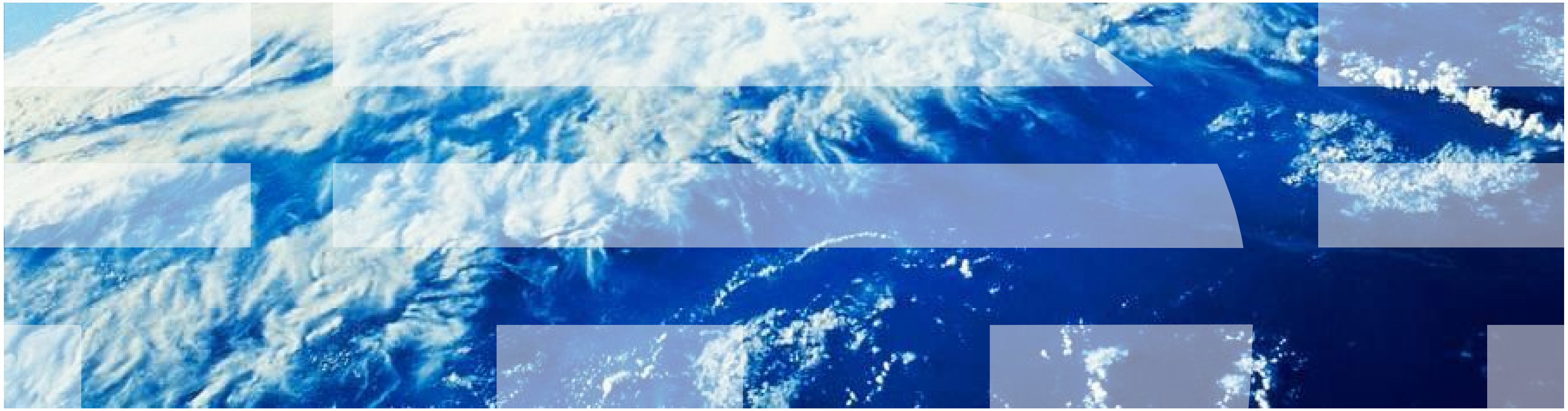
TXSeries performance issues analysis method-part I

TXSeries L2 Zeng Yu



Agenda

l *Background*

l *Performance Analysis Method*

  Ø*Analysis Method*

  Ø*Make the Conclusion*

l *Analysis Method Metrics*

  Ø*Logs Analysis*

  Ø*Sample stand-alone Program*

l *Summary*

Background

Ø***Customer's expecting***

Customer want IBM help check the issue if it related to TXSeries.

Ø***Our thoughts***

We TXSeries need check where problem is first(OS,DB,TX,APP etc).

Ø***TXSeries production ENV very complex***

Maybe related to OS/Database/Application/Network/Compiler/TXSeries Server/TXSeries Client(CTG) etc…

Performance Analysis Method

Analysis Method

ØProblem

Usually, from CTG client side, user found transactions response is slow

ØEnvironment information

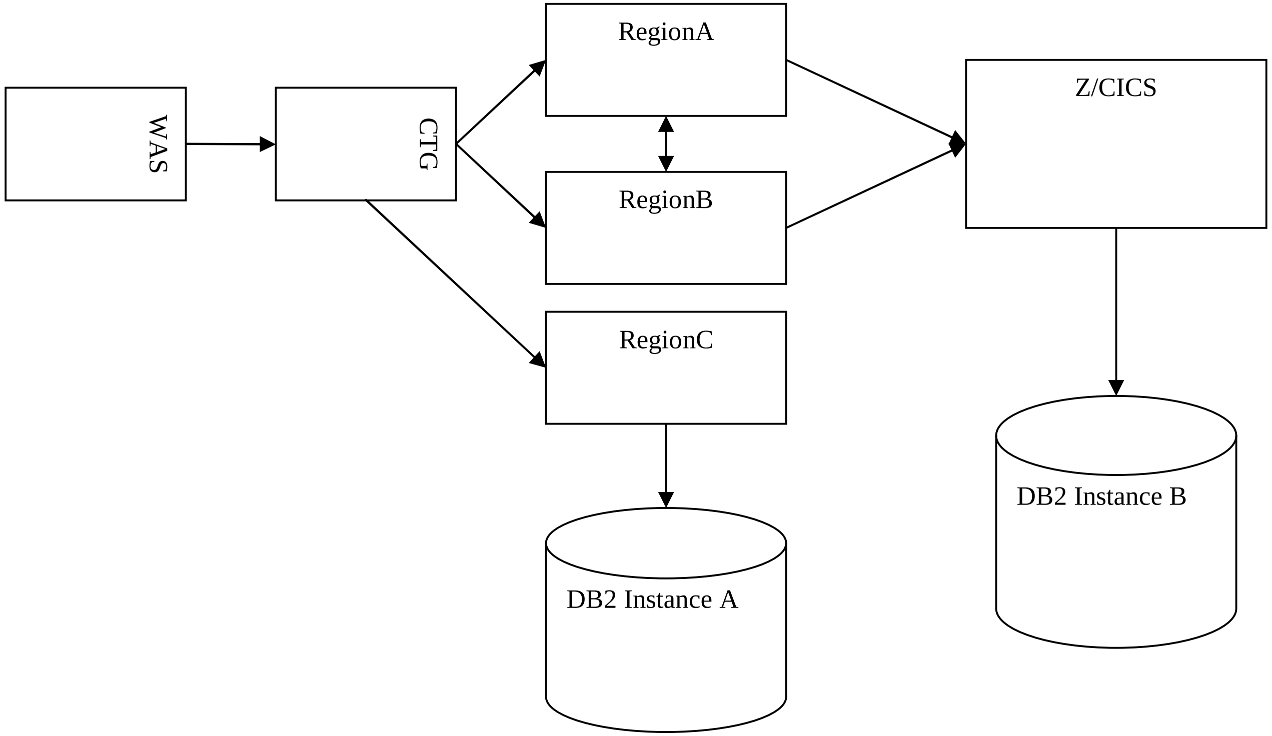
TXSeries Version 7.1 PTF3
AIX 6.1

ØData workflow

WAS(user’s application programs)->CTG ->TXSeries->Z/CICS->Database.

Performance Analysis Method

Analysis Method



Analysis Method Case A---before the checking

ØReaders should to confirm

whether it is a real 'transaction slow' issue first.

ØUnderstand the expected value

Reader needs to compare the real response time and the expected response time .

ØThe performance versus similar system

In TXSeries side, if reader can get another similar system test data, they can compare it with current data.

Case A

Case A

From CTG client side, user found the transaction response is slow.

In the CTG trace, we can find the transaction response time is 405ms ***(16:35:44:295 - 16:35:44:890)*** *¸ but in non-issue time, the response time is* ***about*** *20ms. Let’s check it step by step.*

1. The CTG ***client,*** *conversation ID ConvId is 51927, sent data at 16:35:43.890 and got response data at 16:35:44.295.* ***The whole process takes*** *405 ms.*

The hexadecimal number 0xCAD7=(decimal)51927.

Case A

Picture 2

CTG trace file shows:

```
16:35:43.890 [00050284,00000211] [4410] DRV:CCL4411T TCP/IP (to
<RegionA>) send data: Length=732 ConvId=51927
2005d90e  000002DC0000CAD7 000B90A0270502FF
…
16:35:44.295 [00050284,00000312] [4418] DRV:CCL4412T TCP/IP (to
<RegionA>) receive data, DrvRecv: Length=12
2002beb8  000008540000CAD7 00039001          ...T........      .......P....
TCP data interpretation : TCP header for 00000854 bytes of Data, session CAD7
```

Case A

2. (Since from) Based on the CTG trace, we found the response time at CTG side is 405ms, more then 20ms, then we need check the TXSeries trace files.

Cicsip process got the request at 16:35:44.89616611, and the convid is 51927. TXSeries task number 14235 was assigned for the request.

Case A

```
Picture 3
585     CICS Mon Jan  9 16:35:44 2012 89616611          ABIS01   cicsip          422594/14     (20202020)
96C4009C79C223A    { ComSU_XPGetBuf                  (732)
…
581     CICS Mon Jan  9 16:35:44 89815228    ABIS01   cicsip          422594/14     (20202020)
96C4009C79DAF6F    TCP/IP Listener (ComIP)
cicsip using:
                              Listener       = 422594
                              Connection     = 8
                              Session        = 51927
581     CICS Mon Jan  9 16:35:44 89821916    ABIS01   cicsip          422594/14     (20202020)
96C4009C79DBCC7    TCP/IP Listener (ComIP)
cicsip inbound datastream
000002DC  0000CAD7
585     CICS Mon Jan  9 16:35:44 2012 89934982          ABIS01   cicsip          422594/14     (20202020)
96C4009C79E9EF1   { ConTS_ScheduleWork           (TranId=CPMI, UserId=0, DeviceId=0, Sync=0,
Wait=0, InData=a031c930, Details=0, Fold=0, TClassTran = CPMI)
581     CICS Mon Jan  9 16:35:44 2012 90029273          ABIS01   cicsip          422594/14     (20202020)
96C4009C79F5B84    Scheduler component (ConTS):
Task number 14235 assigned to transaction CPMI
586     CICS Mon Jan  9 16:35:44 2012 90265687          ABIS01   cicsip          422594/14     (20202020)
96C4009C7A1344A   } ConTS_ScheduleWork           return(0, RSN_6)
```

Case A

3. Cicsip process send datastream back

Picture 4

```
<RegionA>.cicsip.n10002.p422594.t11.cicstrc.fmt
581     CICS Mon Jan  9 16:35:44 2012 135205996     ABIS01   cicsip     422594/11
                (20202020)  96C4009C9004CF0      TCP/IP Listener (ComIP)
cicsip using:
                          Listener       = 422594
                          Connection     = 8
                          Session        = 51927
581     CICS Mon Jan  9 16:35:44 2012 135212351     ABIS01   cicsip     422594/11
                (20202020)  96C4009C90059A5      TCP/IP Listener (ComIP)
cicsip outbound datastream
0000 00000854 0000CAD7 00039001
```

Case A

4. So we found TXSeries response quickly, but ,it took long time for CTG to get it. It’s a very interesting phenomena - we know the issue neither in CTG nor in TXSeries.

5. Next step, let's checked the network side. In the case, IP trace logs told us that network didn’t send the data immediate after CTG send data to network side. After change TCP/IP parameter from tcp_nodelayack=0 to tcp_nodelayack=1, the issue was resolved.

The tcp_nodelayack tunable:
The tcp_nodelayack option prompts TCP to send an immediate acknowledgement, rather than the usual 200 ms delay. Sending an immediate acknowledgement might add a little more overhead, but in some cases, greatly improves performance.

Case B

Case B

'transaction slow' case, which seem to occur on region side. In the case

Case B

1. The transaction started at10:10:47.489781869 and finished at 10:10:49.505439646.

Picture 5

Cicsas trace file:

```
585    CICS Wed Jul 11 10:10:47 2012 489781869    TESTZY         cicsas
            1069096/1          CPMI(43504D49)              98908B5E761ECBE
{ TasTA_Run                          (Tranid <CPMI> (49) for user <> on device
<>.... TermD <0>, Indata <0xa00f2d40>)
…
586    CICS Wed Jul 11 10:10:49 2012 505439646    TESTZY         cicsas
            1069096/1          CPMI(43504D49)      98908B624E54054          }
TasTA_Run                           return(0)
```

Case B

2. After checked functions ***between the beginning and the end of the task*** (function TasTA_Run start and finish), there was no TXSeries trace logs found between 10:10:47.497521257 and 10:10:49.497607736.

Picture 6

Cicsas trace file:

```
581     CICS Wed Jul 11 10:10:47 2012 496688892          TESTZY   cicsas               1069096/1      CPMI(43504D49)
98908B5E797E2C3
        Program Control Module (TasPR)

                                                                LINKing Program: TEST1 in Task: 49
585     CICS Wed Jul 11 10:10:47 2012 496694796          TESTZY   cicsas               1069096/1      CPMI(43504D49)
98908B5E797EE99         { TasPR_CallApplication         (void)
…
585     CICS Wed Jul 11 10:10:47 2012 497515648          TESTZY   cicsas               1069096/1      CPMI(43504D49)
98908B5E79E584D          { SupTM_GetThreadCancelData      (void)
586     CICS Wed Jul 11 10:10:47 2012 497521257          TESTZY   cicsas               1069096/1      CPMI(43504D49)
98908B5E79E637D          } SupTM_GetThreadCancelData       return(0x302c1028)
585     CICS Wed Jul 11 10:10:49 2012 497607736          TESTZY   cicsas               1069096/1      CPMI(43504D49)
98908B624A8108A          { SupTM_GetThreadCancelData      (void)
…
586     CICS Wed Jul 11 10:10:49 2012 498334378          TESTZY   cicsas               1069096/1      CPMI(43504D49)
98908B624ADBDBB          } TasPR_CallApplication          return(0, RSN_0)
```

Case B

3. Reference the step2, TXSeries didn’t consume time, we need to check the application log for further tracking.

Picture 7

Application log:

```
1ST ASKTIME date:120711, time:111047
sleep
2ND ASKTIME date:120711, time:111049
```

Case B

4. Check the application program code, and found the root cause as below.

Picture 8

Application program code:

```
   fprintf(stderr,"1ST ASKTIME. date:%s, time:%s\n",wk_c8_date,wk_c6_time);
   sleep(2);
fprintf(stderr,"sleep\n");
…
       fprintf(stderr,"2ND ASKTIME. date:%s,
       time:%s\n",wk_c8_date,wk_c6_time);
```

Make the Conclusion

Case A,

Finally, we know that the issue due to delay of TCP/IP setting. But, users usually just check TXSeries and CTG side, and get confused *after* they can't find cause in TXSeries and CTG.

Here are some keys:

Check the trace step by step followed the data stream from different products

Clearly understand the time cost in different part of the system

Remember that TXSeries is usually used in very complex circumstance.

To track this kinds of issues, it is not enough to check TXSeries regarding logs only.

Make the Conclusion

Case B,

After checked the application program, sleep() in application program was found as the root cause. User need resolve the problem from application program side.

Here are some keys:

Check detailed TXSeries trace timestamp to make sure the main time cost.

If it looks fine in TXSeries trace, go on to check the application program log/trace.

If application program trace looks fine, check the OS or Database side trace files (suc as the output of “errpt -a” on AIX)

Finally, if any TXSeries internal functions issue was found, please contact IBM TXSeries support

Analysis Method Metrics-Logs Analysis

1. At OS /system level, usually we can get more info to check the transaction run slow issue.

Libraries usage statistics “tprof -ktuesjx sleep 60”

Truss output of run slow cicsas process

(“truss -d -f -a -rall -wall -o /tmp/truss_cicsas.log -p process_id”)

Errpt,

env setting

etc…

Analysis Method Metrics-Logs Analysis

2. In TXSeries side, we need to collect the following log/track to check Console, symrecs, CSMT, CCIN,<region_name>.env,<region_name>/database directory and <region_name>/dumps/dir1 directory.

Sometimes, we need trace, for more info, please check mustgather link http://www-01.ibm.com/support/docview.wss?uid=swg21229652#listingMustGather

Analysis Method Metrics-Logs Analysis

3. In application side (database side), usually we need.
The timestamp of application begin/stop time
The timestamp of application main functions to check issue inside application

4. In Network side, usually we need check.
The TCP/IP regarding trace files for CTG/TXSeries communication.

Sample stand-alone Program

Sometimes, a stand-alone program is useful for us to perform a simple performance testing.

Compare the property and action between stand-alone program and similar program running on TXSeries. And we will got more clues..

Summary

1. While encountering a performance issue, user usually only track it from TXSeries perspective. This article introduces the relevant factors, which need to be considered from the integrated environments.

2. This article presents some experience and knowledge, such as Divide the issue among OS system, TXSeries, Application programs and database, to help user to improve division skills. It is helpful for them to make sure which area should be considered .

3. This article introduces how to use stand-alone programs in depth, to help readers to self assist the encountered issue while using TXSeries.

Q & A