WebSphere MQ File Transfer Edition for Solaris V7.0.1

Performance Evaluations

Version 1.0

November 2009

Richard Cumbers

WebSphere MQ File Transfer
IBM UK Laboratories
Hursley Park
Winchester
Hampshire
SO21 2JN

Please take Note!

Before using this report, please be sure to read the paragraphs on “disclaimers”, “warranty and liability exclusion”, “errors and omissions”, and the other general information paragraphs in the "Notices" section below.

First Edition, November 2009.

This edition applies to WebSphere MQ File Transfer Edition for Solaris V7.0.1 (and to all subsequent releases and modifications until otherwise indicated in new editions).

Notices

DISCLAIMERS

The performance data contained in this report were measured in a controlled environment. Results obtained in other environments may vary significantly.

You should not assume that the information contained in this report has been submitted to any formal testing by IBM.

Any use of this information and implementation of any of the techniques are the responsibility of the licensed user. Much depends on the ability of the licensed user to evaluate the data and to project the results into their own operational environment.

WARRANTY AND LIABILITY EXCLUSION

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

In Germany and Austria, notwithstanding the above exclusions, IBM's warranty and liability are governed only by the respective terms applicable for Germany and Austria in the corresponding IBM program license agreement(s).

ERRORS AND OMISSIONS
The information set forth in this report could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; any such change will be incorporated in new editions of the information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information at any time and without notice.

INTENDED AUDIENCE
This report is intended for architects, systems programmers, analysts and programmers wanting to understand the performance characteristics of WebSphere MQ File Transfer Edition V7. The information is not intended as the specification of any programming interface that is provided by WebSphere. It is assumed that the reader is familiar with the concepts and operation of WebSphere MQ File Transfer Edition V7.

LOCAL AVAILABILITY
References in this report to IBM products or programs do not imply that IBM intends to make these available in all countries in which IBM operates. Consult your local IBM representative for information on the products and services currently available in your area.

ALTERNATIVE PRODUCTS AND SERVICES
Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

USE OF INFORMATION PROVIDED BY YOU
IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

TRADEMARKS AND SERVICE MARKS
The following terms used in this publication are trademarks of International Business Machines Corporation in the United States, other countries or both:

- IBM
- WebSphere
- DB2

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

EXPORT REGULATIONS
You agree to comply with all applicable export and import laws and regulations.

How this document is arranged

Performance Headlines

Pages: 2-21

Chapter 2 details the performance headlines for the two scenarios. Each scenario is detailed fully with diagrams in this section. The headline tests show how the altering a property (agentChunkSize) for an FTE agent affects the transfer time. The tests demonstrate the effect of transferring files as a group of transfers versus transferring files as a single transfer. Chapter 2 gives a detailed description of the agentChunkSize property, and its use within an FTE agent.

We detail the time taken for each transfer to complete and the associated CPU utilisation for the hardware in use.

Tuning Recommendations

Pages: 22-24

Chapter 3 discusses the appropriate tuning that should be applied to both the WebSphere MQ network, and File Transfer Edition Agents.

Measurement Environment

Pages: 25-26

Chapter 4 gives an overview of the environment used to gather the performance results. This includes a detailed description of the hardware and software.

Contents

1 - Overview

WebSphere MQ File Transfer Edition is a managed file transfer product that uses WebSphere MQ as it's transport layer. This is the first performance report on Solaris and so there is no comparison to make between versions.

This performance report details WebSphere MQ File Transfer Edition in a range of scenarios, giving the reader information on transfer times and CPU utilisation. The report is based on measurements taken from Sparc hardware, running the Solaris operating system.

2 - Performance Headlines

The measurements for the performance headlines are based on the time taken to transfer a set of files, and the associated CPU cost. A single performance measurement will use 1000MB worth of files, with the size of the files varying as follows:

- 1MB
- 10MB
- 100MB

To illustrate a typical test, if using a test is using a 1MB file then the test will transfer 1000 files in a single performance run. Varying the file size, but keeping the same overall MB transferred demonstrates the cost of the open and close file operations on transfer time and CPU usage.

The performance headlines demonstrate the effect of altering the agent's Chunk Size property. (See http://publib.boulder.ibm.com/infocenter/wmqfte/v7r0/index.jsp?topic=/com.ibm.wmqfte.admin.doc/properties.htm for more details on setting this property). The Chunk Size defines the size of the MQ message that the agent will use to transfer the files. The following Chunk Sizes (defined in bytes) have been used:

- 65536
- 131072
- 262144 (this is the agent's default value)
- 524288

To demonstrate the multi-threaded capability of the agent, a multiple transfer test was run and compared to a single transfer run. The multiple transfer test took divided the number of files transferred in the single transfer test by ten, and submitted them at the same time.

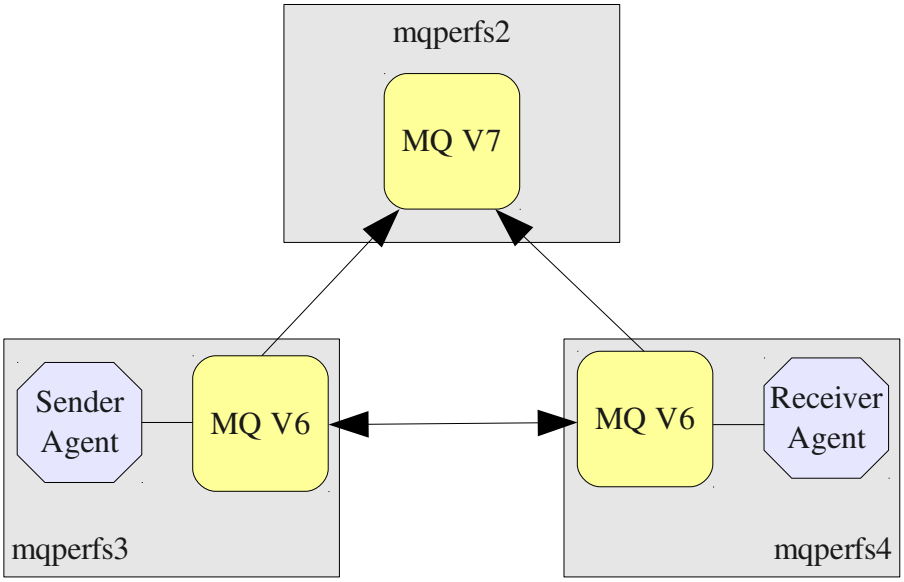
All files were transferred using text mode, as opposed to binary mode. Each file transferred was uniform in size for a given performance run, but contained random data. Transfers were submitted using the documented XML format which can be found in the samples directory of the WebSphere MQ File Transfer Edition Docs and Tools CD.

An agents queue manager was always WebSphere MQ Version 6, with the latest FixPack applied. For this performance report this equated to version 6.0.2.7. For the Coordination queue manager, the latest FixPack for Version 7.0.0 was used. For this performance report this equated to version 7.0.0.1

The results are laid out in the subsequent chapters. Each test case has its own results table, and associated graphs. The first set of tables and figures show the reader the effect that the chunk size (agentChunkSize) property has on the transfer time for a particular file size. These figures are then followed by a second set of tables and figures that compare the combinations of agent connectivity with the single/multiple transfer test at each of the Chunk Sizes. The second set of tables and figures serve to show the reader the difference between the transfer speeds and their associated CPU costs when using different agent connectivity options and single/multiple transfers.

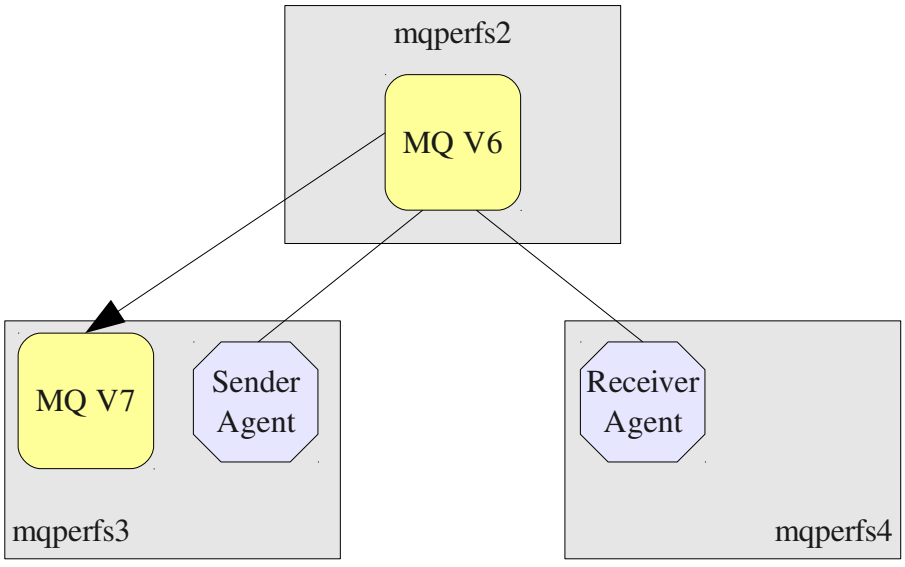
Agents Connecting in Bindings Mode

In this scenario, each agent is connected to a local queue manager in bindings mode. The two local queue managers are connected via sender/receiver channel pairs. A third queue manager is located on another machine, and is used as the coordination queue manager. The following diagram details the exact scenario:



Agents Connecting in Client Mode

In this scenario each agent is connected to the same single remote queue manager in client mode. A second queue manager is placed on the sender machine to act as the coordination queue manager. This coordination queue manager is not highly utilised as it is not directly involved in the transfers and so will have little or no effect on the sender CPU values that are collected. The following diagram details the exact scenario:



2.1 - 1MB File Size

Table 1 shows the full list of results for 1MB file size. Graphs showing the relevant times and cpu utilisation can be seen in Figures 1, 2, 3, 4.

Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Bindings	Single	Solaris	65536	134.09	13.69	13.64	N/A
Bindings	Single	Solaris	131072	70.51	16.03	21.51	N/A
Bindings	Single	Solaris	262144	53.91	15.92	25.7	N/A
Bindings	Single	Solaris	524288	53	13.36	25.21	N/A
Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Bindings	Multiple	Solaris	65536	25.43	30.19	56.15	N/A
Bindings	Multiple	Solaris	131072	22.28	30.14	63.16	N/A
Bindings	Multiple	Solaris	262144	23.29	28.19	53.97	N/A
Bindings	Multiple	Solaris	524288	21.37	27.79	72.98	N/A
Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Client	Single	Solaris	65536	156.71	10.63	12.6	1.93
Client	Single	Solaris	131072	92.65	10.83	15.26	2.03
Client	Single	Solaris	262144	59.81	11.82	21.84	2.03
Client	Single	Solaris	524288	50.85	11.03	24.17	2.99
Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Client	Multiple	Solaris	65536	44.28	12.92	35.93	4.11
Client	Multiple	Solaris	131072	36.89	13.19	33.46	6.63
Client	Multiple	Solaris	262144	31.61	15.55	35.31	4.83
Client	Multiple	Solaris	524288	30.77	14.88	36.54	5.07

Table 1

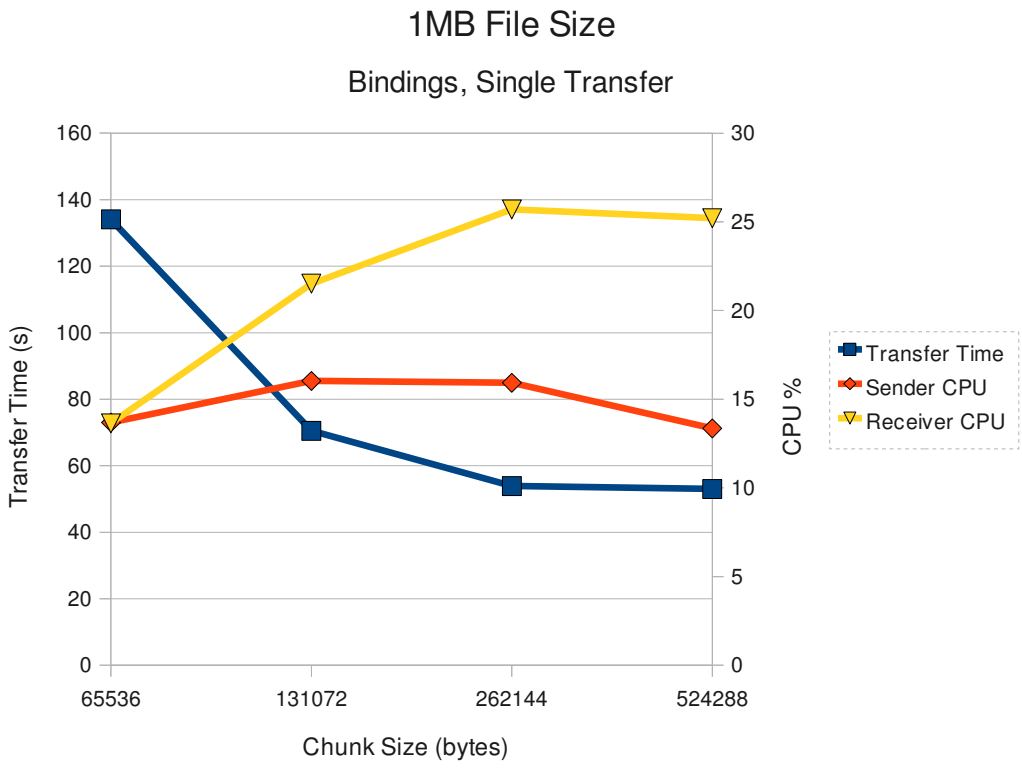


Figure 1

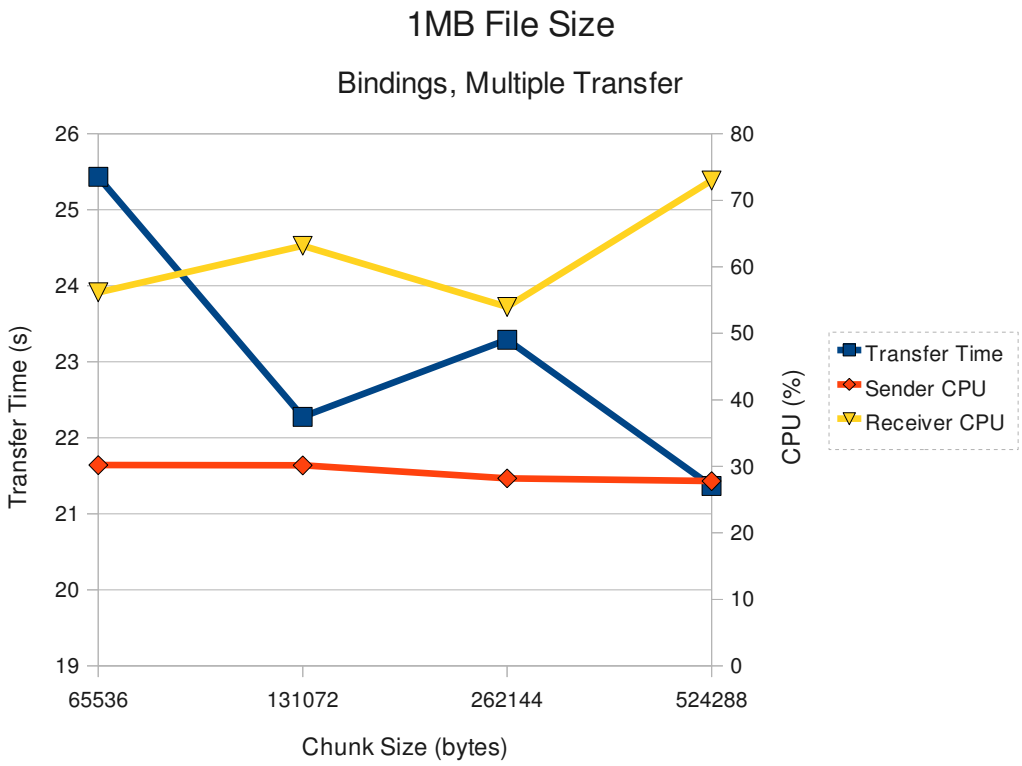


Figure 2

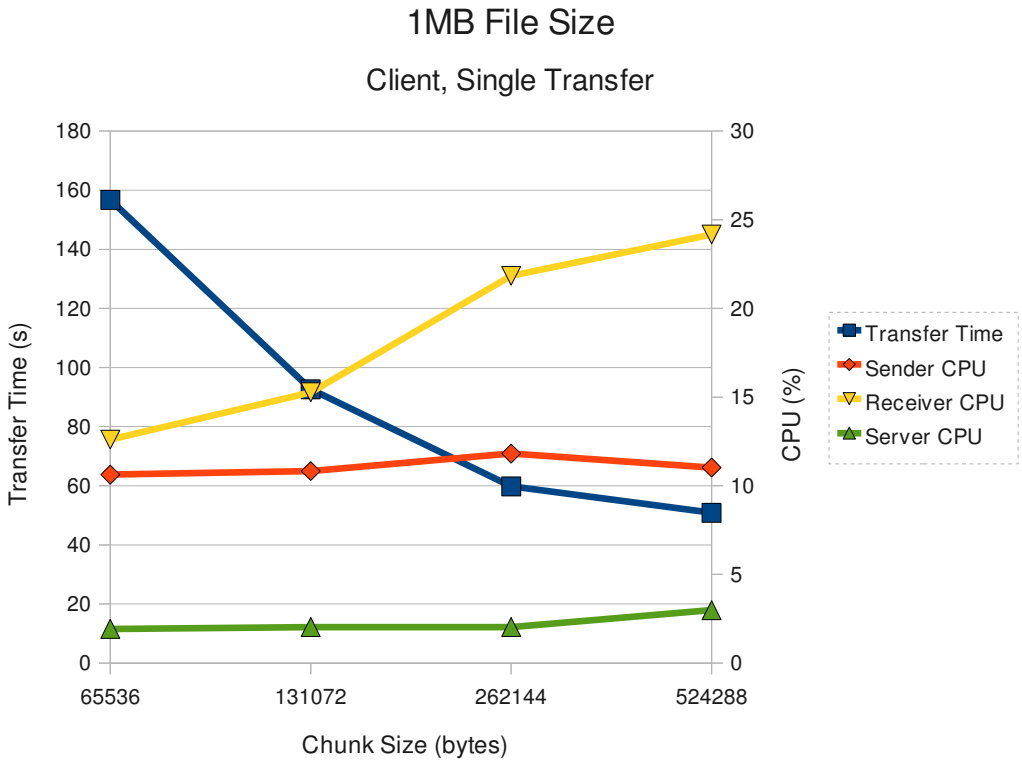


Figure 3

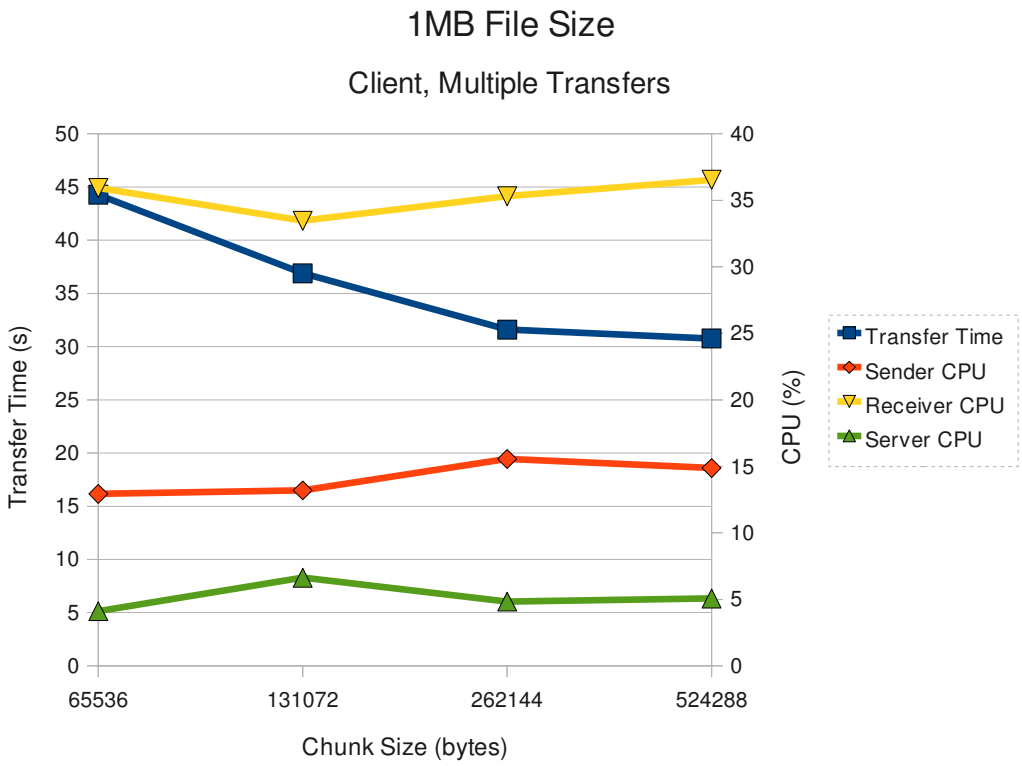


Figure 4

These results clearly demonstrate that larger Chunk Size decrease the transfer time. Single transfer scenarios are greatly affected by Chunk Size changes. The Multiple, Bindings scenario does not show the same level of time reduction, especially when compared to the Client, Multiple scenario.

As expected, the multiple transfer scenario utilises a much higher percentage of CPU overall compared to the single transfer. Sender CPU utilisation actually changes very little, whilst the receiver CPU increase is much more pronounced. This is in keeping with what WebSphere MQ performance reports detail, where an MQGET call is more expensive compared to an MQPUT call.

2.2 - 10MB File Size

Table 2 shows the full list of results for 10MB file size. Graphs showing the relevant times and CPU utilisation can be seen in Figures 5, 6, 7 and 8.

Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Bindings	Single	Solaris	65536	47.09	14.99	27.53	N/A
Bindings	Single	Solaris	131072	48.91	12.13	25.1	N/A
Bindings	Single	Solaris	262144	43.54	15.92	25.7	N/A
Bindings	Single	Solaris	524288	46.38	13.36	25.21	N/A
Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Bindings	Multiple	Solaris	65536	20.23	31.88	63.07	N/A
Bindings	Multiple	Solaris	131072	18.21	30.14	61.78	N/A
Bindings	Multiple	Solaris	262144	18.28	28.19	56.98	N/A
Bindings	Multiple	Solaris	524288	19.04	27.52	74.53	N/A
Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Client	Single	Solaris	65536	73.2	7.25	20.04	2.14
Client	Single	Solaris	131072	47.31	8.67	22.69	3.05
Client	Single	Solaris	262144	48.54	8.15	22.19	3
Client	Single	Solaris	524288	45.3	8.72	22.63	3.05
Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Client	Multiple	Solaris	65536	39.61	11.24	35.71	4.88
Client	Multiple	Solaris	131072	34.52	12.39	32.05	4.94
Client	Multiple	Solaris	262144	30.07	14.25	36.36	5.11
Client	Multiple	Solaris	524288	29.22	17.4	35.18	5.61

Table 2

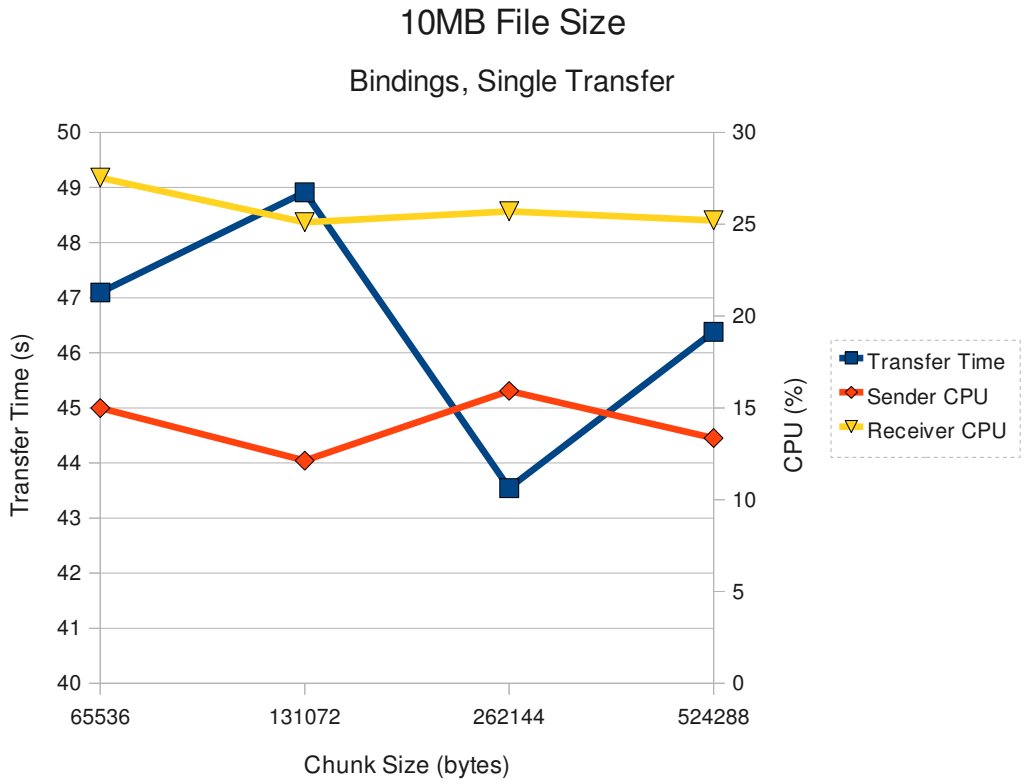


Figure 5

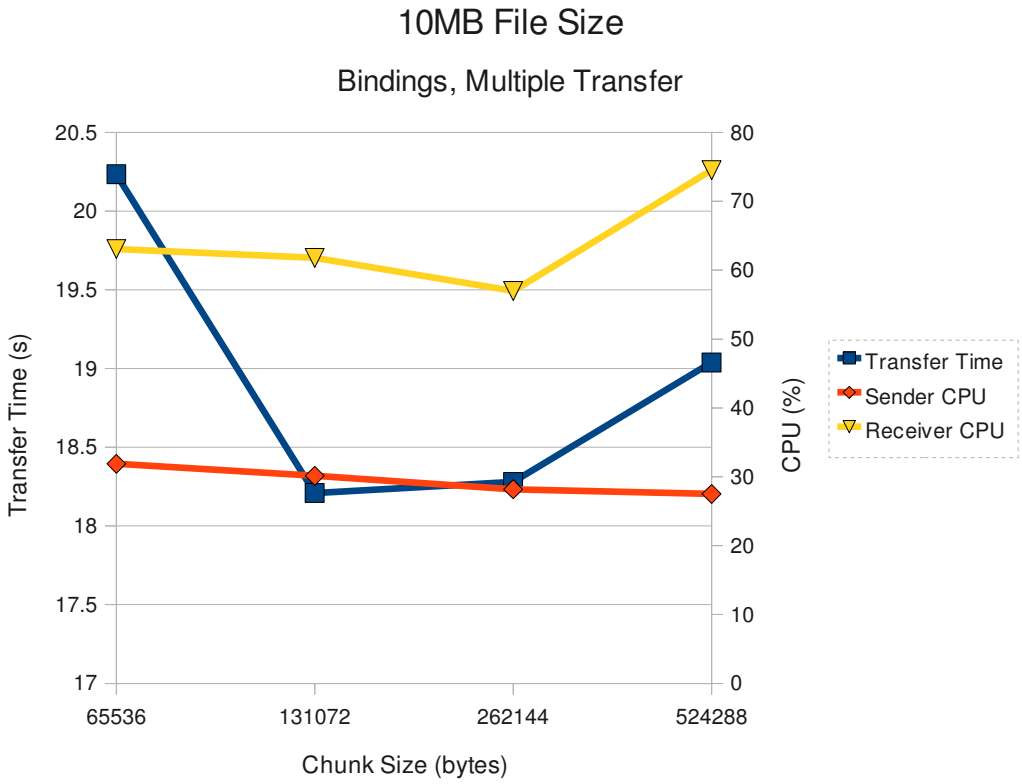


Figure 6

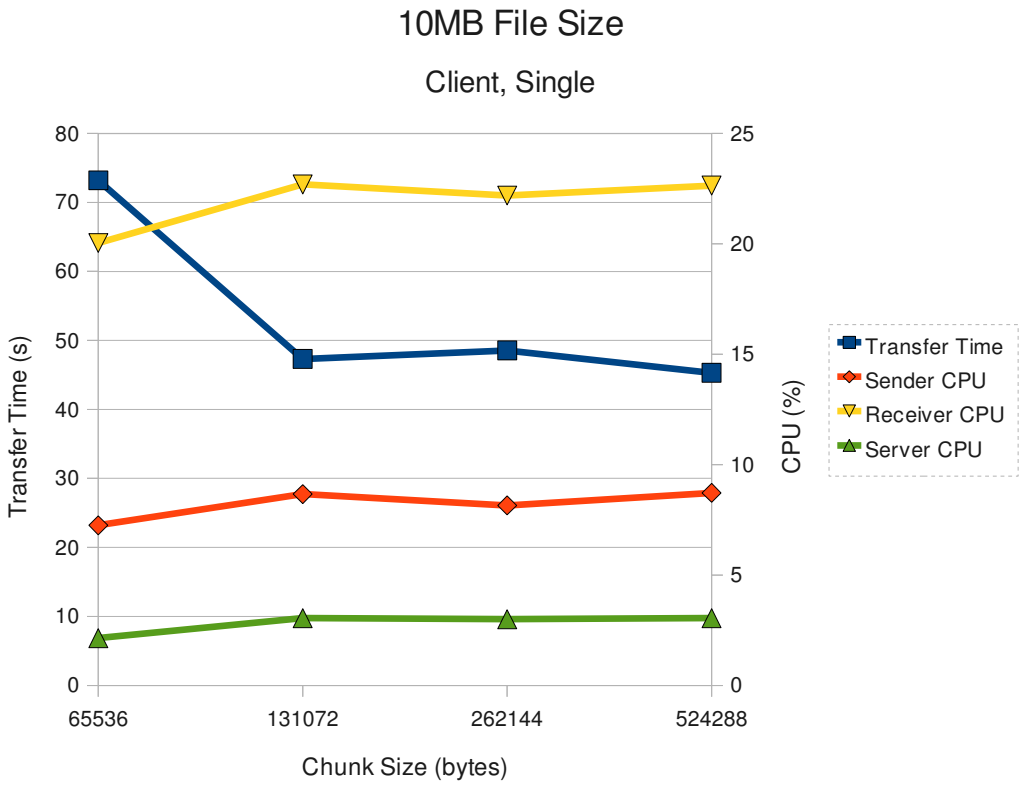


Figure 7

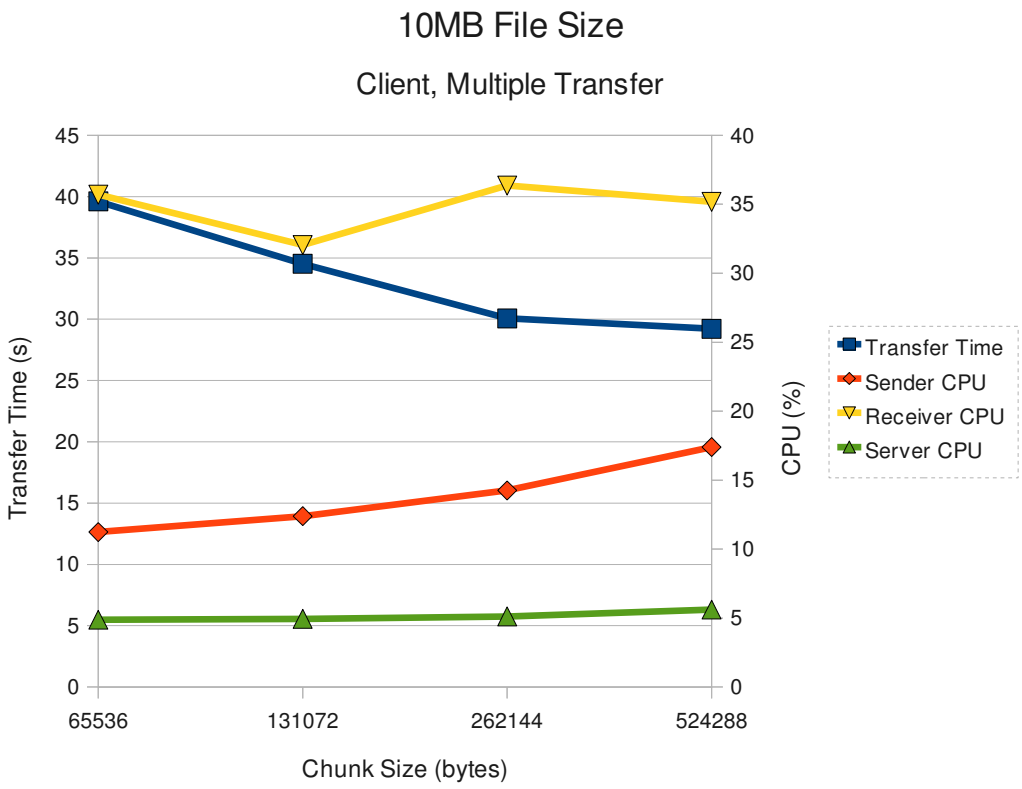


Figure 8

These graphs demonstrate that agents connected in bindings mode have an optimum chunk size of 262144. The client results suggest that a larger chunk size is better, with 524288 byte being the quickest transfer time measured.

2.3 - 100MB File Size

Table 3 shows the full list of results for 100MB file size. Graphs showing the relevant times and cpu utilisation can be seen in Figures 9, 10, 11 and 12.

Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Bindings	Single	Solaris	65536	48.63	12.51	25.58	N/A
Bindings	Single	Solaris	131072	47.43	11.36	25.04	N/A
Bindings	Single	Solaris	262144	45.1	11.63	24.95	N/A
Bindings	Single	Solaris	524288	46.76	10.97	23.56	N/A
Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Bindings	Multiple	Solaris	65536	20.31	31.05	64.02	N/A
Bindings	Multiple	Solaris	131072	16.57	32.74	67.22	N/A
Bindings	Multiple	Solaris	262144	16.52	30.93	63.63	N/A
Bindings	Multiple	Solaris	524288	19.23	28.2	71.77	N/A
Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Client	Single	Solaris	65536	75.93	5.87	18.44	2.03
Client	Single	Solaris	131072	49.18	7.48	21.11	3.02
Client	Single	Solaris	262144	47.9	8.09	22.38	2.99
Client	Single	Solaris	524288	45.3	8.24	22.27	3.09
Bindings/Client	Single/Multiple	Platform	ChunkSize	Transfer Time	Sender CPU	Receiver CPU	Server CPU
Client	Multiple	Solaris	65536	38.21	11.65	36.24	4.98
Client	Multiple	Solaris	131072	34.68	12.04	32.73	4.88
Client	Multiple	Solaris	262144	31.23	13.42	33.17	3.01
Client	Multiple	Solaris	524288	29.8	16.61	35.64	5.21

Table 3

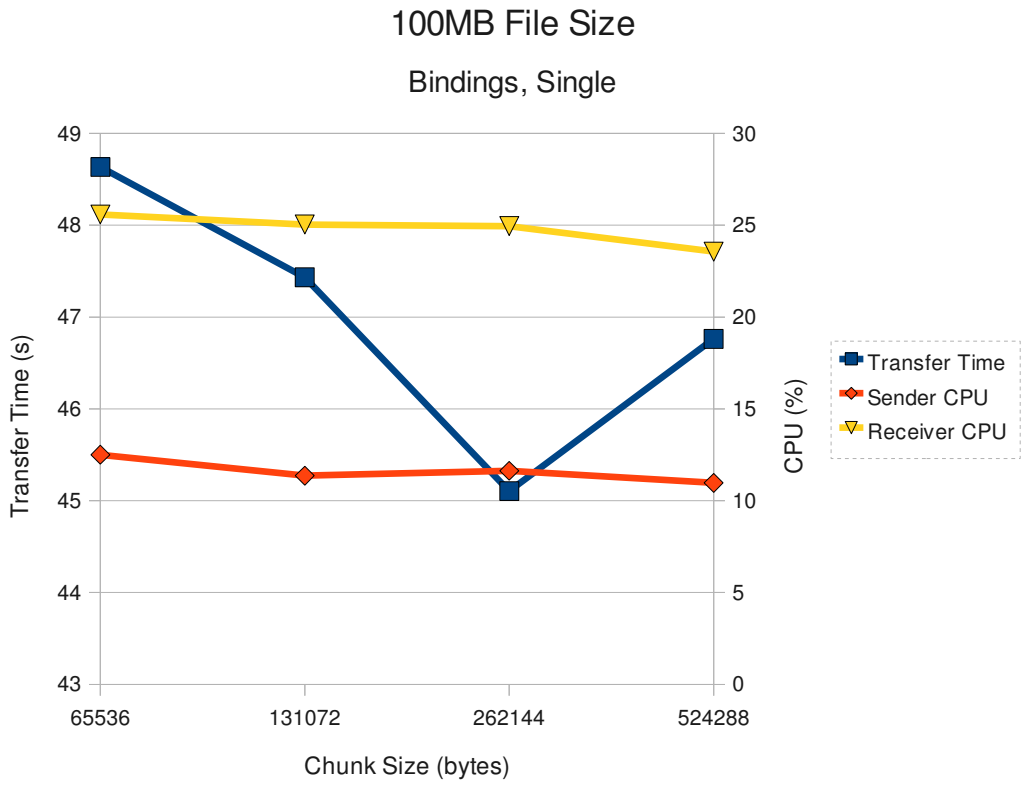


Figure 9

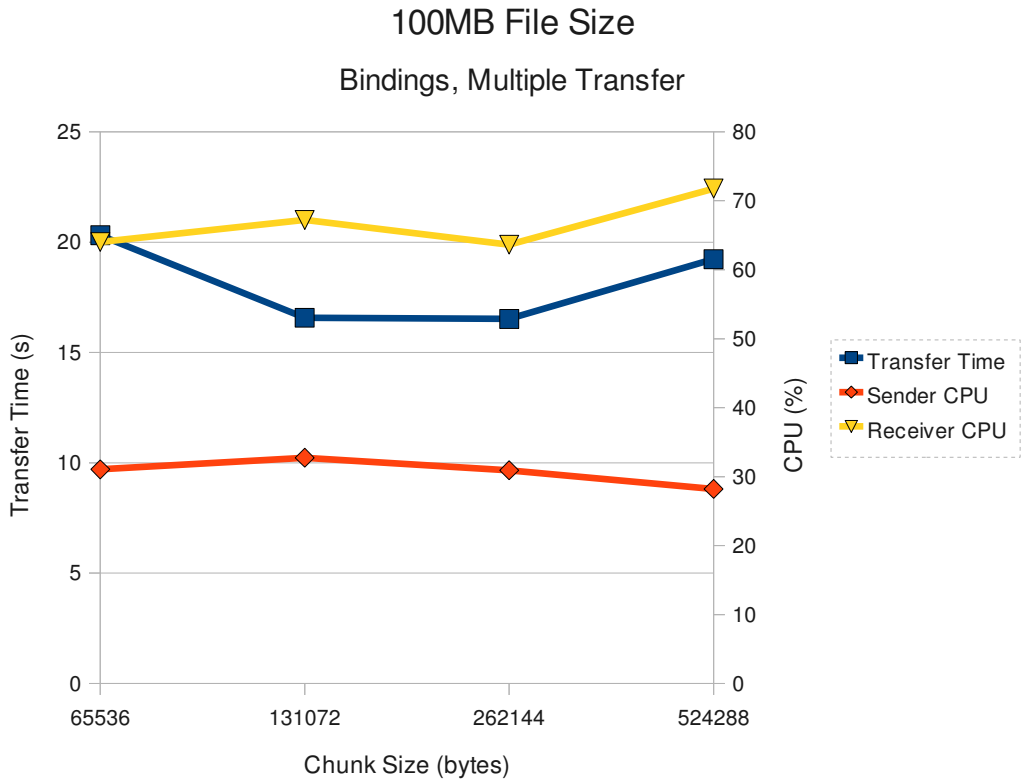


Figure 10

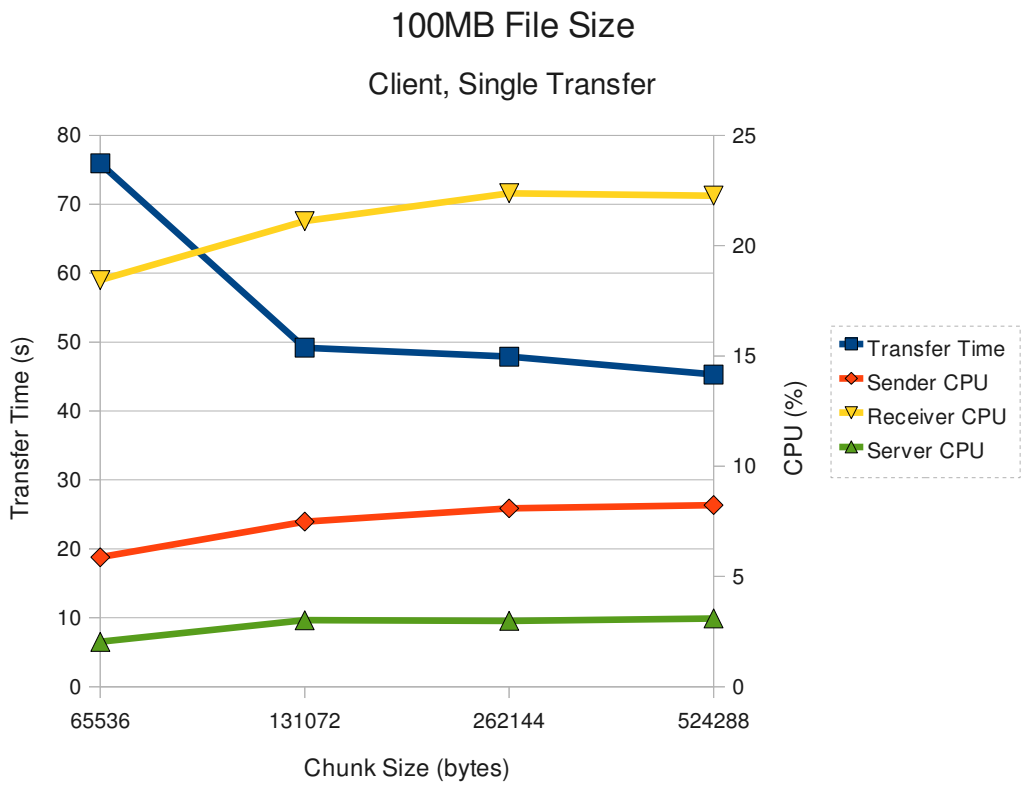


Figure 11

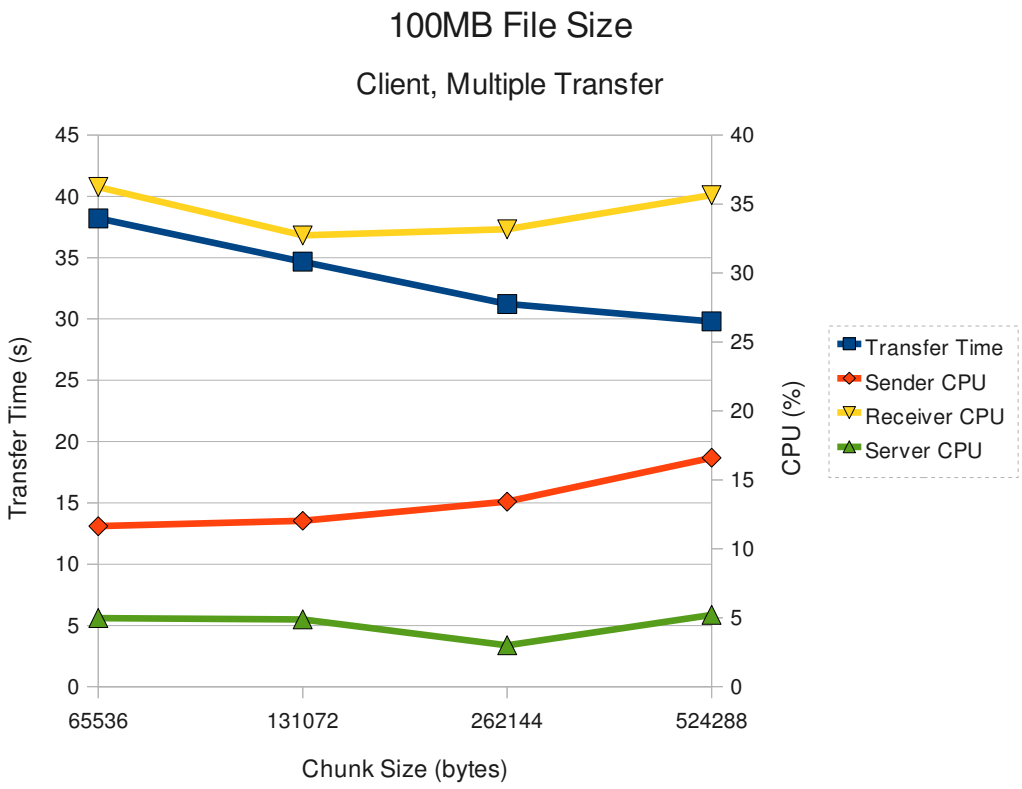


Figure 12

These graphs demonstrate that large files have an optimal chunk size of 262144 when being used by agents connected in Bindings mode. The Client results suggest that a larger chunk size is better, with 524288 bytes resulting in being the quickest transfer time measured.

2.4 - 1MB Scenario Comparison

The following tables and figures show the difference in transfer time and CPU utilisation by scenario when using the1MB file size.

ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
65536	Bindings / Single Transfer	134.09	13.69	13.64	N/A
65536	Bindings / Multiple Transfer	25.43	30.19	56.15	N/A
65536	Client / Single Transfer	156.71	10.63	12.6	1.93
65536	Client / Multiple Transfer	44.28	12.92	35.93	4.11
ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
131072	Bindings / Single Transfer	70.51	16.03	21.51	N/A
131072	Bindings / Multiple Transfer	22.28	30.14	63.16	N/A
131072	Client / Single Transfer	92.65	10.83	15.26	2.03
131072	Client / Multiple Transfer	36.89	13.19	33.46	6.63
ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
262144	Bindings / Single Transfer	53.91	15.92	25.7	N/A
262144	Bindings / Multiple Transfer	23.29	28.19	53.97	N/A
262144	Client / Single Transfer	59.81	11.82	21.84	2.03
262144	Client / Multiple Transfer	31.61	15.55	35.31	4.83
ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
524288	Bindings / Single Transfer	53	13.36	25.21	N/A
524288	Bindings / Multiple Transfer	21.37	27.79	72.98	N/A
524288	Client / Single Transfer	50.85	11.03	24.17	2.99
524288	Client / Multiple Transfer	30.77	14.88	36.54	5.07

Table 4

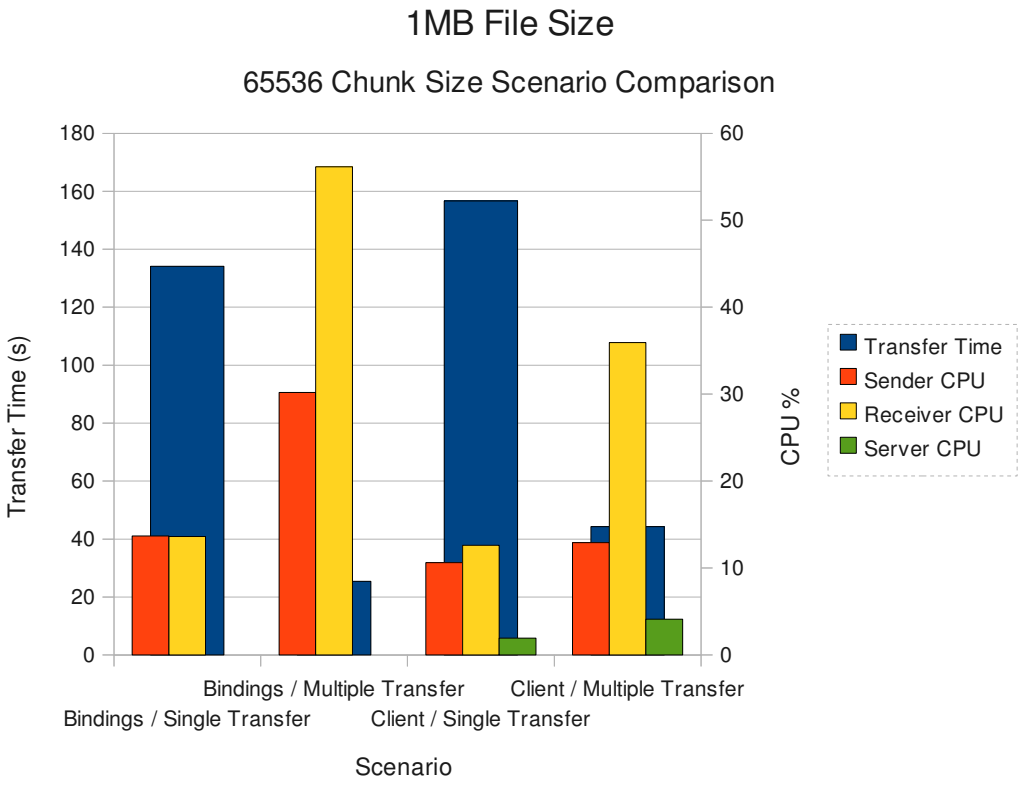


Figure 13

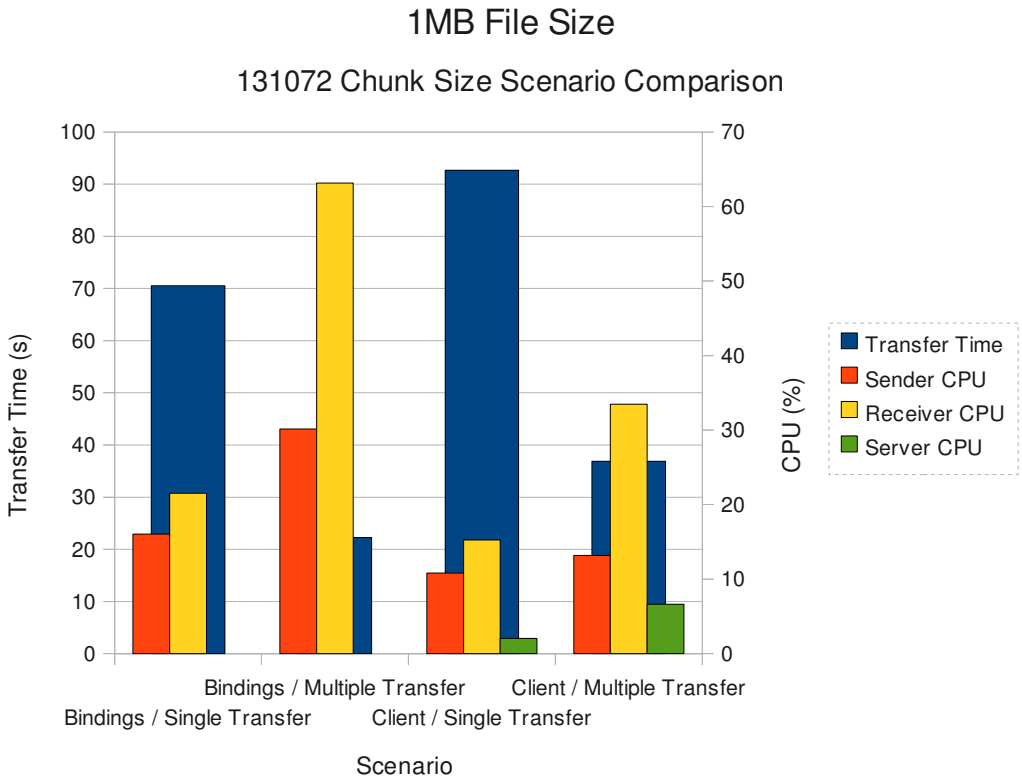


Figure 14

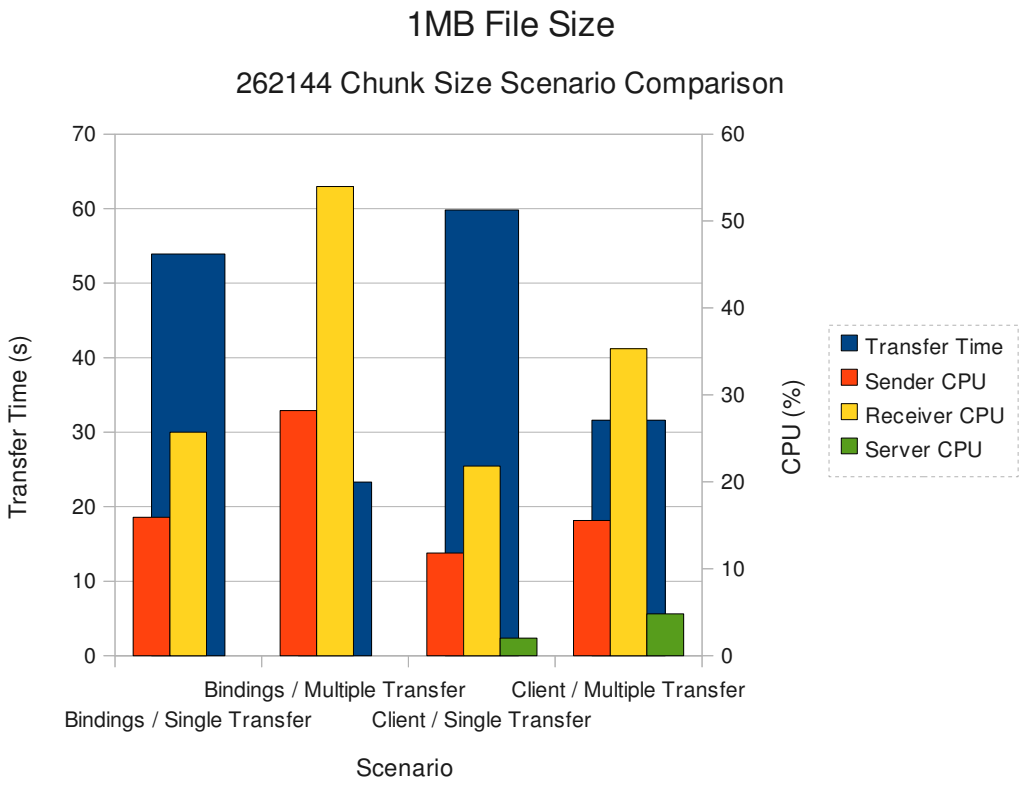


Figure 15

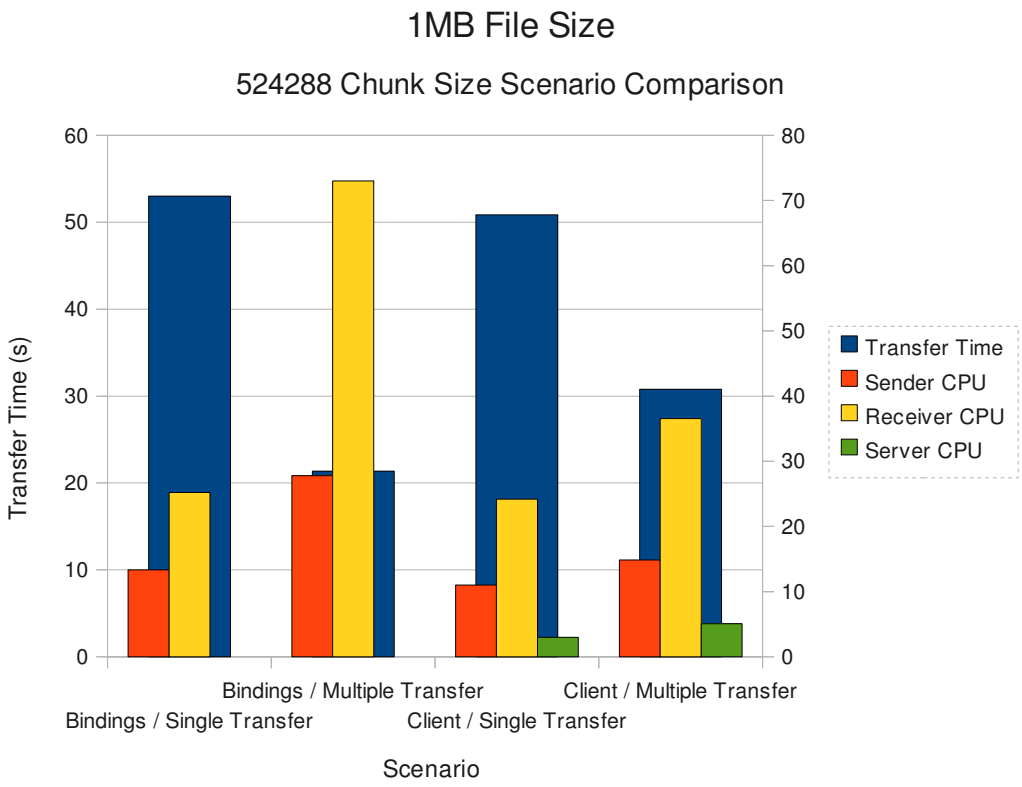


Figure 16

The table and graphs above highlight the benefit of splitting a single large transfer out into multiple smaller transfers. The trade off of high CPU utilisation gains a decrease in transfer time by over 40% in some cases. By using bindings connectivity, rather then client, when using multiple simultaneous transfers there is another increase in agent performance. For a single transfer there is relatively little difference between bindings and client.

2.4 - 10MB Scenario Comparison

The following tables and figures show the difference in transfer time and CPU utilisation by scenario when using the 10MB file size.

ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
65536	Bindings / Single Transfer	47.09	14.99	27.53	N/A
65536	Bindings / Multiple Transfer	20.23	31.88	63.07	N/A
65536	Client / Single Transfer	73.2	7.25	20.04	2.14
65536	Client / Multiple Transfer	39.61	11.24	35.71	4.88
ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
131072	Bindings / Single Transfer	48.91	12.13	25.1	N/A
131072	Bindings / Multiple Transfer	18.21	30.14	61.78	N/A
131072	Client / Single Transfer	47.31	8.67	22.69	3.05
131072	Client / Multiple Transfer	34.52	12.39	32.05	4.94
ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
262144	Bindings / Single Transfer	43.54	15.92	25.7	N/A
262144	Bindings / Multiple Transfer	18.28	28.19	56.98	N/A
262144	Client / Single Transfer	48.54	8.15	22.19	3
262144	Client / Multiple Transfer	30.07	14.25	36.36	5.11
ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
524288	Bindings / Single Transfer	46.38	13.36	25.21	N/A
524288	Bindings / Multiple Transfer	19.04	27.52	74.53	N/A
524288	Client / Single Transfer	45.3	8.72	22.63	3.05
524288	Client / Multiple Transfer	29.22	17.4	35.18	5.61

Table 5

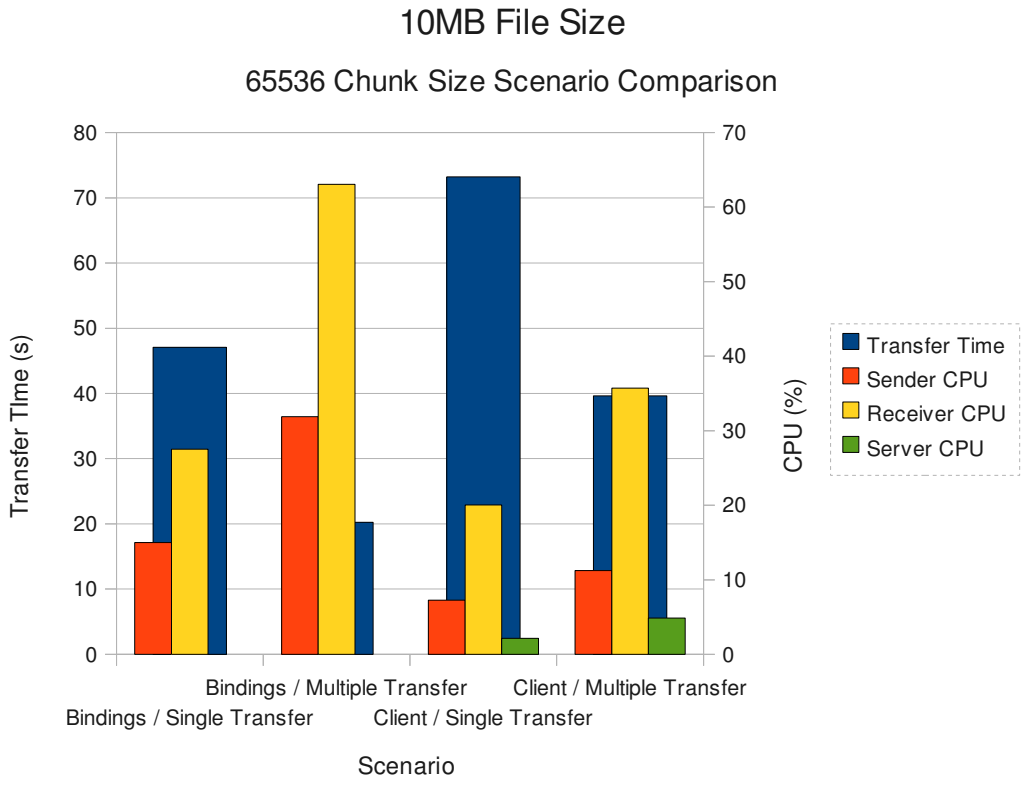


Figure 17

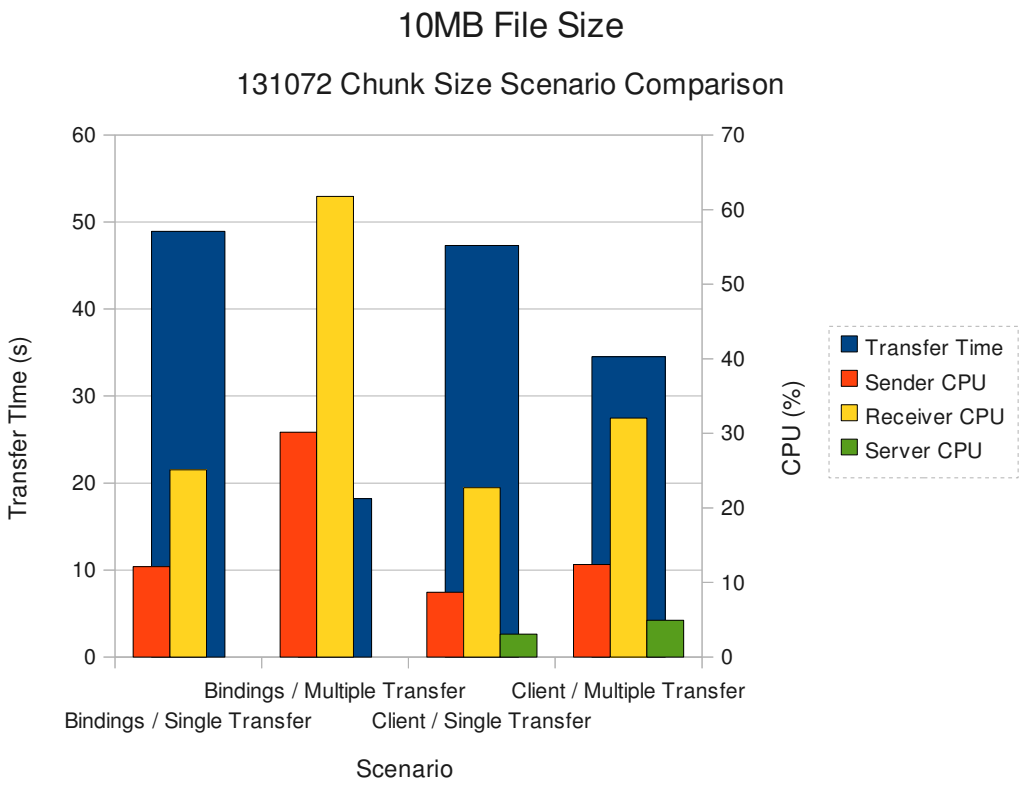


Figure 18

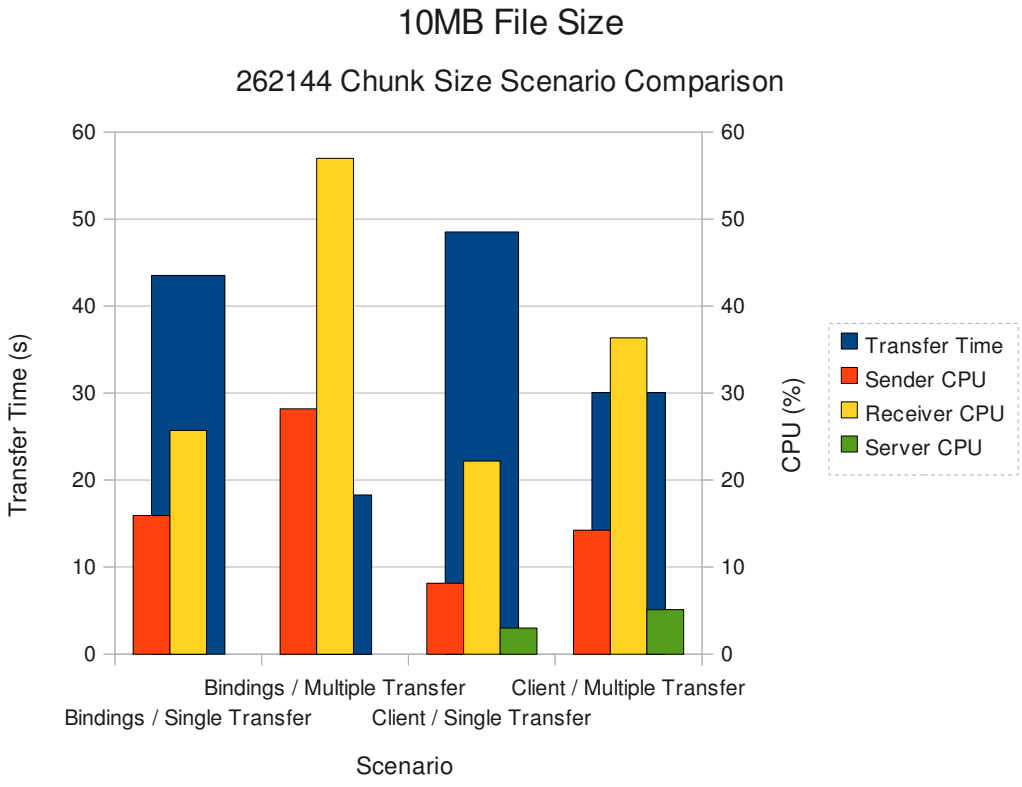


Figure 19

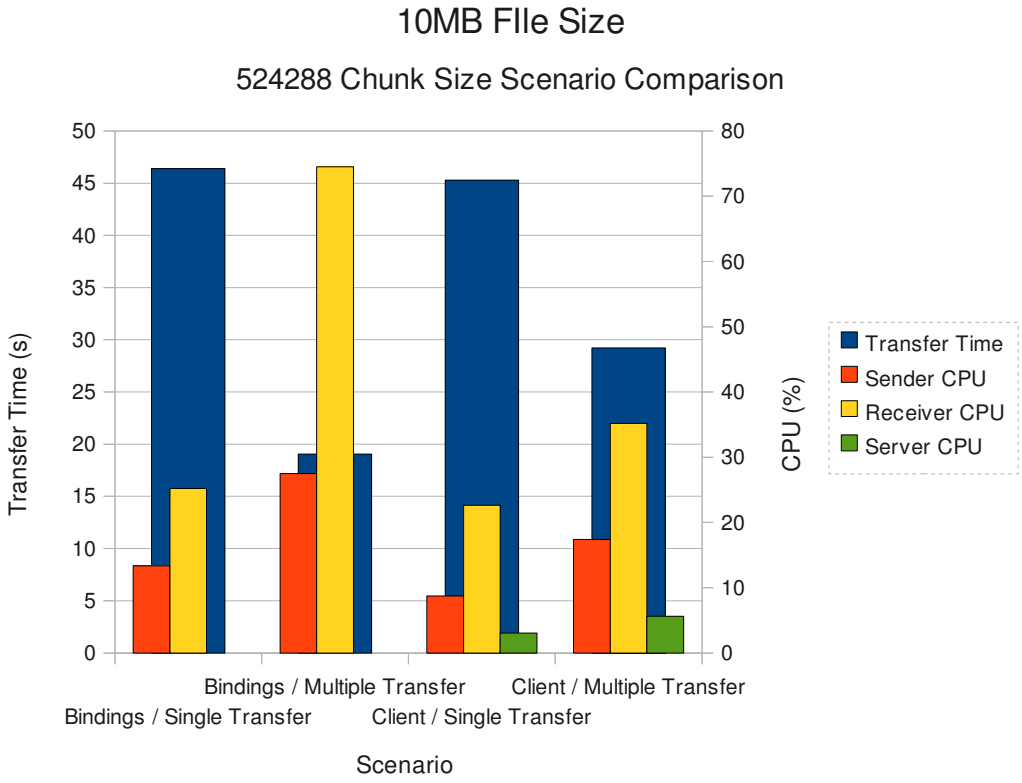


Figure 20

These results confirm the trend that using multiple transfer in bindings mode is the quickest when using any chunk size. Note that the biggest impact that the chunk size has is on receiver CPU utilisation, varying by around 16%, whilst the transfer time varies by less then 2 seconds.

2.4 - 100MB Scenario Comparison

The following tables and figures show the difference in transfer time and CPU utilisation by scenario when using the 10MB file size.

ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
65536	Bindings / Single Transfer	48.63	12.51	25.58	N/A
65536	Bindings / Multiple Transfer	20.31	31.05	64.02	N/A
65536	Client / Single Transfer	75.93	5.87	18.44	2.03
65536	Client / Multiple Transfer	38.21	11.65	36.24	4.98
ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
131072	Bindings / Single Transfer	47.43	11.36	25.04	N/A
131072	Bindings / Multiple Transfer	16.57	32.74	67.22	N/A
131072	Client / Single Transfer	49.18	7.48	21.11	3.02
131072	Client / Multiple Transfer	34.68	12.04	32.73	4.88
ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
262144	Bindings / Single Transfer	45.1	11.63	24.95	N/A
262144	Bindings / Multiple Transfer	16.52	30.93	63.63	N/A
262144	Client / Single Transfer	47.9	8.09	22.38	2.99
262144	Client / Multiple Transfer	31.23	13.42	33.17	3.01
ChunkSize	Scenario	Transfer Time	Sender CPU	Receiver CPU	Server CPU
524288	Bindings / Single Transfer	46.76	10.97	23.56	N/A
524288	Bindings / Multiple Transfer	19.23	28.2	71.77	N/A
524288	Client / Single Transfer	45.3	8.24	22.27	3.09
524288	Client / Multiple Transfer	29.8	16.61	35.64	5.21

Table 6

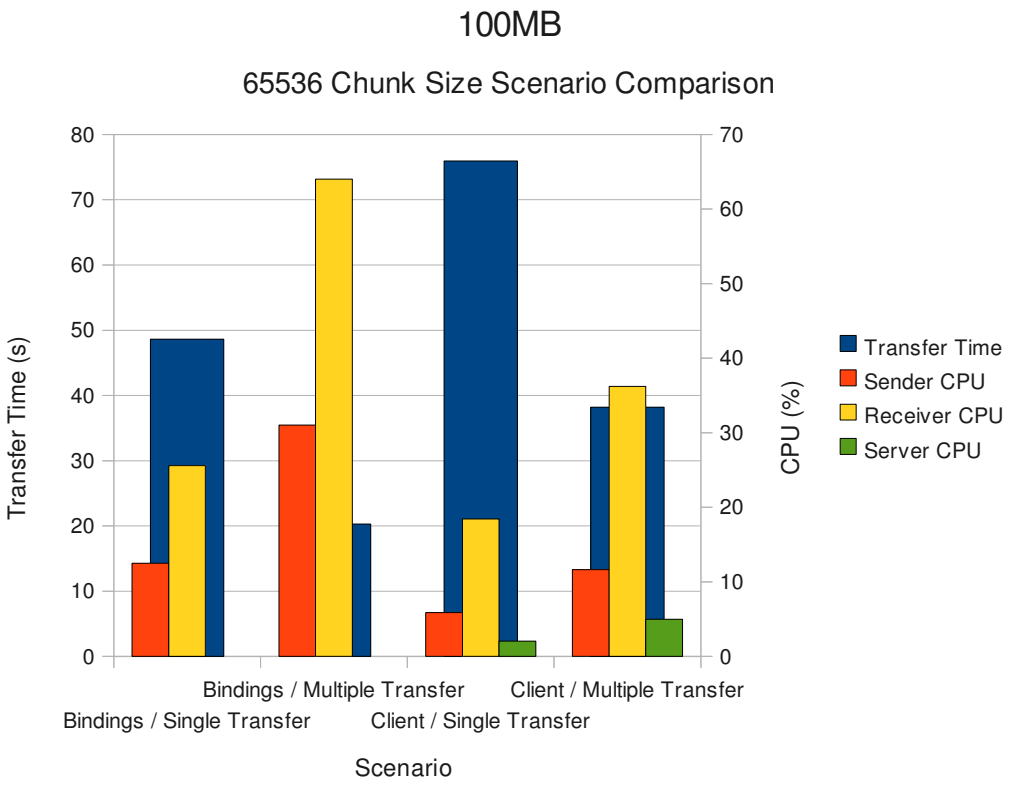


Figure 21

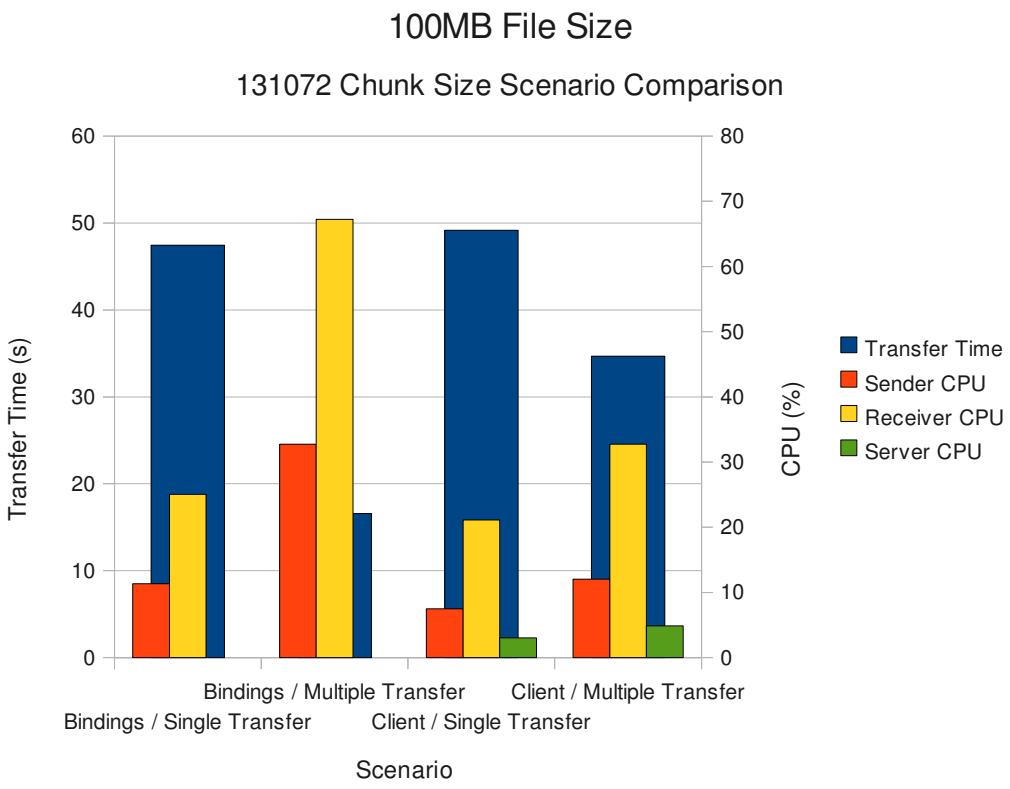


Figure 22

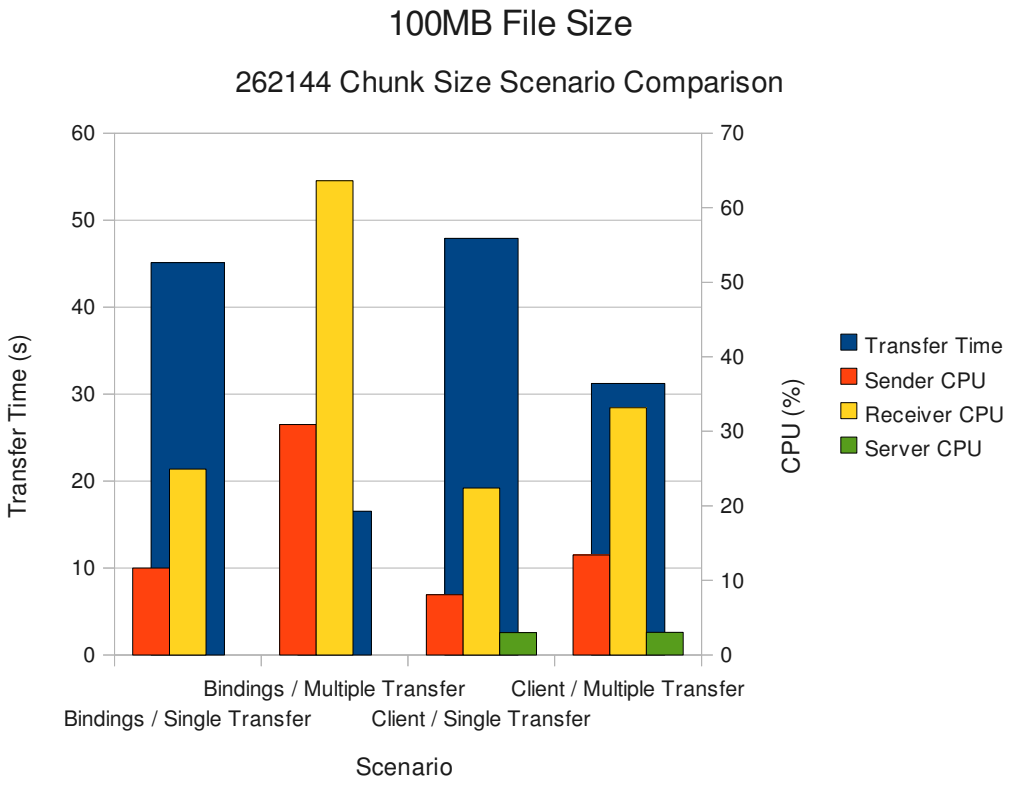


Figure 23

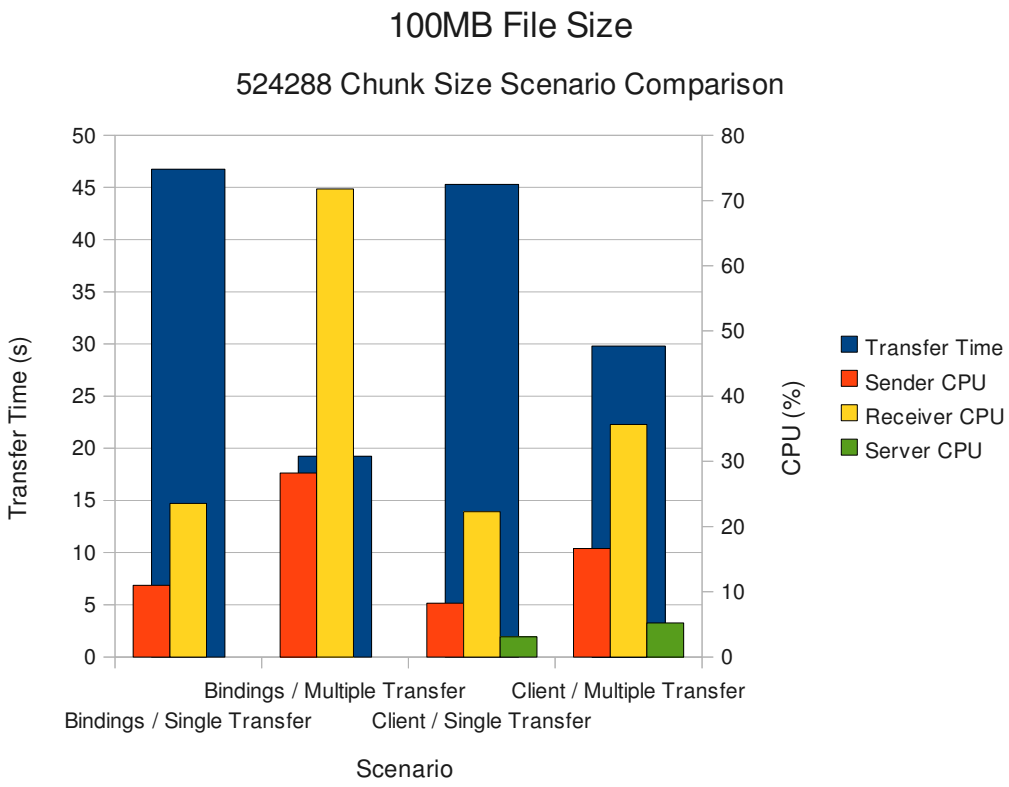


Figure 24

Yet again these results show that using multiple transfers in place of a single transfer is the best way to maximise throughput. Of note within these results is the increased CPU utilisation using 524288 chunk size, compared with the 262144 chunk size. The higher CPU does not match with a quicker transfer, suggesting that 262144 is the optimal chunk size for this scenario.

As with previous file sizes, there is very little difference between the client and bindings, single transfer scenarios.

3 Tuning Recommendations

3.1 - WebSphere MQ Setup

Readers of this performance guide should make themselves familiar with the WebSphere MQ Performance Supportpacs that are continually released. They can be found here: http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg27007197#1. Of particular interest for Solaris are Supportpacs MP6P for MQ Version 7 and MP6M for MQ Version 6.

For this performance report, advice was taken from the aforementioned (MP6M) and applied to the queue managers created accordingly. Queue managers were created using the following *crtmqm* command:

crtmqm -q -u SYSTEM.DEAD.LETTER.QUEUE -lp 16 -lf 16384 <QueueManagerName>

Once the queue manager was created, tuning parameters were added to the queue managers' qm.ini as follows:

Channels:
MQIBindType=FASTPATH

TuningParameters:
DefaultPQBufferSize=1045876
DefaultQBufferSize=1048576

Note that the qm.ini was updated before the queue manager was started (and therefore before the WebSphere MQ Managed File Transfer objects were created).

By increasing the amount of memory available to queues for persistent and non-persistent messages, you can help to avoid writing messages out to disk unnecessarily. Turning on FASTPATH for channels removes the channel process, and enables the channel to run within the main queue manager process. Please consult your documentation to understand what this means for your WebSphere MQ installation.

For more information on tuning a WebSphere MQ queue manager, please refer to the Supportpacs mentioned above.

3.2 - WebSphere MQ File Transfer Edition Setup

When running agents for this performance report, the following environment property was used:

export FTE_JVM_PROPERTIES=”-Xmx2048M -Xms2048M”

This property was set before starting an agent and sets the starting and maximum JVM heapsize to be 2GB. These values were used to ensure that the agent had sufficient memory to allocate when running the multiple transfer scenarios.

As demonstrated in the results, altering the *agentChunkSize* can have a significant impact on both CPU utilisation and transfer time. There is another property *agentWindowSize* that can be used to control the amount of syncpoints commited, and the number of acknowledgements sent between two agents when transferring files. This property has a default value of 10. This means that for every 10 chunks of data sent over WebSphere MQ, the sending agent will take an internal checkpoint, and wait to receive an acknowledgement from the receiving agent before sending more data. The property's default value was determined after extensive performance work during the development of version 7.0.1. Increasing this property increases the amount of data that could potentially need to be re-transmitted if a recovery is required, and is not recommended for unreliable networks.

3.3 – WebSphere MQ File Transfer Edition Transfer Recommendations

The following are a list of bullet pointed recommendations when planning your WebSphere MQ File Transfer Edition network.

- Send large numbers of files over multiple transfers, rather then a single large transfer. This will increase the efficiency of the I/O involved in transferring the files, which will ultimately decrease the transfer time.

- Where possible use Bindings connectivity for agents. If using Client connectivity ensure that multiple transfers are used for maximum efficiency.

- Test your typical transfers using a range of agentChunkSize parameters. Depending on the underlying hardware, you may find an optimum value for your setup.

- Multiple smaller files place the agent under strain due to the operating system open/close costs associated with more files. Where possible configure your file creation processes to generate archives of smaller files, enabling FTE to use less open/close calls.

- Reading and writing to physical disk is often going to be the performance bottleneck. For agents that will see a large number of incoming, and outgoing transfers it would be best if high performance disks were used to read data from and write data to.

- When configuring your MQ network, use the appropriate WebSphere MQ Performance Report to apply optimal settings for your platform.

- Ensure that you have sufficient RAM for your Agent. The performance tests used 2GB of RAM, it is recommended that you read your Operating System guide on memory usage and plan accordingly.

4 - Measurement Environment

4.1 – WebSphere MQ File Transfer Edition Agents

- WebSphere MQ File Transfer Edition Version 7.0.1 was used for this report.
- Agents connected to WebSphere MQ Version 6.0.2.7 queue managers.
- Default properties were used for agents, except for *agentChunkSize*
- Agents were reading/writing files to the local file system, not the SAN.

4.2 - WebSphere MQ

- WebSphere MQ Version 7.0.0.1 was used for the coordination queue manager
- WebSphere MQ Version 6.0.2.7 was used for agent queue managers
- Queue managers created in accordance with Performance report
- /var/mqm and /var/mqm/log were mounted on SAN disks

4.3 - Operating System

- Solaris 5.10. Fully up to date on all security patches.

4.4 - Hardware

Sun T2000:	mqperfs2
Model:	T20|108C-64GA2G SFT2000
Processor:	1.4GHz
Architecture:	8core
Memory (RAM):	64Gb
Disk:	Internal disks for file measurements
	Fibre channel HBA PCI-X 4Gb FC Dual Port HBA
	2 SAN disks on DS6000 (5Gb each, 1 queue, 1 log)
Network:	1Gbit Ethernet Adapter (onboard)

Sun V490:	mqperfs3, mqperfs4
Model:	
Processor:	SPARCV9 @ 1500 HMz
Architecture	4 CPU
Memory (RAM):	32Gb
Disk:	Internal disks for file measurements
	Fibre channel HBA PCI-X 4Gb FC Dual Port HBA)
	SAN with 2 partitions of 5Gb each
Network:	1Gbit Ethernet Adapter (onboard)

The SAN consists of a pair of 2026 model 432 (McDATA ES-4700) switches running at 4Gb/s with 32 ports each. They are connected together via two inter-switch links to form a single SAN fabric.

The MQ hosts attach via this SAN to a DS6800 disk array (1750 model 511) with one expansion drawer.

Each drawer (controller + expansion) contains 16 x 73Gb 15K fibre channel disk drives, so there are a total of 32 physical drives.

The 32 drives are configured as four RAID-5 arrays, each of which is 6+Parity+Spare (the number of spares is defined by the configuration of the DS6800).

The controller has an effective cache size of 2.6Gb plus 0.3Gb of NVS