Highly available iSCSI storage with DRBD and Pacemaker

Florian Haas

Highly available iSCSI storage with DRBD and Pacemaker

Florian Haas

Trademark notice

License information

Table of Contents

Chapter 1. Introduction

SCSI is an implementation of the SCSI protocol over IP. In iSCSI, servers (targets) provide storage services to clients (initiators) over IP based networks using SCSI semantics. On iSCSI initiator nodes, logical units (LUs) appear like any other SCSI block device, where they may be partitioned, used to hold filesystem storage, used to contain raw data, etc.

At the time of writing, several competing iSCSI target implementations exist on the Linux platform. Two of them are covered in this white paper:

- iSCSI Enterprise Target (IET). This was the first production-ready iSCSI target implementation for Linux. It uses a split, part-kernel, part-userland configuration interface that requires both a specific kernel module, and a running management daemon (ietd). The in-kernel implementation has not been merged into the mainline Linux kernel tree. Still, IET is included and fully supported as part of SUSE Linux Enterprise Server (versions 10 and 11), and Debian 5.0 (`lenny`). Although IET development had quieted down at one time, the project is currently quite active and has a small, but very productive core development team.
- Linux SCSI Target Framework (tgt). This aims to be a generic SCSI target framework for Linux, of which an iSCSI target is merely an implementation (or lower-level driver in tgt terms). The generic in-kernel framework that tgt uses is part of mainline Linux since release 2.6.20, and has been back-ported to the Red Hat Enterprise Linux 5 patched 2.6.18 kernel series. As such, it is fully supported on RHEL 5 and CentOS 5 from update 3 onwards; in previous RHEL releases it had been available as a technology preview only. tgt is also available in RHEL 6 and SLES 11.

This whitepaper describes a solution to use either of these target implementations in a highly available iSCSI target configuration.

Chapter 2. Installation

2.1. Installing SCSI Target Framework on Red Hat Enterprise Linux

The SCSI Target Framework (tgt) is a fully supported iSCSI implementation as of Red Hat Enterprise Linux 5 Update 3. To enable iSCSI target functionality on RHEL, you must install the `scsi-target-utils` package, using the following command:

```
yum install scsi-target-utils
```

If, however, you use the older `up2date` package manager instead of YUM, you must issue the following command instead:

```
up2date install scsi-target-utils
```

After installation, you should make sure that the tgtd service on system startup:

```
chkconfig tgtd on
```

2.2. Installing an iSCSI target implementation on SUSE Linux Enterprise Server

SUSE Linux Enterprise Server 11 comes with two iSCSI target implementations: iSCSI Enterprise Target (IET) and the SCSI Target Framework (tgt). You may select either for installation.

2.2.1. Installing IET on SLES 11

To install IET, issue the following commands:

```
zypper install iscsitarget iscsitarget-kmp-<flavor>
```

Replace `<flavor>` with your kernel flavor (usually `default`).

Then, make sure that the IET management daemon is started on system startup:

```
insserv ietd
```

2.2.2. Installing tgt on SLES 11

To install tgt, issue the following command:

```
zypper install tgt
```

tgt requires no additional kernel modules.

Then, to make sure tgt is started automatically on system startup, issue:

```
insserv tgtd
```

2.2.3. Installing iSCSI Enterprise Target on Debian GNU/Linux

iSCSI Enterprise Target (IET) is available as part of Debian GNU/Linux 5.0 (`lenny`), although the IET kernel modules are not distributed as part of the Debian stock kernel. Thus, you must

install two packages, one containing the IET administration utilities, and one containing the kernel modules:

```
aptitude install iscsitarget iscsitarget-modules-2.6-<arch>
```

Replace `<arch>` with your kernel architecture (usually `amd64`). This will install the IET module package to match the latest Debian 2.6 kernel for your architecture. If you are using a stock kernel other than the latest Debian kernel, issue the following command instead:

```
aptitude install iscsitarget iscsitarget-modules-`uname -r`
```

2.3. Installing the Pacemaker cluster manager on Red Hat Enterprise Linux

RHEL 5 packages are provided by the Pacemaker project and are available the project website. Pacemaker is best installed using the `yum` package manager. To be able to do so, you must first add the Pacemaker repository to your repository configuration:

- Download the repository file from http://www.clusterlabs.org/rpm/epel-5/clusterlabs.repo and install it into the `/etc/yum.repos.d` directory.
- Then, install Pacemaker (and dependencies) with `yum install pacemaker.x86_64`.

2.4. Installing Pacemaker on SLES 11

Installing Pacemaker on SLES 11 requires a valid SUSE Linux Enterprise High Availability Extension subscription.

Note

Enabling SLE 11 HAE is beyond the scope of this manual.

Once enabled, you may install Pacemaker with the following command:

```
zypper install pacemaker
```

Then, to make sure Pacemaker is started automatically on system startup, issue:

```
zypper install pacemaker
```

Chapter 3. Initial Configuration

This section describes the configuration of a highly available iSCSI Target and Logical Units (LU) in the context of the Pacemaker cluster manager.

3.1. Configuring a DRBD resource

First, it is necessary to configure a Pacemaker resource that manages a DRBD device. This resource will act as the Physical Volume of an LVM Volume Group to be created later. This example assumes that the LVM Volume Group is to be called `iscsivg01`, hence, the DRBD resource uses that same name.

```
global {
  usage-count yes;
}

common {
  protocol C;
  disk {
    on-io-error detach;
    fencing resource-only;
  }
  net {
    cram-hmac-alg sha1;
    shared-secret "a6a0680c40bca2439dbe48343ddddcf4";
  }
  syncer {
    rate 30M;
    al-extents 3389;
  }
  handlers {
    fence-peer "/usr/lib/drbd/crm-fence-peer.sh";
    after-resync-target "/usr/lib/drbd/crm-unfence-peer.sh";
    pri-on-incon-degr "echo b > /proc/sysrq-trigger";
  }
}

resource iscsivg01 {
  device /dev/drbd1;
  disk /dev/sda1;
  meta-disk internal;

  on alice {
    address 10.0.42.1:7790;
  }
  on bob {
    address 10.0.42.2:7790;
  }
}
```

3.2. LVM Configuration

It is necessary instruct LVM to read Physical Volume signatures from DRBD devices, rather than the underlying backing block devices. The easiest approach for doing this is to mask the underlying block device from the list of devices LVM scans for PV signatures.

To do so, open the LVM configuration file (`/etc/lvm/lvm.conf`) and edit the following entries:

```
filter = [ "r|/dev/sdb.*|" ]
```

In addition, you should disable the LVM cache by setting:

```
write_cache_state = 0
```

After disabling the LVM cache, make sure you remove any stale cache entries by deleting `/etc/lvm/cache/.cache`.

You must repeat the above steps on the peer node.

Now, to be able to create an LVM Volume Group, it is first necessary to initialize the DRBD resource as an LVM Physical Volume. To do so, after you have initiated the initial synchronization of your DRBD resource, issue the following commands on the node where your resource is currently in the Primary role:

```
pvcreate /dev/drbd/by-res/iscsivg01
```

Now, create an LVM Volume Group that includes this PV:

```
vgcreate iscsivg01 /dev/drbd/by-res/iscsivg01
```

3.3. Initial Pacemaker configuration steps

Note

While this manual covers the configuration of the Pacemaker cluster manager, the configuration of the cluster stack that Pacemaker uses is beyond the scope of this manual. Please see Initial Configuration [http://clusterlabs.org/wiki/Initial_Configuration] (from the ClusterLabs Wiki) for details on bootstrapping a Pacemaker cluster configuration.

In a highly available iSCSI target configuration that involves a 2-node cluster, you should

- Disable STONITH;
- Set Pacemaker's no quorum policy to ignore loss of quorum;
- Set the default resource stickiness to 200.

To do so, issue the following commands from the CRM shell:

```
crm(live)# configure
crm(live)configure# property stonith-enabled="false"
crm(live)configure# property no-quorum-policy="ignore"
crm(live)configure# property default-resource-stickiness="200"
crm(live)configure# commit
```

3.4. Creating an Active/Passive iSCSI configuration

An active/passive iSCSI Target consists of the following cluster resources:

- A DRBD resource to replicate data, which is switched from and to the Primary and Secondary roles as deemed necessary by the cluster resource manager;

- An LVM Volume Group, which is made available on whichever node currently holds the DRBD resource in the Primary role;
- A virtual, floating cluster IP address, allowing initiators to connect to the target no matter which physical node it is running on;
- The iSCSI Target itself;
- One or more iSCSI Logical Units (LUs), each corresponding to a Logical Volume in the LVM Volume Group.

The following Pacemaker configuration example assumes that `10.9.9.180` is the virtual IP address to use for a target with the iSCSI Qualified Name (IQN) `iqn.2001-04.com.example:storage.example.iscsivg01`.

The target is to contain two Logical Units with LUNs 1 and 2, mapping to Logical Volumes named `lun1` and `lun2`, respectively.

To start configuring these resources, open the crm shell as `root` (or any non-`root` user that is part of the `haclient` group), and issue the following commands:

```
crm(live)# configure
crm(live)configure# primitive p_drbd_iscsivg01 \
  ocf:linbit:drbd \
    params drbd_resource="iscsivg01" \
    op monitor interval="10"
crm(live)configure# ms ms_drbd_iscsivg01 p_drbd_iscsivg01 \
  meta master-max="1" master-node-max="1" clone-max="2" \
    clone-node-max="1" notify="true"
```

This will create a Master/Slave resource corresponding to the DRBD resource `iscsivg01`.

```
crm(live)configure# primitive p_ip_alicebob01 \
  ocf:heartbeat:IPaddr2 \
    params ip="10.9.9.180" cidr_netmask="24" \
    op monitor interval="10s"
crm(live)configure# primitive p_lvm_iscsivg01 \
  ocf:heartbeat:LVM \
    params volgrpname="iscsivg01" \
    op monitor interval="30s"
crm(live)configure# primitive p_target_iscsivg01 \
  ocf:heartbeat:iSCSITarget \
    params iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      tid="1" \
    op monitor interval="10s"
```

Note

You *must* specify the numeric target id (`tid`) if you are using the tgt implementation. For IET, setting this parameter is optional.

Thus, we have configured a highly available IP address, Volume Group, and iSCSI Target. We can now add Logical Units:

```
crm(live)configure# primitive p_lu_iscsivg01_lun1 \
  ocf:heartbeat:iSCSILogicalUnit \
    params target_iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      lun="1" path="/dev/iscsivg01/lun1" \
    op monitor interval="10s"
crm(live)configure# primitive p_lu_iscsivg01_lun2 \
```

```
  ocf:heartbeat:iSCSILogicalUnit \
    params target_iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      lun="2" path="/dev/iscsivg01/lun2" \
op monitor interval="10s"
```

Now to tie all of this together, we must first create a resource group from the resources associated with our iSCSI Target:

```
crm(live)configure# group rg_iscsivg01 \
  p_lvm_iscsivg01 \
  p_target_iscsivg01 p_lu_iscsivg01_lun1 p_lu_iscsivg01_lun2 \
  p_ip_alicebob01
```

This group, by Pacemaker default, is *ordered* and *colocated*, which means that the resources contained therein will always run on the same physical node, will be started in the order as specified, and stopped in reverse order.

Finally, we have to make sure that this resource group is also started on the node where DRBD is in the Primary role:

```
crm(live)configure# order o_drbd_before_iscsivg01 \
  inf: ms_drbd_iscsivg01:promote rg_iscsivg01:start
crm(live)configure# colocation c_iscsivg01_on_drbd \
  inf: rg_iscsivg01 ms_drbd_iscsivg01:Master
```

Now, our configuration is complete, and may be activated:

```
crm(live)configure# commit
```

3.5. Creating an Active/Active iSCSI configuration

An active/active iSCSI Target consists of the following cluster resources:

Two DRBD resources to replicate data, which are switched from and to the Primary and Secondary roles as deemed necessary by the cluster resource manager;

- Two LVM Volume Groups, which are made available on whichever node currently holds the corresponding DRBD resource in the Primary role;
- Two virtual, floating cluster IP addresses, allowing initiators to connect to the target no matter which physical node it is running on;
- The iSCSI Targets themselves;
- One or more iSCSI Logical Units (LUs), each corresponding to a Logical Volume in one of the two LVM Volume Groups

`10.9.9.180` and `10.9.9.181` are the virtual IP addresses to use for two targets with the iSCSI Qualified Names (IQN) `iqn.2001-04.com.example:storage.example.iscsivg01` and `iqn.2001-04.com.example:storage.example.iscsivg02`, respectively.

The targets are to contain two Logical Units with LUNs 1 and 2, mapping to Logical Volumes named `lun1` and `lun2` in each Volume Group, respectively.

To start configuring these resources, open the `crm` shell as `root` (or any non-`root` user that is part of the `haclient` group), and issue the following commands:

```
crm(live)# configure
```

```
crm(live)configure# primitive p_drbd_iscsivg01 \
  ocf:linbit:drbd \
    params drbd_resource="iscsivg01" \
    op monitor interval="10s"
crm(live)configure# ms ms_drbd_iscsivg01 p_drbd_iscsivg01 \
  meta master-max="1" master-node-max="1" clone-max="2" \
    clone-node-max="1" notify="true"
crm(live)configure# primitive p_drbd_iscsivg02 \
  ocf:linbit:drbd \
    params drbd_resource="iscsivg02" \
  op monitor interval="10s"
crm(live)configure# ms ms_drbd_iscsivg01 p_drbd_iscsivg02 \
  meta clone-max="2" notify="true"
```

This will create Master/Slave resources corresponding to the DRBD resources `iscsivg01` and `iscsivg02`.

```
crm(live)configure# primitive p_ip_alicebob01 \
  ocf:heartbeat:IPaddr2 \
    params ip="10.9.9.180" cidr_netmask="24" \
    op monitor interval="10s"
crm(live)configure# primitive p_ip_alicebob02 \
  ocf:heartbeat:IPaddr2 \
    params ip="10.9.9.181" cidr_netmask="24" \
    op monitor interval="10s"
crm(live)configure# primitive p_lvm_iscsivg01 \
  ocf:heartbeat:LVM \
    params volgrpname="iscsivg01" \
    op monitor interval="30s"
crm(live)configure# primitive p_lvm_iscsivg01 \
  ocf:heartbeat:LVM \
    params volgrpname="iscsivg02" \
    op monitor interval="30s"
crm(live)configure# primitive p_target_iscsivg01 \
  ocf:heartbeat:iSCSITarget \
    params iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      tid="1" \
    op monitor interval="10s"
crm(live)configure# primitive p_target_iscsivg02 \
  ocf:heartbeat:iSCSITarget \
    params iqn="iqn.2001-04.com.example:storage.example.iscsivg02" \
      tid="2" \
    op monitor interval="10s
```

Note

You *must* specify the numeric target id (`tid`) if you are using the tgt implementation. For IET, setting this parameter is optional.

Thus, we have configured a highly available IP address, Volume Group, and iSCSI Target. We can now add Logical Units:

```
crm(live)configure# primitive p_lu_iscsivg01_lun1 \
  ocf:heartbeat:iSCSILogicalUnit \
    params target_iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      lun="1" path="/dev/iscsivg01/lun1" \
    op monitor interval="10s"
crm(live)configure# primitive p_lu_iscsivg01_lun2 \
  ocf:heartbeat:iSCSILogicalUnit \
```

```
    params target_iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      lun="2" path="/dev/iscsivg0
crm(live)configure# primitive p_lu_iscsivg02_lun1 \
  ocf:heartbeat:iSCSILogicalUnit \
    params target_iqn="iqn.2001-04.com.example:storage.example.iscsivg02" \
      lun="1" path="/dev/iscsivg02/lun1" \
    op monitor interval="10s"
crm(live)configure# primitive p_lu_iscsivg02_lun2 \
  ocf:heartbeat:iSCSILogicalUnit \
    params target_iqn="iqn.2001-04.com.example:storage.example.iscsivg02" \
      lun="2" path="/dev/iscsivg02/lun2" \
    op monitor interval="10s"
```

Now to tie all of this together, we must first create resource groups from the resources associated with our iSCSI Targets:

```
crm(live)configure# group rg_iscsivg01 \
  p_lvm_iscsivg01 \
  p_target_iscsivg01 p_lu_iscsivg01_lun1 p_lu_iscsivg01_lun2 \
  p_ip_alicebob01
crm(live)configure# group rg_iscsivg02 \
  p_lvm_iscsivg02 \
  p_target_iscsivg02 p_lu_iscsivg02_lun1 p_lu_iscsivg02_lun2 \
  p_ip_alicebob02
```

These groups, by Pacemaker default, are *ordered* and *colocated*, which means that the resources contained therein will always run on the same physical node, will be started in the order as specified, and stopped in reverse order.

We now have to make sure that this resource group is also started on the node where DRBD is in the Primary role:

```
crm(live)configure# order o_drbd_before_iscsivg01 \
  inf: ms_drbd_iscsivg01:promote rg_iscsivg01:start
crm(live)configure# colocation c_iscsivg01_on_drbd \
  inf: rg_iscsivg01 ms_drbd_iscsivg01:Master
crm(live)configure# order o_drbd_before_iscsivg02 \
  inf: ms_drbd_iscsivg02:promote rg_iscsivg02:start
crm(live)configure# colocation c_iscsivg01_on_drbd \
  inf: rg_iscsivg01 ms_drbd_iscsivg01:Master
crm(live)configure# colocation c_iscsivg02_on_drbd \
  inf: rg_iscsivg02 ms_drbd_iscsivg02:Master
```

Now, our configuration is complete, and may be activated:

```
crm(live)configure# commit
```

Chapter 4. Security Considerations

Access to iSCSI targets may be restricted in one of several fashions:

- By initiator address. Access to iSCSI targets may be restricted to specific initiators, identified by their IP addresses or iSCSI Qualified Name (IQN).
- By initiator credentials. iSCSI Targets may be protected with a username and password. Initiators are then forced to login with those credentials using the Challenge-Response Authentication Protocol (CHAP). This protocol does not transmit passwords in the clear, instead it uses password hashes in a challenge-reponse exchange.
- Combined approach. The two above approaches may be combined, such that targets can be connected to only from specific initiator IP addresses, where the initiators have to additionally pass CHAP authentication.

4.1. Restricting target access by initiator address

To restrict access to a target to one or more initiator addresses, use the `initiators` parameter supported by the iSCSI Target Pacemaker resource agent:

```
crm(live)configure# edit p_target_iscsivg01
```

This will bring up a text editor containing the current configuration parameters for this resource. Edit the resource to include the allowed_initiators parameter, containing a space- separated list of initiator IP addresses allowed to connect to this target. In the example below, access is granted to initiator IP addresses `10.9.9.60` and `10.9.9.170`.

Note

This approach is valid when using iSCSI Enterprise Target (IET) or SCSI Target Framework (tgt) as the underlying iSCSI target implementation.

```
primitive p_target_iscsivg01 \
  ocf:heartbeat:iSCSITarget \
    params iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      allowed_initiators="10.9.9.60 10.9.9.170" \
    op monitor interval="10s"
```

When you close the editor, the configuration changes are inserted into the CIB configuration. To commit these changes, as usual, enter the following command:

```
crm(live)configure# commit
```

After you commit the changes, the target will immediately reconfigure and enable the access restrictions.

Caution

If initiators are connected to the target at the time of re-configuration, and one of the connected initiators is not included in the `initiators` list for this resource, then those initiators will lose access to the target, possibly resulting in disruption on the initiator node. Use with care.

4.2. Restricting target access by using CHAP credentials

To create a username and password which initiators must use to log in to an iSCSI target, use the `username` and `password` parameters supported by the iSCSITarget Pacemaker resource agent:

```
crm(live)configure# edit p_target_iscsivg01
```

This will bring up a text editor containing the current configuration parameters for this resource. Edit the resource to include the `username` and `password` parameters, containing the username and password to access the target. In the example below, access is granted to initiators using a username of `iscsi` and password of `zi1caighaiTo`.

```
primitive p_target_iscsivg01 \
  ocf:heartbeat:iSCSITarget \
    params iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      incoming_username="iscsi" \
      incoming_password="zi1caighaiTo" \
    op monitor interval="10s"
```

Note

Some iSCSI initiator implementations require that the CHAP password is at least 12 bytes long.

When you close the editor, the configuration changes are inserted into the CIB configuration. To commit these changes, as usual, enter the following command:

```
crm(live)configure# commit
```

After you commit the changes, the target will immediately reconfigure and enable the access restrictions.

Caution

If initiators are connected to the target at the time of target re-configuration, they will invariably lose target access until re-configured with matching credentials themselves. As this is likely to cause disruption on the initiator node, you should change usernames and/or passwords only on targets with no initiator activity.

Chapter 5. Setting configuration parameters

This section outlines some of the configuration parameters one may want to set in a highly available iSCSI Target configuration.

5.1. Per-target configuration parameters

You may set configuration parameters at the iSCSI target level by using the `additional_parameters` instance attribute defined for the iSCSITarget resource agent.

To set, for example, the `DefaultTime2Retain` and `DefaultTime2Wait` session parameters to 60 and 5 seconds, respectively, modify your target resource as follows:

```
crm(live)configure# edit p_target_iscsivg01
```

```
primitive p_target_iscsivg01 \
  ocf:heartbeat:iSCSITarget \
    params iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      additional_parameters="DefaultTime2Retain=60 DefaultTime2Wait=5"
    op monitor interval="10s"
```

```
crm(live)configure# commit
```

5.2. Per-LU configuration parameters

5.2.1. SCSI ID and serial number

For some applications, smooth uninterrupted failover requires that the SCSI ID associated with a Logical Unit is identical regardless of which node currently exports the LU. The same applies to the SCSI Serial Number. Examples of such applications are the device-mapper multipath target (`dm-multipath`) on Linux, and the Microsoft iSCSI initiator on Windows.

The `iSCSILogicalUnit` resource agent attempts to select sensible, consistent values for these fields as appropriate for the underlying iSCSI implementation. Still, you may prefer to set the SCSI ID and/or serial number explicitly as part of the LU configuration.

To set a SCSI ID or serial number for an exported LU, edit the `iSCSILogicalUnit` resource to include the `scsi_id` or `scsi_sn` parameter (or both):

```
crm(live)configure# edit p_lu_lun1
```

```
primitive p_lu_lun1 \
  ocf:heartbeat:iSCSILogicalUnit \
    params
    target_iqn="iqn.2001-04.com.example:storage.example.iscsivg01"
      lun="1" path="/dev/iscsivg01/lun1" \
      scsi_id="iscsivg01.lun1" scsi_sn="4711" \
    op monitor interval="10s"
```

```
crm(live)configure# commit
```

5.2.2. Vendor ID and Product ID

Two other SCSI Vital Product Data (VPD) fields that you may wish to set explicitly are the SCSI Vendor ID and Product ID fields. To do so, add the resource parameters `vendor_id` and/or `product_id` to your LU configuration:

```
crm(live)configure# edit p_lu_lun1

primitive p_lu_lun1 \
  ocf:heartbeat:iSCSILogicalUnit \
    params target_iqn="iqn.2001-04.com.example:storage.example.iscsivg01" \
      lun="1" path="/dev/iscsivg01/lun1" \
      vendor_id="STGT" scsi_id="iscsivg01.lun1" scsi_sn="4711" \
    op monitor interval="10s"

crm(live)configure# commit
```

Note

Interestingly, STGT uses a default vendor ID of IET. If you are using the tgt target implementation, you may want to set the vendor ID to a non-default value as shown in the example, to avoid confusion.

Chapter 6. Using highly available iSCSI Targets

This section describes some common usage scenarios for highly available iSCSI Targets.

6.1. Connecting to iSCSI targets from Linux

The recommended way of connecting to a highly available iSCSI Target from Linux is to use the initiator delivered by the Open iSCSI project (http://www.open-iscsi.org).

After installing the Open iSCSI administration utilities, it is first necessary to start the iSCSI initiatior daemon, `iscsid`. To do so, issue one of the following commands (depending on your distribution):

```
/etc/init.d/open-iscsi start
rcopen-iscsi start
service open-iscsi start
```

Now you may start a discovery session on your target portal. Assuming your cluster IP address for the target is 10.9.9.180, you may do so using the following command:

```
iscsiadm -m discovery -p 10.9.9.180 -t sendtargets
```

The output from this command should include the names of all targets you have configured.

```
10.9.9.180:3260,1 iqn.2001-04.com.example:storage.example.iscsivg01
```

Note

If a configured target does not appear in this list, check whether your initiator has been blocked from accessing this target via an initiator restriction (see Section 4.1, “Restricting target access by initiator address” [10]).

Then, if you have configured your iSCSI Target to require authentication (see Section 4.2, “Restricting target access by using CHAP credentials” [11]), you must set a username and password for any target you wish to connect to. To do so, issue the following commands:

```
iscsiadm -m node -p 10.9.9.180 \
  -T iqn.2001-04.com.example:storage.example.iscsivg01 \
  --op update \
  -n node.session.auth.authmethod -v CHAP
iscsiadm -m node -p 10.9.9.180 \
  -T iqn.2001-04.com.example:storage.example.iscsivg01 \
  --op update \
  -n node.session.auth.username -v iscsi
iscsiadm -m node -p 10.9.9.180 \
  -T iqn.2001-04.com.example:storage.example.iscsivg01 \
  --op update \
  -n node.session.auth.password -v zi1caighaiTo
```

Finally, you may log in to the target, which will make all LUs configured therein available as local SCSI devices:

```
iscsiadm -m node -p 10.9.9.180 \
  -T iqn.2001-04.com.example:storage.example.iscsivg01 \
  --login
```

6.2. Connecting to iSCSI targets from Microsoft Windows

On Microsoft Windows, iSCSI Target connections are managed by the Microsoft iSCSI Initiator Service, which is available free of charge from Microsoft and my be installed on Microsoft Windows Server 2003 and 2008.

Important

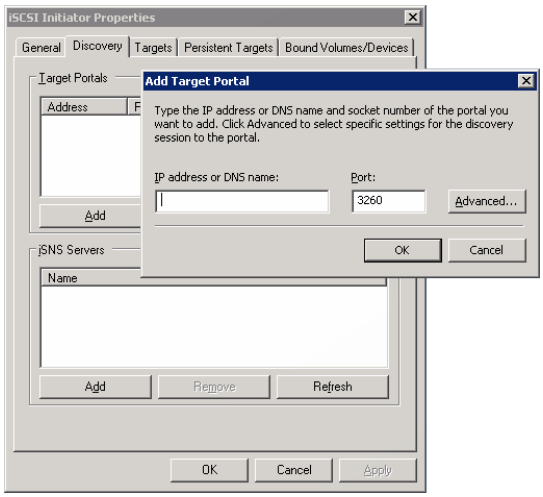
Smooth, uninterrupted target failover in conjunction with the Microsoft iSCSI Initiator is guaranteed only if the Logical Units' SCSI IDs and serial numbers are persistent across the failover process. Refer to the Section 5.2.1, “SCSI ID and serial number” [12] for considerations on consistent SCSI IDs and SNs.

6.2.1. Configuring Microsoft iSCSI Initiator using the Control Panel applet

To configure access to an iSCSI target from Microsoft Windows, open the Control Panel item `Microsoft iSCSI Initiator`. First, click on the `Discovery` tab, then under `Target Portals` click Add to add a connection to a target portal.

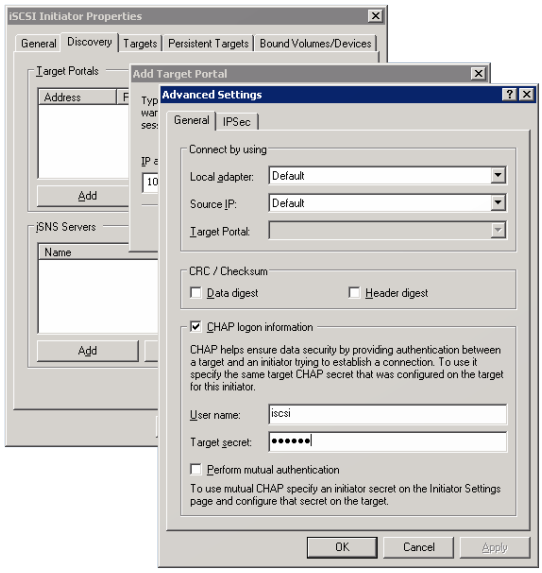
In the `IP address or DNS name` field, enter the floating cluster IP address of your configured iSCSI target as the target IP address. You may of course also use a host name if it resolves to the cluster IP address.

Figure 6.1. Configuring target IP address on Windows



If your target is protected by CHAP authentication, click `Advanced...` to open the advanced settings dialog. Check the CHAP `logon information` checkbox. Enter the username and password configured for target authentication.

Figure 6.2. Configuring CHAP authentication on Windows

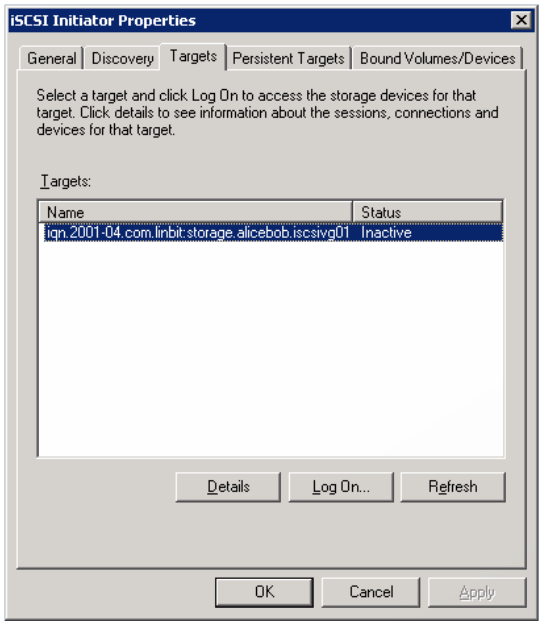


Note

Be sure to leave the `Perform mutual authentication` checkbox unchecked.

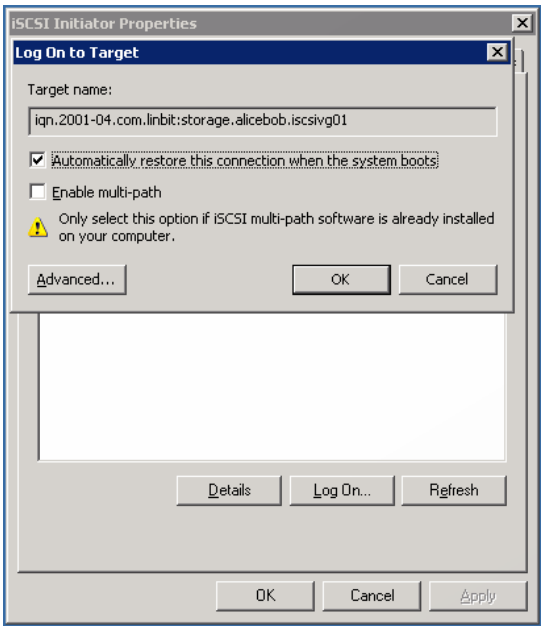
Next, select the Targets tab. It should now list any targets visible to the initiatior under the configured portal address. In the example below, there is one target available. Since the target has not been logged into, its status is listed as `Inactive`:

Figure 6.3. Discovered iSCSI target on Windows



You may now click `Log On...` to log on to the target:

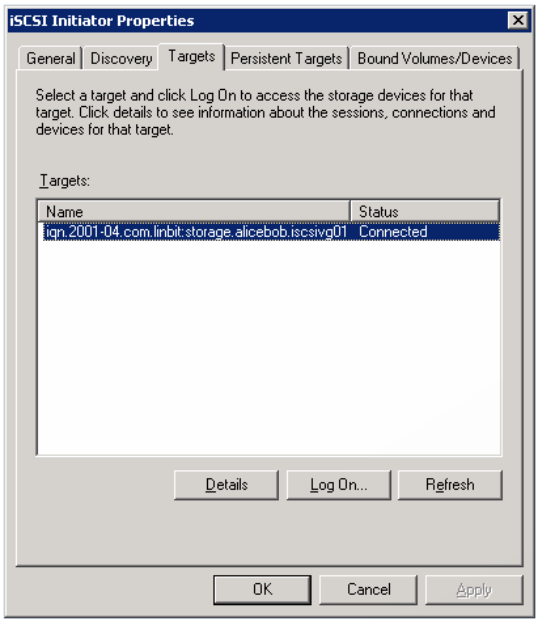
Figure 6.4. iSCSI target logon dialog on Windows



Be sure to check box the labeled `Automatically restore this connection when the system boots`, to ensure that the connection to the configured target portal is automatically restored on system boot.

When you have configured your initiator correctly, the target should be listed as `Connected` in the Targets list:

Figure 6.5. Connected target on Windows



6.2.2. Configuring Microsoft iSCSI Initiator using `iscsicli`

Microsoft iSCSI Initiator comes with a command line utility named `iscsicli`, which may also be used to configure access to iSCSI portals and targets.

To connect to a target portal, use the following command:

```
C:\>iscsicli QAddTargetPortal 10.9.9.180
Microsoft iSCSI Initiator version 2.0 Build 3825

The operation completed successfully.
```

You should now be able to retrieve information associated with the newly added target:

```
C:\>iscsicli ListTargetPortals
Microsoft iSCSI Initiator version 2.0 Build 3825

Total of 1 portals are persisted:

    Address and Socket   : 10.9.9.180 3260
    Symbolic Name        :
    Initiator Name       :
    Port Number          : <Any Port>
    Security Flags       : 0x0
    Version              : 0
    Information Specified: 0x0
    Login Flags          : 0x0

The operation completed successfully.
```

Next, list the targets accessible via this target portal:

```
C:\>iscsicli ListTargets
Microsoft iSCSI Initiator version 2.0 Build 3825

Targets List:
    iqn.2001-04.com.linbit:storage.alicebob.iscsivg01
The operation completed successfully.
```

You may now add the newly discovered target to your configuration, as a persistent target. `iscsicli` requires that you enter the same parameters both for target login, and for making the target persistent:

```
C:\>iscsicli
PersistentLoginTarget iqn.2001-04.com.linbit:storage.alicebob.iscsivg01
T * * * * * * * * * * * * * * * 0
Microsoft iSCSI Initiator version 2.0 Build 3825

LoginTarget to iqn.2001-04.com.linbit:storage.alicebob.iscsivg01
on <no init instance> to <no portal>/0
The operation completed successfully.

C:\>iscsicli
LoginTarget iqn.2001-04.com.linbit:storage.alicebob.iscsivg01
T * * * * * * * * * * * * * * * 0
Microsoft iSCSI Initiator version 2.0 Build 3825
LoginTarget to iqn.2001-04.com.linbit:storage.alicebob.iscsivg01 on
<no init instance> to <no portal>/0
Session Id is 0xfffffadfe5f70018-0x4000013700000010
Connection Id is 0xfffffadfe5f70018-0xf
The operation completed successfully.
```

Note

If your target is configured with CHAP authentication, replace the trailing `* * 0` with `<username> <password> 1`.

Finally, import Logical Units into your local disk configuration:

```
C:\>iscsicli bindpersistentvolumes
Microsoft iSCSI Initiator version 2.0 Build 3825

The operation completed successfully.
```

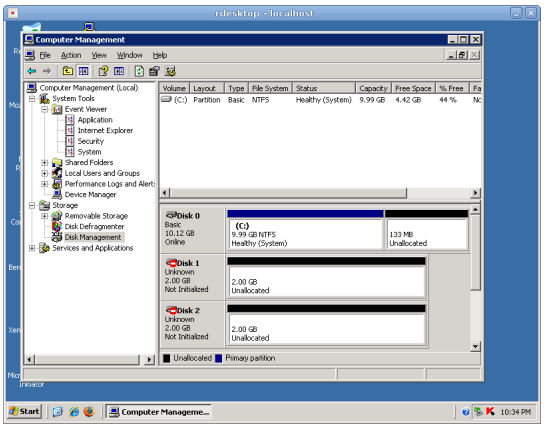
6.2.3. Initializing iSCSI disks on Windows

Note

When you connect a Windows host to a target that uses tgt for the first time, the Windows Plug and Play manager displays a new, unknown storage controller device. This is the virtual "controller" LUN 0 that tgt exposes. No driver for this device exists, nor is one necessary. Simply click through the New Device Wizard and choose not to install any driver. This consideration does not apply if your iSCSI target cluster uses an implementation other than tgt.

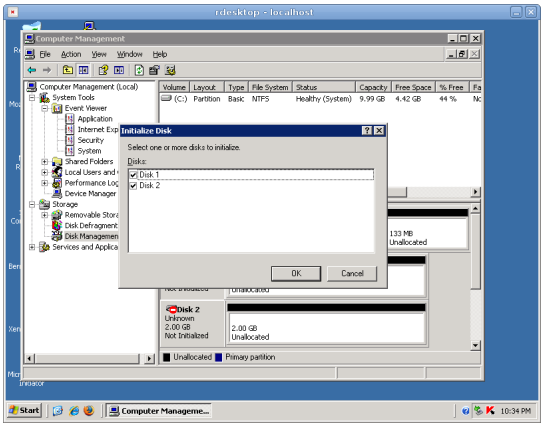
After you have added a target's Logical Units to your computer's configuration as local disks, open the `Computer Management` console from the Administrative Tools menu and select `Logical Disk Manager`. Your new disk should now be listed among the local drives available:

Figure 6.6. New iSCSI disks in Windows Logical Disk Manager



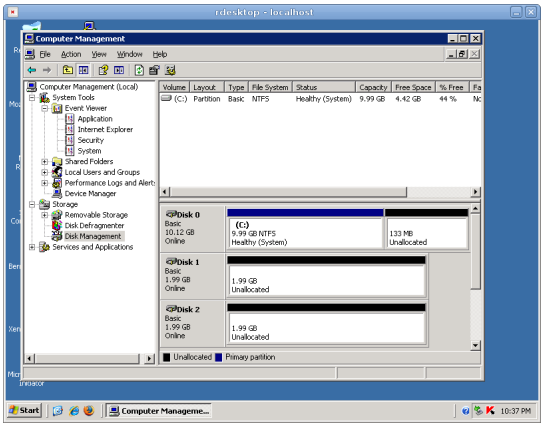
Right-click on one of the new disks labeled `Unknown` and `Not Initialized`, and select `Initialize Disk`. You will be prompted to select one or more disks to initialize:

Figure 6.7. Initializing iSCSI disks in Windows



After drives are initialized, their status changes to `Basic` and `Online`. You may now format the drive, assign a drive letter, or mount point, just as you would with a local disk.

Figure 6.8. iSCSI disks in Windows after initialization



Caution

Do not convert the iSCSI disk to a Dynamic Disk. This is not supported on Windows; iSCSI connected drives should always remain configured as Basic Disks.

Chapter 7. Feedback

Any questions or comments about this document are highly appreciated and much encouraged. Please contact the author(s) directly; contact email addresses are listed on the title page.

For a public discussion about the concepts mentioned in this white paper, you are invited to subscribe and post to the drbd-user mailing list. Please see http://lists.linbit.com/listinfo/ drbd-user for details.