

FS-Cache aware CIFS

SUSE Labs Conference 2010
Jul 14, 2010

Suresh Jayaraman
SUSE Labs

Novell[®]

Overview

- Why Network filesystem caching?
- Some use-cases/scenarios
- Caching data flow
- FS-Cache
- Cache back-ends
- CIFS current implementation
- Q &A

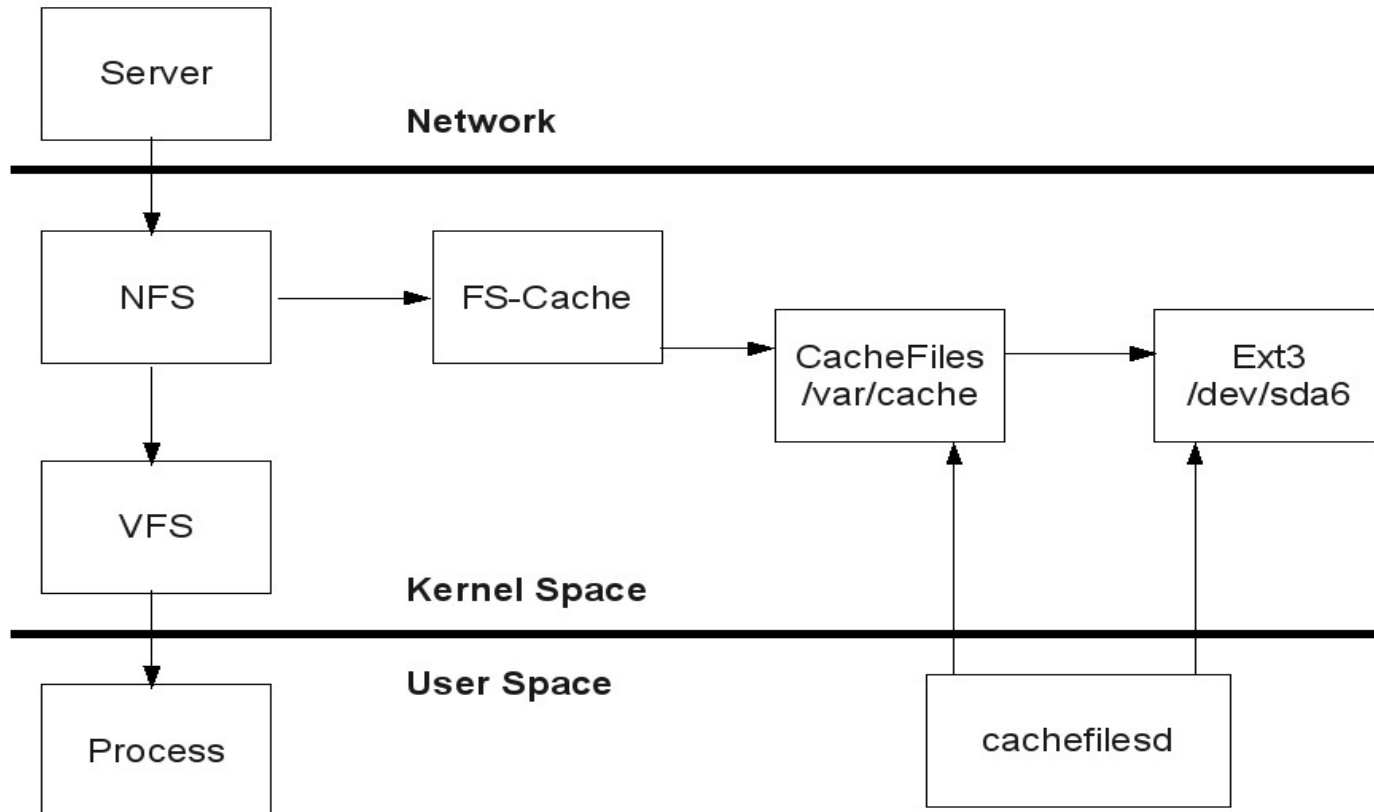
Why Network filesystem caching?

- Network based filesystems over slow network
 - Re-reading over loaded Network/Server can be slow
 - Local caching can help
 - > Reduce Network traffic and Network latency
 - > Reduce Server load
 - > Improve Performance and Scalability
- First step towards making disconnected operations a reality!
 - Still lot needed from Network filesystem
- **Not for every workload or I/O pattern**
 - Trade offs

Some use-cases/scenarios

- Render farms in Entertainment industry
 - Use to distribute textures to individual rendering units
- Read only multimedia workloads
- Accelerate distributed web-servers
 - Web server cluster nodes serve content from the cache
- /usr distributed by a network file system
 - Avoid spamming Servers when there is a power outage
- Caching Server with SSDs reexporting netfs data
- Persistent cache remains across reboots

Caching data flow



FS-Cache

FS-Cache (1)

- Terminology - FS-Cache, CacheFS
- Caching facility for network filesystems, slow media
 - Mediates between netfs and cache back-end
 - Allows persistent local storage for caching data and metadata
 - Cache is transparent
 - Serves out **pages**, does not load file entirely
 - Hides cache I/O errors from the netfs
 - Allows netfs to decide on index hierarchy
 - Index hierarchy used to locate file objects/discard subset of files cached
- Written by David Howells, upstream since 2.6.30

FS-Cache (2)

- Deals with Objects
- Objects have certain attributes
 - Type
 - > Index, Data storage and special
 - > Non-index object can store pages of data
 - Key
 - > Supplied by Netfs, blob of binary data
 - > Used to lookup an object in its parent index
 - > Must be unique in its index's key-space (e.g. share)
 - Auxiliary data (used to validate objects on lookup)
- Filesystem has to be modified to use FS-Cache
- AFS, NFS, Plan 9 (9p) are FS-Cache capable

Cache back-ends

Cache back-ends

- CacheFs (caching filesystem)
 - uses a block device directly rather than a directory under fs
 - can perform its own journaling more efficiently
 - Development stalled due to design issues
- CacheFiles (fs/cachefiles)
 - uses directory on already mounted local filesystem (e.g. ext3)
 - Integrity guarantees only as good as underlying fs
 - /dev/cachefiles as communication channel

CacheFiles

- Cache started by cachedfilesd
- Uses dnotify to monitor and discard inactive objects
- Uses userspace daemon to do reaping of stale nodes and culling
 - Maintains x% of free space (configurable), shrinks
 - Cache culling discards objects on LRU basis
- Configuration file - /etc/cachedfilesd.conf
- Don't create, rename, unlink while cache is active

CacheFiles: Cache structure (2)

/var/cache/fscache

 / \
 cache/ graveyard/

- Active cache objects in cache/
- Retired objects (but not yet unlinked) graveyard/
- Cachefilesd uses dnotify to delete files from graveyard
- Index objects starts with “I...” or “J...”
 - printable Vs encoded
- Data objects starts with “D..” or “E..”
- Special objects starts with “S...” or “T...”

Making a filesystem FS-Cache aware (1)

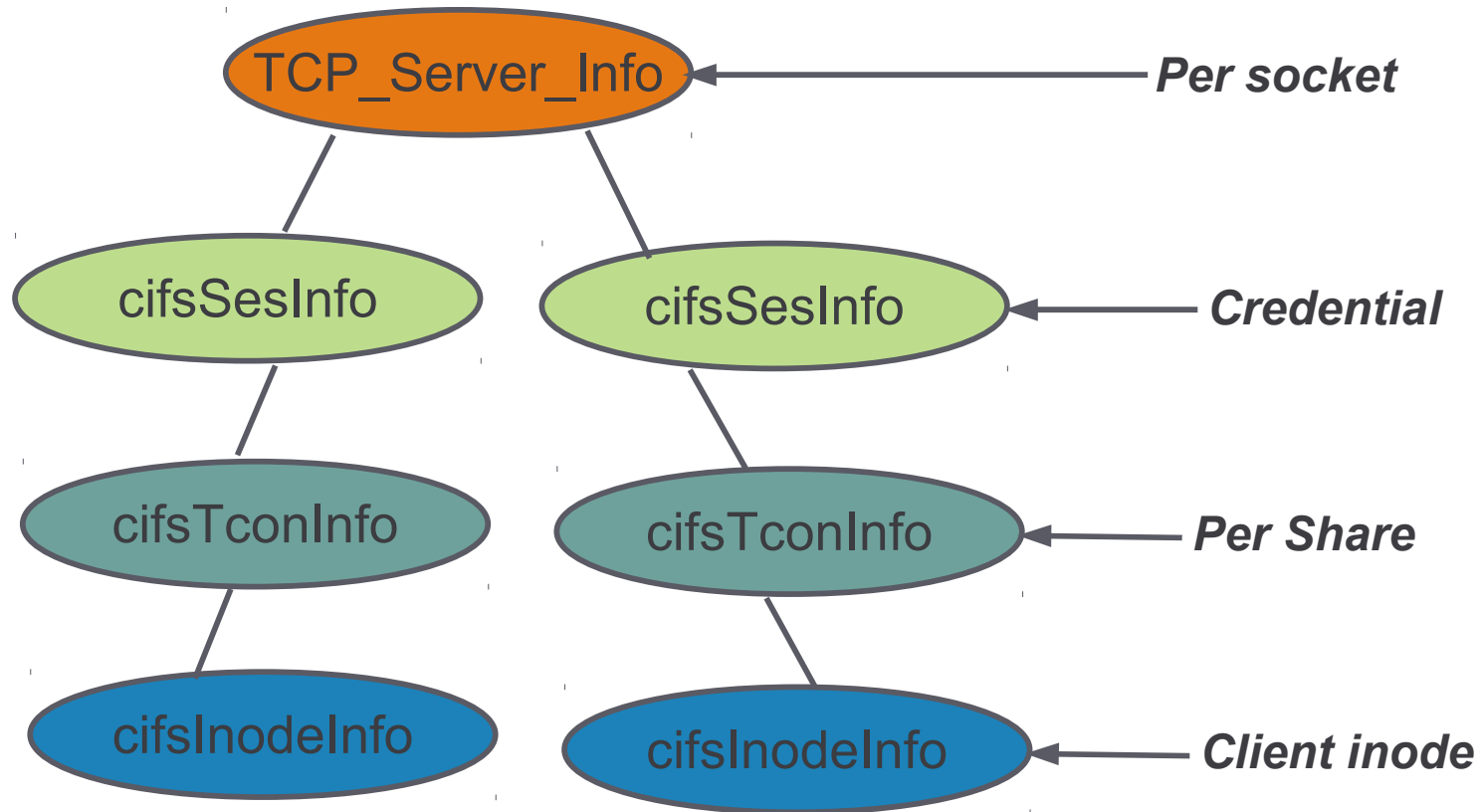
- Define netfs for FS-Cache and register
- Decide on the Index hierarchy
 - Not too deep
 - Index objects can only be children of index objects
- Define objects and methods associated
 - struct fscache_cookie_def
 - Name, type
 - get_key, select_cache, get_aux, check_aux, now_uncached etc.
- All the indices in the index hierarchy and the data file need to be registered

Making filesystem FS-Cache aware (2)

- Functions to store and retrieve pages in the cache
- Cache validation
- Function to release any in-memory representation for the network filesystem page.
- Way to invalidate a data file or index subtree and relinquish cookies.
- Root directory validation (optional)

CIFS current Implementation

CIFS Object hierarchy



CIFS Index hierarchy

- Three level hierarchy
- Server
 - Keyed with Ipaddress, family, port sequence
- Share
 - Superblock keyed with name of the share
- File
 - Keyed with UniqueId
 - > Inode number for Unix filesystems
 - > IndexNumber for FAT-like filesystems

Server Resource validation

- Superblock object keyed by sharename
- Ensure that the exported SMB resource didn't change without client's knowledge
- For e.g.
 - Server exports /vol1/foo
 - Client mounts and access files (persistently cached)
 - Client unmounts
 - Server exports /vol2/foo
 - Client assumes it already has data in cache and provides wrong data
- UniqueId/IndexNumber

Cache validation

- Inodes validation considers
 - LastWriteTime
 - ctime
 - EOF reported by Server
- CreateTime ?

Current status

- RFC posted last month
 - Used Servername/IP as key
 - No way to validate root directory of Server resource (share)
 - Needed an rsize of 4k to work properly
 - Support caching on files opened Read-only
 - Enabled with mount option 'fsc'
- Revised, improved version posted on 05 July
 - Number of Fixes
 - Server key uses Ipaddress, family, port sequence
 - Root dir validation added
 - Better cache validation checks

Open issues

- 'noserverino' mount option
 - Client generates uniqueid (a.k.a inode numbers)
 - will vary after unmount or reboot
 - Used when all server filesystems support does not support unique inode numbers
 - Possible cache aliasing, cache not used
- Cache coherency check
 - When client sets LastWriteTime (tar), some CIFS servers are known not to update mtime until the client closes the filehandle
 - Not a major issue in practice as concurrent access using such application by other machines is uncommon

How to use

- Cachefilesd package (openSUSE repository)
 - Configure `/etc/cachefilesd.conf`
- Grab the patches from
 - <http://www.kernel.org/pub/linux/kernel/people/jays/patches/>
- Kernel Configuration
 - `CONFIG_FSCACHE`, `CONFIG_CACHEFILES`
 - `CONFIG_CIFS_FSCACHE`
- Mount the cifs share
- Try copying a huge file from mount point to local filesystem
 - cache will be initialized the first time
 - it should be read from the local cache the second time

Debugging

- CONFIG_FSCACHE_DEBUG
 - /sys/module/fscache/parameters/debug
 - Bitmask (Cache mgmt, Page handling, operation mgmt etc.)
- CONFIG_CACHEFILES_DEBUG
 - /sys/module/cachefiles/parameters/debug
 - Bitmask (entry, exit, general)

Questions?

References

<http://people.redhat.com/dhowells/fscache/FS-Cache.pdf>

<http://people.redhat.com/steved/fscache/slides/cachefs-new.odp.bz2>

Documentation/filesystems/caching/

Performance

- Reading a 150 MB file
- `time cat /mnt/amuse.zip > /dev/zero (cache init)`
 - real 0m14.019s
 - user 0m0.008s
 - sys 0m1.728s
- `time cat /mnt/egm.zip > /dev/zero (cache hot)`
 - real 0m5.265s
 - user 0m0.008s
 - sys 0m0.320s

Section Break Text Here (32pt)



Novell®

Unpublished Work of Novell, Inc. All Rights Reserved.

This work is an unpublished work and contains confidential, proprietary, and trade secret information of Novell, Inc. Access to this work is restricted to Novell employees who have a need to know to perform tasks within the scope of their assignments. No part of this work may be practiced, performed, copied, distributed, revised, modified, translated, abridged, condensed, expanded, collected, or adapted without the prior written consent of Novell, Inc. Any use or exploitation of this work without authorization could subject the perpetrator to criminal and civil liability.

General Disclaimer

This document is not to be construed as a promise by any participating company to develop, deliver, or market a product. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. Novell, Inc. makes no representations or warranties with respect to the contents of this document, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. The development, release, and timing of features or functionality described for Novell products remains at the sole discretion of Novell. Further, Novell, Inc. reserves the right to revise this document and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes. All Novell marks referenced in this presentation are trademarks or registered trademarks of Novell, Inc. in the United States and other countries. All third-party trademarks are the property of their respective owners.

