Multiuser Mounts with Linux CIFS

Jeff Layton
Red Hat, Inc.

Who am I?

- Decade of experience as UNIX Systems Admin
- Member of Red Hat file system engineering team since 2006
- Joined worldwide Samba team in 2008
- Primarily work on NFS and CIFS, but also in generic VFS layer (and other places)
- Maintain the cifs-utils package

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

File creation confusion

- Mount cifs share with one user's credentials and with unix extensions enabled
- Share is world-writable
- “touch” file in share as another user

$ touch testfile1

touch: cannot touch `testfile1': Permission denied

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

What Happened?

- File was created on server using mount credentials
- CIFS attempts to enforce permissions on client
- That can't fix ownership
- File is created but later ops fail!

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

Permissions Enforcement

- Second test:
  - Mount share with user1's credentials and without unix extensions.
  - As user2, access a file that should be accessible only by user1.
- You can't enforce permissions correctly if you don't know what they should be
- Even if you do, checking on the client is racy – they can change after you check them but before enforcement action.

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

Why is it this way?

- CIFS protocol is **session-based**
- Credentials are handled per-session
- By default, Linux CIFS only has single session per mount
- **Shared Credentials!**

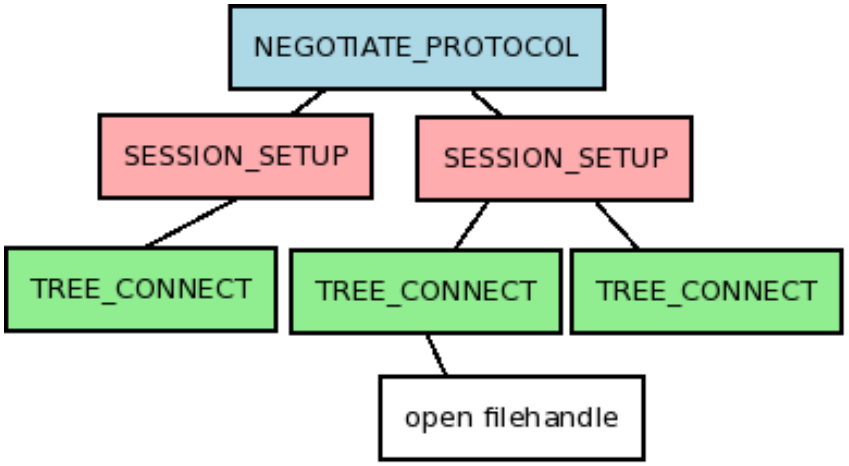
2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

A solution...

- Each user should use their own credentials
- Have multiple SMB sessions per mount
- Establish sessions on an as-needed basis
- Let the server handle permissions
- **Goal**: Easy as NFS!

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

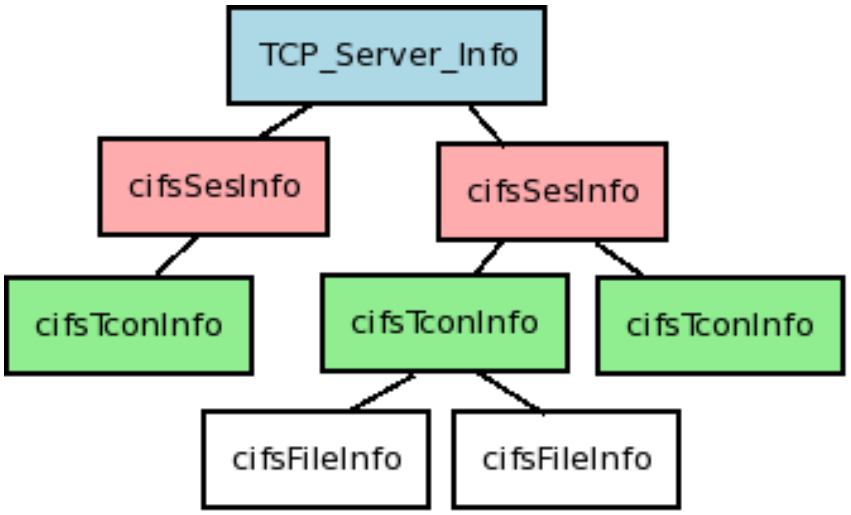
Protocol hierarchy



- The CIFS protocol has a hierarchy of sorts
- NEGOTIATE
- SESSION_SETUP
- TREE_CONNECT
- Open filehandles
- Other path-based ops

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

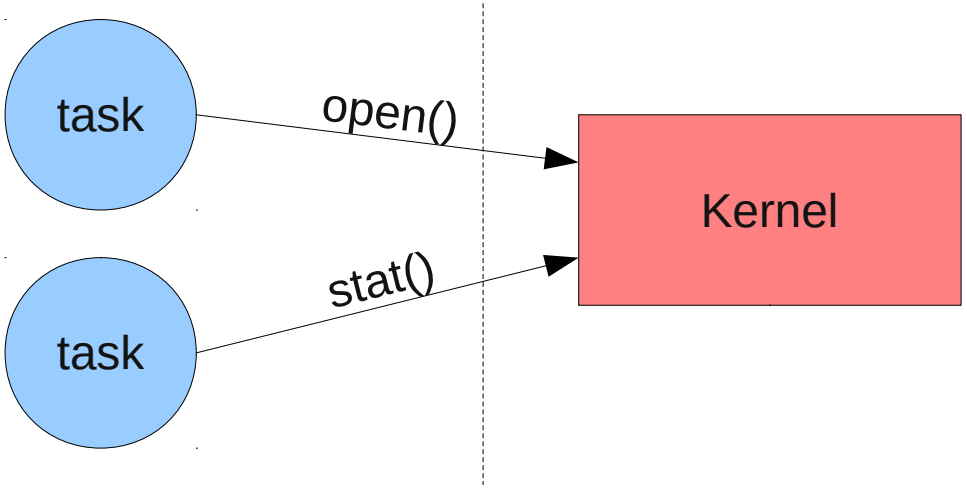
Basic Object Hierarchy



- TCP_Server_Info (socket)
  - NEGOTIATE
- cifs_ses (credential)
  - SESSION_SETUP
- cifs_tcon (share)
  - TREE_CONNECT
- cifsFileInfo (file)
  - open filehandles

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

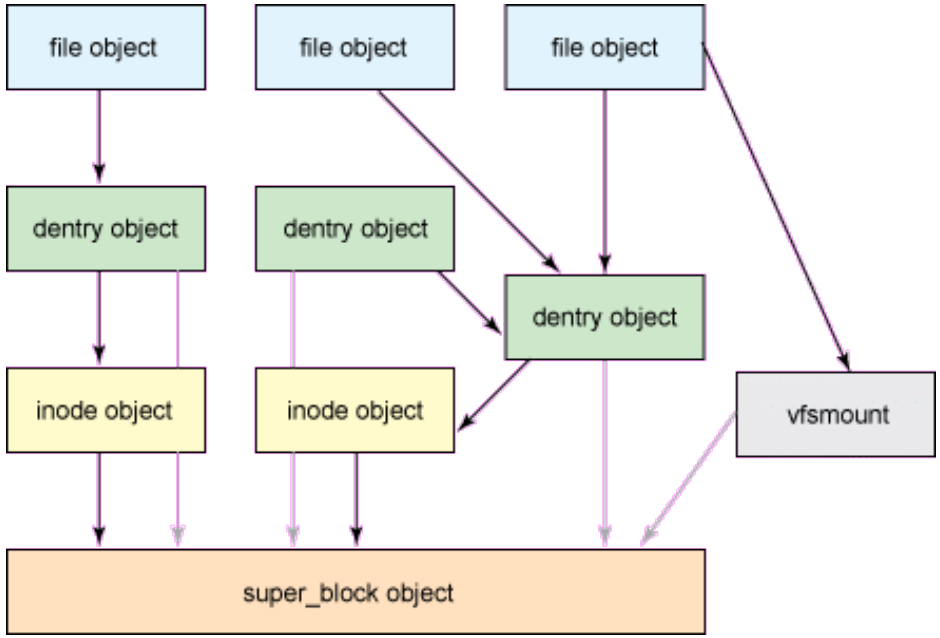
Userspace to Kernel



- To do privileged ops, userland processes have to ask kernel to do it for them
- Typically this is done via system calls
- FS operations go through the VFS layer

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

Basic Linux VFS Anatomy



- **inode** is an actual file or directory
- **dentry** is a path component
- **super_block** is connection to backing store
- **vfsmount** is connection to mount tree
- **file** is an open file descriptor

Image copyright M Tim Jones and IBM

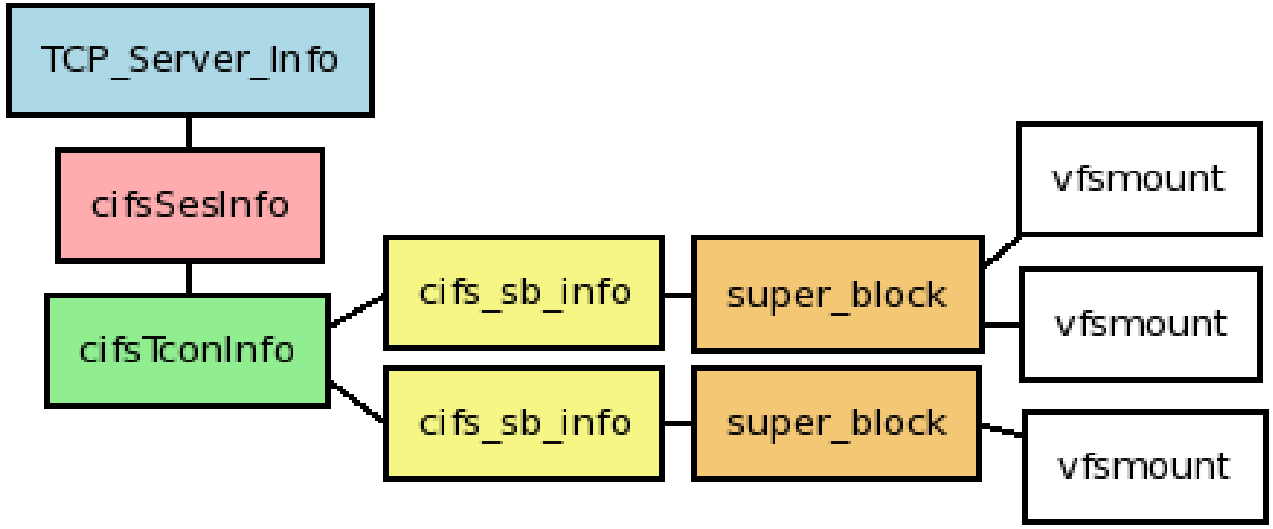
2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

Task Credentials

- "task" in Linux kernel lingo is thread of execution (aka a process or thread)
- Each task has several UIDs associated with it
  - **real:** who owns the process?
  - **effective:** determines permissions when accessing shared resources (shmem, signals, etc.)
  - **saved:** allows task to switch effective uids
  - **filesystem:** permissions for accessing files
- **Goal is to match the fsuid to a session**

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

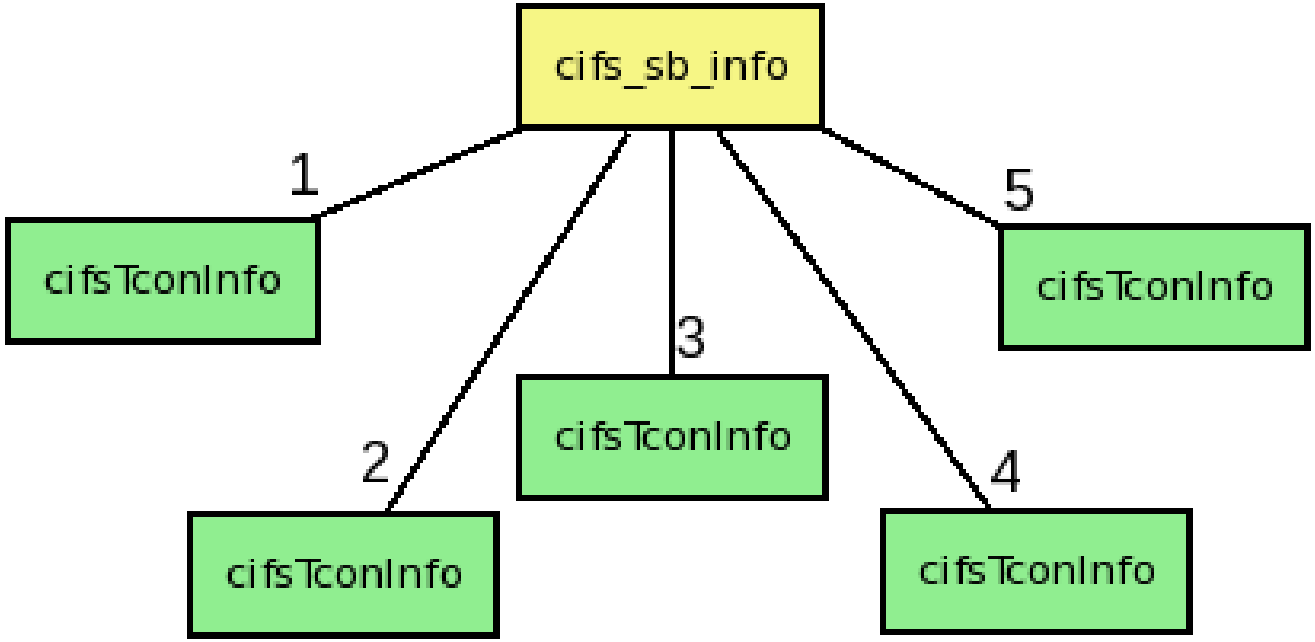
Traditional CIFS-VFS Connection



- a mount in the VFS points to a super_block
- super_block points to a cifs_sb_info
- cifs_sb_info has single pointer to cifs_tcon
- so...each super_block refers to one set of creds

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

More TREE_CONNECTs, Please...

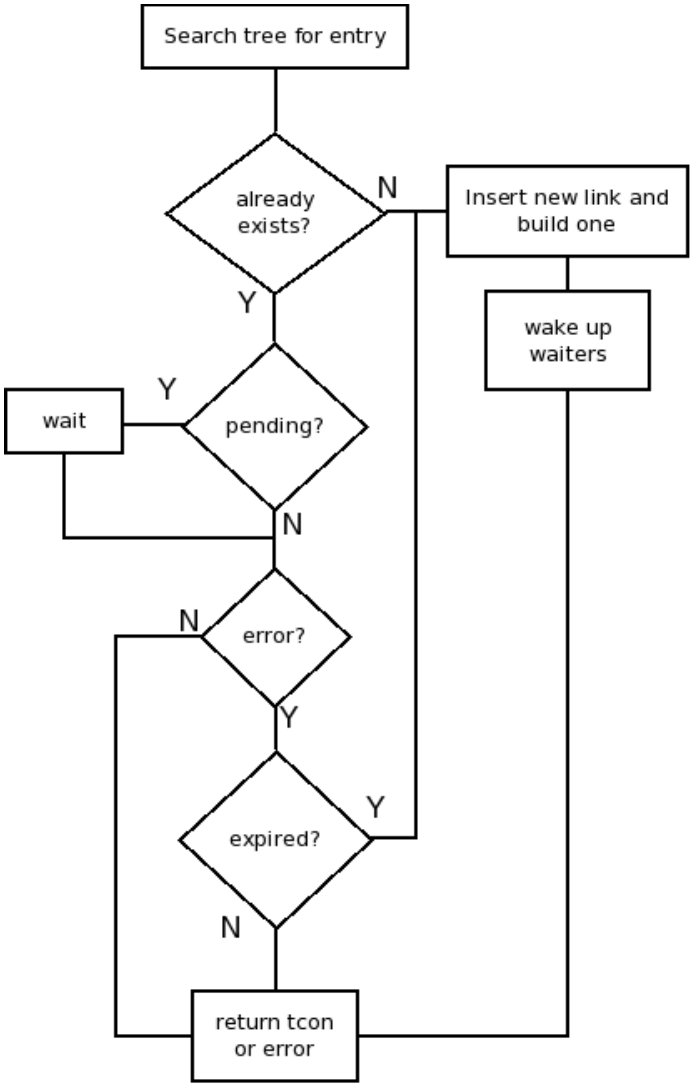


- cifs_sb_info should point to more than one tcon
- convert tcon pointer into a RB tree (tcon_tree)
- use the fsuid as the key

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

Finding/Building New TCons

- Task needs to do a SMB call, so it requests a tcon
- Search for one that matches fsuid
- If not found, then try to build one
- If that fails, error (EACCES)



2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

Building New Tcon's

- tcon established at mount is **master**
- To build new one, use **master** as template
- Build/find new tcon/session for current fsuid
- **Cannot prompt for passwords!**
  - Meshes well with krb5 auth
  - New cifscreds tool in cifs-utils to store user/password in kernel keyring for non-krb5 mounts
- Recurring workqueue job to prune unused tcons after an idle timeout

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

cifscreds

- Originally by Igor Druzhinin in cifs-utils 4.7 and overhauled in 5.3. Kernel support in 3.3
- Allows multiuser mounts to work w/o krb5 auth
- Users stash username/password creds in the kernel session keyring for a host or domain
- Kernel can look for those creds and use them to establish new SMB sessions
- To-do: PAM module

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

Displaying Ownership

- How to handle presentation of file ownership on client? (primarily for stat()-type syscalls)
- With Unix extensions enabled, assume that client and server have uids/gids mapped the same way (similar to NFSv2/3)
- Without Unix extensions, server doesn't send any ownership info. Current code usually sets owner to value of uid= option or root.

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

Displaying Ownership

- For multiuser mounts w/o unix extensions, we always present current fsuid as owner of inodes
- New idmapping upcalls convert SIDs to UID/GID, but you must fetch ACL and upcall which is expensive
- RichACLs may eventually improve this situation, but “ls -l” will always be difficult.

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

What about readdir()?

- Windows has a feature that's used to present different directory contents to different users (Access Based Enumeration -- ABE)
- Is this potentially at risk of exposing that info to users that shouldn't have it?
- Linux CIFS does not cache readdir info. Any readdir() syscall will cause FIND_* to be sent.
- Probably need some work to ensure that other syscalls don't leak info about dentries that are “hidden” by ABE.

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.

Questions?

Questions?

2012 Storage Developer Conference. © Insert Your Company Name. All Rights Reserved.