
Perforce
User Conference
2010

Perforce Tunables

Michael Shields
Performance Lab Manager
Perforce Software

Why Tunables?

- Varied hardware and operating systems
  - Not everyone running on latest and greatest
- Introduced in 2008.2 release
  - A few added with each subsequent release
    - e.g. throttling new features, if necessary

What are Tunables?

- Each provides a value used in Perforce
  - e.g. number of bytes or seconds
  - Possibly used in multiple Perforce products
- Each has default, minimum, and maximum
  - Some also have an increment
  - e.g. net.tcpsize
    - Default: 32KB, minimum: 1KB, maximum: 4MB
    - Increment is 1KB
      - e.g. 65,535 silently adjusted up to 65,536

Working with Perforce

- Customer benefits
  - Leverage others working with same tunable
  - Alerted to potential negative consequences
  - Perhaps better solution other than a tunable
- Perforce benefits
  - Understanding tunable under varied conditions
  - Change default if consistently better behavior

Tunables Not Supported!

- Perforce can't test all combinations of tunables
- We want to help customers work with tunables
  - Technical Support
    - Support agreement must be current
  - Consulting
    - Active engagement
  - Technical Support and Consulting have access to Performance Lab

Methods for Specifying Tunables

- Command line

```
p4d ... -v dbopen.nofsync=1 -v dm.batch.domains=1000 ...
```

- Extension of command line

```
export P4POPTIONS="... -v lbr.bufsize=16384 -v net.tcpsize=262144 ..."
```

- Debugging options

```
export P4DEBUG="... db.isalive=20000 dm.isalive=100000 ..."
```

- P4CONFIG file (as of 2009.2/226798)

```
...
filesys.binaryscan=16384
filesys.bufsize=8192
...
```

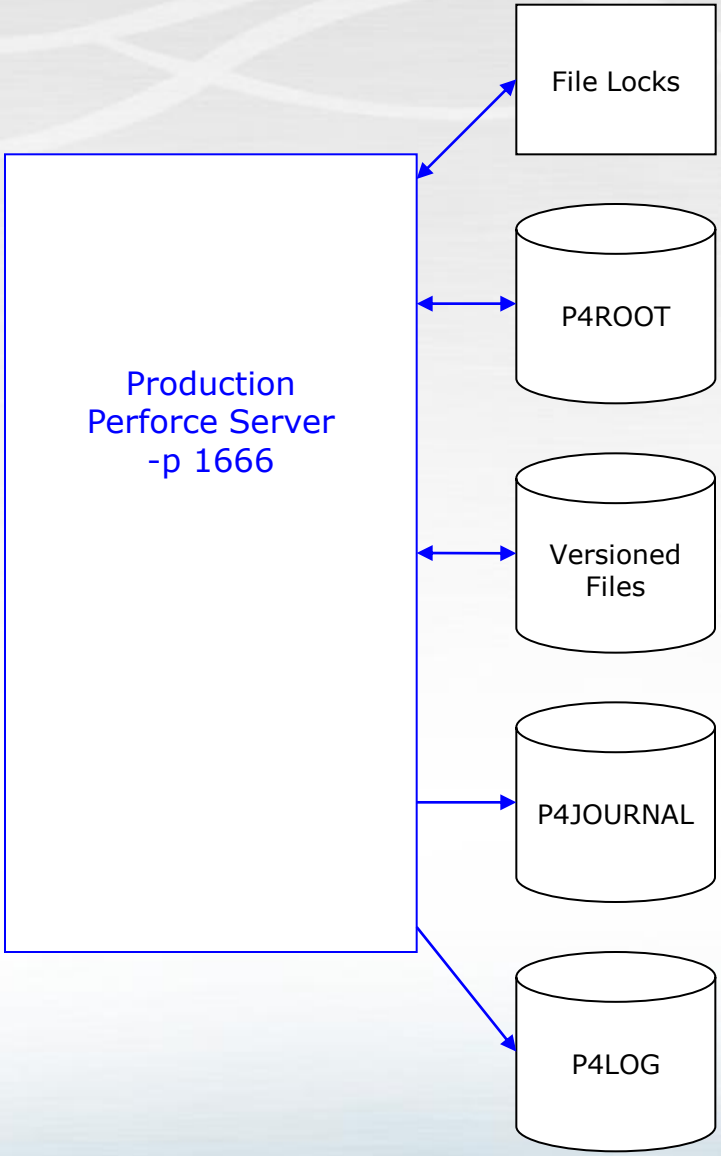
Product	Methods for Specifying Tunables
Perforce Server	P4DEBUG environment variable or registry key specified as **`tunable-name=value ...`** -v command line arguments
Perforce Visual Client	P4CONFIG file
Perforce Command-Line Client	P4CONFIG file -v command line arguments
Perforce Proxy	P4DEBUG environment variable or registry key specified as **`tunable-name=value ...`** P4POPTIONS environment variable or registry key specified as **`-v tunable-name=value ...`** -v command line arguments
Perforce Web Client	P4CONFIG file -v command line arguments
Perforce FTP Plug-in	-v command line arguments P4FTPDEBUG environment variable or registry key specified as **`tunable-name=value ...`** P4CONFIG file
Perforce C/C++ API	**`#include "debug.h"`** **`...`** **`p4debug.SetLevel( "tunable-name=value" );`** P4CONFIG file

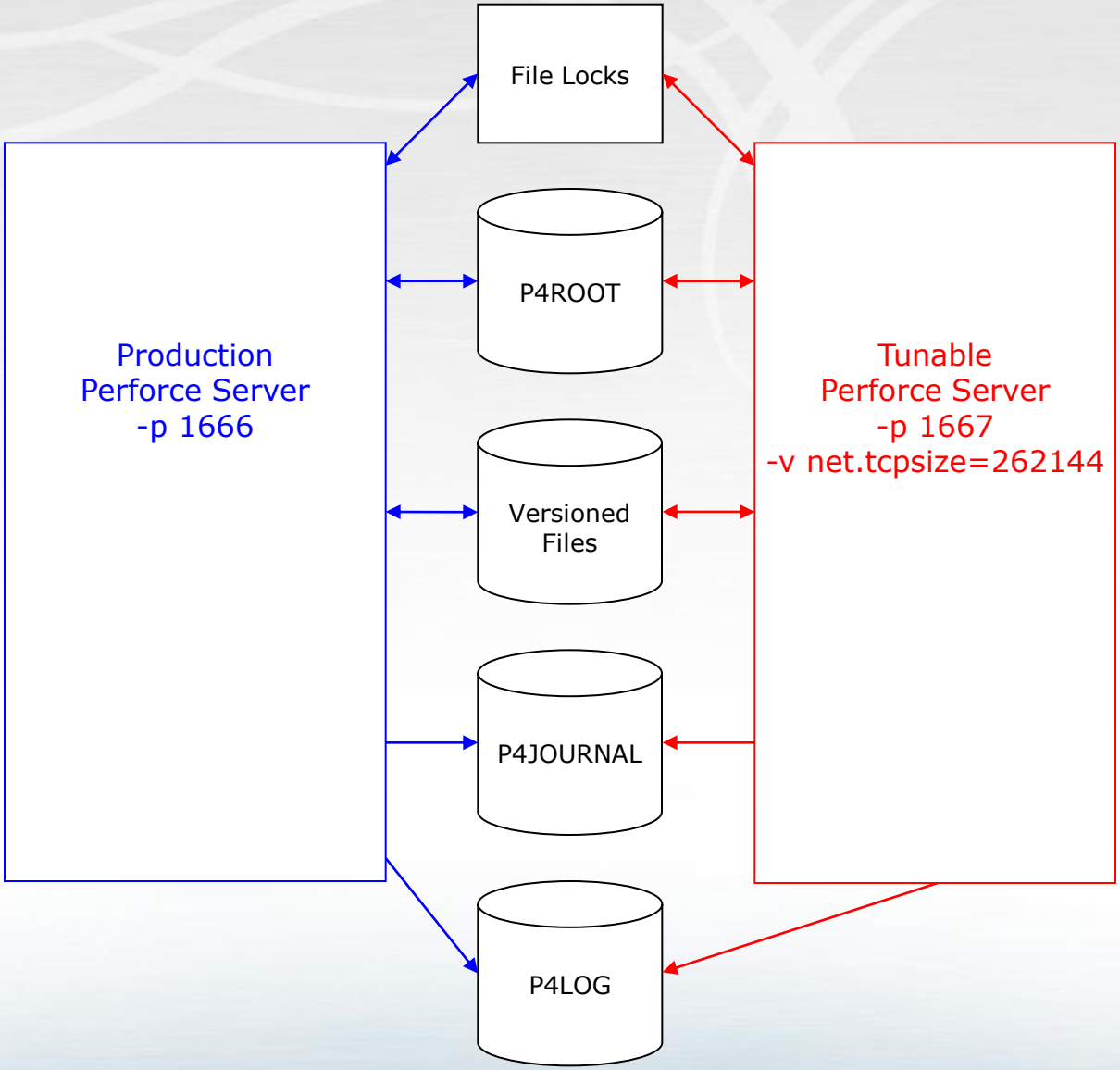
Other Mechanics

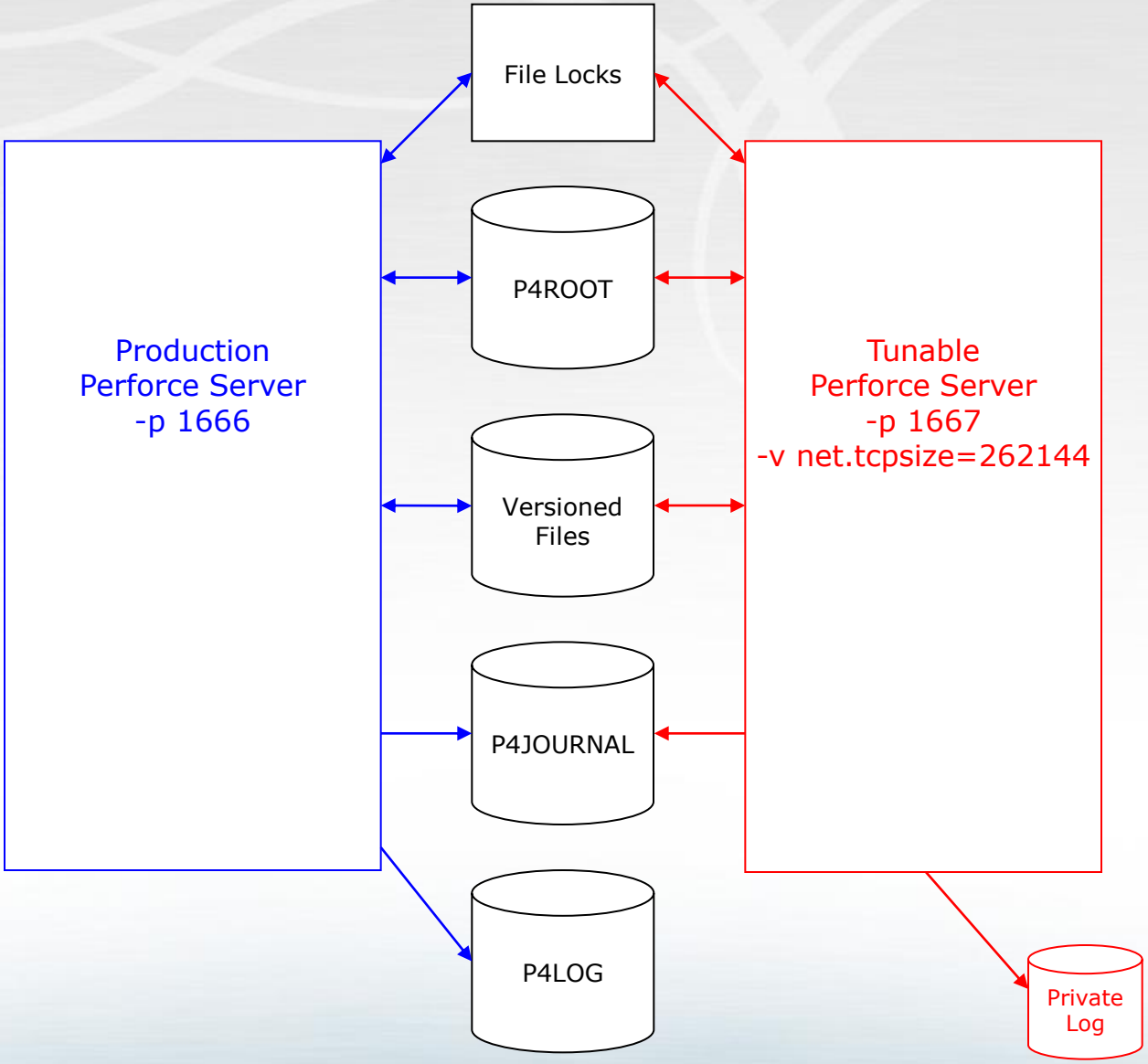
- Values can be specified with “K” or “M” suffix
  - “K” is 1,000 (10^3) or 1,024 (2^10)
  - “M” is 1,000,000 (10^6) or 1,048,576 (2^20)
  - Powers of two for number of bytes
  - Powers of ten for other tunables
- p4 tunables [-a]
  - Show [all] tunable values

Implementing Tunables in Production

- Research and test first!
  - Best if testing on a production snapshot
  - Don’t test on production machines
- Discuss with Perforce
- Then, …


File Locks
Production
Perforce Server
-p 1666
P4ROOT
Versioned
Files
P4JOURNAL
P4LOG


File Locks
Production
Perforce Server
-p 1666
P4ROOT
Tunable
Perforce Server
-p 1667
-v net.tcpsize=262144
Versioned
Files
P4JOURNAL
P4LOG


File Locks
Production
Perforce Server
-p 1666
P4ROOT
Tunable
Perforce Server
-p 1667
-v net.tcpsize=262144
Versioned
Files
P4JOURNAL
P4LOG
Private
Log

Final Steps to Production

- At least two options
  - Specify tunable in production instance
  - Keep new instance as part of production
- Production might consist of multiple instances
  - Each instance tuned for specific purpose

Survey of Tunables

- As of 2009.2
- Interesting tunables shown here
- Whitepaper has more tunables
  - With complete descriptions

dbopen.cache[.wide]

Default: 96[192] pages Minimum: 1 page Maximum: 2^31-1 pages

- Each db.* file has a private cache
  - For each db.* file opened by a process/thread
- Increasing might benefit some commands
  - Repeatedly reading the same rows
- Increasing might decrease other resources
  - e.g. operating system's filesystem cache

dbopen.nofsync

Default: 0 (do fsync) Minimum: 0 (do fsync) Maximum: 1 (don't fsync)

- fsync() called when modified db.* file closed
  - Provide some assurance that data is "written"
    - Might minimize corruption on OS or power failure
- Stable operating system and reasonable UPS
  - Consider dbopen.nofsync=1
    - OS decides when to physically write modified data
    - I/O bandwidth saved could be non-trivial

dm.{domain|user}.access{update|force}

Default: {300|3,600} seconds Minimum: 1 second Maximum: 2^31-1 seconds

- Access timestamp read with read lock
  - Read lock dropped
- If timestamp accessupdate or more old
  - If less than accessforce old
    - Update timestamp if write lock can be acquired
  - Otherwise
    - Wait for write lock and update timestamp
- Increasing access{update|force} might improve concurrency

dm.flush{try|force}

Default: {100|10,000} confirms Minimum: 1 confirm Maximum: 2^31-1 confirms

- For confirmations of:
  - sync, integrate, submit, revert content transfers
  - Opening files for add, edit, move, or delete
- If dm.flushtry confirmations outstanding
  - If less than dm.flushforce confirmations
    - Update metadata if write locks can be acquired
  - Otherwise
    - Wait for write locks and update metadata
- Increasing dm.flush{try|force} might improve concurrency

dm.quick.*

Default: various Minimum: 1 Maximum: 2^31-1

- dm.quick.* specify buffering upper bound
  - If fully-buffered, release read lock early
  - During compute phase
- Only used for some operations
- Increasing might improve concurrency
  - More memory used, decreasing other resources
    - e.g. operating system's filesystem cache

Tunable	Default Value	Operations	Tables Buffered
dm.quick.clients	10,000,000 clients	`opened -a <path>` `obliterate`	db.domain
dm.quick.domains	1,000,000 labels	`labels <path>`	db.domain
dm.quick.have	1,000,000 files	`sync` `flush`	db.have
dm.quick.integ	1,000,000† files	`integrate`	db.have db.locks db.resolve db.working
dm.quick.resolve	1,000 files	`sync` `flush`	db.resolve
dm.quick.rev	100,000 revisions	`labels <path>`	db.domain
dm.quick.working	1,000 files	`sync` `flush`	db.working

†if defined, maxResults is used as the buffering upper bound

dm.batch.domains

Default: 0 (no batching) Minimum: 1,000 labels Maximum: 2^31-1 labels

- Releases and reacquires read lock on db.label
  - Only for **`p4 labels <path>`** command
  - Every dm.batch.domains labels
  - db.domain and db.rev must be fully-buffered
    - See dm.quick.domains and dm.quick.rev
- Allows faster write lock on db.label by others
  - e.g. **`p4 labelsync`** and **`p4 tag`**
  - Might produce inconsistent results

dm.revcx.thresh{1|2}

Default: {4,000|1,000} revisions Minimum: 1 revisions Maximum: 2^31-1 revisions

- <path>@{=|>|>=}change
  - Notably: @x,@y for x <= y
    - but not: <path>@change
- Position in db.revcx and scan, until:
  - dm.revcx.thresh1 unmapped revisions scanned before dm.revcx.thresh2 mapped revisions
    - Position at <path> in db.rev and scan db.rev
  - Changes specified (or implied) scanned
- For very wide paths and slightly older changes, consider increasing dm.revcx.thresh1

filesys.binaryscan

Default: 8 KB Minimum: 0 bytes Maximum: 2^31-1 bytes

- Client uses when determining filetype
- Specifies portion of file sampled
- Increase if larger sample detects correct filetype

filesys.bufsize

Default: 4 KB Minimum: 1 byte Maximum: 2^31-1 bytes

- Size of general buffers for file I/O
- Used by Perforce Server, Proxy, and clients
- Increasing reduces read and write calls
  - Some additional memory usage

lbr.bufsize

Default: 4 KB Minimum: 1 byte Maximum: 2^31-1 bytes

- Size of buffers for archive file I/O
- Used by Perforce Server and Proxy
- Increasing reduces read and write calls
  - Some additional memory usage

net.backlog

Default: 10 requests Minimum: 1 request Maximum: SOMAXCONN requests

- Maximum pending connections queue length
  - Second argument of listen()
- Used by Perforce Server and Proxy
- Consider increasing if under very heavy load and clients see connectivity errors
  - Might also need to increase OS tunable

net.bufsize

Default: 4 KB Minimum: 1 byte Maximum: 2^31-1 bytes

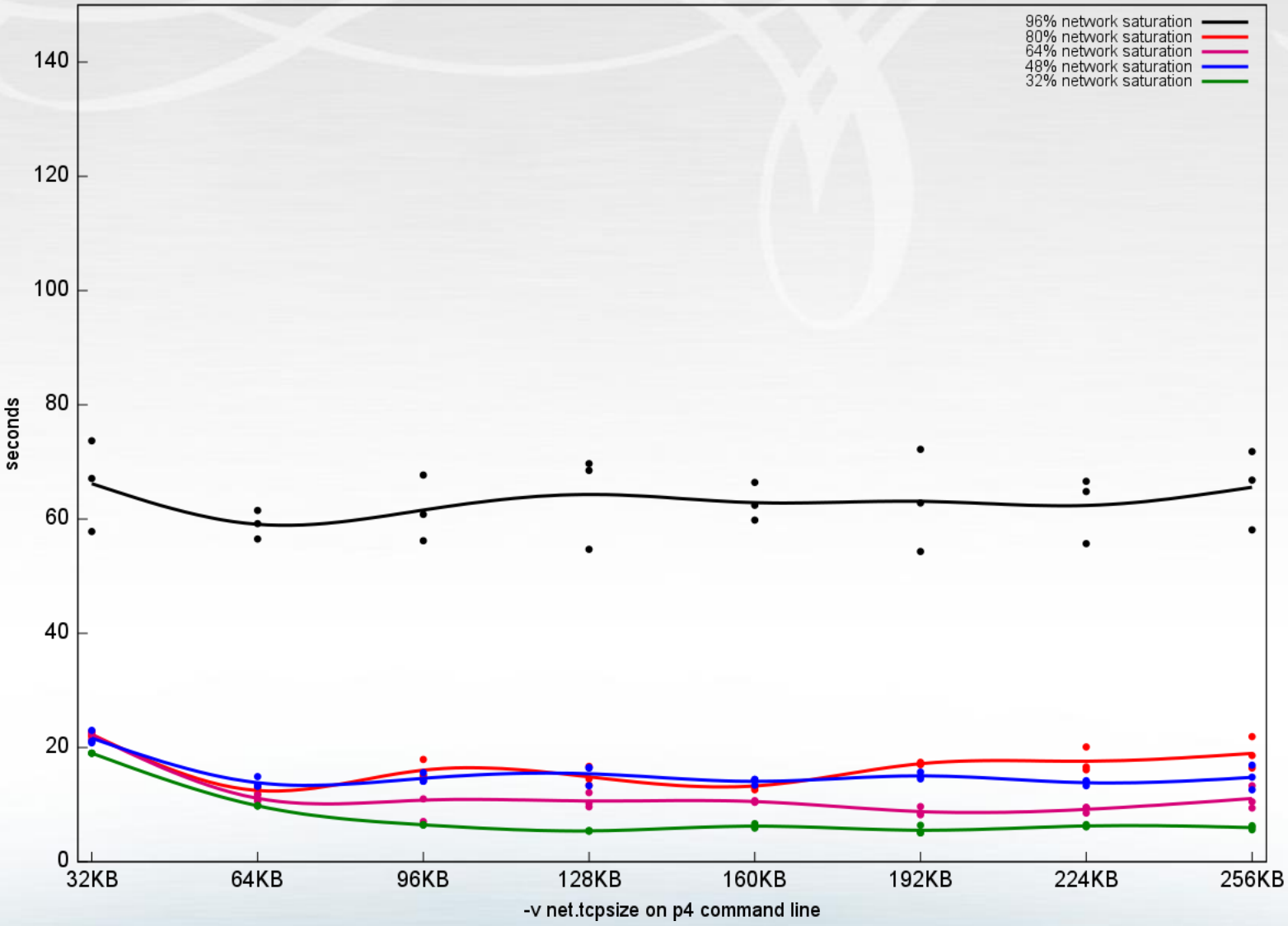
- Initial size of buffers for network I/O
  - Size increased on Perforce Server to maximum of receive buffer sizes and rpc.himark
- Used by Perforce Server, Proxy, and clients

net.tcpsize

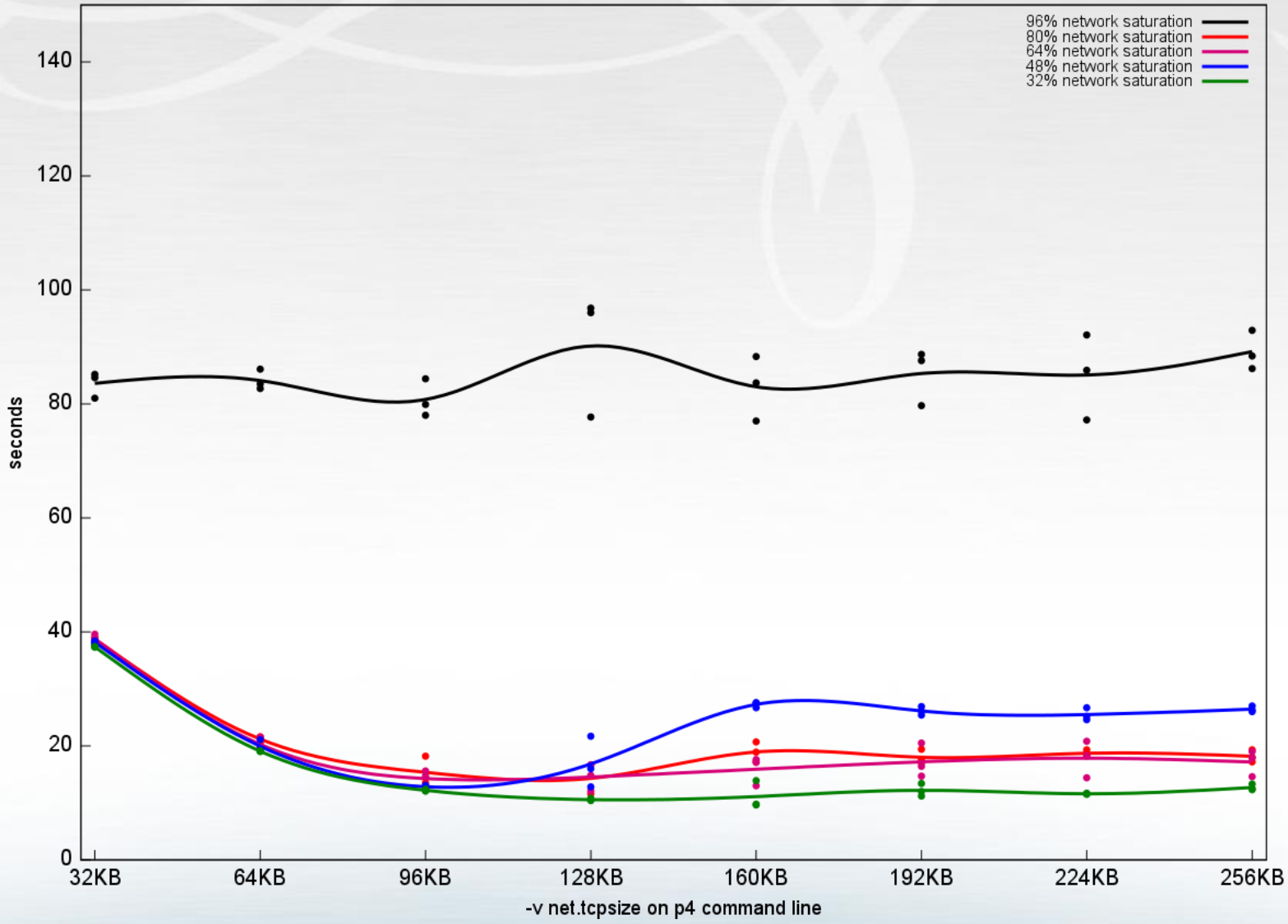
Default: 32 KB Minimum: 1 KB Maximum: 4 MB

- Increases TCP send and receive buffers
  - Unless OS has sized them larger
- Used by Perforce Server, Proxy, and clients
- Increase for high-latency connections
  - Might also need to increase OS tunables

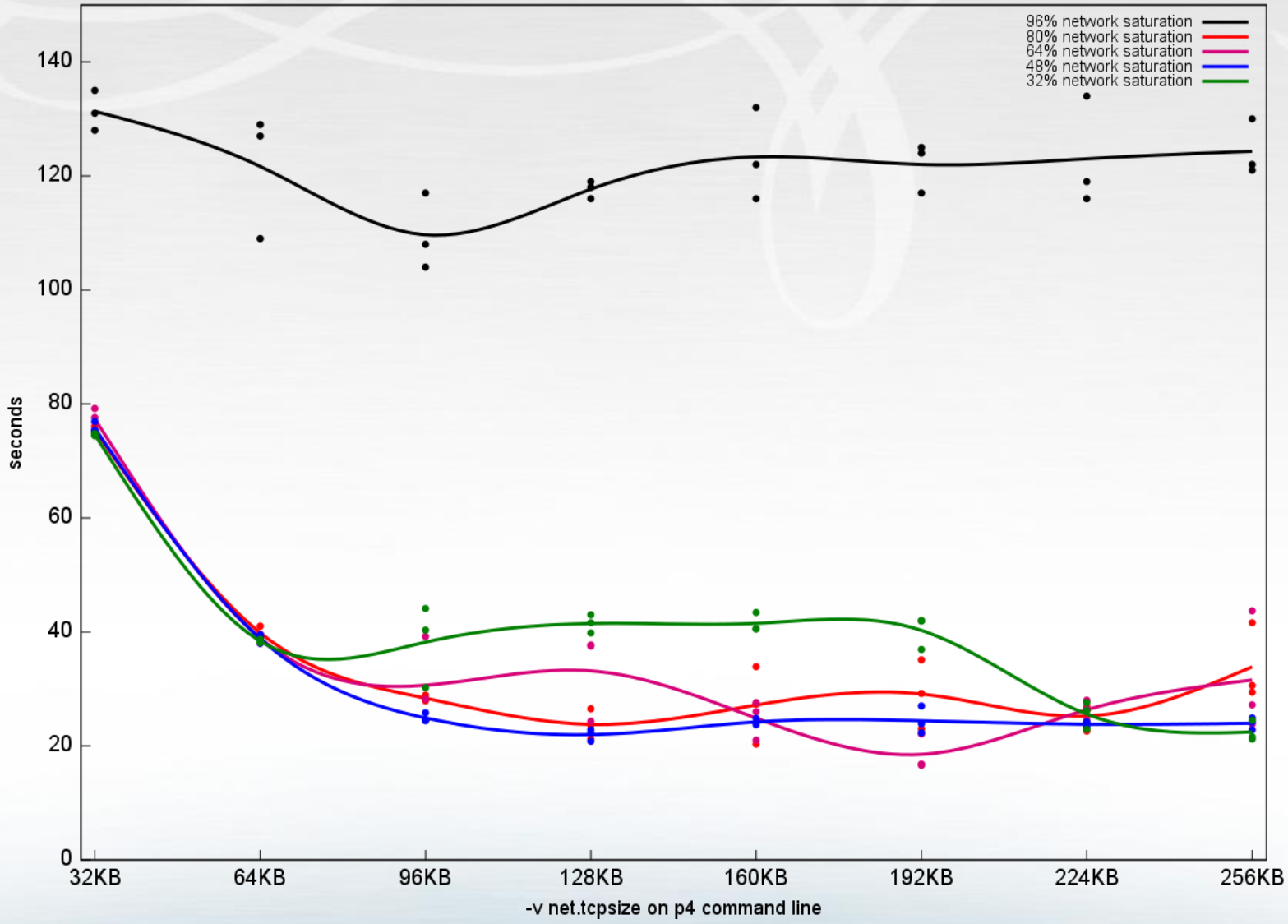
16MB ubinary sync with 32ms round-trip ping



16MB ubinary sync with 64ms round-trip ping



16MB ubinary sync with 128ms round-trip ping



rpc.{low|hi}mark

Default: {700|2,000} bytes Minimum: 1 byte Maximum: 2^31-1 bytes

- If specified, overrides network buffer size mechanics
- As of 2009.2, should not be needed as often
  - Networking layer modifications
- Used by Perforce Server, Proxy, and clients

spec.hashbuckets

Default: 0 buckets Minimum: 0 buckets Maximum: 999 buckets

- Large directories might be problematic
  - Some older filesystems
    - Limited number of directory entries
    - Performance
- Distributes in spec.hashbuckets directories
  - branch, client, label, and job specifications
- Redistribute with **`p4 retype`** command

Summary

- Working with tunables can benefit us all
- Deploy into production carefully, methodically
- Perforce wants to help you!

Questions?


Perforce
User Conference
2010