B-45

システムの挙動に対応して動作の切り替えが可能なイベントトレーサLKSTの開発

畑崎　恵介†　　中村　哲人†　　芹沢　一‡
Keisuke Hatasaki　Tetsuhito Nakamura　Kazuyoshi Serizawa

1．緒言

近年，企業システムにおいて，オープンソースのOSであるLinuxの利用が拡大している。Linuxは機能や性能においては商用OSと同等レベルであるが，企業システムで必須となる障害解析機能に関してはいくつかの課題を残している。

そこで，我々はLinuxシステム障害時の解析やデバッグを加速させ，企業システムへも適用可能な高信頼システムの構築に貢献する機能として，Linuxカーネル状態トレーサLKST(Linux Kernel State Tracer)を開発している。

LKSTの開発においては，企業内のシステム運用を考慮し，(1)常時システム監視を可能とするためオーバヘッドの最適化，(2)デバッグに必要な障害情報の収集と確実な回収，(3)更新の早いオープンソースのデバッグに対応する高い拡張性，の実現に着目した。

上記目的の実現に向け，我々はLKSTに「イベントマスクテーブル機構」の実装を提案した。本稿では，本機構の詳細と，その評価について述べる。

2．LKSTの概要

LKSTはLinuxカーネルの状態変化をトレースし，障害発生原因の迅速な追求を支援する機能である。本機能は，日立，IBM，富士通，日電の4社で進めているエンタープライズLinux機能強化に向けた提案活動の一環として，富士通，日立を中心に開発を進めている。

LKSTの概要を図1に示す。

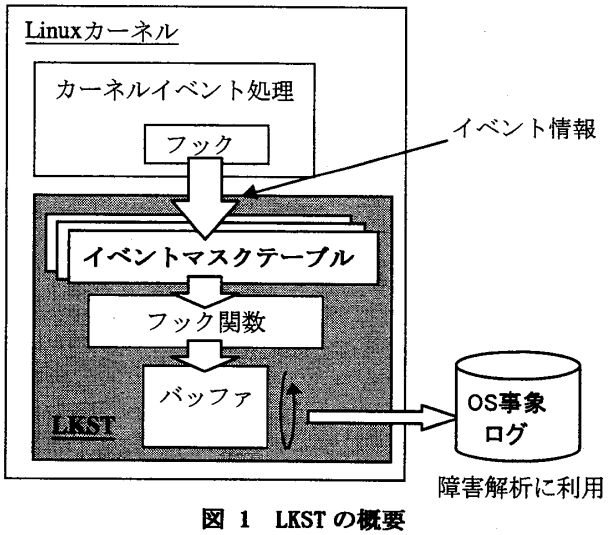

図 1　LKSTの概要

LKSTは，Linuxカーネルのイベント発生時に通過するソースコードにフックを挿入し，イベント発生時にフック関数を呼び出し，LKSTのイベントバッファへと記録する。イベントバッファは一杯まで記録されると，再び先頭から記録を開始する。

カーネル内のフックはプロセスコンテキストスイッチやシステムコールなど，様々なイベント発生個所に挿入されている。収集するイベント種が多ければ，それだけ障害発生時にデバッグに有用な情報が手に入る可能性が高い。しかし，(1)フック関数の呼び出し回数増加によるオーバヘッド増大，(2)バッファにへの記録情報増大による重要情報の上書き消失，という問題が発生する。そこで我々は，「イベントマスクテーブル機構」により，トレースするイベントを選択可能とした。次にイベントマスクテーブル機構の詳細について述べる。

3．イベントマスクテーブル機構

イベントマスクテーブルは，発生したイベントの種類に応じて，呼び出すフック関数を切り替える機能を実現するテーブルである。図2にイベントマスクテーブルを示す。

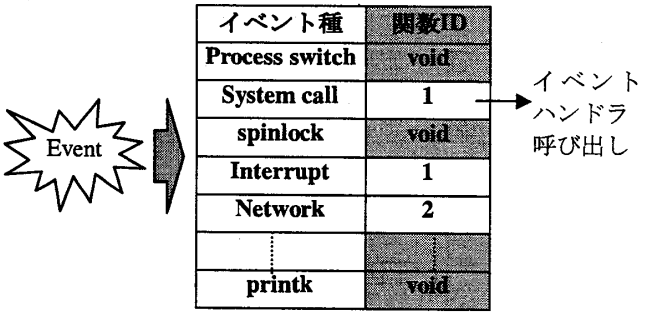

図 2 イベントマスクテーブル

LKSTは，イベントマスクテーブルに定義されているイベント種と関数IDの関係に応じて，イベント発生時に呼び出す関数を決定する。このとき呼び出す関数のことを「イベントハンドラ」と呼ぶ。図の関数IDがvoidとなっているイベントは，イベントハンドラを呼ばないことを意味する。これは，トレースする必要の無いイベントをマスクすることに相当する。イベントマスクテーブルはユーザがコマンドを介して自由に定義可能であり，複数の種類を登録できる。また，トレース中にシステムの稼動状況に合わせてイベントマスクテーブルを動的に切り替えることも可能である。

イベントハンドラには，バッファへの書き込み，イベントバッファの切り替え，イベント発生関数のカウントなど，様々な種類が存在する。また，イベントハンドラ

†（株）日立製作所　中央研究所
‡（株）日立製作所　システム開発研究所

の内部変数を外部から操作することを可能とする「イベントハンドラ制御関数」を実装している（図 3参照）。このイベントハンドラ制御関数は，LKST が提供する API を利用して，ユーザからの呼び出しを可能としている。

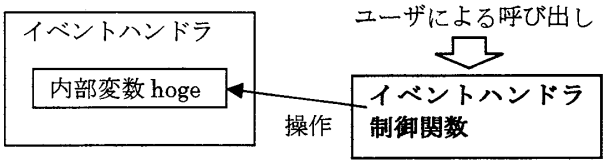

図 3 イベントハンドラ制御関数

このイベントハンドラ制御関数を利用することにより，イベント発生回数をカウントするイベントハンドラを利用中に，イベントのカウント数のリセットする，などの操作が可能である。

さらに，Linux は OS の動作中にカーネルコードに新たにコードを組み込む「カーネルモジュール」と呼ばれる機能をサポートしている。これを利用することで，イベントハンドラとして登録する関数を，ユーザがモジュールとして作成することが出来る。

4．評価・考察

Linux カーネルのコンパイルをベンチマークに，LKST の性能評価を行った。以下に測定環境と結果を示す。

測定環境： CPU: Pentium2 450Mhz x 2 RAM:256MB
System: Redhat 7.2J Kernel Version:2.4.17

表 1 評価結果

#	カーネル	時間[s]	オーバヘッド[%]
1	Original	398.56	-
2	LKST all	419.27	5.20
3	LKST custom	405.38	1.71

表 2 イベントの発生比率

イベント分類	割合[%]
プロセス管理系	0.63
メモリ管理系	4.02
ハードウエア系	7.11
システムコール	6.76
ロック系	80.73
I/O 系	0.74

表 1の#2は全てのイベントをトレースした場合の結果である。表 2は，#2 に対してカーネルコンパイル時に発生するイベントの発生比率を示す。この結果から，ロック系イベントのトレースによるオーバヘッドが大きいことがわかる。ロック系イベントは主にデッドロック解析に利用するため，デバイスドライバ等のデバッグには有効であるが，システム運用時の障害解析に利用することは少ない。そこでイベントマスクテーブルのカスタマイズし，ロック系イベントをトレースしないことで，表 1の#3 のようにオーバヘッドを削減した。このように，ユーザがイベントマスクテーブルを最適化することで，システムの状態や目的に応じた適切なオーバヘッドの下で障害解析が出来る。

また，他の実例として下記のようなデバッグ事例にイベントマスクテーブルは有効である。

（1）特定プロセスのイベント情報の確実な回収

まず，３つのイベントバッファを作成し，次に「プロセスコンテキストスイッチ」イベントに対してイベントバッファを切り替えるイベントハンドラを選択する。これにより，以下の手順でトレースが行われる。

- 最初のバッファにイベント情報が記録される
- 目的のプロセスへコンテキストスイッチすると，２つ目のバッファへと切り替え，このバッファへイベント情報が記録される
- 目的のプロセスから別のプロセスへスイッチすると，３つ目のバッファへの記録を開始

この結果，目的のプロセスのイベント情報のみ２つ目のイベントバッファに記録されることになる。他のプロセスのイベント情報は２つ目のイベントバッファに上書きされることなく，確実に回収することができる。

（2）発生イベントにスタックトレース情報を追加

Linux のモジュール組み込みの機能を用いて，OS 動作中にスタックトレース機能を追加したイベントハンドラを組み込む。これを全てのイベントのイベントハンドラとして選択することで，LKST にスタックトレースの機能を追加することができる。

このように，イベントマスクテーブルの機能を実現により，(1)オーバヘッドの削減，(2)イベント情報の確実な回収，(3)柔軟な拡張を可能とした。

LKST はメインフレーム Linux の高信頼化に向けて Linux for AP8000 に実装済みである。また、各 Linux ディストリビュータへの採用を呼びかけており、一部では採用されている。LKST はオープンソースとして，ホームページ(http://lkst.sourceforge.jp)で公開中である。

5．まとめ

Linux システムのデバッグ加速を目的に，Linux カーネル状態トレーサ LKST を開発した。LKST では，イベント発生時に呼び出すフック関数を動的に切り替えることを可能とするイベントマスクテーブル機構を実装した。これにより，(1)トレースにより発生するオーバヘッドの最適化と，(2)イベント情報の確実な回収を実現した。さらに，(3)トレーサの柔軟な拡張を実現しており，この機能によりカーネルイベントを契機に外部プログラムを駆動することも可能である。

今後，デバッグに有効なイベントハンドラの追加，フックポイントの自由な設定，セキュリティ対策ツールへの拡張など行っていく予定である。

6．参考文献

[1]堀川隆「Linux ソフトウエア・イベント・トレーサ」Linux Conference 2001

[2]Daniel P.Bovet, Marco Cesati "Understanding the Linux Kernel" O'REILLY

[3]Kernel Hooks, http://www-124.ibm.com/developerworks/oss/limix/projects/KernelHooks

[4]LKCD, http://lkcd.sourceforge.net

[5]The Linux Trace Toolkit, http://www.opersys.com/LTT