dm-zero

Device-Mapper's "zero" target provides a block-device that always returns zero'd data on reads and silently drops writes. This is similar behavior to /dev/zero, but as a block-device instead of a character-device.

Dm-zero has no target-specific parameters.

One very interesting use of dm-zero is for creating "sparse" devices in conjunction with dm-snapshot. A sparse device reports a device-size larger than the amount of actual storage space available for that device. A user can write data anywhere within the sparse device and read it back like a normal device. Reads to previously unwritten areas will return a zero'd buffer. When enough data has been written to fill up the actual storage space, the sparse device is deactivated. This can be very useful for testing device and filesystem limitations.

To create a sparse device, start by creating a dm-zero device that's the desired size of the sparse device. For this example, we'll assume a 10TB sparse device:

```
TEN_TERABYTES=`expr 10 \* 1024 \* 1024 \* 1024 \* 2`   # 10 TB in sectors
echo "0 $TEN_TERABYTES zero" | dmsetup create zero1
```

Then create a snapshot of the zero device, using any available block-device as the COW device. The size of the COW device will determine the amount of real space available to the sparse device. For this example, we'll assume /dev/sdb1 is an available 10GB partition:

```
echo "0 $TEN_TERABYTES snapshot /dev/mapper/zero1 /dev/sdb1 p 128" | \
   dmsetup create sparse1
```

This will create a 10TB sparse device called /dev/mapper/sparse1 that has 10GB of actual storage space available. If more than 10GB of data is written to this device, it will start returning I/O errors.