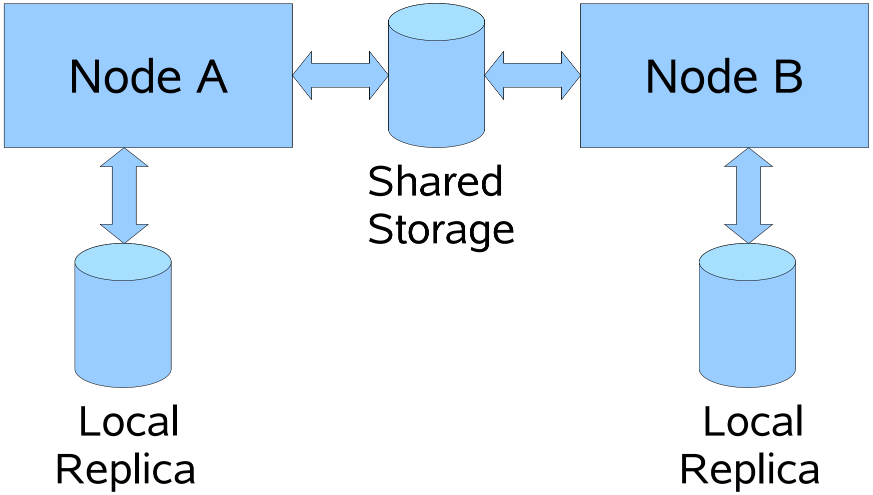
# Solaris Volume Manager: Metaset Creation Example

# Overview

- Hardware Configuration
- Command Execution
- Resulting system configuration
- Replica changes

`

# Hardware Configuration



# Commands Executed

- Create local replicas on NodeA and NodeB
- Command:
  - metaset -s foo -ah NodeA NodeB

# Resulting metaconfiguration

- NodeA & NodeB

Set name = foo, Set number = 1

| Host | Owner |
|---|---|
| NodeA | |
| NodeB | |

# Replica changes

- The only change will be the addition of a set record into each of the nodes' local replicas:

```
RecId 0x00000003: Type:USER      [0005] Type2: Set     Size = 1212
     sr_revision=0x00010000  sr_flags=0x80000000     sr_selfid=0x00000003
     sr_genid=2      sr_setno=1      sr_setname="foo"
     sr_ctime=Wed Jan 11 14:18:38 2006
             1137014318 [    359139]
     sr_mhiargs.mh_ff=1000
     sr_mhiargs.mh_tk.reinstate_resv_delay=6000
     sr_mhiargs.mh_tk.min_ownership_delay=6000
     sr_mhiargs.mh_tk.max_ownership_delay=30000
     sr_driverec=0x00000000
     sr_med.n_cnt=0
            sr_med.n_lst[0].a_cnt=0
            sr_med.n_lst[1].a_cnt=0
            sr_med.n_lst[2].a_cnt=0
     sr_nodes[0]="NodeA"
     sr_nodes[1]="NodeB"
```

# Diskset operations through metaset

- Operations to create disksets and add/delete nodes, disks, and mediators require that all of the nodes in the diskset contain identical information in their local replicas
  - These operations require additional coordination across hosts (eg. Ensuring that a set number and name are not currently used on any of the potential nodes in a diskset before allowing creation of that diskset
- This coordination is done through the daemon, rpc.metad

# Code Structure for RPC calls in metad

- Versioned
  - Rolling upgrade support in SunCluster was a major factor in making this change.  SunCluster will no longer support rolling upgrade.
  - Interfaces
    - When metarpcopen is called it returns a client handle, CLIENT.  This contains the interface version number.
  - Over-the-wire structures
    - A version number is included in the over-the-wire structure
- Code Flow
  - Calls are very similar – walking through one will give great insight into how almost all are structured and operate

# RPC Code Flow

- Clnt_*
  - Entry point for rpc encapsulation
  - Different classes of rpc calls in rpc.metad
    - Change state of local replica (clnt_createset, clnt_adddrvs)
    - Get information (clnt_devinfo, clnt_drvused)
    - Control (clnt_lock_set, clnt_unlock_set)
- Versioned args structure

# clnt_addhosts

```
int
clnt_addhosts(
        char                *hostname,
        mdsetname_t         *sp,
        int                 node_c,
        char                **node_v,
        md_error_t          *ep
)
{
        CLIENT              *clntp;
        mdrpc_host_args     *args;
        mdrpc_host_2_args   v2_args;
        mdrpc_generic_res   res;
        int                 version;

        /* initialize */
        mdclrerror(ep);
        (void) memset(&v2_args, 0, sizeof (v2_args));
        (void) memset(&res, 0, sizeof (res));
```

- hostname is the name of the node to add the specified nodes to
- node_v is the set of node names being added
- mdrpc_host_args is the version 1 over the wire structure
- mdrpc_host_2_args is the version 2 over the wire structure
- mdrpc_generic_res is the structure that contains values returned from this call

# clnt_addhosts – build the arguments

```
/* build args */
v2_args.rev = MD_METAD_ARGS_REV_1;
args = &v2_args.mdrpc_host_2_args_u.rev1;
args->sp = sp;
args->cl_sk = cl_get_setkey(sp->setno, sp->setname);
args->hosts.hosts_len = node_c;
args->hosts.hosts_val = node_v;
```

- The version 2 args are normally a superset of the version 1 arguments so encapsulate them

# clnt_addhosts – run on current node

```
  /* do it */

  if (md_in_daemon &&
strcmp(mynode(), hostname) == 0) {

        int bool;

        bool =
mdrpc_addhosts_2_svc(&v2_args,
&res, NULL);

        assert(bool == TRUE);

        (void) mdstealerror(ep,
&res.status);
```

- If the hostname is the current node then call the function directly rather than through rpc

# clnt_addhosts – set up for rpc call

```
} else {
    if ((clntp = metarpcopen(hostname, CL_LONG_TMO, ep)) == NULL)
        return (-1);
    /*
     * Check the client handle for the version and invoke
     * the appropriate version of the remote procedure
     */
    CLNT_CONTROL(clntp, CLGET_VERS, (char *)&version);
```

- Metarpcopen
  - Verifies that the core SMF services are enabled
  - Try to create a version 2 client handle by default. If this fails then attempt to create a version 1 client handle

# clnt_addhosts – make rpc call

```
            if (version == METAD_VERSION) { /* version 1 */
             if (mdrpc_addhosts_1(args, &res, clntp) != RPC_SUCCESS)
                        (void) mdrpcerror(ep, clntp, hostname,
                        dgettext(TEXT_DOMAIN, "metad add hosts"));
                else
                        (void) mdstealerror(ep, &res.status);
        } else {
                if (mdrpc_addhosts_2(&v2_args, &res, clntp) !=
                    RPC_SUCCESS)
                        (void) mdrpcerror(ep, clntp, hostname,
                        dgettext(TEXT_DOMAIN, "metad add hosts"));
                else
                        (void) mdstealerror(ep, &res.status);
        }

        metarpcclose(clntp);
}
```

# Main Line Flow

- There are variations in the flow based upon whether this is an Oban, autotake, or traditional diskset
- This code walkthrough is for a traditional diskset

# Metaset – setup and cmd line parse

- Bind SunCluster library
  - Proxy commands to primary node if applicable
  - If the dlopen of the libsds_sc.so.1 library fails then all of the sdssc_* functions will be bound to 'not_bound' which simply returns SDSSC_NOT_BOUND
- Open admin device
  - Kernel level called via ioctls
- Install signal handlers

# Metaset – local sanity checks

- Parse the command line parameters
  - Test for conflicting parameters
- Check for root privs
  - Must run as root for anything other than printing set info
- Get a lock on the local set
  - Necessary since the local replica will be updated
- Verify that all of the nodes specified on the command line are unique and valid

# Metaset – create_set checks

- Verify that the current node is in the new diskset node list
- Verify that the setname is not already being used on any of the nodes.  This is done by checking the setrecord cache in rpc.metad (clnt_getset)
- Find a set number that is not being used on any of the nodes
  - Start with the first available on the current node and check on all nodes until an available set number is found or we run out of set numbers (clnt_setnumbusy)
- Check the setname for valid syntax
- Verify that the link, '/dev/md/<diskset>' does not exist on any of the nodes and verify again that the set name is not in the setrecord cache (clnt_setnameok)

# Metaset – create_set create the set

- Get a lock on the set on all nodes (clnt_lock_set)
- Create the set on all of the nodes (clnt_createset)
  - Get the next user record number by calling metaioctl with MD_DB_USERREQ
  - Turn on the SVM diskset SMF services if they are not already on
  - Commit the set USER record
- Release the lock on all of the nodes (clnt_lock_??)

# SVM
# Metaset Creation Example