LTTngを使った情報爆発時代の性能解析の検討

権 奇徳†, 菅谷 みどり†, 大野 有輝‡, 中島 達夫†

早稲田大学基幹理工学研究科†, 早稲田大学基幹理工学部情報理工学科‡

概要

組み込みシステムは我々の実生活に適用されているにも関わらず，日々障害が発生する．例えば携帯電話では，ソフトウェアのバグによる通話障害の発生やシステム停止例といった例が報告されている[1]．こうした問題は，通常障害解析の上対処されるが，カーネル内での障害の場合，ユーザーモードの情報のみならずカーネル内の情報を取得する必要がある．

本論文ではカーネル内のイベントを取得するLTTng(Linux Trace Toolkit next generation [3])を用いたログ分析システムを提案する，本研究で提案するログの取得とイベント分析の仕組みを用いる事で，障害解析を容易にすることができる．本論文では具体的な障害としてタイマ遅延をあげ，提案システムによる原因分析及び解決方案を提示する．

1. 序論

システムにおいては，アプリケーションの予期せぬ動作やハードウェア割込みの増大により負荷が高くなる事がある．負荷の影響を受ける障害の上 1 つとしてタイマの遅延がある．通常，動画処理などのリアルタイムアプリケーションには時間制約があるため，周期的な時間処理を行うためには，周期タイマの利用が不可欠である．しかし，周期タイマが正しい時刻よりも遅れて実行された場合，これらのアプリケーションがデッドラインをミスするなどの深刻な影響がある．

本論文では，タイマの遅延を発見し，さらに要因分析を行う方法について述べる．

Performance analysis of information explosion by using LTTng.

† Kiduk Kwon, Midori Sugaya, Yuuki Ohno and Tatsuo Nakajima,

Department of Computer Science and Engineering, Graduate School of Fundamental Science and Engineering, Waseda University.

2. ログを用いたタイマ分析の提案

2.1 ログの利用

タイマの遅延の発見と分析においては，まず問題発生時の状況を確認する為に，タイマイベントを記録（ログ）する必要がある．特に，タイマはカーネル内部での割込み処理により実現されている為，カーネル領域で発生するイベントデータの取得が必要である．また遅延要因の解析の為にはタイマベント以外の情報も必要である．

そこで，我々はカーネル内部のイベント情報が取得できる LTTng と LTTV(Linux Trace Toolkit Viewer)[3]を利用して実際にタイマの遅延が発生する際のログを取得するものとした．LTTng のオーバーヘッドは低くため，アプリケーションの動作への影響が 1.54%から 2.28%[3]というメリットがある．

しかし，LTTng はカーネルのイベント全体をロギングするため，ログの量が多い(1 秒でテキスト 90MB 位)．その為，タイマイベントを記録したログから障害を特定するのに長時間かかるという問題がある．我々は効率的に問題分析する為には，問題対象の情報を抽出する必要があると考えた．また，抽出したデータはさらに，要因解析が必要である．その為これらを含めた分析システムを提案するものとした．

2.2 システム構成

本研究では，分析システムを作成した．システムでは，イベントのロギング部分とデータ分析部分に分けてシステムを構成した．分析を行うためのシステムを図 1 に示した．

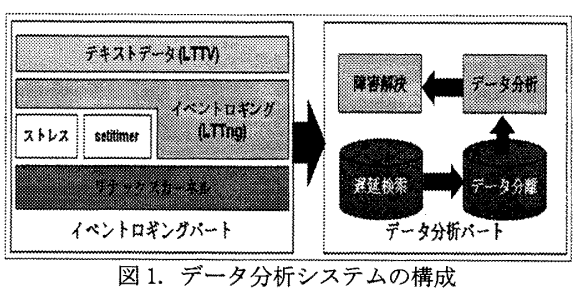

図 1. データ分析システムの構成

図1では，イベントロギング部分にて，LTTngを拡張し，markerを追加して必要なデータをログするものとした．データ分析部分ではログから（1）遅延の発生有無をチェックし，遅延が発生した部分から，その他の情報を分離する．さらに(2) データ分析プログラムにて，データを分析し，遅延要因を特定した．

2.3 高解像度タイマの分析

本研究では計測対象として，Linuxに実装されている高解像度タイマ(High Resolution Timer)を用いた[4]Linux の高解像度タイマはAPIC (Advanced Programmable Interrupt Controller)によるハードウェア割込みと，高解像度タイマのソフト割込みによって実現されている．本研究では高解像度タイマのフトウエア割込みの遅延を，提案システムよって計測・分析を行った．

2.4 計測

システム環境は Pentium4 uniprocessor 1.83GHz，メモリ 1GB で， Linux Kernel 2.6.23 の上でLTTng 0.10.0-pre7[5]と LTTV 0.10.0-pre1[5]を用いて実現した． 高解像度タイマの実行をロギングするために setitimer() システムコールを用いた．本システムコールは，周期を一定時間で繰り返す．

setitimer()に，周期は500μsに設定し，同時にI/O stressプログラム(100MBファイルのread(),write()繰り返し)を1万回実行した．

図2は遅延値と遅延の頻度を集計した結果である． x軸はns単位の時間，y軸はそれぞれ100us の範囲で観測されたデータの個数を示す．

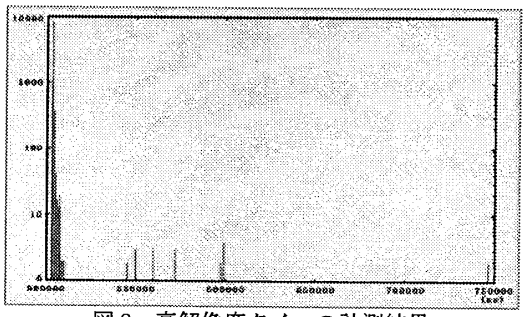

図2. 高解像度タイマの計測結果

実験結果より，500usに対し，100us以上の遅延が3，また，70μs-60μs，60us-50μs，0-50usの範囲での遅延データはそれぞれ2 との結果になった．つまり，100us 以上の遅延は全体の0.01% 生じている．

2.5 遅延の原因分析

高解像度タイマでは，タイマの割込み処理はハードウェアからの割込み処理の後，ソフトウェア割込みにてタイマのハンドラ処理を行う．遅延の生じた箇所でさらにログ内容分析するとソフトウェア割込処理の遅延が見られた．高解像度タイマのソフトウェア割込みは，今回負荷をかけたディスク割込みのソフトウェア割込みより優先度が低いため，ディスク負荷が高い場合に，遅延の発生があることが原因として考えられる．実験では，高解像度タイマの優先度を，ディスク割込みより高い値に変更を行った．結果100us 以上の遅延が無くなるなどの結果が得られたが，本改善手法との因果関係も含め，現在検証中である．

3. 結論

本論文では，障害解析の一手法として，カーネルのロギングを用いた分析システムの提案を行った．実際に，システムを用いて，高解像度タイマの遅延の原因を分析した．またタイマの遅延の原因を特定することができた．今後は組み込みシステムから障害が発生した場合ハードウェア割込みも考慮したシステム分析をする予定である．

参考文献

[1] 携帯情報ページ，http://www.mobiletech review.com/

[2] Linux Kernel State プロジェクトページ,Tracerhttp://lkst.sourceforge.net/

[3] Mathieu Desnoyers, Michel R. Dagenais, "The LTTng tracer: A low impact performance and behavior monitor for GNU/-Linux", Ottawa Linux Symposium 2006.

[4] Thomas Gleixner, Douglas Niehaus,"Hrtimers and Beyond: Transforming the Linux Time Subsystems", Ottawa Linux Symposium 2006.

[5] LTTngプロジェクトページ, http://ltt.polymtl.ca/