

^ZSY: System Status for GT.M/YottaDB

Sam Habel, Pharm.D.

December 30, 2019

Abstract

This paper describes the updates and new functionality in the System Status tools for GT.M/YottaDB.

1 Introduction

The ZSY Routine of unknown provenance – was in unreleased VA patch XU*8.0*349 and thus perhaps in the public domain. Its original purpose was to provide an emulation of %SS in Intersystems Cache. It has been almost entirely re-written and much new functionality has been introduced. While developed with VistA/RPMS in mind, it can be used on any GT.M/YottaDB system with a \$ZINTERRUPT of “I \$\$JOBEXAM^ZSY(\$ZPOS)”

2 Invocation

Enter D ^ZSY on direct mode. This by default sorts by Process ID. You can use D QUERY^ZSY to query by cpu time.

3 Output

The output changes based on the size of the screen: ≤ 80 columns; 80 columns–130 columns; and >130 columns.

```
GT.M System Status users on 24-APR-18 16:03:06 - (stats reflect accessing
DEFAULT region ONLY except *)
PID   PName   Device   Routine           Name                CPU Time
4257   mumps   /dev/pts/1 INTRPTALL+8^ZSY
7369   mumps   BG-0      NEWJOB+5^%ZTM      Taskman ROU 1       00:00:00
7373   mumps   BG-0      G0+26^XMKPLQ       WVEHR,PATCH INSTALL 00:00:00
7391   mumps   BG-0      G0+12^XMTDT        WVEHR,PATCH INSTALL 00:00:00
7397   mumps   BG-0      GETTASK+3^%ZTMS1
7403   mumps   BG-0      GETTASK+3^%ZTMS1   Sub 7403             00:00:00
7409   mumps   BG-0      I6+2^%ZTM
7411   mumps   BG-0      GETTASK+3^%ZTMS1   Sub 7411             00:00:00
```

Listing 1: 80 column output

GT.M System Status users on 24-APR-18 16:35:06 - (stats reflect accessing DEFAULT region ONLY except *)											
PID	PName	Device	Routine	Name	CPU Time	OP/READ	NTR/NTW	NR0123	#L	%LSUCC	%CFAIL
4257	mumps	/dev/pts/1	INTRPTALL+8^ZSY		00:00:00	1.14	10/0	0/0/0/0	1	0/0	0/7
7369	mumps	BG-0	IDLE+3^%ZTM	Taskman ROU 1	00:00:00	493.05	10k/0k	1/0/0/0	0	99.95%	0/426
7373	mumps	BG-0	GO+26^XMKPLQ	WVEHR,PATCH INSTALL	00:00:00	90.14	1905/12	0/0/0/0	2	9/14	0/33
7391	mumps	BG-0	GO+12^XMTDT	WVEHR,PATCH INSTALL	00:00:00	59.73	669/12	0/0/0/0	2	9/9	0/23
7397	mumps	BG-0	GETTASK+3^%ZTMS1		00:00:00	784.04	19k/2k	1/0/0/0	1	99.96%	0/1977
7403	mumps	BG-0	GETTASK+3^%ZTMS1	Sub 7403	00:00:00	0.00	9766/968	3/0/0/0	1	99.35%	0/973
7411	mumps	BG-0	GETTASK+3^%ZTMS1	Sub 7411	00:00:00	0.00	9738/965	3/1/0/0	1	99.53%	0/970
7419	mumps	BG-0	GETTASK+3^%ZTMS1	Sub 7419	00:00:00	0.00	9717/963	7/3/0/0	1	99.02%	0/967
7422	mumps	BG-0	GETTASK+3^%ZTMS1	Sub 7422	00:00:00	0.00	9678/959	13/0/0/0	1	98.25%	0/970

Listing 2: 130 column output

GT.M System Status users on 24-APR-18 17:05:25 - (stats reflect accessing DEFAULT region ONLY except *)														
PID	PName	Device	Routine	Name	CPU Time	OP/READ	NTR/NTW	NR0123	#L	%LSUCC	%CFAIL	R MB*	W MB*	SP MB*
4257	mumps	/dev/pts/1	INTRPTALL+8^ZSY		00:00:00	1.14	10/0	0/0/0/0	1	0/0	0/7	5.66	12.98	0.10
7369	mumps	BG-0	IDLE+3^%ZTM	Taskman ROU 1	00:00:00	955.95	20k/1k	3/0/0/0	0	99.80%	0/721	2.77	0.68	0.10
7373	mumps	BG-0	GO+26^XMKPLQ	WVEHR,PATCH INSTALL	00:00:00	166.86	3516/12	0/0/0/0	2	9/14	0/33	3.21	0.03	0.10
7391	mumps	BG-0	GO+12^XMTDT	WVEHR,PATCH INSTALL	00:00:00	100.64	1119/12	0/0/0/0	2	9/9	0/23	0.23	0.00	0.10
7397	mumps	BG-0	GETTASK+3^%ZTMS1		00:00:01	1505.24	37k/4k	2/0/0/0	1	99.96%	81.36%	0.29	2.51	0.10
7403	mumps	BG-0	GETTASK+3^%ZTMS1	Sub 7403	00:00:00	0.00	18k/2k	8/0/0/0	1	99.47%	89.76%	0.00	0.77	0.10
7411	mumps	BG-0	GETTASK+3^%ZTMS1	Sub 7411	00:00:00	0.00	18k/2k	3/1/0/0	1	99.67%	89.79%	0.00	0.73	0.10
7419	mumps	BG-0	GETTASK+3^%ZTMS1	Sub 7419	00:00:00	0.00	18k/2k	19/4/0/0	1	99.09%	89.80%	0.00	0.79	0.10
7422	mumps	BG-0	GETTASK+3^%ZTMS1	Sub 7422	00:00:00	0.00	18k/2k	16/0/0/0	1	98.50%	89.78%	0.00	0.41	0.10

Listing 3: >130 column output

Here is by contrast the Cache system status:

Cache System Status: 4:00 pm 02 May 2018					
Process	Device	Namespace	Routine	CPU,Glob	Pr User/Location
92	/dev/null	PLA	XMKPLQ	8415,815	0 %System
61	TCP 9430	PLA	%ZISTCPS	2805,187	0 %System
66	/dev/null	PLA	%ZTM	9180,1078	0 %System
68	TCP 8001	PLA	XOBVTCPL	5355,416	0 %System
93	/dev/null	PLA	XMTDT	6885,380	0 %System
94	/dev/null	PLA	%ZTMS1	18615,2779	0 %System

6 user, 0 system, 256 mb global/23 mb routine cache

Listing 4: Cache System Status

4 Explanation of Columns

4.1 PID

This is the Process Number of each process. On GT.M/YottaDB, the process number is the same as job number. This unfortunately is not the same as the Taskman task number.

4.2 PName

Process Name. Most of the time it will be “mumps”, but it can be “java”, “node” or some other process that is opening the database.

4.3 Device

Consider this listing:

```
Device
BG-0
BG-0
BG-0
BG-0
BG-S14823
/dev/pts/22
BG-0
BG-/dev/null
/dev/null
```

Listing 5: Device

This shows the currently active device.

The most obvious thing is the “BG”. This means that the job is a background job (not launched by a user, but launched originally by a Job command). However, even background jobs can interact with users: The last BG device in this listing is actually a CPRS session. The process was created using the Job command.

User “Roll and Scroll” sessions will show as /dev/pty/...

Devices with socket listeners will show as BG-Sport, where port is a numeric port number.

BG-0 devices are usually background jobs without devices in Taskman.

Broker jobs originating from xinetd will show a device of /dev/null unless a device is opened.

4.4 Routine

This shows the currently executing line.

4.5 Name

This is either the name of the user from File 200 if the process has a DUZ variable defined; or process name if defined in ^XUTL, or nothing otherwise.

4.6 CPU Time

Total CPU time (User + System) as reported by ps.

4.7 OP/READ

Operations/Read: Number of global operations per database block read obtained via the following formula:

$$\frac{DTA + GET + ORD + ZPR + QRY}{DRD}$$

DRD is the number of database reads; DTA, GET, ORD, ZPR and QRY are the number of operations on globals of \$DATA, \$GET, \$ORDER, reverse \$ORDER, and \$QUERY (forward and reverse) respectively. This statistic is only for the DEFAULT region of the database.

4.8 NTR/NTW

This is the number of non-transaction reads and non-transactions writes into the DEFAULT region of the database. If NTR > 9999, the values will be divided by 1024 and displayed in kilos. This statistic is only for the DEFAULT region of the database.

4.9 NR0123

This is the number of # of Non-TP transaction Restarts at try 0, 1, 2, and 3. A large number of retries at 2 or 3 is a bad sign. This statistic is only for the DEFAULT region of the database.

4.10 #L

Number of locks held by the process.

4.11 %LSUCC

Lock success percentage calculated by:

$$\frac{LKS}{LKS + LKF}$$

LKS is the number of successful locks; and LKF is the number of unsuccessful locks. If LKS + LKF is < 100, then it will be shown as a fraction; otherwise it will be shown as a percentage with 2 decimal points. This statistic is only for the DEFAULT region of the database.

4.12 %CFAIL

Critical section acquisition failure percentage calculated by:

$$\frac{CFT}{CFT + CAT}$$

CFT is the total of blocked critical section acquisitions; CAT is the total of critical section acquisitions successes. Like locks, if $CFT + CAT < 100$, then it will be shown as a fraction; otherwise, it will be shown as a percentage with 2 decimal points. This statistic is only for the DEFAULT region of the database.

This number is hard to use in isolation. It is only useful as a trend point. My system shows >80% failure rate for taskman processes in a tight loop.

4.13 R MB

Data read from disk in megabytes, as reported by the OS.

4.14 W MB

Data written to disk in megabytes, as reported by the OS.

4.15 SP MB

This is how large the string pool is. This is a good proxy for how much heap memory is being used by GT.M/YDB for the symbol table. This is obtained using `$view("spsize")`.

5 Job Examination Utility

A job examination utility is provided in order to assist with troubleshooting specific jobs. It can be invoked in two ways: either run `D ZJOB^ZSY`, and then select a job number from the list; or run `D ZJOB^ZSY(job_number)` to interrogate a specific job number. You can also use one of these alternate entry points:

- `D EXAMJOB^ZSY[(job_number)]`
- `D VIEWJOB^ZSY[(job_number)]`
- `D JOBVVIEW^ZSY[(job_number)]`

Pressing enter on the screens that are displayed updates them. If you want to exit, type `^`. Following is the screen obtained from `D ZJOB^ZSY`:

```
GT.M System Status users on 02-MAY-18 15:16:38
PID   PName   Device      Routine      Name          CPU Time
11422  mumps    BG-0        IDLE+3^%ZTM   Taskman ICARUS 1 00:00:03
11481  mumps    BG-0        G0+26^XMKPLQ  POSTMASTER     00:00:00
11483  mumps    BG-0        G0+12^XMTDT   POSTMASTER     00:00:00
11489  mumps    BG-S14823   LGTM+25^%ZISTCPS POSTMASTER     00:00:01
13063  mumps    /dev/pts/21 INTRPTALL+8^ZSY USER,ONE       00:03:01
15287  mumps    BG-0        GETTASK+3^%ZTMS1 00:00:00

Total 6 users.

Enter a job number to examine (^ to quit):
```

Listing 6: `D ZJOB^ZSY`

Selecting a job number, or invoking it directly using `D ZJOB^ZSY(PID)` will let you to the follow screen:

```

JOB INFORMATION FOR 11483 (2018-MAY-02 15:26:57)
AT: GO+12^XMTDT: . . H XMHANG

Stack:
1. SUBMGR+3^%ZTMS1          D PROCESS^%ZTMS2 G:$D(ZTQUIT) QUIT^%ZTMS
2. PROCESS+9^%ZTMS2         D TASK^%ZTMS3 I ZTYPE="C"!$D(ZTNONEXT) Q
3. 4+10^%ZTMS3              D RUN
4. RUN+4^%ZTMS3             D @ZTRTN
5. GO+5^XMTDT               F Q:$P($G(^XMB(1,1,0)),U,16) D
6. GO+11^XMTDT              . F D Q:$TSTAMP^XMXUTIL1-XMWAIT>60
7. GO+12^XMTDT:             . . H XMHANG

Locks:
LOCK ^XMBPOST("POST_Tickler") LEVEL=1
LOCK ^%ZTSCH("TASK",3479) LEVEL=1

Devices:
0 OPEN RMS STREAM NOWRAP
0-out OPEN RMS STREAM NOWRAP

Breakpoints:

Global Stats for default region:
BTD: 0          CAT: 26          CFE: 0          CFS: 16k          CFT: 129          CQS: 0
CQT: 0          CTN: 188m        CYS: 1          CYT: 1          DEX: 0          DFL: 0
DFS: 0          DRD: 13          DTA: 30         DWT: 14         GET: 2262        JBB: 1568
JEX: 0          JFB: 0          JFL: 0          JFS: 0          JFW: 0          JRE: 0
JRI: 0          JRL: 12          JRO: 1          JRP: 0          KIL: 4          LKF: 2
LKS: 9          NBR: 10k         NBW: 11         NR0: 0          NR1: 0          NR2: 0
NR3: 0          NTR: 3985        NTW: 12         ORD: 1677       QRY: 0          REG: DEFAULT
SET: 10         TBR: 0          TBW: 0          TC0: 0          TC1: 0          TC2: 0
TC3: 0          TC4: 0          TRO: 0          TR1: 0          TR2: 0          TR3: 0
TR4: 0          TRB: 0          TTR: 0          TTW: 0          ZPR: 4          ZTR: 0

String Pool (size,currently used,freed): 102056,5684,0

Enter to Refersh, V for variables, I for ISVs, K to kill
L to load variables into your ST and quit, ^ to go back:
D to debug (broken), Z to zshow all data for debugging.

```

Listing 7: Job Examination Screen

Pressing enter on this page refreshes the information. Pressing enter frequently will give you a good view of what's happening in an execution.

The sections should be self-explanatory.

Pressing V or I will show you variables and Intrinsic Special Variables(ISV). They are just printed in a vertical list so that they are easy to copy and paste.

```

%=1
%ZPOS="GO+12^XMTDT"
%ZTIME=5596168159
%ZTPFLG("Ballimit")=100
%ZTPFLG("HOME")="ICARUS:foia.2018.02"
%ZTPFLG("LOCKTM")=3
%ZTPFLG("MIN")=1
%ZTPFLG("RT")=0
%ZTPFLG("USER")=18
%ZTPFLG("XUSCNT")=59
%ZTPFLG("ZTPN")=1
%ZTPFLG("ZTREQ")=1
%reference="^XMB(3.9,""AF"",0)"
DILOCKTM=3
DT=3180502
DTIME=1
DUZ=.5
DUZ(0)="@" (etc...)

```

Listing 8: Variable List

```
ISVs :
$DEVICE=" "
$ECODE=" "
$ESTACK=9
$ETRAP="D ERROR^%ZTMS HALT"
$HOROLOG="64770,56506"
$IO=0
$JOB=11483
$KEY=" "
$PRINCIPAL=0
$QUIT=1
$REFERENCE="^XUTL("XUSYS",11483,"JE","I",10)"
$STACK=9
$STORAGE=2147483647
$SYSTEM="47,foia.2018.02"
$TEST=1
$TLEVEL=0 (etc...)
```

Listing 9: ISV List

K will kill the process (actually, it sends it to HALT^ZU after cleaning it up). ^ will take you back; D (debug) is currently broken; and Z is there for the developer's use.

6 Interactive Kill Job (a.k.a RESJOB)

A Utility to interactively kill jobs is provided using the entry point D KILLJOB^ZSY. Select a job number from the list, and it will be sent to HALT^ZU after lock and globals cleanup. You can alternately use D RESJOB^ZSY. This entry point is used by ^%ZOSF(“RESJOB”).

```
GT.M System Status users on 30-DEC-19 14:45:28
PID   PName   Device      Routine          Name              CPU Time
2028   mumps    BG-0        IDLE+3^%ZTM      Taskman ROU 1     00:00:00
2093   mumps    BG-0        GETTASK+3^%ZTMS1 Sub 2093           00:00:00
2187   mumps    /dev/pts/0  INTRPTALL+8^ZSY              00:00:00

Total 3 users.

Enter a job number to kill (^ to quit):
```

Listing 10: D KILLJOB^ZSY

7 Other Entry Points

`^ZSY` includes various other entry points that can be used by developers and system managers. Here they are.

7.1 D TMMGR`^ZSY` [Public]

List Taskman Manager processes only. Output:

```
GT.M System Status users on 02-MAY-18 16:01:02
PID PName Device Routine Name CPU Time
11422 mumps BG-0 IDLE+3^%ZTM Taskman ICARUS 1 00:00:04
```

Listing 11: TMMGR Entry Point

7.2 D TMSUB`^ZSY` [Public]

List Taskman Submanager processes, including those currently “otherwise” engaged.

```
GT.M System Status users on 02-MAY-18 16:03:46
PID PName Device Routine Name CPU Time
11481 mumps BG-0 G0+26^XMKPLQ POSTMASTER 00:00:00
11483 mumps BG-0 G0+12^XMTDT POSTMASTER 00:00:00
11489 mumps BG-S14823 LGTM+25^%ZISTCPS POSTMASTER 00:00:01
15287 mumps BG-0 GETTASK+3^%ZTMS1 00:00:02
```

Listing 12: TMSUB Entry Point

7.3 \$\$UNIXLSOF`^ZSY(.procs)` [Kernel Use Only]

This gives you a listing of all the processes accessing the DEFAULT region of the database. The extrinsic output is the number of processes; while `.procs` contains an M array of the process numbers.

This API should be used by Kernel level applications only.

```
>W $$UNIXLSOF^ZSY(.zzz)
6
>zwrite zzz
zzz(1)=11422
zzz(2)=11481
zzz(3)=11483
zzz(4)=11489
zzz(5)=13063
zzz(6)=15287
```

Listing 13: UNIXLSOF Entry Point

7.4 D INTRPT`^ZSY(PID)` [Kernel Use Only]

Send a GT.M Interrupt to a process specified by its PID.

7.5 D INTRPTALL`^ZSY[(.procs)]` [Kernel Use Only]

Use the `$$UNIXLSOF^ZSY(.procs)` API to find all processes accessing the DEFAULT region and interrupt all of them. You can optionally pass in `.procs` to get a list of all the PIDs that got interrupted.

7.6 D HALTALL`^ZSY` [Kernel Use Only]

“Softly” (`^XUSCLEAN`, `HALT^ZU`) kill all processes.

7.7 D HALTONE`^ZSY(PID)` [Kernel Use Only]

“Softly” kill a single process specified by PID