SAN Tape Shared Storage Option

Bacula Systems Documentation

Contents

Contents

1 Overview

This white paper presents various techniques and strategies for tape backup on SAN networks using the tape Shared Storage option (plugin) with **Bacula Enterprise**.

1.1 Scope

The current version of the Shared Storage plugin supports SCSI persistent reservation using SPC-3 on tape drives in Bacula Enterprise and later. The older form of SPC-2 SCSI reserve is not supported by **Bacula Enterprise**.

2 SAN Backup

A SAN network of SCSI devices has several advantages over a LAN (Ethernet) network.

Faster Generally a SAN is faster than a LAN. SANs are typically 2Gb to 10Gb while LANs are commonly 1Gb (there are, of course, 10GB LANs).

Connectivity In a SAN, each SCSI device appears to each of the hosts connected to the SAN (unless you use zoning or VLAN techniques). This has the advantage that a tape library (autochanger) can appear to be a directly connected SCSI device to two or more servers at the same time.

Backups Since the SCSI devices (tapes) appear to be directly connected, backups are faster and do not involve sending data over the LAN (LAN-free-backup).

The problem with backing up multiple servers at the same time to the same tape library (or autoloader) is that if both servers access the same tape drive same time, you will very likely get data corruption. This is where the **Bacula Enterprise** shared storage plugin comes into play. The plugin ensures that only one server at a time can connect to each device (tape drive) by using the SPC-3 SCSI reservation protocol.

2.1 Using Shared Storage

To use the shared storage plugin with **Bacula Enterprise**, you must have the shared storage plugin (**shstore**) installed and configured, you must have a tape library (autochanger) that supports the SPC-3 protocol, and you must install a package named **sg3_utils** on either a RedHat or Ubuntu system.

The next few sections describes how to install, configure, and use the shared storage option.

2.2 Packages

First you must have a working Bacula with multiple Storage daemons installed, and on each of the machines with the Storage daemons, you must install the **shstore** plugin package.

A package is available for RedHat Enterprise and Ubuntu LTS. Contact Bacula System to get them. A version for Solaris may be available on request, but is not currently implemented.

```
# rpm -ivh bacula-enterprise-shstore_6.0.0.rpm
  or
# dpkg -i bacula-enterprise-shstore_6.0.0.deb
```

These packages will ensure that your **Bacula Enterprise** version is compatible with the shstore plugin and will install **shstore-sd.so** and **storage-ctl** programs.

```
/opt/bacula# ls plugins/shstore* scripts/storage*
-rwxr-x--- 1 bacula tape 60463 Dec 15 12:27 plugins/shstore-sd.so
-rwxr-x--- 1 bacula tape  7147 Dec 15 11:44 scripts/storage-ctl
-rwxr-x--- 1 bacula tape   589 Dec 15 11:44 scripts/storage-ctl.conf
```

It is also possible to install packages through your package managing system (**apt**, or **yum**).

Example:

```
% grep bacula /etc/apt/sources.list
deb https://www.baculasystems.com/dl/<customer>/debs/main/6.0.0/squeeze-64 squeeze main
deb https://www.baculasystems.com/dl/<customer>/debs/shstore/6.0.0/squeeze-64 squeeze␣
↪shstore
```

If you have troubles to setup your package manager, feel free to contact Bacula Systems support team.

2.3 Director Configuration Version 6.0.1

This section only applies to Bacula Enterprise version 6.0.1. For later versions of Bacula Enterprise, this section no longer applies, please see the next section.

The **Shared Storage** directive should be used when using the SAN Shared Storage plugin or when accessing from the Director Storage resources directly to Devices of an Autochanger.

```
Director {
  Name = bacula-dir
  Shared Storage = yes   # Not used in version 6.0.2
...
}
```

When sharing volumes between different Storage resources, you will need to execute the **reset-storageid** script before using the **update slots** command. This script can be scheduled once a day in a CRON or Admin job.

```
$ /opt/bacula/scripts/reset-storageid MediaType StorageName
$ bconsole
* update slots storage=StorageName drive=0
```

The above is not used in version 6.0.2, please see below.

2.4 Director Configuration Version 6.0.2

The **Share Storage** directive is removed in Bacula Enterprise 6.0.2, and the **reset-storageid** script and **update slots** command described above are no longer needed in Bacula Version 6.0.2. However, to make Bacula function properly, you must adapt your **bacula-dir.conf Storage** directives.

Each autochanger that you have defined in an **Autochanger** resource in the Storage daemon's **bacula-sd.conf** file, must have a corresponding **Autochanger** resource defined in the Director's **bacula-dir.conf** file. Normally you will already have a **Storage** resource that points to the Storage daemon's **Autochanger** resource. Thus you need only to change the name of the **Storage** resource to **Autochanger**. In addition no **Autochanger = yes** directive is needed in the Director's **Autochanger** resource.

In addition to the above change (**Storage** to **Autochanger**), you must modify any additional **Storage** resources that correspond to devices that are part of the **Autochanger** device. Instead of the previous **Autochanger = yes** directive they should be modified to be **Autochanger = xxx** where you replace the **xxx** with the name of the Autochanger.

For example, in the bacula-dir.conf file:

```
Autochanger {             # New resource
  Name = Changer-1
  Address = cibou.company.com
  SDPort = 9103
  Password = "xxxxxxxxxx"
  Device = LTO-Changer-1
  Media Type = LTO-9
  Maximum Concurrent Jobs = 50
}

Storage {
```

(continues on next page)

(continued from previous page)

```
  Name = Changer-1-Drive0
  Address = cibou.company.com
  SDPort = 9103
  Password = "xxxxxxxxxx"
  Device = LTO9_1_Drive0
  Media Type = LTO-9
  Autochanger = Changer-1  # New directive
  Maximum Concurrent Jobs = 5
}

Storage {
  Name = Changer-1-Drive1
  Address = cibou.company.com
  SDPort = 9103
  Password = "xxxxxxxxxx"
  Device = LTO9_1_Drive1
  Media Type = LTO-9
  Autochanger = Changer-1  # New directive
  Maximum Concurrent Jobs = 5
}

...
```

Note that Storage resources **Changer-1-Drive0** and **Changer-1-Drive1** are not required since they make up part of an autochanger. However, by referring to those Storage definitions in a job, you will be sure to use only the indicated drive. This is probably most used for reserving a drive for restores. See the Storage daemon example .conf below and the use of **AutoSelect = no**.

So, in summary, the changes are:

- Remove the **Shared Storage = yes** from the **Director** resource.
- Change **Storage** to **Autochanger** in the LTO9 resource.
- Remove the **Autochanger = yes** from the **Autochanger** LTO9 resource.
- Change the **Autochanger = yes** in each of the **Storage** device that belong to the **Autochanger** to point to the **Autochanger** resource with for the example above the directive **Autochanger = LTO9**.

2.5 Sharing an Autochanger

If you have a second Storage daemon that shares an Autochanger with the your first Storage daemon defined above, you define it much the same way as above, with of course different names, but in addition, you must tell the Director that this second Autochanger is shared with the first one. You do so, by adding a new directive: **Shared Storage = <autochanger-name>**.

For example, in the bacula-dir.conf file:

```
Autochanger {                # New resource
  Name = Changer-2
  Address = dir.company.com
  SDPort = 9103
  Password = "yyyyyyy"
  Device = LTO-Changer-2
```

(continues on next page)

(continued from previous page)

```
  Media Type = LTO-9
  Maximum Concurrent Jobs = 50
  Shared Storage = Changer-1  # New directive
}

Storage {
  Name = Changer-2-Drive0
  Address = dir.company.com
  SDPort = 9103
  Password = "yyyyyyy"
  Device = LTO9_2_0
  Media Type = LTO-9
  Autochanger = Changer-2     # New directive
  Maximum Concurrent Jobs = 5
}

Storage {
  Name = Changer-2-Drive1
  Address = dir.company.com
  SDPort = 9103
  Password = "yyyyyyy"
  Device = LTO9_2_1
  Media Type = LTO-9
  Autochanger = Changer-2     # New directive
  Maximum Concurrent Jobs = 5
}
...
```

2.6 Storage Configuration

The **Plugin Directory** option in the **Storage daemon** resource must point where the **shstore-sd.so** plugin is installed. Generally **/opt/bacula/plugins**

```
Storage {
   Name = bacula-sd
   Plugin Directory = /opt/bacula/plugins
...
}
```

2.7 New Device Directives

There are two new directives in the **bacula-sd.conf** Device resource:

Control Device = <device-name> The control device is the SCSI control device that corresponds to the **Archive Device**. See below for details.

Lock Command = <command> This is the command that must be executed to effect the SCSI persistent locking. Generally, it will be the following:

```
Lock Command = "/opt/bacula/scripts/storage-ctl %l %o"
```

Note, the edit codes above are percent small-el percent small-oh.

Your Storage daemons should have an appropriate HBA card that permits direct connections to a SAN. You can examine what SCSI devices are attached with the **lsscsi** command on the first Storage daemon machine **cibou** as follows:

```
/opt/bacula# lsscsi -g
[1:0:0:0] tape    HP       Ultrium 4-SCSI   H61W  /dev/st0  /dev/sg0
[1:0:0:1] tape    HP       Ultrium 4-SCSI   H61W  /dev/st1  /dev/sg1
[1:0:0:2] mediumx HP       MSL G3 Series    E.00  -         /dev/sg2
[1:0:1:0] mediumx NETAPP   VTL              0001  -         /dev/sg3
[1:0:1:1] tape    QUANTUM  DLT7000          022C  /dev/st2  /dev/sg4
[1:0:1:2] tape    QUANTUM  DLT7000          022C  /dev/st3  /dev/sg5
[2:0:0:0] disk    ATA      ST3250824AS      3.AD  /dev/sda  /dev/sg6
[3:0:0:0] cd/dvd  TSSTcorp DVD+-RW TS-H553A DE04  /dev/scd0  /dev/sg7
```

The above machine, **cibou**, is rather simple because it has only a single disk drive, with two autochangers, the first autochanger is an HP LTO-9 with two drives, and the second a NETAPP VTL with two DLT7000 drives.

Then on a different machine named **dir**, you may get something like the following:

```
/opt/bacula# lsscsi -g
[0:0:0:0] cd/dvd  HL-DT-ST DVD-ROM GDR8163B 0B26  /dev/sr0  /dev/sg0
[2:0:0:0] disk    ATA      Hitachi HDS72302 MN6O  /dev/sda  /dev/sg1
[3:0:0:0] disk    ATA      Hitachi HDS72302 MN6O  /dev/sdb  /dev/sg2
[3:0:1:0] disk    ATA      Hitachi HDS72302 MN6O  /dev/sdc  /dev/sg3
[4:0:0:0] tape    HP       Ultrium 4-SCSI   H61W  /dev/st0  /dev/sg4
[4:0:0:1] tape    HP       Ultrium 4-SCSI   H61W  /dev/st1  /dev/sg5
[4:0:0:2] mediumx HP       MSL G3 Series    E.00  /dev/sch0  /dev/sg6
[4:0:1:0] mediumx NETAPP   VTL              0001  /dev/sch1  /dev/sg7
[4:0:1:1] tape    QUANTUM  DLT7000          022C  /dev/st2  /dev/sg8
[4:0:1:2] tape    QUANTUM  DLT7000          022C  /dev/st3  /dev/sg9
```

The above machine, **dir**, has three disk drives, then the same two autochangers as on the **cibou** machine, the first an HP LTO-9 with two drives, and the second a NETAPP VTL with two DLT7000 drives. Note, the HP autochanger and the NETAPP VTL are the same devices, but they appear slightly differently on each machine, because the SCSI device names are quite different.

2.8 Configuring the Devices

Now, let's construct Bacula Device resources in **bacula-sd.conf** for each of these machines starting with **cibou**.

```
Device {
  Name = tape
  Media Type = tape
  AutomaticMount = yes;
  RemovableMedia = yes;
  Archive Device = /dev/nst0

  AlwaysOpen = no;
  Control Device = /dev/sg0
  Lock Command = "/opt/bacula/scripts/storage-ctl %l %o"
}
```

There are three things that are different from a standard tape **Device** resource:

- There is a new **Control Device** directive.
- There is a new **Lock Command** directive.
- We have set **AlwaysOpen** to **no** rather than **yes**.

If you examine the first **lsscsi** output above, you will see that the tape device **Ultrium 4-SCSI** has an archive address of **/dev/st0** while the SCSI control channel is **/dev/sg0**.

For the equivalent for machine **dir**, the second **lsscsi** example shown above, would be:

```
Device {
  Name = tape
  Media Type = tape
  AutomaticMount = yes;
  RemovableMedia = yes;
  Archive Device = /dev/nst0

  AlwaysOpen = no;
  Control Device = /dev/sg4
  Lock Command = "/opt/bacula/scripts/storage-ctl %l %o"
}
```

Note that only the SCSI control channel address has changed.

You might ask: Why do we set **AlwaysOpen = no**. The answer is that when Bacula opens the device, it will be locked thus preventing any other machines (Storage daemons) from accessing it. If we have **AlwaysOpen = yes** the device will be locked as long as the Storage daemon is active. When **AlwaysOpen** is set to **no** the device will be closed when Bacula no longer uses the device and in closing the device, Bacula will also release the lock thus permitting other machines to access the device.

2.9 Complete Storage daemon configuration

A more complete Storage daemon configuration file **bacula-sd.conf** that corresponds to the example of the two autochanger configuration for two Storage daemons given above might be:

```
# Storage definition in bacula-sd.conf on machine cibou.company.com
Storage {
  Name = xxx
  ...
}

Autochanger {
  Name = LTO-Changer-1
  Changer Device = /dev/sg2
  Changer Command ="/opt/bacula/scripts/mtx-changer %c %o %S %a %d"
  Device = LTO9_1_Drive0, LTO9_1_Drive1
}


Device {
  Name = LTO9_1_Drive0
  Media Type = LTO-9
  Archive Device = /dev/nst0
  AutomaticMount = yes
```

(continues on next page)

(continued from previous page)

```
  Autochanger = yes
  Drive Index = 0
  RemovableMedia = yes
  Spool Directory =  /opt/bacula/working
  Device Type = Tape

  AlwaysOpen = no
  Control Device = /dev/sg0
  Lock Command = "/opt/bacula/scripts/storage-ctl %l %o"
}

Device {
  Name = LTO9_1_Drive1
  Media Type = LTO-9
  Archive Device = /dev/nst1
  AutomaticMount = yes
  Autochanger = yes
  Drive Index = 0
  RemovableMedia = yes
  Spool Directory =  /opt/bacula/working
  Device Type = Tape
  AutoSelect = no    # reserved for restores

  AlwaysOpen = no
  Control Device = /dev/sg1
  Lock Command = "/opt/bacula/scripts/storage-ctl %l %o"
}
```

and for the second Storage daemon that uses the same autochanger on a SAN network:

```
# Storage definition in bacula-sd.conf on machine dir.company.com
Storage {
  Name = yyy
  ...
}

Autochanger {
  Name = LTO-Changer-2
  Changer Device = /dev/sg6
  Changer Command ="/opt/bacula/scripts/mtx-changer %c %o %S %a %d"
  Device = LTO9_2_Drive0, LTO9_2_Drive1
}


Device {
  Name = LTO9_2_Drive0
  Media Type = LTO-9
  Archive Device = /dev/nst0
  AutomaticMount = yes
  Autochanger = yes
  Drive Index = 0
  RemovableMedia = yes
```

(continues on next page)

(continued from previous page)

```
  Spool Directory =  /opt/bacula/working
  Device Type = Tape

  AlwaysOpen = no
  Control Device = /dev/sg4
  Lock Command = "/opt/bacula/scripts/storage-ctl %l %o"
}

Device {
  Name = LTO9_2_Drive1
  Media Type = LTO-9
  Archive Device = /dev/nst1
  AutomaticMount = yes
  Autochanger = yes
  Drive Index = 1
  RemovableMedia = yes
  Spool Directory =  /opt/bacula/working
  Device Type = Tape
  AutoSelect = no    # reserved for restores

  AlwaysOpen = no
  Control Device = /dev/sg5
  Lock Command = "/opt/bacula/scripts/storage-ctl %l %o"
}
```

2.10 Configuring the storage-ctl Script

Normally, the file **storage-ctl** found in the Bacula scripts directory should never be changed. On the other hand, it must be configured with the **storage-ctl.conf** file. The default file looks like:

```
#
# This file is sourced by the storage-ctl script every time it runs.
#   You can put your site customization here, and when you do an
#   upgrade, the process should not modify this file.  Thus you
#   preserve your storage-ctl configuration.
#


#
# The following key must be a unique 6 hex digit value on each SD
#
key=bac001

# Set to 1 if you want debug info written to a log
debug_log=0

# Set to path to your sg_persist program
SG_PERSIST=/usr/bin/sg_persist

# Set to the maximum number of seconds to wait for a scsi lock
#  Default = 30*60 = 1800 or 30 minutes
max_lock_wait=1800
```

However, for operational reasons, you will certainly want to change they key (first configuration item) to be different (and unique) on each machine. This key is used by the reservation system, and it is far better to have unique keys on each Storage daemon so that in Bacula status command and in debug output it will be clear which machine holds what reservation.

The key is a 6 digit hexadecimal value (i.e. 0-9 and a-f), where the letters may be either uppercase or lowercase. If you enter a character that is not a valid hexadecimal character or more than 6 characters, the locking will fail, and thus your jobs will fail.

For the examples in this white paper that follow, we have set the **key** to **bac001** on the system named **cibou** and set it to **bac999** on the system named **dir**.

Now assume that machine **dir** has locked the device. If on the machine **cibou**, we issue the following command:

```
sg_persist -r /dev/sg0
```

we will get the following output:

```
HP        Ultrium 4-SCSI    H61W
Peripheral device type: tape
PR generation=0x468, Reservation follows:
  Key=0xbac999
  scope: LU_SCOPE,  type: Exclusive Access
```

This indicates that the device is Exclusively locked by the machine that used the key **bac999**, which we know is our machine named **dir**.

3 Status Command

When running a job on machine **cibou**, if we run a job that wants a tape device that is locked by another machine **dir**, we might get output that looks similar to:

```
Jobs waiting to reserve a drive:
   3612 JobId=2 waiting because device "Drive-0" (/dev/nst0)
   is reserved by: 0xbac999.
====

Device status:
Autochanger "tape" with devices:
   "Drive-0" (/dev/nst0)
Device "Drive-0" (/dev/nst0) is mounted with:
    Volume:      TestVolume001
    Pool:        Default
    Media type:  tape
    Device is BLOCKED by another SD=0xbac999
    Total Bytes Read=64,512 Blocks Read=1 Bytes/block=64,512
    Positioned at File=0 Block=0
    shstore=1 registered=1 locked=0 blockedbySD=0xbac999
```

The important point here is that our job (JobId=2) is BLOCKED by another SD with the key **0xbac999** (i.e. **dir**). We know that shared storage is working and that our Storage daemon was able to register our key because of the last line of the output.

Once **dir** finishes with the drive and releases the lock, our job will continue and the status output will change to:

```
Jobs waiting to reserve a drive:
====

Device status:
Autochanger "tape" with devices:
   "Drive-0" (/dev/nst0)
Device "Drive-0" (/dev/nst0) is mounted with:
    Volume:      TestVolume001
    Pool:        Default
    Media type:  tape
    Total Bytes=48,900,096 Blocks=757 Bytes/block=64,597
    Positioned at File=1 Block=0
    shstore=1 registered=1 locked=1 blockedbySD=no
```

and now the job is no longer blocked and you can see that the last line of the output indicates that this storage daemon has the device locked (**locked=1**).

3.1 Problem Resolution

In general if any component of Bacula crashes, particularly the Storage daemon, any lock that it holds on any and all devices will be released. However, under unusual circumstances (a kill -9) or the OS crashes, the lock may remain.

In your Bacula binary directory (normally /opt/bacula/scripts) you can manually run the storage-ctl script with:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg0 query
```

If a lock is set on the device, you may forcibly remove it by using:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg0 clear
```

Please note that all no program should be actively accessing the device during a **clear** and that any lock that may exist will be forcibly be removed. The program that held the lock will not know the lock is removed and will blindly continue using the device without a lock, then if another program starts using device, it will acquire a new lock, but data corruption will likely occur because two programs will be writing to the same device at the same time (the first one with a broken lock, and the second with the new lock).

Prior to running the above **clear**, you might want to see who has what on the device, which can be done with the **regkeys** and **wholocked** commands:

In your Bacula binary directory (normally /opt/bacula/scripts) you can manually run the storage-ctl script with:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg0 regkeys
0xbac999
0xbac001

./storage-ctl /dev/sg0 wholocked
0xbac001
```

In this case, we see that **bac001** has the device locked, but both **bac001** and **bac999** have registration keys. Thus before trying to break the lock, it might be wise to find out who **bac999** is and why there is still a registration. Since after the **clear**, all locks and all registrations are lost.

3.2 Getting Information on the Locks

You can get a lot of information using the Bacula **status** command about which job is blocking a drive, as seen in examples in a previous section.

However, in some cases, it is easier to examine the overall picture by manually running the storage-ctl script which can normally be found in /opt/bacula/scripts.

Now if we examine from the stand point of the machine **dir** when there are no locks or registrations, using the query command, we get output as follows:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg4 query
  HP        Ultrium 4-SCSI    H61W
  Peripheral device type: tape
  PR generation=0x493
  No full status descriptors
```

If we now manually register a key (defined in **storage-ctl.conf**, then do a query, we will get the following output:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg4 register
./storage-ctl /dev/sg4 query
  HP        Ultrium 4-SCSI    H61W
  Peripheral device type: tape
  PR generation=0x494
    Key=0xbac999
      All target ports bit clear
      Relative port address: 0x1
      not reservation holder
      Transport Id of initiator:
        FCP-2 World Wide Name:
 00     21 00 00 1b 32 1f 43 65    !...2.Ce
```

Note that this time, it shows a key (bac999), the fact that there is no lock (no reservation holder), and the unique identification of the machine that registered the key (Transport Id of initiator).

Now let's actually lock the device:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg4 lock
./storage-ctl /dev/sg4 query
  HP        Ultrium 4-SCSI    H61W
  Peripheral device type: tape
  PR generation=0x494
    Key=0xbac999
      All target ports bit clear
      Relative port address: 0x1
      << Reservation holder >>
      scope: LU_SCOPE,  type: Exclusive Access
      Transport Id of initiator:
        FCP-2 World Wide Name:
 00     21 00 00 1b 32 1f 43 65    !...2.Ce
```

This time the query shows us that we (bac999) is the reservation holder (we have it locked) for Exclusive Access.

If we now switch to another machine, **cibou**, and issue the same query (on a different device address, but for the same physical device), we get:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg0 query
  HP        Ultrium 4-SCSI    H61W
  Peripheral device type: tape
  PR generation=0x494
    Key=0xbac999
      All target ports bit clear
      Relative port address: 0x1
      << Reservation holder >>
      scope: LU_SCOPE,  type: Exclusive Access
      Transport Id of initiator:
        FCP-2 World Wide Name:
 00     21 00 00 1b 32 1f 43 65    !...2.Ce
```

This is the same as what we see from machine **dir**, which means everything is consistent.

Now, let's register a key on the second machine, **cibou**:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg0 register
./storage-ctl /dev/sg0 query
  HP        Ultrium 4-SCSI    H61W
  Peripheral device type: tape
  PR generation=0x495
    Key=0xbac999
      All target ports bit clear
      Relative port address: 0x1
      << Reservation holder >>
      scope: LU_SCOPE,  type: Exclusive Access
      Transport Id of initiator:
        FCP-2 World Wide Name:
 00     21 00 00 1b 32 1f 43 65   !...2.Ce

    Key=0xbac001
      All target ports bit clear
      Relative port address: 0x1
      not reservation holder
      Transport Id of initiator:
        FCP-2 World Wide Name:
 00     21 01 00 1b 32 21 73 ee   !...2!s.
```

This time the query returns us information about two different "initiators" one, **dir**, which registered bac999 and holds the lock (Reservation holder), and the other **cibou**, which has a key (bac001) registered. Note that the two have different unique transport id names (the last line of each section that begins with 00).

If we try to obtain a lock on **cibou** we get the following error:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg0 trylock
persistent reserve out: scsi status: Reservation Conflict
PR out: command failed
```

Now if we remove the lock on **dir** (not shown), then acquire the lock on **cibou** and run another query, the output is as expected:

```
cd /opt/bacula/scripts
./storage-ctl /dev/sg0 lock
./storage-ctl /dev/sg0 query
  HP        Ultrium 4-SCSI    H61W
  Peripheral device type: tape
  PR generation=0x495
    Key=0xbac999
      All target ports bit clear
      Relative port address: 0x1
      not reservation holder
      Transport Id of initiator:
        FCP-2 World Wide Name:
 00     21 00 00 1b 32 1f 43 65    !...2.Ce

    Key=0xbac001
      All target ports bit clear
      Relative port address: 0x1
      << Reservation holder >>
      scope: LU_SCOPE,  type: Exclusive Access
      Transport Id of initiator:
        FCP-2 World Wide Name:
 00     21 01 00 1b 32 21 73 ee    !...2!s.
```

It shows the two machines (initiators), but this time **cibou** with key bac001 holds the Exclusive Access lock.

Index