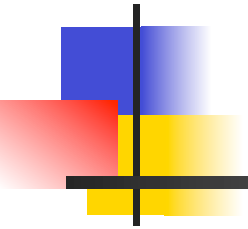
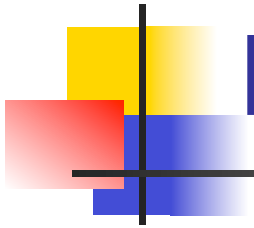


UID/GID remapping

Alessandro Brunengo INFN-Genova

UID/GID remapping

- All'interno di un cluster GPFS assume che ci sia un singolo user identity name space
- Il meccanismo preferito in caso di remote mount e che i cluster condividano lo stesso user identity name space
- GPFS mette a disposizione un meccanismo dinamico per realizzare il remapping degli UID/GID se necessario
  - il meccanismo e' quello di mappare un UID o GID di un cluster in GUN (Global Unique Name) condiviso dai cluster, e di mappare un GUN in un UID o GID nell'altro cluster, e viceversa
  - si implementa tramite la realizzazione di ID Remapping Helper Functions (IRHF) che hanno una interfaccia definita da GPFS



Premessa

- home cluster: il cluster che possiede il file system
- remote cluster: il cluster che accede remotamente al file system
- i valori di UID/GID scritti nei metadati del file system sono sempre quelli dello user space dell'home cluster
- ogni user del remote cluster che ha accesso al file system dispone di un GUN che viene mappato nell'home cluster in un account locale
  - user che non hanno un account possono comunque accedere al file system, rimappati ad esempio come "nobody"
- il comportamento deve essere analogo a quello ottenuto se lo user accedesse al file system dal remote cluster via ssh (ad esempio)

Interfaccia (I)

- L'implementazione si basa sulla creazione di due eseguibili (IRHF):

  mmuid2name <dom> <intent> <nUIDs> <nGIDs>
  mmname2uid <dom> <intent> <nUIDs> <nGIDs>

- <dom>: User Domain del cluster
- <intent>: due valori:
  - "credentials": remapping finalizzato a permission checking e ownership di nuovi file
  - "stat": remapping realizzato in conseguenza di chiamate a stat()
- <nUIDS> e <nGIDs>: numero di UID e GID passati via stdin alla funzione

Interfaccia (II)

- mmuid2name riceve da stdin un elenco di UID e GID, e deve scrivere su stdout i corrispondenti valori di GUN
- mmname2uid riceve da stdin un elenco di GUN (nUIDs relativi a UID, nGIDs relativi a GID) e deve scrivere su stdout i corrispondenti valori di UID o GID
- I due eseguibili devono ritornare 0 in caso di successo, altro valore in caso di failure
  - in caso di failure l'operazione di I/O fallisce con EINVAL
- I due eseguibili devono essere collocati su tutti i nodi del cluster in /var/mmfs/etc ed avere permesso read e execute per tutti gli utenti

intent: credential (I)

- Remote cluster thread:
  - chiama mmuid2name e converte UID/GIDs del processo che tenta l'accesso in GUNs
  - passa i GUNs al thread GPFS dell'home cluster
- Home cluster thread:
  - riceve GUNs dal remote cluster thread
  - chiama mmname2uid e converte i GUNs in UID/GIDs locali
  - esegue l'operazione di I/O utilizzano UID/GIDs rimappati

intent: credential (II)

- mmuid2name: se l'utente ha accesso deve avere un GUN associato all'UID, altrimenti l'UID deve essere mappato in un GUN che corrisponda a qualcosa del tipo "nobody"
- l'approccio suggerito e' di ignorare i GIDs (non e' detto che ci sia mapping per tutti i GIDs): in questo caso sara' mmname2uid che tornera' i GIDs associati all'UID nello user space del cluster
  - questo e' consistente con un comportamento di accesso remoto via credenziali (ssh)
  - il criterio e' comunque lasciato all'implementazione: GPFS lato home cluster si limita a passare a mmname2uid l'output (l'elenco dei GUNs) ottenuto da mmuid2name sul remote cluster

intent: stat (I)

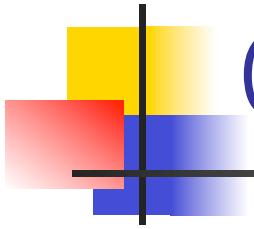
- Remote cluster: esegue stat() su home cluster
- Home cluster:
  - esegue stat() ed ottiene local UID/GID
  - chiama mmuid2name che converte UID/GID in GUNs
  - passa output di stat() e GUNs a remote cluster
- Remote cluster:
  - chiama mmname2uid e converte GUNs in UID/GID
  - ritorna stat() con credenziali rimappate

intent: stat (II)

- senza remapping, GPFS mostra UID/GID dello user space dell'home cluster, che il remote cluster converte in username/groupname in base allo user space cui appartiene il nodo remoto che esegue il comando
  - come fanno altri file system (NFS, AFS, …)
- Il remapping inverso puo' essere fatto, ma:
  - normalmente non tutti gli UID/GID hanno un GUN associato.
  - in caso di mapping parziale, non esiste un criterio realmente soddisfacente (rimappare solo se c'e mapping completo, rimappare quello che si puo' e non il resto, rimappare quello che non ha associazione con "nobody"…)
  - il remapping inverso ha un impatto importante sulle prestazioni

Remapping in altri contesti

- ACL: non si utilizza il remapping
  - le funzioni mmgetacl/mmputacl utilizzano nomi. Fare remapping in assenza di mappatura completa potrebbe essere errato
    - modificare un ACL per uno username il cui UID non ha GUN in “nobody” non e’ corretto
  - l’esecuzione di operazioni sugli ACLda nodo remoto produce un warning
- chown/chgrp: GPFS non permette l’operazione da remote cluster:
  - chown: solo root lo potrebbe fare. Di norma root e’ squashed.
  - chgrp: in situazioni di assenza di mapping dei gruppi, non si puo’ supportare.



Caching

- Una richiesta di remapping comporta due chiamate a funzioni IRHF, una per cluster, ed una connessione tra i demoni di GPFS
- L'impatto sulle prestazioni puo' essere importante, specie per stat
- GPFS ottimizza implementando caching:
  - ID to ID lato remote cluster
  - ID to GUN lato home cluster
- Expiration time per la cache controllato dal parametro di configurazione uidExpiration

Implementazione (I)

- Definire un set di Global Unique Names (GUN)
- Implementare le funzioni **mmuid2name** e **mmname2uid**, e metterle in **/var/mmfs/etc** (eseguibile da tutti gli utenti) su tutti i nodi del cluster
- Eseguire GPFS shutdown su tutti i nodi del cluster
- Abilitare l'UID remapping sul cluster:

  # mmchconfig enableUIDRemap=yes

  - questo va fatto su entrambi i cluster

Implementazione (II)

- Abilitare opzionalmente il remapping per stat (attenzione: ha effetti negativi sulle prestazioni)

  # mmchconfig enableStatUIDRemap=yes

  - anche questo va fatto su entrambi i cluster
- Configurare cache expiration timeout e UID domainname

  # mmchconfig uidExpiration=<timeout in sec>
  # mmchconfig uidDomain=<domain name>

  - il domain name deve essere uguale per cluster che condividono lo user space