CONFIGURATION MANAGEMENT® INC

Whitepaper

Whitepaper

ClearCase Build Performance Degradation

Technical Report

Managing Change for Growing Companies

ClearCase Build Performance Degradation: Technical Report
by
Patrick Dugal

Last Updated

28 July 2004

Restricted Rights

Abstract

This whitepaper presents metrics, a root cause analysis, and resolution strategies regarding the issues surrounding relative build performance of ClearCase dynamic views with respect to the two main underlying data access networking technologies that ClearCase supports on the Microsoft Windows platforms: NFS and CIFS/SMB. The solutions used were NFS Maestro and TAS, but much of the information presented may be generally applicable to other ClearCase environments that use Samba, Windows Servers, NAS Filers, or other access solutions.

For more information, questions or comments regarding the contents of this report please contact:

Configuration Management, Inc.
140 Broad Street
Red Bank, NJ 07701

Telephone: (800) 550-5058
http://www.cmi.com

Contents

List of Tables

References

Ref #	Title	Author / Owner
1	Rational User Conference Training 1999: *Introduction to ClearCase Architecture & Performance Concepts for ClearCase*	Rational Software
2	Internetworking with TCP/IP (Vol. 1), Fourth edition	Douglas E. Comer
3	Samba Unleashed	Litt et al.
4	CCIUG Mailing List -- http://www.rational.com/support/usergroups/index.jsp	Rational Corporation
5	Rational ClearCase – Administrator's Guide, Version 2000.05.00 and later, UNIX/Windows Edition	Rational Corporation
6	Microsoft Windows NT Workstation Resource Kit	Microsoft Corporation
7	ClearCase 2002.05.00 (5.0) Supported Network Access and Storage Platforms http://www.rational.com/support/documentation/release/cc_storage_network.jsp	Rational Corporation

1. Executive Summary

1.1. Purpose

The mission of this whitepaper is to inform others in the global software development community about a root-cause analysis into a build performance issue in ClearCase, conducted at one of CMI's customers in the financial sector, in the hopes of driving solutions that are to the best advantage of organizations the world over.

1.2. Scope

Business Issue

In many organizations, TAS and Samba are seen as superior solutions with ClearCase because of their advantages over other access solutions, chiefly: lower maintenance, lower licensing costs, and an improved security/authentication model that is transparent to the user. In the enterprise ClearCase environment (versions 4.x & 5.0) at one of CMI's customers in the financial sector, a migration from NFS Maestro to TAS was piloted with the intention of simplifying the environment, thereby reducing costs.

Shortly after deployment commenced, early adopters of TAS reported that designer test builds in ClearCase dynamic views completed in over 1560 seconds, which did not meet expectations. The problem was that development teams were used to builds consistently completing in 360 seconds in the NFS Maestro environment. After eliminating all possible sources of poor performance (i.e., virus scanner, network drive in PATH, etc.), the build consistently completed in approximately 900 seconds – 2.5 times longer than expected – and that led the migration project to be put into question.

The problem became a gating issue which impeded the migration and served as the primary impetus for a subsequent investigation into the issue (summarized below). In short, users needed build times in the TAS environment to be comparable to those observed in the NFS Maestro environment.

Business Impact

Procuring additional high-performance hardware resources – which require more effort to support – may become necessary to mitigate the impacts of the issue. There is industry evidence to support the estimated impact of the issue: the slowdown cost is probably in the order of 10s of 1000s of dollars in extra equipment costs to try to get around the problem, and probably 10s of 1000s of dollars of lost programmer productivity as users sit around

waiting for builds. For large-scale companies with many concurrent ClearCase users, costs could be as high as in the millions per year.

1.3. Target Audience

The target audience of this document may include users and managers of the enterprise ClearCase infrastructure who have a vested interest in build performance in their organization, and those involved in resolving the issue in the IBM Rational organization. Target organizations may include Nortel Networks, Motorola, Lucent, Ericsson, HP, CCIUG, cmcrossroads.com, cmtoday.com, and others in the global IT community.

1.4. Results of Investigation

Consequently, extensive research was conducted on the issue affecting the performance of a typical Java build executed on a machine running the Windows NT/2000/XP operating system and accessing ClearCase through CIFS/SMB, instead of NFS. Through in-depth research, the following was discovered:

1. There is a significant difference in the total duration of builds, depending primarily on:
   a) The redirector the Windows client is configured to use (i.e. NFS, CIFS/SMB)
   b) Network latency (i.e. average datagram round-trip-time delay)
   c) How the build tools interact with the client O/S under the hood

2. The main bottleneck in the present case study was the sheer number of time-consuming, over-the-wire CIFS/SMB calls of type QUERY_FILE_INFORMATION (220,000+ calls were observed)

3. Assuming the over-the-wire calls are fundamentally unnecessary and can be averted (using the same avoidance principle applied to the equivalent GETATTR calls observed with NFS), build performance could be improved by anywhere between 10% and a factor of 5

4. The build performance issue likely globally affects many other organizations who execute builds in dynamic views on Windows, in both NT/UNIX interop and Windows homogeneous environments

5. Not all builds will show such significant symptomatic signs of slowness, but under the hood, there may be an alarming amount of time spent on the aforementioned calls; the extent of the slowdown depends much on the factors in point 1 (above)

1.5. Request for Enhancement Submitted to IBM Rational

In addition, several teleconferences were held regarding the issue considered to be a request for enhancement (RFE #RATLC00687665). In response to the problem report, IBM Rational:

1. Acknowledges the issue and has spent a significant amount of time researching how to improve ClearCase performance with CIFS/SMB, however, IBM Rational has not implemented a solution in this regard to date

2. May consider implementing a solution in a future release of ClearCase as the status of the RFE remains OPEN

3. Informed customers present about the RFE process and other related workarounds that may slightly improve performance

4. Understands that customers would require more specific information in order to ascertain whether the calls can in fact be safely removed, however, the IBM Rational Software teams aware of the issue have elected not to provide further details about MVFS internals to customers at this time as the matter is under investigation

1.6. Recommendations

This section lists recommendations that should be communicated to all development teams who typically perform frequent builds inside ClearCase dynamic views on systems running the Windows operating system for software development purposes and who may be suffering from poor performance. Both locally and remotely stored dynamic views in homogeneous Windows and NT/UNIX interop environments – including Samba environments – are affected by the issue. Thus, the following recommendations to consider are applicable to all ClearCase environments that employ CIFS/SMB:

R1. **Build software in locally-stored *snapshot views*** instead of dynamic views on Windows platforms. Become familiar with the caveats and drawbacks associated with snapshot views (i.e. lack of support for VOB symbolic links, etc.)

R2. O**ptimize the build engine, build process, and associated build environment** to reduce excessive application turns over the network.

R3. **Reduce network round-trip-time delays** between the client and server through network upgrades and closer physical proximity of ClearCase servers and clients.

R4. **Deploy ClearCase builds on a UNIX or Linux host** instead of on PCs running Windows (Sun Solaris particularly for Java development). A shared development

UNIX pool configuration or individual development UNIX machines assigned to individuals for builds may be appropriate.

The above work-around recommendations may not answer the needs of some specific teams in other organizations and further analysis into performance may be required. More in-depth information on all the above recommendations is available in the full report.

1.7. Contact Information

For more information, questions or comments regarding the contents of this report please contact:

Configuration Management, Inc.
140 Broad Street
Red Bank, NJ 07701

Telephone: (800) 550-5058
http://www.cmi.com

1.8. Definitions, Abbreviations and Acronyms

The following terms or acronyms will be used throughout the document and are defined here for the convenience of the reader:

Acronym/ Abbreviation	Term	Definition
	Build	The transformation of source codes into executable software products or intermediate machine codes that are further used to create executable software products.
CC	**ClearCase**	An automated software library tool used for implementing commonly used configuration management activities such as configuration identification, identifying project baselines and configuration control. This tool is available in the LINUX/UNIX/Windows (NT/2K/XP) OS platforms.
CCIUG	**ClearCase International Users Group**	Rational provides a global mailing list to ClearCase customers who wish to join. The subject of this list is IBM-Rational ClearCase and configuration management. The following link will take you to the sign-up page: http://www.rational.com/support/usergroups/index.jsp
client	**ClearCase Client**	ClearCase software architecture is based on the client/server model of interaction. The client refers to the type of software that executes on the computer that the user interacts with (e.g., checkin, checkout) to perform work inside ClearCase. The server refers to the type of software that executes on server-class machines which fulfill requests that client computers make through network communication. Often, the host or computer that runs either type of software is commonly referred to as a “client” or a “server”.
CM	**Configuration Management**	The management of a software design as it evolves into a software product or system. IEEE defines configuration management as the discipline of applying technical and administrative direction and surveillance to identify and document the physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements.

	Defect	1. Any unintended characteristic that impairs the utility or worth of an item. 2. Any kind of shortcoming, imperfection or deficiency. 3. Any flaw or imperfection in a software work product or software process. Examples of defects include: • Mistakes • Omissions and imperfections in software artifacts. • Faults contained in software sufficiently mature for test or operation.
	Dynamic View	A view that is always current with the versioned object base (as specified by the config spec). Dynamic views use the MVFS to create and maintain a directory tree that contains versions of versioned object base elements and view-private files. Dynamic views are not supported on all ClearCase platforms.
	Label	A naming convention used within a configuration management repository to mark revisions of files and directories that are of particular interest to the development process. Labels can be used to identify developmental, build and release versions of configuration items.
MVFS	**Multi-Version File System**	A directory tree that, when activated (mounted as a file system of type MVFS), implements a versioned object base. To standard operating system commands, a versioned object base appears to contain a directory hierarchy; ClearCase commands can also access the versioned object base's meta data. Also, MVFS refers to a file system extension to the operating system, which provides access to versioned object base data. MVFS is not supported on all ClearCase platforms.
NFS	**Network File System**	Native protocol that UNIX platforms use for network file system access. ClearCase Windows client computers can communicate with UNIX servers that run NFS services to allow access to ClearCase server file systems. However, every Windows client computer requires an additional piece of software called “NFS Maestro” in order to access ClearCase file systems transparently through NFS.
Oplock	**Opportunistic Lock**	An *opportunistic lock* is a lock placed by a client on a file residing on a server. In most cases, a client

		requests an opportunistic lock so it can cache data locally, thus reducing network traffic and improving apparent response time. Oplocks are used by network redirectors on clients with remote servers, as well as by client applications on local servers. For more information, see the following definition: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/fileio/base/opportunistic_locks.asp
	Software	The entire set of programs, procedures and related documentation associated with a system, especially a computer system.
CIFS/SMB	**Common Internet File System / Server Message Block**	Native protocol that Windows computers use for network file system access. SMB servers (i.e. TAS, Samba) that run on UNIX computers can simplify the implementation of cross-platform file system access for ClearCase users.
	View	A ClearCase object that provides a work area for one or more users. For each element in a versioned object base, a view's config spec selects one version from the element's version tree. Each view can also store view-private files and view-private directories, which do not appear in other views. There are two kinds of views: snapshot views and dynamic views.
	Samba	Samba is an Open Source/Free Software suite that provides seamless file and print services to SMB/CIFS clients. Samba is freely available under the GNU General Public License.
TAS	**TotalNet Advanced Server**	Vendor supported SMB-based network file sharing service which runs on UNIX platforms to allow Windows computers to transparently access ClearCase data. It is one of the recommended interop solutions for use with ClearCase.

Table 1.4-1: Definitions, Abbreviations and Acronyms

2. Build Environment

2.1. Software Releases and Settings

The following table presents details about the environment involved in the Java build.

Component	Release/Setting
TAS version	7.0 patch 1
NFS Maestro Solo version	7.1.1
Windows	XP Pro, patch 1
ClearCase	2002.05.00 patch 15
MVFS mnodes values	800 (max for free list and cleartext list)
MVFS scaling factor (largeinit)	5
Dynamic view cache size	4m
Build software	ant, sun jdk, aspectj, cmd.exe (standard Windows shell)
Number of hops to VOB server	8
VOB server ping time	~2.2 ms
TCP Window Size	Windows XP default value unless specified
Network bandwidth to server	100Mbps / full duplex
Client proximity from VOB server	>20 mi

Table 2.1-1: Software Releases and Settings in Build Environment

2.2. Suspected Sources of Poor Performance Eliminated

On the Windows ClearCase client, the following items which could be causing excessive delays have been disabled/removed from the picture:

- Virus scanner software
- VPN software
- Entrust software
- Network drives from PATH variable
- Firewall software
- Clearmake/omake, express builds, and winkins not used
- No labeling or other cleartool operations occur during build
- Other intensive processes not build-related on client and server
- Class path with too many directories
- No UNIX emulation tools (i.e. Cygwin, Uwin, MKS, or others) are used.

2.3. Build Process

The build first does a standard "clean" operation (removing previously built objects if there are any). Next, it constructs the multiple objects (i.e. byte code .class files) from the source code using *ant* and the Sun JDK, which is one of the operations that takes a significant portion of the total build time. There is a single invocation of the java compiler which contains a large list of .java files (maybe all) in order to avoid having to manage relationships between classes in, say a makefile, for example. Then *AspectJ* is run for a little over half of the build time, and the last step includes some quick *tar*'ing and copying of results files and library files from the view/VOB into a local directory (i.e. C: drive). The clean step and the last step (i.e., tar and copy) take probably less than 10% of the total build time. The actual compiling of java classes and dependency checking that *Javac* is doing takes less time than *AspectJ* does. Again, no ClearCase operations such as labeling, wink-in, auditing, or other such processes are taking place. Omake/clearmake is not used during the Java build. A local or snapshot build takes ~2.5 minutes whereas a build inside ClearCase (NFS Maestro) can take ~6 minutes, and a build in the TAS environment can take ~17 minutes or more.

2.4. How Data was Captured

Network data was collected using a protocol analyzer tool (also known as a sniffer). The tool (Ethereal) is freely available on the Web (http://www.ethereal.com) and can be used to investigate and solve a variety of problems, including performance. There is excellent documentation on the web site about Ethereal (the GUI version) and Tethereal (the text version which is very similar to Sun's snoop). See http://www.ethereal.com/tethereal.1.html.

Please note that running tools like sniffers can be a security risk, so the appropriate approvals and agreements needed to be obtained. It is not recommend others to use sniffers without written prior authorization or without having some basic knowledge of networking protocols.

To install Ethereal on the Windows client, you need to download and run two files: the WinPcap 3.0 from http://winpcap.polito.it/install/bin/WinPcap_3_0.exe and the installer for the Ethereal Win32 package (which includes Tethereal and documentation and more) from http://www.ethereal.com/distribution/win32/ethereal-setup-0.9.11.exe.

Keep in mind that the GUI version can slow executing processes down significantly during a large build. It is therefore not recommended for capturing a large amount of traffic, but it can be useful for viewing individual packets. To capture data using the GUI, click on the menu bar option Capture → Start, ensure the correct interface is being used from the drop down menu and enter the filter string in the filter text box:

```
host client_hostname
```

where client_hostname is the TCP/IP host name or IP address of the client machine's network interface; this limits frames sent by or received by the client to be captured, which is pretty much all you should be interested in seeing unless you have other network problems (i.e. to see broadcasts) in which case you leave the "filter" field blank. All other defaults can be accepted. Click OK to start capturing. Next, launch your build in a timely fashion (it helps to have the build environment set up and the command to execute ready before starting

the capture). In contrast, at the command line, the command to start capturing data can be as simple as the following (except if you have multiple interfaces – you may need to pass more options):

```
Tethereal –w foo.cap
```

Tethreal does not seem to slow processes down like the GUI version does, so it is highly recommended for capturing large amounts of traffic. When the build is done, hit CTRL + C in the cmd.exe command shell window.

To obtain the SMB RTT Statistics, you should use Ethereal version 0.9.11 or later. The version is important otherwise you may not see the SMB option in the menu. From the GUI: in the menu bar, click on Tools → Statistics → RTT. At the command line, you need to provide a capture file (.cap) to read and then redirect output to a file, for example:

```
Tethereal -z smb,rtt –r foo.cap > c:\foo.txt
```

The SMB RTT statistics will be appended to the end of the .txt file. Go to the tail of the file and you will see statistics similar to what you see in the Appendices of this document. You can also get the TCP statistics, for example:

```
Tethereal -z io,users,ip -r foo.cap
```

It is left up to the reader to experiment with various options to get more familiar with the GUI menu options and the command line tool. Although some GUI screen captures may present well in a document, the text based tool (Tethereal) is by far more powerful and seemingly more efficient. In retrospect, because of the fuzziness involved with the GUI snapshots, perhaps text should have been used in the section Observations from Ethereal Captures to improve the clarity of the statistics.

Network packet captures were done on the same client host and the same server VOB server was used throughout both environments. Thus, most factors outside ClearCase software – such as network delay, CPU power, network bandwidth – were controlled and as such remained consistent across interop environments.

To switch between the NFS Maestro and the TAS environments, only the NFS Maestro client software needed to be either installed or uninstalled because the VOB server is running both the TAS server and the PCNFSD server processes. The install settings chosen were the default choices. NFS Maestro's parmset was not run to optimize performance, so the default of 32K packet size and 8 threads was chosen.

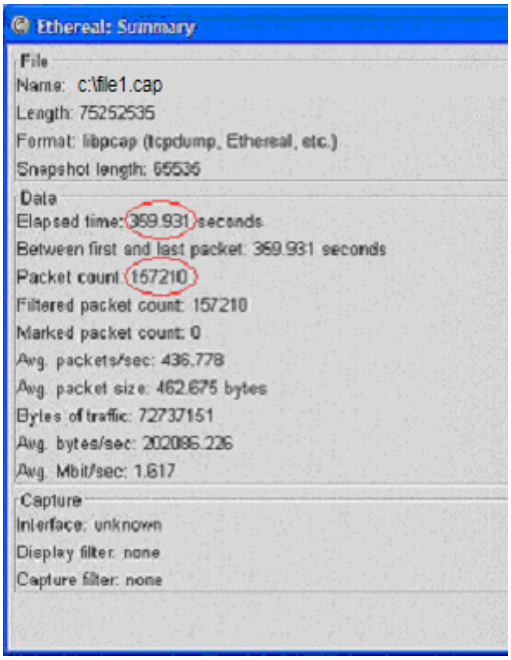
File system call data was gathered using a freely available tool called NTFilemon which was downloaded from the website http://www.sysinternals.com. The version used was 5.01. Unfortunately, this tool seems limited in that it only captures data at higher levels and does not capture calls when performing builds in the NFS Maestro environment. There may be options to enable captures of different levels of calls, but this was not pursued in this analysis.

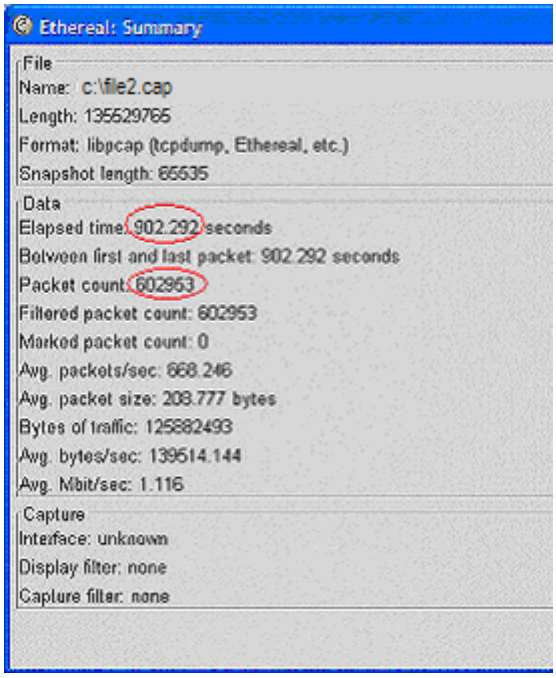
MVFS cache statistics were not gathered because it was felt that everything obvious had already been done to improve MVFS caching (i.e. setting the scaling factor (largeinit) to 5, which is the maximum). We also set the view cache size to 4m, but increasing or decreasing any cache sizes did not improve build times observed.

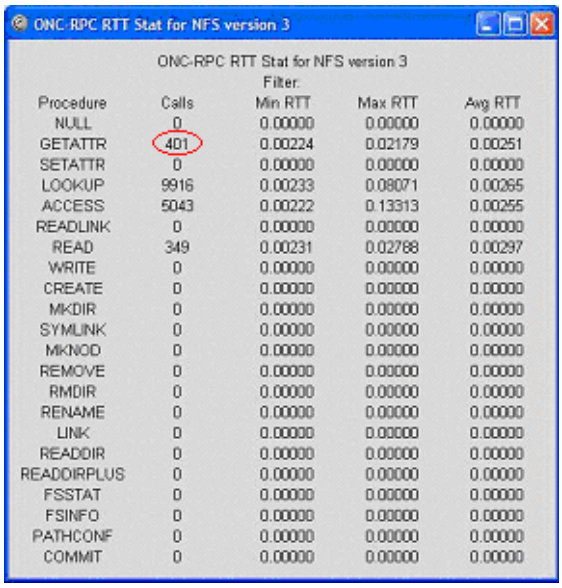
3. Observations from Ethereal Captures

The following snapshots of windows were generated by tools internal to Ethereal and their size has been adjusted for clarity and readability. In the Analysis section, the implications of the data circled in **red** or highlighted in **blue** are discussed.

Build in NFS Maestro environment	**Build in TAS environment**



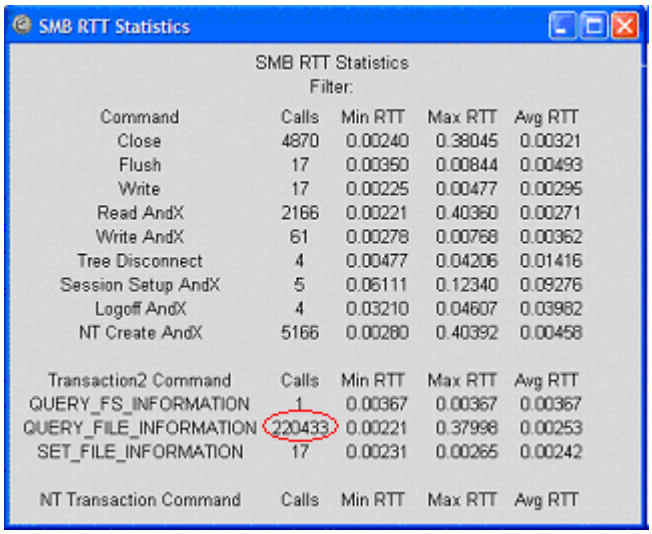




ONC-RPC RTT Stat for NFS version 3

ONC-RPC RTT Stat for NFS version 3
Filter:

Procedure	Calls	Min RTT	Max RTT	Avg RTT
NULL	0	0.00000	0.00000	0.00000
GETATTR	401	0.00224	0.02179	0.00251
SETATTR	0	0.00000	0.00000	0.00000
LOOKUP	9916	0.00233	0.08071	0.00265
ACCESS	5043	0.00222	0.13313	0.00255
READLINK	0	0.00000	0.00000	0.00000
READ	349	0.00231	0.02788	0.00297
WRITE	0	0.00000	0.00000	0.00000
CREATE	0	0.00000	0.00000	0.00000
MKDIR	0	0.00000	0.00000	0.00000
SYMLINK	0	0.00000	0.00000	0.00000
MKNOD	0	0.00000	0.00000	0.00000
REMOVE	0	0.00000	0.00000	0.00000
RMDIR	0	0.00000	0.00000	0.00000
RENAME	0	0.00000	0.00000	0.00000
LINK	0	0.00000	0.00000	0.00000
READDIR	0	0.00000	0.00000	0.00000
READDIRPLUS	0	0.00000	0.00000	0.00000
FSSTAT	0	0.00000	0.00000	0.00000
FSINFO	0	0.00000	0.00000	0.00000
PATHCONF	0	0.00000	0.00000	0.00000
COMMIT	0	0.00000	0.00000	0.00000



SMB RTT Statistics

SMB RTT Statistics
Filter:

Command	Calls	Min RTT	Max RTT	Avg RTT
Close	4870	0.00240	0.38045	0.00321
Flush	17	0.00350	0.00844	0.00493
Write	17	0.00225	0.00477	0.00295
Read AndX	2166	0.00221	0.40360	0.00271
Write AndX	61	0.00278	0.00768	0.00362
Tree Disconnect	4	0.00477	0.04206	0.01416
Session Setup AndX	5	0.06111	0.12340	0.09276
Logoff AndX	4	0.03210	0.04607	0.03982
NT Create AndX	5166	0.00280	0.40392	0.00458

Transaction2 Command	Calls	Min RTT	Max RTT	Avg RTT
QUERY_FS_INFORMATION	1	0.00367	0.00367	0.00367
QUERY_FILE_INFORMATION	220433	0.00221	0.37938	0.00253
SET_FILE_INFORMATION	17	0.00231	0.00265	0.00242

NT Transaction Command	Calls	Min RTT	Max RTT	Avg RTT

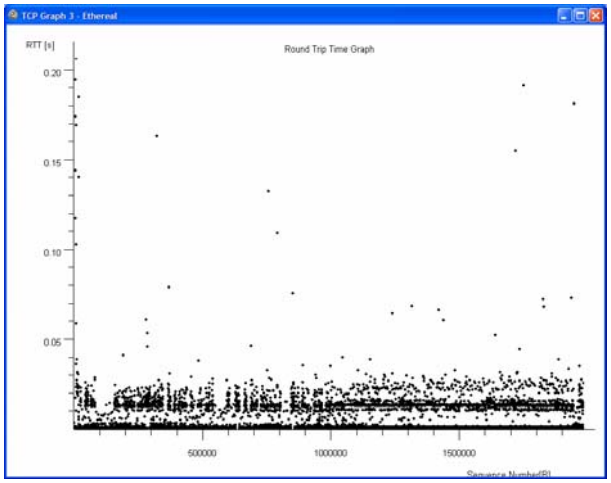
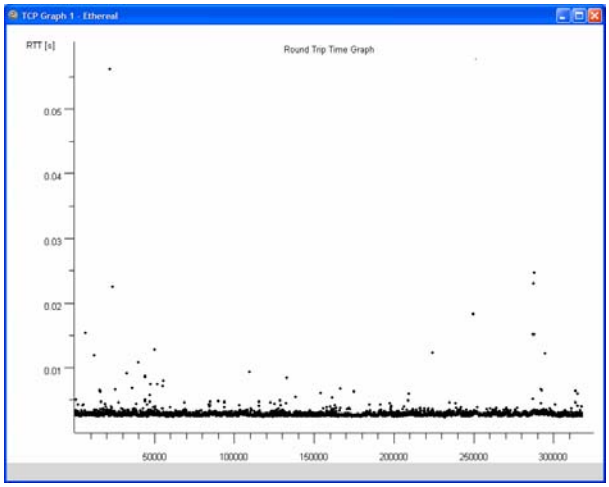
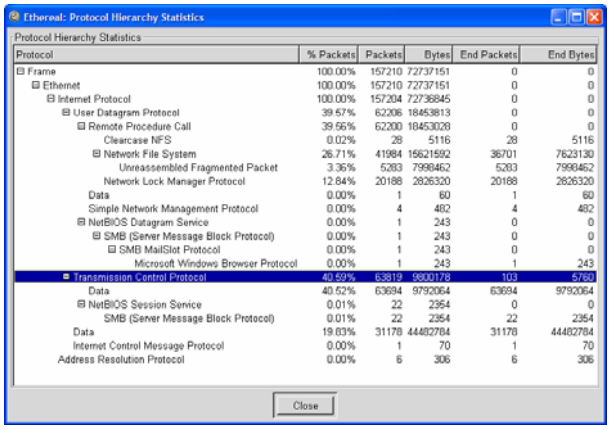
<table>
<tr><th>Build in NFS Maestro environment</th><th>Build in TAS environment</th></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>ONC-RPC Program Statistics
ONC-RPC Program Statistics
<table>
<tr><th>Program</th><th>Version</th><th>Calls</th><th>Min RTT</th><th>Max RTT</th><th>Avg RTT</th></tr>
<tr><td>CLEARCASE</td><td>3</td><td>14</td><td>0.00252</td><td>0.00435</td><td>0.00292</td></tr>
<tr><td>NFS</td><td>3</td><td>15709</td><td>0.00222</td><td>0.13313</td><td>0.00262</td></tr>
<tr><td>NLM</td><td>4</td><td>10094</td><td>0.00225</td><td>0.03565</td><td>0.00268</td></tr>
</table></td><td>Ethereal does not present any useful data in the RPC RTT window for the TAS capture. The snapshot in the table cell on the left of this cell summarizes the RPC NFS RTT stats and includes the information about NLM, which the NFS RTT stats window (above) doesn't seem to provide.</td></tr>
<tr><td> 
</td><td>Protocol hierarchy window from TAS environment packet capture was too large to paste here. It is probably not relevant anyhow because we have all of the information about TCP packets in the SMB RTT statistics window (above).</td></tr>
</table>

Table 3-1: Screenshots from Ethereal

4. Observations from Filemon

Filemon does not seem to capture any useful statistics during an NFS Maestro environment, presumably because the kernel intercepts the calls at a layer that Filemon does not have access to by default in version 5.01. So any statistics from the NFS Maestro environment have not been included.

However, the tool was able to help us capture some potentially relevant data during a build in the TAS environment. For the local-only build outside ClearCase the entire contents of the VOB involved in the build were copied into a local directory on the C: drive. For all intents and purposes, ClearCase was removed from the picture. For the build within ClearCase, a locally stored dynamic view which accesses the VOB in the TAS environment was used. First data captured during a local build outside of ClearCase is presented. Data for the same build from within the ClearCase view is presented in the following subsection.

4.1. Local-only build outside ClearCase

a) **Operations – Breakdown by process**

Command: cat static.LOG | awk '{print $4}' | sort | uniq -c | sort -brn

92769	java.exe:1040
6252	java.exe:836
5075	java.exe:392
711	cmd.exe:1196
570	svchost.exe:956
61	explorer.exe:1128
44	csrss.exe:600
21	Filemon.exe:1772
17	cmd.exe:672
10	mdm.exe:1600
6	csrss.exe:1772
4	svchost.exe:856
3	winlogon.exe:624

b) Breakdown by operation

Command: cat static.LOG | awk '{print $5}' | sort | uniq -c | sort -brn

34363	READ
18701	OPEN
17808	QUERY INFORMATION
17046	CLOSE
15113	WRITE
1519	DIRECTORY
359	CREATE
319	SET INFORMATION
315	DELETE

c) Accesses through MVFS (i.e. \\view\view-tag)

None

d) Accesses through SMB (i.e. \\vobserver\vobstore)

None

4.2. Build in a Local Dynamic View[1] in SMB environment

a) Operations – Breakdown by process

Command: cat local-dynamic.LOG | awk '{print $4}' | sort | uniq -c | sort -brn

183855	java.exe:368
48202	view_server.exe:912
4080	java.exe:408
2590	java.exe:1716
2008	svchost.exe:956
703	vpc32.exe:372
698	cmd.exe:1196
655	taskmgr.exe:1968
476	csrss.exe:600
352	explorer.exe:1128
337	spoolsv.exe:1232
263	vptray.exe:1488
156	winlogon.exe:624
148	services.exe:668
69	cmd.exe:1168
53	Filemon.exe:452
37	lsass.exe:680
20	cccredmgr.exe:1524
18	mdm.exe:1600
12	vpc32.exe:1488
5	svchost.exe:1152
3	csrss.exe:624

c) Accesses through SMB (i.e. \\vobserver\vobstore\...)

Command: cat Filemon-dynamic-view.LOG | grep '\\\\view-name' | awk '{print $5}' | sort | uniq -c | sort -bnr

5057	OPEN
4321	READ

b) Breakdown by operation

Command: cat local-dynamic.LOG | awk '{print $5}' | sort | uniq -c | sort -brn

75760	READ
53179	WRITE
42184	OPEN
34761	CLOSE
28313	QUERY INFORMATION
6964	DIRECTORY
1632	CREATE
969	SET INFORMATION
925	DELETE
43	FLUSH
9	UNLOCK
9	LOCK

d) Accesses through MVFS (i.e. \\view-tag\...)

Command: cat Filemon-dynamic-view.LOG | grep '\\\\view-name' | awk '{print $5}' | sort | uniq -c | sort -bnr

7596	CLOSE
5514	DIRECTORY
1366	WRITE
459	READ
143	DELETE
11	OPEN

[1] Locally stored view and local view_server.exe process running on Windows client in TAS environment

5. Related Observations

The following table presents extra information which was obtained from using the standard tools that can be used to analyze performance.

Item	Observation
Local-only build outside ClearCase	120 seconds
Avg. CPU Utilization – Outside ClearCase	~99%*
Avg. CPU Utilization – Dynamic view	~15% *
Size of most buffered reads by java processes	4096 bytes**
Variance (*v*) in round-trip-times	Factor of 1 < *v* < Factor of 200***
One Open() on a versioned element inside a view generates 3 different types of QUERY_FILE_INFORMATION calls. There are often 10 back to back calls altogether so there are probably more precise correlations, but it has been left up to Rational to determine the relationship of the system calls to network calls.	basic, standard, and extended attributes

Table 5-1: Additional Data to Support Analysis

*This observation was made visually with Windows Task Manager

**This observation was made with Filemon (from www.sysinternals.com)

***Lower and upper bound RTT min/max factors for most significant traffic may account for minor calculation discrepancies in the Analysis section of this document.

6. Analysis

This section provides an overview of the key findings from the previous sections. First, an example of how to calculate where time is being spent is given. Then the bottleneck is focused on. Finally, exploration on whether enabling oplocks[2] or Samba would be beneficial in terms of build performance from the client perspective is covered.

6.1. Where Time is Being Spent

Given the average round-trip-time (RTT), one can calculate roughly how much time is consumed in each type of call using the product:

Average RTT * # of calls = Time spent in seconds

For example,

0.00253	seconds
x 220,433	QUERY_FILE_INFORMATION calls
558	total seconds being spent in calls to QUERY_FILE_INFORMATION

The sum of all products for each type of call reported by Ethereal in addition to the time spent by the local CPU and any other requests, should give a value that approaches the duration of the build in seconds. The values highlighted in green are discussed in the following section.

NFS Maestro Environment		TAS Environment	
Type of Call	**Duration (seconds)**	**Type of call**	**Duration (seconds)**
GETATTR	**1**	**QUERY_FILE_INFORMATION**	**557**
LOOKUP	26	Close	16
ACCESS	13	Read AndX	6
READ	1	NT Create AndX	24
NLM (all)	27	All other SMB calls	7
Client CPU time	120*	Client CPU time	120*
Transparent ClearCase access	172**	Transparent ClearCase access	172**
Total elapsed time of build	**360*** **	**Total elapsed time of build**	**902*** **

Table 6.1-1: Network Calls and Duration

* This value was obtained from the local-only build times from the section Related Observations

**The product of the number of TCP packets (to vobrpc_server) and the average RTT which was obtained from the protocol hierarchy statistics window in Ethereal. See section Observations from Ethereal Captures.

*** The duration of the build is equivalent to the elapsed time of the network capture

[2] See Definitions, Abbreviations and Acronyms for an accurate definition or for more information.

6.2. What is the Bottleneck?

First the possibility that TAS was not the bottleneck needed to be ruled out, so a case was opened with the vendor LSI Logic before opening a case with Rational. After examining the TAS server configuration and running various comparative benchmarks outside ClearCase in their TAS and Windows 2000 Server environments, TAS support stated that the client site was already getting the best performance out of TAS (See Appendix A – E-mail from TAS Support). They were able to tell us that because the benchmark used ran faster with TAS than with a comparable Windows 2000 Server. The validity of their benchmarks and latest statement is likely valid since it is consistent with the rest of the findings herein. Therefore, the TAS server software itself is not the bottleneck in this case.

To be confident that the problem wasn't with the TAS server, the build tools, build process, or networking environment – before presenting the problem to Rational –the number of calls and the round-trip-times needed to be determined to see if there are any numbers that stand out, and then the time spent by each type of call needed to be calculated (as per method described in the section Where Time is Being Spent). Next the network calls and duration data was tabulated, based on the network capture information from Ethereal in section 6.1 – Where Time is Being Spent.

The one key figure that immediately stands out from the SMB RTT statistics in the TAS environment is the sheer number of SMB subcommand calls of the following type:

QUERY_FILE_INFORMATION (Basic, Standard, or Extended)

There are approximately 220,000 calls – 3 orders of magnitude more than the number of equivalent NFS GETATTR calls!

The problem is that each SMB call issued adds a delay of ~2.5 ms to the build while it is waiting for a reply from the server, which adds up to a lot of time spent waiting (557 seconds, or ~9 minutes), especially when it is done that many times for apparently no good reason at all. The statistics from the section Observations from Ethereal Captures show that ClearCase local dynamic view builds in the CIFS/SMB can be approximately 2.5x slower than in the NFS Maestro environment. However, the statistics from Appendix A further demonstrate that they can be more than *3.5x slower*. In both cases, the extra time spent on the calls roughly matches with how much extra time builds take in the CIFS/SMB environment versus the NFS Maestro environment.

Consequently, these calls result in over 440,000 packets being transferred over the network out of a total of ~600,000 packets for the whole build (one for the request, one for the response). As you can see, more than 70% of the total number packets, sent and received during a build, result from these calls. Although the purpose of the extra calls is not clear, ClearCase is querying the remote VOB server (which also runs the SMB service) many times for information on files stored in various VOB filesystem pools. AspectJ seems to be partly to blame for most, if not all of the extra network SMB calls because of the number of stat-like queries on a handful of jar files. However, if the calls were completely eliminated at the system level from the get-go (before they get translated by the MVFS into SMB calls) instead of at the tool/application level, the build time in the CIFS/SMB environment would

practically match – give or take a few inconsequential seconds – the build time that were observed in the NFS Maestro environment currently.

Now that the bottleneck has been determined, what can be done about it? Well, the approach that makes most sense from the technical standpoints is to figure out why the calls need to be done in the first place, what MVFS is doing with the data once it has obtained it, and then find a way to keep network traffic to a minimum by removing the excessive over-the-wire network calls. In Section 8 — Solutions and Recommendations, more detailed work-around solutions are provided while a more sensible solution is investigated by Rational.

6.3. Would Enabling Opportunistic Locks Help?

Oplocks in CIFS/SMB were designed to greatly improve performance from the Windows client perspective. Windows NT/2K/XP, TAS, and Samba servers and clients support this feature. However, Rational recommends disabling oplocks on the VOB server, especially in an interop environment. This recommendation is for all ClearCase CIFS/SMB-based solutions, although they did unofficially state that they are more lenient to allow it to be enabled in the homogeneous Windows environments. In addition, they have recently provided an untested procedure which may help slightly improve performance. The procedure involves some type of splitting of the source, db, and cleartext pools and enabling oplocks on certain pool shares. See section 8.1 – Reduce the # of Application Turns.

The reasoning for being more lenient to allow oplocks to be enabled in a homogeneous environment may be attributed to one Rational customer improving build performance this way (no figure was divulged). This customer has had no issues arising from enabling oplocks in their pure Windows ClearCase environment to date. However, enabling oplocks, even in a pure Windows environment, but especially in an interop environment, remains widely untested.

Some CCIUG members have reportedly found that inadvertently enabling oplocks causes sporadic freezes on user machines, which may lead to data corruption. Rational does not provide any complete documentation on the exact explanation for the contention or incompatibility and data corruption. To further illustrate the point, Doug Graham writes:

> As for oplocks, there are still a lot of things that don't add up, and a lot of details that need to be explained properly. Like which files are in danger of being corrupted if oplocks are turned on (cleartext pool? source pool? database?), and who writes to these files, etc. A client machine using NFS is going to behave very much like a machine using SMB with oplocks enabled, so if there's a problem with turning oplocks on when using SMB, then I'm pretty sure that the same problem is going to exist when using NFS on the clients.
>
> It sorta makes sense in a general and hand-wavy type way that turning on oplocks will have undesirable effects in an interop environment because the Unix clients will not participate in the oplock protocol. An SMB client may think that it has exclusive access to a file, when in reality Unix clients are banging away at it. But even forgetting about interop for a second, the exact same thing can happen in a pure NFS environment, with two or more NFS clients aggressively caching (doing read-ahead and write-behind) data from the same file.

> Oplocks are not a substitute for proper file locking, and they were not designed to be so.
>
> But on second thought (and after a little googling), I may have to amend that somewhat, because I wasn't taking into account actual file locking, and how that interacts with oplocks. If an SMB client that is about to write to a file on the VOB server takes out a lock on the file, and if it holds an exclusive oplock on the file at that point, then it seems that the client does not notify the server of the file lock, because it thinks it's already been granted exclusive access to the file by virtue of the fact that it holds an exclusive oplock. If oplocks are disabled, then the file lock request must be passed to the server so that the server can mediate such lock requests.
>
> The bottom line is that if ClearCase clients actually do issue file locking requests when they are about to modify a file on the server (and I don't know whether they do or not), then it may be important that oplocks be disabled, so that lock requests are sent to the server, where clients that aren't oplock-aware (eg: NFS clients) can find them.
>
> So I think I begin to see how enabling oplocks could cause file corruption, but this is all still just wild guesswork. In the case of ClearCase, as far as I know, the only files on the server that are written by a client are the cleartext files. If that's true, then they are the only files that run the risk of being corrupted if oplocks are enabled. Yet during our conference call with Rational, they claimed that we could move the cleartext files to a share that has oplocks enabled (to improve performance), but that we'd have to leave the database and source pool on a share with oplocks disabled. This is the opposite of what I'd have guessed. If the database and source pool are only ever written by processes on the VOB server, then I can't see how they could be in any danger of corruption regardless of whether or not oplocks are enabled. So if it's ok to enable oplocks on the cleartext pool, it should be ok to enable oplocks everywhere.
>
> This is all very tricky stuff, and my brain hurts from thinking about it. What will and won't cause problems depends on a number of aspects of the ClearCase architecture that I'm not sure I fully understand yet. Like what sort of locking are they doing, if any, and on what files?
>
> I think we've probably gone about as far as we can go with hand-wavy type information, which is all we've ever gotten from Rational. This QUERY_FILE_INFORMATION problem is all about subtleties, and to resolve those subtleties is going to require an understanding of the actual details of what's going on under the hood. Like, perhaps, an exact scenario demonstrating clearly how enabling oplocks in an interop environment could cause file corruption.

Furthermore, it is common knowledge that Rational to make broad conservative statements by erring on the side of caution, and cannot release any written information on relevant performance of NFS and CIFS/SMB due to apparent legal reasons, and lack of rigorous testing, among others. It is also common knowledge that SMB is a “bloated” or “chatty” protocol among engineers who have analyzed network traces. Particularly because oplocks are strongly discouraged by Rational, it is difficult to ascertain if ClearCase itself or the components of the Windows kernel that it uses (i.e. the SMB redirector), are truly and

wholly accountable for the slowdown. Thus, the conjecture that SMB is to blame is likely not very useful or accurate.

Even if oplocks were enabled, given that one customer has tried without much success, the extra calls would still be observed because they are coming directly from the MVFS itself, and not the SMB redirector (SMB client). The difficulty is that there is no trivial way for customers to verify the null-hypothesis of this theory for a fact without an identical test with all variables controlled (i.e. network delay, server performance, disk speed, etc.), and without having a clear picture of how ClearCase works internally.

According to one CCIUG member, David Boyce, part of the predicament with oplocks and ClearCase can be best characterized as follows:

> UNIX doesn't support oplocks and thus interop uses of ClearCase cannot use them, because if there's one thing worse than slow performance it's a network in which some hosts set and trust in locks of which other hosts are blissfully ignorant. There are a couple of solutions if you think oplocks would help:
>
> (a) Move your vobs to a Windows platform (of course this is only an option if you have no Unix clients, or at least no Unix clients that need MVFS).
>
> (b) Use Multisite to keep a copy of the vob on a Windows platofrom, such that Unix clients are server by Unix servers and Windows by Windows. This adds license cost and some admin cost.
>
> (c) I gather that both Linux and SGI have kernel hooks to support oplocks. If you're brave and use only those Unices you could go that way, but you'd be a pioneer.
>
> (d) If you don't have Unix clients at all but just a Unix vob server, you could live dangerously and turn on oplocks. It's still possible to get into trouble as there are some processes on the vob server (e.g. nightly scrubber jobs) which might run afoul of oplocks but you might get lucky. And again you'd be a pioneer. You're probably better off with option (a) in this case, even though Unix vob servers are more robust in general.

To summarize, it is doubtful that enabling oplocks would be a plausible solution: more research is required to make an accurate determination. In short, enabling oplocks may be able to boost performance in some environments, but oplocks are not a viable option today for most modern enterprise-scale heterogeneous ClearCase environments.

6.4. Will Samba Perform Better?

One option to consider is to replace TAS with Samba. However, because the excessive delays are coming from the client side – and not the TAS or ClearCase server software itself – compounded by the fact that enabling oplocks is prohibited, it is doubtful that Samba would ever perform much better than TAS from a designer test build standpoint. There may be other scalability and robustness limitations in the TAS security model which Samba could address. However, the security model issue is likely more of a capacity concern than a build

performance issue, so it is outside the scope of this document. See the section Solutions and Recommendations for more information.

7. Impact & Who Should Solve This Issue

All ClearCase users (not only those in the client's organization) who do builds frequently in ClearCase dynamic views in an SMB-based environment (homogeneous and UNIX/NT Interop) may unknowingly be experiencing the exact same bottleneck in terms of the excessive number of calls to QUERY_FILE_INFORMATION. One experienced CCIUG member (Mark Keil) responded with "the slowdown cost is probably in the order of 10s of 1000s of dollars in extra equipment costs to try to get around this problem, and probably 10s of 1000s of dollars of lost programmer productivity as users sit around waiting for builds." Costs could be as high as in the millions per year for large-scale companies.

The onus should not be on the customer to find a technical solution, but the burden of responsibility should be on the vendor to provide a solution based on the collective needs of the consumers of the product. As previously mentioned, the vendor of TAS has been contacted to determine if the configuration was the problem. However, they have looked into the client's server settings and have made a determination that the client site was already getting the best performance out of TAS. Therefore, the only remaining vendors who may be able to help are Rational, Microsoft, and maybe Sun.

Some may suggest that Microsoft be engaged to resolve this issue. However, to suggest this may not be effective. According to Rational, the extra file attribute queries over SMB that were observed in network traces are being made by code in the MVFS itself and not necessarily by any Microsoft code. The following is an article was recently posted by Doug Graham on the CCIUG mailing list which illustrates this point:

> I don't think that there's anything inherently chatty about the SMB protocol vs. the NFS protocol. In fact, SMB sports a feature (oplocks) that should in theory make it less network intensive than NFS (V3 and less; V4 has something close to oplocks) under most scenarios. Microsoft's implementation of the SMB redirector has its bugs though, and these can defeat any useful caching on the part of an SMB client at times, making it good deal more network intensive that it otherwise would be. For example, last I checked, readonly files were not cached properly; they were reread across the network each time they were opened and read. Similarly for executables and DLLs.
>
> When ClearCase MVFS is thrown into the mix though, it becomes very difficult to tell which piece of software is misbehaving. That's because there is very little technical documentation about MVFS. We don't know what cache consistency model it follows, or even whether or not it needs to make any sort of cache consistency guarantees. For that reason, we don't know what the effects of enabling oplocks would be, nor do we know what sort of network transactions it is required to make to the VOB server to enforce its cache consistency model.
>
> What we see when opening a versioned element inside a view is three different time-consuming QUERY_FILE_INFORMATION SMBs (one for each of basic, standard, and extended attributes). Those look very much like

```
they're supposed to serve the same purpose as the NFS GETATTR
transactions that are required in the implementation of close-to-open
cache consistency.  Whether they really are required for this purpose
or not is something only Rational might know, and even they don't
appear to.

Rational recommend turning off the main feature provided by Microsoft
to improve performance (oplocks), after which MVFS seems to compensate
to some degree for the resulting absence of caching in the redirector
by doing its own caching.  The upshot is that network behaviour under
MVFS is radically different than under straight SMB, which means that
it's not Microsoft to whom we should be looking for answers, but
Rational. According to Rational, the QUERY_FILE_INFORMATION SMBs are a
result of explicit calls in their MVFS code, calls that must have been
put there for a reason.  We know that MVFS does cache file attributes
(a stat() doesn't result in network activity), but we don't know
whether or not MVFS can use that cached information instead of fetching
it a new with another series of QUERY_FILE_INFORMATION SMBs.

It may turn out that the root cause of these extra transactions has
something to do with Microsoft's SMB implementation, but it'll have to
be up to Rational to determine that, because nobody else really knows
what the design philosophy behind MVFS is.

--Doug.
```

How about the vendors of user processes of the MVFS, are they responsible? Since the Sun JDK compiler and AspectJ seem to be triggering many over-the-wire attribute queries on jar library files, it may be wise to verify that senior Sun JDK engineers are also aware of the issue to which they may also be answerable to. The AspectJ development team may also be helpful. Ultimately though, most design tools are not meant to be efficient when run inside a dynamic view (presumably because they are designed to run on locally stored source files). They would be much more efficient if the MVFS was made a little more intelligent about the fact that it relies on the underlying CIFS/SMB redirector to be smart about coherent caching of metadata being requested over the network; but when oplocks are disabled, how can it?. Because of UNIX's apparent incompatibility with oplocks, it may be missing the opportunity that other pure Windows applications have to boost the CIFS/SMB redirector performance when used in a networked environment. In summary, the reasoning for why Rational – and no one else – are answerable to the issue may be characterized as follows:

- UNIX does not support oplocks, which then puts the responsibility on the user of the CIFS/SMB redirector (i.e. the MVFS) to minimize the calls over the network since it cannot be expected to do that work when oplocks are disabled
- For a many obvious reasons, users of the MVFS (i.e. design and build tools like javac, gcc, ant, AspectJ) should not be expected to implement the level of caching required to maximize performance when they are executed inside a heavily networked application such as ClearCase
- It seems plausible that the expensive over-the-wire calls QUERY_FILE_INFORMATION SMB calls can be avoided.

In addition, here is some more information on the questions surrounding whether the calls can in fact be removed or not. Doug Graham wrote:

I think that it's fair to say that we don't know for sure that the calls can be removed, but the evidence seems to point in that direction, provided that the NFS clients are doing everything they're supposed to do. We're told that MVFS makes the same calls to the underlying filesystem regardless of whether that filesystem is SMB or NFS. When that filesystem is NFS, the NFS client does not make any calls to the server, it returns information from its cache instead. If it's ok for NFS to return cached information, then it's hard to imagine why it wouldn't be ok for SMB to return cached information as well. If the SMB redirector doesn't contain its own cache, then it should be possible to add a cache on top of it.

All of that assumes that MVFS actually needs to fetch the attributes of cleartext files this frequently. What we've been begging Rational to tell us is why MVFS cares about the attributes of the cleartext files at all, and what it does with those attributes once it has them. We don't know, and neither, apparently, do they. Without knowing what those attributes are used for, it's impossible to know whether the solution is to just remove the calls altogether, or to add a caching layer, but all indications at this point are that at least one of those solutions should do the job.

All Rational were able to tell us during our last conference call is that MVFS needs these attributes so that it can present accurate information to the application process. But an application process that asks for the relevant attributes explicitly (eg: ls or dir) does *not* cause MVFS to query SMB for these attributes. Or at least SMB does not ask the server for the attributes. We only see the QUERY_FILE_INFORMATION SMBs when the application makes a call such as open()/CreateFile() which does not result in the attributes being passed to the application. That implies that the attributes are being fetched not for the direct benefit of the application, but rather for the internal use of MVFS. Either that, or CreateFile() is grabbing them and caching them at the time a file is opened so that the attributes are available later if the application asks for them (via fstat()/GetFileInformationByHandle()). If it's the latter then they could just remove the call from open() and then only fetch the attributes if and when the application does an fstat() (which is rare for most applications).

MVFS keeps a cache of open file handles to files in the cleartext pool. It's against those files, using one of these open handles, that MVFS issues the QUERY_FILE_INFORMATION calls. I don't know for sure, but I suspect that it's not possible to modify those cleartext files while MVFS has them open. If that's true, then the cleartext file's attributes can't be changing anyway, with the possible exception of the last access timestamp. In that case there's no point in repeatedly fetching them.

--Doug.

8. Solutions and Recommendations

There are three obvious approaches to the problem at hand. Either one of the following solutions or a combination of them, are required in order to maximize performance, from the Windows client computer perspective:

a) Reduce number of application turns (i.e. SMB QUERY_FILE_INFORMATION calls)
b) Reduce round-trip-times between the client and server
c) Increase number of concurrent application turns through parallelism or multi-threading. NOTE: This solution could do a lot of harm to the network infrastructure. Although it would solve the issue from the client perspective, it is probably not the right approach.

Unfortunately, as indicated in the minutes of the teleconference with Rational, it may or may not ever be possible to implement a fix to dramatically reduce the number of application turns. In fact, Rational has no plans to fix the issue at the time of this writing (See Appendix D – Second Teleconference with Rational. In short, if more organizations append to the RFE, it may present Rational with a stronger incentive to increase the priority of the issue accordingly. Users may need to be prepared to live with the inferior work-around solutions presented below in the meantime.

The following subsections contain several ideas for workaround solutions, segregated into three broad approaches, the constraints surrounding each one, and related research ideas for future consideration. Some ideas presented may not be plausible because they may actually hurt performance or bring up other concerns which may not effectively address the issue with the number of over-the-wire SMB calls. They have been included for the sake of completeness despite the fact that they may not be suitable for all teams or be effective solutions. See Section **Error! Reference source not found.** – **Error! Reference source not found.** for the most plausible recommendations. Finally, general recommendations are made to development teams about the ideas presented.

8.1. Reduce the # of Application Turns

1. **Use snapshot views** instead of dynamic views (particularly on a Windows ClearCase client) to perform builds in a lot less time. All network activity is eliminated during a build in a locally stored snapshot view. User documentation regarding caveats of snapshot views may need to be reviewed (i.e. lack of support for VOB symbolic links). New versions of SCMPs which cover snapshot views may need to be created.

2. **Optimize the build engine software or build environment** – reduce time spent on redundant accesses through SMB QUERY_FILE_INFORMATION calls by:

   a) Using build tools which are known to perform better, such a different Java compiler (Jikes). When dynamic views are needed, consider deploying the **Jikes** Java compiler as a replacement for Sun's Javac on Windows PCs. On Windows, Jikes's javac seems to

leak a much lower number of calls over the network than the Sun JDK, presumably because of more intelligent application level caching (See Appendix E – Build Engine Optimization). Have a look the Jikes homepage: http://oss.software.ibm.com/developerworks/opensource/jikes/ for more information. However, Jikes does not seem to solve the problem completely in all cases. When it was used on the build in this case study, it managed to reduce the build times only by 13%, instead of a factor of 3 (which is what is needed). AspectJ seems to be responsible for most, if not all of the network SMB calls. Perhaps it needs to be redesigned for improved performance.

b) Restricting the number of directories in the Java class path or "includes" directories

c) Modifying build engine so that frequently accessed data such as libraries or TMDIR data is accessed by build tools from local disks (i.e. C:\ Drive in Windows) and on different spindles (which presumably maximizes data mover performance) instead of the network. Even better – ensure access from memory instead of local disk (e.g., /tmp on UNIX, or RAMDISK on Windows).

d) Taking advantage of **Express Builds** or build avoidance through winkins with omake or clearmake may be an idea. However, the solution is not as easy as typing "clearmake" instead of "make". For winkins to work, they usually need to be implemented from the beginning of the project. The type of analysis that may be required to really benefit from winkins or build avoidance involves a lot of tedious work, but it may be something worthwhile investigating. Here is more information received from David Boyce:

> One argument against investing in winkin analysis is if you don't want to be deeply invested in ClearCase. I.e. if you ever dump CC and go back to CVS then your winkin work may be partly wasted(*). To the degree you use snapshot views a similar argument would apply. But in your case the client is clearly bought into both CC and MVFS so that doesn't matter.
>
> In general you'll have much better luck tuning your environment towards CC as it stands than begging, pleading and threatening Rational into tuning CC for you. And any fix they do make probably won't be running on your network for a year or more. So do keep the pressure on them but remember there's nothing hack-y about winkin; it's an integral part of MVFS.
>
> (*) Actually most winkin work is fixing hidden bugs in Makefiles. It's like porting C code to a new platform; even if you never actually ship on that platform you end up with much cleaner code. If you've ever done porting work you know that most of it is finding assumptions that were true on platform A but are broken in general (e.g. endian-ness, null ptr derefs, sizeof(int) == sizeof(ptr), etc), and things like uninitialized variables that may happen to work in one environment. A smart, well written Makefile usually has little trouble supporting winkin; you can think of winkin rates as a measure of how clean your Makefiles are. Of course this is complicated by the fact that (at least) your organization is doing Java/AspectJ work.

e) Adding parallelism to the build engine (make –j or clearmake –j) which could increase the number of concurrent requests over the network (and could become a network or server bottleneck issue – not recommended to solve this issue)

f) Distributing builds across machines or CPUs (which can be done in various ways with different tools and languages). Again, this solution could increase the number of concurrent request over the network (and could become a server bottleneck issue – not recommended to solve this issue)

NOTE: See Appendix E – Build Engine Optimization for more information on class path information and Jikes versus Sun's JDK. However, optimization may require significant or undesired changes to the build engine and may not bring the full benefit expected. Some of those changes may or may not be recommended, but have been included for completeness.

3. Provide **NFS Maestro** to heavy build users only. Again, since NFS (especially on a PC client) may not be fully supported by the organization's Risk Management Team (i.e. security group) for security reasons, this may not be the best solution for all teams, and it may cost development teams and support teams more time and $$$ to support.

4. Access ClearCase through **UNIX NFS** instead of Windows SMB (Sun OS particularly for Java development). ClearCase can perform much better with NFS (regardless of platform) than it does in the SMB environment because of various issues discussed in this report. Since NFS may not be fully supported by the organization's Risk Management Team (i.e. security group) for security reasons, and because Java does not perform well on HPs, this may not be the best solution for all users.

5. **Multi-site VOBs to individual desktop** to eliminate traffic over the network. The costs could be extremely high considering licenses, support costs, etc. Certainly not a one size fits all solution, and this is probably not the correct approach.

8.2. Reduce Round-Trip-Times between Client and Server

6. **Create new VOB servers** (UNIX or Windows) within closer proximity to the clients that perform frequent builds and replicate the required VOBs. This could be very costly considering licenses, support, etc.

7. **Move the VOB server** to a site that is in closer proximity (i.e. < 1 mi) to where heavy build users are located. This solution could be very expensive considering moving costs, setup costs, downtime costs, etc. Again, to suggest moving the VOB server would may only help work around the problem and not actually address the core of the issue, so it is not recommended. One exception that can alleviate the cost factor is if the site housing the servers is being shut down and there are already plans in the works to move the ClearCase servers to a different location anyway. Naturally it would be fitting to choose the location wisely since it has considerable implications on build times in this case.

8. **Upgrade network** to improve network latency by reducing number of hops, optimizing network topology and architecture, deploy fiber connection over a long distance – the client's organization has already done that much. Lots of time and $$$ involved here.

8.3. Related Research Ideas

9. **Research server software scalability and robustness of TAS versus Samba** in an enterprise environment (highly recommended as an offshoot of this report!). Understand potential problems regarding server software (i.e. TAS software itself, and interactions with domain controller and WINS server software). Be prepared to deal with any domain controller issues with TAS which may become pressing when more users create a higher load on TAS through normal ClearCase use. The TAS vendor may have difficulty solving this because of any design deficiencies. It may not support the more scalable model and may only support the more inferior model (security=server). Samba may solve the domain controller-related issue because it supports the more scalable model (security=domain). Contact Rational engineers, TAS engineers, the Samba mailing list, and other companies on CCIUG to gather as much information regarding relative merits of TAS and Samba. See The Samba FAQ, Chapters 9 and 10 – "Samba as a ADS Domain Member" and "Samba as an NT4 and W2k Domain Member. See http://us4.samba.org/samba/devel/docs/html/Samba-HOWTO-Collection.html.

10. Developers could benefit from using **Linux NFS** client software on their Intel-based desktop (particularly Java developers) instead of Windows SMB or using a shared UNIX host to harness the power of their own desktop which has cheaper compute power than most average UNIX machines around today. Again, since the Linux OS may not be fully supported internally, this may not be the best solution for all users. Consider whether or not it is necessary to meet Risk Management security requirements by doing a feasibility study of non-NIS *netgroups*, *Kerberos*, or other ways of restricting access.

11. Linux supports **native mounting of CIFS/SMB file systems** which could help enhance security, but other UNIX systems (Sun, HP) do not have that capability without extra software like Sharity, which isn't supported by Rational. Linux is not a prevalent OS at this time for many reasons which could make it difficult to deploy this solution to a large number of users who may depend on software that is only available on the Windows OS (i.e. MsDev Studio). VMware, Open Office, or Crossover Office may solve that problem, but they introduce some complications. Supporting it and purchasing licenses could be more expensive than is necessary to address the security issue.

12. Investigate the relative merits of deploying other existing and upcoming solutions such as **DFS**, **Sharity**, **NAS**, or other data access solutions, or combinations of solutions. However, none of the data access solutions would likely be able to fully address the issue without significant changes to caching behavior with ClearCase MVFS and the CIFS/SMB redirector. See ClearCase 2002.05.00 (5.0) – Supported Network Access and Storage Platforms: http://www.rational.com/support/documentation/release/cc_storage_network.jsp

8.4. Final Recommendations to CM Analysts

Based on the outcome of the evaluation of the above recommendations & solutions, an implementation plan should be developed with minimal impact to the current development process to the extent possible. This plan would include how best to educate users on how to effectively use snapshot views, and how best to internally communicate information about applicable caveats (if any) – be it through learning events, web site, presentations, etc.

Information on when to use dynamic views (i.e. merging, browsing) versus when to snapshot views (i.e. builds), what rules to add to the config spec for a specific build, references to man pages, and other pertinent information should be provided to the user. Information on how snapshot views fit into the overall configuration management strategy should also be documented in the project's SCMP and be communicated to users.

The above work-around solutions may not answer the needs of some specific teams and further analysis into performance may be required. It is recommended that any teams suffering from build performance issues inside ClearCase be made aware that they can contact the enterprise ClearCase support team through the hotline to arrange for further investigation of any outstanding performance issues.

9. How Other Organizations Can Help

The adage "there is power in numbers" or "many hands make light work" can be a fitting way to describe how other organizations that use ClearCase can help. In fact, in order for the RFE to be considered higher priority, more customers should evaluate how the build degradation issue impacts their company and see to it that their needs are expressed by appending to the RFE (i.e. calling in to Rational Technical Support, reference RFE `#RATLC00687665`. It may take a few large ClearCase customers to provide Rational with an incentive to fix their beloved product, or a number of smaller customers expressing their vested interest in this issue. Although only three organizations have appended the RFE to date, there are probably many more small, medium, and large scale organizations that could benefit from getting this RFE addressed sooner rather than later.

One relatively quick deterministic way to make a contribution is to fire up a sniffer which decodes SMB packets well, such as Ethereal (with permission from your network management or risk management group of course), capture network data during a build on a Windows PC when performed inside a locally stored dynamic view and see how many SMB calls are being made. One tell-tale sign of slow performance is that the CPU usage (see Task Manager) is below 90% for most of the build, but that is not always the best indicator of how much time is being spent on SMB calls. Another piece that is important is to calculate the time being spent for each type of SMB call. It is suggested that the procedure as outlined in section 2.4 – How Data was Captured and the formula from section 6 — Analysis, be used by others organizations evaluating the impact of the issue. After doing the analysis in your own environment, you should have a more solid understanding of the number of SMB calls, and what the calls means in terms of the overall duration of the build, and how your business is impacted by the issue.

As for the impact to your organization, another saying is "Money talks!" One heuristic used to assert the impact to development and the business as a whole or for your business unit might be to use the loaded labor rate to the tune of $120 USD per hour per known designer affected in terms of lost productivity per day. Another way to state the impact is that your customers or sanity testing can receive a product build in less time, although it may be more difficult to quantify the impact in this way. For example, it could cost your organization a contract if the build takes you 3x longer than it should every time a build is requested. If the number of time-consuming SMB calls is significant enough (i.e. more than a 10,000 calls – depending on the round-trip-time) to represent significantly high designer productivity loss, there is probably good merit in appending to the RFE that is currently open. ClearCase Administrators or senior development managers may want to contact Rational Technical Support or their Rational representatives (i.e. technical or sales) to provide any SMB RTT statistics on a typical build and the best estimate possible in terms of an impact statement on their organization.

If you think you may be impacted, please contact CMI at info@cmi.com to inform us of the impact on your business, and the information you plan to append to the RFE. This issue may represent a great opportunity to learn more about how ClearCase works under the hood and ultimately, improving productivity inside your organization – and globally!

10. Conclusion

It was observed that the ClearCase client behaves differently during a build based on which redirector (i.e SMB or NFS) is being used on the client, and not based on the OS of the client or VOB server. It has been observed that there are clearly a significantly higher number of application turns (i.e. SMB QUERY_FILE_INFORMATION calls) generated during a Java build inside a locally stored ClearCase dynamic view in a CIFS/SMB environment (i.e. TAS, Samba, or NAS appliances) than does in the a NFS environment – whether the ClearCase client is running on UNIX/Linux or Windows. The client waits for a response from the server for each call, which can add up to a significant time additional delay on top of ClearCase RPC calls (i.e. 2 minutes) to the entire build process (i.e. an additional 10 to 15 minutes for a build that normally takes roughly 3 minutes to complete on local disk and roughly 5 minutes when NFS is used).

Rational engineers have acknowledged the performance issue discovered as valid (see Appendix B – Minutes from Teleconference with Rational, 4/16/03). They are aware that other customers have reported NFS Maestro can be roughly 2x faster or more than CIFS/SMB-based environments with ClearCase -- homogeneous Windows and NT/UNIX interop. Moreover, the problem appears to be part of an architectural deficiency in the ClearCase MVFS, but this problem can undoubtedly be fixed if it receives enough attention and priority.

In summary, the goal of the analysis was to determine more specifically where the slowdown is occurring (i.e. ClearCase, TAS, or some portion of the build). Through analysis it was determined exactly which unnecessary SMB call accounts for virtually all of the extra time (namely, QUERY_FILE_INFORMATION). The bottleneck has been found, it has been suggested who is most answerable to a solution, information on how other organizations can help has been provided, and work-around recommendations have been suggested in order for users to maximize build performance in the ClearCase environment while a more effective solution to the problem is pursued.

Appendix A – Data from Another Client PC

The following statistics were obtained from a Windows 2000 PC with a 10 Mbps connection to the same VOB server, performing the same Java build, as the client used for the statistics from Section 3 – Observations from Ethereal Captures. The environment used was the same CIFS/SMB-based solution as well (i.e. TAS). The build times using NFS Maestro on this PC (not shown in this report) were approximately 360 seconds (6 minutes), which is approximately the same time it takes on the Windows XP PC with a 100 Mbps connection.

The statistics from Section 3 show that ClearCase local dynamic view builds in the CIFS/SMB can be approximately 2.5x slower than in the NFS Maestro environment. The following statistics show that they can be *more than 3.5x slower*. Note that the number of QUERY_FILE_INFORMATION calls has gone up by ~130,000 but the Avg RTT has remained the same. The increase remains unexplained. It may the PC OS or software setup or it may be the network setup.

```
IO-USERS Statistics:

            client <-> server

   |       <-      | |       ->      | |     Total     |
   | Frames  Bytes | | Frames  Bytes | | Frames  Bytes |
   387672  49747664  399162  89489167  786834 139236831

Total elapsed time:   *1340.307382*
================================================================
SMB RTT Statistics:
Filter:
Commands                    Calls   Min RTT   Max RTT   Avg RTT
Close                        4588   0.00027   0.38447   0.00318
Flush                          75   0.00231   0.07230   0.00670
Write                          73   0.00225   0.00322   0.00245
Transaction                    32   0.00065   0.01279   0.00142
Read AndX                    2226   0.00225   0.03255   0.00342
Write AndX                    255   0.00242   0.03671   0.00400
Tree Disconnect                 3   0.00023   0.00909   0.00319
Negotiate Protocol              1   0.00027   0.00027   0.00027
Session Setup AndX              4   0.00119   0.30296   0.08593
Logoff AndX                     3   0.00021   0.03596   0.01236
NT Create AndX               4810   0.00047   0.33061   0.00518

Transaction2 Commands       Calls   Min RTT   Max RTT   Avg RTT
QUERY_FILE_INFORMATION     352129   0.00222   0.28853   0.00258
SET_FILE_INFORMATION           75   0.00234   0.00806   0.00273
```

Appendix B – Email from TAS Support

-----Original Message-----
From: Storage Applications Group
Sent: Wednesday, April 09, 2003 7:43 PM
To: Dugal, Pat
Subject: RE: 2-34823124- Follow-up

Hi Pat,

I still have not heard back from the engineering. Here is the result of the tests I've run.

Case 1.

Server: sparcv9 processor at 400 MHz with 128 MB of ram
Client : windows XP 800 MHz processor with 256 MB od ram

Running a string search took 55 seconds to finish using TAS.

Case 2.

Server : Windows 2000 700 MHz processor with 128 MB of memory
Client : windows XP 800 MHz processor with 256 MB od ram

Running a string search took 1m:55s to finish using microsoft smb client.

I have not been able to run test using NFS. NFS and SMB are different protocols and are not really comparable.
I don't think we are going to be able to match the numbers you see in NFS.
I'd say that you are already getting the best performance out of TAS.

Let me know if you have any questions.

Best Regards,

Storage Applications Group
LSI Logic Storage Systems, Inc
http://www.lsilogicstorage.com

Appendix C – Minutes of Teleconference with Rational, 4/16/03

From: Dugal, Pat
Sent: Friday, April 18, 2003 5:23 PM
To: Undisclosed Recipients
Subject: FW: [SR#173382749] -- Performance issues using TAS that did not exist with NFS Maestro

Hello all,

Please forward minutes (below) from the teleconference of 4/16/03 with Rational on to anyone I missed who may be interested.

Please note that the following important points were made but were left out of the minutes.

1. Rational acknowledged it is common knowledge among engineers that the NFS Maestro environment is known to be 2x faster than in the CIFS/SMB environment.

2. The NT MVFS engineering team is very small. Contention for resources may limit the effort that goes into certain enhancements.

Best Regards,

- Pat

-----Original Message-----
From: IBM Rational Software Customer Services (NA)
Sent: Friday, April 18, 2003 4:27 PM
To: Dugal, Pat
Subject: [SR#173382749] -- Performance issues using TAS that did not exist with NFS Maestro

Dear Patrick,

Here are the key points from the conference call the other day.

1. IBM/Rational requested but did not receive a written confidentiality agreement from your organization indicating it was ok to discuss your internal environment with other organization present. As a result of this, your Senior Manager agreed to have the conference call continue as long as the conversation remained in generalities - no specific information regarding the customers respective environments were discussed.

2. Attendees from IBM Rational software division included:
   - Senior Technical Support Engineer

- Escalation Engineer
- ClearCase Support Manager
- Customer Advocate Engineer
- ClearCase Engineer

3. Organization1 and Organization2 have a vested interest in having a specific RFE fixed. Representatives from both companies were on the conference call at your request.

4. Your organization has logged an SR173382749, April 10th. Upon further discussion, your organization has appended an RFE which was already opened, RFE #RATLC00687665.

5. The decision to implement a RFE (Request for enhancement is based upon a number of factors. Those factors include but are not limited to:
a. Scope of work involved
b. Impact to customer
c. Depending upon scope, impact to the customer and other factors will determine when an RFE is released into the product.

6. The problem is a performance issue when using SMB compared to NFS clients. The customer reports a difference in performance in SMB.

7. Rational acknowledges the issue reported from our customers. Rational has spent a significant amount of time researching how to improve performance in SMB. To date, Rational has not implemented a solution in this regard.

8. Discussion on whether oplocks might be a possible solution was discussed. However, Rational has found that oplocks may not be beneficial to customers who share an interop environment. It was agreed upon that this probably would not option because the slight gain in performance could be nullified by the increased risk factor.

9. We understand the challenges our customers have been facing. Due to this challenge and a fix not being implemented on this RFE, Rational has requested a conference call early next week between organization1 and organization2. The goal of this call is to further explain to our customers the research Rational has done on this RFE so they can understand the types of things we have looked at implementing.

10. At this point, Rational is investigating this RFE. It is still under investigation. At no time during this call did Rational imply this RFE would be fixed at this time.

11. Harry mentioned that Engineering had a white paper with details on MVFS caching. At this time the paper is in rough draft -alpha form. Barbara Niles to further work with engineering on when the paper will be completed. I anticipate at a minimum 90 days from now.

Best regards,

Senior Technical Support Engineer
ClearCase Rational Software
IBM Software Group

Appendix D – Second Teleconference with Rational, 4/28/03

1. Invitation

From: IBM Rational Software Customer Services (NA)
Sent: Monday, April 21, 2003 5:24 PM
To: Dugal, Pat
Subject: [SR#173382749] -- Performance issues using TAS that did not exist with NFS Maestro

Patrick,

I left you a voice mail message regarding our scheduled conference call that was to take place tomorrow. In order to get the right people in the room, we are going to have to move this call to Monday, April 28th at 2pm EST. At this time an MVFS engineer will be available to explain things in greater detail to you regarding our research into SMB performance.

It is our hope that as this call concludes you have a better understanding of why we classified the RFE as an RFE and that you have a better understanding of why we have no plans to fix the RFE at this time.

The conference call number is below.

Conference Call Number
1-866-xxx-xxxx
Participant Passcode: xxxxxx

Sincerely,

Technical Support Manager, ClearCase
Rational Software
IBM Software Group
Email: support@rational.com

2. Minutes of Second Teleconference with Rational 4/28/03

From: IBM Rational Software Customer Services (NA)
Sent: Monday, May 05, 2003 4:42 PM
To: Dugal, Pat
Subject: [SR#173382749] -- Performance issues using TAS that did not exist with NFS Maestro

Patrick

Here are the minutes from last week's conference call. I am still waiting some information and will follow-up with you in a couple of days.

From the meeting: April 28th
Attendees from IBM:
NT MVFS Engineer, Customer Advocate Engineer, ClearCase Support Manager, Senior Technical Support Engineer, Customer Advocate Manager, Escalation Engineer, Product Engineering Manager
Attendees from Nortel: Doug Graham, Eric Boehm
Attendees from CMI: Patrick Dugal

1. Split pool configurations with CIFS -- The write up for this is currently being reviewed an update will be forthcoming. - more information will be provided by the end of this week 5/5/03.

2. During the conference call a discussion took place with regards to sending in a network trace. However, the MVFS team discussed this further, after the conference call, and see no need to review a network trace at this time because this operation results in a readdir within the MVFS and as a result should not generate the QueryInfo calls they see with the file open.

3. The Customer Advocate Engineer has tested an unrelated fix to the MVFS, which has shown an improvement in the areas of build performance. This patch improves the speed of compiles but will not reduce the number of queries on the network. If you are interested in trying a TEST version of this MVFS, you may do so with the understanding that this is test patch and the following requirements must be met, a signature of waiver must be provided and the latest patches must be installed

4. RFE must be handled following the business process -- i.e. begins with Product Management evaluation. Information regarding Product Management will be provided in a follow-up email.

5. Discussed if performance numbers are available. We will not be providing these numbers due to a number of factors that were discussed during the call.

6. Discussed if turning on oplocks is acceptable. Action item to send procedure out on how to do this if the split pool configuration is used (see item 1).

7. Our performance team is aware of this RFE.

8. Information regarding a time-frame for the MVFS performance tuning White Paper will be provided at a later time.

Best Regards,

Senior Technical Support Engineer, ClearCase
Rational Software
IBM Software Group
Phone: (800)433-5444
Email: support@rational.com

3. Follow-up Questions from Doug Graham

```
-----Original Message-----
From: Doug Graham
Sent: Monday, May 05, 2003 6:38 PM
To: Dugal, Pat
Cc: Eric Boehm
Subject: re:FW: [SR#173382749] -- Performance issues using TAS that did not e
xist with NFS Maestro


In message "FW: [SR#173382749] -- Performance issues using TAS that did not e
xist with NFS Maestro", "Dugal, Pat" writes:

>-----Original Message-----
>From: IBM Rational Software Customer Services (NA)
>[mailto:support@rational.com]
>Sent: Monday, May 05, 2003 4:42 PM
>To: Dugal, Pat
>Subject: [SR#173382749] -- Performance issues using TAS that did not exist
>with NFS Maestro
>
>
>Patrick
>
>Here are the minutes from last week's conference call. I am still waiting
>some information and will follow-up with you in a couple of days.

Hmmm.  I wonder what information they're waiting for.  We always seem
to be waiting for something, without being sure what we're going to
get when the waiting is over.  Oh well, I'll ask this question anyway,
even though I suppose there's a chance that this is the question that
they now realize themselves is a question that needs answering.

>From the meeting: April 28th
>
>2.  During the conference call a discussion took place with regards to
>sending in a network trace.  However, the MVFS team discussed
>this further, after the conference call, and see no need to review a network
>trace at this time because this operation results in a readdir within the
>MVFS and as a result should not generate the QueryInfo calls they see with
>the file open.

So, the million dollar question, and the one that obviously needs to
be answered before we can any further with this, is still: what are the
results of the QueryInfo call used for, and why is it apparently necessary
to make this call when files are opened, but not when the attributes of
a file are explicitly asked for?  If readdir can get the attributes from
cache, then why can't open?  I think that Harry mentioned during one call
that he's open to ideas on how to fix this problem, but as I mentioned
during the call, there's no way that we can provide any useful suggestions
without knowing why the heck they're making these calls to begin with.

I don't think it's only readdir calls that use cached information.
The stat() routine in the Microsoft C library calls the Win32 routine
```

FindFirstFile() to obtain atrributes, and that probably does get translated into a readdir, but there's also fstat() (win32 equivalent GetFileInformationByHandle()), which obtains the attributes of an open file. With MVFS, calls to fstat() do not result in network queries being made, so the attributes must be being read from some sort of cache, and I don't think this has anything to do with readdir.

One more comment is in order I think. We waited a week for the MVFS engineer to come back from holidays so that he could answer our questions. When he did get back, we had a conference call during which he provided the wrong answers. I don't necessarily expect anybody to be able to answer all our questions off the top of their heads, which is why I think that detailed technical discussions like this are far better conducted in email, where people have time to think about what they're saying, and time to doublecheck their ideas. Given this new information from Rational, somewhere in the neighbourhood of 8 or so people essentially wasted the better part of an hour during the last call, because nobody came prepared with the answers to the obvious questions (the ones I ask above).

--Doug.

4. Response from Rational on Follow-up Questions

From: IBM Rational Software Customer Services (NA)
Sent: Friday, May 09, 2003 1:14 PM
To: Dugal, Pat
Subject: [SR#173382749] -- Performance issues using TAS that did not exist with NFS Maestro

Dear Patrick,

I wanted to update you:
Regarding the split pool configurations with CIFS.
The write up for this has not yet been approved for distribution to our customers, but I hope to have it soon.

Product management and the engineering team feel that your questions about the inner workings of the MVFS are out of the scope of the particular RFE that we have already identified. As such, we will not be addressing these specific questions.

Best regards,

Senior Technical Support Engineer, ClearCase
Rational Software
IBM Software Group
Phone: (800)433-5444
Email: support@rational.com

Appendix E – Build Engine Optimization

1. Class path may need to be changed

You can see what is happening by looking at the paths that show up when you perform a build outside ClearCase with a copy of the required VOB on an SMB share. The stats for that build are as follows:

```
Same Java build in CIFS/SMB environment - no MVFS
=================================================================
IO-USERS Statistics:

            client <-> server

   |       <-      | |       ->      | |     Total     |
   | Frames  Bytes | | Frames  Bytes | | Frames  Bytes |
   218301  37440956  227593  73935315  445894 111376271

Total elapsed time:   812.747238  (versus 902 seconds in a local view)

=================================================================
SMB RTT Statistics:
Filter:
Commands                    Calls   Min RTT   Max RTT   Avg RTT
Delete Directory                4   0.00550   0.00790   0.00633
Close                        2870   0.00027   0.27456   0.00339
Delete                        139   0.00681   0.02075   0.00918
Write                         140   0.00242   0.00723   0.00352
Transaction                     4   0.00093   0.00105   0.00099
Echo                           25   0.00003   0.00133   0.00016
Read AndX                   31745   0.00064   0.52726   0.00296
Write AndX                   1097   0.00025   0.013385
7   0.00305
Tree Disconnect                 1   0.00019   0.00019   0.00019
Negotiate Protocol              2   0.00024   0.00026   0.00025
Session Setup AndX              2   0.00112   0.00114   0.00113
Logoff AndX                     1   0.00026   0.00026   0.00026
NT Create AndX               2887   0.00019   0.25849   0.00467

Transaction2 Commands       Calls   Min RTT   Max RTT   Avg RTT
FIND_FIRST2                  4655   0.00274   0.09051   0.00421
FIND_NEXT2                      2   0.07156   0.08061   0.07609
QUERY_FS_INFORMATION           38   0.00011   0.00699   0.00239
QUERY_PATH_INFORMATION     140758   0.00249   0.53404   0.00345
QUERY_FILE_INFORMATION      18554   0.00220   0.03255   0.00258
SET_FILE_INFORMATION          143   0.00258   0.06332   0.00499
```

The highest percentage of QUERY_PATH_INFORMATION calls are for jar files which are in the classpath. There are on average 3204 calls for each of the 34 jar files.

The rest of the files are source files stored under a different directory. And there are on average 12 calls per .java file, but the mean might actually be 10 or 12 calls per file. There are 2500 .java and other types of files.

If you add up the number of total number of calls, for all files, you get ~140K, which matches what Ethereal is giving me. Now if you only add up the calls for files being fetched from the lib directory where the jar files are, there are ~120K. If you then multiply that by the Avg RTT (3.5 ms) for that call, you get 414 seconds out of a build that took ~810 seconds. And for the leftover 20K calls, just over a minute is being spent on regular .java source files.

So more than half the total build time is being spent on calls for the java library files. The same files are being queried for attributes with QUERY_FILE_INFORMATION when performing the build inside a local dynamic view, although there are twice as many calls in that case. So what we learned from doing a build outside ClearCase in terms of which files are involved in most of the queries, is probably also valid for builds inside ClearCase, meaning that for a build that takes ~20 minutes, approximately 10-12 minutes is being spent on the above files listed contained in the lib directory.

We wonder why the jar and zip files are being queried for attributes so often. Although it would seem kludgy, perhaps making a local copy of these libraries and pointing the class path to them would help eliminate most of the difference in build time in the CIFS/SMB environment over that of NFS Maestro. The other work-around solution that people don't seem to mind using is snapshot views, despite its drawbacks.

2. Jikes may be more efficient than the Sun JDK inside ClearCase

Excerpt from Doug Graham's e-mail
Sent: Wednesday, April 23, 2003 1:37 AM

```
Another thing you could try is using the jikes Java compiler instead.
It seems to be much more intelligent about reading in and caching
directory contents and file attributes, so that it doesn't need to
keep making calls to GetFileAttributes() to get them again.  I'm not
so sure about Clearcase, but I'm pretty much certain that it's those
GetFileAttributes() calls that are killing you on SMB.  I would expect
that, based on the Win32 calls that it's making, jikes should perform
much better than javac inside a Clearcase view, but I also remember
that I did try jikes at one point inside Clearcase and it didn't help
much.  But that was probably with an older version of jikes, one where
they may not have added the optimizations yet.  I was testing just now
with version 1.14 of jikes, but 1.18 is available.

Here's the SMB summary for javac:
```

```
   ===================================================================
   SMB RTT Statistics:
   Filter:
   Commands                    Calls   Min RTT   Max RTT   Avg RTT
   Close                         627   0.00017   0.00061   0.00027
   Read AndX                    1472   0.00013  14.13344   0.01009
   NT Create AndX                634   0.00024   0.00065   0.00035

   Transaction2 Commands       Calls   Min RTT   Max RTT   Avg RTT
   FIND_FIRST2                 25582   0.00025   0.00363   0.00044
   FIND_NEXT2                      3   0.00036   0.00181   0.00123
   QUERY_FS_INFORMATION         1587   0.00013   4.04616   0.00278
   QUERY_PATH_INFORMATION     112949   0.00015   0.10128   0.00022
   QUERY_FILE_INFORMATION       3162   0.00012   1.54846   0.00071

   NT Transaction Commands     Calls   Min RTT   Max RTT   Avg RTT
   ===================================================================
```

and here it is for jikes:

```
   ===================================================================
   SMB RTT Statistics:
   Filter:
   Commands                    Calls   Min RTT   Max RTT   Avg RTT
   Close                        2585   0.00017   0.00385   0.00021
   Read AndX                    2758   0.00013   0.00292   0.00041
   NT Create AndX               2593   0.00026   0.00070   0.00030

   Transaction2 Commands       Calls   Min RTT   Max RTT   Avg RTT
   FIND_FIRST2                  6492   0.00024   0.00369   0.00069
   FIND_NEXT2                      6   0.00033   0.00184   0.00122
   QUERY_PATH_INFORMATION       6492   0.00018   0.00083   0.00027

   NT Transaction Commands     Calls   Min RTT   Max RTT   Avg RTT
   ===================================================================
```

Here are the high runner system calls for javac:

```
      total    total       total
    seconds  usecs/call    calls  procedure
   ---------- ---------- --------- ---------
     53.120714        527    100665 GetFileAttributes
     25.126398       1077     23324 FindFirstFile
     14.520694       2497      5813 CreateFile
     13.885565          2   5831136 IsDBCSLeadByte
      9.688184       1359      7127 GetFileInformationByHandle
      7.437067       2484      2993 WriteFile
      7.176955        456     15737 ReadFile
      4.097895        986      4155 SetEvent
      2.745178        473      5799 CloseHandle
      1.798993       4263       422 CreateDirectory
      1.589742         68     23320 FindClose
```

and here they are for jikes:

total seconds	total usecs/call	total calls	procedure
13.144443	1235	10639	FindFirstFile
3.266546	597	5464	CreateFile
1.485625	271	5462	CreateFileMapping
1.430201	130	10925	CloseHandle
0.864117	110	7832	SetCurrentDirectory
0.571244	55	10318	FindClose
0.506636	92	5462	UnmapViewOfFile
0.193498	35	5462	MapViewOfFile
0.172698	18	9406	FindNextFileA
0.105956	3	28425	FileTimeToSystemTime
0.074245	2	28425	FileTimeToLocalFileTime
0.054940	20	2702	GetFileSize
0.032662	10887	3	ReadFile

Jikes appears to be reading all the directories and file attributes with a few calls to FindFirstFile(), and caching the information. Javac made 100K calls to GetFileAttributes(), which is pretty dumb of it, and which roughly accounts for the 112K QUERY_PATH_INFORMATION transactions. I can't remember for sure at the moment, but I think that those calls to GetFileAttributes() are expensive inside of Clearcase as well; they result in QUERY_FILE_INFORMATION SMBs. If that's true, then jikes *should* perform a lot better inside a Clearcase dynamic view than does javac. Will have to give that a try again when I get time.

BTW, these tests were run on my home network, so the round-trip-times are much less than they are at work. I was also writing the output class files to local disk.

--Doug.

Whitepaper

Managing Change for Growing Companies

www.cmi.com

About Configuration Management, Inc.

Configuration Management, Inc. is a leading provider of Enterprise Change Management and Software Configuration Management services. Founded in 1992, its standards-based methodology helps organizations manage change across all their digital assets, including software code, software packages, web content and documentation. These services streamline the development process to increase ROI and compress time-to-market. Headquartered in Red Bank NJ, Configuration Management, Inc. has offices worldwide.

For additional copies of this whitepaper, or to explore partnership opportunities with CMI, please contact us as follows.

Configuration Management, Inc.

140 Broad Street
Red Bank, NJ 07701

(732) 450-1100 *tel*
(732) 450-0715 *fax*

info@cmi.com