Access control and ID mapping on the Linux SMB client

Shyam Prasad

Speaker introduction

- Currently working with Azure Files team in Microsoft, India.
- Contributor to the Linux SMB client filesystem project: https://wiki.samba.org/index.php/LinuxCIFS

Agenda

- Authentication
- ID mapping
- Access control (Authorization)
- Future work

SMB filesystem concepts

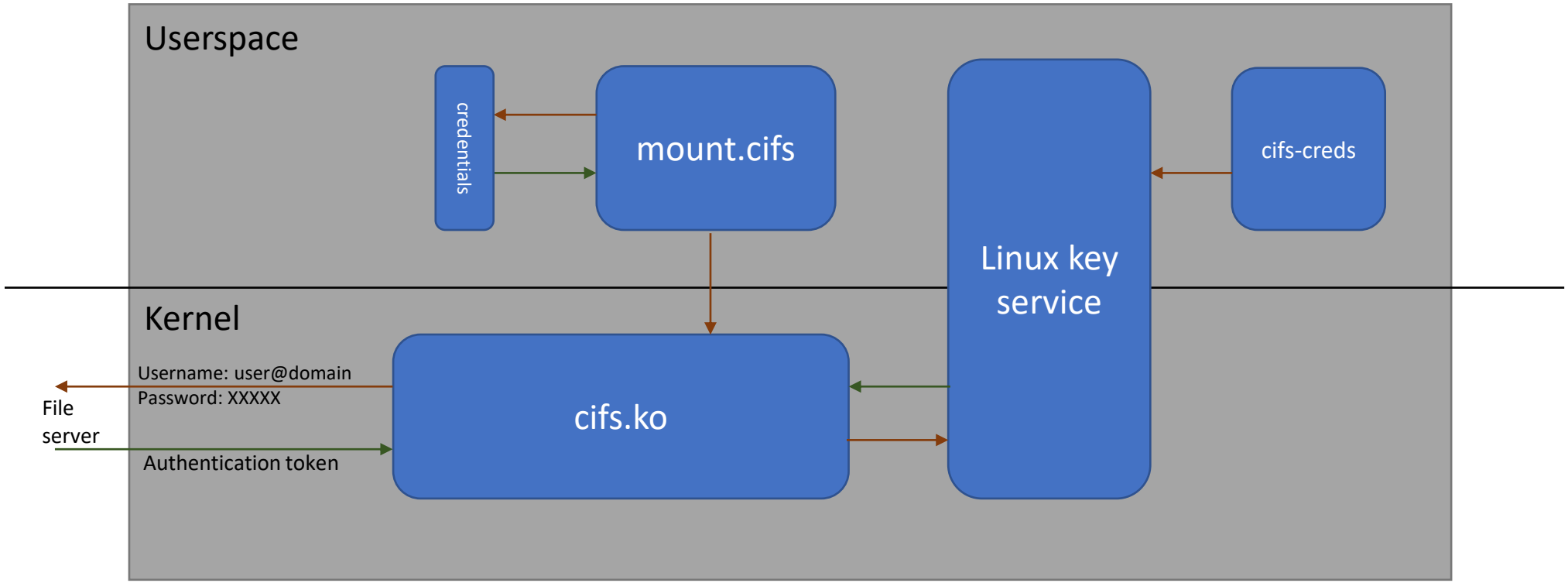
- File server/service: Software stack sharing the files over the network. Ex. Samba, Windows, Azure files.
- File client: Software stack which communicates with the server and exposes the shared files. Ex. Linux SMB client, Windows client, MacOS
- ID-mapping: Mapping between user's identity on client and server.
- Authentication: Can the users prove that they are who they claim to be? (Generally, with some credentials that only they know of)
- Authorization: Do the users have the necessary permissions to access the files in a certain way?

Authentication

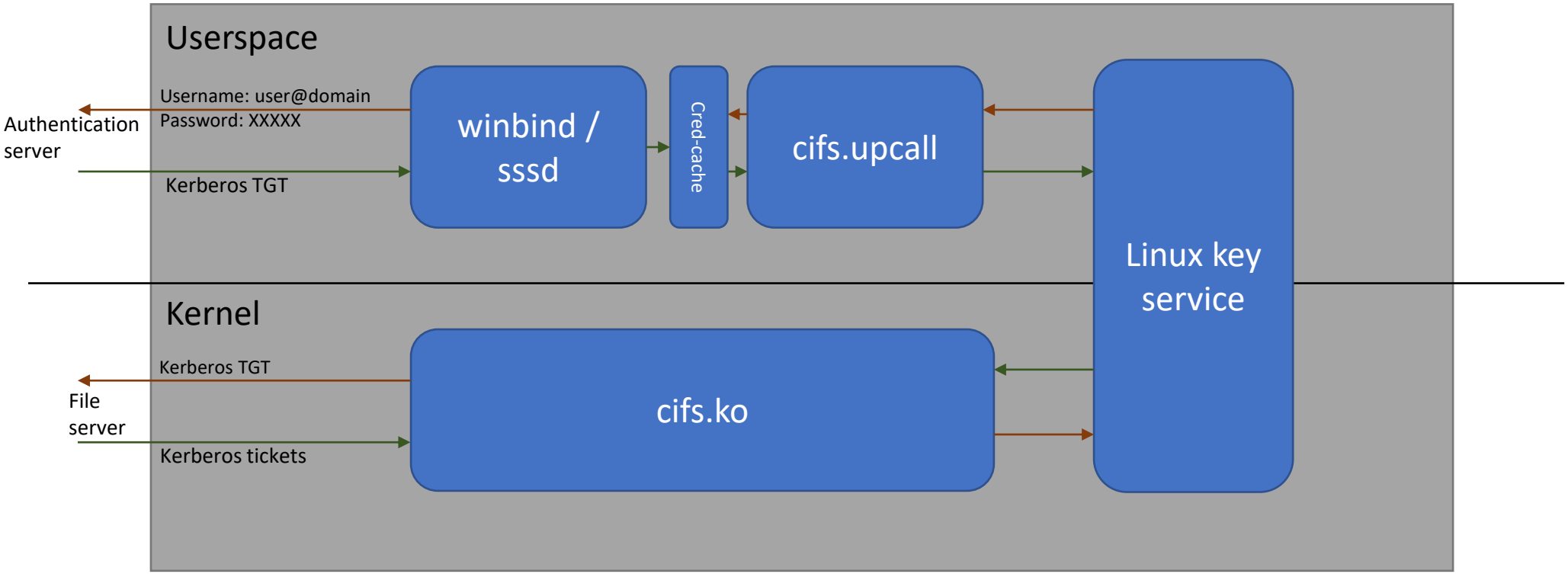
How does Linux SMB client manage authentication?

- Supports NTLMv2 and Kerberos authentication.
- The kernel module cifs.ko caches the credentials in the Linux key service.
- For NTLM, cifscreds allow updating credentials into kernel key service. These credentials are used by cifs.ko for authentication with file server.
- For Kerberos, the user authentication with Kerberos server is expected to be done in advance, before accessing the fileshare. The Kerberos server returns credentials needed to authenticate with the file service.
- When cifs.ko doesn’t find credentials for a user, it can upcall to userspace. (cifs.upcall)
- Can be configured to use winbind/sssd to keep the authentication token up-to-date.

How does Linux SMB client manage authentication? (NTLM)



How does Linux SMB client manage authentication? (Kerberos)



ID mapping

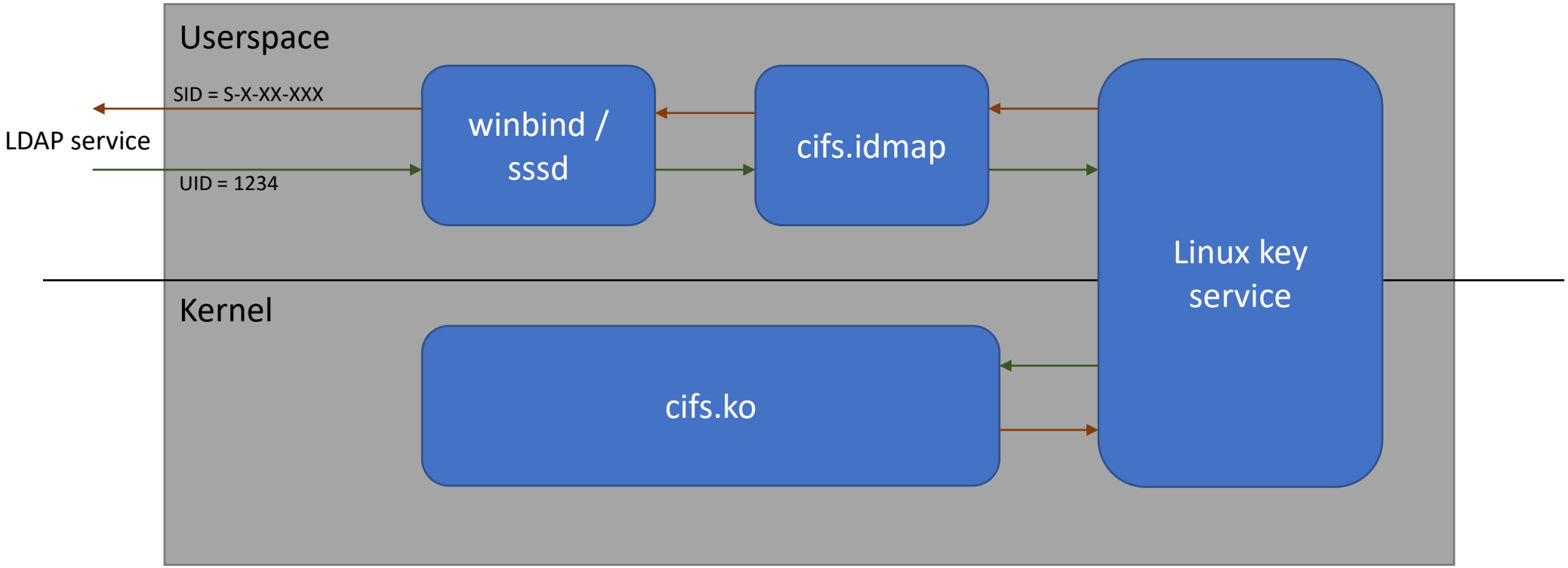
What is ID mapping?

- The task of mapping local users (users on the SMB client) to remote users (users on the SMB server).
- Challenge: Even when the same SMB user maps to different UIDs on different clients, need to map to same server ID.
- Challenge: Even when different SMB users map to same UID on different clients, need to map to different server IDs.
- Relatively simplified with active directory or some other central identity management service.
- If the LDAP server supports RFC2307, the mapping from Unix identities to identities on the server can be maintained centrally.

How does Linux SMB client manage ID mapping?

- Delegates the task to userspace utilities.
- The kernel module cifs.ko caches identities in Linux key service.
- On cache-miss, upcalls to userspace. (cifs.idmap)
- Can easily be integrated with winbind to perform LDAP queries, and map various identities.
- RFC2307 and RFC2307bis are supported by Windows Domain Controllers and Samba Domain Controllers. Linux services like winbind/sssd use them to map unique SIDs to UIDs/GIDs on Linux client (and for reverse mapping).

How does Linux SMB client manage ID mapping?



Access control (Authorization)

What is file system access control?

- When accessing a file or directory on a file system, a user is associated with one or more identities.
- The identities may be associated with the user, or with one or more of the groups that the user belongs to.
- File system objects (files and directories) have rules which define the different permissions granted to users (and users who are members of various groups) accessing that object.

Access control on Linux filesystems

- All processes running in Linux run as a particular user (UID) and a group(GID). Run: ps -eo "pid,user,group,comm"
- File system objects (files and directories) have file ownership (UID:GID pair), and mode bits that describe permissions and authorized users.
- Mode bits broadly define read, write, execute (and a few other) permissions for three set of user types: User, Group, Others.
- For example, mode bits -rwxr-xr-- grant full access to users matching owner UID, read and execute access for users belonging to the owner GID, and only read access to others.
- Some Linux filesystems also support Access Control Lists (ACLs) to allow for more granular access control rules. These are called POSIX ACLs.

Access control in SMB protocols

- The SMB protocol was originally designed at IBM for DOS. However, Microsoft made considerable modifications, based on Windows NTFS. Hence SMB defines the use of NT security descriptors.
- Security descriptors contain ownership information for files and directories, and a list (ACL) of ACEs (access control entries) which define permissions that apply to different users and groups on this file/dir.
- Ownership info are stored as Security IDs (SIDs).

Access control in SMB protocols

- NT ACLs (Rich ACLs) offer much more granular access control than POSIX ACLs. Contains a list of Access control entries (ACEs) with allow/deny permissions that support inheritance while "POSIX ACLs" (implemented by some Linux filesystems natively) allow have "allow" permissions and not deny permissions
- SID of a user is associated with every SMB session.
- Multiple users can share the same TCP connection, but their requests can be distinguished by the SMB "SessionId" in the packet request. A SessionId is obtained when a session is established. i.e. when the user is authenticated.

SMB session handling in Linux

- Linux SMB client maps the user sessions (UID:GID) to an SMB session. Multiple processes can map to the same SMB session.
- Same filesystem mount point can have multiple SMB sessions (multiuser mount option).
- If no suitable SMB session is found, a new one can be created after authentication.
- Each SMB session is associated with a remote user, and the SMB client looks up the mapping using the ID mapping configured.

Linux SMB access control

- Server enforced - default
  - The client does no access checks; the server enforces it for file accesses.
- Client enforced - idsfromsid/modesfromsid
  - Used in environments that access the file shares exclusively from Linux clients.
  - The client encodes the file ownership info (UID/GID) and mode bits inside file/dir ACLs (using reserved SIDs).
  - Client then enforces access control by comparing the UID/GID for the session and mode bits on the file.
- Translated - cifsacl
  - Used in environments that access the file shares both from Windows and Linux clients.
  - The client translates between the user/group/others mode bits on the file and the corresponding ACEs on the file ACL.
  - For example, Active Directory users accessing the file shares will have similar permissions on both Linux and Windows.

Server enforced access control

- Most compatible with SMB protocols.
- SMB file servers can use NT security descriptors as is for access control (e.g. Windows servers), or need to map them to some reversible format (e.g. samba servers).
- May not be ideal for traditional Linux applications, which traditionally rely on mode bits.

Client enforced - idsfromsid/modesfromsid

- Introduced in newer releases of Linux kernel (5.8 onwards).
- The client embeds UID:GID and mode bits into SIDs inside security descriptors.
- Server skips access checks for the files and directories, since SIDs in the reserved range are used.
- **Caution**: Care to be taken that all the SMB client machines accessing the same file shares have matching UID:GID for users and groups. Using winbind helps.

Translated - cifsacl

- Stabilized in Linux kernel 5.12.
- The client translates Linux mode bits into NT security descriptors as closely as possible.
- Useful when interoperability with Windows clients and servers is desired.
- Particularly useful when mounting multiuser (with different authenticated users from the same client to the same server).

Userspace utilities (cifs-utils)

- mount.cifs, an userspace helper for mounting.
- getcifsacl, setcifsacl to query/update the security descriptors (and auditing information) for a file.
- smbinfo is another tool to display detailed informations about files.
- cifscreds to store credentials into kernel keyring.
- smb2-secdesc is a python based GUI tool for managing sec desc.
- cifs.upcall, cifs.idmap: userspace helpers for cifs.ko

Future work

- Add support for newer authentication protocols: pku2u, OAuth.
- Improve regression testing of security features.
- Add a fallback mechanism for mapping IDs via local config file.
- Improve the utilities by adding more features for security management.
- Add support for claims-based ACLs (DAC).
- Add support for mandatory access control (SE-Linux).
- Performance profiling and analysis maybe needed for modefromsid, cifsacl.
- Add support for QUIC protocol, and its security features.

Thank you

Shyam Prasad

(sprasad@microsoft.com)