Grizzly-thrift Benchmarking

Grizzly-Thrift Server/Client Modules Benchmarking

This page is for benchmarking various Thrift Server-Client modules which are TSocketServer/Client, TThreadpoolServer, TTNonblockingServer, Netty Server/Client and Grizzly Server/Client. I used business operations based on Thrift tutorial for test but modified a bit logic for packet size.

Test Information

- Server Type/Client Type: TServer-TSocketClient vs TServer-NettyClient vs TServer-GrizzlyClient vs GrizzlyServer-TSocketClient vs GrizlzyServer-GrizzlyClient vs etc...
- Message Size: About 3M Bytes, 3K Bytes, 300 Bytes
- Thrift Protocol: Binary, Compact
- Client Connections: 40, 20, 60
- Server and Client Test Machine Information
  - CPU: Intel Xeon 3.3G, 8 Processors, 8 * 4 Cores
  - Memory: 16G
  - OS: Linux SentOS
  - JDK: 1.6.0_29
  - Network: 1G
  - Versions: Thrift v0.7.0, Grizzly v2.2(git://java.net/grizzly~git), Netty v4.0.0(git://github.com/netty/netty.git), Netty Tools v1.2.8(https://github.com/cgbystrom/netty-tools.git). Most of all are the lastest version(2011/12/05).
- Scenario
  - After 1min warming-up, testing 5min and collecting total results.
  - Please see the sources which I attached.
    - ThriftServerBenchmark.java: Server modules for benchmarking
    - ThriftClientBenchmark.java: Client modules for benchmarking
    - CalculatorHandler.java: Business logic for Thrift services

Benchmarking Results

- 3M + Compact + 40 Connections

Server Types	TSocket Client	Netty Client	Grizzly Client
TServer	8,637	**478**	8,510
TThreadPoolServer	11,221	2,273	11,220
TNonblockingServer	11,223	1,832	11,221
Netty	11,220	2,311	11,220
Grizzly	11,221	1,765	**11,225**

  - Netty client had the performance problem, so I would exclude it for next benchmarking.
- 3M + Binary + 40 Connections

Server Types	TSocket Client	Grizzly Client
TThreadPoolServer	11,219	11,215
TNonblockingServer	11,221	11,221
Netty	11,213	11,221
Grizzly	11,220	**11,222**

In 3M test, Compact/Binary and Server/Client tests were meaningless for performance.

- *3K + Compact + 40 Connections*

Server Types	Grizzly Client
TThreadPoolServer	8,283,705

TNonblockingServer	5,801,319
Netty	**9,058,550**
Grizzly	**8,964,358**
Grizzly(SameIO)	**9,098,590**

  - TNonblockingServer had the performance problem. And Netty and Grizzlys' results were better than Thrift server modules'.
- 3K + Binary + 40 Connections

Server Types	TSocket Client	Grizzly Client
TThreadPoolServer	7,619,693	8,163,692
TNonblockingServer	5,444,630	6,032,290
Netty	8,254,168	**8,930,896**
Grizzly	8,204,097	**8,833,978**
Grizzly(SameIO)	8,257,918	**8,960,497**

  - Grizzly client module had better performance than TSocket client so I would use only Grizzly client for next benchmarking.

In 3K test, Compact protocol is better than Binary protocol. And Netty and Grizzlys' results were better than Thrift server modules' so I would use only Netty and Grizzly server for next benchmarking.

- 300Bytes + Compact + 20 Connections

Server Types	Grizzly Client
Netty	10,269,876
Grizzly(SameIO)	**10,349,440**
Grizzly(LeaderF)	9,654,216

- *300Bytes + Compact + 40 Connections*

Server Types	Grizzly Client
Netty	14,569,820
Grizzly(SameIO)	**14,770,452**
Grizzly(LeaderF)	13,674,641

- 300Bytes + Compact + 60 Connections

Server Types	Grizzly Client
Netty	15,783,774
Grizzly(SameIO)	**15,962,425**
Grizzly(LeaderF)	15,227,426

- *300Bytes + Compact + 80 Connections*

Server Types	Grizzly Client
Netty	**16,964,578**
Grizzly(SameIO)	16,712,315
Grizzly(Worker)	15,890,537
Grizzly(LeaderF)	16,252,280

- *300Bytes + Compact + 100 Connections*

Server Types	Grizzly Client
Netty	15,879,803
Grizzly(SameIO)	15,781,153
Grizzly(Worker)	16,136,977
Grizzly(LeaderF)	**16,437,650**

- 300Bytes + Compact + 120 Connections

Server Types	Grizzly Client
Netty	15,904,968
Grizzly(SameIO)	15,985,106
Grizzly(Worker)	16,097,609
Grizzly(LeaderF)	**16,164,636**

- 300Bytes + Compact + 150 Connections

Server Types	Grizzly Client
Netty	15,952,442
Grizzly(SameIO)	16,109,154
Grizzly(Worker)	**16,261,584**
Grizzly(LeaderF)	15,923,040

- 300Bytes + Compact + 500 Connections

Server Types	Grizzly Client
Netty	12,463,442
Grizzly(SameIO)	12,499,963
Grizzly(Worker)	12,461,131
Grizzly(LeaderF)	**12,532,517**

- 300Bytes + Compact + 1000 Connections

Server Types	Grizzly Client
Netty	11,867,630
Grizzly(SameIO)	11,903,400
Grizzly(Worker)	**11,906,507**
Grizzly(LeaderF)	11,812,262

In high connections(more than 120 connections), most of servers didn't receive proper requests of client-side because the client machine of this environment used too much resouces such as high CPU usages. So I think that more client machines needs to calculate meaningful data of more connections. In 100 connections, Netty and Grizzly's Same IO Strategy's throughput decreased but Grizzly's Woker Thread IO and Leader Follower IO's throughput increased.

In our test cases and environments, Worker Thread IO Strategy and Leader Follower IO Strategy are more effective than Same IO Thread Strategy if servers should have more than 100 connections.

Conclusion

- Results of 300Bytes + Compact + 40 Connections

Server Types	TSocket Client	Netty Client	Grizzly Client
TServer	741,417		604,558
TThreadPoolServer	14,731,560		12,747,230
TNonblockingServer	6,060,111		6,723,402
Netty	14,749,519		14,569,820
Grizzly(SameIO)	**14,931,745**	9,066,525	**14,770,452**

- Results of 3KBytes + Compact + 40 Connections

Server Types	TSocket Client	Netty Client	Grizzly Client
TServer	631,300		526,341
TThreadPoolServer	7,708,088		8,283,705
TNonblockingServer	5,264,995		5,801,319
Netty	8,372,804		9,058,550
Grizzly(SameIO)	8,381,352	3,718,431	**9,098,590**

- 300Bytes + Compact + 100 Connections

Server Types	Grizzly Client
Netty	15,879,803
Grizzly(SameIO)	15,781,153
Grizzly(Worker)	16,136,977
Grizzly(LeaderF)	**16,437,650**

- Server Module
    - Grizzly Same IO Strategy is best in low and medium connections. Grizzly LeaderFollower IO Strategy is best in high connections.
    - CPU Usages: Netty==GrizzlySameIO < GrizzlyLeaderFollowerIO < GrizzlyWorkerIO
- Client Module
    - In small packets, TSocket is best. In larget packets, Grizzly client is best.
- Thrift Protocol
    - In this scenario, Compact protocol is best.