Update on Android Emulated Storage

Future of Sdcardfs

What does Sdcardfs Do?

- Provides case insensitivity for the lower filesystem, similar to VFAT
- Shared Obb - Was used to share obb contents between users, no longer used for privacy and security reasons
- Special Permissions handling
  - Presents an owner/permission based on file location and mount options
  - Used to prevent a ‘default’, ‘read’, and ‘write’ view
    - Currently done via non-upstreamed changes to mount structure and vfs
  - Access to own data configured via configfs
- Tracking storage
  - We currently have quota uid/gid tracking to track file usage of apps, and of special media types, configured by sdcardfs’s configfs entries.

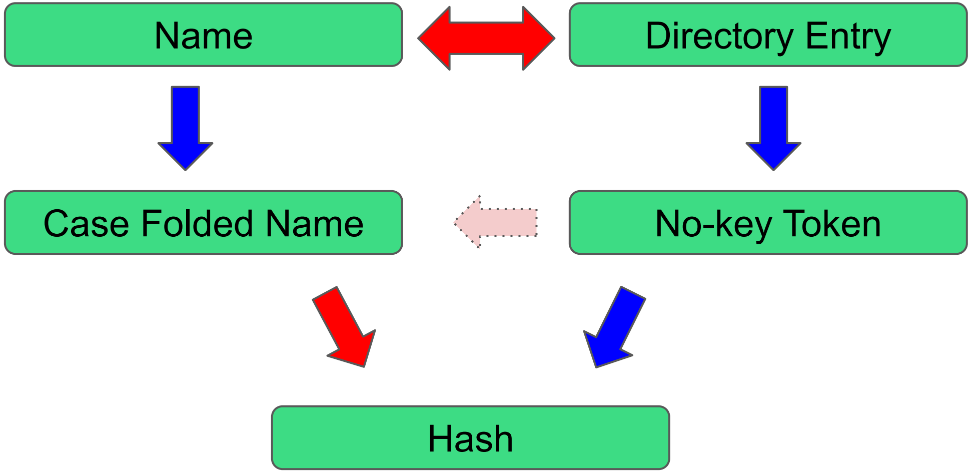
Is Sdcardfs Still needed?

- Shared OBB files now unused
- Storage tracking done by lower FS’s quotas, can be managed elsewhere
- Permissions differences could be done via other means
  - tmpfs core, bind mount relevant parts read-only or read-write
- Case Sensitivity?

Case Insensitivity

- Currently available in Ext4 and F2FS
- Uses custom dentry ops for hash and compare
- Does not support encryption yet

Case folding with Encryption in Ext4 and F2FS



- Case Preserving and Case insensitive
  - No relationship between encrypted name and relevant hash
  - Hash must be stored
- Must decrypt names to check bucket

android

Plans Case folding with Encryption in Ext4 and F2FS

- Roll case folding awareness into default dentry ops

- Use fscrypt2 to derive SipHash-2-4 key to create hash

- Adjust no-key token format to always include hash for casefolded entries
    - Current prototype is base64 encoding of hash and first 149 bytes of encrypted name, and sha256 of the rest if needed

- Ext4 adjustment to store hash (extra space in on disk dentry)

Return of FUSE?

- Align Android Storage APIs with direct access
- Looking towards ExtFUSE for improved performance

Questions ?

android