コードレビュー Vol.14

CODE Review

USP 研究所技術研究員
written by 大内智明

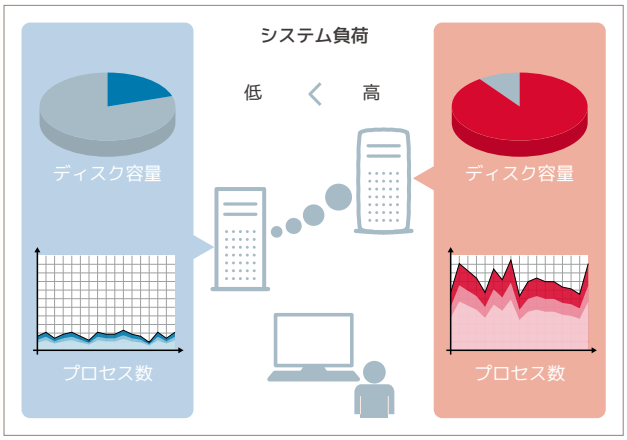
今回は、前回に引き続いて運用機能の一つであるロードアベレージ情報の監視についてお話します。

監視機能の必要性

業務で使用するサーバーは、正常な動作を継続していく必要があります。そのためには、システム情報を定期的に監視して、取得した情報が正常な動作範囲であること、エラーや警告の発生、実行しているプロセス数も許容範囲内であることを確認していく必要があります。

もし、システム利用状況が許容範囲から外れても、常に監視を行っていれば早急にシステム負荷やハードウエア故障に対策／対応ができます（図1参照）。

図1　システム運用状況



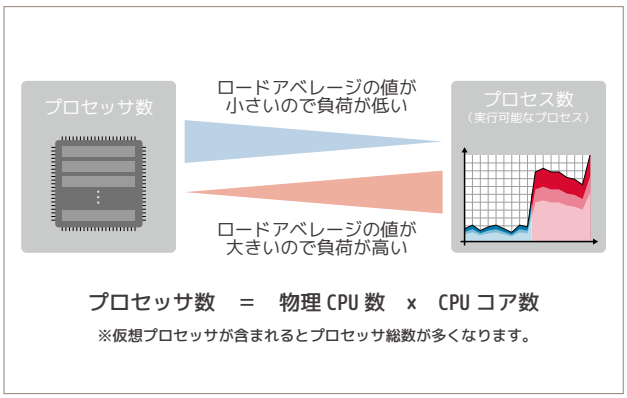
技術的な概要

ロードアベレージとは、CPU のプロセッサ数とプロセス数を元に、システムにかかる負荷を値で表しています。

ロードアベレージの値が、プロセッサ数より大きいとシステム負荷が大きくなります。

CPU がプロセスを1つずつ使用しているので、もし1プロセスが、1CPU を占有するとロードアベレージの値が1になります（図2参照）。ロードアベレージの値が、プロセッサ数を超えるとプロセス数が処理できるプロセッサ数より多い状態となり、システム負荷が高い状態です。また、ロードアベレージの値は、瞬間的に実行しているプロセスではなく、過去1分間・5分間・15分間といった期間で実行しているプロセス数を示しています（画面2参照）。

図2　ロードアベレージと負荷



【補足】
ロードアベレージの値は、uptime コマンド／w コマンド／top コマンド、/proc/loadavg などから取得できます（画面2参照）。

リスト1　定期的に起動して、複数サーバーのロードアベレージを保存するシェル

```
#!/bin/bash -xv
#
# KANSI_LOADAVERAGE
# ロードアベレージ収集
#
#  Written by aoki 20141219

# ログ出力
appd=/home/usp/KANSI
logfile=$appd/LOG/$(basename $0).$(date +  %Y%m%d)
exec 2> $logfile

# 変数設定
export LANG=ja_JP.UTF-8
export PATH=/home/UTL:/home/TOOL:$PATH
cgid=$appd/CGI
htmd=$appd/HTML
logd=$appd/LOG
repd=$appd/REPORT
semd=$appd/SEMAPHORE
shld=$appd/SHELL
tbld=$appd/TABLE
tmp=/tmp/$$
today=$(date +%Y%m%d)
todayhms=$(date +%Y%m%d%H%M%S)

# エラーチェックと終了処理の定義
ERROR_CHECK(){
  [ $(plus ${PIPESTATUS[@]}) -eq 0 ] && return
  rm -rf $tmp-*
  touch $semd/$(basename $0).$HOSTNAME.ERROR.$today
  exit 1
}

# 起動時刻
touch $semd/$(basename $0).$HOSTNAME.START.$today

#####################################################################
# テーブルを読み込んで用意しておく
cat /etc/hosts | decomment | self 2 1 | msort key=1 > $tmp-ip
ERROR_CHECK
touch ${repd}/LOADAVERAGE.${today}

# 各サーバーごとに容量を計測
for host in $(msctrl -ctrl C -print host); do
 ip=$(nameread $host $tmp-ip | itouch _ -)
 ssh ${host} uptime | tr -d , | sed 's/^.* \([0-9]*\) days.* \([0-9]*\) users*/\1 \2/g' | self 1 2 NF-2/NF |
      # 1:起動日数 2:接続ユーザ数  3:５分平均  4:１０分平均  5:１５分平均
 awk 'NF==5{print "'${host}'","'${ip}'","'${todayhms}'",$0}'
done |
# 1:サーバー名  2:ＩＰ          3:年月日時分秒
# 4:起動日数    5:接続ユーザ数  6:５分平均  7:１０分平均  8:１５分平均
LANG=C sort -k1,2                |
up3 key=1/3 $repd/LOADAVERAGE.$today > $repd/LOADAVERAGE.$today.new
```

走行ログを記録します。
ロードアベレージを取得するスクリプトは、定期的に動作するため、記録するログのファイル名は、1日の中で最新のファイルだけ残しています。
走行ログは、プログラムが実行途中、エラーなどにより停止した場合に、原因を発見するのに使用します。

decomment はユニケージコマンド、画面1参照

msctrl はユニケージコマンド。
msctrl -ctrl C -print host は、管理するサーバーの中で稼働中の IP アドレス一覧を出力します。

システムの稼働時間を表示するコマンド、画面2参照

```
55 ERROR_CHECK
56
57 # 最新のものに置き換え
58 mv $repd/LOADAVERAGE.$today.new $repd/LOADAVERAGE.$today
59 ERROR_CHECK
60
61 ###############################################################################
62 # 終了
63 echo "$HOSTNAME $(basename $0) END $(date +%Y%m%d%H%M%S)" >> $logd/UPCNT
64 touch $semd/$(basename $0).$HOSTNAME.END.$today
65 rm -f $tmp-*
66 exit 0
```

❶

画面1　decomment：不要なコメントや空白を除くコマンド

```
$ cat table
# テスト用のテーブルです
# 1:地域コード　2:地域名称　3:ブロックコード
  1 東京都      A
  2 埼玉県      A
  3 神奈川県    B
# 以下は地域単位になります
101 北海道空知 Z
102 北海道後志 Z
103 北海道胆振 Z
```

```
$ decomment table
1 東京都 A
2 埼玉県 A
3 神奈川県 B
101 北海道空知 Z
102 北海道後志 Z
103 北海道胆振 Z
```

画面2　uptime：システムの稼働時間を表示するコマンド

```
$ uptime
22:04:51 up 27 days, 11:50,  1 user,  load average: 0.04, 0.09, 0.03
```

Ⓐ Ⓑ Ⓒ Ⓓ

Ⓐ現在時刻　Ⓑ稼働日時　Ⓒログインユーザー数
Ⓓ過去1分、過去5分、過去15分のロードアベレージ。ロードアベレージとは、1 CPUあたりで実行待ちになっているプロセスの平均値。

リスト2　保存したロードアベレージを取得して、画面出力するCGI

```
 1 #!/bin/bash -xv
 2 #
 3 # KANSI_LOADAVERAGE.CGI
 4 # ロードアベレージ確認
 5 #
 6 # Written by aoki 20130829
 7
 8 #ログの出力
 9 appd=/home/usp/KANSI
10 logfile=$appd/LOG/$(basename $0).$(date +%Y%m%d)
11 exec 2> $logfile
12
13 # 変数設定
14 export LANG=ja_JP.UTF-8
15 export PATH=/home/UTL:/home/TOOL:$PATH
```

```
16 cgid=$appd/CGI
17 htmd=$appd/HTML
18 logd=$appd/LOG
19 repd=$appd/REPORT
20 semd=$appd/SEMAPHORE
21 shld=$appd/SHELL
22 tbld=$appd/TABLE
23 tmp=/tmp/$$
24 today=$(date +%Y%m%d)
25 todayhms=$(date +%Y%m%d%H%M%S)
26
27 # エラーチェックと終了処理の定義
28 ERROR_CHECK(){
29   [ $(plus ${PIPESTATUS[@]}) -eq 0 ] && return
30   rm -rf $tmp-*
31   touch $semd/$(basename $0).$HOSTNAME.ERROR.$today
32   exit 1
33 }
34
35 # 起動時刻
36 touch $semd/$(basename $0).$HOSTNAME.START.$today
37
38 ##########################################################################################################
39 [ "$QUERY_STRING" == "now=1" ] && $shld/KANSI_LOADAVERAGE
40
41 touch $repd/LOADAVERAGE.$today
42 cat $repd/LOADAVERAGE.$today |
43 getlast 1 2 |
44 dayslash HH:MM 3 |
45 # 1:サーバ名  2:ＩＰ          3:年月日時分秒
46 # 4:起動日数  5:接続ユーザ数  6:１分平均  7:５分平均  8:１５分平均
47 awk '{s6="_";for(i=1;i<$6&&i<10;i++) s6=s6"★";
48       s7="_";for(i=1;i<$7&&i<10;i++) s7=s7"★";
49       s8="_";for(i=1;i<$8&&i<10;i++) s8=s8"★";
50       print $0,s6,s7,s8}' > $tmp-list
51 ERROR_CHECK
52 # 9:1minStar  10:5minStar  11:15minStar
53
54 ######################################################
55 # HTML作成
56 ######################################################
57 # 出力
58 echo "Content-Type: text/html"
59 echo ""
60 cat $htmd/KANSI_LOADAVERAGE.HTML         |
61 calsed "###DATE###" $(dayslash -d --output HH:MM:SS $todayhms) |
62 mojihame -l###REPORT_LOOP### - $tmp-list |
63 cat
64
65 ############################################################
66 # 終了
67 touch $semd/$(basename $0).$HOSTNAME.END.$today
68 rm -rf $tmp-*
69 exit
```

getlast はユニケージコマンド。
同一キーの最後の行を出力します。

❷

ロードアベレージの負荷率が高い
ほど、★の数が多くなります。

❸

mojihame はユニケージコマンド、画面3参照

画面 3　mojihame：文字列を行単位にテンプレートにはめるコマンド

```
$ cat data
sv1 192.168.10.11 23:01 354 0 0.00 0.00 0.00 _ _ _
sv2 192.168.10.12 23:01 354 0 0.01 0.02 0.00 _ _ _

$ cat template
<!-- ###REPORT_LOOP### -->
      <tr>
        <td align="left" >%1</td>
        <td align="left" >%2</td>
        <td align="center">%3</td>
        <td align="right">%4</td>
        <td align="right">%5</td>
        <td align="right">%9 %6</td>
        <td align="right">%10 %7</td>
        <td align="right">%11 %8</td>
      </tr>
<!-- ###REPORT_LOOP### -->

$ cat template | mojihame -l###REPORT_LOOP### - data
      <tr>
        <td align="left" >sv1</td>
        <td align="left" >192.168.10.11</td>
        <td align="center">23:01</td>
        <td align="right">354</td>
        <td align="right">0</td>
        <td align="right"> 0.00</td>
        <td align="right"> 0.00</td>
        <td align="right"> 0.00</td>
      </tr>
      <tr>
        <td align="left" >sv2</td>
        <td align="left" >192.168.10.12</td>
        <td align="center">23:01</td>
        <td align="right">354</td>
        <td align="right">0</td>
        <td align="right"> 0.01</td>
        <td align="right"> 0.02</td>
        <td align="right"> 0.00</td>
      </tr>
```

テンプレートに嵌めるデータが2行あります。

データの各列が %1,%2,%3,$4,・・・%11と記載した位置にセットされます。
データの2列目以降は、ラベル ###REPORT_LOOP### の範囲で繰り返し設定されます。

データが繰り返しテンプレートに設定されています。

コードの見どころ

[1] 複数サーバーのロードアベレージ情報の取得及び保存。（❶リスト1. 39 ～ 59行目まで）
[2] 保存したロードアベレージ情報を画像出力用に加工。（❷リスト2. 41 ～ 51行目まで）
[3] HTML データを作成。（❸リスト2. 58 ～ 63行目まで）

まとめ

並列処理の多用や業務が一時期に集中してアクセスすることで、プロセス数の増加でシステムの遅延が発生したり、ログインができないといった問題が起きる場合があります。こうした問題が発生しても、常に運用監視で定期的にログなどを記録して残しておけば、後から解析を行い、原因究明に役立ちます。