File System

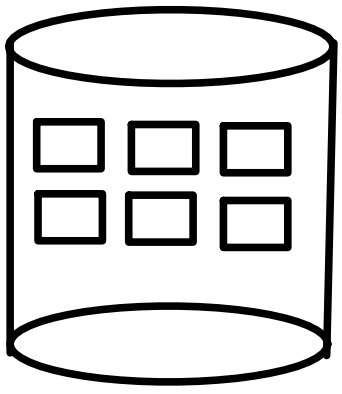
sabato 17 giugno 2023 11:36

Windows File System - NTFS

La struttura di una partizione del dispositivo di memoria di massa dove è installato un file system NTFS è composta in questo modo:

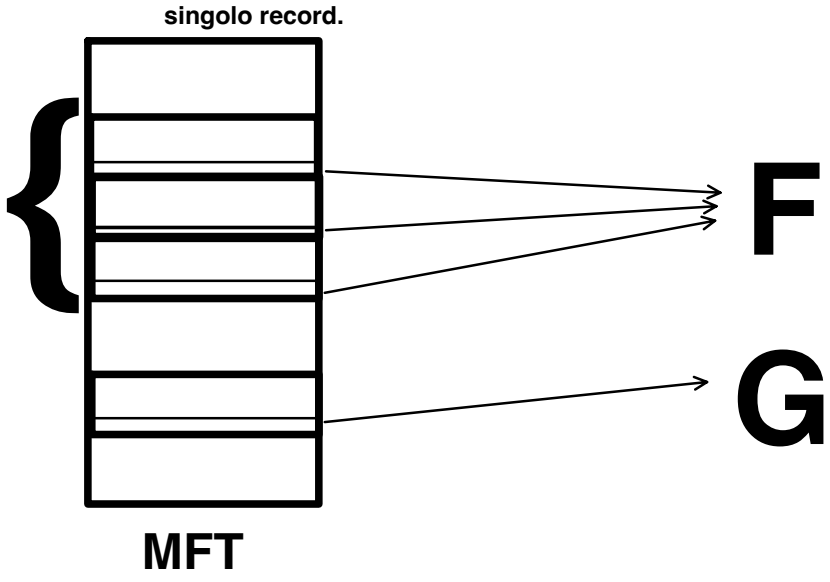
MFT	file di sistema	area dei file

Abbiamo una zona in cui noi manteniamo i dati dei file. Quindi ci sono una serie di blocchi all'interno di questa zona che servono per mantenere i record che sono contenuti all'interno dei file.
La MFT (Master File Table) che è l'equivalente di quello che avevamo all'interno di un file system unix del vettore degli I-NODES. Possiamo identificare i dati ad uno specifico file che sono mantenuti all'interno di un elemento all'interno di questa Master File Table.
Hard-drive divisi in volumi/partizioni e organizzati in CLUSTER da 512 a 64 Kbyte, quindi un blocco può avere questo range di dimensioni.
E per ogni volume si ha una Master File Table corrispondente (MFT).



Nella zona centrale manteniamo le bit-map per andare ad indicare quali sono gli elementi liberi della MFT e della zona dell'area dei file.
Inoltre il file di sistema svolge anche un'attività di logging per andare a registrare alcune informazioni che possono permettere in maniera semplice il recupero dello stato del file system nel caso in cui ci siano dei crash.
Ad ogni file corrisponde almeno un elemento nella Master File Table. Quindi è possibile che abbiamo più di un elemento nella MFT associato ad uno specifico file. È come dire che utilizziamo X I-NODES per mantenere informazioni associate ai dati di un unico file.

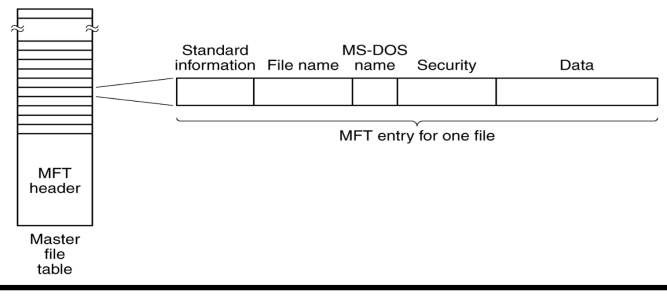
Ogni elemento della master file table contiene:
- Nome del file fino a 255 caratteri unicode: differenza fondamentale che avevamo con il file system UNIX. All'interno di un I-NODE il nome del file non c'è.
- Nome DOS: 8 + 3 caratteri, quindi un nome compatibile con le attività che possono essere svolte dai programmi DOS.
-Informazioni sulla sicurezza: Identiche all'I-node.
-Dati del file o puntatori per il loro accesso: Non è sempre vero che all'interno di uno specifico elemento della MFT puntiamo ai blocchi in memoria di uno specifico file F, c'è la possibilità di mantenere all'interno di questo elemento della Master File Table (MFT) direttamente i dati. Il che implica dire che se noi siamo in grado di associare più di un elemento della MFT ad un unico file siamo in grado di mantenere più dati di un file all'interno della MFT.
Questa è una cosa fondamentale: Fornisce ad NTF la possibilità di gestire file molto grandi di dimensioni, questi file sono direttamente contenuti all'interno di questa master file table.
- Informazioni standard: attributi di accesso/timestamp (aggiornamenti che sono avvenuti su questo file).
- Lista di attributi: dove andare a localizzare all'interno della MFT attributi che non entrano in un singolo record.



Per file di piccole dimensioni i dati sono contenuti direttamente nella sezione dati dell'elemento della MFT, e vengono perciò detti "file immediati".
Questo è un vantaggio non indifferente di Windows: infatti, quando ho un file immediato e carico in cache i suoi metadati, ne carico anche tutto il contenuto, limitando l'accesso al disco rigido.

Per file di grandi dimensioni la sezione dati dell'elemento della MFT contiene gli indirizzi dei blocchi di dispositivo dove questo file è ospitato.

- per piccoli file i dati sono direttamente nella parte dati nell'elemento della MFT (file immediati)
- per file grandi la parte dati dell'elemento della MFT contiene gli indirizzi di cluster o di gruppi di cluster consecutivi
- se un elemento della MFT non basta si aggregano altri



La gestione del buffer cache segue uno schema Least Recently Used (LRU), e lo swap in/out dalla memoria è attuato da un thread di sistema (SU FILE, SI COLLEGA A SWAPPING GIÀ SCRITTO).

Scrittura pigra: gli aggiornamenti vanno su hard drive su base periodica.

Abbiamo un'estensione dell'utilizzo dell'ACL rispetto ai sistemi UNIX: Quando creiamo un oggetto/thread/processo (IN GENERALE UN OGGETTO) su un sistema operativo Windows noi andiamo a definire un ACL, quindi il concetto di ACL è pervasivo cosa che non è all'interno di UNIX dove abbiamo un ACL solo per i file e quindi solo per i programmi, ma non l'abbiamo per i processi. In Windows SI.
Abbiamo un'estensione dell'utilizzo dell'ACL in windows che è sicuramente evidente rispetto a quello che è UNIX.
L'ACL specifica una descrizione per la sicurezza degli oggetti, quindi anche di file, ma questa sicurezza in Windows si divide in due parti differenti:
- DACL: Una riguarda i permessi di accesso/utilizzo ad un'entità che viene ad essere creata all'interno del sistema.
- SACL: Una riguarda le azioni che devono essere eseguite dal sistema operativo quando avvengono alcuni accessi verso questa entità.

Tutto questo è utilizzabile usando il concetto di liste: all'interno di questa lista abbiamo le ACE (Access control Entry) e queste possono essere DACL o SACL.
Un'ACL è formata da una serie di ACE, e queste ACE possono in taluni casi essere delle DACL e in taluni casi essere della SACL.

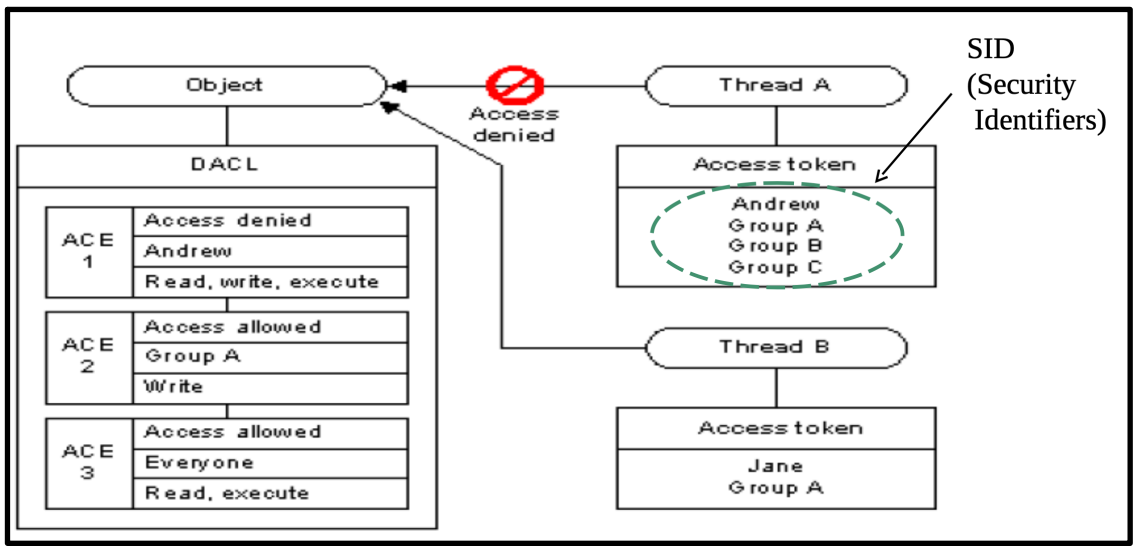
Quando abbiamo un thread attivo questo thread è tale per cui a livello della protezione a questo thread abbiamo associato un accesso token (ossia il token degli accessi) e questo all'interno registra l'identificatore dell'utente per conto di cui stiamo operando e i gruppi di appartenenza dell'utente all'interno del sistema. Un thread che è attivo ha un accesso token che ci dice che ci dice che sta eseguendo

per conto dell'utenza andrew che appartiene all'interno del gruppo A, del gruppo B e del gruppo C.
Quindi l'access token di questo thread non fa altro che registrare questi identificatori.
Ora questo thread potrebbe voler usare una system call per accedere a qualche oggetto (in particolare un file): per capire se questi identificatori che il thread utilizza possano essere compatibili con questo oggetto dobbiamo andare all'interno dell'area DACL che specifica i permessi di accesso a quell'oggetto.
Per specificare i permessi di accesso c'è la zona DACL.
Dentro la zona DACL c'è un'ACE, poi un'altra, poi un'altra… e così via.

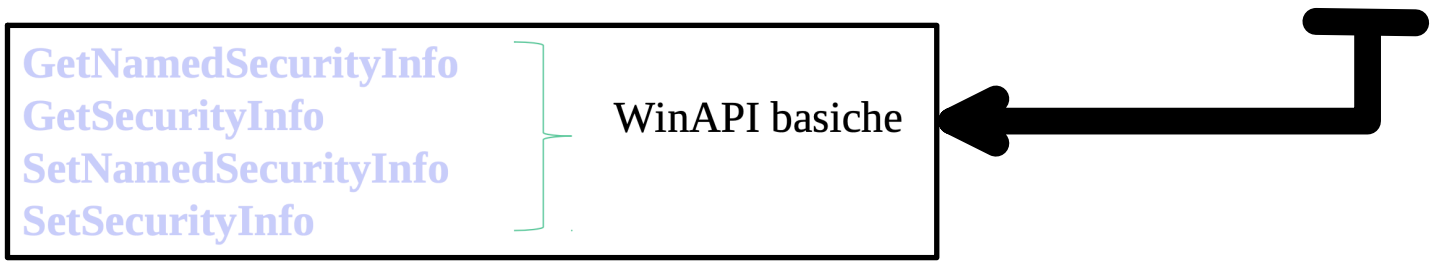
All'interno di queste ACE andiamo a specificare per l'identificatore dell'utente andrew se l'accesso è permesso o negato secondo certe regole.
L'ACE1 dice che per quell'identificatore di utente abbiamo l'accesso negato secondo la regola READ, WRITE ed EXECUTE. Etc..!
Ma in un ACL all'interno di un sistema windows conta l'ordine con cui queste ACE sono presenti: quando noi andiamo a richiedere un'operazione sull'oggetto, l'ACL va guardata in ordine. Noi abiliteremo o bloccheremo questo accesso in funzione della prima entry che troviamo in questo ordine. Nella prima entry c'è scritto che il codice numerico associato all'utente andrew ha in divieto l'operazione di R,W,E. Quando il thread cercherà di accedere a quell'oggetto, l'accesso fallirà. Non ci interessa ora andare a vedere gli altri elementi dell'ACL.

Quindi dobbiamo scandire gli elementi della DACL, ossia gli ACE, e trovare la entry che mi caratterizza quell'access token.
Quindi ogni processo ha associato dei Security Identifier (Utente, Gruppo) che costituiscono l'access token.

Si tratta di trovare il match dell'ACE con l'access token.



Le system call sono le seguenti:



Quelli che non hanno il named va specificata la manigla (handle), quelli con il nome l'oggetto lo possiamo raggiungere con un nome appunto.
Anche in windows possiamo creare degli hard link per i file. Esiste la system call "CreateHardLink" e possiamo specificare gli attributi di sicurezza per quanto riguarda gli hard-link che stiamo creando. E possiamo creare anche dei symbolic link ed esiste la systaem call "CreateSymbolicLink".