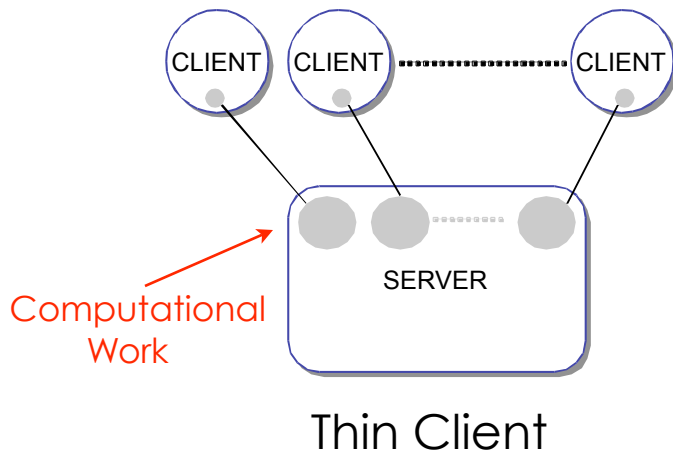


Our goal: find how to improve MDSplus performance on our Linux clusters ...

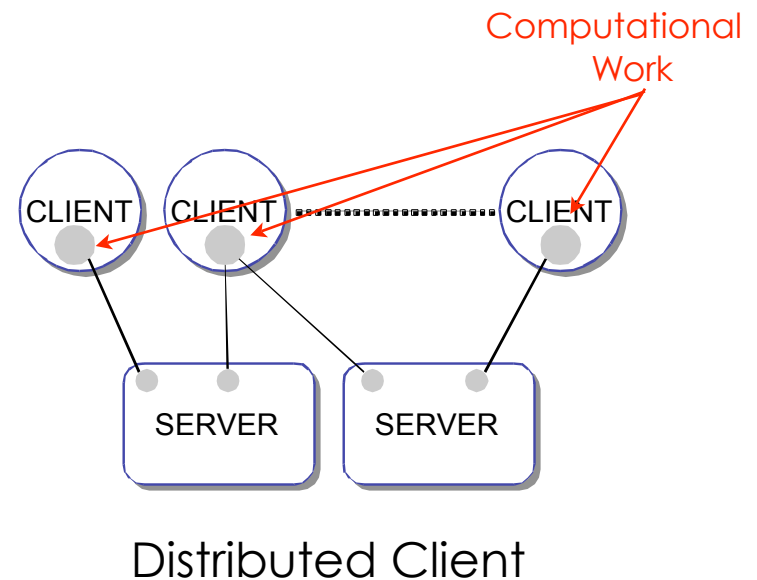
We are investigating the use of grid computing and Linux technology to improve performance in our core data management services. We are in the process of converting much of our functionality to cluster-based and grid-enabled software. One of the most important pieces is a new **distributed client** version of the MDSplus scientific data management system that is presently used to support fusion research in over 30 countries worldwide. The traditional client is known as **thin client**. To improve data handling performance, the staff is investigating the use of Linux clusters for both MDSplus clients and servers. We hope that the new distributed capability will result in better load balancing between these clients and servers, and more efficient use of network resources resulting in improved support of the data analysis needs of the scientific staff.

So, we wanted to find out if the distributed client offers improvements over the thin client ...

The most relevant distinction between the two is that the traditional MDSplus thin client method puts more load on the server, because the server performs all of the expression evaluation, data compression and decompression:



In contrast, the distributed client performs the evaluation, data compression and decompression instead of querying the server:



First we established baseline numbers on the two clients ...

- All tests were run during normal production hours but not during DIII-D operations.
- Tests consisted of one-at-a-time reads of floating point data using 8MB data sizes
- In the simple case of a single client reading 2 million values of compressed floating-point data there is almost no measurable difference between thin client and distributed client:

Thin	Dist.
8.5 MB/s	8.6 MB/s

Next we increased the load by adding six simultaneous data reads ...

- Tests consisted of one-at-time reads of floating point data using 8MB data sizes. This test more closely simulated usage during DIII-D operations.
- This is a *critical test* because it shows that as load increases, distributed client becomes much faster than thin.
- Distributed client outperforms thin by approximately 20%:

Thin	Dist.
2.8 MB/s	3.3 MB/s

Then we ran a test evaluating math usage within node expressions ...

- Expression used was $_data = [...], \sin(data) + \cos(_data) + \tan(_data)$, where [...] is the data array of 2 million floating point values
- Distributed client spreads computational load among all clients rather than on one server.
- Distributed client outperforms thin by approximately 23%:

Thin	Dist.
2.0 MB/s	2.5 MB/s

In the next test we introduced indirection - multi-referencing within the MDSplus trees ...

- Each data node contained a node reference rather than raw data; the subject of this node reference contained another node reference, etc.
- In all we tested nine levels of indirection.
- Distributed client outperforms thin by approximately 10%:

Thin	Dist.
3.0 MB/s	3.3 MB/s

Following this we added calls to PTDATA into the process ...

- Test of reading data from a numeric node containing a ptdata2() TDI function, which points to data in the (separate) PTDATA database to be read.
- Thin client needs an extra network “hop” to PTDATA.
- Distributed client outperforms thin by approximately 16%:

Thin	Dist.
2.3 MB/s	2.6 MB/s

Now, some other types of tests. First we turned off compression (a default setting) ...

- This is another *critical test*, because it demonstrates distributed client's higher reliance on network usage.
- Distributed client performance dropped measurably relative to its baseline testing result.
- Thin client performance stayed the same relative to its baseline testing result.

Thin	Dist.
8.3 MB/s	4.6 MB/s

Finally we tried something completely different: changing a system parameter ...

- We increased the maximum TCP send and receive buffers and the memory reserved for TCP buffers.
- This is a common change on systems hosting databases because it allows the maximum number of data reads and writes.
- Surprisingly, performance went down for both clients!

Summary of results ...

We found the **distributed client** to be superior to **thin client** because:

- It makes best use of compression (an MDSplus default setting)
 - Less data sent over the network
 - Data throughput increased
- Distributes load better
 - More work done on clients, less on server
 - Under normal loads, better performance characteristics
- Eliminates a network hop when using MDSplus as a proxy to retrieve data from (separate) PTDATA database

Our conclusions ...

- In all practical tests, distributed client outperformed thin in measurable ways.
- Using distributed client we can leverage the continued rollout of clustered Linux systems.
- We need to find the best ways to better tune the systems to improve MDSplus performance even more.