消費割合

4.4 関数の呼び出し回数に着目した実験

次にこれらの関数について、ftrace により呼び出し回数を計測した(図 7)。通信に関する処理（handle_tx() と handle_rx()）は、通信を行う VM の増加による影響は少なく、CPU に負荷をかける VM の増加により減少するといった、スループットと類似した傾向を示した。

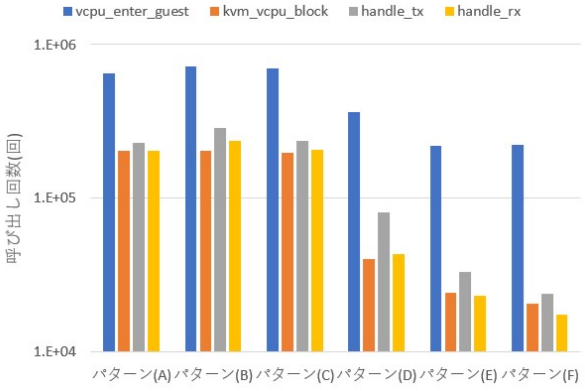

図 7　各パターンにおける関数の呼び出し回数

関数の呼び出し回数とスループットの関係についてさらに詳しく調査するため、次の実験を行った。あらかじめ全ての VM を起動させ十分に時間が経過した後、VM1 のみが、200 秒間の通信を行った。この時間を 25 秒ごとの区間に分割し、各区間が始まるごとに VM2, 3, …, 8 において、

ここまでの負荷をかける VM と同様に yes コマンドの実行することにより、段階的に CPU に対する負荷を増加させ、その際スループットがどのように変動するかを観察した。

プロファイラとして ftrace を用い handle_tx() の関数呼び出しをトレースして得られたタイムスタンプを 0.5 秒ごとに集計した。なお本実験においては、通信を行う VM を収容する物理マシン 1 は主にパケットを送信する側であるため、ソフトウェア割り込みに伴いパケットの送信を行う処理に着目してプロファイリングを実施した。

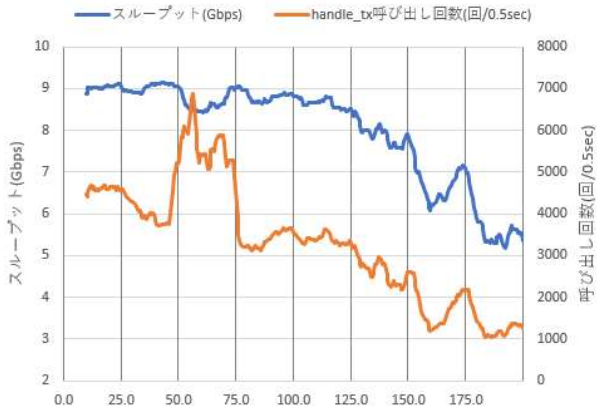

図 8　スループットと handle_tx() 呼び出し回数
（SMA lag = 10sec）

CPU に対する負荷が増大しスループットが低下するにつれ、handle_tx() 呼び出し回数も低下した。0.5 秒間ごとの呼び出し回数をc、スループットをt、これらの平均をそれぞれ $\bar{c}$、$\bar{t}$ とすると、式 1 よりこれらの相関係数は 0.605 であった。

$$C = \frac{\sum(c-\bar{c})(t-\bar{t})}{\sqrt{\sum(c-\bar{c})^2 \sum(t-\bar{t})^2}} \quad \cdots \text{式 1}$$

また、区間 50 ～ 75 において、handle_tx() 呼び出し回数が特異的に増大している。その後区間 75 ～ 100 ではスループットは低下していないにもかかわらず、handle_tx() 呼び出し回数は減少している。これらの区間では、負荷の増大によって新たに何らかの処理が実行された可能性やマシン 1 のリソース割り当てが変化した可能性がある。この点については今後の課題とする。

4.5 考察

iperf3 や yes コマンドの実行によって VM の CPU をエミュレーション負荷が増大する一方、アイドル状態が少なくなることによりポーリングを行う処理が減少しているということが、CPU 時間の消費割合により確認できた。しかし、スループットがあまり変化しないパターンにおいても handle_tx() による消費割合は大きく変化しており、CPU 時間の消費割合は仮想ネットワークの診断を行う上で課題が残る。一方で、関数の呼び出し回数はスループットと一定の相関がみられ、診断の指標として有効であることが期待できる。

5. 結論と今後の課題

5.1 結論

仮想ネットワークの性能を表す指標の一つであるスループットと、ソフトウェアとしての仮想ネットワークに対するプロファイリングにより得られた handle_tx() 呼び出し回数との相関係数は 0.605 であった。プロファイリングは関数の呼び出し回数の他にも様々な指標を計測することが可能であり、仮想ネットワークの診断にプロファイリングは有効であることが期待できる。

しかし handle_tx() の呼び出し回数は非常に多く、そのタイムスタンプを記録することは、物理マシンのハードウェアに対して無視できない負荷を与えているものと考えられる。したがってより低い負荷で記録できるイベントを模索することが必要である。

5.2 今後の課題

プロファイリングを中心にさらに詳細な調査を行い、仮想ネットワークの性能劣化を診断する手法の確立が今後の課題である。

謝辞

本研究は株式会社富士通研究所の助成を受けて行われた。

参考文献

[1] https://www.tcpdump.org/
[2] https://www.wireshark.org/
[3] F. Callegati, W. Cerroni, C. Contoli, G. Santandrea, Performance of Network Virtualization in Cloud Computing Infrastructures: The OpenStack Case, Proc. of 3rd IEEE International Conference on Cloud Networking (CloudNet 2014), Luxemburg, October 2014.
[4] F. Callegati, W. Cerroni, C. Contoli, G. Santandrea, Performance of Multi-tenant Virtual Networks in OpenStack-based Cloud Infrastructures, Proc. of 2nd IEEE Workshop on Cloud Computing Systems, Networks, and Applications (CCSNA 2014), in conjunction with IEEE Globecom 2014, Austin, TX, December 2014.
[5] F. Foresta, W. Cerroni, L. Foschini, G. Davoli, C. Contoli, A. Corradi, F. Callegati, Improving OpenStack Networking: Advantages and Performance of Native SDN Integration, Proc. of 2018 IEEE International Conference on Communications (ICC 2018), Kansas City, MO, May 2018.
[6] https://www.openvswitch.org/
[7] C. Lee, K. Asano, and T. Ishihara, "The Impact of Software-based Virtual Network in the Public Cloud", Proc. IEEE NetSoft'18 Software Defined Networking and Network Function Virtualization Performance (PVE-SDN 2018), pp.494-499, Montreal, Canada, Jun.2018.

[8] C. Lee, H. Abe, T. Hirotsu, and K. Umemura, "Traffic Anomaly Analysis and Characteristics on a Virtualized Network Testbed", IEICE Trans. Information and Systems, Vol.E94-D, No.12, pp. 2353 - 2361, Dec. 2011
[9] C. Lee, R. Mutoh, and N.Oguchi, "The Impact of Microbursts on Throughput of Virtual Switch", IEICE-IN Technical Report, pp. 13 - 18, Hokkaido, Jul.2016 .
[10] K. Suo, Y. Zhao, W. Chen, and J. Rao, "vNetTracer: Efficient and Programmable Packet Tracing in Virtualized Networks", 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS) (pp. 165-175). IEEE.
[11] https://perf.wiki.kernel.org/index.php/Main_Page
[12] https://www.kernel.org/doc/Documentation/trace/ftrace.txt
[13] https://www.kernel.org/doc/Documentation/virtual/kvm/halt-polling.txt