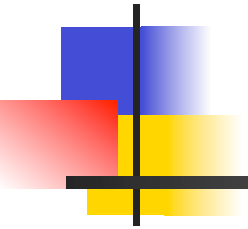
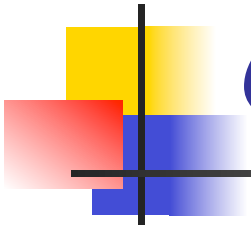


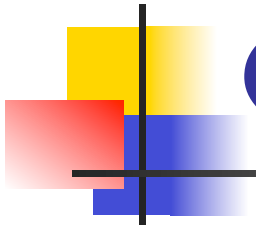
Advanced Features

Alessandro Brunengo INFN-Genova



Contenuto

- Quota management
- Snapshot
- NFS export
- Clustered NFS
- Remote Cluster Export
- Callback
- Access Control List



Quota

Quota management

- GPFS supporta il controllo di quota per l'allocazione di dati e/o i-nodes
- Il controllo di quota puo' essere attivato per file system, relativamente a user, group e fileset
  - a partire dalla release 3.5 puo' essere attivato il controllo di user o group quota all'interno del singolo fileset
- I dati sulla quota vengono mantenuti in memoria e nei file user.quota, group.quota e fileset.quota nella root del file system
- La quota e' imposta attraverso la definizione di soft limit, hard limit, e grace timeout
- Attenzione ai file system con replica di dati/metadati
  - la replica dei dati raddoppia lo spazio disco contreggiato
  - la replica dei metadati raddoppia gli i-nodes conteggiati

Abilitazione del controllo di quota

- Per abilitare il controllo di quota su un file system alla sua creazione, si deve eseguire **mmcrfs** con il parametro -Q yes
  - questo attiva automaticamente la quota ad ogni mount
- Per abilitare e attivare il controllo di quota su un file system precedentemente creato senza, si deve modificare il parametro a file system smontato su tutti i nodi:

```
# mmumount <device-filesystem> -a
# mmchfs <device-filesystem> -Q yes
# mmmount <device-filesystem> -a
```

una volta attivato il controllo di quota si deve compilare la statistica sull'attuale utilizzo di file e data blocks:

```
# mmcheckquota <device-filesystem>
```

Disabilitazione del controllo di quota

- Per disabiltare il quota management, si deve effettuare l'operazione inversa:

```
# mmumount <device-filesystem> -a
# mmchfs <device-filesystem> -Q no
# mmmount <device-filesystem> -a
```

I file della quota rimarranno nel file system, ma non saranno piu' utilizzati

Attivazione/disattivazione del controllo di quota

- E' possibile disattivare temporaneamente il controllo di quota su un file system, anche limitatamente a user, group o fileset quota:

  # mmquotaoff [-u | -g | -j] <filesystem>

- Per riattivare il controllo di quota precedentemente disattivato:

  # mmquotaon [-u | -g | -j] <filesystem>

- Si puo' visualizzare lo stato del quota management sul file system con il comando

  # mmlsfs <filesystem> -Q

```
flag  value             description
------------------      --------------------
-Q    group;fileset     Quotas enforced
```

Default quota

- E' possibile definire valori di quota di default ed abilitare il filesystem ad assegnare tali valori a nuovi user, group o fileset
- I comandi per attivare o disattivare l'assegnazione della quota per default sono **mmdefquotaon** ed **mmdefquotaof**
  - il file system deve avere il controllo di quota abilitato (**-Q yes**)
  - l'assegnazione di limiti di default puo' essere fatta indipendentemente per file system e per user, group o fileset
- per visualizzare lo stato della attivazione di quota di default:

```
# mmdefquotaon -g fs1

# mmlsfs  fs1 -Q

flag value          description
---- -------------- ------------------------
-Q  group           Quotas enforced
     group          Default quotas enabled
```

Assegnazione della quota

- La quota si assegna tramite il comando **mmedquota**
  - il comando permette di definire i parametri della quota tramite una sessione di editing
  - e' possibile assegnare la quota uguagliandola ad altri user, group o fileset (**mmedquota -p**)
    - questo permette l'utilizzo del comando da script
- **mmdefedquota** deve essere utilizzato per definire i valori della **quota di default** per user, group o fileset (**mmedquota -d**)

Visualizzazione della quota

- Il comando che visualizza lo stato di un singolo utente/ gruppo/fileset e' **mmlsquota**
  - il comando mostra blocchi ed inodes utilizzati, utilizzabili (hard e soft limit), ed ***in_doubt***
    - i valori ***in_doubt*** sono la quantita' di quota assegnata come disponibile ai client del cluster ma non ancora conteggiata
    - puo' essere quota usata o non usata
    - in occasione di una failure di un nodo, i valori ***in_doubt*** potrebbero rimanere non aggiornati
  - la somma di **quota utilizzata** ed ***in_doubt*** non puo' superare l'**hard limit**
  - utilizzare **mmlsquota -d** per visualizzare i valori di default quota
- Per avere un report sull'utilizzo della quota in un file system, utilizzare **mmrepquota**

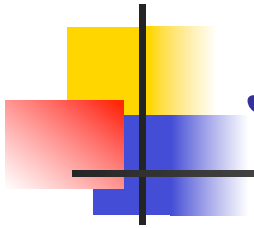
Controllo della quota

- mmcheckquota esegue un controllo della quota utilizzata effettivamente sul file system
  - **mmlsquota** fornisce i valori attualmente memorizzati sul file system manager, che potrebbero essere **piu' o meno out-of-date**
  - in particolare, mmcheckquota esegue un controllo sulla quota ***in_doubt***, rimuovendo quella parte di quota ***in_doubt*** che potrebbe essere rimasta erroneamente conteggiata a causa di **node failure**
- **mmcheckquota** deve essere utilizzato qualora la percentuale della quota ***in_doubt*** dovesse divenire importante
  - questo e' un segno di un possibile conteggio errato di quota in_doubt

Restore dei quota file

- I file contenenti la configurazione delle quote possono corrompersi
  - in questo caso compaiono errori MMFS_QUOTA nei log file
- E' possibile recuperare tali file da backup
  - ad esempio, per recuperare la confiugrazione della user quota:
    - restorare il file user.quota in userquota.bck
    - eseguire il comando mmchechquota -u userquota.bck fs1
    - ricalcolare l'utilizzo attuale: mmcheckquota fs1
- In assenza di backup:

1. **# mmumount fs1 -a**
2. **# mmchfs -Q no**
3. **# mmmount fs1**
4. **# rm /gpfs/fs1/*.quota**
5. **# mmumount fs1**
6. **# mmchfs -Q yes**
7. **# mmount fs1 -a**
8. **eseguire mmedquota per ridefinire le quote**
9. **# mmcheckquota fs1**



Snapshot

Snapshot

- GPFS supporta la creazione di **snapshot di un intero file system o di un fileset** (a partire dalla 3.5)
- La snapshot e' readonly
  - e' la **fotografia** del file system o fileset al momento della creazione della snapshot
  - puo' essere utilizzata da programmi di backup per ottenere **backup consistenti** durante le normali operazioni di I/O degli utenti, o come area di backup per **recupero rapido** di file perduti o per realizzare confronto di file con versioni vecchie

Snapshot

- Lo storage necessario alla snapshot viene preso **dai blocchi del file system** (copy-on-write)
- Il contenuto della snapshot mantiene **tutte le caratteristiche** dei file (**permission**, **ACL**, **attributes**)
- Le snapshot possono essere oggetto di applicazione di policy
  - in questo caso le operazioni definite vengono applicate ai file della snapshot
  - va ricordato che la snapshot e' readonly

Creare una snapshot

- Il comando per creare una snapshot e'

  # mmcrsnapshot <dev> <snap-name> [-j <fileset>]

- Possono essere create fino a 256 snapshot contemporanee su un file system

 # mmcrsnapshot /dev/home_dev testsnap

Writing dirty data to disk
Quiescing all file system operations
Writing dirty data to disk again
Resuming operations.
#

Accesso al contenuto della snapshot

- Il contenuto della snapshot e' accessibile in

  /<fs-mount-point>/.snapshots/<snap-name>
  /<fset-junction>/.snapshot/<snap-name>

- E' possibile (**mmsnapdir**) creare un link **.snapshots** in ogni directory del file system, in modo da accedere alla snapshot di ogni directory tramite il link

  /<dir>/.snapshots/<snap-name>

  - le directory .snapshots **non sono visibili** tramite ls, ma e' possibile listare il suo contenuto o attraversarla con cd

Gestione delle snapshot

- Si possono visualizzare le snapshot definite tramite il comando **mmlssnapshot**
  - e' possibile visualizzarne anche l'occupazione di dati e metadati
- Si rimuove una snapshot tramite il comando

```
# mmdelsnapshot <device> <snap-name> [-j <fileset>]
```

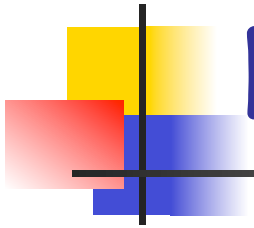
  - viene liberato tutto lo storage occupato dalla snapshot, che non sara' piu' accessibile

Recovery del file system da snapshot

- E' possibile eseguire un restore del file system a partire da una snapshot globale:

  # mmrestorefs <device> <snap-name>

  - il file system deve essere smontato su tutti i nodi
  - le snapshot non sono coinvolte nel processo di restore, quindi rimangono visibili anche dopo il restore di una snapshot vecchia



NFS

NFS export di file system GPFS

- Un file system GPFS puo' essere esportato via NFS da uno o piu' nodi del cluster
- Gli accessi al file system via NFS e via GPFS coesistono senza problemi
  - NFS basa sul timestamp dei metadati la consistenza della sua cache
    - questo richiede **sincronizzazione tra i nodi** che accedono al file system sia via NFS che via GPFS

Considerazioni per l'export via NFS

- Sul server NFS il file system deve essere montato via GPFS **prima** di essere esportato via NFS
  - il file system deve essere montato **automaticamente** allo startup di GPFS
  - e' utile inserire il comando exportfs –ra nel file (da creare) **/var/mmfs/etc/mmfsup** (script eseguito dopo che GPFS e' partito ed i file system sono stati montati)

Considerazioni per l'export via NFS

- per kernel 2.6 o sup, per esportare file system GPFS, in /etc/exports deve essere specificato il parametro fsid = <num>:

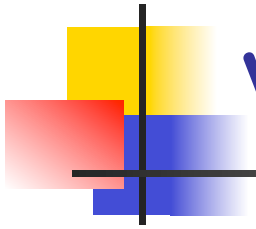
  /gpfs/dir1 cluster1(rw,fsid=745)

- fsid deve essere unico per ciascun file system
- fsid non deve cambiare nemmeno al reboot
- se la stessa directory e' esportata da piu' NFS server, l'fsid deve essere lo stesso su tutti i nodi che lo esportano
- se si esportano due subtree diversi dello stesso file system GPFS, i due fsid devono essere differenti
  - ATTENTI alla documentazione errata in "***A Guide to the IBM Clustered Network File System***"

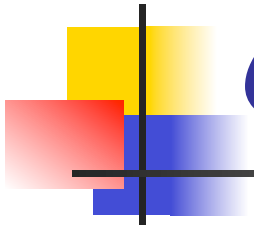
Considerazioni per l'export via NFS

- Il nodo che agisce come NFS server puo' avere benefici nell'incremento della cache GPFS
  - maxFileToCache = 1000
  - maxStatCache = 50000
    - attenzione al token manager per le grosse installazioni
- Se il file system GPFS deve essere esportato via NFS v4, il file system deve essere configurato opportunamente:
  - deve supportare il deny-write open lock (-D nfs4)
    - si DEVE usare -D posix per l'export via NFS v3
  - deve supportare gli ACL di tipo NFS v4 (-k nfs4 o -k all)

Considerazioni per l'export via NFS



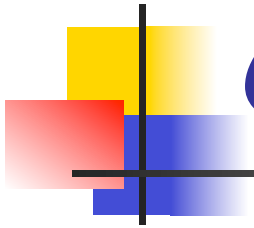
- Si deve valutare l'utilizzo di opzioni che penalizzano le prestazioni
  - **write sincroni**: -o sync,no_wdelay
  - **disabilitare caching** dei metadati sull'NFS client: -o noac
- nfsd puo' allocare file, ed impedire l'umount del file system GPFS
  - si deve fare shutdown di nfs **prima** di eseguire mmumount del file system GPFS sull'NFS server



CNFS

Clustered NFS

- In aggiunta all'export via NFS tradizionale, GPFS supporta la configurazione di un export via NFS realizzato tramite un subset di nodi del cluster che operano in modo da fornire un servizio ad alta affidabilita (Clustered NFS)
  - Il protocollo di trasporto e' NFS (ordinario): non ci sono requisiti sui client che montano il file system
  - I nodi che partecipano all'export sono designati come **CNFS member nodes**
  - il loro insieme e' indicato come **CNFS cluster**
- In questa soluzione i membri del CNFS cluster devono esportare gli stessi volumi ai client NFS (stesso file /etc/exports)
- In caso di failure (o di shutdown) di un membro del cluster CNFS, l'infrastruttura di clustering GPFS sottostante viene utilizzata per implementare un meccanismo di failover:
  - il cluster CNFS si accorgera' della failure del server
  - un altro nodo del cluster CNFS si fara' carico delle attivita' pendenti del nodo in failure senza interruzione di servizio
  - la gestione del failover e' completamente trasparente ai client
- Attualmente CNFS e' supportato solo su piattaforma Linux

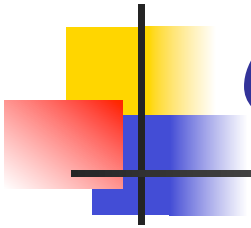


CNFS

- I meccanismi su cui si basa CNFS sono
  - **NFS monitoring**: ogni nodo del cluster CNFS esegue una utility che controlla lo stato di NFS, GPFS e della rete. In caso di malfunzionamenti l'utility innesca la procedura di NFS failover
  - **NFS failover**: il meccanismo di NFS failover e' eseguito tra i nodi del cluster CNFS in aggiunta ai meccanismi del GPFS recovery
    - identifica un nodo del cluster CNFS per il takeover
    - trasferisce il carico NFS del nodo indisponibile sul nodo selezionato
    - il meccanismo di failover si basa sull'***IP address failover***
    - supporta il recovery degli NFS lock
  - **IP address failover**: il cluster CNFS necessita di un set di indirizzi IP (usualmente uno per ogni nodo) in aggiunta a quelli principali
    - l'IP address failover consiste nella migrazione di questi indirizzi tra i nodi del cluster

Network setup per CNFS

- Ad ogni nodo del cluster CNFS deve essere assegnato un altro indirizzo IP
  - l'indirizzo puo' essere reale o virtuale (alias)
    - se e' un alias, non deve essere configurato
  - l'indirizzo deve essere configurato staticamente
  - l'interfaccia di rete relativa non deve partire al boot
- L'insieme di tali indirizzi deve essere registrato nel DNS, con un alias
  - l'alias sara' il nome utilizzato dai client per specificare il server NFS da cui montare
  - l'alias permette di realizzare
    - load balancing via DNS
    - una configurazione (lato client) indipendente dai nodi effettivamente configurati nel cluster CNFS



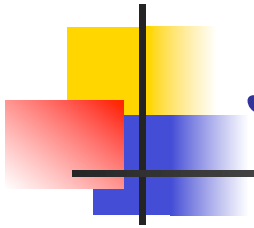
CNFS: failover

- CNFS failover nel dettaglio:
  - l'NFS monitor utility identifica un problema relativo ad NFS
  - l'NFS monitor utility ferma l'NFS serving e ferma (kill) il daemon GPFS.
  - Il cluster GPFS identifica il node failure e, come parte del recovery, tutti i nodi del cluster CNFS entrano in stato di grace period (fermano tutte le richieste di lock dei client)
  - il cluster GPFS completa il recovery rilasciando tutti i lock posseduti dal nodo che e' in failure
  - il cluster CNFS sposta tutti i lock posseduti dal nodo in failure su un nuovo nodo ed invoca l'NFS recovery
  - Il cluster CNFS opera l'IP address takeover (inclusa emissione di gratuitous ARP)
  - i nodi del cluster CNFS notificano ai client di iniziare un lock reclamation, secondo il protocollo NFS
  - alla fine del grace period le operazioni ricominciano normalmente

CNFS: prerequisiti

- linux 2.6 kernel
  - distribuzioni supportate: RHEL 4, 5 e 6, SLES 9 e 10
- OS patches
  - patch del ***lockd*** daemon per la propagazione delle informazioni di lock al file system sottostante
  - patch per permettere a ***statd*** di inviare ai client reclaim messages provenienti da un indirizzo IP specifico del server (necessario in conseguenza dell'IP failover)
    - richiede l'utilizzo di opportune versioni di util-linux o nfs-utils
- Verificare sulla documentazione in base alla distribuzione ed alla release

Setup di CNFS (I)



- Definire una directory per i file shared del cluster CNFS
  - **# mmchconfig cnfsSharedRoot=*directory***
  - <dir> e' il path ad una directory su file system GPFS che:
    - non sia esportato via NFS (preferibilmente un piccolo file system dedicato)
    - sia montato automaticamente allo startup di GPFS (-A yes)
  - GPFS deve essere down sui nodi ? (documentazione ambigua: pagg. 12 e 148 di *Administration and Programming Reference 3.5*)
- Configurare i file system da esportare in /etc/exports dei nodi del cluster CNFS
  - la configurazione **deve essere uguale per tutti**
  - nfsd non deve essere configurato per partire al boot (GPFS si incarica di fare start/stop)

Setup di CNFS (II)

- Aggiungere i nodi al cluster CNFS
  - **# mmchnode --cnfs-interface=<nfs_ip> -N <node>**
  - <nfs_ip> e' l'indirizzo virtuale (puo' essere piu' d'uno)
  - <node> e' il nome del nodo da aggiungere al cluster CNFS
- Definire parametri di configurazione
  - **# mmchconfig cnfsMountdPort=<port>** (la porta a cui risponde rpc.mountd)
  - **# mmchconfig cnfsNFSDprocs=<nproc>** (numero processi NFS, default 32)
  - **# mmchconfig cnfsVIP=<aliasDNSname>** (inutile)

Failover groups

- E' possibile forzare la selezione del nodo subentrante in funzione del nodo andato in failure
- Questo e' implementato attraverso i CNFS group id:
  - **# mmchnode --cnfs-groupid=<nn> -N <node>**
  - <nn>: CNFS group ID (numero di due cifre)
  - <node>: il nodo del cluster a cui assegnare il CNFS group id
- Quando un nodo fallisce, la selezione del nodo a cui assegnare il carico viene determinata:
  - uno dei nodi con lo stesso group id
  - uno dei nodi con group id nella stessa decina
    - ad esempio, se fallisce un nodo con group id = 15, si cerca un nodo
      - con group id = 15
      - con group id tra 10 e 19
  - in mancanza di nodi con group id nella stessa decina il failover fallisce
- Per default tutti i nodi hanno CNFS group id = 0 (tutti possono subentrare a tutti)

CNFS management

- La configurazione del cluster CNFS viene visualizzata tramite il comando:

 # mmlscluster --cnfs

```
GPFS cluster information
========================
  GPFS cluster name:         grid-ge.grid03
  GPFS cluster id:           13965265603593503612

Cluster NFS global parameters
-----------------------------
  Shared root directory:     /mnt/atlas2_mnt/HA
  Virtual IP address:        (undefined)
  rpc.mountd port number:    (undefined)
  nfsd threads:              32
  Reboot on failure enabled: yes
  CNFS monitor enabled:      yes

 Node  Daemon node name    IP address       CNFS state  group  CNFS IP address list
-------------------------------------------------------------------------------------
   15   cnfs1.ge.infn.it   193.206.151.4     enabled       0      193.206.151.5
   16   cnfs2.ge.infn.it   193.206.151.6     enabled       0      193.206.151.7
```

CNFS management

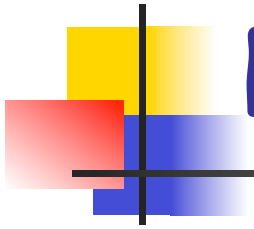
- E' possibile abilitare e disabilitare nodi del cluster CNFS:

  # mmchnode --cnfs-enable -N <node-list>
  # mmchnode --cnfs-disable -N <node-list>

- Per rimuovere un nodo dal cluster CNFS:

  # mmchnode --cnfs-interface=DELETE

- Disabilitare un nodo significa escluderlo dai meccanismi di failover
  - l'export ordinario via NFS non e' coinvolto dal comando: il server NFS e le connessioni restano attivi
  - per disabilitare o rimuovere anche l'export NFS senza dare disservizio, si deve prima fermare GPFS sul nodo tramite **mmshutdown**
    - questo innesca la procedura di failover
- Per la rimozione vale lo stesso discorso
  - il nodo deve anche essere rimosso dal round-robin del DNS



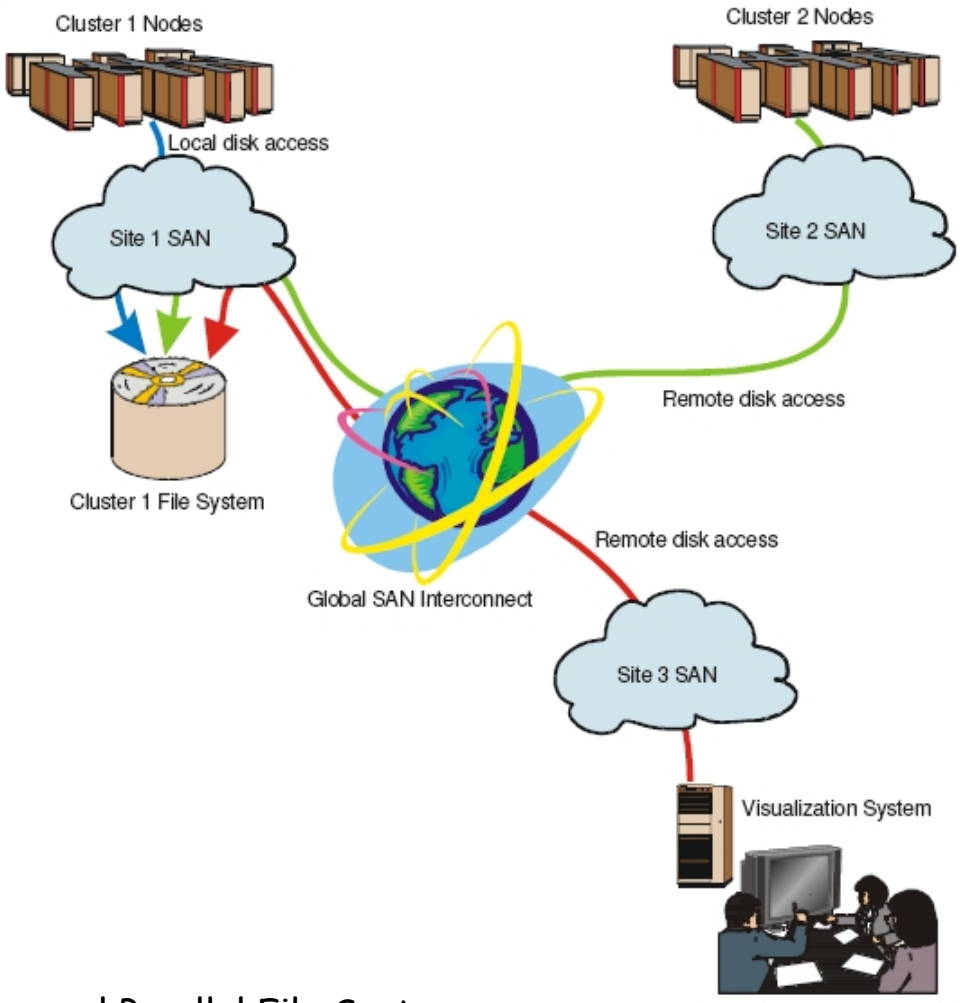
Remote cluster export

Remote cluster export

- Il file system GPFS creato su un cluster e' accessibile implicitamente a tutti i nodi del cluster
- GPFS permette di rendere un file system accessibile anche ai membri di un cluster GPFS diverso
  - l'accesso e' nativo (non mediato da protocollo client/server)
  - se i client del cluster remoto vedono gli NSD del file system , l'accesso e' diretto
  - altrimenti l'accesso e' mediato dagli NSD server del cluster che possiede il file system
- Prerequisito e' che ogni nodo del cluster che vuole montare il file system deve avere connettivita' TCP verso tutti i nodi del cluster proprietario del file system
  - la comunicazione avviene tra i demoni GPFS (non c'e' comunicazione di management via ssh
  - se due cluster A e B montano un file system del cluster C, i nodi di A e B non devono comunicare tra loro

GPFS Multi-Cluster Feature
The Big Picture

- Problem: nodes outside the cluster need access to GPFS files
- Solution: allow nodes outside the cluster to natively (i.e., no NFS) mount the file system
  - "Home" cluster responsible for admin, managing locking, recovery, etc.
  - Separately administered remote nodes have limited status
    - Can request locks and other metadata operations
    - Can do I/O to file system disks over global SAN
    - Are trusted to enforce access control, map user Ids, ...
- Uses:
  - High-speed data ingestion, postprocessing (e.g. visualization)
  - Sharing data among clusters
  - Separate data and compute sites (Grid)
  - Forming multiple clusters into a "supercluster" for grand challenge problems
- Scaling: max supported GPFS cluster size



GPFS Multi-Cluster
Example

Mount a GPFS file system from Cluster_A onto Cluster_B

On Cluster_A

1. Generate public/private key pair
   `mmauth genkey new`
   COMMENTS
   - key pair is placed in /var/mmfs/ssl
   - public key default file name `id_rsa.pub`
2. Enable authorization
   `mmauth update . -l AUTHONLY`
3. Sysadm gives following file to Cluster_B
   `/var/mmfs/ssl/id_rsa.pub`
   COMMET: rename as `cluster_A.pub`
7. Authorize Cluster_B to mount file systems owned by Cluster_A
   `mmauth add cluster_B -k cluster_B.pub`
8. Authorize Cluster_B to mount a particular FS owned by Cluster_A
   `mmauth grant cluster_B -f /dev/fsA`

On Cluster_B

4. Generate public/private key pair
   `mmauth genkey`
   COMMENTS
   - key pair is placed in /var/mmfs/ssl
   - public key default file name `id_rsa.pub`
5. Enable authorization
   `mmauth update . -l AUTHONLY`
6. Sysadm gives following file to Cluster_A
   `/var/mmfs/ssl/id_rsa.pub`
   COMMENT: rename as `cluster_B.pub`

9. Define cluster name, contact nodes and public key for cluster_A
   `mmremotecluster add cluster_A -n nsd_A1,nsd_A2 -k Cluster_A.pub`
10. Identify the FS to be accessed on cluster_A
    `mmremotefs add /dev/fsAonB -f /dev/fsA -C Cluster_A -T /fsAonB`
11. mount FS locally
    `mmmount /dev/fsAonB`

Communication Between Clusters

All nodes in both clusters must have TCP/IP connectivity between each other; this is used only for "daemon to daemon" (*n.b.*, mmfsd) communication via the GPFS protocol. This does not allow remote shell (e.g., ssh or rsh) access between nodes; remote users can **not** access the home cluster by this mechanism. OpenSSL guarantees secure communications.

Contact Nodes

The contact nodes are used only when a remote cluster first tries to access the home cluster; one of them sends configuration information to the remote cluster after which there is no further communication. It is recommended that the primary and backup cluster manager be used as the contact nodes.

mmauth

- **# mmauth update . -l AUTHONLY**
  - il comando configura il cluster per utilizzare il livello di encription specificato (authonly: encription per l'autorizzazione all'accesso)
  - quando si passa da un livello non sicuro a sicuro e viceversa, **il cluster deve essere down**
- **# mmauth delete <remote-cluster>**
  - elimina un cluster (e la sua chiave) tra quelli precedentemente autorizzati
- **# mmauth grant <remote-cluster>**
  - opzione per specificare solo un file system (**-f**)
  - opzione per specificare accesso rw o ro (**-a**)
  - opzione per specificare root-squash (**-r [uid:gid | no]**)
- **# mmauth deny <remote-cluster> [-f <device>]**
  - rimuove una autorizzazione concessa
- **# mmauth show**
  - visualizza i cluster a cui si e' dato accesso a file system locali, ed i file system relativi

Problema con libreria ssl

- Si puo' verificare un problema con libreria SSL
  - GPFS non trova libssl.so.4
  - RHEL5/6 troppo uptodate: hanno libssl.so.1.0.0
- Si deve modificare un parametro di configurazione undocumented:

  mmchconfig openssllLibName=libssl.so.1.0.0

- E' meglio aggiungere il nome della libreria attuale a quelli preconfigurati; per vedere l'attuale valore (a cluster su):

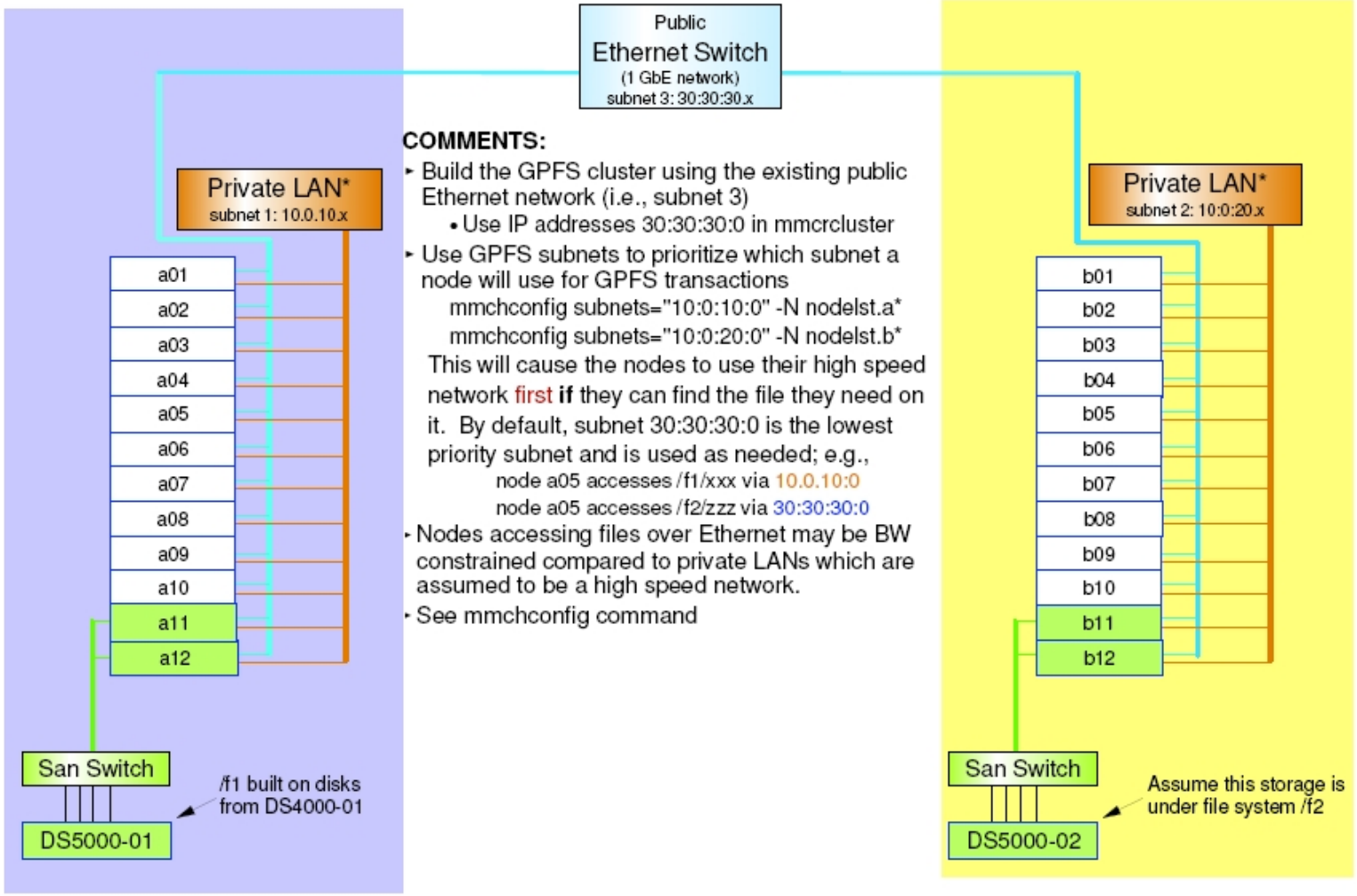
  # mmdiag --config | grep openssllLibName
  openssllLibName libssl.so:libssl.so.0:libssl.so.4

- Per aggiungere il nome in piu':
  - mmchconfig openssllLibName=<old string>:<new file name>

mmremote*

- **# mmremotecluster add|update|delete|show**
  - gestisce i cluster remoti da cui si desidera montare un file system
- **# mmremotefs add <device>**
  - crea un "device" corrispondente al file system remoto sui nodi del cluster locale
  - permette di specificare le seguenti opzioni ide3ntiche a quelle del file system locale:
    - il mount point (**-T <dir>**), omogeneo sul cluster locale
    - quando montare (**-A yes|no|automount**)
    - opzioni del mount (**-o <MountOotions>**)
    - priority del mount (**--mountPriority <prio>**)
- **# mmremotefs update** modifica le opzioni
- **# mmremotefs delete** rimuove un remote file system

GPFS Subnets



COMMENTS:

- Build the GPFS cluster using the existing public Ethernet network (i.e., subnet 3)
  - Use IP addresses 30:30:30:0 in mmcrcluster
- Use GPFS subnets to prioritize which subnet a node will use for GPFS transactions
  - mmchconfig subnets="10:0:10:0" -N nodelst.a*
  - mmchconfig subnets="10:0:20:0" -N nodelst.b*

  This will cause the nodes to use their high speed network first **if** they can find the file they need on it. By default, subnet 30:30:30:0 is the lowest priority subnet and is used as needed; e.g.,
  - node a05 accesses /f1/xxx via 10.0.10:0
  - node a05 accesses /f2/zzz via 30:30:30:0
- Nodes accessing files over Ethernet may be BW constrained compared to private LANs which are assumed to be a high speed network.
- See mmchconfig command

* The private LAN is generally a high speed switch; e.g., IB, Myrinet, Federation
Assume nodelst.a contains the nodes a01-a12 and nodelst.b contains nodes b01-b12.

Subnets in multi-cluster

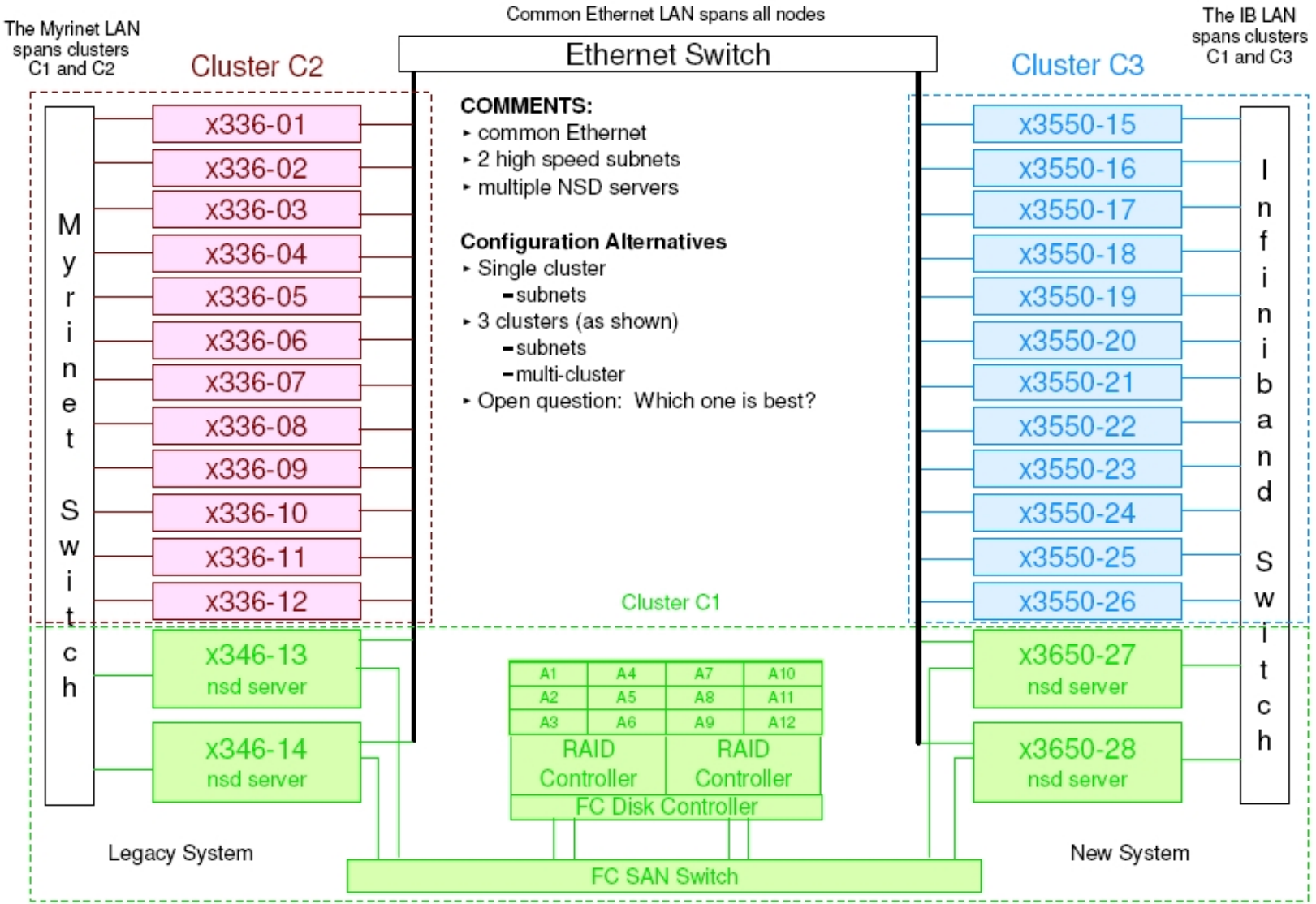
- Il parametro di configurazione subnets configura una sequenza di reti IP preferenziali per accedere ai nodi del cluster
  - quando due nodi si contattano utilizzano gli IP corrispondenti al nome con cui si conoscono (tipicamente IP pubblici)
  - dopo si scambiano tutti gli IP che hanno, per vedere se esiste una subnet configurata in comune: se c'e', la usano
  - la scelta e' definitiva: in caso di failure di una connessione di rete non si cerca di tornare indietro su un'altro indirizzo
- E' possibile configurare tale parametro anche per accedere a nodi di cluster remoti:

  # mmchconfig subnets="10.10.10.0/cl1;cl2"

  indica di utlilzzare quella rete per connettersi con i nodi dei cluster cl1 e cl2
  - se si configurano reti IP private per remote cluster, GPFS assume che siano reti disgiunte
  - per operare tra cluster remoti su una rete privata connessa, la subnet va configurata specificandi i due cluster name assieme, come nell'esempio

Subnet *vs.* Multi-Cluster

Combining Subnets with Multi-Clusters to Support Multiple Fabrics



Modifica delle chiavi SSL

- E' possibile modificare le chiavi SSL di un cluster in modo da non influire sulle connessioni attive tra cluster
  - **# mmauth genkey new**
    questo genera una nuova coppia di chiavi, mantenendo la vecchia coppia ancora usabile
  - il file **/var/mmfs/ssl/id_rsa.pub** che contiene la parte pubblica della nuova chiave va copiata sul cluster remoto
  - sul cluster remoto va eseguito il comando

    # mmremotecluster update <cname> -k <file>
    se la chiave e' modificata su chi esporta il file system, o

    # mmauth update <cname> -k <file>
    se la chiave e' modificata su chi importa il file system
  - infine si rende la nuova chiave come unica chiave valida con il comando
    # mmauth genkey commit

Da tenere presente…

- Un file system e' amministrato unicamente dal cluster in cui e' stato creato
  - le uniche operazioni eseguibili da un cluster che monta in file system remoto sono
    - accedere ai dati in read/write secondo gli accessi
    - eseguire comandi di **visualizzazione**: mmlsfs, mmlsdisk, mmlsmount, mmdf
- Ogni cluster e' gestito in modo indipendente, quindi modifiche di configurazione non sono propagate automaticamente tra due cluster
  - se il cluster proprietario del file system cancella o rinomina il file system, il cluster remoto non potra' piu' montarlo
    - ci deve essere coordinamento e update delle informazioni in mmauth e mmremotefs
- Update al database dei cluster remoti deve essere eseguito se
  - cambia il nome del cluster remoto
  - vengono rimossi i contact nodes del cluster remoto
  - cambiano le chiavi SSL (visto prima)
- Possono insorgere problemi se il parametro **maxblocksize** dei due cluster non e' uguale
  - in generale non sara' possibile montare un file system remoto con block size maggiore della maxblocksize del cluster che vuole montare
  - si deve modificare il parametro in modo opportuno (**richiede cluster restart**)

User mapping

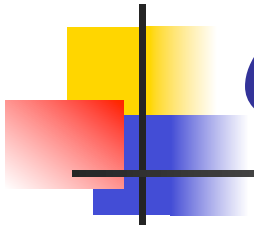
- GPFS supporta la realizzazione di uno UID remapping tra l'home cluster (quello che possiede il file system) ed il remote cluster
- Il meccanismo si basa sulla definizione di un set di Global Unique Name tale che
  - ogni GUN sia associato ad un UID dell'home cluster
    - il contrario non e' indispensabile
  - per ogni UID del cluster remoto esista un GUN, o esista una convenzione per mappare gli utenti senza corrispondente (nobody)
- Si devono implementare due funzioni (mmuid2name e mmname2uid) per ciascun cluster che associno agli UID locali il GUN corrispondente
  - l'interfaccia delle funzioni deve osservare specifiche definite
- Vi sono diverse problematiche non ovvie (default mapping, caching, remapping per output di stat()
  - vedi al riguardo "UID Mapping for GPFS in a Multi-cluster Environment"

Implementazione dello user mapping

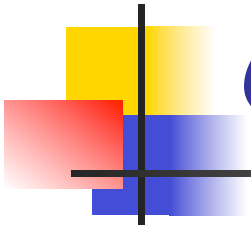
- Definire un set di Global Unique Names (GUN)
- Implementare le funzioni **mmuid2name** e **mmname2uid**, e metterle in **/var/mmfs/etc** (eseguibile da tutti gli utenti) su tutti i nodi del cluster
- Eseguire GPFS shutdown su tutti i nodi del cluster
- Abilitare l'UID remapping sul cluster:
  # mmchconfig enableUIDRemap=yes
  - questo va fatto su entrambi i cluster
- 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> (default: cluster name)

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



Callback



Callback

- Funzionalita' aggiunta dalla versione 3.3
- GPFS e' capace di eseguire scripts in occasione di certi eventi
  - l'amministratore puo' creare script per automatizzare operazioni
- Gli eventi monitorati possono essere
  - eventi globali (comportano l'esecuzione dello script su tutti i nodi del cluster)
  - eventi locali (comportano l'esecuzione dello script solo sul nodo locale

Eventi globali

- Eventi globali
  - nodeJoin, nodeLeave, quorumReached, quorumLoss, quorumNodeJoin, quorumNodeLeave, clusterManagerTakeover (un nuovo cluster manager viene eletto, sia allo startup che in occasione della failure del cluster manager)
- Eventi locali
  - lowDiskSpace (in base ad una policy rule), noDiskSpace, softQuotaExceeded (user o fileset), preMount, preUnmount, mount (successfully), unmount (successfully), preStartup (dopo il join al cluster, ma prima di effettuare qualsiasi recovery), startup, preShutdown, shutdown

Callback management

- Dopo aver realizzato lo script, si deve eseguire **mmaddcallback**, a cui si specificano
  - il nome (unico nel cluster) che si assegna alla callback
  - il path completo dello script (installato su tutti i nodi su cui deve essere eseguito, su file system locale)
  - l'evento (o lista di eventi) che triggerano l'esecuzione della callback
  - priorita' (se ci sono piu' callback per lo stesso evento)
  - se GPFS deve aspettare, eventualmente quanto, e cosa fare in caso di errore (shutdoun o continue)
  - la lista dei nodi su cui installare la callback
  - parametri da passare allo script (vedi man page)
- **# mmlscallback** per visualizzare le callback installate
- **# mmdelcallback** per rimuovere una callback

Esempio di callback

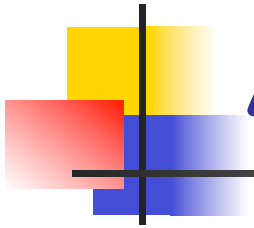
- In occasione di network failure o OS failure, i dischi interni del nodo vengono messi in stato "down" e devono essere messi online manualmente (mmchdisk)
- E' possibile definire una callback per eseguire questa operazione automaticamente:

```
# mmaddcallback downDiskHandler --command  \
  /root/recover_down_disk.sh --event startup -N all  \
  --sync --parms "%eventName %eventNode.shortName"
```

- Questa callback e' installata su tutti i nodi (quindi deve essere copiata su tutti i nodi), e viene eseguita al termine dello startup di GPFS.
- Il contenuto dello script e' mostrato nella prossima slide

/root/recover_down_disk.sh

```
#!/bin/ksh
DAT=`date`
gdd='gpfs_down_disk_'
rsfs='RestripeFS_log'
# get file system device list
/usr/lpp/mmfs/bin/mmlsconfig | grep dev >/tmp/gpfsFS
while read n1 n2
do
   /usr/lpp/mmfs/bin/mmlsdisk $n1 -e | grep -v grep | grep nsd | grep down | \
       awk '{print $1}' >/tmp/$gdd$n1
   while read m1 m2
   do
      echo working on $n1 down disk $m1
      /usr/lpp/mmfs/bin/mmchdisk $n1 start -d "$m1"
   done</tmp/$gdd$n1
   nohup /usr/lpp/mmfs/bin/mmrestripefs $n1 -b >/tmp/$n1$rsfs 2>&1 &
done </tmp/gpfsFS
```



ACL

Access Control List

- Il file system GPFS supporta l'utilizzo di ACL su file e directory in una o entrambe le seguenti modalita'
  - supporto ACL in modalita' **posix** (con estensioni)
  - supporto ACL in modalita' **NFS4** (con limiti)
    - necessari per l'utilizzo del file system su Windows
    - necessari per esportare il file system via NFS v4
- Il file system deve avere l'opzione –k opportunamente definita ("posix", "nfs4" o "all")
- Sono supportati i comandi ordinari getfacl e setfacl per la gestione degli ACL posix
  - per la gestione delle estensioni o degli ACL NFS4 devono essere utilizzati i comandi GPFS: **mmgetacl**, **mmputacl**, **mmeditacl**, **mmdelacl**

ACL posix (tradizionali)

- L'ACL posix (o tradizionale) e' un ACL del tipo:

  user::rwxc
  group::rwxc
  other:rwxc
  mask:rwxc
  user:<uid>:rwxc
  group:<gid>:rwxc

- La flag **c** e' una estensione GPFS agli ACL posix, e riguarda il permesso di modificare l'ACL stesso (in posix solo l'owner puo' farlo)
- ***mask*** contiene i permessi massimi ottenibili dalle altre entry, esclusi ***user::*** (l'owner) e ***other***
  - **gli effettivi permessi per *owner-group*, *named-group* e *named-user* sono calcolati facendo l'AND tra il permesso specifico ed i permessi della *mask***
- L'esecuzione del comando **chmod** si riflette in una corrispondente modifica dell'ACL

ACL posix base

- Su file e directory viene definito un ACL base corrispondente alle permission unix

```
# ls -ld storage

drwxr-xr-x 9 root root 32768 Apr 25  2011 storage

# mmgetacl storage

#owner:root
#group:root
user::rwxc
group::r-x-
other::r-x-
```

- Nell'ACL base non compare l'entry per ***mask***

ACL posix espliciti

- La presenza di ACL espliciti viene segnalata con un + in coda alla stringa delle permission nell'output di "ls –l":

```
# ls -ld atlashotdisk/

drwxr-x--x+ 19 storm storm 32768 Mar 28  2011
atlashotdisk/

# mmgetacl atlashotdisk/

#owner:storm
#group:storm
user::rwxc
group::----
other::--x-
mask::r-x-
group:atlast3:r-x-
group:atlas:--x-
```

Default ACL (posix)

- In aggiunta agli ACL di accesso, una directory puo' anche avere definito un **default ACL**, che viene utilizzato come ACL base per tutte le entry create dentro la directory stessa
  - quando una entry viene creata dentro la directory dotata di default ACL, questo viene **copiato come ACL di accesso** per questa entry
  - le permission per user/group/other vengono quindi definite come **intersezione** tra il default ACL e le permission imposte **dall'umask** in vigore e **dal *mode*** della funzione che ha creato l'oggetto
  - se l'oggetto creato e' una directory, essa eredita il default ACL della parent anche come **default ACL per essa stessa**

NFS v4 ACL

- Gli ACL di tipo NFS v4 sono specificati in tre linee:
  - la prima linea indica diversi campi, separati da ":"
    - il destinatario dell'ACL, specificato come **user:<username>**, **group:<groupname>** o tramite gli identificativi **special:owner@** (owner), **special:group@** (owner group) e **special:everyone@** (tutti gli altri)
    - la traduzione dell'ACL in termini di permission rwxc degli ACL tradizionali
    - il tipo di ACL (**allow** o **deny**)
    - una eventuale flag che puo' avere uno o piu' dei valori **DirInherit**, **FileInherit**, **Inherited** e **InheritOnly**
  - Le seguenti due linee contengono l'elenco dei permessi da garantire o negare

NFS v4 ACL

Esempio di ACL NFS v4

```
#owner:smithj
#group:staff
special:owner@:rwxc:allow:FileInherit
(X)READ/LIST (X)WRITE/CREATE (X)MKDIR (-)SYNCHRONIZE (X)READ_ACL (X)READ_ATTR (-)READ_NAMED
(X)DELETE (X)DELETE_CHILD (X)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (X)WRITE_ATTR (-)WRITE_NAMED

special:owner@:rwxc:allow:DirInherit:InheritOnly
(X)READ/LIST (X)WRITE/CREATE (X)MKDIR (-)SYNCHRONIZE (X)READ_ACL (X)READ_ATTR (-)READ_NAMED
(X)DELETE (X)DELETE_CHILD (X)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED

user:smithj:rwxc:allow
(X)READ/LIST (X)WRITE/CREATE (X)MKDIR (-)SYNCHRONIZE (X)READ_ACL (X)READ_ATTR (-)READ_NAMED
(X)DELETE (X)DELETE_CHILD (X)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

Visualizzare gli ACL

- Gli ACL si visualizzano tramite il comando **mmgetacl**
  - Per default, il comando mostra l'ACL in un formato consistente con il file system setting (flag **k**):
    - se **k** = **posix**, l'ACL e' mostrato nel formato tradizionale
    - se **k** = **nfs4**, l'ACL e' mostrato in formato NFS v4
    - se **k** = **all**, l'ACL e' mostrato nella sua forma nativa
  - Il commando **mmgetacl -k nfs4** mostra sempre un ACL in formato NFS v4
  - Il commando **mmgetacl -k posix** mostra sempre l'ACL in fomrato tradizionale
  - Il commando **mmgetacl -k native** mostra sempre l'ACL nel suo formato nativo
- Il comando **mmgetacl –d** mostra il default ACL (solo posix: NFS v4 non supporta default ACL)

ACL management

- I comandi per la gestione degli ACL sono
  - **mmgetacl**: visualizza un ACL
  - **mmeditacl** o **mmputacl**: crea o modifica un ACL
  - **mmdelacl**: rimuove un ACL, lasciando solo l'ACL base corrispondente alle permissioni unix sull'entry in questione
- Tutti i comandi accettano le opzioni **–d** per i default ACL, e **–k** (con l'eccezione di mmdelacl) per definire il formato di input/ output