Lessons Learned Tuning HP's SMB Server with the FSCT Benchmark

Bret McKee
Hewlett-Packard
bret.mckee@hp.com

Vinod Eswaraprasad
Wipro Technologies
vinod.eswaraprasad@hp.com

Disclaimers

- There are many people authorized to make official statements for Hewlett-Packard and Wipro. We are NOT those people. Opinions expressed here are strictly those of the authors.
- All brands and trademarks mentioned are the property of their respective owners.
- No electrons were injured during the preparation of this presentation.

About the Authors

Bret McKee

- Worked at Hewlett-Packard > 25 Years
- Unix kernel – FS/PM/VM/utilities
- System design for performance
- HP Labs
- Storage

About the Authors

Vinod Eswaraprasad

- 15 Years at Wipro Technologies
- Operating systems for fault tolerant servers
- HAL, Kernel, and File Systems
- 4 Years of involvement with SMB

Objectives (of this talk)

- Introduction
  - Brief overview of our hardware
  - Describe our methodology
- Share some things we learned about FSCT
- Conclusion
  - FSCT next steps
  - Describe our results

Viewer Discretion Advised

Warning: The following slide contains an image generated by “Marketing” and may be offensive to some viewers.

Viewer Discretion is Advised

System Overview

- Servers
  - 2 or more servers/system
  - CPU – Intel Xeon
    - 2 Sockets/Server, >= 4 Cores/Socket, Hyper-threaded
    - >= 16 Logical Cores/Server
  - RAM >= 48 GB/Server
- Storage
  - Fibre Channel or built-in controller
  - All tests were run with 96 or 192 HDD (didn't seem to matter)

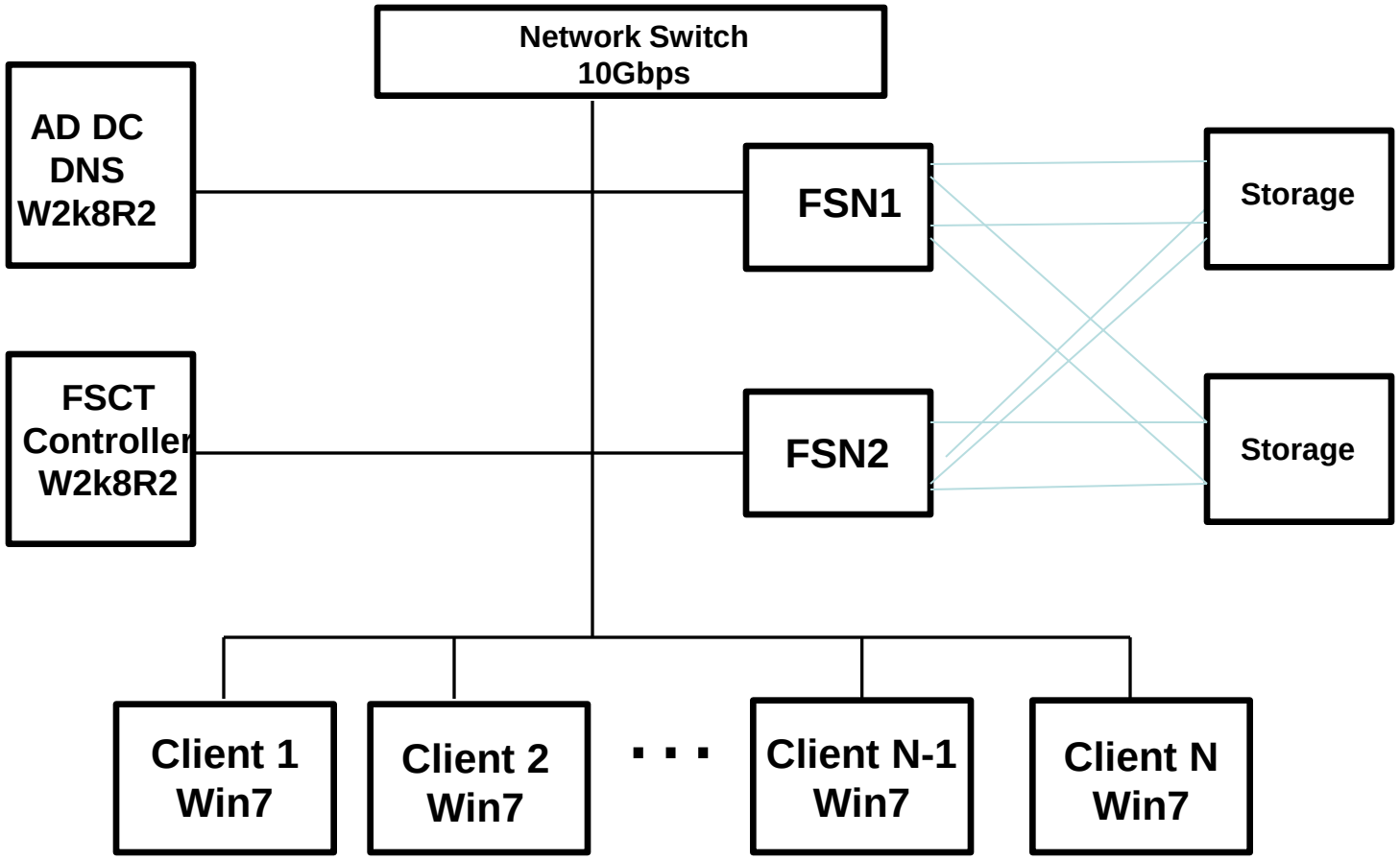
Objectives (of our effort)

- Improve the performance of home directory work load
  - Understand the load placed on the system
    - Use FSCT as the primary benchmark
  - Optimize the protocol server and file system
  - Recommend hardware/system configuration changes if appropriate

FSCT Overview

- File Server Capacity Tool (FSCT) is a Microsoft proprietary benchmark/tool
- Attempts to measure how many users a server will support without “Overloading”
- Created from network traces
- Runs sequences of operations called “Scenarios”
  - There are 12 different scenarios
- Our measurements were on FSCT v1.2
- “Non-trivial” to setup and run

FSCT Hardware Layout



All FSCT systems were 2U servers, 2X4X2 CPU, 48GB Memory

Sample FSCT Output

*** Results

Users	Overload	Throughput	Errors	Errors [%]	Duration [ms]
100	0%	9	0	0%	600556
200	0%	18	0	0%	600166
300	0%	27	0	0%	600228
400	0%	36	0	0%	600291
500	0%	45	0	0%	600353
600	0%	55	0	0%	600213
700	0%	64	0	0%	600119
800	0%	73	0	0%	600556
900	0%	82	0	0%	600588
1000	8%	85	0	0%	608652
1100	31%	79	0	0%	610883
1200	55%	73	0	0%	613333
1300	78%	68	0	0%	615236
1400	121%	60	2	0%	614362

Our Methodology

- Measure, Analyze and *Apply Computer Science*
- Component level instrumentation.
  - Simple and complete set of instrumentation
  - Frequency, time delay, service time
  - Random vs. sequential nature of work performed
  - Layered counters – easy to split and aggregate
    - Provide multiple views of the data
- Understand request/response timing
  - Time spent on queues
  - Time spent in request handling dispatch
  - Time spent in response handling and dispatch
- Iteratively explore and eliminate bottlenecks
- Once you change something, something else rises to the top

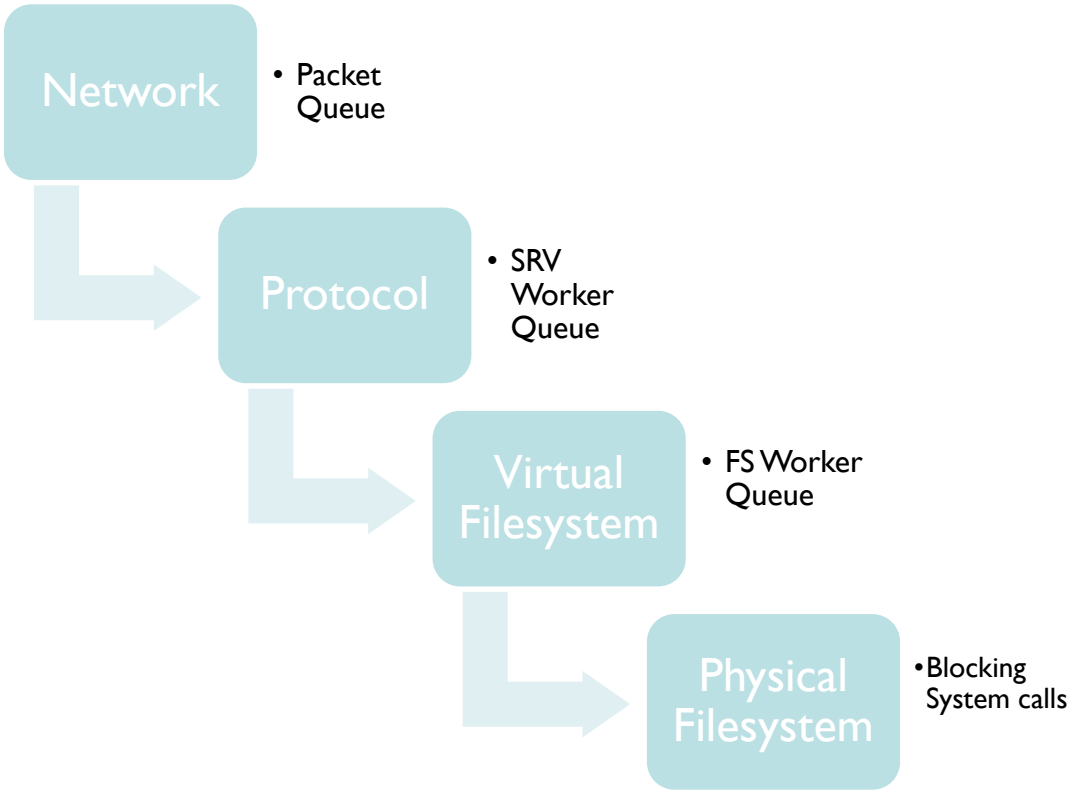
“Apply Computer Science?”

- Changes needed depend on the situation, but often include:
  - Change data structures
    - List -> Tree
    - Tree -> Hash of Trees
  - Change Locking structure
    - Mutex -> Reader/Writer Lock
    - Reader Writer Lock -> array of Locks
    - Single Lock to Many
  - Change Algorithm
  - Improve caching efficiency
  - Etc.

FSCT On Disk footprint

- An FSCT User's on disk data consists of:
  - 92 Directories
    - Nesting Depth is 3
  - 290 Files
    - Total Size is ~ 85MB
    - Average Size is ~307K
    - Minimum Size is ~50 bytes
    - Maximum Size is ~6MB
  - ~3 files/directory

SMB Request Lifecycle



Sample Queue Data

Protocol thread pool - queue depth requests

depth	count	percent
0	0	0.000%
1	44,776,384	47.044%
2	23,666,803	24.866%
4	6,590,288	6.924%
8	2,540,343	2.669%
16	1,915,186	2.012%
32	1,828,684	1.921%
64	1,989,446	2.090%
128	2,450,090	2.574%
256	2,639,634	2.773%
512	2,647,632	2.782%
1024	2,011,154	2.113%
2048	1,915,932	2.013%
4096	194,559	0.204%
8192	12,988	0.014%

Sample SMB Op data


```
SMB2 Create - service time seconds
samplesRecorded=11,487,498 samplesNotRecorded=0
min=0.000000 max=5.565202 ave=0.104203
total=1,197,036.001
 distribution
   Time              Count % of count % of time
   0.000032             56    0.000%    0.000%
   0.000064         26,598    0.232%    0.000%
   0.000128        185,239    1.613%    0.003%
   0.000256      1,039,569    9.050%    0.033%
   0.000512      2,162,783   18.827%    0.139%
[snip]
```

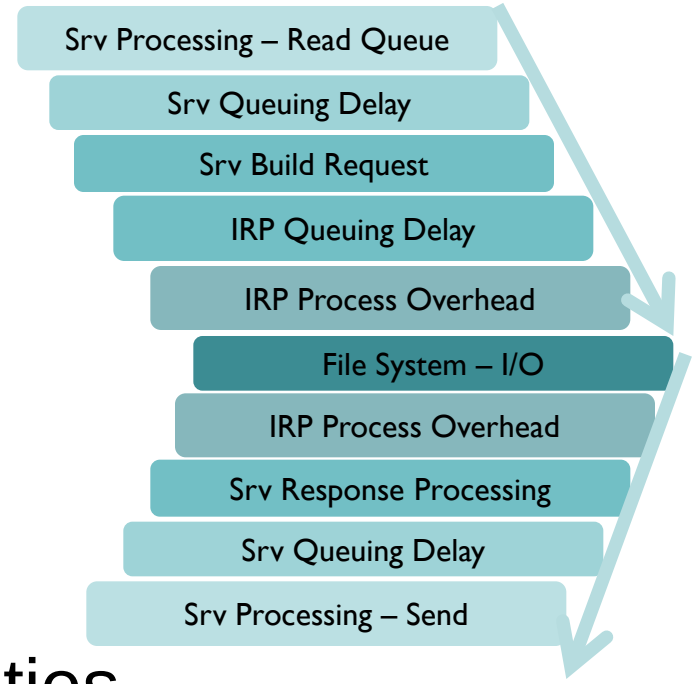
Instrumentation at every layer

- Instrumentation at several layers to get visibility into runtime behavior and bottleneck analysis
- Measure round-trip of SMB requests and IRPs
- Measure work item service time, work item wait time, IRP service time and IRP wait time

- Protocol layer interactions with kernel and filesystem
- Measure the system call response time
- Insight into non-obvious behavior of FSCT workload

Know what your computer is doing

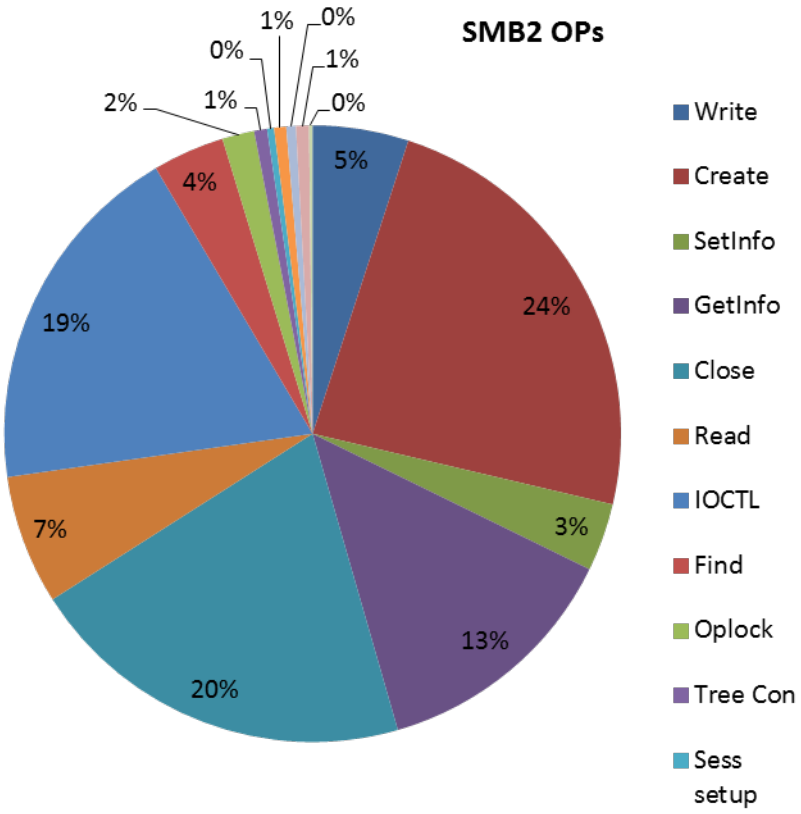
- What does the SMB server do internally?
  - Nature of home folder workload
  - Workload scenarios
  - Are they really rare operations?
    - Create/Close
    - Query Volume Info

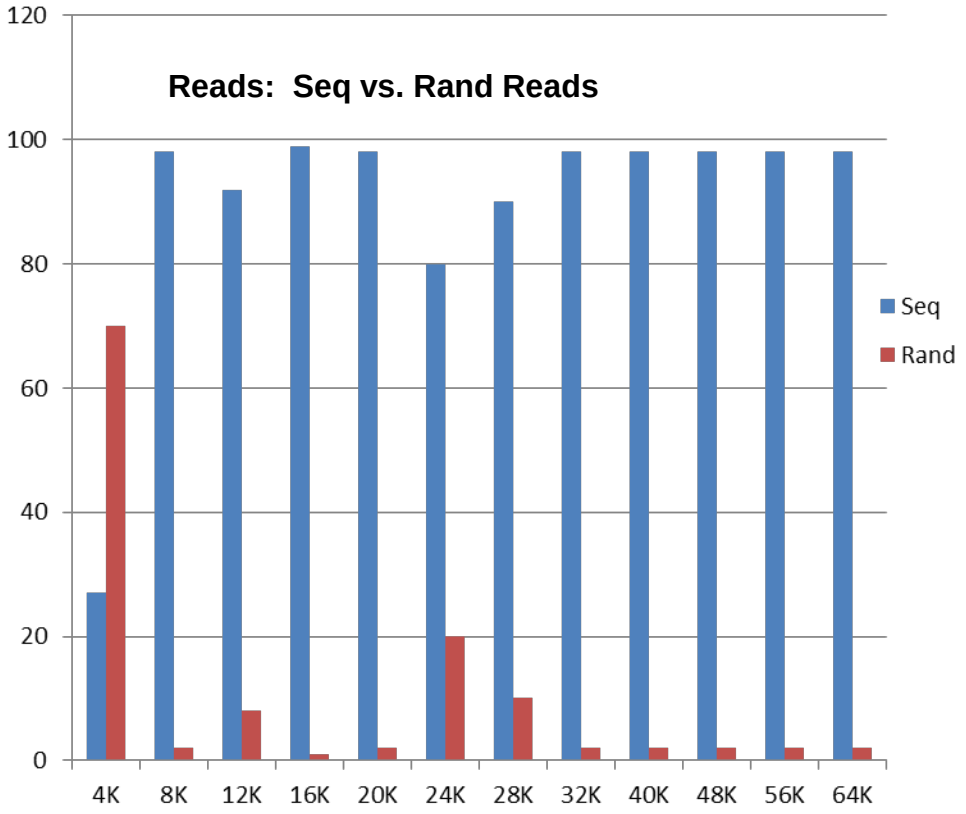


- Obvious optimization opportunities specific to workload characteristics

What does it do?
Analysis of the workload

- More metadata operations than read/write ops
- Mostly sequential reads and writes
- More than 90% of the reads and writes were above 64K





Graph Highlights

- By Operation counts, 90% covered by:
  - Create is 24%
  - Close is 20%
  - Ioctl is 19% (QueryVolumeInfo mostly)
  - Tree Connect is 13%
  - Read is 7%
  - Write is 5%
  - Find is 4%
- Notice that only 12% are Read/Write

FSCT I/O Characteristics

- Read
  - Average size is ~40K
  - Average rate is ~15K/second/user
- Write
  - Average size is ~61K
  - Average rate is ~13K/second/user

FSCT is NOT a disk benchmark…

- FSCT is metadata heavy
  - Disks do get hit, but not for data
- File/directory metadata retrieval are key to performance
- The file system metadata operations (open, stat, etc.) are more important than the read/write operations
- The protocol server, kernel and file system eventually become CPU bound

CPU Tuning -Two Phases

1. Remove lock contention to allow saturation of the CPUS
   - During this phase, generally “making things faster” does not get any more users
   - Balance thread pool sizes
2. Shorten paths to allow more throughput
   - Once CPUs are saturated, the only way to get more users it to reduce per request CPU usage

FSCT is Highly Parallel

- FSCT schedules scenarios based on frequency so that it can do 1 scenario every 11 seconds
- If the “back end” is slow, a huge number of requests will queue at the “front end”
- At 5000 users, we see queue lengths of
  - ~1-2 SMB requests for 0% overload runs
  - > 8K SMB requests when there is overload

FSCT is Throttled

- FSCT throughput is throttled (per user)
  - Making paths faster shows no immediate results like it does in unthrottled benchmarks
    - However, it will allow you to increase the number of requests/second that can be serviced
    - May require increasing thread counts

FSCT Maps/Unmaps shares!

- FSCT does not keep shares mapped
- Shares are mapped/unmapped for every scenario
  - Session setup time and authentication delays can be significant
  - 4% of the total SMB Requests are tree connect/disconnect
  - Query Volume Info request is executed frequently
    - 12 % of the IRPs are query volume info.
- Large burst of these operations during a test start/stop
  - Not an anticipated scenario during normal testing/use

SMB protocol features and FSCT behavior

- Experimented with File and Directory Leases
  - Leases
    - Did not see differences in overload point
    - Very small Ops reduction observed - Create
  - Directory Leases
    - 20-30% reduction in certain metadata Ops
      - Create and Close Ops – dropped by (20%)
      - Find Ops – dropped by 31%
    - Equivalent frame reduction was observed
    - However, the overload point remained same
      - More work during child object state changing ops
- The overall performance did not change

Errors matter

- The test and overload numbers are meaningful only with near zero error counts during test scenarios
- Reaching 0 errors caused a 33% reduction in our users without overload number
  - Our initial benchmark numbers were invalid
- Investigate all the errors in the summary and detail reports and eliminate them
  - Example: FSCTL_CREATE_OR_GET_OBJECT_ID IOCTL related errors
  - Turn on compatibility mode

That Overloaded "overload" Term…

- FSCT Performance is measured by the number of users supported without "overload"
- The definition of overload is not simple, and it is not obvious how to address issues with it
  - Ability to schedule/complete 1 user scenario in 11 seconds
  - 1-2% overload for a range of users, after a point
  - Remained after all bottlenecks were avoided
  - No direct correlation to the measurements RTT or latency
- More visibility into the overload state would be useful…

Increased throughput ⇨ Increased stress

- FSCT was a good system test to flush out issues at high load
  - A multi-threaded daemon to service requests
  - Multi-thread race vs. Multi-process overhead
- Exposed race conditions in simultaneous overlapping/request handling
- Causes sequences/patterns that are not caused by regular IO workload testing
  - Read/Write/Close overlapping requests
  - Durable open re-connect and close races
  - No-waited read requests and their interaction with serialization/lock

Finally, Storage, Network, or CPU bound?

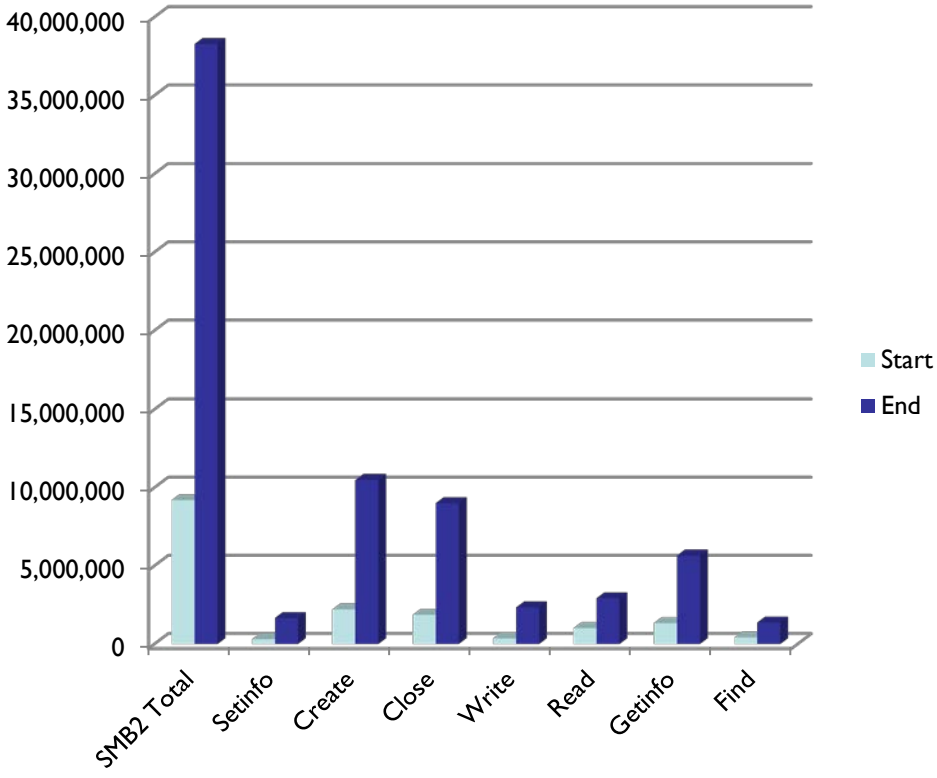
- In the limit FSCT is a CPU benchmark
- Metadata related operations
  - Large number of system calls in a Linux implementation
  - Interaction with file system becomes key
  - Tends to become CPU bound as other bottlenecks are eliminated
- High level of lock contention
  - Create/Close processing rate
  - Starts making the synchronization locks hot

Future Home Directory workload benchmark?

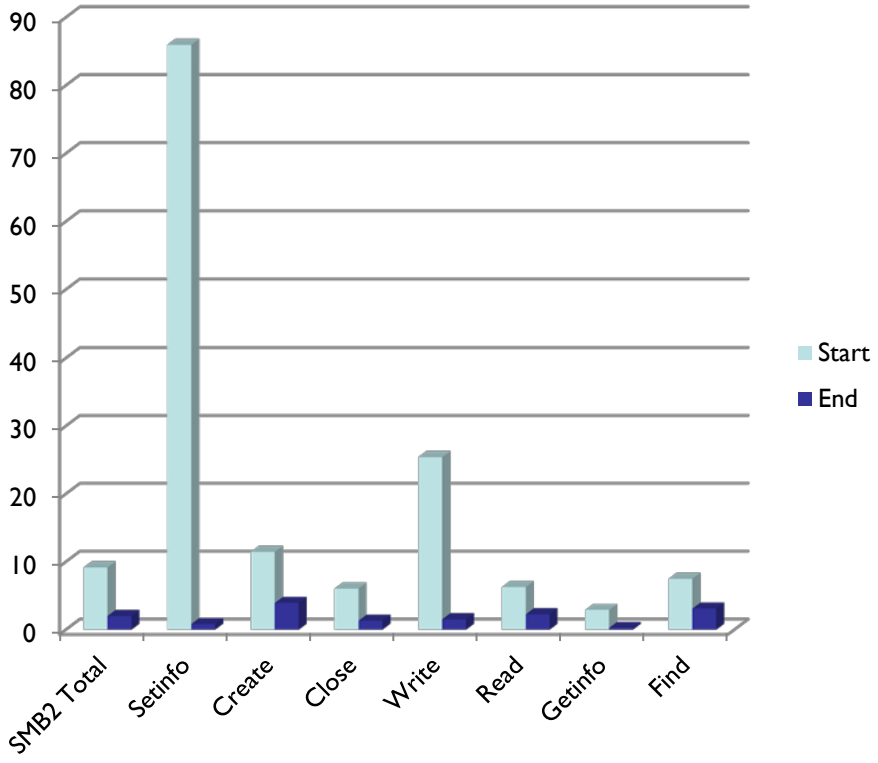
- FSCT model is based on measurements from 2006
- Multi-protocol environment? Authentication and authorization complexity in the FSD?
- Need newer workload scenarios such as streaming to model modern Home Directory workload?
- How about Virtual machine images, other “block like” SMB3 workloads?
- Better logging
- More transparent reporting
- Easier to setup/run
- Other topics?

Results

Number of Operations in 60 minutes at overload



Average time in milliseconds per SMB operation



Lessons and Results

- Lessons
  - Instrumentation is important
  - Reach 0% errors before measuring overload
  - Serves as a regression suite for all the changes that deal with the data path
  - FSCT IS NOT AN IO BENCHMARK!!!
- Primary results
  - ~8x improvement on our SMB stack (~800 -> ~6400)
- Secondary results
  - ~3x improvement in the sequential read performance
  - Improved Stability – Found and fixed some rare race conditions in the product

Questions?

Thank You