



OpenSolaris Share Management

Frank Batschulat

MTS, Solaris Sustaining Engineering

SUN Microsystems

27/05/2009

Why are we here ? Because of this:

```
#man dfstab(4)
```

.....

NOTES

Do not modify this file directly. This file is reconstructed and only maintained for backwards compatibility. Configuration lines could be lost.

Use the `sharemgr(1M)` command for all share management.

Agenda

- Theory
 - > Improved, centralized share management with *sharemgr*(1M), *sharectl*(1M) and *zfs*(1M)
 - > *sharemgr*(1M) - NFS support / CIFS support
 - > *sharectl*(1M) - file sharing protocol configuration
 - > *zfs*(1M) - share management support for ZFS datasets
 - > *share*(1M)/*unshare*(1M) - ???
- Real life examples:
 - > *sharemgr*(1M) “on-the-fly”
 - > *zfs*(1M) share management
 - > *sharemgr*(1M) and share groups

What is a share?

- **Object** defined by the **path** used to make it publicly available (**exported**) (eg. */export/home*)
- possible **protocol specific** configurations/settings (NFS, CIFS)
- Can be part of one share group only
- multiple aliases or “*resource*” names possible

New share management

- introduces concept of “*share groups*”
- Integrated with *smf(5)*
- extensible via plugin modules (NFS, CIFS)
- scripting enabled CLI Tool *sharemgr(1M)*
- CLI to manage protocol aspects separately *sharectl(1M)*
- compatible with traditional *share(1M)*

New: share groups

- Group describing a collection of individual shares
- configuration of shares happens at the group level
 - > common configurations/settings
 - > activation/deactivation per share group
- overwrite of group configurations/settings still possible at individual share level
- 2 share groups created by default
 - > “*default*” - for all existing or new manual */etc/dfs/dfstab* NFS shares
 - > “*zfs*” – for all ZFS datasets with property “*sharenfs=on*”
- Groups can be for NFS, CIFS or NFS and CIFS protocols

Example:

```
#sharemgr show
```

```
default
```

```
    /export/home
```

```
    /spare
```

```
    /spare2
```

```
zfs
```

```
    zfs/pool/zfs1
```

```
        /pool/zfs1
```

```
    zfs/pool/zfs2
```

```
        /pool/zfs2
```

```
#share
```

```
-    /export/home sec=sys,rw=client1:client2:client3 "server_home"
```

```
-    /spare sec=sys,ro=client4 "server_spare"
```

```
-    /spare2 sec=sys,ro=client4 "server_spare2"
```

```
-@pool/zfs1 /pool/zfs1 sec=sys,rw=client1:client2 "server_zfs1"
```

```
-@pool/zfs2 /pool/zfs2 sec=sys,rw=client3:client4 "server_zfs2"
```

Resource in *sharemgr(1M)*

- In NFS, a “*resource*” is just an alias for the share path, optional
- In CIFS, “*resources*” are the CIFS share itself inclusively any options
 - > CIFS can share one path multiple times
 - > every export may have different configurations/settings
 - > settings can be inherited/left from a share or entire share group
- “*resource*” names must be unique (“-r” option for *add-share*)
- “*resource*” names are case insensitive
- “*resource*” names are mandatory for CIFS share groups

smf(5) integration

- every share group is implemented as a *smf(5)* service instance

#svcs group

online - 9:03:42 - svc:/network/shares/group:zfs

online - 9:03:47 - svc:/network/shares/group:default

- > start/stop in parallel
 - > configuration saved in *smf(5)* repository
 - > New *smf(5)* service instance created for every share group
 - > “Shares, Resources” and “Option Sets” are implemented as *smf(5)* “property groups”
- start/stop of ZFS shares via *smf(5)* as well, but configuration stored in *zfs(1M)* “properties”
 - > every ZFS dataset with “*sharenfs/sharesmb*” option is a sub-group
 - > There's only one “*zfs*” *smf(5)* service instance

sharectl(1M)

- Used to configure protocol specific settings
 - > for NFS everything listed in *nfs(4)* / */etc/default/nfs*
 - > for CIFS everything listed in *smb(4)* / *nsmbrc(4)*
- *sharectl(1M)* automatically restarts changed services
- *sharectl get* / *sharectl set* [-p option] [protocol] to read/change settings

```
#sharectl get nfs
```

```
[...]
```

```
servers=16
```

```
server_versmin=2
```

```
server_versmax=4
```

```
client_versmin=2
```

```
client_versmax=4
```

```
server_delegation=on
```

```
nfsmapid_domain=
```

zfs(1M) Share Management

- Possible via 2 different ways:
 - > using *sharemgr*(1M) / *share*(1M)
 - > using *zfs*(1M)

zfs(1M) sharing:

- configuration/settings stored in *zfs*(1M)
“*sharenfs/sharesmb*” “properties” per dataset
- every exported dataset will be a sub-group in *sharemgr*(1M) (within share group “*zfs*”)
- sub-datasets which inherit “*sharenfs/sharesmb*” property will be part of the same sub-group in *sharemgr*(1M)

...zfs(1M) sharing

- *zfs(1M)* can automatically export datasets during *mount(2)*
- “*sharenfs/sharesmb*” property “= *on*”, during mount, export via *zfs share* using *sharemgr(1M)*(CIFS) or *share(1M)*(NFS) (without specific options, R/W for everyone per default) or manually via *zfs share/unshare*
- “*sharenfs/sharesmb*” property “= *off*”, export manually only via *share(1M)/sharemgr(1M)*
- You still use the usual NFS *share(1M)* configuration parameters via *zfs set sharenfs = 'share(1M) opt1, opt2, ...'*, they will be automatically handed over to *share(1M)* by *zfs share* during mount
- Change of “*sharenfs/sharesmb*” dataset property will automatically result in re-share of this dataset (and all its sub-datasets that inherit this property) using the new settings

share(1M) / unshare(1M) ?

- usual *share(1M)* behavior has not changed, internally uses new *libshare* like *sharemgr(1M)*
- Do not edit */etc/dfs/dfstab* file, see *dfstab(4)*
- Same for *unshare(1M)*

Real life examples

sharemgr(1M) “on-the-fly”

- manually, without share groups, ad-hoc same way like traditional *share*(1M), same syntax (regardless if ZFS or UFS export)

```
#sharemgr share -F nfs -o sec=sys,rw=client1:client2 -d zfstest /pool/nfs
```

```
#share
```

```
- /pool/nfs sec=sys,rw=client1:client2 "zfstest"
```

```
#sharemgr show -vp
```

```
default nfs=()
```

```
* /pool/nfs nfs:sys=(rw="client1:client2") "zfstest"
```

```
#sharemgr unshare /pool/nfs
```

- Hint: *sharemgr share* “-p” option makes export persistent

sharing with zfs(1M)

```
#zfs create pool/nfs
```

```
#zfs create pool/nfs/share1
```

```
#zfs create pool/nfs/share2
```

```
#zfs set sharenfs=on pool/nfs
```

(automatic export happens here, "sharenfs" will be left !)

```
#sharemgr show -vp zfs
```

```
zfs
```

```
zfs/pool/nfs nfs=()
```

```
  /pool/nfs
```

```
  /pool/nfs/share1
```

```
  /pool/nfs/share2
```

```
#share
```

```
-@pool/nfs  /pool/nfs rw ""
```

```
-@pool/nfs  /pool/nfs/share1 rw ""
```

```
-@pool/nfs  /pool/nfs/share2 rw ""
```

```
#unshareall      (unshare of all exports)
```

...sharing with *zfs*(1M)

```
#zfs share pool/nfs/share1
```

```
#zfs share pool/nfs/share2
```

```
#share
```

```
-@pool/nfs /pool/nfs/share1 rw ""
```

```
-@pool/nfs /pool/nfs/share2 rw ""
```

```
#zfs unshare -a      (unshare of all zfs exports)
```

```
#zfs share -a       (share of all zfs exports)
```

```
#share
```

```
-@pool/nfs /pool/nfs rw ""
```

```
-@pool/nfs /pool/nfs/share1 rw ""
```

```
-@pool/nfs /pool/nfs/share2 rw ""
```

...sharing with zfs(1M) - changing protocol specific settings

- #share

```
-@pool/nfs/share1 /pool/nfs/share1 rw ""
```

```
-@pool/nfs/share2 /pool/nfs/share2 rw ""
```

```
#sharemgr show -vp zfs
```

```
zfs
```

```
zfs/pool/nfs/share1 nfs=()
```

```
    /pool/nfs/share1
```

```
zfs/pool/nfs/share2 nfs=()
```

```
    /pool/nfs/share2
```

```
#zfs set sharenfs='rw=client1,sec=sys' pool/nfs/share1
```

(delete with `zfs set sharenfs=off` (unshare) or `=on` (unshare/re-share))

```
#sharemgr show -vp zfs
```

```
zfs
```

```
zfs/pool/nfs/share1 nfs=() nfs:sys=(rw="client1")
```

```
    /pool/nfs/share1
```

```
zfs/pool/nfs/share2 nfs=()
```

```
    /pool/nfs/share2
```

```
#share
```

```
-@pool/nfs/bat /pool/nfs/share1 sec=sys,rw=client1 ""
```

```
-@pool/nfs/non /pool/nfs/share2 rw ""
```

...sharing with *zfs*(1M) - inheritance of protocol specific settings

- *zfs*(1M) datasets can leave/inherit settings

```
#zfs set sharenfs='sec=dh,rw=client1,ro=client2' pool/nfs
```

```
#sharemgr show -vp zfs
```

```
zfs
```

```
zfs/pool/nfs nfs=() nfs:dh=(rw="client1" ro="client2")
```

```
  /pool/nfs
```

```
  /pool/nfs/share1
```

```
#share
```

```
-@pool/nfs  /pool/nfs  sec=dh,rw=client1,ro=client2  ""
```

```
-@pool/nfs  /pool/nfs/share1  sec=dh,rw=client1,ro=client2  ""
```

```
#zfs get sharenfs pool/nfs
```

NAME	PROPERTY	VALUE	SOURCE
pool/nfs	sharenfs	sec=dh,rw=client1,ro=client2	local

```
#zfs get sharenfs pool/nfs/share1
```

NAME	PROPERTY	VALUE	SOURCE
pool/nfs/share1	sharenfs	sec=dh,rw=client1,ro=client2	inherited from pool/nfs

sharemgr(1M) and share groups

sharemgr create -P nfs ufsexports (SMB and NFS will be enabled by default without protocol info !)

sharemgr set -P nfs -p anon=0 ufsexports

sharemgr add-share [-d server_test] -s /test ufsexports (delete with **sharemgr remove-share** -s path)

sharemgr add-share [-d server_home] -s /export/home ufsexports

sharemgr show -vp ufsexports

ufsexports nfs=(anon="0")

 /test "server_test"

 /export/home "server_home"

share

-@ufsexports /test anon=0 "server_test"

-@ufsexports /export/home anon=0 "server_home"

svcs group

STATE STIME FMRI

online 14:01:02 svc:/network/shares/group:zfs

online 14:01:02 svc:/network/shares/group:default

online 15:40:38 svc:/network/shares/group:ufsexports

sharemgr(1M) – NFS share group configuration

- ```

sharemgr set -P nfs -S sys -p rw=client1 ufsexports (-S optionset for protocol defined by -P, here for NFS, are nfssec(5) Security Flavours)
sharemgr show -vp ufsexports
ufsexports nfs=(anon="0") nfs:sys=(rw="client1")
 /test "server_test"
 /export/home "server_home"

share
-@ufsexports /test sec=sys,rw=client1 "server_test"
-@ufsexports /export/home sec=sys,rw=client1 "server_home"

sharemgr set -P nfs -S sys -p ro=client2,client3 ufsexports
sharemgr show -vp ufsexports
ufsexports nfs=() nfs:sys=(rw="client1" ro="client2:client3")
 /test "server_test"
 /export/home "server_home"

share
-@ufsexports /test sec=sys,rw=client1,ro=client2:client3 "server_test"
-@ufsexports /export/home sec=sys,rw=client1,ro=client2:client3 "server_home"

```

## ...sharemgr(1M) – NFS share group configuration

- Removing a NFS share configuration setting:

```
sharemgr unset -P nfs -S sys ufsexports
```

```
sharemgr show -vp ufsexports
```

```
ufsexports nfs=(anon="0")
```

```
 /test "server_test"
```

```
 /export/home "server_home"
```

```
share
```

```
-@ufsexports /test anon=0 "server_test"
```

```
-@ufsexports /export/home anon=0 "server_home"
```

```
sharemgr unset -P nfs -p anon ufsexports
```

```
sharemgr show -vp ufsexports
```

```
ufsexports nfs=()
```

```
 /test "server_test"
```

```
 /export/home "server_home"
```

```
share
```

```
-@ufsexports /test rw "server_test"
```

```
-@ufsexports /export/home rw "server_home"
```

## Unshare with *sharemgr(1M)* and share groups

- deactivating a share group (unshare ***and*** no longer export automatically at boot time)

**#sharemgr disable** ufsexports

- deactivating a share group (unshare ***without*** configuration change)

**#sharemgr stop** ufsexports

- removing a share group

**#sharemgr delete** ufsexports (-f option if not empty)

## more informations:

- <http://www.opensolaris.org/os/community/nfs/>
- [http://opensolaris.org/os/project/brosug/nfs4\\_frankb-1.pdf](http://opensolaris.org/os/project/brosug/nfs4_frankb-1.pdf)
- [http://developers.sun.com/solaris/articles/nfs\\_zfs.html](http://developers.sun.com/solaris/articles/nfs_zfs.html)
- <http://blogs.sun.com/doug/m/>
- <http://www.opensolaris.org/os/project/cifs-server>
- <http://frsun.downloads.edgesuite.net/sun/08B01361>
- <http://docs.sun.com/app/docs/doc/819-1634/rfstm-1?a=view>
- <http://docs.sun.com/app/docs/doc/820-2429/managingsmbsharestm?a=view>
- <http://docs.sun.com/app/docs/doc/817-2271/gamnd?a=view>
- Man pages: *sharemgr*(1M), *sharectl*(1M), *zfs*(1M), *share*(1M), *share\_nfs*(1M), *sharefs*(7FS)



**#unshareall ; init 5**

**Frank Batschulat**

**frank.batschulat@sun.com**