Internet Small Computer System Interface (iSCSI)

賴羿茗 (ymlai)

What is iSCSI?

- A protocol that transports **SCSI block storage commands over standard IP networks**.
  - Encapsulates those same SCSI commands inside **TCP/IP** packets
  - Allowing servers to treat remote disks as locally attached block devices
  - RFC 3720

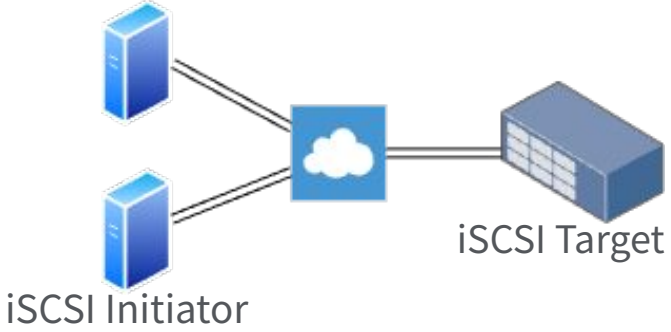
- SCSI (Small Computer System Interface)
  - A set of **standards** for physically connecting and transferring data between computers and peripheral devices (best known for its use with **storage devices** such as **HDD**)

Why Use iSCSI?

- Advantages
  - Easy to configure
  - Supports multipathing (MPIO)
  - Widely supported (Linux, Windows, VMware)
  - Can export any block device including:
    - Disk partitions
    - **LVM volumes**
    - ZFS volumes
    - Files as block devices
- Use Cases
  - Virtualization (KVM/Proxmox/VMware shared storage)
  - Clustered environments (shared disks)
  - Centralized storage for servers
  - Backup appliances

Feature	iSCSI	NFS	SMB (CIFS)
Access Type	Block-level	File-level	File-level
OS Compatibility	Widely supported	Primarily Linux/Unix (Windows supports via client)	Best on Windows; Linux supports via Samba
Performance	**High** (block I/O; depends on network	Good; varies by version (NFSv4+)	Good; SMB3 adds significant improvements
Concurrency Handling	Filesystem handled by host OS	Built-in file locking	Built-in file locking, versioning, leases
Centralized Management	Via storage arrays or SAN tools	Simple exports	Active Directory integration
Typical Use	Servers needing raw block devices	Unix/Linux file sharing	Windows file sharing & enterprise file services

Core Components / Key Terms



- iSCSI Initiator
  - Client that sends SCSI commands over the network
- iSCSI Target
  - Server that receives commands and provides storage resources
- iSCSI Qualified Name (IQN)
  - A unique, global name for each initiator and target
  - Used for authentication and to establish connections.
- Logical Unit Number (LUN)
  - A logical volume or partition on an iSCSI target
- Portal
  - An IP address and port number combination on the iSCSI target or initiator

Initiator Types

- Software Initiator
  - The iSCSI code runs on the host and allows an Ethernet NIC to handle iSCSI traffic
  - Offers low cost with a performance penalty and CPU overhead
  - Most common on Windows, Linux, macOS, ESXi
- Hardware Initiator (iSCSI HBA)
  - A high-performance HBA integrates both TCP/IP and iSCSI functions
  - Provides high-speed iSCSI transport and minimal CPU overhead

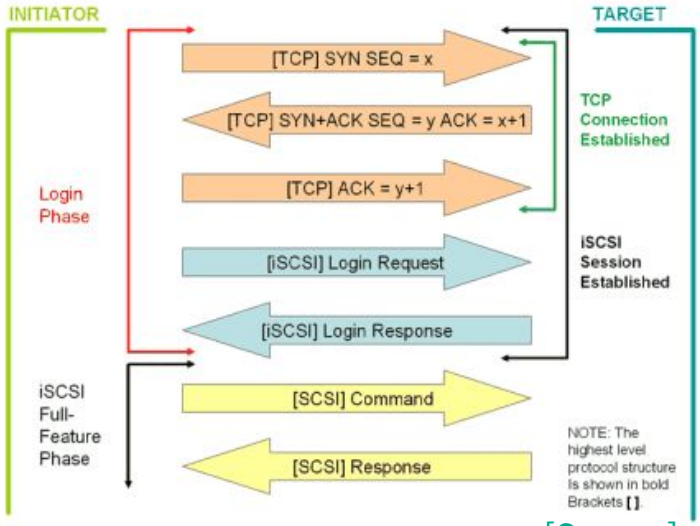
How iSCSI Works

SCSI commands are encapsulated into TCP/IP packets and transmitted over Ethernet networks.

- Initiator sends SCSI commands in iSCSI Protocol Data Units (PDUs)
- PDUs encapsulated in TCP segments, IP packets, and Ethernet frames
- OS perceives storage as local block devices

Sessions & Authentication

- Communication organized in iSCSI Sessions over **TCP** connections
- **CHAP protocol** used for secure authentication
- Session phases:
  - Login Phase
  - Full Feature Phase
- IQN identifies initiators and targets



[Source]

Target Setup (targetcli)

iSCSI Target on Debain

- Install the iSCSI Target (LIO)

```
sudo apt update
sudo apt install targetcli-fb
```

- Start the service

```
sudo systemctl enable --now targetclid.service
```

- Open the management shell

```
sudo targetcli
```

Configure: `/etc/rtslib-fb-target/saveconfig.json`

Creating Storage Backends (1/2)

- File-backed storage

```
mkdir -p /var/iscsi
truncate -s 5G /var/iscsi/disk01.img

targetcli
/backstores/fileio create disk01 /var/iscsi/disk01.img 5G
```

- Block device

```
targetcli
/backstores/block create name=blk01 dev=/dev/sdc
```

Creating Storage Backends (2/2)

- LVM volume

```
pvcreate /dev/sdb1
vgcreate vg_iscsi /dev/sdb1
lvcreate -L 10G -n lv_sdb vg_iscsi

targetcli
/backstores/block create name=lvm_sdb dev=/dev/vg_iscsi/lv_sdb
```

- Verify

```
targetcli
/> ls
```

Creating the iSCSI Target

- Create a new IQN

```
targetcli
/iscsi create iqn.2025-12.cs.nycu.edu.tw:target01
```

- An iSCSI IQN format is **iqn**.yyyy-mm.naming-authority:unique-name
  - iqn is a literal string
  - yyyy-mm is the year and month the naming authority was established
  - naming-authority is the domain name
  - unique-name is a string to ensure uniqueness.

Attach Backstore to Target

- Create a new LUN

```
targetcli
cd /iscsi/iqn.2025-12.cs.nycu.edu.tw:target01/tpg1/
luns/ create /backstores/fileio/disk01
luns/ create /backstores/block/blk01
luns/ create /backstores/block/lvm_sdb
```

```
/iscsi/iqn.20...target01/tpg1> ls
o- tpg1 ..... [no-gen-acls, no-auth]
  o- acls ..... [ACLs: 0]
  o- luns ..... [LUNs: 3]
  | o- lun0 ..... [fileio/disk01 (/var/iscsi/disk01.img) (default_tg_pt_gp)]
  | o- lun1 ..... [block/blk01 (/dev/sdc) (default_tg_pt_gp)]
  | o- lun2 ..... [block/lvm_sdb (/dev/vg_iscsi/lv_sdb) (default_tg_pt_gp)]
  o- portals ..... [Portals: 1]
    o- 0.0.0.0:3260 ..... [OK]
```

Configure Network Portal / ACL

- Default portal uses 0.0.0.0:3260
- To use a specific IP:

```
targetcli
cd /iscsi/iqn.2025-12.cs.nycu.edu.tw:target01/tpg1/
portals/ delete  0.0.0.0 3260
portals/ create 10.0.8.5 3260
```

- Allow an initiator to connect:

```
targetcli
cd /iscsi/iqn.2025-12.cs.nycu.edu.tw:target01/tpg1/
acls/ create iqn.2025-12.cs.nycu.edu.tw:node01
```

Enable CHAP Authentication (recommended)

- Set userid and password for each initiator

```
targetcli
cd /iscsi/iqn.2025-12.cs.nycu.edu.tw:target01/tpg1/
set attribute authentication=1
acls/iqn.2025-12.cs.nycu.edu.tw:node01 set auth userid=cscc
acls/iqn.2025-12.cs.nycu.edu.tw:node01 set auth password=2025SA
acls/iqn.2025-12.cs.nycu.edu.tw:node01 info
```

- Save Configuration and Exit

```
targetcli
saveconfig
exit
```

Example

- Create new taget portal group and set lun as read-only

```
targetcli
cd /iscsi/iqn.2025-12.cs.nycu.edu.tw:target01/
create tag=2
tpg2/portals create 192.168.20.20:3260
tpg1/luns/ delete 2
tpg2/luns/ create /backstores/block/lvm_sdb
tpg2/acls/iqn.2025-12.cs.nycu.edu.tw:target01/ delete 0
tpg2/acls/iqn.2025-12.cs.nycu.edu.tw:target01 create mapped_lun=0 \
    tpg_lun_or_backstore=/backstores/block/lvm_sdb write_protect=1
```

Initiator Setup (iscsiadm)

iSCSI Initiator on Debain

- Install the iSCSI Initiator

```
sudo apt update
sudo apt install open-iscsi
```

- Discover Targets

```
$ sudo iscsiadm -m discovery -t sendtargets -p 10.0.8.5
10.0.8.5:3260,1 iqn.2025-12.cs.nycu.edu.tw:target01
```

- Login

```
sudo iscsiadm -m node --login
```

Set the Initiator IQN

- Set the Initiator IQN

```
sudo vim /etc/iscsi/initiatorname.iscsi
InitiatorName=iqn.2025-12.cs.nycu.edu.tw:node01
```

- Restart iSCSI

```
sudo systemctl restart iscsid
sudo systemctl restart iscsi
```

Configure CHAP authentication

- Configure for the specific target

```
sudo iscsiadm -m node -T iqn.2025-12.cs.nycu.edu.tw:target01 \
    -p 10.0.8.5 --op=update -n node.session.auth.authmethod -v CHAP

sudo iscsiadm -m node -T iqn.2025-12.cs.nycu.edu.tw:target01 \
    -p 10.0.8.5 --op=update -n node.session.auth.username -v <USERID>

sudo iscsiadm -m node -T iqn.2025-12.cs.nycu.edu.tw:target01 \
    -p 10.0.8.5 --op=update -n node.session.auth.password -v <PASSWORD>

sudo iscsiadm -m node -T iqn.2025-12.cs.nycu.edu.tw:target01 -p 10.0.8.5
```

Log in to the target

- Login the target

```
sudo iscsiadm -m node -T iqn.2025-12.cs.nycu.edu.tw:target01 -p 10.0.8.5 --login
Logging in to [iface: default, target: iqn.xxx:target, portal: 10.0.8.5,3260]
Login to [iface: default, target: iqn.xxx:target, portal: 10.0.8.5,3260] successful
```

- Check sessions

```
sudo iscsiadm -m session
```

- Enable automatic login at boot

```
sudo iscsiadm -m node -T iqn.2025-12.cs.nycu.edu.tw:target01 \
    -p 10.0.8.5 --op=update -n node.startup -v automatic
```

Log out from the target

- Check sessions

```
sudo iscsiadm -m session
tcp: [1] 10.0.8.5:3260,1 iqn.2025-12.cs.nycu.edu.tw:target01 (non-flash)
```

- Log out by session ID

```
sudo iscsiadm -m session --sid=1 --logout
```

- Log out by target name and portal

```
sudo iscsiadm -m node -T iqn.2025-12.cs.nycu.edu.tw:target01 \
    -p 10.0.8.5:3260 --logout
```

Thanks