OpenVMS Technical
Journal V11

RMS Collector for T4 and Friends

Gorazd Kikelj, HP Account Support Manager

Overview

This article provides a technique of expanding T4 and Friends with additional data collectors and with minimal impact to the core T4 procedures and utilizing services provided by them. The example described is uses a MONITOR RMS extension to monitor RMS file statistics.

Introduction

When analyzing a customer's system performance problem, it is common for some hot files to arise as key factors in the solution. Customers usually have no historical data on real file performance. Also, it is difficult to estimate from T4 cumulative data how much performance loss is caused by a specific file.

The RMS collector enables T4 to track RMS file-related statistics that are on line with other system performance data and ready for examination with TlViz. Figure 1 shows that, if relevant data is captured, how file performance is tied to system performance and also can be compared with application performance.

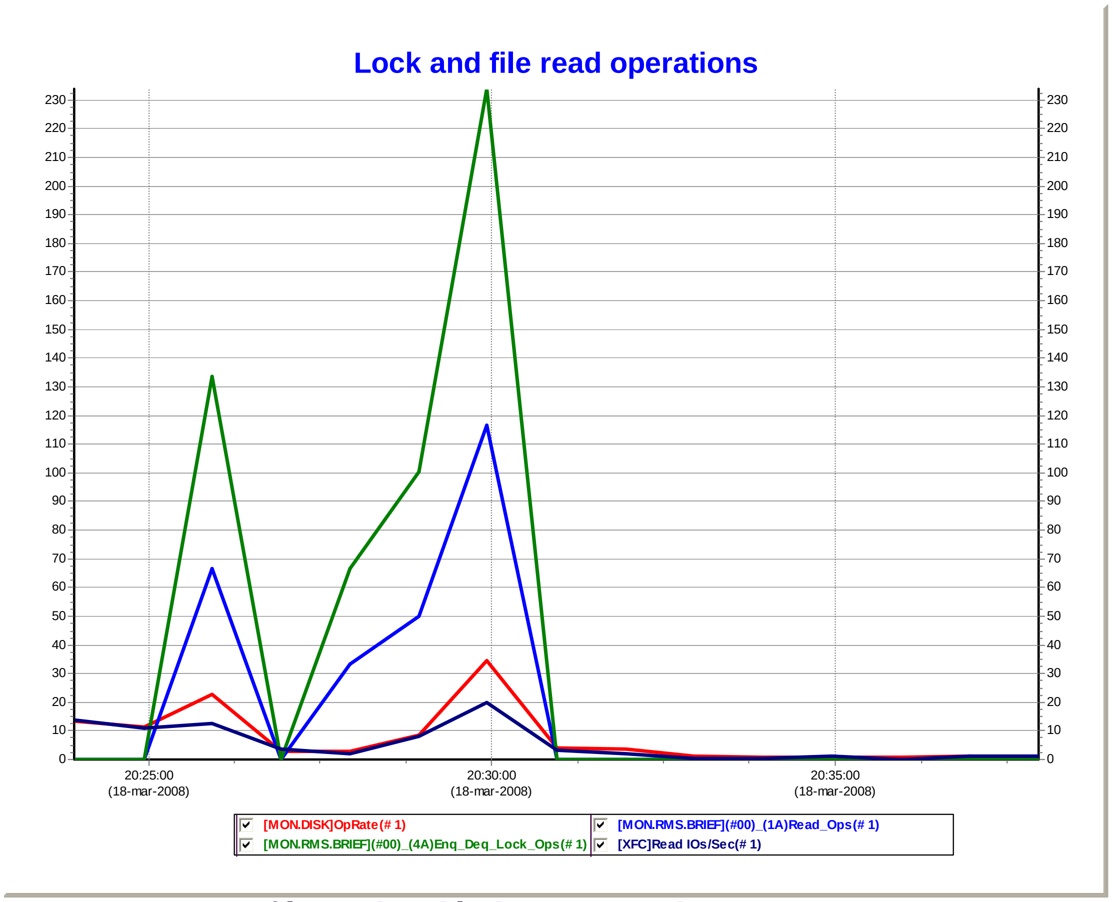

Figure 1 – Comparing file read and lock activity with system I/O operation rate

Trying to add a new collector into the T4 and Friends framework can be time consuming and requires modification of the core T4 command procedures. Also, some reverse engineering is needed to utilize facilities already in the framework. For this article, we used a technique that allows easy addition of new collectors in T4 and Friends framework by utilizing existing T4 data and collection process handling.

Work on this article is heavily based on Pat McConnell's article, "Adding a Friend to T4 and Friends", OpenVMS Technical Journal V4, 2004.

T4 and Friends version V4.2 is used in this article.

Expanding T4$COLLECT.COM to allow new Friends

The main procedure for data collection is the T4$SYS:T4$COLLECT.COM. It provides all necessary functions for dispatching data collection, processing collected performance data, maintaining the file repository, and recovering from errors. It is a complex procedure and is replaced with every new T4 upgrade. Consequently, every modification to the procedure must be reimplemented with any new release of T4 and Friends.

The T4$COLLECT.COM procedure provides many useful functions and symbols that can be used with Friends. The following table lists just some of the available symbols you can use:

Symbol	Description	Format
Today	Date when T4$COLLECT starts	DDMMMYYYY
Start_Time	Date and time when collection starts	DD-MMM-YYYY HH:MM
End_Time	Date and time when collection stops	DD-MMM-YYYY HH:MM
St_Et	Start and end times of data collection	HHMM_HHMM
This_Node	Node name of the T4 main process	Character string
This_Pid	PID of the main T4 process	Character string
Work_Dir	Working directory where data-collection files reside	Directory specification, by default T4$DATA
P6	Collection interval in seconds	Integer

For collected data to be useful, it must be gather in the same time period and at the same time intervals. You can specify these values by the Start_Time, End_Time and P6 symbols in the T4$COLLECT.COM command procedure. Those symbols must be passed to a Friend data-collection procedure and must be used in the collection process.

This article focuses on two sections of the T4$COLLECT.COM procedure that are of interest when adding a Friend: the Start_Data_Collection section and the Post_Process_the_Data section.

The Start_Data_Collection section starts and monitors all data collectors. If you want to add a Friend, add it to this section.

To minimize changes that are required after an upgrade, a user-maintainable procedure is called from this section. The T4$SYSTARTUP procedure includes a set of parameters that enable synchronization of the data-collection schedule and a working directory for data collection. By default, T4$SYSTARTUP.COM resides in the T4$SYS directory or must be pointed to by the logical name T4$SYSTARTUP. Here is an example of the Start_Data_Collection section of the T4$COLLECT.COM procedure.

```
$ Start_Data_Collection:
$! Skipped all lines before Network Monitor starts
$! ...
$!      Start user data collectors – Friends
$       User_Startup = F$Edit( F$Search( F$Parse( “T4$SYSTARTUP”,-
                      ”T4$SYS:.COM”,,,”SYNTAX_ONLY”)),”TRIM” )
$       If “’’User_Startup’” .Nes. “”
$       Then
$        @’User_Startup “’’Start_Time’” “’’End_Time’” “’’This_Pid’”
-
                            “” “’’St_Et’” “’’P6’” “’’Work_Dir’”
$       EndIf
$
$!      Last, but not least spawn a Network Monitor data collection
event
```

The following parameters are passed to the user-maintainable procedure:

P1 – Start date and time for data collection
P2 – End date and time for data collection
P3 – PID of the master T4 process
P4 – Blank
P5 – Start and end times of data collection (used to form a file name)
P6 – Collection interval
P7 – Data-collection working directory, defined in T4$CONFIGURE.COM

In the Post_Process_the_Data section, all data collectors are already stopped and data is available for post-processing. A user-modifiable procedure is added at the end of standard post-processing procedures, right before consolidating data. For example:

```
$ post_Process_the_Data:
$! Skipped all lines before glue the .CSV files with T4$ApRc
$! ...
$!      User post processing data procedures
$       User_Shutdown = F$Edit( F$Search( F$Parse( “T4$SYSHUTDOWN”,-
                       ”T4$SYS:.COM”,,,”SYNTAX_ONLY”)),”TRIM” )
$       If “’’User_Shutdown’” .Nes. “”
$       Then
$        @’User_Shutdown “’’Start_Time’” “’’End_Time’” “’’This_Pid’”
-
                           “” “’’St_Et’” “” “’’Work_Dir’”
$       EndIf
$
$!      Last, but not least, we glue all the .CSV records together
$!      horizontally.  This is done using T4$APRC, which takes two
$!      parameters (P1 and P2), where each record in file P1 is
prefixed
$!      with a comma and then appended to the end of the
```

The following parameters are passed to the user-maintainable procedure are:

P1 – Start date and time for data collection
P2 – End date and time for data collection
P3 – PID of the master T4 process
P4 – Blank
P5 – Start and end time of data collection (used to form a file name)
P6 – Blank
P7 – Data-collection working directory, defined in T4$CONFIGURE.COM

These two changes are the only ones needed for enabling new Friends to be added to T4.

The following is the syntax for log files to be automatically processed by T4$COLLECT.COM procedure:

```
<Work_Dir>T4_<nodename>_<Today>_<St_Et>_SUBP_<collector>.LOG
```

For example:

T4$DATA:T4_DS10L_26JUN2007_1500_1530_SUBP_R001.LOG

User-modifiable procedures

The only function of the T4$SYSTARTUP and T4$SYSHUTDOWN procedures is to call new T4 Friends (collectors) added by the user.

T4$SYSTARTUP example:

```
$       Set Verify ! T4$SYSTART.COM
$       Set Noon
$       Start_Time = P1
$       End_Time   = P2
$       This_Pid   = P3
$       St_Et      = P5
$       Interval   = P6
$       Work_Dir   = P7
$
$!      Call RMS Collector
$       Rms_Collector = F$Edit( F$Search( F$Parse( “T4$RMS_START”,-
                      ”T4$SYS:.COM”,,,”SYNTAX_ONLY”)),”TRIM” )
$       If “’’Rms_Collector’” .Nes. “”
$       Then
$        @’Rms_Collector “’’Start_Time’” “’’End_Time’” “’’This_Pid’”
-
                          “” “’’St_Et’” “’’Interval’” “’’Work_Dir’”
$       EndIf
```

T4$SYSHUTDOWN example:

```
$       Set Verify ! T4$SYSHUTDOWN.COM
$       Set Noon
$       Start_Time = P1
$       End_Time   = P2
$       This_Pid   = P3
$       St_Et      = P5
$       Interval   = P6
$       Work_Dir   = P7
$
$!      Call RMS Collector Postprocessor
$       Rms_Collector_End = F$Edit( F$Search( F$Parse(
“T4$RMS_END”,-
                     ”T4$SYS:.COM”,,,”SYNTAX_ONLY”)),”TRIM” )
$       If “’’Rms_Collector_End’” .Nes. “”
$       Then
$      @’Rms_Collector_End “’’Start_Time’” “’’End_Time’”
“’’This_Pid’” –
                              “” “’’St_Et’” “” “’’Work_Dir’”
```

RMS Collector

The RMS collector consists of three command files and a data file. Monitored files need to be entered into T4$SYS:RMS_FILES.DAT text file with your text editor of choice. Each record represents one monitored file specification. For each file specification, add 1 to the Prclm quota in SYSUAF record for the user account that will be used for the data collection.

T4$SYS:RMS_FILES.DAT example:

```
$ type T4$SYS:RMS_FILES.DAT

DISK$$DATA:[APPLICATION]MASTER_DATA.DAT
DISK$$DATA:[APPLICATION]INDEX_DATA.DAT
DISK$$USER:[HOTSPOT]HOTFILE.IDX
```

The T4$RMS_START.COM command procedure reads the T4$SYS:RMS_FILES.DAT file and spawns a subprocess for each record in a file. For example:

```
$! T4$RMS_START.COM
$! Parameters:
$!      P1 - Begin time
$!      P2 - End time
$!      P3 – Pid
$!      P4 -
$!      P5 - StopTime_EndTime
$!      P6 - Sample interval
$!      P7 - Working Directory
$!
$       Set Verify
$       On Error then goto Exit
$       Set Process/Priv=(All,NoBypass)
$       This_Node = F$GetSyi("NodeName")
$       Today = F$CvTime(P1,"ABSOLUTE","DATE")
$       If (F$Length(Today) .Eq. 10)
$       Then
$             Today = "0" + Today
$       EndIf
$
$       Today = Today - "-" - "-"
$       Work_Dir = P7
$       if ( F$Edit(F$Search(“T4$Sys:Rms_Files.dat”),”TRIM”) .Nes. “”
)
$       Then
$         File_Index = 1
$         Open/Read/Share t4_rms t4$sys:rms_files.dat
$ Loop:
$         Read/End=End_Read/Error=End_Read t4_rms rec
$         Idx = "''F$Fao("R!3XL",File_Index)'"
$         Spawn/NoSymbols/NoWait/Process="T4''P3'_''Idx'" -

/OutPut='Work_Dir'T4_'This_Node'_'Today'_'P5'_SUBP_'Idx'.log -
           @T4$Sys:T4$Rms_Mon "''P1'" "''P2'" "''Idx'" "''rec'"
"''P5'" –
           "''P6'" "''P7'"
$
$         File_Index = File_Index + 1
$         goto loop
```

In addition to the Prclm quota, another limiting factor is file-name syntax. The collector part of the file name (that is, the last part) consists of the letter R for RMS and a 3-byte hexadecimal-encoded running sequence from the RMS_FILES.DAT file. This sequence starts from 1 for the first record and progresses until the end of file. This imposes a limit of 4094 (%xFFF-1) files that can be monitored simultaneously.

The T4$RMS_MON.COM command procedure is the data collector. It uses a MONITOR RMS to monitor a file. Data are recorded into Work_Dir. Monitored files must have RMS statistics enabled via the SET FILE/STATISTIC command. To collect data about the RMS file provided in P4, define the Start_Time (P1) and End_Time (P2) parameters and the Sample Interval (P6) value. The process name and the collection file name use the File Index (P3) parameter to uniquely identify a collection. For example, T40009960_R001 is the process that collects statistics for the first file listed, and T4_DS10L_26JUN2007_1500_1530_R001.DAT is the name of the collection data file.

The following is the syntax for the data file name:

```
<Work_Dir>T4_<nodename>_<Today>_<St_Et>_<collector>.DAT
```

For example:

T4$DATA:T4_DS10L_26JUN2007_1500_1530_R001.DAT

This syntax enables T4$COLLECT.COM to automatically manage DAT files.

```
$! T4$RMS_MON.COM
$! Parameters:
$!      P1 - Begin time
$!      P2 - End time
$!      P3 - File Index
$!      P4 - File name
$!      P5 - StopTime_EndTime
$!      P6 - Sample interval
$!
$       Set Verify
$       Set Noon
$       Set Process/Priv=(All,NoBypass)
$       This_Node = F$GetSyi("NodeName")
$       Today = F$CvTime(P1,"ABSOLUTE","DATE")
$       If (F$Length(Today) .Eq. 10)
$       Then
$             Today = "0" + Today
$       EndIf
$
$       Today = Today - "-" - "-"
$
$       write sys$output "File ''P4'"
$       File_Name =
f$edit(f$search(f$parse(P4,,,"SYNTAX_ONLY")),"TRIM")
$       if "''File_Name'".nes.""
$       then -
          Monitor/Record=T4_'This_Node'_'Today'_'P5'_'P3'.Dat-
          /Interval='P6 -
           /Flush_Interval='P6 -
           /Begin="''P1'" -
           /End="''P2'" -
           /Comment="''This_Node' ''File_Name'" -
```

The T4$RMS_END.COM command procedure is a post-processing step for integrating recorded data into the file-specific comma-separated values (CSV) and composite CSV file. To create a CSV file, the T4$RMS_END.COM procedure utilizes the T4$Mon_Extract utility. For merging a CSV file into a composite CSV file, the procedures utilizes the T4$ApRc utility.

The CSV file syntax is the same as the .DAT file syntax:

```
<Work_Dir>T4_<nodename>_<Today>_<St_Et>_<collector>.CSV
```

For example:

T4$DATA:T4_DS10L_26JUN2007_1500_1530_R001.CSV

This syntax enables the T4$COLLECT.COM procedure to automatically manage CSV files.

```
$! T4$RMS_END.COM
$! Parameters:
$!      P1 - Begin time
$!      P2 - End time
$!      P3 – Pid
$!      P4 -
$!      P5 - StopTime_EndTime
$!      P6 - Sample interval
$!      P7 - Working Directory
$!
$       Set Verify
$       Set NoOn
$       Set Process/Priv=(All,NoBypass)
$       This_Node = F$GetSyi("NodeName")
$       Today = F$CvTime(P1,"ABSOLUTE","DATE")
$       If (F$Length(Today) .Eq. 10)
$       Then
$             Today = "0" + Today
$       EndIf
$
$       Today = Today - "-" - "-"
$       Work_Dir = P7
$       Set Command T4$Sys:T4$Mon_Extract
$       Set Command T4$Sys:T4$ApRc
$
$loop:
$      File_Name = -
       F$Edit(F$Search("T4_''This_Node'_''Today'_''P5'_R%%%.dat",-
              112233),"TRIM")
$       if File_Name .Eqs. "" then goto end_loop
$       File_Csv = F$Parse(File_Name,,,"NAME","SYNTAX_ONLY")
$       T4Extr 'File_Name' /Csv_File='File_Csv'.Csv –
                           /Class=(NoAll,RMS) –
                           /Format=Extended
$       T4Aprc 'Work_Dir''File_Csv'.Csv; -
               T4_'This_Node'_'Today'_'P5'_Comp.csv;
$       goto loop
```

The original CSV files are retained and combined with the composite CSV file in the archive ZIP file that T4$COLLECT provides at the end of the processing. This was done deliberately because there is no other way to find out what file was monitored once the RMS_FILES.DAT is changed.

Summary

This article shows that it is easy to add a Friend to T4 and Friends. This approach adds a user entry point to the T4 core execution process. Adding a simple, additional collector that utilizes existing utilities in OpenVMS and T4 is demonstrated.

For more information

- The following article is the standard reference for understanding the Timeline Collaboration concept, which has driven the development of T4 and Friends. This article provides the background necessary for the development of extensions to the default T4 implementation.

  "Timeline-Driven Collaboration with T4 and Friends: A Timesaving Approach to OpenVMS Performance," OpenVMS Technical Journal, Volume 3, February 2004.

- The following article describes how to integrate a Friend into T4:

  “Adding a Friend to T4 and Friends Incorporating BEA WebLogic Server 8.1 Performance Data, OpenVMS Technical Journal, Volume 4, June 2004.