Using MySQL

Version 2.1.4715.4

January 31, 2014

MyARM GmbH
Altkönigstraße 7
65830 Kriftel
Germany

Web: http://www.myarm.com/
Mail: info@myarm.com

Contents

1 Introduction

This document describes how to configure MyARM for using MySQL database with high transaction measurement numbers. For general information regarding MyARM or MySQL please consult the appropriate user manuals.

2 Motivation

MyARM is designed to measure and store (unlike other performance monitoring tools) all measured transactions of an ARM instrumented application into a connected database. This can result in high number of measurements which need to be stored into the database. This document describes how to configure MyARM using the MySQL database to achive an optimal transaction `INSERT` throughput.

3 Scenarios

The ARM standard defines a variety number of optional features which influence the throughput of storing transactions into a database we need to define typical classes of scenarios. The following list defines the most common (items 1 to 5) used and worth case (items 6 and 7) ARM transaction measurements:

1. Simple transaction measurement without any additional data (Simple).
2. Simple transaction measurement with a parent correlator (ParentCorr).
3. Transaction measurement with 3 gauge metrics attached (Metrics).
4. Transaction measurement with 1 context value string with a size of 32 Bytes (Context 1).
5. Transaction measurement with 5 context value string with a size of 32 Bytes (Context 5).
6. Transaction measurement with 10 context value string with a size of 32 Bytes (Context 10).
7. Transaction measurement with 20 context value string with a size of 32 Bytes (Context 20).

Some of these scenarios are likely to be combined. For example it is quite usual that a transaction measurement as correlators as well as metrics and context properties. The tested scenarios should help to get an overview how additional data influence the overall database storing transaction throughput.

4 Test design

The following test design is used to get the transaction storing rate into a MySQL database. A simple C program is used to generate ARM transactions at a maximum rate. MyARM is configured to buffer all transaction measurements and write the buffered transactions into the MySQL database. When all transactions are written into the MySQL database the application terminates.

The elapsed real time devided by the number of transactions stored in the MySQL database gives an good overview of the maximal transaction throughput into the MySQL database.

To increase the maximal transaction throughput MyARM supports a so-called `thread` datasink which uses a defined number of threads to write data to a destination, here MySQL, datasink. Each thread open its own connection to the MySQL database. All test scenarios described above are executed with two, four or six datasink threads.

The one thread scenario is executed without the `thread` datasink and the MyARM ARM agent is configured to use the `mysql` datasink directly.

4.1 Hardware

All tests were executed on an Intel Core2 Quad Q9550 (2.0/2.83 GHz) running Debian Linux 5.0 (amd64) with 8GB of memory.

4.2 MySQL configuration

The MySQL version 5.0.51a (x86_64) is used and databases were created using the INNODB storage engine with the following tuned INNODB parameters:

innodb_buffer_pool_size – is set to `128MB`.

innodb_thread_concurrency – is set to `16`.

innodb_log_buffer_size – is set to `32MB`.

For a detailed description of the modified INNODB variables please consult the MySQL user manual.

4.3 MyARM configuration

The standard MyARM 1.3.x.5 configuration is used with the following property changes:

si_thread.number – is set to `2`, `4` or `6`.

si_thread.queue.size – is set to `256`.

basic.armdata.buffer.size – is set to `196608`.

basic.armdata.buffer.pool.max – is set to `1024`.

agent.transaction.pool.max – is set to `2048`.

agent.metric.pool.max – is set to `4096`.

agent.sink.thread.queue.size – is set to `1024`.

These property changes are made to support very high transaction measurement rates of the MyARM ARM agent. For a detailed description of the modified MyARM properties please consult the MyARM user manual.

5 Results

The Figure 1 shows the results of all tests as described above. First of all the number of MySQL connection threads influence the overall transaction throughput in any scenario. Thus if high transaction rates are expected configure MyARM to use the `thread` datasink. Within this test setup four MySQL datasink threads performs best.

The shown results are the average transaction per second values from the last 10 MyARM builds of the version MyARM 1.3.x.5.

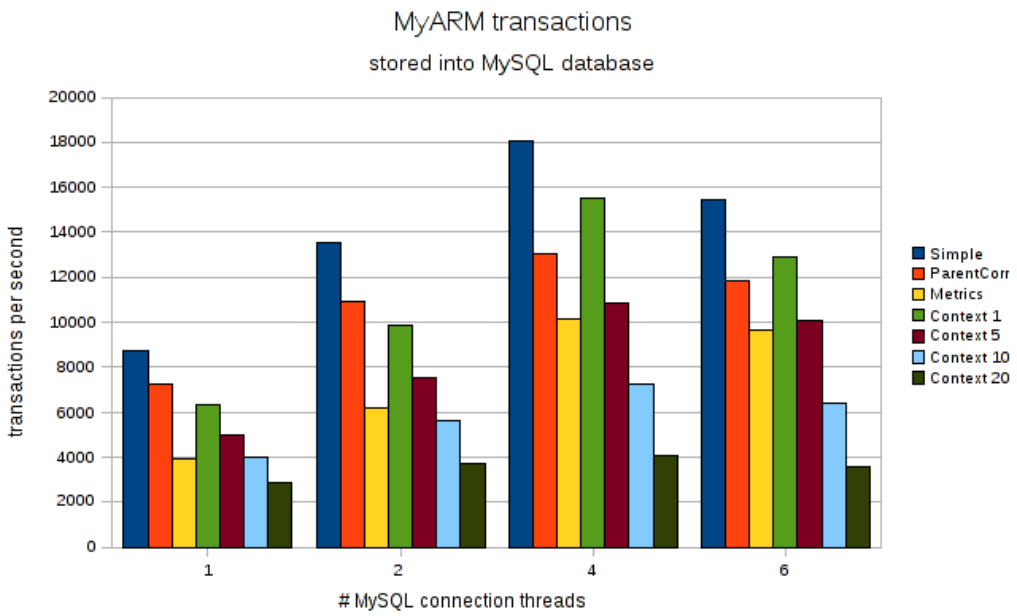

Figure 1: MyARM transaction MySQL throughput

In real world ARM instrumented application a mixure of the outlined scenarios are present. However the following rules should be taken into account for instrumenting applications with ARM:

- Add only information (metrics, context properties) which are of real interest
- Generate only a correlator if its passed to a sub-transaction.
- Prefer context properties instead of metrics if possible.
- Use diagnostic detail for failed or aborted transactions instead of repeating information in any transaction for diagnosing errors.

The following table lists all average transaction throughput rates in detail:

THREADS	SIMPLE	PARENTCORR	METRICS	CONTEXT 1	CONTEXT 5	CONTEXT 10	CONTEXT 20
1	8746	7234	3927	6308	4970	4018	2879
2	13539	10913	6175	9875	7537	5616	3742
4	18044	13028	10145	15535	10842	7257	4040
6	15435	11826	9645	12935	10050	6424	3605

Table 1: **MyARM transaction MySQL average througput (transactions per seconds)**

5.1 One datasink thread result details

VERSION	SIMPLE	PARENTCORR	METRICS	CONTEXT 1	CONTEXT 5	CONTEXT 10	CONTEXT 20
1.3.2436.5	8830	7117	3948	6387	5000	4043	2891
1.3.2435.5	8691	7165	3885	6230	4943	3930	2860
1.3.2434.5	8691	7307	3913	6285	4941	4041	2897
1.3.2433.5	8853	7315	3920	6291	5044	4075	2892
1.3.2432.5	8658	7275	3945	6373	4898	3975	2927
1.3.2431.5	8884	7352	3936	6311	5005	4039	2886
1.3.2430.5	8718	7194	3930	6371	5001	4031	2851
1.3.2429.5	8669	7194	3900	6323	4939	3987	2829
1.3.2428.5	8771	7285	3941	6269	4928	4064	2900
1.3.2427.5	8691	7135	3947	6240	4998	3996	2856

Table 2: **MyARM transaction MySQL througput with one datasink thread (transactions per seconds)**

5.2 Two datasink threads result details

VERSION	SIMPLE	PARENTCORR	METRICS	CONTEXT 1	CONTEXT 5	CONTEXT 10	CONTEXT 20
1.3.2436.5	13504	10554	6114	9784	8406	6167	3691
1.3.2435.5	14388	10666	6036	9661	7262	5557	3954
1.3.2434.5	13080	10570	6129	9666	7309	5418	3885
1.3.2433.5	13140	10666	6153	10565	7275	5457	3736
1.3.2432.5	13504	10548	6236	9813	7451	5502	3619
1.3.2431.5	13262	10863	6125	9857	7399	5936	3725
1.3.2430.5	13368	10952	6071	9920	7323	5529	3729
1.3.2429.5	13342	11363	6131	9813	8250	5505	3685
1.3.2428.5	13063	11383	6112	9779	7459	5530	3703
1.3.2427.5	14738	11567	6642	9891	7233	5560	3694

Table 3: **MyARM transaction MySQL througput with two datasink threads (transactions per seconds)**

5.3 Four datasink threads result details

VERSION	SIMPLE	PARENTCORR	METRICS	CONTEXT 1	CONTEXT 5	CONTEXT 10	CONTEXT 20
1.3.2436.5	19342	13175	9596	15128	11025	7312	4010
1.3.2435.5	17825	13253	9794	15060	10712	7238	4181
1.3.2434.5	17761	12944	10000	15673	10626	7127	3951
1.3.2433.5	17482	13080	10389	15723	10465	7307	4042
1.3.2432.5	18050	13012	10509	15467	10729	7238	4082
1.3.2431.5	17436	12492	10362	15673	11074	7092	3949
1.3.2430.5	18248	13413	10416	15661	10822	7264	3990
1.3.2429.5	18450	12853	10230	15673	11148	7358	4048
1.3.2428.5	18639	12812	10106	15408	10970	7222	4097
1.3.2427.5	17211	13245	10045	15885	10845	7415	4046

Table 4: **MyARM transaction MySQL througput with four datasink threads (transactions per seconds)**

5.4 Six datasink threads result details

VERSION	SIMPLE	PARENTCORR	METRICS	CONTEXT 1	CONTEXT 5	CONTEXT 10	CONTEXT 20
1.3.2436.5	15576	11792	9551	12836	9789	6470	3613
1.3.2435.5	15540	11655	9689	12586	9980	6495	3564
1.3.2434.5	15163	11454	9389	12853	9857	6038	3459
1.3.2433.5	15772	12019	9633	12836	9519	6371	3674
1.3.2432.5	14771	11848	9592	13166	10111	6697	3675
1.3.2431.5	15515	11897	10400	13262	10162	6339	3588
1.3.2430.5	15515	12055	9541	13192	10303	6633	3541
1.3.2429.5	15588	11709	9487	12722	10405	6635	3597
1.3.2428.5	15910	11983	9652	12978	10178	6174	3703
1.3.2427.5	15003	11848	9519	12919	10193	6383	3635

Table 5: **MyARM transaction MySQL througput with six datasink threads (transactions per seconds)**