SAN ASSIGNMENT

1)AKSHAY UBALE : 01FB16ECS048
2)ALTAMASH SAMEER : 01FB16ECS051
3)SHADAN ALAM KAIFEE : 01FB16ECS346

iscsi

1) INTRODUCTION
2) HOW IT WORKS
3) INSTALLATIION
4) DIFFICULTIES
5) ADVANTAGES
6) DISADVANTAGES

Introduction

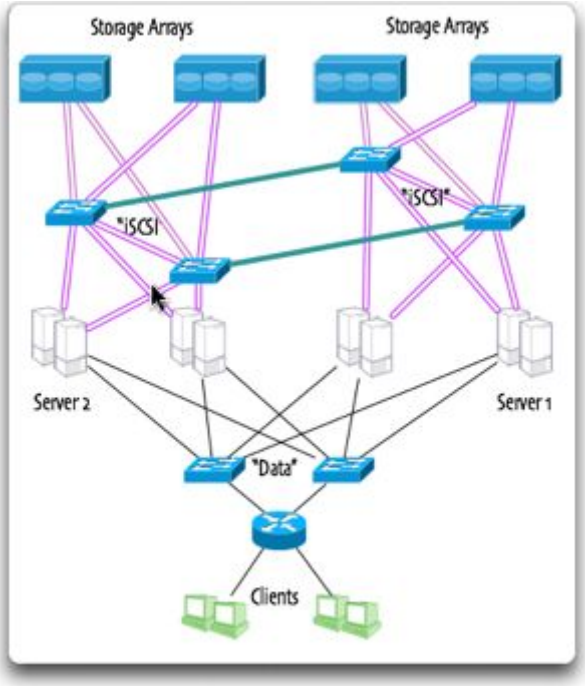
iSCSI is Internet SCSI (Small Computer System Interface), an Internet Protocol (IP)-based storage networking
standard for linking data storage facilities,and used to send block storage from storage arrays or devices to
client computers that aren't directly connected to those devices.It is a transport layer pratotocol that describes
how SCSI packets should be transported over a TCP/IP network.
iSCSI, which stands for Internet Small Computer System Interface, works on top of the Transport Control

Protocol (TCP) and allows the SCSI command to be sent end-to-end over local-area networks (LANs), wide-
area networks (WANs) or the Internet. IBM developed iSCSI as a proof of concept in 1998, and presented the

first draft of the iSCSI standard to the Internet Engineering Task Force (IETF) in 2000. The protocol was ratified
in 2003.

How it works

An iSCSI storage area network consists of iSCSI targets on storage array controllers and iSCSI initiators on storage clients. These targets and initiators are used by the iSCSI protocol to connect storage to clients and are represented by a unique name called the iSCSI Qualified Name or IQN. Computer operating systems include or can be installed with iSCSI clients and initiators. While some storage arrays are designed to only provide storage over iSCSI, many storage devices come equipped with iSCSI targets in addition to other common protocols. iSCSI works by transporting block-level data between an iSCSI Initiator on a server and an iSCSI target on a storage device.The iSCSI protocol encpsulates SCSI commands and assembles the data in packets and also encryption procedures for the TCP/IP layer. Packets are sent over the network using a point-to-point connection. Upon arrival, the iSCSI protocol disassembles the packets and decrypted, separating the SCSI
commands so the operating system (OS) will see the storage as a local SCSI device that can be formatted as
usual. Today, some of iSCSI's popularity in small to midsize businesses (SMBs) has to do with the way server virtualization makes use of storage pools. In a virtualized environment, the storage pool is accessible to all the hosts within the cluster and the cluster nodes nodes communicate with the storage pool over the network through the use of the iSCSI protocol.



The iSCSI protocol describes how blocks of data are to be transmitted between "initiators" on an Ethernet network and their storage "targets". The following section further describes the role of initiators, targets and the iSCSI components typically required for an initial IP SAN implementation. An initiator is typically a server hosting an application, where the application makes periodic requests for data to a related storage device. Initiators are also referred to as servers or host computers. The iSCSI device driver that resides on the server may also be called an initiator. Initiators "initiate" (or begin) iSCSI data transport transactions by making an application request to send/receive data either to or from one or more storage devices. The application request is immediately converted into SCSI commands, then encapsulated into iSCSI where a packet and header are added for transport via TCP/IP over either the Internet or traditional Ethernet networks.

Targets are one or more storage devices that reside on the network. Targets receive iSCSI commands from various initiators (or servers) on the network. On the target's side, these commands are then broken down into their original SCSI format to allow block data to be transported between the initiator and the storage device. The target will respond to a server's data request by sending SCSI commands back to that server. These commands are again encapsulated via iSCSI for transport over the Ethernet. Targets can be any type of storage device, such as a storage array that is part of a larger IP-based storage area network, or IP SAN. They could also be a separate tape library residing on either the SAN or elsewhere on the network. A basic Ethernet port or Host Bus Adapter is all that's needed to connect iSCSI targets and initiators to a network. Special HBAs can be installed in computer servers that offload iSCSI transactions from the CPU to the adapter for greater efficiency or that support multiple Ethernet storage protocols such as both iSCSI and Fiber Channel over Ethernet. An iSCSI LUN is created on the storage device and assigned to an initiator group or client definition at this point assuming the target and initiator are on the same IP network, the client may be able to automatically
discover the target. Once the initiator is connected to the target, the iSCSI LUN at that target IQN is available
for use by the client. iSCSI LUNs are configured and used the same as any other block storage by the client operating system. The iSCSI protocol supports many features to improve security and performance.

Installation

->Common used
definitions:
IQN (iSCSI Qualified

Name) — these are the names used to identify both the target and the initiator.
Backend Storage — defines storage device an iSCSI target is providing access to.
Target — service on an iSCSI server that provides access to the backend storage devices.
Initiator — iSCSI client that connects to the target.
ACL — Access Control List that lists iSCSI clients to be granted access to the storage device.
LUN (logical unit number) — backend storage devices (disks, partitions, logical volumes, files, or tape drives) shared through the target.
Portal — an IP address and port used by the target or the initatior to establish a connection.
TPG (target portal group) — group of IP addresses and TCP ports to which specific iSCSI target will listen.
Login — Authentication that gives an initiator access to LUNs on the target.
->Steps(For UBUNTU):
Configure iSCSI target(Using targetcli)
Install the targetcli, administration tool for viewing and managing the target configurations.
apt -y install targetcli-fb
On the targetcli administration tool is installed, proceed to configure the iSCSI target.

Create the backend storage device (backstore)

To configure the backend storage launch the admin tool, targetcli. When run, it launches an interactive
prompt. # create a
directory
root@dlp:~# mkdir /var/lib/iscsi_disks
enter the admin console
root@dlp:~# targetcli
targetcli shell version 2.1.fb43
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.
/> cd backstores/fileio
create a disk-image with the name "disk01" on /var/lib/iscsi_disks/disk01.img with 10G
/backstores/fileio> create disk01 /var/lib/iscsi_disks/disk01.img 10G
Created fileio disk01 with size 10737418240
/backstores/fileio> cd /iscsi
Create the iSCSI Target and Portal
To create an iSCSI target, navigate to scsi directory run the create command specifying the IQN of the target followed by the name of the target itself as shown below.
create a target
naming rule : [iqn.(year)-(month).(reverse of domain name):(any name you like)]
/iscsi> create iqn.2018-05.world.srv:dlp.target01
Created target iqn.2018-05.world.srv:dlp.target01.
Created TPG 1.

Global pref auto_add_default_portal=true
Created default portal listening on all IPs (0.0.0.0), port 3260.
/iscsi> cd iqn.2018-05.world.srv:dlp.target01/tpg1/luns

Set up the LUN

LUN is needed to associate a block device with a specific TPG. To create the LUN, naviate to the target portal group (tpg1) created above and create a LUN for both the block and fileio backstores created above.
set LUN
/iscsi/iqn.20...t01/tpg1/luns> create /backstores/fileio/disk01
Created LUN 0.
/iscsi/iqn.20...t01/tpg1/luns> cd ../acls
Set up Access Control Lists (ACL)
ACL is used to specify the clients (iSCSI intiators) allowed to access the iSCSI target backstores. To cteate an ACL for an intiator, run create command with IQN of the initiator just like as did above.
set ACL (it's the IQN of an initiator you permit to connect)
/iscsi/iqn.20...t01/tpg1/acls> create iqn.2018-05.world.srv:www.initiator01
Created Node ACL for iqn.2018-05.world.srv:www.initiator01
Created mapped LUN 0.
/iscsi/iqn.20...t01/tpg1/acls> cd iqn.2018-05.world.srv:www.initiator01

Set User ID and password for iSCSI initiator authentication
Set the userid and password as shown below and exit the targetcli admin console. When you exit the console, all the changes are saved automatically.
set UserID for authentication
/iscsi/iqn.20...w.initiator01> set auth userid=username
Parameter userid is now 'username'.
/iscsi/iqn.20...w.initiator01> set auth password=password
Parameter password is now 'password'.
/iscsi/iqn.20...w.initiator01> exit
Global pref auto_save_on_exit=true
Last 10 configs saved in /etc/rtslib-fb-target/backup.
Configuration saved to /etc/rtslib-fb-target/saveconfig.json
Once the changes are saved, start the target service and it should now be listening on port 3206.
If firewall is running, allow iscsi target through it.
ufw allow
3260 #
ufw
reload
ss -napt | grep
3260 LISTEN
0 256 0.0.0.0:3260 0.0.0.0:*

Configure iSCSI target(Using tgt)

1)Install administration tools first.
root@dlp:~# apt -y install tgt
2)Configure iSCSI Target.
For example, create an disk-image under the [/var/lib/iscsi_disks] directory and set it as a SCSI

device.
create a disk-image
root@dlp:~# mkdir /var/lib/iscsi_disks

root@dlp:~# dd if=/dev/zero of=/var/lib/iscsi_disks/disk01.img count=0 bs=1 seek=10G
[root@dlp ~]# vi /etc/tgt/conf.d/target01.conf # create
new # naming rule : [iqn.(year)-(month).
(reverse of domain name):(any name you like)]
<target iqn.2018-05.world.srv:dlp.target01>
provided devicce as a iSCSI target
backing-store /var/lib/iscsi_disks/disk01.img
iSCSI Initiator's IQN you allow to connect to this Target
initiator-name iqn.2018-05.world.srv:www.initiator01

authentication info (set any name you like for "username", "password")
incominguser username password
</target>
root@dlp:~# systemctl restart tgt # show status
root@dlp:~# tgtadm --mode target --op show Target 1: iqn.2018-
05.world.srv:dlp.target01
System information:
Driver: iscsi
State: ready
I_T nexus information:
LUN information:
LUN: 0
Type: controller
SCSI ID: IET 00010000
SCSI SN: beaf10
Size: 0 MB, Block size: 1
Online: Yes
Removable media: No
Prevent removal: No
Readonly: No
SWP: No
Thin-provisioning: No
Backing store type: null
Backing store path: None
Backing store flags:
LUN: 1
Type: disk
SCSI ID: IET 00010001
SCSI SN: beaf11
Size: 10737 MB, Block size: 512
Online: Yes
Removable media: No
Prevent removal: No
Readonly: No
SWP: No
Thin-provisioning: No
Backing store type: rdwr

Backing store path: /var/lib/iscsi_disks/disk01.img
Backing store flags:
Account information:username
ACL information:
ALL
iqn.2018-05.world.srv:www.initiator01

Configure iSCSI Initiator

1)Install iSCSI initiator packages
apt -y install open-iscsi
2)Edit the /etc/iscsi/initiatorname.iscsi and set the IQN of the initiator to be just the same as the one we created on the target above you can comment out the existing and add a new one as shown below.
vim /etc/iscsi/initiatorname.iscsi
#InitiatorName=iqn.1993-08.org.debian:01:02068a9161c
# change to the same IQN you set on the iSCSI target server
InitiatorName=iqn.2018-05.world.srv:www.initiator01

3)Edit the /etc/iscsi/iscsid.conf configuration file to set the authentication method and specify the username and password defined above, under the CHAP settings
# vim /etc/iscsi/iscsid.conf
# line 56: uncomment
node.session.auth.authmethod = CHAP
# line 60,61: uncomment and specify the username and password you set on the iSCSI target
node.session.auth.username = username
node.session.auth.password = password
Save the configuration file and restart and enable both iscsid and iscsi services.

systemctl restart iscsid open-
iscsi # systemctl enable

iscsid open-iscsi
Next, run target discovery against our iSCSI target server to find out the shared targets.
10.0.0.30:3260,1 iqn.2018-05.world.srv:dlp.target01
confirm status after discovery
root@www:~# iscsiadm -m node -o show
# BEGIN RECORD 2.0-874
node.name = iqn.2018-05.world.srv:dlp.target01
node.tpgt = 1
node.startup = manual
node.leading_login = No
.....
.....
node.conn[0].iscsi.IFMarker = No
node.conn[0].iscsi.OFMarker = No
# END RECORD
# login to the target
root@www:~# iscsiadm -m node --login
Logging in to [iface: default, target: iqn.2018-05.world.srv:dlp.target01, portal:
10.0.0.30,3260] (multiple)

Login to [iface: default, target: iqn.2018-05.world.srv:dlp.target01, portal:
10.0.0.30,3260] successful.
confirm the established session
root@www:~# iscsiadm -m session -o show
tcp: [1] 10.0.0.30:3260,1 iqn.2018-05.world.srv:dlp.target01 (non-flash)

confirm the partitions
root@www:~# cat /proc/partitions
major minor #blocks name
252 0 31457280 sda
252 1 31455232 sda1
253 0 30441472 dm-0
253 1 999424 dm-1
8 0 10485760 sdb
added new device provided from the target server as [sdb]
To make these devices usable, we need to partition them, create filesystems on them
and mount them.
To partition the devices, you can use any partitioning system you are comfortable with.
In our case we used parted in a scripted format as shown below.
create label
root@www:~# parted --script /dev/sdb "mklabel msdos"
create partiton
root@www:~# parted --script /dev/sdb "mkpart primary 0% 100%"
format with ext4
root@www:~# mkfs.ext4 /dev/sdb1
mke2fs 1.44.1 (24-Mar-2018)
Creating filesystem with 2619392 4k blocks and 655360 inodes
Filesystem UUID: 841e57a2-b0d7-4431-945b-7e0ceb15b0f7
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632
Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
mount device
root@www:~# mount /dev/sdb1 /mnt
Confirm the mounting
root@www:~# df -hT
Filesystem Type Size Used Avail Use% Mounted on
udev devtmpfs 1.9G 0 1.9G 0% /dev
tmpfs tmpfs 395M 612K 394M 1% /run
/dev/mapper/ubuntu--vg-root ext4 29G 1.7G 26G 7% /
tmpfs tmpfs 2.0G 0 2.0G 0% /dev/shm
tmpfs tmpfs 5.0M 0 5.0M 0% /run/lock
tmpfs tmpfs 2.0G 0 2.0G 0% /sys/fs/cgroup
tmpfs tmpfs 395M 0 395M 0% /run/user/0
/dev/sdb1 ext4 9.8G 37M 9.3G 1% /mnt

What were the difficulties encountered?

Connection timed out Error:
The intiator was setup easily on say ‘System A’ , as well as the target on ‘System B’ with either methods of using tgt or targetcli.But the problem was halfway through ,mainly in the discovery of the target by the
inititor given the target IP.
Wherein the terminal just waited for a few seconds before giving the “iscisadm : connect to x.x.x.x timed out”.The same command is displayed for about 5 times before finally giving up as the no of max retries=5 was reached.
Searching the given error on web was futile as it mainly pertained to firstly other operating systems like CentOS and Red Hat,and secondly did not refer to our error context properly,the few solutins found involved small alterations that did not help the situation in any further way.

ADVANTAGES

1. Less expensive than Fibre Channel.
2. Easy to implement (set up) than Fibre Channel SAN as it possesses well understood IP infrastructure.
3. Fast, reliable transport services for SCSI commands and data using Internet & LAN protocols.
4. No distance limitation to implement.
5. Simpler to deploy and manage than a Fibre Channel.
6. Fastest possible network connections and h/w based TCP processors are used.
7. Best suited for less frequent used data.
8. Fibre Channel technology can also be combined with it to implement the positive aspects of both.

DISADVANTAGES

Deploying iSCSI in mission-critical situations isn't a risk-free proposition. Customers may have to cope with
higher latency and performance issues in mixed networks after an iSCSI implementation.

One of the challenges, especially for mission-critical performance-sensitive applications, is that there is higher latency in iSCSI. Specifically, wrapping SCSI packets around TCP/IP protocols has some overhead. It is also difficult to guarantee the performance of mixed networks. For example, if your VoIP, iSCSI, email and Microsoft Word documents are all on the same wire without some form of quality assurance for performance, it's difficult to guarantee it. That's one of the nice things about Fibre Channel SANs. It's almost guaranteed there won't be any other traffic except disk traffic on that network

Results:

After discovery,login,mount phases comes the file transfer phase that is carried out using cp command for copying files from initiator local system to storage as mounted in /mnt. Used tools like tcpdump,Wireshark and NetSpeed(Ubuntu tool) to detect and record the the transfer and its parmeters like packets and via which protocol(as shown in tcpdump,Wireshark) and transfer rate and on which network got from NetSpeed.