B-042

OS スローダウン発生時のプロセス情報取得方式

A study of method for acquiring process information under OS slowdown situation

亀井 仁志† 中村 隆喜†‡ 薦田 憲久‡
Hitoshi Kamei Takaki Nakamura Norihisa Komoda

1. はじめに

データセンタなどで用いられる高機能サーバに障害が発生した場合，管理者は管理プログラムを用いて障害情報を取得し，その情報を元に障害の要因を特定して対策を行う．障害情報にはメモリダンプ，システムログ，プロセス情報などがある．これらの障害情報のうち，プロセス情報は障害要因の一次切り分けに利用できるため有用である．

しかし，OS スローダウン障害が発生している場合，プロセス情報を取得するプログラム自体の負荷により，プロセス情報取得処理が規定の時間内に完了しないことがある（タイムアウトと呼ぶ）．ここでは，OS は動作しているが OS の応答性能が悪化することを OS スローダウン障害とする．

特に，1 台の高機能サーバ上で大量のプロセスを起動している場合，プロセス情報取得処理がタイムアウトする可能性が高くなる．

本論文では，OS スローダウン障害発生時に動作可能なプロセス情報取得方式を提案する．

2. プロセス情報取得の課題

Linux 等の UNIX 系 OS にはプロセス情報を表示する ps コマンドがある．本章は ps コマンドの動作概要を述べ，負荷の要因について述べる．

2.1. ps コマンドの概要

ps コマンドは OS のカーネルプログラムが管理するプロセス情報を proc ファイルシステム（procfs）のファイルから取得し，データを整形して画面へ出力する．画面ではなくファイルへ出力する場合はリダイレクトを用いる（図 1）．

ps コマンドは出力情報の整形方法を変更するオプションを多数サポートしている．本論文では，動作中の全プロセスに対して，状態フラグやメモリサイズなどを表示する最も一般的な「-elf」オプションを用いたプロセス情報取得を検討対象とする．

2.2. ps コマンドの負荷と要因

ps コマンドの負荷と要因について以下に述べる．

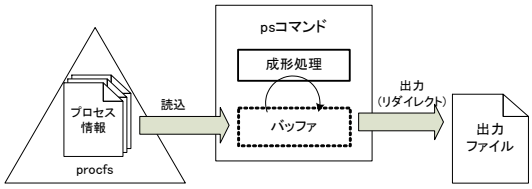

図 1 ps コマンドの動作概要

(ア) システムコールの大量発行

ps コマンドは procfs からプロセス情報とメモリ容量やカーネルバージョンといったシステム情報を取得し，出力データを整形して表示する．ユーザプログラムである ps コマンドが procfs から情報を取得して画面へ出力するには open，read，write などのシステムコールを OS へ発行する必要がある．システムコール発行は CPU 負荷が高いため[1]，システムコール発行数が増加すると OS スローダウン障害の影響を受けやすくなる．

(イ) メモリ確保

ps コマンドは出力データを整形するために procfs から取得したプロセス情報をバッファする．特に，ソートオプションを追加すると，数 M バイトのメモリを動的に確保するため，メモリ確保の負荷が高い．

(ウ) 出力データの整形処理

ps コマンドの出力データ整形処理による CPU 負荷が高い．特に，ソートオプションを追加して，出力結果を UID 順にソートする場合，バッファされたデータを並べ替えるために比較処理やリスト処理などが必要となり大きな負荷がかかる．

OS スローダウン障害が発生している場合，上記の負荷がプロセス情報取得のタイムアウトを引き起こす要因となる．

3. 軽量化 ps コマンドの提案

本論文では，OS スローダウン障害が発生していても動作可能な軽量化 ps コマンド群を提案する．軽量化 ps コマンド群の動作概要について図 2 に示し詳細を以下に述べる．

(ア) ps コマンド処理の分割

ps コマンドの処理を，(a)procfs からの情報取得処理を行う軽量 ps コマンド（lps，lightweight ps）と，(b)出力情報の整形処理を行うアナライザ（lpsa，lightweight ps analyzer）へ分割する．

† （株）日立製作所 横浜研究所，Yokohama Research Laboratory，Hitachi, Ltd.

‡ 大阪大学大学院情報科学研究科，Graduate School of Information Science and Technologies, Osaka University

そして，OS スローダウン障害が発生している高機能サーバで lps コマンドのみ動作させ，処理の分割により出力情報の整形処理の負荷を抑える．

(イ)システムコール発行回数の削減

procfs からの取得情報量とプロセス情報の出力情報量を削減してシステムコール発行回数を抑える．

高機能サーバは，組み込み機器と同様に，ハードウェア仕様や OS は予め決まっているため，ps コマンドが取得しているシステム情報は既知である．従って，lps コマンドは ps コマンドが取得する以下のシステム情報を取得しない．

- カーネル情報：/proc/version
- システムステータス情報：/proc/stat
- 実行プロセスステータス情報：/proc/self/stat
- メモリ情報：/proc/meminfo
- 最大プロセス数：/proc/sys/kernel/pid_max

プロセス情報の取得では，ユーザ/カーネルモードの実行時間や実行ステータスなど OS スローダウン障害に関する情報のみ取得することとした．つまり，lps コマンドはプロセス毎に以下の情報を取得する．

- /proc/<PID>/stat
  - 保持情報：pid, cmd, status, ppid,flags, utime, ktime, priority, nice, stime, vsize, rsize, wchan
- /proc/<PID>/status
  - 保持情報：euid
- /proc<PID>/cmdline
  - 保持情報：先頭から 256 バイト

さらに，プロセス情報の表示では，ソート等の整形を行わずに標準出力へ csv 形式で出力する．その際，空白等の整形に関する文字を出力結果へ入れず，出力情報量を削減する．

(ウ)静的メモリ確保

lps コマンドの動作中にメモリ確保が失敗することを防ぐため，メモリを動的に確保せず，lps コマンドの起動時にメモリを静的に確保する．

lps コマンドでは，8K バイトの入出力用バッファと約 400 バイトのデータ保持用メモリを静的に確保し，そのメモリ領域のみを使用して，procfs からの情報取得とプロセス情報の出力を行う．

さらに，ライブラリ関数のうち，動的メモリを確保する opendir 関数，readdir 関数，closedir 関数，fopen 関数，fclose 関数を使用せず，それらの関数が提供する同等の機能を lps コマンドへ作りこむ．

4. 評価

3 章で提案した lps コマンドを実装して評価した．評価では，1G バイトのメモリを割り当てた Linux 導入済み仮想化環境を作成し，高 Disk I/O 負荷を発生させて OS をスローダウンさせた．その環境で lps コマンドと ps コマンドを 80 回動作させて動作完了時間を比較した．評価結果を図 3 に示す．

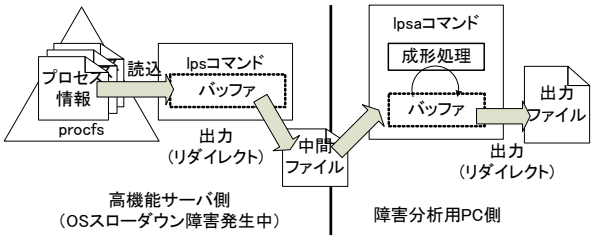

図 2　提案方式の動作概要

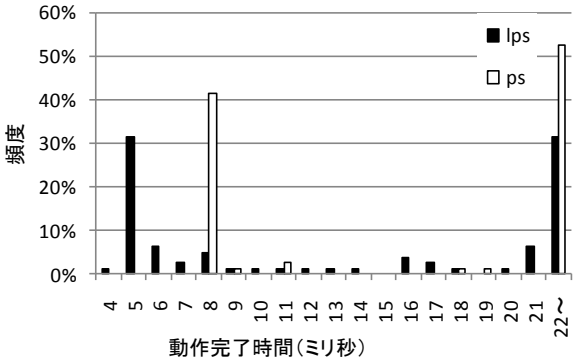

図 3　動作完了時間のヒストグラム

評価の結果，lps コマンドは 5 ミリ秒の動作完了頻度が最も高く，31.25%を占めていることがわかった．また，ps コマンドは 8 ミリ秒の動作完了頻度が最も高く，41.25%を占めていることがわかった．一方，80 回実行したコマンドの平均動作完了時間は，lps コマンドが 6.09 秒であり ps コマンドが 16.22 秒であった．また，lps コマンドの最大動作完了時間は 109.80 秒であったのに対し，ps コマンドは 161.06 秒であった．また，プロセス情報取得タイムアウトを 5 秒とすると，lps コマンドは 85.00%がタイムアウトせずに動作した．一方，ps コマンドは 62.50%がタイムアウトせずに動作した．

評価の結果，コマンドの取得情報と出力情報を厳選し，メモリ確保方式を動的メモリ確保から静的メモリ確保へ置きかえることで，動作完了時間を削減できることが分かった．

5. おわりに

本論文では，OS スローダウン障害が発生した際のプロセス情報取得方式について検討した．検討では，ps コマンドの負荷を分析し，プロセス情報取得の軽量化手法を検討した．そして，評価プログラムを作成して ps コマンドと比較した結果，提案内容の有効性を確認した．

6. 参考文献

[1] L. Soares and M. Stumm: FlexSC: flexible system call scheduling with exception-less system calls, *Proc. of the 9th USENIX conference on Operating systems design and implementation* (2010).